Question 1

Assume that student\_ID is the number that corresponds to your student ID number.

1. (2pts) What is student\_ID \% 4? Call this key1

   Possible answers are: 0, 1, 2, 3

2. (2pts) What is student\_ID \% 5? Call this key2

   Possible answers are: 0, 1, 2, 3, 4
Question 2

Assume the timer/counter equal to your key1.

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 frequency of counting of the timer/counter?

\[
t (\text{t/socks/sec}) = \frac{16,000,000 \text{ ticks/sec}}{p \text{ ticks/tick}}
\]

So:

<table>
<thead>
<tr>
<th>key2</th>
<th>p</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>16 MHz</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>2 MHz</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>250 KHz</td>
</tr>
<tr>
<td>3</td>
<td>256</td>
<td>62.5 KHz</td>
</tr>
<tr>
<td>4</td>
<td>1024</td>
<td>15.625 KHz</td>
</tr>
</tbody>
</table>

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

\[
f (\text{sec/int}) = \frac{p \text{ ticks/tick} \times x \text{ ticks/int}}{16,000,000 \text{ ticks/sec}}
\]

So:

<table>
<thead>
<tr>
<th>key1</th>
<th>key2</th>
<th>x</th>
<th>p</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0</td>
<td>256</td>
<td>1</td>
<td>16 \mu s</td>
</tr>
<tr>
<td>0.2</td>
<td>1</td>
<td>256</td>
<td>8</td>
<td>128 \mu s</td>
</tr>
<tr>
<td>0.2</td>
<td>2</td>
<td>256</td>
<td>64</td>
<td>1024 \mu s</td>
</tr>
<tr>
<td>0.2</td>
<td>3</td>
<td>256</td>
<td>256</td>
<td>4096 \mu s</td>
</tr>
<tr>
<td>0.2</td>
<td>4</td>
<td>256</td>
<td>1024</td>
<td>16.384 ms</td>
</tr>
<tr>
<td>1.3,4</td>
<td>0</td>
<td>256</td>
<td>1</td>
<td>4.096 ms</td>
</tr>
<tr>
<td>1.3,4</td>
<td>1</td>
<td>256</td>
<td>8</td>
<td>32.768 ms</td>
</tr>
<tr>
<td>1.3,4</td>
<td>2</td>
<td>256</td>
<td>64</td>
<td>262.144 ms</td>
</tr>
<tr>
<td>1.3,4</td>
<td>3</td>
<td>256</td>
<td>256</td>
<td>1.0486 s</td>
</tr>
<tr>
<td>1.3,4</td>
<td>4</td>
<td>256</td>
<td>1024</td>
<td>4.1943 s</td>
</tr>
</tbody>
</table>
Question 3

Suppose that we want to produce an overflow interrupt frequency of 488 Hz. Assume that we are using a 16 MHz crystal for our clock.

1. (5 pts) Which timer should we use?
   Timer 2.

2. (5 pts) Which prescaler should we use?
   Prescaler: 128
Question 4

1. (15pts) Suppose that we want a function – called \texttt{control()} – to be executed approximately once every second, and another function – called \texttt{sense()} – to be executed approximately once every 5 minutes. We will use the timer1 overflow interrupt 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 256. This gets us down to an interrupt every 1.0486s, which is close to the desired control frequency (within 5%). We then need an interrupt routine with an additional counter that expires at 286. So, we are left with an interrupt interval of: $286 \times 256 \times 256 \times 256/16000000 = 299.8927s \approx 5\text{min}$.

\begin{verbatim}
ISR(TIMER1_OVF_vect) {
    static uint16_t counter = 0; // Must be 16 bits

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

    ++counter;
    if(counter == 286) {
        // Execute this function only once out of every 286 interrupts
        sense();
        counter = 0;
    }
}

Somewhere in the main program:
// Interrupt occurs every
// $(256 \times 256 \times 256)/16000000 = 1.0486$ sec
timer1_config(TIMER1_PRE_256);
// Enable the timer interrupt
timer1_enable();
// Enable global interrupts
sei();
\end{verbatim}
Question 5

Consider the following code:

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

    if (counter == 0) {
        switch (phase) {
            case 0:
                PORTC = PORTC & 0xFC | 1;
                counter = 75;
                phase = 1;
                break;
            case 1:
                PORTC = PORTC & 0xFC | 2;
                counter = 100;
                phase = 2;
                break;
            case 2:
                PORTC = PORTC & 0xFC;
                counter = 25;
                phase = 0;
                break;
        }
    }
    --counter;
}

Somewhere in the main program:

    // Initialization
    timer1_config(TIMER1_PRE_64);
    // Enable the timer interrupt
    timer1_enable();
    // Enable global interrupts
    sei();

    DDRC = 0x3;
    PORTC = 0;

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

This program produces two PWM signals at a frequency of 0.0191 Hz on pins C0 and C1.

\[ f(\text{cycle/sec}) = \frac{16,000,000 \text{ ticks/sec}}{64 \text{ticks/tick} \times 256^2 \text{ticks/int} \times (75+100+25) \text{int/cycle}} \]

C0 is high for the first 19.66 s (a duty cycle of 37.5%). After C0 is turned off, C1 is high for 26.21 s (a duty cycle of 50%). Then, both are off for 6.55 s.

![Diagram of PWM signals on pins C0 and C1 with time ticks 0, 19.7, 45.9, 52.4]