CSE-3103: Microprocessor and Microcontroller

Dept. of Computer Science and Engineering
University of Dhaka

Prof. Sazzad M.S. Imran, PhD
Dept. of Electrical and Electronic Engineering
sazzadmsi.webnode.com

Purpose of Interrupts

Interfacing I/O devices with microprocessor →

- I/O devices have low data transfer rates.
- 2 methods are used →
  - 1) Strobed or polling technique.
  - 2) Interrupt processing.

Strobed technique →

- microprocessor keeps checking input device to see if new data has arrived.
- it looks at status bit to decide whether data were available.

- keyboard using strobed input operation of 82C55 →
  - software polled 82C55 and its IBF bit (input buffer full),
  - keyboard is typed 1 cps →
    - CPU will waste almost whole second,
    - waited entire second between each keystroke.

Purpose of Interrupts

Interrupt processing →

microprocessor does not continuously check for input.
allows microprocessor to execute other software between each keystroke.
user presses key →

keyboard encoder debounces switch,
puts out one pulse that interrupts microprocessor.
this interrupt signal tells CPU that "data is ready".
CPU temporarily stops what it was doing,
jumps to special routine (called Interrupt Service Routine, ISR),
reads key data,
returns to continue executing software.

What is interrupt?

signal that temporarily stops normal execution of program,
forces processor to run special program called ISR.
after finishing ISR →

processor returns to main program.

Interrupts

Hardware interrupts pins →

INTR →

interrupt request,
generated by devices like keyboard, printer, etc.
CPU can ignore it if interrupts are disabled (IF = 0).

NMI →

non-maskable interrupt,
cannot be disabled.
used for emergencies like power failure or memory parity error.

$\overline{\text{INTA}}$ →

interrupt acknowledge.
sent by CPU to acknowledge that
it has received interrupt request from device.

Interrupts

- Software interrupts →
  - initiated by special instructions in program itself.
  - INT n →
    - executes interrupt number n.
    - INT 21H for DOS system calls.
    - INT 10H calls video service interrupt for screen operations.
  - INTO →
    - executes ISR if Overflow Flag (OF) = 1.
  - INT 3 →
    - breakpoint interrupt used for debugging.
  - BOUND →
    - checks array bounds,
    - triggers interrupt if out of range.

Interrupts

2 flag bits are used →

- IF (interrupt flag) →
  - interrupt enable flag.
  - IF = 1 →
    - CPU accepts maskable interrupts (INTR).
  - IF = 0 →
    - CPU ignores them.
- TF (trap flag) →
  - single step flag.
  - used for debugging.
  - TF = 1 →
    - CPU executes one instruction and then interrupts,
    - used for step-by-step execution.

IRET →

- interrupt return.
- used at end of ISR.
- restores previous program's flags, CS, and IP from stack,
- CPU continues exactly where it left off before interrupt.

Interrupt can be initiated by →

- i) hardware,
- ii) software, or
- iii) processor.

