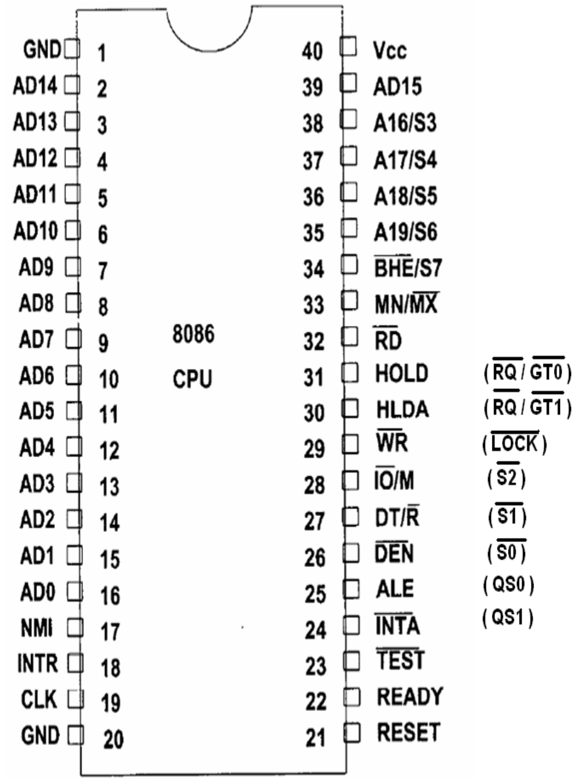
Microprocessors Techniques II

YAMAMA A. SHAFEEK

8086 Pin Diagram

The 8086, announced in 1978, was the first16-bit microprocessor introduced by Intel Corporation. The 8086 is manufactured using high-performance metal-oxide semiconductor (HMOS) technology, and the circuitry on its chips is equivalent to approximately 29000 transistors. It is housed in a 40-pin dual in-line package.

Many of 8086's pins have multiple functions. For example, the address bus lines **A_0** through **A_{15}** and data bus lines **D_0** through **D_{15}** are multiplexed. For this reason, these leads are labeled **AD_0** through **AD_{15}**. By multiplexed we mean that the same physical pin carries an address bit at one time and the data bit at another time.



8086 Modes of Operation

The 8086 can be configured to work in either of two modes:

- ***Minimum Mode*** is selected by applying logic 1 to the $MN/\overline{MX}$ input lead. Minimum mode 8086 systems are typically smaller and contain a single microprocessor.
- ***Maximum Mode*** is selected by applying logic 0 to the $MN/\overline{MX}$ input lead. Maximum mode configures 8086 systems for use in larger systems and with multiple processors.

Depending on the mode of operation selected, the assignments for a number of pins on the microprocessor package are changed. As shown above, the pin function of the 8086 specified in parentheses relate to a maximum-mode system.

8086 Common Signals

Name	Function	Type
AD15-AD0	Address /data bus	Bidirectional
A19/S6-A16/S3	Address / status	Output
MN/$\overline{MX}$	Minimum/Maximum mode control	Input
$\overline{RD}$	Read control	Output
$\overline{TEST}$	Wait on test control	Input
READY	Wait state control	Input
RESET	System reset	Input
NMI	Non-maskable interrupt request	Input
INTR	Interrupt request	Input
CLK	System clock	Input
V_{CC}	+5 volt	Input
GND	Ground	Input

Minimum Mode Interface Signals

The minimum-mode signals are shown in Table 1. They can be divided into the following basic groups:

Name	Function	Type
HOLD	Hold request	Input
HLDA	Hold acknowledgment	Output
$\overline{WR}$	Write control	Output
$M/\overline{IO}$	IO/memory control	Output
$DT/\overline{R}$	Data transmit /receive	Output
$\overline{DEN}$	Data enable	Output
$\overline{BHE}$ /S7	Bank high enable/Status line 7	Output
ALE	Address latch enable	Output
$\overline{INTA}$	Interrupt acknowledgment	Output

Maximum Mode Interface Signals

When the 8086 microprocessor is set for the maximum-mode configuration, it produces signals for implementing a *multiprocessor/coprocessor system environment*. By multiprocessor system environment we mean that multiple microprocessors exist in the system and that each processor executes its own program. During the maximum mode operation, the $\overline{\mathrm{WR}}$, M/$\overline{\mathrm{IO}}$, DT/$\overline{\mathrm{R}}$, $\overline{\mathrm{DEN}}$, ALE, and $\overline{\mathrm{INTA}}$ signals are no longer produced by the 8086. Instead, it outputs a status code on three signals lines, $\overline{\mathrm{S}}_0$, $\overline{\mathrm{S}}_1$, and $\overline{\mathrm{S}}_2$, prior to the initiation of each bus cycle.

Name	Function	Type
$\overline{\mathbf{RQ/GT1,0}}$	**Request/grant bus access control**	**Bidirectional**
$\overline{\mathbf{LOCK}}$	**Bus priority lock control**	**Output**
$\overline{\mathbf{S2}} - \overline{\mathbf{S0}}$	**Bus cycle status**	**Output**
QS1, QS0	**Instruction queue status**	**Output**

System Clock

The time base for synchronization of the internal and external operations of the microprocessor in a microcomputer system is provided by the *clock* (CLK) input signal.

An external crystal oscillator is connected to the X1 and X2 inputs of an 8284 clock generator as shown in Figure 4. CLK (Clock) output of 8284 Provides CLK input signal to the 8086 microprocessor and other components in the system. fclk =1/3 fcrystal so that Duty cycle is 33% which is required by 8086.

The 8086 microprocessor is manufactured in three speeds: the **5-MHz** 8086, the **8-MHz** 8086-2 and the **10-MHz** 8086-1.

Bus Cycle and Time State

A *bus cycle* defines the basic operation that a microprocessor performs to communicate with external devices such as Memory read, Memory write, I/O read, and I/O write.

The bus cycle of 8086 microprocessors consists of at least four clock periods (T_1, T_2, T_3, and T_4):

- During T_1 the 8086 puts an address on the bus.
- During T_2 the 8086puts the data on the bus (for write memory cycle) and maintained through T_3 and T_4.
- During T_2 the 8086puts the bus in high-Z state (for read cycle) and then the data to read must be available on the bus during $T_3$and T_4.

These four clock states give a bus cycle duration of 125 ns × 4= 500 ns in an 8-MHz system.

Idle States and Wait States

- ***Write Cycle*** The write bus cycle is similar to the read bus cycle except that signal $\overline{\mathrm{WR}}$ is set to 0 instead of the signal $\overline{\mathrm{RD}}$ and signal $\mathrm{DT}/\overline{\mathrm{R}}$ is set to 1.

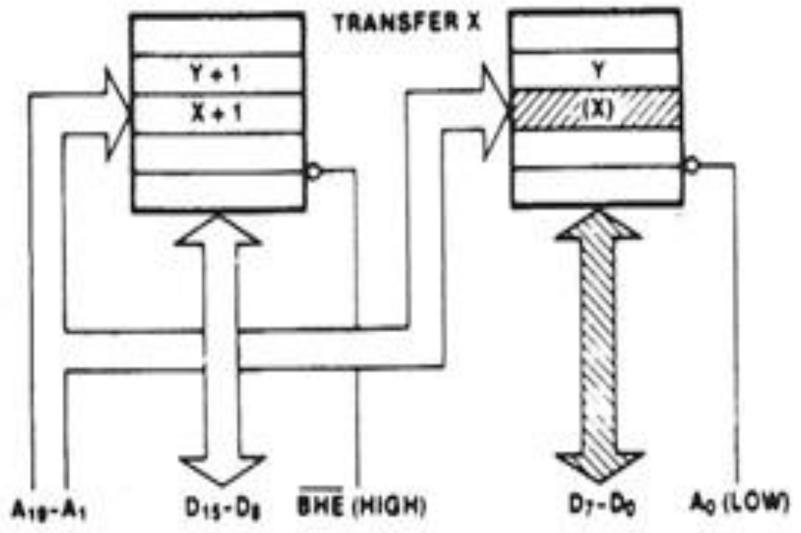
- ***Idle States*** If no bus cycles are required, the microprocessor performs what is known as *idle state*. During these states, no bus activity takes place. Each idle state is one clock period long, and any number of them can be inserted between bus cycles. Idle states are performed if the instruction queue inside the microprocessor is full and it does not need to read or write operands from memory.

- ***Wait States*** Wait states can be inserted into a bus cycle. This is done in response to request by an event in external hardware instead of an internal event such as a full queue. The READY input of the 8086is provided specifically for this purpose. As long as READY is held at the 0 level, wait states are inserted between states $T_3$and T_4 of the current bus cycle, and the data that were on the bus during T_3 are maintained. The bus cycle is not completed until the external hardware returns READY back to the 1 logic level.

Hardware Organization of the 8086 Memory Address Space

The 8086's 1Mbyte memory address space is implemented as two independent 512 Kbyte banks: the *low (even) bank* and the *high (odd) bank.*

- When a **byte** of data at an **even** address (such as X) is to be accessed:
- A_0 is set to logic 0 to enable the low bank of memory.
- $\overline{\text{BHE}}$ is set to logic 1 to disable the high bank.

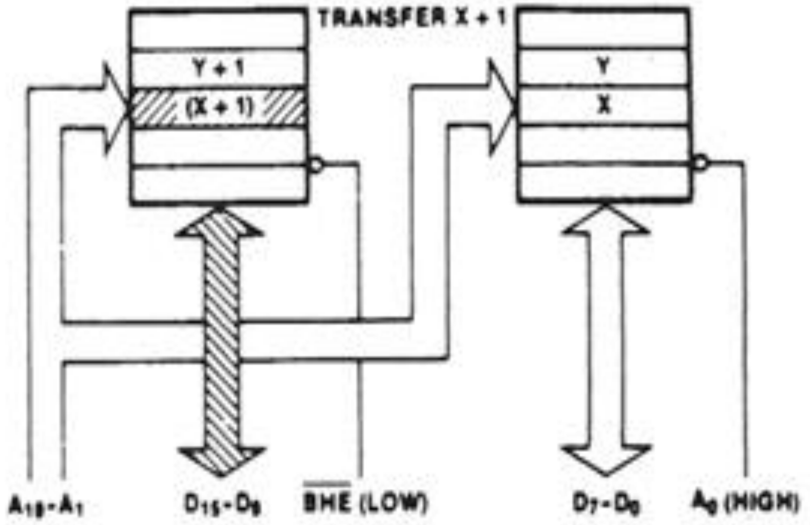


Hardware Organization of the 8086 Memory Address Space

- When a **byte** of data at an **odd** address

(such as X+1) is to be accessed:

- A_0 is set to logic 1 to disable the low bank of memory.
- $\overline{BHE}$ is set to logic 0 to enable the high bank.



