INTERRUPTS AND EXCEPTIONS

Page No.
Date

1) Explain the process of Enabling and Disabling Interrupts in 80386.

→

Certain conditions and flag settings cause the processor to inhibit certain interrupts and exceptions at instruction boundaries. These ~~or~~ conditions and settings are :

NMI masks further NMIs :

- While an NMI handler is executing, the processor ignores further interrupt signals at the NMI pin until the next IRET instruction is executed.

IE mask INTR :

- The external interrupts that are controlled through the INTR pin are controlled by the IE (Interrupt-Enable) flag.
- If IE = 1, the INTR pin is enabled and if IE = 0, the INTR pin is disabled.

RF masks debug faults :

The RF bit in EFLAGS controls the recognition of debug faults. This permits debug faults to be raised for a given instruction at most once, no matter how many times the instruction started.

Page No.
Date

MOV or POP to SS masks some Interrupts & Exception:

- software that needs to change stack segments often uses a pair of instructions; for eg.

```
MOV SS,AX
MOV ESP, stacktop.
```

- If an interrupt or ~~exp~~ exception is processed after SS has been changed but before ESP has received the corresponding change, the two parts of the stack pointer ~~ES~~ SS:ESP are inconsistent for the duration of the interrupt handler or exception handler.

- To prevent this situation, the 80386 after both a MOV to SS and POP to SS instruction inhibits NMI, INTR, debug exceptions, and single-step traps at the instruction boundary following the instruction that changes SS. some exception may still occur, namely page fault and general protection fault. Always use the 80386 LSS instruction and the problem will not occur.

Page No.
Date

2) With the help of necessary diagram, explain structure of IDT in 80386.

→

- In protection mode, each interrupt or exception is associated with a descriptor which gives the information about interrupt service routine. These descriptor are stored in special descriptor table called Interrupt Descriptor Table or IDT.

- This table can be located anywhere in memory like the GDT and LDTs, the IDT is an array of 8 byte descriptors. The base address and limit for the interrupt descriptors tables are loaded into the interrupt Descriptor Table Register (IDTR).

