



รายงาน

Mini Project Phase 1

วิชา ไมโครโพรเซสเซอร์และระบบคอมพิวเตอร์แบบฝังตัว

(Microprocessor and Embedded Computer System)

เสนอ

อาจารย์ วังระ ภัคมาตร์

จัดทำโดย

นายโสภณ สุขสมบูรณ์ รหัสนักศึกษา 6201011631188

นักศึกษาชั้นปีที่3 สาขาวิชาวิศวกรรมไฟฟ้า (โทรคมนาคม)

วิชาไมโครโพรเซสเซอร์และระบบคอมพิวเตอร์แบบฝังตัว ประจำปีการศึกษา 2/2564

สาขาวิชาวิศวกรรมไฟฟ้า(โทรคมนาคม) ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

Pseudocode

Global

เริ่มต้น

- ประกาศตัวแปร l , j ,k และ m เป็นตัวแปรชนิด integer
- ประกาศตัวแปร finale_out เป็นตัวแปรชนิด integer แบบ 8 bit
- ประกาศตัวแปร signed_ADC , A[16] , B[16] , C เป็นตัวแปรชนิด float
- ระบุขนาดของ CPU สำหรับประมวลผล 16 MHz

adc_init function

- เปิดใช้งาน ADC mode และทำการ Pre-scale /128
- เลือกใช้ Vref จาก AVCC pin

Timer_init function

- เปิดใช้งาน Timer0 Overflow Interrupt mode
- กำหนดค่าเริ่มต้นเป็น 0xC2 สำหรับ delay 1 ms
- ทำการ Prescale /256

b_array_value function

- for loop
- i = 0 , i<15 , i++
- เช็คบิต EEPE ว่าพร้อมหรือไม่
- กำหนด addr สำหรับทำ LUT ที่ตำแหน่ง 0x10+i
- เก็บเลข 1 ไว้ใน EEPROM
- เปิดใช้งาน EEPROM write mode
- จบ

-for loop

- j=0 , j<15, j++

-เช็คบิต EEPE ว่าพร้อมหรือไม่

-กำหนด addr สำหรับดึงข้อมูลจาก LUT ที่ตำแหน่ง 0x10+i

-เปิดใช้งาน EEPROM read mode

- B[j]= EEDR

Main function

- DDRD เป็น output port

-DDRC เป็น input port

-เซ็ตให้ Port C บิตที่ 5 เป็น 1

-เรียกใช้ฟังก์ชัน adc_init()

-เรียกใช้ฟังก์ชัน b_array_value()

-เรียกใช้ฟังก์ชัน timer_init()

-เซ็ต interrupt flag (เปิดใช้งาน interrupt)

-while(1)

-จบ

ISR(TIMER_OVF_vect)

-กำหนดค่าเริ่มต้น 0xC2

-แปลง ADC จาก unsigned ให้เป็น signed number และเก็บค่าไว้ที่ signed_ADC

-เรียกใช้ฟังก์ชัน adc_conv()

-if (PINC5 = 0)

-for loop

-k=15,k>0,k—

-A[k]=A[k-1]

-จบ

-A[0]=signed_ADC

-C=0

-for loop

-m=0,m<16,m++

-C+=A[m]*B[m]

