Department of Mechanical, Materials and Manufacturing Engineering

Computer Engineering and Mechatronics MMME3085

Exercise Sheet 9: Interrupts on AVR microcontrollers

The solution to Q2 on Exercise Sheet 3 includes an alternative solution to the generation of 20 kHz fast PWM on the Arduino Uno, using some "tricks" to get Timer 1 to operate the LED on pin 13 even though (on the Uno) there is no direct output from a timer to that pin. How does this program work? You will need to revisit the end of Lecture 3 where the Arduino Mega version of this PWM approach is described. Hint: you need to look at the Atmega 328 data sheet to understand the fast PWM mode and the interrupts used, though for some reason the bit names used there are slightly different from the ones that actually work in the compiler (**`TOEI`** and **`OCIEA`** rather than **`TOIE1`** and **`OCIE1A`**). Note also that in this particular PWM mode, matching of **`TCNT1`** with **`ICR1`** is treated as an overflow (as the counter register **`TCNT1`** is reset to zero) and hence triggers an overflow interrupt.

```
include <avr/io.h>

void setup()
{
  // Note: there isn't an OCR1C etc. on the Uno so for OCR1C, COM1C0..1
  // etc. in hints, read OCR1A, COM1A0..1 etc. noting this operates pin 9
  DDRB = (1 << PB1) | (1 << PB5); // Make both pin 9 and 13 output
  TCCR1A = (1 << WGM11) | (1 << COM1A1);
  TCCR1B = (1 << WGM12) | (1 << WGM13) | (1 << CS10);
  TCCR1C = 0; // No force output compare
  ICR1 = 799; // Set PWM frequency taking account of prescaler if any
  OCR1A = 79;  // Set PWM duty cycle (act'y OCR1A+1) as fraction of OCR1+1

  // So we can see results on LED, call interrupts to switch it on & off
  // on the events that also affect pin 9.
  TIMSK1 = (1 << TOIE1) | (1 << OCIE1A);
}

void loop()
{
  // Loop is deliberately empty.
}

ISR(TIMER1_OVF_vect)
{
  PORTB |= (1 << PB5);
}

ISR(TIMER1_COMPA_vect)
{
  PORTB &= ~(1 << PB5);
}
```