BIRSA INSTITUTE OF TECHNOLOGY (TRUST)

NH-33, GETLATU, RANCHI

Department: - Electronics and Communication Engineering
Question Set: -11
Semester: - 4th
Subject: - Digital Technologies and Microprocessor
Lecturer: - Alok Kumar Singh Date:-15/04/2020

1. **How many interrupts can be implemented using 8086 µP?**

Ans. A total of 256 interrupts can be implemented using 8086 µP.

2. **Mention and tabulate the different types of interrupts that 8086 can implement.**

Ans. 8086 µP can implement seven different types of interrupts.

- NMI and INTR are external interrupts implemented via *Hardware*.
- INT n, INTO and INT3 (breakpoint instruction) are software interrupts implemented through *Program*.
- The 'divide-by-0' and 'Single-step' are interrupts *initiated by CPU*.

Table 18.1 shows the seven interrupt types implemented by 8086.

Table 18.1: The Seven different types of 8086 interrupts

Name	Initiated by:	Maskable?	Trigger	Priority	Acknowledge signal?	Vector table address–	Interrupt latency
NMI	External hardware	No	↑Edge, hold 2 T states min.	2	None	00008H–0000BH	Current instruction + 51 T states
INTR	External hardware	Yes via IF	High level until acknowledged	3	$\overline{\text{INTA}}$	n * 4[b]	Current instruction + 61 T states
INT n	Internal via software	No	None	1	None	n * 4	51 T states
INT 3 (break point)	Internal via software	No	None	1	None	0000CH–0000FH	52 T states
INTO	Internal via software	No	None	1	None	00010H–00013H	53 T states
Divide-by-0	Internal via CPU	Yes via OF	None	1	None	00000H–00003H	51 T states
Single-step	Internal via CPU	Yes via TF	None	4	None	00004H–00007H	51 T states

a. All interrupt types cause the flags, CS, and IP registers to be pushed onto the stack. In addition, the IF and TF flags are cleared.
b. n is an 8-bit type number read during the second INTA pulse.

3. **Distinguish between the two hardware interrupts of 8086.**

Ans. The distinction between the two hardware interrupts of 8086 are as follows, shown in Table 18.2.

Table18.2: Comparison of NMI and INTR interrupts

NMI	INTR
1. Non-maskable type.	1. Maskable type.
2. Higher priority.	2. Lower priority.
3. Edge triggered interrupt initiated on Low to High transition.	3. Level triggered interrupt.
4. Must remain high for more than 2 CLK cycles.	4. Sampled during last CLK cycle of each instruction.
5. The rising edge of NMI input is latched on-chip and is serviced at the end of current instruction.	5. No latching. Must stay high until acknowledged by CPU.
6. No acknowledgement.	6. Acknowledged by INTA output signal.

4. How many bytes are needed to store the starting addresses of ISS for 8086 µP?

Ans. 8086 µP can implement 256 different interrupts. To store the starting address of a single ISS (Interrupt Service Subroutine), four bytes of memory space are required—two bytes to store the value of CS and two bytes to store the IP value. Thus to store the starting address of 256 ISS, in all 256 × 4 = 1024 bytes = 1 KB will be required.

5. Indicate the number of memory spaces needed in stack when an interrupt occurs.

Ans. When an interrupt occurs, before moving over to starting address of the corresponding ISS, the following are pushed into the stack: the contents of the flag register, CS and IP. Since each one of the three are 2 bytes, hence a total of 6 bytes of memory space is needed in the stack to accommodate the flag register, CS and IP contents.

6. What are meant by interrupt pointer and interrupt pointer table?

Ans. The starting address of an ISS in the 1 KB memory space is known as the interrupt pointer or interrupt vector corresponding to that interrupt.

The 1 KB memory space needed to store the starting addresses of all the 256 ISS is called the interrupt pointer table.

7. Write down the steps, sequentially carried out by the systems when an interrupt occurs.

Ans. When an interrupt occurs (hardware or software), the following things happen:

- The contents of flags register, CS and IP are pushed on to the stack.
- TF and IF are cleared which disable single step and INTR interrupts respectively.
- Program jumps to the starting address of ISS.
- At the end of ISS, when IRET is executed in the last line, the contents of flag register, CS and IP are popped out of the stack and placed in the respective registers.
- When the flags are restored, IF and TF get back their previous values.

