Anantha Lakshmi
INSTITUTE OF TECHNOLOGY AND SCIENCES
(Approved by A.I.C.T.E, New Delhi, Accredited by NAAC and Affiliated to J.N.T.University Ananthapuramu.)
Near SK University, Itikalapalli-515 721. (V), Ananthapuramu (Dist.)
Website : www.alits.ac.in

Department of Electronics and Communication Engineering

MICROPROCESSORS&MICROCONTROLLERS

(19A04601T)

LECTURE NOTES

B.TECH(ECE)

(III YEAR–II SEM)

(2019-20)

ANANTHA LAKSHMIINSTITUTE OF TECHNOLOGY AND SCIENCES

IIIYearB.Tech.ECE-IISem

	L	T/P/D	C
	3	1/-/-	4

(19A04601T)MICROPROCESSORSANDMICROCONTROLLERS

COURSEOBJECTIVES:

1. To understand the basics of microprocessors and microcontrollers architectures and

itsfunctionalities.
2. To develop an in-depth understanding of the operation of microprocessors andmicrocontrollers,machinelanguageprogramming& interfacingtechniques.
3. To design and develop Microprocessor/ microcontroller based systems for real timeapplicationsusinglow level language likeALP.
4. Tounderstandtheconcepts ofARMprocessor.

UNIT-I

8086ARCHITECTURE: Architectureof8086,RegisterOrganization,ProgrammingModel,Memory addresses,MemorySegmentation,PhysicalMemoryOrganization,Signaldescriptions of 8086-Common Function Signals, Minimum and Maximum mode signals,Timing diagrams.

UNIT-II

INSTRUCTION SET AND ASSEMBLY LANGUAGE PROGRAMMING OF 8086: Instruction formats,Addressingmodes,InstructionSet,AssemblerDirectives,Procedures,Macros,SimplePrograms involvingLogical,BranchandCallInstructions,Sorting,EvaluatingArithmeticExpressions, StringManipulations.

UNIT-III

I/O INTERFACE: 8255 PPI, Various Modes of Operation and Interfacing to 8086, D/A and A/DConverter,Stepper motor, InterfacingofDMA controller8257

INTERFACINGWITHADVANCEDDEVICES: MemoryInterfacingto8086,InterruptStructureof8086, Vector Interrupt Table, InterruptServiceRoutine, architectureof8259.

COMMUNICATIONINTERFACE: SerialCommunicationStandards,SerialDataTransferSchemes,8251 USARTArchitectureandInterfacing.

UNIT-IV

INTRODUCTION TO MICROCONTROLLERS: Overview of 8051 Microcontroller, Architecture,I/O Ports, Memory Organization, Addressing Modes and Instruction set of 8051, SimplePrograms,memory interfacing to8051

UNIT-V

8051REALTIMECONTROL: ProgrammingTimerInterrupts,ProgrammingExternalHardware Interrupts,ProgrammingtheSerialCommunicationInterrupts,Programming8051TimersandCounters

ARMPROCESSOR:

Fundamentals,Registers,Currentprogramstatusregister,Pipeline,Interrupt andthe vectortable.

TEXT BOOKS:

1. D. V. Hall,Microprocessorsand Interfacing,TMGH, 2ndEdition2006.
2. Kenneth. J. Ayala,The8051 Microcontroller,3rd Ed.,CengageLearning.

3. ARMSystemDeveloper'sGuide:DesigningandOptimizingSystemSoftware-AndrewN.Slo ss, Dominic Symes,Chris Wright, Elsevier Inc.,2007

REFERENCEBOOKS:

1. AdvancedMicroprocessorsandPeripherals–A.K.RayandK.M.Bhurchandani,TMH,2ndEditio n2006.
2. The8051Microcontrollers,ArchitectureandProgrammingandApplications-K.UmaRao,Andhe Pallavi, Pearson,2009.
3. MicroComputerSystem8086/8088FamilyArchitecture,ProgrammingandDesign-LiuandGA Gibson,PHI,2ndEd.
4. MicrocontrollersandApplication-Ajay.V. Deshmukh,TMGH,2005.

COURSEOUTCOMES:

Aftergoingthroughthiscoursethe studentwill

1. Learntheinternalorganizationofpopular8086/8051microprocessors/microcontrollers.
2. Learnhardwareandsoftware interactionandintegration.
3. Learn the designof microprocessors/microcontrollers-basedsystems

UNIT-I

8086Architecture

- Architectureof8086
- RegisterOrganization
- ProgrammingModel
- Memoryaddresses
- MemorySegmentation
- PhysicalMemoryOrganization
- Signaldescriptionsof8086-CommonFunctionSignals
- MinimumandMaximum modesignals
- Timingdiagrams

UNIT-I

8086Architecture

IntroductiontoMicroprocessors

A microprocessor isa computerprocessor whichincorporatesthefunctions of a computer's central processing unit (CPU) on asingleintegratedcircuit(IC),orat mostafewintegratedcircuits

Themicroprocessorisamultipurpose, clock driven, register based,digital-integratedcircuitwhichaccepts binary dataasinput,processesitaccordingto instructions storedinits memory,andprovidesresultsasoutput.Microprocessorscontainboth combinationallogic and sequentialdigital logic. Microprocessors operate on numbers and symbols representedin thebinarynumeralsystem.

GenerationofMicroprocessors:

- **INTEL 4004(1971)**
  - 4-bitmicroprocessor
  - 4KBmainmemory
  - 45instructions
  - PMOStechnology
  - wasfirstprogrammabledevicewhichwasusedincalculators
- **INTEL8008(1972)**
  - 8-bitversionof4004
  - 16KBmainmemory
  - 48instructions
  - PMOStechnology
  - Slow
- **Intel8080**(1973)
  - 8-bitmicroprocessor
  - 64KBmainmemory
  - 2microsecondsclockcycle time
  - 500,000instructions/sec
  - 10Xfasterthan8008
  - NMOStechnology
  - Drawbackwasthatitneededthreepowersupplies.

- Smallcomputers(Microcomputers)weredesignedinmid 1970's

  Using8080asCPU.

- **INTEL8086/8088**

  Yearofintroduction1978for 8086 and1979for8088

  - 16-bitmicroprocessors
  - Databuswidthof8086is16bitand8 bitfor8088
  - 1MBmainmemory
  - 400nanosecondsclockcycletime
  - 6byteinstructioncachefor8086 and4bytefor8088
  - Otherimprovementsincludedmoreregistersandadditionalinstructions
  - In1981IBMdecidedtouse8088initspersonalcomputer

- **INTEL80186**(1982)

  - 16-bitmicroprocessor-upgradedversionof8086
  - 1MBmainmemory
  - Contained special hardware like programmable counters,interruptcontrolleretc.
  - Never usedinthePC
  - Butwasidealforsystemsthatrequiredaminimumofhardware.

- **INTEL80286**(1983)

  - 16-bit high performance microprocessor with memorymanagement& protection
  - 16MBmainmemory
  - Fewadditionalinstructionstohandleextra15MB
  - Instructionexecutiontimeisaslittleas 250ns
  - Concentrates on the features needed to implementMULTITASKING

- **Intel80386**(1986)
- **Intel80486**(1989)

- **Pentium**(1993)
- **Pentiumpro**(1995)
- **Pentiumii**(1997)
- **Pentiumiii**(1999)
- **Pentiumiv**(2002)
- **Latestis Inteli9processor**

GeneralArchitectureofMicroprocessors

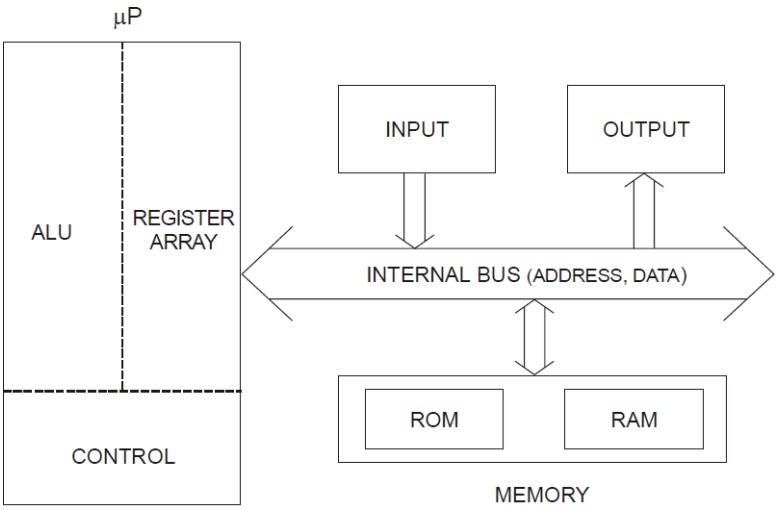


Figure 1.2 Architecture of Microprocessor

Buses

Figure shows busses interconnecting various blocks. These busses allow exchange of words between the blocks. A bus has a wire or line for each bit and thus allows exchange of all bits of a word in parallel. The processing of bits in the µP is also in parallel. The busses can thus be viewed as data highways. The width of a bus is the number of signal lines that constitute the bus.

Arithmetic-Logic Unit (ALU)

The arithmetic-logic unit is a combinational network that performs arithmetic and logical operations on the data.

Internal Registers

A number of registers are normally included in the microprocessor. These are used for temporary storage of data, instructions and addresses during execution of a program. Those in the Intel

RegisterOrganizationof8086

8086 has a powerful set of registers containing general purpose andspecial purpose registers. All the registers of 8086 are 16-bit registers. Thegeneralpurposeregisters,canbeusedeither8-bitregistersor16-bitregisters. The general purpose registers are either used for holding the data,variablesand intermediate resultstemporarily or for other purpose likecounter or for storing offset address for some particular addressing modesetc. The special purpose registers are used as segment registers, pointers,indexregistersorasoffsetstorageregistersforparticularaddressingmodes. Fig 1.4 shows register organization of 8086. We will categorize theregistersetinto fourgroups as follows:

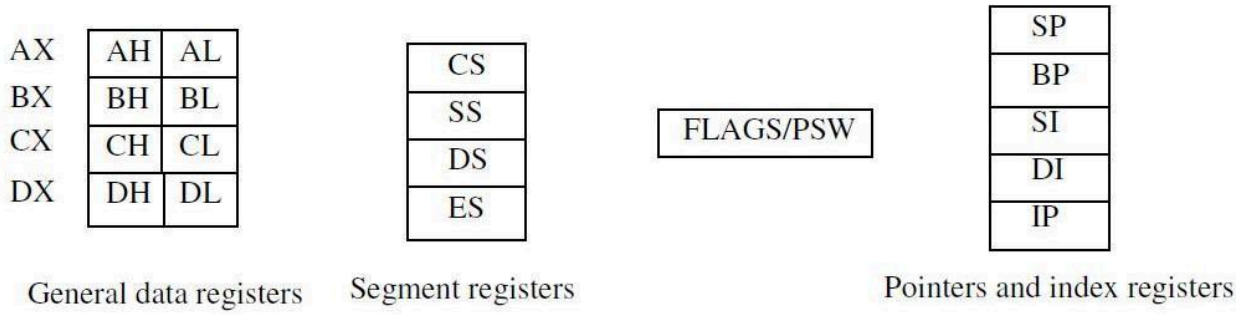


Fig.1.4 Register organization of 8086 Microprocessor

GeneraldataRegisters:

TheregistersAX,BX,CX,andDX are thegeneral 16-bitregisters.

AX Register: Accumulator register consists of two 8-bit registers AL and AH,which can be combined together and used as a 16- bit register AX. AL in thiscase contains the low-order byte of the word, and AH contains the high-order byte. Accumulator can be used for I/O operations, rotate and stringmanipulation.

BX Register: This register is mainly used as a base register. It holds thestarting base location of a memory region within a data segment. It is usedas offset storage for forming physical address in case of certain addressingmode.

CXRegister: Itisusedasdefaultcounteror countregisterincaseofstringand loopinstructions.

DX Register: Data register can be used as a port number in I/O operationsand implicit operand or destination in case of few instructions. In integer32-bit multiply and divide instruction the DX register contains high-orderword of theinitialorresultingnumber.

Segmentregisters:

Tocomplete1Mbytememoryisdividedinto16logicalsegments.Thecomplete1Mbytememorysegmentationisasshowninfig1.5.Eachsegmentcontains64Kbyteofmemory.Therearefoursegmentregisters.

Code segment (CS) is a 16-bit register containing address of 64 KB segmentwith processor instructions. The processor uses CS segment for all accessesto instructions referenced by instruction pointer (IP) register. CS registercannot be changed directly. The CS register is automatically updated duringfar jump, far call and far return instructions. It is used for addressing amemorylocationinthecodesegmentofthememory,wheretheexecutableprogramis stored.

Stack segment (SS) is a 16-bit register containing address of 64KB segmentwithprogramstack.Bydefault,theprocessorassumesthatalldatareferencedbythestackpointer(SP)andbasepointer(BP)registersislocated in the stack segment. SS register can be changed directly using POPinstruction. It is used for addressing stack segment of memory. The stacksegmentisthatsegmentofmemory,whichisused tostorestackdata.

Data segment (DS) is a 16-bit register containing address of 64KB segmentwithprogramdata.Bydefault,theprocessorassumesthatalldatareferenced by general registers (AX, BX, CX, DX) and index register (SI, DI) islocated in the data segment. DS register can be changed directly using POPand LDS instructions. It points to the data segment memory where the dataisresided.

Extra segment (ES) is a 16-bit register containing address of 64KB segment,usually with program data. By default, the processor assumes that the DIregister references the ES segment in string manipulation instructions. ESregister canbechangeddirectlyusingPOP andLESinstructions.It alsoreferstosegmentwhichessentiallyisanotherdata segmentofthememory.Italsocontainsdata.

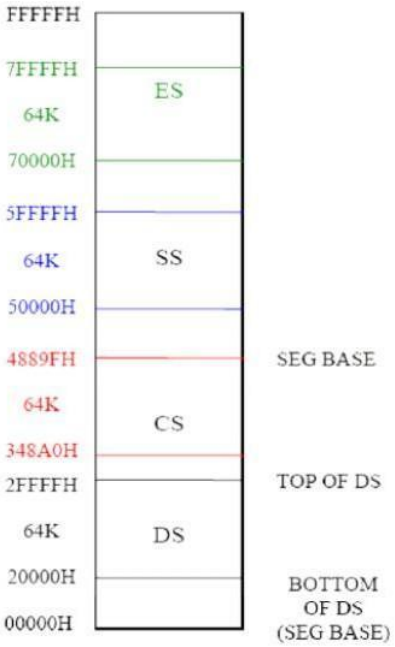


Fig1.5. Memory segmentation

Pointersandindexregisters.

The pointers contain within the particular segments. The pointers IP, BP, SPusuallycontainoffsetswithinthecode,dataandstacksegmentsrespectively

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

Base Pointer (BP) is a 16-bit register pointing to data in stack segment. BPregisterisusuallyusedforbased,basedindexedorregisterindirectaddressing.

Source Index (SI) is a 16-bit register. SI is used for indexed, based indexedand register indirect addressing, as well as a source data addresses in stringmanipulation instructions.

Destination Index (DI) is a 16-bit register. DI is used for indexed, basedindexedandregisterindirectaddressing,aswellasadestinationdataaddressinstringmanipulationinstructions.

FlagRegister:

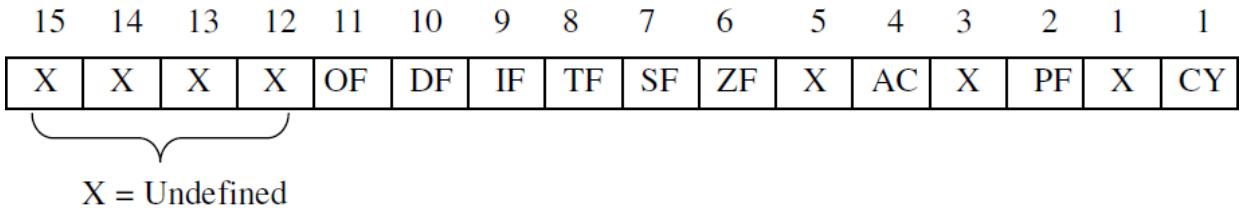


Fig1.6 . Flag Register of 8086

Flags Register determines the current state of the processor. They aremodified automatically by CPU after mathematical operations, this allows todetermine the type of the result, and to determine conditions to transfercontrol to otherparts of theprogram.The 8086 flag register as shown inthefig1.6.8086has 9activeflagsandthey aredividedintotwocategories:

1. ConditionalFlags
2. ControlFlags

Conditionalflagsareasfollows:

CarryFlag(CY): Thisflagindicatesanoverflowconditionforunsignedintegerarithmetic. Itis also usedinmultiple-precisionarithmetic.

AuxiliaryFlag(AC): IfanoperationperformedinALUgeneratesacarry/barrow from lower nibble (i.e. D0 – D3) to upper nibble (i.e. D4 – D7),the AC flag is set i.e. carry given by D3 bit to D4 is AC flag. This is not ageneral-purpose flag, it is used internally by the Processor to perform Binaryto BCDconversion.

Parity Flag (PF): This flag is used to indicate the parity of result. If lowerorder 8-bits of the result contains even number of 1's, the Parity Flag is setandforoddnumberof 1's,theParityflagis reset.

Zero Flag (ZF): It is set; if the result of arithmetic or logical operation is zeroelseitis reset.

Sign Flag (SF): In sign magnitude format the sign of number is indicated byMSB bit. Iftheresultofoperationisnegative,sign flagisset.

ControlFlags

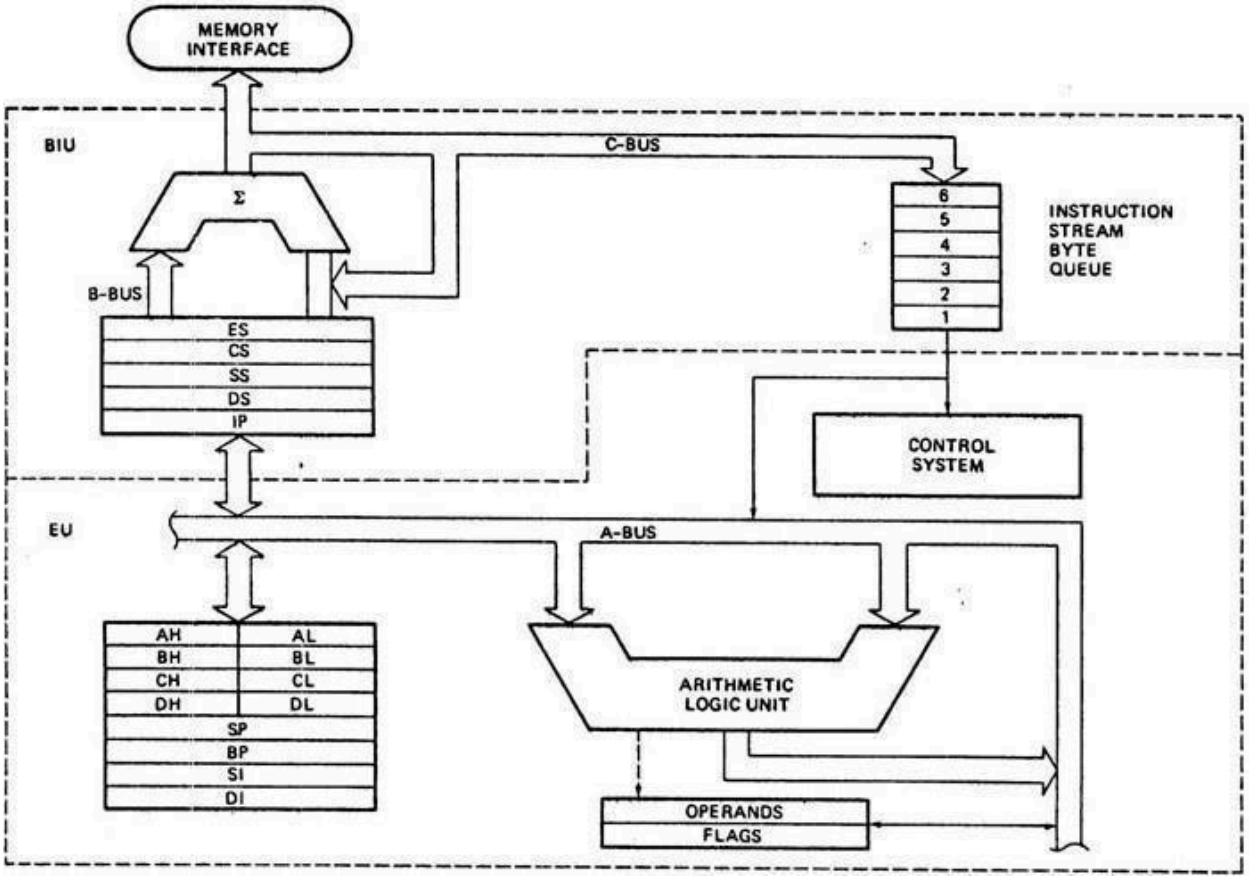
Control flags are set or reset deliberately to control the operations of theexecution unit.Controlflags areas follows:

Trap Flag (TF): It is used for single step control. It allows user to execute oneinstruction of a program at a time for debugging. When trap flag is set,programcanberunin singlestepmode.

InterruptFlag(IF): Itisaninterruptenable/disableflag.Ifitisset,themaskable interrupt of 8086 is enabled and if it is reset, the interrupt isdisabled. It can be set by executing instruction sit and can be cleared byexecutingCLIinstruction.

Direction Flag (DF): It is used in string operation. If it is set, string bytes areaccessed from higher memory address to lower memory address. When it isreset, the string bytes are accessed from lower memory address to highermemoryaddress.

8086Architecture



The8086ismainlydividedintomainlytwoblocks

1. Execution Unit
(EU)2.BusinterfaceUnit(B
IU)

Dividingtheworkbetweenthesetwowillspeedupthe processing

1) EXECUTIONUNIT(EU)

The ExecutionunittellstheBIUwheretofetchinstructionsor data
from

- decodesinstructionsand
- Executes

instructionsTheExecutionun

itcontains:

1) Controlcircuitry
2) ALU
3) FLAGS
4) GeneralpurposeRegisters

5) PointerandIndexRegisters

ControlCircuitry:

- Itdirectsinternaloperations.

- AdecoderintheEUtranslatesinstructionsfetchedfrommemoryIntoseries ofactionswhich theEUcarriesout

ArithmeticLogicUnit:

16bitALU

Usedtocarrytheoperations

ADD

SUBTRACT

XOR

INCREMENT

DECREMENT

COMPLEMENT

SHIFTBINARYNUMBERS

FLAGREGISTERS:

A flag is a flip flop that indicates some condition produced byexecutionofaninstructionorcontrolscertainoperationof the EU.

Itis16bit

Ithasnineactiveflag

sDividedintotwo types

1. Conditionalflags
2. Controlflags

ConditionalFlags

CarryFlag(CY): Thisflagindicatesanoverflowconditionforunsignedintegerarithmetic. Itis also usedinmultiple-precisionarithmetic.

AuxiliaryFlag(AC): IfanoperationperformedinALUgeneratesacarry/barrow from lower nibble (i.e. D0 – D3) to upper nibble (i.e. D4 – D7),the AC flag is set i.e. carry given by D3 bit to D4 is AC flag. This is not ageneral-purpose

flag, it is used internally by the Processor to perform Binaryto BCDconversion.

Parity Flag (PF): This flag is used to indicate the parity of result. If lowerorder 8-bits of the result contains even number of 1's, the Parity Flag is setandforoddnumberof 1's, theParityflagis reset.

Zero Flag (ZF): It is set; if the result of arithmetic or logical operation is zeroelseitis reset.

Sign Flag (SF): In sign magnitude format the sign of number is indicated byMSBbit. Iftheresultofoperationisnegative,sign flagisset.

ControlFlags

Control flags are set or reset deliberately to control the operations of theexecution unit.Controlflags areas follows:

Trap Flag (TF): It is used for single step control. It allows user to execute oneinstruction of a program at a time for debugging. When trap flag is set,programcanberunin singlestepmode.

InterruptFlag(IF): Itisaninterruptenable/disableflag.Ifitisset,themaskable interrupt of 8086 is enabled and if it is reset, the interrupt isdisabled. It can be set by executing instruction sit and can be cleared byexecutingCLIinstruction.

Direction Flag (DF): It is used in string operation. If it is set, string bytes areaccessed from higher memory address to lower memory address. When it isreset, the string bytes are accessed from lower memory address to highermemoryaddress.

GeneralPurposeRegisters:
The 8086 general purpose registers are similar to those of earliergenerations 8080 and 8085 .It was designed in such a way that manyprograms written for 8080 and 8085 could easily be translated to run on8086.Theadvantageofusinginternalregistersforthe temporarystorageofdata is that since data already in the EU ., it can be accessed much morequicklythanitcould beaccessedfromexternal memory.
GeneralPurposeRegisters

TheregistersAX,BX,CX,andDX are thegeneral 16-bitregisters.

AX Register: Accumulator register consists of two 8-bit registers AL and AH,whichcanbecombinedtogether andusedasa16-bitregisterAX.ALinthiscase contains the low-order byte of the word, and AH contains the high-order byte. Accumulator can be used for I/O operations, rotate and stringmanipulation.
BX Register: This register is mainly used as a base register. It holds thestartingbase locationofamemoryregionwithina datasegment.Itisusedas offset storage for forming physical address in case of certain addressingmode.

CXRegister: Itisusedasdefaultcounteror countregisterincaseofstringand loopinstructions.

DXRegister: Dataregistercanbeusedasaportnumberin I/Ooperationsand implicit operand or destination in case of few instructions. In integer32-bit multiply and divide instruction the DX register contains high-orderword of theinitialorresultingnumber.

2) BUSINTERFACEUNIT(BIU)

TheBIUsendsout

- Addresses
- Fetchesinstructionsfrommemory
- ReaddatafromportsandmemoryOr

TheBIUhandlesalltransferofdataandaddressesonthe busesfortheExecutionUnit

TheBusinterfaceunitcontains

1) InstructionQueue
2) Instructionpointer
3) Segmentregisters
4) AddressGenerator

InstructionQueue:

BIU gets upto 6 bytes of next instructions and stores them in theinstruction queue. When EU executes instructions and is ready for its nextinstruction, then it simply reads the instruction from this instruction queueresulting in increased execution speed. Fetching the next instruction whilethe current instruction executes is called **pipelining.**(based on FIFO) .This ismuch faster than sending out an addressestothesystem memory andwaiting for memoryto send back the next instruction byte or bytes .HeretheQueuewillbedumpedandthenreloadedfromthenewAddress.

SegmentRegister:

The 8086 20 bit addresses So it can address upto 2^{20} in memory (1 Mbyte)but at any instant it can address upto 4 64 KB segments. This four segmentsholds the upper 16 bits of the starting address of four memory segmentsthat the 8086 is working with it at particular time .The BIU always insertszerosforthelowest4bitsofthe20bitstartingaddress

Example: Ifthecodesegmentregistercontains348AHthenthecodesegmentstart sat348A0H.Inotherwordsa64KbytesegmentcanbelocatedanywherewithinaM ByteaddressSpacebutthesegmentwillalwaysstartsatanaddresswithzerosinthe lowest4bits

Stack: is a section of memory set aside to store addresses and data whilesubprogram executesisoften called segmentbase .The stacksegmentregisteralwaysholdstheupper16bitstarting addressofprogramstack**.**

The extra segment register and data segment register is used to hold theupper 16 bit starting addressesof two memory segments that are used fordata.

Instruction Pointer holds the 16 bit address or offset of the next code bytewithin the code segment.The value contained in the Instruction Pointercalled as Offset because the value must be added to the segment baseaddressin CStoproducetherequired20bitaddress.

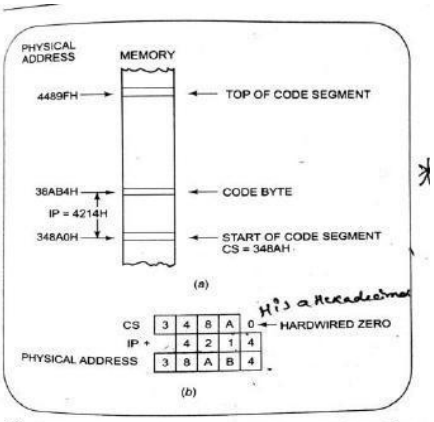


Fig. 2.10 *Addition of IP to CS to produce the physical address of the code byte, (a) Diagram, (b) Computation.*

CS register contains the Upper 16 bit of the starting address of thecode segment in the 1 Mbyte address range the instruction pointer containsa 16 bit offset which tells wherein that 64 Kbyte code segment the nextinstruction bytehastobefetchedfrom.

StackRegisterandStackPointer:

Stack: is a section of memory set aside to store addresses and datawhilesubprogramexecutesisoftencalledsegmentbase.Thestacksegment register always holds the upper 16 bit starting address of programstack. The Stack pointer (SP) holds the 16 bit offset from the starting of thesegmenttothememorylocationwhereawordwasmostrecentlystored .The memorylocationwhere thewordisstorediscalledastopofthestack

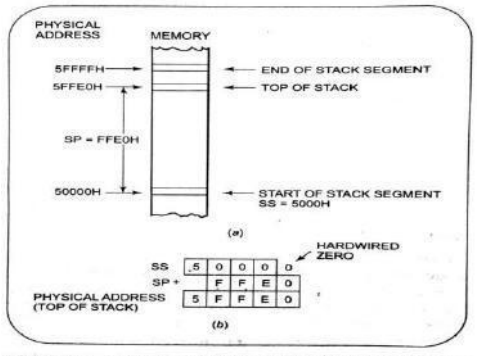


Fig. 2.11 *Addition of SS and SP to produce the physical address of the top of the stack. (a) Diagram, (b) Computation.*

PointerandIndexregisters:
InadditiontostackpointerregisterEUhasBas
epointerRegister(BP)
Source Pointer
Register(SP)DestinationPointerR
egister(DP)

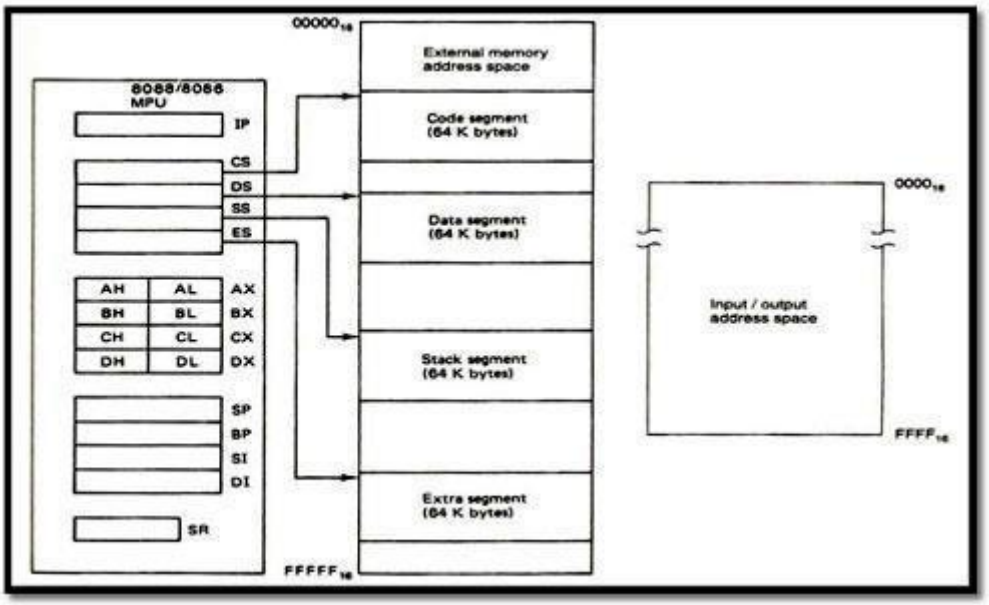
Thesethreeregistersareusedtostoretemporarystorageofdatalikegeneral purpose registers .They hold the 16 bit offset data of the data wordinoneof thesegment

Programmingmodel

Howcana20-bitaddressbeobtained,iftherearеonly16-bitregisters? However, the largest register is only 16 bits (64k); so physical addresseshave to be calculated. These calculations are done in hardware within themicroprocessor.
The 16-bit contents of segment register gives the starting/ base address ofparticular segment. To address a specific memory location within a segmentwe need an offset address. The offset address is also 16-bit wide and it isprovidedbyoneoftheassociatedpointerorindexregister.

Tobeabletoprogramamicroprocessor,onedoesnotneedtoknowallofits hardwarearchitecturalfeatures.Whatisimportanttothe programmerisbeingawareofthevarious registerswithinthedeviceandtounderstandtheirpurpose,functions,operating capabilities,andlimitations.

The above figure illustrates the software architecture of the 8086microprocessor. From this diagram, we see that it includes fourteenl6-bitinternal registers: the instruction pointer (IP), four data registers (AX, BX,CX, and DX), two pointer registers (BP and SP), two index registers (SI andDI),foursegmentregisters(CS,DS,SS,andES)andstatusregister(SR),withnin eofitsbits implementedas status andcontrolflags.



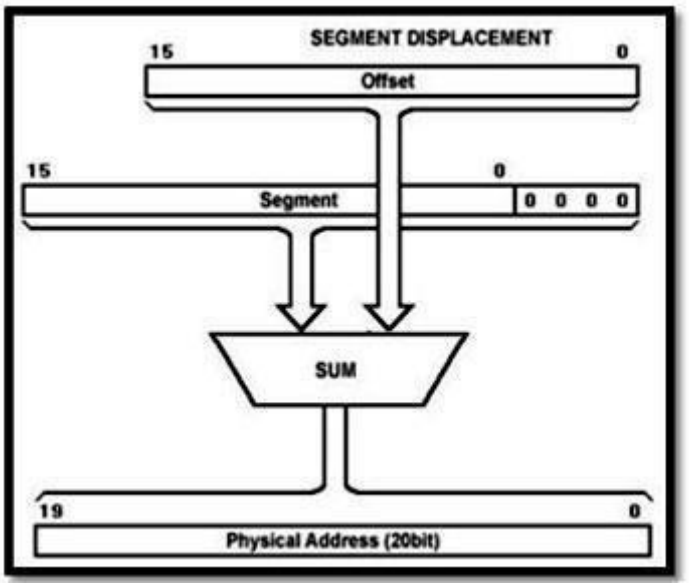
Thepointtonoteisthatthebeginningsegmentaddressmustbeginat an address divisible by 16.Also note that the four segments need not bedefinedseparately.Itisallowableforallfoursegmentstocompletelyoverlap (CS=DS=ES=SS).

LogicalandPhysical Address

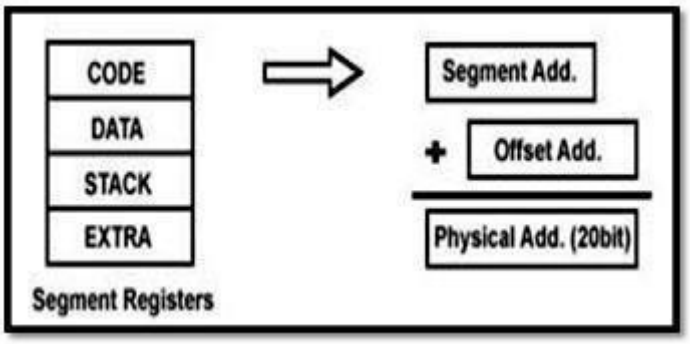
Addresseswithinasegmentcanrangefromaddress00000htoaddress 0FFFFh. This corresponds to the 64K-bytelength of the segment. Anaddresswithinasegmentis
calledanoffsetor logicaladdress.

A logical address gives the displacement from the base address of thesegment to the desired location within it, as opposed to its "real" address,which maps directly anywhere into the 1 MByte memory space. This "real"address is called the physical address.

