3 - Microcontroller Function Generation:

/\*

\* sine\_wave.c

\*

\* Created: 10/6/2015 8:02:58 PM

\* Author: haorshao

\*/

#include <avr/io.h>

#include "m\_general.h"

#include<math.h>

double sin(double \_\_x);

#define pi 3.1415926

volatile float i;

int main(void)

{m\_clockdivide(3);//set the system clock frequency as 2MHz

clear(TCCR1B,CS12);//set prescaler to /64

set(TCCR1B,CS11);

set(TCCR1B,CS10);

OCR1A=100;//set the value of OCR1A as 100

set(TCCR1B,WGM13);//(mode 15) UP to OCR1A, PWM mode

set(TCCR1B,WGM12);

set(TCCR1A,WGM11);

set(TCCR1A,WGM10);

set(TCCR1A,COM1B1);//set at OCR1B, clear at rollover

set(TCCR1A,COM1B0);

set(TIMSK1,OCIE1A);//call an interrupt whenever (TCNT1 matches OCR1A)

sei();//enable global interrupt

set(DDRB,6);

while(1) {

if (i<OCR1A){

OCR1B=OCR1A/2.0+sin(i/OCR1A\*2\*pi)\*OCR1A/2.0;//set the value of OCR1B

}else{

i=0;

}

}

}

ISR(TIMER1\_COMPA\_vect){

i++;

}

The basic idea is to generate a sine wave by passing the square output waveform from the microcontroller trough a low pass filter. I changed the duty cycle of the PWM in sine fashion, so I can get a sine waveform at the filter output. The duty cycle directly corresponds to the amplitude of the wave in that division. Divide the sine wave into OCR1A divisions. Use an interrupt to increase the value of i per unit and use a while loop to calculate the value of OCR1B. The value of OCR1B is calculated by a sine function and adding a constant which is related to OCR1A. Use Timer1 to control the frequency of the square waveform.