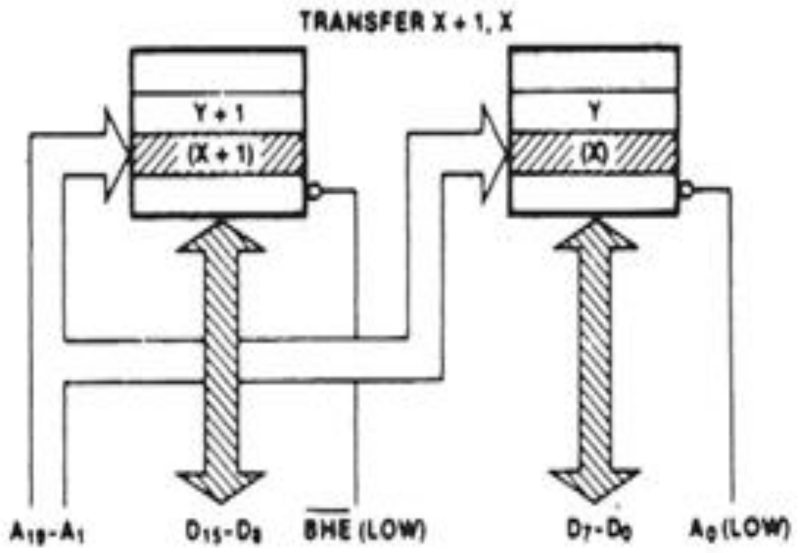
Hardware Organization of the 8086 Memory Address Space

➢When a **word** of data at an **even** address

(**aligned word**) is to be accessed:

❖A_0 is set to logic 0 to enable the low bank of memory.

❖$\overline{\text{BHE}}$ is set to logic 0 to enable the high bank.



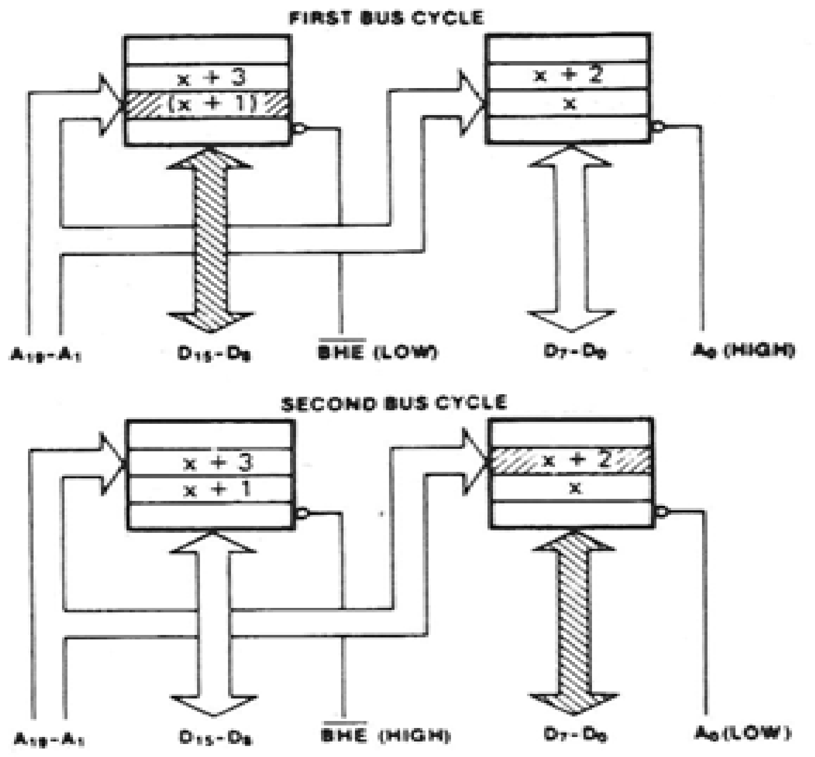
Hardware Organization of the 8086 Memory Address Space

- When a **word** of data at an **odd** address (**misaligned word**) is to be accessed the 8086 needs **two bus cycles** to access it:

- During the first bus cycle, the odd byte of the word (in the high bank) is addressed
  - A_0 is set to logic 1 to disable the low bank of memory.
  - $\overline{\text{BHE}}$ is set to logic 0 to enable the high bank.

- During the second bus cycle, the even byte of the word (in the low bank) is addressed
  - A_0 is set to logic 0 to enable the low bank of memory.
  - $\overline{\text{BHE}}$ is set to logic 1 to disable the high bank.

Hardware Organization of the 8086 Memory Address Space



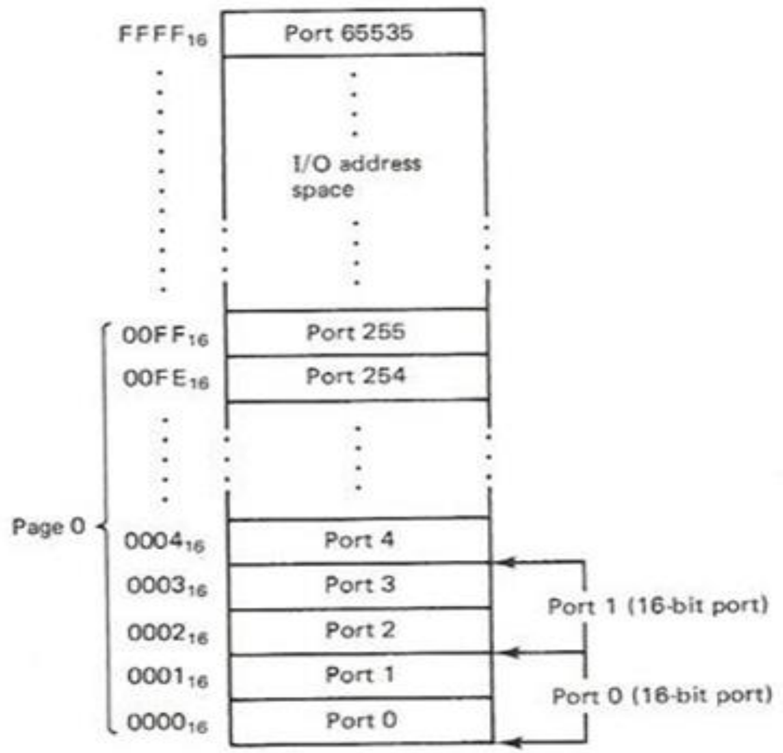
Input / Output Types

The input/output (I/O) system of the microprocessor allows peripherals to provide data or receive results of processing the data; this is done using I/O ports. Through the I/O interface, the MPU can input or output data in *bit*, *byte,* or *word*.

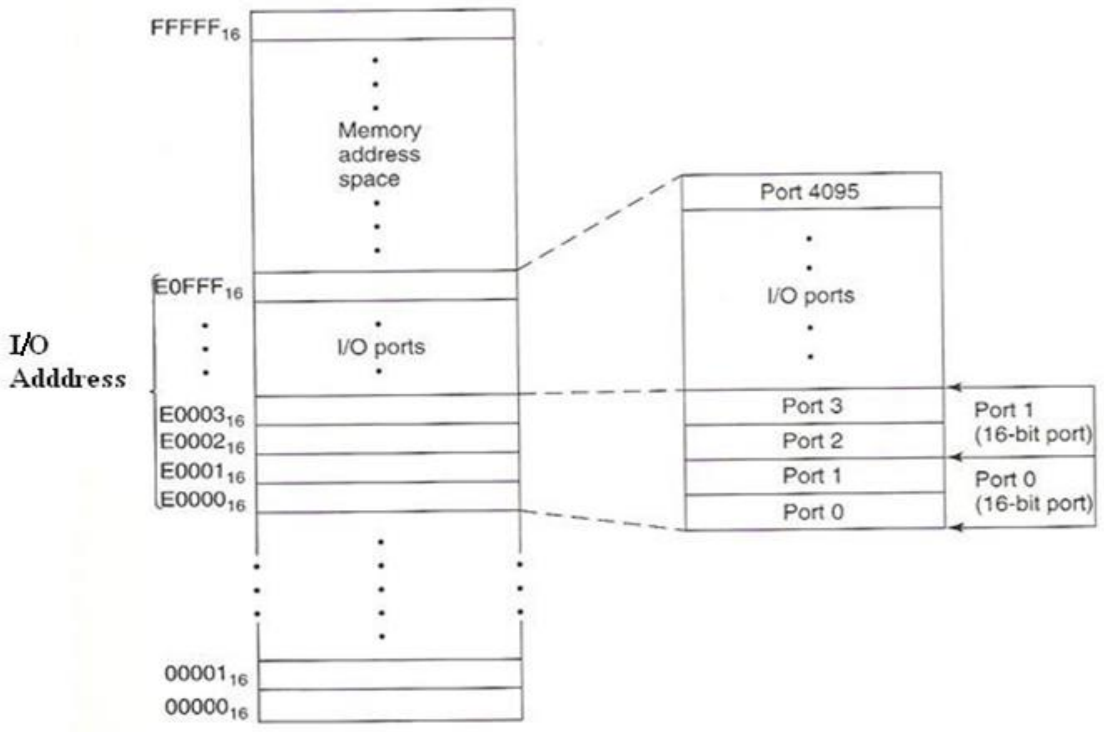
The 8086 microcomputers can employ two different types of (I/O); *Isolated I/O* and *Memory-mapped I/O*. These I/O methods differ in how I/O ports are mapped into the 8086's address spaces.

- ***Isolated Input/Output*** the I/O devices are treated separate from memory. The address space from a software point of view for the I/O ports is organized as bytes of data in the range 0000H through FFFFH. The part of the I/O address space from address 0000H through 00FFH is referred to as *Page 0*.
- ***Memory-Mapped Input/Output*** the I/O devices are placed in the memory address space of the microcomputer. The MPU looks at the I/O port as though it is a storage location in memory. For example, in the next figure the 4096 memory addresses in the range form E0000H through E0FFFH are assigned to I/O devices.

Input / Output Types



Isolated I/O Ports



Memory-Mapped I/O Ports

Input / Output Types

Isolated I/O	Memory-mapped I/O
Use only the special input/output instructions (IN and OUT)	All memory instructions and addressing modes are available to perform I/O operation (MOV, AND, XCHG, SUB…)
Faster because I/O instructions is specifically designed to run faster than memory instructions	Slower because memory instructions execute slower than the special I/O instructions
The memory address space is not affected	Part of the memory address space is lost
All data transfers must take place between AL or AX register and the I/O port	Data transfers can take place between an I/O port and an internal register other than just AL or AX

Input / Output Instructions

Isolated Input/output operations are performed using special input and output instructions (IN and OUT). All data transfers take place between an I/O device and the accumulator (byte transfers involve AL, and word transfers involve AX).