8. Draw and discuss the interrupt pointer table for 8086 µP.

Ans. The interrupt pointer table for 8086 is shown in Fig. 18.1.

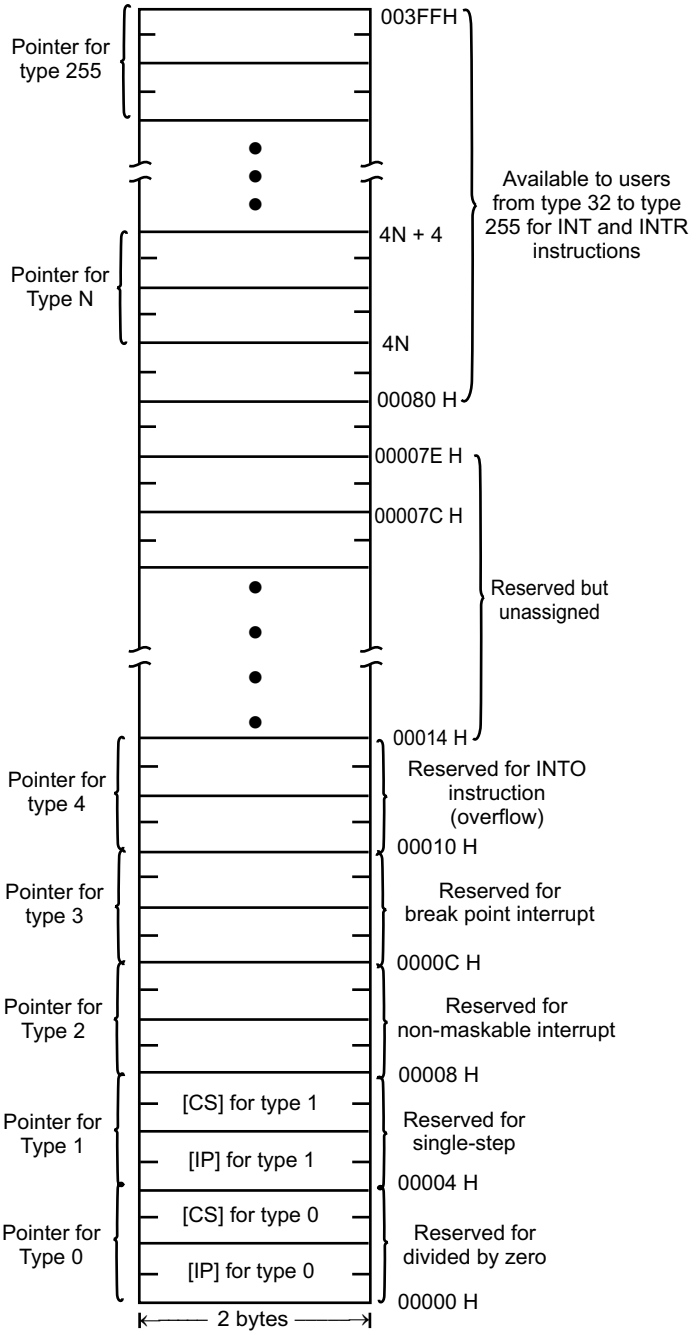

Fig. 18.1: Interrupt Pointer Table for intel 8086

The 256 interrupt pointers are stored in memory locations starting from 00000 H to 003FF H (1 KB memory space). The number assigned to an interrupt pointer is called

the Type of the corresponding interrupt. As for example, Type 0 interrupt, Type 1 interrupt ... Type 255 interrupt. Type 0 interrupt has a memory address 00000 H, Type 1 has a memory address 00004 H, while Type 255 has a memory address 003FF H. The first five pointers (Type 0 to Type 4) are dedicated pointers used for divide by zero, single step, NMI, break point and overflow interrupts respectively. The next 27 pointers (Type 5 to Type 31) are reserved pointers—reserved for some special interrupts. The remaining 224 interrupts—from Type 32 to Type 255 are available to the programmer for handling hardware and software interrupts.

