MEAM 5100: Laboratory Assignment-1

Student Saurav Agrawal
ID 58987928
Professor Dr.Mark Yim

1.1.1 Choose an LED and a resistor from the ministore. What ranges of resistance values will work well and why? [Hint: use the LED datasheets from Digikey.] Use one of the three LED's listed below available in the GM lab with Digikey.com part number and description:
160-1133-ND 5mm yellow LED, LTL-4253
160-1128-ND 5mm red LED, LTL-4224
160-1709-ND LED 3MM YELLOW TRANSPARENT, LTL-1CHYE
Calculate the smallest value resistor that you can safely use with the chosen LED in series with 5V power. Submit a copy of the page from the datasheet you used highlighting the specification you used to determine this value along with the calculation you used to get it. The lecture on LED's will be relevant here.

I have taken the red LED: 160-1128-ND 5mm red LED, LTL-4224

Electrical / Optical Characteristics at T_A=25℃

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Condition
Luminous Intensity	I_V	29	90		mcd	I_F = 10mA Note 1,4
Viewing Angle	$2\theta_{1/2}$		16		deg	Note 2 (Fig.6)
Peak Emission Wavelength	λ_P		635		nm	Measurement @Peak (Fig.1)
Dominant Wavelength	λ_d		623		nm	Note 3
Spectral Line Half-Width	$\Delta\lambda$		40		nm	
Forward Voltage	V_F		2.0	2.6	V	I_F = 20mA
Reverse Current	I_R			100	μA	V_R = 5V
Capacitance	C		20		pF	V_F = 0 , f = 1MHz

Figure 1: Specification highlighted used in calculations

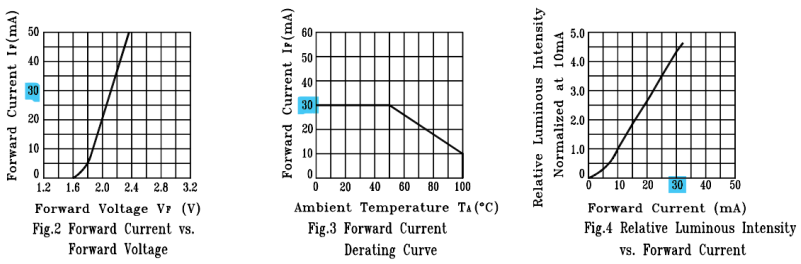

Figure 2: Specification highlighted used in calculations

Observing the data from the data sheet, to find the smallest resistor that can be used to find the we can assume the forward current to be close to 30 mA. looking at the absolute continuous forward current table we found that maximum rating is 30 mA. Since in our lectures we were advised not be close to the maximum ratings that are

found in the datasheet. Taking into consideration we will choose the current to be 20mA. (which is used in the test conditions of the optical characteristics table)
Using I = 0.020 A for my calculations We have a 5 voltage power supply
Considering forward voltage of 2 V
so 5 = 2 (forward voltage) - I *(resistance)
R = 150 ohms
Now considering forward voltage of 2.6 V
so 5 = 2.6 (forward voltage) - I *(resistance)
R = 120 ohms
Therefore resistance can be in the range of 120-150 ohms (note: Since the graphs helps us observe forward current to be 30 ohms we can using 30 as current the range of resistance would be 80-100 0hms)

1.1.2 Use a breadboard and hook up the LED you chose in series with a resistor that has a value close to but higher than the value you calculated above. Connect it to 5V and ground and observe the brightness and verify that the LED does not get too hot, smoke or explode. Increase the resistance (using more resistors in series or resistors with larger values) and find what value of resistance will give a very dim but still visible light. Submit the value of resistance you observed and calculate the current that gives this dim light.
THe lowest value found in the above calculation is 120 ohms. so have taken 2 resistors of 100 ohm and 22 ohms in series to get 122 ohms resistance. We got current flowing to be 21 mA at constant voltage of 5 V proving the above work. Using multiple resistors in series I was able to see light till the resistance of 509 K ohms.

