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

April 30, 2012

Question 1

Assume: timer 4 and a prescaler of 256.

1. (5 pts) What is the time period between counts of the timer/counter? Show your work

   $t\ (sec/tock) = \frac{256\ ticks/tock}{16{,}000{,}000\ ticks/sec} = 16\mu s$

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

   $f\ (int/sec) = \frac{16{,}000{,}000\ ticks/sec}{256\ ticks/tock \times 256^2\ tocks/int}$

   So:

   0.954 int/sec

Question 2

Assume that we are using a 16 MHz crystal for our clock.

1. (10 pts) Suppose that we want to produce an overflow interrupt period of 4.096 ms. Which timer should we use and with which prescaler?

   Timer 0 or 2

   Prescaler: 256

2. (5 pts) Suppose that we want to produce an overflow interrupt frequency of approximately $976Hz$. Which timer should we use and with which prescaler?

   Timer 0 or 2

   Prescaler: 64

Question 3

1. (15pts) Suppose that we want a function – called *control*1() – to be executed at approximately 30.5 Hz, and another function – called *control*2() – to be executed at approximately 2.77 Hz. We will use the timer1 overflow interrupt service routine to call both of these. 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 8. This gets us down to an interrupt frequency of 30.5176 Hz*, which is close to the desired control frequency. We then need an interrupt routine with an additional counter that expires at 11. So, we are left with an interrupt interval of:* $16000000/(256*256*8*11) = 1.7743\ Hz$.

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

    // Execute the control function each time
    control1();

    ++counter;
    if(counter == 11) {
      // Execute this function only once out of every 11 interru
      control2();
      counter = 0;
   };
};

Somewhere in the main program:
// Interrupt occurs at a frequency of
//      16000000/(256*256*8) = 30.5176 Hz
timer1_config(TIMER1_PRE_8);
// Enable the timer interrupt
timer1_enable();
// Enable global interrupts
sei();
```

Question 4

Consider the following code:

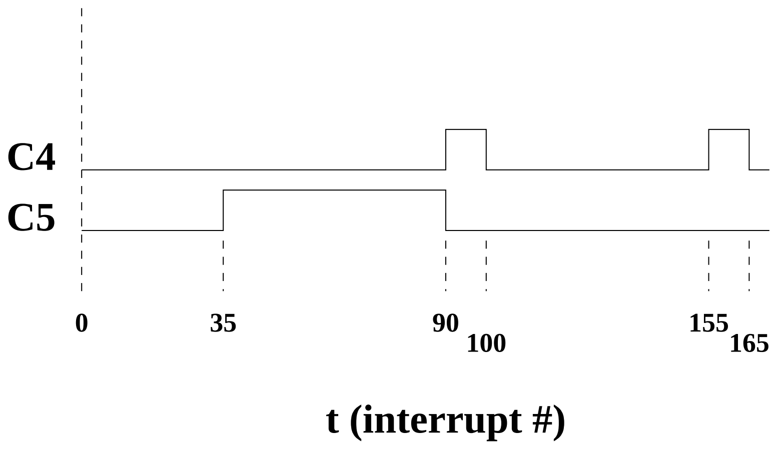
```
ISR(TIMER1_OVF_vect) {
    static uint8_t counter = 0;
    static uint8_t phase = 0;
    switch(phase) {
      case 0:
           if(counter == 35) {
              PORTC = PORTC & 0xCF | 0x20;
              counter = 0;
              phase = 1;
           }
           break;
      case 1:
           if(counter == 55) {
              PORTC = PORTC & 0xCF | 0x10;
              counter = 0;
              phase = 2;
           }
           break;
      case 2:
           if(counter == 10) {
              PORTC = PORTC & 0xCF;
              counter = 0;
              phase = 1;
           break;
        }
    }
    counter = counter + 1;
};

Somewhere in the main program:

// Initialization
timer1_config(TIMER1_PRE_256);
// Enable the timer interrupt
timer1_enable();
// Enable global interrupts
sei();

DDRC = 0x30;
PORTC = 0;
while(1){};
```

1. (15 pts) What is the timing diagram for C4 and C5?

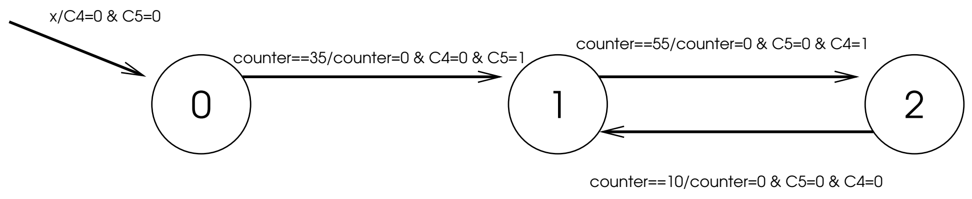
   This program results in an interrupt interval of:

   $f(sec/cycle) = \frac{256 ticks/tock \times 256^2 tock/int}{16{,}000{,}000\ ticks/sec} = 1.05 sec/int$

   After 36 interrupts (0...35), C5 goes high and is held there for 55 more interrupts. This pin is then turned off for the duration of the program. At interrupt 90 (the time at which C4 goes low), C5 enters into a periodic behavior of an interval of 165 interrupts and a duty cycle of 16.5%.



2. (15 pts) Draw the FSM diagram that represents the ISR functionality.


x/C4=0 & C5=0
counter==35/counter=0 & C4=0 & C5=1
counter==55/counter=0 & C5=0 & C4=1
counter==10/counter=0 & C5=0 & C4=0
0
1
2