Generating PWM signals on GPIO pins of PIC Microcontroller

Submitted To : Qazi Mushtaq Ahmad

Submitted by : Group # 10

Mohammad Sameer Ishaq-190401054

ES LAB : Project Final

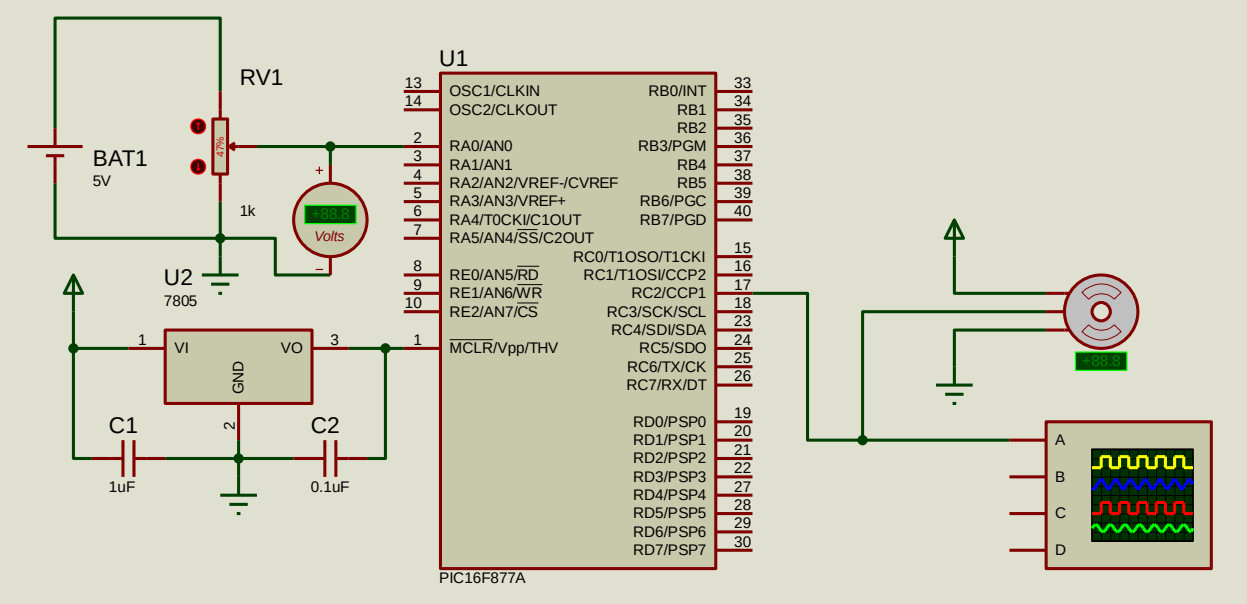
Objective:

To learn how the PWM signal and servo motor respond when the ADC value is changed through the potentiometer.

List of components:

1. Microcontroller-16F877A
2. Pot-Hg
3. Oscilloscope
4. Oscillator
5. Power +12V, 5V
6. Servo motor
7. Positive power supply regulator-7805

Circuit diagram:



Detailed explanation of the circuit:

The PIC16F877A can generate PWM signals only on pins RC1 and RC2, if we use the CCP modules. But we might encounter situations, where we need more pins to have PWM functionality. In these scenarios we can program the GPIO pins to produce PWM signals using timer modules. This way we can generate as many PWM signals with any required pin. So, this project we will help us to learn how to convert a PIC GPIO pin into a PWM pin and to test it we will simulate it on proteus with digital oscilloscope and also control the position of Servo motor using the PWM signal and vary its duty cycle by varying a potentiometer. So for this purpose we will use an analogue to digital converter function and transfer its output to the oscilloscope and servo motor.

Applications:

1. Controlling the position of servo motor

2. Switching few power electronic ICs in converters/invertors
3. Even for a simple LED brightness control

Shortcoming / Limitation:

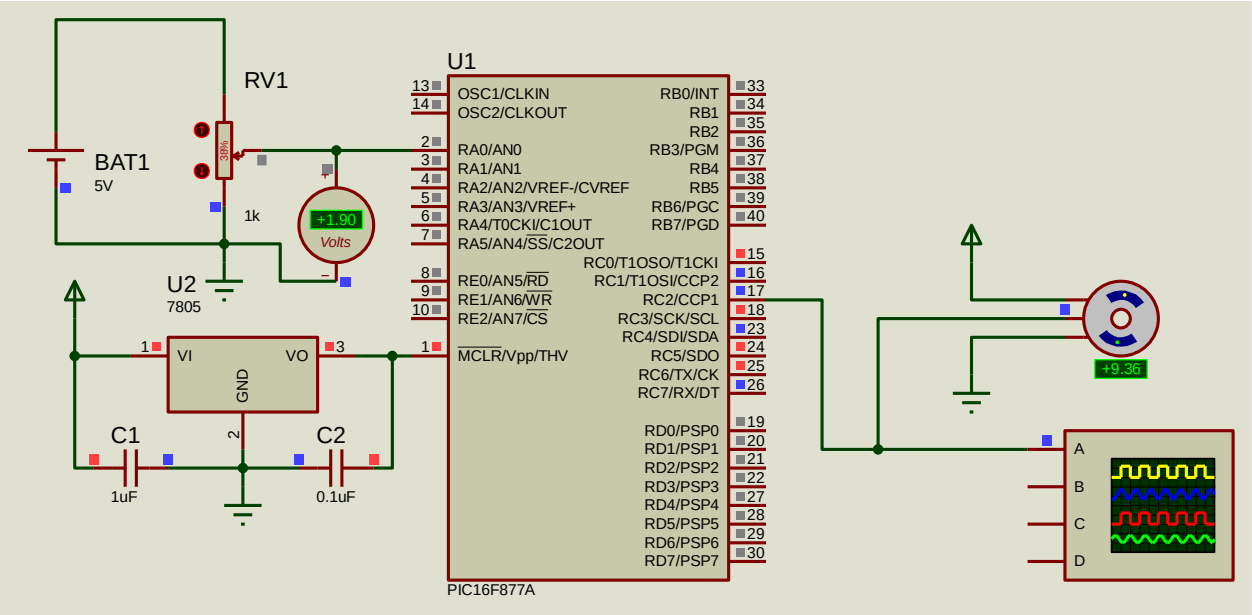
Electromagnetic Interference issues due to rise and fall of the current in PWM dimming.

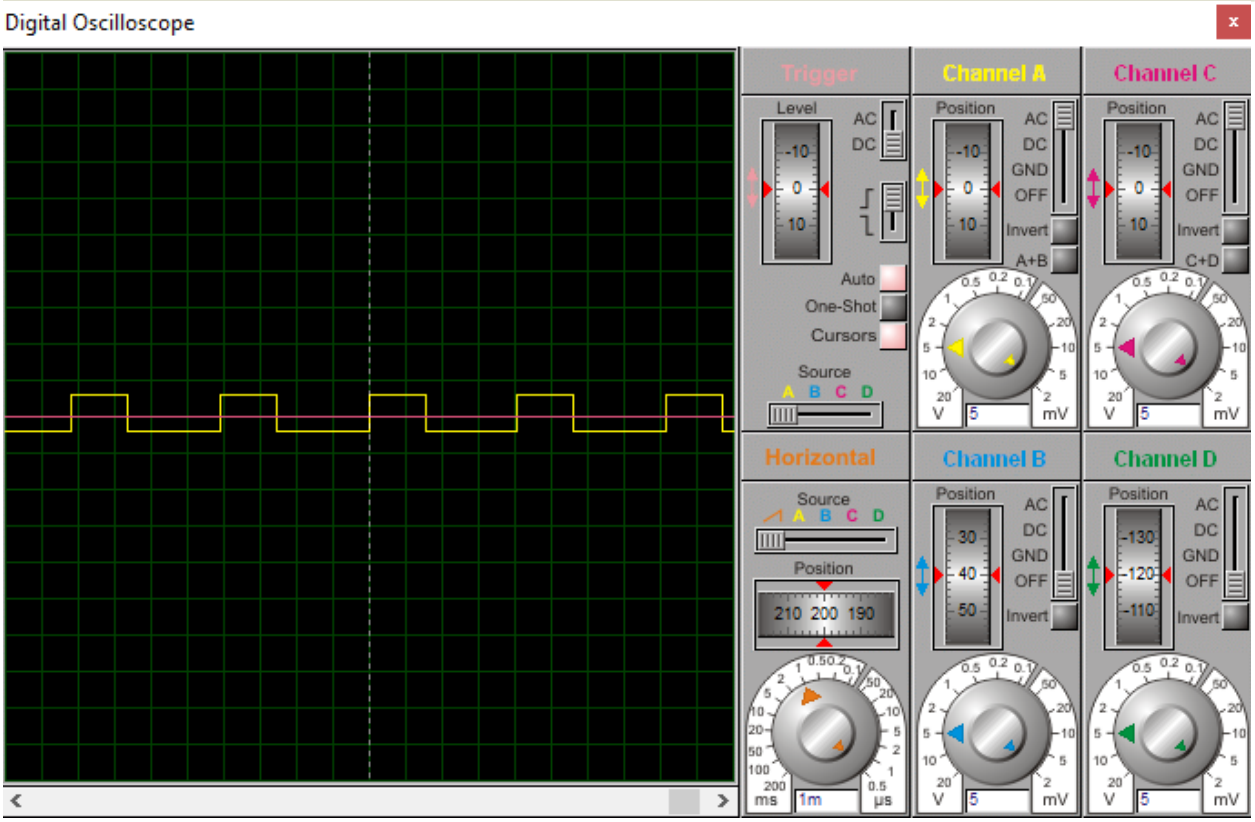
Code:

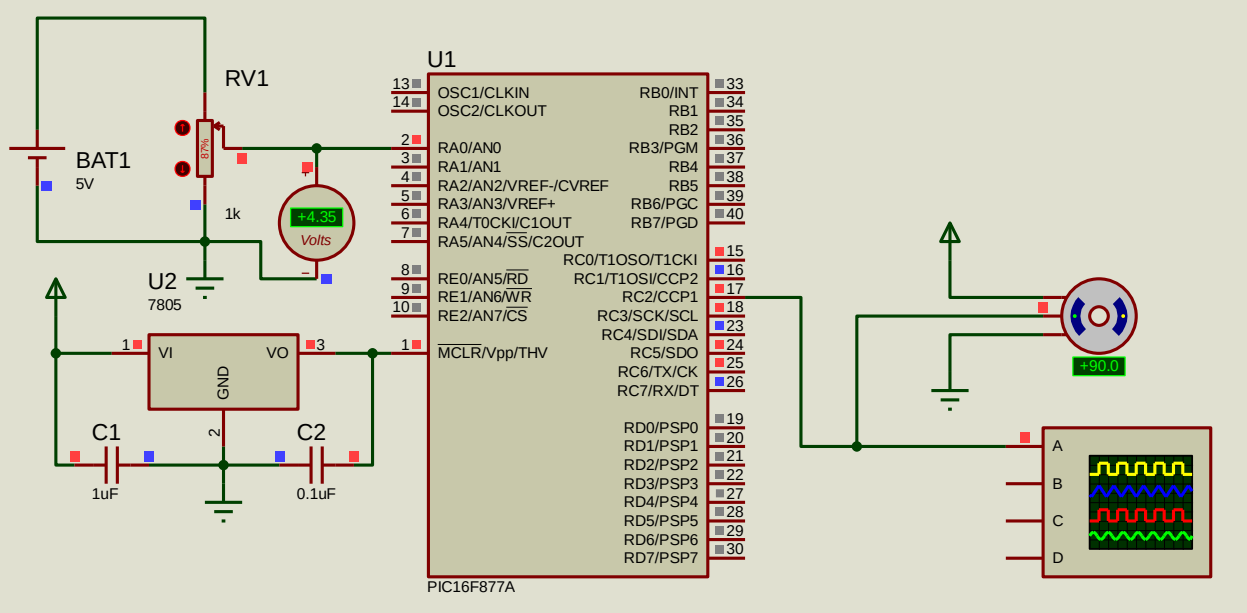
```
#include <16f877A.h>//header files
#fuses XT,NOWDT//configuration
#use delay(clock=4000000)//oscilator of 4MHz

#byte ADRESH=0x1E//Equating the ADRESH
#byte ADRESL=0x9E//Equating the ADRESL
#byte PORTA=0x05//Equating the PORTA
#byte PORTB=0x06//Equating the PORTB
#byte PORTC=0x07//Equating the PORTC
#byte PORTD=0x08//Equating the PORTD
#byte TRISA=0x85//Equating the TRISA
#byte TRISB=0x86//Equating the TRISB
#byte TRISC=0x87//Equating the TRISC
#byte TRISD=0x88//Equating the TRISD
#byte TMR2=0x11//Equating the TMR2
#byte ADCON0=0x1f//Equating the ADCON0
#byte ADCON1=0x9f//Equating the ADCON1
#byte T2CON=0x12//Equating the T2CON
#byte CCPR1L=0x15//Equating the CCPR1L
#byte CCP1CON=0x17//Equating the CCP1CON
#byte PR2=0x92//Equating the timer 2 module period register

#bit GO=0x1f.2      //bit of ADCON1 is go
#bit ON=0x1f.0      //bit of ADCON0 is on
#bit CCP1Y=CCP1CON.4
#bit CCP1X=CCP1CON.5
#bit PWM_PIN=TRISC.2

int RD_ADC(); //Prototype of a function

void main()   //starting main loop
{  float x=0; //defining float variable
   PR2=249;  //set periode frequency
   TRISC=0;  //making portc data direct register output
```