1.1.3 Take any two LED's with two different colors different from the ones you used in 1.1.1. Put them in parallel with a 330 ohm resistor as in the figure below. Measure the voltage at point A. Submit what you observe: include which LED's you chose; which LED would you expect to be brighter based on the specifications; which LED is brighter; what values on the two different datasheets for each LED explains the voltage at point A?
As the other two LED's form the sheet given have same color so we cannot clearly distinguish between the two of the intensity of the light visually. So for this part I have used a Yellow LED with the red one mentioned in the 1.1.1 question. Form that it can be observed that the red LED is brighter compared to the yellow LED. looking at the typical forward voltages of Red and yellow LED we can find it to be 2V and 2.1V respectively. Since Red had the lower Forward voltage Red will have higher intensity compared to the yellow. The voltage at point A is 1.91 V with respect to ground which is seen in the DMM. Since the typical forward voltage of red is closer to the voltage at point A so the red will have a brighter light compared to yellow.

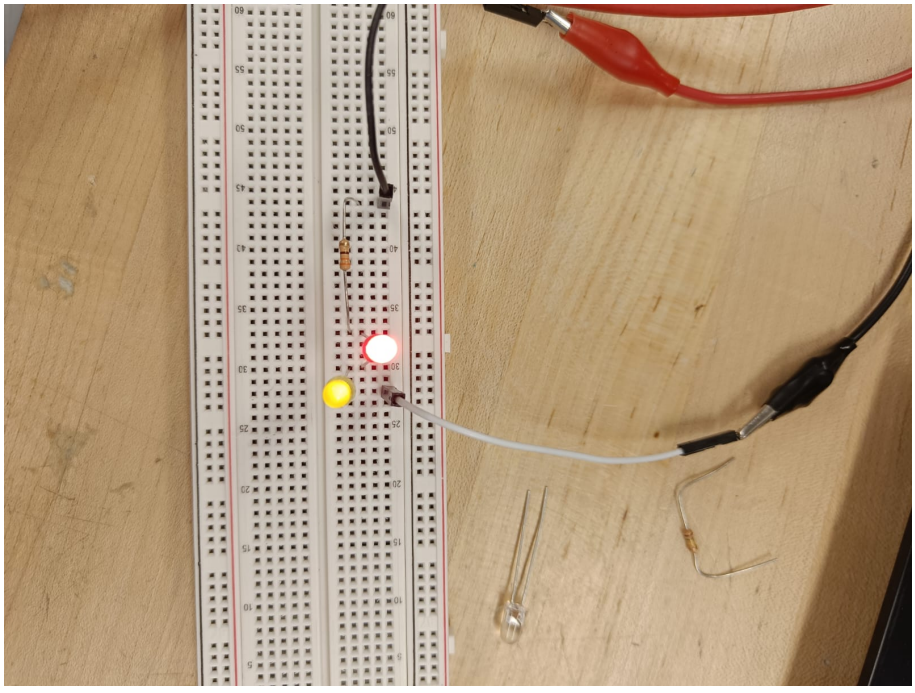

Figure 3: Brightness of 2 LED's in parallel

1.1.4 Take the same circuit, move LED1 to be in series with LED2. Increase the power supply voltage if necessary to get them to turn on. Measure the voltage at point A and the point between the two LED's. Submit what you observe: include the values you read and the values you expect based on the datasheet.

The circuit contains a resistor of 330 ohms followed by Yellow LED and Red LED in the end.
Voltage at point A (Resistor and Yellow LED) = 3.75 V
Voltage at point between 2 LED's(yellow LED and Red) = 1.86 V
Further we try to calculate the potential difference between point A and point between LED's. We find the difference to be 1.89 V which is the forward potential for yellow LED and 1.86 V is the forward voltage for Red LED. It is observed that both the yellow light and Red Light have similar brightness. Using the data sheet we found that they have similar intensities at the forward voltages seen in the DMM which confirms the visual observation.

1.1.5 Pretend you are designing an LED powered light source. This will be like a desk lamp (wide area (not focused beam). Go to www.digikey.com Find the brightest or cheapest or smallest that suits your needs for this lamp. Submit a write up that describes the reason you chose it, and how it is optimal (either bright, cheap or small). Full points if the TA's cannot find a more optimal solution in less than 4 minutes.

A desk lamp should be bright and economical. For these reasons I would use LED 1416-L1RX-LME1000000000TR-ND - Tape Reel (TR) (Digi key part number) for my Lamp. It has 501 lm intensity which is very high. THe cost of LED unit is 0.9 dollars which is very cheap and it also serves lime light good for lamp.

1.2.1 What registers are you changing and why?

I am changing PortB 6 register. I am making it active. PortB pin 6 is a GPIO.