9. Discuss the priority of interrupts of 8086.

Ans. 8086 tests for the occurence of interrupts in the following hierarchical sequence:

- Internal interrupts (divide-by-0, single step, break point and overflow)
- Non-maskable interrupt—via NMI
- Software interrupts—via INTn
- External hardware interrupt—via INTR

Hence, internal interrupts belong to the highest priority group and internal hardware interrupts are the lowest priority group. Again, different interrupts are given different priorities by assigning a type number corresponding to each priority—starting from Type 0 (highest priority interrupt) to Type 255 (lowest priority interrupt). Thus, Type 40 interrupt is having more priority than Type 41 interrupt. If we presume that at any instant a Type 40 interrupt is in progress, then it can be interrupted by any software interrupt, the non-maskable interrupt, all internal interrupts or any external interrupt with a Type number less than 40.

10. Outline the events that take place when 8086 processes an interrupt.

Ans. Fig. 18.2 shows the manner in which 8086 processes an interrupt while the following are the events that take place sequentially when the processor receives an interrupt (from an *external* device via INT 32 through INT 255:

- Receiving an interrupt request (from an external device)
- Generation of interrupt acknowledge bus cycles.
- Servicing the Interrupt Service Subroutine (corresponding to the external device which has interrupted the CPU.)

Again the difference between simultaneous interrupt and interrupt within an ISS is to be understood. Occurrence of more than one interrupt within the *same instruction* is called simultaneous interrupt while if an interrupt occurs while the ISS is in progress, it is an interrupt occurring within an ISS.

Internal interrupts (except single step) have priority over simultaneous external requests. For example, let the current instruction causes a divide-by-zero interrupt when an INTR (a hardware interrupt) occurs, the former will be serviced. Again, if simultaneous interrupts occur on INTR and NMI, then NMI will be serviced first.

For simultaneous interrupts occurring, the priority structure of Fig.18.2 will be honoured, with the highest priority interrupt being serviced first.

Although it has been commented that software interrupts get priority over external hardware interrupts, even then if an interrupt on NMI occurs as soon as the interrupt's ISS begins, it (i.e., NMI) will be recognised and hence serviced.

