MPMC
Course File

Contents required for course file

1. Cover Page
2. Syllabus copy
3. Vision of the Department
4. Mission of the Department
5. PEOs and POs
6. Course objectives and outcomes
7. Brief notes on the importance of the course and how it fits into the curriculum
8. prerequisites
9. Instructional Learning Outcomes
10. Course mapping with PEOs and POs
11. Class Time Table
12. Individual Time Table
13. Micro Plan with dates and closure report
14. Detailed notes
15. Additional topics
16. University Question papers of previous years
17. Question Bank
18. Assignment topics
19. Unit wise Quiz Questions
20. Tutorial problems
21. Known gaps ,if any
22. Discussion topics
23. References, Journals, websites and E-links
24. Quality Control Sheets
25. Student List
26. Group-Wise students list for discussion topics

1.cover page

GEETHANJALI COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF *Electronics and Communication Engineering*

(Name of the Subject) : Microprocessors and Microcontrollers **Course file**

(JNTU CODE – 56012) **Programme : *UG***

Branch: EEE **Version No : *0***

Year: *III* **Document No: GCET/ECE/56012/01**

Semester: II **No. of pages :**

Classification status (Unrestricted / Restricted) : Unrestricted

Distribution List : Dept. Library, Dept Office, Concerned Faculty

Prepared by	**Updated by:**
1) Name : M.Laxmi	**1) Name**
2) Sign :	**2) Sign :**
3) Design : Assoc. Professor.	**3) Design**
4) Date : 12/12/2014	**4) Date :**

Verified by : 1) Name :	*** For Q.C Only.**
2) Sign :	**1) Name :**
3) Design :	**2) Sign :**
4) Date :	**3) Design :**
	4) Date :

Approved by : (HOD) 1) Name :

2) Sign :

3) Date :

2. Syllabus copy

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY

III YEAR B.TECH.EEE-IISEM

(56012) MICROPROCESSORS AND MICROCONTROLLERS

UNIT-1

8080 Architecture: Introduction to 8085 microprocessor, Architecture of 8086 Microprocessor – functional diagram description, Register organization, memory segmentation, Programming model, memory addresses, Physical memory organization, Signal description of 8086,Common function signals, maximum and minimum mode signals, Timing diagrams, Interrupts of 8086

UNIT-2

Instruction set and assembly language programming of 8086: Instruction formats, addressing modes, Instruction set, Assembler directives, macros, Simple programs involving logical, branch and call instructions, sorting, evaluating arithmetic expressions, string manipulations

UNIT-3

I/O interface:8255-PPI,various Modes of operation and interfacing to 8086 ,Modes of operation and interfacing to 8086, Interfacing keyboard, display, Stepper motor interfacing, D/A and A/D converter

UNIT-4

Interfacing with advanced devices: Memory interfacing to 8086, Interrupt structure of 8086, vector interrupt table, interrupt service routine, Introduction to DOS and BIOS interrupts ,interfacing Interrupt controller-8259, DMA controller 8257 to 8086

UNIT-5

Communication interface: Serial communication standards, serial data transfer schemes, 8251 USART architecture and interfacing, RS-232, IEEE-488, Prototyping and troubleshooting

UNIT-6

Introduction to microcontrollers: Overview of 8051 microcontrollers, Architecture, I/O ports, memory organization, Addressing modes, Instruction set of 8051, Simple programs on 8051

UNIT-7

8051 real time control: Interrupts, Timer/counters and Serial communication, Programming timer interrupts, Programming external hardware interrupts, Programming the serial communication interrupts, Programming 8051 timers/counters.

UNIT-8

The AVR RISC microcontroller architecture: Introduction, AVR family architecture, registers file, the ALU, memory access and Instruction Execution, I/O memory. EEPROM, I/O ports, timers, UART, Interrupt structure.

3.Vision of the Department

To impart quality technical education in Electronics and Communication Engineering emphasizing analysis, design/synthesis and evaluation of hardware/embedded software using various Electronic Design Automation (EDA) tools with accent on creativity, innovation and research thereby producing competent engineers who can meet global challenges with societal commitment.

4. Mission of the Department

i. To impart quality education in fundamentals of basic sciences, mathematics, electronics and communication engineering through innovative teaching-learning processes.
ii. To facilitate Graduates define, design, and solve engineering problems in the field of Electronics and Communication Engineering using various Electronic Design Automation (EDA) tools.
iii. To encourage research culture among faculty and students thereby facilitating them to be creative and innovative through constant interaction with R & D organizations and Industry.
iv. To inculcate teamwork, imbibe leadership qualities, professional ethics and social responsibilities in students and faculty

5.PEOs and POs

Program Educational Objectives of B. Tech (ECE) Program :

I. To prepare students with excellent comprehension of basic sciences, mathematics and engineering subjects facilitating them to gain employment or pursue postgraduate studies with an appreciation for lifelong learning.
II. To train students with problem solving capabilities such as analysis and design with adequate practical skills wherein they demonstrate creativity and innovation that would enable them to develop state of the art equipment and technologies of multidisciplinary nature for societal development.
III. To inculcate positive attitude, professional ethics, effective communication and interpersonal skills which would facilitate them to succeed in the chosen profession exhibiting creativity and innovation through research and development both as team member and as well as leader.

Program Outcomes of B.Tech ECE Program

1. An ability to apply knowledge of Mathematics, Science, and Engineering to solve complex engineering problems of Electronics and Communication Engineering systems.
2. An ability to model, simulate and design Electronics and Communication Engineering systems, conduct experiments, as well as analyze and interpret data and prepare a report with conclusions.
3. An ability to design an Electronics and Communication Engineering system, component, or process to meet desired needs within the realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability and sustainability.
4. An ability to function on multidisciplinary teams involving interpersonal skills.
5. An ability to identify, formulate and solve engineering problems of multidisciplinary nature.
6. An understanding of professional and ethical responsibilities involved in the practice of Electronics and Communication Engineering profession.
7. An ability to communicate effectively with a range of audience on complex engineering problems of multidisciplinary nature both in oral and written form.
8. The broad education necessary to understand the impact of engineering solutions in a global, economic, environmental and societal context.
9. A recognition of the need for, and an ability to engage in life-long learning and acquire the capability for the same.
10. A knowledge of contemporary issues involved in the practice of Electronics and Communication Engineering profession
11. An ability to use the techniques, skills and modern engineering tools necessary for engineering practice.
12. An ability to use modern Electronic Design Automation (EDA) tools, software and electronic equipment to analyze, synthesize and evaluate Electronics and Communication Engineering systems for multidisciplinary tasks.
13. Apply engineering and project management principles to one's own work and also to manage projects of multidisciplinary nature.

6..Course objectives and outcomes

Course objectives :

1. To understand the basic 8, 16 bit microprocessor architecture and its functionalities.
2. To understand the programming model of microprocessor.
3. To develop the microprocessor based programs for various applications.
4. To make the interfacing in between microprocessor and various peripherals.
5. To develop DOS/BIOS programs.
6. To develop the microcontroller based programs for various applications.
7. To enable the students to understand basic feature of 8051 and AVR controller
8.

course outcomes:

1. Basic understanding of microprocessors and microcontrollers architectures and its functionalities.
2. esign and develop Microprocessor/ microcontroller based systems for real time applications using low level language like ALP

7. Brief notes on the importance of the course and how it fits into the curriculum

1. The student will be able to understand the basic microprocessors architecture its functionality.and to interface with I/O devices.
2. This courset will gives the knowledge to understand the advanced processors and its interfacing with various devices.
3. It will be the base for embedded system design.
4. The course will be usefull for understanding computer organization and advanced systems.

8.Pre Requisites: Swithing Theory & Logic Design ,Computer Organization.

9.Instructional Learning Outcomes

1) students will gain knowledge on Architecture of 8086 Microprocessor and its function and flags.
2) students will gain knowledge on programming using logical and call instructions.students will gain knowledge on how to interface 8086 with 8257 and need for memory reigsters.
3) students will gain knowledge on 8255 and D/A and A/D interfacing.
4) students will gain knowledge on Interrupt structure of 8086 & 8259 PIC Architecture.
5) students will gain knowledge on 8251 USART architecture and interfacing.

6) students will gain knowledge on 8051 microcontrollers, Architecture and i/o ports, Addressing modes, Instruction set, Sample programs.
7) , students will gain knowledge on Serial communication control in 8051, Interrupt structure of 8051,Memory and i/o interfacing of 8051
8) , students will gain knowledge on Serial communication control in 8051, Interrupt structure of 8051,Memory and i/o interfacing of 8051

10.Course mapping with PEOs and POs

Course mapping with POs

a) an ability to apply the knowledge of Mathematics, science and engineering in Electronics and communications	√
b) an ability to Design & Conduct Experiments, as well as analyze & Interpret Data	√
c) an ability to design a system, component, or process to meet desired needs with in realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability	√
d) an ability to function on multidisciplinary teams	√
e) an ability to Identify, Formulate & Solve problems in the area of Electronics and Communications Engineering	√
f) an understanding of professional and ethical responsibility	
g) an ability to communicate effectively	
h) the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context	√
i) a recognition of the need for, and an ability to engage in life-long learning	√
j) a knowledge contemporary issues	
k) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice	√

Course mapping with PEOs

	PROGRAMME EDUCATIONAL OBJECTIVES					
	Domain knowledge	**Professional Employment**	**Higher Degrees**	**Engineering citizenship**	**Lifelong Learning**	**Research and Development**
a) an ability to apply the knowledge of Mathematics, science and engineering in Electronics and communications	√	√	√	√	√	√
b) an ability to Design & Conduct Experiments, as well as analyze & Interpret Data	√	√	√	√	√	√
c) an ability to design a	√	√	√		√	√

system, component, or process to meet desired needs with in realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability						
d) an ability to function on multidisciplinary teams	√	√			√	√
e) an ability to Identify, Formulate & Solve problems in the area of Electronics and Communications Engineering	√	√	√		√	√
h) the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context	√	√	√		√	√
i) a recognition of the need for, and an ability to engage in life-long learning	√	√	√	√	√	√
k) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice	√	√	√	√	√	√

12. Class Time Table

A-section

13. Individual Time Table

13.Micro Plan:

Name of the FacultyM.Laxmi

Branch: ECE

Year & Sem: III-II

Subject: Microprocessors and microcontrollers

Lesson Plan

S.L no	Unit No	Total no of Periods	Topics to be covered	Reg/ Additional	Teaching aids LCD/OHP/BB	Remarks
1	**I**	08	Introduction to 8085 microprocessor,	Regular	OHP,BB	
2			Architecture of 8086 Microprocessor – functional diagram description	Regular	OHP,BB	
3			Register organization, memory segmentation	Regular	OHP,BB	
4			Programming model, memory addresses	Regular	OHP,BB	
5			Physical memory organization	Regular	BB	
6			Signal description of 8086, Common function signals, maximum and minimum mode signals	Regular	BB	
7			Timing diagrams, Interrupts of 8086	Regular	OHP,BB	
8			Tutorial class	Regular	BB	
9	**II**	08	Instruction formats, addressing modes	Regular	OHP,BB	
10			Instruction set of 8086	Regular	BB	
11			Instruction set of 8086	Regular	BB	
12			Assembler directives, macros	Regular	OHP,BB	
13			Simple programs on logical ,branch and call instructions	Regular	BB	
14			Programs on sorting, string manipulations	Regular	BB	
15			Tutorial class	Regular	BB	
16			**Introduction to Pentium and dual core processors**	**Additional**	OHP,BB	
17	**III**	08	8255-PPI	Regular	OHP,BB	
18			Modes of operation and interfacing to 8086	Regular	BB	
19			Modes of operation and interfacing to 8086	Regular	BB	
20			Interfacing keyboard, display to 8086	Regular	OHP,BB	
21			Stepper motor interfacing to 8086		BB	
22			D/A and A/D converter interfacing to 8086	Regular	OHP,BB	
23			D/A and A/D converter interfacing to 8086	Regular	BB	
24			Tutorial class	Regular	LCD,OHP,BB	
25	**IV**	09	Memory interfacing to 8086	Regular	OHP,BB	
26			Memory interfacing to 8086	Regular	BB	
27			Interrupt structure of 8086,vector	Regular	OHP,BB	

			interrupt table, interrupt service routine			
28			Introduction to DOS and BIOS interrupts	Regular	BB	
29			Interrupt controller-8259	Regular	BB	
30			Interrupt controller-8259	Regular	BB	
31			Interfacing DMA controller 8257 to 8086	Regular	OHP,BB	
32			Tutorial class	Regular	OHPBB	
33			**USB**	**Additional**	OHP,BB	
34	**V**	06	Serial communication standards, serial data transfer schemes	Regular	OHP,BB	
35			8251 USART architecture	Regular	BB	
36			8251 USART interfacing	Regular	BB	
37			RS-232,IEEE-488	Regular	OHP,BB	
38			Prototyping and troubleshooting	Regular	OHP,BB	
39			**TTL to RS 232 and RS-232 to TTL conversion**	**Missing**	BB	
40	**VI**	09	Overview of 8051 microcontrollers	Regular	OHP,BB	
41			Architecture of 8051microcontroller	Regular	BB	
42			I/O ports, memory segmentation of 8051	Regular	BB	
43			Addressing modes of 8051	Regular	BB	
44			Instruction set of 8051	Regular	BB	
45			Simple programs on 8051	Regular	BB	
46			Simple programs on 8051	Regular	OHP,BB	
47			Tutorial class	Regular	OHP,BB	
48			**Introduction to PIC and ARM controllers**	**Additional**	OHP,BB	
49	**VII**	08	Interrupts of 8051 microcontroller	Regular	OHP,BB	
50			Timer/counters of 8051	Regular	BB	
51			Serial communication of 8051	Regular	BB	
52			Programming timer interrupts	Regular	OHP,BB	
53			Programming external hardware interrupts	Regular	OHP,BB	
54			Programming the serial communication interrupts	Regular	BB	
55			Programming 8051 timers/counters	Regular	BB	
56			Tutorial class	Regular	BB	
57	**VIII**	06	AVR RISC microcontroller – architecture, register file	Regular	OHP,BB	
58			ALU and memory access And Instruction Execution	Regular	BB	
59			I/O memory, EEPROM	Regular	BB	
60			I/O ports, timers, UART	Regular	OHP,BB	
61			Interrupt structure	Regular	OHP,BB	
62			Tutorial class	Regular	BB	

Micro Plan:

SL. No	Unit No.	Total No.of Periods	Date	Topic to be covered in One lecture	Regular/ Additional/ missing	Teaching aids LCD/OHP/ BB	Remarks
1	**I**	08		Introduction to 8085 microprocessor,	Regular	OHP,BB	
2				Architecture of 8086 Microprocessor – functional diagram description	Regular	OHP,BB	
3				Register organization, memory segmentation	Regular	OHP,BB	
4				Programming model, memory addresses	Regular	OHP,BB	
5				Physical memory organization	Regular	BB	
6				Signal description of 8086, Common function signals, maximum and minimum mode signals	Regular	BB	
7				Timing diagrams, Interrupts of 8086	Regular	OHP,BB	
8				Tutorial class	Regular	BB	
9	**II**	08		Instruction formats, addressing modes	Regular	OHP,BB	
10				Instruction set of 8086	Regular	BB	
11				Instruction set of 8086	Regular	BB	
12				Assembler directives, macros	Regular	OHP,BB	
13				Simple programs on logical ,branch and call instructions	Regular	BB	
14				Programs on sorting, string manipulations	Regular	BB	
15				Tutorial class	Regular	BB	
16				**Introduction to Pentium and dual core processors**	**Additional**	OHP,BB	
17	**III**	08		8255-PPI	Regular	OHP,BB	
18				Modes of operation and interfacing to 8086	Regular	BB	
19				Modes of operation and interfacing to 8086	Regular	BB	
20				Interfacing keyboard, display to 8086	Regular	OHP,BB	
21				Stepper motor interfacing to 8086	Regular	BB	
22				D/A and A/D converter interfacing to 8086	Regular	OHP,BB	
23				D/A and A/D converter interfacing to 8086	Regular	BB	
24				Tutorial class	Regular	LCD,OHP,B	
25	**IV**	09		Memory interfacing to 8086	Regular	OHP,BB	
26				Memory interfacing to 8086	Regular	BB	
27				Interrupt structure of 8086,vector interrupt table, interrupt service routine	Regular	OHP,BB	
28				Introduction to DOS and BIOS interrupts	Regular	BB	
29				Interrupt controller-8259	Regular	BB	
30				Interrupt controller-8259	Regular	BB	
31				Interfacing DMA controller 8257 to 8086	Regular	OHP,BB	
32				Tutorial class	Regular	OHPBB	
33				**USB**	**Additional**	OHP,BB	
34	**V**	06		Serial communication standards, serial data transfer schemes	Regular	OHP,BB	
35				8251 USART architecture	Regular	BB	
36				8251 USART interfacing	Regular	BB	

37				RS-232,IEEE-488	Regular	OHP,BB	
				TTL to RS 232 and RS-232 to TTL conversion	**Missing**	BB	
38				Prototyping and troubleshooting	Regular	OHP,BB	
39				Tutorial class	Regular	OHP,BB	
40	**VI**	09		Overview of 8051 microcontrollers	Regular	OHP,BB	
41				Architecture of 8051microcontroller	Regular	BB	
42				I/O ports, memory segmentation of 8051	Regular	BB	
43				Addressing modes of 8051	Regular	BB	
44				Instruction set of 8051	Regular	BB	
45				Simple programs on 8051	Regular	BB	
46				Simple programs on 8051	Regular	OHP,BB	
47				Tutorial class	Regular	OHP,BB	
48				**Introduction to PIC and ARM controllers**	**Additional**	OHP,BB	
49	**VII**	08		Interrupts of 8051 microcontroller	Regular	OHP,BB	
50				Timer/counters of 8051	Regular	BB	
51				Serial communication of 8051	Regular	BB	
52				Programming timer interrupts	Regular	OHP,BB	
53				Programming external hardware interrupts	Regular	OHP,BB	
54				Programming the serial communication interrupts	Regular	BB	
55				Programming 8051 timers/counters	Regular	BB	
56				Tutorial class	Regular	BB	
57				Introduction to PIC and ARM controllers	**Additional**	BB	
58	**VII**	06		AVR RISC microcontroller –architecture, register file	Regular	BB	
59				ALU and memory access And Instruction Execution,I/O memory, EEPROM	Regular	BB	
60				I/O ports, timers, UART	Regular	OHP,BB	
61				Interrupt structure	Regular	OHP,BB	
62				Tutorial class	Regular	BB	

14.Detailed notes

UNIT-I

8085 Microprocessor *Contents*

•General definitions

•Overview of 8085 microprocessor

•Overview of 8086 microprocessor

•Signals and pins of 8086 microprocessor

The salient features of 8085 µp are:

•It is a 8 bit microprocessor.

•It is manufactured with N-MOS technology.

•It has 16-bit address bus and hence can address up to 216 =65536 bytes (64KB) memory locations through A_0-A_{15}.

•The first 8 lines of address bus and 8 lines of data bus are multiplexed AD0 – AD7.

•Data bus is a group of 8 lines D0 – D7.

•It supports external interrupt request.

•A 16 bit program counter (PC)

•A 16 bit stack pointer (SP)

•Six 8-bit general purpose register arranged in pairs: BC, DE, HL.

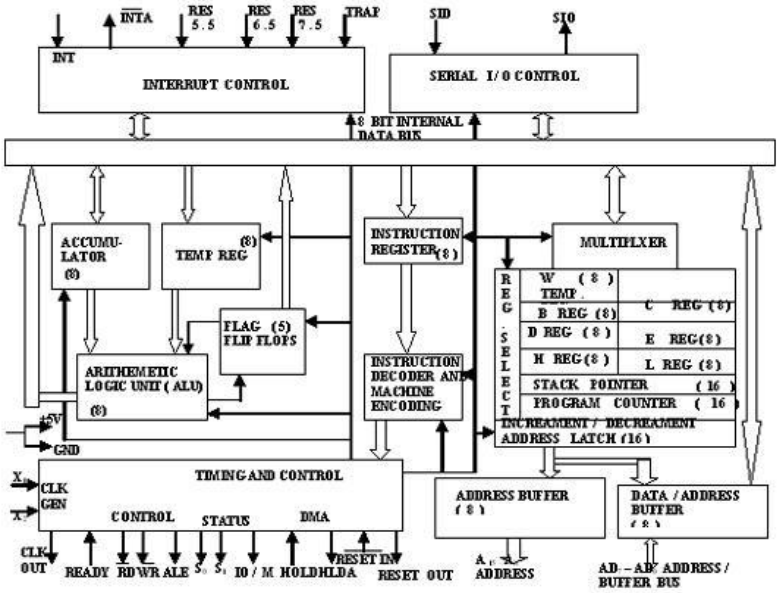
•It requires a signal +5V power supply and operates at 3.2 MHZ single phase clock.

•It is enclosed with 40 pins DIP (Dual in line package).

Overview of 8085 microprocessor

8085 Architecture

- Pin Diagram
- Functional Block Diagram



Instruction Set

8085 instruction set consists of the following instructions:

- Data moving instructions.
- Arithmetic - add, subtract, increment and decrement.
- Logic - AND, OR, XOR and rotate.
- Control transfer - conditional, unconditional, call subroutine, return from subroutine and restarts.
- Input/Output instructions.
- Other - setting/clearing flag bits, enabling/disabling interrupts, stack operations, etc.

Addressing mode

•**Register** - references the data in a register or in a register pair.

Register indirect - instruction specifies register pair containing address, where the data is located.

Direct, Immediate - 8 or 16-bit data.

Advantages of 8086 over 8085 are:

1. pipelining is employed making the execution faster.

2. data bus width increased to 16 bits.

3. higher memory of 1MB.

4. some instructions such as MUL or DIV are available for multiplication and division.

5. increased instruction set making the programming easier.

•EU executes instructions from the instruction system byte queue.

•Both units operate asynchronously to give the 8086 an overlapping instruction fetch and execution mechanism which is called as Pipelining. This results in efficient use of the system bus and system performance.

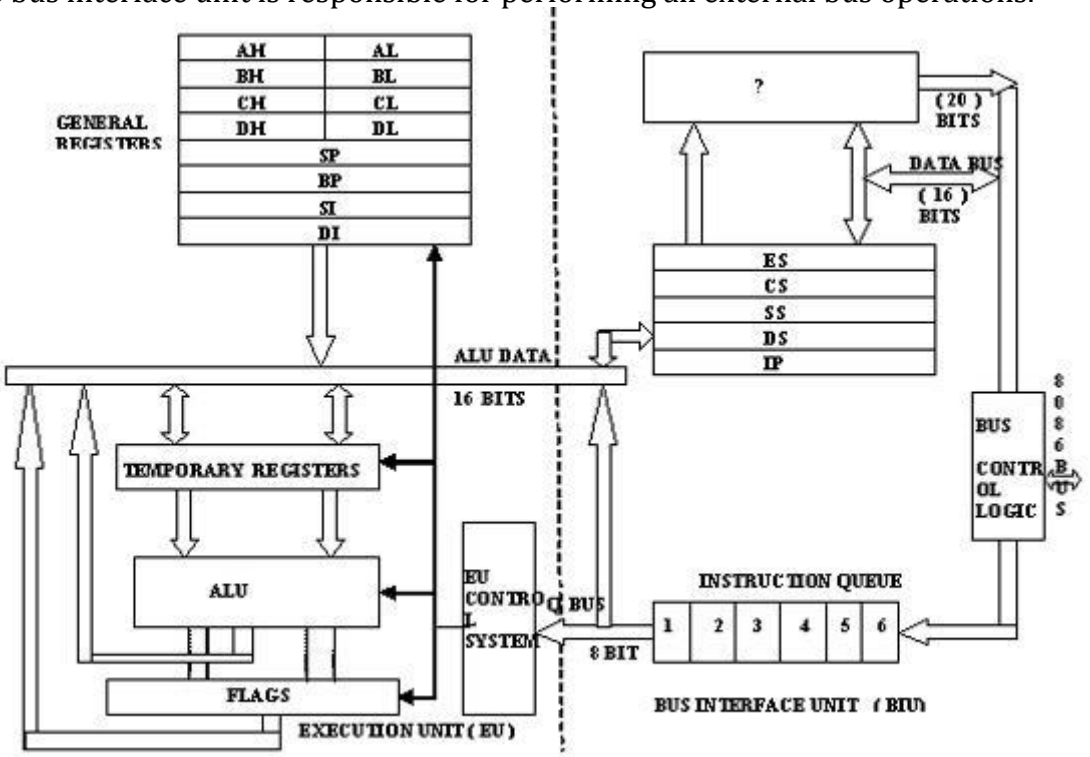
•BIU contains Instruction queue, Segment registers, Instruction pointer, Address adder.

•EU contains Control circuitry, Instruction decoder, ALU, Pointer and Index register, Flag register.

BUS INTERFACR UNIT:

• It provides a full 16 bit bidirectional data bus and 20 bit address bus.

•The bus interface unit is responsible for performing all external bus operations.



Specifically it has the following functions:

•Instruction fetch, Instruction queuing, Operand fetch and storage, Address relocation and Bus control.

•The BIU uses a mechanism known as an instruction stream queue to implement a *pipeline architecture.*

•This queue permits prefetch of up to six bytes of instruction code. When ever the queue of the BIU is not full, it has room for at least two more bytes and at the same time the EU is not requesting it to read or write operands from memory, the BIU is free to look ahead in the program by prefetching the next sequential instruction.

•These prefetching instructions are held in its FIFO queue. With its 16 bit data bus, the BIU fetches two instruction bytes in a single memory cycle.

•After a byte is loaded at the input end of the queue, it automatically shifts up through the FIFO to the empty location nearest the output.

•The EU accesses the queue from the output end. It reads one instruction byte after the other from the output of the queue. If the queue is full and the EU is not requesting access to operand in memory.

•These intervals of no bus activity, which may occur between bus cycles are known as

Idle state.

•If the BIU is already in the process of fetching an instruction when the EU request it to read or write operands from memory or I/O, the BIU first completes the instruction fetch bus cycle before initiating the operand read / write cycle.

•The BIU also contains a dedicated adder which is used to generate the 20bit physical address that is output on the address bus. This address is formed med by combining the current contents of the code segment CS register and the current contents of the instruction pointer IP register.

•The BIU is also responsible for generating bus control signals such as those for memory read or write and I/O read or write.

EXECUTION UNIT

The Execution unit is responsible for decoding and executing all instructions.

•The EU extracts instructions from the top of the queue in the BIU, decodes them, generates operands if necessary, passes them to the BIU and requests it to perform the read or write bys cycles to memory or I/O and perform the operation specified by the instruction on the operands.

•During the execution of the instruction, the EU tests the status and control flags and updates them based on the results of executing the instruction.

•If the queue is empty, the EU waits for the next instruction byte to be fetched and shifted to top of the queue.

•When the EU executes a branch or jump instruction, it transfers control to a location corresponding to another set of sequential instructions.

SPECIAL FUNCTIONS OF GENERAL PURPOSE REGISTERS

Accumulator register consists of 2 8-bit registers AL and AH, which can be combined together and used as a 16-bit register AX. AL in this case contains the low-order byte of the word, and AH contains the high-order byte. Accumulator can be used for I/O operations and string manipulation.

Base register consists of 2 8-bit registers BL and BH, which can be combined together and used as a 16-bit register BX. BL in this case contains the low-order byte of the word, and BH contains the high-order byte. BX register usually contains a data pointer used for based, based indexed or register indirect addressing.

Count register consists of 2 8-bit registers CL and CH, which can be combined together and used as a 16-bit register CX. When combined, CL register contains the low-order

byte of the word, and CH contains the high-order byte. Count register can be used as a counter in string manipulation and shift/rotate instructions.

Data register consists of 2 8-bit registers DL and DH, which can be combined together and used as a 16-bit register DX. When combined, DL register contains the low-order byte of the word, and DH contains the high-order byte. Data register can be used as a port number in I/O operations. In integer 32-bit multiply and divide instruction the DX register contains high-order word of the initial or resulting number.

SPECIAL FUNCTIONS OF SPECIAL PURPOSE REGISTERS

Stack Pointer (SP) is a 16-bit register pointing to program stack.

Base Pointer (BP) is a 16-bit register pointing to data in stack segment. BP register is usually used for based, based indexed or register indirect addressing.

Source Index (SI) is a 16-bit register. SI is used for indexed, based indexed and register indirect addressing, as well as a source data address in string manipulation instructions.

Destination Index (DI) is a 16-bit register. DI is used for indexed, based indexed and register indirect addressing, as well as a destination data address in string manipulation instructions. The si and di registers (Source Index and Destination Index) have some special purposes as well. You may use these registers as pointers (much like the bx register) to indirectly access memory. You'll also use these registers with the 8086 string instructions when processing character strings.

The bp register (Base Pointer) is similar to the bx register. You'll generally use this register to access parameters and local variables in a procedure.

The sp register (Stack Pointer) has a very special purpose - it maintains the *program stack*. Normally, you would not use this register for arithmetic computations. The proper operation of most programs depends upon the careful use of this register.

SEGMENTATION:

Since address registers and address operands are only 16 bits they can only address 64k bytes. In order toaddress the 20-bit address range of the 8086, physicaladdresses (those that are put on the address bus)are always formed by adding the values of one of the

instruction is executed? The use of segment registers reduces the size ofpointers to 16 bits.

This reduces the code size but also restricts the addressing range of a pointer to 64k bytes. Performing address arithmetic within data structures larger than 64k is

awkward. This is the biggest drawback of the 8086 architecture. We will restrict ourslves to short programs where all of the code, data and stack are placed into thesame 64k segment (i.e. CS=DS=SS).

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

Memory

•Program, data and stack memories occupy the same memory space. As the most of the processor instructions use 16-bit pointers the processor can effectively address only 64 KB of memory.

• To access memory outside of 64 KB the CPU uses special segment registers to specify where the code, stack and data 64 KB segments are positioned within 1 MB of memory (see the "Registers" section below).

•16-bit pointers and data are stored as:

address: low-order byte

address+1: high-order byte

•**Program memory** - program can be located anywhere in memory. Jump and call instructions can be used for short jumps within currently selected 64 KB code segment, as well as for far jumps anywhere within 1 MB of memory.

•All conditional jump instructions can be used to jump within approximately +127 to -127 bytes from current instruction.

•**Data memory** - the processor can access data in any one out of 4 available segments, which limits the size of accessible memory to 256 KB (if all four segments point to different 64 KB blocks).

•Accessing data from the Data, Code, Stack or Extra segments can be usually done by prefixing instructions with the DS:, CS:, SS: or ES: (some registers and instructions by default may use the ES or SS segments instead of DS segment).

•Word data can be located at odd or even byte boundaries. The processor uses two memory accesses to read 16-bit word located at odd byte boundaries. Reading word data from even byte boundaries requires only one memory access.

•**Stack memory** can be placed anywhere in memory. The stack can be located at odd memory addresses, but it is not recommended for performance reasons (see "Data

Memory" above).

Reserved locations:

•0000h - 03FFh are reserved for interrupt vectors. Each interrupt vector is a 32-bit pointer in format segment: offset.

•FFFF0h - FFFFFh - after RESET the processor always starts program execution at the FFFF0h address.

segment registers to the 16-bit address to form a 20-bit address.

The segment registers themselves only contain themost-significant 16 bits of the 20-bit value that iscontributed by the segment registers. The least significantfour bits of thesegment address are alwayszero.

By default, the DS (data segment) is used fordata transfer instructions (e.g. MOV), CS(codesegment) is used with control transfer instructions(e.g. JMP or CALL), and SS is used with the stackpointer (e.g. PUSH or to save/restore addresses duringCALL/RET or INT instructions).

Exercise: If DS contains 0100H, what address will be written by the instruction MOV [2000H],AL? If CX contains 1122H, SP contains 1234H, and SS contains 2000H, what memory values will change and what will be their values when the PUSH CX

Code segment (CS) is a 16-bit register containing address of 64 KB segment with processor instructions. The processor uses CS segment for all accesses to instructions referenced by instruction pointer (IP) register. CS register cannot be changed directly. The CS register is automatically updated during far jump, far call and far return instructions.

Stack segment (SS) is a 16-bit register containing address of 64KB segment with program stack. By default, the processor assumes that all data referenced by the stack pointer (SP) and base pointer (BP) registers is located in the stack segment. SS register can be changed directly using POP instruction.

Data segment (DS) is a 16-bit register containing address of 64KB segment with program data. By default, the processor assumes that all data referenced by general registers (AX, BX, CX, DX) and index register (SI, DI) is located in the data segment. DS register can be changed directly using POP and LDS instructions.

Extra segment (ES) is a 16-bit register containing address of 64KB segment, usually with program data. By default, the processor assumes that the DI register references

the ES segment in string manipulation instructions. ES register can be changed directly using POP and LES instructions.

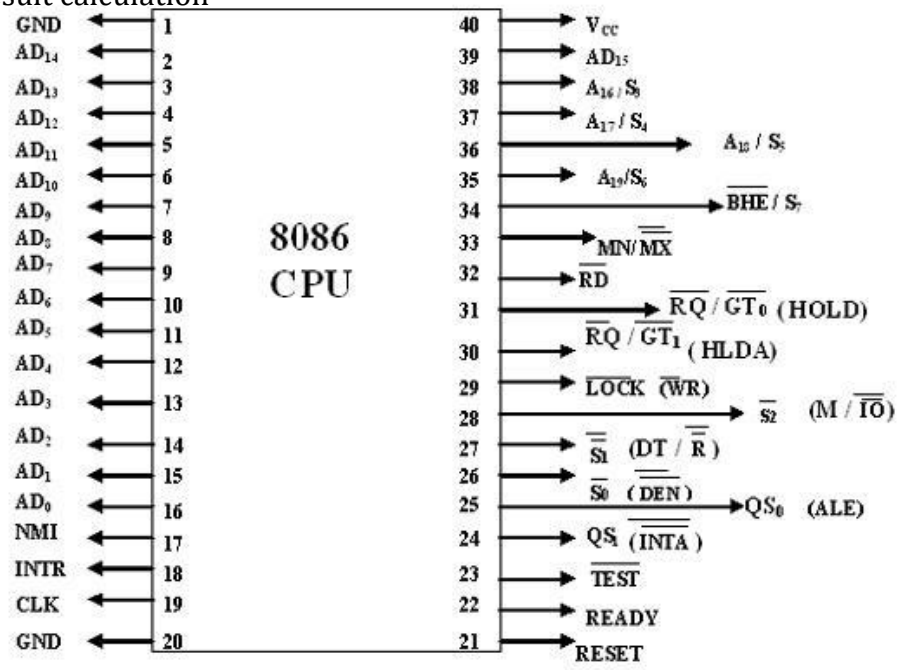
It is possible to change default segments used by general and index registers by prefixing instructions with a CS, SS, DS or ES prefix.

8086 FLAG REGISTER

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



Signal Description of 8086

The Microprocessor 8086 is a 16-bit CPU available in different clock rates and packaged in a 40 pin CERDIP or plastic package.

•The 8086 operates in single processor or multiprocessor configuration to achieve high performance. The pins serve a particular function in minimum mode (single processor mode) and other function in maximum mode configuration (multiprocessor mode).

•The 8086 signals can be categorized in three groups. The first are the signal having common functions in minimum as well as maximum mode

•The second are the signals which have special functions for minimum mode and third are the signals having special functions for maximum mode.

The following signal descriptions are common for both modes.

AD15-AD0 : These are the time multiplexed memory I/O address and data lines.

• Address remains on the lines during T1 state, while the data is available on the data bus during T2, T3, Tw and T4.

• These lines are active high and float to a tristate during interrupt acknowledge and local bus hold acknowledge cycles

A19/S6,A18/S5,A17/S4,A16/S3 : These are the time multiplexed address and status lines.

• During T1 these are the most significant address lines for memory operations.

•During I/O operations, these lines are low. During memory or I/O operations, status information is available on those lines for T2,T3,Tw and T4.

• The status of the interrupt enable flag bit is updated at the beginning of each clock cycle.

•The S4 and S3 combinedly indicate which segment register is presently being used for memory accesses as in below fig.

•These lines float to tri-state off during the local bus hold acknowledge. The status line S6 is always low .

•The address bit are separated from the status bit using latches controlled by the ALE signal.

BHE /S7 : The bus high enable is used to indicate the transfer of data over the higherorder (D15-D8) data bus as shown in table. It goes low for the data transfer over

D15- D8 and is used to derive chip selects of odd address memory bank or peripherals. BHE is low during T1 for read, write and interrupt acknowledge cycles, whenever a byte is to be transferred on higher byte of data bus. The status informatIon is available during T2, T3 and T4. The signal is active low and tristated during hold. It is low during T1 for the first pulse of the interrupt acknowledge cycle.

BHE	A_0	Indication
0	0	Whole word
0	1	Upper byte from or to even address
1	0	Lower byte from or to even address
1	1	None

RD – Read : This signal on low indicates the peripheral that the processor is performing s memory or I/O read operation. RD is active low and shows the state for T2, T3, Tw of any read cycle. The signal remains tristated during the hold acknowledge.

•**READY** : This is the acknowledgement from the slow device or memory that they have completed the data transfer. The signal made available by the devices is synchronized by the 8284A clock generator to provide ready input to the 8086. the signal is active high.

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

•This can be internally masked by resulting the interrupt enable flag. This signal is active high and internally synchronized.

• **TEST** : This input is examined by a 'WAIT' instruction. If the TEST pin goes low, execution will continue, else the processor remains in an idle state. The input is synchronized internally during each clock cycle on leading edge of clock.

•**CLK**- Clock Input : The clock input provides the basic timing for processor operation and bus control activity. Its an asymmetric square wave with 33% duty cycle

MN/ MX : The logic level at this pin decides whether the processor is to operate in either minimum or maximum mode.

•The following pin functions are for the minimum mode operation of 8086.

•M/ IO – Memory/IO : This is a status line logically equivalent to S2 in maximum mode. When it is low, it indicates the CPU is having an I/O operation, and when it is high, it indicates that the CPU is having a memory operation. This line Becomes active high in the previous T4 and remains active till final T4 of the current cycle. It is tri stated

during local bus “hold acknowledge “.

INTA – Interrupt Acknowledge : This signal is used as a read strobe for interrupt acknowledge cycles. i.e. when it goes low, the processor has accepted the interrupt.

ALE – Address Latch Enable :This output signal indicates the availability of the valid address on the address/data lines, and is connected to latch enable input of latches. This signal is active high and is never tri stated.

•**DT/ R – Data Transmit/Receive**: This output is used to decide the direction of data flow through the transceivers (bidirectional buffers). When the processor sends out data, this signal is high and when the processor is receiving data, this signal is low.

•**DEN – Data Enable** :This signal indicates the availability of valid data over the address/data lines. It is used to enable the transceivers (bi directional buffers) to separate the data from the multiplexed address/data signal. It is active from the middle of T2 until the middle of T4. This is tri stated during hold acknowledge’ cycle.

HOLD, HLDA- Acknowledge : When the HOLD line goes high, it indicates to the processor that another master is requesting the bus access.•The processor, after receiving the HOLD request, issues the hold acknowledge signal on HLDA pin, in the middle of the next clock cycle after completing the current bus cycle.•At the same time, the processor floats the local bus and control lines. When the processor detects the HOLD line low, it lowers the HLD**A** signal. HOLD is an asynchronous input, and is should be externally synchronized.•If the DMA request is made while the CPU is performing a memory or I/O cycle, it will release the local bus during T4 provided :

1.The request occurs on or before T2 state of the current cycle.

2.The current cycle is not operating over the lower byte of a word.

3.The current cycle is not the first acknowledge of an interrupt acknowledge sequence.

4. A Lock instruction is not being executed

The following pin function are applicable for maximum mode operation of 8086.

•S2, S1, S0 – Status Lines : These are the status lines which reflect the type of operation, being carried out by the processor. These become activity during T4 of the previous cycle and active during T1 and T2 of the current bus cycles.

S_2	S_1	S_0	Indication
0	0	0	Interrupt Acknowledge
0	0	1	Read I/O port
0	1	0	Write I/O port
0	1	1	Halt
1	0	0	Code Access
1	0	1	Read memory
1	1	0	Write memory
1	1	1	Passive

LOCK: This output pin indicates that other system bus master will be prevented from gaining the system bus, while the LOCK signal is low

•The LOCK signal is activated by the 'LOCK' prefix instruction and remains active until the completion of the next instruction. When the CPU is executing a critical instruction which requires the system bus, the LOCK prefix instruction ensures that other processors connected in the system will not gain the control of the bus.

• The 8086, while executing the prefixed instruction, asserts the bus lock signal output, which may be connected to an external bus controller.

•**QS1, QS0 – Queue Status:** These lines give information about the status of the code-prefetch queue. These are active during the CLK cycle after while the queue operation is performed.

•This modification in a simple fetch and execute architecture of a conventional microprocessor offers an added advantage of pipelined processing of the instructions.

•The 8086 architecture has 6-byte instruction prefetch queue. Thus even the largest (6-bytes) instruction can be prefetched from the memory and stored in the prefetch. This results in a faster execution of the instructions.

•In 8085 an instruction is fetched, decoded and executed and only after the execution of this instruction, the next one is fetched.

•By prefetching the instruction, there is a considerable speeding up in instruction execution in 8086. This is known as ***instruction pipelining***.

•At the starting the CS:IP is loaded with the required address from which the execution is to be started. Initially, the queue will be empty an the microprocessor starts a fetch operation to bring one byte (the first byte) of instruction code, if the CS:IP address is

odd or two bytes at a time, if the CS:IP address is even.

•The first byte is a complete opcode in case of some instruction (one byte opcode instruction) and is a part of opcode, in case of some instructions (two byte opcode instructions), the remaining part of code lie in second byte.

•The second byte is then decoded in continuation with the first byte to decide the instruction length and the number of subsequent bytes to be treated as instruction data.

•The queue is updated after every byte is read from the queue but the fetch cycle is initiated by BIU only if at least two bytes of the queue are empty and the EU may be concurrently executing the fetched instructions

• **RQ / GT0 , RQ / GT1 – Request/Grant :** These pins are used by the other local bus master in maximum mode, to force the processor to release the local bus at the end of the processor current bus cycle.

•Each of the pin is bidirectional with RQ/GT0 having higher priority than RQ/GT1.

•RQ/GT pins have internal pull-up resistors and may be left unconnected.

Request/Grant sequence is as follows:

1.A pulse of one clock wide from another bus master requests the bus access to 8086.

2.During T4(current) or T1(next) clock cycle, a pulse one clock wide from 8086 to the requesting master, indicates that the 8086 has allowed the local bus to float and that it will enter the 'hold acknowledge' state at next cycle. The CPU bus interface unit is likely to be disconnected from the local bus of the system.

3.A one clock wide pulse from the another master indicates to the 8086 that the hold request is about to end and the 8086 may regain control of the local bus at the next clock cycle. Thus each master to master exchange of the local bus is a sequence of 3 pulses. There must be at least one dead clock cycle after each bus exchange.

•The request and grant pulses are active low.

•For the bus request those are received while 8086 is performing memory or I/O cycle, the granting of the bus is governed by the rules as in case of HOLD and HLDA in minimum mode.

General Bus Operation:

•The 8086 has a combined address and data bus commonly referred as a time multiplexed address and data bus.

• The main reason behind multiplexing address and data over the same pins is the maximum utilization of processor pins and it facilitates the use of 40 pin standard DIP

package. Maximum utilization of processor pins and it facilitates the use of 40 pin standard DIP package.

•The bus can be de multiplexed using a few latches and transceivers, when ever required.

•Basically, all the processor bus cycles consist of at least four clock cycles. These are refered to as T1, T2, T3, T4. The address is transmitted by the processor during T1. It is present on the bus only for one cycle.

•The negative edge of this ALE pulse is used to separate the address and the data or status

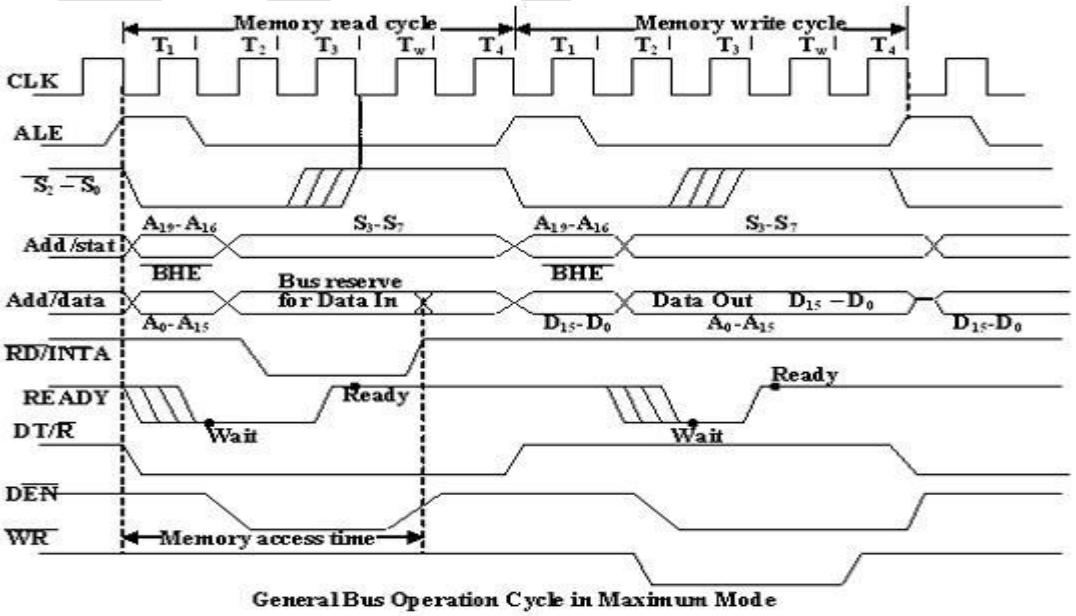
information. In maximum mode, the status lines S0, S1 and S2 are used to indicate the type of operation.

•In a minimum mode 8086 system, the microprocessor 8086 is operated in minimum mode by strapping its MN/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.

•In a minimum mode 8086 system, the microprocessor 8086 is operated in minimum mode by strapping its MN/MX pin to logic 1.



General Bus Operation Cycle in Maximum Mode

•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.

•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 arecontrolled by the ALE signal generated by 8086.

•Transreceivers 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 signals.

•They are controlled by two signals namely, DEN and DT/R The DEN signal indicates the direction of data, i.e. from or to the processor. The system contains memory for the monitor and users program storage.

•Usually, EPROM are used for monitor storage, while RAM for users program storage. A system may contain I/O devices.

•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 T1 with the assertion of address latch enable (ALE) signal and also M / IO 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 T1 to T4 , the M/IO signal indicates a memory or I/O operation.

•At T2, the address is removed from the local bus and is sent to the output. The bus is then tristated. The read (RD) control signal is also activated in T2.

•The read (RD) signal causes the address device to enable its data bus drivers. After 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 M/IO signal is again asserted to indicate a memory or I/O operation. In T2, after sending the address in T1, the processor sends the data to be written to the addressed location.

•The data remains on the bus until middle of T4 state. The WR becomes active at the beginning of T2 (unlike RD is somewhat delayed in T2 to provide time for floating).

•The BHE and A0 signals are used to select the proper byte or bytes of memory or I/O word to be read or write.

•The M/IO, RD and WR signals indicate the type of data transfer as specified in table below.

Hold Response sequence: The HOLD pin is checked at leading edge of each clock pulse. If it is received active by the processor before T4 of the previous cycle or during T1 state of the current cycle, the CPU activates HLDA in the next clock cycle and for succeeding bus cycles, the bus will be given to another requesting master.

•The control of the bus is not regained by the processor until the requesting master does not drop the HOLD pin low. When the request is dropped by the requesting master, the HLDA is dropped by the processor at the trailing edge of the next clock.

Maximum Mode 8086 System

•In the maximum mode, the 8086 is operated by strapping the MN/MX pin to ground.

•In this mode, the processor derives the status signal S2, S1, S0. Another chip called bus controller derives the control signal using this status information .

•In the maximum mode, there may be more than one microprocessor in the system configuration.

•The components in the system are same as in the minimum mode system.

•The basic function of the bus controller chip IC8288, is to derive control signals like RD and WR (for memory and I/O devices), DEN, DT/R, ALE etc. using the information by the processor on the status lines.

•The bus controller chip has input lines S2, S1, S0 and CLK. These inputs to 8288 are driven by CPU.

•It derives the outputs ALE, DEN, DT/R, MRDC, MWTC, AMWC, IORC, IOWC and AIOWC. The AEN, IOB and CEN pins are specially useful for multiprocessor systems.

•AEN and IOB are generally grounded. CEN pin is usually tied to +5V. The significance of the MCE/PDEN output depends upon the status of the IOB pin.

•If IOB is grounded, it acts as master cascade enable to control cascade 8259A, else it acts as peripheral data enable used in the multiple bus configurations.

•INTA pin used to issue two interrupt acknowledge pulses to the interrupt controller or to an interrupting device

IORC, 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 address port.

•The MRDC, MWTC are memory read command and memory write command signals respectively and may be used as memory read or write signals.

•All these command signals instructs the memory to accept or send data from or to the bus.

•For both of these write command signals, the advanced signals namely AIOWC and AMWTC are available.

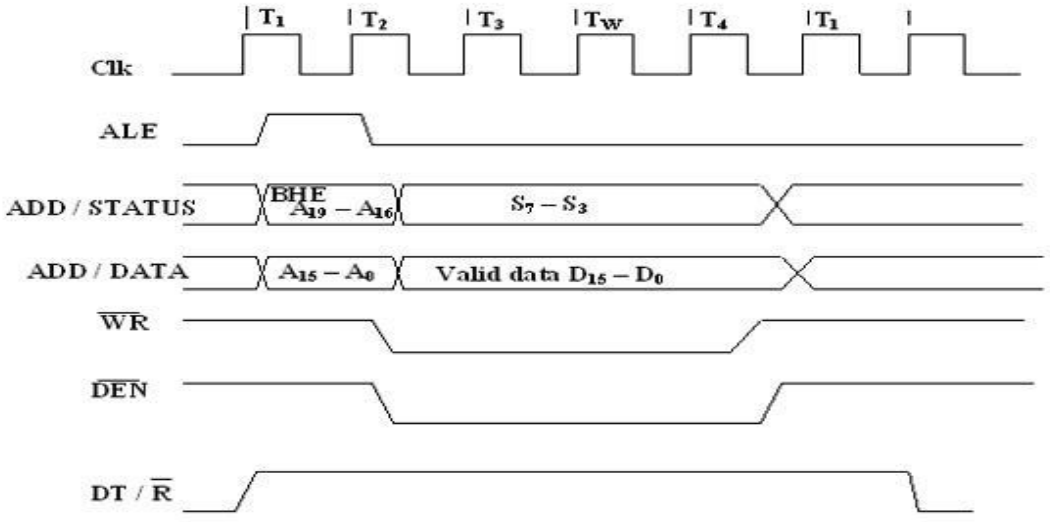
•Here the only difference between in timing diagram between minimum mode and maximum mode is the status signals used and the available control and advanced command signals.

•R0, S1, S2 are set at the beginning of bus cycle.8288 bus controller will output a pulse as on the ALE and apply a required signal to its DT / R pin during T1.

•In T2, 8288 will set DEN=1 thus enabling transceivers, and for an input it will activate MRDC or IORC. These signals are activated until T4. For an output, the AMWC or AIOWC is activated from T2 to T4 and MWTC or IOWC is activated from T3 to T4.

•The status bit S0 to S2 remains active until T3 and become passive during T3 and T4.

•If reader input is not activated before T3, wait state will be inserted between T3 and T4.



Write Cycle Timing Diagram for Minimum Mode

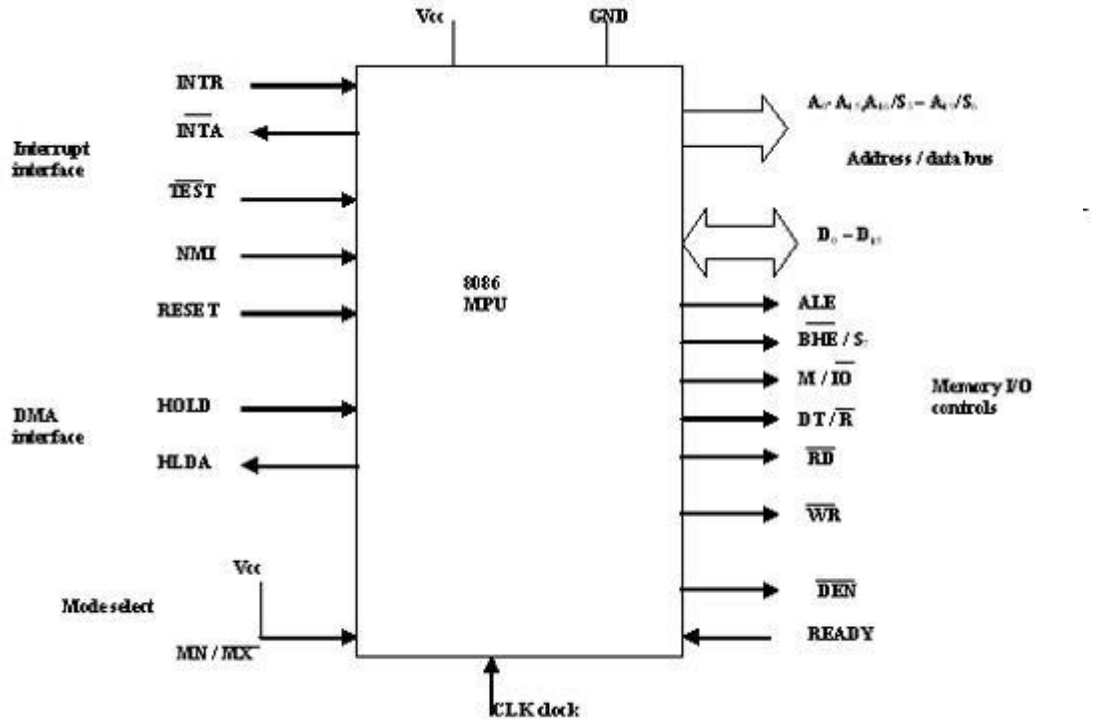
•Timings for RQ/ GT Signals :

The request/grant response sequence contains a series of three pulses. The request/grant pins are checked at each rising pulse of clock input.

When a request is detected and if the condition for HOLD request is satisfied, the processor issues a grant pulse over the RQ/GT pin immediately during T4 (current) or

T1 (next) state.

When the requesting master receives this pulse, it accepts the control of the bus; it sends a release pulse to the processor using RQ/GT pin.



Block Diagram of the Minimum Mode 8086 MPU

Minimum Mode Interface

•When the Minimum mode operation is selected, the 8086 provides all control signals needed to implement the memory and I/O interface.The minimum mode signal can be divided into the following basic groups : address/data bus, status, control, interrupt and DMA.

•**Address/Data Bus** : these lines serve two functions. As an address bus is 20 bits long and consists of signal lines A0 through A19. A19 represents the MSB and A0 LSB. A 20bit address gives the 8086 a 1Mbyte memory address space. More over it has an independent

I/O address space which is 64K bytes in length.

•The 16 data bus lines D0 through D15 are actually multiplexed with address lines A0 through A15 respectively. By multiplexed we mean that the bus work as an address bus during first machine cycle and as a data bus during next machine cycles. D15 is the MSB and D0 LSB.

•When acting as a data bus, they carry read/write data for memory, input/output data

for

I/O devices, and interrupt type codes from an interrupt controller.

S_4	S_3	Segment Register
0	0	Extra
0	1	Stack
1	0	Code / none
1	1	Data

Memory segment status codes.

Status signal:

The four most significant address lines A19 through A16 are also multiplexed but in this case with status signals S6 through S3. These status bits are output on the bus at the same

time that data are transferred over the other bus lines.

•Bit S4 and S3 together from a 2 bit binary code that identifies which of the 8086 internal

segment registers are used to generate the physical address that was output on the address

bus during the current bus cycle.

•Code S4S3 = 00 identifies a register known as ***extra segment register*** as the source of the segment address.

•Status line S5 reflects the status of another internal characteristic of the 8086. It is the logic level of the internal enable flag. The last status bit S6 is always at the logic 0 level

•**Control Signals** :

The control signals are provided to support the 8086 memory I/O interfaces. They

control functions such as when the bus is to carry a valid address in which direction data
are to be transferred over the bus, when valid write data are on the bus and when to put read data on the system bus.
ALE is a pulse to logic 1 that signals external circuitry when a valid address word is on the bus. This address must be latched in external circuitry on the 1-to-0 edge of the pulse
•Another control signal that is produced during the bus cycle is BHE bank high enable. Logic 0 on this used as a memory enable signal for the most significant byte half of the data bus D8 through D1. These lines also serves a second function, which is as the S_7 status line.
•Using the M/IO and DT/R lines, the 8086 signals which type of bus cycle is in progress and in which direction data are to be transferred over the bus.
•The logic level of M/IO tells external circuitry whether a memory or I/O transfer is taking place over the bus. Logic 1 at this output signals a memory operation and logic 0 an I/O operation.
•The direction of data transfer over the bus is signaled by the logic level output at DT/R. When this line is logic 1 during the data transfer part of a bus cycle, the bus is in the transmit mode. Therefore, data are either written into memory or output to an I/O device.
•On the other hand, logic 0 at DT/R signals that the bus is in the receive mode. This corresponds to reading data from memory or input of data from an input port.
•The signal read RD and write WR indicates that a read bus cycle or a write bus cycle is in progress. The 8086 switches WR to logic 0 to signal external device that valid write or output data are on the bus
• On the other hand, RD indicates that the 8086 is performing a read of data of the bus. During read operations, one other control signal is also supplied. This is DEN (data enable) and it signals external devices when they should put data on the bus.
•There is one other control signal that is involved with the memory and I/O interface. This is the READY signal.
•READY signal is used to insert wait states into the bus cycle such that it is extended by a number of clock periods. This signal is provided by an external clock generator device and can be supplied by the memory or I/O sub-system to signal the 8086 when they are

ready to permit the data transfer to be completed

Maximum Mode Interface

•When the 8086 is set for the maximum-mode configuration, it provides signals for implementing a multiprocessor / coprocessor system environment.

•By multiprocessor environment we mean that one microprocessor exists in the system and that each processor is executing its own program.

• Usually in this type of system environment, there are some system resources that are common to all processors.

•They are called as ***global resources***. There are also other resources that are assigned to

specific processors. These are known as ***local or private resources***.

•Coprocessor also means that there is a second processor in the system. In this two processor does not access the bus at the same time.

•One passes the control of the system bus to the other and then may suspend its operation.

•In the maximum-mode 8086 system, facilities are provided for implementing allocation of global resources and passing bus control to other microprocessor or coprocessor .

UNIT-II

ADDRESSING MODES OF 8086:

Implied - the data value/data address is implicitly associated with the instruction.

Direct - the instruction operand specifies the memory address where data is located.

•**Register indirect** - instruction specifies a register containing an address, where data is located. This addressing mode works with SI, DI, BX and BP registers.

•**Register** - references the data in a register or in a register pair.

•**Immediate** - the data is provided in the instruction.

•**Based** :- 8-bit or 16-bit instruction operand is added to the contents of a base register (BX or BP), the resulting value is a pointer to location where data resides.

•**Indexed** :- 8-bit or 16-bit instruction operand is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides

•**Based Indexed** :- the contents of a base register (BX or BP) is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.

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

Instruction set of 8086:

Data transfer instructions

GENERAL – PURPOSE BYTE OR WORD TRANSFER INSTRUCTIONS:

MOV

PUSH

POP

XCHG

XLAT

SIMPLE INPUT AND OUTPUT PORT TRANSFER INSTRUCT

IN

OUT

SPECIAL ADDRESS TRANSFER INSTRUCTIONS

LEA

LDS

LES

FLAG TRANSFER INSTRUCTIONS:

LAHF

SAHF

PUSHF

POPF

ADITION INSTRUCTIONS:

ADD

ADC

INC

AAA

DAA

SUBTSUBTRACTION INSTRUCTIONS:

SUB

SBB

DEC

NEG

CMP

AAS

DAS

MULTIPLICATION INSTRUCTIONS:

MUL

IMUL

AAM

DIVISION INSTRUCTIONS:

DIV

IDIV

AAD

CBW

CWD

BIT MANIPULATION INSTRUCTIONS

LOGICAL INSTRUCTIONS:

NOT

AND

OR

XOR

TEST

SHIFT INSTRUCTIONS:

SHL / SAL

SHR

SAR

PROGRAM EXECUTION TRANSFER INSTRUCTIONS

UNCONDITIONAL TRANSFER INSTRUCTIONS:

CALL

RET

JMP

CONDITIONAL TRANSFER INSTRUCTIONS:

JA / JNBE

JAE / JNB

JB / JNAE

JBE / JNA

JC

JE / JZ

JG / JNLE

JGE / JNL

JL / JNGE

JLE / JNG

JNC

JNE / JNZ

JNO

JNP / JPO

JNS

JO

JP / JPE

JS

ITERATION CONTROL INSTRUCTIONS:

LOOP

LOOPE / LOOPZ

LOOPNE / LOOPNZ

JCXZ

INTERRUPT INSTRUCTIONS:

INT

INTO

IRET

PROCESS CONTROL INSTRUCTIONS

FLAG SET / CLEAR INSTRUCTIONS:

STC

CLC

CMC

STD

CLD

STI

CLI

EXTERNAL HARDWARE SYNCHRONIZATION INSTRUCTIONS:

HLT

WAIT

ESC

LOCK

NOP

Instruction Description

AAA Instruction - ASCII Adjust after Addition

AAD Instruction - ASCII adjust before Division

AAM Instruction - ASCII adjust after Multiplication

AAS Instruction - ASCII Adjust for Subtraction

ADC Instruction - Add with carry.

ADD Instruction - ADD destination, source

AND Instruction - AND corresponding bits of two operands

Unit-III

8255-PPI:

The 8255 is a general purpose programmable I/O device used for parallel data transfer. It has 24 I/O pins which can be grouped in three 8-bit parallel ports : Port A, Port B and Port C. The eight bits of port C can be used as individual bits or be grouped in two 4-bit ports : C_{upper} (C_U) and C_{lower} (C_L).

The 8255, primarily, can be programmed in two basic modes : Bit Set/Reset (BSR) mode and I/O mode. The BSR mode is used to set or reset the bits in port C. The I/O mode is further divided into three modes :

Mode 0 : Simple Input/Output
Mode 1 : Input/Output with handshake
Mode 2 : Bi-directional I/O data transfer

The function of I/O pins (input or output) and modes of operation of I/O ports can be programmed by writing proper control word in the control word register. Each bit in the control word has a specific meaning and the status of these bits decides the function and operating mode of the I/O ports.