1.2.2 Attach an LED from the ministore to this pin with appropriate additional hardware. Submit a drawing of your circuit using circuitlab and justify any component values you use

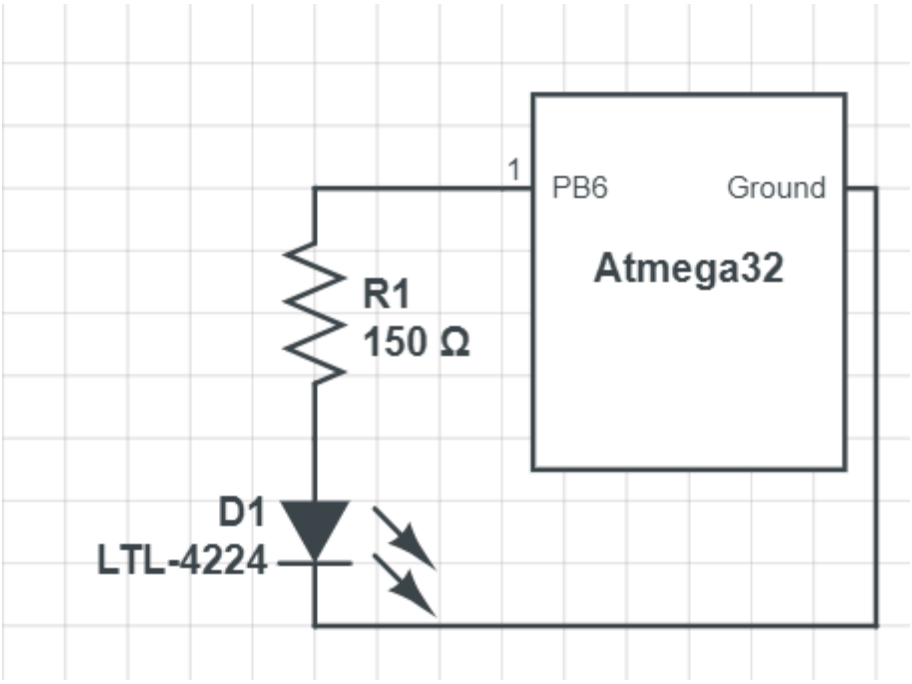

Figure 4: Circuit diagram

We have used the a resistance of 150 ohms and LED LTL 4224 (red LED). Since the current limit per pin is

40 mA. SO to be on the safe side using the the data from question 1.1.1 I have taken Resistor as 150 ohms and Red LED so that it draws approx 20 mA current across 5 V.

1.2.3 Using the delay ms() routine, create code that will blink an LED (you may modify the Blinky example to start with). Submit your (well commented) C code for this portion

```
/* Name: main.c
 * Author: Saurav Agrawal
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file

int main(void)
{
    float duty_cycle; //initialize duty cycle variable
    float time_period; // initialize time period
    time_period = 1000; //Set time period
    duty_cycle = 20; // set duty cycle percentage

    _clockdivide(0); //set the clock speed to 16Mhz
    set_led(ON); // turn on the on board LED
    _delay_ms(1000); // wait 1000 ms when at 16 MHz
    float on_time = (duty_cycle*time_period/100); /* calculate the time LED stays on
                                                    using duty cycle and time period*/
    float off_time = time_period - on_time; // calculate off time using time period and on time
    /* insert your hardware initialization here */
    DDRB = 0x40; //Set the direction pin to Port B6

    for(;;){
        //set_led(TOGGLE); // switch the led state
        toggle(PORTB,6);        // Toggle the state of the LED pin
        _delay_ms(on_time); // keep it on for the on time
        toggle(PORTB,6);        // Toggle the state of the LED pin
        _delay_ms(off_time); // delay for time period off
    }
    return 0;   /* never reached */
}
```

1.2.4 Create a variable in your code so you can easily change the duty cycle (100 percent duty cycle is always on, 0 percent duty cycle is always off, 50 percent is half on half off). Verify with an oscilloscope that the duty cycle changes appropriately. Ensure your code handles all cases including the special cases of 0 percent where the LED is completely off and 100 percent where it is completely on. Submit your (well commented) C code for this portion.

```
#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file

int main(void)
{
    float duty_cycle; //initialize duty cycle variable

    float time_period; // initialize time period

    time_period = 1000; //Set time period
```

```
    duty_cycle = 80; // set duty cycle percentage


    _clockdivide(0); //set the clock speed to 16Mhz
    set_led(ON); // turn on the on board LED
    _delay_ms(1000); // wait 1000 ms when at 16 MHz

    float on_time = (duty_cycle*time_period/100); // calculate the time LED stays on
    float off_time = time_period - on_time; // calculate off time using time period and on time

    /* insert your hardware initialization here */
    DDRB = 0x40; //Set the direction pin to Port B6




    for(;;){

        toggle(PORTB,6);        // Toggle the state of the LED pin

        _delay_ms(on_time); // keep it on for the on time

        toggle(PORTB,6);        // Toggle the state of the LED pin

        _delay_ms(off_time); // delay for time period off
    }
    return 0;   /* never reached */
}
```