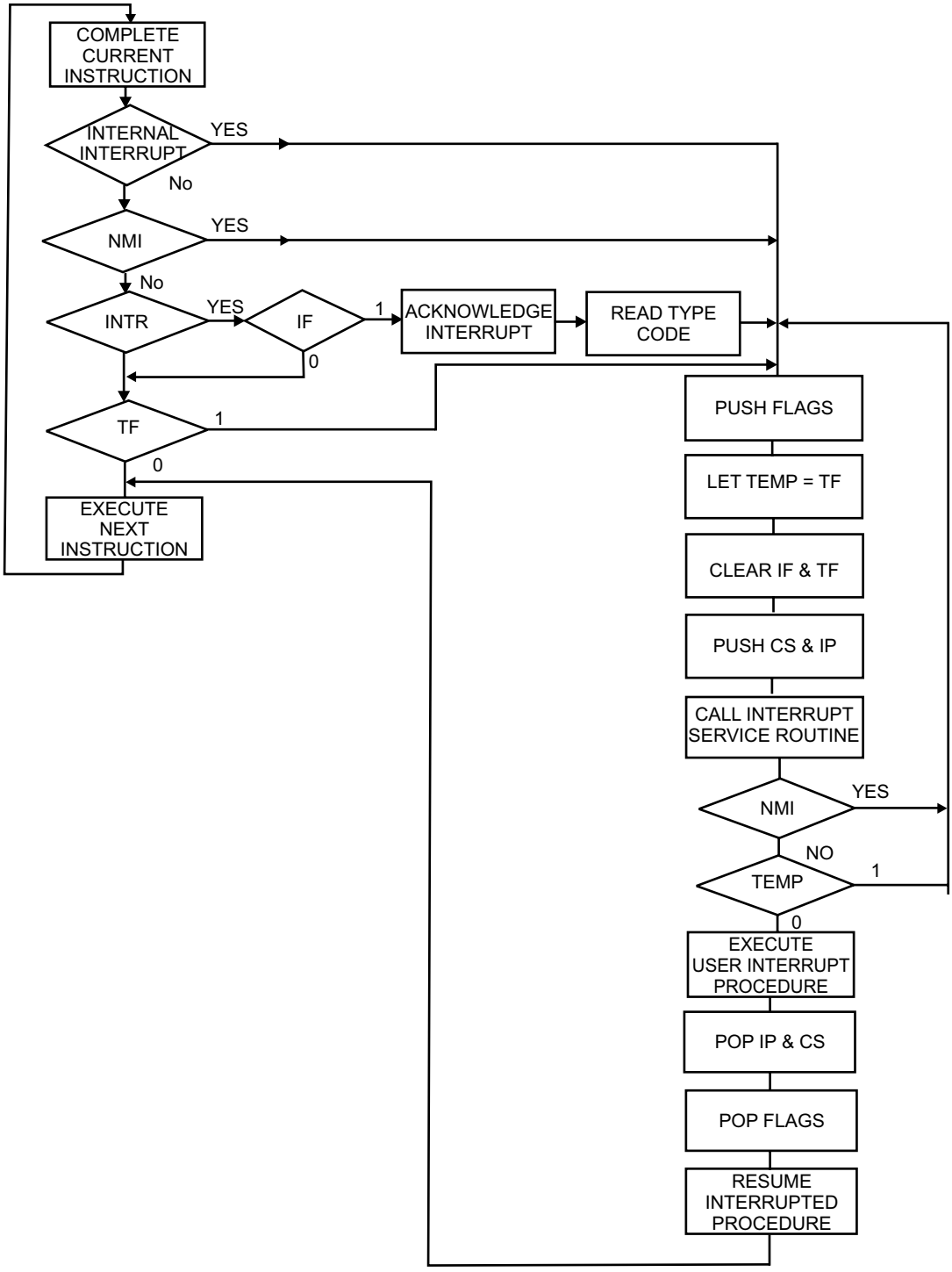

Fig.18.2: Interrupt processing sequence of the 8086 microprocessors

11. List the different interrupt instructions associated with 8086 μP.

Ans. Table 18.3 lists the different interrupts of 8086 μP along with a brief description of their functions.

Table18.3 : The different interrupt instructions of 8086 μP

Mnemonic	Meaning	Format	Operation	Flags affected
CLI	Clear interrupt flag	CLI	0→(IF)	IF
STI	Set interrupt flag	STI	1→(IF)	IF
INT n	Type n software interrupt	INT n	(Flags)→((SP) − 2) 0→TF, IF. (CS)→((SP) − 4) (2 + 4, n)→(CS) (IP)→((SP)–6) (4 . n)→(IP)	TF, IF
IRET	Interrupt return	IRET	((SP))→(IP) ((SP)+2)→(CS) ((SP)+4)→(Flags) (SP)+6→(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

12. Show the internal interrupts and their priorities.

Ans. The internal interrupts are: Divide-by-0, single step, break point and overflow corresponding to Type 0, Type 1, Type 3 and Type 4 interrupts respectively.

Since a type with lesser number has higher priority than a type with more number, thus the mentioned internal interrupts can be arranged in a decreasing priority mode, with highest priority mentioned first: Divide-by-0, single step, break point, overflow.

13. What are the characteristics associated with internal interrupts?

Ans. The following are the characteristics associated with internal interrupts:

- The interrupt type code is either contained in the instruction itself or is predefined.
- No $\overline{\text{INTA}}$ bus cycles are generated, as in the case of INTR interrupt input.
- Apart from single step interrupt, no other internal interrupt can be disabled.
- Internal interrupts, except single step have higher priority than external interrupts.

14. Discuss the two interrupts HLT and WAIT.

Ans. On execution of HLT (halt) instruction by 8086, CPU suspends its instruction execution and enters into an idle state. It waits for either an external hardware interrupt or a reset input (interrupt). When any one of these occurs, CPU starts executing again.

When the WAIT instruction is executed by 8086, it internally checks the logic level existing at its $\overline{\text{TEST}}$ input. If $\overline{\text{TEST}}$ is at logic 1 state, then CPU goes into an idle state. When $\overline{\text{TEST}}$ input assumes a zero state, execution resumes from the next sequential

instruction in the program. $\overline{\text{TEST}}$ input is normally connected to the BUSY output signal of 8087 NDP.

15. Mention the addresses at which CS_{40} and IP_{40} corresponding to vector 40 would be stored in memory.

Ans. INT 40, for its storage, requires four memory locations—two for IP_{40} and two for CS_{40}.

The addresses are calculated as follows:

$$4 \times 40 = 160_{10} = 1010\ 0000_2 = A0\ H.$$

Thus, IP_{40} is stored starting at 000A0 H and CS_{40} is stored starting at 000A2 H.

16. Explain in detail the external hardware interrupt sequence.

Ans. The external device can request for service via INT 32 through INT 255 by pulling the corresponding INT n (n = 32 to 255) high. The interrupt request gives rise to generation of interrupt acknowledge bus cycles and then moving into ISS corresponding to the device which has interrupted the system. The presently requested interrupt is recognised provided no higher priority interrupt is pending and IF is already set via software.