Features of 8255A

1. The 8255A is a widely used, programmable, parallel I/O device.
2. It can be programmed to transfer data under various conditions, from simple I/O to interrupt I/O.
3. It is compatible with all Intel and most other microprocessors.
4. It is completely TTL compatible.
5. It has three 8-bit ports : Port A, Port B, and Port C, which are arranged in two groups of 12 pins. Each port has an unique address, and data can be read from or written to a port. In addition to the address assigned to the three ports, another address is assigned to the control register into which control words are written for programming the 8255 to operate in various modes.
6. Its bit set/reset mode allows setting and resetting of individual bits of Port C.
7. The 8255 can operate in 3 I/O modes : (i) Mode 0, (ii) Mode 1, and (iii) Mode 2.

a) In Mode 0, Port A and Port B can be configured as simple 8-bit input or output ports without handshaking. The two halves of Port C can be programmed separately as 4-bit input or output ports.

b) In Mode 1, two groups each of 12 pins are formed. Group A consists of Port A and the upper half of Port C while Group B consists of Port B and the lower half of Port C. Ports A and B can be programmed as 8-bit Input or Output ports with three lines of Port C in each group used for handshaking.

c) In Mode 2, only Port A can be used as a bidirectional port. The handshaking signals are provided on five lines of Port C (PC_3 - PC_7). Port B can be used in Mode 0 or in Mode 1.

8. All I/O pins of 8255 has 2.5 mA DC driving capacity (i.e. sourcing current of 2.5 mA).

Block Diagram

Fig. 7.2 shows the internal block diagram of 8255A. It consists of data bus buffer. control logic and Group A and Group B controls.

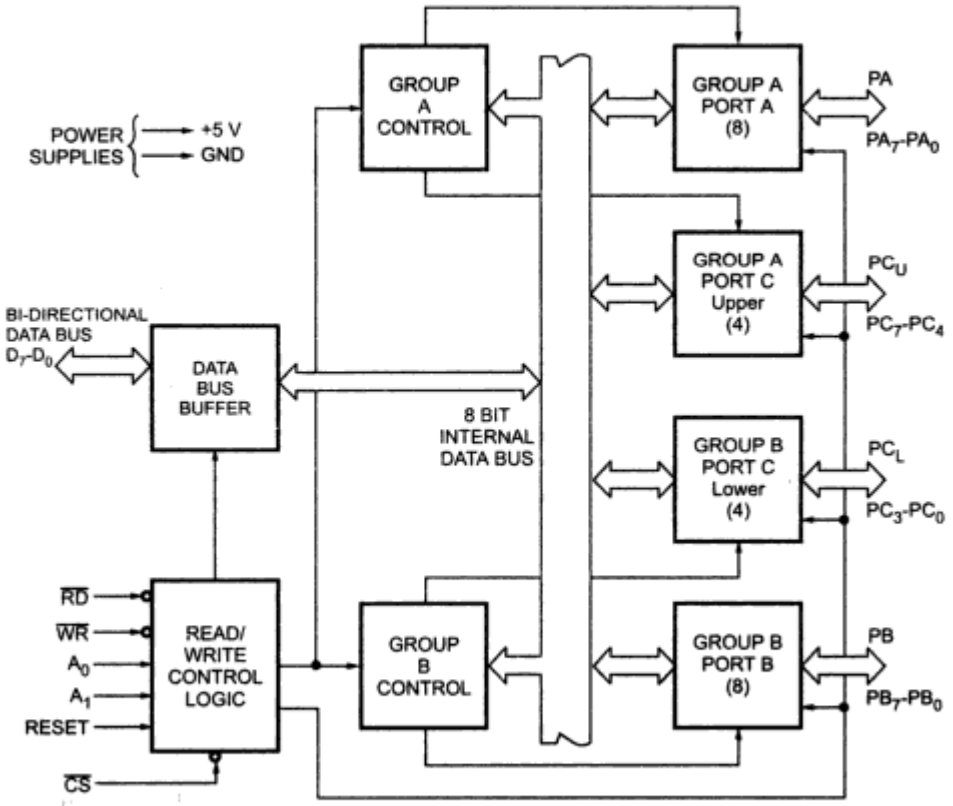


Fig. 7.2 Block diagram of 8255A

Mode 1 : Input/Output with handshake

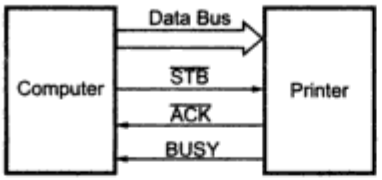


Fig. 7.3 Data transfer between computer and printer using handshaking signals

In this mode, input or output data transfer is controlled by handshaking signals. Handshaking signals are used to transfer data between devices whose data transfer speeds are not same. For example, computer can send data to the printer with large speed but printer can't accept data and print data with this rate. So computer has to send data with the speed with which printer can accept. This type of data transfer is achieved by using handshaking signals alongwith data signals. Fig. 7.3 shows data transfer between computer and printer using handshaking signals.

These handshaking signals are used to tell computer whether printer is ready to accept the data or not. If printer is ready to accept the data then after sending data on data bus, computer uses another handshaking signal ($\overline{STB}$) to tell printer that valid data is available on the data bus.

The 8255 mode 1 which supports handshaking has following features.

1. Two ports (A and B) function as 8-bit I/O ports. They can be configured either as input or output ports.
2. Each port uses three lines from Port C as handshake signals. The remaining two lines of Port C can be used for simple I/O functions.
3. Input and output data are latched.
4. Interrupt logic is supported.

Mode 2 : Bi-directional I/O data transfer

This mode allows bi-directional data transfer (transmission and reception) over a single 8-bit data bus using handshaking signals. This feature is available only in Group A with Port A as the 8-bit bi-directional data bus; and $PC_3 - PC_7$ are used for handshaking purpose. In this mode, both inputs and outputs are latched. Due to use of a single 8-bit data bus for bi-directional data transfer, the data sent out by the CPU through Port A appears on the bus connecting it to the peripheral, only when the peripheral requests it. The remaining lines of Port C i.e. PC_0-PC_2 can be used for simple I/O functions. The Port B can be programmed in mode 0 or in mode 1. When Port B is programmed in mode 1, PC_0-PC_2 lines of Port C are used as handshaking signals.

7.5 Control Word Formats

A high on the RESET pin causes all 24 lines of the three 8-bit ports to be in the input mode. All flip-flops are cleared and the interrupts are reset. This condition is maintained even after the RESET goes low. The ports of the 8255 can then be programmed for any other mode by writing a single control word into the control register, when required.

For Bit Set/Reset Mode

Fig. 7.4 shows bit set/reset control word format.

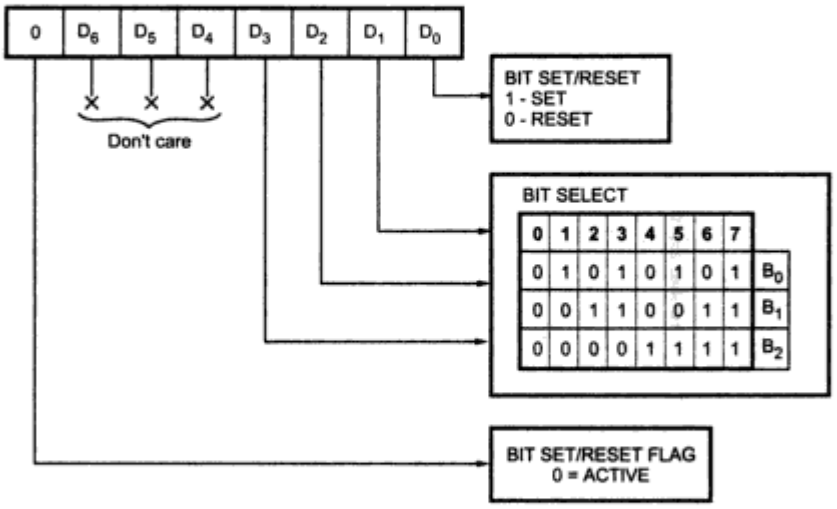


Fig. 7.4 Bit set/reset control word format

The eight possible combinations of the states of bits D_3 - D_1 (B_2 B_1 B_0) in the Bit Set-Reset format (BSR) determine particular bit in PC_0 - PC_7 being set or reset as per the status of bit D_0. A BSR word is to be written for each bit that is to be set or reset. For example, if bit PC_3 is to be set and bit PC_4 is to be reset, the appropriate BSR words that will have to be loaded into the control register will be, 0XXX0111 and 0XXX1000, respectively, where X is don't care.

The BSR word can also be used for enabling or disabling interrupt signals generated by Port C when the 8255 is programmed for Mode 1 or 2 operation. This is done by setting or resetting the associated bits of the interrupts. This is described in detail in next section.

For I/O Mode

The mode definition format for I/O mode is shown in Fig. 7.5. The control words for both, mode definition and Bit Set-Reset are loaded into the same control register, with bit D_7 used for specifying whether the word loaded into the control register is a mode

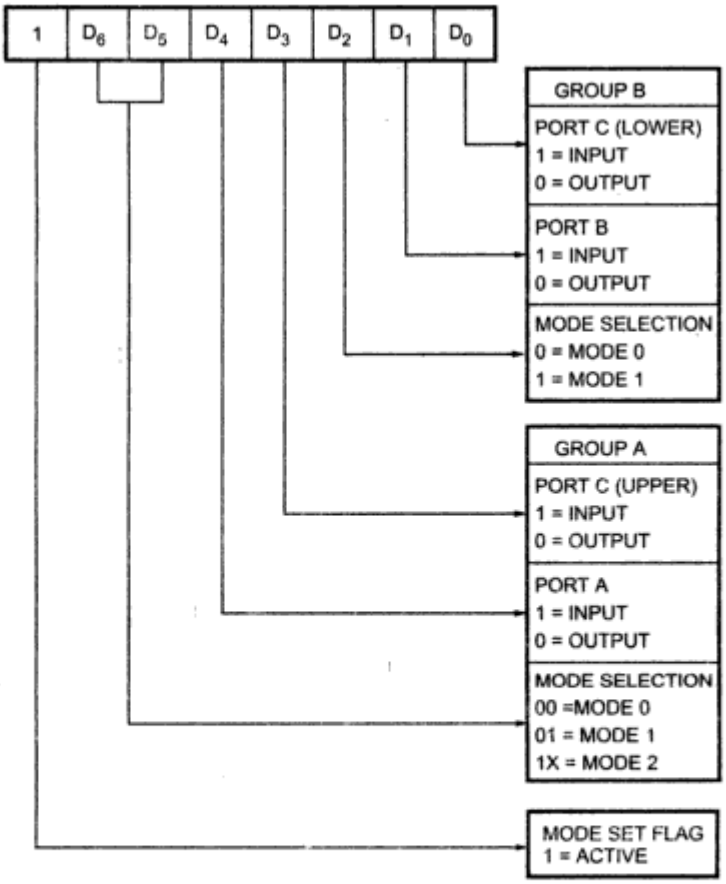


Fig. 7.5 8255 Mode definition format

definition word or Bit Set-Reset word. If D_7 is high, the word is taken as a mode definition word, and if it is low, it is taken as a Bit Set-Reset word. The appropriate bits are set or reset depending on the type of operation desired, and loaded into the control register.

Interfacing 8255 to 8086 in I/O Mapped I/O Mode

The 8086 has four special instructions IN, INS, OUT, and OUTS to transfer data through the input/output ports in I/O mapped I/O system. M/$\overline{IO}$ signal is always low when 8086 is executing these instructions. So M/$\overline{IO}$ signal is used to generate separate addresses for, memory and input/output. Only 256 (2^8) I/O addresses can be generated when direct addressing method is used. By using indirect address method this range can be extended upto 65536 (2^{16}) addresses.

Fig. 7.17 shows the interfacing of 8255 with 8086 in I/O mapped I/O technique. Here, $\overline{RD}$ and $\overline{WR}$ signals are activated when M/$\overline{IO}$ signal is low, indicating I/O bus cycle. Only lower data bus (D_0 – D_7) is used as 8255 is 8-bit device. Reset out signal from clock generator is connected to the Reset signal of the 8255. In case of interrupt driven I/O INTR signal (PC_3 or PC_0) from 8255 is connected to INTR input of 8088.

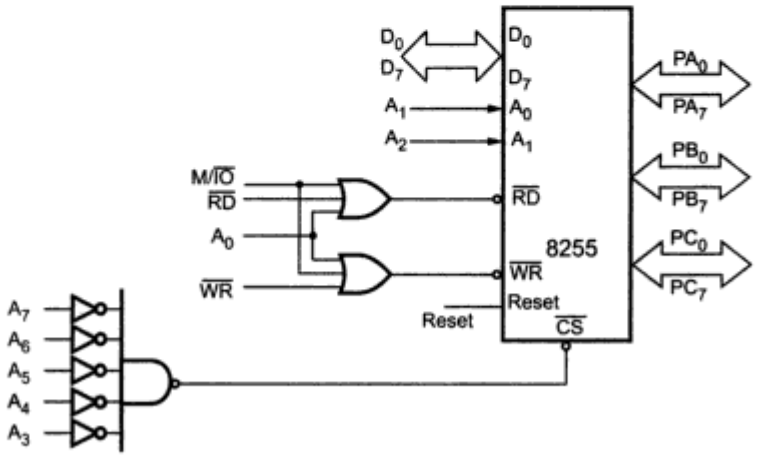


Fig. 7.17 I/O mapped I/O

I/O Map :

Port / control Register	Address lines								Address
	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
Port A	0	0	0	0	0	0	0	0	00H
Port B	0	0	0	0	0	0	1	0	02H
Port C	0	0	0	0	0	1	0	0	04H
Control register	0	0	0	0	0	1	1	0	06H

Note : It is assumed that the direct addressing is used.

Interfacing 8255 to 8086 in Memory Mapped I/O

In this type of I/O interfacing, the 8086 uses 20 address lines to identify an I/O device; an I/O device is connected as if it is a memory register. The 8086 uses same control signals and instructions to access I/O as those of memory. Fig. 7.18 shows the interfacing of 8255 with 8086 in memory mapped I/O technique. Here $\overline{RD}$ and $\overline{WR}$ signals are activated when M/$\overline{IO}$ signal is high, indicating memory bus cycle. Address lines A_0 - A_1 are used by 8255 for internal decoding. To get absolute address, all remaining address lines (A_3 - A_{19}) are used to decode the address for 8255. Other signal connections are same as in I/O mapped I/O.

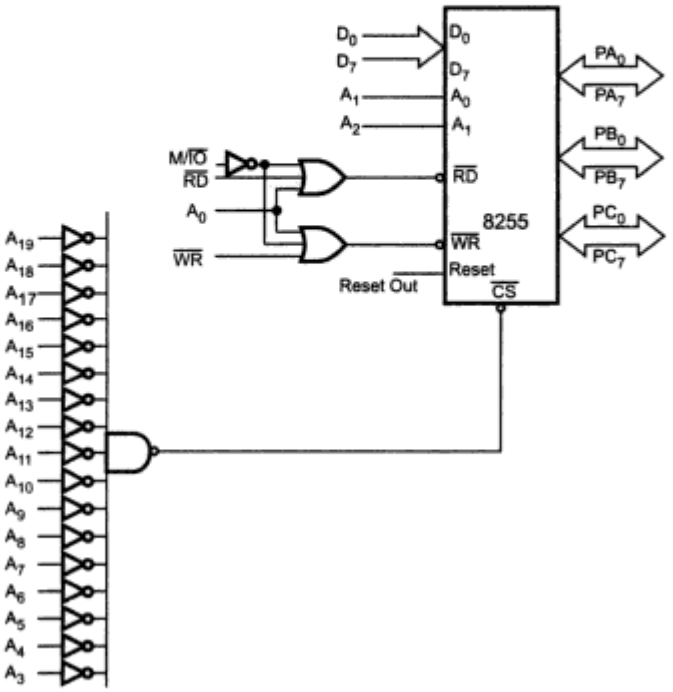


Fig. 7.18 Memory mapped I/O

I/O Map :

Register	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Address
Port A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000H
Port B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	00002H
Port C	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	00004H
Control register	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	00006H

ADC 0808/0809

The ADC 0808 and ADC 0809 are monolithic CMOS devices with an 8-channel multiplexer. These devices are also designed to operate from common microprocessor control buses, with tri-state output latches driving the data bus. The main features of these devices are :

Features

- 8-bit successive approximation ADC.
- 8-channel multiplexer with address logic.
- Conversion time 100 μs.
- It eliminates the need for external zero and full-scale adjustments.
- Easy to interface to all microprocessors.
- It operates on single 5 V power supply.
- Output meet TTL voltage level specifications.

Pin Diagram

Fig. 7.37 shows pin diagram of 0808/0809 ADC.

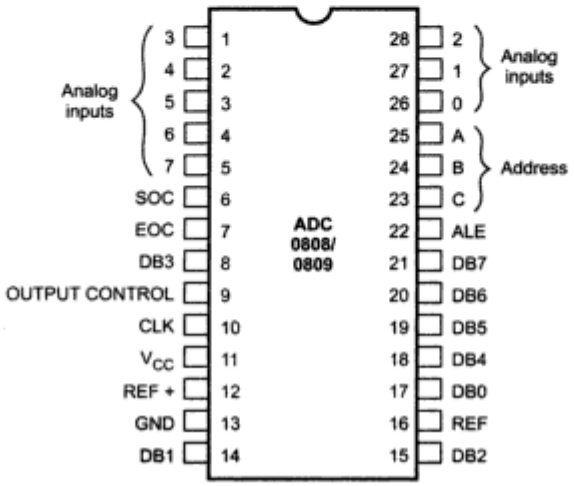


Fig. 7.37 Pin diagram of 0808/0809

Operation

ADC 0808/0809 has eight input channels, so to select desired input channel, it is necessary to send 3-bit address on A, B and C inputs. The address of the desired channel is sent to the multiplexer address inputs through port pins. After at least 50 ns, this address must be latched. This can be achieved by sending ALE signal. After another 2.5 μs, the start of conversion (SOC) signal must be sent high and then low to start the conversion process. To indicate end of conversion ADC 0808/0809 activates EOC signal. The microprocessor system can read converted digital word through data bus by enabling the output enable signal after EOC is activated. This is illustrated in Fig. 7.38.

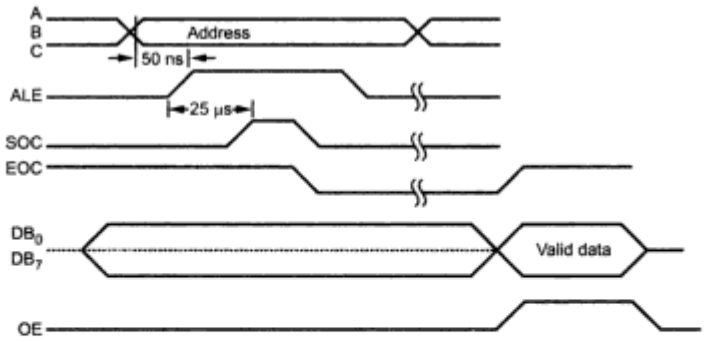


Fig. 7.38 Timing waveforms for ADC 0808

Interfacing

Fig. 7.39 shows typical interfacing circuit for ADC 0808 with microprocessor system.

The zener diode and LM 308 amplifier circuitry is used to produce a V_{CC} an $+V_{REF}$ of 5.12 V for the A/D converter. With this reference voltage the A/D converter will have 256 steps of 20 mV each.

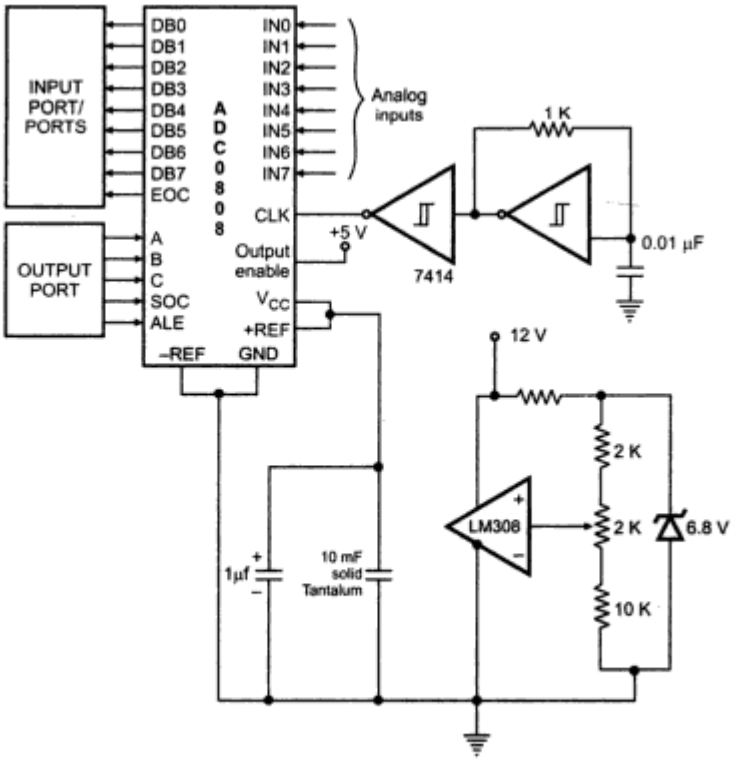


Fig. 7.39 Typical interface for 0808/0809

Stepper Motor Interfacing

A stepper motor is a digital motor. It can be driven by digital signal. Fig. 7.40 shows the typical 2 phase motor interfaced using 8255. Motor shown in the circuit has two phases, with center-tap winding. The center taps of these windings are connected to the 12 V supply. Due to this, motor can be excited by grounding four terminals of the two windings. Motor can be rotated in steps by giving proper excitation sequence to these windings. The lower nibble of port A of the 8255 is used to generate excitation signals in the proper sequence.

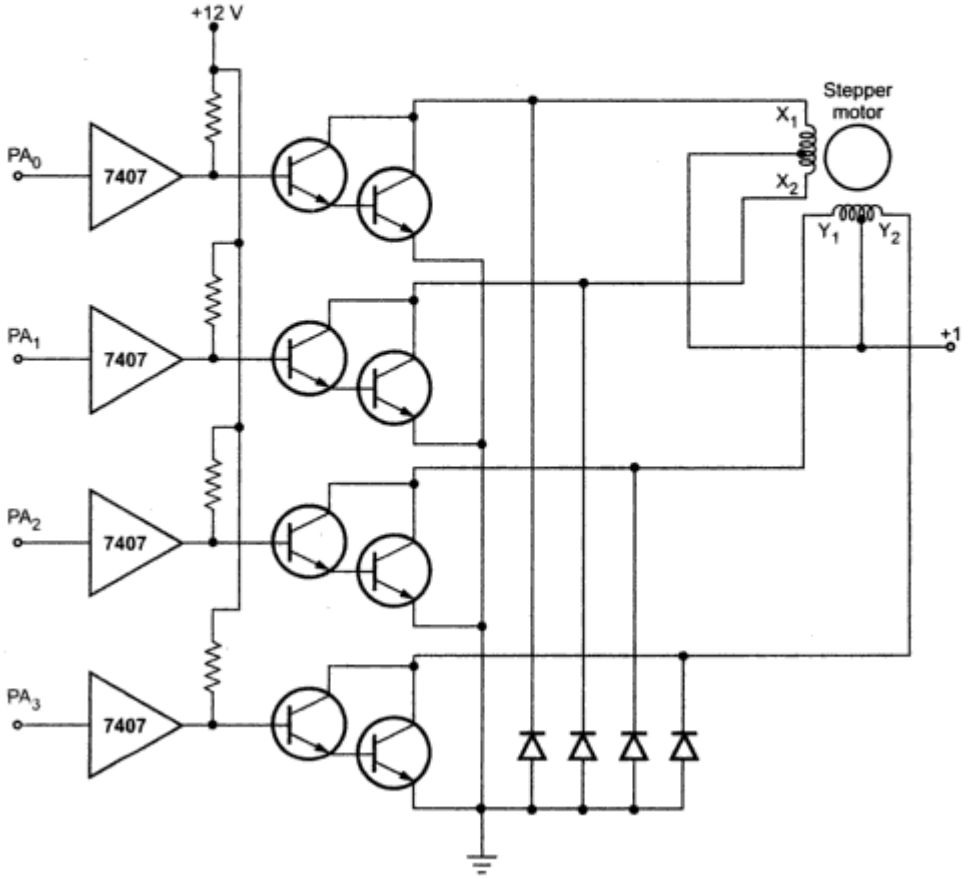


Fig. 7.40 Stepper motor interface

The Table 7.3 shows typical excitation sequence. The given excitation sequence rotates the motor in clockwise direction. To rotate motor in anticlockwise direction we have to excite motor in a reverse sequence. The excitation sequence for stepper motor may change due to change in winding connections. However, it is not desirable to excite both the ends

of the same winding simultaneously. This cancels the flux and motor winding may damage. To avoid this, digital locking system must be designed. Fig. 7.41 shows a simple digital locking system. Only one output is activated (made low) when properly excited; otherwise output is disabled (made high).

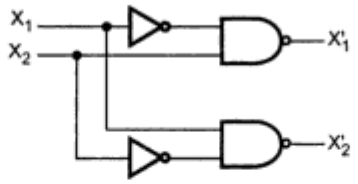


Fig. 7.41 Digital locking system

Step	X_1	X_2	Y_1	Y_2
1	0	1	0	1
2	1	0	0	1
3	1	0	1	0
4	0	1	1	0
1	0	1	0	1

Table 7.3 Full step excitation sequence

The excitation sequence given in Table 7.3 is called **full step sequence** in which excitation ends of the phase are changed in one step. The excitation sequence given in Table 7.4 takes two steps to change the excitation ends of the phase. Such a sequence is called **half step sequence** and in each step the motor is rotated by 0.9°.

Step	X_1	X_2	Y_1	Y_2
1	0	1	0	1
2	0	0	0	1
3	1	0	0	1
4	1	0	0	0
5	1	0	1	0
6	0	0	1	0
7	0	1	1	0
8	0	1	0	0
1	0	1	0	1

Table 7.4 Half step excitation sequence

We know that stepper motor is stepped from one position to the next by changing the currents through the fields in the motor. The winding inductance opposes the change in current and this puts limit on the stepping rate. For higher stepping rates and more torque, it is necessary to use a higher voltage source and current limiting resistors as shown in Fig. 7.42. By adding series resistance, we decrease L/R time constant, which allows the current to change more rapidly in the windings. There is a power loss across series resistor, but designer has to compromise between power and speed.

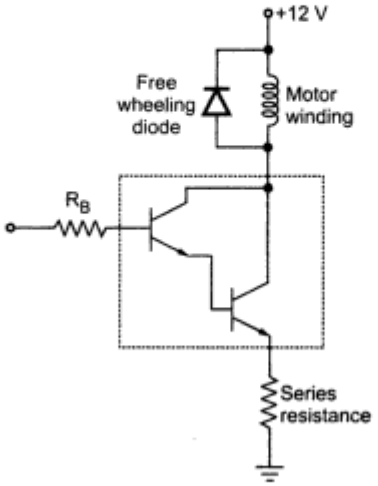


Fig. 7.42 Excitation circuit with series resistance

UNIT-IV

MEMORY INTERFACING:

•***Control connections***: A ROM usually has only one control input, while a RAM often has one or two control inputs.

•The control input most often found on the ROM is the output enable (OE) or gate (G), this allows data to flow out of the output data pins of the ROM.

•If OE and the selected input are both active, then the output is enable, if OE is inactive, the output is disabled at its high-impedance state.

•The OE connection enables and disables a set of three-state buffer located within the memory device and must be active to read data.

•A RAM memory device has either one or two control inputs. If there is one control input it is often called R/ W .

•This pin selects a read operation or a write operation only if the device is selected by the selection input (CS). If the RAM has two control inputs, they are usually labeled WE or W and OE or G .

•(WE) write enable must be active to perform a memory write operation and OE must be active to perform a memory read operation.

•When these two controls WE and OE are present, they must never be active at the same time.

•The ROM read only memory permanently stores programs and data and data was always present, even when power is disconnected.

•It is also called as nonvolatile memory.

•EPROM (erasable programmable read only memory) is also erasable if exposed to high intensity ultraviolet light for about 20 minutes or less, depending upon the type of EPROM.

•We have PROM (programmable read only memory)

•RMM (read mostly memory) is also called the flash memory.

•The flash memory is also called as an EEPROM (electrically erasable programmable ROM), EAROM (electrically alterable ROM), or a NOVROM (nonvolatile ROM).

•These memory devices are electrically erasable in the system, but require more time to erase than a normal RAM.

•EPROM contains the series of 27XXX contains the following part numbers :

2704(512 * 8), 2708(1K * 8), 2716(2K * 8), 2732(4K * 8), 2764(8K * 8), 27128(16K * 8) etc..

•Each of these parts contains address pins, eight data connections, one or more chip selection inputs (CE) and an output enable pin (OE).

•This device contains **11** address inputs and **8** data outputs.

•If both the pin connection CE and OE are at logic 0, data will appear on the output connection . If both the pins are not at logic 0, the data output connections remains at their high impedance or off state.

•To read data from the EPROM Vpp pin must be placed at a logic 1.

Static RAM Interfacing

The semiconductor RAM is broadly two types – Static RAM and Dynamic RAM.

•The semiconductor memories are organized as two dimensional arrays of memory locations.

•For example 4K * 8 or 4K byte memory contains 4096 locations, where each locations contains 8-bit data and only one of the 4096 locations can be selected at a time. Once a location is selected all the bits in it are accessible using a group of conductors called Data bus.

•For addressing the 4K bytes of memory, 12 address lines are required.

•In general to address a memory location out of N memory locations, we will require at least n bits of address, i.e. n address lines where n = Log2 N.

•Thus if the microprocessor has n address lines, then it is able to address at the most N locations of memory, where 2^n=N. If out of N locations only P memory locations are to be interfaced, then the least significant p address lines out of the available n lines can be directly connected from the microprocessor to the memory chip while the remaining (n-p) higher order address lines may be used for address decoding as inputs to the chip selection logic.

•The memory address depends upon the hardware circuit used for decoding the chip select (CS). The output of the decoding circuit is connected with the CS pin of the memory chip.

•The general procedure of static memory interfacing with 8086 is briefly described as follows:

1.Arrange the available memory chip so as to obtain 16- bit data bus width. The upper 8-bit bank is called as odd address memory bank and the lower 8-bit bank is called as

even address memory bank.

2.Connect available memory address lines of memory chip with those of the microprocessor and also connect the memory RD and WR inputs to the corresponding processor control signals. Connect the 16-bit data bus of the memory bank with that of the microprocessor 8086.

3.The remaining address lines of the microprocessor, BHE and A0 are used for decoding the required chip select signals for the odd and even memory banks. The CS of memory is derived from the o/p of the decoding circuit.

•As a good and efficient interfacing practice, the address map of the system should be continuous as far as possible, i.e. there should not be no windows in the mapand no fold back space should be allowed. A memory location should have a single address corresponding to it, i.e. absolute decoding should be preferred and minimum hardware should be used for decoding large capacity memory is required in a microcomputer system, the memory subsystem is generally designed using dynamic RAM because there are various advantages of dynamic RAM.

• E.g. higher packing density, lower cost and less power consumption. A typical static RAM cell may require six transistors while the dynamic RAM cell requires only a transistors along with a capacitor. Hence it is possible to obtain higher packaging density and hence low cost units are available.

•The basic dynamic RAM cell uses a capacitor to store the charge as a representation of data. This capacitor is manufactured as a diode that is reverse- biased so that the storage capacitance comes into the picture.

•This storage capacitance is utilized for storing the charge representation of data but the reverse-biased diode has leakage current that tends to discharge the capacitor giving rise to the possibility of data loss. To avoid this possible data loss, the data stored in a dynamic RAM cell must be refreshed after a fixed time interval regularly. The process of refreshing the data in RAM is called as

Refresh cycle.

•The refresh activity is similar to reading the data from each and every cell of memory, independent of the requirement of microprocessor. During this refresh period all other operations related to the memory subsystem are suspended. Hence the refresh activity causes loss of time, resulting in reduce system performance.

•However keeping in view the advantages of dynamic RAM, like low power consumption, high packaging density and low cost, most of the advanced computing system are designed using dynamic RAM, at the cost of operating speed.

• A dedicated hardware chip called as dynamic RAM controller is the most important part of the interfacing circuit.

•The ***Refresh cycle*** is different from the memory read cycle in the following aspects.

1.The memory address is not provided by the CPU address bus, rather it is generated by a refresh mechanism counter called as refresh counter.

2.Unlike memory read cycle, more than one memory chip may be enabled at a time so as to reduce the number of total memory refresh cycles.

3.The data enable control of the selected memory chip is deactivated, and data is not allowed to appear on the system data bus during refresh, as more than one memory units are refreshed simultaneously. This is to avoid the data from the different chips to appear on the bus simultaneously.

4.Memory read is either a processor initiated or an external bus master initiated and carried out by the refresh mechanism. Dynamic RAM is available in units of several kilobits to megabits of memory.

This memory is arranged internally in a two dimensional matrix array so that it will have n rows and m columns. The row address n and column address m are important for the refreshing operation.

• For example, a typical 4K bit dynamic RAM chip has an internally arranged bit array of dimension 64 * 64 , i.e. 64 rows and 64 columns. The row address and column address will require 6 bits each. These 6 bits for each row address and column address will be generated by the refresh counter, during the refresh cycles.

•A complete row of 64 cells is refreshed at a time to minimizes the refreshing time. Thus the refresh counter needs to generate only row addresses. The row address are multiplexed, over lower order address lines.

•The refresh signals act to control the multiplexer, i.e. when refresh cycle is in process the refresh counter puts the row address over the address bus for refreshing. Otherwise, the address bus of the processor is connected to the address bus of DRAM, during normal processor initiated activities.

• A timer, called refresh timer, derives a pulse for refreshing action after each refresh

interval.

•Refresh interval can be qualitatively defined as the time for which a dynamic RAM cell can hold data charge level practically constant, i.e. no data loss takes place.

• Suppose the typical dynamic RAM chip has 64 rows, then each row should be refreshed after each refresh interval or in other words, all the 64 rows are to refreshed in a single refresh interval.

•This refresh interval depends upon the manufacturing technology of the dynamic RAM cell. It may range anywhere from 1ms to 3ms.

• Let us consider 2ms as a typical refresh time interval. Hence, the frequency of the refresh pulses will be calculated as follows:

•Refresh Time (per row) tr = (2 * 10 -3) / 64.

•Refresh Frequency fr = 64 / (2 * 10 -3) = 32 * 103 Hz.

•The following block diagram explains the refreshing logic and 8086 interfacing with dynamic RAM.

•Each chip is of 16K * 1-bit dynamic RAM cell array. The system contains two 16K byte dynamic RAM units. All the address and data lines are assumed to be available from an 8086 microprocessor system.

•The OE pin controls output data buffer of the memory chips. The CE pins are active high chip selects of memory chips. The refresh cycle starts, if the refresh output of the refresh timer goes high, OE and CE also tend to go high.

•The high CE enables the memory chip for refreshing, while high OE prevents the data from appearing on the data bus, as discussed in memory refresh cycle. The 16K * 1-bit dynamic RAM has an internal array of 128*128 cells, requiring 7 bits for row address. The lower order seven lines A_0-A_6 are multiplexed with the refresh counter output A10-A16.

•If the RAM has two control inputs, they are usually labeled WE or W and OE or G .

•(WE) write enable must be active to perform a memory write operation and OE must be active to perform a memory read operation.

•When these two controls WE and OE are present, they must never be active at the same time

•The ROM read only memory permanently stores programs and data and data was always present, even when power is disconnected.

•It is also called as nonvolatile memory.

•EPROM (erasable programmable read only memory) is also erasable if exposed to high intensity ultraviolet light for about 20 minutes or less, depending upon the type of EPROM.

•We have PROM (programmable read only memory)

•RMM (read mostly memory) is also called the flash memory.

•The flash memory is also called as an EEPROM (electrically erasable programmable ROM), EAROM (electrically alterable ROM), or a NOVROM (nonvolatile ROM).

•These memory devices are electrically erasable in the system, but require more time to erase than a normal RAM.

•EPROM contains the series of 27XXX contains the following part numbers : 2704(512 * 8), 2708(1K * 8), 2716(2K * 8), 2732(4K * 8), 2764(8K * 8), 27128(16K * 8) etc..

•Each of these parts contains address pins, eight data connections, one or more chip selection inputs (CE) and an output enable pin (OE).

•This device contains **11** address inputs and **8** data outputs.

•If both the pin connection CE and OE are at logic 0, data will appear on the output connection . If both the pins are not at logic 0, the data output connections remains at their high impedance or off state.

•To read data from the EPROM Vpp pin must be placed at a logic 1.

Interrupt Cycle of 8086/88

An 8086 interrupt can come from any one the three sources :

- External signal
- Special Instruction in the program
- Condition produced by instruction

8.2.1 External Signal (Hardware Interrupt)

An 8086 can get interrupt from an external signal applied to the nonmaskable interrupt (NMI) input pin, or the interrupt (INTR) input pin.

8.2.2 Special Instruction

8086 supports a special instruction, INT to execute special program. At the end of the interrupt service routine, execution is usually returned to the interrupted program.

8.2.3 Condition Produced by Instruction

An 8086 is interrupted by some condition produced in the 8086 by the execution of an instruction. For example divide by zero : Program execution will automatically be interrupted if you attempt to divide an operand by zero.

At the end of each instruction cycle 8086 checks to see if there is any interrupt request. If so, 8086 responds to the interrupt by performing series of actions (Refer Fig. 8.1).

1. It decrements stack pointer by 2 and pushes the flag register on the stack .
2. It disables the INTR interrupt input by clearing the interrupt flag in the flag register.
3. It resets the trap flag in the flag register.
4. It decrements stack pointer by 2 and pushes the current code segment register contents on the stack.
5. It decrements stack pointer by 2 and pushes the current instruction pointer contents on the stack.
6. It does an indirect far jump at the start of the procedure by loading the CS and IP values for the start of the interrupt service routine (ISR).

An IRET instruction at the end of the interrupt service procedure returns execution to the main program.

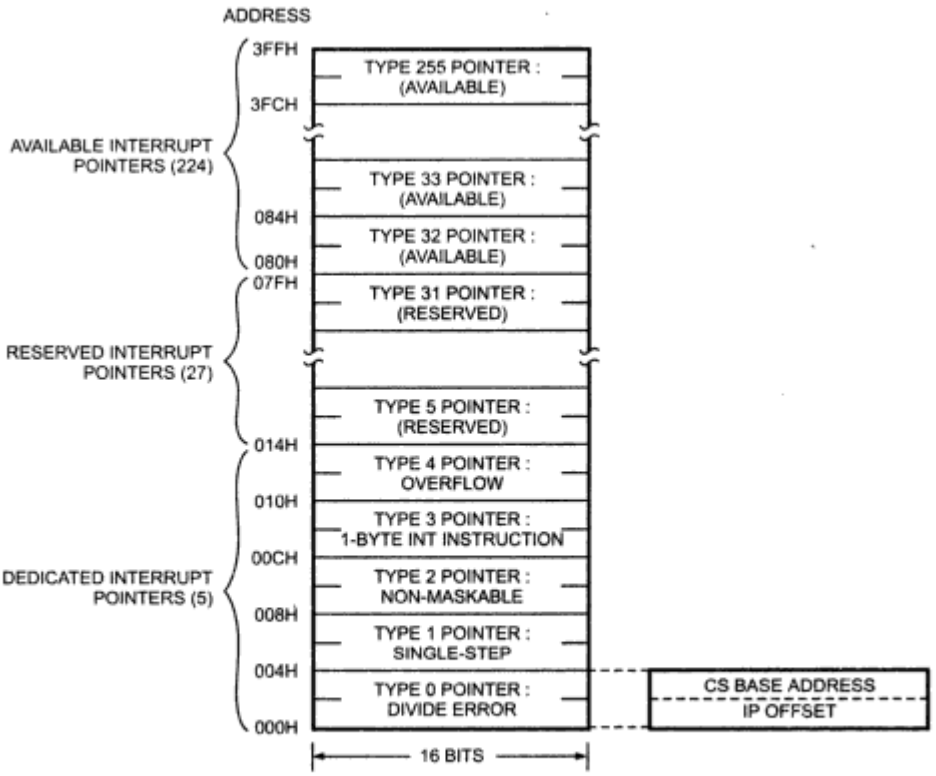


Fig. 8.2 8086 interrupt vector table

8086 Interrupt Types

Divide by Zero Interrupt (Type 0)

When the quotient from either a DIV or IDIV instruction is too large to fit in the result gister; 8086 will automatically execute type 0 interrupt.

Single Step Interrupt (Type 1)

The type 1 interrupt is the single step trap. In the single step mode, system will execute one instruction and wait for further direction from user. Then user can examine the contents of registers and memory locations and if they are correct, user can tell the system to execute the next instruction. This feature is useful for debugging assembly language programs.

Software Interrupts

Type 0 - 255 :

The 8086 INT instruction can be used to cause the 8086 to do one of the 256 possible interrupt types. The interrupt type is specified by the number as a part of the instruction. You can use an INT2 instruction to send execution to an NMI interrupt service routine. This allows you to test the NMI routine without needing to apply an external signal to the NMI input of the 8086.

With the software interrupts you can call the desired routines from many different programs in a system e.g. BIOS in IBM PC. The IBM PC has in its ROM collection of routines, each performing some specific function such as reading character from keyboard, writing character to CRT. This collection of routines referred to as **Basic Input Output System** or **BIOS**.

The BIOS routines are called with INT instructions. We will summarize interrupt response and how it is serviced by going through following steps.

1. 8086 pushes the flag register on the stack.
2. It disables the single step and the INTR input by clearing the trap flag and interrupt flag in the flag register.
3. It saves the current CS and IP register contents by pushing them on the stack.
4. It does an indirect far jump to the start of the routine by loading the new values of CS and IP register from the memory whose address calculated by multiplying 4 to the interrupt type, For example, if interrupt type is 4 then memory address is $4 \times 4 = 10_{10} = 10H$. So 8086 will read new value of IP from 00010H and CS from 00012H.
5. Once these values are loaded in the CS and IP, 8086 will fetch the instruction from the new address which is the starting address of interrupt service routine.
6. An IRET instruction at the end of the interrupt service routine gets the previous values of CS and IP by popping the CS and IP from the stack.
7. At the end the flag register contents are copied back into flag register by popping the flag register form stack.

Features of 8259

1. It can manage eight priority interrupts. This is equivalent to provide eight interrupt pins on the processor in place of INTR pin.
2. It is possible to locate vector table for these additional interrupts any where in the memory map. However, all eight interrupts are spaced at the interval of either four or eight locations.
3. By cascading 8259s it is possible to get 64 priority interrupts.
4. Interrupt mask register makes it possible to mask individual interrupt request.
5. The 8259A can be programmed to accept either the level triggered or the edge triggered interrupt request.
6. With the help of 8259A user can get the information of pending interrupts, in-service interrupts and masked interrupts.
7. The 8259A is designed to minimize the software and real time overhead in handling multi-level priority interrupts.

Block Diagram of 8259A

Fig. 8.6 shows the internal block diagram of the 8259A. It includes eight blocks : data bus buffer, read/write logic, control logic, three registers (IRR, ISR and IMR), priority resolver, and cascade buffer.

Data Bus Buffer

The data bus allows the 8086 to send control words to the 8259A and read a status word from the 8259A and read a status word from the 8259A. The 8-bit data bus also allows the 8259A to send interrupt types to the 8086.

Priority Resolver

The priority resolver determines the priorities of the bits set in the IRR. The bit corresponding to the highest priority interrupt input is set in the ISR during the $\overline{INTA}$ input.

Cascade Buffer Comparator

This section generates control signals necessary for cascade operations. It also generates Buffer-Enable signals. As stated earlier, the 8259 can be cascaded with other 8259s in order to expand the interrupt handling capacity to sixty-four levels. In such a case, the former is called a **master**, and the latter are called **slaves**. The 8259 can be set up as a master or a slave by the $\overline{SP}$ / $\overline{EN}$ pin.

CAS 0 - 2

For a master 8259, the CAS_0-CAS_2 pins are outputs, and for slave 8259s, these are inputs. When the 8259 is a master (that is, when it accepts interrupt requests from other 8259s), the CALL opcode is generated by the Master in response to the first $\overline{INTA}$. The vectoring address must be released by the slave 8259. The master sends an identification code of three-bits (to select one out of the eight possible slave 8259s) on the CAS_0-CAS_2 lines. The slave 8259s accept these three signals as inputs (on their CAS_0-CAS_2 pins) and compare the code sent by the master with the codes assigned to them during initialisation. The slave thus selected (which had originally placed an interrupt request to the master 8259) then puts out the address of the interrupt service routine during the second and third $\overline{INTA}$ pulses from the CPU.

$\overline{SP}$ / $\overline{EN}$ (Slave Program /Enable Buffer)

The $\overline{SP}$ / $\overline{EN}$ signal is tied high for the master. However, it is grounded for the slave.

In large systems where buffers are used to drive the data bus, the data sent by the 8259 in response to $\overline{INTA}$ cannot be accessed by the CPU (due to the data bus buffer being disabled).

If an 8259 is used in the buffered mode (buffered or non-buffered modes of operation can be specified at the time of initialising the 8259), the $\overline{SP}$ / $\overline{EN}$ pin is used as an output which can be used to enable the system data bus buffer whenever the 8259's data bus outputs are enabled (when it is ready to send data).

Means, in non-buffered mode, the $\overline{SP}/\overline{EN}$ pin of an 8259 is used to specify whether the 8259 is to operate as a master or as a slave, and in the buffered mode, the $\overline{SP}/\overline{EN}$ pin is used as an output to enable the data bus buffer of the system.

8.5.3 Interrupt Sequence

The events occur as follows in an 8086 system :

1. One or more of the INTERRUPT REQUEST lines (IR0-IR7) are raised high, setting the corresponding IRR bit(s).

2. The priority resolver checks three registers : The IRR for interrupt requests, the IMR for masking bits, and the ISR for the interrupt request being served. It resolves the priority and sets the INT high when appropriate.
3. The CPU acknowledges the INT and responds with an $\overline{INTA}$ pulse.
4. Upon receiving an $\overline{INTA}$ from the CPU, the highest priority ISR bit is set and the corresponding IRR bit is reset. The 8259A does not drive data bus during this cycle.
5. A selection of priority modes is available to the programmer so that the manner in which the requests are processed by the 8259A can be configured to match his system requirements. The priority modes can be changed or reconfigured dynamically at any time during the main program. This means that the complete interrupt service structure can be defined as required, based on the total system environment.
6. The 8086 will initiate a second INTA pulse. During this pulse, the 8259A releases a 8-bit pointer (interrupt type) onto the Data Bus where it is read by the CPU.
7. This completes the interrupt cycle. In the AEOI mode the ISR bit is reset at the end of the second INTA pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

8.5.4 Priority Modes and Other Features

The various modes of operation of the 8259 are :

(a) Fully Nested Mode,

(b) Rotating Priority Mode,

(c) Special Masked Mode, and

(d) Polled Mode.

a) Fully Nested Mode :

After initialization, the 8259A operates in fully nested mode so it is called as default mode. The 8259 continues to operate in the Fully Nested Mode until the mode is changed through Operation Command Words. In this mode, IR0 has highest priority and IR7 has lowest priority. When the interrupt is acknowledged, it sets the corresponding bit in ISR. This bit will prevent all interrupts of the same or lower level, however it will accept higher priority interrupt requests. The vector address corresponding to this interrupt is then sent. The bit in the ISR will remain set until an EOI command is issued by the microprocessor at the end of interrupt service routine.

But if AEOI (Automatic End of Interrupt) bit is set, the bit in the ISR resets at the trailing edge of the last $\overline{INTA}$.

(i) Automatic Rotation

In this mode, a device, after being serviced, receives the lowest priority. Assuming that IR_3 has just been serviced, it will receive the seventh priority.

IR_0	IR_1	IR_2	IR_3	IR_4	IR_5	IR_6	IR_7
4	5	6	7	0	1	2	3

(ii) Specific Rotation

In the Automatic Rotation mode, the interrupt request last serviced is assigned the lowest priority, whereas in the Specific Rotation mode, the lowest priority can be assigned to any interrupt input (IR_0 to IR_7) thus fixes all other priorities.

For example if the lowest priority is assigned to IR_2, other priorities are as shown below.

IR_0	IR_1	IR_2	IR_3	IR_4	IR_5	IR_6	IR_7
5	6	7	0	1	2	3	4

d) Special Mask Mode :

If any interrupt is in service then the corresponding bit is set in ISR and the lower priority interrupts are inhibited. Some applications may require an interrupt service routine to dynamically alter the system priority structure during its execution under software control, for example, the routine may wish to inhibit lower priority requests for a portion of its execution but enable some of them for another portion. In these cases we have to go for special mask mode.

In the special mask mode it inhibits further interrupts at that level and enables interrupts from all other levels (lower as well as higher) that are not masked. Thus any interrupt may be selectively enabled by loading the mask register.

e) Poll Mode :

In this mode the INT output is not used. The microprocessor checks the status of interrupt requests by issuing poll command. The microprocessor reads contents of 8259A after issuing poll command. During this read operation the 8259A provides polled word and sets ISR bit of highest priority active interrupt request FORMAT.

I	X	X	X	X	W_2	W_1	W_0

$I = 1 \rightarrow$ One or more interrupt requests activated.

$I = 0 \rightarrow$ No interrupt request activated.

$W_2\ W_1\ W_0 \rightarrow$ Binary code of highest priority active interrupt request.

Programming the 8259A

The 8259A requires two types of command words. Initialization Command Words (ICWs) and Operational Command Words (OCWs).

The 8259A can be initialized with four ICWs; the first two are compulsory, and the other two are optional based on the modes being used. These words must be issued in a given sequence. After initialization, the 8259A can be set up to operate in various modes by using three different OCWs; however, they no longer need to be issued in a specific sequence.

Flow chart :

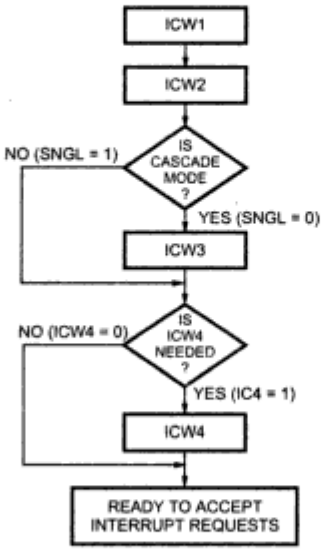


Fig. 8.7 8259 A initialization flowchart

Initialization Command Word 1 (ICW1)

Fig. 8.8 shows the Initialization Command Word 1 (ICW1).

A write command issued to the 8259 with $A_0 = 0$ and $D_4 = 1$ is interpreted as ICW1, which starts the initialization sequence.

It specifies

1. Single or multiple 8259As in the system.
2. 4 or 8 bit interval between the interrupt vector locations.
3. The address bits A_7 - A_5 of the CALL instruction.
4. Edge triggered or level triggered interrupts.
5. ICW4 is needed or not.

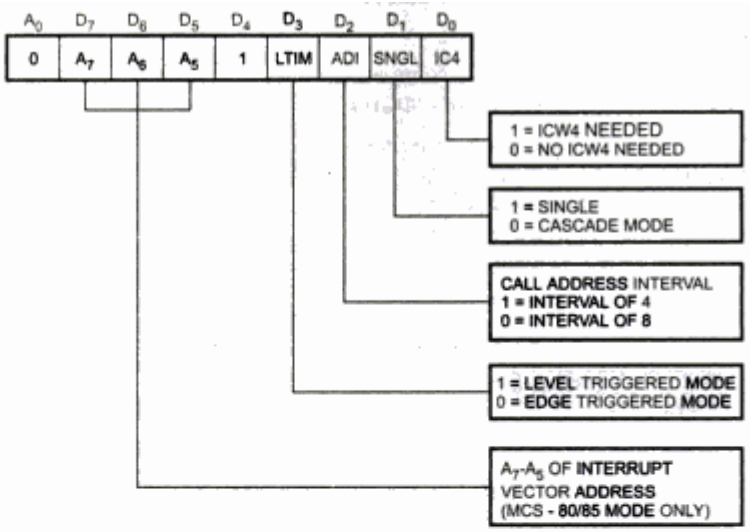


Fig. 8.8 Initialization command word 1 (ICW1)

Initialization Command Word 2 (ICW2)

Fig 8.9 shows the Initialization Command Word 2 (ICW2).

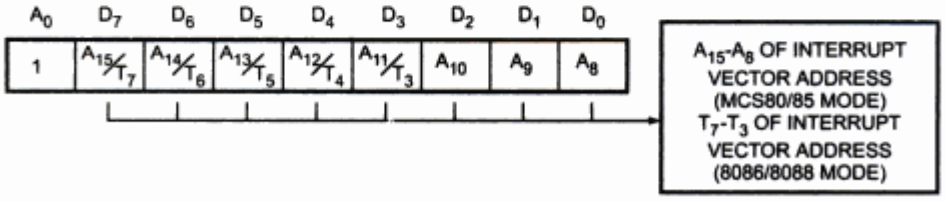


Fig. 8.9 Initialization command word 2 (ICW2)

A write command following ICW1, with A0 = 1 is interpreted as ICW2. This is used to load the high order byte of the interrupt vector address of all the interrupts.

Initialization Command Word 3 (ICW3)

ICW3 is required only if there is more than one 8259 in the system and if they are cascaded. An ICW3 operation loads a slave register in the 8259. The format of the byte to be loaded as an ICW3 for a master 8259 or a slave is shown in the Fig. 8.10. For master, each bit in ICW3 is used to specify whether it has a slave 8259 attached to it on its corresponding IR (Interrupt Request) input. For slave, bits D_0-D_2 of ICW3 are used to assign a slave identification code (slave ID) to the 8259.

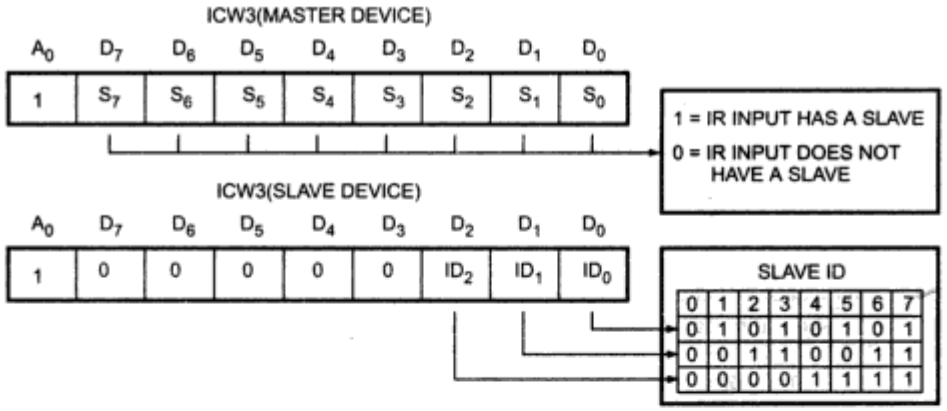


Fig. 8.10 Initialization command word 3 (ICW3)

Initialization Command Word 4 (ICW4)

It is loaded only if the D_0 bit of ICW1 (IC 4) is set. The format of ICW4 is shown in Fig. 8.11.

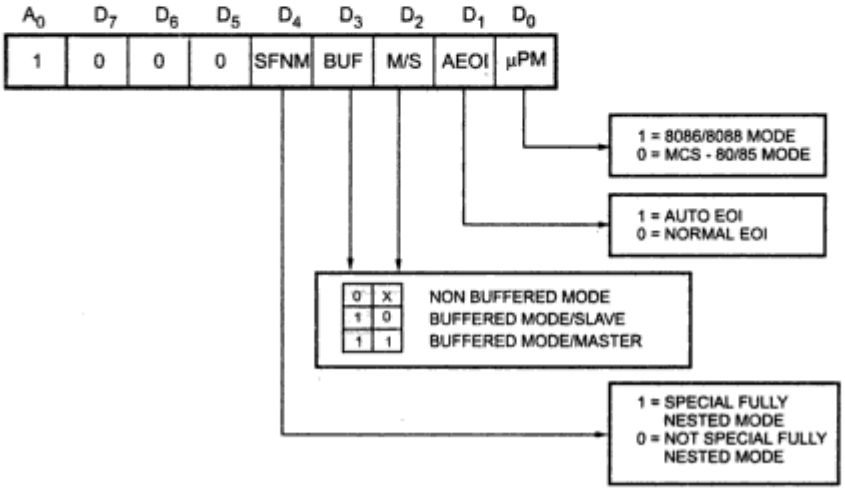


Fig. 8.11 Initialization command word 4 (ICW4)

It specifies.

1) Whether to use special fully nested mode or non special fully nested mode.
2) Whether to use buffered mode or non buffered mode.
3) Whether to use Automatic EOI or Normal EOI
4) CPU used, 8086/8088 or 80810.

UNIT-V

SERIAL COMMUNICATION:

Classification

Serial data transmission can be classified on the basis of how transmission occurs.

1. Simplex
2. Half duplex
3. Full duplex

Simplex

In simplex, the hardware exists such that data transfer takes place only in one direction. There is no possibility of data transfer in the other direction. A typical example is transmission from a computer to the printer.

10.1.2 Half Duplex

The half duplex transmission allows the data transfer in both directions, but no simultaneously. A typical example is a walkie-talkie.

10.1.3 Full Duplex

The full duplex transmission allows the data transfer in both direction simultaneously The typical example is transmission through telephone lines.

10.2 Transmission Formats

The data in the serial communication may be sent in two formats :

a) Asynchronous b) Synchronous

10.2.1 Asynchronous

Fig. 10.1 shows the transmission format for asynchronous transmission. Asynchronous formats are character oriented. In this, the bits of a character or data word are sent at a constant rate, but characters can come at any rate (asynchronously) as long as they do not overlap. When no characters are being sent, a line stays high at logic 1 called **mark**, logic 0 is called **space**. The beginning of a character is indicated by a start bit which is always low. This is used to synchronize the transmitter and receiver. After the start bit, the data bits are sent with least significant bit first, followed by one or more stop bits (active high). The stop bits indicate the end of character. Different systems use 1, 1 1/2 or 2 stop bits. The combination of start bit, character and stop bits is known as **frame.** The start and stop bits carry no information, but are required because of the asynchronous nature of data. Fig. 10.2 illustrates how the data byte CAH would look when transmitted in the asynchronous serial format.

10.1.2 Half Duplex

The half duplex transmission allows the data transfer in both directions, but no simultaneously. A typical example is a walkie-talkie.

10.1.3 Full Duplex

The full duplex transmission allows the data transfer in both direction simultaneously The typical example is transmission through telephone lines.

10.2 Transmission Formats

The data in the serial communication may be sent in two formats :

a) Asynchronous b) Synchronous

10.2.1 Asynchronous

Fig. 10.1 shows the transmission format for asynchronous transmission. Asynchronou formats are character oriented. In this, the bits of a character or data word are sent at constant rate, but characters can come at any rate (asynchronously) as long as they do no overlap. When no characters are being sent, a line stays high at logic 1 called **mark**, logic is called **space**. The beginning of a character is indicated by a start bit which is alway low. This is used to synchronize the transmitter and receiver. After the start bit, the dat bits are sent with least significant bit first, followed by one or more stop bits (active high) The stop bits indicate the end of character. Different systems use 1, 1 1/2 or 2 stop bits The combination of start bit, character and stop bits is known as **frame.** The start and stop bits carry no information, but are required because of the asynchronous nature of data Fig. 10.2 illustrates how the data byte CAH would look when transmitted in the asynchronous serial format.

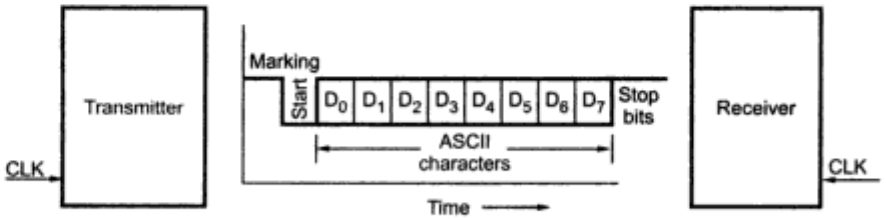


Fig. 10.1 Transmission format for asynchronous transmission

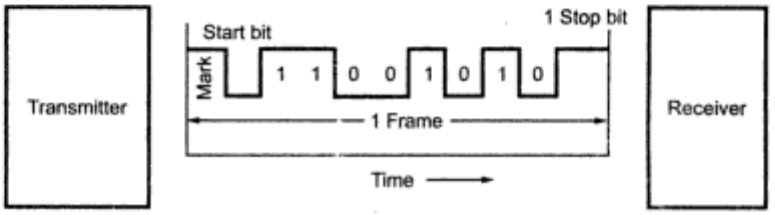


Fig. 10.2 Asynchronous format with data byte CAH

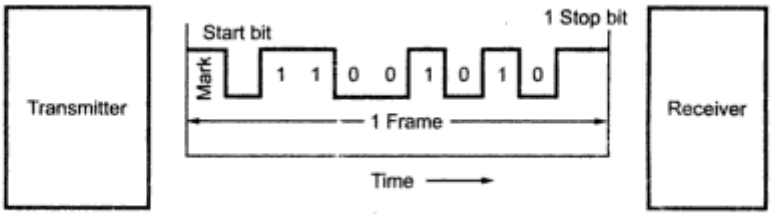


Fig. 10.2 Asynchronous format with data byte CAH

The data rate can be expressed as bits/sec. or characters/sec. The term bits/sec is also called the **baud rate**. The asynchronous format is generally used in low-speed transmission (less than 20 Kbits/sec).

10.2.2 Synchronous

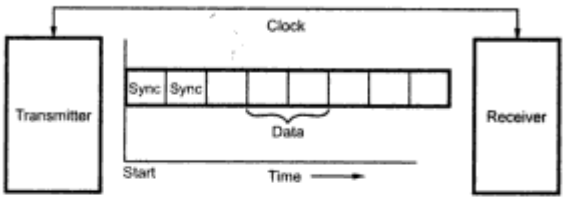


Fig. 10.3 Synchronous transmission format

The start and stop bits in each frame of asynchronous format represents wasted overhead bytes that reduce the overall character rate. These start and stop bits can be eliminated by synchronizing receiver and transmitter. They can be synchronized by having a common clock signal. Such a communication is called **synchronous serial communication**. The Fig. 10.3 shows the transmission format of synchronous serial communication. In this transmission synchronous bits are inserted instead of start and stop bits.