What is the difference between the physical and the logical address?The physical address is 20 bits long and corresponds to the actual binarycode output by the BIU on the address bus lines. The logical address is anoffsetfromlocation0ofagivensegment.

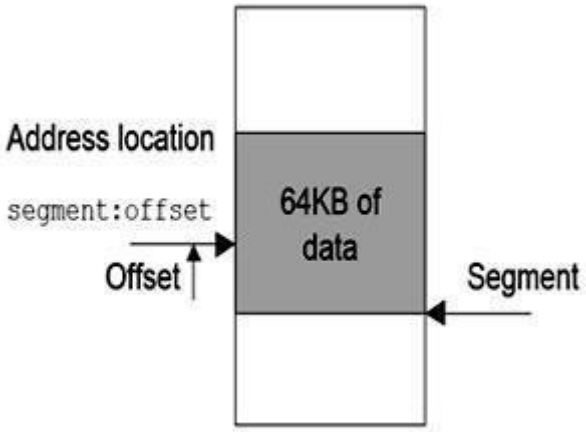


You should also be careful when writing addresses on paper to do soclearly. To specify the logical address XXXX in the stack segment, use theconvention SS:XXXX, whichisequal to[SS]*16+XXXX.



Logical address is in the form of: Base Address: Offset Offset is thedisplacementofthememorylocationfromthestartinglocationofthesegmen t. To calculate the physical address of the memory, BIU uses thefollowingformula:

PhysicalAddress=Base AddressofSegment*16+Offset



Example:

ThevalueofDataSegmentRegister (DS)is2222H.

To convert this 16-bit address into 20-bit, the BIU appends 0H to the LSB (bymultiplying with 16) of the address. After appending, the starting address oftheData Segmentbecomes 22220H.

Dataatanylocationhasalogicaladdressspecifiedas:2222H:0016H

Where 0016H is the offset, 2222 H is the value of DS Therefore the physicaladdress:22220H+0016H
:22236H

Thefollowingtabledescribesthedefaultoffsetvalues tothecorrespondingmemorysegments.

Segment	Offset Registers	Function
CS	IP	Address of the next instruction
DS	BX, DI, SI	Address of data
SS	SP, BP	Address in the stack
ES	BX, DI, SI	Address of destination data (for string operations)

Someoftheadvantagesofmemorysegmentationinthe 8086areasfollows:

- Withthehelpofmemorysegmentationa userisable to workwithregistershavingonly16-bits.
- Thedataandthe user'scodecanbestoredseparatelyallowingformore flexibility.

- Alsodue tosegmentationthelogicaladdressrange isfrom0000HtoFFFFHthecodecanbeloadedatanylocationinthememor y.

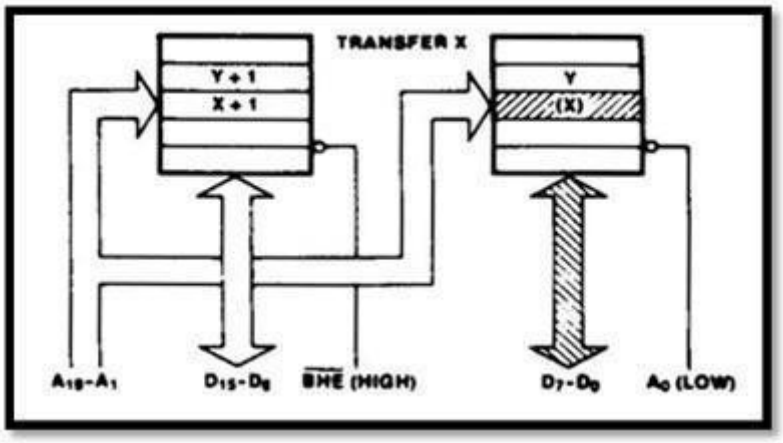
Physicalmemoryorganization:

The8086's1Mbytememoryaddressspaceisdividedintotwoindependent 512Kbyte banks: the low (even) bank and the high (odd) bank.Data bytes associated with an even address (0000016, 0000216, etc.) residein the low bank, and those with odd addresses (0000116, 0000316, etc.)resideinthehigh bank.

Address bits A1 through A19 select the storage location that is to beaccessed.Theyareappliedtobothbanksinparallel.A0andbankhighenable(BH E)areusedasbank-selectsignals.

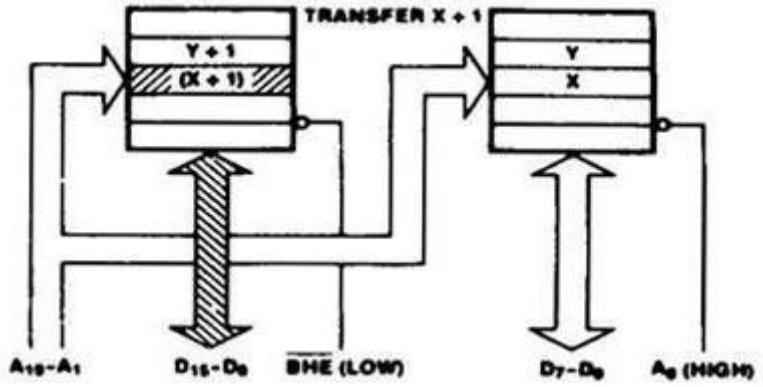
Thefourdifferentcasesthathappenduringaccessingdata:

Case1: Whenabyteofdataatanevenaddress(suchasX)istobeaccessed:



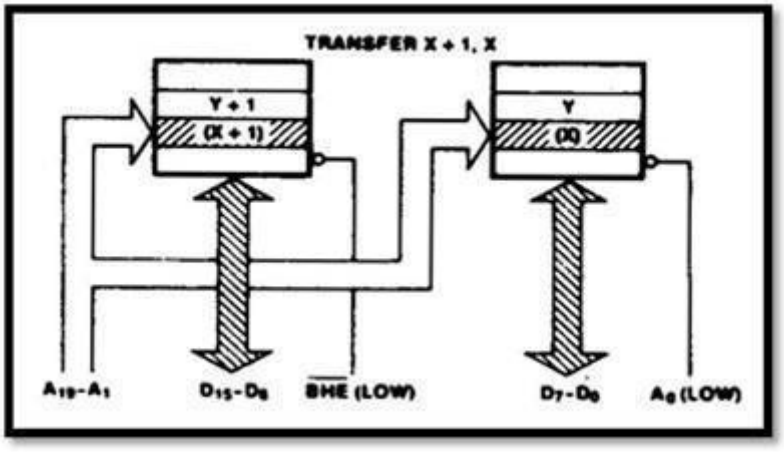
- A0issettologic0toenablethelowbankofmemory.
- BHEissettologic 1todisablethe highbank.

Case2: Whenabyteofdataatanoddaddresses(suchasX+1)istobeaccessed:



- A0issettologic1todisablethelowbankofmemory.
- BHEissettologic0toenable thehighbank.

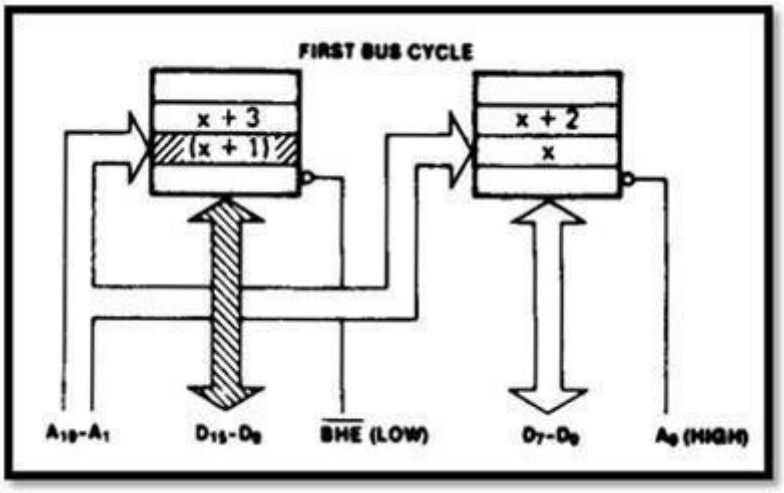
Case3: Whenawordofdataatanevenaddress(alignedword)istobeaccessed:



- A0issettc
- BHEissettologic0toenable thehighbank.

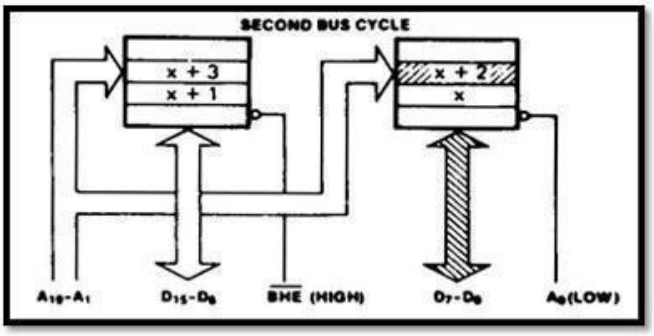
Case4: Whenawordofdataatanodd address(misalignedword)istobeaccessed,then the8086 need two bus cycles to accessit:

a) Duringthefirstbuscycle,theoddbyteofthe word(inthe highbank)isaddressed



- A0issettologic1to disable the lowbankofmemory
- BHEissettologic0toenable thehighbank.

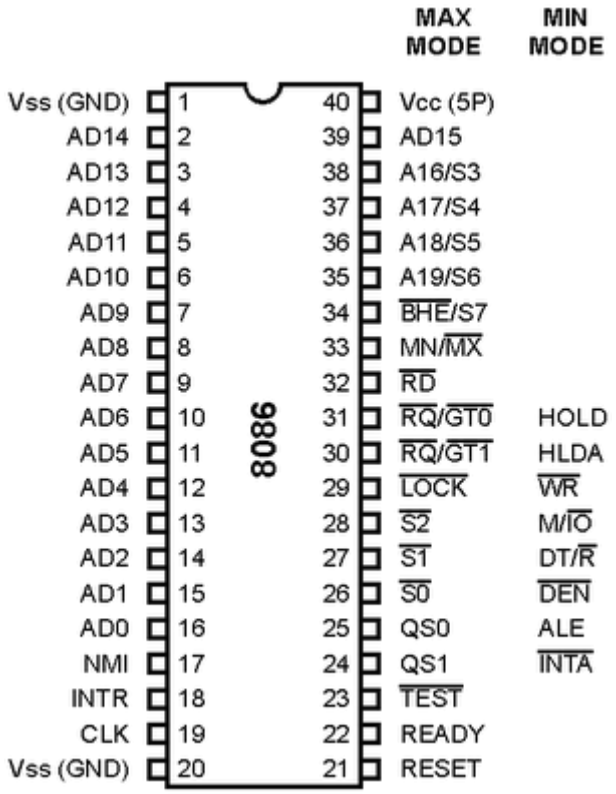
b) Duringthesecondbuscycle,theoddbyteoftheword(inthelowbank)isaddressed



- A0issettologic0toenablethelowbankofmemory.
- BHEissetto logic 1to disablethehighbank.

SignalDescriptionof8086Microprocessor

The 8086 Microprocessor is a 16-bit CPU available in 3 clock rates, i.e.5, 8 and 10MHz, packaged in a 40 pin CERDIP or plastic package. The 8086Microprocessoroperatesinsingleprocessorormultiprocessorconfigurations toachievehighperformance.Thepin configuration isasshown in fig1. Some of the pins serve a particular function in minimummode(singleprocessormode)andothersfunctioninmaximummode(multiprocessormode)configuration.



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

Thefollowingsignaldescriptioniscommonforboththeminimumand maximummodes.

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 onthe data bus during T2, T3, TW and T4. Here T1, T2, T3, T4 and TW are theclockstatesofamachinecycle.TWisawaitstate.Theselinesareactivehigh and float to a tristate during interrupt acknowledge and local bus holdacknowledgecycles.

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 or memory operations. DuringI/O operations, these lines are low. During memory or I/O operations, statusinformationisavailableonthoselinesforT2,T3,TWandT4.Thestatusof

the interrupt enable flag bit(displayed on S5) is updated at the beginning ofeach clock cycle. The S4 and S3 combinedly indicate which segment registerispresentlybeingusedformemoryaccessesas shown inTable1.1.

These lines float to tri-state off (tristated) during the local bus holdacknowledge. The status line S6 is always low(logical). The address bits areseparatedfromthestatusbitsusinglatchescontrolledby theALEsignal.

S4	S3	Indication
0	0	Alternate Data
0	1	Stack
1	0	Code or none
1	1	Data

Table 1.1 Bus High Enable/Status

BHE/S7(ActiveLow):

The bus high enable signal is used to indicate the transfer of data over thehigher order (D15-D8) data bus as shown in Table 1.2. It goes low for thedata transfers over D15-D8 and is used to derive chip selects of odd addressmemory bank or peripherals. $\overline{BHE}$ is low during T1 for read, write andinterrupt acknowledge cycles, when- ever a byte is to be transferred on thehigherbyteofthedatabus.ThestatusinformationisavailableduringT2,T3 and T4. The signal is active low and is tristated during 'hold'. It is lowduring T1forthefirstpulseoftheinterruptacknowledgecycle.

$\overline{BHE}$	A_0	Indication
0	0	Whole Word
0	1	Upper byte from or to odd address
1	0	Upper byte from or to even address
1	1	None

$\overline{RD}$-Read:

Read signal, when low, indicates the peripherals that the processor isperforming a memory or I/O read operation. $\overline{RD}$ is active low and showsthe state for T2,T3, TW of any read cycle.The signalremainstristatedduringthe'holdacknowledge'.

READY:

This is the acknowledgement from the slow devices or memory thatthey have completed the data transfer. The signal made available by thedevices is synchronized by the 8284A clock generator to provide ready inputto the8086. Thesignalis activehigh.

INTR-InterruptRequest:

This is a level triggered input. This is sampled during the last clockcycle of each instruction to determine the availability of the request. If anyinterruptrequestispending,theprocessorentherstheinterruptacknowledge cycle. This can be internally masked by resetting the interruptenableflag.Thissignalisactivehighandinternallysynchronized.

TEST:

This input is examined by a 'WAIT' instruction. If the TEST input goeslow, execution will continue, else, theprocessor remainsin an idle state.The input is synchronized internally during each clock cycle on leading edgeofclock.

NMI-Non-maskableInterrupt:

This is an edge-triggered input which causes a Type2 interrrupt. TheNMI is not maskable internally by software. A transition from low to highinitiates the interrupt response at the end of the current instruction. Thisinputisinternallysynchronized.

RESET:

This input causes the processor to terminate the current activity andstart execution from FFFF0H. The signal is active high and must be active forat least four clock cycles. It restarts execution when the RESET returns low.RESETis also internallysynchronized.

CLK-ClockInput:

The clock input provides the basic timing for processor operation andbus control activity. Its an asymmetric squarewavewith 33% duty cycle.Therangeoffrequencyfordifferent8086versionsisfrom5MHzto10MHz.

VCC:

+5Vpowersupplyfortheoperationofthe internalcircuit.GNDground fortheinternalcircuit.

MN/MX:

The logic level at this pin decides whether the processor is to operatein either minimum (single processor) or maximum (multiprocessor) mode.Thefollowingpinfunctionsareforthemodeoperationof8086.

M/IO-Memory/IO:

Thisis a statuslinelogically equivalentto S2 inmaximummode.Whenitislow,itindicatestheCPUishavinganI/Ooperation,and whenitis high, it indicates that the CPU is having a memory operation. This linebecomes active in the previous T4 and remains active till final T4 of thecurrentcycle. Itistristatedduringlocalbus"holdacknowledge".

$\overline{\text{INTA}}$-InterruptAcknowledge:

This signal is used as a read strobe for interrupt acknowledge cycles.In other words, when it goes low, it means that the processor has acceptedtheinterrupt.ItisactivelowduringT2,T3andTWofeachinterruptackno wledgecycle.

ALE-AddresslatchEnable:

This output signal indicates the availability of the valid address on theaddress/data lines, and is connected to latch enable input of latches. Thissignalis activehighandisnevertristated.

DT $\overline{\text{/R}}$-DataTransmit/Receive:

This output is used to decide the direction of data flow through thetransreceivers (bidirectional buffers). When the processor sends out data,this signal is high and when the processor is receiving data, this signal is low.Logically,thisisequivalenttoS1inmaximummode.ItstimingisthesameasM/I /O. Thisis tristatedduring'holdacknowledge'.

$\overline{\text{DEN}}$-Data Enable

Thissignalindicatestheavailabilityofvaliddataovertheaddress/datalines. It isusedtoenablethetransreceivers(bidirectionalbuffers) to separate the data from the multiplexed address/data signal. It isactive from the middle ofT2 until the middle of T4 DEN is tristated during'holdacknowledge' cycle.

HOLD,HLDA-Hold/HoldAcknowledge:

Whenthe HOLDline goes high, it indicates to theprocessor thatanother master is requesting the bus access. The processor, after receivingthe HOLD request, issues the hold acknowledge signal on HLDA pin, in themiddleofthenextclockcycleaftercompletingthecurrentbus(instruction)

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 HLDA signal.HOLDisanasynchronousinput,anditshouldbeexternallysynchronized.

S2, S1,S0-StatusLines:

These are the status lines which reflect the type of operation, beingcarried out by the processor. These become active during T4 of the previouscycleandremainactiveduringT1 andT2 ofthecurrentbuscycle.Thestatus lines return to passive state during T3 of the current bus cycle so thatthey may again become active for the next bus cycle during T4. Any changein these lines during T3 indicates the starting of a new cycle, and return topassive state indicates end of the bus cycle. These status lines are encodedin table1.3

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	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

Table 1.3

$\overline{\text{LOCK}}$:

This output pin indicates that other system bus masters will beprevented from gaining the system bus, whilethe $\overline{\text{LOCK}}$ signal is low.The $\overline{\text{LOCK}}$ signal is activated by the 'LOCK' prefix instruction and remainsactive until the completion of the next instruction. This floats to tri-stateoff during"hold acknowledge".Whenthe CPUisexecuting a criticalinstruction which requires the system bus, the LOCK prefix instructionensures that other processors connected in the system will not gain thecontrol of the bus. The 8086, while executing the prefixed instruction,assertsthebuslocksignaloutput,whichmaybeconnectedtoanex ternal bus controller.

QS1,QS0-QueueStatus:

These lines give information about the status of the codeprefetchqueue. These areactiveduringtheCLKcycle afterwhich thequeueoperationisperformed.Theseareencodedasshowninable1.4.

QS_1,	QS_0	Indication
0	0	No operation
0	1	First byte of opcode from the queue
1	0	Empty queue
1	1	Subsequent byte from the queue

Table 1.4

$\overline{RQ}/\overline{GT_0}$, $\overline{RQ}/\overline{GT_1}$- ReQuest/Grant:

These pins are used by other local bus masters, in maximum mode, toforce the processor to release the local bus at the end of the processor'scurrentbuscycle.Eachofthepinsisbidirectionalwith $\overline{RQ}/\overline{GT_0}$ having higher priority than $\overline{RQ}/\overline{GT_1}$, $\overline{RQ}/\overline{GT}$ pins have internal pull-up resistors andmay beleft unconnected.Therequest!Grantsequenceisasfollows:

1. Apulseoneclockwidefromanotherbusmaster requests thebusaccessto 8086.

2. During T4 (current) or T1 (next) clock cycle, a pulse one clock wide from8086 to the requesting master, indicates that the 8086 has allowed the localbus to float and that it will enter the "hold acknowledge" state at next clockcycle. The CPU's bus interface unit is likely to be disconnected from the localbusofthesystem.

3. A one clock wide pulse from the another master indicates to 8086 thatthe 'hold' request is about to end and the 8086 may regain control of thelocalbus atthenextclockcycle.

MinimumMode8086SystemandTimings

Inaminimummode8086system,themicroprocessor8086isoperated in minimum mode by strapping its MN/MX* pin to logic1. In thismode, all the control signals are given out by the microprocessor chip itself.Thereisasinglemicroprocessorintheminimummodesystem.Theremainin gcomponentsinthesystemarelatches,transreceivers,clockgenerator, memory and I/O devices. Some type of chip selection logic mayberequiredforselectingmemoryorI/Odevices,dependingupontheaddress mapof thesystem.

Latches:

Thelatchesaregenerallybufferedoutput D-typeflip-flops,like,74LS373 or 8282. They are used for separating the valid address from themultiplexedaddress/datasignalsandarecontrolledbytheALEsignalgenerated by8086.

Transreceivers

Transreceivers are the bidirectional buffers and some times they arecalled as data amplifiers. They are required to separate the valid data fromthetimemultiplexedaddress/datasignal.Theyarecontrolledbytwosignals, namely, DEN* and DT/R*. The DEN* signal indicates that the validdataisavailableonthedatabus,whileDT/Rindicatesthedirectionofdata, i.e.fromor totheprocessor.

Memory:

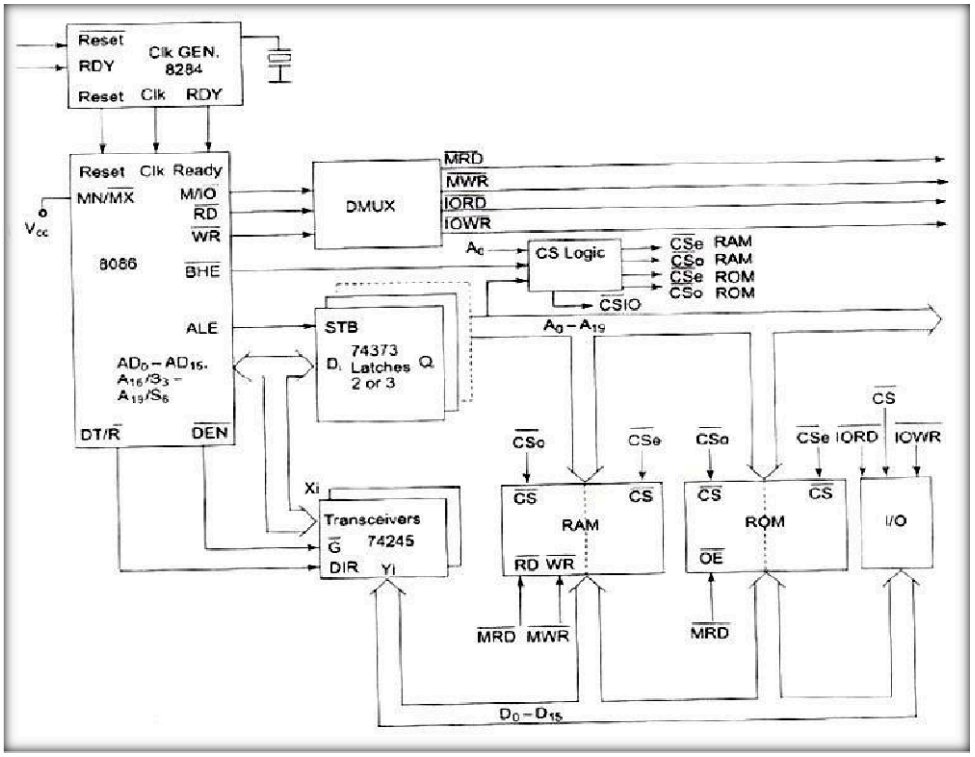
Thesystemcontainsmemoryforthemonitorandusersprogramstorage. Usually, EPROMS are used for monitor storage, while RAMs forusersprogramstorage.

IODevices:

AsystemmaycontainI/Odevicesforcommunicationwiththeprocessoraswellasso mespecialpurposeI/Odevices.

ClockGenerator:

Theclock generatorgeneratestheclock from thecrystal oscillatorandthenshapes it anddividesto makeit moreprecise sothat it canbeused as an accurate timing reference for the system. The clock generatoralsosynchronizessomeexternalsignalswiththesystemclock.



The general system organization is shown in above fig .Since it has 20address lines and 16 data lines, the 8086 CPU requires three octal addresslatchesandtwooctaldatabuffersforthecompleteaddressanddatasepar ation.

Theworkingoftheminimummodeconfigurationsystemcanbebetter described in terms of the timing diagrams rather than qualitativelydescribing the operations. The opcode fetch and read cycles are similar.Hencethetimingdiagramcan becategorizedintwoparts.

1) Timingdiagramforreadcycle
2) Timingdiagramforwritecycle.

TimingdiagramforReadcycle:

The read cycle begins in T1 with the assertion of the address latchenable (ALE) signal and also M/IO* signal. During the negative going edge ofthissignal,thevalidaddressislatchedon thelocalbus.The BHE* andA0 signals address low, high or both bytes. From Tl to T4, the M/IO* signalindicates a memory or I/O operation. At T2 the address is removed from thelocal bus and is sent to the output. The bus is then tristated. The read (RD*)controlsignalis alsoactivatedinT2.

The read (RD) signal causes the addressed device to enable its databusdrivers.AfterRD*goeslow,the validdataisavailableonthedata bus.

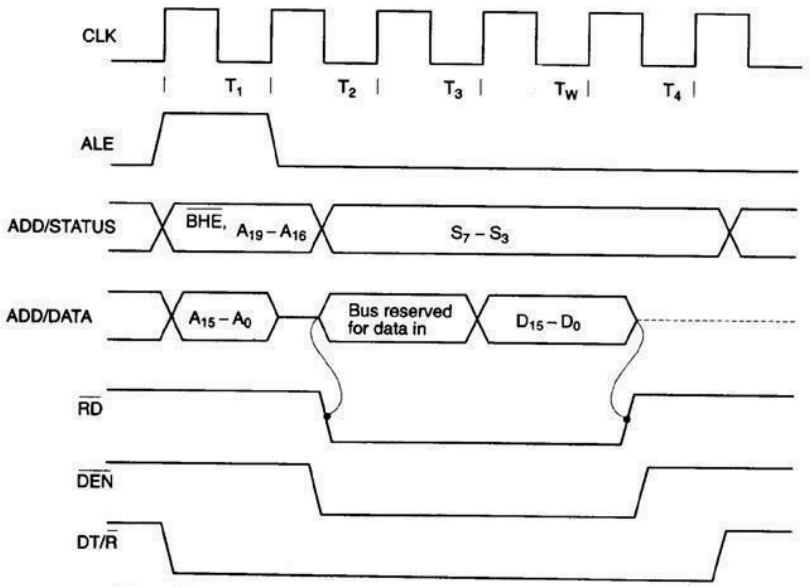
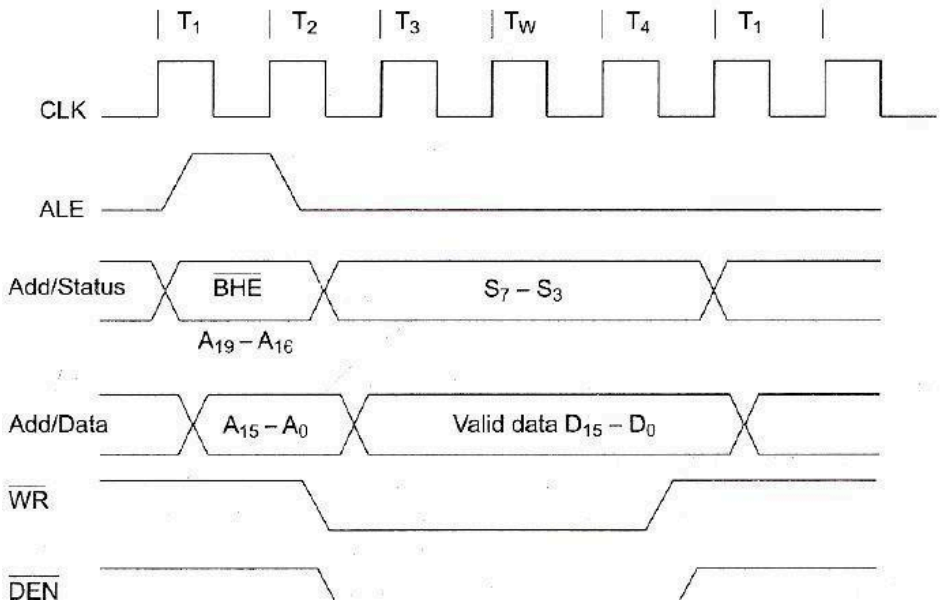


Fig. 1.9(a) *Read Cycle Timing Diagram for Minimum Mode*

TheaddresseddevicewilldrivetheREADYlinehigh,whentheprocessor returns the read signal to high level, the addressed device willagain tristateits busdrivers.

Timingdiagramfor writecycle:

A write cycle also begins with theassertion of ALE and the emissionof the address. The M/IO* signal is again asserted to indicate a memory orI/O operation. In T2 after sending the address in Tl the processor sends thedata to be written to the addressed location. The data remains on the busuntilmiddleofT4 state.TheWR*becomesactiveatthebeginningofT2.

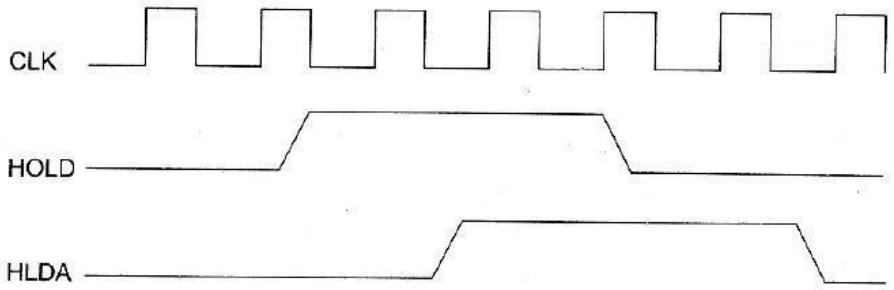


TheBHE*andA0signalsareusedtoselecttheproperbyteorbytesof memory or I/O word to be read or written. The M/IO*, RD* and WR*signalsindicatethetypes ofdatatransferasspecifiedinTable

M/IO	RD	WR	Transfer Type
0	0	1	I/O read
0	1	0	I/O write
1	0	1	Memory read
1	1	0	Memory write

HOLDResponseSequence

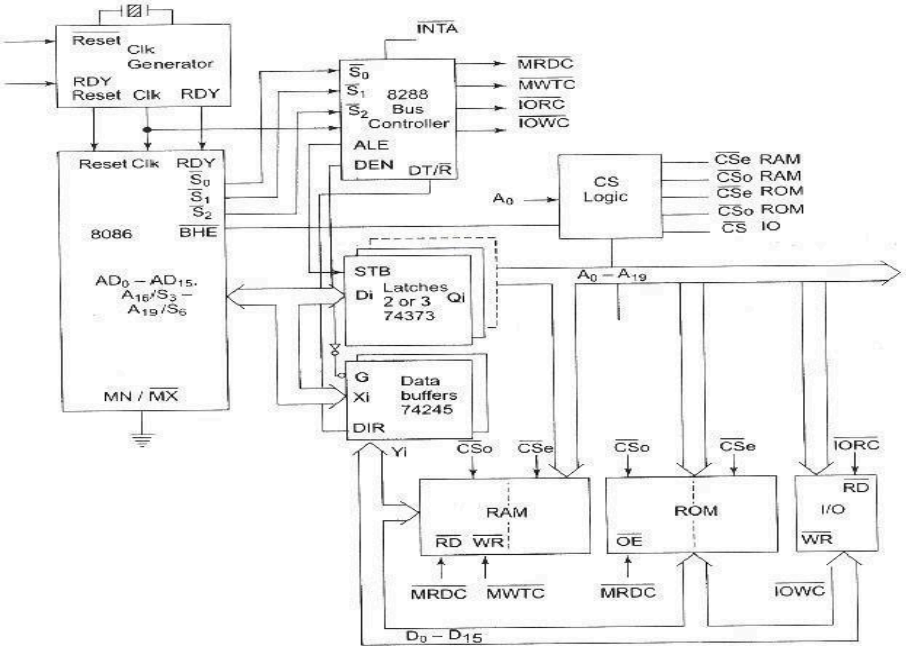
The HOLD pin is checked at the end of the each bus cycle. If it isreceivedactivebytheprocessorbeforeT4ofthepreviouscycleorduringT1 state of the current cycle, the CPU activities HLDA in the next clock cycleandforthesucceedingbuscycles,thebuswillbegiventoanotherrequesting master The control control of the bus is not regained by theprocessor until the requesting master does not drop the HOLD pin low.Whentherequestisdroppedbytherequestingmaster,theHLDAisdropped by the processor at the trailing edge of the next clock as shown infig



MaximumMode8086SystemandTimings

Inthemaximummode,the8086isoperatedbystrappingtheMN/MX*pin toground. In thismode, theprocessor derivesthe statussignals S2*, S1* and S0*. Another chip called bus controller derives thecontrol signals using this status information. In the maximum mode, theremay be more than one microprocessor in the system configuration. Theother components in the system are the same as in the minimum modesystem.Thegeneralsystemorganizationisas showninthefig1.1

The basic functions of the bus controller chip IC8288, is to derivecontrolsignalslikeRD*andWR*(formemoryandI/Odevices),DEN*,DT/R* , ALE, etc. using the information made available by the processor onthe status lines. The bus controller chip has input lines S2*, S1* and S0* andCLK. These inputs to 8288 are driven by the CPU. It derives the outputs ALE,DEN*,DT/R*,MWTC*,AMWC*,IORC*,IOWC*andAIOWC*.TheAEN*,IOB andCENpinsarespeciallyusefulformultiprocessorsystems.AEN*andIOBaregener allygrounded. CENpin isusuallytiedto +5V.



INTA* pin is used to issue two interrupt acknowledge pulses to theinterrupt controller or to an interrupting device.IORC*, IOWC* are I/O readcommand and I/O write command signals respectively. These signals enablean IO interface to read or write the data from or to the addressed port. TheMRDC*, MWTC* are memory read command and memory write commandsignals respectively and may be used as memory read and write signals. Allthese command signals instruct the memory to accept or send data from orto the bus. For both of these write command signals, the advanced signalsnamelyAIOWC*andAMWTC*areavailable.Theyalsoservethesamepurpose,butareactivatedoneclockcycleearlierthantheIOWC*andMWTC* signals, respectively. The maximum mode system is shown in fig.1.1.

The maximum mode system timing diagrams are also divided in twoportionsasread(input)andwrite(output)timingdiagrams.Theaddress/data and address/status timings are similar to the minimum mode.ALE is asserted in T1, just likeminimummode. The only differenceliesinthe status signals used and the available control and advanced commandsignals.Thefig.1.2showsthemaximummodetimingsforthereadoperationwhilethefig.1.3showsthesameforthewriteoperation.