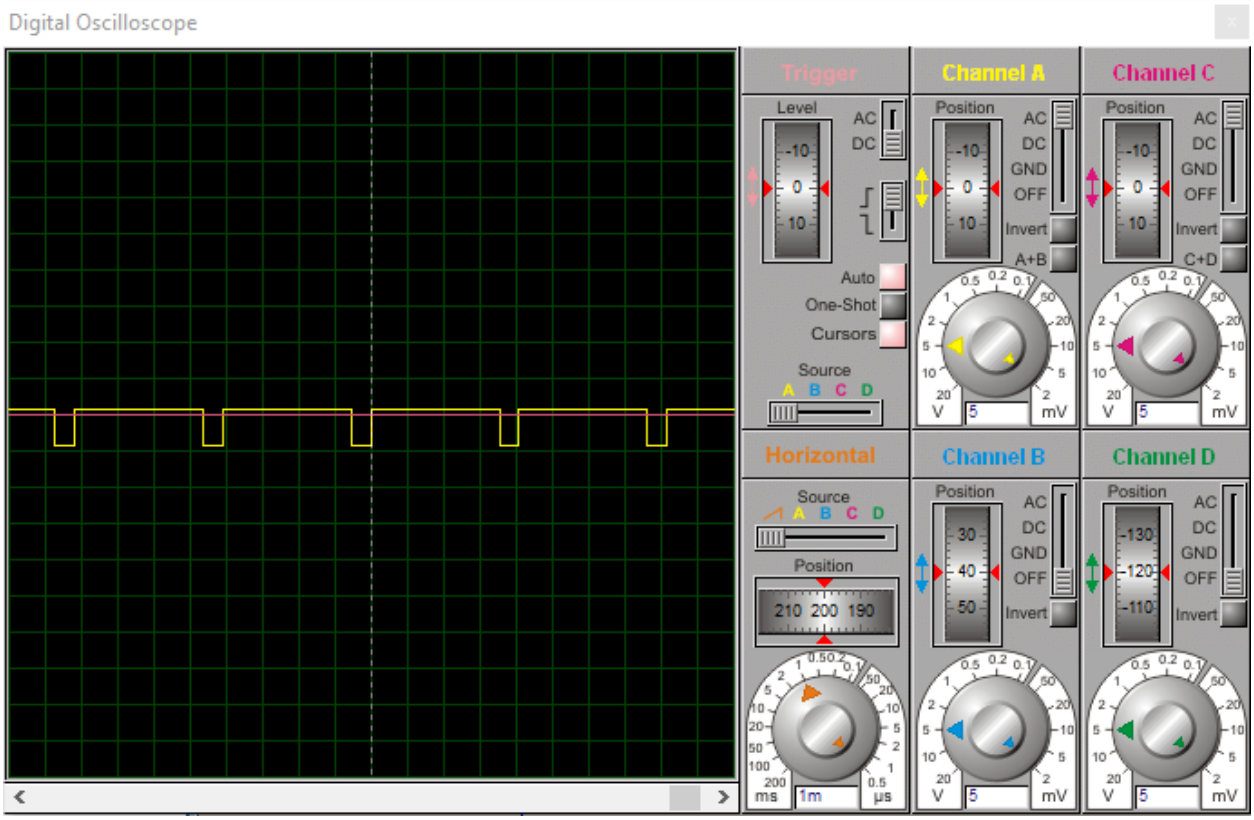
```
  ADCON0=0b01000000;//setting up clock of ADC
  ADCON1=0; //all the adcon1 bits are 0
  while(1)//infinite loop
  {
  x=RD_ADC(); // storing the returined value in x
  CCPR1L=x; //eight mode select bits//setting up the duty cycle
  CCP1Y=0; //LSbs of PWM duty cycle
  CCP1X=0; //LSbs of PWM duty cycle
  PWM_PIN=0;//set trisc,2 as output
  CCP1CON=0b00001100;   //Module mode
  T2CON=0b00000101;    //tmr2=on & prescaler=1:4
  }
}
int16 RD_ADC()      //function for A-D Conversion
{
int16 value=0;      //defining variable
bit_set(ADCON0,0); //setting the bit 0 which is on
bit_set(ADCON0,2);//setting the bit 1 which is go
while(bit_test(ADCON0,2)){} //while the adcon0 is working
value=make16(ADRESH,ADRESL);//storing the ad-result-high and low in value for 10 bits
return value; //returing the value
}
```