1.3.1 Instead of delay functions, use a timer of the ATmega32U4 to get an LED to blink at 20Hz. (Note: don't use OCR registers which we will use later). Verify the driving frequency on a scope. Submit a photo of the oscilloscope screen showing the 20Hz signal and the code that generates this

```
 /* Name: main.c
 * Author: Saurav Agrawal
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */
#include <avr/io.h>
#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file
# define compval 1562 /* Since we have prescaled the timer now
                       we have have to find compvalue for 20 hz*/
/*
for 20 hz 1 cycle takes 1/20 th of second and for 62.5 Khz it takes 1/62.5 Khz to find
* counter value we need to devide 1/20 to 1/62500 to get number of cycles counter.
* Since we are slowing the clock cycle as well by factor of 256 we need to divide by 256 too.
* Since it is acomplete cycle we need half time on and half time off so we
* divide the answer got form above by 2.
*/
int main(void)
{



    DDRB = 0x40; // port B6 as output
    set(TCCR3B,CS32);     // To set pre scaler to 62.5 Khz that is divide by 256
    clear(TCCR3B,CS31);    // To set pre scaler to 62.5 Khz that is divide by 256
    clear(TCCR3B,CS30);   // To set pre scaler to 62.5 Khz that is divide by 256
```

```
    TCNT3 =0;            // initialize the timer counter to 0


    /* insert your hardware initialization here */
    for(;;){
        /* insert your main loop code here */
        if (TCNT3 >compval)  //compare the counter to comp value
        {
            toggle(PORTB,6);   // toggle when the TCNT3 value is greater than compvalue
            TCNT3 =0; //Reset timer to 0
        }

    }
    return 0;   /* never reached */
}
```

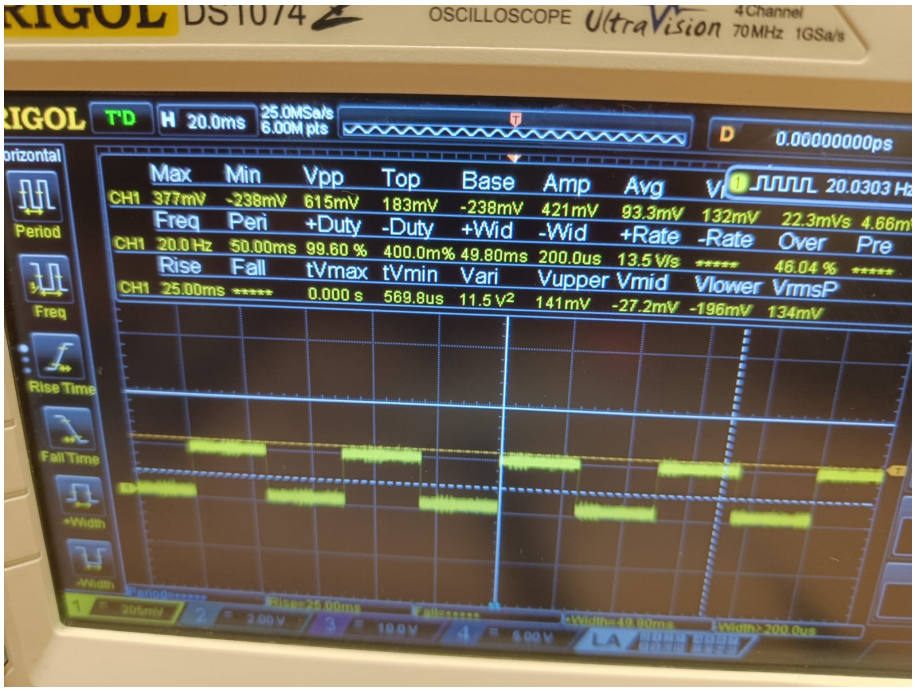

Figure 5: Ossiloscope showing 20 Hz frequency

1.3.2 Using the timer registers, adjust the system clock pre-scaler. (Note: don't use OCR registers which we will use later). Does the output change as you would expect, why or why not? What is the default system clock frequency? Submit your answers