Mnemonic	Meaning	Format	Operation
IN	Input direct	IN Acc, Port	(Acc) $\leftarrow$ (Port)
	Input indirect (variable)	IN Acc, DX	(Acc) $\leftarrow$ ((DX))
OUT	Output direct	OUT Port, Acc	(Port) $\leftarrow$ (Acc)
	Ouput indirect (variable)	OUT DX, Acc	((DX)) $\leftarrow$ (Acc)

Input / Output Instructions

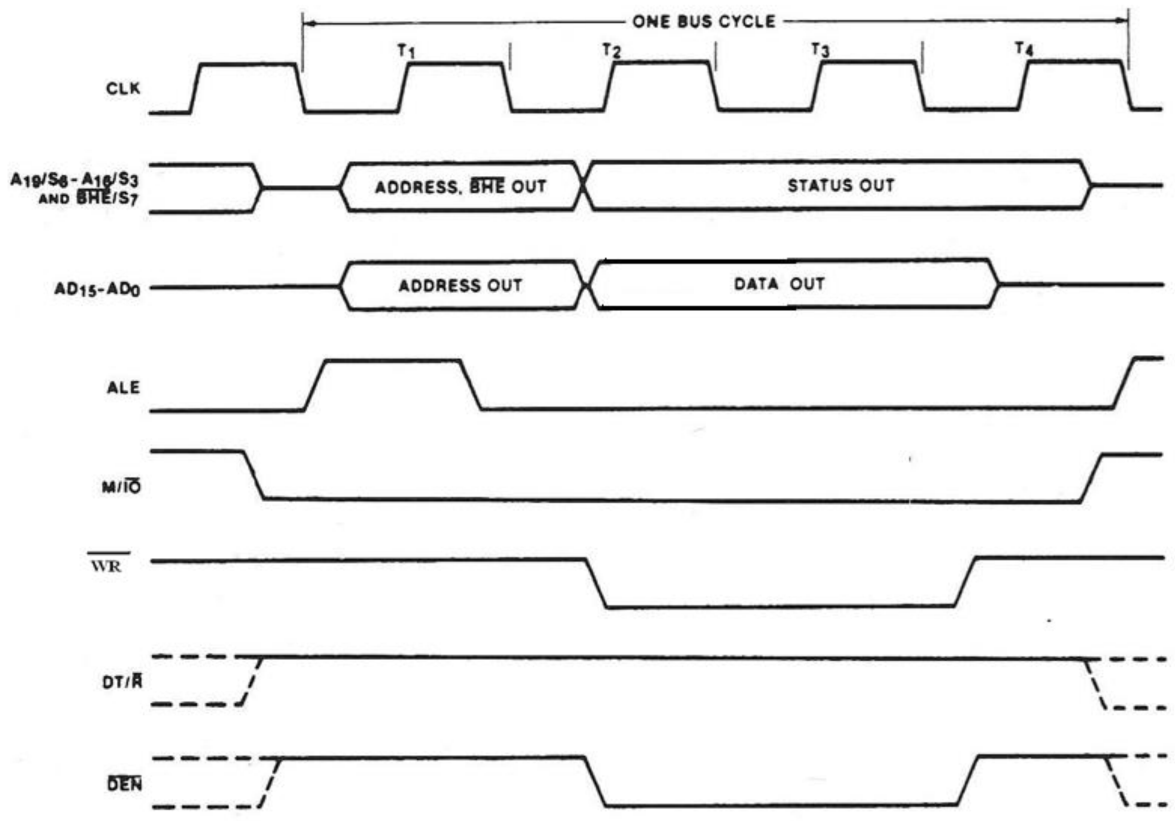
There are two different forms of IN and OUT instructions; the *Direct* I/O instructions and *Variable* I/O instructions.

- ***Direct I/O instructions*** The address of the I/O port is specified as part of the instruction. Eight bits are provided for this direct address. For this reason, its value is limited to the address range from 00H to FFH (Page 0).

- ***Variable I/O instructions*** use a 16-bit address that resides in the DX register within the MPU (the value in DX is not an offset). Since the address is 16 bits in length, variable I/O instructions can access ports located anywhere in the 64 KB I/O address space.

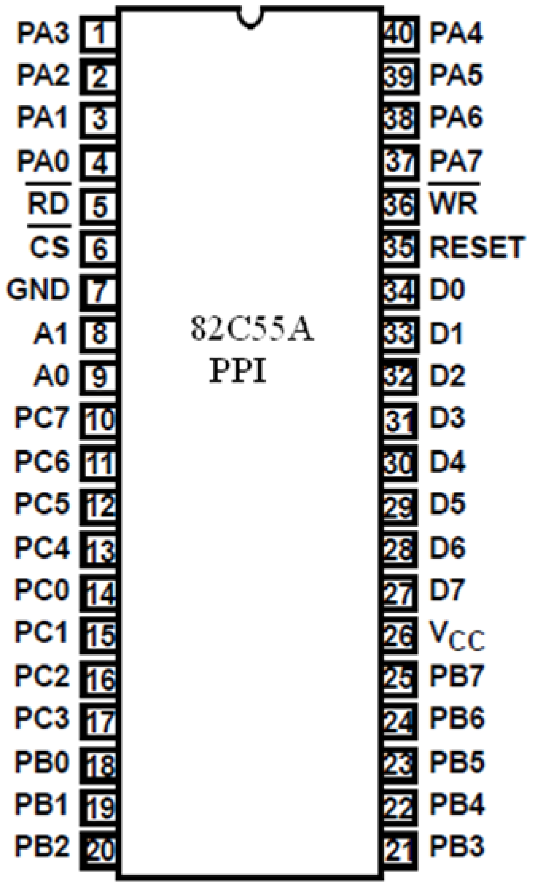
Input / Output Bus cycles

The input/output bus cycles are essentially the same as those involved in the memory interface. This figure shows the output bus cycle of the 8086. It's similar to the write cycle except for the signal M/$\overline{\text{IO}}$.



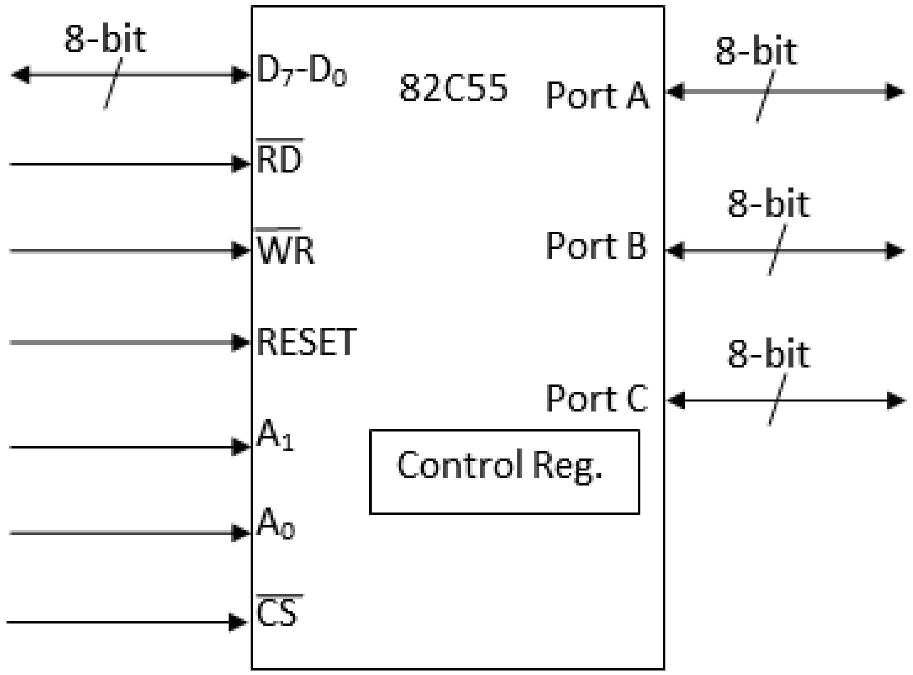
82C55A Programmable Peripheral Interface

The 82C55A interfaces peripheral I/O devices to the microcomputer system. It is programmable by the system software. It reduces the external logic normally needed to interface peripheral devices and supports a variety of byte oriented I/O interfaces (printers, keyboards, D-to-A and A-to-D converters, etc.). It is manufactured using CMOS technology, and the circuitry on its chips is equivalent to approximately 5200 transistors. The standard configuration of the 82C55A makes it compatible with the 8086, 8088 and other microprocessors.



82C55A Programmable Peripheral Interface

- 82C55A consists of 3 (8-bit) I/O ports (A, B, and C). They are grouped as group A (Port A with the 4 upper bits of port C) and group B (Port B with the 4 lower bits of port C).
- D_0-D_7 are bidirectional three-state data bus lines.
- $\overline{RD}$ is an active low control signal used by the CPU to read status information or data via the data bus.
- $\overline{WR}$ is an active low control signal used by the the
- CPU to load control words and data into the 82C55A.

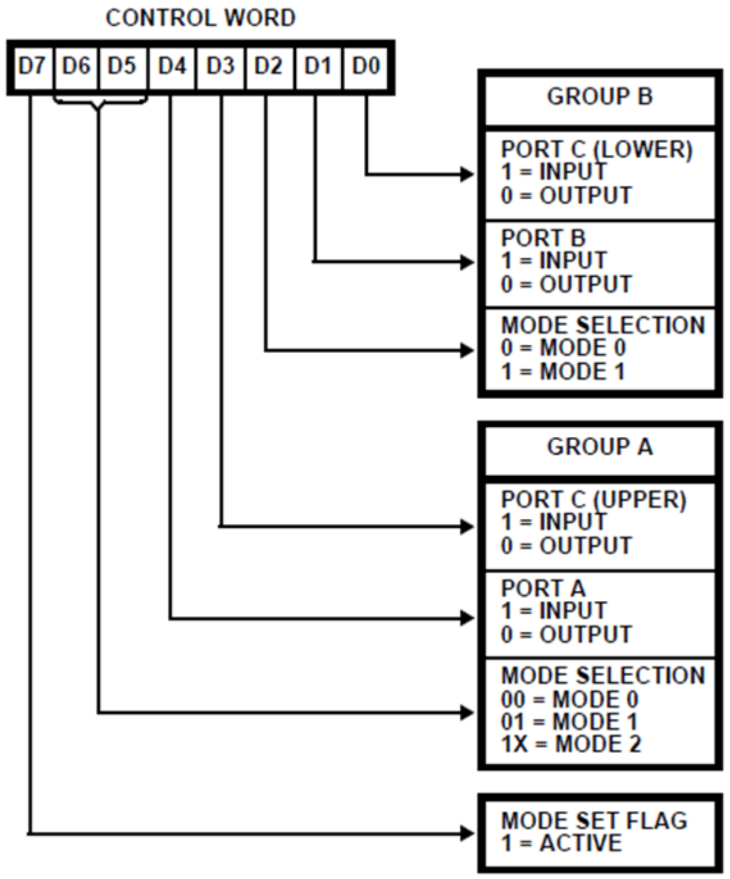


82C55A Programmable Peripheral Interface

- The RESET signal (active high) clears the control register and all ports (A, B, and C) are set to the input mode.
- Chip select $(\overline{CS})$ is an active low input used to enable the 82C55A for communications.
- A_0 and A_1 signals control the selection of one of the three ports or the control word register.