Once the interrupt is recognised (since for any INTR to be recognised, the corresponding INT must stay high till the last clock cycle of the presently executed instruction). 8086 initiates interrupt acknowledge bus cycles, shown in Fig. 18.3.

During T_1 of the first interrupt bus cycle, ALE is put to low state and remains so till the end of the cycle. During the whole of this cycle, address/data bus is driven into Z-state. During T_2 and T_3 of this first interrupt bus cycle, $\overline{\text{INTA}}$ is put to low state—indicating that the request for service has been granted so that the requesting device can withdraw its high logic which is connected to INTR pin 8086.

$\overline{\text{LOCK}}$ signal is of importance only in maximum mode. This signal goes low during T_2 of the first INTA bus cycle and is maintained in zero state until T_2 of the second INTA bus cycle.

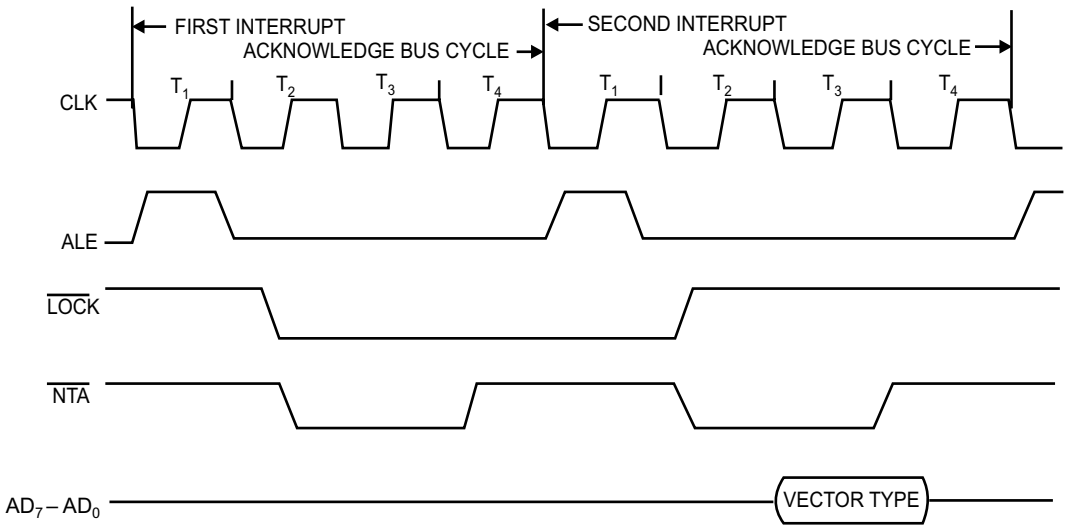

Fig. 18.3: Interrupt acknowledge bus cycle

8086 is prevented from accepting a HOLD request. The $\overline{\text{LOCK}}$ output, in conjunction with external logic, is used to lock off other devices from the system bus. This ensures completion of the current interrupt till its completion.

During the second interrupt bus cycle, the external circuit puts in the interrupt code (32_{10} = 20 H through 255_{10} = FF H) on the data bus AD_0– AD_7 during T_3 and T_4 and is read by 8086.

Before moving to ISS, CPU saves the contents of the flag register along with the current CS and IP values. Now by reading the number from the data bus, the corresponding CS and IP values are placed in them (for instance, if the external device has interrupted via INT 60, then CS_{60} and IP_{60} would be loaded into CS and IP register respectively). Thus the ISS would be run to its completion because before moving into ISS, IF and single-stepping have been disabled.

In the last line of ISR, an IRET instruction is there, which on its execution pops the old CS and IP values from the stack and put them in CS and IP registers. This thus ensures that the main program starts at the very memory location that was left off because of ISS.

17. Indicate two applications where NMI interrupt can be applied.

Ans. NMI is a non-maskable hardware interrupt, i.e., it cannot be masked or disabled. Hence, it is used for very important system exigencies like (a) detection of power failure or (b) detection of memory read error cases.