Sr. No.	Asynchronous Serial Communication	Synchronous Serial Communication
1.	Transmitters and receivers are not synchronized by clock.	Transmitter and receivers are synchronized by clock.
2.	Bits of data are transmitted at constant rate.	Data bits are transmitted with synchronisation of clock.
3.	Character may arrive at any rate at receiver.	Character is received at constant rate.
4.	Data transfer is character oriented.	Data transfer takes place in blocks.
5.	Start and stop bits are required to establish communication of each character.	Start and stop bits are not required to establish communication of each character; however, synchronisation bits are required to transfer the data block.
6.	Used in low-speed transmissions at about speed less than 20 kbits/sec.	Used in high-speed transmissions

Table 10.1 Comparison between asynchronous and synchronous serial data transfer

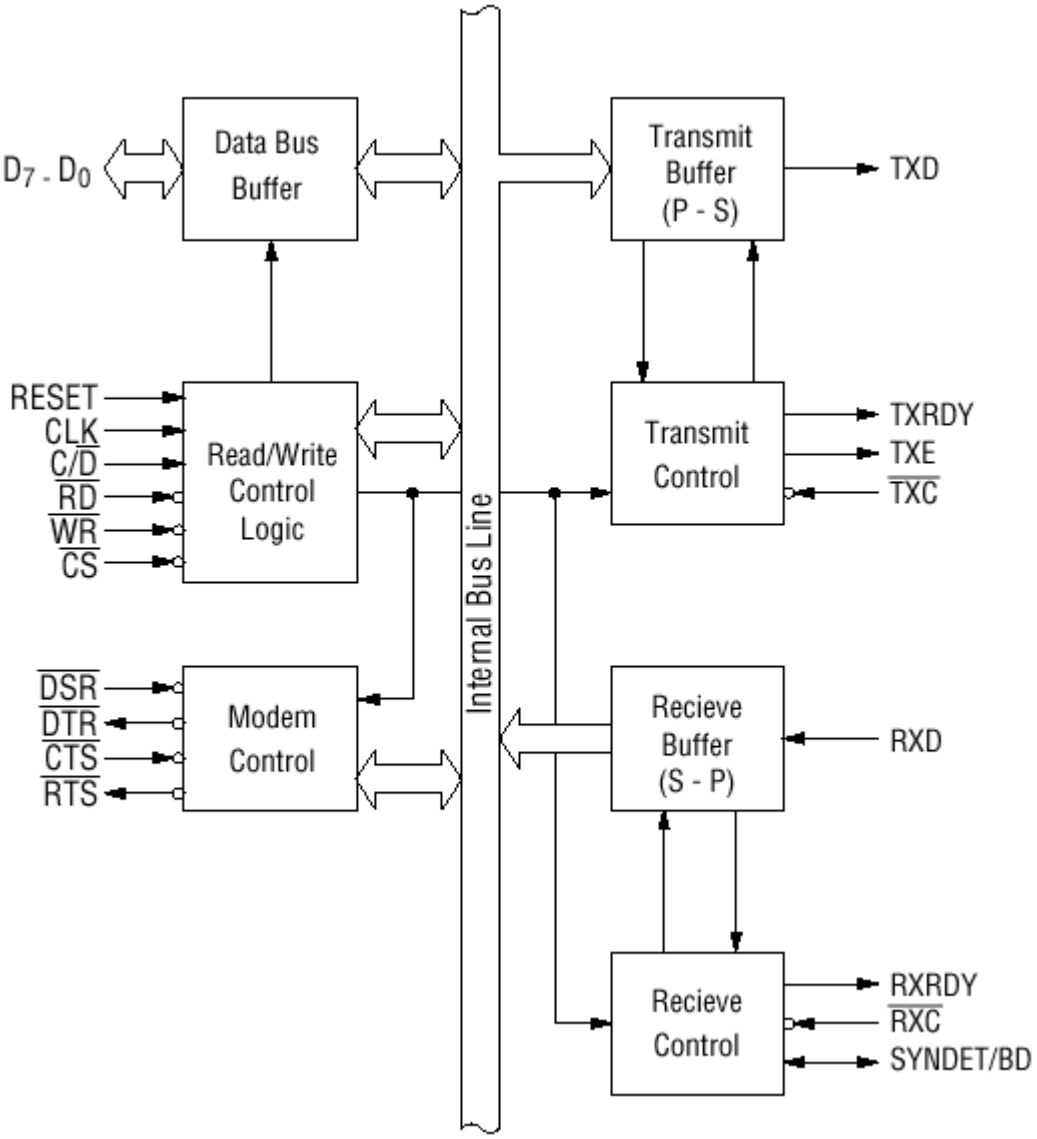
Command Instruction

After the mode instruction, command character should be issued to the USART. It controls the operation of the USART within the basic frame work established by the mode instruction. Fig. 10.7 shows command instruction format.

It does function such as : Enable Transmit/Receive, Error Reset and modem control.

Universal Synchronous Asynchronous Receiver Transmitter (USART)

The 8251 is a USART (Universal Synchronous Asynchronous Receiver Transmitter) for serial data communication. As a peripheral device of a microcomputer system, the 8251 receives parallel data from the CPU and transmits serial data after conversion. This device also receives serial data from the outside and transmits parallel data to the CPU after conversion.



Block diagram of the 8251 USART

The 8251 functional configuration is programed by software. Operation between the 8251 and a CPU is executed by program control. Table 1 shows the operation between a CPU and the device.

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	
1	×	×	×	Data Bus 3-State
0	×	1	1	Data Bus 3-State
0	1	0	1	Status → CPU
0	1	1	0	Control Word ← CPU
0	0	0	1	Data → CPU
0	0	1	0	Data ← CPU

Table 1 Operation between a CPU and 8251

Control Words

There are two types of control word.

1. Mode instruction (setting of function)

2. Command (setting of operation)

1) Mode Instruction

Mode instruction is used for setting the function of the 8251. Mode instruction will be in "wait for write" at either internal reset or external reset. That is, the writing of a control word after resetting will be recognized as a "mode instruction."

Items set by mode instruction are as follows:

• Synchronous/asynchronous mode

• Stop bit length (asynchronous mode)

• Character length

• Parity bit

• Baud rate factor (asynchronous mode)

• Internal/external synchronization (synchronous mode)

• Number of synchronous characters (Synchronous mode)

The bit configuration of mode instruction is shown in Figures 2 and 3. In the case of synchronous mode, it is necessary to write one-or two byte sync characters. If sync characters were written, a function will be set because the writing of sync characters constitutes part of mode instruction.

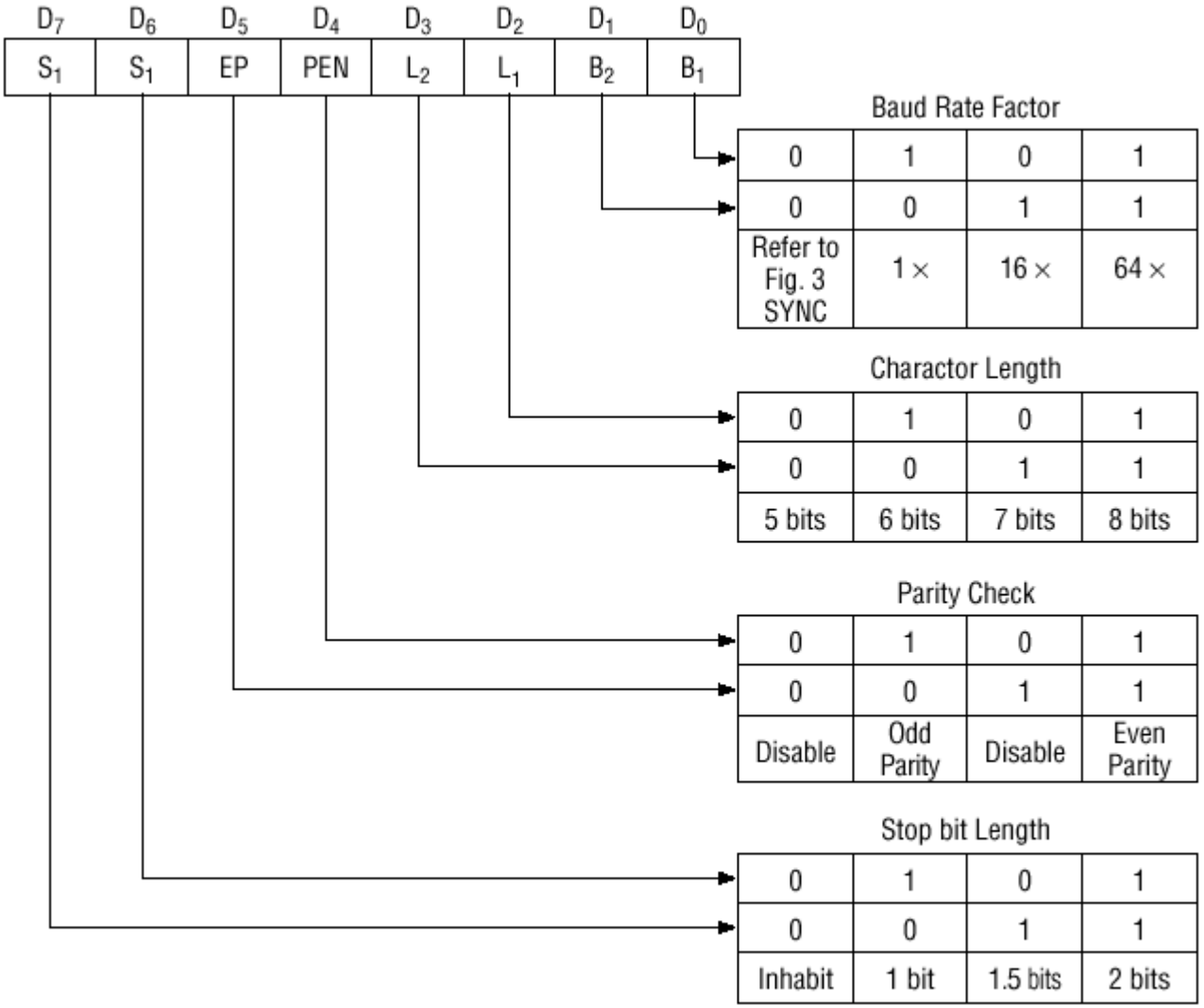


Fig. 2 Bit Configuration of Mode Instruction (Asynchronous)

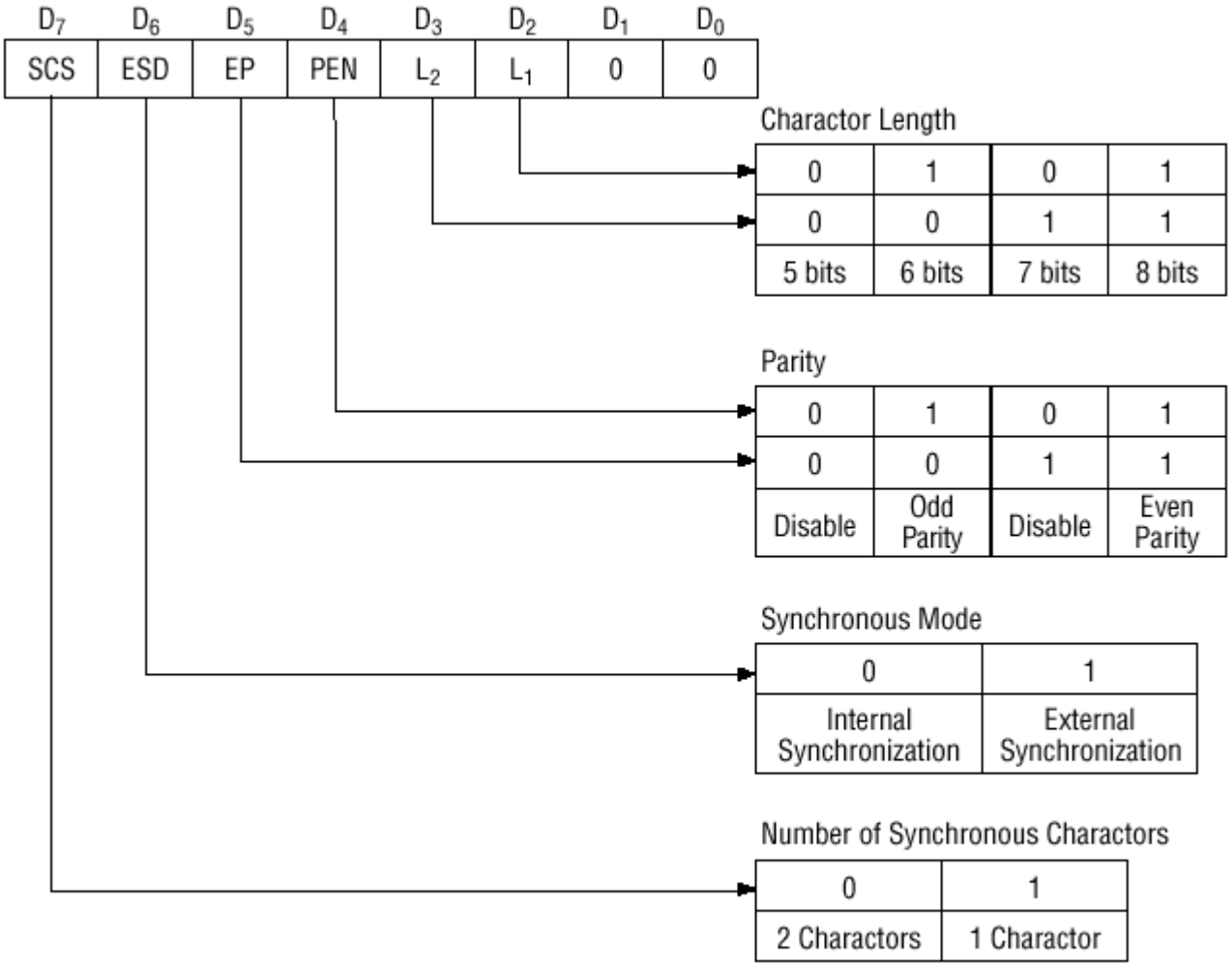


Fig. 3 Bit Configuration of Mode Instruction (Synchronous)

2) Command

Command is used for setting the operation of the 8251. It is possible to write a command whenever necessary after writing a mode instruction and sync characters.

Items to be set by command are as follows:

• Transmit Enable/Disable

• Receive Enable/Disable

• DTR, RTS Output of data.

• Resetting of error flag.

• Sending to break characters

• Internal resetting

• Hunt mode (synchronous mode)

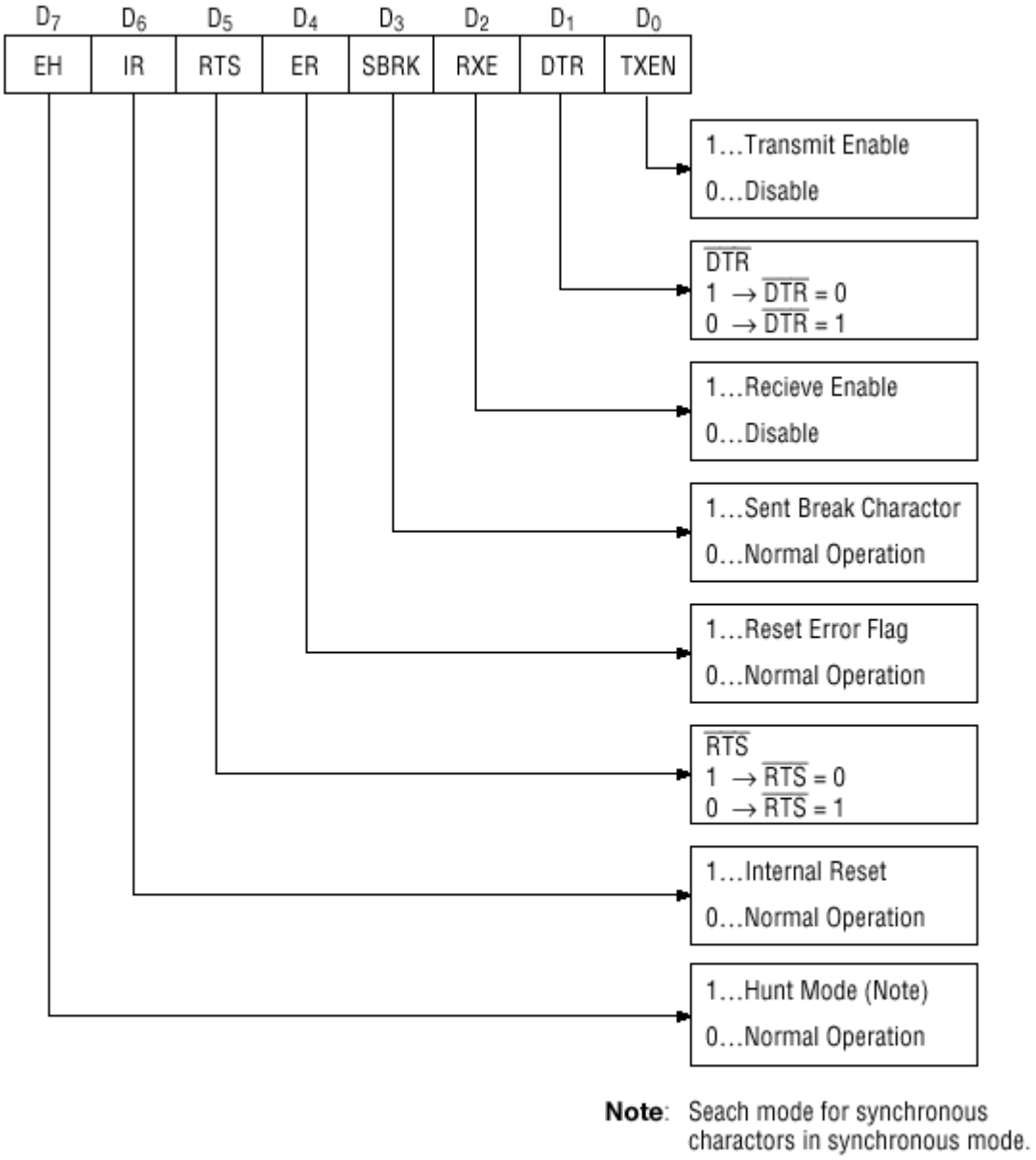


Fig. 4 Bit Configuration of Command

Status Word

It is possible to see the internal status of the 8251 by reading a status word. The bit configuration of status word is shown in Fig. 5.

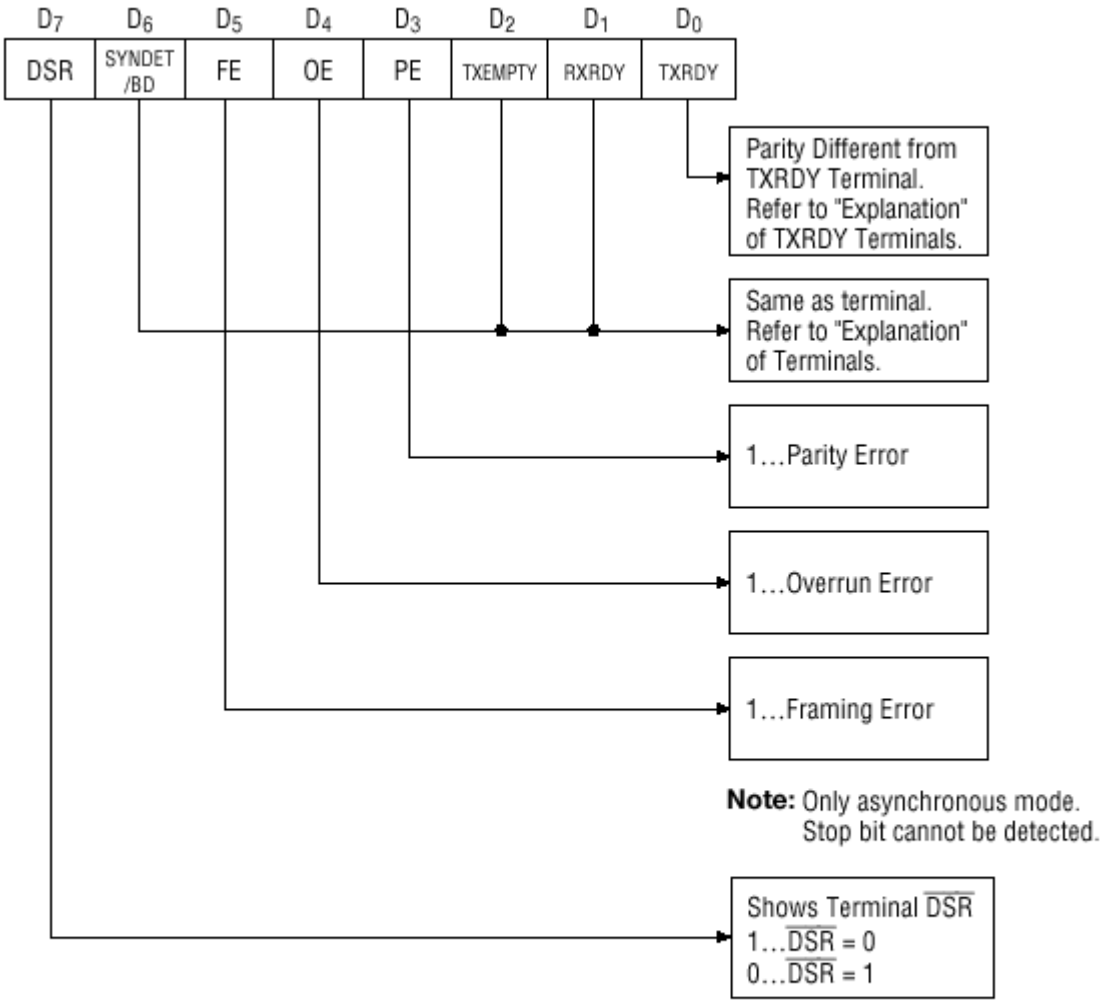


Fig. 5 Bit Configuration of Status Word

Pin Description

D 0 to D 7 (l/O terminal)

This is bidirectional data bus which receive control words and transmits data from the CPU and sends status words and received data to CPU.

RESET (Input terminal)

A "High" on this input forces the 8251 into "reset status." The device waits for the writing of "mode instruction." The min. reset width is six clock inputs during the operating status of CLK.

CLK (Input terminal)

CLK signal is used to generate internal device timing. CLK signal is independent of RXC or TXC. However, the frequency of CLK must be greater than 30 times the RXC and TXC at Synchronous mode and Asynchronous "x1" mode, and must be greater than 5 times at Asynchronous "x16" and "x64" mode.

WR (Input terminal)

This is the "active low" input terminal which receives a signal for writing transmit data and control words from the CPU into the 8251.

RD (Input terminal)

This is the "active low" input terminal which receives a signal for reading receive data and status words from the 8251.

C/D (Input terminal)

This is an input terminal which receives a signal for selecting data or command words and status words when the 8251 is accessed by the CPU. If C/D = low, data will be accessed. If C/D = high, command word or status word will be accessed.

CS (Input terminal)

This is the "active low" input terminal which selects the 8251 at low level when the CPU accesses. Note: The device won't be in "standby status"; only setting CS = High.

TXD (output terminal)

This is an output terminal for transmitting data from which serial-converted data is sent out. The device is in "mark status" (high level) after resetting or during a status when transmit is disabled. It is also possible to set the device in "break status" (low level) by a command.

TXRDY (output terminal)

This is an output terminal which indicates that the 8251is ready to accept a transmitted data character. But the terminal is always at low level if CTS = high or the device was set in "TX disable status" by a command. Note: TXRDY status word indicates that transmit data character is receivable, regardless of CTS or command. If the CPU writes a data character, TXRDY will be reset by the leading edge or WR signal.

TXEMPTY (Output terminal)

This is an output terminal which indicates that the 8251 has transmitted all the characters and had no data character. In "synchronous mode," the terminal is at high level, if transmit data characters are no longer remaining and sync characters are automatically transmitted. If the CPU writes a data character, TXEMPTY will be reset by the leading edge of WR signal. Note : As the transmitter is disabled by setting CTS "High" or command, data written before disable will be sent out. Then TXD and TXEMPTY will be "High". Even if a data is written after disable, that data is not sent out and TXE will be "High".After the transmitter is enabled, it sent out. (Refer to Timing Chart of Transmitter Control and Flag Timing)

TXC (Input terminal)

This is a clock input signal which determines the transfer speed of transmitted data. In "synchronous mode," the baud rate will be the same as the frequency of TXC. In "asynchronous mode", it is possible to select the baud rate factor by mode instruction. It can be 1, 1/16 or 1/64 the TXC. The falling edge of TXC sifts the serial data out of the 8251.

RXD (input terminal)

This is a terminal which receives serial data.

RXRDY (Output terminal)

This is a terminal which indicates that the 8251 contains a character that is ready to READ. If the CPU reads a data character, RXRDY will be reset by the leading edge of RD signal. Unless the CPU reads a data character before the next one is received completely, the preceding data will be lost. In such a case, an overrun error flag status word will be set.

RXC (Input terminal)

This is a clock input signal which determines the transfer speed of received data. In "synchronous mode," the baud rate is the same as the frequency of RXC. In "asynchronous mode," it is possible to select the baud rate factor by mode instruction. It can be 1, 1/16, 1/64 the RXC.

SYNDET/BD (Input or output terminal)

This is a terminal whose function changes according to mode. In "internal synchronous mode." this terminal is at high level, if sync characters are received and synchronized. If a status word is read, the terminal will be reset. In "external synchronous mode, "this is an input terminal. A "High" on this input forces the 8251 to start receiving data characters.

In "asynchronous mode," this is an output terminal which generates "high level"output upon the detection of a "break" character if receiver data contains a "low-level" space between the stop bits of two continuous characters. The terminal will be reset, if RXD is at high level. After Reset is active, the terminal will be output at low level.

DSR (Input terminal)

This is an input port for MODEM interface. The input status of the terminal can be recognized by the CPU reading status words.

DTR (Output terminal)

This is an output port for MODEM interface. It is possible to set the status of DTR by a command.

CTS (Input terminal)

This is an input terminal for MODEM interface which is used for controlling a transmit circuit. The terminal controls data transmission if the device is set in "TX Enable" status by a command. Data is transmitable if the terminal is at low level.

RTS (Output terminal)

This is an output port for MODEM interface. It is possible to set the status RTS by a command.

Data Communication Types

We know that, 8251A is Universal Synchronous, Asynchronous, Receiver, and Transmitter. Therefore communication can take place with four different ways.

1. Asynchronous transmission
2. Asynchronous reception
3. Synchronous transmission
4. Synchronous reception

These communication modes can be enabled by writing proper mode and command instructions. The mode instruction defines the baud rate (in case of asynchronous mode), character length, number of stop bit(s) and parity type. After writing proper mode instruction it is necessary to write appropriate command instruction depending on the communication type.

Asynchronous Transmission

Transmission can be enabled by setting transmission enable bit (bit 0) in the command instruction. When transmitter is enabled and $\overline{CTS}$ = 0 the transmitter is ready to transfer data on TxD line.

Operation : When transmitter is ready to transfer data on TxD line, CPU sends data character and it is loaded in the transmit buffer register. The 8251A then automatically adds a start bit (low level) followed by the data bits (least significant bit first), and the programmed number of STOP bit(s) to each character. It also adds parity information prior to STOP bit(s), as defined by the mode instruction. The character is then transmitted as a serial data stream on the TxD output at the falling edge of $\overline{TxC}$. The rate of transmission is equal to 1, $1/16$ or $1/64$ that of the $\overline{TxC}$, as defined by the mode instruction. Fig. 10.9 shows the transmitter output in the asynchronous mode.

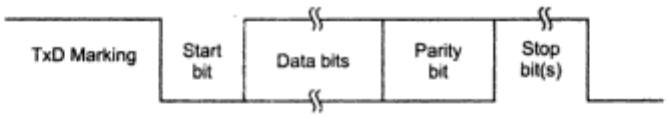


Fig. 10.9 Transmitter output in asynchonous mode

Asynchronous Reception

Reception can be enabled by setting receive enable bit (bit 2) in the command instruction.

Operation :

The RxD line is normally high. 8251A looks for a low level on the RxD line. When it receives the low level, it assumes that it is a START bit and enables an internal counter. At a count equivalent to one-half of a bit time, the RxD line is sampled again. If the line is still low, a valid START bit is detected and the 8251A proceeds to assemble the character. After succesful reception of a START bif the 8251A receives data, parity, and STOP bits and then transfers the data on the receiver input register. The data is then transferred into the receiver buffer register. Fig. 10.10 shows the receiver input in the asynchronous mode.

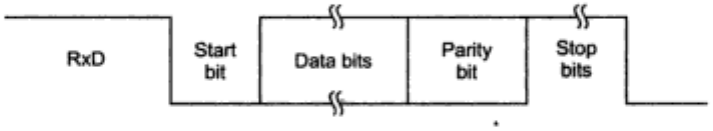


Fig. 10.10 Receiver input in asynchronous mode

Synchronous Transmission

Transmission can be enabled by setting transmission enable bit (bit 0) in the command instruction. When transmitter is enabled and $\overline{CTS}$ = 0, the transmitter is ready to transfer data on TxD line.

Operation : When transmitter is ready to transfer data on TxD line, 8251A transfers characters serially out on the TxD line at the falling edge of the $\overline{TxC}$. The first character usually is the SYNC character.

Once transmission has started, the data stream at the TxD output must continue at the $\overline{TxC}$ rate. If CPU does not provide 8251A with a data character before transmitter buffers become empty, the SYNC characters will be automatically inserted in the TxD data stream, as shown in the Fig. 10.11. In this case, the TxEMPTY pin is raised high to indicate CPU that transmitter buffers are empty. The TxEMPTY pin is internally reset when CPU writes data character in the transmitter buffer.

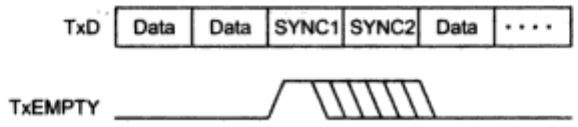


Fig. 10.11 Insertion of SYNC characters

Synchronous Reception : Reception can be enabled by setting receive enable bit (bit 2) in the command instruction.

Operation : In this mode character synchronization can be achieved internally or externally.

Internal SYNC To detect the SYNC character 8251A should be programmed in the 'Enter HUNT' mode by setting bit 7 in the command insturction. Once 8251A enters in the 'Enter HUNT' mode it starts sampling data on the RxD pin on the rising edge of the $\overline{RxC}$. The content of the receiver buffer is compared at every bit boundary with the first SYNC character until a match occurs. If the 8251A has been programmed for two SYNC characters, the subsequent SYNC characters are compared until the match occurs. Once 8251A detects SYNC character(s) it enters from 'HUNT' mode to character synchronization mode, and starts receiving the data characters on the rising edge of the next $\overline{RxC}$. To indicate that the synchronization is achieved 8251A sets the SYNDET pin high. It is reset automatically when CPU reads the status register.

External SYNC

In the external SYNC mode, synchronization is achieved by applying a high level on the SYNDET pin, thus forcing the 8251A out of the HUNT mode.

Serial Communication Protocol (RS232C)

In response to the need for signals and handshake standards between DTE and DCE, the Electronic Industries Association (EIA) introduced EIA standard RS-232 in 1962. It was revised and named as RS-232C, in 1969 by EIA. It is widely accepted for single ended data transmission over short distances with low data rates.

This standard describes the functions of 25 signal and handshake pins for serial data transfer. It also describes the voltage levels, impedance levels, rise and fall times, maximum bit rate, and maximum capacitance for these signal lines. RS-232C specifies 25 signal pins and it specifies that the DTE connector should be male, and the DCE connector should be a female. The most commonly used connector should be a female. The most commonly used connector, DB-25P is shown in the Fig. 10.16.

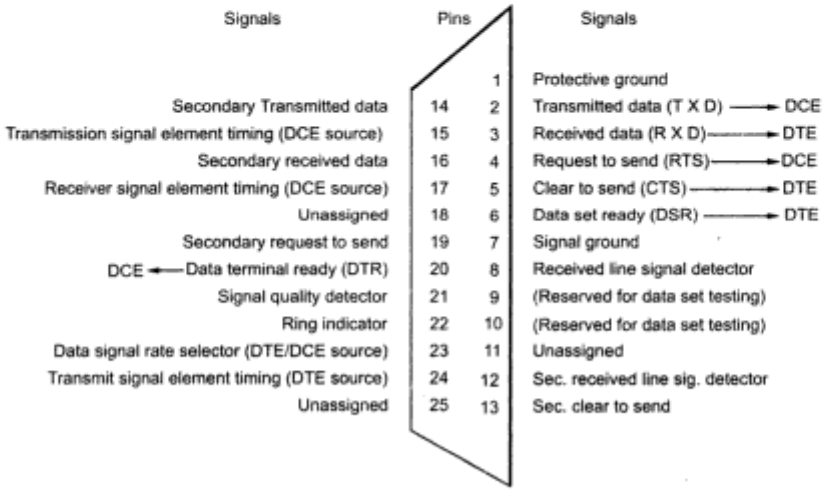


Fig. 10.16 RS 232C 25 pin connector

The Table 10.2 shows pins and signals description for RS-232C for data lines, The voltage level + 3V to + 15V is defined as logic 0; from –3 V to –15 V is defined as logic 1. The control and timing signals are compatible with the TTL level. Because of the incompatibility of the data lines with the TTL logic, voltage translators, called line drivers and line receivers, are required to interface TTL logic with the RS-232 signals. Fig. 10.17 shows the interfacing between TTL and RS-232 signals. The line driver, MC1488, converts logic 1 into approximately 9 V. These levels at the receiving end are again converted by the line receiver, MC1489, into TTL-compatible logic.

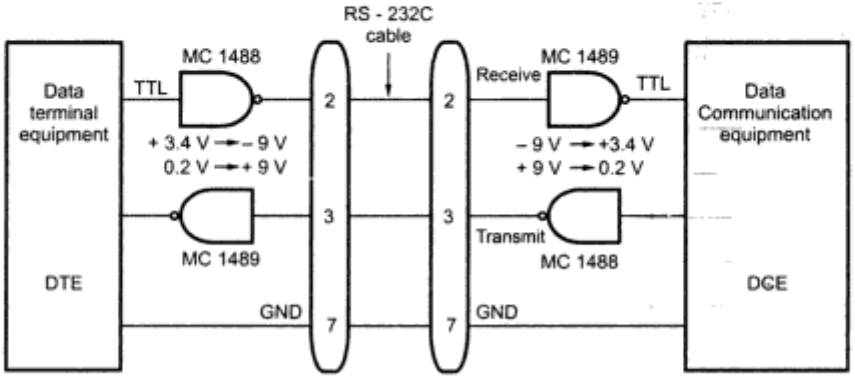


Fig. 10.17 Line drivers and receivers

UNIT-VI

8051 MICROCONTROLLER:

The Intel 8051 microcontroller is one of the most popular general purpose microcontrollers in use today. The success of the Intel 8051 spawned a number of clones which are collectively referred to as the MCS-51 family of microcontrollers, which includes chips from vendors such as Atmel, Philips, Infineon, and Texas Instruments

The Intel 8051 is an 8-bit microcontroller which means that most available operations are limited to 8 bits. There are 3 basic "sizes" of the 8051: Short, Standard, and Extended. The Short and Standard chips are often available in DIP (dual in-line package) form, but the Extended 8051 models often have a different form factor, and are not "drop-in compatible". All these things are called 8051 because they can all be programmed using 8051 assembly language, and they all share certain features (although the different models all have their own special features).

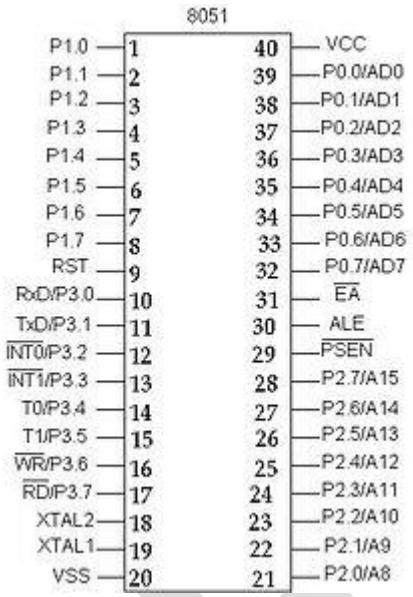
Some of the features that have made the 8051 popular are:

- 64 KB on chip program memory.
- 128 bytes on chip data memory(RAM).
- 4 reg banks.
- 128 user defined software flags.
- 8-bit data bus
- 16-bit address bus
- 32 general purpose registers each of 8 bits
- 16 bit timers (usually 2, but may have more, or less).
- 3 internal and 2 external interrupts.
- Bit as well as byte addressable RAM area of 16 bytes.
- Four 8-bit ports, (short models have two 8-bit ports).
- 16-bit program counter and data pointer.
- 1 Microsecond instruction cycle with 12 MHz Crystal.

8051 models may also have a number of special, model-specific features, such as UARTs, ADC, OpAmps, etc...

Typical applications

8051 chips are used in a wide variety of control systems, telecom applications, robotics as well as in the automotive industry. By some estimations, 8051 family chips make up over 50% of the embedded chip market.



Pin diagram of the 8051 DIP

Basic Pins

PIN 9: PIN 9 is the reset pin which is used reset the microcontroller's internal registers and ports upon starting up. (Pin should be held high for 2 machine cycles.)

PINS 18 & 19: The 8051 has a built-in oscillator amplifier hence we need to only connect a crystal at these pins to provide clock pulses to the circuit.

PIN 40 and 20: Pins 40 and 20 are VCC and ground respectively. The 8051 chip needs +5V 500mA to function properly, although there are lower powered versions like the Atmel 2051 which is a scaled down version of the 8051 which runs on +3V.

PINS 29, 30 & 31: As described in the features of the 8051, this chip contains a built-in flash memory. In order to program this we need to supply a voltage of +12V at pin 31. If external memory is connected then PIN 31, also called EA/VPP, should be connected to ground to indicate the presence of external memory. PIN 30 is called ALE (address latch enable), which is used when multiple memory chips are connected to the controller and only one of them needs to be selected.We will deal with this in depth in the later chapters. PIN 29 is called PSEN. This is "program store enable". In order to use the external memory it is required to provide the low voltage (0) on both PSEN and EA pins.

Ports

There are 4 8-bit ports: P0, P1, P2 and P3.

PORT P1 (Pins 1 to 8): The port P1 is a general purpose input/output port which can be used for a variety of interfacing tasks. The other ports P0, P2 and P3 have dual roles or additional functions associated with them based upon the context of their usage.

PORT P3 (Pins 10 to 17): PORT P3 acts as a normal IO port, but Port P3 has additional functions such as, serial transmit and receive pins, 2 external interrupt pins, 2 external counter inputs, read and write pins for memory access.

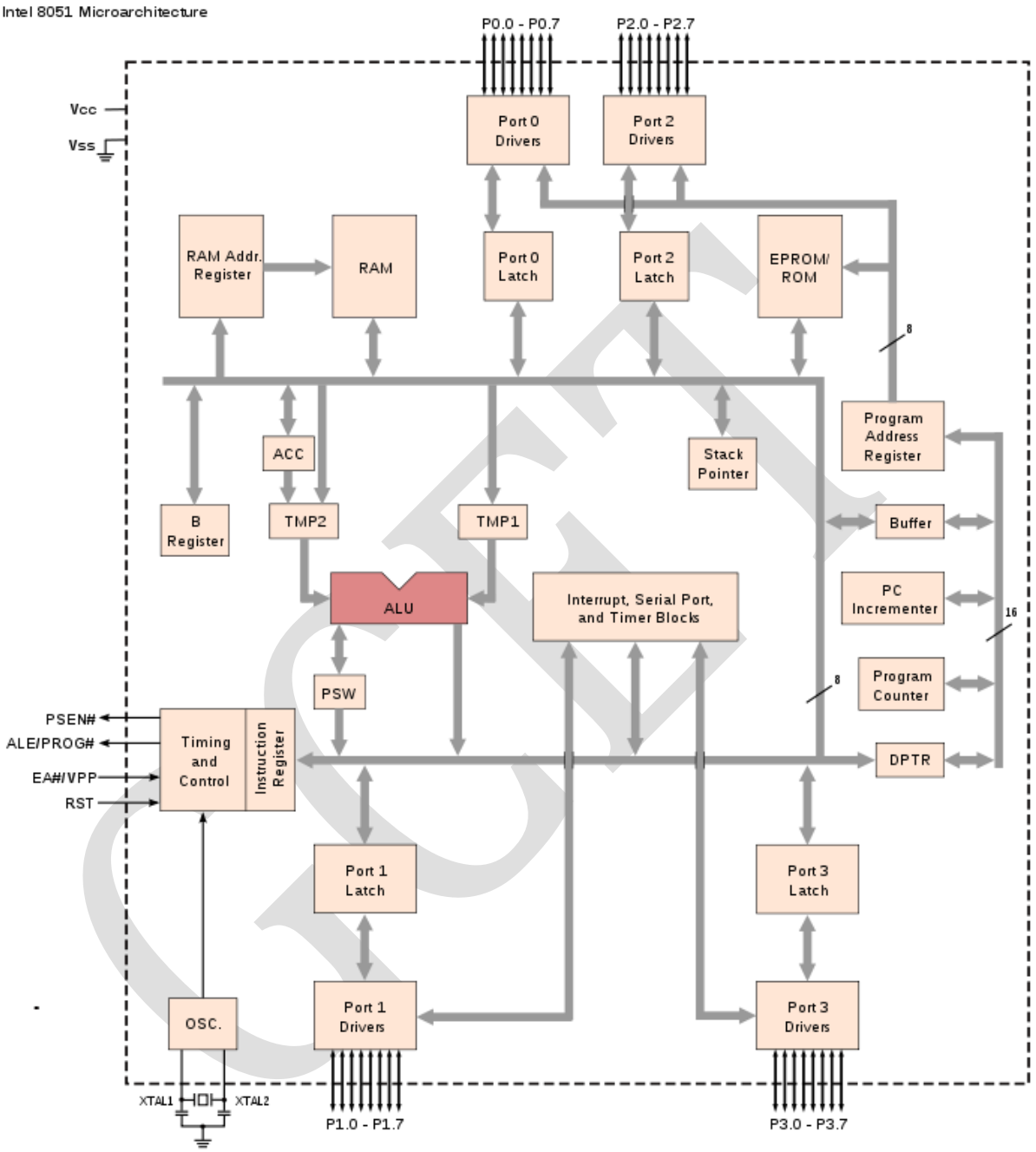
PORT P2 (pins 21 to 28): PORT P2 can also be used as a general purpose 8 bit port when no external memory is present, but if external memory access is required then PORT P2 will act as an address bus in conjunction with PORT P0 to access external memory. PORT P2 acts as A8-A15, as can be seen from fig 1.1

PORT P0 (pins 32 to 39) PORT P0 can be used as a general purpose 8 bit port when no external memory is present, but if external memory access is required then PORT P0 acts as a multiplexed address and data bus that can be used to access external memory in conjunction with PORT P2. P0 acts as AD0-AD7, as can be seen from fig 1.1

Oscillator Circuits

The 8051 requires the existence of an external oscillator circuit. The oscillator circuit usually runs around 12MHz, although the 8051 (depending on which specific model) is capable of running at a maximum of 40MHz. Each machine cycle in the 8051 is 12 clock cycles, giving an effective cycle rate at 1MHz (for a 12MHz clock) to 3.33MHz (for the maximum 40MHz clock).

Internal Architecture



Data and Program Memory

The 8051 Microprocessor can be programmed in PL/M, 8051 Assembly, C and a number of other high-level languages. Many compilers even have support for compiling C++ for an 8051.

Program memory in the 8051 is read-only, while the data memory is considered to be read/write accessible. When stored on EEPROM or Flash, the program memory can be rewritten when the microcontroller is in the special programmer circuit.

Program Start Address

The 8051 starts executing program instructions from address 0000 in the program memory.

Direct Memory

The 8051 has 256 bytes of internal addressable RAM, although only the first 128 bytes are available for general use by the programmer. The first 128 bytes of RAM (from 0x00 to 0x7F) are called the **Direct Memory**, and can be used to store data.

Special Function Register

The **Special Function Register** (SFR) is the upper area of addressable memory, from address 0x80 to 0xFF. A, B, PSW, DPTR are called SFR.This area of memory cannot be used for data or program storage, but is instead a series of memory-mapped ports and registers. All port input and output can therefore be performed by memory **mov** operations on specified addresses in the SFR. Also, different status registers are mapped into the SFR, for use in checking the status of the 8051, and changing some operational parameters of the 8051.

General Purpose Registers

The 8051 has 4 selectable banks of 8 addressable 8-bit registers, R0 to R7. This means that there are essentially 32 available general purpose registers, although only 8 (one bank) can be directly accessed at a time. To access the other banks, we need to change the current bank number in the flag status register.

A and B Registers

The A register is located in the SFR memory location 0xE0. The A register works in a similar fashion to the AX register of x86 processors. The A register is called the **accumulator**, and by default it receives the result of all arithmetic operations. The B register is used in a similar manner, except that it can receive the extended answers from the multiply and divide operations. When not being used for multiplication and Division, the B register is available as an extra general-purpose register.

Comparison between Microprocessor and Microcontroller

We have discussed what is a microprocessor and a microcontroller. Let us see the points of differences between them.

No.	Microprocessor	Microcontroller
1.	Microprocessor contains ALU, control unit (clock and timing circuit), different register and interrupt circuit.	Microcontroller contains microprocessor, memory (ROM and RAM), I/O interfacing circuit and peripheral devices such as A/D converter, serial I/O, timer etc.
2.	It has many instructions to move data between memory and CPU.	It has one or two instructions to move data between memory and CPU.
3.	It has one or two bit handling instructions.	It has many bit handling instructions.
4.	Access times for memory and I/O devices are more.	Less access times for built-in memory and I/O devices.
5.	Microprocessor based system requires more hardware.	Microcontroller based system requires less hardware reducing PCB size and increasing the reliability.
6.	Microprocessor based system is more flexible in design point of view.	Less flexible in design point of view.
7.	It has single memory map for data and code.	It has separate memory map for data and code.
8.	Less number of pins are multifunctioned.	More number pins are multifunctioned.

Features of 8051

The features of the 8051 family are as follows :

1) 4096 bytes on - chip program memory.
2) 128 bytes on - chip data memory.
3) Four register banks.
4) 128 User-defined software flags.
5) 64 Kilobytes each program and external RAM addressability.
6) One microsecond instruction cycle with 12 MHz crystal.
7) 32 bidirectional I/O lines organized as four 8-bit ports (16 lines on 8031).
8) Multiple mode, high-speed programmable serial port.
9) Two multiple mode, 16-bit Timers/Counters.
10) Two-level prioritized interrupt structure.
11) Full depth stack for subroutine return linkage and data storage.
12) Direct Byte and Bit addressability.
13) Binary or Decimal arithmetic.
14) Signed-overflow detection and parity computation.
15) Hardware Multiple and Divide in 4 μsec.
16) Integrated Boolean Processor for control applications.
17) Upwardly compatible with existing 8084 software.

8051 Microcontroller Hardware

The Fig. 11.1 shows the internal block diagram of 8051. It consists of a CPU, two kinds of memory sections (data memory - RAM and program memory - EPROM/ROM), input/output ports, special function registers and control logic needed for a variety of peripheral functions. These elements communicate through an eight bit data bus which runs throughout the chip referred as internal data bus. This bus is buffered to the outside world through an I/O port when memory or I/O expansion is desired.

Central Processing Unit (CPU)

The CPU of 8051 consists of eight-bit Arithmetic and Logic unit with associated registers like A, B, PSW, SP, the sixteen bit program counter and "Data pointer" (DPTR) registers. Alongwith these registers it has a set of special function registers. Along with these registers it has a set of special function registers.

The 8051's ALU can perform arithmetic and logic functions on eight bit variables. The arithmetic unit can perform addition, subtraction, multiplication and division. The logic unit can perform logical operations such as AND, OR, and Exclusive-OR, as well as rotate, clear, and complement. The ALU also looks after the branching decisions. An important and unique feature of the 8051 architecture is that the ALU can also manipulate one bit as well as eight-bit data types. Individual bits may be set, cleared, complemented, moved, tested, and used in logic computation.

Internal RAM

The 8051 has 128-byte internal RAM. It is accessed using RAM address register. The Fig. 11.3 shows the organisation of internal RAM. As shown in the Fig. 11.3, internal RAM of 8051 is organised into three distinct areas :

- Working registers
- Bit Addressable
- General Purpose

1. First thirty-two bytes from address 00H to 1FH of internal RAM constitute 32 working registers. They are organised into four banks of eight registers each. The four register banks are numbered 0 to 3 and are consists of eight registers named R_0 to R_7. Each register can be addressed by name or by its RAM address. Only one register bank is in use at a time. Bits RS_0 and RS_1 in the PSW determine which bank of registers is currently in use. Register banks when not selected can be used as general purpose RAM. On reset, the Bank 0 is selected.
2. The 8051 provides 16 bytes of a bit-addressable area. It occupies RAM byte addresses from 20H to 2FH, forming a total of 128 (16 × 8) addressable bits. An addressable bit may be specified by its bit address of 00H to 7FH, or 8 bits may form any byte address from 20H to 2FH. For example, bit address 4EH refers bit 6 of the byte address 29H.
3. The RAM area above bit addressable area from 30H to 7FH is called general purpose RAM. It is addressable as byte.

See Fig. 11.3 on next page.

11.3.4 Internal ROM

The 8051 has 4 Kbyte of internal ROM with address space from 0000H to 0FFFH. It is programmed by manufacturer when the chip is built. This part cannot be erased or altered after fabrication. This is used to store final version of the program.

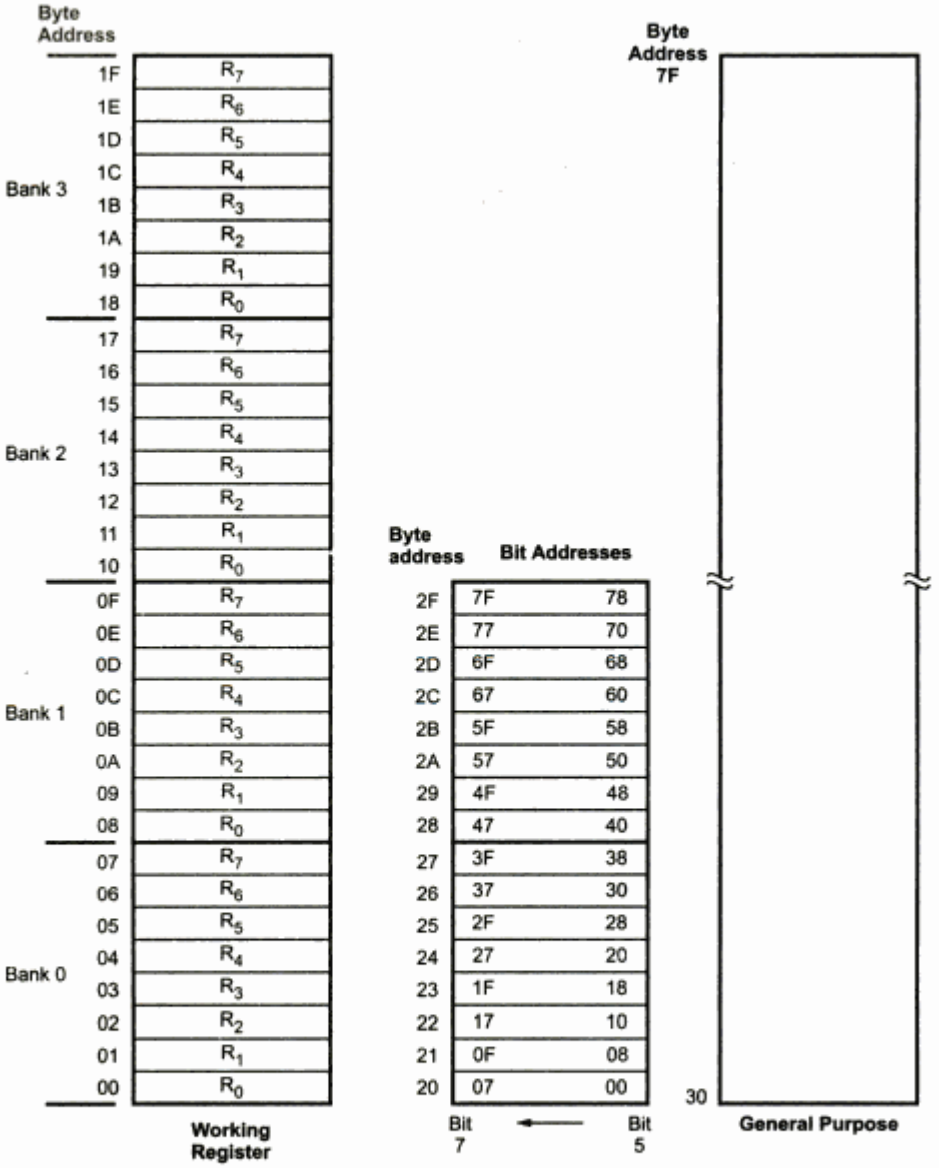


Fig. 11.3 Organisation of Internal RAM of 8051

0000H

Input/Output Ports

The 8051 has 32 I/O pins configured as four eight-bit parallel ports (P0, P1, P2, and P3). All four ports are bidirectional, i.e. each pin will be configured as input or output (or both) under software control. Each port consists of a latch, an output driver, and an input buffer.

The output drives of Ports 0 and 2 and the input buffers of Port 0, are used to access external memory. As mentioned earlier, Port 0 outputs the low order byte cf the external memory address, time multiplexed with the data being written or read, and Port 2 outputs the high order byte of the external memory address when the address is 16 bits wide. Otherwise Port 2 gives the contents of special function register P2.

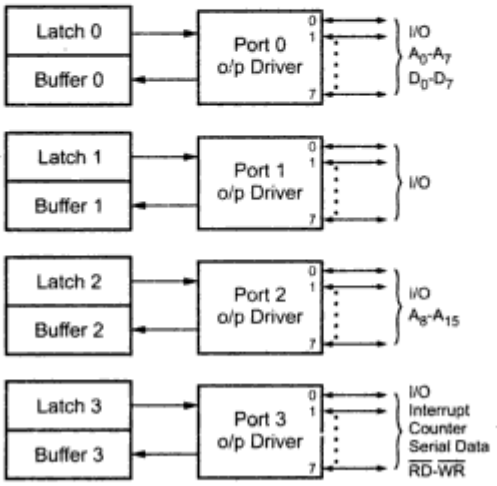


Fig. 11.4 I/O Ports

11.3.6 Register Set of 8051

11.3.6.1 Register A (Accumulator)

It is an 8-bit register. It holds a source operand and receives the result of the arithmetic instructions (addition, subtraction, multiplication, and division). The accumulator can be the source or destination for logical operations and a number of special data movement instructions, including look-up tables and external RAM expansion. Several functions apply exclusively to the accumulator : rotate, parity computation , testing for zero , and so on.

11.3.6.2 Register B

In addition to accumulator, an 8-bit B-register is available as a general purpose register when it is not being used for the hardware multiply/divide operation.

11.3.6.3 Program Status Word (Flag Register)

Many instructions implicitly or explicitly affect (or are affected by) several status flags, which are grouped together to form the Program Status Word. Fig. 11.5 shows the bit pattern of the program status word. It is an 8-bit word, containing the information as follows.

B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0
CY	AC	F0	RS1	RS0	OV	-	P

CY Bit 7 - Carry flag

AC Bit 6 - Auxiliary carry flag for BCD operations

F0 Bit 5 - User defined flag (Flag zero)

RS1, RS0 Bit 4-3 - Select the working register banks as follows :

RS1	RS0	Bank Selection	
0	0	00H - 07H	Bank 0
0	1	08H - 0FH	Bank 1
1	0	10H - 17H	Bank 2
1	1	18H - 1FH	Bank 3

OV Bit 2 - Overflow flag

– Bit 1 - Reserved

P Bit 0 - Parity flag (1 = Even parity)

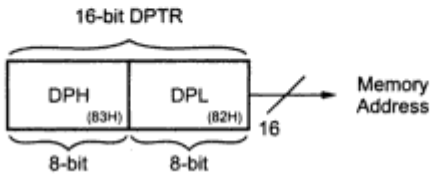
Fig. 11.5 Program status word

11.3.6.4 Stack and Stack Pointer

The stack refers to an area of internal RAM that is used in conjunction with certain opcodes data to store and retrieve data quickly. The stack pointer register is used by the 8051 to hold an internal RAM address that is called **top of stack.** The stack pointer register is 8-bit wide. It is increased **before** data is stored during PUSH and CALL instructions and decremented **after** data is restored during POP and RET instructions. Thus stack array can reside anywhere in on-chip RAM. The stack pointer is initialized to 07H after a reset. This causes the stack to begin at location 08H. The operation of stack and stack pointer is illustrated in Fig. 11.6.

11.3.6.5 Data Pointer (DPTR)

The data pointer (DPTR) consists of a high byte (DPH) and a low byte (DPL). Its function is to hold a 16 bit address. It may be manipulated as a 16 bit data register or as two independent 8 bit registers. It serves as a base register in indirect jumps, lookup table instructions and external data transfer. The DPTR does not have a single internal address; DPH (83H) and DPL (82H) have separate internal addresses.



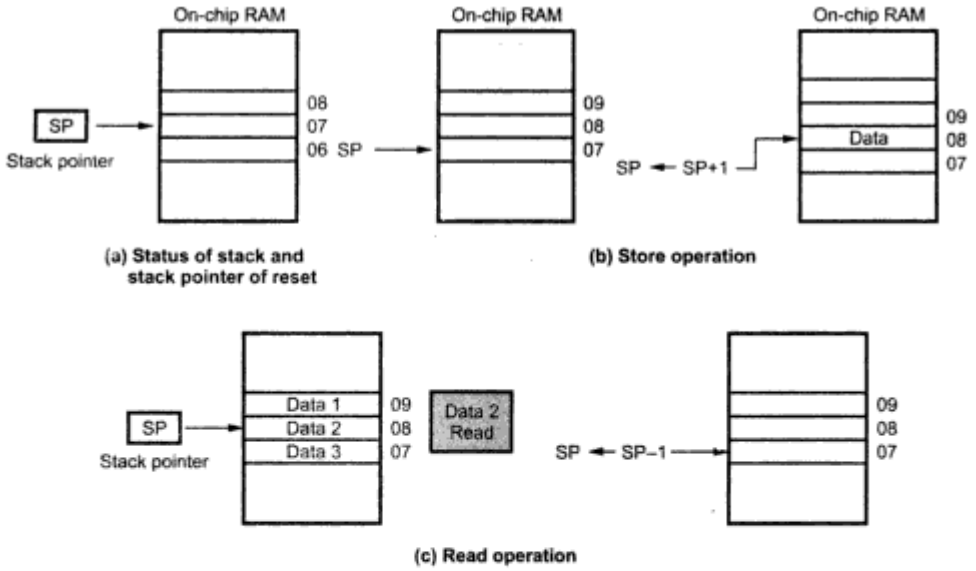


Fig. 11.6

11.3.6.6 Program Counter

The 8051 has a 16-bit program counter. It is used to hold the address of memory location from which the next instruction is to be fetched. Due to this the width of the program counter decides the maximum program length in bytes. For example, 8051 is 16-bit hence it can address upto 2^{16} bytes (64 K) of memory.

The PC is automatically incremented to point the next instruction in the program sequence after execution of the current instruction. It may also be altered by certain instructions. The PC is the only register that does not have an internal address.

11.3.6.7 Special Function Registers

Unlike other microprocessors in the Intel family, 8051 uses memory mapped I/O through a set of special function registers that are implemented in the address space immediately above the 128 bytes of RAM. Fig. 11.7 shows special function bit addresses. All access to the four I/O ports, the CPU registers, interrupt-control registers, the timer/counter, UART, and power control are performed through registers between 80H and FFH.

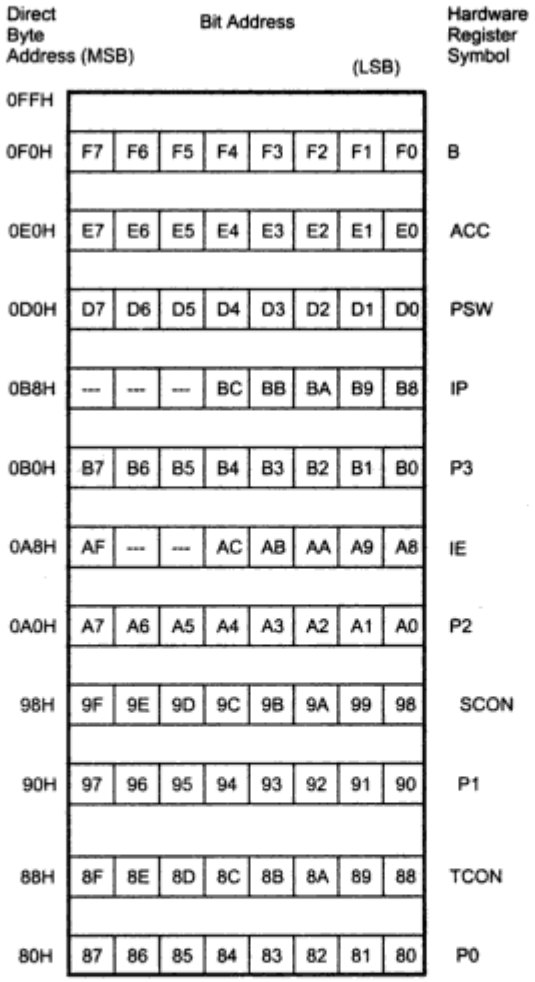


Fig. 11.7 SFR bit address

Symbol	Name	Address	Value in Binary
*ACC	Accumulator	0E0H	0 0 0 0 0 0 0 0
*B	B Register	0F0H	0 0 0 0 0 0 0 0
*PSW	Program Status Word	0D0H	0 0 0 0 0 0 0 0
SP	Stack Pointer	81H	0 0 0 0 0 1 1 1
DPTR DPL DPH	Data Pointer 2 Bytes Low Byte High Byte	 82H 83H	 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
*P0	Port 0	80H	1 1 1 1 1 1 1 1
*P1	Port 1	90H	1 1 1 1 1 1 1 1
*P2	Port 2	0A0H	1 1 1 1 1 1 1 1
*P3	Port 3	0B0H	1 1 1 1 1 1 1 1
*IP	Interrupt Priority Control	0B8H	8051 X X X 0 0 0 0 0 8052 X X 0 0 0 0 0 0
*IE	Interrupt Enable Control	0A8H	8051 0 X X 0 0 0 0 0 8052 0 X 0 0 0 0 0 0
TMOD	Timer/Counter Mode Control	89H	0 0 0 0 0 0 0 0
*TCON	Timer/Counter Control	88H	0 0 0 0 0 0 0 0
* + T2CON	Timer/Counter 2 Control	0C8H	0 0 0 0 0 0 0 0
TH0	Timer/Counter 0 High Byte	8CH	0 0 0 0 0 0 0 0
TL0	Timer/Counter 0 Low Byte	8AH	0 0 0 0 0 0 0 0
TH1	Timer/Counter 1 High Byte	8DH	0 0 0 0 0 0 0 0
TL1	Timer/Counter 1 LowByte	8BH	0 0 0 0 0 0 0 0
+ TH2	Timer/Counter 2 High Byte	0CDH	0 0 0 0 0 0 0 0
+ TL2	Timer/Counter 2 Low Byte	0CCH	0 0 0 0 0 0 0 0
+ RCAP2H	T/C 2 Capture Reg. High Byte	0CBH	0 0 0 0 0 0 0 0
+ RCAP2L	T/C 2 Capture Reg. Low Byte	0CAH	0 0 0 0 0 0 0 0
* SCON	Serial Control	98H	0 0 0 0 0 0 0 0
SBUF	Serial Data Buffer	99H	Interminate
PCON	Power Control	87H	HMOS 0 X X X X X X X CHMOS 0 X X X 0 0 0 0

Table 11.3 List of all SFRs (* = Bit addressable, + = 8052 only)

Memory Organization in 8051

Fig. 11.8 shows the basic memory structure for 8051. It can access upto 64 K program memory and 64 K data memory. The 8051 has 4 Kbytes of internal program memory and 256 bytes of internal data memory.

