****

Proj_1E_Timer_interrupt

A PROGRAM THAT DOES MULTI-TASKING

The program continues to drive the display but the vast majority of time which was spent waiting to increment the display is used to do something useful. In this case doing some arithmetic and sending the results to the PC to be displayed.

IT INTRODUCES

1. Project subroutine T1_65ms_clock_tick() This generates an interrupt every 65 mS.

2. ISR(TIMER1_OVF_vect){} This is the Timer1 interrupt service routine. Here, it is the ISR that increments the LED display rather than the main routine.

3. Code other than a timer subroutine that can safely be interrupted. Here the code is executed indefinitely by a while(1) loop.

4. Project subroutines Char_to_PC() and Num_to_PC_U() These send data to the PC and will be considered in more detail later on.

5. Timer T2 This slows down data flow to the PC.

6. long numbers: These occupy 32 bits of data memory and can hold numbers between -2^31 and 2^31 – 1 or between 0 and 2^32 - 1 = 4,294,967, 295.

Note: The logic is necessary for the display and may be found interesting but is not really the important thing here.

A QUICK LOOK AT SOME OF THE LOGIC

Consider the statement PORT_1 = ~(~PORT_1 >> 1);
If PORT_1 = 0 then (~PORT_1 >> 1) = 0b0111111111111111 and ~(~PORT_1 >> 1) = 0b1000000000000000

The next time the loop runs (~PORT_1 >> 1) = 0b0011111111111111 and ~(~PORT_1 >> 1) = 0b1100000000000000