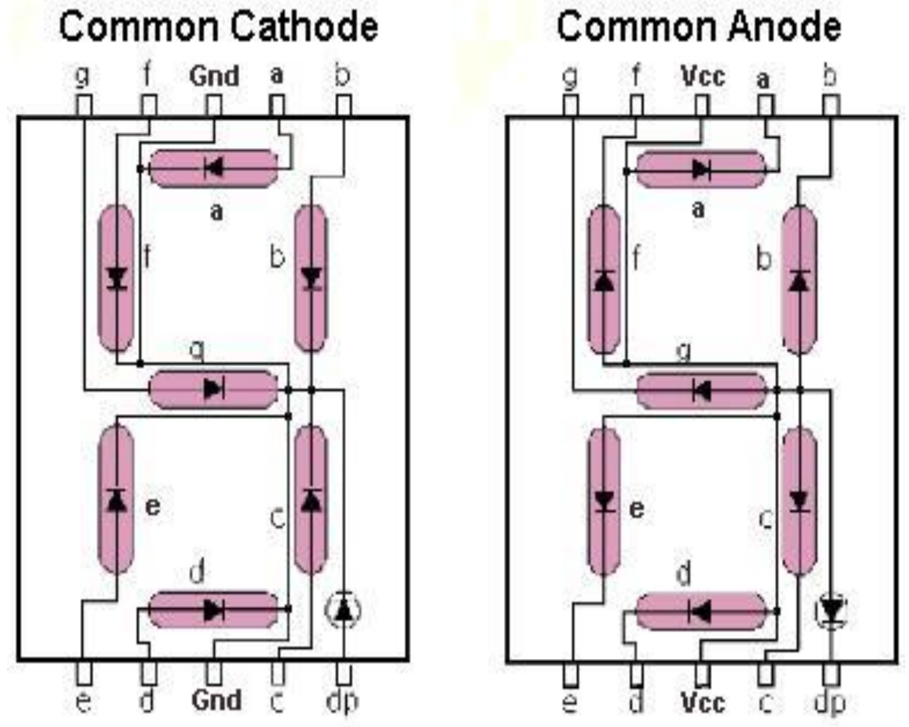
A_1A_0	82C55A Register
00	A
01	B
10	C
11	Control register

Control Word of the 82C55 PPI



Seven-Segment Display Interface

A seven-segment display is a form of electronic display device for displaying decimal numerals. It is widely used in digital clocks, electronic meters, and other electronic devices for displaying numerical information.



Seven-Segment Display Interface

There are two types of LED seven-segment displays: common cathode (CC) and common anode (CA). The difference between the two displays is the common cathode has all the cathodes of the seven-segments connected directly together and the common anode has all the anodes of the seven-segments connected together.

When working with a CA seven segment display, power must be applied externally to the anode connection that is common to all the segments. Then by applying a ground to a particular segment connection (a-g), the appropriate segment will light up.

A common cathode seven-segment is different from a common anode segment in that the cathodes of all the LEDs are connected together. For the use of this seven-segment the common cathode connection must be grounded and power must be applied to appropriate segment in order to illuminate that segment.

Memory Interface Circuits

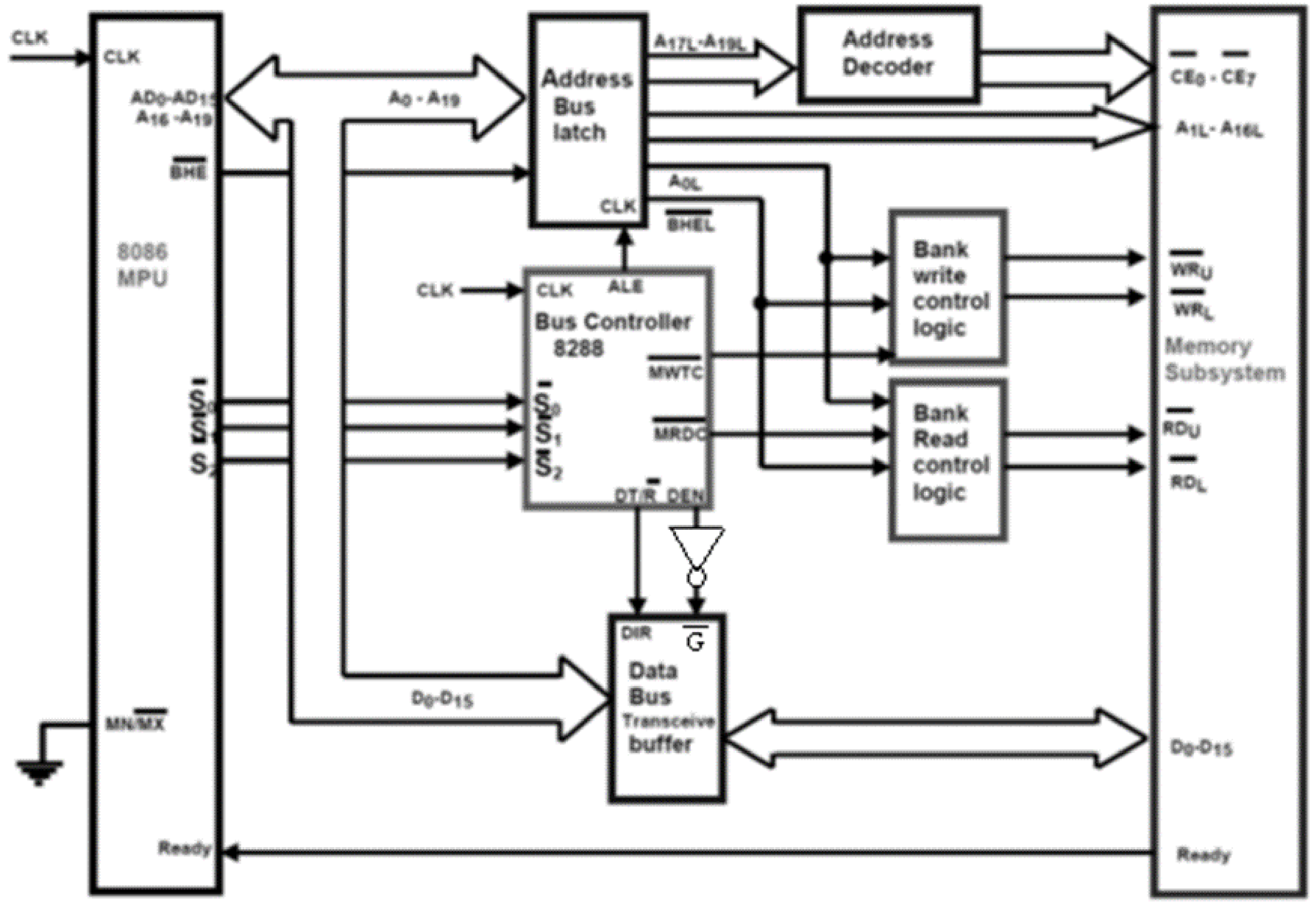
The memory interface circuit of an 8086-based (maximum mode) microcomputer system is shown below.

We find that the interface includes:

- 8288 bus controller
- Address bus latches
- Address decoders
- Bank write and read control logic
- Data bus transceiver/buffer

The status signals $\bar{S}_0$,$\bar{S}_1$, and $\bar{S}_2$are outputs of the 8086 and inputs to the 8288 bus controller which decodes them to produce command and control signals needed to coordinate data transfer over the bus.

Memory Interface Circuits (Maximum Mode)



Memory Interface Circuits (Maximum Mode)

The address bus is latched and decoded. The address lines A_0 through A_{19} are latched along with control signal $\overline{BHE}$ in the address bus latch. The latched address lines A_{17L} through A_{19L} are decoded to produce chip enable output $\overline{CE}_0$ through $\overline{CE}_7$.

The 8288 bus controller produces the address latch enable *ALE* control signal from $\bar{S}_2\ \bar{S}_1\bar{S}_0$. It is applied to the *CLK* input of latches.

During read operations, the bank read control logic determines whether the data are read from one of the two memory banks or from both (depending on whether a byte or word data transfer). Similarly, the bank write control logic determines to which memory bank the data are written.