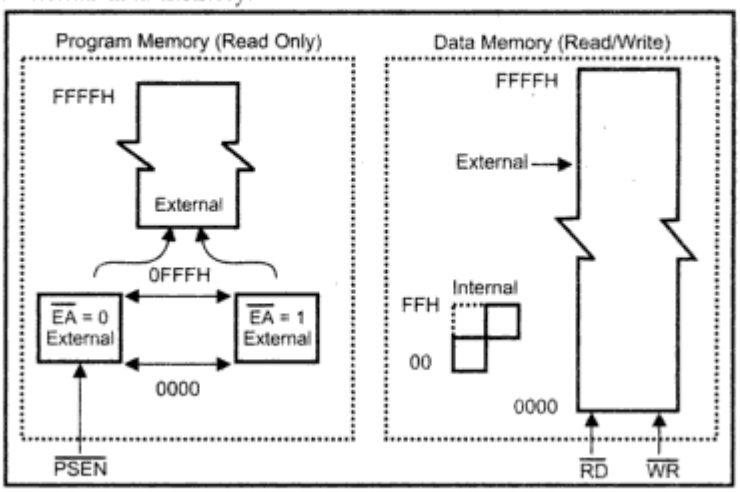


Fig. 11.8 Memory structure

External Data Memory and Program Memory

We have seen that 8051 has internal data and code memory with limited memory capacity. This memory capacity may not be sufficient for some applications. In such situations, we have to connect external ROM/EPROM and RAM to 8051 microcontroller to increase the memory capacity. We also know that ROM is used as a program memory and RAM is used as a data memory. Let us see how 8051 accesses these memories.

External Program Memory

Fig. 11.10 shows a map of the 8051 program memory.

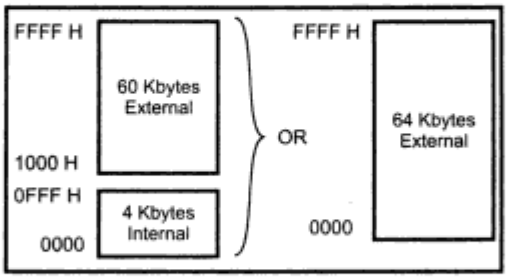


Fig. 11.10 The 8051 program memory

In 8051, when the $\overline{EA}$ pin is connected to V_{CC}, program fetches to addresses 0000H through 0FFFH are directed to the internal ROM and program fetches to addresses 1000H through FFFFH are directed to external ROM/EPROM. On the other hand when $\overline{EA}$ pin is grounded, all addresses (0000H to FFFFH) fetched by program are directed to the external ROM/EPROM. The $\overline{PSEN}$ signal is used to activate output enable signal of the external ROM/EPROM, as shown in the Fig. 11.11.

External Data Memory and Program Memory

We have seen that 8051 has internal data and code memory with limited memory capacity. This memory capacity may not be sufficient for some applications. In such situations, we have to connect external ROM/EPROM and RAM to 8051 microcontroller to increase the memory capacity. We also know that ROM is used as a program memory and RAM is used as a data memory. Let us see how 8051 accesses these memories.

External Program Memory

Fig. 11.10 shows a map of the 8051 program memory.

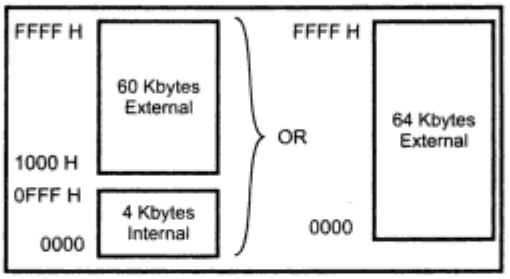


Fig. 11.10 The 8051 program memory

In 8051, when the $\overline{EA}$ pin is connected to V_{CC}, program fetches to addresses 0000H through 0FFFH are directed to the internal ROM and program fetches to addresses 1000H through FFFFH are directed to external ROM/EPROM. On the other hand when $\overline{EA}$ pin is grounded, all addresses (0000H to FFFFH) fetched by program are directed to the external ROM/EPROM. The $\overline{PSEN}$ signal is used to activate output enable signal of the external ROM/EPROM, as shown in the Fig. 11.11.

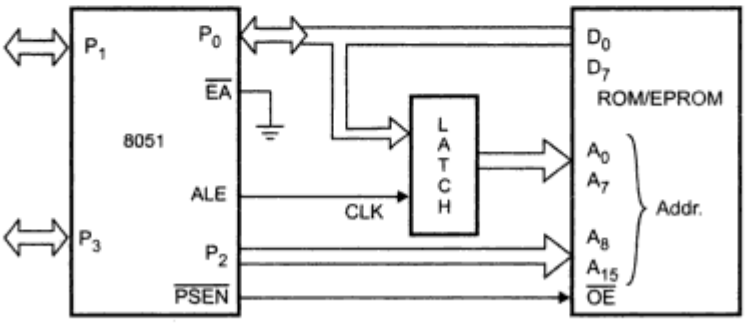


Fig. 11.11 Accessing external program memory

Timer 0 and Timer 1

In this timers, "Timer" or Counter" mode is selected by control bits $C/\overline{T}$ in the Special Function Register TMOD (Fig. 11.18). These two Timer/Counters have four operating modes, which are selected by bit-pairs (M1, M0) in TMOD. Modes 0, 1 and 2 are same for both Timer/Counters. Mode 3 is different. The four operating modes are described as follows :

(MSB) (LSB)

GATE	$C/\overline{T}$	M1	M0	GATE	$C/\overline{T}$	M1	M0
Timer 1				Timer 0			

GATE Gating control when set. Timer/Counter "x" is enabled only while "INTx" pin is high and "TRx" control bit is set. When cleared Timer "x" is enabled whenever "TRx" control bit is set.

$C/\overline{T}$ Timer or Counter selector cleared for Timer operation (input from internal system clock). Set for Counter operation (input from "Tx" input pin).

M1	M0	Operating Mode
0	0	8-bit Timer/Counter "THx" with "TLx"s 5-bit prescaler.
0	1	16-bit Timer/Counter "THx" with "TLx" are cascaded; there is no prescaler.
1	0	8-bit auto-reload Timer/Counter "THx" holds a value which is to be reloaded into "TLx" each time it overflows.
1	1	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit timer only controlled by Timer 1 control bits.
1	1	(Timer 1) Timer/Counter 1 stopped.

TMOD : Timer/counter mode control register

MODE 0

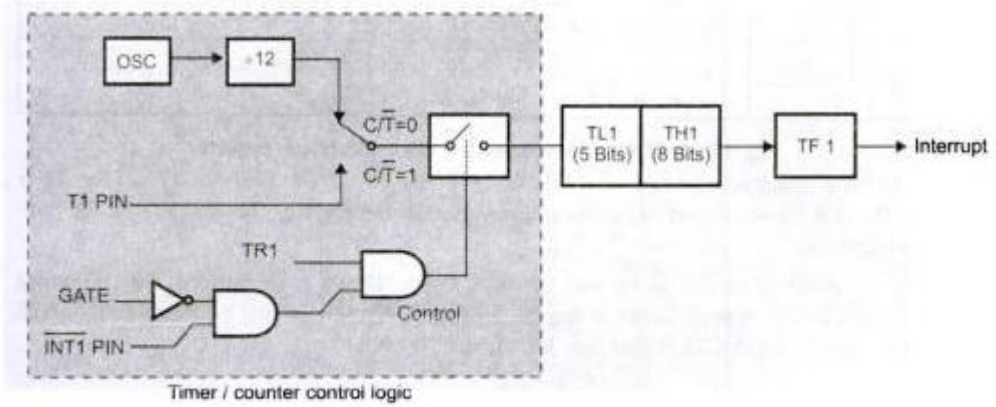


Fig. 11.19 Timer/counter 1 mode 0 : 13-bit counter

Both Timers in Mode 0 is an 8-bit Counter with a divide-by-32 prescaler. This 13-bit timer is MCS-48 compatible. Fig. 11.19 shows the Mode 0 operation as it applies to Timer 1. In this mode, the Timer register is configured as a 13-bit register. As the count rolls over from all 1s to all 0s, it sets the Timer interrupt flag TF1. The counted input is enabled to the Timer when TR1 = 1 and either GATE = 0 or $\overline{INT1}$ = 1. (Setting GATE = 1 allows the Timer to be controlled by external input $\overline{INT1}$, to facilitate pulse width measurements.) TR1 is a control bit in the Special Function Register TCON (Fig. 11.20) GATE is in TMOD.

(MSB) (LSB)

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

Symbol	Position	Name and Significance
TF1	TCON.7	Timer 1 Overflow Flag. Set by hardware on timer/counter overflow. Cleared when interrupt processed.
TR1	TCON.6	Timer 1 Run control bit. Set/cleared by software to turn timer/counter on/off.
TF0	TCON.5	Timer 0 Overflow Flag. Set by hardware on timer/counter overflow. Cleared when interrupt processed.
TR0	TCON.4	Timer 0 Run control bit. Set/cleared by software to turn timer/counter on/off.
IE1	TCON.3	Interrupt 1 Edge Flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
IT1	TCON.2	Interrupt 1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
IE0	TCON.1	Interrupt 0 Edge Flag. Set by hardware when external interrupt edge detected. Cleared when interrupt processed.
IT0	TCON.0	Interrupt 0 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.

Fig 11.20 TCON-timer/counter control/status register

The 13-bit register consists of all 8 bits of TH1 and the lower 5 bits of TL1. The upper 3 bits of TL1 are indeterminate and should be ignored. Setting the run flag (TR1) does not clear the registers.

RI	SCON.0	Receive Interrupt flag. Set by hardware when byte received. Cleared by software after servicing.
	Note : **Mode** 0 1 2 3	The state of (SM0, SM1) selects ; **SM0 SM1** 0 0 - Shift register ; baud = f/12 0 1 - 8-bit UART, variable data rate. 1 0 - 9-bit UART, fixed data rate ; baud = f/32 or f/64 1 1 - 9-bit UART, variable data rate.

Fig. 11.24 (a) SCON-serial port control/status register

(MSB) (LSB)

7	6	5	4	3	2	1	0
SMOD	–	–	–	GF1	GF0	PD	IDL

Symbol	Position	Name and Significance
SMOD	PCON.7	Serial baud rate modify bit. It is 0 at reset. It is set to 1 by program to double the baud rate.
–	PCON.6-4	Not defined
GF1	PCON.3	General purpose user flag bit 1. Set/cleared by program.
GF0	PCON.2	General purpose user flag bit 0. Set/cleared by program.
PD	PCON.1	Power down bit. It is set to 1 by program to enter power down configuration for CHMOS microcontrollers.
IDL	PCON.0	Idle mode bit. It is set to 1 by program to enter idle mode configuration for CHMOS microcontrollers.
Note : PCON is not bit addressable		

Fig. 11.24 (b) PCON register

Operating Modes for Serial Port

MODE 0

In this mode, serial data enters and exits through RXD. TXD outputs the shift clock. 8 bits are transmitted/received : 8 data bits (LSB first). The baud rate is fixed at 1/12 the oscillator frequency.

MODE 1

In this mode, 10 bits are transmitted (through TXD) or received (through RXD) : a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB8 in Special Function Register SCON. The baud rate is variable.

MODE 2

In this mode, 11 bits are transmitted (thorugh TXD) or received (through RXD) : a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On Transmit, the 9th data bit (TB8 in SCON) can be assigned the value of 0 or 1. Or, for example, the parity bit (P, in the PSW) could be moved into TB8. On receive, the 9th data bit goes into RB8 in Special Function Regsiter SCON, while the stop bit is ignored. The baud rate is programmable to either $\frac{1}{32}$ or $\frac{1}{64}$ the oscillator frequency.

MODE 3

In this mode, 11 bits are transmitted (through TXD) or received (through RXD) : a start bit (0), 8 data bits (LSB first), a programmable 9th data bit and a stop bit (1). In fact, Mode 3 is the same as Mode 2 in all respects except the baud rate. The baud rate in Mode 3 is variable.

In all four modes, transmission is initiated by any instruction that uses SBUF as a destination register. Reception is initiated in Mode 0 by the condition RI = 0 and REN = 1. Reception is initiated in the other modes by the incoming start bit if REN = 1.

The Table 11.9 summaries the four serial port modes provided by 8051.

Mode	Transmission Format	Baud Rate
0	8-data bits	$\frac{1}{12}$ oscillator frequency
1	10-bit (start bit + 8-data bits + stop bit)	Variable
2	11-bit (start bit + 8-data bits + programmable 9^{th} data bit + stop bit)	Programmable to either $\frac{1}{32}$ or $\frac{1}{64}$ oscillator frequency
3	11-bit (start bit + 8 data bit + programmable 9^{th} data bit + stop bit)	Variable

Table 11.9 Summary of serial port modes

UNIT-VIII

Atmel AVR RISC microcontroller:

The **AVR** is a modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to one-time programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time. However, it is commonly accepted that AVR stands for **A**lf (Egil Bogen) and **V**egard (Wollan)'s **R**isc processor" the use of "AVR" generally refers to the 8-bit RISC line of Atmel AVR Microcontrollers.

Among the first of the AVR line was the AT90S8515, which in a 40-pin DIP package has the same pinout as an 8051 microcontroller, including the external multiplexed address and data bus. The polarity of the RESET line was opposite (8051's having an active-high RESET, while the AVR has an active-low RESET) but other than that, the pinout was identical.

overview

The AVR is a modified Harvard architecture machine where program and data are stored in separate physical memory systems that appear in different address spaces, but having the ability to read data items from program memory using special instructions.

Basic families

AVRs are generally classified into five broad groups:

- **tinyAVR** — the ATtiny series
  - 0.5–8 kB program memory
  - 6–32-pin package
  - Limited peripheral set
- **megaAVR** — the ATmega series
  - 4–256 kB program memory
  - 28–100-pin package
  - Extended instruction set (Multiply instructions and instructions for handling larger program memories)
  - Extensive peripheral set
- **XMEGA** — the ATxmega series
  - 16–384 kB program memory
  - 44–64–100-pin package (A4, A3, A1)

    - Extended performance features, such as DMA, "Event System", and cryptography support.
    - Extensive peripheral set with DACs
- **Application-specific AVR**
    - megaAVRs with special features not found on the other members of the AVR family, such as LCD controller, USB controller, advanced PWM, CAN etc.
- **FPSLIC™ (AVR with FPGA)**
    - FPGA 5K to 40K gates
    - SRAM for the AVR program code, unlike all other AVRs
    - AVR core can run at up to 50 MHz [5]
- **32-bit AVRs**

  In 2006 Atmel released microcontrollers based on the new, 32-bit, AVR32 architecture. They include SIMD and DSP instructions, along with other audio and video processing features. This 32-bit family of devices is intended to compete with the ARM based processors. The instruction set is similar to other RISC cores, but is not compatible with the original AVR or any of the various ARM cores.

Architecture

Flash, EEPROM, and SRAM are all integrated onto a single chip, removing the need for external memory in most applications. Some devices have a parallel external bus option to allow adding additional data memory or memory-mapped devices. Almost all devices (except the smallest TinyAVR chips) have serial interfaces, which can be used to connect larger serial EEPROMs or flash chips. The fast-access register file concept contains 32 x 8-bit general-purpose working registers With a single clock cycle access time. This means that during one single clock cycle, one Arithmetic Logic Unit (ALU)operation is executed. Two operands are output from the register file, the operation is executed and the result is stored back in the register file–in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect addresses register pointers for Data Space addressing, enabling efficient address calculations. One of the three address pointers is also used as the address pointer for the constant table look-up function.

These addedfunctionregistersarethe16-bitX-register, Y-register, and Z-register.

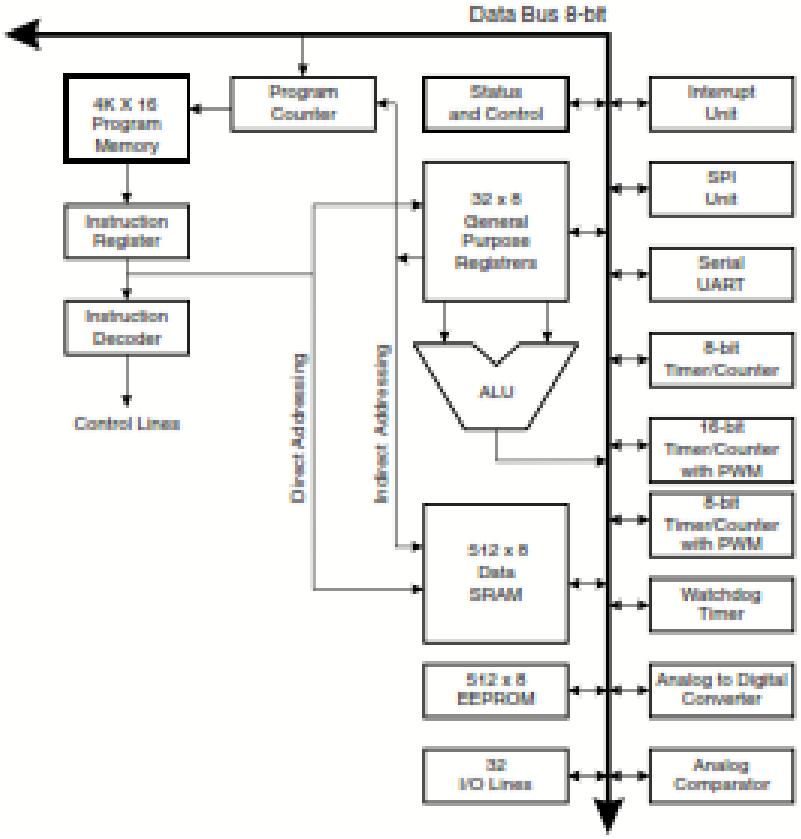


Figure. The AT90S8535 AVR RISC Architecture

The ALU supports arithmetic and logic functions between registers or between a constant

And a register. Single register operations are also executed in the ALU.Figure4 shows the AT90S8535 AVR RISC microcontroller architecture.

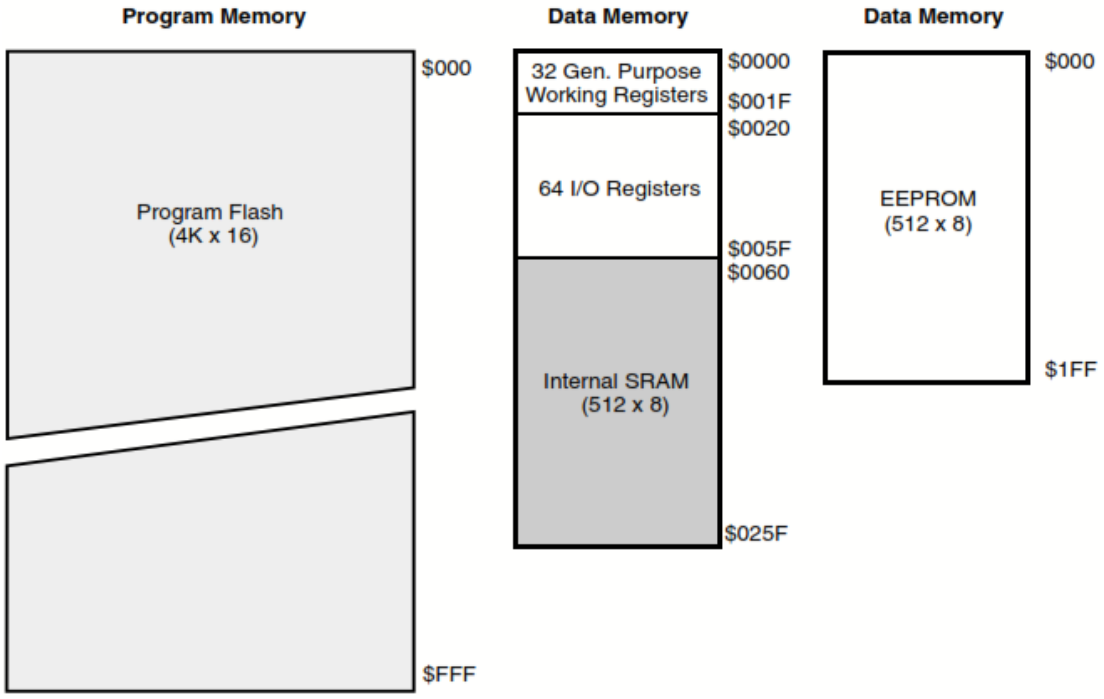
Memory Maps

In addition to the register operation, the conventional memory addressing modes can be used on the register file as well. This is enabled by the fact that the register file is assigned the 32 lowermost Data Space addresses ($00 - $1F), allowing them to be accessed as though they were ordinary memory locations. The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, Timer/Counters, A/D converters and other I/O functions. The I/O memory can be accessed directly or as the Data Space locations following those of the register file, $20 - $5F. The AVR uses a Harvard architecture concept – with separate memories and buses for program and data. The program memory is executed with a two-stage pipeline. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is in-system downloadable Flash memory. With the relative jump and call instructions, the whole 4K address space is directly accessed. Most AVR

instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction. During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the stack. The stack is effectively allocated in the general data SRAM and consequently, the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The 10-bit stack pointer (SP) is read/write-accessible in the I/O space.

The 512 bytes data SRAM can be easily accessed through the five different addressing modes supported in the AVR architecture.The memory spaces in the AVR architecture are all linear and regular memory maps.

Figure 5. Memory Maps



Program memory

Program instructions are stored in non-volatile flash memory. Although the MCUs are 8-bit, each instruction takes one or two 16-bit words. The size of the program memory is usually indicated in the naming of the device itself (e.g., the ATmega64x line has 64 kB of flash while the ATmega32x line has 32 kB).

There is no provision for off-chip program memory; all code executed by the AVR core must reside in the on-chip flash. However, this limitation does not apply to the AT94 FPSLIC AVR/FPGA chips.

Internal data memory

The data address space consists of the register file, I/O registers, and SRAM.

Internal registers

- Atmel ATxmega128A1 in 100-pin TQFP package
- The AVRs have 32 single-byte registers and are classified as 8-bit RISC devices.
- In most variants of the AVR architecture, the working registers are mapped in as the first 32 memory addresses (0000_{16}–$001F_{16}$) followed by the 64 I/O registers (0020_{16}–$005F_{16}$).

Actual SRAM starts after these register sections (address 0060_{16}). (Note that the I/O register space may be larger on some more extensive devices, in which case the memory mapped I/O registers will occupy a portion of the SRAM address space.) Even though there are separate addressing schemes and optimized opcodes for register file and I/O register access, all can still be addressed and manipulated as if they were in SRAM.

In the XMEGA variant, the working register file is not mapped into the data address space; as such, it is not possible to treat any of the XMEGA's working registers as though they were SRAM. Instead, the I/O registers are mapped into the data address space starting at the very beginning of the address space. Additionally, the amount of data address space dedicated to I/O registers has grown substantially to 4096 bytes (0000_{16}–$0FFF_{16}$). As with previous generations, however, the fast I/O manipulation instructions can only reach the first 64 I/O register locations (the first 32 locations for bitwise instructions). Following the I/O registers, the XMEGA series sets aside a 4096 byte range of the data address space which can be used optionally for mapping the internal EEPROM to the data address space (1000_{16}–$1FFF_{16}$). The actual SRAM is located after these ranges, starting at 2000_{16}.

EEPROM

Almost all AVR microcontrollers have internal EEPROM for semi-permanent data storage. Like flash memory, EEPROM can maintain its contents when electrical power is removed.In most variants of the AVR architecture, this internal EEPROM memory is not mapped into the MCU's addressable memory space. It can only be accessed the same way an external peripheral device is, using special pointer registers and read/write instructions which makes EEPROM access much slower than other internal RAM.

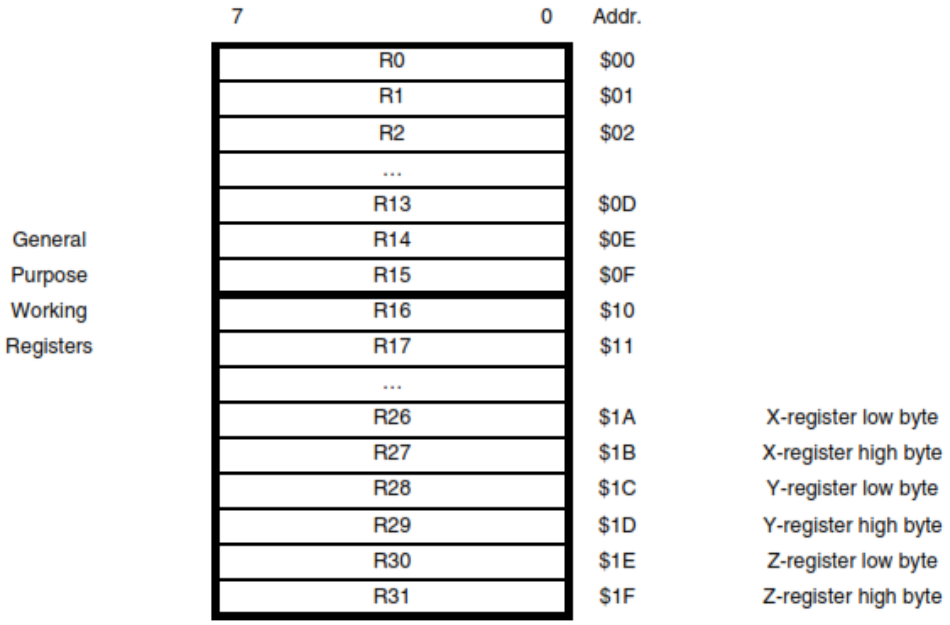
However, some devices in the SecureAVR (AT90SC) family [6] use a special EEPROM mapping to the data or program memory depending on the configuration. The XMEGA family also allows the EEPROM to be mapped into the data address space. Since the number

of writes to EEPROM is not unlimited — Atmel specifies 100,000 write cycles in their datasheets — a well designed EEPROM write routine should compare the contents of an EEPROM address with desired contents and only perform an actual write only if contents need to be changed.

General-purpose Register File

Figure 6 shows the structure of the 32 general-purpose working registers in the CPU.

Figure 6. AVR CPU General-purpose Working Registers



All the register operating instructions in the instruction set have direct and single-cycle

access to all registers. The only exception is the five constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI and ORI between a constant and a register and the LDI instruction for load immediate constant data. These instructions apply to the second half of the registers in the register file (R16..R31). The general SBC, SUB, CP, AND, and OR and all other operations between two registers or on a single register apply to the entire register file.

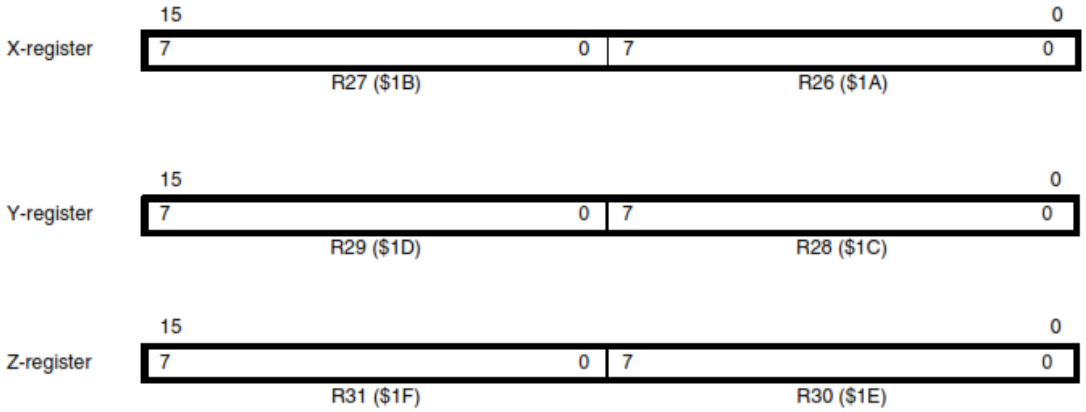
As shown in Figure 6, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically

implemented as SRAM locations,this memory organization provides great flexibility in access

of the registers, as the X-, Y-, and Z-registers can be set to index any register in the file. X-register, Y-register and Z-register The registers R26..R31 have some added functions to their

general-purpose usage.These registers are address pointers for indirect addressing of the Data Space. The three indirect address registers, X, Y, and Z, are defined in Figure 7.

Figure 7. X-, Y-, and Z-register



ALU – Arithmetic Logic Unit

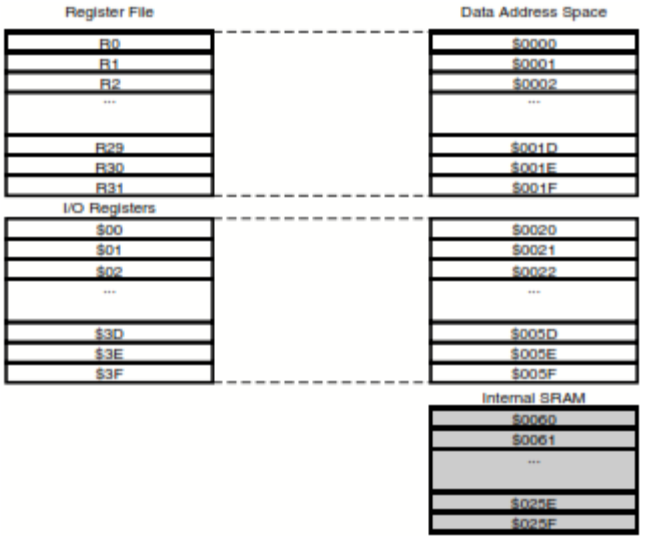
The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, ALU operations between registers in the register file are executed.The ALU operations are divided into three main categories: arithmetic, logical and bit functions.

EEPROM Data Memory

The AT90S8535 contains 512 bytes of data EEPROM memory. It is organized as a separate Data space, in which single bytes can be read and written. The EEPROM has an endurance of At least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described On page 51 specifying the EEPROM address registers, the EEPROM data register and the EEPROM control register.

SRAM Data Memory Figure 8 shows how the AT90S8535 SRAM memory is organized.

Figure 8. SRAM Organization



The lower 608 data memory locations address the Register file, the I/O memory and the internal data SRAM. The first 96 locations address the Register file + I/O memory, and the next 512 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect,Indirect with Pre-decrement and Indirect with Post-increment. In the Register file, registers R26 to R31 feature the indirect addressing pointer registers.

The direct addressing reaches the entire data space. The Indirect with Displacement mode features 63 address locations reached from the base address given by the Y- or Z-registers. When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y, and Z are decremented and incremented.

The 32 general-purpose working registers, 64 I/O registers and the 512 bytes of internal data SRAM in the AT90S8535 are all accessible through all these addressing modes.Memory Access Times and Instruction Execution Timing This section describes the general access timing concepts for instruction execution and internal memory access.

The AVR CPU is driven by the System Clock Ø, directly generated from the external clock crystal for the chip. No internal clock division is used. Figure 20 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks and functions per power-unit.

Figure 20. The Parallel Instruction Fetches and Instruction Executions

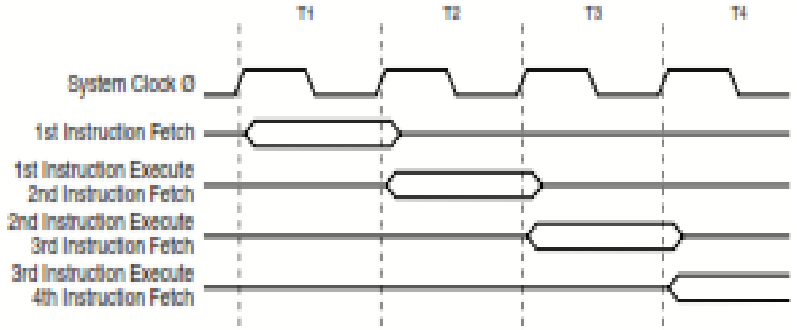
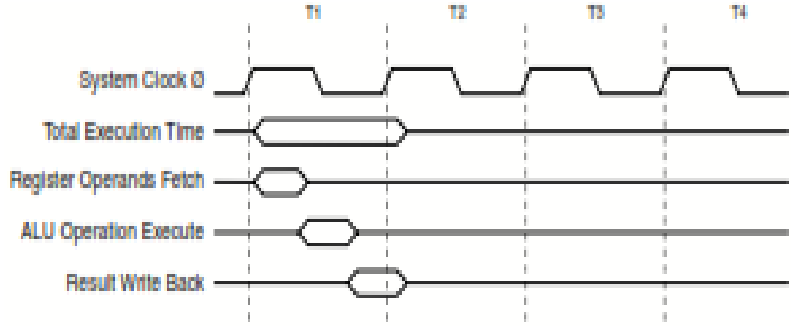


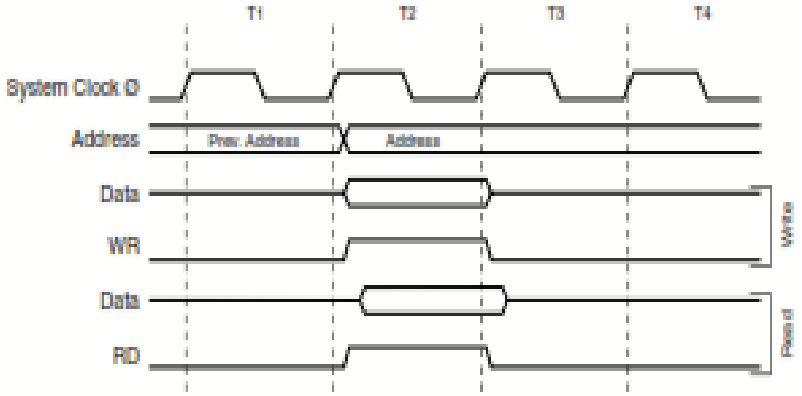
Figure 21 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed and the result is stored back to the destination register.

Figure 21. Single Cycle ALU Operation



The internal data SRAM access is performed in two System Clock cycles as described in Figure 22.

Figure 22. On-chip Data SRAM Access Cycles



I/O Memory The I/O space definition of the AT90S8535 is shown in Table 1.

Table 1. AT90S8535 I/O Space

I/O Address (SRAM Address)	Name	Function
$3F ($5F)	SREG	Status REGister
$3E ($5E)	SPH	Stack Pointer High
$3D ($5D)	SPL	Stack Pointer Low
$3B ($5B)	GIMSK	General Interrupt MaSK register
$3A ($5A)	GIFR	General Interrupt Flag Register
$39 ($59)	TIMSK	Timer/Counter Interrupt MaSK register
$38 ($58)	TIFR	Timer/Counter Interrupt Flag register
$35 ($55)	MCUCR	MCU general Control Register
$34 ($45)	MCUSR	MCU general Status Register
$33 ($53)	TCCR0	Timer/Counter0 Control Register
$32 ($52)	TCNT0	Timer/Counter0 (8-bit)
$2F ($4F)	TCCR1A	Timer/Counter1 Control Register A
$2E ($4E)	TCCR1B	Timer/Counter1 Control Register B
$2D ($4D)	TCNT1H	Timer/Counter1 High Byte
$2C ($4C)	TCNT1L	Timer/Counter1 Low Byte
$2B ($4B)	OCR1AH	Timer/Counter1 Output Compare Register A High Byte
$2A ($4A)	OCR1AL	Timer/Counter1 Output Compare Register A Low Byte
$29 ($49)	OCR1BH	Timer/Counter1 Output Compare Register B High Byte
$28 ($48)	OCR1BL	Timer/Counter1 Output Compare Register B Low Byte
$27 ($47)	ICR1H	T/C 1 Input Capture Register High Byte
$26 ($46)	ICR1L	T/C 1 Input Capture Register Low Byte
$25 ($45)	TCCR2	Timer/Counter2 Control Register
$24 ($44)	TCNT2	Timer/Counter2 (8-bit)
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register
$22 ($42)	ASSR	Asynchronous Mode Status Register
$21 ($41)	WDTCR	Watchdog Timer Control Register
$1F ($3E)	EEARH	EEPROM Address Register High Byte
$1E ($3E)	EEARL	EEPROM Address Register Low Byte
$1D ($3D)	EEDR	EEPROM Data Register
$1C ($3C)	EECR	EEPROM Control Register
$1B ($3B)	PORTA	Data Register, Port A
$1A ($3A)	DDRA	Data Direction Register, Port A
$19 ($39)	PINA	Input Pins, Port A
$18 ($38)	PORTB	Data Register, Port B

Table 1. AT90S8535 I/O Space (Continued)

I/O Address (SRAM Address)	Name	Function
$17 ($37)	DDRB	Data Direction Register, Port B
$16 ($36)	PINB	Input Pins, Port B
$15 ($35)	PORTC	Data Register, Port C
$14 ($34)	DDRC	Data Direction Register, Port C
$13 ($33)	PINC	Input Pins, Port C
$12 ($32)	PORTD	Data Register, Port D
$11 ($31)	DDRD	Data Direction Register, Port D
$10 ($30)	PIND	Input Pins, Port D
$0F ($2F)	SPDR	SPI I/O Data Register
$0E ($2E)	SPSR	SPI Status Register
$0D ($2D)	SPCR	SPI Control Register
$0C ($2C)	UDR	UART I/O Data Register
$0B ($2B)	USR	UART Status Register
$0A ($2A)	UCR	UART Control Register
$09 ($29)	UBRR	UART Baud Rate Register
$08 ($28)	ACSR	Analog Comparator Control and Status Register
$07 ($27)	ADMUX	ADC Multiplexer Select Register
$06 ($26)	ADCSR	ADC Control and Status Register
$05 ($25)	ADCH	ADC Data Register High
$04 ($24)	ADCL	ADC Data Register Low

Note: Reserved and unused locations are not shown in the table.

All AT90S8535 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions transferring data between the 32 general-purpose working registers and the I/O space. I/O registers within the address range $00 $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses $00- $3F must be used. When addressing I/O registers as SRAM, $20 must be added to these addresses. All I/O register addresses throughout this document are shown with the SRAM address in parentheses.

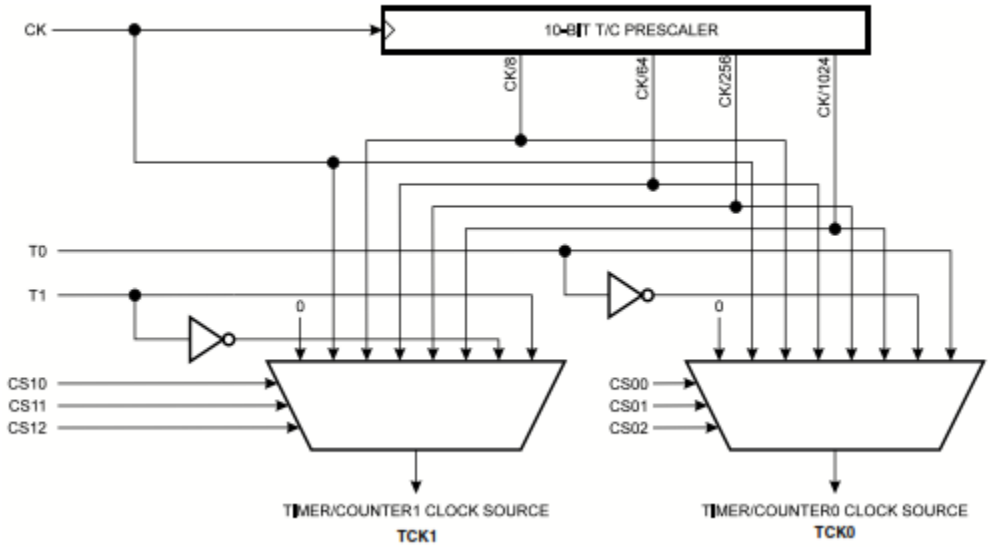
Timer/Counters the AT90S8535 provides three general-purpose Timer/Counters – two 8-bit T/Cs and one 16-bit T/C. Timer/Counter2 can optionally be asynchronously clocked from an external oscillator. This oscillator is optimized for use with a 32.768 kHz watch crystal, enabling

Use of Timer/Counter2 as a Real-time Clock (RTC). Timer/Counters 0 and 1 have individual pre scaling selection from the same 10-bit prescaling timer. Timer/Counter2 has its own prescaler.

These Timers/Counters

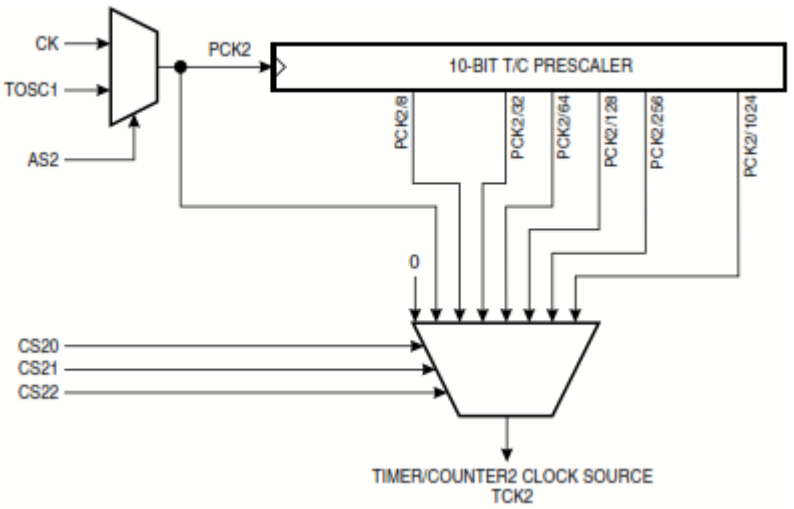
can either be used as a timer with an internal clock time base or as a counter with an external pin connection that triggers the counting. Timer/Counter Prescalers

Figure 28. Prescaler for Timer/Counter0 and 1



For Timer/Counters 0 and 1, the four different prescaled selections are: CK/8, CK/64, CK/256 and CK/1024, where CK is the oscillator clock. For the two Timer/Counters 0 and 1, CK, external source and stop can also be selected as clock sources.

Figure 29. Timer/Counter2 Prescaler



The clock source For Timer/Counter2 prescaler is named PCK2. PCK2 is by default connected to the main system clock (CK). By setting the AS2 bit in ASSR, Timer/Counter2 prescaler is asynchronously clocked from the PC6(TOSC1) pin. This enables use ofTimer/Counter2 as aReal-time Clock (RTC). When AS2 is set, pins PC6 (TOSC1) and PC7 (TOSC2) are disconnected from Port C. A crystal can then be connected between the PC6(TOSC1) and PC7 (TOSC2) pins to serve as an independent clock source for Timer/Counter2. The

oscillator Is optimized for use with a 32.768 kHz crystal. Applying an external clock source to TOSC1 is Not recommended. 8-bit Timer/Counter0 Figure 30 shows the block diagram for Timer/Counter0.

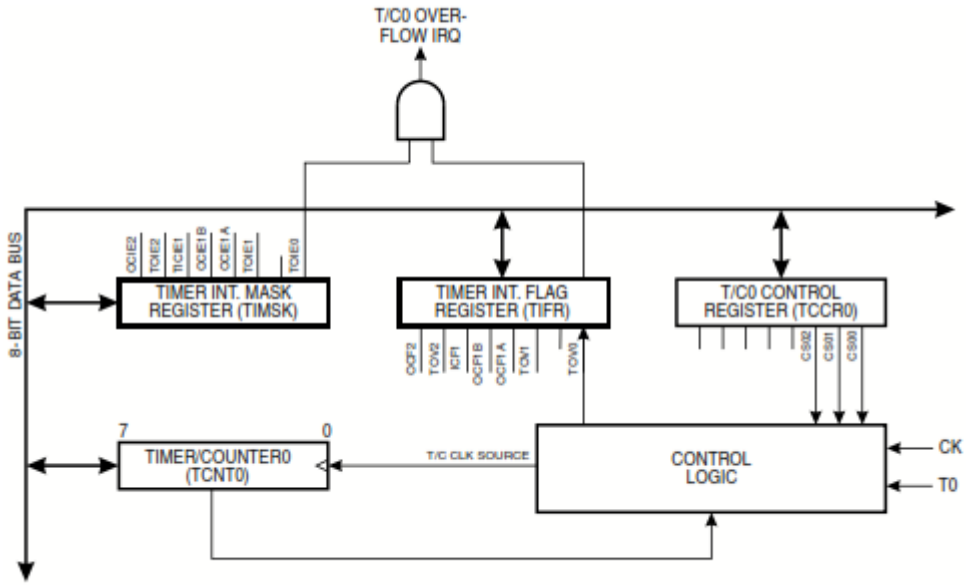
The 8-bit Timer/Counter0 can select clock source from CK, prescaled CK or an external

pin. In addition, it can be stopped as described in the specification for the Timer/Counter0 control Register (TCCR0). The overflow status flag is found in the Timer/Counter Interrupt Flag Register (TIFR). Control signals are found in the Timer/Counter0 Control Register (TCCR0). The interrupt enable/disable settings for Timer/Counter0 are found in the Timer/Counter Interrupt Mask Register (TIMSK).

When Timer/Counter0 is externally clocked, the external signal is synchronized with the

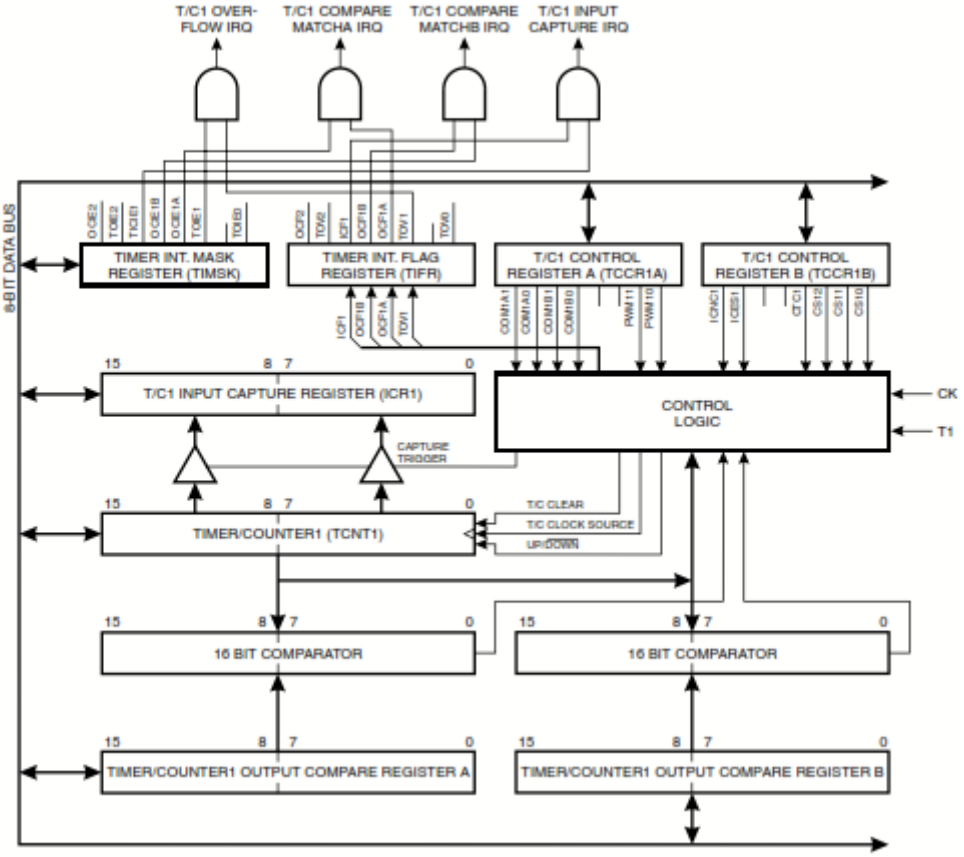
oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock. The 8-bit Timer/Counter0 features both a high-resolution and a high-accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make the Timer/Counter0 useful for lower speed functions or exact timing functions with infrequent actions.

Figure 30. Timer/Counter0 Block Diagram



16-bit Timer/Counter1 Figure 31 shows the block diagram for Timer/Counter1.

Figure 31. Timer/Counter1 Block Diagram



The 16-bit Timer/Counter1 can select clock source from CK, prescaled CK or an external

pin. In addition, it can be stopped as described in the specification for the Timer/Counter1 Control Registers (TCCR1A and TCCR1B). The different status flags (Overflow, Compare Match and Capture Event) and control signals are found in the Timer/Counter1 Control Registers (TCCR1A and TCCR1B). The interrupt enable/disable settings for Timer/Counter1 are found in the Timer/Counter Interrupt Mask Register (TIMSK). When Timer/Counter1 is externally locked, the external signal is synchronized with the oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock. The 16-bit Timer/Counter1 features both a high-resolution and a high-accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities makes the Timer/Counter1 useful for lower speed functions or exact timing functions with infrequent actions.

The Timer/Counter1 supports two Output Compare functions using the Output Compare Register 1A and B (OCR1A and OCR1B) as the data sources to be compared to the Timer/Counter1 contents. The Output Compare functions include optional clearing of I/O

Ports All AVR ports have true read-modify-write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies for changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input).

Port A

Port A is an 8-bit bi-directional I/O port.

Three I/O memory address locations are allocated for Port A, one each for the Data Register – PORTA, $1B($3B), Data Direction Register – DDRA, $1A($3A) and the PortA Input Pins – PINA, $19($39). The Port A Input Pins address is read-only, while the Data Register and the Data Direction Register are read/write.

All port pins have individually selectable pull-up resistors. The Port A output buffers can

sink 20 mA and thus drive LED displays directly. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. Port A has an alternate function as analog inputs for the ADC. If some Port A pins are configured as outputs, it is essential that these do not switch when a conversion is in progress. This might corrupt the result of the conversion.

During Power-down mode, the Schmitt trigger of the digital input is disconnected. This allows analog signals that are close to V /2 to be present during power-down without causing excessive power consumption.

Port A Data Register – PORTA

Bit	7	6	5	4	3	2	1	0	
$1B ($3B)	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port A Data Direction Register – DDRA

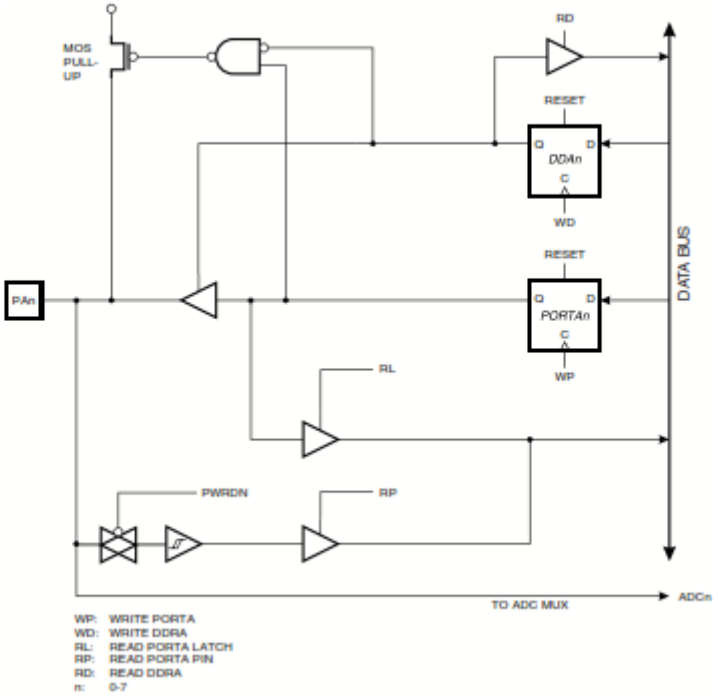
Bit	7	6	5	4	3	2	1	0	
$1A ($3A)	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port A Input Pins Address – PINA

Bit	7	6	5	4	3	2	1	0	
$19 ($39)	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

The Port A Input Pins address (PINA) is not a register; this address enables access to the physical value on each Port A pin. When reading PORTA, the Port A Data Latch is read and when reading PINA, the logical values present on the pins are read.

Figure 51. Port A Schematic Diagrams (Pins PA0 - PA7)



Port B

Port B is an 8-bit bi-directional I/O port.

Three I/O memory address locations are allocated for the Port B, one each for the Data

Register – PORTB, $18($38), Data Direction Register – DDRB, $17($37) and the Port B Input Pins – PINB, $16($36). The Port B Input Pins address is read-only, while the Data Register and the Data Direction Register are read/write.

All port pins have individually selectable pull-up resistors. The Port B output buffers can sink 20 mA and thus drive LED displays directly. When pins PB0 to PB7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

Port B Data Register – PORTB

Bit	7	6	5	4	3	2	1	0	
$18 ($38)	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

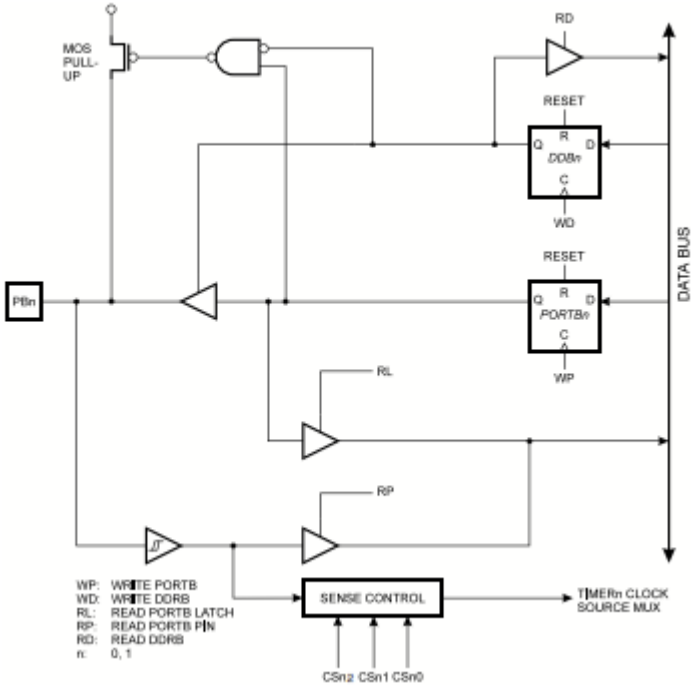
Port B Data Direction Register – DDRB

Bit	7	6	5	4	3	2	1	0	
$17 ($37)	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port B Input Pins Address – PINB

Bit	7	6	5	4	3	2	1	0	
$16 ($36)	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Figure 52. Port B Schematic Diagram (Pins PB0 and PB1)



Port C : Port C is an 8-bit bi-directional I/O port.

Three I/O memory address locations are allocated for the Port C, one each for the Data

Register – PORTC, $15($35), Data Direction Register – DDRC, $14($34) and the Port C Input Pins – PINC, $13($33). The Port C Input Pins address is read-only, while the Data Register and the Data Direction Register are read/write.

All port pins have individually selectable pull-up resistors. The Port C output buffers can

sink 20 mA and thus drive LED displays directly. When pins PC0 to PC7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated

.

Port C Data Register – PORTC

Bit	7	6	5	4	3	2	1	0	
$15 ($35)	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	PORTC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

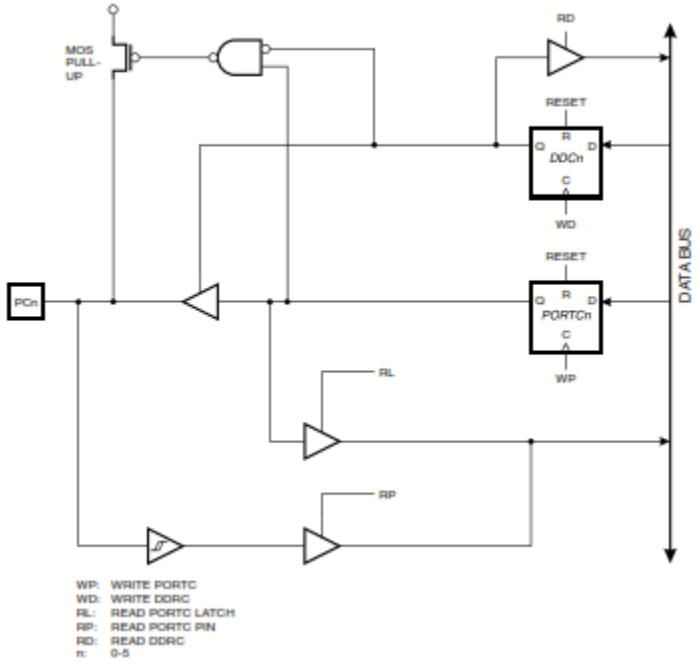
Port C Data Direction Register – DDRC

Bit	7	6	5	4	3	2	1	0	
$14 ($34)	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port C Input Pins Address – PINC

Bit	7	6	5	4	3	2	1	0	
$13 ($33)	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Figure 58. Port C Schematic Diagram (Pins PC0 - PC5)



Port D : Port D is an 8-bit bi-directional I/O port with internal pull-up resistors.

Three I/O memory address locations are allocated for Port D, one each for the Data Register – PORTD, $12($32), Data Direction Register – DDRD, $11($31) and the Port D Input Pins – PIND, $10($30). The Port D Input Pins address is read-only, while the Data Register and the Data Direction Register are read/write.

The Port D output buffers can sink 20 mA. As inputs, Port D pins that are externally

pulled low will source current if the pull-up resistors are activated. Some Port D pins have alternate functions

Port D Data Register – PORTD

Bit	7	6	5	4	3	2	1	0	
$12 ($32)	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

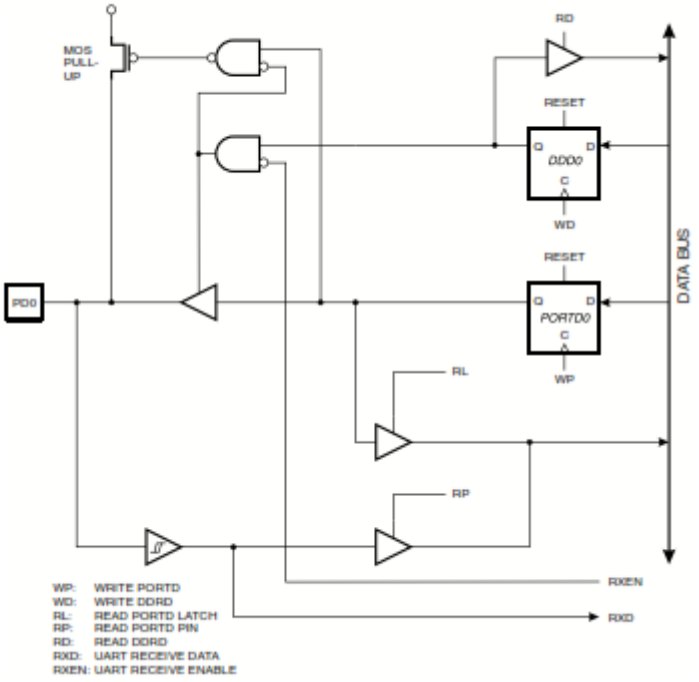
Port D Data Direction Register – DDRD

Bit	7	6	5	4	3	2	1	0	
$11 ($31)	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port D Input Pins Address – PIND

Bit	7	6	5	4	3	2	1	0	
$10 ($30)	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Figure 61. Port D Schematic Diagram (Pin PD0)



UART

The AT90S8535 features a full duplex (separate receive and transmit registers) Universal Asynchronous Receiver and Transmitter (UART). The main features are:

- Baud Rate Generator that can Generate a Large Number of Baud Rates (bps)

• High Baud Rates at Low XTAL Frequencies

• 8 or 9 Bits Data

• Noise Filtering

• Overrun Detection

• Framing Error Detection

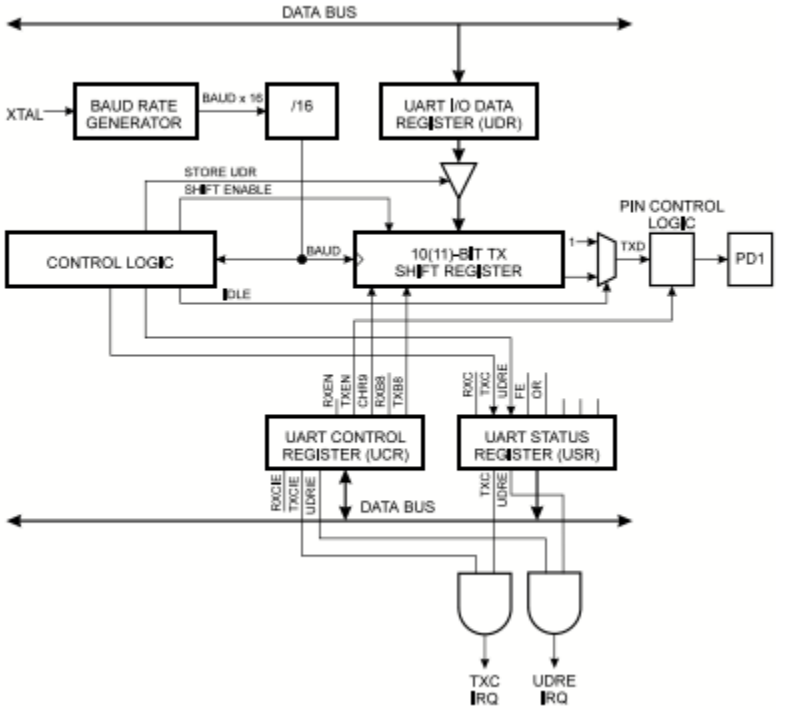
• False Start Bit Detection

• Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete

• Buffered Transmit and Receive

Data Transmission A block schematic of the UART transmitter is shown in Figure 41.