-จบ

-แปลงตัวแปร C จาก float ให้เป็น integerแล้วทำค่าไปเก็บที่ finale_out

-ส่งค่าจากตัวแปร finale_out ออกไปยังพอร์ต D

-else

-PORTD=0x00

โปรแกรมภาษาซีและการจำลองการทำงานบน Proteus

Code

```
/*
*miniproject_phase1.c
*
*Created:24/5/2565 8:34:45
*Author :ASAS
*/

#include <avr/io.h>
#include <avr/interrupt.h>
int8_t finale_out=0;
float signed_ADC;
float A[16],B[16];
float C;
#define F_CPU 16000000UL
int i,j,k,m;

void adc_init(void)
{
    ADCSRA=(1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0);//set ADC and Prescaler
    ADMUX |=(1<<REFS0);//set Vref =AVCC pin
}
void adc_convs(void)
{
    ADCSRA |=(1<<ADSC);//set ADC conversion
}
void timer_init(void)
{
    TIMSK0=(1<<TOIE0);//set Timer OVF interrupt
    TCNT0 =(1<<7)|(1<<6)|(1<<1);//set initial value is 0xC2 for delay 1 ms
    TCCR0A=0x00;
    TCCR0B=(1<<CS02);//set prescale is clk/256
}
void b_array_value(void)
{
    for(i=0;i<15;i++)
    {
        //do Look up table
        while(EECR&(1<<EEMPE));
        EEAR =0x10+i;//set address
        EEDR =1;//import data in EEPROM
        EECR |=(1<<EEMPE);
        EECR |=(1<<EEPE);//set write mode
    }
}
```

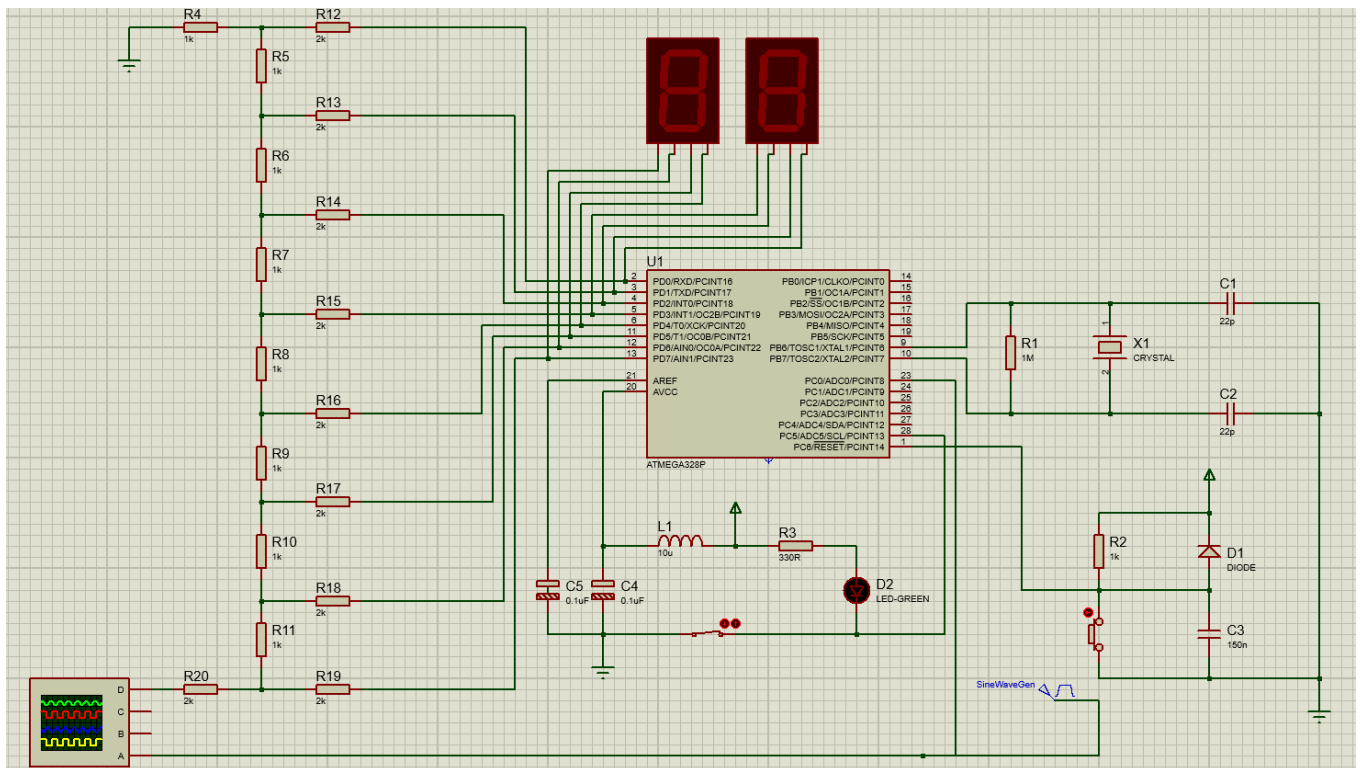
```

    for(j=0;j<15;j++)
    {
        //call data in EEPROM
        while(EECR&(EEPE));
        EEAR=0x10+j;
        EECR |=(1<<EERE); //set read mode
        B[j]=EEDR;
    }
}
int main(void)
{
    DDRD |=0xFF;
    DDRC |=0x00;
    PORTC |=(1<<5);
    adc_init();
    b_array_value0;
    timer_init();
    sei();
    while(1);
    return 0;
}
ISR(TIMER0_OVF_vect)//timer OVF interrupt
{
    TCNT0 =(1<<7)|(1<<6)|(1<<1);
    signed_ADC=((ADC/1023.0)*2.0)-1.0; //change unsigned into signed number
    adc_convs0;
    if((PINC&(1<<PINC5))==0)//PORTC.5 is high
    {
        for(k=15;k>0;k--)
        {
            A[k]=A[k-1]; //when new value comes dis old value
        }
        A[0]=signed_ADC; //put value from T_ADC into A[0]
        C=0;
        for(m=0;m<16;m++)
        {
            C+=A[m]*B[m]; //sum of A*B
        }
        finale_out=(int)(C*127/16); //change float into int
        PORTD=finale_out;
    }
    else PORTD=0x00;
}

