LDR ile (PWM Kullanılarak) Power LED Kontrolü

LDR ( Light Dependent Resistor ) ışığa bağlı direnç demektir. LDR ışık fotonlarını emebilecek yüksek dirençli bir yarı iletken malzemeden yapılmıştır. LDR üzerine ışık düştüğünde yarı iletken malzeme üzerindeki serbest elektronlar yer değiştirirler.  Bu değişim LDR nin direncinin düşmesine neden olur. Yer değiştiren elektron sayısı ışık frekansına bağlıdır.

Çok karanlık bir ortamda LDR nin direnci yaklaşık 1Mohmdur(1 milyon ohm).  Üzerine ışık düştüğünde direnci birkaç kOhm’a kadar düşer (1kilo ohm= 1000 ohm).

Yaptığım devrede LDR karanlık ortamda iken direnci aşırı artacağından akım geçirmez hale gelir ve PIC in AN1 girişine yaklaşık 5v gelir. Bu da max. PWM duty cycle anlamına gelir ve power led en parlak haliyle yanar.

PWM biriminden önce, gerçek dünyadan bir veri alacağımız için ADC modülünü kullanmak zorundayız. LDR analog veri üreteceğinden ADC birimiyle aldığımız analog veriyi dijital veriye dönüştürmeliyiz ki bu veriyi donanımlarımızda kullanabilelim. Analog veri ADC birimi vasıtasıyla dijital veri olarak ilgili register’lara ( ADRESH , ADRESL ) kaydedilir. Sonra bu veriyi register lardan belirlediğimiz değişkenler atarız. Bu değişkenleri ise PWM Duty Cycle ını belirleme de kullanılırız.

PIC te PWM modulü TMR2 ile kullanılmaktadır. TMR2 kaydedicisi okunabilir ve yazılabilirdir. TMR2 Modülünün PR2 adında 8 bitlik bir kaydecisi vardır. İstediğimiz PWM periyodu için PR2 register ına gereken değeri yazdırırız.

PWM periyodunu bu formül vasıtasıyla ayarlayabiliyoruz. Ben programımda PWM periyodunu 1ms (1000 ) olarak ayarladım. PIC 16f877a’ da 2 adet PWM modülü bulunmaktadır. Yaptığım devrede CCP1 (RC2) çıkışını kullandım. PWM çıkışından alacağımız Duty Cycle ise bu formülle hesaplanır.

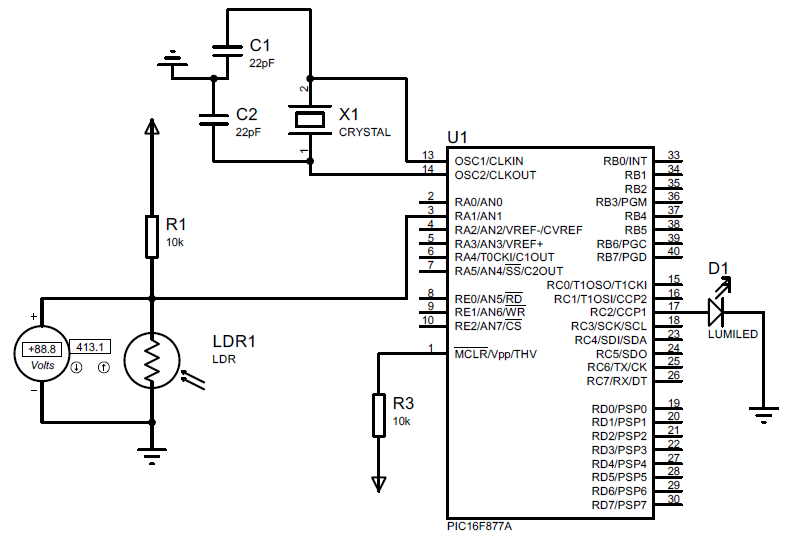
PWM Duty Cycle=[CCPRxL:CCPxCON<5:4>]\*Tosc\*[TMR2 Prescale değeri]

Analog veri sürekli bir şekilde değişeceğinden program bu veriyi dışarıdan okur ve dijitale çevirir. Elde edilen bu veri ile PWM in Duty Cycle ı altta yer alan register lara kaydedilerek kare dalga üretilmesi sağlanır.