The system clock prescaler reduces the system clock speed by2 power N where N belongs to (0-8). The system clock slows the whole system by the factor. This is generally used when we want to work with lower frequency and are not using high calculations. Even the reset key slows down and we have to give gap between 2 presses to reset the code. I tried clock prescaler to solve question 1.3.1 and found that we can use a very low compare value. for 1.3.1 the compare value calculate was 1526 whereas the same effect can be seen with compare value of 7. I am also attaching the code for the reference. We can use clockdivide function to slw the system.

```
 /* Name: main.c
 * Author: <insert your name here>
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file
# define compval 7  //
```

```
/*
* for 20 hz 1 cycle takes 1/20 th of second and for 62.5 Khz it takes 1/62.5 Khz to find
* counter value we need to devide 1/20 to 1/62500 to get number of cycles counter.
* Sice the clock speed is slowed by a factor of 256 Since it is acomplete cycle we need half
* time on and half time off so we divide the answer got form above by 2.
*/

int main(void)
{


    _clockdivide(8); //set the clock speed to 62.5Khz
    DDRB |= 0x40; // port B6 as output
    set(TCCR3B,CS32);    // To set pre scaler to 62.5 Khz that is divide by 256
    clear(TCCR3B,CS31);   // To set pre scaler to 62.5 Khz that is divide by 256
    clear(TCCR3B,CS30);   // To set pre scaler to 62.5 Khz that is divide by 256
    TCNT3 =0;          // initialsie the counter


    /* insert your hardware initialization here */
    for(;;){
        /* insert your main loop code here */
        if (TCNT3 >compval)  //compare the counter to the compare value defined above
        {
            toggle(PORTB,6);  // toggle when the TCNT3 value is greater than compvalue
            TCNT3 =0; //Reset timer to 0
        }

    }
    return 0;   /* never reached */
}
```

1.3.3 Use the PWM functions (the WGM modes of timer registers) of the ATmega32U4 timer to do the same as you did in 1.2.4. Explain which timer options you used for generating the PWM. Show that you can generate 0 and 100 duty cycle. Submit your (well-commented) code, and your explanation. Also submit a short video of the system working

```
     /* Name: main.c
 * Author: Saurav Agrawal
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file

int main(void)
{
    DDRC |= 0x40; // port C6 as output
    float duty_cycle;  // initialise duty cycle
    duty_cycle = 50;   // provide a value to duty cycle
    set(TCCR3B,CS32);    // set prescaler /256
    clear(TCCR3B,CS31);   // set prescaler /256
    clear(TCCR3B,CS30);    // set prescaler /256


    set(TCCR3B,WGM32);   // for time mode 5
    set(TCCR3A,WGM30);    // for time mode 5
    clear(TCCR3B,WGM33);   // for time mode 5
```

```
    clear(TCCR3A,WGM31);    // for time mode 5

    int pmm = (duty_cycle*255/100);
    /* Calculate the integer value for counter using duty cycle.
    Since the timer is only 255 counts and duty cycle in percentage the pmm
    = duty cycle *255*0.01 */
    set(TCCR3A,COM3A1);    // to clear at OCR3A and set at rollover
    clear(TCCR3A,COM3A0);  // to clear at OCR3A and set at rollover

    OCR3A = pmm; // set OC3Ra as pmm

 for(;;){ } //endless loop
 return 0;

}
```

Figure 6: Duty cycle of 75 percentage seen

1.4.1 Create code to make the external LED (from part 1.2) pulse. The pulse should start at 0 intensity, take 0.5 seconds to increase in intensity until it is full brightness, then 0.5 seconds to decrease in brightness and repeat.

```
    /* Name: main.c
 * Author: Saurav Agrawal
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file


int main(void)
{
    DDRC |= 0x40; // port C6 as output set pin as output pin
   clear(TCCR3B,CS32);     // set prescaler by 64
    set(TCCR3B,CS31);       // set prescaler by 64
```

```
    set(TCCR3B,CS30);      // set prescaler by 64

    set(TCCR3B,WGM32);   // for time mode 5
    set(TCCR3A,WGM30);    // for time mode 5
    clear(TCCR3B,WGM33);   // for time mode 5
    clear(TCCR3A,WGM31);    // for time mode 5

    set(TCCR3A,COM3A1);    // to clear at OCR3A and set at rollover
    clear(TCCR3A,COM3A0);  // to clear at OCR3A and set at rollover

 while(1)
 {
    for(int i=0; i<=255;i++) //for loop for increasing brightness
    {
        OCR3A = i;  // counter
        _delay_ms(1.2);
         /*Since we have divided the time of 0.3 s equall so we get 0.3/255
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
    for(int i =255;i>=0;i--) //for loop for decreasing brightness
    {
        OCR3A =i; // counter
        _delay_ms(2.4);
        /*Since we have divided the time of 0.6 s equall so we get 0.6/255
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
 }
 return 0;

}
```