Figure 41. UART Transmitter



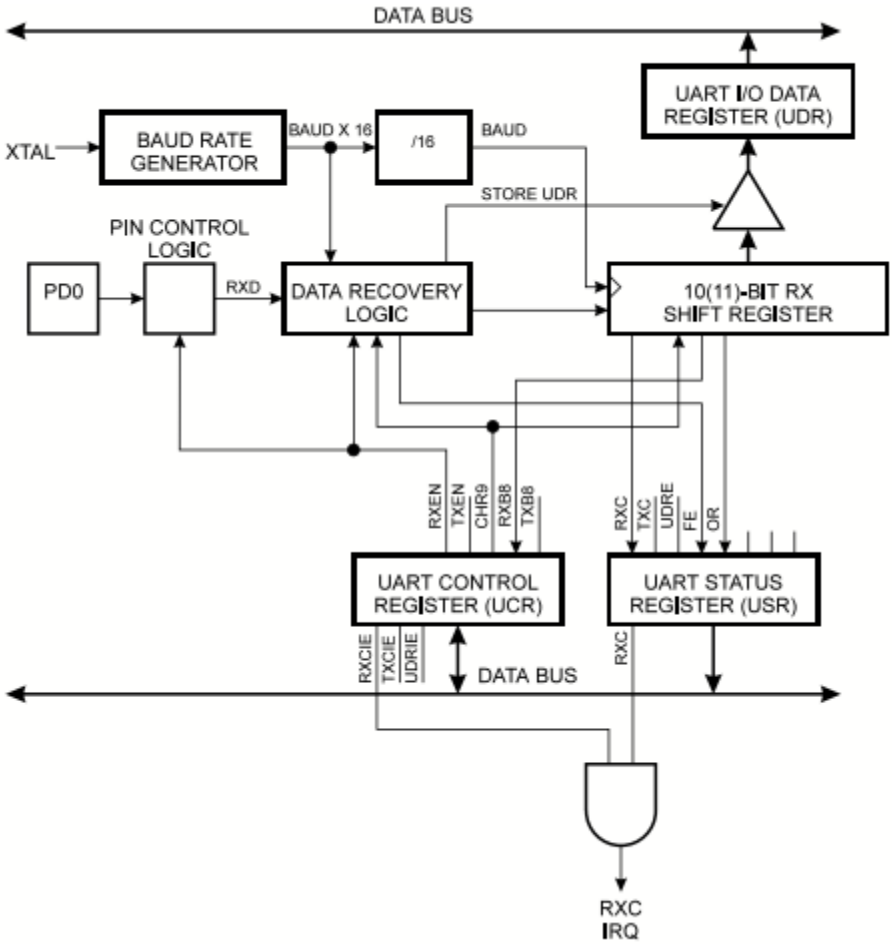
Data transmission is initiated by writing the data to be transmitted to the UART I/O Data Register, UDR. Data is transferred from UDR to the Transmit shift register when:

• A new character is written to UDR after the stop bit from the previous character has been shifted out. The shift register is loaded immediately.

• A new character is written to UDR before the stop bit from the previous character has been shifted out. The shift register is loaded when the stop bit of the character currently being transmitted is shifted out.

Data Reception Figure 42 shows a block diagram of the UART Receiver.

Figure 42. UART Receiver

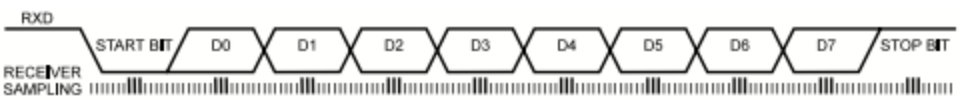


The receiver front-end logic samples the signal on the RXD pin at a frequency 16 times the baud rate. While the line is idle, one single sample of logical “0” will be interpreted as the falling edge of a start bit and the start bit detection sequence is initiated. Let sample 1 denote the first zero-sample. Following the 1-to-0 transition, the receiver samples the RXD pin at samples 8, 9 and 10. If two or more of these three samples are found to be logical “1”s, the start bit is rejected as a noise spike and the receiver starts looking for the next 1-to-0 transition.

If however, a valid start bit is detected, sampling of the data bits following the start bit is

performed. These bits are also sampled at samples 8, 9 and 10. The logical value found in at least two of the three samples is taken as the bit value. All bits are shifted into the Transmitter Shift register as they are sampled. Sampling of an incoming character is shown in Figure 43.

Figure 43. Sampling Received Data



When the stop bit enters the receiver, the majority of the three samples must be one to accept the stop bit. If two or more samples are logical “0”s, the Framing Error (FE) flag in the UART Status Register (USR) is set. Before reading the UDR register, the user should always check the FE bit to detect framing errors.

Whether or not a valid stop bit is detected at the end of a character reception cycle, the data is transferred to UDR and the RXC flag in USR is set. UDR is in fact two physically separate registers, one for transmitted data and one for received data. When UDR is read, the Receive Data register is accessed and when UDR is written, the Transmit Data register is accessed. If 9-bit data word is selected (the CHR9 bit in the UART Control Register, UCR is set), the RXB8 bit in UCR is loaded with bit 9 in the Transmit Shift register When data is transferred to UDR. Reset and Interrupt Handling The AT90S8535 provides 16 different interrupt sources. These interrupts and the separate reset vector each have a separate Program vector in the program memory space. All interrupts are assigned individual enable bits That must be set (one) together with the I-bit in the Status Register in order to enable the interrupt. The lowest addresses in the program memory space are automatically defined as the Reset and Interrupt vectors. The complete list of vectors is shown in Table 2. The list also determines the priority levels of the different interrupts. The lower the address, the higher the priority level. RESET has the highest priority, and next is INT0 (the External Interrupt Request 0), etc.

Table 2. Reset and Interrupt Vectors

Vector No.	Program Address	Source	Interrupt Definition
1	$000	RESET	Hardware Pin, Power-on Reset and Watchdog Reset
2	$001	INT0	External Interrupt Request 0
3	$002	INT1	External Interrupt Request 1
4	$003	TIMER2 COMP	Timer/Counter2 Compare Match
5	$004	TIMER2 OVF	Timer/Counter2 Overflow
6	$005	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$006	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$007	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$008	TIMER1 OVF	Timer/Counter1 Overflow
10	$009	TIMER0 OVF	Timer/Counter0 Overflow
11	$00A	SPI, STC	SPI Serial Transfer Complete
12	$00B	UART, RX	UART, Rx Complete

Table 2. Reset and Interrupt Vectors (Continued)

Vector No.	Program Address	Source	Interrupt Definition
13	$00C	UART, UDRE	UART Data Register Empty
14	$00D	UART, TX	UART, Tx Complete
15	$00E	ADC	ADC Conversion Complete
16	$00F	EE_RDY	EEPROM Ready
17	$010	ANA_COMP	Analog Comparator

The most typical and general program setup for the Reset and Interrupt vector addresses are:

Reset Sources The AT90S8535 has three sources of reset:

• Power-on Reset. The MCU is reset when the supply voltage is below the Power-on

• External Reset. The MCU is reset when a low level is present on the RESET pin for Reset threshold (V POT more than 50 ns)..

• Watchdog Reset. The MCU is reset when the Watchdog timer period expires and the Watchdog is enabled.

During reset, all I/O registers are set to their initial values and the program starts execu-

tion from address $000. The instruction placed in address $000 must be an RJMP (relative jump) instruction to the reset handling routine. If the program never enables an interrupt source, the interrupt vectors are not used and regular program code can be used.

Instruction set

Main article: Atmel AVR instruction set

The AVR Instruction Set is more orthogonal than those of most eight-bit microcontrollers, in particular the 8051 clones and PIC microcontrollers with which AVR competes today. However, it is not completely regular:

- Pointer registers X, Y, and Z have addressing capabilities that are different from each other.
- Register locations R0 to R15 have different addressing capabilities than register locations R16 to R31.
- I/O ports 0 to 31 have different addressing capabilities than I/O ports 32 to 63.
- CLR affects flags, while SER does not, even though they are complementary instructions. CLR set all bits to zero and SER sets them to one. (Note that CLR is pseudo-op for EOR R, R; and SER is short for LDI R,$FF. Math operations such as EOR modify flags while moves/loads/stores/branches such as LDI do not.)

- Accessing read-only data stored in the program memory (flash) requires special LPM instructions; the flash bus is otherwise reserved for instruction memory.

Additionally, some chip-specific differences affect code generation. Code pointers (including return addresses on the stack) are two bytes long on chips with up to 128 kBytes of flash memory, but three bytes long on larger chips; not all chips have hardware multipliers; chips with over 8 kBytes of flash have branch and call instructions with longer ranges; and so forth. The mostly-regular instruction set makes programming it using C (or even Ada) compilers fairly straightforward. GCC has included AVR support for quite some time, and that support is widely used. In fact, Atmel solicited input from major developers of compilers for small microcontrollers, to determine the instruction set features that were most useful in a compiler for high-level languages.

Features

Current AVRs offer a wide range of features:

- Multifunction, bi-directional general-purpose I/O ports with configurable, built-in pull-up resistors
- Multiple internal oscillators, including RC oscillator without external parts
- Internal, self-programmable instruction flash memory up to 256 kB (384 kB on XMega)
    - In-system programmable using serial/parallel low-voltage proprietary interfaces or JTAG
    - Optional boot code section with independent lock bits for protection
- On-chip debugging (OCD) support through JTAG or debugWIRE on most devices
    - The JTAG signals (TMS, TDI, TDO, and TCK) are multiplexed on GPIOs. These pins can be configured to function as JTAG or GPIO depending on the setting of a fuse bit, which can be programmed via ISP or HVSP. By default, AVRs with JTAG come with the JTAG interface enabled.
    - debugWIRE uses the /RESET pin as a bi-directional communication channel to access on-chip debug circuitry. It is present on devices with lower pin counts, as it only requires one pin.
- Internal data EEPROM up to 4 kB
- Internal SRAM up to 16 kB (32 kB on XMega)
- External 64 kB little endian data space on certain models, including the Mega8515 and Mega162.

    - The external data space is overlaid with the internal data space, such that the full 64 kB address space does not appear on the external bus. An accesses to e.g. address 0100_{16} will access internal RAM, not the external bus.
    - In certain members of the XMega series, the external data space has been enhanced to support both SRAM and SDRAM. As well, the data addressing modes have been expanded to allow up to 16 MB of data memory to be directly addressed.
    - AVRs generally do not support executing code from external memory. Some ASSPs using the AVR core do support external program memory.
- 8-bit and 16-bit timers
    - PWM output (some devices have an enhanced PWM peripheral which includes a dead-time generator)
    - Input capture
- Analog comparator
- 10 or 12-bit A/D converters, with multiplex of up to 16 channels
- 12-bit D/A converters
- A variety of serial interfaces, including
    - I²C compatible Two-Wire Interface (TWI)
    - Synchronous/asynchronous serial peripherals (UART/USART) (used with RS-232, RS-485, and more)
    - Serial Peripheral Interface Bus (SPI)
    - Universal Serial Interface (USI) for two or three-wire synchronous data transfer
- Brownout detection
- Watchdog timer (WDT)
- Multiple power-saving sleep modes
- Lighting and motor control (PWM-specific) controller models
- CAN controller support
- USB controller support
    - Proper full-speed (12 Mbit/s) hardware & Hub controller with embedded AVR.
    - Also freely available low-speed (1.5 Mbit/s) (HID) bitbanging software emulations
- Ethernet controller support

- LCD controller support
- Low-voltage devices operating down to 1.8 V (to 0.7 V for parts with built-in DC–DC upconverter)
- picoPower devices
- DMA controllers and "event system" peripheral communication.
- Fast cryptography support for AES and DES

Memory

The memory of the Atmel AVR processors is a Modified Harvard architecture, in which the program and data memory are on separate buses to allow faster access and increased capacity. The AVR uses internal memory for data and program storage, and so does not require any external memory.

The four types of memories in a Atmel AVR are:

- Data memory: registers, I/O registers, and SRAM
- Program flash memory
- EEPROM
- Fuse bits

All these memories are on the same chip as the CPU core. Each kind of memory is separated from each other, in different locations on the chip. Address 0 in data memory is distinct from address 0 in program flash and address 0 in EEPROM.

Program Memory

All AVR microcontrollers have some amount of 16 bit wide non-volatile flash memory for program storage, from 1 KB up to 256 KB (or, 512-128K typical program words). The program memory holds the executable program opcodes and static data tables. Program memory is linearly addressed, and so mechanisms like page banking or segment registers are not required to call any function, regardless of its location in program memory. AVRs cannot use external program memory; the flash memory on the chip is the only program memory available to the AVR core.

The flash program memory can be reprogrammed using a programming tool, the most popular being those that program the chip *in situ* and are called in-system programmers (ISP). Atmel AVRs can also be reprogrammed with a high-voltage parallel or serial programmer, and via JTAG (again, *in situ*) on certain chips. The flash memory in an AVR can be re-programmed at least 10,000 times.

Many of the newer AVRs (MegaAVR series) have the capability to self-program the flash memory. This functionality is used mainly by bootloaders.

Data Memory

Data Memory includes the registers, the I/O registers, and internal SRAM. The AVR has thirty-two general purpose eight-bit registers (R0 to R31), six of which can be used in pairs as sixteen-bit pointers (X, Y, and Z).

All AVR microcontrollers have some amount of RAM, from 32 bytes up to several KB. This memory is byte addressable. The register file (both general and special purpose) is mapped into the first addresses and thus accessible also as RAM. Some of the tiniest AVR microcontrollers have only the register file as their RAM.

The data address space consists of the register file, I/O registers, and SRAM. The working registers are mapped in as the first thirty-two memory spaces (0000_{16}-$001F_{16}$) followed by the reserved space for up to 64 I/O registers (0020_{16}-$005F_{16}$). The actual usable SRAM starts after both these sections (address 0060_{16}). (Note that the I/O register space may be larger on some more extensive devices, in which case the beginning address of SRAM will be higher.) Even though there are separate addressing schemes and optimized opcodes for register file and I/O register access, they can still be addressed and manipulated as if they were SRAM.

The I/O registers (and the program counter) are reset to their default starting values when a reset occurs. The registers and the rest of SRAM have initial random values, so typically one of the first things a program does is clear them to all zeros or load them with some other initial value.The registers, I/O registers, and SRAM never wear out, no matter how many times they are written.

External Data Memory

Some of the higher pin-count AVR microcontrollers allow for external expansion of the data space, addressable up to 64 KB. When enabled, external SRAM is overlaid by internal SRAM; an access to address 0000_{16} in the data space will always resolve to on-chip memory. Depending on the amount of on-chip SRAM present in the particular AVR, anywhere from 512 bytes to several KB of external RAM will not be accessible. This usually does not cause a problem.

The support circuitry required is described in the datasheet for any device that supports external data memory, such as the Mega 162, in the "External Memory Interface" section. The support circuitry is minimal, consisting of a '573 or similar latch, and potentially some chip select logic. The SRAM chip select may be tied to a logic level that permanently enables the chip, or it may be driven by a pin from the AVR. For an SRAM of 32 KB or less, one option is to use a higher-order address line to drive the chip select line to the SRAM.

EEPROM Storage

Almost all AVR microcontrollers have internal EEPROM memory for non-volatile data storage. Only the Tiny11 and Tiny28 have no EEPROM.

EEPROM memory is not directly mapped in either the program or data space, but is instead accessed indirectly as a peripheral, using I/O registers. Many compilers available for the AVR hide some or all of the details of accessing EEPROM. IAR's C compiler for the AVR recognizes the compiler-specific keyword `__eeprom` on a variable declaration. Thereafter, a person writes code to read and write that variable with the same standard C syntax as normal variables (in RAM), but the compiler generates code to access the EEPROM instead of regular data memory.

Atmel's datasheets indicate that the EEPROM can be re-written a minimum of 100,000 times. An application must implement a wear-leveling scheme if it writes to the EEPROM so frequently that it will reach the write limit before it reaches the expected lifetime of the device. AVRs ship from the factory with the EEPROM erased, i.e. the value in each byte of EEPROM is FF_{16}.Many of the AVRs have errata about writing to EEPROM address 0 under certain power conditions (usually during brownout), and so Atmel recommends that programs not use that address in the EEPROM.

General Purpose I/O Ports

General Purpose I/O, or GPIO, pins are the digital I/O for the AVR family. These pins are true push-pull outputs. The AVR can drive a high or low level, or configure the pin as an input with or without a pull-up. GPIOs are grouped into "ports" of up to 8 pins, though some AVRs do not have enough pins to provide all 8 pins in a particular port, e.g. the Mega48/88/168 does not have a PortC7 pin. Control registers are provided for setting the data direction, output value (or pull-up enabled), and for reading the value on the pin itself. An individual pin can be accessed using bitwise manipulation instructions.

Each port has 3 control registers associated with it, DDRx, PORTx, and PINx. Each bit in those registers controls one GPIO pin, i.e. bit 0 in DDRA controls the data direction for PortA0 (often abbreviated PA0), and bit 0 in PORTA will control the data (or pullup) for PA0.

The DDR (Data Direction Register) controls whether the pin is an input or an output. When the pin is configured as an output, the corresponding bit in the PORT register will control the drive level to the pin, high or low. When the pin is configured as an input, the bit in the PORT register controls whether a pull-up is enabled or disabled on that pin. The PIN (Port Input) register was read-only on earlier AVRs, and was used to read the value on the

port pin, regardless of the data direction. Newer AVRs allow a write to the PIN register to toggle the corresponding PORT bit, which saves a few processor cycles when bit-banging an interface.

Timer/Counters

All AVRs have at least one 8-bit timer/counter. For brevity, a timer/counter is usually referred to as simply a timer. Some of the Tiny series have only one 8-bit timer. At the high end of the Mega series, there are chips with as many as six timers (two 8-bit and four 16-bit).

A timer can be clocked directly by the system clock, by a divided-down system clock, or by an external input (rising or falling edge). Some AVRs also include an option to use an external crystal, asynchronous to the system clock, which can be used for maintaining a real-time clock with a 32.768 kHz crystal.

The basic operation of a timer is to count up to FF_8 (or $FFFF_{16}$), roll over to zero, and set an overflow bit, which may cause an interrupt if enabled. The interrupt routine reloads the timer with the desired value in addition to any other processing required.

The value of a timer can be read back at any time, even while it is running. (There is a specific sequence documented in the datasheets to read back a 16-bit timer so that a consistent result is returned, since the AVR can only move 8 bits at a time.) A timer can be halted temporarily by changing its clock input to "disabled," then resumed by re-selecting the previous clock input.

PWM

Many of the AVRs include a compare register for at least one of the timers. The compare register can be used to trigger an interrupt and/or toggle an output pin (i.e. OC1A for Timer 1) when the timer value matches the value in the compare register. This may be done separately from the overflow interrupt, enabling the use of pulse-width modulation (PWM).Some AVRs also include options for phase-correct PWM, or phase- and frequency-correct PWM.

The Clear Timer on Compare (CTC) mode allows for the timer to be cleared when it matches a value in the compare register, before the timer overflows. Clearing the timer prior to overflow manipulates the timer resolution, allowing for greater control of the output frequency of a compare match. It can also simplify the counting of an external event.

The ATtiny26 is unique in its inclusion of a 64 MHz high-speed PWM mode. The 64 MHz clock is generated from a PLL, and is independent of, and asynchronous to, the processor clock.

Some AVRs also include complementary outputs suitable for controlling some motors. A dead-time generator (DTG) inserts a delay between one signal falling and the other signal rising so that both signals are never high at the same time. The high-end AT90PWM series allows the dead time to be programmed as a number of system clock cycles, while other AVRs with this feature simply use 1 clock cycle for the dead time.

Output Compare Modulator

An Output Compare Modulator (OCM), which allows generating a signal that is modulated with a carrier frequency. OCM requires two timers, one for the carrier frequency, and the second for the signal to be modulated. OCM is available on some of the Mega series.

Serial Communication

AVR microcontrollers are in general capable of supporting a plethora of serial communication protocols and serial bus standards. The exact types of serial communication support varies between the different members of the AVR microcontroller family.On top of support in hardware there is also often the option to implement a particular serial communication mechanism entirely in software. Typically this is used in case a particular AVR controller does not support some serial communication mechanism in hardware, the particular hardware is already in use (e.g. when two RS-232 interfaces are needed, but only one is supported in hardware), or the chip's hardware can't be used, because it shares pins with other chip functions, and such a function is already in used for the particular hardware. The latter often happens with the low-pin count AVRs in DIP packages.

Finally, there is also the possibility to use additional logic to implement a serial communication function. For example, most AVRs don't support the USB bus (some later ones do so, however). When using an AVR which doesn't support USB directly, a circuit designer can add USB functionality with a fixed-function chip such as the FTDI232 USB to RS-232 converter chip, or a general-purpose USB interface such as the PDIUSB11. Adding additional electronics is in fact necessary for some supported communication protocols, e.g. standard-compliant RS-232 communication requires adding voltage level converters like the MAX232.

The number of serial communication possibilities supported by a particular AVR can be confusing at times, in particular if the pins are shared with other chip functions. An intensive study of the particular AVR's datasheet is highly recommended. The serial communication features most commonly to be found on AVRs are discussed in the following sections.

15.ADDITIONAL TOPICS:

UNIVERSAL SERIAL BUS(USB)

Universal Serial Bus (**USB**) is an industry standard developed in the mid-1990s that defines the cables, connectors and communications protocols used in a bus for connection, communication and power supply between computers and electronic devices.[2]
USB was designed to standardize the connection of computer peripherals, such as keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters to personal computers, both to communicate and to supply electric power. It has become commonplace on other devices, such as smartphones, PDAs and video game consoles.[3] USB has effectively replaced a variety of earlier interfaces, such as serial and parallel ports, as well as separate power chargers for portable devices.
As of 2008, approximately 6 billion USB ports and interfaces are currently in the global marketplace, and about 2 billion were being sold each year.[4]

History

A group of seven companies began development on USB in 1994: Compaq, DEC, IBM, Intel, Microsoft, NEC and Nortel. The goal was to make it fundamentally easier to connect external devices to PCs by replacing the multitude of connectors at the back of PCs, addressing the usability issues of existing interfaces, and simplifying software configuration of all devices connected to USB, as well as permitting greater data rates for external devices. The first silicon for USB was made by Intel in 1995.[5]
The original USB 1.0 specification, which was introduced in January 1996, defined data transfer rates of 1.5 Mbit/s "Low Speed" and 12 Mbit/s "Full Speed".[5] The first widely used version of USB was 1.1, which was released in September 1998. The 12 Mbit/s data rate was intended for higher-speed devices such as disk drives, and the lower 1.5 Mbit/s rate for low data rate devices such as joysticks.[6]
The USB 2.0 specification was released in April 2000 and was ratified by the USB Implementers Forum (USB-IF) at the end of 2001. Hewlett-Packard, Intel, Lucent Technologies (now Alcatel-Lucent), NEC and Philipsjointly led the initiative to develop a higher data transfer rate, with the resulting specification achieving 480 Mbit/s, a fortyfold increase over the original USB 1.1 specification.
The USB 3.0 specification was published on 12 November 2008. Its main goals were to increase the data transfer rate (up to 5 Gbit/s), to decrease power consumption, to increase power output, and to be backwards-compatible with USB 2.0.[7] USB 3.0 includes a new, higher speed bus called SuperSpeed in parallel with the USB 2.0 bus.[8] For this reason, the new version is also called SuperSpeed.[9] The first USB 3.0 equipped devices were presented in January 2010.[9][10]

Prereleases

The USB standard evolved through several versions before its official release in 1995:

- *USB 0.7*: Released in November 1994.
- *USB 0.8*: Released in December 1994.
- *USB 0.9*: Released in April 1995.
- *USB 0.99*: Released in August 1995.
- *USB 1.0 Release Candidate*: Released in November 1995.

USB 1.0

- *USB 1.0*: Released in January 1996.
  Specified data rates of *1.5 Mbit/s* (*Low-Bandwidth*) and *12 Mbit/s* (*Full-Bandwidth*).

Does not allow for extension cables or pass-through monitors (due to timing and power limitations). Few such devices actually made it to market.

- *USB 1.1*: Released in August 1998.
  Fixed problems identified in 1.0, mostly relating to hubs. Earliest revision to be widely adopted.

USB 2.0

- *USB 2.0*: Released in April 2000.
  Added higher maximum bandwidth of *480 Mbit/s* (60 MB/s) (now called *"Hi-Speed"*). Further modifications to the USB specification have been done via Engineering Change Notices (ECN). The most important of these ECNs are included into the USB 2.0 specification package available from USB.org:
  - *Mini-A and Mini-B Connector ECN*: Released in October 2000.
    Specifications for Mini-A and B plug and receptacle. Also receptacle that accepts both plugs for On-The-Go. These should not be confused with Micro-B plug and receptacle.
  - *Errata as of December 2000*: Released in December 2000.
  - *Pull-up/Pull-down Resistors ECN*: Released in May 2002.
  - *Errata as of May 2002*: Released in May 2002.
  - *Interface Associations ECN*: Released in May 2003.
    New standard descriptor was added that allows multiple interfaces to be associated with a single device function.
  - *Rounded Chamfer ECN*: Released in October 2003.
    A recommended, compatible change to Mini-B plugs that results in longer lasting connectors.
  - *Unicode ECN*: Released in February 2005.
    This ECN specifies that strings are encoded using UTF-16LE. USB 2.0 did specify that Unicode is to be used but it did not specify the encoding.
  - *Inter-Chip USB Supplement*: Released in March 2006.
  - *On-The-Go Supplement 1.3*: Released in December 2006.
    USB On-The-Go makes it possible for two USB devices to communicate with each other without requiring a separate USB host. In practice, one of the USB devices acts as a host for the other device.
  - *Battery Charging Specification 1.1*: Released in March 2007 (Updated 15 Apr 2009).
    Adds support for dedicated chargers (power supplies with USB connectors), host chargers (USB hosts that can act as chargers) and the No Dead Battery provision which allows devices to temporarily draw 100 mA current after they have been attached. If a USB device is connected to dedicated charger, maximum current drawn by the device may be as high as 1.8 A. (Note that this document is not distributed with USB 2.0 specification package only USB 3.0 and USB On-The-Go.)
  - *Micro-USB Cables and Connectors Specification 1.01*: Released in April 2007.
  - *Link Power Management Addendum ECN*: Released in July 2007.
    This adds a new power state between enabled and suspended states. Device in this state is not required to reduce its power consumption. However, switching between enabled and sleep states is much faster than switching between enabled and suspended states, which allows devices to sleep while idle.
  - *Battery Charging Specification 1.2*[11]: Released in December 2010.
    Several changes and increasing limits including allowing 1.5A on charging

ports for unconfigured devices, allowing High Speed communication while having a current up to 1.5A and allowing a maximum current of 5A.

USB 3.0

USB 3.0 was released in November 2008. The standard specifies a maximum transmission speed of up to 5 Gbit/s (625 MB/s), which is more than 10 times as fast as USB 2.0 (480 Mbit/s, or 60 MB/s), although this speed is typically only achieved using powerful professional grade or developmental equipment. USB 3.0 reduces the time required for data transmission, reduces power consumption, and is backward compatible with USB 2.0. The USB 3.0 Promoter Group announced on 17 November 2008 that the specification of version 3.0 had been completed and had made the transition to the USB Implementers Forum (USB-IF), the managing body of USB specifications.[12] This move effectively opened the specification to hardware developers for implementation in future products. A new feature is the "SuperSpeed" bus, which provides a fourth transfer mode at 5.0 Gbit/s. The raw throughput is 4 Gbit/s (using 8b/10b encoding), and the specification considers it reasonable to achieve around 3.2 Gbit/s (0.4 GB/s or 400 MB/s), increasing as hardware advances in the future take hold. Two-way communication is also possible. In USB 3.0, full-duplex communications are done when using SuperSpeed (USB 3.0) transfer. In previous USB versions (i.e., 1.x or 2.0), all communication is half-duplex and directionally controlled by the host.

- *Battery Charging Specification 1.2*[11]: Released in December 2010. Several changes and increasing limits including allowing 1.5A on charging ports for unconfigured devices, allowing High Speed communication while having a current up to 1.5A and allowing a maximum current of 5A.

System design

The design architecture of USB is asymmetrical in its topology, consisting of a host, a multitude of downstream USB ports, and multiple peripheral devices connected in a tiered-star topology. Additional USB hubsmay be included in the tiers, allowing branching into a tree structure with up to five tier levels. A USB host may implement multiple host controllers and each host controller may provide one or more USB ports. Up to 127 devices, including hub devices if present, may be connected to a single host controller.[13][14]

USB devices are linked in series through hubs. One hub is known as the root hub which is built into the host controller.

A physical USB device may consist of several logical sub-devices that are referred to as *device functions*. A single device may provide several functions, for example, a webcam (video device function) with a built-in microphone (audio device function). This kind of device is called *composite device*. An alternative for this is *compound device* in which each logical device is assigned a distinctive address by the host and all logical devices are connected to a built-in hub to which the physical USB wire is connected.

USB device communication is based on *pipes* (logical channels). A pipe is a connection from the host controller to a logical entity, found on a device, and named an *endpoint*. Because pipes correspond 1-to-1 to endpoints, the terms are sometimes used interchangeably. A USB device can have up to 32 endpoints: 16 into the host controller and 16 out of the host controller. The USB standard reserves one endpoint of each type, leaving a theoretical maximum of 30 for normal use. USB devices seldom have this many endpoints.

There are two types of pipes: stream and message pipes depending on the type of data transfer.

- *isochronous transfers*: at some guaranteed data rate (often, but not necessarily, as fast as possible) but with possible data loss (e.g., realtime audio or video).
- *interrupt transfers*: devices that need guaranteed quick responses (bounded latency) (e.g., pointing devices and keyboards).

- *bulk transfers*: large sporadic transfers using all remaining available bandwidth, but with no guarantees on bandwidth or latency (e.g., file transfers).
- *control transfers*: typically used for short, simple commands to the device, and a status response, used, for example, by the bus control pipe number 0.

A stream pipe is a uni-directional pipe connected to a uni-directional endpoint that transfers data using an *isochronous*, *interrupt*, or *bulk* transfer. A message pipe is a bi-directional pipe connected to a bi-directional endpoint that is exclusively used for *control* data flow. An endpoint is built into the USB device by the manufacturer and therefore exists permanently. An endpoint of a pipe is addressable with a tuple *(device_address, endpoint_number)* as specified in a TOKEN packet that the host sends when it wants to start a data transfer session. If the direction of the data transfer is from the host to the endpoint, an OUT packet (a specialization of a TOKEN packet) having the desired device address and endpoint number is sent by the host. If the direction of the data transfer is from the device to the host, the host sends an IN packet instead. If the destination endpoint is a uni-directional endpoint whose manufacturer's designated direction does not match the TOKEN packet (e.g., the manufacturer's designated direction is IN while the TOKEN packet is an OUT packet), the TOKEN packet will be ignored. Otherwise, it will be accepted and the data transaction can start. A bi-directional endpoint, on the other hand, accepts both IN and OUT packets.
Endpoints are grouped into *interfaces* and each interface is associated with a single device function. An exception to this is endpoint zero, which is used for device configuration and which is not associated with any interface. A single device function composed of independently controlled interfaces is called a *composite device*. A composite device only has a single device address because the host only assigns a device address to a function.
When a USB device is first connected to a USB host, the USB device enumeration process is started. The enumeration starts by sending a reset signal to the USB device. The data rate of the USB device is determined during the reset signaling. After reset, the USB device's information is read by the host and the device is assigned a unique 7-bit address. If the device is supported by the host, the device drivers needed for communicating with the device are loaded and the device is set to a configured state. If the USB host is restarted, the enumeration process is repeated for all connected devices.
The host controller directs traffic flow to devices, so no USB device can transfer any data on the bus without an explicit request from the host controller. In USB 2.0, the host controller polls the bus for traffic, usually in a round-robin fashion. The throughput of each USB port is determined by the slower speed of either the USB port or the USB device connected to the port.
High-speed USB 2.0 hubs contain devices called transaction translators that convert between high-speed USB 2.0 buses and full and low speed buses. When a high-speed USB 2.0 hub is plugged into a high-speed USB host or hub, it will operate in high-speed mode. The USB hub will then either use one transaction translator per hub to create a full/low-speed bus that is routed to all full and low speed devices on the hub, or will use one transaction translator per port to create an isolated full/low-speed bus per port on the hub.
Because there are two separate controllers in each USB 3.0 host, USB 3.0 devices will transmit and receive at USB 3.0 data rates regardless of USB 2.0 or earlier devices connected to that host. Operating data rates for them will be set in the legacy manner.

Communication

During USB communication data is transmitted as packets. Initially, all packets are sent from the host, via the root hub and possibly more hubs, to devices. Some of those packets direct a device to send some packets in reply.
After the sync field, all packets are made of 8-bit bytes, transmitted least-significant bit first. The first byte is a packet identifier (PID) byte. The PID is actually 4 bits; the byte consists of

the 4-bit PID followed by its bitwise complement. This redundancy helps detect errors. (Note also that a PID byte contains at most four consecutive 1 bits, and thus will never need bit-stuffing, even when combined with the final 1 bit in the sync byte. However, trailing 1 bits in the PID may require bit-stuffing within the first few bits of the payload.)
Packets come in three basic types, each with a different format and CRC (cyclic redundancy check):

Handshake packets

Handshake packets consist of a PID byte, and are generally sent in response to data packets. The three basic types are *ACK*, indicating that data was successfully received, *NAK*, indicating that the data cannot be received and should be retried, and *STALL*, indicating that the device has an error condition and will never be able to successfully transfer data until some corrective action (such as device initialization) is performed.

Token packets

Token packets consist of a PID byte followed by 2 payload bytes: 11 bits of address and a 5-bit CRC. Tokens are only sent by the host, never a device.
IN and *OUT* tokens contain a 7-bit device number and 4-bit function number (for multifunction devices) and command the device to transmit DATAx packets, or receive the following DATAx packets, respectively.
USB 2.0 added a *PING* token, which asks a device if it is ready to receive an OUT/DATA packet pair. The device responds with ACK, NAK, or STALL, as appropriate. This avoids the need to send the DATA packet if the device knows that it will just respond with NAK.
USB 2.0 also added a larger 3-byte *SPLIT* token with a 7-bit hub number, 12 bits of control flags, and a 5-bit CRC. This is used to perform split transactions. Rather than tie up the high-bandwidth USB bus sending data to a slower USB device, the nearest high-bandwidth capable hub receives a SPLIT token followed by one or two USB packets at high bandwidth, performs the data transfer at full or low bandwidth, and provides the response at high bandwidth when prompted by a second SPLIT token.

Data packets

A data packet consists of the PID followed by 0–1,023 bytes of data payload (up to 1,024 in high bandwidth, at most 8 at low bandwidth), and a 16-bit CRC.
There are two basic data packets, *DATA0* and *DATA1*. They must always be preceded by an address token, and are usually followed by a handshake token from the receiver back to the transmitter. The two packet types provide the 1-bit sequence number required by Stop-and-wait ARQ. If a USB host does not receive a response (such as an ACK) for data it has transmitted, it does not know if the data was received or not; the data might have been lost in transit, or it might have been received but the handshake response was lost.
When a device is reset with a SETUP packet, it expects an 8-byte DATA0 packet next.
USB 2.0 added *DATA2* and *MDATA* packet types as well. They are used only by high-bandwidth devices doing high-bandwidth isochronous transfers which need to transfer more than 1024 bytes per 125 µs microframe (8,192 kB/s).

PRE packet

Low-bandwidth devices are supported with a special PID value, *PRE*. This marks the beginning of a low-bandwidth packet, and is used by hubs which normally do not send full-bandwidth packets to low-bandwidth devices. Since all PID bytes include four 0 bits, they leave the bus in the full-bandwidth K state, which is the same as the low-bandwidth J state. It is followed by a brief pause during which hubs enable their low-bandwidth outputs, already idling in the J state, then a low-bandwidth packet follows, beginning with a sync sequence and PID byte, and ending with a brief period of SE0. Full-bandwidth devices other than hubs can simply ignore the PRE packet and its low-bandwidth contents, until the final SE0 indicates that a new packet follows.

INTRODUCTION TO PENTIUM AND DUAL CORE PROCESSORS

DUAL CORE PROCESSORS

A multi-core processor is a single computing component with two or more independent actual processors (called "cores"), which are the units that read and execute program instructions.[1] The instructions are ordinary CPU instructions such as add, move data, and branch, but the multiple cores can run multiple instructions at the same time, increasing overall speed for programs amenable to parallel. Manufacturers typically integrate the cores onto a single integrated circuit die (known as a chip multiprocessor or CMP), or onto multiple dies in a single chip package.

In which the number of cores is large enough that traditional multi-processor techniques are no longer efficient— largely because of issues with congestion in supplying instructions and data to the many processors. Processors were originally developed with only one core. A **many-core** processor is a multi-core processor The many-core threshold is roughly in the range of several tens of cores; above this threshold network on chip technology is advantageous. Tilera processors feature a switch in each core to route data through an on-chip mesh network to lessen the data congestion, enabling their core count to scale up to 100 cores.

A **dual-core processor** has two cores (e.g. AMD Phenom II X2, Intel Core Duo), a **quad-core processor** contains four cores (e.g. AMD Phenom II X4, Intel's quad-core processors, see i3, i5, and i7at Intel Core), a **hexa-core processor** contains six cores (e.g. AMD Phenom II X6, Intel Core i7 Extreme Edition 980X), an **octa-core processor** contains eight cores (e.g. Intel Xeon E7-2820, AMD FX-8150) A multi-core processor implements multiprocessing in a single physical package. Designers may couple cores in a multi-core device tightly or loosely. For example, cores may or may not share caches, and they may implement message passing or shared memory inter-core communication methods. Commonnetwork topologies to interconnect cores include bus, ring, two-dimensional mesh, and crossbar.*Homogeneous* multi-core systems include only identical cores, heterogeneous multi-core systems have cores that are not identical. Just as with single-processor systems, cores in multi-core systems may implement architectures such as superscalar, VLIW, vector processing, SIMD, or multithreading.

Multi-core processors are widely used across many application domains including general-purpose,embedded, network, digital signal processing (DSP), and graphics.

Terminology

The terms *multi-core* and *dual-core* most commonly refer to some sort of central processing unit (CPU), but are sometimes also applied todigital signal processors (DSP) and system-on-a-chip (SoC circuit, unless otherwise noted.). The terms are generally used only to refer to multi-core microprocessors that are manufactured on the *same* integrated circuit die; separate microprocessor dies in the same package are generally referred to by another name, such as *multi-chip module*. This article uses the terms "multi-core" and "dual-core" for CPUs manufactured on the *same* integrated circuits.

Commercial incentives

Several business motives drive the development of dual-core architectures. For decades, it was possible to improve performance of a CPU by shrinking the area of the integrated circuit, which drove down the cost per device on the IC. Alternatively, for the same circuit area, more transistors could be utilized in the design, which increased functionality, especially for CISC architectures. Clock rates also increased by orders of magnitude in the decades of the late 20th century, from several megahertz in the 1980s to several gigahertz in the early 2000s.

As the rate of clock speed improvements slowed, increased use of parallel computing in the form of multi-core processors has been pursued to improve overall processing performance. Multiple cores were used on the same CPU chip, which could then lead to better sales of CPU chips with two or more cores. Intel has produced a 48-core processor for research in cloud computing

Technical factors

Since computer manufacturers have long implemented symmetric multiprocessing (SMP) designs using discrete CPUs, the issues regarding implementing multi-core processor architecture and supporting it with software are well known.

Additionally:

Utilizing a proven processing-core design without architectural changes reduces design risk significantly.

- For general-purpose processors, much of the motivation for multi-core processors comes from greatly diminished gains in processor performance from increasing the operating frequency. This is due to three primary factors:
    1. The *memory wall*; the increasing gap between processor and memory speeds. This effect pushes cache sizes larger in order to mask the latency of memory. This helps only to the extent that memory bandwidth is not the bottleneck in performance.
    2. The *ILP wall*; the increasing difficulty of finding enough parallelism in a single instructions stream to keep a high-performance single-core processor busy.
    3. The *power wall*; the trend of consuming exponentially increasing power with each factorial increase of operating frequency. This increase can be mitigated by "shrinking" the processor by using smaller traces for the same logic. The *power wall* poses manufacturing, system design and deployment problems that have not been justified in the face of the diminished gains in performance due to the *memory wall* and *ILP wall*.

In order to continue delivering regular performance improvements for general-purpose processors, manufacturers such as Intel and AMD have turned to multi-core designs, sacrificing lower manufacturing-costs for higher performance in some applications and systems. Multi-core architectures are being developed, but so are the alternatives. An especially strong contender for established markets is the further integration of peripheral functions into the chip.

The **Pentium Dual-Core** brand was used for mainstream x86 architecture microprocessors fromIntel from 2006 to 2009 when it was renamed to Pentium. The processors are based on either the 32-bit *Yonah* or (with quite different microarchitectures) 64-bit *Merom-2M*, *Allendale*, and*Wolfdale-3M* core, targeted at mobile or desktop computers.

In terms of features, price and performance at a given clock frequency, Pentium Dual-Core processors were positioned above Celeron but below Core and Core 2 microprocessors in Intel's product range. The Pentium Dual-Core was also a very popular choice for overclocking, as it can deliver high performance (when overclocked) at a low price.

Processor cores

In 2006, Intel announced a plan[1] to return the Pentium trademark from retirement to the market, as a moniker of low-cost Core microarchitecture processors based on the single-core Conroe-L but with 1 MiB of cache. The identification numbers for those planned Pentiums were similar to the numbers of the latter Pentium Dual-Core microprocessors, but with the first digit "1", instead of "2", suggesting their single-core functionality. A single-core Conroe-L with 1 MiB cache was deemed as not strong enough to distinguish the planned Pentiums from the Celerons, so it was replaced by dual-core CPUs, adding "Dual-Core" to the line's name. Throughout 2009, Intel changed the name back from Pentium Dual-Core to Pentium in its publications. Some processors were sold under both names, but the newer E5400 through E6800 desktop and SU4100/T4x00 mobile processors were not officially part of the Pentium Dual-Core line.

Comparison to the Pentium D

Although using the *Pentium* name, the desktop Pentium Dual-Core is based on the Core microarchitecture, which can clearly be seen when comparing the specification to the Pentium D, which is based on the NetBurst microarchitecture first introduced in the Pentium 4. Below the 2 or 4 MiB of shared-L2-cache-enabled Core 2 Duo, the desktop Pentium Dual-Core has 1 or 2 MiB of shared L2 Cache. In contrast, the Pentium D processors have either 2 or 4 MiB of non-shared L2 cache. Additionally, the fastest-clocked Pentium D has a factory boundary of 3.73 GHz, while the fastest-clocked desktop Pentium Dual-Core reaches 3.2 GHz. A major difference among these processors is that the desktop Pentium Dual Core processors have a TDP of only 65 W while the Pentium D ranges between 95 to 130 W. Despite the reduced clock speed, and lower amounts of cache, Pentium dual-core outperfomed Pentium D by a fairly large margin.

Advantages

The proximity of multiple CPU cores on the same die allows the cache coherency circuitry to operate at a much higher clock-rate than is possible if the signals have to travel off-chip. Combining equivalent CPUs on a single die significantly improves the performance of cache snoop (alternative: Bus snooping) operations. Put simply, this means that signals between different CPUs travel shorter distances, and therefore those signals degrade less. These higher-quality signals allow more data to be sent in a given time period, since individual signals can be shorter and do not need to be repeated as often.The largest boost in performance will likely be noticed in improved response-time while running CPU-intensive

processes, like antivirus scans, ripping/burning media (requiring file conversion), or file searching. For example, if the automatic virus-scan runs while a movie is being watched, the application running the movie is far less likely to be starved of processor power, as the antivirus program will be assigned to a different processor core than the one running the movie playback.

Assuming that the die can fit into the package, physically, the multi-core CPU designs require much less printed circuit board (PCB) space than do multi-chip SMP designs. Also, a dual-core processor uses slightly less power than two coupled single-core processors, principally because of the decreased power required to drive signals external to the chip. Furthermore, the cores share some circuitry, like the L2 cache and the interface to the front side bus (FSB). In terms of competing technologies for the available silicon die area, multi-core design can make use of proven CPU core library designs and produce a product with lower risk of design error than devising a new wider core-design. Also, adding more cache suffers from diminishing returns.[*citation needed*]

Multi-core chips also allow higher performance at lower energy. This can be a big factor in mobile devices that operate on batteries. Since each core in multi-core is generally more energy-efficient, the chip becomes more efficient than having a single large monolithic core. This allows higher performance with less energy. The challenge of writing parallel code clearly offsets this benefit.[4]

Disadvantages

Maximizing the utilization of the computing resources provided by multi-core processors requires adjustments both to the operating system(OS) support and to existing application software. Also, the ability of multi-core processors to increase application performance depends on the use of multiple threads within applications. The situation is improving: for example the Valve Corporation's Source engine offers multi-core support,[5][6] and Crytek has developed similar technologies for CryEngine 2, which powers their game, *Crysis*. Emergent Game Technologies' Gamebryo engine includes their Floodgate technology,[7] which simplifies multicore development across game platforms. In addition, Apple Inc.'s second latest OS, Mac OS X Snow Leopard has a built-in multi-core facility called Grand Central Dispatch for Intel CPUs.

Integration of a multi-core chip drives chip production yields down and they are more difficult to manage thermally than lower-density single-chip designs. Intel has partially countered this first problem by creating its quad-core designs by combining two dual-core on a single die with a unified cache, hence any two working dual-core dies can be used, as opposed to producing four cores on a single die and requiring all four to work to produce a quad-core. From an architectural point of view, ultimately, single CPU designs may make better use of the silicon surface area than multiprocessing cores, so a development commitment to this architecture may carry the risk of obsolescence. Finally, raw processing power is not the only constraint on system performance. Two processing cores sharing the same system bus and memory bandwidth limits the real-world performance advantage. If a single core is close to being memory-bandwidth limited, going to dual-core might only give 30% to 70% improvement. If memory bandwidth is not a problem, a 90% improvement can be expected[*citation needed*]. It would be possible for an application that used two CPUs to end up

running faster on one dual-core if communication between the CPUs was the limiting factor, which would count as more than 100% improvement.

Hardware

Trends

The general trend in processor development has moved from dual-, tri-, quad-, hexa-, octo-core chips to ones with tens or even hundreds of cores. In addition, multi-core chips mixed with simultaneous multithreading, memory-on-chip, and special-purpose "heterogeneous" cores promise further performance and efficiency gains, especially in processing multimedia, recognition and networking applications. There is also a trend of improving energy-efficiency by focusing on performance-per-watt with advanced fine-grain or ultra fine-grain power management and dynamic voltage and frequency scaling (i.e. laptop computers and portable media players).

Architecture

The composition and balance of the cores in multi-core architecture show great variety. Some architectures use one core design repeated consistently ("homogeneous"), while others use a mixture of different cores, each optimized for a different, "heterogeneous" role.

The article "CPU designers debate multi-core future" by Rick Merritt, EE Times 2008,[8] includes these comments:

Chuck Moore suggested computers should be more like cellphones, using a variety of specialty cores to run modular software scheduled by a high-level applications programming interface.

Atsushi Hasegawa, a senior chief engineer at Renesas, generally agreed. He suggested the cellphone's use of many specialty cores working in concert is a good model for future multi-core designs.

Anant Agarwal, founder and chief executive of startup Tilera, took the opposing view. He said multi-core chips need to be homogeneous collections of general-purpose cores to keep the software model simple.

Managing concurrency acquires a central role in developing parallel applications. The basic steps in designing parallel applications are:

Partitioning

The partitioning stage of a design is intended to expose opportunities for parallel execution. Hence, the focus is on defining a large number of small tasks in order to yield what is termed a fine-grained decomposition of a problem.

Communication

The tasks generated by a partition are intended to execute concurrently but cannot, in general, execute independently. The computation to be performed in one task will typically require data associated with another task. Data must then be transferred between tasks so as to allow computation to proceed. This information flow is specified in the communication phase of a design.

Agglomeration

In the third stage, development moves from the abstract toward the concrete. Developers revisit decisions made in the partitioning and communication phases with a view to obtaining an algorithm that will execute efficiently on some class of parallel computer. In particular, developers consider whether it is useful to combine, or agglomerate, tasks identified by the partitioning phase, so as to provide a smaller number of tasks, each of greater size. They also determine whether it is worthwhile to replicate data and/or computation.

Mapping

In the fourth and final stage of the design of parallel algorithms, the developers specify where each task is to execute. This mapping problem does not arise on uniprocessors or on shared-memory computers that provide automatic task scheduling.

On the other hand, on the server side, multicore processors are ideal because they allow many users to connect to a site simultaneously and have independent threads of execution. This allows for Web servers and application servers that have much better throughput.

PENTIUM PROCESSORS

Pentium is a brand used for a series of x86-compatible microprocessors produced by Intel. In its most current form, a Pentium processor is a consumer-level product with a two-star rating[1], above the low-end Atom and Celeron products but below the faster Core i3, i5 and i7 lines as well as the high-end Xeon processors.

The name Pentium is originally derived from the Greek word *pente* (πέντε), meaning 'five' (as the series was Intel's 5th generation microarchitecture, the P5), and the Latin ending *-ium*. The current Pentium processors only share the name but are in fact based on the same processor chips that are used in the Intel Core but are typically used with a lower clock frequency, a partially disabled L3 cache and some of the advanced features such as Hyper-threading andVirtualization disabled.

Overview

During development Intel generally identifies processors with codenames, such as *Prescott*,*Willamette*, *Coppermine*, *Katmai*, *Klamath* or *Deschutes*. These usually become widely known,] even after the processors are given official names on launch.

History

The original Pentium branded CPUs were expected to be named 586 or i586, to follow the naming convention of previous generations (286,i386, i486). However, as the company wanted to prevent their competitors from branding their processors with similar names, as AMD had done with their Am486, Intel attempted to file a trademark on the name in the United States, only to be denied because a series of numbers was not considered distinct.

Following Intel's previous series of 8086, 80186, 80286, 80386, and 80486 microprocessors, the company's first P5-based processor was released as the original Intel Pentium on March 22, 1993. Due to its success, the Pentium brand would continue through several generations of high-end processors beyond the original. In 2006, the Pentium brand briefly disappeared from Intel's roadmaps,[3][4] only to re-emerge in 2007.[5]

In 1998, Intel introduced the Celeron[6] brand for low-priced microprocessors. With the 2006 introduction of the Intel Core brand as the company's new flagship line of processors, the Pentium series was to be discontinued. However, due to a demand for mid-range dual-core processors, the Pentium brand was re-purposed to be Intel's mid-range processor series, in between the Celeron and Core series, continuing with the Pentium Dual-Core line.[7] [8][9]

In 2009, the "Dual-Core" suffix was dropped, and new x86 microprocessors started carrying the plain *Pentium* name again.

Pentium-branded processors

P5 microarchitecture based

The original **Pentium** and **Pentium MMX** processors were the superscalar follow-on to the 80486 processor and were marketed from 1993 to 1999. Some versions of these were available as Pentium OverDrive that would fit into older CPU sockets.

Pentium

P6 microarchitecture based

In parallel with the P5 microarchitecture, Intel developed the **P6 microarchitecture** and started marketing it as the **Pentium Pro** for the high-end market in 1995. It introduced out-of-order execution and an integrated second level cache on dual-chip processor package. The second P6 generation replaced the original P5 with the **Pentium II** and rebranded the high-end version as **Pentium II Xeon**. It was followed by a third version called the **Pentium III** and **Pentium III Xeon**, respectively. The Pentium II line added the MMX instructions that were also present in the Pentium MMX.

Versions of these processors for the Laptop market were initially called **Mobile Pentium II** and **Mobile Pentium III**, later versions were called **Pentium III-M**. Starting with the Pentium II, the Celeron brand was used for low-end versions of most Pentium processors with a reduced feature set such as a smaller cache or missing power management features.

Pentium Pro

Pentium II

Pentium III

Netburst microarchitecture based

In 2000, Intel introduced a new microarchitecture called **NetBurst**, with a much longer pipeline enabling higher clock frequencies than the P6 based processors. Initially, these were called **Pentium 4** and the high-end versions have since been called just Xeon. As with Pentium III, there are both **Mobile Pentium 4** and **Pentium 4 M** processors for the laptop market, with Pentium 4 M denoting the more power-efficient versions. Enthusiasts version of the Pentium 4 with the highest clock frequency were called **Pentium 4 Extreme Edition**.

The **Pentium D** was the first multi-core Pentium, integrating two Pentium 4 chips in one package and was also available as the enthusiast**Pentium Extreme Edition**.

Pentium 4

Pentium D

[**Pentium M microarchitecture based**

In 2003, Intel introduced a new processor based on the P6 microarchitecture called **Pentium M**, which was much more power efficient than the Mobile Pentium 4, Pentium 4 M and

Pentium III M. Dual-core version of the Pentium M was developed under the code name Yonah and sold under the marketing names Core Duo and **Pentium Dual-Core**. Unlike Pentium D, it integrated both cores on a single chip. From this point, the Intel Core brand name was used for the mainstream Intel processors and the Pentium brand became a low-end version between Celeron and Core. All Pentium M based designs including Yonah are for the mobile market.

Pentium M

Pentium Dual-Core

Core microarchitecture based

The **Pentium Dual-Core** name continued to be used when the Yonah design was extended with 64 bit support, now called the **Core microarchitecture**. This microarchitecture eventually replaced all NetBurst based processors across the four brands, Celeron, Pentium, Core and Xeon. Pentium Dual-Core processors based on the Core microarchitecture use the Allendale and Wolfdale-3M designs for desktop processors and Merom-2M for mobile processors.

Pentium Dual-Core

Pentium (2009)

In 2009, Intel changed the naming system for Pentium processors, renaming the Wolfdale-3M based processors to **Pentium**, without the Dual-Core name and introduced new single- and dual-core processors based on Penryn under the Pentium name.

The Penryn core is the successor to the Merom core and Intel's 45 nm version of their mobile series of Pentium microprocessors. The FSB is increased from 667 MHz to 800 MHz and the voltage is lowered. Intel released the first Penryn Core, the Pentium T4200, in December, 2008. In June 2009, Intel released the first single-core processor to use the Pentium name, a Consumer Ultra-Low Voltage (CULV) Penryn core called the Pentium SU2700.

In September 2009, Intel introduced the Pentium SU4000 series together with the Celeron SU2000 and Core 2 Duo SU7000 series, which are dual-core CULV processors based on Penryn-3M and using 800 MHz FSB. The Pentium SU4000 series has 2 MB L2 cache but is otherwise basically identical to the other two lines.

Nehalem microarchitecture based

The Nehalem microarchitecture was introduced in late 2008 as a successor to the Core microarchitecture, and in early 2010, a new **Pentium**G6950 processor based on the **Clarkdale** design was introduced based on the **Westmere** refresh of Nehalem, which were followed by the mobile P6xxx based on Arrandale a few months later.

On January 7, 2010, Intel launched a new Pentium model using the Clarkdale chip in parallel with other desktop and mobile CPUs based on their new Westmere microarchitecture. The first model in this series is the Pentium G6950. The Clarkdale chip is also used in the Core i3-5xx and Core i5-6xx series and features a 32 nm process (as it is based on the Westmere microarchitecture), integrated memory controller and 45 nm graphics controller and a third-level cache. In the Pentium series, some features of Clarkdale are disabled. Compared to Core i3, it lacks Hyper-Threading and the graphics controller in the Pentium runs at 533 MHz, while in the Core i3 i3-5xx series they run at 733 MHz. Dual Video Decode that enables Blu-ray picture-in picture hardware acceleration is disabled as well as Deep Color and xvYCCsupport]. The memory controller in the Pentium supports DDR3-1066 max same

as the Core i3 i3-5xx series (ref:http://ark.intel.com/products/43529). The L3 cache is also 1 MB less than in the Core i3-5xx series.

Sandy Bridge microarchitecture based

The Sandy Bridge microarchitecture was released in the Pentium line on May 22, 2011.

- [a]All models share the following details: 2 cores, 2 logical processors (4 on Pentium 3xx with Hyper-threading), CPUID signature 206A7, family 6 (06h), model 42 (02Ah), stepping 7 (07h)
- [b]TLB / cache 64-byte Prefetching; Data TLB0 2-MB or 4-MB pages, 4-way associative, 32 entries; Data TLB 4-KB Pages, 4-way set associative, 64 entries; Instruction TLB 4-KB Pages, 4-way set associative, 128 entries, L2 TLB 1-MB, 4-way set associative, 64-byte line size; Shared 2nd-level TLB 4 KB pages, 4-way set associative, 512 entries.
- [c]All models feature: On-chip Floating Point Unit, Enhanced Intel SpeedStep Technology (EIST), Intel 64, XD bit (an NX bitimplementation), Intel VT-x, Smart Cache.
- [d]All models support: *MMX*, *SSE*, *SSE2*, *SSE3*, *SSSE3*, *SSE4.1*, *SSE4.2*
- [e]HD Graphics (Sandy Bridge) contain 6 EUs as well as HD Graphics 2000, but does not support the following technologies: Intel Quick Sync Video, InTru 3D, Clear Video HD, Wireless Display, and it doesn't support 3D Video or 3D graphics acceleration.

Pentium compatible Intel processors

Due to its prominence, the term "Pentium compatible" is often used to describe any x86 processor that supports the IA-32 instruction set and architecture. Even though they do not use the Pentium name, Intel also manufacturers other processors based on the Pentium series for other markets. Most of these processors share the core design with one of the Pentium processor lines, usually differing in the amount ofCPU cache, power efficiency or other features. The notable exception is the Atom line, which is an independent design.

- Celeron, a low-end version
- Core, the mainstream version including Core 2 and Core i7, now placed above Pentium
- Xeon, a high-end version used in servers and workstations
- A100 (discontinued), an ultra-mobile version of Pentium M
- EP80579, A system-on-a-chip based on Pentium M
- Atom, current ultra-mobile processors

PIC MICROCONTROLLER

INTRODUCTION:

PIC microcontrollers are popular processors developed by Microchip Technology with built-in RAM, memory, internal bus, and peripherals that can be used for many applications. PIC originally stood for "Programmable Intelligent Computer" but is now generally regarded as a "Peripheral Interface Controller".

PIC microcontrollers can be programmed in Assembly, C or a combination of the two. Other high-level programming languages can be used but embedded systems software is primarily written in C.

TYPES OF PIC'S:

PIC microcontrollers are broken up into two major categories: 8-bit microcontrollers and 16-bit microcontrollers. Each category is further subdivided into product families as shown in the

following table:

8-bit MCU Product Family	16-bit MCU Product Family
PIC10 PIC12 PIC14 PIC16 PIC18	PIC24F PIC24H dsPIC30 dsPIC33

The microcontrollers in the PIC10 through PIC14 families are considered low-end microcontrollers. PIC microcontrollers in the PIC16 and PIC18 families are considered mid-level microcontrollers while 16-bit PICs are considered high-end microcontrollers.

FEATURES:

- Family includes controllers from 16c61/62/64/71/74/710/715 etc.
- They are RISC processors and uses Harvard architecture.
- Different bus widths of data and program memory.
- Data memory is 8 bit wide where as program memory is 12,14,16 bits wide.
- The instruction holds immediate data along with instruction code.
- Only 35 instructions.
- Most instructions take 0.2 microseconds to execute when operated at 20 MHz.
- Machine cycle consist of 4 clock pulses.
- Instruction set is highly orthogonal.
- 1-3 Timers with 8/16 bit prescalar.
- Watch Dog timer (WDT)
- 13-33 I/O pins.

- 3-12 interrupt Sources.
- 4/8 Channel, 8 bit on chip ADC.
- Power on Reset. (POR)
- Brown out Reset (BOR).
- Capture/Compare/ PWM modules.
- USART
- Synchronous serial port (SSP) with SPI and I2C.
- Power saving SLEEP mode.
- Wide operating Voltage range 2.5 V to 6.0 V.Very Low power consumption.
- Commercial,Industrial and Extended Temperature ranges.
- Parallel slave port (PSP),8 bits wide with external RD,WR and CS controls.

PICs also come in several types of packages:

Plastic Dual Inline Package (PDIP),Small-Outline Transistor (SOT),Dual Flat No-lead (DFN),Mini Small Outline Package (MSOP),Thin Quad Flat Pack (TQFP),Plastic Leaded Chip Carrier (PLCC),CERamic QUADpack (CERQUAD),

The reason for the number of packages is that there are some PICs with 100 I/O pins! The microcontrollers are basically rectangular or square shaped. The easiest package to work with is DIP or PDIP because it is easily bread boardable and can easily be soldered.Writing your code completely in C because it is much faster and easier than writing your code in Assembly or a combination of languages.

PIC ARCHITECTURE:

ALU

Size is 8 bit Performs operations with temporary working register and (W register) and any register file. W register 8 bit wide. It contains one of source operands during execution of instruction and may serve as the destination for the result of operation. Used only for ALU operations.

STATUS Register

7	6	5	4	3	2	1	0
0	0	RP0	T0	PD	Z	DC	C

C = Carry Bit

DC = Digits Carry (Same as AC)

Z = Zero

TO = Reset status bit (Time out bit)

PD = Reset status bit (Power down)

These bits are used along with SLEEP mode. After coming out from SLEEP processor checks these bits to determine which kind of event is responsible for bringing out of SLEEP mode.RP0 = Register Bank Select bit.If 0 selects bank 0 otherwise bank 1.

FSR Register (File Selection Register):

FSR is a pointer used for indirect memory addressing in the whole register file. In indirect addressing mode one has to write address byte in FSR and then use INDF (Indirect thro FSR).INDF is used in instruction.

PCLATH (PC Latch):

Can be independently read or written like any other register.It's different from PC and is separate entity. It is 5 bits.This is added with PCL (Program Counter Lower) so as to get 13 bit address.

PROGRAM MEMORY

- PIC 16c6X/7X is 2K or 4K.
- 11 bit or 12 bit address is used out of
- 13 bits in PC.
- Maximum memory that can be accessed is 8K.
- After reset program counter is cleared.
- At 0000h there is "goto Mainline" Instruction which takes PC to 0005h.

DATA MEMORY:

Register File Structure:They are the memory locations that are addressed Be instruction.There is general purpose and special purpose register file.General purpose are 8 bit RAM locations and special purpose are I/O ports and control registers.