Memory Interface Circuits (Maximum Mode)

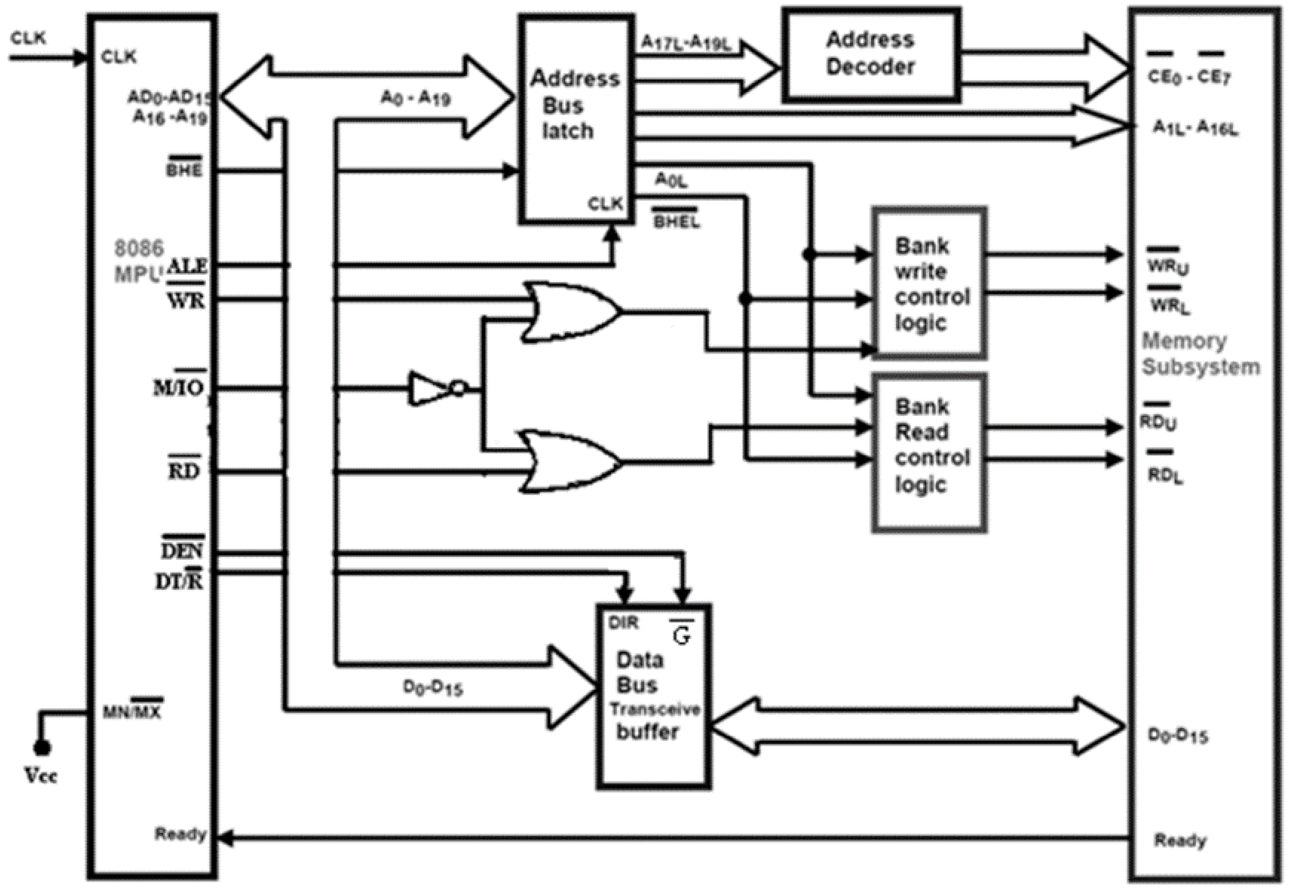
The bus transceivers control the direction of data transfer between the microprocessor and memory subsystem.

DEN is applied to the *EN* input of the transceivers to enable them for operation.

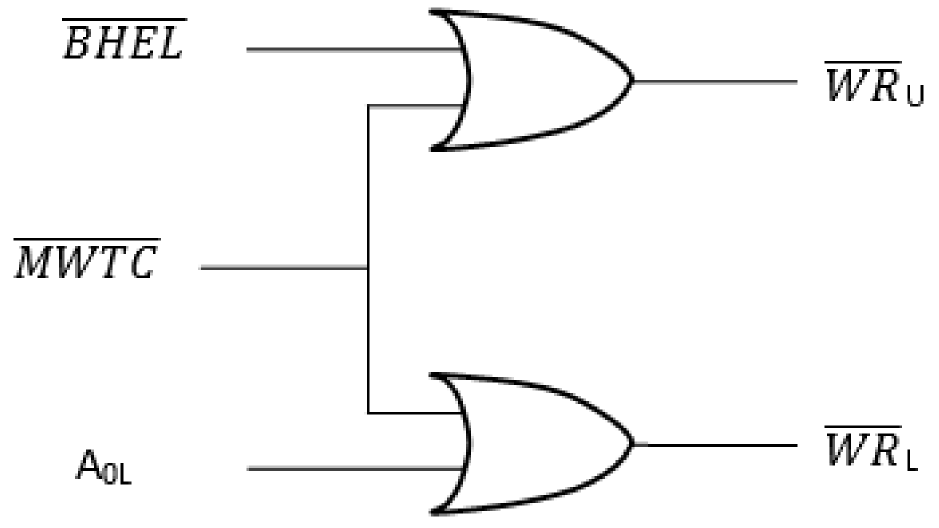
$DT/\bar{R}$ is applied to the *DIR* input of the bus transceivers to select the direction of data transfer (logic 0 makes data pass from memory to microprocessor, logic 1 makes data be carried from the microprocessor to the memory).

For the minimum mode, the memory interface is similar to above except that the signals $ALE, \overline{DEN}$, and $DT/\bar{R}$ are delivered by 8086 directly. $\overline{MWTC}$ and $\overline{MRDC}$ are produced using $\overline{RD}$, $\overline{WR}$, and $M/\overline{IO}$ signals as shown below.

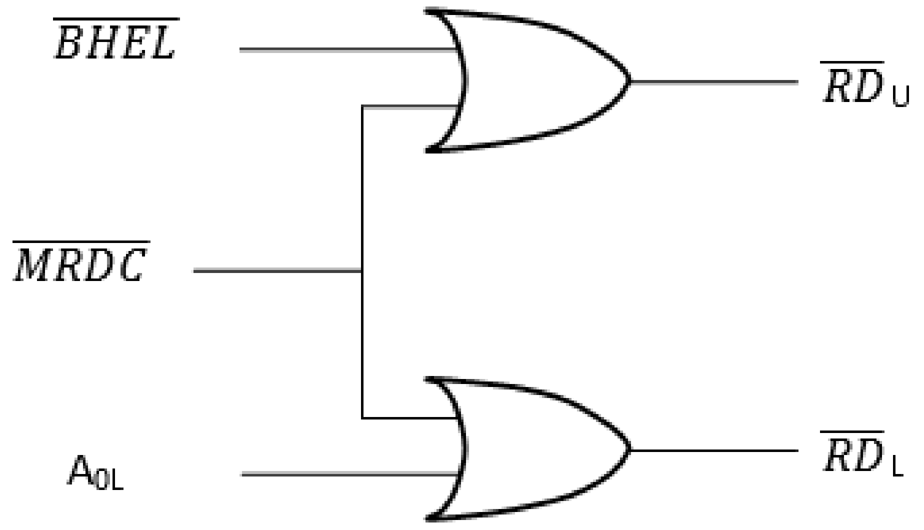
Memory Interface Circuits (Minimum Mode)



Bank Write and Bank Read Control Logic

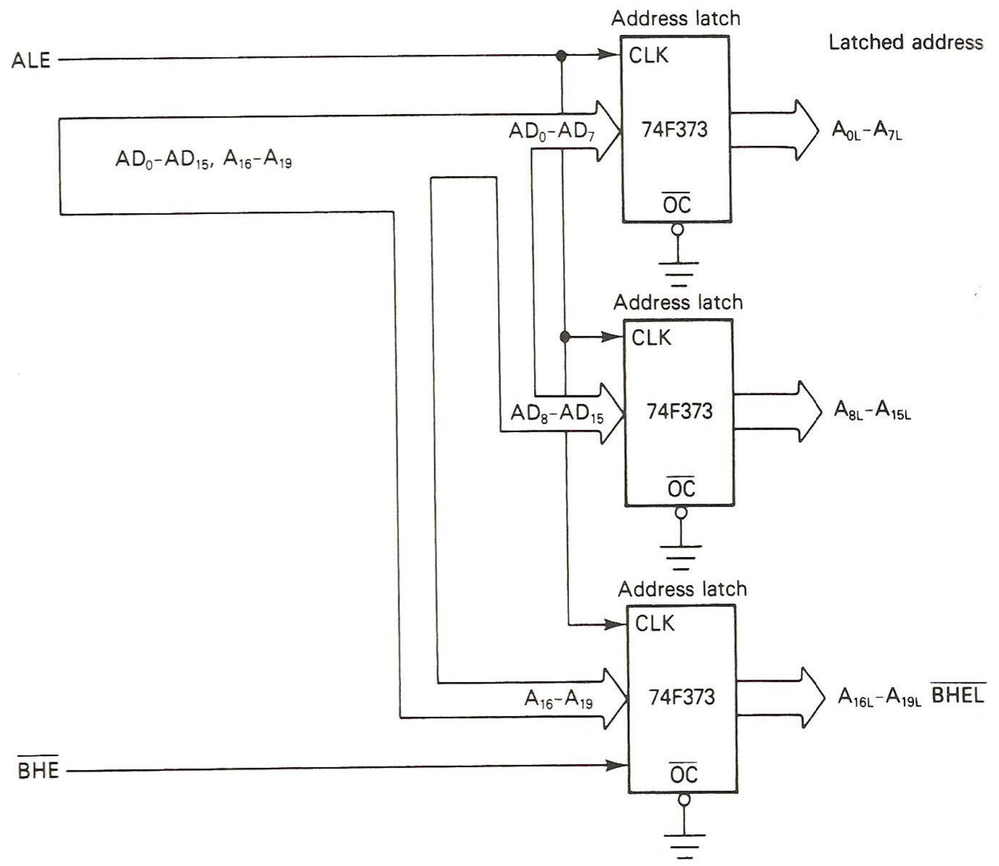


Bank Write Control Logic



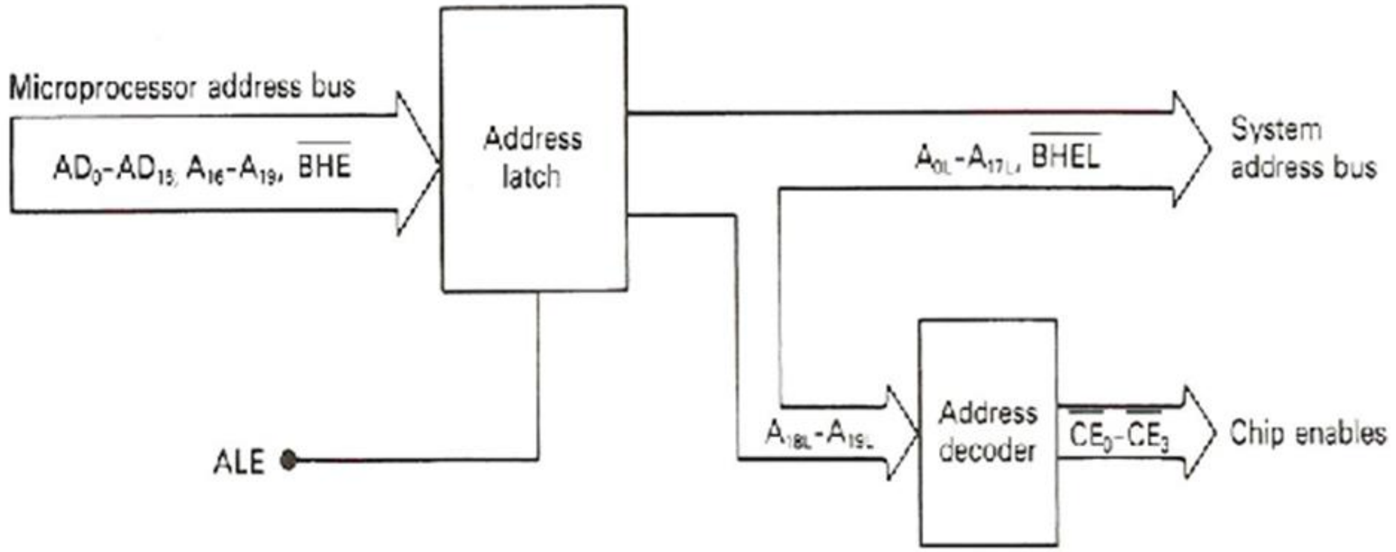
Bank Read Control Logic

Address Bus Latches



Address Decoders

The address decoder in the 8086 microcomputer system is located at the output side of the address latch.