Interrupts

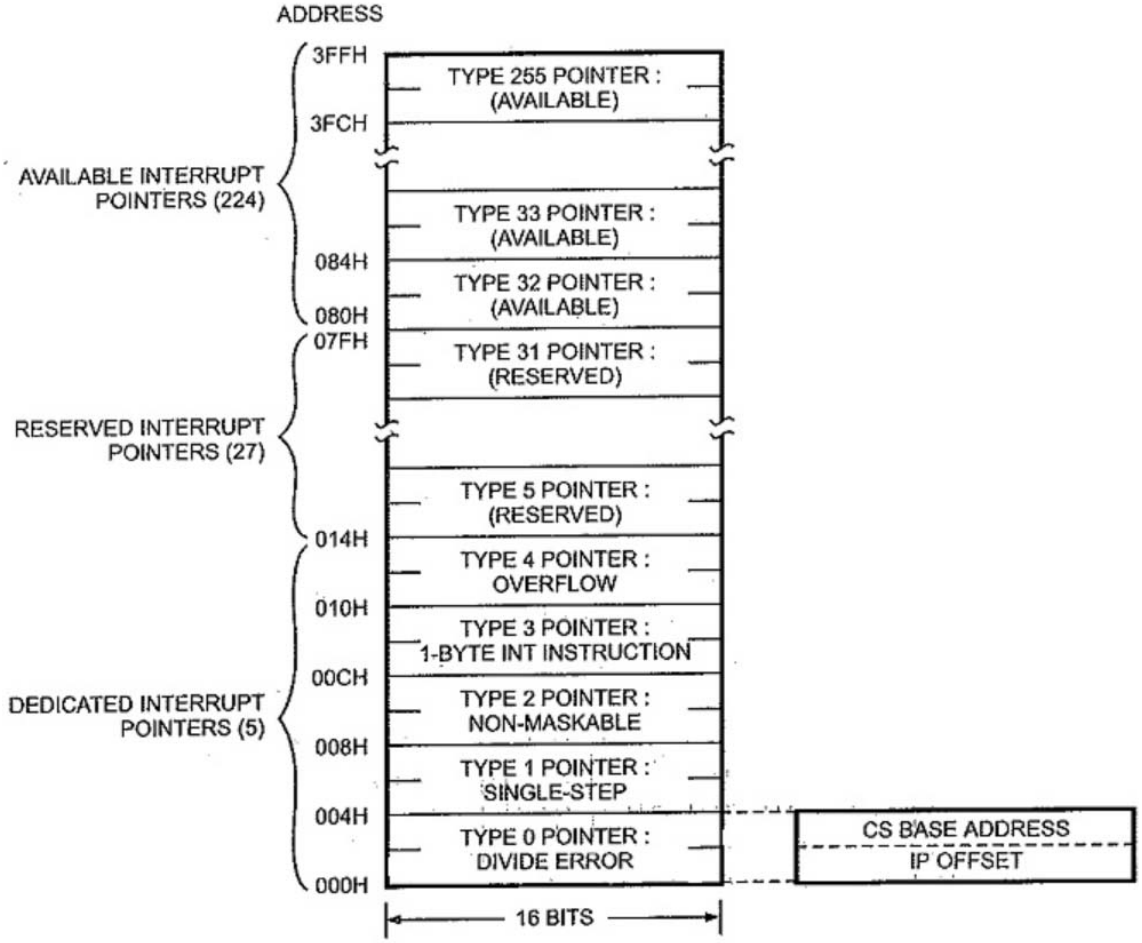
Interrupt vector table (IVT) →

structured list containing addresses of ISR for various interrupts.

each ISR has unique address = interrupt vector.
256 different 4-byte interrupt vectors.

located in first 256×4 = 1024 bytes = 1 kB of memory,
addresses = 0000:0000H – 0000:03FFH.

Intel reserves first 32 interrupt vectors for their use.
last 224 vectors are available as user interrupt vectors.