CCPR1L : En yüksek değerlikli bitlerin kaydeğildiği register

CCP1Y , CCP1X : En düşük değerlikli iki bitin kaydedildiği registerlar. (5. ve 4. Bitler)

İSİS ŞEMASI



Şekil 1: İsis Şeması

#pragma config FOSC = XT // Oscillator Selection bits (XT oscillator)

#pragma config WDTE = OFF // Watchdog Timer Enable bit (WDT disabled)

#pragma config PWRTE = OFF // Power-up Timer Enable bit (PWRT disabled)

#pragma config BOREN = ON // Brown-out Reset Enable bit (BOR enabled)

#pragma config LVP = ON // Low-Voltage (Single-Supply) In-Circuit Serial Programming Enable bit (RB3/PGM pin has PGM function; low-voltage programming enabled)

#pragma config CPD = OFF // Data EEPROM Memory Code Protection bit (Data EEPROM code protection off)

#pragma config WRT = OFF // Flash Program Memory Write Enable bits (Write protection off; all program memory may be written to by EECON control)

#pragma config CP = OFF // Flash Program Memory Code Protection bit (Code protection off)

#include <xc.h>

#define \_XTAL\_FREQ 4000000

#define TMR2PRESCALER 4

void duty\_gir(int deger1){

CCP1X = deger1 & 2; //duty nin dusuk degerlikli bitleri bu register lara yazilacak.

CCP1Y = deger1 & 1;

CCPR1L = deger1>>2; //duty nin yüksek degerlikli 8 biti bu register'a yazilacak.

}

PWM1\_TMR2\_giris()

{

CCP1M3 = 1; // capture pwm modunda. Mod secme bitleri

CCP1M2 = 1;

// Prescaler 4

T2CKPS0 = 1;

T2CKPS1 = 0;

TMR2ON = 1; // TM2 açma biti

TRISC2 = 0; // PWM çikisini kullanabilmek için CCP ucunu çikis yaptim.

}

void ADC\_Deger()

{

ADCON1=0b11000000; // AN ler analog, ADFM=1,ADcs2=1

ADCON0=0b00001001; //fosc/4, AN1 giris , GODONE=0, ADON=1

}

void main(void) {

unsigned int adc\_deger,deger=0;

double volt\_degeri;

PR2=249; // PWM periyodu= (PR2+1)\*4\*Tosc\*(TMR2Prescaler degeri)

duty\_gir(0);

PWM1\_TMR2\_giris();

TRISA = 0xFF; // PORTA input

ADC\_Deger();

while(1){

ADCON0bits.GO=1; // çevrime basla

while (ADCON0bits.GO); //bitene kadar bekle

adc\_deger=(ADRESH<<8)+ADRESL; //ADC de?eri de?i?kene at

volt\_degeri=adc\_deger\*0.004882; //Degeri ölçülen volta çevir. ADC den okunan deger\* adim degeri

deger=(volt\_degeri\*1000/5); // volt degeri duty e göndermek icin max. 1000 olan duty seviyesine gore hesaplamasi

// MAX duty= (Kaydediciler)\* Tosc \* (TMR2 Prescaler degeri)

if(volt\_degeri<0.4){

deger=0;

}

if(volt\_degeri>4.6){

deger=1000;

}

duty\_gir(deger);

}

return;

}

Sürekli bir şekilde okunan very dönüştürülerek kare dalga üretilmesi için PWM modülü çıkışına (CCP1) aktarılır.

volt\_degeri değişkenine atanan değer PWM üretmek için ilgili kaydedicilere yazdırılır. (CCP1X, CCP1Y, CCP1L)

Okunan değer voltaj birimi olarak dijitale dönüştürülür. (volt\_degeri=adc\_deger\*0.004882)

AN1 girişinden analog veri okunur **ve register lara kaydedilir.**