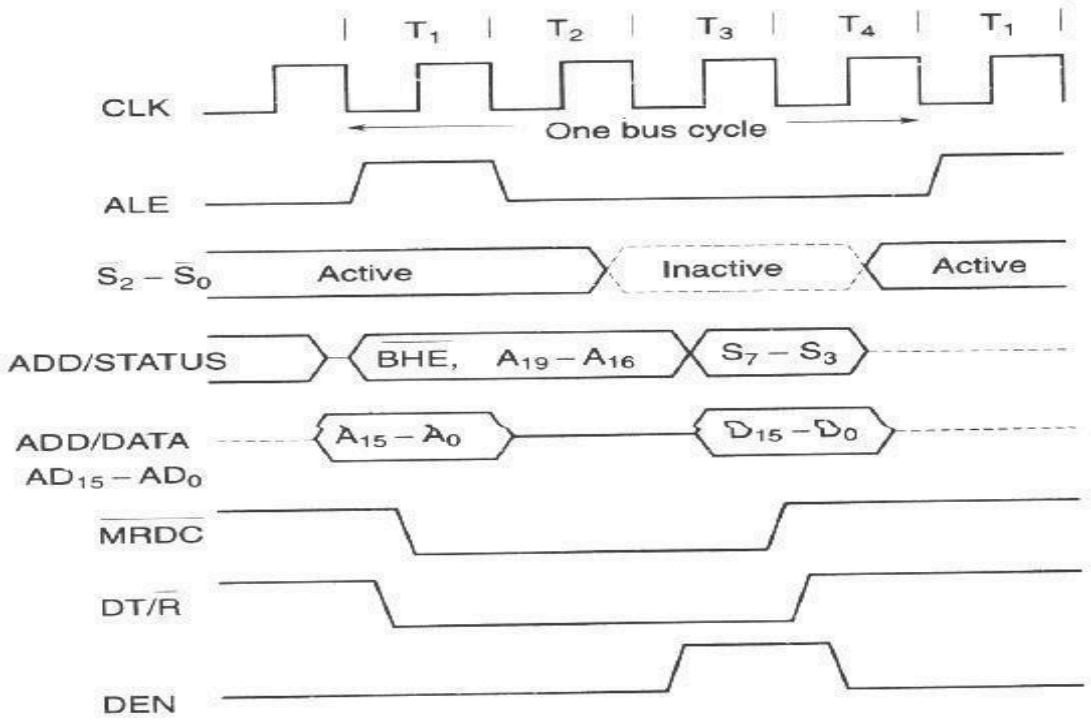


Fig.1.2MemoryReadTiminginMaximumMode

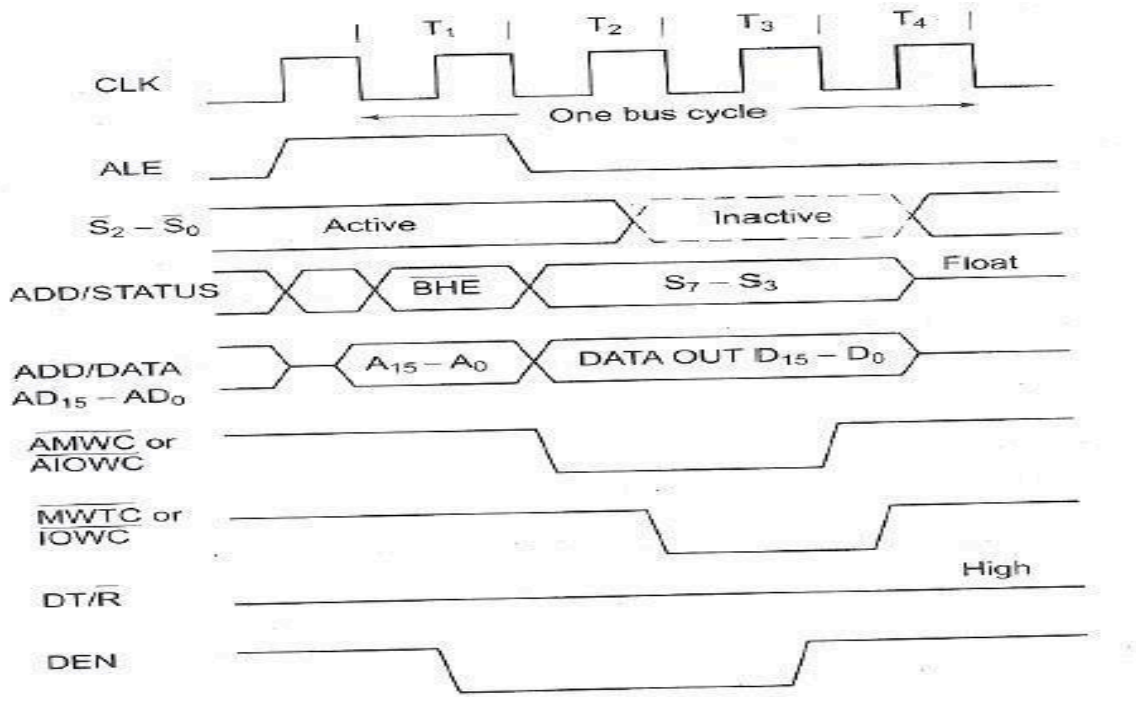


Fig.1.3MemoryWriteTiminginMaximumMode

UNIT-II

- InstructionSetandAssemblyLanguageProgrammingof8086
- Instructionformats,Addressingmodes,
- InstructionSet
- AssemblerDirectives,
- Procedures,Macros
- SimplePrograms involvingLogical
- BranchandCallInstructions
- SortingEvaluatingArithmeticExpressions
- StringManipulations

UNIT-II

Theinstructionformatcontainstwofields

- operationcode/opcode
- Operandfield

OPERATIONCODE/OPCODE:

- Itindicates thetypeof theoperationto beperformedbyCPU
- Example: MOV,ADD…

OPERAND:

- TheCPUexecutestheinstructionusingtheinformationresidesinthesefields.

Therearesixgeneralformatsofinstructionsin8086instructionset.Theinstr uctionof 8086varyfrom1to 6bytes length

ONEBYTEINSTRUCTION:

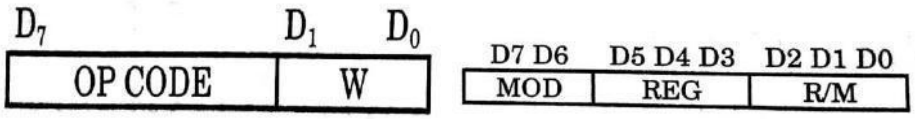
- Itisonlyonebytelongandmayhaveimplieddataorregisteroperands.
- Theleastthreesignificant3bitsoftheopcodeareusedforspecifyingregisteroperandifanyot herwiseallthe8bitsforman opcodeandtheoperandsareimplied.

REGISTERTOREGISTER

- Theformatis2bytelong
- Thefirstbyteofthecodespecifiestheopcodeandwidth
- Thesecondbyteofthecodeshowstheregisteroperandand R/Mfield
- TheRegisterrepresentedbyREGisoneoftheoperands.TheR/Mfieldspecifiesanother registerormemory location.ietheotheroperand

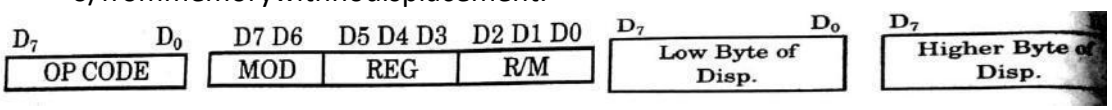
REGISTERTO/FROMMEMORYWITHNODISPLACEMENT

- Theformatis2bytelong
- Thisissimilartotheregistertoregisterformatexceptfortheb MODfieldisshown.
- TheMODfieldshowsthemodeofaddressing



REGISTERTO/FROMMEMORYWITHDISPLACEMENT

- Theformatcontainsoneortwoadditionalbytesfordisplacementalongwith2bytesRegistert o/frommemorywithnodisplacement.



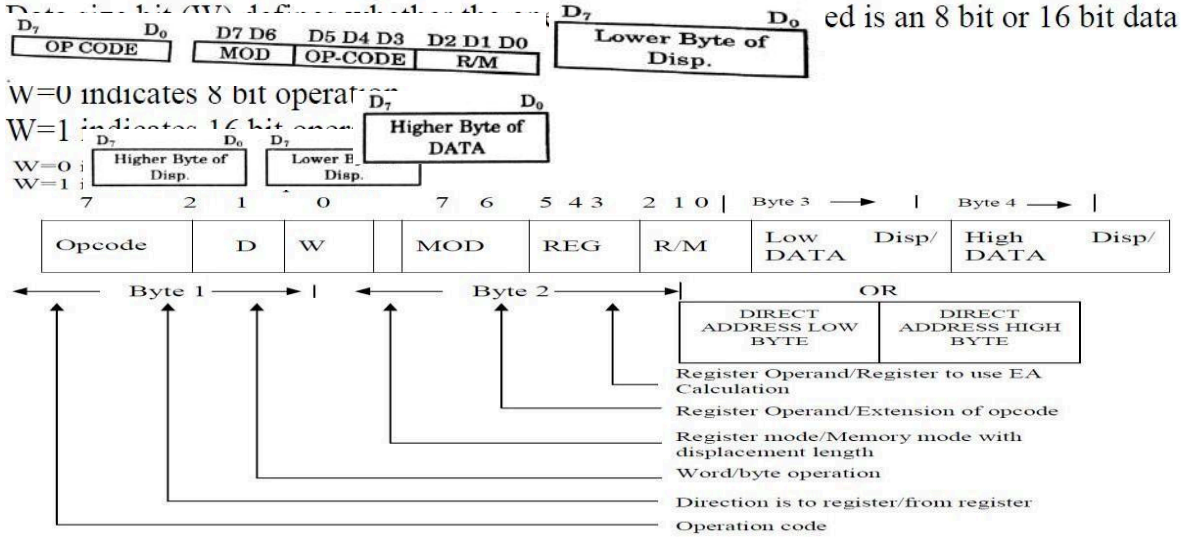
bitsfromthesecondbytewhichareusedforREGfieldincaseofRegistertoregisterformator usedforOPCODE.

- Italsocontainsonearetwobytesofdata.

The 8086 instruction sizes vary from one to six bytes. Depending on the type of coding, an instruction may have more than one Hexcode, (not unique as in 8085)

The OP code field occupies 6-bits.It defines the operation to be carried out by the instruction.

Register Direct bit (D) occupies one bit. It defines whether the register operand in byte 2 is the source or destination operand.
D=1 Specifies that the register operand is the destination operand.
D=0 indicates that the register is a source operand.



This byte contains 3 fields. These are the mode (MOD) field, the register (REG) field and the Register/Memory (R/M) field.

MOD (2 bits)	**Interpretation**
00	Memory mode with no displacement follows except for 16 bit displacement when R/M=110
01	Memory mode with 8 bit displacement
10	Memory mode with 16 bit displacement
11	Register mode (no displacement)

Register field occupies 3 bits. It defines the register for the first operand which is specified as source or destination by the D bit.

REG	W=0	W=1
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

The R/M field occupies 3 bits. The R/M field along with the MOD field defines the second operand as shown below.

MOD 11

R/M	W=0	W=1
000	AL	AX
001	CL	CX
010	DL	DX
011	BL	BX
100	AH	SP
101	CH	BP
110	DH	SI
111	BH	DI

Effective Address Calculation

R/M	MOD=00	MOD 01	MOD 10
000	(BX) + (SI)	(BX)+(SI)+D8	(BX)+(SI)+D16
001	(BX)+(DI)	(BX)+(DI)+D8	(BX)+(DI)+D16
010	(BP)+(SI)	(BP)+(SI)+D8	(BP)+(SI)+D16
011	(BP)+(DI)	(BP)+(DI)+D8	(BP)+(DI)+D10
100	(SI)	(SI) + D8	(SI) + D16
101	(DI)	(DI) + D8	(DI) + D16
110	Direct address	(BP) + D8	(BP) + D16
111	(BX)	(BX) + D8	(BX) + D16

In the above, encoding of the R/M field depends on how the mode field is set. If MOD=11 (register to register mode), this R/M identifies the second register operand.

MOD / R/M	Memory Mode (EA Calculation)			Register Mode	
	00	01	10	W=0	W=1
000	(BX)+(SI)	(BX)+(SI)+d8	(BX)+(SI)+d16	AL	AX
001	(BX) + (DI)	(BX)+(DI)+d8	(BX)+(DI)+d16	CL	CX
010	(BP)+(SI)	(BP)+(SI)+d8	(BP)+(SI)+d16	DL	DX
011	(BP)+(DI)	(BP)+(DI)+d8	(BP)+(DI)+d16	BL	BX
100	(SI)	(SI) + d8	(SI) + d16	AH	SP
101	(DI)	(DI) + d8	(DI) + d16	CH	BP
110	d16	(BP) + d8	(BP) + d16	DH	SI
111	(BX)	(BX) + d8	(BX) + d16	BH	DI

MOD selects memory mode, then R/M indicates how the effective address of the memory operand is to be calculated. Bytes 3 through 6 of an instruction are optional fields that

segment override prefix byte (SOP byte) to start with. The SOP byte is **001 xx 110**, where SR value is provided as per table shown below.

xx	Segment register
00	ES
01	CS
10	SS
11	DS

To specify DS register, the SOP byte would be 001 11 110 = 3E H. Thus the 5 byte code for this instruction would be 3E 89 96 45 23 H.

SOP	Opcode	D	W	MOD	REG	R/M	LB disp.	HD disp.
3EH	1000 10	0	1	10	010	110	45	23

Suppose we want to code MOV SS : 2345 (BP), DX. This generates only a 4 byte code, without SOP byte, as SS is already the default segment register in this case.

Example 5:
Give the instruction template and generate code for the instruction ADD 0FABE [BX], [DI], DX (code for ADD instruction is 000000)

ADD 0FABE [BX] [DI], DX
Here we have to specify DX using REG field. The bit D is 0, indicating that DX is the source register. The REG field must be 010 to indicate DX register. The w must be 1 to indicate it is a word operation. FABE (BX + DI) is specified using MOD value of 10 and R/M value of 001 (from the summary table). The 4 byte code for this instruction would be

Opcode	D	W	MOD	REG	R/M	16 bit disp.		=01 91 BE FAH
000000	0	1	10	010	001	BEH	FAH	

Example 6 :
Give the instruction template and generate the code for the instruction MOV AX, [BX] (Code for MOV instruction is 100010)
AX destination register with D=1 and code for AX is 000 [BX] is specified using 00 Mode and R/M value 111. It is a word operation

Opcode	D	W	Mod	REG	R/M	=8B 07H
100010	1	1	00	000	111	

INPUT/OUTPUT INSTRUCTIONS:

IN acc, port : In transfers a byte or a word from input port to the AL register or the AX register respectively. The port number my be specified either with an immediate byte constant, allowing access to ports numbered 0 through 255 or with a number previously placed in the DX register allowing variable access (by changing the value in DX) to ports numbered from 0 through 65,535.

In Operands	Example
acc, immB	IN AL, 0E2H (OR) IN AX, PORT
acc, DX	IN AX, DX (OR) IN AL, DX

ADDRESSINGMODESOF8086

Accordingtwotheflowofinstructionsmaybecategorizedas

1. SequentialControlflowinstructions
2. Controltransferinstructions

Sequentialcontrolflowinstructionsaretheinstructionswhichafterexecutiontransfer control to the next instruction appearing immediately. The control transferinstructions transfer control to some predefined address or the address somehowspecifiedintheinstructionaftertheirexecution.

Whatisaddressingmode?

The different ways in which a source operand is denoted in an instruction are knownas addressing mode theaddressing modes for sequential control flowinstructionsare

1. ImmediateAddressingMode
2. DirectAddressingmode
3. RegisterAddressingmode
4. RegisterIndirectAddressingmode
5. IndexedAddressingMode
6. RegisterRelativeaddressingmode
7. Basedindexedaddressingmode
8. RelativebasedindexedAddressingmode

IMMEDIATEADDRESSINGMODE

Theaddressingmodeinwhichthedataoperandisapartoftheinstructionitselfisknownas immediateaddressing mode.

Example

MOVDL,08H

The8-bitdata

(08_H)giveninstructionismovedtoDL(DL) $\leftarrow 08_H$

MOVAX,0A9FH

The16-bitdata(0A9FH)giveninthe

instructionismovedtoAXregister(AX) $\leftarrow$ 0A9FH

DIRECTADDRESSINGMODE

The addressing mode in which the effective address of the memory location at whichthe data operand is stored is given in the instruction. The effective address (Offset) isjusta16-bitnumberwrittendirectlyintheinstruction.

Example:MOVBX,[1354H]M
OVBL,[0400H]

The square brackets around the 1354H denote the contents of the memory location.When executed, this instruction will copy the contents of the memory location into BXregister. This addressing mode is called direct because the displacement of the operandfromthesegmentbaseisspecifieddirectlyintheinstruction.

REGISTERADDRESSINGMODE

Theinstructionwillspecifythenameoftheregisterwhichholdsthedatatobeoperatedbytheinstruction.AllregistersexceptIPmaybeusedinthismode

Example:

MOVCL,DH
ThecontentofB-bitregisterDHismovedtoanother8-bitregisterCL(
CL) ← (DH)

REGISTERINDIRECTADDRESSINGMODE

Thisaddressingmodeallowsdatatobeaddressedatanymemorylocationthroughanoffsetaddress held inany ofthefollowingregisters:BP,BX,DI &SI.

Example

```
MOVAX,[BX];supposetheregisterBXcontains4895H,thenthe contents
                    ; 4895HaremovedtoAX
ADDCX,{BX}
```

INDEXEDADDRESSINGMODE

In this addressing mode, the operands offset address is found by adding the contents ofSI or DI register and 8-bit/16-bit displacements. DS and ES are the default segments forindex registers SI and DI respectively. This is the special case of the of register indirectaddressingmode.

Example

MOVBX,[SI+16],ADDAL,[DI+16]

REGISTERRELATIVEADDRESSINGMODE

In register relative Addressing, BX, BP, SI and DI is used to hold the base value foreffective address and a signed 8-bit or unsigned 16-bit displacement will be specified inthe instruction. In case of 8-bit displacement, it is sign extended to 16-bit before addingtothebasevalue.WhenBXholdsthebasevalueofEA,20-bitphysicaladdressiscalculated fromBXandDS.WhenBPholdsthebasevalueofEA, BPand SSisused.

Example:

MOVAX,[BX+08H] MOVAX,08H [BX]

BASEDINDEXEDADDRESSINGMODE

In this addressing mode, the offset address of the operand is computed by summing thebase register to the contents of an Index register. The default segment registers may beESorDS

Example:

MOVDX,[BX+SI] MOVDX, [BX][SI]

RELATIVEBASEDINDEXEDADDRESSINGMODE

Inthisaddressingmode,theoperandsoffsetiscomputedbyaddingthebaseregistercontents.AnIndexregisters contentsand8or16-bitdisplacement.

Example

MOVAX,[BX+DI+08]
ADDCX,[BX+SI+16]

CONTROLTRANSFERINSTRUCTIONSADDRESSINGMODES/BRANCHADDRESSINGMODE

Thecontroltransferinstructionstransfercontroltosomepredefinedaddressortheaddresssomehowspecifiedintheinstructionaftertheirexecution

Examples:INT,CALL,RETandJUMPinstructions

The control transfer instruction the addressing modes depend upon whether destinationlocation is within the same segment or a different one .It also depends on the method ofpassingthedestinationaddress totheprocessor

Basicallytherearetwomethodsforpassingcontroltransferinstructions

1. Intersegmentaddressingmode
2. Intrasegmentaddressingmode

INTRASEGMENTADDRESSINGMODE

Ifthedestinationlocationiswithinthesamesegmentthemodeiscalledintrasegmentaddressing mode

Therearetwotypes

1. Intrasegementdirectmode
2. Intrasegmentindirectmode

INTRASEGMENTDIRECTMODE:

Inthismodetheaddresstowhichthecontrolistobetransferredlieswithinthesegmentinwhichthecontroltransferinstructionliesandappearsdirectlyintheinstruction as an immediate displacement value .The displacement is computed relativetothecontentoftheinstructionpointerIP.

JMPSHORTLABEL;

is a control transfer instruction following intra segment direct mode. Here, SHORT LABELrepresentsasigneddisplacement.

INTRASEGMENTINDIRECTMODE:

In this mode the displacement to which the control is to be transferred isin the samesegment in which the control transfer instruction lies but it is passed to the instructionindirectly Here the branch address is found as the content of a register or a memorylocation.

Example

JMP[AX]

INTERSEGMENTADDRESSINGMODE

Ifthedestinationlocationisinthedifferentsegmentthemodeiscalledintersegmentaddressingmode

Therearetwotypes

1. Intersegmentdirectmode
2. Intersegmentindirectmode

INTERSEGMENTDIRECTMODE:

In this mode the address to which the control isto be transferredis in a differentsegment this addressing mode provides a means of branching from one code segment toanother code segment. Here the CS and IP of the destination address are specifieddirectlyintheinstruction.

Example

JMP2000H:3000H;

INTERSEGMENTINDIRECTMODE:

In this the address to which the control is to be transferred lies in a different segmentand it is passed to the instruction indirectly .Content of memory block containing fourbytes IP(LSB) ,IP(MSB),CS(LSB)andCS(MSB) sequentially Thestarting address ofthememory block may be referred using any of the addressing mode except immediatemode.

Example

JMP[5000H];

INSTRUCTIONSETOF8086

The8086microprocessorsupports8typesofinstructions−

- DataTransferInstructions
- ArithmeticInstructions
- logicalInstructions
- StringInstructions
- ProgramExecutionTransferInstructions(Branch&LoopInstructions)
- ProcessorControlInstructions
- IterationControlInstructions
- InterruptInstructions

1. DATATRANSFERINSTRUCTIONS

Theseinstructionsareusedtotransferthedatafromthesourceoperandtothedestinationoperand.Followingarethelistofinstructionsunder thisgroup−

INSTRUCTIONTOTRANSFERAWORD

- **MOV**−Usedtocopythebyteorwordfromtheprovidedsourcetotheprovided destination.
- **PPUSH** −Usedtoputawordatthetop of thestack.
- **POP**−Usedtogetawordfromthetopofthestacktotheprovidedlocation.
- **PUSHA**−Usedtoputalltheregistersinto thestack.
- **POPA**−Usedtogetwords fromthestacktoallregisters.

- **XCHG**−Usedtoexchangethedatafromtwo locations.
- **XLAT**−UsedtotranslateabyteinALusingatableinthememory.

INSTRUCTIONSFORINPUTANDOUTPUTPORTTRANSFER

- **IN**−Usedtoreadabyteorword fromtheprovidedporttotheaccumulator.
- **OUT**−Usedtosendoutabyteorwordfromtheaccumulatortotheprovided port.

INSTRUCTIONSTOTRANSFERTHEADDRESS

- **LEA**−Usedtoloadtheaddressofoperandintotheprovidedregister.
- **LDS**−Usedtoload DSregisterandotherprovided registerfromthememory
- **LES**−UsedtoloadESregisterandotherprovided registerfromthememory.

INSTRUCTIONSTOTRANSFERFLAGREGISTERS

- **LAHF**−UsedtoloadAHwiththelowbyteoftheflag register.
- **SAHF**−UsedtostoreAHregistertolowbyteoftheflagregister.
- **PUSHF**−Usedtocopytheflagregisteratthetopofthestack.
- **POPF**−Usedtocopyawordat thetopofthestacktotheflagregister.

2.ARITHMETICINSTRUCTIONS

Theseinstructionsareusedtoperformarithmeticoperationslikeaddition,subtraction ,multiplication, division,etc.

Followingisthelistofinstructionsunderthisgroup−

INSTRUCTIONSTOPERFORMADDITION

- **ADD** −Usedtoaddtheprovidedbyteto byte/wordtoword.
- **ADC**−Usedtoaddwith carry.
- **INC**−Usedtoincrementtheprovidedbyte/word by1.
- **AAA**−UsedtoadjustASCII afteraddition.
- **DAA**−Usedtoadjustthedecimalafterthe addition/subtractionoperation.

INSTRUCTIONSTOPERFORMSUBTRACTION

- **SUB**−Usedtosubtractthebytefrombyte/wordfromword.
- **SBB**−Usedtoperformsubtractionwithborrow.
- **DEC**−Usedto decrementtheprovidedbyte/word by1.
- **NPG**−Usedtonegateeachbitoftheprovidedbyte/wordandadd1/2's complement.
- **CMP**−Usedtocompare2provided byte/word.
- **AAS**−UsedtoadjustASCIIcodesaftersubtraction.
- **DAS**−Usedtoadjustdecimalaftersubtraction.

INSTRUCTIONTOPERFORMMULTIPLICATION

- **MUL**−Usedtomultiplyunsignedbytebybyte/wordbyword.
- **IMUL**−Usedtomultiplysignedbytebybyte/wordbyword.
- **AAM**−UsedtoadjustASCIIcodes aftermultiplication.

INSTRUCTIONSTOPERFORMDIVISION

- **DIV**−Usedtodividetheunsignedwordbybyteorunsigneddoublewordby word.
- **IDIV**−Usedtodividethesignedwordbybyteorsigneddoublewordbyword.
- **AAD**−UsedtoadjustASCIIcodes afterdivision.
- **CBW**−Usedtofilltheupperbyteofthewordwiththecopiesofsignbitofthe lowerbyte.
- **CWD**−Usedtofilltheupperwordofthedoublewordwiththesignbitofthelowerword.

3.LOGICALINSTRUCTIONS

Theseinstructionsareusedtoperformoperationswheredatabitsareinvolved, i.e.operationslikelogical,shift,etc.

Followingisthelistofinstructionsunderthisgroup−

INSTRUCTIONSTOPERFORMLOGICALOPERATION

- **NOT**−Usedtoinverteach bitofabyteorword.
- **AND**−Usedforaddingeachbitinabyte/wordwiththecorrespondingbitin anotherbyte/word.
- **OR**−Usedtomultiplyeachbitinabyte/wordwiththecorrespondingbitinanotherbyte /word.
- **XOR** − Used to perform Exclusive-OR operation over each bit in a byte/word withthecorresponding bitinanother byte/word.
- **TEST**−Usedtoadd operandstoupdateflags,withoutaffectingoperands.

INSTRUCTIONSTOPERFORMSHIFTOPERATIONS

- **SHL/SAL**−Usedtoshiftbitsofabyte/wordtowardsleftandputzero(S)inLSBs.
- **SHR**−Usedtoshiftbitsofabyte/wordtowardstherightandputzero(S)in MSBs.
- **SAR**−Usedtoshiftbitsofabyte/wordtowardstherightandcopytheoldMSBintothene w MSB.

INSTRUCTIONSTOPERFORMROTATEOPERATIONS

- **ROL**−Usedtorotatebitsofbyte/wordtowardstheleft,i.e.MSBtoLSBandto CarryFlag[CF].
- **ROR**−Usedtorotatebitsofbyte/wordtowardstheright,i.e.LSBtoMSBand to CarryFlag[CF].

- **RCR**−Used to rotatebitsofbyte/word towardstheright,i.e.LSB toCFand CFto MSB.
- **RCL**−Usedtorotatebitsof byte/wordtowardstheleft,i.e.MSBtoCFandCFto LSB.

4.STRINGINSTRUCTIONS

Stringisagroupofbytes/wordsandtheirmemoryisalwaysallocatedinasequentialorder.

Followingisthelistofinstructionsunderthisgroup−

- **REP**−Usedtorepeatthegiveninstruction tillCX≠0.
- **REPE/REPZ**−UsedtorepeatthegiveninstructionuntilCX=0 orzeroflagZF=1.
- **REPNE/REPNZ**−UsedtorepeatthegiveninstructionuntilCX=0orzeroflagZF =1.
- **MOVS/MOVSB/MOVSW**−Usedtomovethebyte/wordfromonestringto another.
- **COMS/COMPSB/COMPSW**−Usedtocompare twostringbytes/words.
- **INS/INSB/INSW**−Usedasaninputstring/byte/wordfromtheI/Oporttotheprovided memorylocation.
- **OUTS/OUTSB/OUTSW**−Usedasanoutputstring/byte/wordfromtheprovided memorylocationtotheI/Oport.
- **SCAS/SCASB/SCASW**−Usedtoscanastringandcompareitsbytewithabytein ALorstringword withawordin AX.
- **LODS/LODSB/LODSW**−UsedtostorethestringbyteintoALorstringwordinto AX.

5. **PROGRAM EXECUTIONTRANSFERINSTRUCTIONS (BRANCH AND LOOPINSTRUCTIONS)**Theseinstructionsareusedtotransfer/branchtheinstructionsduringanexecution.Itincludesthefollowinginstructions−

Instructionstotransfertheinstructionduringanexecutionwithoutanycondition−

- **CALL**−Usedtocallaprocedureand savetheirreturnaddresstothestack.
- **RET**−Usedtoreturnfromtheproceduretothemainprogram.
- **JMP**−Usedtojumptotheprovidedaddresstoproceedtothenextinstruction.Instructionstotransferthe instructionduringanexecutionwithsomeconditions−
- **JA/JNBE**−Usedtojumpifabove/notbelow/equalinstructionsatisfies.
- **JAE/JNB**−Usedtojumpifabove/notbelowinstructionsatisfies.
- **JBE/JNA**−Usedtojumpifbelow/equal/notaboveinstructionsatisfies.
- **JC**−Usedtojumpif carryflagCF=1
- **JE/JZ**−Usedtojump ifequal/zeroflagZF=1
- **JG/JNLE**−Usedtojump ifgreater/notlessthan/equalinstructionsatisfies.

- **JGE/JNL**−Usedtojumpifgreaterthan/equal/not lessthaninstructionsatisfies.
- **JL/JNGE**−Usedtojump iflessthan/notgreaterthan/equalinstructionsatisfies.
- **JLE/JNG**−Usedtojumpiflessthan/equal/ifnotgreaterthaninstruction satisfies.
- **JNC**−Usedtojump if no carryflag(CF=0)
- **JNE/JNZ**−Usedtojump if notequal/zeroflagZF=0
- **JNO**−Usedtojumpifnooverflowflag OF=0
- **JNP/JPO**−Usedtojumpif notparity/parityoddPF=0
- **JNS**−UsedtojumpifnotsignSF=0
- **JO**−UsedtojumpifoverflowflagOF=1
- **JP/JPE**−Usedtojumpifparity/parityevenPF=1
- **JS** −Usedtojumpifsignflag SF=1

6.PROCESSORCONTROLINSTRUCTIONS

These instructionsareused tocontroltheprocessoractionbysetting/resettingtheflagvalues.

Followingaretheinstructions underthisgroup−

- **STC**−Usedtoset carryflag CFto1
- **CLC**−Usedtoclear/resetcarryflagCFto0
- **CMC**−Usedto putcomplementatthestateofcarryflagCF.
- **STD** −UsedtosetthedirectionflagDFto1
- **CLD**−Usedtoclear/resetthedirectionflagDFto0
- **STI**−Usedtosettheinterruptenableflagto1,i.e.,enableINTR input.
- **CLI**−Usedtocleartheinterruptenableflagto0,i.e.,disableINTR input.

7.ITERATIONCONTROLINSTRUCTIONS

Theseinstructionsareusedtoexecutethegiveninstructionsfornumberoftimes. Followingisthelistofinstructionsunderthisgroup−

- **LOOP**−Usedtoloopagroupofinstructionsuntiltheconditionsatisfies,i.e.,CX =0
- **LOOPE/LOOPZ**−Used toloopagroupofinstructionstillitsatisfiesZF=1&CX=0
- **LOOPNE/LOOPNZ**−UsedtoloopagroupofinstructionstillitsatisfiesZF=0& CX=0
- **JCXZ**−Usedtojumpto theprovided addressif CX=0

8.INTERRUPTINSTRUCTIONS

Theseinstructionsareusedtocalltheinterruptduringprogramexecution.

- **INT−**Usedtointerrupttheprogramduringexecutionandcallingservice specified.
- **INTO−**Usedtointerrupttheprogramduringexecutionif OF=1
- **IRET−**Usedtoreturnfrominterruptservicetothemainprogram

ASSEMBLERDIRECTIVES

Assembler directives are the Instructions to the Assembler, linker and loaderregarding theprogram being executed. also called‘pseudo instructions. Control thegenerationofmachine codes andorganizationof theprogram; but no machinecodesaregeneratedforassemblerdirectives.

Theyareusedto

›specifythestartandend of aprogram
›attachvaluetovariables
›allocatestoragelocationstoinput/outputdata
›definestartandend ofsegments,procedures,macrosetc..

ASSUME

Used to tell the assembler the name of thelogical segment it should use for aspecified segment. You must tell the assemblerthat what to assume for any segmentyouuseintheprogram.

Example

ASSUME:CODE

Tellstheassemblerthattheinstructionsfortheprogramareinsegmentnamed CODE.

DB–DefinedByte

Usedtodeclareabytetypevariableortosetasideoneormorelocationsof typebyteinmemory.

Example

PRICESDB49H,98H,29H:

Declarearrayof3bytesnamedPRICESand initialize3bytesasshown.

DD–DefineDoubleWord

Usedtodeclareavariable

oftypedoublewordortoreserveamemorylocationwhichcanbeaccessedas doubleword.

DQ–DefineQuadword

Usedtotelltheassemblertodeclarethevariableas4wordsofstoragein memory.

DT–DefineTenBytes

Usedtotelltheassemblertodeclarethevariablewhichis10bytes inlengthor reserve10bytesofstorageinmemory.

DW–DefineWord

Usedtotelltheassemblertodefineavariabletypeaswordor reserveword inmemory.

DUP:usedtoinitializeseverallocationsandto assignvaluestolocation

END –EndtheProgram

Totelltheassemblertostopfetchingtheinstructionandendtheprogramexecution.

ENDP–itisusedtoendtheprocedure.

ENDS–usedtoendthesegment.

EQU–EQUATE

Usedtogivenametosomevalueorsymbol.

EVEN–AlignOnEvenMemoryAddress

Tellstheassemblertoincrementthelocationcountertothenextevenaddressif itisnotalreadyatanevenaddress.

EXTRN

Usedtotelltheassemblerthatthename orlabelsfollowingthedirectiveare insomeotherassembly module.

GLOBAL–DeclaresSymbolsAsPublicOrExtrn

Usedtomakethesymbolavailabletoother modules.ItcanbeusedinplaceofEXTRNorPUBLICkeyword.

GROUP –Grouprelatedsegment

Usedtotelltheassemblertogroupthelogicalsegmentsnamedafterthedirective into one logical segment. This allows the content of all the segments to beaccessedfromthesamegroup.

INCLUDE–includesourcecodefromfile

Usedtotelltheassemblertoinsertablockofsourcecodefromthenamedfileintothec urrentsourcemodule.This shortensthesourcecode.

LABEL

Used to give the name to the current value in the location counter. The LABEL directivemust be followedby a termwhich specifies the type you want associated with thatname.

LENGTH

Usedtodeterminethenumberofitems insomedatasuchasstringorarray.

NAME

Used to give a specific name to a module when the programs consisting of severalmodules.

OFFSET

Itisanoperatorwhichtellstheassemblertodeterminetheoffsetordisplacementofnameddataite morprocedurefromthestartof thesegmentwhichcontainsit.

ORG–Originate

Tellstheassemblertosetthelocationcountervalue.
Example,ORG7000Hsetsthelocationcountervaluetopointto7000H locationinmemory.

$ is often used to symbolically represent the value of the location counter. It isused with ORG to tell the assembler to change the location according to the currentvalueinthelocation counter. E.g. ORG$+100.

UNIT-III

I/OInterface

8255PPI

VariousModesofOperationandInterfacingto8086

D/AandA/DConverter

Steppermotor

InterfacingofDMAcontroller8257

Interfacingwithadvanceddevices

MemoryInterfacingto8086

InterruptStructureof8086

InterruptVectorTable,InterruptService Routine

architectureof8259.

CommunicationInterface

SerialCommunicationStandards

SerialDataTransferSchemes

8251USARTArchitectureandInterfacing.

UNIT-3

I/OInterface

Introduction:

Any application of a microprocessor based system requires the transfer of databetween external circuitry to the microprocessor and microprocessor to the externalcircuitry. User can give information to the microprocessor based system using keyboardand user can see the result or output information from the microprocessor based systemwiththehelpofdisplaydevice.Thetransferofdatabetweenkeyboardandmicroprocessor,andmicroprocessoranddisplaydeviceiscalledinput/outputdatatransferorI/Odatatransfer.Thisdatatransfer isdonewith thehelpofI/Oports.