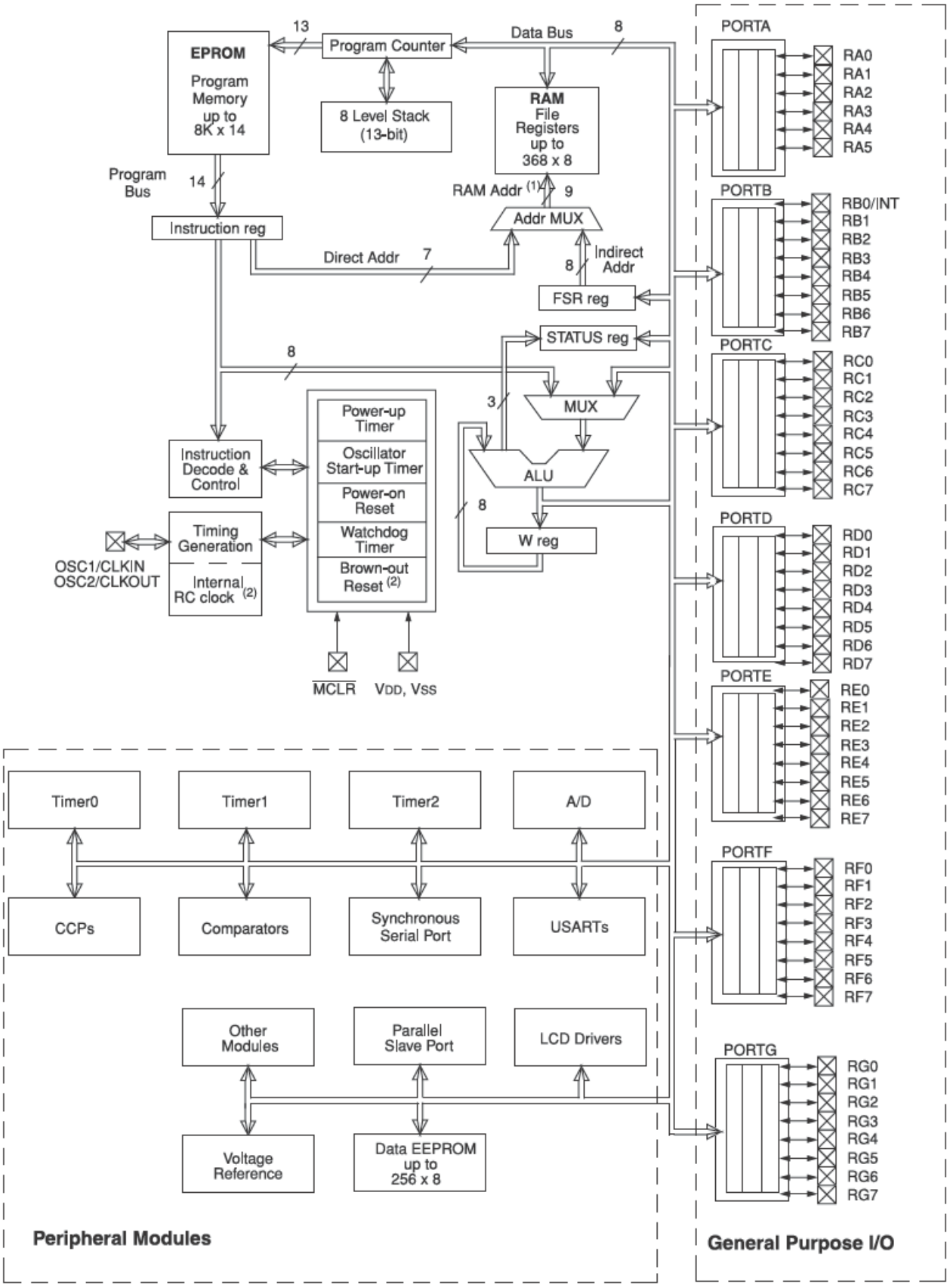
ADDRESSING MODES:

Direct Addressing:

It uses 7 bits of instruction and the 8 bit from RP0.If bit is 0 then bank otherwise bank 1.

Indirect addressing

In this mode the 8 bit address of the location in register file to be accessed is written in FSR and use INDF.


EPROM
Program Memory up to 8K x 14
13
Program Counter
Data Bus
8
8 Level Stack (13-bit)
RAM File Registers up to 368 x 8
Program Bus
14
RAM Addr (1)
9
Addr MUX
Instruction reg
Direct Addr
7
8
Indirect Addr
FSR reg
STATUS reg
8
3
MUX
Power-up Timer
Oscillator Start-up Timer
Power-on Reset
Watchdog Timer
Brown-out Reset (2)
Instruction Decode & Control
ALU
8
W reg
Timing Generation
Internal RC clock (2)
OSC1/CLKIN
OSC2/CLKOUT
MCLR
VDD, VSS
PORTA
RA0
RA1
RA2
RA3
RA4
RA5
PORTB
RB0/INT
RB1
RB2
RB3
RB4
RB5
RB6
RB7
PORTC
RC0
RC1
RC2
RC3
RC4
RC5
RC6
RC7
PORTD
RD0
RD1
RD2
RD3
RD4
RD5
RD6
RD7
PORTE
RE0
RE1
RE2
RE3
RE4
RE5
RE6
RE7
PORTF
RF0
RF1
RF2
RF3
RF4
RF5
RF6
RF7
PORTG
RG0
RG1
RG2
RG3
RG4
RG5
RG6
RG7
Timer0
Timer1
Timer2
A/D
CCPs
Comparators
Synchronous Serial Port
USARTs
Other Modules
Parallel Slave Port
LCD Drivers
Voltage Reference
Data EEPROM up to 256 x 8
Peripheral Modules
General Purpose I/O

I/O PORTS:

Port A

RA0 to RA4 (5 lines)(Address 05) RA4 has alternate function. TRISA(85H) is SFR used to configure these lines individually as either inputs or outputs.Setting bit in TRIS will configure as input and 0 will configure as output.

Port B

RB0 to RB7(8 lines).TRISB It has weak internal pull up which is to be enabled.POR disables pull ups.

INTERRUPTS:

3 Interrupt Sources for 16C61.

External Interrupt –Due to external source.Edge Sensitive RB0/INT causes this interrupt.This interrupt wakes up processor from SLEEP.This must be set before going into SLEEP mode.

Timer 0 –Timer 0 overflow. FF to 00 overflow.

Port B Change Interrupt – A change from high to low or low to high on port B pins RB4 to RB7 causes this interrupt.This interrupt can wake device from SLEEP.

ADC – For 16C71 series as EOC.

TIMERS:

- 8 Bit wide.
- Clocked internally by system clock which is Fosc/4 or by external clock on RA4/TOCKI.(Frequency 0 – 50 MHz)
- Incrementing. Overflows from FF-00 and generates interrupt.
- 2 cycle delay after prescaler for purpose of synchronization of external with internal clock.
- Timer0 delay= { [Timer0 count] x Prescaler Value x4/Fosc}
- Timer0 preload = 256 – [Timer 0 delay x Fosc]/Prescale value x 4]

16. University previous Question papers

Code No: 07A6EC02 R07

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

B. Tech III Year II Semester Examinations, May – 2013

Microprocessors and Microcontrollers

(Electrical and Electronics Engineering)

Time: 3 hours **Max. Marks: 80**

Answer any five questions

All questions carry equal marks

* * *

1. What is meant by an addressing mode? Explain the different addressing modes supported by 8086 with suitable examples. [16]

2.a) Write an 8086 assembly language program to arrange a given series of hexadecimal bytes in ascending order.

b) Explain the following assembler directives:
i) SEGMENT, ii) ORG,
iii) OFFSET, iv) DW [8+8]

3.a) Explain the control word format of 8255 in I/O and BSR mode.

b) Explain the A/D converter interface to 8086 microprocessor. [8+8]

4.a) With the help of the internal block diagram, explain the working of 8259 priority interrupt controller.

b) Explain the various hardware and software interrupts in 8086 microprocessor. [8+8]

5.a) Give an overview of RS-232C serial data standard.

b) Write short notes on IEEE-488 standard. [8+8]

6. Draw and explain the internal architecture of 8051 microcontroller. [16]

7.a) Explain the format and bit definitions of the following SFRs in 8051:
i) TMOD, ii) SCON

b) Explain the interrupt structure of 8051. [8+8]

8. List out the various human and keyboard factors that effect the keyboard application programs. Discuss how a lead per key keyboard is interfaced to the 8051 microcontroller with the help of a diagram and explain the "Getkey" routine for scanning. [16]

Code No: 09A60204 | R09 | SET-1

B. Tech III Year II Semester Examinations, April/May - 2012
MICROPROCESSORS AND MICROCONTROLLERS
(Common to EEE, ECE, EIE, ETM, ECM, ICE)

Time: 3 hours **Max. Marks: 75**

Answer any five questions
All questions carry equal marks

---

1. Explain different registers used in 8086 and its memory segmentation. What are the registers used to access memory. [15]

2. Explain different addressing modes in 8086. [15]

3. Show 8255 PPI in mode1 operation and interface to 8086. [15]

4. Explain the interrupt structure of 8086. [15]

5. Explain the need for RS232C interface. Explain serial communication standards with respect to voltage levels. [15]

6. What is the difference between microprocessor and microcontroller? Give 8051 architecture. [15]

7. Explain how the interrupts are used in real-time. [15]

8. Explain standard AVR architecture. [15]

--oo0oo--

Code No: 09A60204	R09	SET-2

B. Tech III Year II Semester Examinations, April/May - 2012

MICROPROCESSORS AND MICROCONTROLLERS

(Common to EEE, ECE, EIE, ETM, ECM, ICE)

Time: 3 hours **Max. Marks: 75**

Answer any five questions

All questions carry equal marks

1.	Explain memory mapping techniques in 8086.	[15]
2.	Explain instruction format for 8086.	[15]
3.	Interface 8086 to keyboard and display unit.	[15]
4.	Explain interrupt service routine concept.	[15]
5.	Explain 8251 USART.	[15]
6.	Explain the timers in 8051 and its modes.	[15]
7.	Explain the timers and counters in 8051.	[15]
8.	Explain memory organization in AVR.	[15]

--ooOoo--

Code No: 09A60204	R09	SET-3

B. Tech III Year II Semester Examinations, April/May - 2012

MICROPROCESSORS AND MICROCONTROLLERS

(Common to EEE, ECE, EIE, ETM, ECM, ICE)

Time: 3 hours **Max. Marks: 75**

Answer any five questions

All questions carry equal marks

---

1. Explain interrupts of 8086 its minimum and maximum mode signals and common function signals. [15]

2. What are assembler directives and micros? Consider one example and show how they are used? [15]

3. Explain A/D and D/A conversion mechanism. [15]

4. Explain 8259 interrupt controller along with control registers. [15]

5. Explain the interfacing of RS-232 to 8086. [15]

6. Explain internal and external memory of 8051. [15]

7. Explain programming serial communication interrupts. [15]

8. Explain interrupt structure in AVR. [15]

--ooOoo--

Code No: 09A60204	R09	SET-4

B. Tech III Year II Semester Examinations, April/May - 2012

MICROPROCESSORS AND MICROCONTROLLERS

(Common to EEE, ECE, EIE, ETM, ECM, ICE)

Time: 3 hours **Max. Marks: 75**

Answer any five questions

All questions carry equal marks

---

1. Explain the architecture of 8086. [15]

2. Explain sorting technique with an example. [15]

3. Explain I/O mapped I/O and memory mapped I/O. [15]

4. Explain the interface of 8257 to 8086. [15]

5. Explain IEEE-488 bus and its protocols. [15]

6. Give any five instructions of 8051 and explain each. [15]

7. Explain special function registers in 8051 and show how they are used for programming. [15]

8. Explain register file of AVR. [15]

--ooOoo--

Code No: 09A60204 R09

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD

B. Tech III Year II Semester Examinations, May – 2013

Microprocessors and Microcontrollers

(Common to EEE, ECE, EIE, ETM, E.COMP.E, ICE)

Time: 3 hours **Max. Marks: 75**

Answer any five questions

All questions carry equal marks

- - -

1.a) With a neat block diagram explain the architecture of 8086 processor.

b) Explain the following instructions of 8086

i) TEST ii) CMP

iii) XLAT iv) CBW. [8+7]

2.a) What is the purpose of the Instruction Queue of BIU in 8086? Why is the size of the queue limited to 6 Bytes?

b) Explain addressing modes of 8086 with an example. [5+10]

3. An 8086-1 system with 8255 is interfaced at port A address FFF8H and Port C address at FFFEH. 8086-2 system with another 8255 is interfaced at port A address FFF8H and Port C address at FFFEH. Give necessary hardware and software for transferring 10 bytes of data in parallel from 8086-1 to another 8086-2. Assume that both systems run on the same clock rate. [15]

4.a) Explain how to interface Digital-to-Analog Converter (DAC) to 8086 processor. Give the hardware and software for it.

b) Explain type 3 interrupt in 8086. [12+3]

5.a) Draw and explain interrupt acknowledgement cycle of 8086.

b) Explain the mode word format and the command word format of 8251A. [5+10]

6.a) Assume that ROM space of 8051 starting at 250H contains "Hello", write a program to transfer the bytes into RAM locations starting at 40H.

b) Explain the stack operation in 8051 microcontroller and also discuss necessary instructions to access the stack. [9+6]

7.a) List out the steps involved in programming the 8051 to transfer data serially.

b) Write an 8051 program to find Y where $Y = x^2+5$ and x is between 0 and 5. [5+10]

8.a) Explain the salient features of AVR RISC controller with reference to its architecture.

b) Discuss the interrupt structure of RISC controller. [15]

III B. Tech II Semester Regular Examinations, Apr/May 2010

MICROPROCESSORS AND INTERFACING
(Common to ECE, BME, EIE)

Time: 3 Hours Max Marks: 80

Answer Any FIV Questions

All Questions Carry Equal Marks

1. a) Draw and explain the flag register of 8086 microprocessor.

   b) Explain the different logical instructions of 8086 microprocessor. [8+8]

2. Write an assembly language program in 8086 to arrange the given series of hexadecimal numbers in ascending order. [16]

3. With neat circuit diagram and timing diagrams, explain the minimum mode operation of 8086 microprocessor. [16]

4. Interface an 8255 with 8086 to work as an I/O port. Initialize port A as output port, port B as input port and port C as output port. Port A address should be 0740H.Write a program to sense switch positions SW_0 –SW_7 connected at port B. The sensed pattern is to be displayed in port A, to which 8 LEDs are connected, which the port C lower displays number of on switches out of the total eight switches. [16]

5. a) What is the difference between maskable and non-makeable interrupts? Give some examples?

   b) Discussabout the followingcontrol word formats of 8259:

   i. Initialization Command Words (ICWs).

   ii. Operational Command Words (OCWs) [6+10]

6. a) Discuss the types of serial communication.

   b) With a neat circuit diagram, explain the interfacing of 8251 with 8086. [6+10]

7. a) What is meant by paging? Explain its advantages and disadvantages.

   b) Explain the procedure of converting linear address into physical address. [8+8]

8. a)Discuss the advantages of microcontroller based systems over microprocessor based systems.

b) Draw and discuss the formats and bit definitions of the following SFRs

i. PCON ii. TCON iii. TMOD [4+12]

III B. Tech II Semester Regular Examinations, Apr/May 2010

MICROPROCESSORS AND INTERFACING
(Common to ECE, BME, EIE)

Time: 3 Hours Max Marks: 80

Answer Any FIVE Questions

All Questions Carry Equal Marks

1. Explain different addressing modes of 8086 microprocessors with suitable examples. [16]
2. a) Write an assembly language program in 8086 for the addition of a series of 8-bit numbers.

   b) Write an assembly language program to display a message “J N T U KAKINADA” on the CRT stream of a micro computer. [8+8]
3. Draw the functional pin diagram of 8086 and explain the functions of different pins. [16]
4. Interface a 4*4 keyboard with 8086 using 8255 and write an ALP for detecting a key closure and return the key code in AL. The debouncing period for a key in 10ms. Use software key bouncing technique. [16]
5. What is interrupt service routine? Draw the interrupt vector table and explaindifferentinterruptsavailablein8086.[16]
6. a) Discuss the asynchronous and synchronous data transfer schemes. b) Explain the interfacing of 8251 with 8086 with necessary circuit diagram.[8+8]

. a) Explain the salient features of the 80386 microprocessors.

b) Explain how the linear address is converted into physical address[8+8]

8. a) Discuss the register set of 8051 microcontroller.

b) Explain the interrupt structure of the 8051 microcontroller.[8+8]

III B. Tech II Semester Regular Examinations, Apr/May 2010

MICROPROCESSORS AND INTERFACING (Common to ECE, BME, EIE)

Time: 3 Hours Max Marks: 80

Answer Any FIVE Questions

All Questions Carry Equal Marks

1. List the register set of 8086 microprocessor and explain the functions of each of them.[16]
2. a) Write an ALP in 8086 for the addition of a series of 8-bit numbers.

   b) Write an ALP in 8086 to move a string of data words from offset 2000H to offset 3000H and the length of the string in 0FH. [8+8]
3. With neat circuit and timing diagrams, explain the maximum mode operation of 8086 microprocessor. [16]
4. Interface ADC 0808 with 8086 using 8255 ports. Use port A of 8255 for transferring digital data output of ADC to the CPU and port C for control signals. Assume that an analog input is present at I/P2 of the ADC and a clock input of suitable frequency in available for ADC. Draw the schematic and write required ALP. [16]
5. Draw and explain the architecture of 8259 programmable Interrupt Controller. [16]
6. a) Explain different modes data transmission.

   b) Draw and explain the internal architecture of 8251 USART[8+8]
7. a) Differentiate RISC and CISC processors

b) Explain the real mode and protected mode operations of 80386 microprocessor. [6+10]

8. a) Discuss various addressing modes of 8051.

b) Explain in detail about serial port operation in 8051 microcontroller. [8+8]

III B. Tech II Semester Regular Examinations, Apr/May 2010

MICROPROCESSORS AND INTERFACING
(Common to ECE, BME, EIE)

Time: 3 Hours Max Marks: 80

Answer Any FIVE Questions
All Questions Carry Equal Marks

1. a) Explain different data transfer instructions of 8086 microprocessor. b) Define and explain macros.[10+6]

2. a) Write an assembly language program in 8086 to find out the largest numbers from a given unordered array of 8-bit numbers, stored in the locations starting from a known address.

b) Write an assembly language program in 8086 to perform a one bit BCD addition. [8+8]

3. a) What is DMA? Explain the need for DMA.

b) With a neat block diagram, explain the working of 8257 DMA controller. [8+8]

4. Interface DAC AD7532 with an 8086 CPU resuming at 8MHz and write an assembly language program to generate a saw tooth waveform of period 1ms with Vmax 5V. [16]

5. a) Describe some important features of 8259 interrupt controller.

b) Distinguish between Master and Slave mode operation of 8259.[8+8]

6. Design a hardware interfacing circuit for interfacing 8251 with 8086. Set the 8251A in asynchronous mode as a transmitter and receiver with even

parity enabled, 2 stop bits, 8-bit character length, frequency 160 KHz and baud rate 10K. Write an ALP to transmit 100 bytes of data string starting at location 2000:5000H. [16]

7. a) Explain the salient features of the Pentium processor.

   b) Differentiate the paging and segmentation.[8+8]

8.Draw and discuss the formats and bit definitions of the following SFRs

   i. PCON

   ii. TCON

   iii. TMOD

   iv. SCON [4+4+4+4]

III B. Tech II Semester Regular Examinations, NOV/DEC 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) Draw the architectural diagram of 8085 and explain the function of each block in detail

   (b) Discuss about Multiplexing in 8086 microprocessor [10+6]

2. (a) Discuss about the following addressing modes of 8086 with some examples i. Directing addressing

   ii. Register addressing

   iii. Immediate addressing

   iv. Based addressing with displacement

(b) Explain why assembler directives are required? [8+8]

3. (a) Write a program in 8086 to add two 8-bytes of data available in memory location array1 and array 2. Store the result in array3

(b) Write an ALP to count number of 0s in a 16 bit binary string [10+6]

4. (a) With a neat pin diagram explain the minimum mode operation of 8086 (08)
(b) With a neat timing diagram explain how a READ operation is performed by 8086[8+8]

5. (a) Explain the following pins of 8255

. i. A0 and A1

ii. WR

iii. RESET

iv. PC0-PC7

(b) Explain the use of handshaking signals used in 8255 [8+8]

6. (a) How many Initialization Command words are required for a single 8259 In an 8086 based system? Explain their format?

(b) Discuss the following interrupts?

i. Single step Execution

ii. Interrupt on Overflow. [10+6]

7. (a) Draw the circuit of TTL to RS232 conversion and explain the necessity of this interface.

(b) Draw the necessary circuit to interface 8251 to an 8086 based system with an address A0H. Write the sequence of instructions to initialize 8251 for syn- chronous transmission with odd parity, single SYNC character,8bit data character.[6+10]

8. (a) Explain the internal and external program memory as well as data memory of8051 with the diagram showing their capacities.

(b) Draw the diagram to Interface Program memory of 16K x 8 EPROM to 8051and give its memory map. The address of memory map should start from 0000H.[8+8]

*****

III B.Tech II Semester Supplementary Examinations, Nov/Dec 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) With a neat architectural diagram explain the functioning of an 8086 micro- processor

   (b) Compare the flag resisters of 8086 & 8085 [10+6]

2. (a) Explain the following 8086 instructions with examples. i. MUL

   ii. IMUL

   iii. DIV

   iv. IDIV

   (b) Differentiate between procedures and macros using certain examples. [8+8]

3. (a) Write a program in 8086 to add two 8-bytes of data available in memory location array1 and array 2. Store the result in array3

   (b) Write an ALP to count number of 0s in a 16 bit binary string[10+6]

4. (a) Explain how static RAMs are interfaced to 8086. Give necessary interface diagram assuming appropriate signals and memory size

   (b) Explain the need of DMA. Discuss in detail about DMA data transfer method[8+8]

5. (a) Explain the following pins of 8255. i. A0 and A1

   ii. WR

iii. RESET

iv. PC0-PC7

(b) Explain the use of handshaking signals used in 8255 [8+8]

6. (a) How many Initialization Command words are required for a single 8259 in an8086 based system? Explain their format?
(b) Discuss the following interrupts?
i. Single step Execution
ii. Interrupt on Overflow. [10+6]

7. (a) Explain why serial data transfer is mostly preferred over parallel data transfer.Give reasons.
   (b) Distinguish between data formats used for Synchronous and Asynchronous serial data transfer modes. [8+8]
8. (a) Draw and discuss the formats and bit definitions of the following registers of 8051.
      i. IP
      ii. IE
   (b) The internal RAM memory (128 bytes) is divided into three parts. Explain it with neat diagram? [8+8]

****

III B.Tech II Semester Supplementary Examinations, Nov/Dec 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) Draw the architectural diagram of 8085 and explain the function of each block in detail
   (b) Discuss about Multiplexing in 8086 microprocessor [10+6]
2. (a) Discuss about the following addressing modes of 8086 with some examples i. Directing addressing
      ii. Register addressing
      iii. Immediate addressing
      iv. Based addressing with displacement
   (b) Explain why assembler directives are required? [8+8]
3. (a) Write an ALP in 8086 to sort a given set of 8-bit unsigned integers into as- cending order by bubble sort method
   (b) Write an ALP in 8086 to display the string "WELCOME" on the screen [10+6]

4. (a) With relevant pin diagrams explain the minimum and maximum mode oper- ations of 8086

   (b) Explain briefly about DMA data transfer method. [12+4]

5. (a) Explain the modes of operation of 8255 with relevant diagrams?

   (b) With a neat diagram explain how a 7-segment display is interfaced to 8086 using 8255? [8+8]

6. (a) Differentiate between Initialization Command Words and Operation Com- mand Words of 8259.

   (b) Discuss about the interrupt priority schemes used in 8259. [10+6]

7. (a) Draw the circuit of RS232 to TTL conversion and explain this interface?

   (b) Draw the internal block diagram of 8251 USART and explain in detail about each block. [6+10]

8. (a) Explain the alternate functions of Port 0, Port 2 and Port 3.

(b) Draw and discuss the formats and bit definitions of TCON register in 8051. [8+8]

*****

III B.Tech II Semester Supplementary Examinations, Nov/Dec 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. Explain the function of following registers in 8086 microprocessor :

[4×4=16] (a)

AX,BX,CX,DX

(b) CS,DS,SS,ES

(c) BP,SP,SI,DI

(d) IP and Instruction Queue

2. (a) Discuss about the following addressing modes of 8086 with some examples i. Directing addressing

ii. Register addressing

iii. Immediate addressing

iv. Based addressing with displacement

(b) Explain why assembler directives are required? [8+8]

3. (a) Write a program in 8086 to add two 8-bytes of data available in memory location array1 and array 2. Store the result in array3

(b) Write an ALP to count number of 0s in a 16 bit binary string[10+6]

4. (a) With relevant pin diagrams explain the minimum and maximum mode oper- ations of 8086

(b) Explain briefly about DMA data transfer method. [12+4]

5. (a) Write an ALP in 8086 to generate triangular waveform and give the necessary circuit setup with a DAC.

(b) Write an algorithm for driving a stepper motor. Assume that the desired direction stored in BL and the number of steps is stored in CL. Write a delay routine for 1 millisecond after each step movement. [8+8]

6. (a) How many Initialization Command words are required for a single 8259 in an

8086 based system? Explain their

format? (b) Discuss the following

interrupts?

i. Single step Execution

ii. Interrupt on Overflow. [10+6]

7. (a) Explain with a neat diagram the working of 8251

PCI. (b) Draw the interface circuits for data conversion

from

i. TTL to RS232C and

ii. RS232C to TTL [8+8]

8. (a) Discuss about various addressing modes of 8051.

(b) Explain the interrput structure of 8051 [8+8]

III B.Tech I Semester Examinations, November 2010

MICROPROCESSORS AND INTERFACING

Common to Information Technology, Instrumentation And Control Engineering, Electronics And Computer Engineering, Computer Science And Engineering

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

1. (a) Explain the Interrupts of 80286 in the order of priority?

(b) Explain the salient features of Pentium Processor? [8+8]

2. (a) Explain memory addressing modes of 8086? Give an example for each ad- dressing mode?

(b) What is the purpose of Trap flag? Discuss how debugging feature is provided with the help of Trap flag in 8086? [8+8]

3. Describe the function of the following pins in 8086 maximum mode of operation. (a) MN/ MX

(b) RQ/GT_0 and RQ/GT_1

(c) QS_0 & QS_1

(d) LOCK [16]

4. (a) Using REPEAT-UNTIL construct, develop a sequence of 8086 instructions that reads a character string from the keyboard and

after pressing the enter key the character string is to be displayed again.

(b) What is a procedure? Give an example to declare a procedure as near? Make this procedure as PUBLIC procedure? [8+8]

5. (a) Explain the application of stepper motor in microcomputers?

(b) Explain with a neat block diagram the working of dual slope ADC? How do you interface the dual slope ADC to microprocessor? Give the required instruction sequence to acquire one sample from ADC? [8+8]

6. (a) Discuss the following signal descriptions?

i. $\overline{INT_0}/\overline{INT_1}$

ii. TXD

iii. T_0 AND T_1

iv. $\overline{RD}$

(b) Draw and discuss the formats and bit definitions of the following SFRs in 8051 microcontroller?

i. TMOD

ii. PSW [8+8]

7. In an 8086 based system it is necessary to serve 64 IRQs from different initiators.The allocated address space for 8259s is from 0700h to 070FH. Give the complete design by choosing the appropriate address locations in the above range? Give the initialization sequence for all 8259's with each IRQ activated in level triggered mode and the starting interrupt is type 40H? [16]

8. (a) What are the MODEM control lines? Explain the function of each line?

Discuss how MODEM is controlled using these lines with necessary sequence of instructions?

(b) Interface 8251 with 8086 at address 0F0H. Initialize it in asynchronous mode, with 8 bit character size, baud rate factor 16, one start bit, two stop bits, even parity enable? [8+8]

III B.Tech I Semester Examinations,November 2010

MICROPROCESSORS AND INTERFACING

Common to Information Technology, Instrumentation And Control Engineering, Electronics And Computer Engineering, Computer Science And Engineering

Time: 3 hours

Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

1. (a) Draw the block diagram of 8086 and explain each block.

   (b) Discuss the addressing modes provided by 8086 and explain with examples. [8+8]

2. (a) Using DF flag and string instructions, write an assembly language program to move a block of data of length 'N from source to destination. Assume all possible conditions.

   (b) Discuss how procedures are defined and involved in assembly language programming. [8+8]

3. Why do we prefer interrupt driven data transfer than programmed I/O transfer?

   Show the complete hardware design to resolve the multiple interrupts based on priority? [16]

4. (a) Explain USB operation?

   (b) Interface 8251 with 8086 at address 0A010H. Initialize it in asynchronous mode, with 6 bit character size, baud rate factor 16, one start bit, two stop bits, odd parity enable? [8+8]

5. (a) Explain DOS interrupt 21H and its functions?

   (b) Under what conditions type 0 interrupt is initiated? List out the instructions that may cause type 0 interrupt? [8+8]

6. It is necessary to initialize interrupt for mode 2 operation of port-A

and mode

1 operation of port-B with the 8255 address map of 0600H to 0603H. Give the complete hard ware design to interface 8255 to 8086 processor with this address map? Write the instruction sequence for the initialization of 8255 in the above modes? Give the instruction sequence to change the operation modes of port A and Port B to mode 1? [16]

7. (a) Explain the paging system of 80386.

   (b) Explain the protected virtual address mode of 80286 and show how 24 bit physical address is generated. [8+8]

8. Interface two 8255s to 8051 with starting address of 0FFF0H? Show the hardware design? Write the instruction sequence to initialize all ports of 8255s as input ports in mode 0. [16]

III B.Tech I Semester Examinations,November 2010

MICROPROCESSORS AND INTERFACING

Common to Information Technology, Instrumentation And Control Engineering, Electronics And Computer Engineering, Computer Science And Engineering

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

1. (a) What is a recursive procedure? Write a recursive procedure to calculate the factorial of number N, where N is a two-digit Hex number?

   (b) Give the assembly language implementation of the

following. i. REPEAT – UNTIL ii. FOR [8+8]

2. (a) Explain initialization command words and their sequence of operation?

(b) Under what conditions type 0 interrupt is initiated? List out the instructions that may cause type 0 interrupt? [8+8]

3. (a) How do we connect RS-232C equipment

i. To data terminal type devices?

ii. To serial port of SDK 86, RS-232C connection?

(b) Give the specifications of

i. RS-232C

ii. RS-423A [8+8]

4. Discuss the following signal descriptions? (a) ALE/PROG

(b) $\overline{EA}$ / V_{PP}

(c) $\overline{PSEN}$

(d) RXD

(e) $\overline{INT_0}$ / $\overline{INT_1}$

(f) TXD

(g) T_0 AND T_1

(h) $\overline{RD}$ [16]

5. 8086 processor do not provide memory indirect addressing mode. Show all possible ways to access a word from memory where the segment address is given in location C000H:1000H and the offset is given in location C000H:1002H. Give the instruction sequence for every addressing mode of 8086.? [16]

6. (a) How many local and global descriptors can be defined in 80286 and explain how to access them?

(b) Discuss the branch prediction logic of Pentium processor? [8+8]

7. What is function of ready pin in 8086. Draw the circuit diagram for wait state generation between 0 and 7 wait status and draw the corresponding timing diagram. [16]

8. Interface an 8-bit DAC to 8255 with an address map of 0804H to 0807H. The DAC provides output in the range of +5V to - 5V. Write the instruction sequence for the following?

(a) For generating a square wave with a peak to peak voltage of 2V and the frequency will be selected from memory location 'FREQ'.

(b) For generating a triangular wave with a maximum voltage of +4V and a minimum of -2V.[8+8]

III B.Tech I Semester Examinations,November 2010

MICROPROCESSORS AND INTERFACING

Common to Information Technology, Instrumentation And Control Engineering, Electronics And Computer Engineering, Computer Science And Engineering

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

1. (a) Write a sequence of instructions to communicate to a modem using 8251 at address 080H.

(b) Give the specifications of

i. RS-232C

ii. RS-423A [8+8]

2. (a) What is the purpose of operational command words of 8259? Explain their format and the use.

(b) Explain the following terms with reference to 8259.

i. Fully nested mode

ii. Automatic rotation [8+8]

3. Describe the function of the following pins and their use in 8086 based system.

(a) $\overline{NMI}$

(b) $\overline{LOCK}$

(c) $\overline{TEST}$

(d) RESET [16]

4. (a) Discuss various branch instruction of 8086 microprocessor, that are useful for relocation?

(b) Using a do-while construct, develop a sequence of 8086 instructions that reads a character string from the keyboard and after pressing the enter key the character string is to be displayed again. [8+8]

5. (a) Draw and discuss the formats and bit definitions of the following SFRs in 8051 microcontroller?

i. SCON

ii. TCON

(b) Discuss the following signal descriptions?

i. ALE/PROG

ii. $\overline{EA}$ / V_{pp}

iii. $\overline{PSEN}$

v. RXD [8+8]

6. It is necessary to check weather the word stored in location 6000H:6000H is zero or not. Show all possible ways of testing the above condition using different address-

ing modes and store 0FFH if the condition is satisfied in location A000H:1002H. Otherwise store 00H. [16]

7.(a) Discuss memory management of virtual 8086 mode in 80386.

(b) Bring out the architectural differences between 80386 and Pentium processors.

8. Interface a stepper motor with 8-step input sequence to 8086 based system and write the instruction sequence to move the stepper motor 20 steps in clockwise and 12 steps in anti-clockwise direction. [16]

III B.Tech II Semester Regular Examinations, April/May 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) Draw the architectural diagram of 8085 and explain the function of each block in detail

   (b) Discuss about Multiplexing in 8086 microprocessor [10+6]

2. (a) Explain in detail about the following addressing modes of 8086 with examples. i. I/O port addressing

   ii. Based indexed addressing with displacement

   (b) Write an ALP in 8086 to add two 16- bit hexa decimal numbers [10+6]

3. (a) Write an ALP in 8086 to sort a given set of 8-bit unsigned integers into as- cending order by bubble sort method

   (b) Write an ALP in 8086 to display the string "WELCOME" on the screen [10+6]

4. (a) Explain how static RAMs are interfaced to 8086. Give necessary interface diagram assuming appropriate signals and memory size

(b) Explain the need of DMA. Discuss in detail about DMA data transfer Method

5. (a) Distinguish between Mode set control word and BSR control Word of 8255? (b) Write an ALP in 8086 to generate a symmetrical square wave form with 1KHz frequency? Give the necessary circuit setup with a DAC? [8+8]

6. (a) Draw the internal block diagram of 8259, and explain in detail about each block?

(b) With examples, discuss briefly about software and hardware interrupts. [10+6]

7. (a) Discuss the types of serial communication?

(b) Write an 8086 instruction sequence for receiving 50 characters using 8251 and store them in memory at location 2080H.. [8+8]

8. (a) Discuss about various addressing modes of 8051.

(b) Explain the interrput structure of 8051 [8+8]

*****

III B.Tech II Semester Regular Examinations, April/May 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. Explain the function of following registers in 8086 microprocessor : [4×4=16]
   (a) AX,BX,CX,DX
   (b) CS,DS,SS,ES
   (c) BP,SP,SI,DI
   (d) IP and Instruction Queue

2. (a) Explain the following 8086 instructions with examples. i. MUL
   ii. IMUL
   iii. DIV
   iv. IDIV

   (b) Differentiate between procedures and macros using certain examples. [8+8]

3. (a) Write a program in 8086 to add two 8-bytes of data available in memory location array1 and array 2. Store the result in array3

(b) Write an ALP to count number of 0s in a 16 bit binary string [10+6]

4. (a) With a neat block diagram, explain the working of 8257 DMA controller

(b) Explain briefly about memory interfacing with 8086 microprocessor [10+6]

5. (a) Write an ALP in 8086 to generate triangular waveform and give the necessary circuit setup with a DAC.

(b) Write an algorithm for driving a stepper motor. Assume that the desired direction stored in BL and the number of steps is stored in CL. Write a delay routine for 1 millisecond after each step movement. [8+8]

6. (a) Distinguish between interrupts and exceptional handling interrupts

(b) Discuss about the following terms:

i. Vector interrupt table and lookup table.

ii. Interrupt service routines and subroutines. [8+8]

7. (a) Discuss the types of serial communication?

(b) Write an 8086 instruction sequence for receiving 50 characters using 8251 and store them in memory at location 2080H.. [8+8]

8. (a) Explain the internal and external program memory as well as data memory of 8051 with the diagram showing their capacities.

(b) Draw the diagram to Interface Program memory of 16K x 8 EPROM to 8051and give its memory map. The address of memory map should start from 0000H. [8+8]

*****

III B.Tech II Semester Regular Examinations, April/May 2009

MICROPROCESSORS AND INTERFACING

(Common to Electronics & Communication Engineering, Electronics & Instrumentation Engineering, Bio-Medical Engineering, Electronics & Control Engineering and Electronics & Telematics)

Time: 3 hours Max Marks: 80

Answer any FIVE Questions

All Questions carry equal marks

*****

1. (a) Compare the Interrupt structure of 8086 and 8085 Microprocessors. Also discuss about priorities of the interrupts in both the cases.

   (b) With examples ,explain how multiplexing is implemented in 8086 Microproces- sor [10+6]

2. (a) Explain in detail about the following addressing modes of 8086 with examples. i. I/O port addressing

   ii. Based indexed addressing with displacement

   (b) Write an ALP in 8086 to add two 16- bit hexa decimal numbers [10+6]

3. (a) Write an ALP in 8086 to count number of positive and negative numbers from an array of 8-bit integers

   (b) Write an ALP in 8086 to exchange a block of N bytes of data between source and destination [8+8]

4. (a) With relevant pin diagrams explain the minimum and maximum mode oper- ations of 8086

   (b) Explain briefly about DMA data transfer method. [12+4]

5. (a) With a neat internal architectural diagram, explain the features of 8255?

(b) Explain the interfacing of a stepper motor with 8086 using the ports of 8255? [8+8]

6. (a) What is the purpose of operational command words of 8259? Explain their format and the use?

   (b) Discuss the following interrupts. i. Divide Error

   ii. Non Maskable interrupt

   iii. Break Point interrupt [10+6]

7. (a) Write the instruction sequence in 8086 to initialize the 8251

   (b) Draw and explain the block diagram of 8251. [8+8]

8. (a) Draw the architectural diagram of 8051 microcontroller and explain in detail about each block.

(b) Explain the basic differences between a microprocessor and a microcontroller.

[10+6]

17.Question bank:

UNIT-1

General definitions of mini computers etc,.

1. What are the advantages and the limitations microcontroller over a Microprocessor?
2. Describe the main blocks in a digital signal processor that are not in a general Microprocessor?
3. Overview of 8085 Microprocessor
4. List the internal registers in 8085 microprocessor and their abbreviations and lengths. Describe the primary function of each register.
5. Differentiate between NMI and MI interrupts
6. Explain how with external hardware TRAP can be masked
7. Explain the timing diagrams of 8085 when it is executing Memory mapped I/O and I/O mapped I/O instructions

   B) Explain the following pins of 8085 microprocessor

   i) SID,SOD ii) ALE iii) HOLD,HLD iv) TRAP

8.Explain the features of 8086

9. Explain architecture if 8086
10. Explain the function of following registers in 8086 microprocessor.

    a) AX, BX, CX, DX. b) CS, DS, SS, ES. C) BP, SP, SI ,DI.

    d) IP & instruction queue e) Flags.

UNIT-2

1. List out the addressing modes of 8085, & Explain each with an example
2. Write a algorithm & ALP to reverse the bits in a 16-bit number & check weather it is a Palindrome or not.
3. Explain all branch and call instructions

4. Write an ALP for Ascending and Descending of a series

5.Write an ALP for Matrix multiplication

6. Explain the meaning of the following 8086 instructions.

i) MOV [3845H],BX ii) ADD AX,[SI]

iii) MOV BX,2956H iv) ADC AX,BX.

7. Explain string instructions supported by 8086 processor.

8. Write a algorithm & ALP to reverse the bits in a 16-bit number & check weather it is a Palindrome or not.

9. If BH = 0F3H what is the value of BH in hex after the instruction SAR BH, 1

10. IF AL = 78H and BL=73H explain how DAS instruction (after subtracting BL From AL) adjusts to the BCD result

11. If CL=78H what is the value of CL after the instruction ROL CL, 3

12. Why AAD is to be executed before DIV instruction while converting unpacked BCD to Binary number

13. Under what conditions REPE MOVS executes

14. Explain XLAT instruction to linearize transducer characteristics

15. Explain intra segment and inter segment branch instructions with examples the instructions related to arithmetic and logical shift.

16. Explain all addressing modes with the assembler syntax and how effective address is calculated

UNIT-3

1.How do you configure 8086 into minimum and maximum modes

2.Bring out the differences between 8086 and 8088 processors

3.Explain all the features in 8284

4.Why and when wait states are required. How do you insert wait states1.

5. . In an SDK-86 kit **128KB SRAM** & **64KB EPROM** is provided on system and provision

Expansion of another **128KB SRAM** is given. Then on system SRAM address starts from **00000H** & EPROM ends with **FFFFFH.** The expansion slot address map is from **80000H** To **9FFFFH.** The size of **SRAM chip is 64KB, EPROM chip size is 16KB**.Give the Complete memory interface and also the address map for complete memory map for Individual chips..

UNIT-4

1. What are the steps in interfacing peripherals with the micro processor?
2. Sketch and explain the interface of PPI 8255 to the 8086 microprocessor in minimum mode. Interface 4 7 segment LEDs to display as a BCD counter
3. In the above question Q2 interface two keys UP and DOWN to the PPI. Write an 8086 assembly program segment such that when UP is pressed the counter counts up every second. Similarly when DOWN key is pressed the counter decrements every second
4. Sketch and explain the interface of 8279 to the 8086 microprocessor in minimum mode. Interface 8x8 key pad and 16x 7 Seg LED display. Write an 8086 assembly program to read the key codes of keys and display -NPTEL-INDIA
5. Sketch and explain the interface of PIT 8254 to the 8086 microprocessor in minimum mode. Cascade two counters in the PIT. Write a program segment two get one minute delay
6. What are the differences in interfacing RWMs while 8086 is in minimum and maximum modes?
7. Sketch and explain the interface of 32K x 16 RWMs using a decoder in minimum mode. What is the maximum access time of ROMs such that it does not require wait states when 8086 operates at 8 MHz?
8. Sketch and explain the timing diagrams in the above interface Question 2
9. Sketch and explain the 8086 bus activities during write machine cycle

10.Review on Interrupt Structure of 8086

11.POS & BIOS interrupts

12.PIC (8259)

13.Cascading of 8259

UNIT-5

1.Explain serial data transfer schemes

2.Pin configuration of 8251

3.Architectural features of 8251

4.Sending and receiving a character

UNIT-6

1. What are the advantages and disadvantages of using Harvard architecture in 8051?
2. How much maximum external program memory can be interfaced?
3. Explain PSW SFR. Give the application differences between Carry and Overflow flags
4. What are the power consumptions in power down and idle modes
5. Explain Quasi Bidirectional ports of 8051
6. What is the status of all registers on reset?
7. What is the maximum delay the Timer0 produces when 8051 is operated at 12MHz?
8. Explain how in Serial communication mode 0 expands I/O lines with the help of

Shift

UNIT-7

1.explain timers/counters?

2.explain different modes of timers/counters?

3.explain serial communication concepts in 8051?

4.explain different modes in serial communication?

5.explain interrupt structure of 8051?

UNIT-8

1.AVR family architecture?

2. AVR family registers file and ALU?

3. explain memory access and Instruction Execution in AVR controllers?

4. explain I/O memory, EEPROM in AVR controllers?,

5.explain I/O ports, timers in AVR controllers?

6.explain UART, Interrupt structure in AVR controllers?

18.Assignment Questions

UNIT 1:

1. Explain 8085 architecture. Describe the Registers in 8085.

2. Draw the 8086 architecture diagram and explain various blocks of the diagram. Detail the Register organization in 8086.

3. What is memory segmentation? Describe physical memory organization. How to calculate the physical memory address.

4. Describe minimum mode and maximum mode operation in 8086 along with Timing diagrams.

UNIT 2:

1. Explain the instruction formats in 8086. give the complete instruction set of 8086 along with examples.
2. What is an addressing mode? Explain various addressing modes of 8086 along with examples.
3. What is an assembler directive and Macro? Give the differences between Procedure and Macro.Explain Assembler directives, Procedures and Macros with the help of examples.
4. Write ALP for
a. to perform multi-byte addition.
b. to perform the sorting N numbers.
c. to display the string "GEETHANJALI"
d. to move a block of data from source to destination.

UNIT 3:

1.8255 architecture

2. Keyboard display interfacing

3. ADC interfacing

4. Stepper motor interfacing

UNIT-4:

1. Explain how memory is interfaced with 8086. Write memory connection diagrams for minimum mode and maximum mode operation of 8086.
2. Explain interrupt structure of 8086. What is vector table? what are the operations done during handling an interrupt service routine?
3. Explain 8259 PPI, along with block diagram, functions and operations?
4. Explain 8257 DAM controller, with block diagram, features and operation?

UNIT-5:

1. Explain why serial data transfer is mostly preferred over parallel data transfer. Give reasons? Discuss the types of serial communication?

2. Draw the internal block diagram of 8251 USART and explain in detail about each block.

3. With a neat circuit diagram, explain the interfacing of 8251 with 8086

4. Design a hardware interfacing circuit for interfacing 8251 with 8086. Set the 8251A in asynchronous mode as a transmitter and receiver with even parity enabled, 2 stop bits, 8-bit character length, frequency 160 KHz and baud rate 10K. Write an ALP to transmit 100 bytes of data string starting at location 2000:5000H.

5. Give the description about the following.

(I) RS-232

(II)IEEE-488 GPIB

(III) Prototyping and troubleshooting

.

UNIT-6:

1.8051 microcontroller architecture

2. Register organization of 8051

3. Addressing modes and i/o ports of 8051.

4. Memory segmentation of 8051.

UNIT-7:

1. Timer/counter operations

2. Serial communication operation

3. Memory interfacing

4. Various modes of operation of timers.

UNIT8:

1.AVR family architecture?

2. AVR family registers file and ALU?

3. explain memory access and Instruction Execution in AVR controllers?

4. explain I/O memory, EEPROM in AVR controllers?,

19. Unit-wise quiz questions

1. Mode 1 of 8255 is used for which of these I/O methods
(a) Bi-Directional Handshake
(b) DMA I/O
(c) Strobed or Hand Shake mode
(d) Simple Input/Output
2 . If the 8255 is selected for addresses 0F800H to 0F806H, which one of the following is the address of PORT C
(a) 0F806H
(b) 0F802H
(c) 0F804H
(d) 0F800H
3 . If 8 seven segment led multiplexed displays including the decimal point are connected. How many wires are needed to connect the segments (not including power supply wires)?
(a) 15
(b) 16
(c) 56
(d) 64
4 . The diode kept across the coil of a step per motor is for ?
(a) Shorting the Back EMF
(b) To stop the step per motor
(c) Stopping flow of current
(d) Allowing flow of current
5 . In the A/D converter ADC0808 which of these signals tri states the digital output of ADC
(a) EOC
(b) ALE
(c) SOC
(d) OE
6 . The fourth response of the 8086 to an interrupt is which one of the following ?
(a) Decrements the SP by 2 & pushes flag register
(b) It disables interrupt flag
(c) Decrements the SP& pushes current CS contents
(d) Resets Trap Flag
7 . What is the size of the vector interrupt table?
(a) 256 bytes
(b) 65536 bytes
(c) 1024 bytes
(d) 4096 bytes
8 . Which of these values in AH perform input from keyboard ?
(a) 02 (b) 09 (c) 17 **(d) 01**
9 . Which one of these BIOS services is used for Mouse Operation?
(a) 10H
(b) 13H
(c) 16H
(d) 33H

10. The PIC interrupt controller which Masks the Interrupt is?
(a) IRR
(b) IMR
(c) ISR
(d) PR
11. Data line which can be transfer in only one direction is referred as
(a) Half Duplex
(b) Com plex
(c) Full Duplex
(d) Simplex
12. In the Synchronous mode the 8251 which signal indicates that the specified Synchronous character is received?
(a) Tx C
(b) SYNDET/BD
(c) CLK
(d) C/D
13. The outer loop counter for N numbers is
(a) N-1 (b) N-2 (c) N (d) N/2
1 4. If the CPU has not read a character from the 8251 before the arrival of the next character. Which of the following errors is supposed to have occurred?
(a) Even Parity error
(b) Framing Error
(c) Over Run Error
(d) Odd Parity Error
15. USB stands for which one of the following?
(a) Universal Sequential Bus
(b) Universal Serial Bus
(c) College Bus
(d) Universal Sequential Bus
16. The 8051 is a
(a) 32 bit micro controller
(b) 24 bit micro controller
(c) 1 6 bit micro controller
(d) 8 bit micro controller
17. If the bit 8 th bit of memory location 20H is to be set, what will be its bit addressable memory location?
(a) 09
(b) 07
(c) 06
(d) 08
18. Counter or timer operation is chosen in which of these registers?
(a) TLx
(b) TCON
(c) THx
(d) TMOD

19. Serial port operates in Multi Processor mode in which of these modes?
(a) Mode3
(b) Mode2
(c) Mode0
(d) Mode1
20. The upper 8 bits of address bus are generated on which of these ports?
(a) Port2
(b) Port3
(c) Port1
(d) Port0
21 . Mode 1 of 8255 is used for which of these I/O methods
(a) Simple Input/ Output
(b) DMA I/O
(c) Bi-Directional Hand shake
(d) Strobed or Hand Shake mode
22 . Two 8255‘s IC1&IC2 are connected to 8086 16 bit bus and D0 to D7 are connected to IC1 and D8 to D15 are connected to IC2. The base address signals A0&A1 goto IC1&IC2‘s A0&A1 pins respectively. If a decoded signal with a range of 0FFE0H to 0FFE3H then What is the address of IC1PORT B register
(a) 0FFE2H
(b) 0FFE0H
(c) 0FFE3H
(d) 0FFE1H
23 . If 8 seven segment led multiplexed displays including the decimal point are all lighted up and each of these is biased to carry 5mA. What is the total current drawn from the power supply when all the segments are ON?
(a) 280 mA
(b) 80 mA
(c) 40 mA
(d) 300 mA
24 . The diode kept across the coil of a stepper motor is for?
(a) Stopping flow of current
(b) To stop the stepper motor
(c) Allowing flow of current
(d) Shorting the Back EMF
25 . In the A/D converter ADC 0808 which of these signals tri states the digital output of ADC
(a) OE
(b) SOC
(c) EOC
(d) ALE
..
..
26 . The fourth response of the 8086 to an interrupt is which

one of the following?
(a) Decrements the SP by 2 & pushes flag register
(b) Resets Trap Flag
(c) It disables interrupt flag
(d) Decrements the SP& pushes current CS contents
27 . What is the segment address of the vector interrupt table?
(a) 0F0000H
(b) 00000H
(c) 10000H
(d) 0F000H
28 . Which of these values in AH performs output on video display of Ascii character in DL?
(a) 01
(b) 09
(c) 02
(d) 17
29 . Which one of these BIOS services is used for Key Board Operation ?
(a) 33H
(b) 13H
(c) 16H
(d) 10H
30. The PIC interrupt controller connects the single 8086 IRQ to how many IRQ input lines?
(a) 4
(b) 8
(c) 32
(d) 16
31. Data line which can be transfer in only one direction is referred as as
(a) Full Duplex
(b) Half Duplex
(c) Complex
(d) Simplex
32. USART stands for which of the following?
(a) Unidirectional Synchronous Asynchronous Receiver Transmitter
(b) Unidirectional Sequential Asynchronous Receiver Transmitter
(c) Universal Synchronous Asynchronous Receiver Transmitter
(d) Universal Serial Addressed Receiver Transmitter
33. Which combination of the flags indicate that the source is greater than the destination
(a) CY=1,Z=1
(b) CY=0,Z=0
(c) CY=0,Z=1
(d) CY=1,Z=0

34. When a valid stop bit is not detected at the end of every character in a synchronous mode, the error is called as which one of the following?
(a) Over Run Error
(b) Odd Parity Error
(c) Even Parity error
(d) Framing Error
35. USB stands for which one of the following?
(a) Universal Sequential Bus
(b) Unidirectional Serial Bus
(c) Universal Serial Bus
(d) SMCE bus
36. The size of the ROM in 8051 is
(a) 32 K bytes
(b) 16 K bytes
(c) 4 K bytes
(d) 8 K bytes
37. If the bit 8th bit of memory location 20H is to be set, what will be its bit addressable memory location?
(a) 06
(b) 09
(c) 08
(d) 07
38. The TF bit is in which of these registers?
(a) TCON
(b) TLx
(c) THx
(d) TMOD
39. The interrupt vector address for timer 0 is which one of the following addresses?
(a) 000BH
(b) 0003H
(c) 0023H
(d) 0013H
40. The instruction used to access data from external memory is which one of the following?
(a) MOV @
(b) MOV
(c) MOVC
(d) MOV X
41 . The number of ports 8255 PPI has is
(a) 1 **(b) 3** (c) 2 (d) 4
42 . If the control word 09BH is given to the control register of the 8255 PPI then which of these options are the condition of the ports
(a) PORT‘s A&B INPUT,PORT C OUTPUT
(b) ALL PORT‘s INPUT
(c) BIT SET RESET

(d) ALL PORT‘s OUTPUT
43 . A key board is made using 12 keys connected not as a matrix. How many pins of the I/O port are used?
(a) 12
(b) 07
(c) 11
(d) 09
44 . What is the angle per step of a step per motor in full step mode?
(a) 1.0 degrees
(b) 1.8 degrees
(c) 0.45 degrees
(d) 0.9 degrees
45 . In the A/D converter ADC 0808 which of these signals is used select an analog input channel to be digit is used
(a) ALE
(b) SOC
(c) OE
(d) EOC
46 . The second response of the 8086 to an interrupt is which one of the following?
(a) Resets Trap Flag
(b) Decrements the SP by 2&pushes flag register
(c) Decrements the SP & pushes current CS contents
(d) It disables interrupt flag
47 . Which of these is called when an NMI occurs?
(a) INT3
(b) INT2
(c) INT0
(d) INT1
48 . The DOS function AH=9 INT2 1H displays a message till which one of this character is reached
(a) $ (b) × (c) # (d) *
49 . Which one of these BIOS services is used for Communication Operation?
(a) 17H
(b) 15H
(c) 16H
(d) 14H
50. The total number of interrupts available if all the 8 inputs of PIC are each having a slave is?
(a) 128
(b) 64
(c) 256
(d) 16
..
..
51. The USART used as Serial Communication controller is

(a) 8259
(b) 8257
(c) 8250
(d) 8251
52. USART stands for which of the following ?
(a) Unidirectional Synchronous Asynchronous Receiver Transmitter
(b) Universal Synchronous Asynchronous Receiver Transmitter
(c) Unidirectional Sequential Asynchronous Receiver Transmitter
(d) Universal Serial Addressed Receiver Transmitter
53. Given the program shown below , What is the expression evaluated by this program?

```
P db 5
Q db 6
R db 7
MOV AL,[P]
MUL AL
MOV BL,AL
MOV AL,[Q]
MUL AL
ADD AL,BL
MOV [R],AL
```

(a) R = P + Q2
(b) R =(P + Q)2
(c) R=(P2+ Q2)
(d) R=(P+Q)
54.If the CPU has not read a character from the 8251 before the arrival of the next character. Which of the following errors is supposed to have occurred?
(a) Even Parity error
(b) Odd Parity Error
(c) Framing Error
(d) Over Run Error
55. Which one of these serial data transmission standards is single ended and does not use low impedance drivers?
(a) RS449
(b) RS232C
(c) RS423A
(d) RS422A
56. Which one of the following is the 8051 architecture based upon?
(a) Princeton Architecture
(b) Param Architecture
(c) Harvard Architecture
(d) Von Neumann Architecture
57. General purpose registers R0 to R7 can be referred as sets

of four banks (0to3) and also by memory location. If a register R3 in Bank2 is referred, what is its internal ram location address?
(a) 14H
(b) 10H
(c) 12H
(d) 13H
58. Timer0 is functional as an 8 bit counter while timer 1 is stopped in which of the following modes?
(a) Mode2
(b) Mode1
(c) Mode3
(d) Mode0
59. Which of the following is the correct order of priority in decreasing order from left to right?
(a) EX0,EX1,ET0,ET1,ES,
(b) ES,EX1,ET0,E01,ET1
(c) ET0,EX0,ET1,EX1,ES
(d) EX0,ET0,EX1,ET1,ES
60. One of these signals indicates that address is available on Port0?
(a) EA
(b) RESET
(c) ALE
(d) PSEN
61 . The 8255 PPI has
(a) 28 pins
(b) 16 pins
(c) 40 pins
(d) 64 pins
62 . Two 8255‘s IC1&IC2 are connected to 8086 16 bit bus and D0 to D7 are connected to IC1 and D8 to D15 are connected to IC2. The base address signals A0&A1 goto IC1&IC2‘s A0&A1 pins respectively. If a decoded signal with a range of 0FFE0H to 0FFE3H then What is the address of IC1 PORTB register
(a) 0FFE2H
(b) 0FFE3H
(c) 0FFE1H
(d) 0FFE0H
63 . A key board is made using 12 keys connected not as a matrix. How many pins of the I/O port are used?
(a) 12
(b) 07
(c) 09
(d) 11
64 . How many wires does as tepper motor effectively have?
(a) 4
(b) 2

(c) 3
(d) 5
65 . An A/D converter which has a maximum output digital value of 1023 need show many wires for connecting its digital output to a peripheral chip
(a) 10
(b) 13
(c) 8
(d) 12
66 . Which of these signals has the highest priority?
(a) IRQ
(b) Hold
(c) NMI
(d) Reset
67 . What is the number of Software interrupts available in the Interrupt Vector Table?
(a) 4096
(b) 256
(c) 1024
(d) 512
68 . DOS is an abbreviation which stands for?
(a) Data Output Storage
(b) Disk Operating Systems
(c) Disk Output Storage
(d) Device Operating System
69 . Which one of these BIOS services is used for Mouse Operation?
(a) 33H
(b) 13H
(c) 16H
(d) 10H
70. The PIC interrupt controller which indicates that a Interrupt is being requested is?
(a) ISR **(b) IRR** (c) IMR (d) PR
71. The device from which data originates or terminates is called
(a) DSR
(b) DTR
(c) DTE
(d) DCE
72. USART stands for which of the following ?
(a) Unidirectional Synchronous Asynchronous Receiver Transmitter
(b) Universal Synchronous Asynchronous Receiver Transmitter
..
..
(c) Unidirectional Sequential Asynchronous Receiver

Transmitter
(d) Universal Serial Addressed Receiver Transmitter
73. Given the program shown below, What is the value or R after this program executes?

```
P db 5
Q db 6
R db 7
MOV AL,[P]
MUL AL
MOV BL,AL
MOV AL,[Q]
ADD AL,BL
MOV [R],AL
```

(a) R=(P2+Q1)[P2+Q]
(b) R=(P+Q)
(c) R=(P+Q)2
(d) R=P+Q2
74. When a valid stop bit is not detected at the end of every character in asynchronous mode, the error is called as which one of the following?
(a) Odd Parity Error
(b) Over Run Error
(c) Even Parity error
(d) Framing Error
75. USB stands for which one of the following?
(a) Universal Sequential Bus
(b) Uni directional Serial Bus
(c) Universal Serial Bus
(d) Universal Sequential Bus
76. The 8051 micro controller does not have one of the following as its built in peripheral?
(a) Timers
(b) Counters
(c) UART
(d) DMA controller
77. Which of these registers cannot be decremented?
(a) SP
(b) B register
(c) Accumulator A
(d) DPTR
78. One of these registers is loaded with the upper byte of the terminal count?
(a) THx
(b) TMOD
(c) TLx
(d) TCON
79. The bit which disables reception of serial data is which one of the following?

(a) RB8
(b) TB8
(c) RI
(d) REN
80. The instruction which activates the PSEN signal is which one of the following?
(a) MOVC
(b) MOV @
(c) MOV
(d) MOV X
81 .Which of these is Double Hand Shake
(a) STB followed by ACK
(b) STB followed by ACK and removal of ACK
(c) STB only
(d) ACK followed by STB
82 . Two 8255‘s IC1&IC2 are connected to 8086 16 bit bus and D0 to D7 are connected to IC1 and D8 to D15 are connected to IC2. The base address signals A0&A1 goto IC1&IC2‘s A0&A1 pins respectively. If a decoded signal with arrange of 0FFE0H to 0FFE3H then What is the address of IC1 PORTB register
(a) 0FFE2H
(b) 0FFE0H
(c) 0FFE3H
(d) 0FFE1H
83 . A key board is made using12 keys connected as a matrix of 4X3. How many pins of the I/O port are used?
(a) 12 (b) 09 **(c) 07** (d) 11
84 .How many steps per revolution does a stepper motor in full step mode take?
(a) 200
(b) 400
(c) 380
(d) 220
85 . A n A/D converter which has a maximum output digital value of 1023 need show many wires for connecting its digital output to a peripheral chip
(a) 12 **(b) 10** (c) 8 (d) 13
86 . Which of these signals has the second highest priority?
(a) NMI
(b) Hold
(c) Reset
(d) Irq
87 . Which of these is called for doing Single Stepping?
(a) INT0
(b) INT2
(c) INT3
(d) INT1
88 . Which of these values in AH perform input from key

board?
(a) 09 (b) 02 (c) 17 **(d) 01**
89 . Which one of these BIOS services is used for Video Operation?
(a) 13H **(b) 10H** (c) 33H (d) 16H
90. The PIC interrupt controller which Masks the Interrupt is?
(a) PR
(b) IRR
(c) IMR
(d) ISR
91. The device from which data originates or terminates is called
(a) DSR (b) DTR **(c) DTE** (d) DCE
92. In the Synchronous mode the 8251 which signal indicates that the specified Synchronous character is received?
(a) CLK
(b) C/D
(c) TxC
(d) SYNDET/BD
93. Given the program shown below , What is the value or R after this program executes?
P db 5
Q db 6
R db 7
MOV AL,[P]
MUL AL
MOV BL,AL
MOV AL,[Q]
ADD AL,BL
MOV [R],AL
(a) R =(P+Q)
(b) R=(P+Q)2
(c) R=(P2+Q1)[P2+Q]
(d) R = P + Q 2
94. When a valid stop bit is not detected at the end of every character in asynchronous mode, the error is called as which one of the following?
(a) Over Run Error
(b) Even Parity error
(c) Framing Error
(d) Odd Parity Error
95. USB connector has how many pins?
(a) 3 (b) 2 (c) 5 **(d) 4**
96. The 8051 micro controller is?
(a) 8 bits
(b) 32 bits
(c) 16 bits
(d) 24 bits

97. Which of these registers cannot be decremented?
(a) SP
(b) Accumulator A
(c) DPTR
(d) B register
98. Which one of the following starts the timer?
(a) IE1
(b) TF
(c) TR
(d) IT1
99. The interrupt vector address for serial reception is which one of the following addresses?
(a) 0003H
(b) 0023H
(c) 000B H
(d) 0013H
100. The instruction used to access internal RAM is which one of the following?
(a) MOV @
(b) MOV
(c) MOV X
(d) MOVC
101. The number of ports 8255 PPI has is
(a) 3 (b) 2 (c) 1 (d) 4
102. When 8255 port is to be read then which of these combinations is valid
(a) RD= 1 , W R= 0, R E S E T =0 , C S =0
(b) RD= 0 , W R= 1, R E S E T =1 , C S =0
(c) R D= 0 , W R= 1, R E S E T =0 , C S =1
(d) R D= 0 , W R= 1, R E S E T =0 , C S =0
103 . Which of these key switches is non mechanical contact type?
(a) MECHANICAL KEY SWITCHES
(b) HALL EFFECT KEY SWITCHES
(c) MAGNETIC REED KEY SWITCHES
(d) MEMBRANE KEY SWITCHES
104 . How many steps per revolution does a stepper motor in full step mode take?
(a) 220 **(b) 200** (c) 400 (d) 380
105 . An A/D converter which has a maximum output digital value of 1023 need show many wires for connecting its digital output to a peripheral chip
(a) 10 (b) 13 (c) 12 (d) 8
106 . Which of these is used for Power brown out management?
(a) Hold
(b) NMI
(c) Irq

(d) Reset
107 . Which of these is called when a Divide Error occurs?
(a) INT3
(b) INT1
(c) INT2
(d) INT0
108 . DOS Interrupts reside in the?
(a) ROM
(b) Floppy Disk
(c) Hard Disk
(d) RAM
109 . For which one of these values of AL does the Video interrupt sets 80X25 color video mode?
(a) 02H (b) 00H **(c) 03H** (d) 01H
11 0. Which of these Initialisation Command Word indicates whether slave connected to the PIC IRQ pins?
(a) ICW3
(b) ICW2
(c) ICW4
(d) ICW1
111. The device from which data originates or terminates is called
(a) DSR (b) DTR **(c) DTE** (d) DCE
112. When the MODEM is ready to Transmit Data it Asserts which one of these signals is asserted?
(a) DSR (b) CD (c) RTS **(d) CTS**
113. Given the program shown below, What is the expression evaluated by this program?

```
P db 5
Q db 6
R db 7
MOV AL,[P]
MUL AL
MOV BL,AL
MOV AL,[Q]
MUL AL
ADD AL,BL
MOV [R],AL
```

(a) R=(P2+ Q2)
(b) R=(P+Q)
(c) R= P+Q2
(d) R=(P+Q)2
114. The L2 and 1 bits in the Mode word are used for setting which of these parameters?
(a) NUMBER OF STOP BITS
(b) CHARACTER LENGTH
(c) BAUD RATE FACTOR
(d) PARITY

115. The data signals used by USB are of which type of the following?
(a) Uni-Phase
(b) Quad Phase
(c) Bi-Phase
(d) Tri-phase
116. The 8051 micro controller built in timer is of how many bits?
(a) 16 (b) 8 (c) 32 (d) 24
117. Which of these register of 8051 is 16 bit?
(a) DPTR (b) PSW (c) SP (d) A
118. One of these registers is loaded with the upper byte of the terminal count?
(a) TLx
(b) TCON
(c) THx
(d) TMOD
119. Which of the following is the correct order of priority in decreasing order from left to right?
(a) ET0,EX0,ET1,EX1,ES
(b) EX0,EX1,ET0,ET1,ES,
(c) EX0,ET0,EX1,ET1,ES
(d) ES,EX1,ET0,E01,ET1
120. Which of these signals is used for enabling the external ROM?
(a) ALE
(b) RESET
(c) EA
(d) PSEN
121 .Which of these is Single Hand Shake
(a) STB followed by ACK and removal of ACK
(b) STB only
(c) STB followed by ACK
(d) ACK followed by STB
122 . At power on or after reset the ports are in which of these conditions
(a) PORT A=INPUT,PORT B=INPUT,PORT C=INPUT
(b) PORT A=INPUT,PORT B=OUTPUT,PORT C=OUTPUT
(c) PORT A=OUTPUT,PORT B=INPUT,PORT C=OUTPUT
(d) PORT A= INPUT,PORT B=INPUT, PORT C=OUTPUT
123 . Which of these key switches is non mechanical contact type ?
(a) MEMBRANE KEY SWITCHES
(b) HALL EFFECT KEY SWITCHES
(c) MECHANICAL KEY SWITCHES
(d) MAGNETIC REED KEY SWITCHES
124 . How many steps per revolution does a stepper motor in half step mode take?

