Embedded Real-Time Systems (AME 3623)
Homework 3 Solutions

April 27, 2010

Question 1

Assume that *student_ID* is the number that corresponds to your student ID number.

1. (2pts) What is *student_ID % 2*? Call this *key1*

   Possible answers are: 0, 1

2. (2pts) What is *student_ID % 5*? Call this *key2*

   Possible answers are: 0, 1, 2, 4

Question 2

Assume timer/counter 0 (if key1 == 0) or 1 (if key1 == 1).

Assume a prescaler of 1 (if key2 == 0), 8 (key2 == 1), 64 (key2 == 2), 256 (key2 == 3) or 1024 (key2 == 4).

1. (5 pts) What is the interval between counts of the timer/counter (tocks)?

   $t\ (sec/tock) = \frac{p\ ticks/tock}{16{,}000{,}000\ ticks/sec}$

   So:

key2	**p**	**t**
0	1	0.0625 μs
1	8	0.5 μs
2	64	4 μs
3	256	16 μs
4	1024	64 μs

2. (5 pts) Assume that we have the overflow interrupt enabled. What is the frequency of interrupts?

   $f\ (int/sec) = \frac{16{,}000{,}000\ ticks/sec}{p\ ticks/tock \times x\ tocks/int}$

   So:

key1	**key2**	**x**	**p**	**f**
0	0	256	1	$62,500\ Hz$
0	1	256	8	$7,812\ Hz$
0	2	256	64	976.6 Hz
0	3	256	256	244.1 Hz
0	4	256	1024	61 Hz
1	0	256^2	1	244.14 Hz
1	1	256^2	8	30.52 Hz
1	2	256^2	64	3.81 Hz
1	3	256^2	256	0.95 Hz
1	4	256^2	1024	0.24 Hz

Question 3

Suppose we want to produce a regular interrupt every $262ms$. Assume that we are using a 16 MHz crystal for our clock.

1. (5 pts) Which timer should we use?

   Timer 1.

2. (5 pts) Which prescaler should we use?

   Prescaler: 64

Question 4

1. (15pts) Suppose we want a function – called $myfunc()$ – to be executed once every 20.97s. Assume a system clock of 16MHz. What is the timer1 prescaler configuration and the code for the interrupt routine (the code does not need to be syntactically correct)? Also - show the code in your main function that configures the timer.

   We will use a prescaler of 1024. This gets us down to an interrupt every 4.19 s. We then need an interrupt routine with an additional counter that expires at 5. So, we are left with an interrupt interval of: $5 * 1024 * 256 * 256/16000000 = 20.97s$.

```
ISR(TIMER1_OVF_vect) {
    static uint8_t counter = 0;

    ++counter;
    if(counter == 5) {
      myfunc();
      counter = 0;
   };
};
```

```
Somewhere in the main program:


// Interrupt occurs every
//      (1024*256*256)/16000000 = 4.19 sec
timer1_config(TIMER1_PRE_1024);
// Enable the timer interrupt
timer1_enable();
// Enable global interrupts
sei();
```

Question 5

Consider the following code:

```
ISR(TIMER0_OVF_vect) {
    static uint8_t counter = 0;
    static uint8_t phase = 0;

    if(counter == 0) {
        switch(phase) {
            case 0:
                PORTC |= 3;
                counter = 100;
                phase = 1;
                break;
            case 1:
                PORTC &= ~1;
                counter = 50;
                phase = 2;
                break;
            case 2:
                PORTC &= ~2;
                counter = 75;
                phase = 0;
                break;
        }
    }
    --counter;
};

Somewhere in the main program:

// Initialization
timer0_config(TIMER0_PRE_8);
// Enable the timer interrupt
timer0_enable();
// Enable global interrupts
sei();

DDRC = 0x3;

while(1)
{
}
```

1. (15 pts) Explain in detail what the program does. You are welcome to provide a picture.

   Produce two PWM signals at a frequency of $34.722Hz$ *on pins C0 and C1.*

   $f(cycle/sec) = \frac{16{,}000{,}000 \; ticks/sec}{8ticks/tock \times 256int/tock \times (100+50+75) \; int/cycle}$

   Both pins go high at the same type.

   C0 has a duty cycle of 44.44%.

   C1 has a duty cycle of 66.67%.