Inputport:

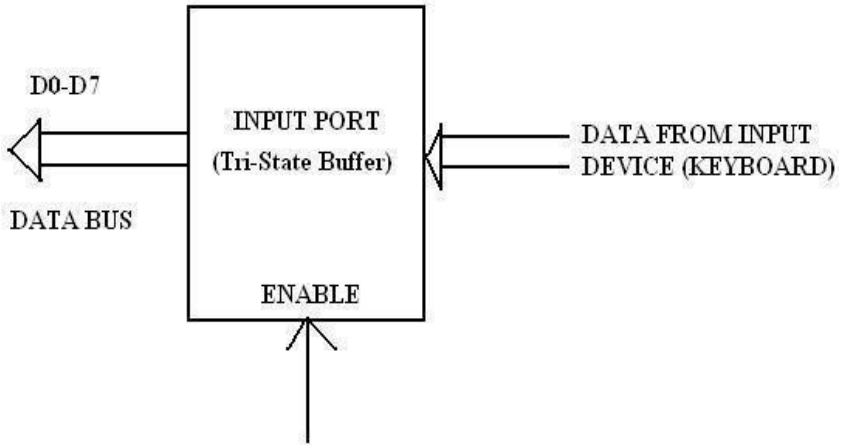


FIG.1 INPUT PORT

It is used to read data from the input device such as keyboard. The simplest formof input port is a buffer. The input device is connected to the microprocessor throughbuffer, as shown in the fig.1. This buffer is a tri-state buffer and its output is availableonly when enable signal is active. When microprocessor wants to read data from theinput device (keyboard), the control signals from the microprocessor activates the bufferby asserting enable input of the buffer. Once the buffer is enabled, data from the inputdevice is available on the data bus. Microprocessor reads this data by initiating readcommand.

Outputport:

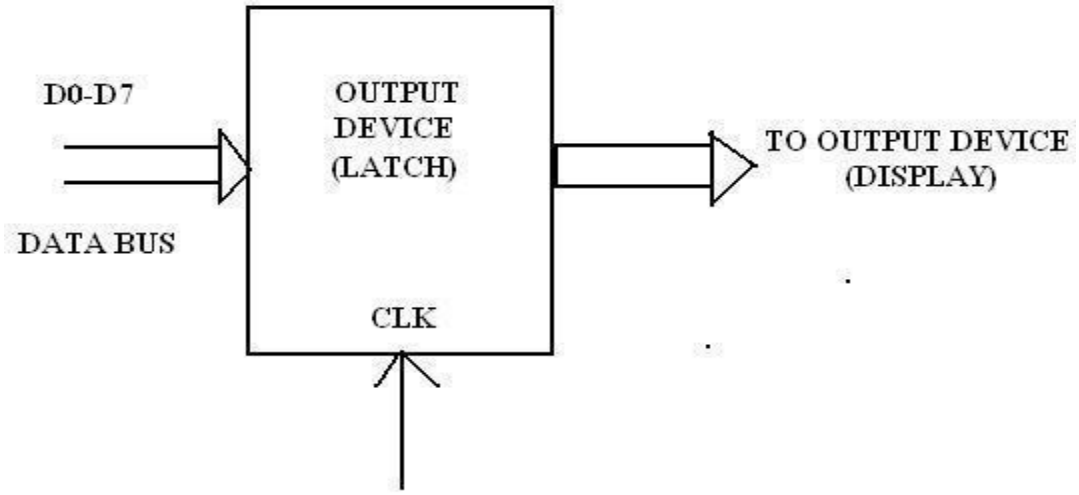


FIG.2 OUTPUT PORT

Itisusedtosenddatatotheoutputdevicesuchasdisplayfromthemicroprocessor.Thesimplestformofoutputportisalatch.Theoutputdeviceisconnectedtothemicroprocessorthroughlatch,asshowninthefig.2.Whenmicroprocessor wants to send data to the output device is puts the data on the data busand activates the clock signal of the latch, latching the data from the data bus at theoutputoflatch.Itisthenavailableattheoutputoflatch for theoutputdevice.

SerialandParallelTransmission:

Intelecommunications,serialtransmissionisthesequentialtransmissionofsignal elements of a group representing a character or other entity of data. Digital serialtransmissions are bits sent over a single wire, frequency or optical path sequentially.Becauseitrequireslesssignalprocessingandlesschanceforerrorthanparalleltransmission, the transfer rate of each individual path may be faster. This can be usedoverlongerdistances as acheckdigitorparitybitcanbesentalongiteasily.

In telecommunications, parallel transmission is the simultaneous transmission ofthe signal elements of a character or other entity of data. In digitalcommunications,parallel transmission is the simultaneous transmission of related signal elements overtwoormoreseparatepaths.Multipleelectricalwiresareusedwhichcantransmitmultiple bits simultaneously, which allows for higher data transfer rates than can beachieved with serial transmission. This methodis used internally within the computer,for example the internal buses, and sometimes externally for such things as printers, Themajor issue with this is "skewing" because the wires in parallel data transmission haveslightly different properties (not intentionally) so some bits may arrive before others,whichmay corrupt

the message. A parity bit can help to reduce this. However, electrical wire parallel datatransmission is therefore less reliable for long distances because corrupt transmissionsarefarmorelikely.

InterruptdrivenI/O:

In this technique, a CPU automatically executes one of a collection of specialroutineswhenever certain condition exists within a program or a processor system.Example CPU gives response to devices such as keyboard, sensor and other componentswhen they request for service. When the CPU is asked to communicate with devices, itservices the devices. Example each time you type a character on a keyboard, a keyboardservice routine is called. It transfers the character you typed from the keyboard I/O portintotheprocessorandthentoadatabufferinmemory.

The interrupt driven I/O technique allows the CPU to execute its main programand only stop to service I/O device when it is told to do so by the I/O system as shown
infig.3.Thismethodprovidesanexternalasynchronousinputthatwouldinformtheprocessor that it should complete whatever instruction that is currently being executedand fetch a new routine that will service the requesting device. Once this servicing iscompleted,theprocessorwould resumeexactlywhereitleftoff.

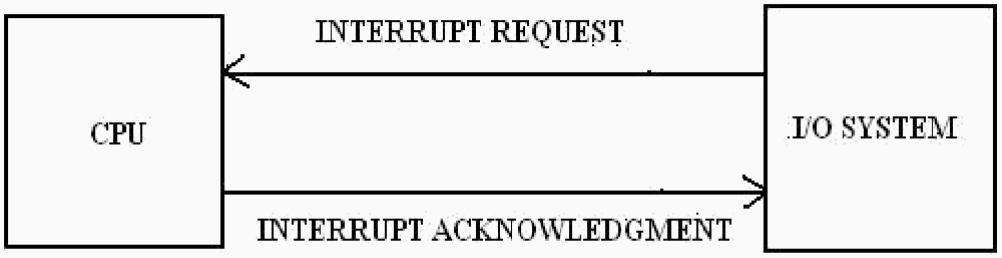


FIG.3 INTERRUPT DRIVEN I/O

An analogy to the interrupt concept is in the classroom, where the professor servesas CPU and the students as I/O ports. The classroom scenario for this interrupt analogywill be such that the professor is busy in writing on the blackboard and delivering hislecture.

The student raises his finger when he wants to ask a question (student requesting forservice).Theprofessorthencompleteshissentenceandacknowledgesstudent‟srequestbys aying“YES”(professoracknowledgestheinterruptrequest).Afteracknowledgement from the professor, student asks the question and professor givesanswer to the question (professor services the interrupt). After that professor continuesitsremaining lectureformwhereitwas left.

PIO8255:

Theparallelinput-outputportchip8255isalsocalledasprogrammable**peripheral input-output port.** The Intel‟s 8255 are designed for use withIntel‟s8-bit,16-bitandhighercapabilitymicroprocessors.Ithas24input/outputlineswhich may be individually programmed in two groups of twelve lines each, orthreegroupsofeightlines.

The two groups of I/O pins are named as Group A and Group B. Each of thesetwogroups contains a subgroup of eight I/O lines called as 8-bit port and anothersubgroup offour lines or a 4-bit port. Thus Group A contains an 8-bit port Aalong with a 4-bit port Cupper.

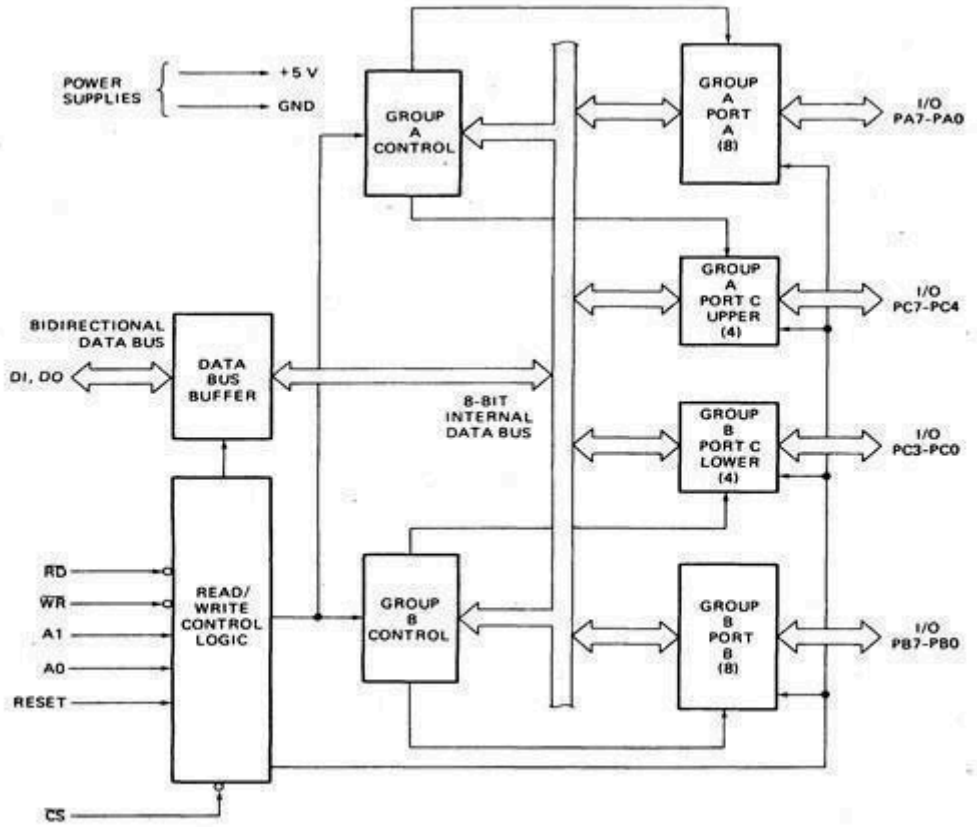
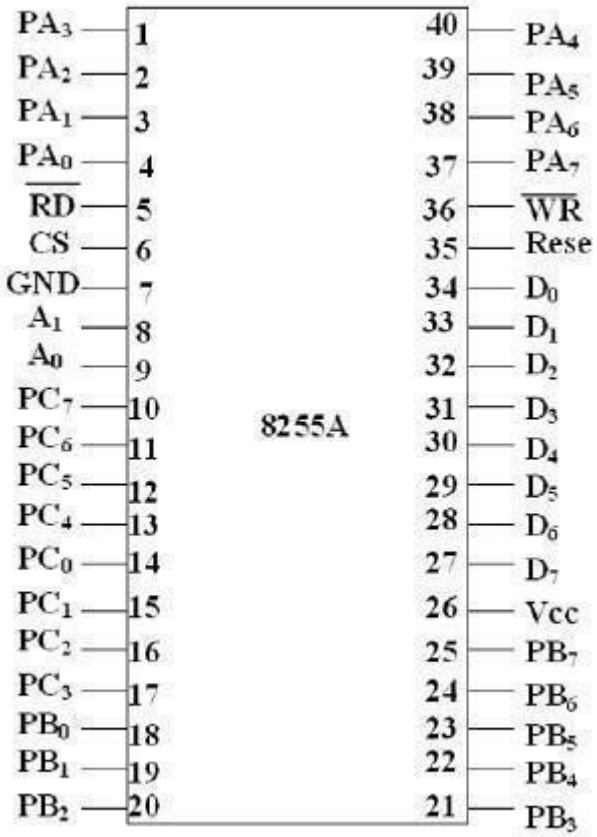


FIGURE Internal block diagram of 8255A programmable parallel port device. *(Intel Corporation)*

The port A lines are identified by symbols PA0-PA7 while the port C lines areidentified as PC4-PC7 similarly. Group B contains an 8-bit port B, containing lines PB0-PB7 and a 4-bit port C with lower bits PC0-PC3. The port C upper and port C lower can beused in combination as an 8-bit port C. Both the port Cs is assigned the same address.Thus one may have either three 8-bit I/O ports or two 8-bit and two 4-bit I/O ports from8255. All of these ports can function independently either as input or as output ports.This can be achieved by programming the bits of an internal register of 8255 called ascontrol word register (CWR). The internal block diagram and the pin configuration of8255 areshowninfigs.

The 8-bit data bus buffer is controlled by the read/write control logic. The read/writecontrol logic manages all of the internal and external transfer of both data and controlwords. RD, WR, A1, A0 and RESET are the inputs, provided by the microprocessor toREAD/WRITE controllogicof8255.The8-bit,3-state bidirectionalbufferisused tointerface the 8255 internal data bus with theexternal system data bus. This bufferreceives or transmits data upon the execution of input or output instructions by themicroprocessor. The control words or status information is also transferred through thebuffer.

PinDiagramof8255A



8255A Pin Configuration

Thepinconfigurationof8255isshowninfig.

- The port A lines are identified by symbols PA0-PA7 while the port C lines areIdentifiedasPC4-PC7.Similarly,GroupBcontainsan8-bitportB,containinglinesPB0-PB7anda4-bitportCwithlowerbitsPC0-PC3.TheportCupperandportClowercanbe usedin combinationas an8-bitportC.
- Both the port C is assigned the same address. Thus one may have either three8-bit I/O ports or two 8-bit and two 4-bit ports from 8255. All of these portscanfunctionindependentlyeitherasinputorasoutputports.Thiscan be

achievedbyprogrammingthebitsofan internalregisterof8255calledascontrolwordregister(CWR).

The 8-bit data bus buffer is controlled by the read/write control logic. The read/writecontrol logic managesall of theinternal and external transfersof both data andcontrolwords.

RD,WR, A1, A0 and RESET are the inputsprovided bythe microprocessor to theREAD/ WRITE control logic of 8255. The 8-bit, 3-state bidirectional buffer is used tointerfacethe8255internaldatabuswiththeexternalsystemdatabus.

Thisbufferreceivesortransmitsdataupontheexecutionofinputoroutputinstructions by the microprocessor. The control words or status information is alsotransferredthroughthebuffer.

Thesignaldescriptionof8255isbrieflypresented asfollows:

PA7-PA0:TheseareeightportAlinesthatactsaseitherlatchedoutputorbufferedinputlines dependinguponthecontrolwordloadedintothecontrolwordregister.

PC7-PC4: UppernibbleofportClines.Theymayactaseitheroutputlatchesorinputbufferslin es.

Thisportalsocan beusedforgenerationofhandshakelinesinmode1 ormode2.

PC3-PC0: ThesearethelowerportClines;otherdetailsarethesameasPC7-PC4lines.

PB0-PB7: TheseareteeightportBlineswhichareusedaslatchedoutputlinesorbufferedin putlinesinthesameway asportA.

RD: Thisistheinputlinedrivenbythemicroprocessorandshouldbelowtoindicatereadoper ationto8255.

WR: Thisisaninputlinedrivenbythemicroprocessor.Alowonthislineindicateswriteoperati on.

CS: This is a chip select line. If this line goes low, it enables the 8255 to respond to RDandWRsignals, otherwiseRDandWRsignalareneglected.

D0-D7: Thesearethedatabuslinesthosecarrydataorcontrolwordto/fromthemicroprocessor.

RESET: Logichighonthislineclearsthecontrolwordregisterof8255.Allportsaresetas inputports by defaultafterreset.

A1-A0: Thesearethe addressinput linesandaredrivenbythe microprocessor.

These lines A1-A0 with RD, WR and CS from the following operations for 8255. Theseaddress lines are used for addressing any one of the four registers, i.e. three portsandacontrolwordregisteras givenintablebelow.

Incaseof8086 systems,ifthe8255istobeinterfacedwith lowerorderdatabus,theA0and A1pinsof8255 areconnectedwithA1 andA2respectively.

$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	A_1	A_0	Input (Read) cycle
0	1	0	0	0	Port A to Data bus
0	1	0	0	1	Port B to Data bus
0	1	0	1	0	Port C to Data bus
0	1	0	1	1	CWR to Data bus

$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	A_1	A_0	Output (Write) cycle
1	0	0	0	0	Data bus to Port A
1	0	0	0	1	Data bus to Port B
1	0	0	1	0	Data bus to Port C
1	0	0	1	1	Data bus to CWR

$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	A_1	A_0	Function
X	X	1	X	X	Data bus tristated
1	1	0	X	X	Data bus tristated

Control Word Register

ModesofOperationof8255

Thesearetwobasicmodesofoperationof8255.I/OmodeandBitSet-Resetmode(BSR).

In I/O mode, the 8255 ports work as programmable I/O ports, while in BSR mode onlyportC(PC0-PC7)canbeusedtoset orresetits individualportbits.

Under the I/O mode of operation, further thereare threemodes of operation of 8255,soastosupportdifferenttypes of applications,mode0,mode1andmode2.

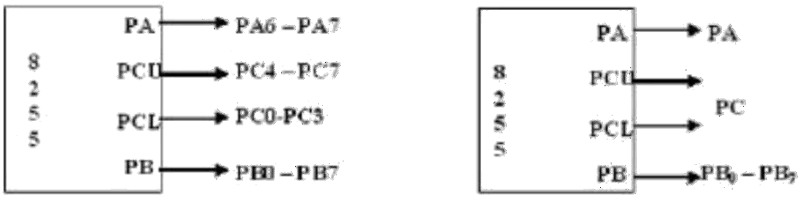
BSR Mode: In this mode any of the 8-bits of port C can be set or reset depending on D0of the control word. The bit to be set or reset is selected by bit select flags D3, D2 and D1oftheCWRas givenintable.

I/OModes:

a) **Mode 0 (Basic I/O mode):** This mode is also called as basic input/output Mode. Thismode provides simple input and output capabilities using each of the threeports. Datacan be simply read from and written to the input and output portsrespectively, afterappropriateinitialization.

D_3	D_2	D_1	Selected bits of port C
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

BSR Mode : CWR Format



Mode 0

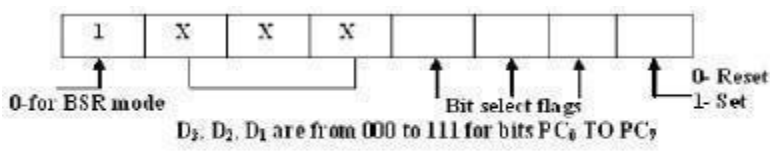
Thesalientfeaturesofthismodeareaslistedbelow:

1. Two8-bitports(portAandportB)andtwo4-bitports(portCupperandlower)areavailable.Thetwo 4-bitportscanbecombinedusedasathird8-bitport.
2. Anyportcanbe usedasaninputoroutputport.
3. Outputportsarelatched.Inputportsarenotlatched.
4. Amaximumoffourportsareavailablesothatoverall16I/Oconfigurationsarepossible.

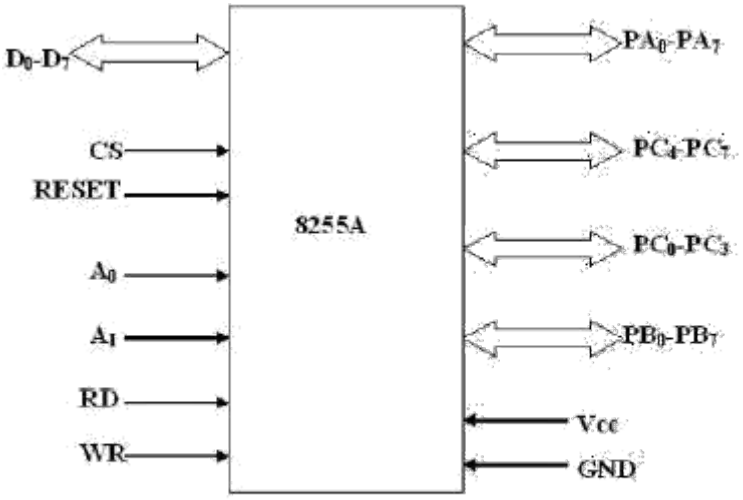
All these modes can be selected by programming a register internal to 8255known asCWR.

The control word register has two formats. The first format is valid for I/O modesofoperation, i.e. modes 0, mode 1 and mode 2 while the second format is validfor bitset/reset(BSR)modeofoperation.

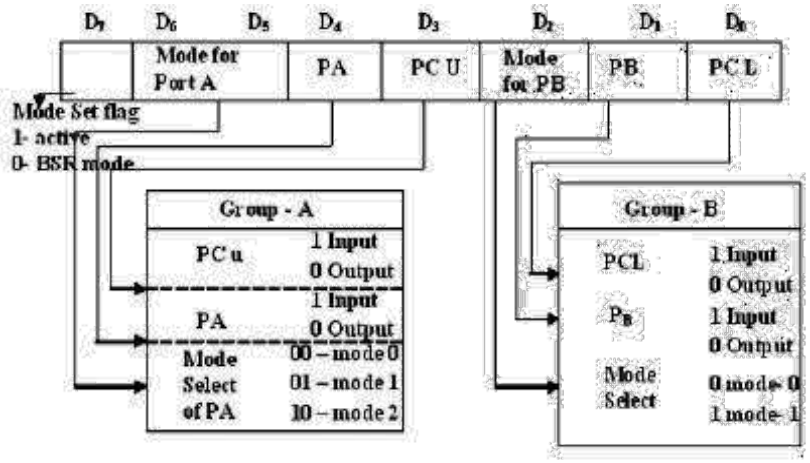
Theseformatsareshowninfollowingfig.



I/O Mode Control Word Register Format and BSR Mode Control Word Register Format



Signals of 8255



Control Word Format of 8255

b) **Mode 1: (Strobed input/output mode)** In this mode the handshaking control theinputandoutput actionofthe specifiedport.Port Clines PC0-PC2, provide strobeorhandshake lines for port B. This group which includes port B and PC0-PC2 is calledasgroup B for Strobed data input/output. Port C lines PC3-PC5 provides strobe lines forportA.This group including port A and PC3-PC5 from group A. Thus port C is utilizedforgeneratinghandshakesignals.

Thesalientfeaturesofmode1arelistedasfollows:

1. Twogroups –groupAandgroupBareavailableforstrobeddatatransfer.
2. Eachgroupcontainsone8-bitdataI/Oportandone4-bitcontrol/dataport.
3. The 8-bit data port can be either used asinput and output port. The inputsandoutputsbotharelatched.
4. Out of 8-bit portC, PC0-PC2 areused to generate control signals for port BandPC3-PC5 are used to generate control signals for port A. the lines PC6, PC7maybeusedas independentdatalines.

Thecontrolsignalsforboththegroupsininputandoutputmodesareexplainedasfollows:

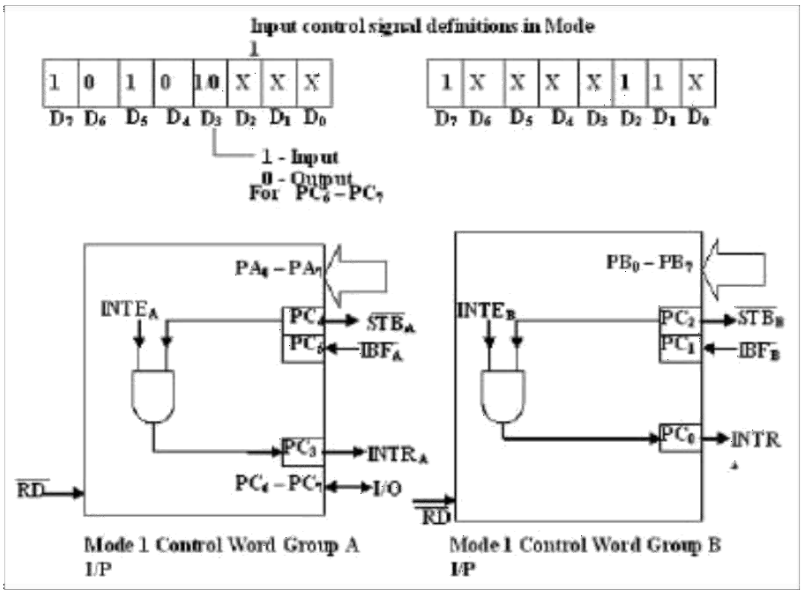
Inputcontrolsignaldefinitions(mode1):

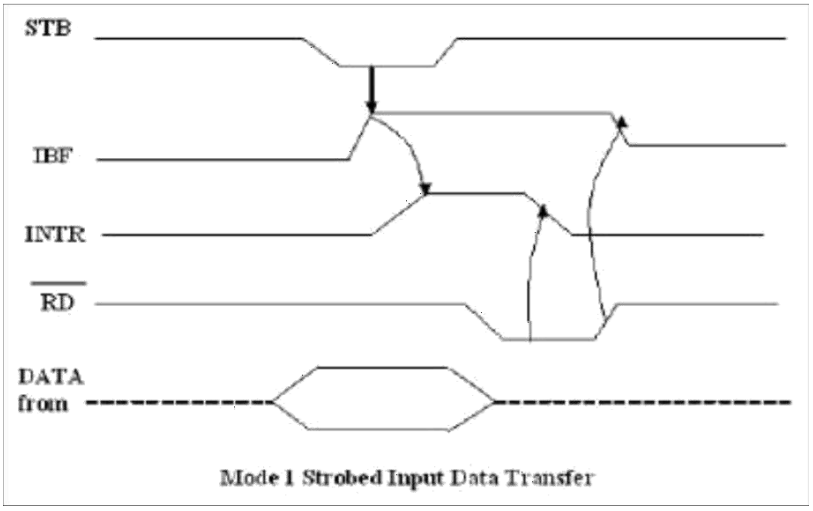
- **STB** (Strobeinput) – If this lines falls to logic low level, the data available at 8-bitinputportis loadedinto inputlatches.
- **IBF** (Input buffer full) – If this signal rises to logic 1, it indicates that datahasbeen loaded into latches, i.e. itworks as an acknowledgement. IBF is setbyalowonSTBand is resetby therising edgeofRDinput.
- **INTR**(Interruptrequest)–ThisactivehighoutputsignalcanbeusedtointerrupttheCPUwheneveraninputdevicerequeststheservice.INTRisset by ahigh STBpin and a high at IBF pin. INTE is an internal flag that can becontrolledby the bit set/reset mode of either PC4 (INTEA) or PC2 (INTEB) asshowninfig.
- INTR is reset by a falling edge of RD input. Thus an external input device canberequest the service of the processor by putting the data on the bus andsendingthestrobesignal.

Outputcontrolsignaldefinitions(mode1):

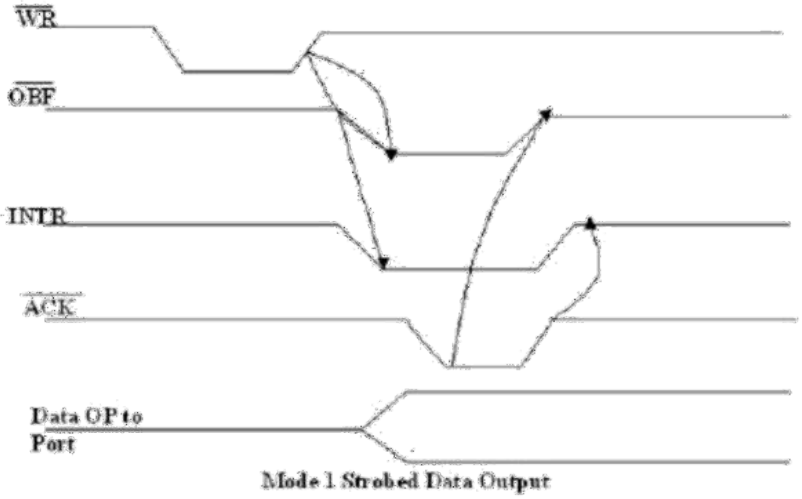
- **OBF**(Outputbufferfull)–Thisstatussignal,wheneverfallstolow,indicatesthat CPU has written data to the specified output port. The OBF flip-flop will beset by a rising edge of WR signal and reset by a low going edge attheACKinput.
- ACK (Acknowledgeinput) – ACK signal acts as an acknowledgement to begivenby an output device. ACK signal, whenever low, informs the CPU that thedatatransferred by the CPU to the output device through the port is receivedbytheoutputdevice.
- **INTR**(Interruptrequest)–Thusanoutputsignalthatcanbeusedtointerruptthe CPU when an output device acknowledges the data receivedfromtheCPU.INTRis setwhenACK,OBFandINTE are1. Itisresetby a

fallingedgeonWRinput.The INTEAandINTEBflagsarecontrolledbythebitset-resetmodeofPC6andPC2respectively.



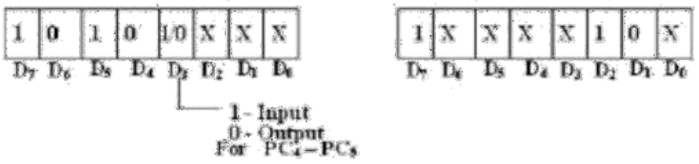


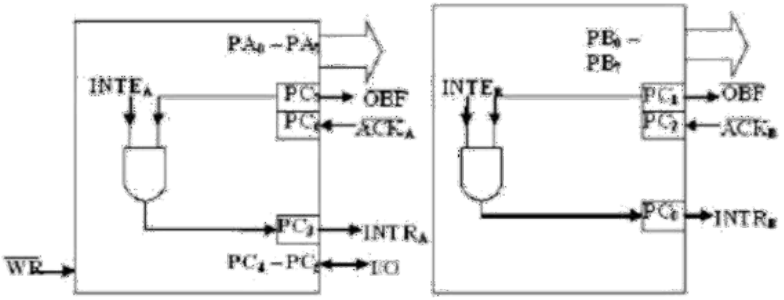
Mode 1 Strobed Input Data Transfer



Mode 1 Strobed Data Output

Output control signal definitions Mode 1





Mode 1 Control Word Group A Mode 1 Control Word Group B

c) **Mode 2 (Strobed bidirectional I/O):**This mode of operation of 8255 is alsocalled asstrobed bidirectional I/O. This mode of operation provides 8255 with additional features forcommunicatingwithaperipheraldeviceonan8-bitdatabus.Handshakingsignalsareprovided to maintain proper data flow andsynchronization between the data transmitterandreceiver.Theinterruptgenerationand other functionsaresimilar tomode1.

Inthismode,8255isabidirectional8-bitportwithhandshakesignals.TheRdandWRsignalsdecidewhether the8255 isgoingto operateasaninputportoroutputport.

TheSalientfeaturesofMode 2of8255are listedasfollows:

1. Thesingle8-bitportingroupAisavailable.
2. The8-bitportisbidirectionalandadditionallya5-bitcontrolportisavailable.
3. ThreeI/Olines areavailableatportC.(PC2–PC0)
4. Inputsandoutputsarebothlatched.

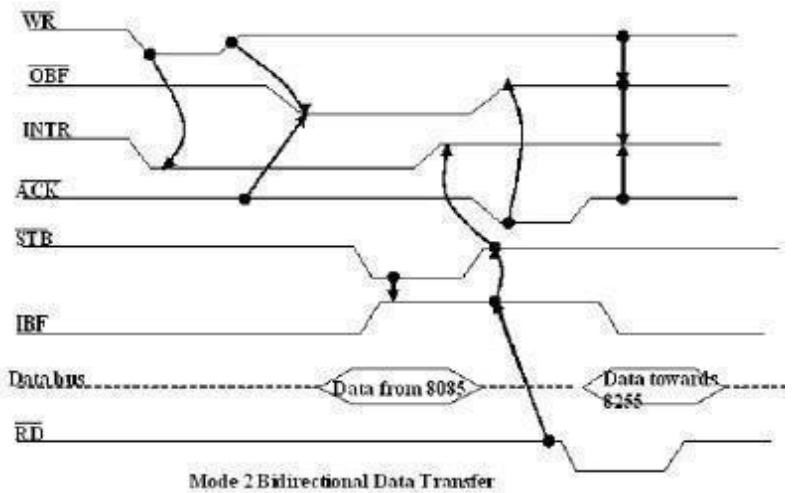
5. The 5-bit control port C (PC3-PC7) is used for generating / acceptinghandshakesignalsforthe8-bitdatatransferonportA.

Controlsignaldefinitionsinmode2:

- **INTR** – (Interrupt request) As in mode 1, this control signal is active high andisusedtointerruptthemicroprocessortoaskfortransferofthenextdatabyteto/fro mit.Thissignalisusedforinput(read)aswellasoutput(write)operations.
- **ControlSignalsforOutputoperations**:
- OBF (Outputbuffer full)– This signal, when falls tolow level, indicatesthattheCPUhas writtendatatoportA.
- ACK(Acknowledge)Thiscontrolinput,whenfallstologiclowlevel,Acknowledges that the previous data byte is received by the destination andnextbyte may be sent by the processor. This signal enables the internal tristatebufferstosendthenextdatabyteonportA.
- **INTE1** (A flag associated with OBF) This can be controlled by bit set/resetmodewithPC6.

Controlsignalsforinputoperations:

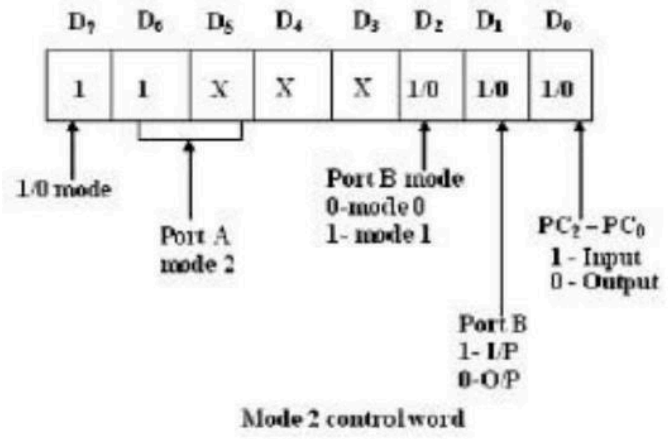
- STB(Strobeinput)alowonthislineisusedtostrobeinthedataintotheinputLatches of8255.
- **IBF** (Input buffer full) when the data is loaded into input buffer, this signal risestologic „1". This can be used as an acknowledge that the data has been receivedbythereceiver.
- ThewaveformsinfigshowtheoperationinMode2foroutputaswellasinputport.
- Note:WRmustoccur beforeACKandSTBmustbeactivatedbeforeRD.



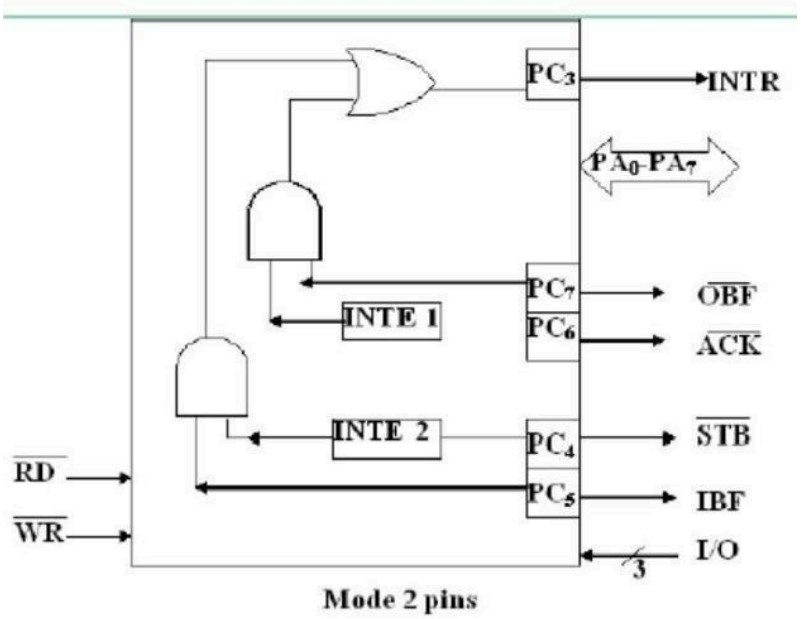
Mode 2 Bidirectional Data Transfer

- Thefollowingfigshowsaschematicdiagramcontainingan8-bitbidirectionalport, 5-bit control port and the relation of INTR with the controlpins. Port B caneither be set to Mode 0 or 1 with port A(Group A) is in Mode2.
- Mode2isnotavailable
- forportB.Thefollowingfigshowsthecontrolword.TheINTRgoeshighonly ifIBF,
- INTE2,STBandRDgohighorOBF,

  INTE1, ACK and WR go high. The port C can be read to know the status oftheperipheraldevice,intermsofthecontrolsignals,usingthenormalI/Oinstructions.