Microprocessor address bus
AD_0-AD_{15}, A_{16}-A_{19}, $\overline{BHE}$
Address latch
A_{0L}-A_{17L}, $\overline{BHEL}$
System address bus
ALE
A_{18L}-A_{19L}
Address decoder
$\overline{CE}_0$-$\overline{CE}_3$
Chip enables

The 74LS138 decoder can be used to perform the address decode function. The circuit in above uses the 74F13 9to generate chip enable signals $\overline{CE}_0$ through $\overline{CE}_3$ by decoding address lines A_{18L}, and A_{19L}.

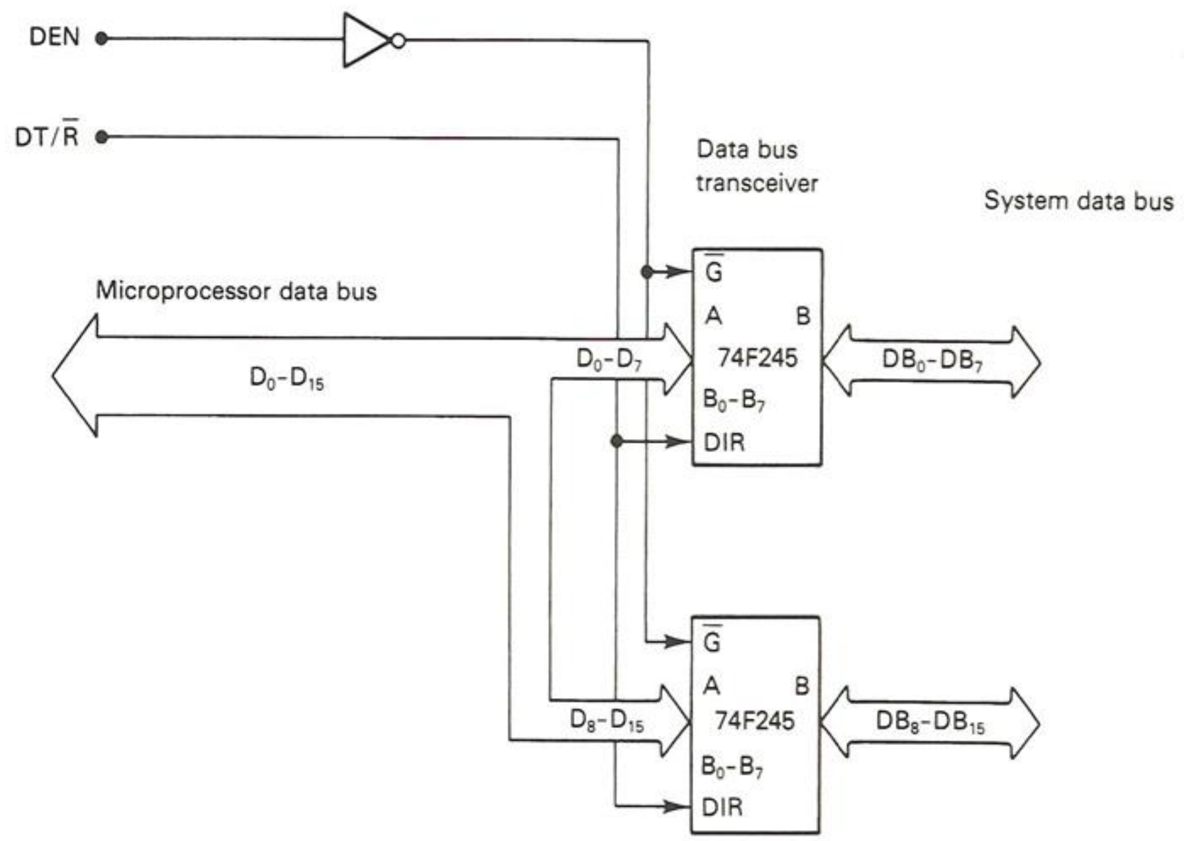
Data Bus Transceivers

The data bus transceiver block of the bus interface circuit can be implemented with 74LS245 octal bus transceiver ICs. The $\bar{G}$input is used to enable the buffer for the operation. On the other hand, DIR input is used to select the direction in which data are transferred through the device (if *DIR* = 0 the data pass from B lines to A lines, else if *DIR* =1 data pass from A lines to B lines).

Figure below shows a circuit that implements the data bus transceiver block of the bus interface circuit using the 74LS245. For the 16-bit data bus of the 8086 microcomputer, two devices are required.

Here the *DIR* input is driven by the signal *DT/*$\bar{R}$, and $\bar{G}$ is supplied by data bus enable *DEN* (from the bus controller 8288 in the maximum mode) or by $\overline{DEN}$ (form 8086 in the minimum mode).

Data Bus Transceivers



Memory Types

Every microprocessor-based system has a memory system. Memory provides the ability to store and retrieve digital information. The memory unit of the microcomputer is partitioned into a primary storage section and secondary storage section.

Primary Storage Memory	Secondary Storage Memory
Used for working information, such as the instruction of the program currently being run and data that it is processing	Used for storage of data, information, and programs that are not in use
This part normally requires high speed operation but does not normally require very large storage capacity	This part of the memory unit can be slow speed, but it requires very large storage capacity
It is implemented with semiconductor memory devices, such as ROM , RAM and FLASH	It is normally implemented with magnetic storage device, such as the hard disk drive

Memory Types

Almost all systems contain two main types of memory: Read-Only Memory ROM and Random Access Memory RAM. ROM contains system software and permanent system data, while RAM contains temporary data and application software.

- ***Read Only Memory*** is one type of semiconductor memory device. It is most widely used in microcomputer systems for storage of the program that determines overall system operation. The information stored within a ROM integrated circuit is permanent (non-volatile).

  Three types of ROM devices are in wide use:

- The mask programmable read only memory (ROM)
- The one time programmable read only memory (PROM)
- The erasable programmable read only memory (EPROM)

Memory Types

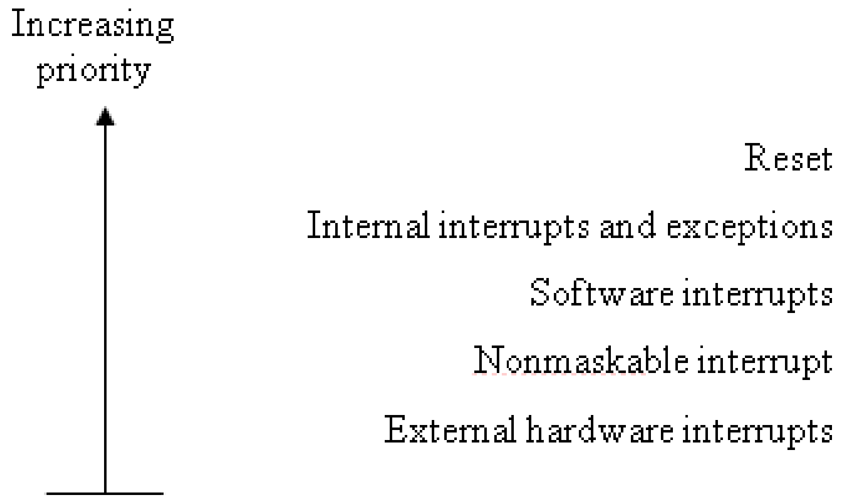
- ***Random Access Memory*** RAM is similar to ROM in that its storage location can be accessed in a random order, but it is different from ROM in that data stored in RAM is not permanent (volatile).

  Two types of RAMs are in wide use today:

- *Static RAM (SRAM)* data remain valid as long as the power supply is not turned off.
- *Dynamic RAM (DRAM)* to retain data in a DRAM, it is not sufficient just to maintain the power supply; we must periodically restore the data in each storage location (Refreshing the DRAM).

Interrupt Mechanism, Types, and Priority

Interrupts provide a mechanism for quickly changing program environment. Transfer of program control is initiated by the occurrence of either an event internal to the microprocessor or an event in its external hardware. The 8086 microcomputers are capable of implementing any combination of up to 256 interrupts. They are divided into five groups; *external hardware interrupts*, *Nonmaskable interrupt*, *Software interrupts*, *Internal interrupts*, and *Reset*. Their priority hierarchy is shown below.



Interrupt Mechanism, Types, and Priority

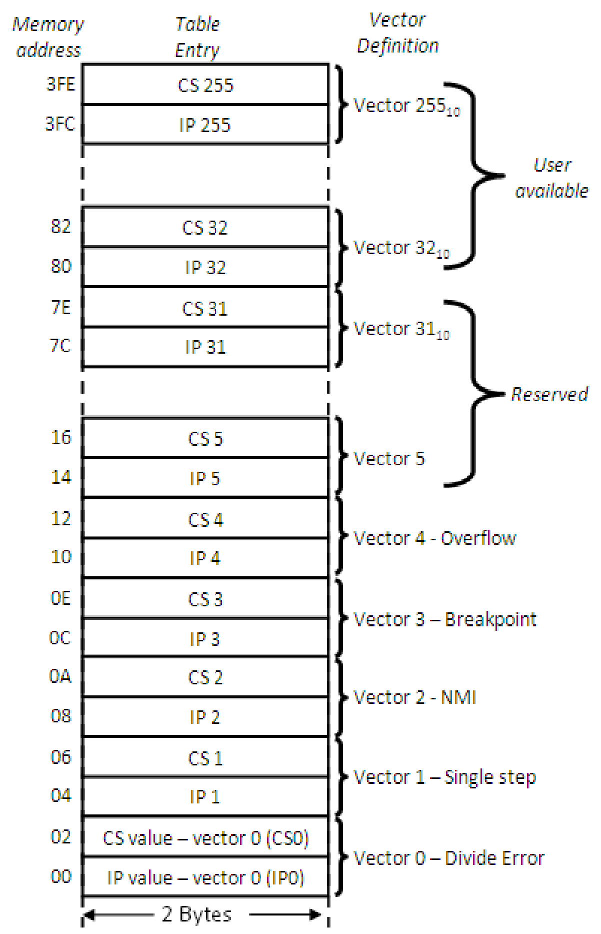
The user defines the function of the external hardware, software, and nonmaskable interrupt. For instance, hardware interrupts are often assigned to devices such as the keyboard, printer, and timers. On the other hand, the functions of the internal interrupts and reset are not user defined. They perform dedicated system functions.