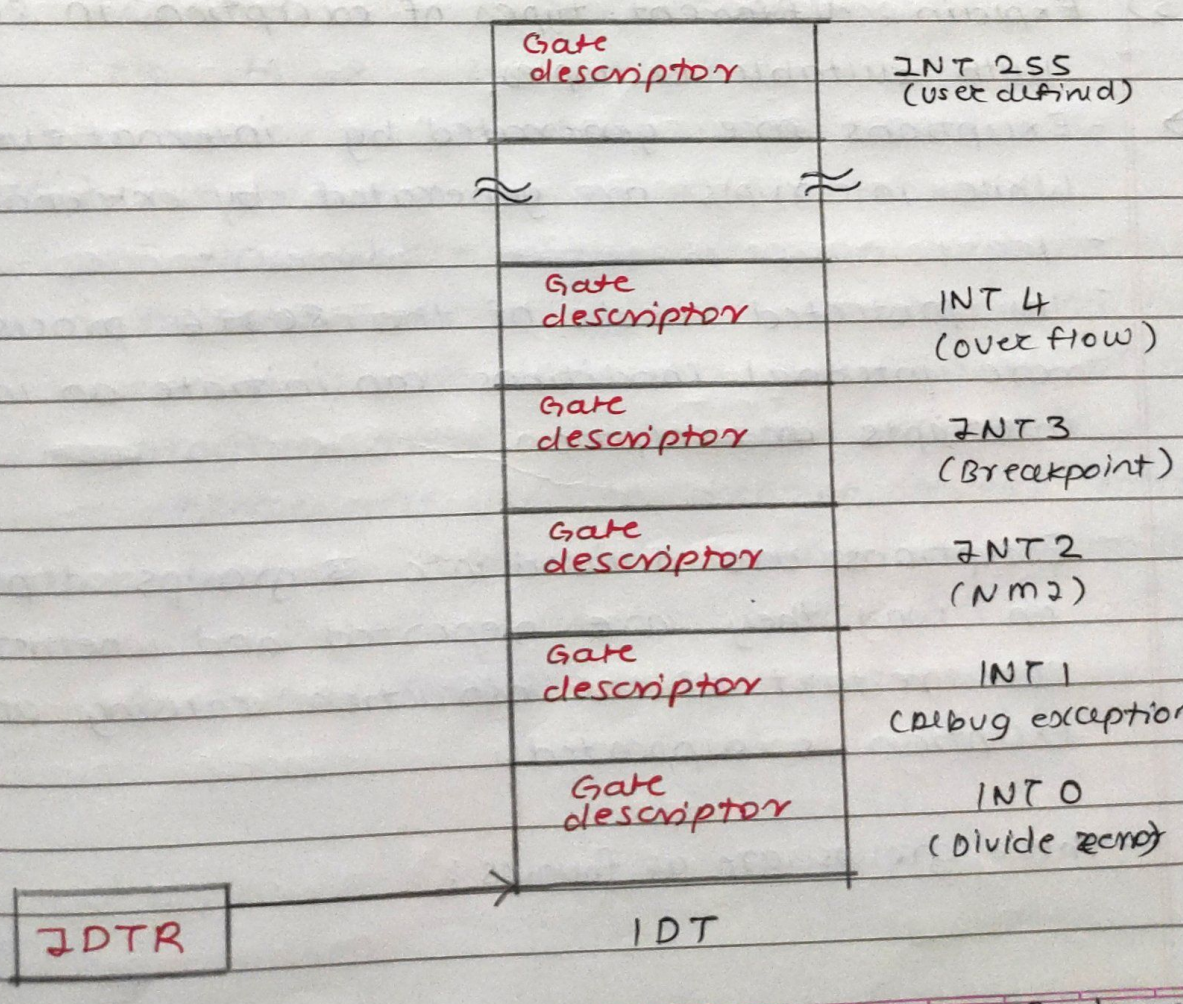


fig. Interrupt descriptor table & IDTR

- LIDT (load IDT register) / SIDT (store IDT register) instructions are used to load / store the IDT register with linear base address and limit values, respectively.

- Because there are only 256 identifiers, the IDT need not contain more than 256 descriptors. There are three types of descriptors can be used in the IDT
  - Trap gate descriptor
  - Interrupt gate descriptor
  - Task gate descriptor.

3) Explain different types of exception in 80386 with suitable examples.

→ - Exceptions are generated by internal events while interrupts are generated by external events.
- ;
- In protected mode of the 80386 processor, more internal conditions can initiate an internal interrupts or exception.

- Exceptions are divided into 3 groups depending on way they are reported and whether or not restart of the instruction causing an exception is supported.

- These groups are as follows:

(i) Fault:

- Fault are exceptions that are reported "before" the instruction causing the exception.
- Faults are either detected before the instruction begins to execute, or during execution of the instruction. If detected during the instruction, the fault is reported with the machine restored to a state that permits the instruction to be restarted.
- Eg. INT0, INT5, INT6

(ii) Traps:

- A trap is an exception that is reported at the instruction boundary immidiately after the instruction in which the exception was detected.
- eg. INT2, INT3, INT4

(iii) Aborts:

- An abort is an exception that permits neither precise location of the instruction causing the exception nor restart of the program that caused the exception.
- Aborts are used to report severe errors, such as hardware errors and inconsistent or illegal values in system tables.
- Eg. INT8, INT9

MICROPROCESSOR	MICROCONTROLLER
(i) Microprocessor is heart of computer system	(i) Microcontroller is a heart of embedded system.
(ii) It is a processor in which memory and I/O output component is connected externally	(ii) It is a controlling device in which memory and I/O output component are present internally
(iii) The circuit is more complex	(iii) The circuit is less complex
(iv) Microprocessors are expensive	(iv) Microprocessors are cheaps
(v) It cannot be used in compact system. Therefore microprocessor is ~~inefficient~~ inefficient.	(v) It can be used in a compact system. Therefore microcontroller is more efficient.
(vi) speed is ~~slower~~ fast	(vi) Speed is ~~fast~~ Slower
(vii) Easy to replace	(vii) Not easy to replace
(viii) Do not have power saving mode	(viii) Have power saving mode
(ix) Based on Von-Neuman Model	(ix) Based on the Harvard architecture
(x) Used in personal computers	(x) Used in embedded system.

Page No.
Date

* How are interrupts handled in protected mode? Explain with help of neat diagram?