Mode 2 controlword



Mode 2 pins

InterfacingAnalogtoDigitalDataConverters:

Inmostofthecases,thePIO8255isusedforinterfacingtheanalogtodigitalconverterswithmicroprocessor.

We have already studied 8255 interfacing with 8086 as an I/O port, in previous section.This section we will only emphasize the interfacing techniques of analog to digitalconverterswith8255.

The analog to digital converters is treated as aninputdevice by the microprocessorthatsendsaninitializingsignaltotheADCtostarttheanalogytodigitaldataconversationprocess.Thestartofconversationsignalisapulseofaspecific duration.

Theprocessofanalogtodigitalconversionisaslow

Process and the microprocessor have to wait for the digitaldata till the conversion isover. After the conversion isover, the ADC sends end of conversion EOC signal toinformthemicroprocessor that the conversion is over andthe result is ready at the outputbuffer of the ADC. Thesetasks of issuing an SOC pulse to ADC, reading EOC signalfromthe ADC and reading thedigital output of the ADCare carried out by the CPU using8255 I/Oports.

The time taken by the ADC from the active edge of SOCpulse till the active edge of EOCsignalis calledas theconversion delay oftheADC.

It may range anywhere from a few microseconds in caseof fast ADC to even a fewhundredmillisecondsincaseofslowADCs.

The available ADC in the market use different conversiontechniques for conversion ofanalogsignaltodigitals.Successiveapproximationtechniquesanddualslopeintegration techniques are the most popular techniquesused in the integratedADCchip.

GeneralalgorithmforADCinterfacingcontainsthefollowingsteps:

Ensurethestability ofanaloginput,appliedtotheADC.

IssuestartofconversionpulsetoADC

Readend ofconversion signaltomarktheendofconversion processes.

ReaddigitaldataoutputoftheADCasequivalentdigitaloutput.

Analog input voltage must be constant at the input of theADC right from the start ofconversion till the end of theconversion to get correct results. This may be ensured byasample and hold circuit which samples the analog signaland holds it constant forspecifictimeduration.Themicroprocessormayissueaholdsignaltothesampleandholdcircuit.

If the applied input changes before the completeconversion process is over, the digitalequivalentoftheanaloginputcalculatedby theADCmaynotbecorrect.

ADC0808/0809:

- Theanalogtodigitalconverterchips0808and0809are8-bitCMOS,successiveapproximationconverters.Thistechniqueisoneofthefasttechniques for analog to digital conversion. The conversion delay is 100µs at aclockfrequencyof640KHz,whichisquitelowascomparedtootherconverters.Theseconvertersdonotneedanyexternalzeroorfullscaleadjustments astheyarealreadytakencareofbyinternalcircuits.
- These converters internally have a 3:8 analog multiplexer so that at a timeeight different analog conversion by using address lines - ADD A, ADD B,ADD C, as shown. Using these address inputs, multichannel data acquisitionsystem can be designed using a single ADC. The CPU may drive these linesusingoutput portlinesincase of multichannelapplications.In caseofsingle inputapplications,these maybe hardwired toselecttheproperinput.
- There are unipolar analog to digital converters, i.e. they are able to convertonly positive analog input voltage to their digital equivalent. These chips donotcontainanyinternalsampleandholdcircuit.
- If one needs a sample and hold circuit for the conversion of fast signal intoequivalent digital quantities, it has to be externally connected at each of theanaloginputs.

Fig(1)andFig(2)showtheblockdiagramsandpindiagramsforADC0808/0809.

Table.1

AnalogI/Pselected	**Addresslines**		
	C	**B**	**A**
I/P0	0	0	0
I/P1	0	0	1
I/P2	0	1	0
I/P3	0	1	1
I/P4	1	0	0
I/P5	1	0	1
I/P6	1	1	0
I/P7	1	1	1

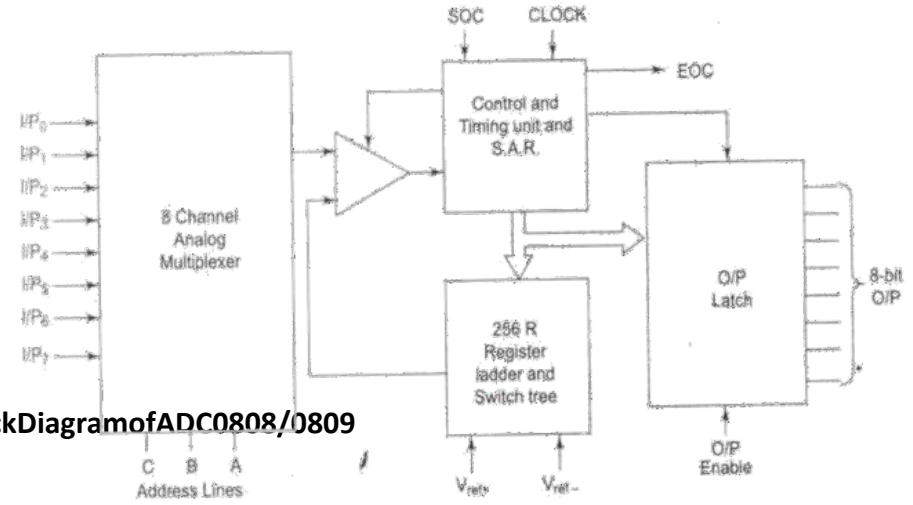
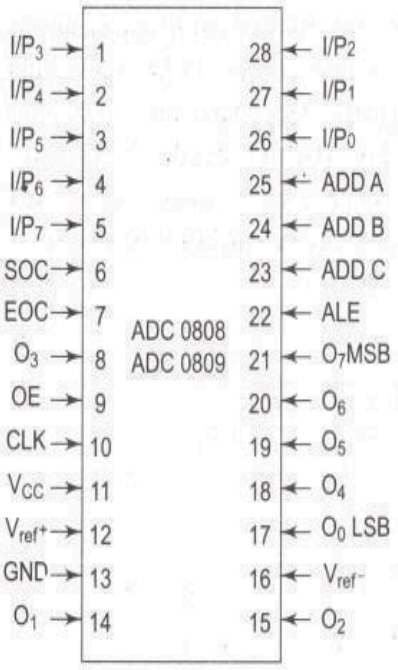


Fig.1BlockDiagramofADC0808/0809



I/P_0 - I/P_7	Analog inputs
ADD A, B, C	Address lines for selecting analog inputs
O_7 - O_0	Digital 8-bit output with O_7 MSB and O_0 LSB
SOC	Start of conversion signal pin
EOC	End of conversion signal pin
OE	Output latch enable pin, if high enable output
CLK	Clock input for ADC
V_{CC}, GND	Supply pins +5V and GND
V_{ref+} and V_{ref-}	Reference voltage positive (+5 Volts maximum) and Reference voltage negative (0V minimum)

Fig.2Pi

SomeElectricalSpecificationsOfTheADC 0808/0809AreGivenInTable.2.

Table.2

Minimum SOC pulse width	100 ns
Minimum ALE pulse width	100 ns
Clock frequency	10 to 1280 kHz
Conversion time	100 ms at 640 kHz
Resolution	8-bit
Error	+/–1 LSB
V_{ref+}	Not more than +5V
V_{ref-}	Not less than GND
+ V_{cc} supply	+ 5 V DC
Logical 1 i/p voltage	minimum V_{cc} –1.5 V
Logical 0 i/p voltage	maximum 1.5 V
Logical 1 o/p voltage	minimum V_{cc}–0.4 V
Logical 0 o/p voltage	maximum 0.45 V

TheTimingDiagramOfDifferentSignalsOf Adc0808IsShownInFig.3

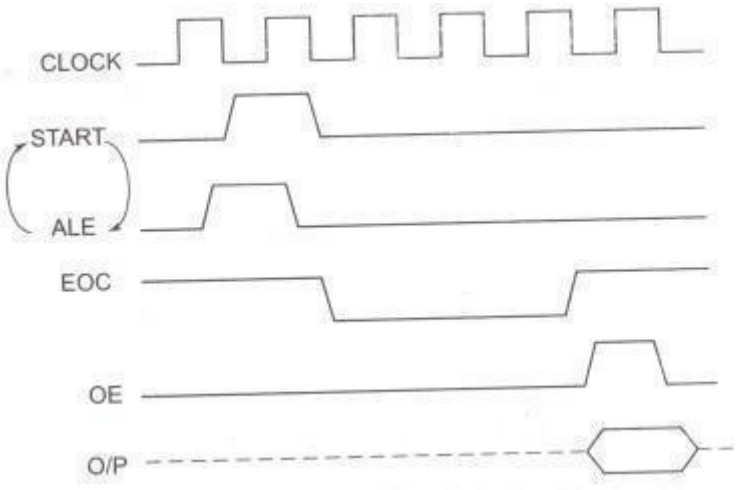
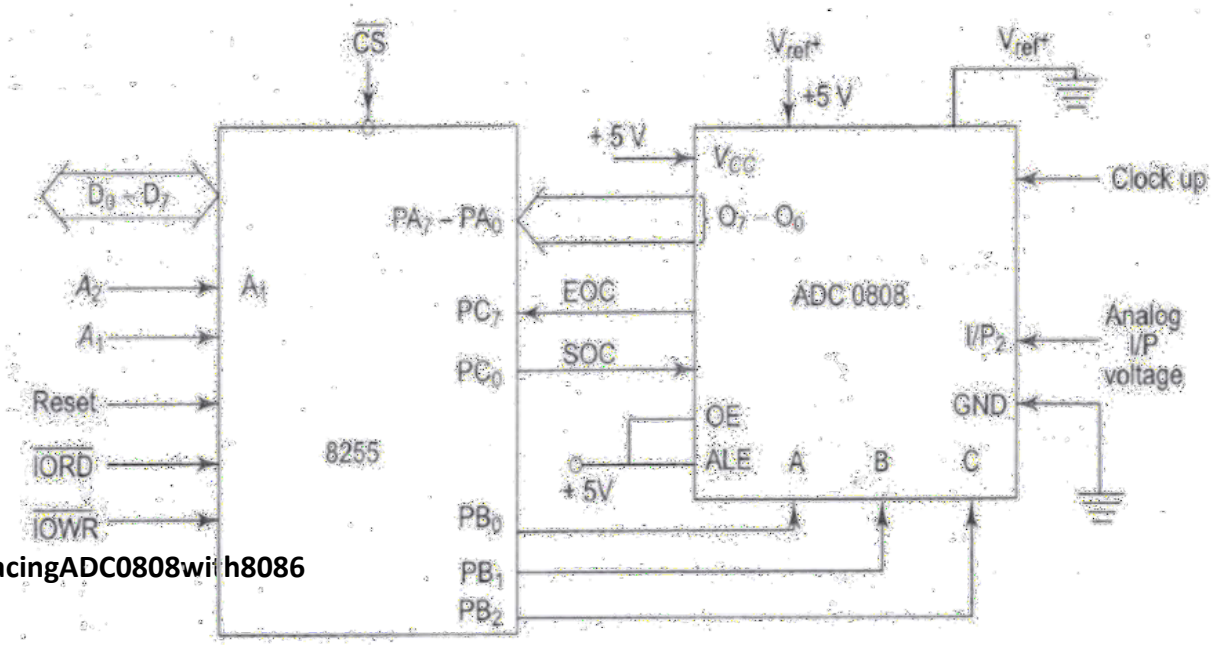


Fig.3TimingDiagramOfADC0808.



InterfacingADC0808wi h8086

InterfacingDigitalToAnalogConverters:

The digital to analog converters convert binary numbersinto their analogequivalent voltages. The DAC find applications in areas like digitally controlled gains,motorspeedcontrols, programmablegainamplifiers,etc.

DAC08008-bitDigitaltoAnalogConverter

The DAC 0800 is a monolithic 8-bit DAC manufacturedbyNationalSemiconductor.

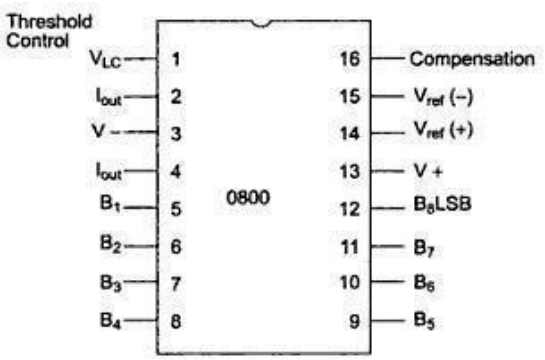
Ithassettlingtimearound100msand canoperate on

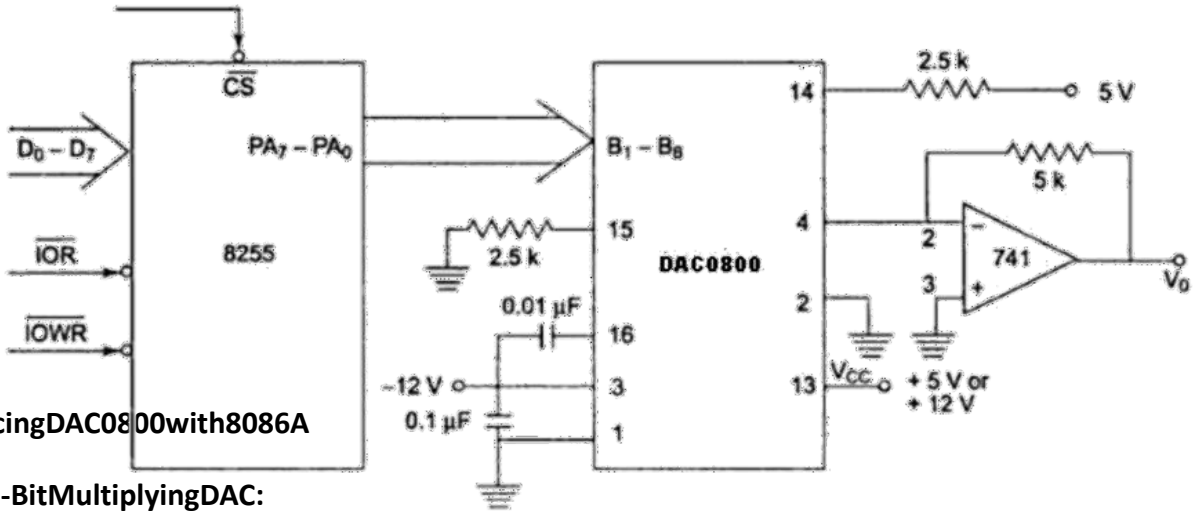
arangeofpowersupplyvoltagesi.e.from4.5Vto+18V.

UsuallythesupplyV+ is5Vor+12V.

TheV-pincanbekeptataminimumof-12V.

PinDia

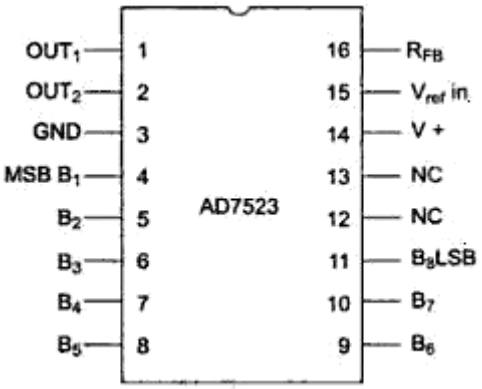




InterfacingDAC0800with8086A

d75238-BitMultiplyingDAC:

Intersil"s AD 7523 is a 16 pin DIP, multiplying digital to analog converter,containing R-2R ladder(R=10KΩ) for digital to analog conversion along withsinglepoledoublethroughNMOSswitchestoconnectthedigitalinputstotheladdder.

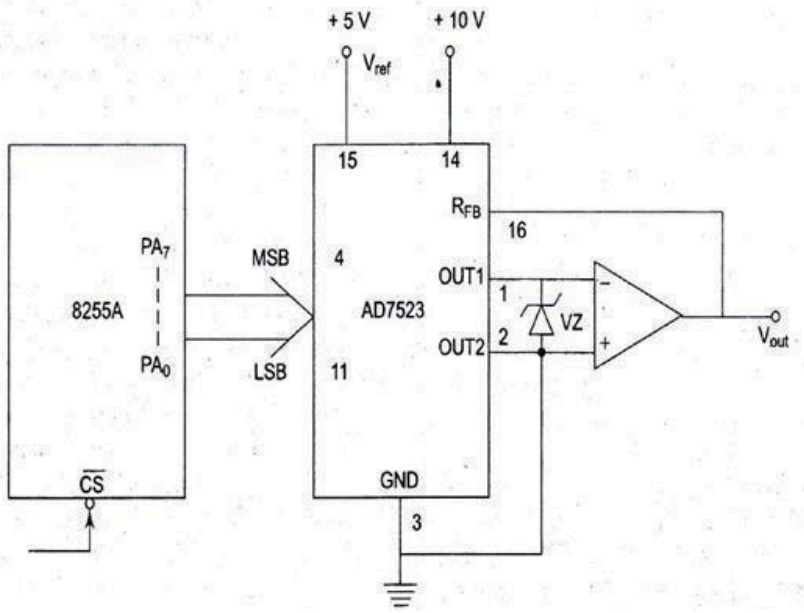


PinDiag

The supply range extends from +5V to +15V , while Vref may be anywherebetween -10V to +10V. The maximum analog output voltage will be +10V,when all the digital inputs are at logic high state. Usually a Zener is connectedbetweenOUT1andOUT2tosavetheDAC fromnegativetransients.

An operational amplifier is used as a current to voltage converter at the outputof AD 7523 to convert the current output of AD7523 to a proportional outputvoltage.

- ItalsooffersadditionaldrivecapabilitytotheDACoutput.Anexternalfeedback resistor acts to control the gain. One may not connect any externalfeedbackresistor,ifnogaincontrol is required.



InterfacingAD7523with8086

StepperMotorInterfacing:

- Asteppermotorisadeviceusedtoobtainanaccuratepositioncontrolofrotating shafts. It employs rotation of its shaft in terms of steps, rather thancontinuous rotation as in case of AC or DC motors. To rotate the shaft of thestepper motor, a sequence of pulses is needed to be applied to the windings ofthesteppermotor, inapropersequence.
- The number of pulses required for one complete rotation of the shaft of thestepper motor is equal to its number of internal teeth on its rotor. The statorteethandtherotorteeth lockwith each otherto fixaposition oftheshaft.
- With a pulse applied to the winding input, the rotor rotates by one teeth positionorananglex. Theanglexmay becalculatedas:

$$X=360^0/\text{no.ofrotorteeth}$$

- After the rotation of the shaft through angel x, the rotor locks itself with the nexttoothinthesequenceontheinternalsurfaceofstator.
- The internal schematic of a typical stepper motor with four windings is shown infig.1.
- The stepper motors have been designed to work with digital circuits. Binary levelpulses of 0-5V are required at its winding inputs to obtain the rotation of shafts.The sequence of the pulses can be decided, depending upon the required

motionoftheshaft.

- Fig.2showsatypicalwindingarrangementofthe steppermotor.

- Fig.3showsconceptualpositioningoftherotorteethonthesurfaceofrotor,forasixteeth rotor.

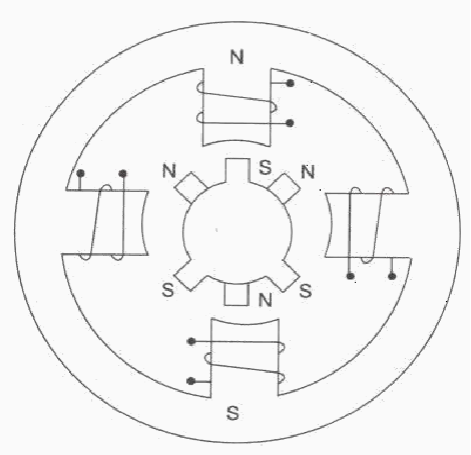


Fig.1Internalschematicofafourwindingsteppermotor

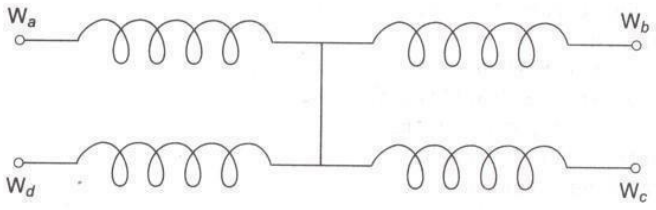


Fig.2Windingarrangementofasteppermotor.

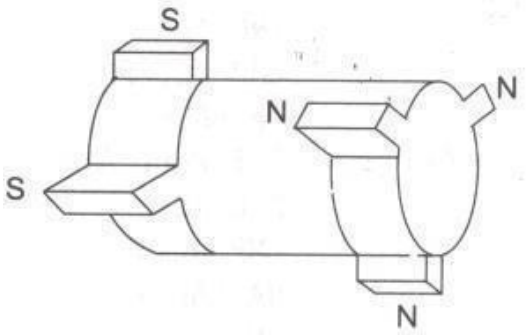


Fig.3Steppermotorrotor

- ThecircuitforinterfacingawindingWnwithanI/Oportisgiveninfig.4.Eachofthewinding sofasteppermotorneedsthiscircuitforitsinterfacingwiththeoutput port. A typical stepper motor may have parameters like torque 3 Kg-cm,operatingvoltage12V,currentrating0.2Aandastepangle1.8^{0}i.e.200steps/revo lution (numberofrotorteeth).

- A simple schematic for rotating the shaft of a stepper motor is called a wavescheme. In this scheme, the windings Wa, Wb, Wc and Wd are applied with therequiredvoltagespulses,inacyclicfashion.Byreversingthesequenceofexcitation,thedirectionofrotationofthesteppermotorshaftmaybereversed.
- Table.1 shows the excitation sequences for clockwise and anticlockwise rotations.Another popular scheme for rotation of a stepper motor shaft applies pulses totwo successive windings at a time but these are shifted only by one position at atime.Thisschemefor rotationof steppermotorshaftisshown intable2.

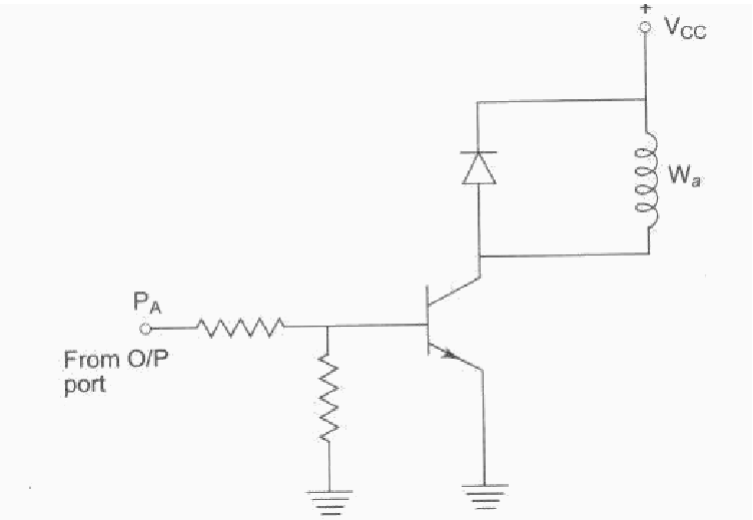


Fig.4interfacingsteppermotorwinding.

Table.1Excitationsequenceofasteppermotorusingwaveswitchingscheme.

Motion	step	A	B	C	D
Clockwise	1	1	0	0	0
	2	0	1	0	0
	3	0	0	1	0
	4	0	0	0	1
	5	1	0	0	0
Anticlock wise	1	1	0	0	0
	2	0	0	0	1
	3	0	0	1	0
	4	0	1	0	0
	5	1	0	0	0

Table.2Analternativeschemeforrotatingsteppermotorshaft

Motion	step	A	B	C	D
Clockwise	1	0	0	1	1
	2	0	1	1	0
	3	1	1	0	0
	4	1	0	0	1
	5	0	0	1	1
Anticlock wise	1	0	0	1	1
	2	1	0	0	1
	3	1	1	0	0
	4	0	1	1	0
	5	0	0	0	0

KeyboardInterfacing

- Inmostkeyboards,thekeyswitchesareconnectedinamatrixofRowsandColumns.
- Gettingmeaningfuldatafrom akeyboardrequiresthreemajortasks:

1. Detectakey pres s
2. Debouncethekeypress.
3. Encode thekeypress(produceastandardcodeforthepressedkey).

Logic„0"isreadbythemicroprocessorwhenthekeyispressed.

KeyDebounce:

Wheneveramechanicalpush-bottomispressedorreleasedonce,themechanicalcomponents ofthekeydonotchangethepositionsmoothly;ratheritgeneratesatransient response. These may be interpreted as the multiple pressures and respondedaccordingly.

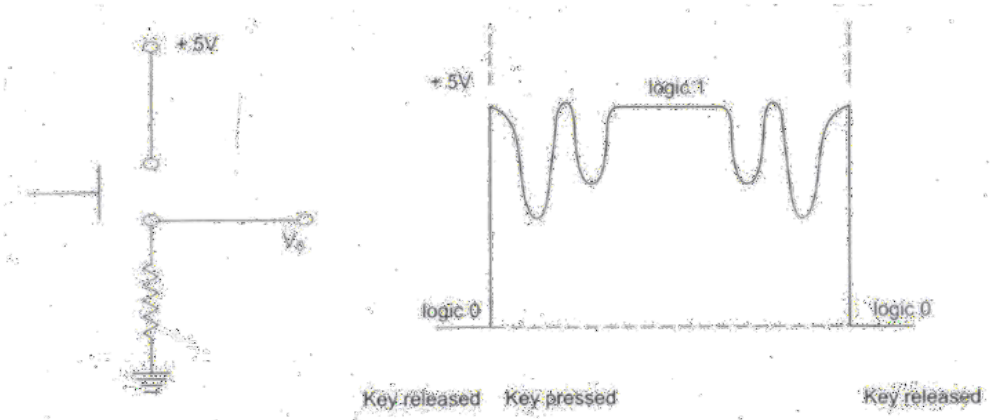


Fig. 5.23 *A Mechanical Key and Its Response*

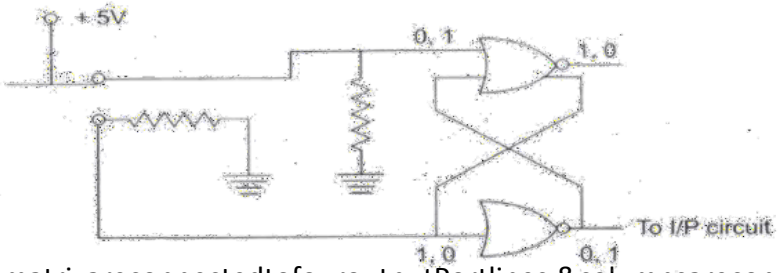


Fig. 5.24 *Hardware Debouncing Circuit*

Therowsofthematrixareconnectedtofouroutput Portlines,&columnsareconnected tofourinput Portlines.

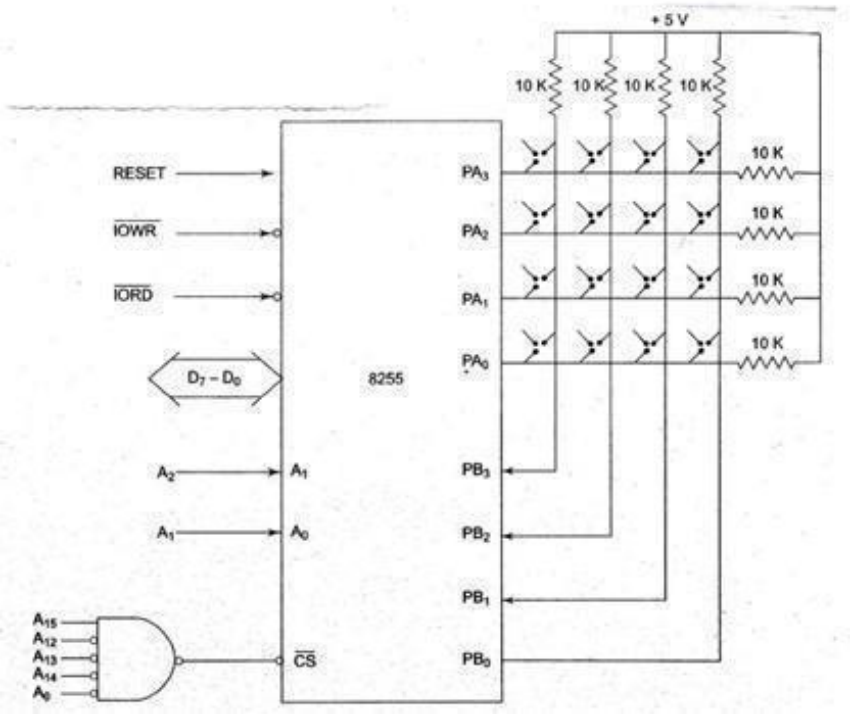
When no keys are pressed, the column lines are held high by the pull-up resistorsconnectedto+5v.

- Pressingakeyconnectsarow& acolumn.
- Todetectifanykeyispressedistooutput0"stoallrows&thencheckcolumns toseeitapressedkeyhasconnectedalow(zero)toacolumn.
- Oncethecolumnsarefoundtobeallhigh,theprogramentersanotherloop,which waits until a low appears on one of the columns i.e indicating a key press.Asimple20/10msecdelayis executedtodebouncetask.
- Afterthedebouncetime,anothercheckismadetoseeifthekeyisstillpressed.If the columns are now all high, then no key is pressed & the initial detection wascausedby anoisepulse.
- Toavoidthisproblem,twoschemesaresuggested:
  1. UseofBistablemultivibratorattheoutputofthekeytodebounceit.
  2. The microprocessor has to wait for the transient period (at least for 10ms), so that the transient response settles down and reaches a steadystate.

Ifanyofthecolumnsarelownow,thentheassumptionismadethatitwasavalidkey press.

- The final task is to determine the row & column of the pressed key &convert thisinformationtoHex-codeforthepressedkey.

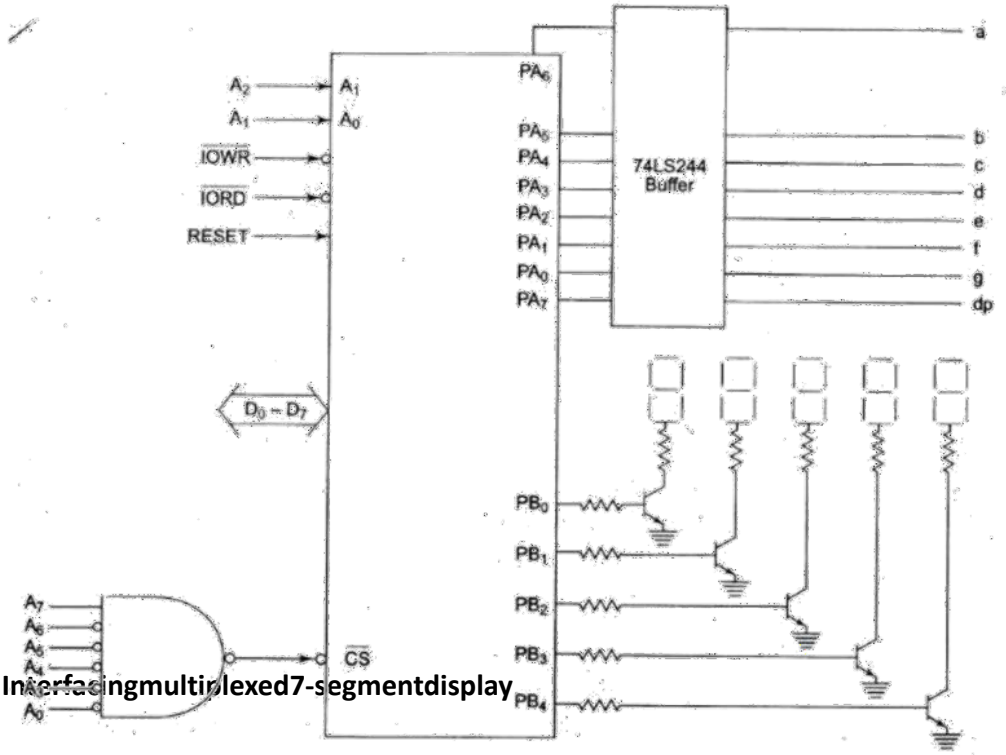
- The4-bitcodefromI/Pport&the4-bitcodefromO/Pport(row&column)areconvertedtoHex-code.



Interfacing4x4keyboard

DisplayInterface

Number to be displayed	PA7dp	PA6a	PA5b	PA4c	PA3d	PA2e	PA1f	PA0g	Code
1	1	1	0	0	1	1	1	1	CF
2	1	0	0	1	0	0	1	0	92
3	1	0	0	0	0	1	1	0	86
4	1	1	0	0	1	1	0	0	CC
5	1	0	1	0	0	1	0	0	A4



Interfacingmultiplexed7-segmentdisplay

InterfacingwithAdvanceddevices

MEMORYANDI/OINTERFACING

(Ref:InterfacingthroughMicroprocessorsbyK.SubbaRao,Hi-tech publishers,P.163-166)

I/OInterface

Anyapplicationofamicroprocessorsystemrequiresthe transferofdatabetweenmicroprocessorandexternalenvironmentandalsowithinthemicroprocessor. This is known as Input/Output. There are three different ways that the data transfercantakeplace. Theyare

(1) ProgramcontrolledI/O
(2) InterruptProgramControlledI/O
(3) HardwarecontrolledI/O

In program controlled I/O data transfer scheme the transfer ofdata is completely under the control of the microprocessor program. In this case an I/O operationtakesplaceonlywhenanI/Otransferinstruction isexecuted.

InaninterruptprogramcontrolledI/Oanexternaldeviceindicates directly to the microprocessor its readiness to transfer data by a signal at an interrupt inputof the microprocessor. When microprocessor receives this signal the control is transferred to ISS(Interruptservicesubroutine)whichperformsthedatatransfer.

HardwarecontrolledI/Oisalsoknownasdirectmemoryaccess DMA. In this case the data transfer takes place directly between an I/O device and memory butnot through microprocessors. Microprocessor only initializes the process of data transfer by indicatingthestartingaddressandthenumberofwordstobetransferred.

Theinstruction.setofanymicroprocessorcontainsinstructions that transfer information to an I/O device and to read information from an I/O device. In8086 we have IN, OUT instructions for this purpose. OUT instruction transfers information to an I/Odevice where as IN instruction is used to read information from an I/O device. Both the instructionsperformthedatatransferusingaccumulatorALorAX.TheI/OaddressisstoredinregisterDX.

The port number is specified along with IN or OUT instruction.The external I/O interface decodes to find the address of the I/O device. The 8 bit fixed port numberappearsonaddressbusA_0-A_7withA_8-A_{15}allzeros.TheaddressconnectionsaboveA15are undefined for an I/O instruction. The 16 bit variable port number appears on address connections A_0 -A_{15}. The above notation indicates that first 256 I/O port addresses 00 to FF are accessed by both thefixedandvariableI/Oinstructions.TheI/Oaddressesfrom0000toFFFFareaccessedbythevariable I/Oaddress.

I/O devicescanbeinterfacedtothemicroprocessorsusingtwomethods.TheyareI/OmappedI/Oandmemory mappedI/O.TheI/OmappedI/Oisalsoknownas isolated I/O or direct I/O. In I/O mapped I/O the IN and OUT instructions transfer data between theaccumulator or memory and I/O device. In memory mapped I/O the instruction that refers memorycanperformthedatatransfer.