(a) 200 (b) 220 **(c) 400** (d) 380
125 . An 8 bit A/D converter with a reference voltage of 5 Volts will be able to read a lowest non zero voltage of
(a) 20 mV
(b) 19.5 mV
(c) 19 mV
(d) 21 mV
126 . The third response of the 8086 to an interrupt is which one of the following?
(a) Resets Trap Flag
(b) Decrements the SP by 2&pushes flag register
(c) Decrements the SP & pushes current CS contents
(d) It disables interrupt flag
127 . What is the size of the vector interrupt table?
(a) 4096 bytes
(b) 256 bytes
(c) 65536 bytes
(d) 1024 bytes
128 . In the single character display using Interrupt 21H,DL is the register which contains the character to display. If DL contains 61 H which of these characters is displayed?
(a) B **(b) a** (c) b (d) A
129 . For which one of these values of AH the key board interrupt wait for a key to be pressed before returning?
(a) 03H (b) 01H (c) 02H **(d) 00H**
130. Which of these Initialisation Command Word indicates whether PIC is being used in Single or Cascaded mode?
(a) ICW3
(b) ICW1
(c) ICW4
(d) ICW2
131. The device from which data originates or terminates is called
(a) DTE
(b) DCE
(c) DSR
(d) DTR
132. Serial data is received by the DTE from the DCE through which one of these signals?
(a) RTS
(b) CD
(c) CTS
(d) RxD
133. Given the program shown below, What is the expression evaluated by this program?
P db 5
Q db 6
R db 7

MOV AL,[P]
MUL AL
MOV BL,AL
MOV AL,[Q]
MUL AL
ADD AL,BL
MOV [R],AL
(a) R=(P+Q)
(b) R=(P+Q)2
(c) R=(P2+Q2)
(d) R=P+Q2
134. The L2 and 1 bits in the Mode word are used for setting which of these parameters?
(a) CHARACTER LENGTH
(b) NUMBER OF STOP BITS
(c) PARITY
(d) BAUD RATE FACTOR
135. The data signals used by USB are of which type of the following?
(a) Uni-Phase
(b) Bi-Phase
(c) Quad Phase
(d) Tri-phase
136. The size of the ROM in 8051 is
(a) 4K bytes
(b) 16K bytes
(c) 32K bytes
(d) 8K bytes
137. Which of these registers cannot be decremented?
(a) Accumulator A
(b) DPTR
(c) B register
(d) SP
138. The lower byte of a 16 bit count is loaded in which of these registers?
(a) TLx
(b) THx
(c) TCON
(d) TMOD
139. The Serial Interrupt is enabled by which of these bits?
(a) ES (b) EA (c) ET (d) EX
140. This input pin is used for accessing Internal or External Memory and also generating the bus signals for micro processor operation of the micro controller?
(a) PSEN
(b) EA
(c) RESET
(d) ALE

141 . The 8255 PPI has
(a) 64 pins
(b) 16 pins
(c) 40 pins
(d) 28 pins
142 . If the control word 09BH is given to the control register of the 8255 PPI then which of these options are the condition of the ports
(a) BIT SET RESET
(b) ALL PORT‘s INPUT
(c) ALL PORT‘s OUTPUT
(d) PORT‘s A&B INPUT,PORT C OUTPUT
143 . If 8 seven segment led non multiplexed displays including the decimal point are connected. How many wires are needed to connect the segments(not including power supply wires)?
(a) 64 (b) 15 (c) 56 (d) 16
144 . What is the angle per step of a stepper motor in full step mode?
(a) 0.45 degrees
(b) 1.0 degrees
(c) 0.9 degrees
(d) 1.8 degrees
145 . In the A/D converter ADC 0808 which of these signals is used select an analog input channel to be digit is
(a) EOC (b) OE **(c) ALE** (d) SOC
146 . When an interrupt service is over, what is the instruction which makes 8086 continue executing the program it was executing before the Interrupt occurred
(a) JMP (b) ESC **(c) IRET** (d) RET
147 . What is the offset address of INT 20H in the vector interrupt table?
(a) 0050D
(b) 0050H
(c) 0080D
(d) 0080H
148 . The DOS function AH=9 INT 21H displays a message till which one of this character is reached
(a) × **(b) $** (c) # (d) *

149 . Which one of these BIOS services is used for Printer Operation?
(a) 15H (b) 14H **(c) 17H** (d) 16H
150. The PIC interrupt controller connects the single 8086 IRQ to how many IRQ input lines?
(a) 16 **(b) 8** (c) 4 (d) 32
151. The UART used as Serial Communication controller is
(a) 8257 (b) 8251 **(c) 8250** (d) 8259

152. How many registers does 8251 have?
(a) 3 **(b) 2** (c) 1 (d) 4
153. Which one of these instructions could be used for swapping in a sort program
(a) XCHG AX,DX
(b) SBB AX,DX
(c) SUB AX¡ DX
(d) MOV AX,BX
154. If B1=0 and B0=0 in the mode register of the 8251.The USART is working in which of these modes?
(a) Synchronous mode
(b) Asynchronous mode with BRF=16
(c) Asynchronous mode with BRF=64
(d) Asynchronous mode with BRF=1
155. The serial data transmission standards which uses differential Rx & Tx signals is which one of the following?
(a) RS232C
(b) RS449
(c) RS423A
(d) RS422A
156. The 8051 is a
(a) 16 bit micro controller
(b) 24 bit micro controller
(c) 8 bit micro controller
(d) 32 bit micro controller
157. Which of the registers R0 to R7 support indirect addressing?
(a) R6 & R7
(b) R0 & R6
(c) R0 & R7
(d) R0 & R1
158. Which of these modes the timer is a 16 bit timer?
(a) Mode 2
(b) Mode 3
(c) Mode 1
(d) Mode 0
159. The mode in which serial port operates like a shift register is which one of the following?
(a) Mode0
(b) Mode2
(c) Mode3
(d) Mode1
160. The instruction used to access data from external memory is which one of the following?
(a) MOV @
(b) MOV
(c) MOVC
(d) MOV X

161 . Group B port assignments of 8255 is
(a) Port B & Lower Port C
(b) Port A & Upper Port C
(c) Port B & Upper Port C
(d) Port A & Lower Port C
162 . If the control word 00 is given to the control register of the 8255 PPI then which of these options are the condition of the ports
(a) BIT SET RESET
(b) ALL PORT‘s OUTPUT
(c) PORT‘s A&B INPUT, PORT C OUTPUT
(d) ALL PORT‘s INPUT
163 . If 8 seven segment led non multiplexed displays including the decimal point are all lighted up and each of these is biased to carry 5mA.What is the total current drawn from the power supply when all the segments are ON?
(a) 280 mA
(b) 40 mA
(c) 80 mA
(d) 320 mA
164 . The electro mechanical device which produces mechanical displacement in proportion to an applied voltage is know as
(a) Solenoid
(b) Contactor
(c) Actuator
(d) Relay
165 . In a D/A converter the output voltage is
(a) Voltage
(b) Pulses
(c) current
(d) Ramp
166 . When an interrupt service is over, what is the instruction which makes 8086 continue executing the program it was executing before the Interrupt occurred
(a) JMP
(b) IRET
(c) ESC
(d) RET
167 . Which of these is called Break Point Interrupt?
(a) INT2
(b) INT1
(c) INT0
(d) INT3
168 . Which of these values in AH performs output on video display of Ascii character in DL?
(a) 17 (b) 01 (c) 09 **(d) 02**

169 . Which one of these BIOS services is used for Printer Operation?
(a) 16H (b) 14H (c) 15H **(d) 17H**
170. The PIC interrupt controller which indicates that an Interrupt is already in service is?
(a) ISR (b) IRR (c) IMR (d) PR
171. Communications which take place in either direction between two systems, but can only occur in one direction at a time is known as
(a) Complex
(b) Half Duplex
(c) Full Duplex
(d) Simplex
172. The 8251 gets its clock from which of these signals?
(a) SYNDET/BD
(b) C/D
(c) CLK
(d) TxC
173. Which combination of the flags indicate that the source is equal to the destination
(a) CY=1,Z=1
(b) CY=0,Z=0
(c) CY=1,Z=0
(d) CY=0,Z=1
174. Which of these bits in status register is used to clear all errors?
(a) ER (b) EN (c) RTS (d) IR
175. The serial data transmission standards which uses differential Rx& Tx signals is which one of the following?
(a) RS449
(b) RS232C
(c) RS423A
(d) RS422A
176. The size of the internal data RAM in 8051 is
(a) 128 bytes
(b) 256 bytes
(c) 32 bytes
(d) 1K bytes
177.Which of these register of 8051 is 16 bit?
(a) DPTR
(b) A
(c) PSW
(d) SP
178. Counter or timer operation is chosen in which of these registers?
(a) TLx
(b) THx
(c) TCON

(d) TMOD

179. Which of these modes is a standard UART mode?

(a) Mode3

(b) Mode0

(c) Mode1

(d) Mode2

180. The instruction used to access data from external memory is which one of the following?

(a) MOV @

(b) MOV

(c) MOV X

(d) MOVC

181 . Which of these is Double Hand Shake

(a) STB only

(b) STB followed by ACK

(c) STB followed by ACK and removal of ACK

(d) ACK followed by STB

182 . At power on or after reset the ports are in which of these conditions

(a) PORT A=INPUT,PORT B=INPUT, PORT C=INPUT

(b) PORT A=INPUT,PORT B=INPUT, PORT C=OUT PUT

(c) PORT A=INPUT, PORT B=OUTPUT, PORT C=OUTPUT

(d) PORT A=OUTPUT, PORT B=INPUT, PORT C=OUTPUT

183 . Which of these Peripheral ICs is made for interfacing Keyboards and Multiplexed LED displays?

(a) 8259 (b) 8237 (c) 8251 **(d) 8279**

184 . Which of these motors works like a digital system

(a) DC MOTOR

(b) STEPPER MOTOR

(c) INDUCTION MOTOR

(d) SYNCHRONOUS MOTOR

185 . An 8 bit A/D converter with a reference voltage of 5 Volts will be able to read a lowest non zero voltage of

(a) 20 mV **(b) 19.5 mV** (c) 21 mV (d) 19 mV

186 . The third response of the 8086 to an interrupt is which one of the following?

(a) Decrements the SP& pushes current CS contents

(b) Resets Trap Flag

(c) It disables interrupt flag

(d) Decrements the SP by 2& pushes flag register

187 . Which of these is called for doing Single Stepping?

(a) INT3

(b) INT1

(c) INT0

(d) INT2

188 . In the single character display using Interrupt 21H, DL is the register which contains the character to display. If DL contains 41H which of these characters is displayed?

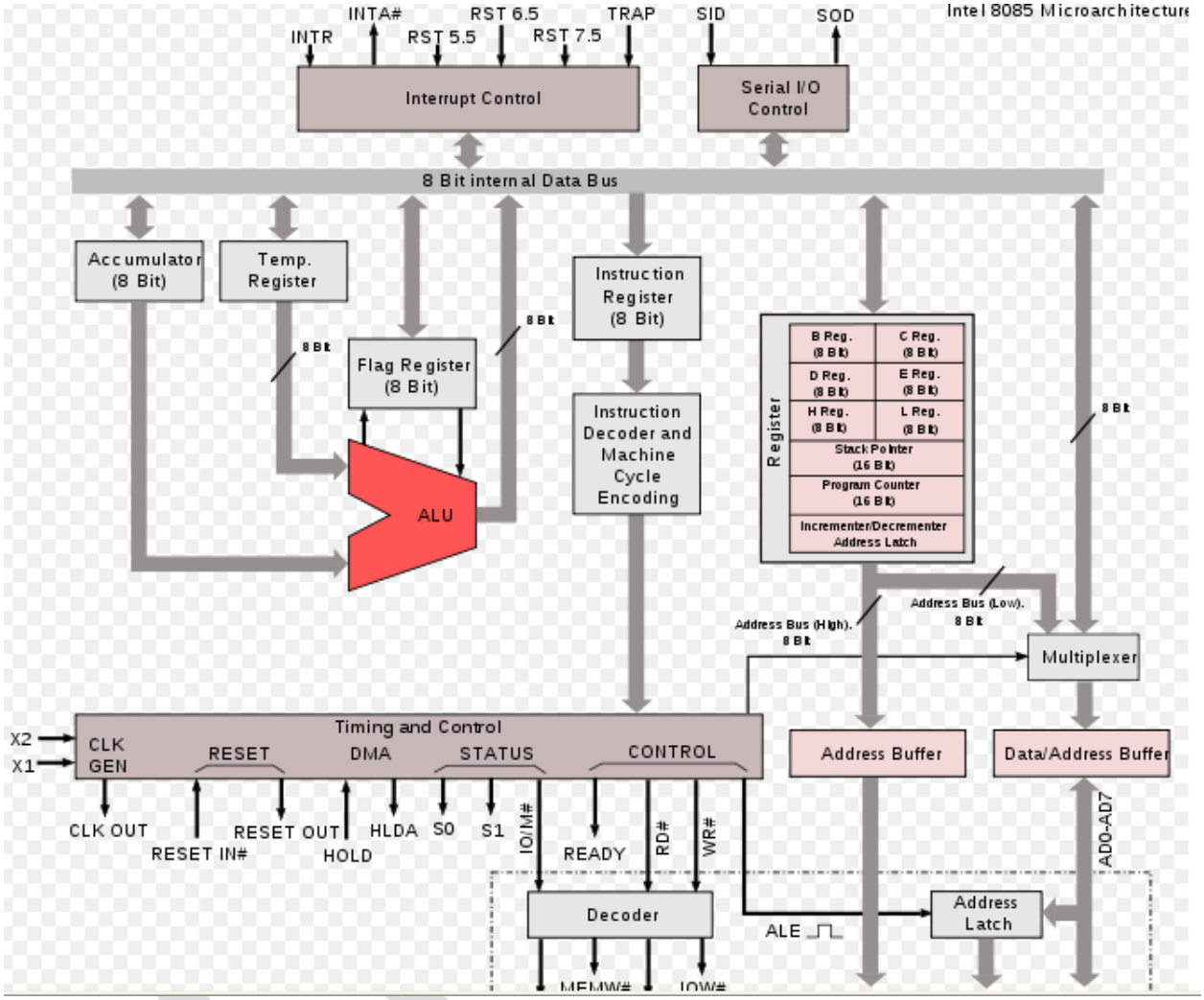
(a) A (b) B (c) b (d) a
189 . For which one of these values of AL does the Video interrupt sets 80X25 color video mode?
(a) 03H (b) 00H (c) 02H (d) 01H
190. Which of these Initialisation Command Word indicates whether PIC is being used in Single or Cascaded mode?
(a) ICW2
(b) ICW4
(c) ICW1
(d) ICW3
191. The device from which data originates or terminates is called
(a) DT R
(b) DCE
(c) DTE
(d) DSR
192. Serial data is received by the DTE from the DCE through which one of these signals?
(a) RxD
(b) CD
(c) RTS
(d) CTS
193. Given the program shown below, What is the expression evaluated by this program?
P db 5
Q db 6
R db 7
S db 6
MOV AL,[P]
MOV BL,[Q]
ADD AL,BL
MUL AL
ADD AL,R
MOV CL,S
DIV CL
(a) ((P+Q)2+R/S
(b) (P+Q2+R)/s
(c) ((P +Q)2+R)/S
(d) (P+(Q+R)2)/S
194. Which of these parameters are set by the S2& S1 bits?
(a) BAUD RATE FACTOR
(b) PARITY
(c) CHARACTER LENGTH
(d) NUMBER OF STOP BITS
195. The data signals used by USB are of which type of the following?
(a) Uni-Phase
(b) Quad Phase

(c) Bi - Phase
(d) Tri-phase
196. The 8051 is a
(a) 24 bit micro controller
(b) 16 bit micro controller
(c) 8 bit micro controller
(d) 32 bit micro controller
197. Which of these registers cannot be decremented?
(a) B register
(b) SP
(c) Accumulator A
(d) DPTR
198. Which of these modes the timer is a 16 bit timer?
(a) Mode 3
(b) Mode 0
(c) Mode 2
(d) Mode 1
199. The mode in which serial port operates like a shift register is which one of the following?
(a) Mode 0
(b) Mode 1
(c) Mode 3
(d) Mode 2
200. Which of these signals is used for enabling the external ROM ?
(a) EA
(b) RESET
(c) ALE
(d) PSEN

20.Tutorial classes

Tutorial-1

8085 ARCHITECTURE



The 8085 is a conventional von Neumann design based on the Intel 8080. Unlike the 8080 it does not multiplex state signals onto the data bus, but the 8-bit data bus was instead multiplexed with the lower part of the 16-bit address bus to limit the number of pins to 40. Pin #40 is used for the power supply (+5v) and pin #20 for ground. Pin #39 is used as the hold pin. Pins #15 to #8 are generally used for address buses. The processor was designed using nMOS circuitry and the later "H" versions were implemented in Intel's enhanced nMOS process called HMOS, originally developed for fast static RAM products. Only a 5 Volt supply is needed, like competing processors and unlike the 8080. The 8085 uses approximately 6,500 transistors.[1]

The 8085 incorporates the functions of the 8224 (clock generator) and the 8228 (system controller), increasing the level of integration. A downside compared to similar contemporary designs (such as the Z80) was the fact that the buses required demultiplexing; however, address

latches in the Intel 8155, 8355, and 8755 memory chips allowed a direct interface, so an 8085 along with these chips was almost a complete system.

The 8085 has extensions to support new interrupts, with three maskable interrupts (RST 7.5, RST 6.5 and RST 5.5), one non-maskable interrupt (TRAP), and one externally serviced interrupt (INTR). The RST n.5 interrupts refer to actual pins on the processor, a feature which permitted simple systems to avoid the cost of a separate interrupt controller.

Like the 8080, the 8085 can accommodate slower memories through externally generated wait states (pin 35, READY), and has provisions for Direct Memory Access (DMA) using HOLD and HLDA signals (pins 39 and 38). An improvement over the 8080 was that the 8085 can itself drive a piezoelectric crystal directly connected to it, and a built in clock generator generates the internal high amplitude two-phase clock signals at half the crystal frequency (a 6.14 MHz crystal would yield a 3.07 MHz clock, for instance)

Programming model

The 8085 is a binary compatible follow up on the 8080, using the same basic instruction set as the 8008 (developed by Computer Terminal Corporation). Only a few minor instructions were new to the 8085 above the 8080 set.

Registers

The processor has seven 8-bit registers named A, B, C, D, E, H, and L, where A is the 8-bit accumulator and the other six can be used as independent byte-registers or as three 16-bit register pairs, BC, DE, and HL, depending on the particular instruction. Some instructions use HL as a (limited) 16-bit accumulator. As in the 8080, the contents of the memory address pointed to by HL could be accessed as pseudoregister M. It also has a 16-bit stack pointer to memory (replacing the 8008's internal stack), and a 16-bit program counter. HL pair is called the primary data pointers.

Commands/instructions

As in many other 8-bit processors, all instructions are encoded in a single byte (including register-numbers, but excluding immediate data), for simplicity. Some of them are followed by one or two bytes of data, which could be an immediate operand, a memory address, or a port number. Like larger processors, it has CALL and RET instructions for multi-level procedure calls and returns (which can be conditionally executed, like jumps) and instructions to save and restore any 16-bit register-pair on the machine stack. There are also eight one-byte call instructions (RST) for subroutines located at the fixed addresses 00h, 08h, 10h,...,38h. These were intended to be supplied by external hardware in order to invoke a corresponding interrupt-service routine, but are also often employed as fast system calls. The most sophisticated command was XTHL, which is used for exchanging the register pair HL with the value stored at the address indicated by the stack pointer.

8-bit instructions

Most 8-bit operations work on the 8-bit accumulator (the A register). For two operand 8-bit operations, the other operand can be either an immediate value, another 8-bit register, or a memory cell addressed by the 16-bit register pair HL. Direct copying is supported between any two 8-bit registers and between any 8-bit register and a HL-addressed memory cell. Due to the regular encoding of the MOV-instruction (using a quarter of available opcode space) there are redundant codes to copy a register into itself (MOV B,B, for instance), which are of little use, except for delays. However, what would have been a copy from the HL-addressed cell into itself (i.e., MOV M,M) instead encodes the HLT instruction, halting execution until an external reset or interrupt occurred.

16-bit operations

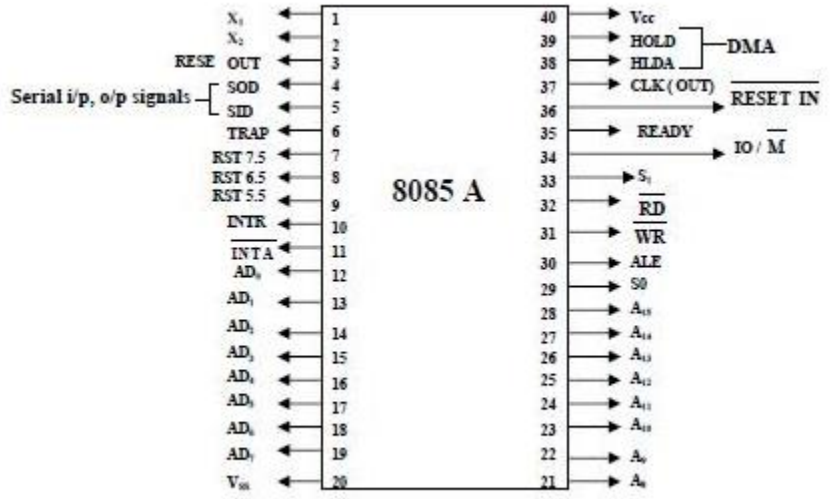
Although the 8085 is an 8-bit processor, it also has some 16-bit operations. Any of the three 16-bit register pairs (BC, DE, HL) or SP could be loaded with an immediate 16-bit value (using LXI), incremented or decremented (using INX and DCX), or added to HL (using DAD). LHLD loaded HL from directly-addressed memory and SHLD stored HL likewise. The XCHG operation exchanges the values of HL and DE. Adding HL to itself performs a 16-bit arithmetical left shift with one instruction. The only 16 bit instruction that affects any flag was DAD (adding HL to BC, DE, HL or SP), which updates the carry flag to facilitate 24-bit or larger additions and left shifts (for a floating point mantissa for instance). Adding the stack pointer to HL is useful for indexing variables in (recursive) stack frames. A stack frame can be allocated using DAD SP and SPHL, and a branch to a computed pointer can be done with PCHL. These abilities make it feasible to compile languages such as PL/M, Pascal, or C with 16-bit variables and produce 8085 machine code.

Subtraction and bitwise logical operations on 16 bits is done in 8-bit steps. Operations that have to be implemented by program code (subroutine libraries) included comparisons of signed integers as well as multiply and divide.

Input/output scheme

The 8085 supported up to 256 input/output (I/O) ports, accessed via dedicated I/O instructions—taking port addresses as operands. This I/O mapping scheme was regarded as an advantage, as it freed up the processor's limited address space

8085 PIN DESCRIPTION



Pin Diagram of 8085

Single + 5V Supply
4 Vectored Interrupts (One is Non Maskable)
Serial In/Serial Out Port
Decimal, Binary, and Double Precision Arithmetic
Direct Addressing Capability to 64K bytes of memory
The Intel 8085A is a new generation, complete 8 bit parallel central processing unit (CPU). The 8085A uses a multiplexed data bus. The address is split between the 8bit address bus and the 8bit data bus. Figures are at the end of the document.

Pin Description

The following describes the function of each pin:

A6 - A1s (Output 3 State)

Address Bus**;** The most significant 8 bits of the memory address or the 8 bits of the I/0 address,3 stated during Hold and Halt modes.

AD0 - 7 (Input/Output 3state)

Multiplexed Address/Data Bus; Lower 8 bits of the memory address (or I/0 address) appear on the bus during the first clock cycle of a machine state. It then becomes the data bus during the second and third clock cycles. 3 stated during Hold and Halt modes.

ALE (Output)

Address Latch Enable: It occurs during the first clock cycle of a machine state and enables the address to get latched into the on chip latch of peripherals. The falling edge of ALE is set to

guarantee setup and hold times for the address information. ALE can also be used to strobe the status information. ALE is never 3stated.

SO, S1 (Output)

Data Bus Status. Encoded status of the bus cycle:

S1 S0

O O HALT

0 1 WRITE

1 0 READ

1 1 FETCH

S1 can be used as an advanced R/W status.

RD (Output 3state)

READ; indicates the selected memory or 1/0 device is to be read and that the Data Bus is available for the data transfer.

WR (Output 3state)

WRITE; indicates the data on the Data Bus is to be written into the selected memory or 1/0 location. Data is set up at the trailing edge of WR. 3stated during Hold and Halt modes.

READY (Input)

If Ready is high during a read or write cycle, it indicates that the memory or peripheral is ready to send or receive data. If Ready is low, the CPU will wait for Ready to go high before completing the read or write cycle.

HOLD (Input)

HOLD; indicates that another Master is requesting the use of the Address and Data Buses. The CPU, upon receiving the Hold request. will relinquish the use of buses as soon as the completion of the current machine cycle. Internal processing can continue.

The processor can regain the buses only after the Hold is removed. When the Hold is acknowledged, the Address, Data, RD, WR, and IO/M lines are 3stated.

HLDA (Output)

HOLD ACKNOWLEDGE; indicates that the CPU has received the Hold request and that it will relinquish the buses in the next clock cycle. HLDA goes low after the Hold request is removed. The CPU takes the buses one half clock cycle after HLDA goes low.

INTR (Input)

INTERRUPT REQUEST; is used as a general purpose interrupt. It is sampled only during the next to the last clock cycle of the instruction. If it is active, the Program Counter (PC) will be inhibited from incrementing and an INTA will be issued. During this cycle a RESTART or CALL instruction can be inserted to jump to the interrupt service routine. The INTR is enabled and disabled by software. It is disabled by Reset and immediately after an interrupt is accepted.

INTA (Output)

INTERRUPT ACKNOWLEDGE; is used instead of (and has the same timing as) RD during the Instruction cycle after an INTR is accepted. It can be used to activate the 8259 Interrupt chip or some other interrupt port.

RST 5.5

RST 6.5 - (Inputs)

RST 7.5

RESTART INTERRUPTS; These three inputs have the same timing as I NTR except they cause an internal RESTART to be automatically inserted.

RST 7.5 ~~ Highest Priority

RST 6.5

RST 5.5 o Lowest Priority

The priority of these interrupts is ordered as shown above. These interrupts have a higher priority than the INTR.

TRAP (Input)

Trap interrupt is a nonmaskable restart interrupt. It is recognized at the same time as INTR. It is unaffected by any mask or Interrupt Enable. It has the highest priority of any interrupt.

RESET IN (Input)

Reset sets the Program Counter to zero and resets the Interrupt Enable and HLDA flipflops. None of the other flags or registers (except the instruction register) are affected The CPU is held in the reset condition as long as Reset is applied.

RESET OUT (Output)

Indicates CPlJ is being reset. Can be used as a system RESET. The signal is synchronized to the processor clock.

X1, X2 (Input)

Crystal or R/C network connections to set the internal clock generator X1 can also be an external clock input instead of a crystal. The input frequency is divided by 2 to give the internal operating frequency.

CLK (Output)
Clock Output for use as a system clock when a crystal or R/ C network is used as an input to the CPU. The period of CLK is twice the X1, X2 input period.

IO/M (Output)
IO/M indicates whether the Read/Write is to memory or l/O Tristated during Hold and Halt modes.

SID (Input)
Serial input data line The data on this line is loaded into accumulator bit 7 whenever a RIM instruction is executed.

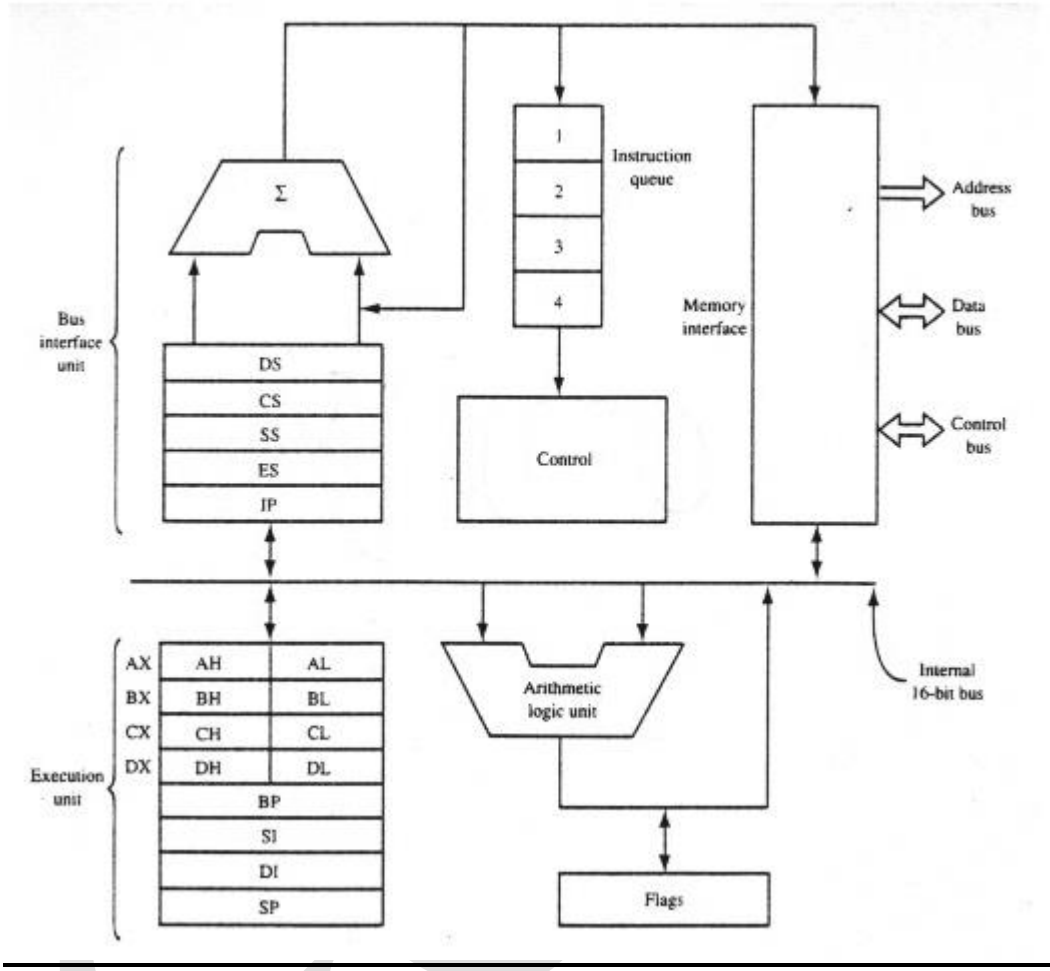
SOD (output)
Serial output data line. The output SOD is set or reset as specified by the SIM instruction.

Vcc
+5 volt supply.

Vss
Ground Reference.

Tutorial-2

8086 Architecture



Memory

Program, data and stack memories occupy the same memory space. The total addressable memory size is 1MB KB. As the most of the processor instructions use 16-bit pointers the processor can effectively address only 64 KB of memory. To access memory outside of 64 KB the CPU uses special segment registers to specify where the code, stack and data 64 KB segments are positioned within 1 MB of memory (see the "Registers" section below).

16-bitpointersanddataarestoreas:
address:low-orderbyte address+1: high-order byte

32-bitaddressesarestoredin"segment:offset"formaas:
address:low-orderbyteofsegment

address+1high-orderbyteofsegment
address+2:low-orderbyteofoffset
address+3: high-order byte of offset

Physical memory address pointed by segment:offset pair is calculated as:

address = (* 16) + <offset>

Program memory - program can be located anywhere in memory. Jump and call instructions can be used for short jumps within currently selected 64 KB code segment, as well as for far jumps anywhere within 1 MB of memory. All conditional jump instructions can be used to jump within approximately +127 - -127 bytes from current instruction.

Data memory - the 8086 processor can access data in any one out of 4 available segments, which limits the size of accessible memory to 256 KB (if all four segments point to different 64 KB blocks). Accessing data from the Data, Code, Stack or Extra segments can be usually done by prefixing instructions with the DS:, CS:, SS: or ES: (some registers and instructions by default may use the ES or SS segments instead of DS segment).

Word data can be located at odd or even byte boundaries. The processor uses two memory accesses to read 16-bit word located at odd byte boundaries. Reading word data from even byte boundaries requires only one memory access.

Stack memory can be placed anywhere in memory. The stack can be located at odd memory addresses, but it is not recommended for performance reasons (see "Data Memory" above).

Reserved locations:

- 0000h - 03FFh are reserved for interrupt vectors. Each interrupt vector is a 32-bit pointer in format segment:offset.
- FFFF0h - FFFFFh - after RESET the processor always starts program execution at the FFFF0h address.

Interrupts

The processor has the following interrupts:

INTR is a maskable hardware interrupt. The interrupt can be enabled/disabled using STI/CLI instructions or using more complicated method of updating the FLAGS register with the help of the POPF instruction. When an interrupt occurs, the processor stores FLAGS register into stack, disables further interrupts, fetches from the bus one byte representing interrupt type, and jumps to interrupt processing routine address of which is stored in location 4 * <interrupt type>. Interrupt processing routine should return with the IRET instruction.

NMI is a non-maskable interrupt. Interrupt is processed in the same way as the INTR interrupt. Interrupt type of the NMI is 2, i.e. the address of the NMI processing routine is stored in location 0008h. This interrupt has higher priority then the maskable interrupt.

Software interrupts can be caused by:

- INT instruction - breakpoint interrupt. This is a type 3 interrupt.
- INT <interrupt number> instruction - any one interrupt from available 256 interrupts.
- INTO instruction - interrupt on overflow
- Single-step interrupt - generated if the TF flag is set. This is a type 1 interrupt. When the CPU processes this interrupt it clears TF flag before calling the interrupt processing routine.
- Processor exceptions: divide error (type 0), unused opcode (type 6) and escape opcode (type 7).

Software interrupt processing is the same as for the hardware interrupts.

I/O ports

65536 8-bit I/O ports. These ports can be also addressed as 32768 16-bit I/O ports.

Registers

Most of the registers contain data/instruction offsets within 64 KB memory segment. There are four different 64 KB segments for instructions, stack, data and extra data. To specify where in 1 MB of processor memory these 4 segments are located the 8086 microprocessor uses four segment registers:

Code segment (CS) is a 16-bit register containing address of 64 KB segment with processor instructions. The processor uses CS segment for all accesses to instructions referenced by instruction pointer (IP) register. CS register cannot be changed directly. The CS register is automatically updated during far jump, far call and far return instructions.

Stack segment (SS) is a 16-bit register containing address of 64KB segment with program stack. By default, the processor assumes that all data referenced by the stack pointer (SP) and base pointer (BP) registers is located in the stack segment. SS register can be changed directly using POP instruction.

Data segment (DS) is a 16-bit register containing address of 64KB segment with program data. By default, the processor assumes that all data referenced by general registers (AX, BX, CX, DX) and index register (SI, DI) is located in the data segment. DS register can be changed directly using POP and LDS instructions.

Extra segment (ES) is a 16-bit register containing address of 64KB segment, usually with program data. By default, the processor assumes that the DI register references the ES segment in string manipulation instructions. ES register can be changed directly using POP and LES instructions.

It is possible to change default segments used by general and index registers by prefixing instructions with a CS, SS, DS or ES prefix.

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

Accumulator register consists of 2 8-bit registers AL and AH, which can be combined together and used as a 16-bit register AX. AL in this case contains the low-order byte of the word, and AH contains the high-order byte. Accumulator can be used for I/O operations and string manipulation.

Base register consists of 2 8-bit registers BL and BH, which can be combined together and used as a 16-bit register BX. BL in this case contains the low-order byte of the word, and BH contains the high-order byte. BX register usually contains a data pointer used for based, based indexed or register indirect addressing.

Count register consists of 2 8-bit registers CL and CH, which can be combined together and used as a 16-bit register CX. When combined, CL register contains the low-order byte of the word, and CH contains the high-order byte. Count register can be used as a counter in string manipulation and shift/rotate instructions.

Data register consists of 2 8-bit registers DL and DH, which can be combined together and used as a 16-bit register DX. When combined, DL register contains the low-order byte of the word, and DH contains the high-order byte. Data register can be used as a port number in I/O operations. In integer 32-bit multiply and divide instruction the DX register contains high-order word of the initial or resulting number.

The following registers are both general and index registers:

Stack Pointer (SP) is a 16-bit register pointing to program stack.

Base Pointer (BP) is a 16-bit register pointing to data in stack segment. BP register is usually used for based, based indexed or register indirect addressing.

Source Index (SI) is a 16-bit register. SI is used for indexed, based indexed and register indirect addressing, as well as a source data address in string manipulation instructions.

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

Other registers:

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

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.

Instruction Set

Instruction set of Intel 8086 processor consists of the following instructions:

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

Addressing modes

Implied - the data value/data address is implicitly associated with the instruction.

Register - references the data in a register or in a register pair.

Immediate - the data is provided in the instruction.

Direct - the instruction operand specifies the memory address where data is located.

Register indirect - instruction specifies a register containing an address, where data is located. This addressing mode works with SI, DI, BX and BP registers.

Based - 8-bit or 16-bit instruction operand is added to the contents of a base register (BX or BP), the resulting value is a pointer to location where data resides.

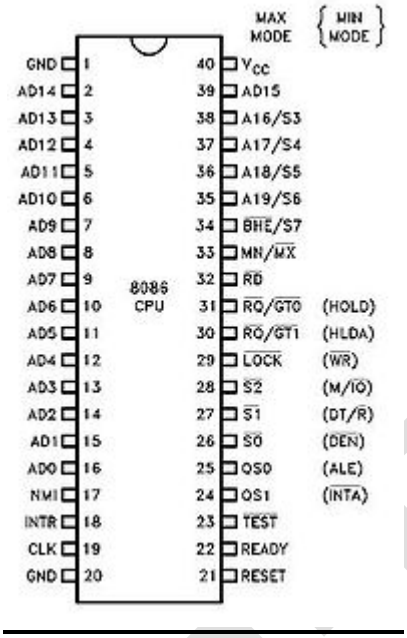
Indexed - 8-bit or 16-bit instruction operand is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.

Based Indexed - the contents of a base register (BX or BP) is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.

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

Tutorial-3

8086 PIN CONFUGURATION



- The Microprocessor 8086 is a 16-bit CPU available in different clock rates and packaged in a 40 pin CERDIP or plastic package.
- The 8086 operates in single processor or multiprocessor configuration to achieve high performance. The pins serve a particular function in minimum mode (single processor mode) and other function in maximum mode configuration (multiprocessor mode).
- The 8086 signals can be categorised in three groups.
    - The first are the signal having common functions in minimum as well as maximum mode.
    - The second are the signals which have special functions for minimum mode
    - The third are the signals having special functions for maximum mode.
- The following signal descriptions are common for both modes.
- **AD15-AD0 :** These are the time multiplexed memory I/O address and data lines.
    - Address remains on the lines during T1 state, while the data is available on the data bus during T2, T3, Tw and T4. These lines are active high and float to a tristate during interrupt acknowledge and local bus hold acknowledge cycles.
- **A19/S6,A18/S5,A17/S4,A16/S3 :** These are the time multiplexed address and status lines.
    - During T1 these are the most significant address lines for memory operations.
    - During I/O operations, these lines are low.
    - During memory or I/O operations, status information is available on those lines for T2,T3,Tw and T4.

- The status of the interrupt enable flag bit is updated at the beginning of each clock cycle.
- The S4 and S3 combinely indicate which segment register is presently being used for memory accesses as in below fig.
- These lines float to tri-state off during the local bus hold acknowledge. The status line S6 is always low.
- The address bit are separated from the status bit using latches controlled by the ALE signal.

S4	S3	Indication
0	0	Alternate Data
0	1	Stack
1	0	Code or None
1	1	Data
0	0	Whole word
0	1	Upper byte from or to even address
1	0	Lower byte from or to even address

- **BHE/S7 :** The bus high enable is used to indicate the transfer of data over the higher order (D15-D8) data bus as shown in table. It goes low for the data transfer over D15-D8 and is used to derive chip selects of odd address memory bank or peripherals. BHE is low during T1 for read, write and interrupt acknowledge cycles, whenever a byte is to be transferred on higher byte of data bus. The status information is available during T2, T3 and T4. The signal is active low and tristated during hold. It is low during T1 for the first pulse of the interrupt acknowledge cycle.

- **RD – Read :** This signal on low indicates the peripheral that the processor is performing memory or I/O read operation. RD is active low and shows the state for T2, T3, Tw of any read cycle. The signal remains tristated during the hold acknowledge.

- **READY :** This is the acknowledgement from the slow device or memory that they have completed the data transfer. The signal made available by the devices is synchronized by the 8284A clock generator to provide ready input to the 8086. the signal is active high.

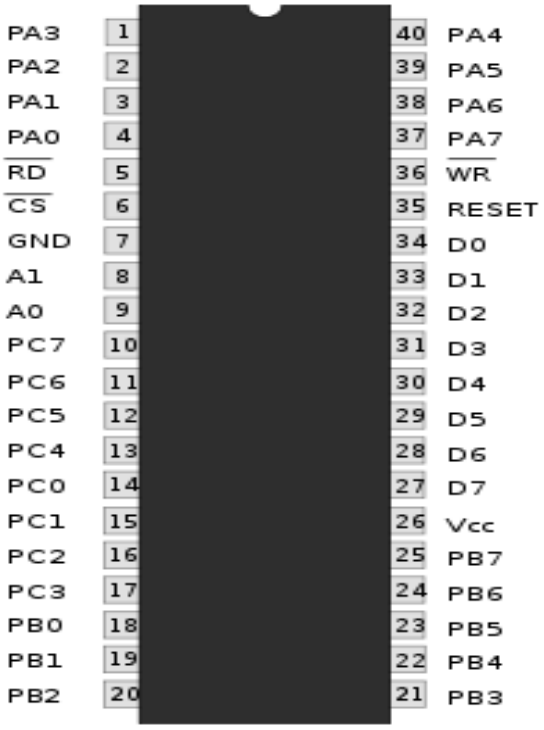
- **INTR-Interrupt Request :** This is a triggered input. This is sampled during the last clock cycles of each instruction to determine the availability of the request. If any interrupt request is pending, the processor enters the interrupt acknowledge cycle. This can be internally masked by resulting the interrupt enable flag. This signal is active high and internally synchronized.

- **TEST :** This input is examined by a 'WAIT' instruction. If the TEST pin goes low, execution will continue, else the processor remains in an idle state. The input is synchronized internally during each clock cycle on leading edge of clock.

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

Tutorial-4

8255:



Pinout of i8255

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. It is made in DIP 40 and PLCC 44 pins encapsulated versions.

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 Input/Output) from the Motorola 6800 family, the Western Design Center WDC 65C21, an enhanced 6520, and many others.

The 8255 is widely used not only in many microcomputer/microcontroller systems especially Z-80 based, home computers such as SV-328 and all MSX, but also in the system

board of the best known original IBM-PC, PC/XT, PC/jr, etc. and clones, along with numerous homebuilt computer computers such as the N8VEM.

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.

Functional

The 8255 has 24 input/output pins in all. These are divided into three 8-bit ports. Port A and port B can be used as 8-bit input/output ports. Port C can be used as an 8-bit input/output port or as two 4-bit input/output ports or to produce handshake signals for ports A and B.

The three ports are further grouped as follows:

1. Group A consisting of port A and upper part of port C.
2. Group B consisting of port B and lower part of port C.

Eight data lines (D0 - D7) are available (with an 8-bit data buffer) to read/write data into the ports or control register under the status of the "¬**RD**" (pin 5) and ¬**WR**" (pin 36), which are active low signals for read and write operations respectively. The address lines A1 and A0 allow to successively access any one of the ports or the control register as listed below:

A1	**A0**	**Function**
0	0	port A
0	1	port B
1	0	port C
1	1	control register

The control signal "'¬CS**" (pin 6) is used to enable the 8255 chip. It is an active low signal, i.e., when ¬CS = '0'** , the 8255 is enabled. The **RESET** input (pin 35) is connected to a system (like 8085, 8086, etc.) reset line so that when the system is reset, all the ports are initialized as input lines. This is done to prevent 8255 and/or any peripheral connected to it, from being destroyed due to mismatch of ports. This is explained as follows. Suppose an input device is connected to 8255 at port A. If from the previous operation, port A is initialized as an output port and if 8255 is not reset before using the current configuration,

then there is a possibility of damage of either the input device connected or 8255 or both since both 8255 and the device connected will be sending out data.

The control register or the control logic or the command word register is an 8-bit register used to select the modes of operation and input/output designation of the ports.

Operational modes of 8255

There are two main operational modes of 8255:

1. Input/output mode
2. Bit set/reset mode

Input/output mode

There are three types of the input/output mode which are as follows:

Mode 0

In this mode, the ports can be used for simple input/output operations without handshaking. If both port A and B are initialized in mode 0, the two halves of port C can be either used together as an additional 8-bit port, or they can be used as individual 4-bit ports. Since the two halves of port C are independent, they may be used such that one-half is initialized as an input port while the other half is initialized as an output port. The input/output features in mode 0 are as follows:

1. O/p are latched.
2. I/p are buffered not latched.
3. Port do not have handshake or interrupt capability.

Mode 1

When we wish to use port A or port B for handshake (strobed) input or output operation, we initialise that port in mode 1 (port A and port B can be initilalised to operate in different modes, i.e., for e.g., port A can operate in mode 0 and port B in mode 1). Some of the pins of port C function as handshake lines.

For port B in this mode (irrespective of whether is acting as an input port or output port), PC0, PC1 and PC2 pins function as handshake lines.

If port A is initialised as mode 1 input port, then, PC3, PC4 and PC5 function as handshake signals. Pins PC6 and PC7 are available for use as input/output lines.

The mode 1 which supports handshaking has following features:

1. Two ports i.e. port A and B can be use as 8-bit i/o port.

2. Each port uses three lines of port c as handshake signal and remaining two signals can be function as i/o port.
3. Interrupt logic is supported.
4. Input and Output data are latched.

Mode 2

Only group A can be initialised in this mode. Port A can be used for *bidirectional handshake* data transfer. This means that data can be input or output on the same eight lines (PA0 - PA7). Pins PC3 - PC7 are used as handshake lines for port A. The remaining pins of port C (PC0 - PC2) can be used as input/output lines if group B is initialised in mode 0. In this mode, the 8255 may be used to extend the system bus to a slave microprocessor or to transfer data bytes to and from a floppy disk controller.

Bit set/reset (BSR) mode

In this mode only port B can be used (as an output port). Each line of port C (PC0 - PC7) can be set/reset by suitably loading the command word register.no effect occurs in input-output mode. The individual bits of port c can be set or reset by sending the signal OUT instruction to the control register.

Control word format

Input/output mode format

- The figure shows the control word format in the input/output mode. This mode is selected by making **D7 = '1'** .
- **D0, D1, D3, D4** are for lower port C, port B, upper port C and port A respectively. When D0 or D1 or D3 or D4 are *"SET"*, the corresponding ports act as input ports. For e.g., if D0 = D4 = '1', then lower port C and port A act as input ports. If these bits are *"RESET"*, then the corresponding ports act as output ports. For e.g., if D1 = D3 = '0', then port B and upper port C act as output ports.
- **D2** is used for mode selection for group B (Port B and Lower Port C). When D2 = '0', mode 0 is selected and when D2 = '1', mode 1 is selected.
- **D5, D6** are used for mode selection for group A (Upper Port C and Port A). The format is as follows:

D6	D5	mode
0	0	0

0	1	1
1	x	2

Example: *If port B and upper port C have to be initialised as input ports and lower port C and port A as ouput ports (all in mode 0), what is the control word?*

1. Since it is an input/ouput mode, **D7 = '1'**.
2. Mode selection bits, **D2, D5, D6 are all '0'** for mode 0 operation.
3. Port B should operate as input port, hence, **D1 = '1'**.
4. Upper port C should also be an input port, hence, **D3 = '1'**.
5. Port A has to operate output port, hence, **D4 = '0'**.
6. Lower port C should also operate as output port, hence, **D0 = '0'**.

Applying the corresponding values to the format in input/output mode, we get the control word as **"8A (hex)"**

BSR mode format

- The figure shows the control word format in BSR mode. This mode is selected by making **D7='0'**.
- **D0** is used for bit set/reset. When D0= '1', the port C bit selected (*selection of a port C bit is shown in the next point*) is **SET**, when D0 = '0', the port C bit is **RESET**.
- **D1, D2, D3** are used to select a particular port C bit whose value may be altered using D0 bit as mentioned above. The selection of the port C bits are done as follows:

D3	D2	D1	Bit/pin of port C selected
0	0	0	PC0
0	0	1	PC1
0	1	0	PC2
0	1	1	PC3
1	0	0	PC4
1	0	1	PC5
1	1	0	PC6
1	1	1	PC7

- **D4, D5, D6** are not used.

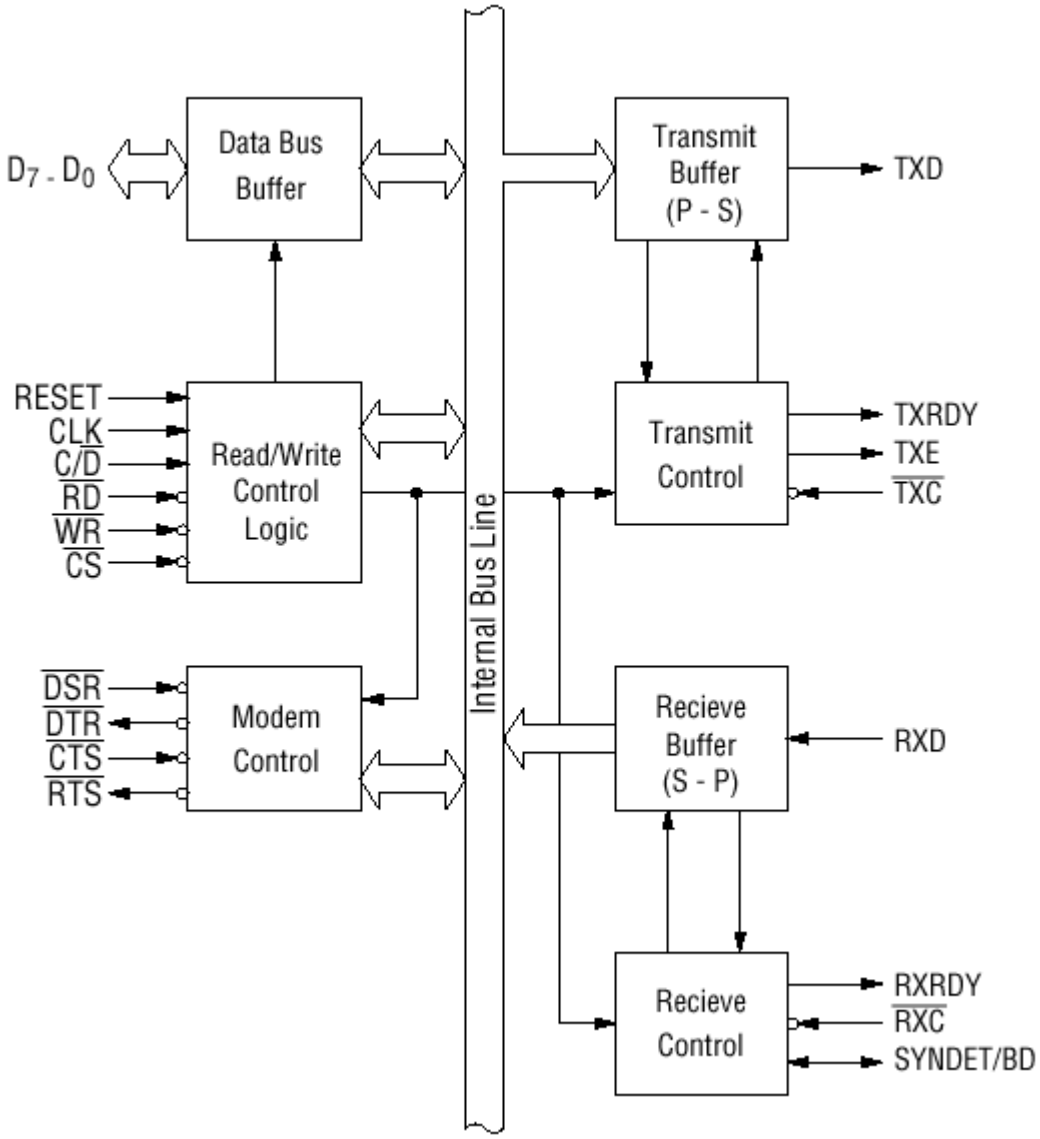
Example: *If the 5th bit (PC5) of port C has to be "SET", then what is the control word?*

1. Since it is BSR mode, **D7 = '0'**.
2. Since D4, D5, D6 are not used, assume them to be '**0**'.
3. PC5 has to be selected, hence, **D3 = '1', D2 = '0', D1 = '1'**.
4. PC5 has to be set, hence, **D0 = '1'**.

Tutorial-5

8251 Universal Synchronous Asynchronous Receiver Transmitter (USART):

The 8251 is a USART (Universal Synchronous Asynchronous Receiver Transmitter) for serial data communication. As a peripheral device of a microcomputer system, the 8251 receives parallel data from the CPU and transmits serial data after conversion. This device also receives serial data from the outside and transmits parallel data to the CPU after conversion.



Block diagram of the 8251 USART (Universal Synchronous Asynchronous Receiver Transmitter)

The 8251 functional configuration is programed by software. Operation between the 8251 and a CPU is executed by program control. Table 1 shows the operation between a CPU and the device.

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	
1	×	×	×	Data Bus 3-State
0	×	1	1	Data Bus 3-State
0	1	0	1	Status → CPU
0	1	1	0	Control Word ← CPU
0	0	0	1	Data → CPU
0	0	1	0	Data ← CPU

Table 1 Operation between a CPU and 8251

Control Words

There are two types of control word.

1. Mode instruction (setting of function)

2. Command (setting of operation)

1) Mode Instruction

Mode instruction is used for setting the function of the 8251. Mode instruction will be in "wait for write" at either internal reset or external reset. That is, the writing of a control word after resetting will be recognized as a "mode instruction."

Items set by mode instruction are as follows:

• Synchronous/asynchronous mode

• Stop bit length (asynchronous mode)

• Character length

• Parity bit

• Baud rate factor (asynchronous mode)

• Internal/external synchronization (synchronous mode)

• Number of synchronous characters (Synchronous mode)

The bit configuration of mode instruction is shown in Figures 2 and 3. In the case of synchronous mode, it is necessary to write one-or two byte sync characters. If sync characters were written, a function will be set because the writing of sync characters constitutes part of mode instruction.

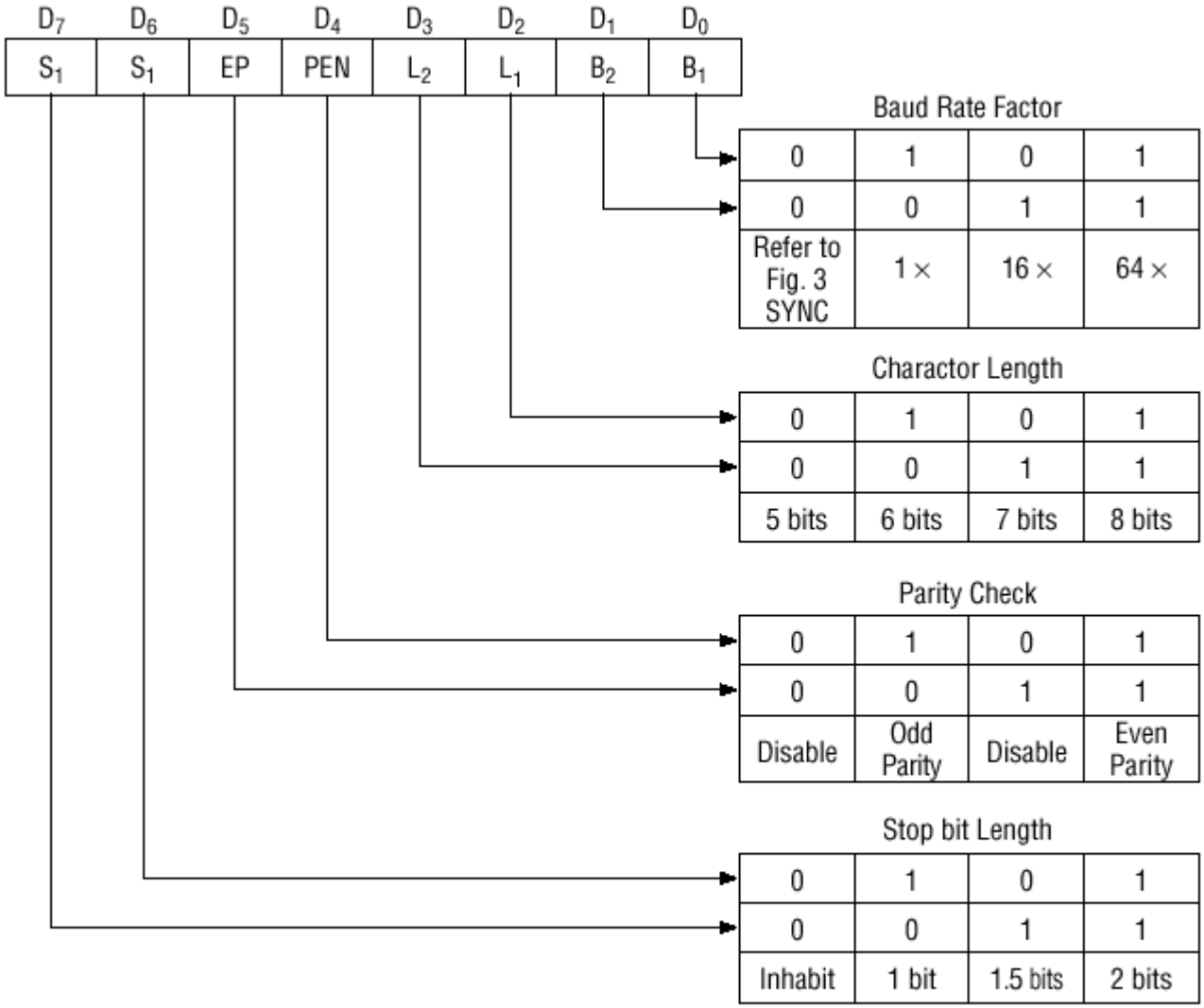


Fig. 2 Bit Configuration of Mode Instruction (Asynchronous)

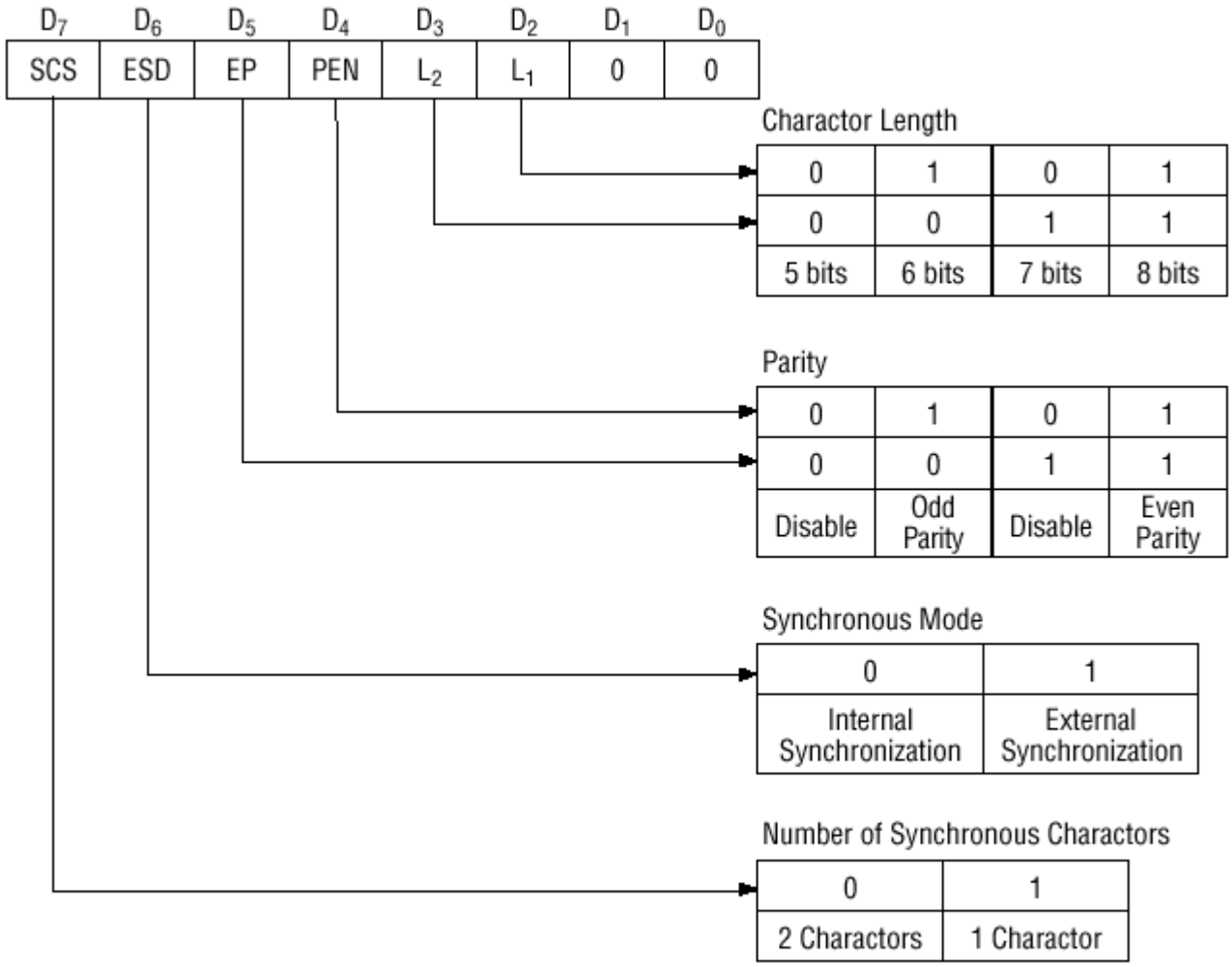


Fig. 3 Bit Configuration of Mode Instruction (Synchronous)

2) Command

Command is used for setting the operation of the 8251. It is possible to write a command whenever necessary after writing a mode instruction and sync characters.