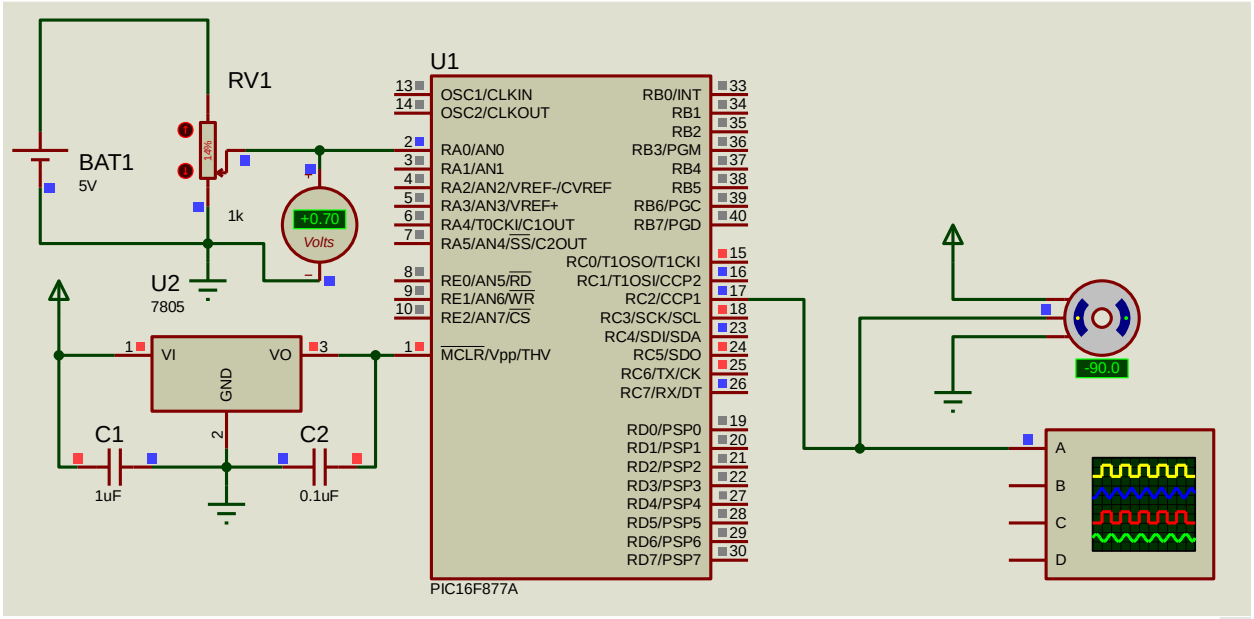
Simulations:






RV1
BAT1
5V
87%
1k
+4.35
Volts
U2
7805
VI
VO
GND
C1
1uF
C2
0.1uF
U1
OSC1/CLKIN
OSC2/CLKOUT
RA0/AN0
RA1/AN1
RA2/AN2/VREF-/CVREF
RA3/AN3/VREF+
RA4/T0CKI/C1OUT
RA5/AN4/SS/C2OUT
RE0/AN5/RD
RE1/AN6/WR
RE2/AN7/CS
MCLR/Vpp/THV
RB0/INT
RB1
RB2
RB3/PGM
RB4
RB5
RB6/PGC
RB7/PGD
RC0/T1OSO/T1CKI
RC1/T1OSI/CCP2
RC2/CCP1
RC3/SCK/SCL
RC4/SDI/SDA
RC5/SDO
RC6/TX/CK
RC7/RX/DT
RD0/PSP0
RD1/PSP1
RD2/PSP2
RD3/PSP3
RD4/PSP4
RD5/PSP5
RD6/PSP6
RD7/PSP7
PIC16F877A
+90.0
A
B
C
D


Digital Oscilloscope
Trigger
Channel A
Channel C
Horizontal
Channel B
Channel D
Level
Position
AC
DC
GND
OFF
Invert
A+B
C+D
Auto
One-Shot
Cursors
Source
210 200 190
V
mV
ms
µs



Digital Oscilloscope