I/O mapped I/O is the most commonly used I/O transfer technique. In this method I/O locations areplaced separately from memory. The addresses for isolated I/O devices are separate from memory.Usingthismethodusercanusetheentirememory.Thismethodallowsdatatransferonlybyusing instructions IN, OUT. The pins M/ $\overline{IO}$ and W/R are used to indicate I/O read or an I/O write operations.Thesignalsontheselinesindicatethattheaddressontheaddressbusisfor I/Odevices.

Memory mapped I/O does not use the IN, OUT instruction itusesonlytheinstructionthattransfersdatabetweenmicroprocessorandmemory.Amemorymapped I/O device is treated as memory location. The disadvantage in this systemis the overallmemory is reduced. The advantage of this system is that any memory transfer instruction can be usedfor data transfer and control signals like I/O read and I/O write are not necessary which simplify thehardware.

Memoryinterfacing Memoryisanintegralpartofamicrocomputersystem.There

aretwomaintypesofmemory.

- **(i) Readonlymemory(ROM):**Asthenameindicates thismemoryisavailableonlyforreading purpose.The varioustypes available under thiscategory arePROM,EPROM,EEPROMwhichcontainsystemsoftwareand permanentsystemdata.
- **(ii) Random Access memory (RAM):** This is also known as Read Write Memory. It is a volatilememory. RAM contains temporary data and software programs generally for differentapplications.

Whileexecutingparticulartaskitisnecessarytoaccessmemory to get instruction codes and data stored in memory. Microprocessor initiates the necessarysignals when read or write operation is to be performed. Memory device also requires some signals toperformreadandwriteoperationsusingvariousregisters.Todotheabovejobitisnecessarytohavea device and a circuit, which performs this task is known as interfacing device and as this is involvedwith memory it-is known as memory interfacing device. The basic concepts of memory interfacinginvolve three different tasks.The microprocessorshould be ableto readfromor write intothespecified register. To do this it must be able to select the required chip, identify the required registeranditmustenabletheappropriatebuffers.

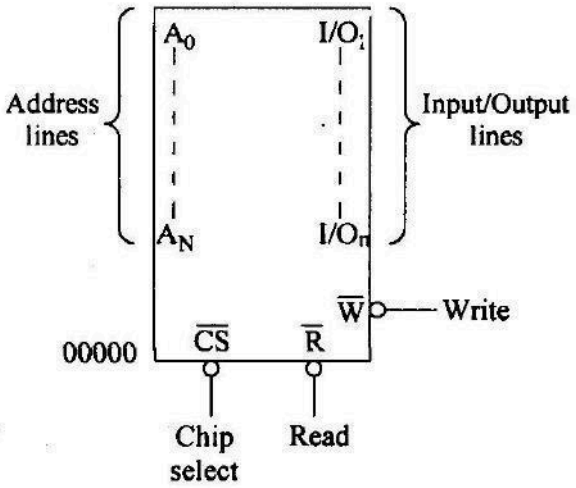


Fig. 3.13 Simple memory device

Anymemorydevice mustcontainaddresslinesandInput,output lines, selection input, control input to perform read or write operation. All memory deviceshaveaddressinputsthatselectmemorylocationwithinthememory device. Theselinesarelabeledas

A_0 A_N.Thenumberofaddresslinesindicatesthetotalmemorycapacityofthememorydevice.A 1K memory requires 10 address lines A_0-A_9. Similarly a 1MB requires 20 lines A_0-A_{19} (in the case of8086). The memory devices may have separate I/O lines or a common set of bidirectional I/O lines.Using these lines data can be transferred in either direction. Whenever output buffer is activated theoperationisreadwheneverinputbuffersareactivatedtheoperationiswrite. Theselinesarelabelled

asI/O,.....I/OnorD0 Dn.Thesizeofamemorylocationisdependentuponthenumberof data bits. If the number of data lines are eight D0 - D7 then 8 bits or 1 byte of data can be stored ineach location. Similarly if numbers of data bits are 16 (D0 - D15) then the memory size is 2 bytes. Forexample2Kx8indicatesthereare2048memorylocationsandeachmemorylocationcanstore8bits ofdata.

Memorydevicesmaycontainoneormoreinputswhichareused toselect thememorydeviceortoenablethememorydevice. ThispinisdenotedbyCS(Chip select)orCE(Chipenable).Whenthispinisatlogic'0'thenonlythememorydeviceperforms aread orawriteoperation.Ifthispinisatlogic‘1’ thememorychipisdisabled.Iftherearemorethanone CSinputthenall thesepinsmustbeactivatedtoperformreadorwriteoperation.

Allmemorydeviceswillhaveoneormorecontrolinputs.When ROM is used we find OE output enable pin which allows data to flow out of the output datapins.Toperformthistaskboth CS and OE mustbeactive.ARAMcontainsoneortwocontrolinputs. TheyareR/WorRD andWR.IfthereisonlyoneinputR/Wthenit performsread operationwhenR/ W pinisatlogic1. Ifitisatlogic0itperformswriteoperation. Notethat thisispossibleonlywhenCSisalsoactive.

MemoryInterfaceusingRAMS,EPROMSandEEPROMS

(Ref: Advanced Microprocessors and Peripherals by A.K. Ray & K.M. Bhurchandi, McGraw-Hill, 2nd Edition.P.158-164)

SemiconductorMemoryInterfacing:

Semiconductor memories are of two types, viz. RAM (Random Access Memory) and ROM (Read OnlyMemory).

StaticRAMInterfacing:

ThesemiconductorRAMsareofbroadlytwotypes-staticRAManddynamicRAM.Thesemiconductor memories are organised as two dimensional arrays of memory locations. For example,4K x 8 or 4K byte memory contains 4096 locations, where each location contains 8-bit data and onlyone of the 4096 locations can be selected at a time. Obviously, for addressing 4K bytes of memory,twelveaddresslinesarerequired.Ingeneral,toaddressamemorylocationout ofNmemorylocations ,wewillrequireatleast*n*bitsofaddress,i.e.*n*addresslineswhere*n*=Log2N.Thusifthemicroprocessor has *n* address lines, then it is able to address at the most N locations of memory,where 2^n = N. However, if out of *N* locations only *P* memory locations are to be interfaced, then theleastsignificant*p*addresslinesoutoftheavailable*n*linescanbedirectlyconnectedfromthemicroprocessor to the memory chip while the remaining *(n-p)* higher order address lines may be usedforaddressdecoding(asinputstothechipselectionlogic).Thememoryaddressdependsuponthe hardwarecircuitusedfordecodingthechipselect(CS).Theoutputofthedecodingcircuitis connected with the CS pin of the memory chip. The general procedure of static memory interfacingwith8086isbrieflydescribedasfollows:

1. Arrange the available memory chips so as to obtain 16-bit data bus width. The upper 8-bitbank is called ‘odd address memory bank’ and the lower 8-bit bank is called ‘even address memorybank’.
2. Connectavailablememoryaddress linesofmemorychipswiththoseofthemicroprocessor andalsoconnectthememoryRDandWRinputsto thecorrespondingprocessorcontrolsignals.Connectthe16-bitdatabusofthememorybankwiththatofth emicroprocessor8086.
3. Theremaining addresslinesofthemicroprocessor,BHEandA0areusedfordecodingthe requiredchipselectsignalsfortheoddandevenmemorybanks.CSofmemoryisderivedfromtheO/Pofth edecodingcircuit.

Asagoodandefficientinterfacingpractice,theaddressmapofthesystemshouldbecontinuous as far as possible, i.e. there should be no windows in the map. A memory location shouldhave a single address corresponding to it, i.e. absolute decoding should be preferred, and minimumhardware should be used for decoding. In a number of cases, linear decoding may be used to minimisethe required hardware.Let usnowconsider afew example problems on memory interfacing with8086.

Problem 5.1

Interface two 4K × 8 EPROMS and two 4K × 8 RAM chips with 8086. Select suitable maps.

Solution We know that, after reset, the IP and CS are initialised to form address FFFF0H. Hence, this address must lie in the EPROM. The address of RAM may be selected any where in the 1MB address space of 8086, but we will select the RAM address such that the address map of the system is continuous, as shown in Table 5.1.

Table 5.1 *Memory Map for Problem 5.1*

Address	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_{09}	A_{08}	A_{07}	A_{06}	A_{05}	A_{04}	A_{03}	A_{02}	A_{01}	A_{00}
FFFFFH	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
						EPROM							8K × 8							
FE000H	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
FDFFFH	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
						RAM							8K × 8							
FC000H	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Total 8K bytes of EPROM need 13 address lines $A_0 - A_{12}$ (since $2^{13} = 8K$). Address lines $A_{13} - A_{19}$ are used for decoding to generate the chip select. The $\overline{BHE}$ signal goes low when a transfer is at odd address or higher byte of data is to be accessed. Let us assume that the latched address, $\overline{BHE}$ and demultiplexed data lines are readily available for interfacing. Figure 5.1 shows the interfacing diagram for the memory system.

The memory system in this example contains in total four 4K × 8 memory chips.

The two 4K × 8 chips of RAM and ROM are arranged in parallel to obtain 16-bit data bus width. If A_0 is 0, i.e. the address is even and is in RAM, then the lower RAM chip is selected indicating 8-bit transfer at an even address. If A_0 is 1, i.e. the address is odd and is in RAM, the $\overline{BHE}$ goes low, the upper RAM chip is selected, further indicating that the 8-bit transfer is at an odd address. If the selected addresses are in ROM, the respective ROM chips are selected. If at a time A_0 and $\overline{BHE}$ both are 0, both the RAM or ROM chips are selected, i.e. the data transfer is of 16 bits. The selection of chips here takes place as shown in Table 5.2.

Table 5.2 *Memory Chip Selection for Problem 5.1*

Decoder I/P → *Address/$\overline{BHE}$ →*	A_2 A_{13}	A_1 A_0	A_0 $\overline{BHE}$	*Selection/ Comment*
Word transfer on $D_0 - D_{15}$	0	0	0	Even and odd addresses in RAM
Byte transfer on $D_7 - D_0$	**0**	**0**	**1**	**Only even address in RAM**
Byte transfer on $D_8 - D_{15}$	0	1	0	Only odd address in RAM
Word transfer on $D_0 - D_{15}$	**1**	**0**	**0**	**Even and odd addresses in ROM**
Byte transfer on $D_0 - D_7$	1	0	1	Only even address in ROM
Byte transfer on $D_8 - D_{15}$	**1**	**1**	**0**	**Only odd address in ROM**

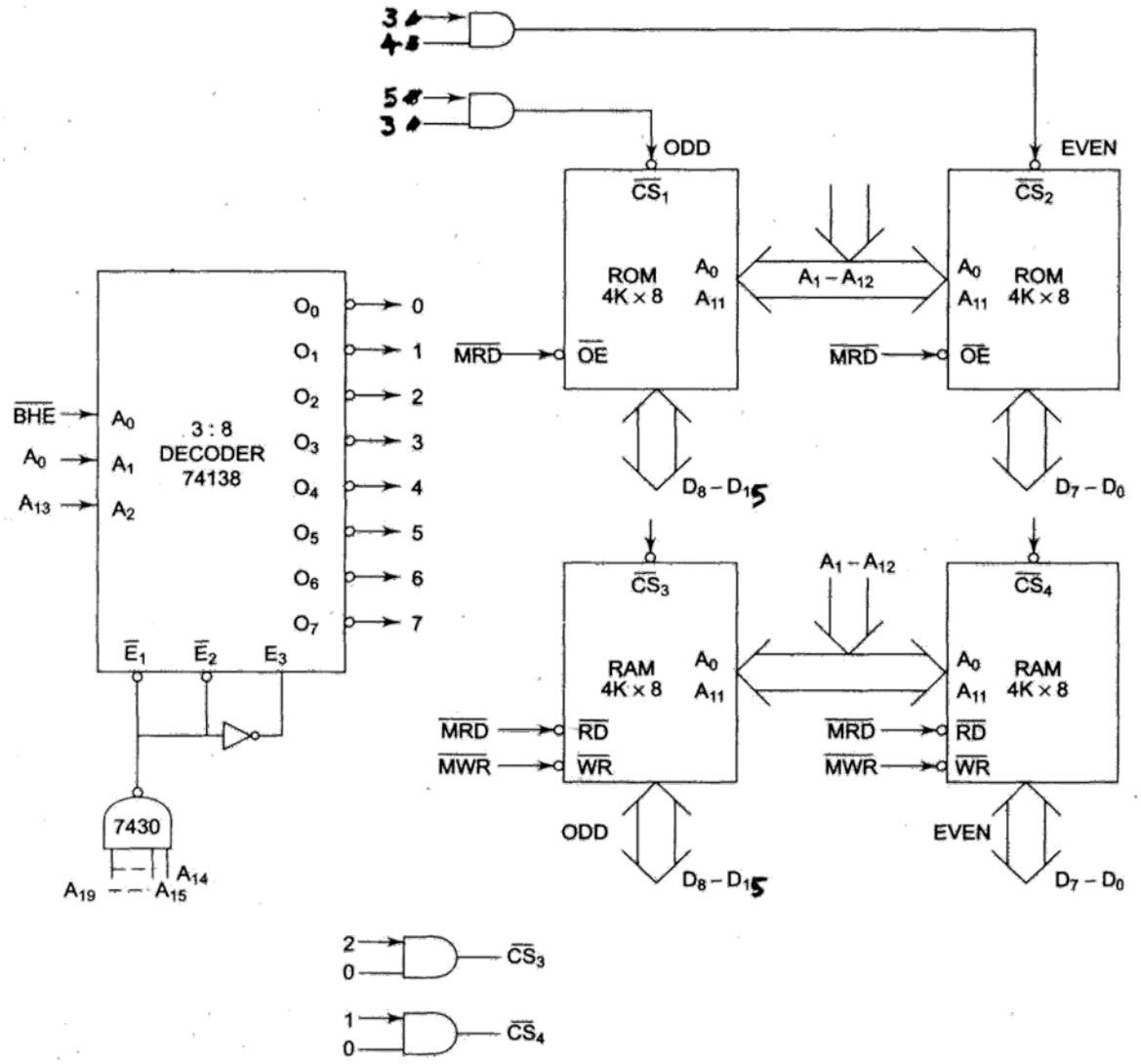


Fig. 5.1 *Interfacing Problem 5.1*

Problem 5.2

Design an interface between 8086 CPU and two chips of 16K × 8 EPROM and two chips of 32K × 8 RAM. Select the starting address of EPROM suitably. The RAM address must start at 00000H.

Solution: The last address in the map of 8086 is FFFFFH. After resetting, the processor starts from FFFF0H. Hence this address must lie in the address range of EPROM. Figure 5.2 shows the interfacing diagram, and Table 5.3 shows complete map of the system.

Table 5.3 *Address Map for Problem 5.2*

Addresses	A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_{09}	A_{08}	A_{07}	A_{06}	A_{05}	A_{04}	A_{03}	A_{02}	A_{01}	A_{00}
FFFFFH	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
						32KB EPROM														
F8000H	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0FFFFH	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
						64KB RAM														
00000H	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

It is better not to use a decoder to implement the above map because it is not continuous, i.e. there is some unused address space between the last RAM address (0FFFFH) and the first EPROM address F8000H). Hence the logic is implemented using logic gates, as shown in Fig. 5.2.

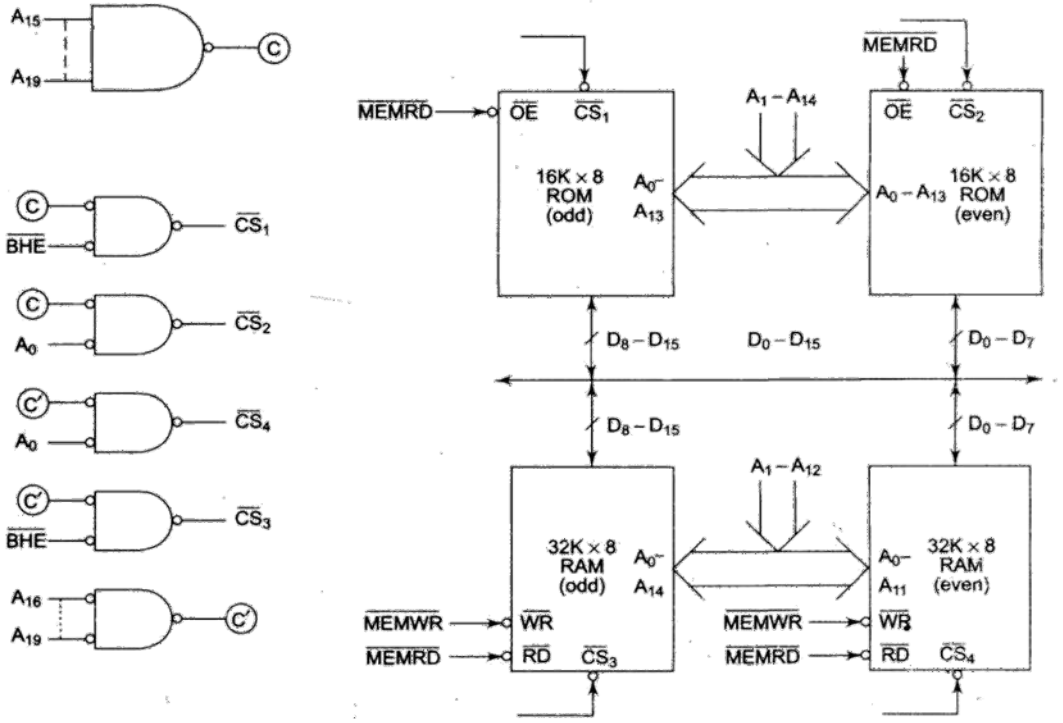


Fig. 5.2 *Interfacing Problem 5.2*

Problem 5.3

It is required to interface two chips of 32K × 8 ROM and four chips of 32K × 8 RAM with 8086, according to the following map.

```
ROM 1 and 2 F0000H - FFFFFH, RAM 1 and 2 D0000H - DFFFFH
RAM 3 and 4 E0000H - EFFFFH
```

Show the implementation of this memory system.

Solution Let us write the memory map of the system as shown in Table 5.6.

The implementation of the above map is shown in Fig. 5.3 using the same technique as in Problem 5.1 and Problem 5.2. All the address, data and control signals are assumed to be readily available.

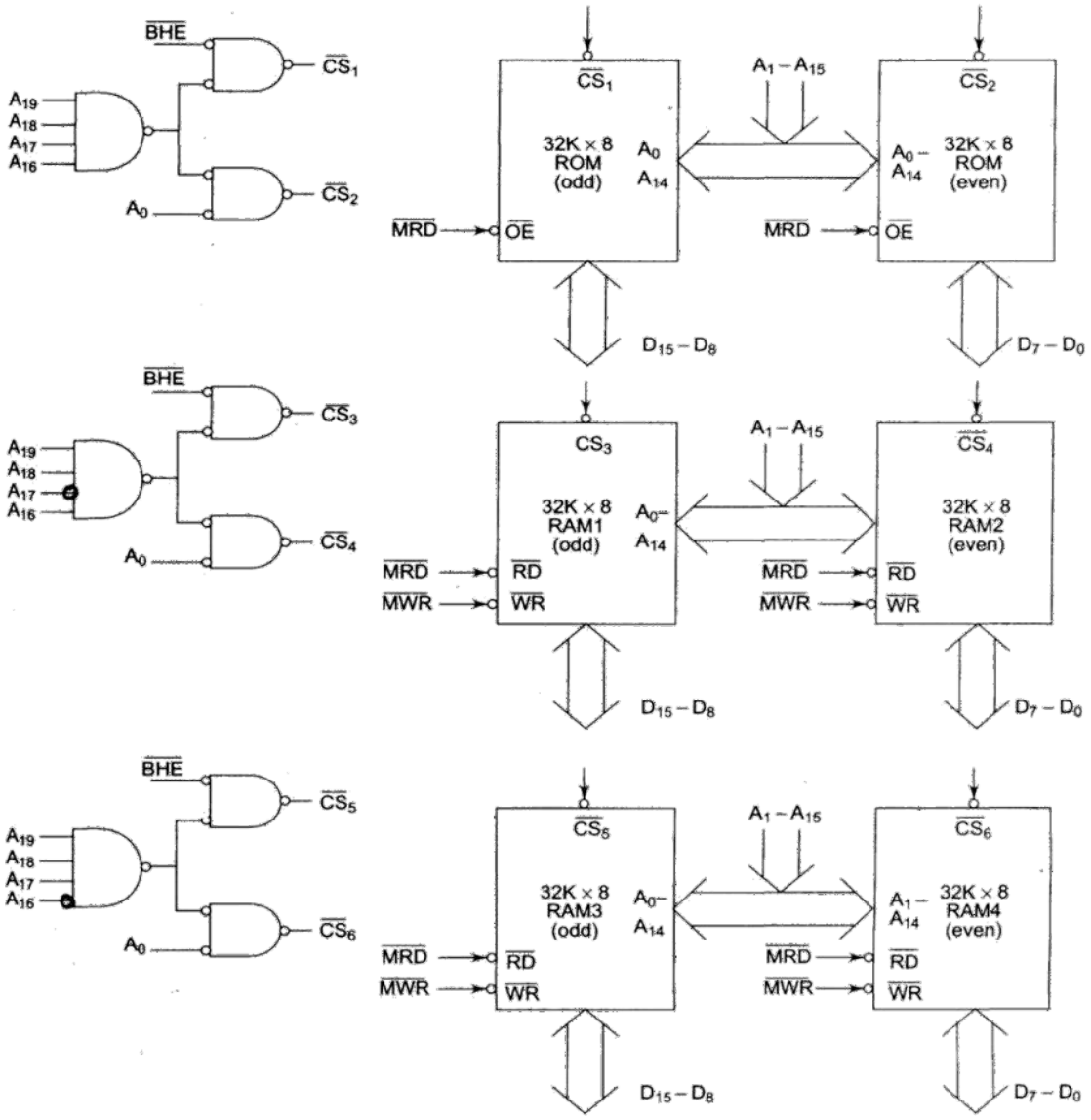


Fig. 5.3 *Interfacing Problem 5.3*

SERIALCOMMUNICATIONSTANDARDS

(Ref:InterfacingthroughMicroprocessorsbyK.SubbaRao,Hi-tech publishers,P.250-260)

Most of devices are parallel in nature. These devices transfer data simultaneously on datalines. But parallel data transfer process is very complicated and expensive. Hence in some situationsthe serial I/O mode is used where one bit is transferred over a single line at a time. In this type oftransmission parallel word is converted into a stream of serial bits which is known as parallel to serialconversion. The rate of transmission in serial mode is BAUD, i.e., bits per second. The serial datatransmission involves starting, end of transmission, error verification bits along with the data. AnyserialI/Oinvolvesthefollowingconcepts.

(a) Interfacing requirements (b) Alphanumeric codes(c) Transmission format (d)Errorchecksin datacommunication(e)Datacommunicationoverlines(f)StandardsinserialI/O

The microprocessor has to identify the port address to perform read or write operation. SerialI/Ousesonlyonedataline,chipselect, read,writecontrol signals.

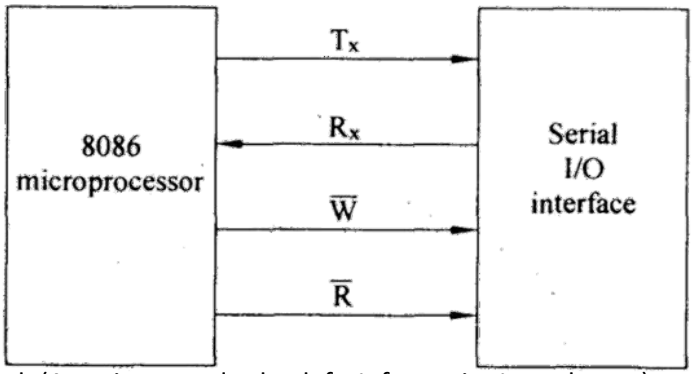
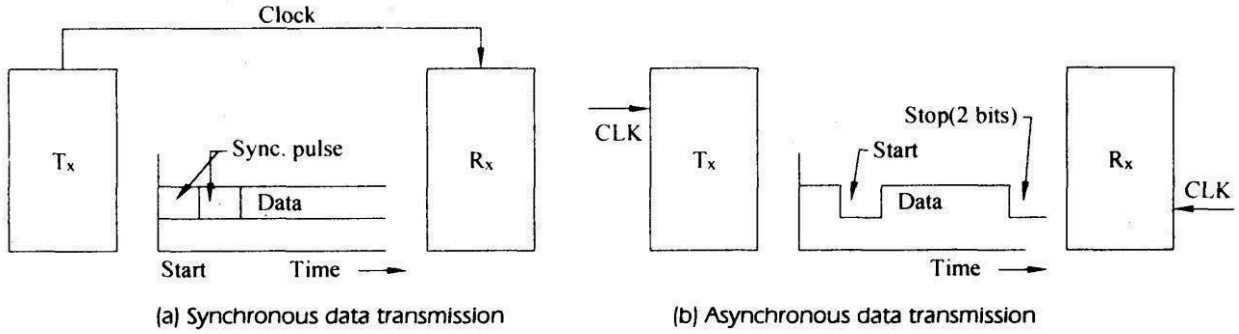


Fig. 5.1 Block diagram of serial I/O interfacing

DatatransfertakesplaceusingASCIIcode(AmericanstandardcodeforInformationInterchange) which is 7 bit code with 128 combinations. The data can be transmitted by taking variousparameters into consideration such as synchronization or asynchronization, direction of data flowspeed, errors, medium of data transmission etc. In synchronous transmission both transmitter andreceiveroperate,insynchronoustoeachother.

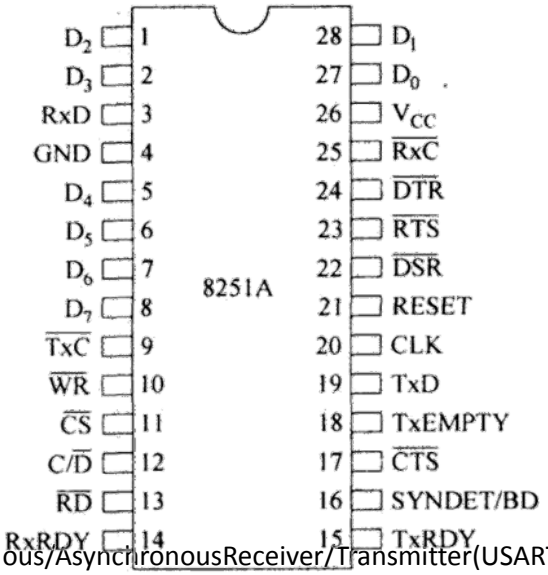
Synchronization used for high speed operations. In asynchronous data transmission data istransmitted between Start and Stop bits with logic 1 as mark logic 0 as space. In asynchronous we getaround11bitsfordatatransmissiononestart,8bitsofdata,2stopbits.Asynchronousdatatransmissioni susedforlessthan20Kbits/second transmission.



(a) Synchronous data transmission (b) Asynchronous data transmission

Fig. 5.2 Data communication

S.No.	Synchronous	Asynchronous
1.	Same clock pulse is applied to both Tx & Rx simultaneously	Different clock pulses are applied to Tx & Rx sepeartely
2.	Only hardware is required to implement this.	Both hardware and software are requied for this.
3.	Group or a set of characters can be transmitted at a time.	Only one character is transmitted at a time.
4.	Synchronous pulses are required	Synchronous pulses are not required but uses start and stop bits.
5.	Uses for high speed Tx.	Used for low speed Tx.



Pin diagram of 8521

The8251AisUniversalSynchronous/AsynchronousReceiver/Transmitter(USART)designedfor the data communication with Intel's family of microprocessor such as 8085, 8086 and 8088. Likeother I/O devices in a microcomputer system, its functional configuration is programmed by thesystem's software for maximum flexibility. The USART accepts data characters from the CPU in theparallelformatandconvertsthemintocontinuousserialdatastreamfortransmission.Simultaneously, it can receive serial data streams arid convert them into parallel data characters fortheCPU.TheCPUcanreadthecompletestatusofUSARTatanytime,theseincludesdatatransmissionerr ors,control signalsetc.

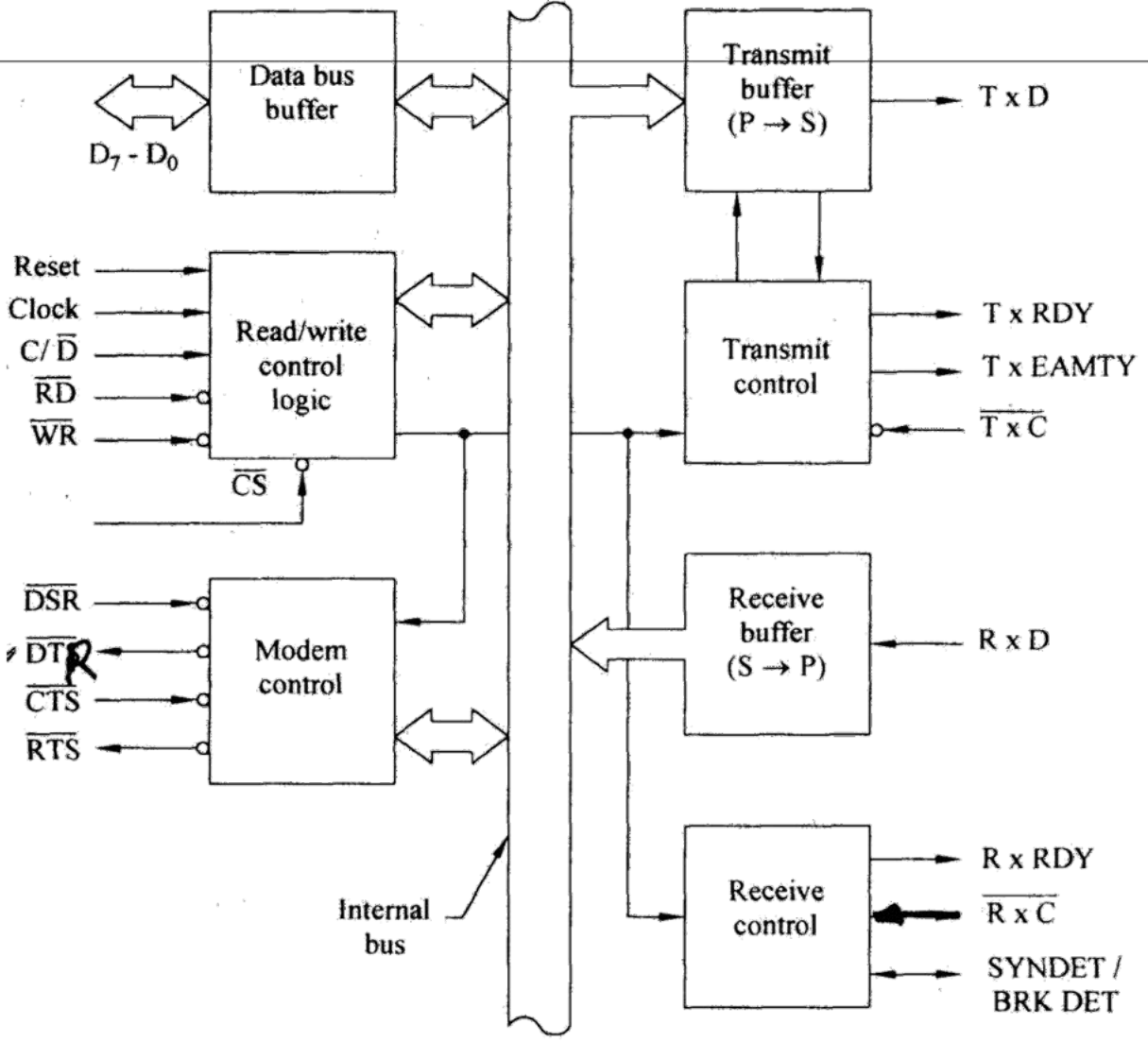


Fig.5.7 Blockdiagramof8251

Fig. 5.7 shows the block diagram of 8251 A. The block diagram shows all the elements of aprogrammable chip; it includes the interfacing signals, the control register and the status register. Thefunctionsofvariousblocksaredescribedbelow:

(A) Data bus buffer: This 3-state, bidirectional buffer is used to interface the 8251A to the system databus. Data is transmitted or received by the buffer upon execution of input and output instruction ofthe CPU Command words and status information are also transferred through the data bus buffer. Thecommand,statusanddatainanddataoutareseparate8-bitregisterstoprovidedoublebuffering.

The functional block accepts inputs form the control bus and generates control signals foroverall device operation. It contains the control word register and command word register that storethevariouscontrol formatsforthedevicefunctionaldefinition.

(B) Read/Write logic and Registers:

This section includes R/W control logic, six input signals, control logic, and three buffer registers; data register, control register and status register. The input signals to control logic are as follows:

RESET: A high on this input forces the 8251A into an idle mode. The device will remain at idle until a new set of control words is written into the 8251A to program its functional definition.

A command reset operation also puts the device into the idle state.

CLK (Clock): The CLK input is used to generate internal device timing and is normally connected to the phase 2 (TTL) output of the Clock Generator. No external inputs or outputs are referenced to CLK but the frequency of CLK must be greater than 30 times the Receiver or Transmitter data bit rates.

$\overline{WR}$ (Write): A "low" on this input informs the 8251A that the CPU is writing data or control words to the 8251A.

$\overline{RD}$ (Read): A "low" on this input informs the 8251A that the CPU is reading data or status information from the 8251A.,

C/$\overline{D}$	RD	WR	CS	
0	0	1	0	8251 data-data bus
0	1	0	0	Data bus-8251A data
1	0	1	0	Status-data bus
1	1	0	0	Data bus-control
×	1	1	0	Data bus-3 state
×	×	×	1	Data bus 3 state

C/$\overline{D}$ (Control/Data): This input, in conjuction with the $\overline{WR}$ and $\overline{RD}$ inputs informs the 8251A that the word on the Data bus is either a data character, control word or status information.

1 = CONTROL/STATUS; 0 = DATA

$\overline{CS}$ (Chip Select) : A "low" on this input selects the 8251A. No reading or writing will occur unless the device is selected. When $\overline{CS}$ is high, the Data Bus is in the float state and $\overline{RD}$ and $\overline{WR}$ have no effect on the chip.

(C) Modem Control:

The 8251A has a set of control inputs and outputs that can be used to simplify the interface to almost any modem. The modem control signals are general purpose in nature and can be used for functions other than modem control, if necessary.

$\overline{DSR}$ (Data Set Ready) : The $\overline{DSR}$ input signal is a general-purpose, 1-bit inverting input port. Its condition can be tested by the CPU using a Status Read operation. The DSR input is normally used to test modem condition such as Data Set Ready.

$\overline{\text{DTR}}$ (Data Terminal Ready): The $\overline{\text{DTR}}$ output signal is a general-purpose, 1-bit inverting output port. It can be set "low" by programming the appropriate bit in the Command instruction word. The $\overline{\text{DTR}}$ output signal is normally used for modem control such as Data Terminal Ready.

$\overline{\text{RTS}}$ (Request to Send): The $\overline{\text{RTS}}$ output signal is a general-purpose, 1-bit inverting output port. It can be set "low" by programming the appropriate bit in the Command instruction word. The $\overline{\text{RTS}}$ output signal is normally used for modem control such as Request to send.

$\overline{\text{CTS}}$ (Clear to Send): A "low" on this input enables the 8251A to transmit serial data if the Tx Enable bit in the Command byte is set to a "one", if either a T× Enable off or $\overline{\text{CTS}}$ off condition occurs while the T× is in operation, the T× will transmit all the data in the USART, written prior to T× Disable command before shutting down.