Items to be set by command are as follows:

- Transmit Enable/Disable
- Receive Enable/Disable
- DTR, RTS Output of data.
- Resetting of error flag.

• Sending to break characters

• Internal resetting

• Hunt mode (synchronous mode)

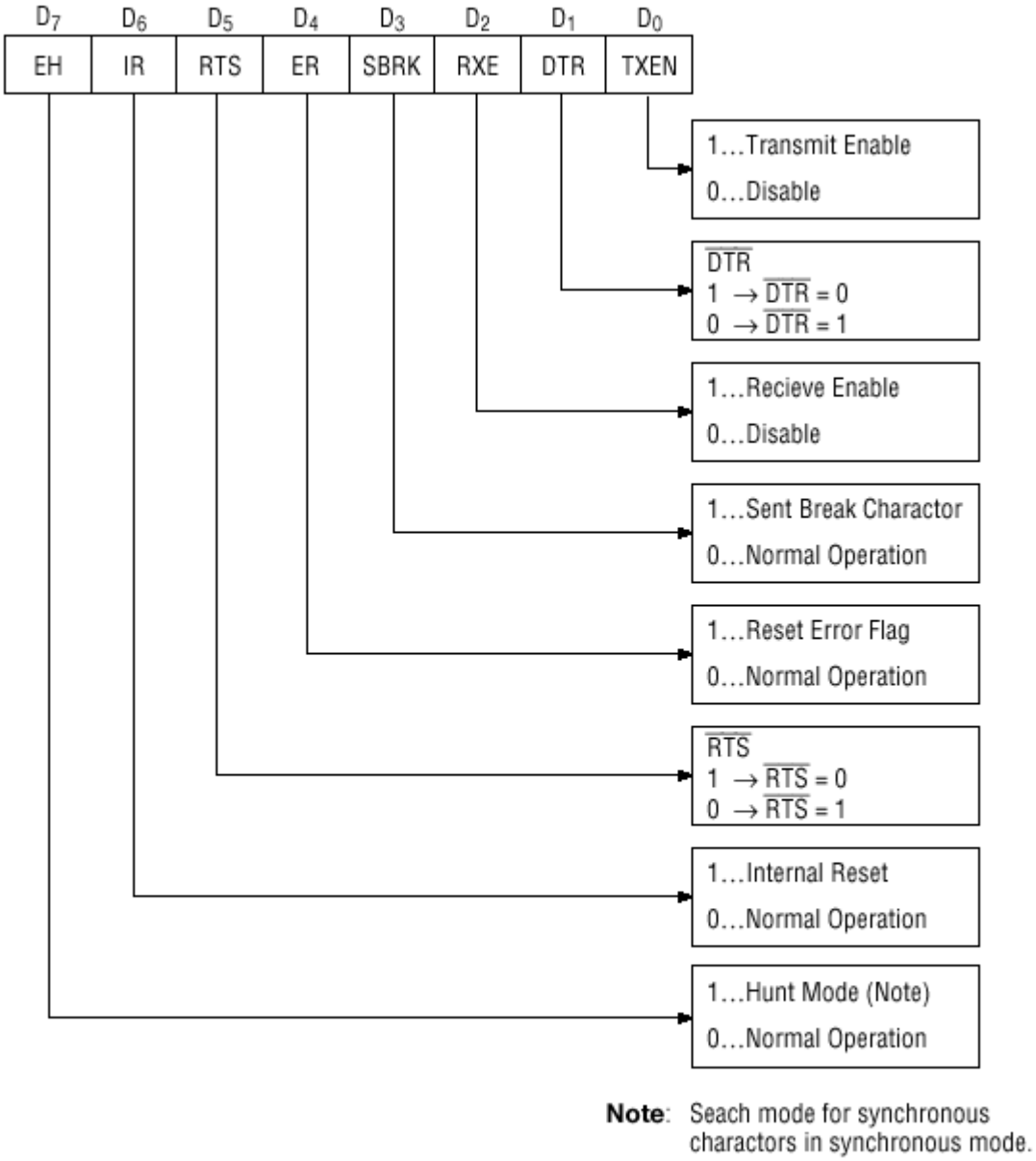


Fig. 4 Bit Configuration of Command

Status Word

It is possible to see the internal status of the 8251 by reading a status word. The bit configuration of status word is shown in Fig. 5.

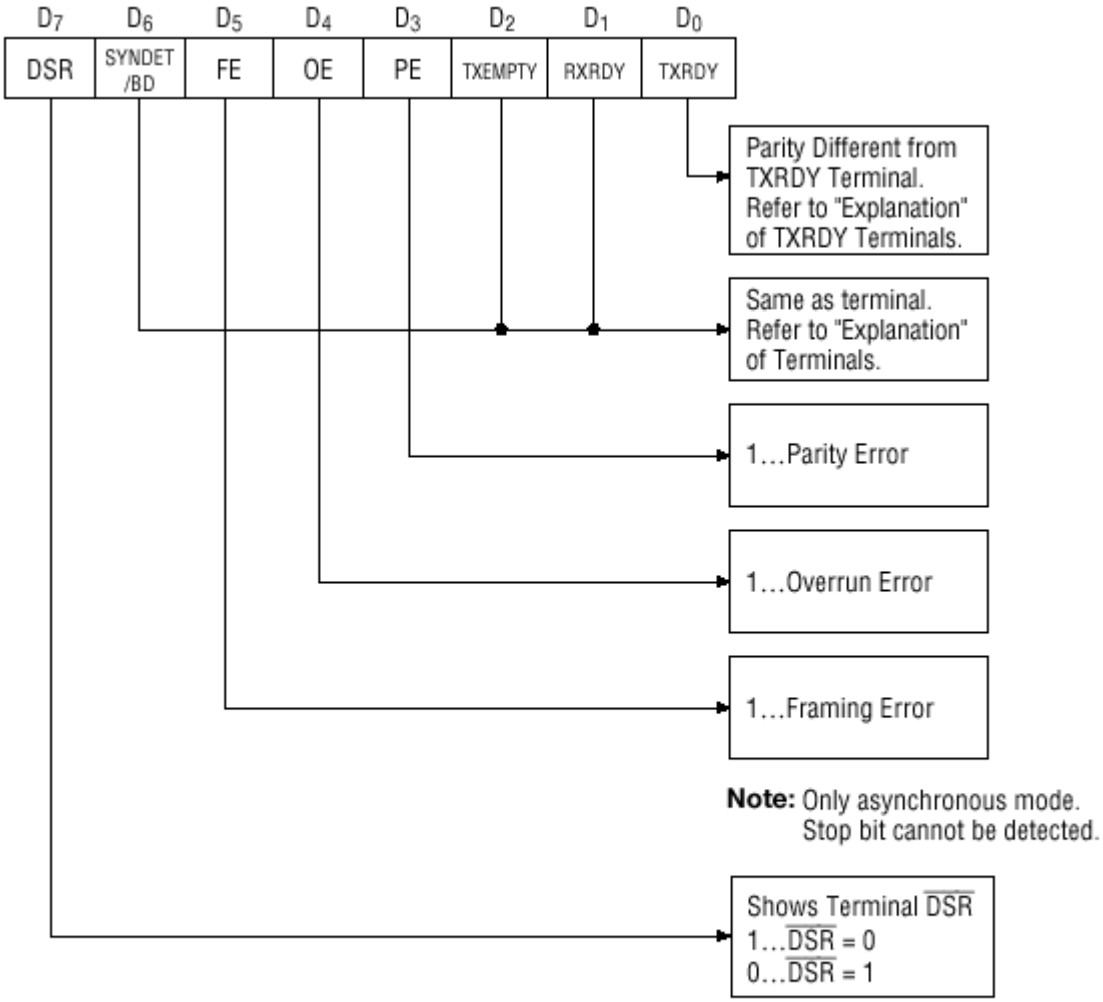


Fig. 5 Bit Configuration of Status Word

Pin Description

D 0 to D 7 (I/O terminal)

This is bidirectional data bus which receive control words and transmits data from the CPU and sends status words and received data to CPU.

RESET (Input terminal)

A "High" on this input forces the 8251 into "reset status." The device waits for the writing of "mode instruction." The min. reset width is six clock inputs during the operating status of CLK.

CLK (Input terminal)

CLK signal is used to generate internal device timing. CLK signal is independent of RXC or TXC. However, the frequency of CLK must be greater than 30 times the RXC and TXC at Synchronous mode and Asynchronous "x1" mode, and must be greater than 5 times at Asynchronous "x16" and "x64" mode.

WR (Input terminal)

This is the "active low" input terminal which receives a signal for writing transmit data and control words from the CPU into the 8251.

RD (Input terminal)

This is the "active low" input terminal which receives a signal for reading receive data and status words from the 8251.

C/D (Input terminal)

This is an input terminal which receives a signal for selecting data or command words and status words when the 8251 is accessed by the CPU. If C/D = low, data will be accessed. If C/D = high, command word or status word will be accessed.

CS (Input terminal)

This is the "active low" input terminal which selects the 8251 at low level when the CPU accesses. Note: The device won't be in "standby status"; only setting CS = High.

TXD (output terminal)

This is an output terminal for transmitting data from which serial-converted data is sent out. The device is in "mark status" (high level) after resetting or during a status when transmit is disabled. It is also possible to set the device in "break status" (low level) by a command.

TXRDY (output terminal)

This is an output terminal which indicates that the 8251is ready to accept a transmitted data character. But the terminal is always at low level if CTS = high or the device was set in "TX disable status" by a command. Note: TXRDY status word indicates that transmit data character is receivable, regardless of CTS or command. If the CPU writes a data character, TXRDY will be reset by the leading edge or WR signal.

TXEMPTY (Output terminal)

This is an output terminal which indicates that the 8251 has transmitted all the characters and had no data character. In "synchronous mode," the terminal is at high level, if transmit data characters are no longer remaining and sync characters are automatically transmitted. If the CPU writes a data character, TXEMPTY will be reset by the leading edge of WR signal. Note : As the transmitter is disabled by setting CTS "High" or command, data written before disable will be sent out. Then TXD and TXEMPTY will be "High". Even if a data is written after disable, that data is not sent out and TXE will be "High".After the transmitter is enabled, it sent out. (Refer to Timing Chart of Transmitter Control and Flag Timing)

TXC (Input terminal)

This is a clock input signal which determines the transfer speed of transmitted data. In "synchronous mode," the baud rate will be the same as the frequency of TXC. In "asynchronous mode", it is possible to select the baud rate factor by mode instruction. It can be 1, 1/16 or 1/64 the TXC. The falling edge of TXC sifts the serial data out of the 8251.

RXD (input terminal)

This is a terminal which receives serial data.

RXRDY (Output terminal)

This is a terminal which indicates that the 8251 contains a character that is ready to READ. If the CPU reads a data character, RXRDY will be reset by the leading edge of RD signal. Unless the CPU reads a data character before the next one is received completely, the preceding data will be lost. In such a case, an overrun error flag status word will be set.

RXC (Input terminal)

This is a clock input signal which determines the transfer speed of received data. In "synchronous mode," the baud rate is the same as the frequency of RXC. In "asynchronous mode," it is possible to select the baud rate factor by mode instruction. It can be 1, 1/16, 1/64 the RXC.

SYNDET/BD (Input or output terminal)

This is a terminal whose function changes according to mode. In "internal synchronous mode." this terminal is at high level, if sync characters are received and synchronized. If a status word is read, the terminal will be reset. In "external synchronous mode, "this is an input terminal. A "High" on this input forces the 8251 to start receiving data characters.

In "asynchronous mode," this is an output terminal which generates "high level"output upon the detection of a "break" character if receiver data contains a "low-level" space between the stop bits of two continuous characters. The terminal will be reset, if RXD is at high level. After Reset is active, the terminal will be output at low level.

DSR (Input terminal)

This is an input port for MODEM interface. The input status of the terminal can be recognized by the CPU reading status words.

DTR (Output terminal)

This is an output port for MODEM interface. It is possible to set the status of DTR by a command.

CTS (Input terminal)

This is an input terminal for MODEM interface which is used for controlling a transmit circuit. The terminal controls data transmission if the device is set in "TX Enable" status by a command. Data is transmitable if the terminal is at low level.

RTS (Output terminal)

This is an output port for MODEM interface. It is possible to set the status RTS by a command.

TUTORIAL-6

The Intel **8259** is a family of Programmable Interrupt Controllers (PICs) designed and developed for use with the Intel 8085 and Intel 8086 8-bit and 16-bit microprocessors. The family originally consisted of the 8259, 8259A, and 8259B PICs, though a number of manufacturers make a wide range of compatible chips today. The 8259 acts as a multiplexer, combining multiple interrupt input sources into a single interrupt output to interrupt a single device.

History

The 8259 was included in the original PC introduced in 1981 and maintained by the PC/XT when introduced in 1983. A second 8259 was added with the introduction of the PC/AT. The 8259 has coexisted with the Intel APIC Architecture since its introduction in Symmetric Multi-Processor PCs. Modern PCs have since begun to completely phase out the use of the 8259 family in favor of the exclusive use of the Intel APIC Architecture. However, while not anymore a separate chip, the 8259 interface is still provided by the Southbridge chipset on modern x86 motherboards.The main connectors on an 8259 are as follows: eight interrupt input request lines named IRQ0 through IRQ7, an interrupt request output line named INTR, interrupt acknowledgment line named INTA, D0 through D7 for communicating the interrupt level or vector offset. Other connectors include CAS0 through CAS2 for cascading between 8259s.Up to eight slave 8259s may be cascaded to a *master* 8259 to provide up to 64 IRQs. 8259s are cascaded by connecting the INT line of one *slave* 8259 to the IRQ line of one *master* 8259.

There are three registers, an Interrupt Mask Register (IMR), an Interrupt Request Register (IRR), and an In-Service Register (ISR). The IRR maintains a mask of the current interrupts that are pending acknowledgement, the ISR maintains a mask of the interrupts that are pending an EOI, and the IMR maintains a mask of interrupts that should not be sent an acknowledgement.

End Of Interrupt (EOI) operations support specific EOI, non-specific EOI, and auto-EOI. A specific EOI specifies the IRQ level it is acknowledging in the ISR. A non-specific EOI resets the IRQ level in the ISR. Auto-EOI resets the IRQ level in the ISR immediately after the interrupt is acknowledged.

Edge and level interrupt trigger modes are supported by the 8259A.

Fixed priority and rotating priority modes are supported.

The 8259 may be configured to work with an 8085 or an 8086. On the 8086, the interrupt controller will provide an interrupt number on the data bus when an interrupt occurs; instead, the interrupt cycle of the 8085 will issue three bytes on the data bus (corresponding to a CALL instruction in the 8085 instruction set).

The 8259A provides additional functionality compared to the 8259 (in particular buffered mode and level-triggered mode) and is upward compatible with it. It is believed that the NEC Corporation created the 8259A and the 8259B may be nothing more than a mnemonic for the second 8259A introduced in the PC/AT

Programming Considerations
DOS and Windows

Programming an 8259 in conjunction with DOS and Microsoft Windows has introduced a number of confusing issues for the sake of backwards compatibility, which extends as far back as the original PC introduced in 1981.

The first issue is more or less the root of the second issue. DOS device drivers are expected to send a non-specific EOI to the 8259s when they finish servicing their device. This prevents the use of any of the 8259's other EOI modes in DOS, and excludes the differentiation between device interrupts rerouted from the master 8259 to the slave 8259.

The second issue deals with the use of IRQ2 and IRQ9 from the introduction of a slave 8259 in the PC/AT. The slave 8259's INT output is connected to the master's IR2. The IRQ2 line of the ISA bus, originally connected to this IR2, was rerouted to IR1 of the slave. Thus the

old IRQ2 line now generates IRQ9 in the CPU. To allow backwards compatibility with DOS device drivers that still set up for IRQ2, a handler is installed by the BIOS for IRQ9 that redirects interrupts to the original IRQ2 handler.

On the PC, the BIOS (and thus also DOS) traditionally maps the master 8259 interrupt requests (IRQ0-IRQ7) to interrupt vector offset 8 (INT08-INT0F) and the slave 8259 (in PC/AT and later) interrupt requests (IRQ8-IRQ15) to interrupt vector offset 112 (INT70-INT77). This was done despite the first 32 (INT00-INT1F) interrupt vectors being reserved by the processor for internal exceptions (this was ignored for the design of the PC for some reason). Because of the reserved vectors for exceptions most other operating systems map (at least the master) 8259 IRQs (if used on a platform) to another interrupt vector base offset.

Other Operating Systems

Since most other operating systems allow for changes in device driver expectations, other 8259 modes of operation, such as Auto-EOI, may be used. This is especially important for modern x86 hardware in which a significant amount of time may be spent on I/O address space delay when communicating with the 8259s. This also allows a number of other optimizations in synchronization, such as critical sections, in a multiprocessor x86 system with 8259s.

Edge/Level Triggered Mode

Since the ISA bus does not support level triggered interrupts, level triggered mode may not be used for interrupts connected to ISA devices. This means that on PC/XT, PC/AT, and compatible systems the 8259 must be programmed for edge triggered mode. On MCA systems, devices use level triggered interrupts and the interrupt controller is hardwired to always work in level triggered mode. On newer EISA, PCI, and later systems the Edge/Level Control Registers (ELCRs) control the mode per IRQ line, effectively making the mode of the 8259 irrelevant for such systems with ISA buses. The ELCR is programmed by the BIOS at system startup for correct operation.

The ELCRs are located 0x4d0 and 0x4d1 in the x86 I/O address space. They are 8-bits wide, each bit corresponding to an IRQ from the 8259s. When a bit is set, the IRQ is in level triggered mode; otherwise, the IRQ is in edge triggered mode.

Spurious Interrupts

The 8259 generates spurious interrupts in response to a number of conditions.

The first is an IRQ line being de-asserted before it is acknowledged. This may occur due to noise on the IRQ lines. In edge triggered mode, the noise must maintain the line in the low state for 100nS. When the noise diminishes, a pull-up resistor returns the IRQ line to high, thus generating a false interrupt. In level triggered mode, the noise may cause a high signal level on the systems INTR line. If the system sends an acknowledgment request, the 8259 has nothing to resolve and thus sends an IRQ7 in response. This first case will generate spurious IRQ7's.

A similar case can occur when the 8259 unmask and the IRQ input de-assertion are not properly synchronized. In many systems, the IRQ input is de-asserted by an I/O write, and the processor doesn't wait until the write reaches the I/O device. If the processor continues and unmasks the 8259 IRQ before the IRQ input is de-asserted, the 8259 will assert INTR again. By the time the processor recognizes this INTR and issues an acknowledgment to read the IRQ from the 8259, the IRQ input may be de-asserted, and the 8259 returns a spurious IRQ7.

The second is the master 8259's IRQ2 is active high when the slave 8259's IRQ lines are inactive on the falling edge of an interrupt acknowledgment. This second case will generate spurious IRQ15's, but is very rare.

PC/XT and PC/AT

The PC/XT ISA system had one 8259 controller, while PC/AT and later systems had two 8259 controllers, master and slave. IRQ0 through IRQ7 are the master 8259's interrupt lines, while IRQ8 through IRQ15 are the slave 8259's interrupt lines. The actual names on the pins on an 8259 are IR0 through IR7. IRQ0 through IRQ15 are the names of the ISA bus's lines to which the 8259's are historically attached.

- Master 8259
    - IRQ0 – Intel 8253 or Intel 8254 Programmable Interval Timer, aka the system timer
    - IRQ1 – Intel 8042 keyboard controller

    - IRQ2 – *not assigned in PC/XT; cascaded to slave 8259 INT line in PC/AT*
    - IRQ3 – 8250 UART serial port COM2 and COM4
    - IRQ4 – 8250 UART serial port COM1 and COM3
    - IRQ5 – hard disk controller in PC/XT; Intel 8255 parallel port LPT2 in PC/AT
    - IRQ6 – Intel 82072A floppy disk controller
    - IRQ7 – Intel 8255 parallel port LPT1 / spurious interrupt

- Slave 8259 (PC/AT and later only)
    - IRQ8 – real-time clock (RTC)
    - IRQ9 – no common assignment
    - IRQ10 – no common assignment
    - IRQ11 – no common assignment
    - IRQ12 – Intel 8042 PS/2 mouse controller
    - IRQ13 – math coprocessor
    - IRQ14 – hard disk controller 1
    - IRQ15 – hard disk controller 2

Initially IRQ7 was a common choice for the use of a sound card, but later IRQ5 was used when it was found that IRQ7 would interfere with the printer port (LPT1). The serial ports are frequently disabled to free an IRQ line for another device.

IRQ2/9 is the traditional interrupt line for an MPU-401 MIDI port, but this conflicts with the ACPI system control interrupt (SCI is hardwired to IRQ9 on Intel chipsets); this means ISA MPU-401 cards with a hardwired IRQ 2/9, and MPU-401 device drivers with a hardcoded IRQ 2/9, cannot be used in interrupt-driven mode on a system with ACPI enabled.

TUTORIAL-7

The Intel 8051 microcontroller is one of the most popular general purpose microcontrollers in use today. The success of the Intel 8051 spawned a number of clones which are collectively referred to as the MCS-51 family of microcontrollers, which includes chips from vendors such as Atmel, Philips, Infineon, and Texas Instruments

The Intel 8051 is an 8-bit microcontroller which means that most available operations are limited to 8 bits. There are 3 basic "sizes" of the 8051: Short, Standard, and Extended. The Short and Standard chips are often available in DIP (dual in-line package) form, but the Extended 8051 models often have a different form factor, and are not "drop-in compatible". All these things are called 8051 because they can all be programmed using 8051 assembly language, and they all share certain features (although the different models all have their own special features).

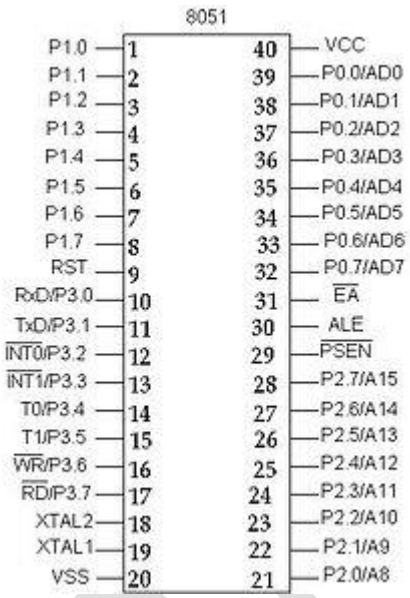
Some of the features that have made the 8051 popular are:

- 64 KB on chip program memory.
- 128 bytes on chip data memory(RAM).
- 4 reg banks.
- 128 user defined software flags.
- 8-bit data bus
- 16-bit address bus
- 32 general purpose registers each of 8 bits
- 16 bit timers (usually 2, but may have more, or less).
- 3 internal and 2 external interrupts.
- Bit as well as byte addressable RAM area of 16 bytes.
- Four 8-bit ports, (short models have two 8-bit ports).
- 16-bit program counter and data pointer.
- 1 Microsecond instruction cycle with 12 MHz Crystal.

8051 models may also have a number of special, model-specific features, such as UARTs, ADC, OpAmps, etc...

Typical applications

8051 chips are used in a wide variety of control systems, telecom applications, robotics as well as in the automotive industry. By some estimations, 8051 family chips make up over 50% of the embedded chip market.



Pin diagram of the 8051 DIP

Basic Pins

PIN 9: PIN 9 is the reset pin which is used reset the microcontroller's internal registers and ports upon starting up. (Pin should be held high for 2 machine cycles.)

PINS 18 & 19: The 8051 has a built-in oscillator amplifier hence we need to only connect a crystal at these pins to provide clock pulses to the circuit.

PIN 40 and 20: Pins 40 and 20 are VCC and ground respectively. The 8051 chip needs +5V 500mA to function properly, although there are lower powered versions like the Atmel 2051 which is a scaled down version of the 8051 which runs on +3V.

PINS 29, 30 & 31: As described in the features of the 8051, this chip contains a built-in flash memory. In order to program this we need to supply a voltage of +12V at pin 31. If external memory is connected then PIN 31, also called EA/VPP, should be connected to ground to indicate the presence of external memory. PIN 30 is called ALE (address latch enable), which is used when multiple memory chips are connected to the controller and only one of them needs to be selected.We will deal with this in depth in the later chapters.

PIN 29 is called PSEN. This is "program store enable". In order to use the external memory it is required to provide the low voltage (0) on both PSEN and EA pins.

Ports

There are 4 8-bit ports: P0, P1, P2 and P3.

PORT P1 (Pins 1 to 8): The port P1 is a general purpose input/output port which can be used for a variety of interfacing tasks. The other ports P0, P2 and P3 have dual roles or additional functions associated with them based upon the context of their usage.

PORT P3 (Pins 10 to 17): PORT P3 acts as a normal IO port, but Port P3 has additional functions such as, serial transmit and receive pins, 2 external interrupt pins, 2 external counter inputs, read and write pins for memory access.

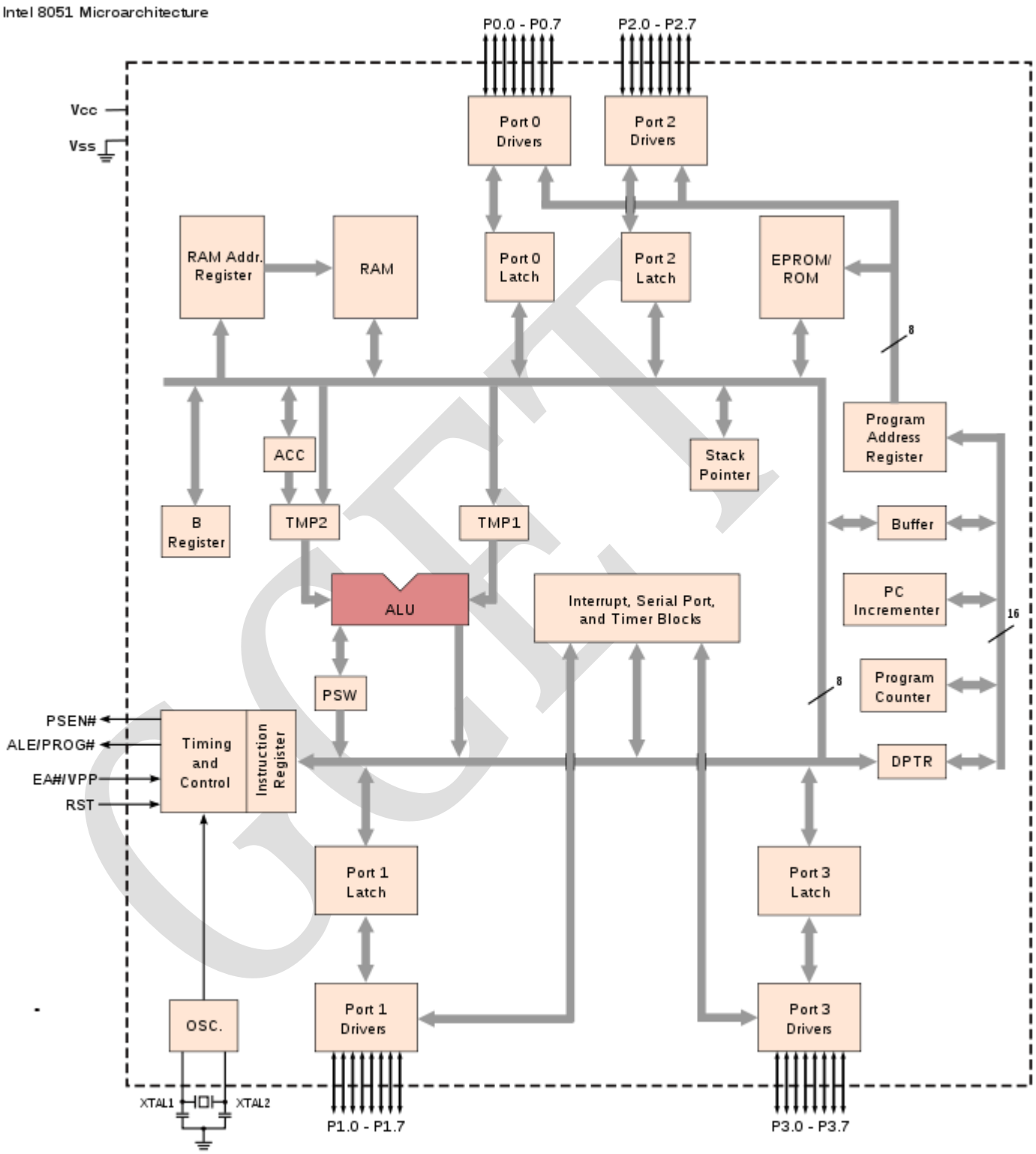
PORT P2 (pins 21 to 28): PORT P2 can also be used as a general purpose 8 bit port when no external memory is present, but if external memory access is required then PORT P2 will act as an address bus in conjunction with PORT P0 to access external memory. PORT P2 acts as A8-A15, as can be seen from fig 1.1

PORT P0 (pins 32 to 39) PORT P0 can be used as a general purpose 8 bit port when no external memory is present, but if external memory access is required then PORT P0 acts as a multiplexed address and data bus that can be used to access external memory in conjunction with PORT P2. P0 acts as AD0-AD7, as can be seen from fig 1.1

Oscillator Circuits

The 8051 requires the existence of an external oscillator circuit. The oscillator circuit usually runs around 12MHz, although the 8051 (depending on which specific model) is capable of running at a maximum of 40MHz. Each machine cycle in the 8051 is 12 clock cycles, giving an effective cycle rate at 1MHz (for a 12MHz clock) to 3.33MHz (for the maximum 40MHz clock).

Internal Architecture



Internal schematics of the 8051.

Data and Program Memory

The 8051 Microprocessor can be programmed in PL/M, 8051 Assembly, C and a number of other high-level languages. Many compilers even have support for compiling C++ for an 8051.

Program memory in the 8051 is read-only, while the data memory is considered to be read/write accessible. When stored on EEPROM or Flash, the program memory can be rewritten when the microcontroller is in the special programmer circuit.

Program Start Address

The 8051 starts executing program instructions from address 0000 in the program memory.

Direct Memory

The 8051 has 256 bytes of internal addressable RAM, although only the first 128 bytes are available for general use by the programmer. The first 128 bytes of RAM (from 0x00 to 0x7F) are called the **Direct Memory**, and can be used to store data.

Special Function Register

The **Special Function Register** (SFR) is the upper area of addressable memory, from address 0x80 to 0xFF. A, B, PSW, DPTR are called SFR.This area of memory cannot be used for data or program storage, but is instead a series of memory-mapped ports and registers. All port input and output can therefore be performed by memory **mov** operations on specified addresses in the SFR. Also, different status registers are mapped into the SFR, for use in checking the status of the 8051, and changing some operational parameters of the 8051.

General Purpose Registers

The 8051 has 4 selectable banks of 8 addressable 8-bit registers, R0 to R7. This means that there are essentially 32 available general purpose registers, although only 8 (one bank) can be directly accessed at a time. To access the other banks, we need to change the current bank number in the flag status register.

A and B Registers

The A register is located in the SFR memory location 0xE0. The A register works in a similar fashion to the AX register of x86 processors. The A register is called the **accumulator**, and by default it receives the result of all arithmetic operations. The B register is used in a similar manner, except that it can receive the extended answers from

the multiply and divide operations. When not being used for multiplication and Division, the B register is available as an extra general-purpose register.

TUTORIAL-8

Atmel AVR RISC microcontroller:

The **AVR** is a modified Harvard architecture 8-bit RISC single chip microcontroller which was developed by Atmel in 1996. The AVR was one of the first microcontroller families to use on-chip flash memory for program storage, as opposed to one-time programmable ROM, EPROM, or EEPROM used by other microcontrollers at the time. However, it is commonly accepted that AVR stands for **A**lf (Egil Bogen) and **V**egard (Wollan)'s **R**isc processor" the use of "AVR" generally refers to the 8-bit RISC line of Atmel AVR Microcontrollers.

Among the first of the AVR line was the AT90S8515, which in a 40-pin DIP package has the same pinout as an 8051 microcontroller, including the external multiplexed address and data bus. The polarity of the RESET line was opposite (8051's having an active-high RESET, while the AVR has an active-low RESET) but other than that, the pinout was identical.

overview

The AVR is a modified Harvard architecture machine where program and data are stored in separate physical memory systems that appear in different address spaces, but having the ability to read data items from program memory using special instructions.

Basic families

AVRs are generally classified into five broad groups:

- **tinyAVR** — the ATtiny series
  - 0.5–8 kB program memory
  - 6–32-pin package
  - Limited peripheral set
- **megaAVR** — the ATmega series
  - 4–256 kB program memory
  - 28–100-pin package
  - Extended instruction set (Multiply instructions and instructions for handling larger program memories)
  - Extensive peripheral set
- **XMEGA** — the ATxmega series
  - 16–384 kB program memory

- 44–64–100-pin package (A4, A3, A1)
- Extended performance features, such as DMA, "Event System", and cryptography support.
- Extensive peripheral set with DACs

- **Application-specific AVR**
  - megaAVRs with special features not found on the other members of the AVR family, such as LCD controller, USB controller, advanced PWM, CAN etc.
- **FPSLIC™ (AVR with FPGA)**
  - FPGA 5K to 40K gates
  - SRAM for the AVR program code, unlike all other AVRs
  - AVR core can run at up to 50 MHz [5]
- **32-bit AVRs**

  In 2006 Atmel released microcontrollers based on the new, 32-bit, AVR32 architecture. They include SIMD and DSP instructions, along with other audio and video processing features. This 32-bit family of devices is intended to compete with the ARM based processors. The instruction set is similar to other RISC cores, but is not compatible with the original AVR or any of the various ARM cores.

Architecture

Flash, EEPROM, and SRAM are all integrated onto a single chip, removing the need for external memory in most applications. Some devices have a parallel external bus option to allow adding additional data memory or memory-mapped devices. Almost all devices (except the smallest TinyAVR chips) have serial interfaces, which can be used to connect larger serial EEPROMs or flash chips. The fast-access register file concept contains 32 x 8-bit general-purpose working registers With a single clock cycle access time. This means that during one single clock cycle, one Arithmetic Logic Unit (ALU)operation is executed. Two operands are output from the register file, the operation is executed and the result is stored back in the register file–in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect addresses register pointers for Data Space addressing, enabling efficient address calculations. One of the three address pointers is also used as the address pointer for the constant table look-up function.

These addedfunctionregistersarethe16-bitX-register, Y-register, and Z-register.

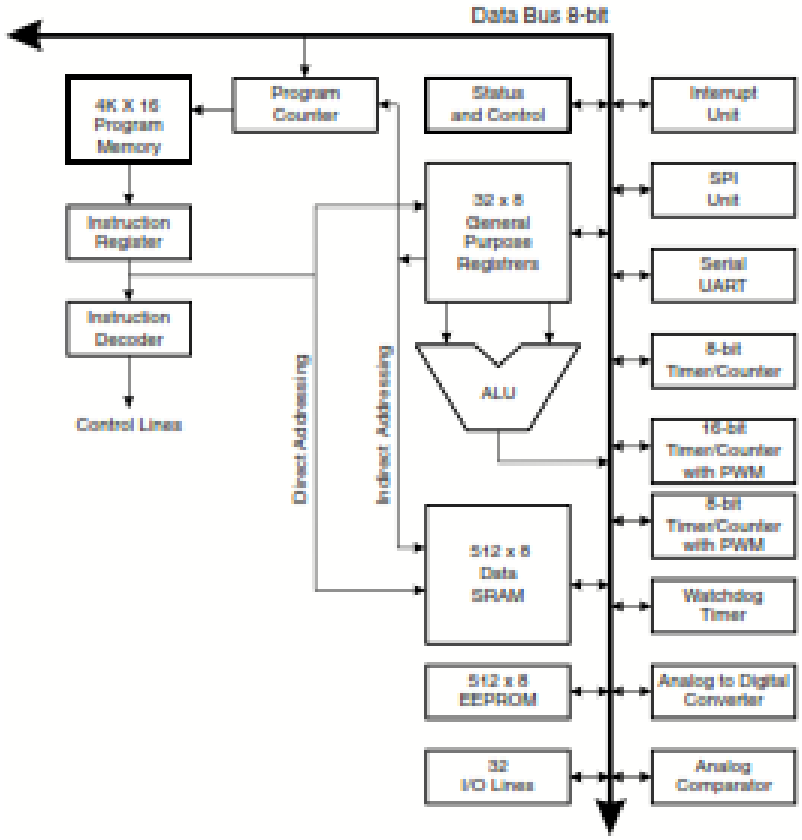


Figure. The AT90S8535 AVR RISC Architecture

The ALU supports arithmetic and logic functions between registers or between a constant And a register. Single register operations are also executed in the ALU.Figure4 shows the AT90S8535 AVR RISC microcontroller architecture.

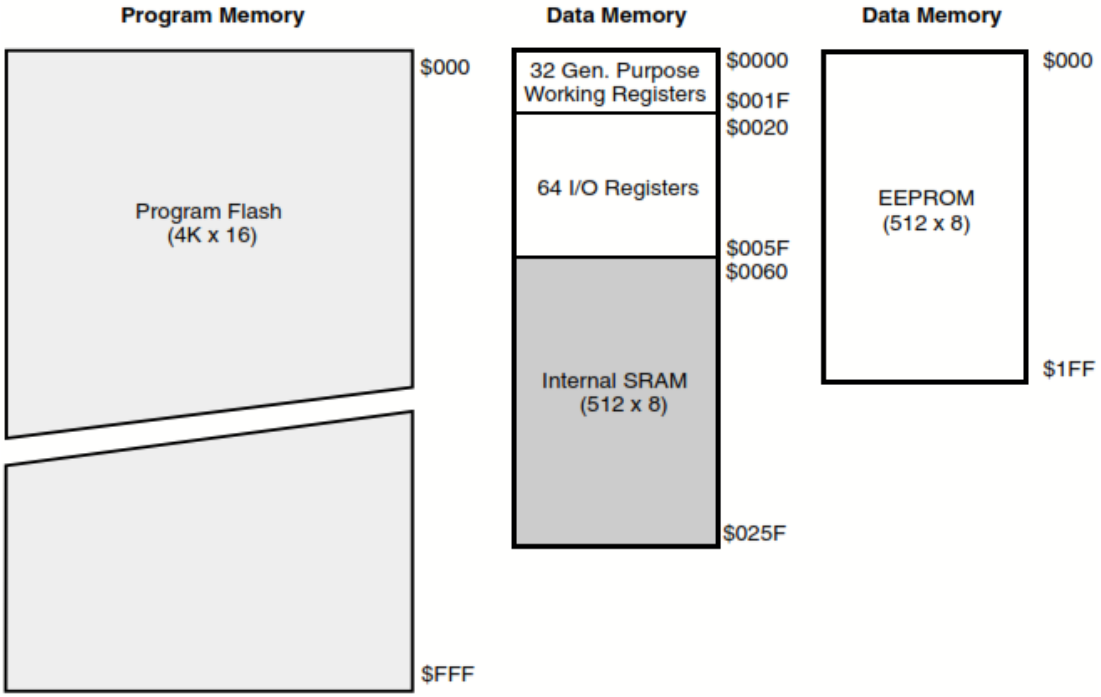
Memory Maps

In addition to the register operation, the conventional memory addressing modes can be used on the register file as well. This is enabled by the fact that the register file is assigned the 32 lowermost Data Space addresses ($00 - $1F), allowing them to be accessed as though they were ordinary memory locations. The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, Timer/Counters, A/D converters and other I/O functions. The I/O memory can be accessed directly or as the Data Space locations following those of the register file, $20 - $5F. The AVR uses a Harvard architecture concept – with separate memories and buses for program and data. The program memory is executed with a two-stage pipeline. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is in-system downloadable Flash memory. With the relative jump and call instructions, the whole

4K address space is directly accessed. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction. During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the stack. The stack is effectively allocated in the general data SRAM and consequently, the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The 10-bit stack pointer (SP) is read/write-accessible in the I/O space.

The 512 bytes data SRAM can be easily accessed through the five different addressing modes supported in the AVR architecture.The memory spaces in the AVR architecture are all linear and regular memory maps.

Figure 5. Memory Maps



Program memory

Program instructions are stored in non-volatile flash memory. Although the MCUs are 8-bit, each instruction takes one or two 16-bit words. The size of the program memory is usually indicated in the naming of the device itself (e.g., the ATmega64x line has 64 kB of flash while the ATmega32x line has 32 kB).

There is no provision for off-chip program memory; all code executed by the AVR core must reside in the on-chip flash. However, this limitation does not apply to the AT94 FPSLIC AVR/FPGA chips.

Internal data memory

The data address space consists of the register file, I/O registers, and SRAM.

Internal registers

- Atmel ATxmega128A1 in 100-pin TQFP package
- The AVRs have 32 single-byte registers and are classified as 8-bit RISC devices.
- In most variants of the AVR architecture, the working registers are mapped in as the first 32 memory addresses (0000_{16}–$001F_{16}$) followed by the 64 I/O registers (0020_{16}–$005F_{16}$).

Actual SRAM starts after these register sections (address 0060_{16}). (Note that the I/O register space may be larger on some more extensive devices, in which case the memory mapped I/O registers will occupy a portion of the SRAM address space.) Even though there are separate addressing schemes and optimized opcodes for register file and I/O register access, all can still be addressed and manipulated as if they were in SRAM.

In the XMEGA variant, the working register file is not mapped into the data address space; as such, it is not possible to treat any of the XMEGA's working registers as though they were SRAM. Instead, the I/O registers are mapped into the data address space starting at the very beginning of the address space. Additionally, the amount of data address space dedicated to I/O registers has grown substantially to 4096 bytes (0000_{16}–$0FFF_{16}$). As with previous generations, however, the fast I/O manipulation instructions can only reach the first 64 I/O register locations (the first 32 locations for bitwise instructions). Following the I/O registers, the XMEGA series sets aside a 4096 byte range of the data address space which can be used optionally for mapping the internal EEPROM to the data address space (1000_{16}–$1FFF_{16}$). The actual SRAM is located after these ranges, starting at 2000_{16}.

EEPROM

Almost all AVR microcontrollers have internal EEPROM for semi-permanent data storage. Like flash memory, EEPROM can maintain its contents when electrical power is removed.In most variants of the AVR architecture, this internal EEPROM memory is not mapped into the MCU's addressable memory space. It can only be accessed the same way an external peripheral device is, using special pointer registers and read/write instructions which makes EEPROM access much slower than other internal RAM.

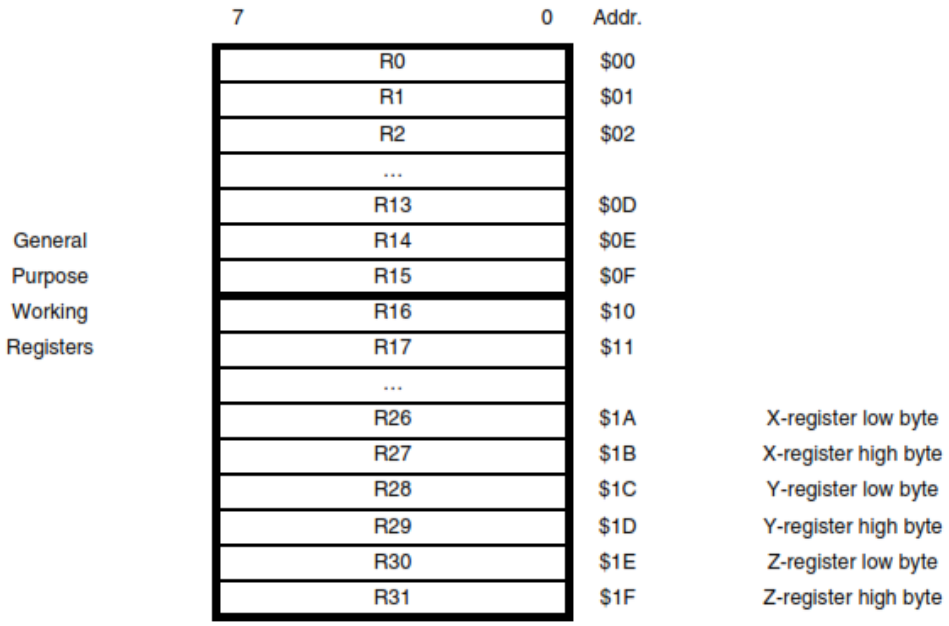
However, some devices in the SecureAVR (AT90SC) family [6] use a special EEPROM mapping to the data or program memory depending on the configuration. The XMEGA family also allows the EEPROM to be mapped into the data address space. Since the number of writes to EEPROM

is not unlimited — Atmel specifies 100,000 write cycles in their datasheets — a well designed EEPROM write routine should compare the contents of an EEPROM address with desired contents and only perform an actual write only if contents need to be changed.

General-purpose Register File

Figure 6 shows the structure of the 32 general-purpose working registers in the CPU.

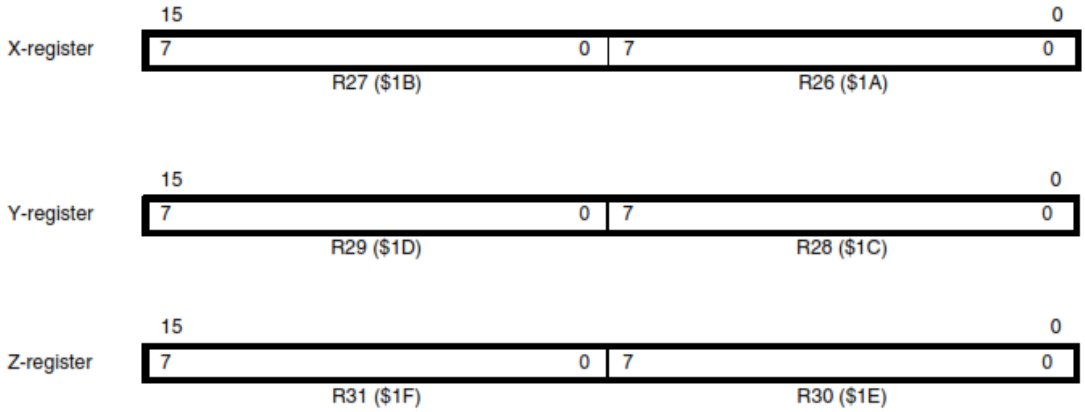
Figure 6. AVR CPU General-purpose Working Registers



All the register operating instructions in the instruction set have direct and single-cycle access to all registers. The only exception is the five constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI and ORI between a constant and a register and the LDI instruction for load immediate constant data. These instructions apply to the second half of the registers in the register file (R16..R31). The general SBC, SUB, CP, AND, and OR and all other operations between two registers or on a single register apply to the entire register file.

As shown in Figure 6, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations,this memory organization provides great flexibility in access of the registers, as the X-, Y-, and Z-registers can be set to index any register in the file. X-register, Y-register and Z-register The registers R26..R31 have some added functions to their general-purpose usage.These registers are address pointers for indirect addressing of the Data Space. The three indirect address registers, X, Y, and Z, are defined in Figure 7.

Figure 7. X-, Y-, and Z-register



ALU – Arithmetic Logic Unit

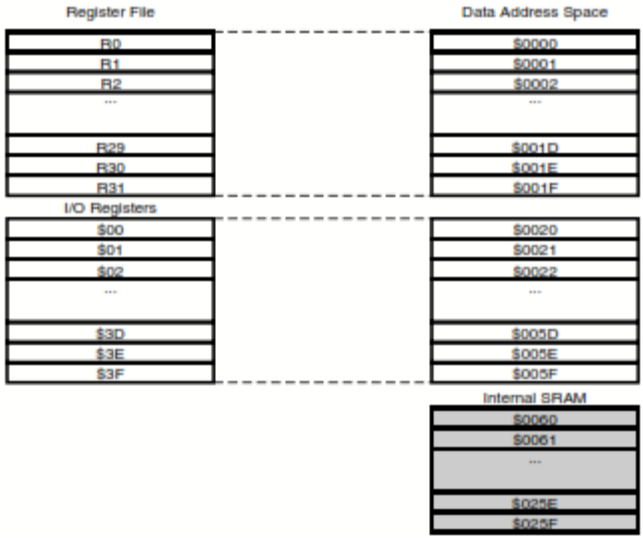
The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, ALU operations between registers in the register file are executed.The ALU operations are divided into three main categories: arithmetic, logical and bit functions.

EEPROM Data Memory

The AT90S8535 contains 512 bytes of data EEPROM memory. It is organized as a separate Data space, in which single bytes can be read and written. The EEPROM has an endurance of At least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described On page 51 specifying the EEPROM address registers, the EEPROM data register and the EEPROM control register.

SRAM Data Memory Figure 8 shows how the AT90S8535 SRAM memory is organized.

Figure 8. SRAM Organization



The lower 608 data memory locations address the Register file, the I/O memory and the internal data SRAM. The first 96 locations address the Register file + I/O memory, and the next 512 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect,Indirect with Pre-decrement and Indirect with Post-increment. In the Register file, registers R26 to R31 feature the indirect addressing pointer registers.

The direct addressing reaches the entire data space. The Indirect with Displacement mode features 63 address locations reached from the base address given by the Y- or Z-registers. When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y, and Z are decremented and incremented.

21. Known gaps

1. The MPMC subject as per the curriculum is not matching with the advance microcontrollers like ARM.

2. As per industry applications the known gap of TTL compatible device is not present in MPI which in the JNTU curriculum.

3. As per industry applications the known gap of timers supported with microprocessor not present in MPI which is in the JNTU curriculum.

4. As per industry applications the known gap of processor in embedded applications not present in the JNTU curriculum.

5. As per industry applications the known gap of microprocessor real time applications are not present in MPI subject which in the JNTU curriculum like home applications.

Actions taken:

The following topics are taken to fill the known gaps

1. ARM architecture

2. Introduction to dual core and Pentium processors.

3. USB

4. Embedded processor

5. Real time applications.

22. Discussion topics

UNIT 1:

- 8086 architecture
- Flag register organization

UNIT 3:

- Pin diagram of 8086
- Minimum mode and maximum mode operations

UNIT-4:

1.8255 architecture

2. Keyboard display interfacing

UNIT-5:

- Interrupt structure of 8086
- 8259 PIC architecture

UNIT-6:

Architecture of 8251 USART

- Serial communication
- Universal serial bus(USB)

UNIT-7:

1.8051 microcontroller architecture

2. Register organization of 8051

UNIT8:

1. Timer/counter operations

2. Serial communication operation

23. References, Journals, websites and E-links

Text books:-

1. Advanced Microprocessors and peripherals, A.K Ray and K.M Burch and TMH 2000

2. Micro controllers-Deshmukh, TMH

3. 8086 and 80286 microprocessors, hardware and software interfacing, -Walter A. Triebel Singh

4. Microprocessor architecture, programming and applications with 8085, by Ramesh Goankar

5. Advanced Microprocessors and Interfacing, Badri Ram, TMH

Reference Text Books:-

1. Microprocessor and interfacing-Douglas V.Hall, 2007

2. The 8088 and 8086 microprocessor-PHI, 4th Edition, 2003

3. Microcomputer systems, the 8086/8088 Family, architecture, Programming & Design, Yu-Chang Liu & Glenn A Gibson, PHI

4. The Intel Microprocessor, 8086/8088, 80186, 80286, 80386 and 80486 programming and interfacing by Barry B. Brey

5. Microprocessors, Theory and Applications, Intel and Motorola – Rafiquzzaman

6.Atmel AVR microcontroller primer: programming and interfacing- steven f.barret, Daniel j. pack

Websites:-

1. www.embedded-computing.com

2. www.mcjournal.com

3. www.atmel.com

4. www.keil.com

5. http://8085projects.info/Interfacing-of-PIC-8259-with-8085.html

Journals:-

1. www.mcjournal.com (web journal on microcontrollers)

2. Microprocessors and Microcomputer System

3. Embedded Hardware Design

24. Quality Control Sheets

Hard copy will be attached

25.Students List

3-1 A:

SlNo	AdmnNo	StudentName
1	12R11A0401	A BALA KRISHNA
2	12R11A0402	A J V SUDHESHNA
3	12R11A0403	A PAVAN REDDY

4	12R11A0404	ALETI VIJAY KUMAR
5	12R11A0405	AMBERPETA SOWJANYA
6	12R11A0406	ARSHID SHEETAL
7	12R11A0407	BANAPURAM SHIVA SANTH
8	12R11A0408	BANDARI MOUNIKA
9	12R11A0409	BATTINIGARI LEELADHARA
10	12R11A0410	BOMMINENI VAMSHIDHAR REDDY
11	12R11A0411	BORAMPETA NISHANTH REDDY
12	12R11A0412	CHILLIJUVURU SATYANARANAYANA RAJU
13	12R11A0413	CHITYALA SAIROOPA
14	12R11A0414	SAI PRANEETH D
15	12R11A0415	DHEERAVATH RAJU
16	12R11A0416	DIVYA DEENDAYAL
17	12R11A0417	DODDA SRAVANTH KUMAR REDDY
18	12R11A0418	G MOUNIKA
19	12R11A0419	G RAVIKANTH SAGAR
20	12R11A0420	GADWALA SWAROOPA
21	12R11A0421	GANNARAM ARJUN
22	12R11A0422	GIRIMALLA NIKITHA
23	12R11A0423	GITTA BHUVAN KUMAR
24	12R11A0424	GOLLAPUDI SURYANARAYANA KARTHIK
25	12R11A0425	GORASA SRISHA
26	12R11A0426	GORENTLA TEJASWINI
27	12R11A0427	GUDE SHIVANADAM
28	12R11A0428	GUJJALA SRINATH

29	12R11A0429	GURRAM RAHUL KUMAR
30	12R11A0430	GUTTAMEEDA SOUBHAGYA
31	12R11A0431	IPPALA TEJA SRI
32	12R11A0432	J DIVYA
33	12R11A0433	K GNANA SARASWATHI
34	12R11A0434	KAVURI RAGHU RAM
35	12R11A0435	KALLI RAMA CHANDRA TEJA
36	12R11A0436	KANNOJU CHARANKUMAR
37	12R11A0437	KANUMURI DINESH VARMA
38	12R11A0438	KARAN PANDRE
39	12R11A0439	KARNAKANTI BHASKER
40	12R11A0440	KARNATI DEVI VARA PRASAD
41	12R11A0441	KASTURI ROHINI
42	12R11A0442	KATAPALLY RAMA KRISHNA REDDY
43	12R11A0443	KATIKELA SAI KUMAR
44	12R11A0444	KEERTHANA T
45	12R11A0445	KHODE VIVEKANAND
46	12R11A0446	KISTAIGARI SAMATHA
47	12R11A0447	KURA VIVEK
48	12R11A0448	MAHENDRA HARSHA VARDHAN
49	12R11A0449	M SOWMYA
50	12R11A0450	M MANOJ JOSHI
51	12R11A0451	MACHARLA ARUN KUMAR CHARY
52	12R11A0452	MADHAGANI KEERTHI
53	12R11A0453	MADIREDDY SIRISHA

54	12R11A0454	RAGULA RANI
55	12R11A0456	S CHAITANYA
56	12R11A0457	SHIVAM
57	12R11A0458	SINGIREDDY RAVINDER REDDY
58	12R11A0459	S ALEKHYA
59	12R11A0460	TENALI VISHAL ANURAG
60	13R15A0401	GOWRAVAJHALA MEGHASHYAMA SARMA
61	13R15A0402	TANNEERU SRINIVAS
62	13R15A0403	SANALA MOUNIKA
63	13R15A0404	ADEPU PANDU
64	13R15A0405	NYALAM RAM

26. Group-Wise students list for discussion topics

Section 3-1A

Group 1

1	12R11A0401	A BALA KRISHNA	1
2	12R11A0402	A J V SUDHESHNA	1
3	12R11A0403	A PAVAN REDDY	1
4	12R11A0404	ALETI VIJAY KUMAR	1
5	12R11A0405	AMBERPETA SOWJANYA	1
6	12R11A0406	ARSHID SHEETAL	1

Group 2

7	12R11A0407	BANAPURAM SHIVA SANTH	2
8	12R11A0408	BANDARI MOUNIKA	2
9	12R11A0409	BATTINIGARI LEELADHARA	2
10	12R11A0410	BOMMINENI VAMSHIDHAR REDDY	2
11	12R11A0411	BORAMPETA NISHANTH REDDY	2
12	12R11A0412	CHILLIJUVURU SATYANARANAYANA RAJU	2

Group 3:

13	12R11A0413	CHITYALA SAIROOPA	3
14	12R11A0414	SAI PRANEETH D	3
15	12R11A0415	DHEERAVATH RAJU	3
16	12R11A0416	DIVYA DEENDAYAL	3
17	12R11A0417	DODDA SRAVANTH KUMAR REDDY	3
18	12R11A0418	G MOUNIKA	3

Group 4:

19	12R11A0419	G RAVIKANTH SAGAR	4
20	12R11A0420	GADWALA SWAROOPA	4
21	12R11A0421	GANNARAM ARJUN	4
22	12R11A0422	GIRIMALLA NIKITHA	4
23	12R11A0423	GITTA BHUVAN KUMAR	4
24	12R11A0424	GOLLAPUDI SURYANARAYANA KARTHIK	4

Group 5:

25	12R11A0425	GORASA SRISHA	5

26	12R11A0426	GORENTLA TEJASWINI	5
27	12R11A0427	GUDE SHIVANADAM	5
28	12R11A0428	GUJJALA SRINATH	5
29	12R11A0429	GURRAM RAHUL KUMAR	5
30	12R11A0430	GUTTAMEEDA SOUBHAGYA	5

Group 6:

31	12R11A0431	IPPALA TEJA SRI	6
32	12R11A0432	J DIVYA	6
33	12R11A0433	K GNANA SARASWATHI	6
34	12R11A0434	KAVURI RAGHU RAM	6
35	12R11A0435	KALLI RAMA CHANDRA TEJA	6
36	12R11A0436	KANNOJU CHARANKUMAR	6

Group 7:

37	12R11A0437	KANUMURI DINESH VARMA	7
38	12R11A0438	KARAN PANDRE	7
39	12R11A0439	KARNAKANTI BHASKER	7
40	12R11A0440	KARNATI DEVI VARA PRASAD	7
41	12R11A0441	KASTURI ROHINI	7
42	12R11A0442	KATAPALLY RAMA KRISHNA REDDY	7

Group 8:

43	12R11A0443	KATIKELA SAI KUMAR	8
44	12R11A0444	KEERTHANA T	8
45	12R11A0445	KHODE VIVEKANAND	8
46	12R11A0446	KISTAIGARI SAMATHA	8
47	12R11A0447	KURA VIVEK	8
48	12R11A0448	MAHENDRA HARSHA VARDHAN	8

Group 9:

49	12R11A0449	M SOWMYA	9
50	12R11A0450	M MANOJ JOSHI	9
51	12R11A0451	MACHARLA ARUN KUMAR CHARY	9
52	12R11A0452	MADHAGANI KEERTHI	9
53	12R11A0453	MADIREDDY SIRISHA	9
54	12R11A0454	RAGULA RANI	9

Group 10:

55	12R11A0456	S CHAITANYA	10
56	12R11A0457	SHIVAM	10
57	12R11A0458	SINGIREDDY RAVINDER REDDY	10
58	12R11A0459	S ALEKHYA	10
59	12R11A0460	TENALI VISHAL ANURAG	10
60	13R15A0401	GOWRAVAJHALA MEGHASHYAMA SARMA	10

Group 11:

61	13R15A0402	TANNEERU SRINIVAS	11
62	13R15A0403	SANALA MOUNIKA	11
63	13R15A0404	ADEPU PANDU	11
64	13R15A0405	NYALAM RAM	11