****
Proj_1C1 Pin_Change_Interrupt PROGRAM 3 A REACTION TIME TESTER

Test your reactions by attempting to shoot the LEDS down using switch_2 (the middle one).

SOME OF THE THINGS THIS PROGRAM INTRODUCES:

1. Interrupts (Signals generated by the Atmega HW):
An interrupt is a signal that interrupts normal program flow, which then temporarily leaves the main routine and jumps to a special subroutine known as an ISR (Interrupt Service Routine). On this PCB Interrupts can be generated by timers, by pressing a key on the PC or by operating one of the project pcb switches. At the end of the ISR program flow returns to where it was when the interrupt occurred.

2 config_sw3_for_PCI This is a project macro that sets up switch_3 to generate Pin Change interrupts (PCI).

3. ISR(PCINT0_vect){}: This is the Interrupt Service Routine (ISR) that is called every time sw_2 is pressed or released. Note that program flow returns immediately when the switch is released so that it is only switch presses that have any effect.

4. Global variables: Note that variables PORT_1 and mask are available to both the main routine and also to the ISR (Note they are also volatile because of their use by the ISR).

5. More complex logic: Not really of interest at this stage. This has been added in an attempt to produce something interesting and its study may usefully be delayed.

6. The statement if(m == 1) which means execute the next statement if m equals 1, but if it equals anything else skip the next statement.

MORE ON THE LOGIC

The -^- symbol know as Exclusive-OR is similar to OR except that 1^1 = 0 where as 1|1 = 1 For example 10101010 ^ 11110000 = 01011010

Variable mask starts of as 1111111111111111

Assume that sw3 is pressed when PORT_1 is 0000000000000100

~PORT_1 is 1111111111111011 and mask which is set to mask & ~PORT_1 becomes 1111111111111111 & 1111111111111011 = 111111111111011

Assume that sw3 is pressed again when PORT_1 is 0000000000000010 mask now becomes 1111111111111011 & 1111111111111101 = 1111111111111001

After pressing sw3 several times assume that the mask is 1111111111000011 and therefore ~mask is 0000000000111100

PORT_1 & mask is zero for the following values of PORT_1 0000000000000100, 0000000000001000, 0000000000010000 and 0000000000100000
Therefore the upper half of the display skips leds numbered 2,3,4 and 5 counting from zero i.e those that have been shot down

{(~mask) ^ PORT_1} always illuminates leds numbered 2,3,4 and 5 except for the following values of PORT_1 0000000000000100, 0000000000001000, 0000000000010000 and 0000000000100000

i.e. the lower half of the display remembers the leds that have been shot down Note however that the lower leds flicker when their upper companion is passed by