(D) Transmitter Buffer:

The transmitter Buffer accepts parallel data from the Data Bus Buffer, converts it to a serial bit stream, inserts the appropriate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the TxD output pin on the falling edge of $\overline{\text{T x C}}$. The transmitter will begin transmission upon being enabled, if $\overline{\text{CTS}}$ = 0. The $\overline{\text{T x C}}$ line will be held in the marking state immediately upon a master Reset or when Tx Enable or $\overline{\text{CTS}}$ is off or the transmitter is empty.

(E) Transmitter Control:

The Transmitter Control manages all activities associated with the transmission of serial data. It accepts and issues signals both externally and internally to accomplish this function.

T x RDY (Transmitter Ready): This output signals the CPU that the transmitter is ready to accept a data character. The T x RDY output pin can be used as an interrupt to the system, since it is masked by T x Enable; or, for Polled operation, the CPU can check T x RDY using a Status Read operation. T x RDY is automatically reset by the leading edge of $\overline{\text{WR}}$ when a data character is loaded from the CPU.

T x E (Transmitter Empty): When the 8251A has no character to send, the T x empty will go high. It resets upon receiving a character from CPU if the transmitter is enabled. Tx empty remains high when the transmitter is disabled. T x Empty can be used to indicate the end of transmission node, so that the CPU knows when to turn around in the half-duplex operation mode.

In the synchronous mode, a high on this output indicates that a character has not been loaded and the SYNC character or characters are about to be are being transmitted as filters. T x Empty does not go low when the Sync characters are being shifted out.

T x C (Transmitter Clock): The Transmitter Clock control the rate at which the character is to be transmitted. In the Synchronous transmission mode, the Baud Rate (1x) is equal to the T x C frequency. In Asynchronous transmission mode, the baud rate is a fraction of the actual T x C frequency. A portion of the mode instruction selects this factor it can be 1,1/16 or 1/64 the T x C.

Forexample
IfBaudrateequals220Baud
TXC equals 220 Hz in the 1x
mode.TXCequals3.52KHzinthe16x
mode.TXCequals14.08KHzinthe
64xmode.

(F) Receiver Buffer:

The Receiver accepts serial data, converts this serial input to parallel format checks for bits or characters that are unique to the communication technique and sends an "assembled" character to the CPU. Serial data is input to R × D pin, and is clocked in on this rising edge of $\overline{R \times C}$.

(G) Receiver Control:

This functional block shown in Fig. manages all receiver - related activities which consists of the following features.

The R × D initialization circuits prevents the 8251A from mistaking and unused input line for an active low data line in the break condition. Before starting to receive serial characters on the R × D line, a valid '1' must first be detected after a chip master reset. Once this has been determined, a search for a valid low bit (start bit) is enabled. This feature is only active in the asynchronous mode, and is only done once for each master reset.

The false start bit detection circuit prevents false starts due to a transient noise spike by first detecting the falling edge and then strobing the nominal center of the start (R × D = low).

Parity error detection sets the corresponding status bit.

The framing error status bit is set if the stop bit is absent at the end of the data byte (asynchronous mode).

R × RDY (Receiver Ready) : This output indicates the the 8251A contains a character that is ready to be input to the CPU. R × RDY can be connected to the interrupt structure of the CPU or, for polled operation, the CPU can check the condition of R × RDY using a status read operation.

R × Enable, when off, holds R × RDY in the reset condition. For asynchronous mode, to set R × RDY, the receiver must be enabled to sense a start bit and a complete character must be assembled and transferred to the data output register.

Failure to read the received character from the R × Data output register prior to the assembly of the next R × data character will set overrun condition error and the previous character will be written over and lost. If the R × data is being read by the CPU when the internal transfer is occurring, overrun error will be set and the old character will be lost.

Thefalling edge of8251A.
TXCshiftstheserial dataout ofthe

$\overline{R \times C}$ (Receiver Clock) : The receiver clock controls the rate at which the character is to be received. In synchronous mode, the baud rate (1×) is equal to the actual frequency of $\overline{R \times C}$. In asynchronous mode, the baud rate is a fraction of the actual $\overline{R \times C}$ frequency. A portion of the mode instruction selects this factor: 1, 1/16 of 1/64 the $\overline{R \times C}$.

For Example:

Baud rate equals 200 baud, if

$\overline{R \times C}$ equals 200 Hz in the 1 × mode.

$\overline{R \times C}$ equals 3200 Hz in the 16 × mode.

$\overline{R \times C}$ equals 128 KHz in the 64 × mode.

SYNDET (SYNC Detect/BRKDET Break Detect): This is used in Synchronous Mode for SYNDET and may be used as either input or output, programmable through the Control Word. It is reset to output mode low upon RESET. When used as an output (internal Sync mode), the SYNDET pin will go "high" to indicate that the 8251A has located the SYNC character in the Receive mode. If the 8251A is programmed to use double Sync characters (bi-sync), then SYNDET will go "high" in the middle of the last bit of the second Sync character SYNDET is automatically reset upon a status Read operation.

When used as an input (external SYNC detect mode), a positive going signal will cause the 8251A to start assembling data characters on the rising edge of the next $\overline{R \times C}$. Once in SYNC, the "high" input signal can be removed. When External SYNC Detect is programmed, internal SYNC Detect is disabled.

BREAK (Async Mode Only): This output will go high whenever the receiver remains low through two consecutive stop bit sequences (including the start bits, data bits, and parity bits). Break Detect may also be read as a Status bit. It is reset only upon a master chip Reset of R × Data returning to a "one" state.

INTERFACINGSTANDARDS

(Ref:InterfacingthroughMicroprocessorsbyK.SubbaRao,Hi-tech publishers,P.266)

Serial I/O is used to interface various devices or for connecting various equipment to thesystem.Common understandingis necessaryamongvarious manufacturerssuchthat astandardnotation is followed for interfacing these components. These standards may be provided by IEEE or byany standard professional organisation. The serial I/O standards must specify clearly voltage levels,speed of data transfer, length of cables etc. In serial I/O data can be transmitted as either current orvoltage 20 mA or 60 mA current loops are used if data is transmitted using current. Current flow takesplace when the system is at logic 1. The current flow is stopped when the system is at logic 0. In thecurrent loop method the signals are relatively noise-free and they are best suited for long distancetransmission.

RS-232 is developed long before which is used for communication between terminals andmodems. Using RS-232C data can be transmitted as voltage. The data terminals equipment and datacommunication equipment are used to communicate using RS-232C cable. RS-232C is not compatiblewithTTLlogicandcannotbeusedforlongdistancetransmission.

RS-232CSerialDataStandard

(Ref:Microprocessorsandinterfacing byDouglasV.Hall,2nd edition,TMH,P.494-495)

OVERVIEW

Modemsweredevelopedsothatterminalscouldusephonelinestocommunicatewithdistant computers. As we stated earlier, modems and other devices used to send serial data are oftenreferred to as *data communication equipment* or DCE. The terminals or computers that are sending orreceiving the data are referred to as *data terminal equipment* or DTE. In response to the need forsignal and handshake standards between DTE and DCE, the Electronic Industries Association (EIA)developed EIA standard RS-232C. This standard describes the function of 25 signal and handshake pinsfor serial-data transfer. It also describesthe voltage levels, impedance levels, rise and fall times,maximumbitrate,andmaximumcapacitanceforthesesignallines.

RS-232C specifies 25 signal pins, and it specifies that the DTE connector should be a male andthe DCE connector should be a female. A specific connector is not given, but the most commonly usedconnectorsare theDB-25P male shown in Figure 14-7a. For systems where many of the25 pins arenotneeded,a9-pinDINconnectorsuchastheDE-9PmaleconnectorshowninFigure14-7bisused.

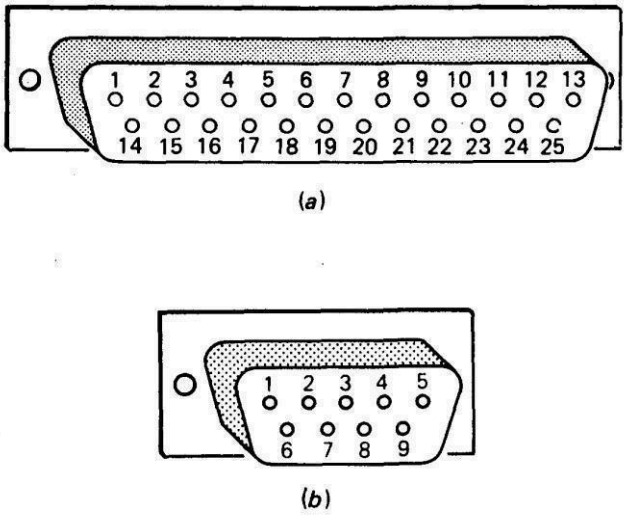


FIGURE 14-7 Connectors often used for RS-232C connections. (*a*) DB-25P 25-pin male. (*b*) DE-9P 9-pin male DIN connector.

The voltage levels for all RS-232C signals are as follows. A logic high, or mark, is a voltagebetween-3Vand-15Vunderload(-25Vnoload).Alogic loworspaceisavoltagebetween+3Vand +15Vunderload(+25Vnoload). Voltagessuchas±12Varecommonlyused.

RS-232CtoTTLINTERFACING

Obviously a USART such as the 8251A is not directly compatible with RS-232C signal levels. ThestandardwaytointerfacebetweenRS-232CandTTLlevels iswithMCI488quadTTL-to-RS-232CdriversandMCI489quadRS-232C-to-TTLreceiversshowninFigure14 -8.

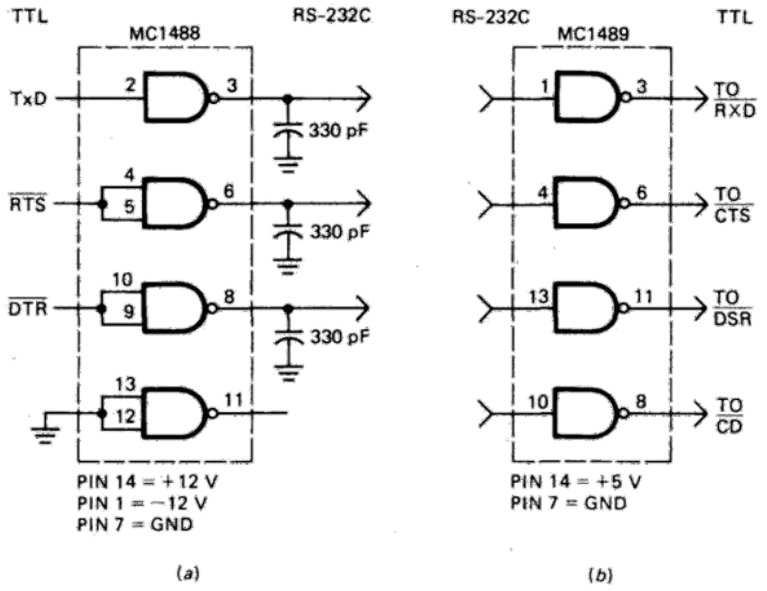


FIGURE 14-8 TTL to RS-232C to TTL signal conversion. (a) MC1488 used to convert TTL to RS-232C. (b) MC1489 used to convert RS-232C to TTL.

TheMCI488srequire+and-supplies,buttheMCI489srequireonly+5V.Notethecapacitortogroun dontheoutputsoftheMCI488driversistoreducecrosstalkbetweenadjacentwires,theriseandfall timesforRS-232C signalsarelimitedto30V/µs.

RS-232CSIGNALDEFINITIONS

PIN NUMBERS FOR 9 PINS	PIN NUMBERS FOR 25 PINS	COMMON NAME	RS-232C NAME	DESCRIPTION	SIGNAL DIRECTION ON DCE
	1		AA	PROTECTIVE GROUND	–
3	2	TXD	BA	TRANSMITTED DATA	IN
2	3	RXD	BB	RECEIVED DATA	OUT
7	4	$\overline{\text{RTS}}$	CA	REQUEST TO SEND	IN
8	5	$\overline{\text{CTS}}$	CB	CLEAR TO SEND	OUT
6	6	$\overline{\text{DSR}}$	CC	DATA SET READY	OUT
5	7	GND	AB	SIGNAL GROUND (COMMON RETURN)	–
1	8	$\overline{\text{CD}}$	CF	RECEIVED LINE SIGNAL DETECTOR	OUT
	9		–	(RESERVED FOR DATA SET TESTING)	–
	10		–	(RESERVED FOR DATA SET TESTING)	–
	11			UNASSIGNED	–
	12		SCF	SECONDARY RECEIVED LINE SIGNAL DETECTOR	OUT
	13		SCB	SECONDARY CLEAR TO SEND	OUT
	14		SBA	SECONDARY TRANSMITTED DATA	IN
	15		DB	TRANSMISSION SIGNAL ELEMENT TIMING (DCE SOURCE)	OUT
	16		SBB	SECONDARY RECEIVED DATA	OUT
	17		DD	RECEIVER SIGNAL ELEMENT TIMING (DCE SOURCE)	OUT
	18			UNASSIGNED	–
	19		SCA	SECONDARY REQUEST TO SEND	IN
4	20	$\overline{\text{DTR}}$	CD	DATA TERMINAL READY	IN
	21		CG	SIGNAL QUALITY DETECTOR	OUT
9	22		CE	RING INDICATOR	OUT
	23		CH/CI	DATA SIGNAL RATE SELECTOR (DTE/DCE SOURCE)	IN/OUT
	24		DA	TRANSMIT SIGNAL ELEMENT TIMING (DTE SOURCE)	IN
	25			UNASSIGNED	–

FIGURE 14-9 RS-232C pin names and signal directions.

thestandard. Note that there is both a chassis ground (pin 1) and a signal ground (pin 7). To prevent largeac-induced ground currents in the signal ground, these two should be connected together only at thepowersupplyintheterminalorthecomputer.

The TxD, RxD, and handshake signals shown with common names in Figure 14-9 are the ones most oftenused for simple systems. These signals control what is called the *primary or forward* communicationschannel of the modem. Some modems allow communication over a secondary or *backward* channel,which operates in the reverse direction from the forward channel and at a much lower baud rate. Pins12,13,14,16,and19arethedataandhandshakelinesforthisbackwardchannel.Pins15,17,21,and24are usedforsynchronousdatacommunication.

UNIT-IV

IntroductiontoMicrocontrollers:

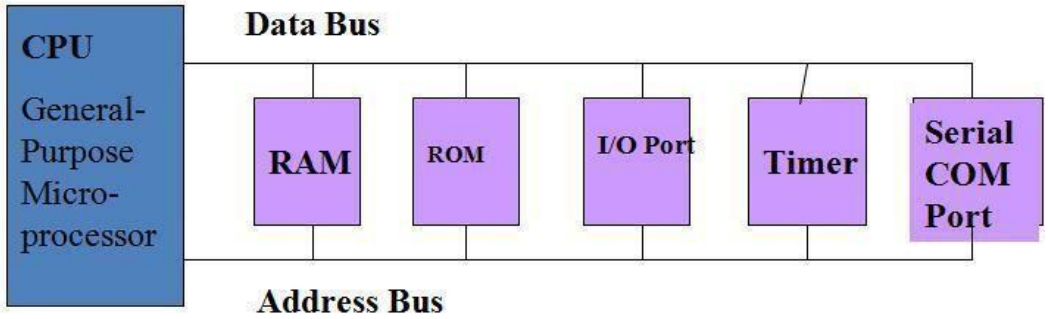
- **Overviewof8051 Microcontroller**
- **Architecture**
- **I/OPorts**
- **MemoryOrganization**
- **AddressingModesandInstruction set of8051**
- **SimplePrograms**
- **memoryinterfacing to 8051**

Thenecessarytoolsforamicroprocessor/controller:

- CPU:CentralProcessingUnit
- I/O:Input /Output
- Bus:Addressbus&Databus
- Memory:RAM &ROM
- Timer
- Interrupt
- SerialPort
- ParallelPort
-

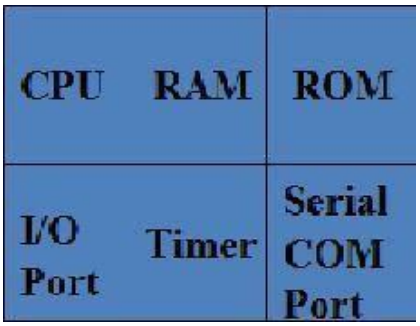
Microprocessors:
General-purposemicroprocessor:

- CPUfor Computers
- NoRAM,ROM, I/OonCPUchipitself
- Example : Intel'sx86,Motorola's680x0



Microcontroller:

- Asmallercomputer
- On-chipRAM, ROM, I/Oports...
- Example : Motorola's6811, Intel's8051,Zilog'sZ8andPIC16X
-



Microprocessorvs.Microcontroller:

Microprocessor

- CPU is stand-alone, RAM, ROM, I/O, timer are separate
- designer can decide on the amount of ROM, RAM and I/O ports.
- expansive
- versatility
- general-purpose

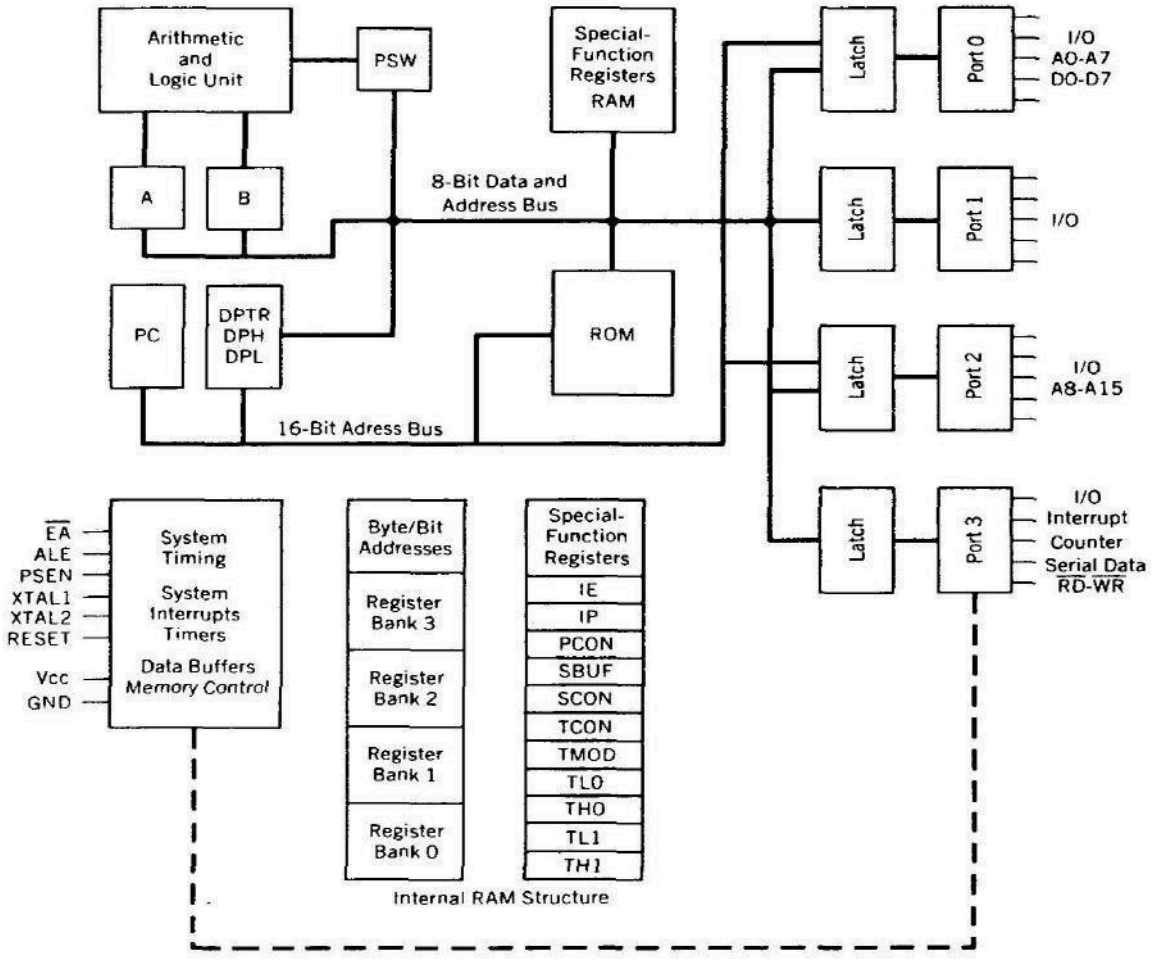
Microcontroller

- CPU, RAM, ROM, I/O and timer are all on a single chip
- fix amount of on-chip ROM, RAM, I/O ports
- for applications in which cost, power and space are critical
- single-purpose

8051MicrocontrollerHardware:

The 8051 microcontroller actually includes a whole family of microcontrollersthat have numbers ranging from 8031 to 8751 and are available in N-ChannelMetal Oxide Silicon (NMOS) and Complementary Metal Oxide Silicon (CMOS)construction inavarietyof

FIGURE 2.1a 8051 Block Diagram



housedina40-pinDIP,anddirecttheinvestigationofaparticulartypetothedatabook
s.

Theblockdiagramofthe8051inFigure 2.la showsallofthefeatures uniquetomicrocontrollers:

1. InternalROM andRAM
2. I/Oportswithprogrammablepins
3. Timersandcounters
4. Serialdatacommunication

ThefigurealsoshowstheusualCPUcomponents:programcounter,ALU,workingre
gisters,andclock circuits.'

The8051architectureconsistsofthesespecificfeatures:

Eight-bitCPUwithregistersA(theaccumulator)andB

Sixteen-bitprogramcounter(PC)anddatapointer(DPTR)

Eight-bitprogramstatus word(PSW)

Eight-bitstackpointer(SP)

InternalROMorEPROM(8751)of0(8031)to 4K(8051)

InternalRAMof128 bytes:

Fourregisterbanks,eachcontainingeightregisters

Sixteenbytes, whichmaybeaddressedatthebitlevel

Eightybytes ofgeneral-purposedatamemory

Thirty-twoinput/outputpinsarrangedasfour8-bitports:PO-P3

Two16-bittimer/counters:TOandTl

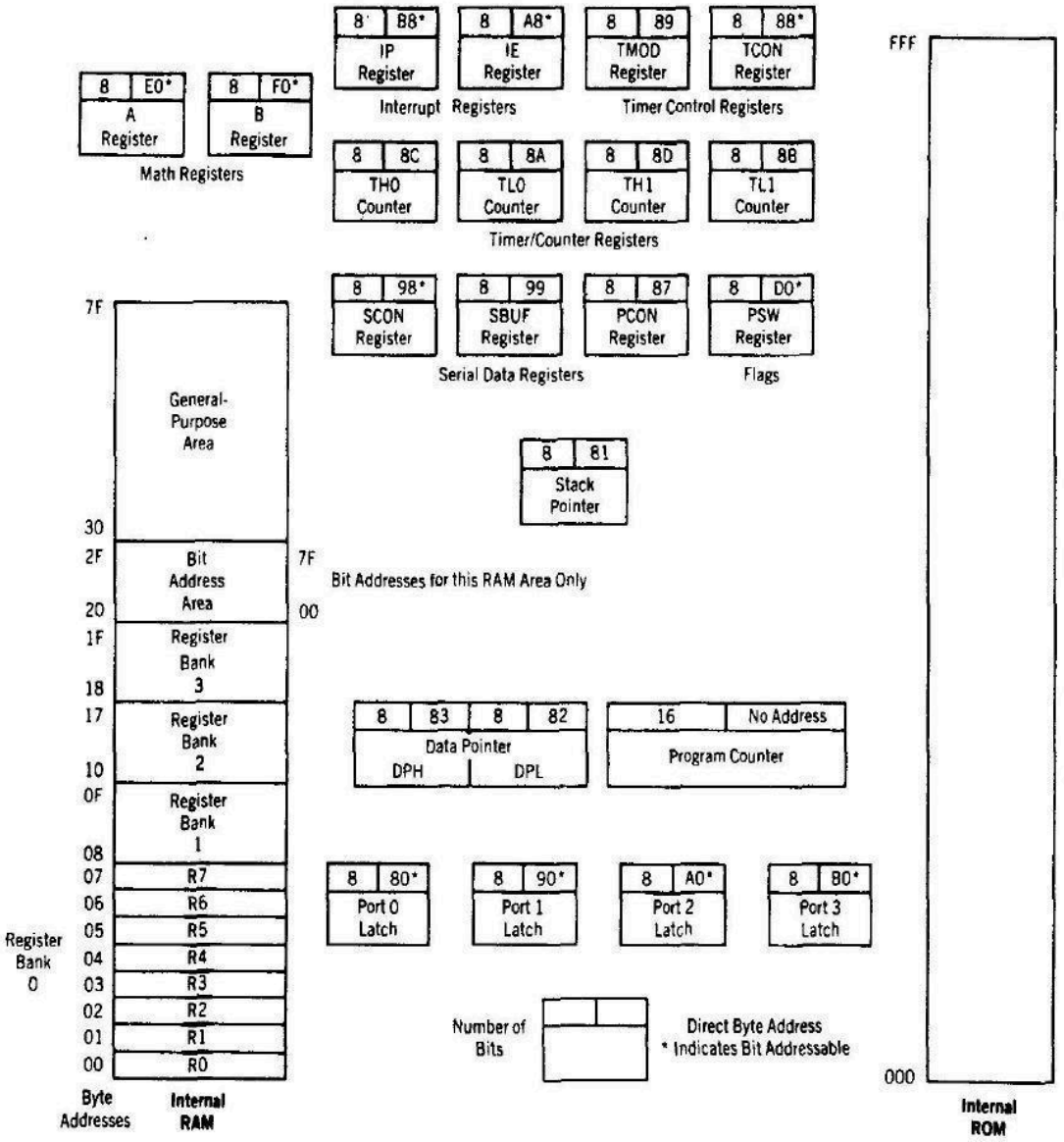
Fullduplexserialdatareceiver/transmitter:SBUF

Controlregisters:TCON,TMOD,SCON,PCON,IP,andIE

Twoexternalandthreeinternalinterruptsources

Oscillatorandclockcircuits

FIGURE 2.1b 8051 Programming Model



Theprogrammingmodelofthe8051inFigure2.Ibshowsthe8051asacollection of 8- and 16-bit registers and 8-bit memory locations. These registersand memory locations can be made to operate using the software instructions thatare incorporated as part of the design. The program instructions have to do withthe control of the registers and digital data paths that are physically containedinside the 8051, as well as memory locations that are physically located outsidethe8051.

The model is complicated by the number of special-purpose registers that must bepresent to make a microcomputer a microcontroller. A cursory inspection of themodel is recommended for the first-time viewer; return to the model as neededwhileprogressingthroughtheremainder of thetext.

Mostoftheregistershaveaspecificfunction;thosethatdooccupyanindividualbloc k with a symbolic name, such as A or THO or PC. Others, which aregenerally indistinguishable from each other, are grouped in a larger block, suchasinternalROMor RAMmemory.

Each register, with the exception of the program counter, has an internal 1-byteaddress assigned to it. Some registers (marked with an asterisk * in Figure 2.1b)are both byte and bit addressable. That is, the entire byte of data at such registeraddressesmay be read or altered, or individual bitsmay be read or altered.Software instructions are generally able to specify a register by its address, itssymbolic name, or both. A pinout of the 8051 packaged in a 40-pin DIP is shownin Figure 2.2 with the full and abbreviated names of the signals for each pin. It isimportant to note that many of the pins are used for more than one function (thealternate functions are shown in parentheses in Figure 2.2). Not all of the possible8051featuresmaybeused*atthesametime.*

Programming instructions or physical pin connections determine the use of anymultifunction pins. For example, port 3 bit 0 (abbreviated P3.0) may be used as ageneral purpose I/O pin, or as an input (RXD) to SBUF, the serial data receiverregister. The system designer decides which of these two functions is to be usedanddesignsthe hardwareandsoftware affectingthatpinaccordingly.

ProgramCounterandDataPointer

The 8051 contains two 16-bit registers: the program counter (PC) and the datapointer(DPTR).Eachisusedtoholdthe addressofa byte inmemory.

Programinstructionbytesarefetchedfromlocationsinmemorythatareaddressed by the PC. Program ROM may be on the chip at addresses OOOOh toOFFFh, external to the chip for addresses that exceed OFFFh, or totally externalfor all addresses from OOOOh to FFFFh. The PC is automatically incrementedaftereveryinstructionbyteisfetchedandmayalsobealteredbycertainin structions. ThePCistheonlyregisterthatdoesnothaveaninternaladdress.

The DPTR register is made up of two 8-bit registers, named DPH and DPL, thatare

used tofurnish memory addressesforinternal and external codeaccessandexternal data access. The DPTR is under the control of program instructions andcan be specified by its 16-bitname,DPTR,or by each individual bytename,DPH and DPL. DPTR does not have a single internal address; DPH and DPL areeachassignedanaddress.

AandB CPURegisters

The8051contains34general-purpose,orworking,registers.Twoofthese,registers A and B, comprise the mathematical core of the 8051 central processingunit (CPU). The other 32 are arranged as part of internal RAM in four banks,BO-B3,ofeightregisterseach,named ROto R7.

The A (accumulator) register is the most versatile of the two CPU registers and isused for many operations, including addition, subtraction, integer multiplicationand division, and Boolean bit manipulations. The A register is also used for alldata transfers between the 8051 and any external memory. The B register is usedwith the A register for multiplication and division operations and has no otherfunctionotherthanasalocationwheredatamaybestored.

FlagsandtheProgramStatusWord(PSW):

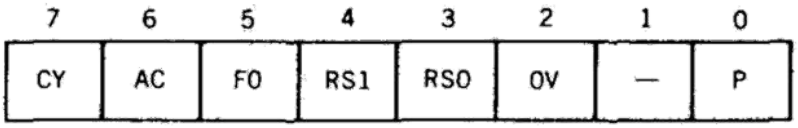
FlagsareI-bitregistersprovidedtostoretheresultsofcertainprograminstructions.Otherinstructionscan testthecondition of theflagsandmakedecisions based upon the flag states. In order that the flags may be convenientlyaddressed, they are grouped inside the program status word (PSW) and the powercontrol(PCON) registers.

The 8051 has four math flags that respond automatically to the outcomes of mathoperations and three general-purpose user flags that can be set to I or cleared to 0by the programmer as desired. The math flags include carry (C), auxiliary carry(AC), overflow (OV), and parity (P). User flags are named FO, GFO, and GF1;they are general-purpose flags that may be used by the programmer to recordsome event in the program. Note that all of the flags can be set and cleared by theprogrammeratwill.Themathflags,however,arealsoaffectedbymathoperations.

The program status word is shown in Figure 2.4. The PSW contains the mathflags,

user program flag FO, and the register select bits that identify which of the fourgeneralpurpose register banks is currently in use by the program. The remainingtwo user flags, GFO and GFl, arestored in PCON, which is shown in Figure2.13.

PSW Program Status Word Register



THE PROGRAM STATUS WORD (PSW) SPECIAL FUNCTION REGISTER

Bit	Symbol	Function
7	CY	Carry flag; used in arithmetic, JUMP, ROTATE, and BOOLEAN instructions
6	AC	Auxilliary carry flag; used for BCD arithmetic
5	F0	User flag 0
4	RS1	Register bank select bit 1
3	RS0	Register bank select bit 0

RS1	RS0	
0	0	Select register bank 0
0	1	Select register bank 1
1	0	Select register bank 2
1	1	Select register bank 3

2	OV	Overflow flag; used in arithmetic instructions
1	—	Reserved for future use
0	P	Parity flag; shows parity of register A: 1 = Odd Parity

Bit-addressable as PSW.0 to PSW.7

Detailed descriptions of the math flag operations will be discussed in chaptersthat

cover the opcodes that affect the flags. The user flags can be set or cleared usingdatamoveinstructionscoveredinChapter3.

InternalMemory:

A functioning computermust havememory for program code bytes, commonlyin ROM, and RAM memory for variable data that can be altered as the

programruns.The8051hasinternalRAMandROMmemoryforthesefunctions.Additional memorycanbeadded externally usingsuitablecircuits.

Unlike microcontrollers with Von Neumann architectures, which can use a *single*memory address for either program code or data, *but not for both,* the 8051 has aHarvard architecture, which uses the *same address,* in *different* memories, forcode and data. Internal circuitry accesses the correct memory based upon thenatureoftheoperation inprogress.

InternalRAM:

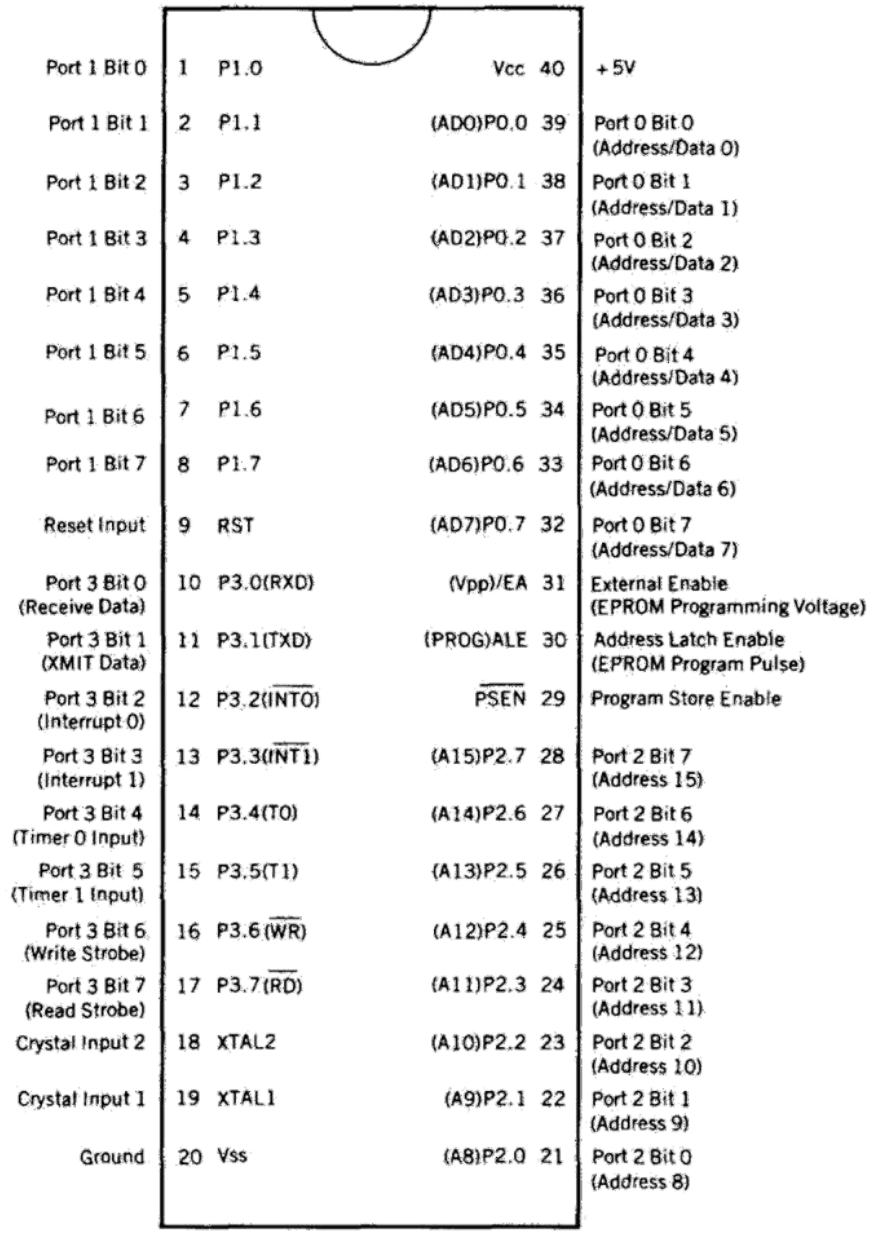
The128-byte internalRAM,whichis showngenerallyinFigure2.1andindetailin Figure2.5,isorganizedinto threedistinct areas:

1. Thirty-two bytes from address OOh to I Fh that make up 32 working registersorganized asfour banksof eightregisterseach.Thefour register banksarenumbered 0 to 3 and are made up of eight registers named RO to R7. Eachregister canbeaddressedbyname(whenitsbankisselected)orbyitsRAMaddress.Thus RO of bank 3 is RO (if bank 3 is currently selected) or address 18h(whether bank3isselectedor not). BitsRSOandRSIinthePSWdeterminewhichbankofregistersiscurrentlyinuseata nytime whenthe programisrunning. Registerbanksnotselectedcanbeusedasgeneral-purposeRAM.Bank0isselectedup onreset.