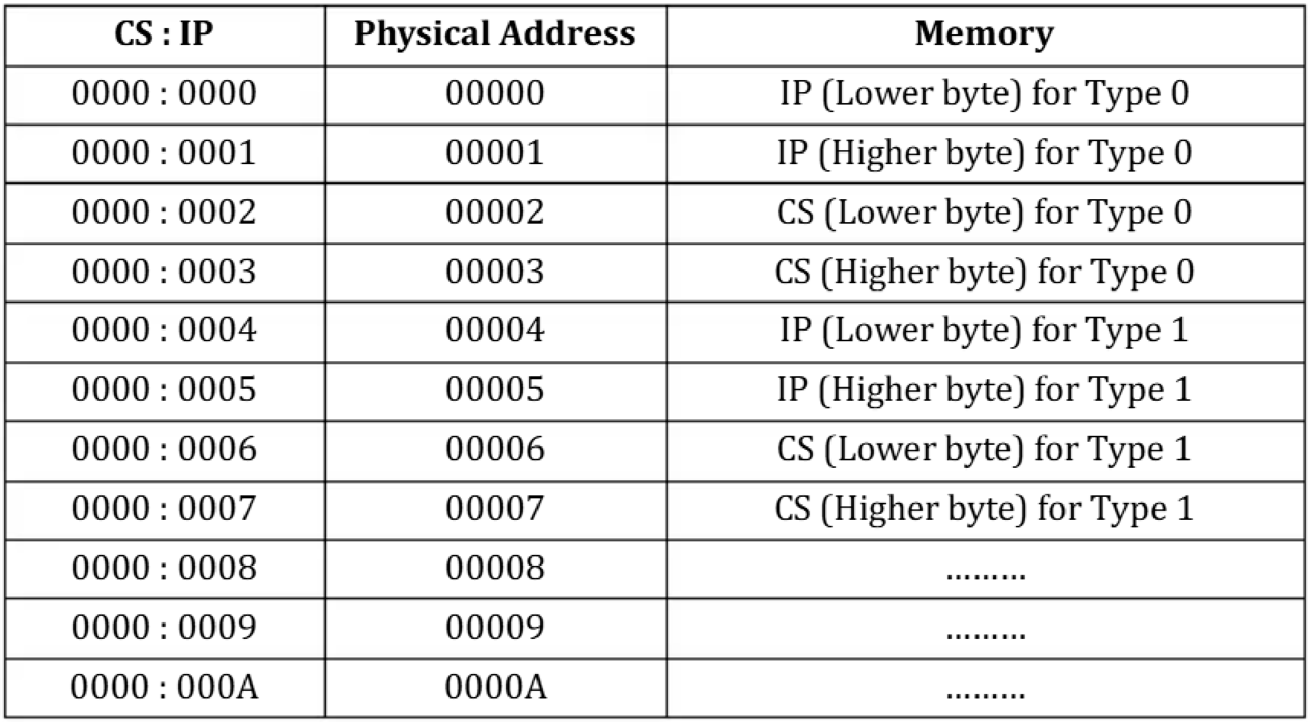

ADDRESS
3FFH
3FCH
084H
080H
07FH
014H
010H
00CH
008H
004H
000H
AVAILABLE INTERRUPT POINTERS (224)
RESERVED INTERRUPT POINTERS (27)
DEDICATED INTERRUPT POINTERS (5)
TYPE 255 POINTER : (AVAILABLE)
TYPE 33 POINTER : (AVAILABLE)
TYPE 32 POINTER : (AVAILABLE)
TYPE 31 POINTER : (RESERVED)
TYPE 5 POINTER : (RESERVED)
TYPE 4 POINTER : OVERFLOW
TYPE 3 POINTER : 1-BYTE INT INSTRUCTION
TYPE 2 POINTER : NON-MASKABLE
TYPE 1 POINTER : SINGLE-STEP
TYPE 0 POINTER : DIVIDE ERROR
CS BASE ADDRESS
IP OFFSET
16 BITS

Interrupts

Interrupt vector →

each vector is 4 bytes long in real mode.
contains starting address of interrupt service procedure.
first 2 bytes = offset address.
last 2 bytes = segment address.

IP for ISR of Type N = 4N
CS for ISR of Type N = 4N+2



CS : IP	Physical Address	Memory
0000 : 0000	00000	IP (Lower byte) for Type 0
0000 : 0001	00001	IP (Higher byte) for Type 0
0000 : 0002	00002	CS (Lower byte) for Type 0
0000 : 0003	00003	CS (Higher byte) for Type 0
0000 : 0004	00004	IP (Lower byte) for Type 1
0000 : 0005	00005	IP (Higher byte) for Type 1
0000 : 0006	00006	CS (Lower byte) for Type 1
0000 : 0007	00007	CS (Higher byte) for Type 1
0000 : 0008	00008	....
0000 : 0009	00009	....
0000 : 000A	0000A	....

Interrupts

Function of some reserved interrupt →

TYPE 0 to TYPE 4 are dedicated interrupts.

TYPE 5 to TYPE 31 are reserved interrupts.

TYPE 0	divide by zero, result from division overflows.
TYPE 2	NMI occurs when logic 1 is placed on NMI input pin.
TYPE 3	used to store breakpoint in program for debugging.
TYPE 4	used with INTO instruction, interrupts program if overflow condition exists.
TYPE 6	interrupt occurs whenever undefined opcode is encountered.
TYPE 8	two separate interrupts occur during same instruction, double fault interrupt is activated.
TYPE 11	segment not present interrupt. P = 0 in descriptor → segment is not present or not valid.

Interrupt Instructions

5 software interrupt instructions →

BOUND	compares register with 2 words of memory data. BOUND AX, DATA → If AX < DATA+1:DATA (16-bit lower limit) or AX > DATA+3:DATA+2 (16-bit upper limit), type 5 interrupt occurs.
INTO	generate overflow trap if OF = 1 (interrupt TYPE 4). used for arithmetic overflow checking.
INT n	generate software interrupt, vector specified by immediate byte. vector address of INT 5 → 4×5 or 20 (14H), vector for INT 5 is at address 0014H – 0017H.
INT 3	generate breakpoint trap. helps debuggers pause program at specific points.
IRET	used to return from Interrupt Service Procedure (ISR). IRET retrieves → 4 bytes (2 for IP, 2 for CS) return address, 2 bytes flag register from stack.

Operation of Real Mode Interrupt

Microprocessor completes executing current instruction →

determines whether an interrupt is active.

one or more interrupt conditions are present →

1) contents of flag register are pushed onto stack.
2) both IF and TF flags are cleared.
3) contents of CS are pushed onto stack.
4) contents of IP are pushed onto stack.
5) interrupt vector contents are fetched, placed into IP and CS.

return address points →

1) where interrupt occurred. Type 0, 5, 6, 7, 8, 10, 11, 12, and 13 interrupts.
2) next instruction in program for other interrupts.