```

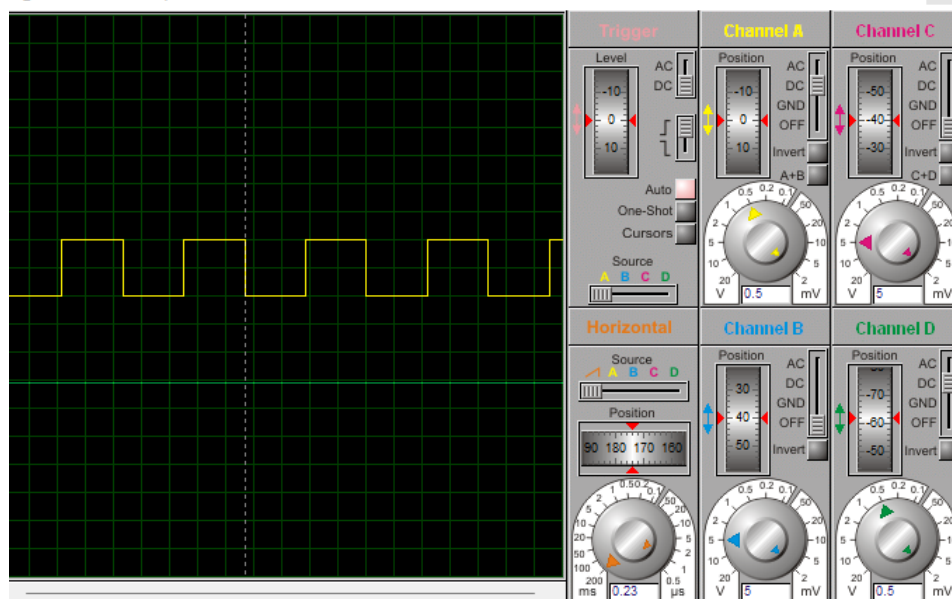
การจำลองการทำงานบน Proteus

Schematic



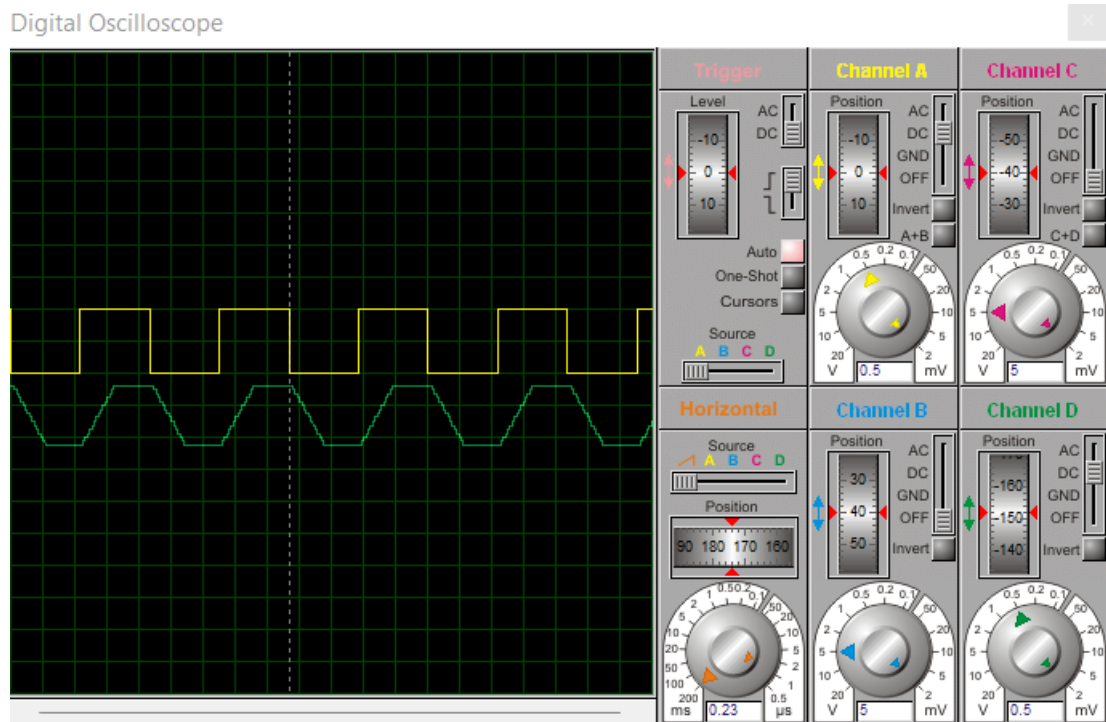
Display (Switch off)

Digital Oscilloscope



กรณีนี้เกิดจากการที่ไม่ทำการกด Switch ที่ต่อกับ PINC.5 และ AVCC ทำให้ไม่มีการทำ A/D Conversion

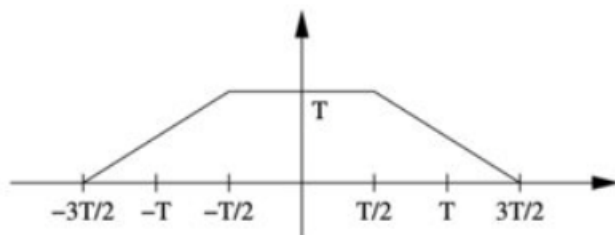
Display (Switch on)



อธิบาย

เนื่องจาก Input ของเราเป็น Impulse Train หรือ Rectangular Pulse เมื่อทำการคูณกับ 1 และทำการบวก เปรียบเสมือนการทำ Integration ซึ่งผลลัพธ์ของการทำ Integration ของ Rectangular function คือ

But the solution of the convolution $s(t)$ is known to be the function shown below:



(ขอขอบคุณภาพประกอบจาก : <https://math.stackexchange.com>)

ซึ่งกล่าวได้ว่า คำสั่งที่เราป้อนให้กับวงจรนี้ถูกต้องตามทฤษฎีนั่นเอง ซึ่งจากภาพเราจะสังเกตว่า ผลของการทำ D/A หรือ ผลลัพธ์ที่เราได้นั้น เกิดการคลาดเคลื่อนหรือดีเลย์จาก Rectangular Pulse เล็กน้อย ซึ่งเป็นผลมาจากระบบทำการคำนวณกินเวลาสักเล็กน้อย