2. A Wf-addressable area of 16 bytes occupies RAM *byte* addresses 20h to 2Fh,formingatotalof128addressablebits.Anaddressablebitmaybespecifiedbyits *bit* address of OOh to 7Fh, or 8 bits may form any *byte* address from 20h to2Fh. Thus, for example, bit address 4Fh is also bit 7 of byte address 29h.Addressablebitsareusefulwhentheprogramneedonlyrememberabinaryevent (switch on, light off, etc.). Internal RAM is in short supply as it is, so why use abytewhenabitwilldo?

3. Ageneral-purposeRAMareaabovethebitarea,from30hto7Fh,addressable asbytes.

FIGURE 2.2 8051 DIP Pin Assignments



Note: Alternate functions are shown below the port name (in parentheses). Pin numbers and pin names are shown inside the DIP package.

PortO:

Port0pinsmayserveasinputs,outputs,or,whenusedtogether,asabi-directional loworder address and data bus for external memory. For example,when a pin is to be used as an input, a 1 *must be* written to the corresponding port0 latch by the program, thus turning both of the output transistors off, which inturncausesthepinto"float" inahighimpedancestate, andthepinis essentially

connectedtotheinputbuffer.

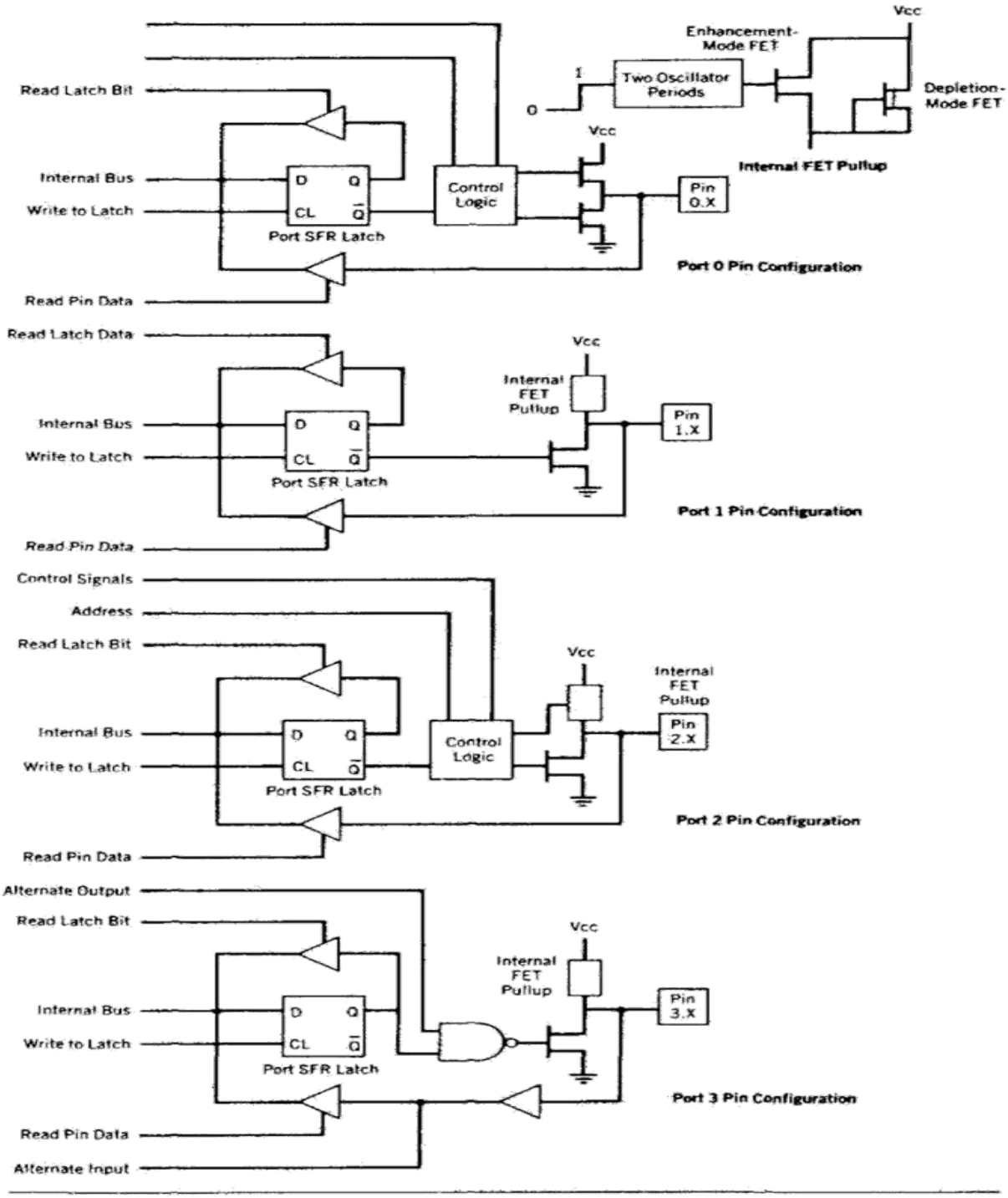
When used as an output, the pin latches that are programmed to a 0 will turn onthe

lower FET, grounding the pin. All latches that are programmed to a 1 still float;thus,external pullup resistors will be needed to supply a logic high when usingport 0asanoutput.

When port 0 isused as an address bus to external memory, internal controlsignals

switch the address lines to the gates of the Field Effect Transistories (FETs). Alogic I on an address bit will turn the upper FET on and the lower FET off toprovide a logic high at the pin. When the address bit is azero, the lower FET isonandtheupperFEToffto

Port Pin Circuits



provide a logic low at the pin. After the address has been formed and latched intoexternal circuits by the Address Latch Enable (ALE) pulse, the bus is turnedaround to become a data bus. Port 0nowreads data from the external memoryand must be configured as an input, so a logic 1 is automatically written byinternalcontrollogictoallport 0latches.

Portl

Port 1 pins have no dual functions. Therefore, the output latch is connecteddirectly to the gate of the lower FET, which has an FET circuit labeled "InternalFETPull up"asanactivepull upload.

Usedasaninput,a1iswrittentothelatch,turningthelowerFEToff;thepinand theinputto the pin buffer are pulled high by theFET load. An externalcircuitcanovercomethehighimpedancepullupanddrivethepinlowtoinput a0 or leavetheinputhighfor a1.

If used as an output, the latches containing a I can drive the input of an externalcircuithighthroughthepullup. Ifa0iswrittentothelatch,thelowerFET ison,thepullupisoff,andthe pincandrivethe inputofthe externalcircuitlow.

To aid in speeding up switching times when the pin is used as an output, theinternal FET pull uphasanotherFETin parallel with it.ThesecondFET isturned on for two oscillator time periods during a low-to-high transition on thepin,asshownin Figure2.7.

This arrangement provides a low impedance path to the positive voltage supply tohelpreduce rise timesin charging any parasitic capacitancesin the externalcircuitry.

Port2

Port 2 may be used as an input/output port similar in operation to port 1. Thealternate use of port 2 is to supply a high-order address byte in conjunction withtheport 0 low-order byteto addressexternal memory.

Port2pinsaremomentarilychangedbytheaddresscontrolsignalswhensupplying the high byte of a 16-bit address. Port 2 latches remain stable whenexternal memory is addressed, as they do not have to be turned around (set to 1)fordatainputasisthecaseforport0.

Port3

Port3 isaninput/outputportsimilartoportI.Theinputandoutputfunctionscan

be programmed under the control of the P3 latches or under the control of variousotherspecialfunctionregisters.Theport3alternateusesareshowninthefollowing table:-

PIN	ALTERNATE USE	SFR
P3.0–RXD	Serial data input	SBUF
P3.1–TXD	Serial data output	SBUF
P3.2–$\overline{\text{INT0}}$	External interrupt 0	TCON.1
P3.3–$\overline{\text{INT1}}$	External interrupt 1	TCON.3
P3.4–T0	External timer 0 input	TMOD
P3.5–T1	External timer 1 input	TMOD
P3.6–$\overline{\text{WR}}$	External memory write pulse	—
P3.7–$\overline{\text{RD}}$	External memory read pulse	—

Unlike ports 0 and 2, which can have external addressingfunctions and changeall

eight port bits when in alternate use, each pin of port 3 may be individuallyprogrammedtobe usedeitheras I/Oorasoneofthe alternatefunctions.

ExternalMemory

The system designer is not limited by the amount of internal RAM and ROMavailable on chip. Two separate external memory spaces are made available bythe 16-bit PC and DPTR and by different control pins for enabling external ROMand RAM chips. Internal control circuitry accesses the correct physical memory,dependinguponthemachine cycle state andthe opcode beingexecuted.

Thereareseveralreasonsforaddingexternalmemory,particularlyprogrammemory, whenapplyingthe8051inasystem.Whentheprojectisintheprototype stage, the expense—in time and money—of having a masked internalROMmadeforeachprogram"try"isprohibitive.

To alleviate this problem, the manufacturers make available an EPROM version,the 8751, which has 4K of on-chip EPROM that may be programmed and erasedas needed as the program is developed. The resulting circuit board layout will beidentical to one that uses a factory-programmed 8051. The only drawbacks to the8751 are the specialized EPROM programmers that must be used to program thenon-standard 40-pin part, and thelimit of "only" 4096 bytes of program code.The8751solutionworkswelliftheprogramwillfitinto4Kbytes.Unfortunately, many times, particularly if the program is written in a high-levellanguage, the program size exceeds 4K bytes, and an external program memory isneeded. Again, the manufacturers provide a version for the job, the ROMIess8031. The EA pin is grounded when using the 8031, and all program code iscontained in an external EPROM that may be as large as 64K bytes and that canbeprogrammed usingstandard EPROMprogrammers.

External RAM, which is accessed by the DPTR, may also be needed when 128bytes of internal data storage is not sufficient. External RAM, up to 64K bytes,mayalso beaddedto anychipinthe8051family.

ConnectingExternalMemory

Figure 2.8 shows the connections between an 8031 and an external memoryconfiguration consisting of I6K bytes of EPROM and 8K bytes of static RAM.The8051accessesexternal RAMwhenevercertain program instructionsareexecuted. External ROM is accessed whenever the EA (external access) pin isconnected to ground or when the PC contains an address higher than the lastaddressin theinternal 4K bytesROM(OFFFh).8051 designscan thususeinternal and external ROM automatically; the 8031, having no internal ROM,musthaveEAgrounded.

Figure 2.9 shows the timing associated with an external memory access cycle.During any memory access cycle, port 0 is time multiplexed. That is, it firstprovidesthelowerbyteofthe16-bitmemoryaddress, thenactsasabidirectional

data bus to write or read a byte of memory data. Port 2 provides the high byte ofthememoryaddressduringtheentirememoryread/writecycle.

The lower address byte from port 0 must be latched into an external register tosave
thebyte.AddressbytesaveisaccomplishedbytheALEclockpulsethatprovidesthec orrecttimingforthe'373typedatalatch.Theport0pinsthenbecomefreetoserveasada tabus.
If the memory access is for a byte of program code in the ROM, the PSEN(program store enable) pin will go low to enable the ROM to place a byte ofprogram code on the data bus. If the access is for a RAM byte, the WR (write) orRD (read) pins will go low, enabling data to flow between the RAM and the databus.

TheROMmaybeexpandedto64Kbyusinga27512typeEPROMandconnectingthe remainingport2upperaddresslinesAI4-A15tothe chip.
At this time the largest static RAMs available are 32K in size; RAM can beexpanded to 64K by using two 32K RAMs that are connected through addressA14 ofport2.The

FIGURE 2.8 External Memory Connections

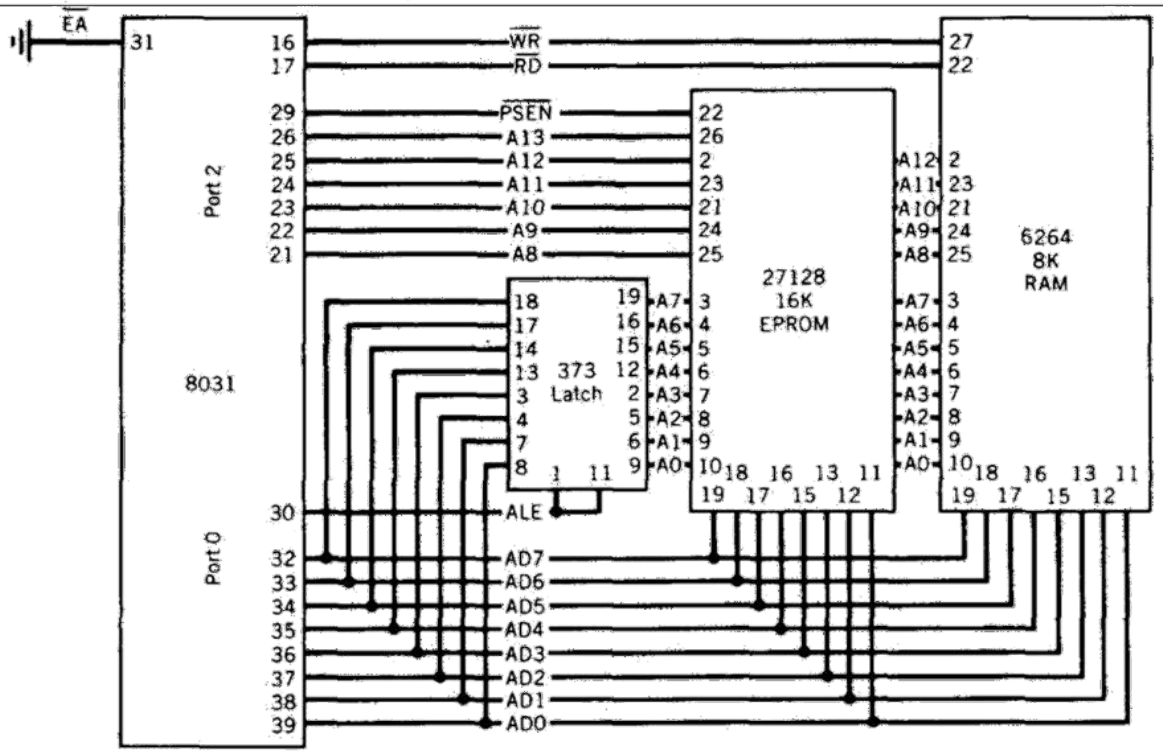
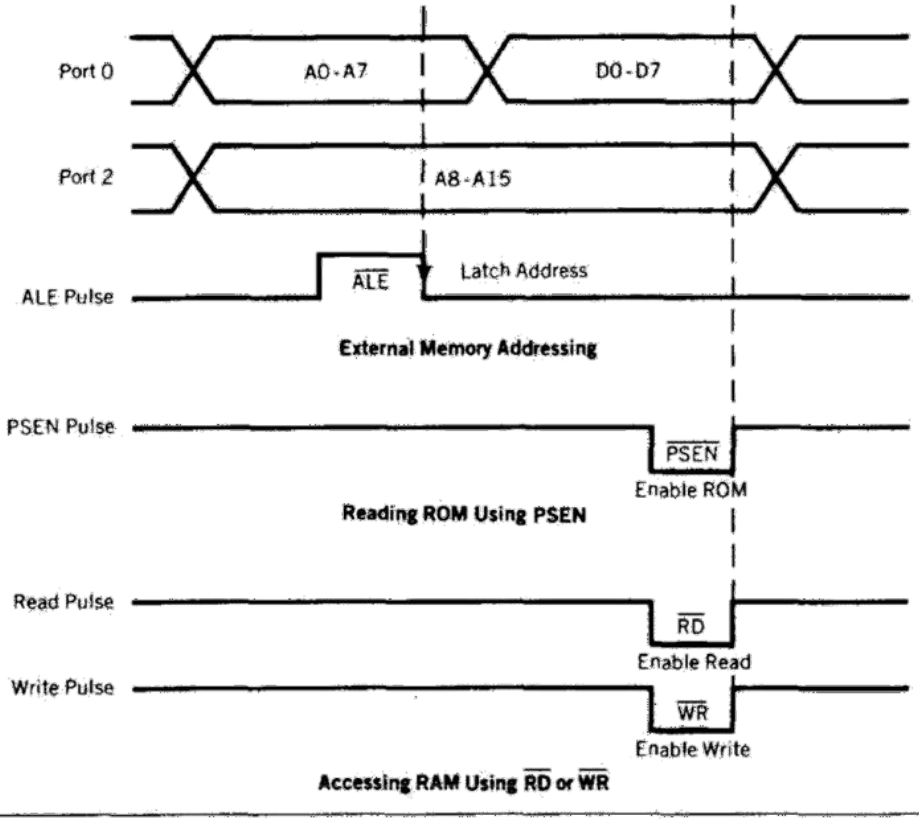


FIGURE 2.9 External Memory Timing



first 32K RAM (OOOOh-7FFFh) can then be enabled when AI5 of port 2 is low,and the second 32K RAM (SOOOh-FFFFh) when A15 ishigh, by using aninverter.

Note that the WR and RD signals are alternate uses for port 3 pins 16 and 17.Also,

port 0 is used for the lower address byte and data; port 2 is used for upper addressbits. The use of external memory consumes many of the port pins, leaving onlyport 1and partsofport3forgeneralI/O.

8051 INSTRUCTIONSET

8051hasabout 111instructions.Thesecanbegroupedintothefollowingcategories

- ArithmeticInstructions
- LogicalInstructions
- DataTransferinstructions
- BooleanVariableInstructions
- ProgramBranchingInstructions

Thefollowingnomenclaturesforregister,data,addressandvariablesareusedwhilewriteinstructions

- A:Accumulator
- B: "B"register
- C: Carry bit
- Rn:RegisterR0-R7ofthecurrentlyselectedregisterbank
- Direct:8-bitinternaldirectaddressfordata.Thedatacouldbeinlower128bytesof RAM(00 -7FH) oritcouldbein thespecialfunctionregister(80 - FFH).
- @Ri:8-bitexternalorinternalRAMaddressavailableinregisterR0orR1.Thisisused forindirectaddressingmode.
- #data8:Immediate8-bitdataavailableintheinstruction.
- #data16:Immediate16-bitdata availableintheinstruction.
- Addr11:11-bitdestinationaddressforshortabsolutejump.UsedbyinstructionsAJMP &ACALL.Jumprangeis2kbyte(onepage).
- Addr16:16-bitdestinationaddressforlongcallorlongjump.
- Rel:2'scomplement8-bitoffset(one-byte)usedforshortjump(SJMP) andall conditionaljumps.
- bit:DirectlyaddressedbitininternalRAMorSFR

SomeSimpleInstructions:

MOVdest,source

;dest=sourceMOVA,#72H

;A=72H

MOVR4,#62H ;R4=62H

MOV B,0F9H ;B=thecontentofF9'thbyteofRAM

MOV

DPTR,#7634HMO

V

DPL,#34HMOV

DPH,#76H

MOV P1,A ;movAtoport1

Note1: MOV A,#72H ≠MOVA,72H

Afterinstruction"MOVA,72H "thecontentof72'thbyteofRAMwillreplace

in

Accumulato

r.Note 2:

MOV A,R3≡ MOVA,3

ADD A,Source ;A=A+SOURCE

ADD A,#6 ;A=A+6

ADD A,R6 ;A=A+R6

ADD A,6 ;A=A+[6]orA=A+R6

ADD A,0F3H ;A=A+[0F3H]

SUBBA,Source ;A=A-SOURCE-C

SUBB A,#6 ;A=A-6

SUBB A,R6 ;A=A+R6

MUl&Div:

- MUL AB ;B|A=A*B

MOV A,#25H

MOV B,#65H

MUL AB ;25H*65H=0E99

- DIV AB ;A=A/B,B=AmodB ;B=0EH,A=99H

```
MOV   A,#25
      MOV   B,#10
      DIV   AB          ;A=2,B=5
SETB  bit               ;bit=1
CLR   bit               ;bit=0
SETB  C                 ;CY=1
SETB  P0.0              ;bit0fromport0=1
SETB  P3.7              ;bit7fromport3=1
SETB  ACC.2             ;bit2fromACCUMULATOR=1
SETB  05                ;sethighD5ofRAMloc.20h
```

Note:

CLRinstructionisassamea

sSETBi.e.:

```
      CLR   C           ;CY=0
```

ButfollowinginstructionisonlyforCLR:

```
      CLR   A           ;A=0
DEC   byte              ;byte=byte-1
INC   byte              ;byte=byte+1
INC   R7
DEC   A
DEC   40H               ;[40]=[40]-1
```

RR – RL – RRC – RLC A

EXAMPLE:

```
  RR    A
```

RR:

RRC:

RL:

RLC:

ANL-ORL–XRL

BitwiseLogicalOperatio

ns:AND,OR, XOR

EXAMPLE:

```
      MOV  R5,#89H
      ANL  R5,#08H
```

```
CPL     A       ;1'scomplement
```

Example:

```
        MOV     A,#55H;A=01010101B
L01:    CPL     A
        MOV     P1,A
        ACALL   DELAY
        SJMP    L01
```

UNIT-V

8051RealTimeControl:

- ProgrammingTimerInterrupts
- ProgrammingExternalHardware
- Interrupts
- ProgrammingtheSerialCommunicationInterrupts
- Programming8051TimersandCounters

ARMProcessor:

- Fundamentals
- Registers
- currentprogramstatusregister,
- pipeline
- Interruptandthevectortable

Interrupts:

1. EnablingandDisablingInterrupts
2. InterruptPriority
3. WritingtheISR(InterruptServiceRoutine)

InterruptEnable(IE)Register:

- EA:Globalenable/disable.
- --- :Undefined.
- ET2:EnableTimer2interrupt.
- ES:EnableSerialport interrupt.
- ET1:EnableTimer1interrupt.
- EX1:EnableExternal1interrupt.
- ET0:EnableTimer0interrupt.
- EX0:EnableExternal0interrupt.

EA	—	ET2	ES	ET1	EX1	ET0	EX0

InterruptVectors:

Interrupt	Vector Address
System Reset	0000H
External 0	0003H
Timer 0	000BH
External 1	0013H
Timer 1	001BH
Serial Port	0023H
Timer 2	002BH

PeripheralControlRegist

ersPCON

(PowerControl)

The PCON or Power Control register, as the name suggests is used to control the 8051Microcontroller'sPowerModesandislocatedat87HoftheSFRMemorySpace.Usingtwobitsin thePCON Register,themicrocontrollercanbeset toIdleModeandPowerDownMode.

During Idle Mode, the Microcontroller will stop the Clock Signal to the ALU (CPU) but it isgiventoother peripheralslikeTimer,Serial,Interrupts,etc.InordertoterminatetheIdleMode,youhavetouseanInte rruptorHardwareReset.

InthePowerDownMode,theoscillatorwillbestoppedandthepowerwillbe reduced to 2V.ToterminatethePowerDownMode,youhave tousethe HardwareReset.

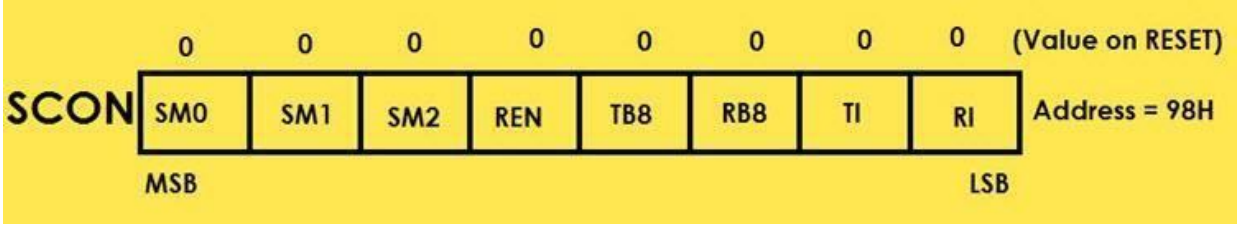
Apartfromthesetwo,thePCONRegister canalsobeusedforfewadditionalpurposes.TheSMOD Bitinthe PCON Registerisusedtocontrolthe BaudRateofthe SerialPort.

TherearetwogeneralpurposeFlagBitsinthePCONRegister,whichcan beusedbytheprogrammerduringexecution.

SCON(SerialControl)

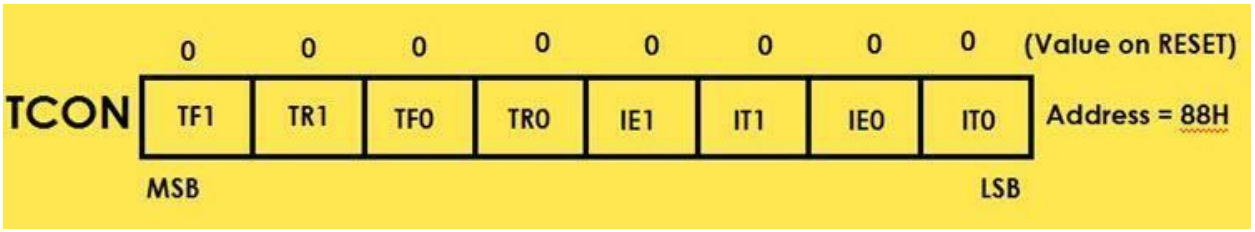
TheSerialControlor SCONSFRisusedtocontrolthe8051Microcontroller'sSerialPort.Itislocated as an address of 98H. Using SCON, you can control the Operation Modes of the SerialPort,BaudRateoftheSerialPortandSendorReceiveData usingSerialPort.

SCONRegister alsoconsistsofbitsthatareautomaticallySETwhenabyteofdataistransmittedorreceived.



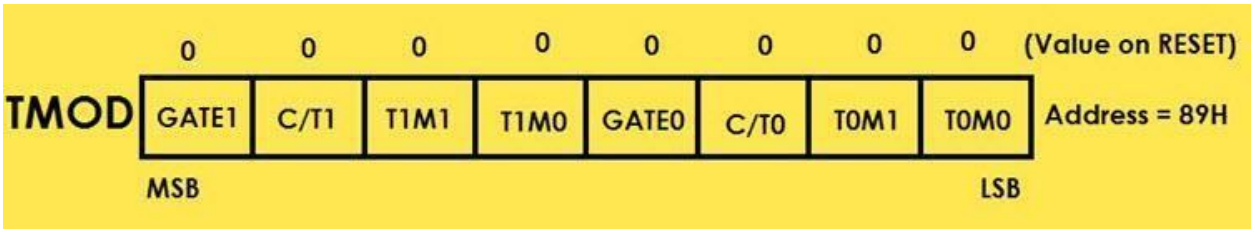
TCON(TimerControl)

Timer ControlorTCONRegister isused to startorstoptheTimersof8051 Microcontroller.Italso contains bits to indicate if the Timers has overflowed. The TCON SFR also consists ofInterruptrelatedbits.



TMOD(TimerMode)

TheTMODorTimer ModeregisterorSFRisusedtosettheOperatingModesoftheTimersT0and T1. The lower four bits are used to configure Timer0 and the higher four bits are used toconfigure Timer1.



TheGatexbitisusedto operatetheTimerxwithrespect to theINTxpinorregardlessoftheINTxpin.

GATE1 = 1 ==> Timer1 is operated only if INT1 is

SET.GATE1=0==>Timer1isoperatesirrespectiveofINT1pi

n.GATE0 = 1 ==> Timer0 is operated only if INT0 is

SET.GATE0=0==>Timer0isoperatesirrespectiveofINT0

pin.

The C/Tx bit is used selects the source of pulses for the Timer to

count.C/T1 = 1 ==> Timer1 counts pulses from Pin T1 (P3.5) (Counter

Mode)C/T1=0==>Timer1countspulses frominternaloscillator(Timer

Mode)

C/T0 = 1 ==> Timer0 counts pulses from Pin T0 (P3.4) (Counter

Mode)C/T0=0==>Timer0countspulses frominternaloscillator(Timer

Mode)

TxM0	Description
0	13-bitTimerMode(THx–8-bitandTLx–5-bit)
0	16-bitTimerMode
1	8-bitAutoReloadTimerMode
1	Two8-bitTimerModeorSplitTimerMode

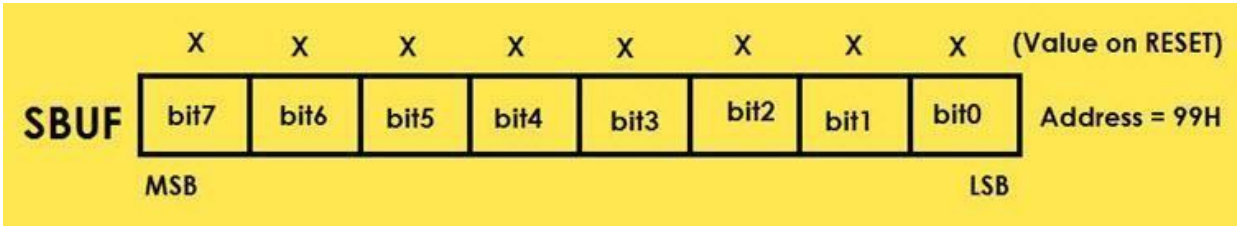
IP(InterruptPriority)

The IP or Interrupt Priority Register is used to set the priority of the interrupt as High or Low. Ifa bit is CLEARED, the corresponding interrupt is assigned low priority and if the bit is SET, theinterruptis assignedhighpriority.

TxM1	Mode
0	0
1	1
0	2
1	3

Peripheral Data RegistersSBUF(Serial DataBuffer)

TheSerialBuffer orSBUFregister isusedtoholdtheserialdatawhiletransmissionorreception.



ARMprocessor

AnARMprocessorisone ofafamilyofCPUsbased onthe RISC(reduced instructionsetcomputer)architecturedevelopedbyAdvancedRISCMachines(ARM).

ARM makes 32-bit and 64-bit RISC multi-core processors. RISC processors are designed toperform a smaller number of types of computer instructions so that they can operate at a higherspeed, performingmoremillionsofinstructionsper second(MIPS).Bystrippingoutunneededinstructions and optimizing pathways, RISC processors provide outstanding performance at afractionofthe powerdemand ofCISC (complexinstructionsetcomputing)devices.

ARMprocessorsareextensivelyusedinconsumer electronicdevicessuch as smartphones, tablets, multimedia players and other mobile devices, such as wearables.Because of their reduced instruction set, they require fewer transistors, which enables a smallerdie size for the integrated circuitry (IC). The ARM processor's smaller size, reduced complexityandlower powerconsumption makesthemsuitablefor increasinglyminiaturized devices.

ARMprocessorfeaturesinclude:

- Load/storearchitecture.
- Anorthogonalinstructionset.
- Mostlysingle-cycleexecution.
- Enhancedpower-savingdesign.
- 64and32-bitexecutionstatesforscalablehighperformance.
- Hardwarevirtualizationsupport.

The simplified design of ARM processors enables more efficient multi-core processing andeasier coding for developers. While they don't have the same raw compute throughput as theproducts of x86 market leader Intel, ARM processors sometimes exceed the performance of Intelprocessorsforapplicationsthatexistonbotharchitectures.

Thehead-to-headcompetitionbetweenthevendorsisincreasingasARMisfindingitswayintofull size notebooks.Microsoft, for example, offers ARM-based versions of Surface computers.The cleaner code base of Windows RT versus x86 versions may be also partially responsible --WindowsRTismorestreamlinedbecauseitdoesn'thaveto supportanumberoflegacyhardwares.

ARM is also moving into the server market,a move that represents a large change in directionand a hedging of bets on performance-per-watt over raw compute power. AMD offers 8-coreversions of ARM processors for its Opteron series of processors. ARM serversrepresent animportant shift in server-based computing. A traditional x86-class server with 12, 16, 24 or morecores increases performance by scaling up the speed and sophistication of each processor, usingbrute force speedandpowertohandle demandingcomputingworkloads.

In comparison, an ARM server uses perhaps hundreds of smaller, less sophisticated, low-powerprocessors that share processing tasks among that large number instead of just a few higher-capacityprocessors.
Thisapproachissometimesreferredtoas"scalingout,"incontrastwiththe"scalingup"ofx86-baseds ervers.

ARMregisters

ARMprocessorsprovidegeneral-purposeandspecial-purposeregisters.Someadditionalregister sareavailableinprivilegedexecutionmodes.

InallARMprocessors,thefollowingregistersareavailableandaccessibleinanyprocessormode:

- 13general-purposeregistersR0-R12.
- One*StackPointer*(SP).
- One*LinkRegister*(LR).
- One*ProgramCounter*(PC).
- One*ApplicationProgramStatusRegister*(APSR).

ARM processors, with the exception of ARMv6-M and ARMv7-M based processors, have atotalof37 registers,with3additionalregistersiftheSecurityExtensionsareimplemented,andin ARMv7-A only, 3 more if the Virtualization Extensions are implemented. The registers arearranged in partially overlapping banks. There is a different register bank for each processormode. The banked registers give rapid context switching for dealing with processor exceptionsandprivilegedoperations.

Theadditionalregistersthatareavailableinprivilegedsoftwareexecution,withtheexceptionofARMv6-MandARMv7-M,are:

- TwoSupervisormoderegistersforbankedSPandLR.
- TwoAbortmoderegistersforbankedSPandLR.
- TwoUndefinedmoderegistersforbankedSPandLR.
- TwoInterruptmoderegistersforbankedSPandLR.
- SevenFIQmoderegisters forbankedR8-R12, SPandLR.
- TwoMonitormoderegistersfor bankedSPandLR.TheseareonlypresentiftheSecurityExtensionsareimplemented.
- TwoHypmoderegisters forbankedSP, andtoholdthereturnaddress fromHypmode.Theseare onlypresentifthe VirtualizationExtensions areimplemented.
- One*SavedProgramStatusRegister*(SPSR)for eachexceptionmode.

CurrentProgramStatusRegister

The*CurrentProgramStatusRegister*(CPSR) holdsthesameprogramstatusflagsastheAPSR,andsomeadditionalinformation.

TheCPSRholds:

- TheAPSRflags.
- Theprocessormode.
- Theinterruptdisableflags.
- Theinstructionsetstate(ARM,Thumb, ThumbEE,orJazelle®).
- Theendiannessstate(onARMv4Tandlater).
- TheexecutionstatebitsfortheITblock(onARMv6T2andlater).
-

Theexecutionstatebitscontrolconditionalexecutioninthe ITblock.
Only the APSR flags are accessible in all modes. ARM deprecates using an MSRinstruction tochangetheendiannessbit(E)oftheCPSR,inanymode.SETENDisthepreferredinstructiontowritetotheEbit.

Theexecutionstatebits fortheIT block(IT[1:0]),Jazellebit(J), andThumbbit(T)canbeaccessedbyMRSonlyinDebugstate.

Theinstructionpipeline

The ARM uses a pipeline to increase the speed of the flow of instructions to the processor. Thisallowsseveraloperationstotakeplacesimultaneously, andtheprocessing,andmemorysystemstooperatecontinuously.

Athree-stagepipelineisused, soinstructionsareexecutedinthreestages:

- Fetch
- Decode
- Execute.

Duringnormaloperation,whileoneinstructionisbeingexecuted, itssuccessorisbeingdecoded,anda thirdinstructionisbeingfetchedfrommemory.