18. Draw a circuit that will terminate the INTR when interrupt request has been acknowledged.

Ans. Fig. 18.4 makes INTR input of 8086 to go into 1 state once the interrupt request comes from some external agency. The falling edge of the peripheral clocks the flip-flop which makes INTR to become 1. The first $\overline{\text{INTR}}$ pulse then resets Q, making INTR to become 0. This ensures that no second interrupt request is recognised by the system. The reset input sees to it that INTR remains in the 0 state when the system is reset.

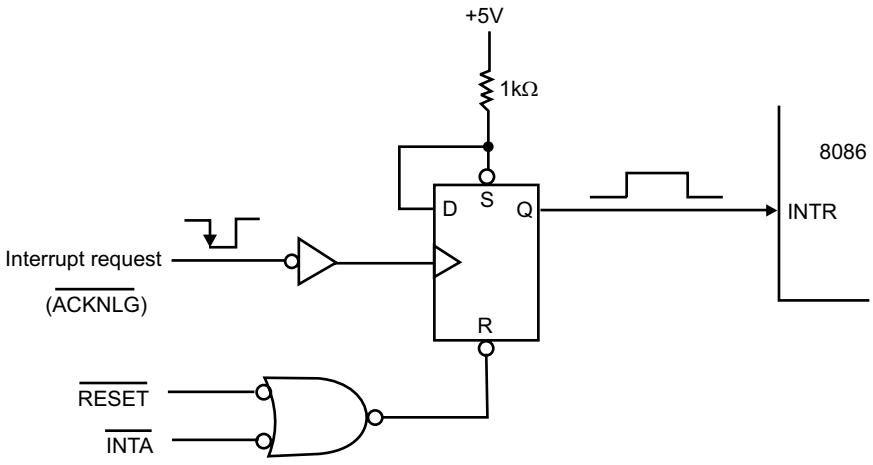

Fig. 18.4: Termination of INTR request once interrupt has been acknowledged

19. Discuss the following (a) Type 0 interrupt (b) Type 1 interrupt (c) Type 2 interrupt (d) Type 3 interrupt and (e) Type 4 interrupt.

Ans. (a) Type 0 interrupt (or Divide-by-zero interrupt)

If the quotient resulting from a DIV (divide) instruction or an IDIV (integer divide) instruction is too large such that it cannot be accommodated in the destination register, a divide error occurs. Then 8086 perform a Type 0 interrupt. This then passes the control to a service subroutine at addresses corresponding to IP_0 and CS_0 at 0000 H and 0002 H respectively in the pointer table.

(b) Type 1 interrupt (Single Step interrupt)

The single step interrupt will be enabled only if the trap flag (TF) bit is set (= 1). The TF bit can be set/reset by software.

Single step control is used for debugging in assembly language. In this mode the processor executes one instruction and then stops. The contents of various registers and memory locations can be examined. If the results are found to be ok, then a command can be inserted for execution of the next instruction. Trap flag cannot be set directly. This is done by pushing the flags on the stack, changes are made and then they are popped.

(c) Type 2 interrupt (non-maskable NMI interrupt)

Type 2 interrupt is the non-maskable NMI interrupt and is used for some emergency situations like power failure. When power fails, an external circuit detects this and sends an interrupt signal via NMI pin of 8086. The DC supply remains on for atleast 50 ms via capacitor banks so that the program and data remaining in RAM locations can be saved, which were being executed at the time of power failure.

(d) Type 3 interrupt (break point interrupt)

Type 3 interrupt is a break point interrupt. The program runs up to the break point when the interrupt occurs. This is achieved by inserting INT 3 at the point the break is desired. The ISS corresponding to Type 3 interrupt saves the register contents in the stack and can also be displayed on CRT and the control is returned to the user. This is used as a software debugging tool, like single stepping method.

(e) Type 4 interrupt (overflow interrupt)

The software instruction INTO (interrupt on overflow) is inserted in a program immediately after an arithmetic operation is performed. Insertion of INTO implements a Type 4 interrupt. When the signed result of an arithmetic operation on two signed numbers is too large to be stored in the destination register or else in a memory location, an overflow occurs and the OF (overflow flag) is set. This initiates INT4 instruction and the program control moves over to the starting address of the ISS, which corresponds to IP_4 and CS_4. These two are stored at address locations 0010 H and 0012 H respectively.

20. In what way the INTO instruction is different from others?

Ans. The INTO instruction is different in that no type number is needed to be mentioned.

To explain the difference, for executing any INT instruction, type no. is needed, like INT 10, INT 23, etc.