→

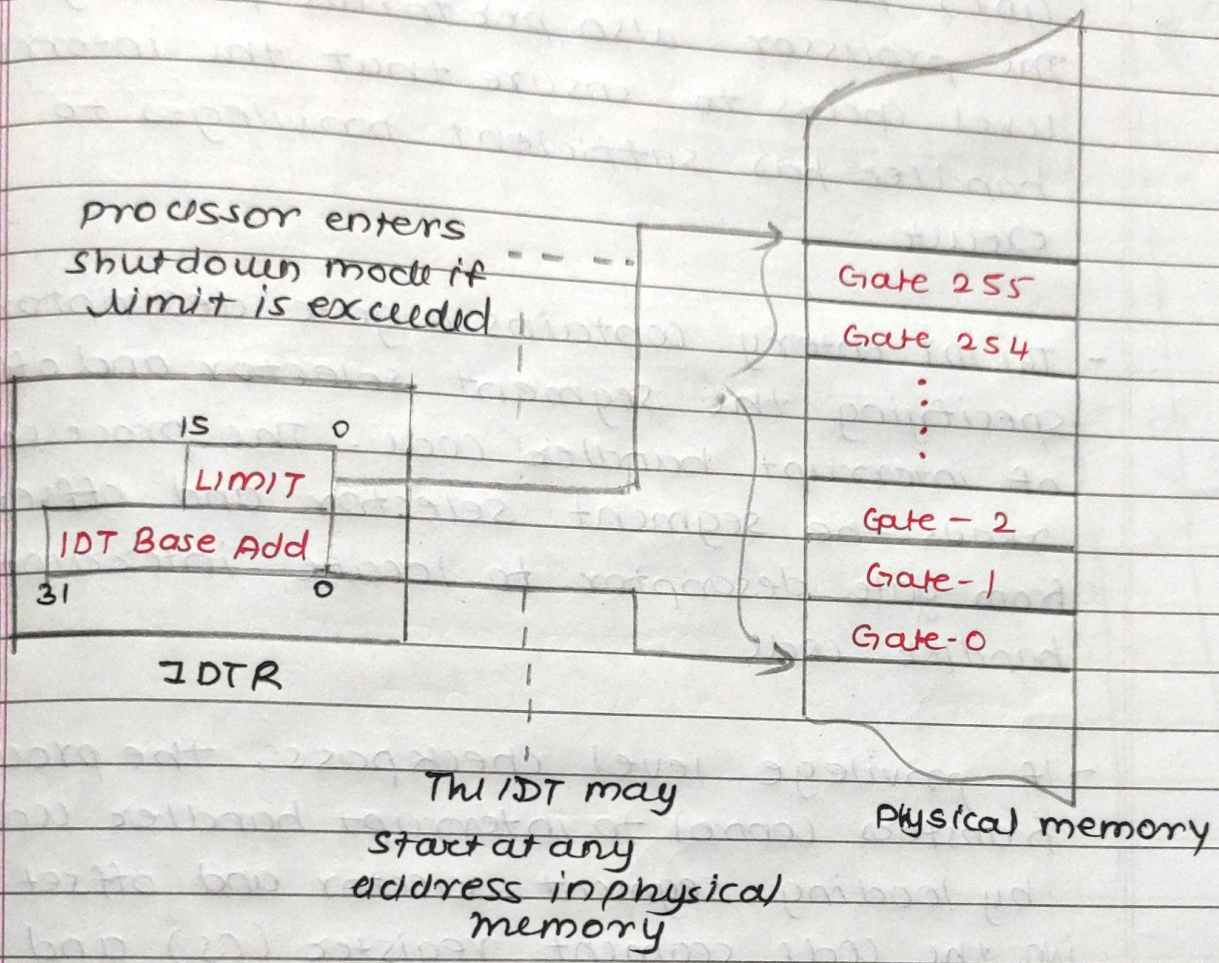


fig: Protected mode Interrupt descriptor table

- An external device or an ~~interal~~ internal event triggers an interrupt. The processor saves the current execution state onto the stack. This include the contents of the flags register and instruction pointer (IP).

- The processor sends an interrupt request (IRQ) signal to interrupt controller to handle the interrupt. Then the controller prioritizes and forwards the interrupt to the processor.

Date

~~Steps involved~~

- The processor looks up the corresponding entry in the Interrupt descriptor table (IDT) based on the interrupt number. The processor also performs privilege level checks to ensure that the interrupt handler has sufficient privileges to execute.

- The IDT entery contains a gate descriptor, specifying the segment selector and offset of interrupt handler code. The processor reads the segment selector and offset from gate descriptor to locate interrupt handler code

- If privilege level check pass, the processor transfers control to interrupt handler code by loading segment selector and offset into the code segment register (CS) and Instruction pointer (IP), respectively.