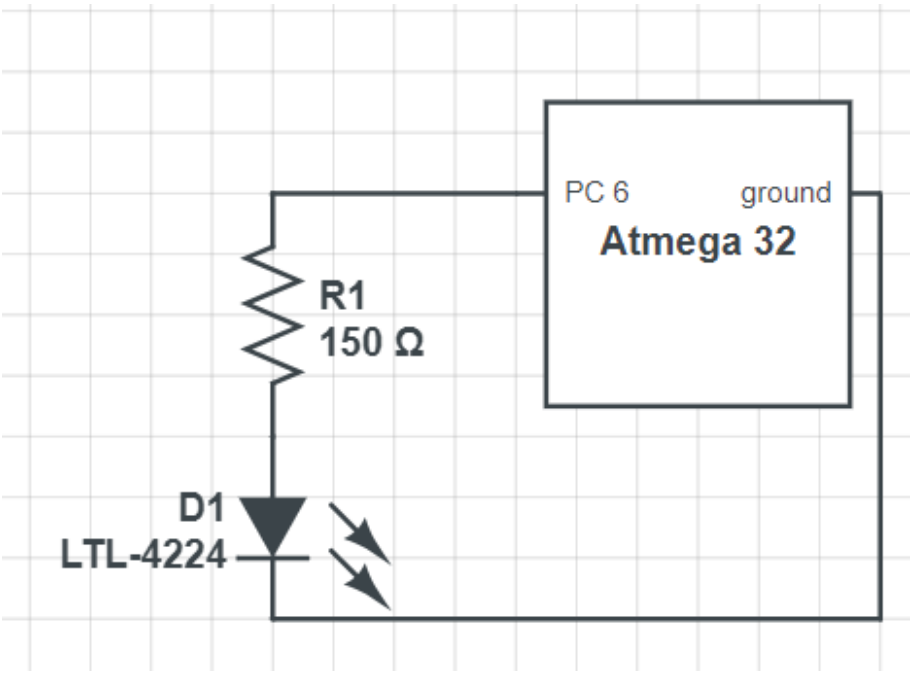

Figure 7: Circuit Diagram

1.4.2 Use subroutines so that the pulsing of the LED with variable increasing and decreasing intensity is easy to call from another routine. Modify the code above so that the maximum intensity can also be

hanged. Create a routine that causes the LED to blink as if it were a heartbeat (e.g, large then small intensity – ala lub dub). That is the LED percentage intensity i should follow the pattern below at time t seconds but smoothly interpolated intensities between each value: t=0 i = 0 t=0.1 i = 100 t=0.5 i = 0 t=0.6 i = 50 t=1.0 i = 0 t=3.0 i = 0 t=3.1 i = 100 t=3.5 i = 0 t=3.6 i = 50 t=4.0 i = 0 Submit a video and your (well commented) C code for this portion. Look on Canvas for sample videos of what this should look like

```
    /* Name: main.c
 * Author: Saurav Agrawal
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file

void heartbeat1()  //creating a function hearbeat to keep track of step
{
    for(int i=0; i<=255;i++)
    {
        OCR3A = i;
        _delay_ms(0.4); // 0.1sec /255 *1000
        /*Since we have divided the time of 0.1 s equall so we get 0.1/255
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
    for(int i =255;i>=0;i--)
    {
        OCR3A =i;
        _delay_ms(1.6);  // 0.4sec /255 *1000
        /*Since we have divided the time of 0.4 s equall so we get 0.4/255
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
}

void heartbeat2()
{
    for(int i=0; i<=127;i++)
    {
        OCR3A = i;
        _delay_ms(0.78); // 0.1sec /127 *1000
        /*Since we have divided the time of 0.1 s equall so we get 0.1/127
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }

    for(int i =127;i>=0;i--)
    {
        OCR3A =i;
        _delay_ms(3.15);  // 0.4 sec/127*1000
        /*Since we have divided the time of 0.4 s equall so we get 0.4/127
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
    _delay_ms(2000);
}
```

```
int main(void)
{
    DDRC |= 0x40; // port C6 as output set pin C6 as aoutput pin
    clear(TCCR3B,CS32);     // set prescaler by 64
    set(TCCR3B,CS31);       // set prescaler by 64
    set(TCCR3B,CS30);      // set prescaler by 64

   set(TCCR3B,WGM32);   // for time mode 5
    set(TCCR3A,WGM30);    // for time mode 5
    clear(TCCR3B,WGM33);   // for time mode 5
    clear(TCCR3A,WGM31);    // for time mode 5


    set(TCCR3A,COM3A1);    // to clear at OCR3A and set at rollover
    clear(TCCR3A,COM3A0);  // to clear at OCR3A and set at rollover

 while(1)
 {
    heartbeat1();   // call function hearbeat1 in main
    heartbeat2();   // call  function hearbeat2 in main


 }
 return 0;

}
```