To explain further,

Opcode	Operand	Object Code	Mnemonic
INT	Type	CD 23	INT 23 H (assuming Type 23 H is employed)
INTO	none	CE	INT

21. Draw the schemes of (a) Min and (b) Max mode 8086 system external hardware interrupt interface and explain.

Ans. (a) The scheme of interconnections of Min-mode 8086 system external hardware interrupt interface is shown below in Fig. 18.5.

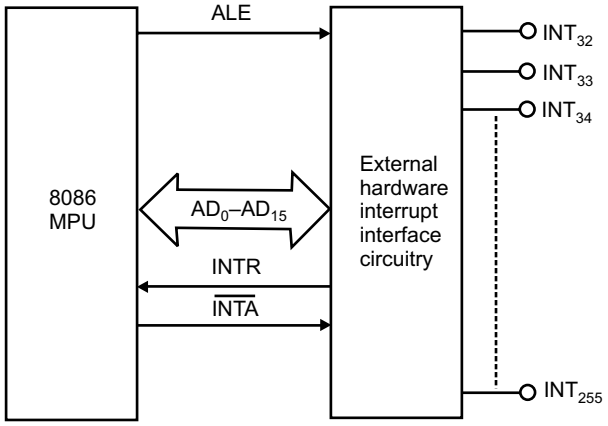

Fig. 18.5: Minimum-mode external hardware interrupt interface

The interconnecting signals to be considered for 8086 are ALE, INTR, $\overline{\text{INTA}}$ and the data bus $AD_0 - AD_{15}$.

The external device requests the service of 8086 via the INTR line. INTR is level triggered and must stay at logic 1 until recognised by the processor. Two interrupt acknowledge bus cycles are generated in response to INTR. At the end of the first bus cycle, the INTR should be removed so that it does not interrupt the 8086 a second time and the ISS can run without interruption. In this second bus cycle CPU puts the type number on the data bus of the active interrupt.

(b) The scheme of interconnections of Max-mode 8086 system external hardware interrupt interface is shown below in Fig. 18.6.

In this mode, the bus controller IC 8288 generates the $\overline{\text{INTA}}$ and ALE signals. $\overline{\text{INTA}}$ is generated when at the input of 8288, a status 000 H is applied via the status lines $\overline{S_2}\ \overline{S_1}\ \overline{S_0}$.

The $\overline{\text{LOCK}}$ signal in the figure is the bus priority lock signal and is the input to bus arbiter circuit. This circuit ensures that no other device can take control of the system buses until the presently run interrupt acknowledge cycle is completed.

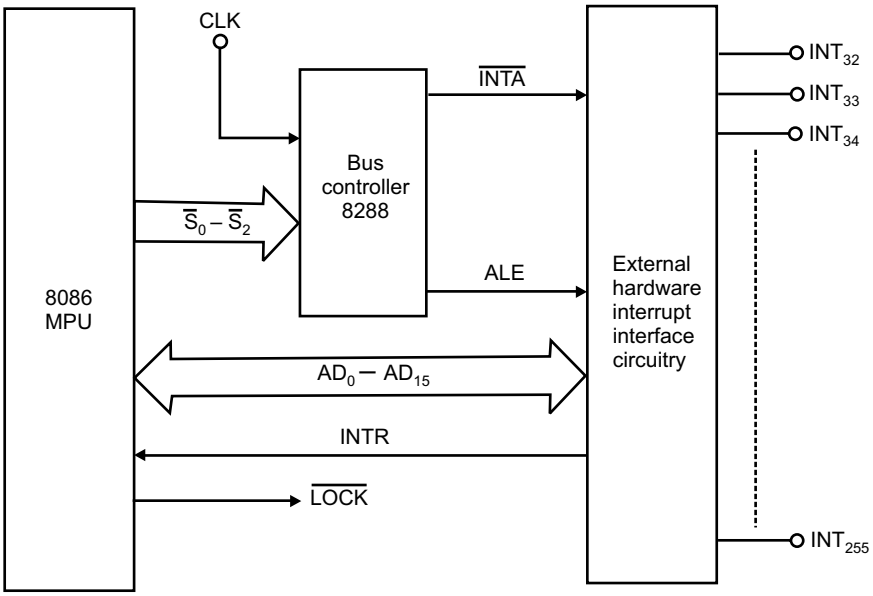

Fig.18.6: Maximum-mode 8086 system external hardware interrupt interface