- The interrupt handler code executes to handle the interrupt or exception as necessary, after completion is restores the saved execution state from the stack,

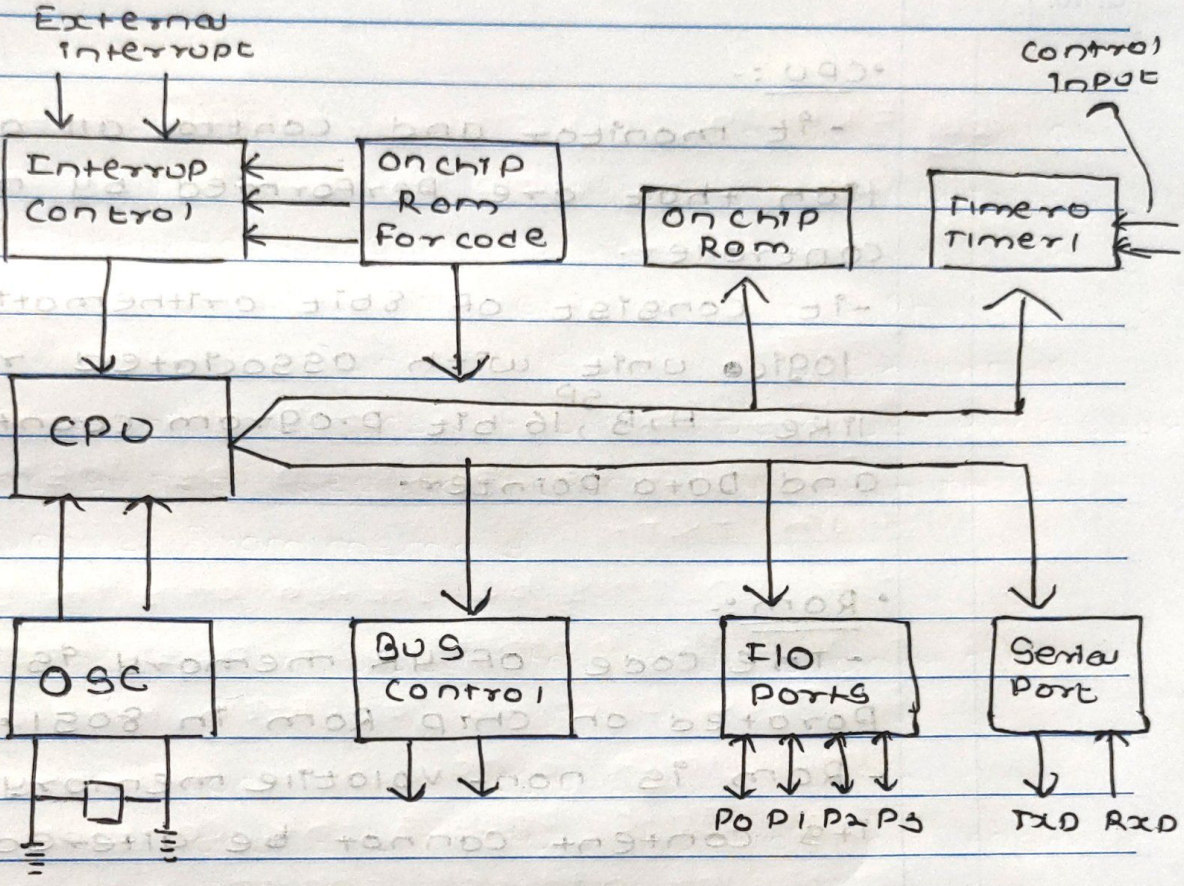
- Control is returned to interrupted code, allowing it to resum execution from where it was interrupted.

Do not write your name or seat no. below this line

Q. No.						Q. No.					

प्र. क्र.
Q. No.

MICRO-CONTROLLER ARCHITECTURE



- This figure shows architecture of typical microcontroller
- It consist of CPU and two memory section Rom and Rom, input/output ports, Serial port, Interrupt control, oscillator etc
- All these element communicate through an eight bit data bus.
- The bus is connected to outside world via I/O ports.

प्र. क्र.
Q. No.

•CPU :-

- It monitor and control all operation that are performed by micro controller.
- It consist of 8bit arithematic and logic unit with associated register like A, B, SP, 16 bit program counter and Data Pointer.

• Rom :-

- The code of 4K memory is incorporated on chip Rom in 8051.
- Rom is non-volatile memory that it's content cannot be altered.