An example of a high-priority service routine that should not be interrupted is that for a power failure. Once initiated, this routine should be quickly run to completion to assure that the microcomputer goes through an orderly power-down.

Interrupt Vector Table

An address pointer table is used to link the interrupt type numbers to the locations of their service routines in the program-storage memory. Figure below shows a map of the pointer table in the memory of the 8086 microcomputer.

Interrupt Vector Table



Interrupt Vector Table

The interrupt vector table contains 256 address pointers (vectors). Which are identified as vector 0 through vector 255. That is, one pointer corresponds to each of the interrupt types 0 through 255. These address pointers identify the starting location of their service routines in program memory.

The pointer table is located at the low-address end of the memory address space. It starts at address 00000H and ends at 003FEH. This represents the first 1Kbytes of the memory.

Each of the 256 pointers requires two words (4 bytes) of memory and is always stored at an even-address boundary.

The first 31 pointers either have dedicated functions or are reserved. The 27 reserved pointers, 5 through 31, represent a reserved portion of the pointer table and should not be used. The remainder of the table, the 224 pointers in the address range 00080H through 003FFH, is available to the user for storage of software or hardware interrupt vectors.

Interrupt Types

❖ ***Software Interrupts***

The 8086 microcomputer system is capable of implementing software interrupts their service routines are initiated (without interrupt acknowledgments bus cycles) in response to the execution of a software interrupt instruction (not external hardware). Software interrupts are not masked out by IF.

❖ ***Interrupt Instructions***

A number of instructions are provided in the instruction set of the 8086 microprocessors for use with interrupt processing.

Interrupt Types

Mnemonic	Meaning	Format	Operation	Flags affected
CLI	Clear interrupt flag	CLI	$0 \longrightarrow (IF)$	IF
STI	Set interrupt flag	STI	$1 \longrightarrow (IF)$	IF
INT n	Type n software interrupt	INT n	$(Flags) \longrightarrow ((SP)-2)$ $0 \longrightarrow TF,IF$ $(CS) \longrightarrow ((SP)-4)$ $(2+4*n) \longrightarrow CS$ $(IP) \longrightarrow ((SP)-6)$ $(4*n) \longrightarrow (IP)$	TF, IF
IRET	Interrupt return	IRET	$((SP)) \longrightarrow (IP)$ $((SP)+2) \longrightarrow (CS)$ $((SP)+4) \longrightarrow (Flags)$ $(SP)+6 \longrightarrow (SP)$	All
INTO	Interrupt on overflow	INTO	INT 4 steps	TF , IF
HLT	Halt	HLT	Wait for an external interrupt or reset to occur	None
WAIT	Wait	Wait	Wait for $\overline{\text{TEST}}$ input to go active	None

Interrupt Types

- ***CLI Instruction*** permits manipulation of interrupt flag. It disables the external interrupt input (INTR) by resetting IF.
- ***STI Instruction*** also permits manipulation of interrupt flag. It enables the external interrupt input (INTR) for operation-that is, it sets IF.
- ***INT n Instruction*** used to initiate a vectored call of a subroutine. It causes program control to be transferred to the subroutine pointed to by the vector for the number n specified in the instruction.

For example, execution of the instruction INT 50 initiates execution of a subroutine whose starting point is identified by vector 50 in the pointer table. First the MPU saves the old flags on the stack, clears TF and IF, and saves the old program context (CS and IP) on the stack. Then it reads the values of IP_{50} and CS_{50} from addresses 000C8H and 000CAH, respectively, in memory, loads them into the IP and CS registers, calculates the physical address CS_{50}:IP_{50}, and starts to fetch instruction from this new location in program memory.

Interrupt Types

- ***IRET Instruction*** must be included at the end of each interrupt service routine. It causes the old values of IP, CS, and flags to be popped from the stack back into the internal register of the MPU.

- ***INTO Instruction*** must be included after arithmetic instructions that can result in an overflow condition, such as divide. It tests the overflow flag, and if the flag is found to be set, a type 4 internal interrupt is initiated.

- ***HALT Instruction*** makes the MPU suspends operation and enters the idle state.

- ***WAIT Instruction*** makes the MPU checks the logic level of the $\overline{\text{TEST}}$ input prior to going into the idle state. Only if it is logic 1 then the MPU will go into the idle state.

Interrupt Types

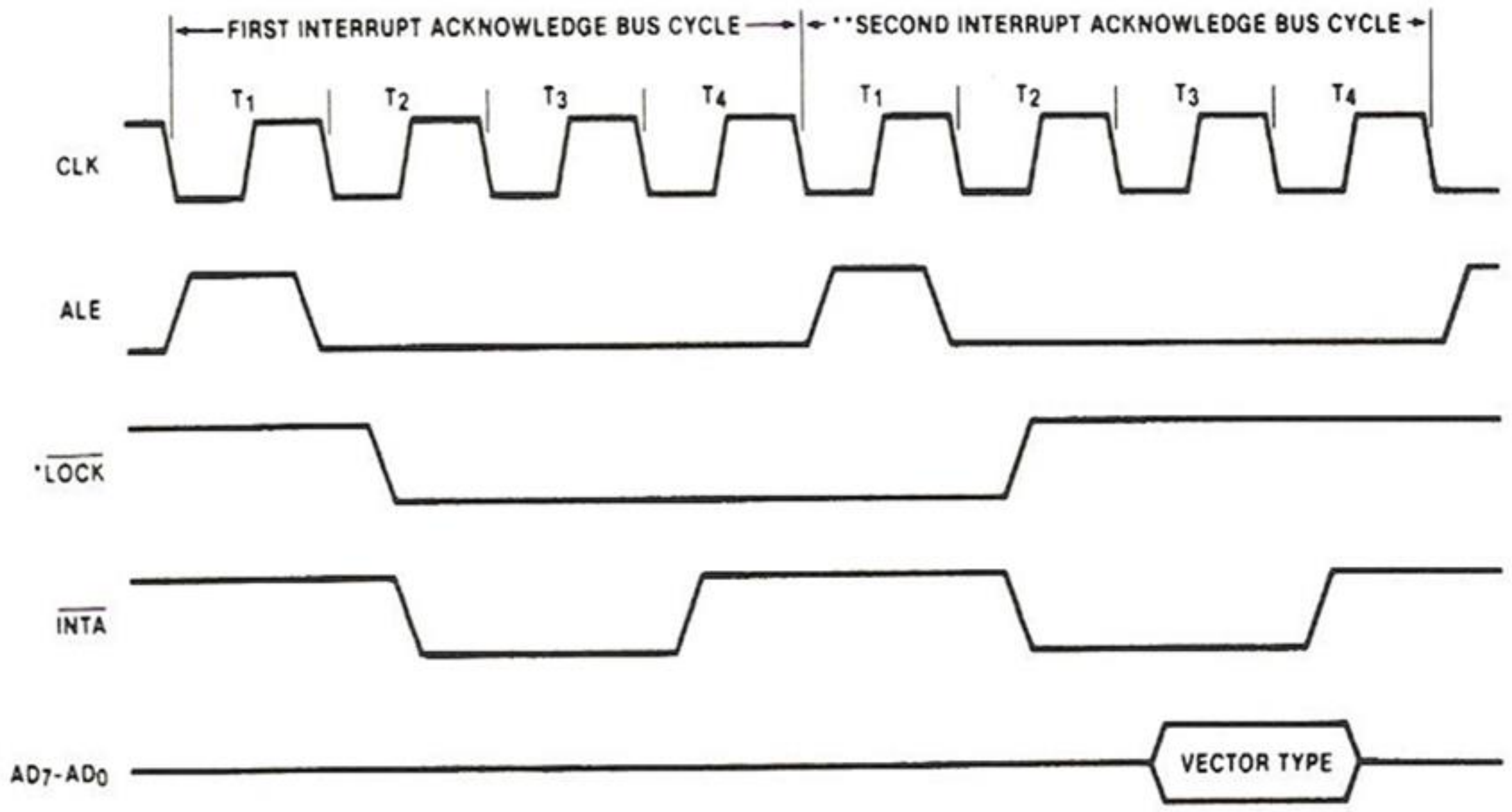
❖ ***External Hardware-Interrupt Interface Sequence***

The interrupt interface is a special input interface. When an interrupt request has been recognized on the ***NMI*** pin, the 8086 initiate type 2 interrupt (CS_2:IP_2). The NMI cannot be masked out with IF. Its input is positive edge triggered. Therefore, a request for service is automatically latched internal to the MPU.

If the ***INTR*** pin is logic 1, a request for service is recognized. The 8086 checks the IF if it = 0 then the interrupt request is ignored and the next sequential instruction is executed. If IF = 1 then the service routine is initiated and the 8086:

- saves the flag register on the stack
- saves the old program context on the stack
- clears TF and IF
- responds with two pulses at $\overline{\text{INTA}}$ during interrupt acknowledge bus cycle

Interrupt Types



FIRST INTERRUPT ACKNOWLEDGE BUS CYCLE
**SECOND INTERRUPT ACKNOWLEDGE BUS CYCLE
T1
T2
T3
T4
T1
T2
T3
T4
CLK
ALE
*LOCK
INTA
AD7-AD0
VECTOR TYPE

Interrupt Types

During T_1 of the first bus cycle, a pulse is output on ALE along with putting the address/data bus on high impedance state and for the rest of the bus cycle. During T2 and T3, $\overline{\text{INTA}}$ is switched to logic 0. This signals external circuitry that the request is granted and the logic 1 at INTR can be removed.

The $\overline{\text{LOCK}}$ signal is produced only in the maximum mode. It is used to lock other devices off the system bus, ensuring that the interrupt acknowledge sequence continues to completion without interruption.

During the second bus cycle, a similar sequence occurs. External circuitry puts the type number of the active interrupt on the data bus AD_0-AD_7. This code must be valid during periods T_3 and T_4 of the second bus cycle.