1.4.3 Modify the above code so that the heartbeat stays at a constant frequency, but the maximum intensity slowly fades as if the heartbeat is getting weaker. Have it beat 20 times before it is reduced to 0 intensity. Show a TA your LED blinking; answer his/her questions, and get signed off. (NOTE YOU MUST BE CHECKED OFF BY A TA BEFORE THIS PART WILL BE CONSIDERED FINISHED). Submit a short video and the commented code.

```
    /* Name: main.c
 * Author: Saurav Agrawal
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "MEAM_general.h"  // includes the resources included in the MEAM_general.h file

void heartbeat(int b) //creating a function hearbeat to keep track of step
{
    for(int i=0; i<=255;i++) // loop for increasing intensity
    {
        OCR3A = i/b;  // decreasing intensity so divided by b
        _delay_ms(0.4); // 0.1sec /255 *1000
        /*Since we have divided the time of 0.1 s equall so we get 0.1/255
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }

    for(int i =255;i>=0;i--) // loop for decreasing intensity
    {
        OCR3A =i/b;  // decreasing intensity so divided by b
        _delay_ms(1.6);  // 0.4sec /255 *1000
        /*Since we have divided the time of 0.4 s equall so we get 0.4/255
```

```
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
    for(int i=0; i<=127;i++)  // loop for increasing intensity for 50%
    {
        OCR3A = i/b; // decreasing intensity so divided by b
        _delay_ms(0.78); // 0.1sec /127 *100
        /*Since we have divided the time of 0.1 s equall so we get 0.1/127
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
     for(int i =127;i>=0;i--) //loop for decreasing intensity of 50%
    {
        OCR3A =i/b;  // decreasing intensity so divided by b
        _delay_ms(3.15);  // 0.4 sec/255 *1000
        /*Since we have divided the time of 0.4 s equall so we get 0.4/127
        * for each step. but it is in ms so we multiply by 1000 to get the time
        * to delay in milliseconds*/
    }
    _delay_ms(2000); // delay for 2 seconds
}

int main(void)
{
    DDRC |= 0x40; // port C6 as output set pin C6 as aoutput pin
    clear(TCCR3B,CS32);     // set prescaler by 64
    set(TCCR3B,CS31);       // set prescaler by 64
    set(TCCR3B,CS30);      // set prescaler by 64

    set(TCCR3B,WGM32);   // for time mode 5
    set(TCCR3A,WGM30);    // for time mode 5
    clear(TCCR3B,WGM33);   // for time mode 5
    clear(TCCR3A,WGM31);    // for time mode 5

    set(TCCR3A,COM3A1);    // to clear at OCR3A and set at rollover
    clear(TCCR3A,COM3A0);  // to clear at OCR3A and set at rollover

 while(1)
 {
    for(int j=1; j<=20;j++) //the beat should fade in 20 steps
    {
        heartbeat(j); //function calling
    }
 }
 return 0;

}
```

For videos demos of the questions go the URL:
https://www.youtube.com/playlist?list=PLI1MCD4tyefNgBUZaV8rZCJI8XEX5PSbW

Reference: For the assignment I have collaborated with Mr.Tejendra, Mr. Jalaj, Mr. Kevin.