• RAM :-

- The microcontroller composed 128 bytes of internal memory.
- This is volatile memory since the content will be lost if power is switch off.
- These 128 bytes of internal RAM is divided into 32 working register in which there is 4 register bank each bank consist 8 register.

Q. No.						Q. No.					

प्र. क्र.
Q. No.

- I/O Ports:-

The 8051 microcontroller has four 8 bit I/O port : P0, P1, P2, P3 which is used to communicate the microcontroller with outside world.

- Interrupt Control:-

- It support both internal and external interrupts.
- 5 source of interrupt are provided.

- Timers:-

8051 support two multiple mode, timer/~~counter~~ counter.

- Timer mode is used to generate delay and counter mode is used to count external pulse.

- Serial Port:-

- Provide a method of establishing a serial communication by transmitting and receiving bits.
- It uses TXD and RXD pin to transmit and receive bits.

- Oscillator:-

- It is used to provide clock to 8051.

Q. No.						Q. No.					

प्र. क्र.
Q. No.

Q4] Explain and Differentiate Interrupt gate and trap gate.

→

*~~Date~~

*Trap Gate:-

- Trap gate is called by instruction INT.
- It can only be stored only in IDT.
- This is nothing but basic type of gate because it often has a small task to do.
- Trap gate passes the control to particular address specified in trap gate descriptor.

*Uses of trap gate

① System call activation

② Management application

*Interrupt gate:-

- Interrupt gate is called by INT instruction.
- It is also stored in IDT.
- Interrupt gate is similar to trap gate except there is one difference

that is all gate additionally prohibit future interrupt acceptance by automatic clearing of IF flag in EFlag register.

feature	Interrupt gate	Trap gate
Purpose	Handle external interrupt	Handle Software generated exception
Trigger	Triggered by external device or CPU	Trigger by CPU executing specific instruction
Source	external device or hardware signal	Software generated exception or interrupt
Used	used for event like I/O operation, timer, interrupt	used for event like divide by zero, over flow etc.

Q. Explain the structure of V86 task in detail. How is protection provided within the V86 task?

→

- A V86 task consists partly of the 80386 program to be executed partly of 80386 "native mode" code that serves as the virtual-machine monitor.

Q. Explain the following exceptions in brief.

i) Divide error:

- The divide-error fault occurs during a DIV or an IDIV instruction when the divisor is zero.

ii) Overflow error:

- Interrupt on Overflow (INTO) is a conditional software interrupt which tests the overflow (O) flag. IF 'O' = 0, the INTO performs no operations.

- But IF 'O' = 1, an INTO instruction executes.

- Another way to detect & respond to an overflow error in a program is to put the jump if overflow instruction (JO) immediately after arithmetic instruction.

- If overflow flag is set as a result of arithmetic operation, execution will jump to address specified in JO instruction.

iii) Invalid opcode:

- This fault occurs when an invalid opcode is detected by the execution unit.

- This exception also occurs when the type of operand is invalid for the given opcode.

- Examples include an intersegment JMP referencing a register operand, or an LES instruction with a register source operand.

- No error code is pushed on stack. The exception can be handled within the same task.

6M

PAGE NO.
DATE:

Q. List & elaborate on different applications of microcontroller.

→

1) Home appliances:
Washing machine, refrigerators, microwave ovens etc.

2) Calculators, Keyboards, printers, modems, mobile phones etc.

3) Industrial controllers, data acquisition systems communication systems etc.

4) Automobile engines, flight control system, traffic light control systems, etc. military applications.

---

1. Consumer Electronics:

- Home Appliances:
  Washing machines, microwave ovens, refrigerators, & air conditioners.

- Personal Gadgets:
  Smartphones, cameras, fitness tracker using microcontroller to handle sensor input & display management.

PAGE NO.
DATE:

2. Medical Devices:

Heart rate monitors, blood pressure devices & glucose meter, ultrasound machines.

3. Communication Systems:

- Network Devices:

Routers, modems, & other networking hardware ~~etc~~

- Wireless Communication:

Bluetooth, zigbee, & WiFi for wireless data transmission

4. Automotive Industy:

- Engine Control Units:

critical in managing engine performance, fuel injections, & emission control system.

- Safety Systems:

Airbag development, antilock braking system (ABS), & ECS.

5. Industrial Automation:

Industrial controllers, data acquisition system communication system, PLC (Programmable Logic Controllers), robitics

PAGE NO.

DATE:

6. Sensor monitoring:

They collect data from various sensors for temperature, pressure, & humidity monitoring.