Interrupt Types

- ***Divide Error*** represents an error condition that can occur in the execution of the division instruction. If the quotient of division result is larger than the specified destination it causes automatic initiation of type 0 interrupt.
- ***Single Step*** relates to an operation option of the 8086. If the trap flag (TF) is set, the single-step mode of operation is enabled and the MPU initiates type 1 interrupt service routine at the completion of execution of every instruction of the user program.
- ***Breakpoint*** is inserted at strategic points in a program that is being debugged to cause execution to be stopped automatically. The breakpoint service routine can stop execution of the main program, permit the programmer to examine the contents of registers and memory, and allow for the resumption of execution of the program down to the next breakpoint.
- ***Overflow Error*** can result from the execution of any arithmetic instruction. The transfer of program control to a service routine is not automatic at occurrence of overflow condition. Instead, INTO instruction must be executed to test the OF, if it is found to be set, a type 4 interrupt service routine is initiated.

The 4004, 8080, 80186 Microprocessors

- ***The 4004*** was the first commercially available microprocessor produced by Intel, its clock rate is 740 KHz. It had 4 data bits multiplexed with 12 address bits. It was originally designed to be used as calculator.

- ***The 8080's*** clock rate was 2MHz. It had 8 data bits and 16 address bits so the addressable memory size is 64KB. Same is as in 8085 microprocessor but the clock rate is 3MHz and wider instruction set.

- ***The 80186's*** original clock rate was 6 MHz. It is like 8086 has 16 data bits multiplexed with 20 address bits. It included two timers, a DMA controller, and an interrupt controller on the chip in addition to the microprocessor.it was mostly used in embedded applications as controller.

The 80286 Microprocessor

- The 80286 microprocessor has 16-bit data bus and 24-bit address bus so it can address memory up to 16 MB. None of its signal lines are multiplexed with another signal. The instruction set of the 80286 is almost identical to the 8086, except for a few additional instructions that managed the extra 15M bytes of memory. The clock speed of the 80286 was increased, so it executed some instructions in as little as 250 ns with the original release 8 MHz. The 80286 was designed to run multitasking applications, including communications, real-time process control, and multi-user systems.

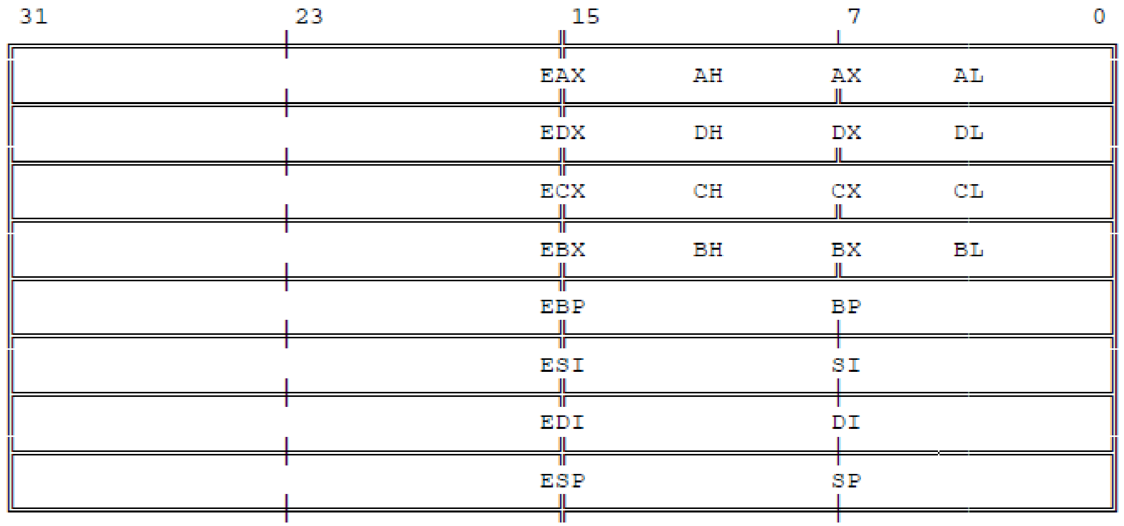
The 80386 Microprocessor

- The 80386 was Intel’s first practical 32-bit microprocessor that contained a 32-bit data bus and a 32-bit memory address that allowed it to address up to 4GB of memory. The instruction set of the 80386 was upward-compatible with the earlier 8086, 80286 microprocessors. The 80386 (and its variants) are common in aerospace technology and electronic musical instruments. Some mobile phones also used 80386 processor.

The 80386 Microprocessor

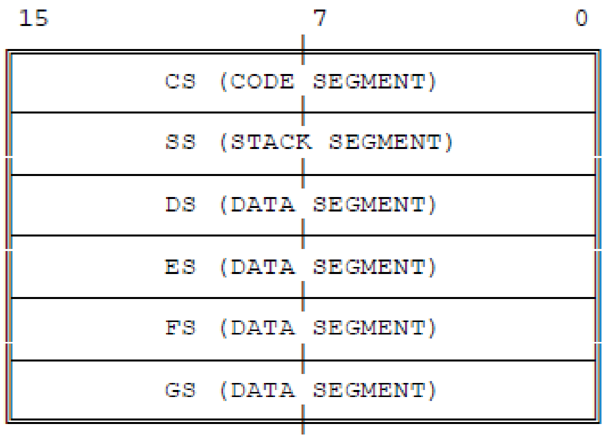
The 80386 contains a total of sixteen registers that may be grouped into three basic categories:

- ***General registers*** These eight 32-bit general-purpose registers are used primarily to contain operands for arithmetic and logical operations.

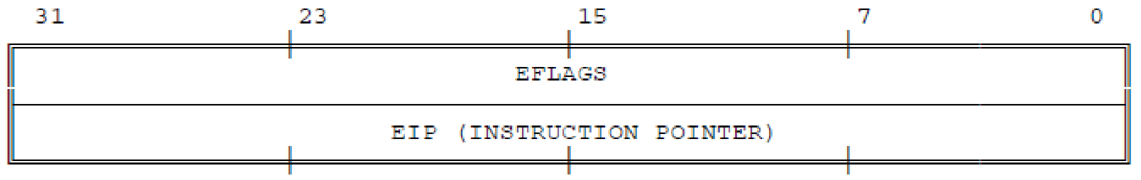


The 80386 Microprocessor

- ***Segment registers*** These special-purpose registers determine, at any given time, which segments of memory are currently addressable.



- ***Status and instruction registers*** These special-purpose registers are used to record and alter certain aspects of the 80386 processor state.



The 80386 Microprocessor

The 80386 was available in a few modified versions such as 80386CX, 80386DX, 80386EX, 80386SX, and others.

- ***80386SX*** is a low cost version of the 80386 with a 16-bit data bus to simplify circuit board layout and reduce total cost. This simplified designs but hampered performance. Only 24 pins were connected to the address bus, therefore limiting addressing to 16 MB.
- ***80386SL*** was introduced as a power efficient version for laptop computers. The processor offered several power management options, as well as different "sleep" modes to conserve battery power. It also contained support for an external cache of 16 to 64 KB.

The 80486 Microprocessor

- The 80486 family is the Intel’s second generation of 32-bit microprocessors. The 80486 has a 32-bit data bus and a 32-bit address bus. The 32-bit address bus of the 80486 enabled up to 4 GB of memory to be directly addressed. An 8 KB on-chip cache stores the most recently used instructions and data.

  The internal architecture of the 80486 was modified so that about half of its instructions executed in one clock instead of two clocks. Also the 80486 was available in different versions.

The Pentium Microprocessors

- The Pentium introduced in 1993, the Pentium processor’s 32-bit architecture is enhanced with a 64-bit external data bus and a variety of internal data paths that are 64-bits, 128-bits, or 256-bits wide. It is able to address 4 GB and has 16 KB cache.

  The Pentium processors employ an advanced *superscaler* pipelined internal architecture which gives the Pentium processors the ability to execute more than one instruction per clock cycle by simultaneously dispatching multiple instructions to different execution units .

  Another feature that enhances performance is a *jump prediction* technology that speeds the execution of programs that include jump.

Intel Microprocessors Modes

- ***Real Mode*** The 80286 and above operate in either the *real* or *protected* mode. Only the 8086 operate exclusively in the real mode. Real mode operation allows the microprocessor to address only the first 1M byte of the memory space. The DOS operating system requires the microprocessor to operate in the real mode. Real mode operation allows application software written for the 8086, which contain only 1M byte of memory, to function in the 80286 and above without changing the software. In all cases, each of these microprocessors begins operation in the real mode by default whenever power is applied or microprocessor is reset.
- ***Protected Mode*** When configured for protected-mode the microprocessor provides more instruction and advanced software architecture (like memory management, paging and multitasking for 80386).
- ***Virtual 8086 Mode*** This special mode is designed so that multiple 8086 real-mode software applications can execute at one time. The virtual 8086 mode can be used to share one microprocessor with many users by portioning the memory so that each user has its own DOS partition.

Intel Microprocessors Modes

When in this mode, the 80386DX (and above) support an 8086 microprocessor programming model and can directly run programs written for the 8086. That is, it creates a virtual 8086 machine for executing the program.

The Core and Xeon Microprocessors

- A multi-core processor is a single computing component with two or more independent processing units. The processor can run multiple instructions on separate cores at the same time increasing overall speed of the program execution. Multi core processors are widely used across many application domains, including general purpose, network, digital signal processing, and graphics.

- Xeon microprocessors have some advanced features such as support for error correcting code memory (ECC memory) that can detect and correct internal data corruption, larger amount of RAMs, larger cache memory.

Microprocessors and Microcontrollers

Microprocessor	Microcontroller
Hardware architecture: Microprocessor is a single-chip CPU. It requires external circuitry to implement memory and input/output components.	Hardware architecture: Microcontrollers contain, in a single IC, a CPU, RAM, ROM, a serial interface, a parallel interface, timer, and interrupt scheduling circuitry.
Instruction set features: Instruction sets are designed to operate on large volumes of data. Their instructions operate on bytes, words and double words. Addressing modes provide access to large arrays of data, using address pointers and offset.	Instruction set features: The instructions are highly compact. They are mostly dedicated to the control of inputs and outputs. They have instructions to set and clear individual bits and perform other bit-oriented operations such as logically ANDing, ORing or XORing bits.
Applications: Microprocessors are most commonly used as the CPU in microcomputer systems. They are suited to processing information in computer systems.	Applications: Microcontrollers are found in small, minimum-component designs performing control-oriented activities. They are suited to control of I/O devices and digital signal processing.
Example: 8086, 80286 . . .	Example: 8048, 8051 . . .