**Lab Manual – Timer Interrupt**

**Important Instructions:** Best apply all the concepts studied so far.

**Activity 1:** Write a TSR to make an asterisks travel the border of the screen, from upper left to upper right to lower right to lower left and back to upper left indefinitely, making each movement after one tick. Note: Do not use any loop.

**Activity 2:** Open your tsr.com in DOSBOX then run PrintTriangle.com and 182.com one by one. The programs should run successfully and asterisk should keep moving on the border along with both the programs.

**Practice Problems**

1. Write a TSR to clear the screen when CTRL key is pressed and restore it when it is released.
2. Write a TSR to rotate the screen (scroll up and copy the old top most line to the bottom) while F10 is pressed. The screen will keep rotating while F10 is pressed at 18.2 rows per second. As soon as F10 is released the rotation should stop and the original screen restored. A secondary buffer of only 160 bytes (one line of screen) can be used.
3. Write a keyboard interrupt handler that disables the timer interrupt (no timer interrupt should come) while Q is pressed. It should be re-enabled as soon as Q is released. Hook timer to print timer ticks on screen and verify both functionalities.
4. Write a TSR to calculate the current typing speed of the user. Current typing speed is the number of characters typed by the user in the last five seconds. The speed should be represented by printing asterisks at the right border (80th column) of the screen starting from the upper right to the lower right corner (growing downwards). Draw n asterisks if the user typed n characters in the last five seconds. The count should be updated every second.
5. Write a TSR to show a clock in the upper right corner of the screen in the format HH:MM:SS.DD where HH is hours in 24 hour format, MM is minutes, SS is seconds and DD is hundredth of second.

HINT: IBM PC uses a Real Time Clock (RTC) chip to keep track of time while switched off. It provides clock and calendar functions through its two I/O ports 70h and 71h. It is used as follows:

mov al, <command>

out 0x70, al ; command byte written at first port

jmp D1 ; waste one instruction time

D1: in al, 0x71 ; result of command is in AL now

Following are few commands: 00 - Get current second, 02 - Get current minute, 04 - Get current hour

All numbers returned by RTC are in BCD. E.g. if it is 6:30 the second and third command will return 0x30 and 0x06 respectively in al.

1. Write a TSR that hooks software interrupt 0x80 and the timer interrupt. The software interrupt is called by other programs with the address of a far function in ES:DI and the number of timer ticks after which to call back that function in CX. The interrupt records this information and returns to the caller. The function will actually be called by the timer interrupt after the desired number of ticks. The maximum number of functions and their ticks can be fixed to 8.
2. Chapter 8: Q 1 and 2