ÖZDiSAN ELEKTRONIK A.S.

ELAN Demo Board Özellikleri ve Yazılım Uygulamaları Rehberi

Doc.Version: 1.0

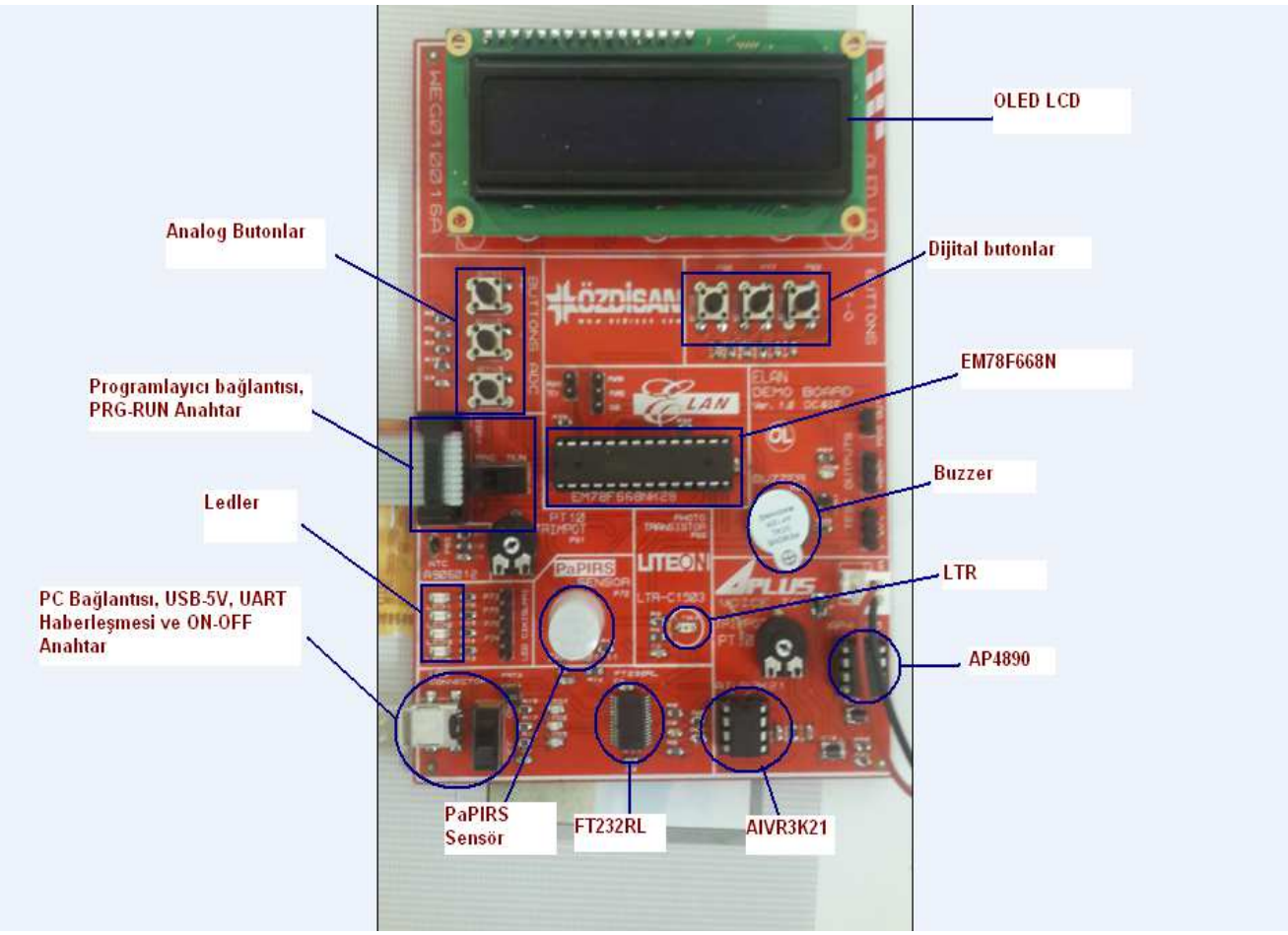
OLAB

Özdisan Elektronik Ar-Ge ve Teknik Destek Departmanı

Email: ts@ozdisan.com Tel: +90 2164201882

www.ozdisan.com

1.ELAN DEMO BOARD



Şekil 1.1 Elan Demo Board

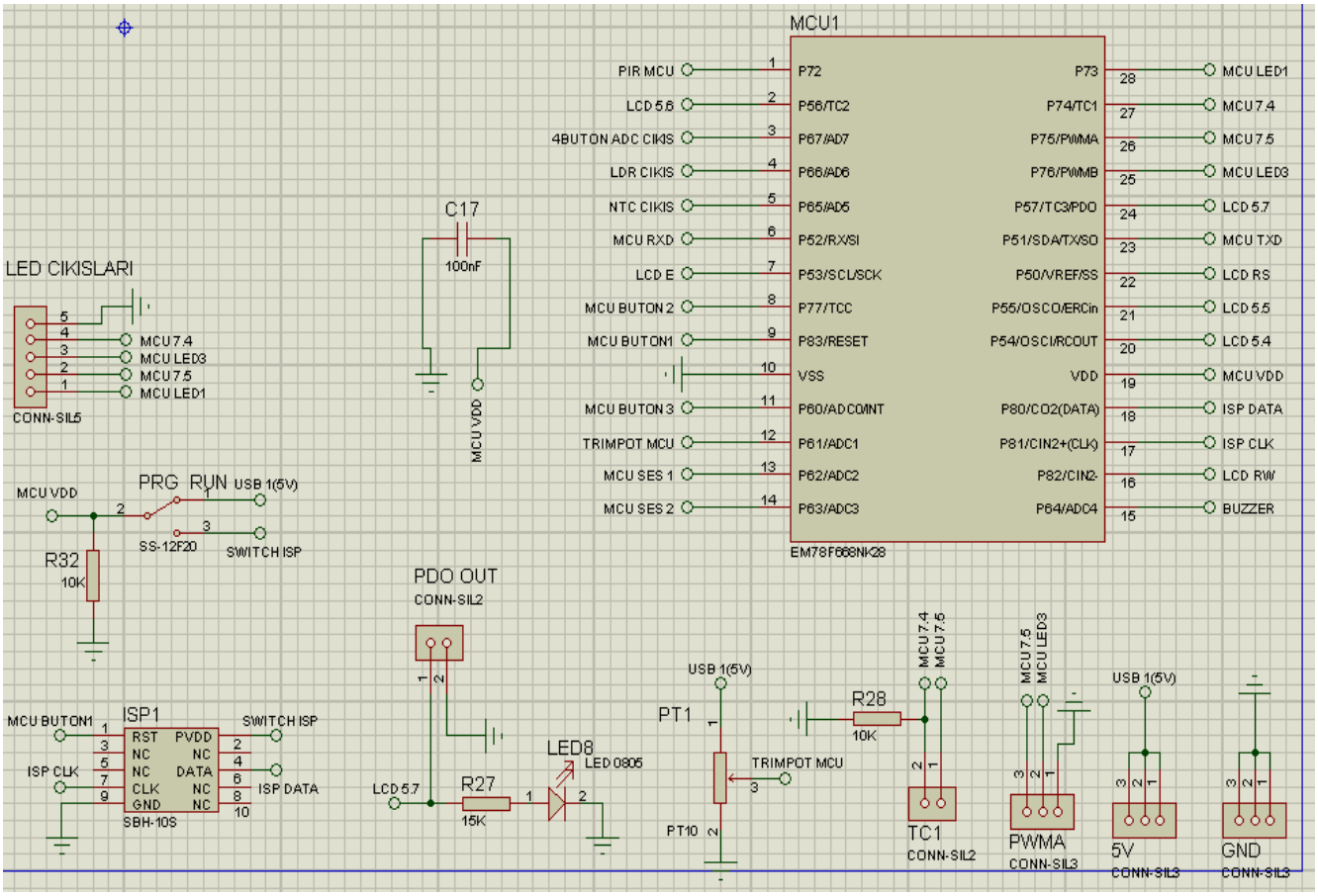
ELAN Demo Board, üzerinde mikrodenetleyici, buttonlar, LDR, ses entegresi, buzzer, hareket sensörü, ledler, OLED LCD ve FT232RL USB UART IC bulunan bir geliştirme bordu olarak tasarlanmıştır. Üzerinde bulunan malzemelerin kullanım amaçları şöyledir:

- Ledler ile dijital çıkışları kontrol etmek.
- Butonlar analog ve dijital girişleri kontrol etmektedir.
- Ses entegresi melodili ses çıkışı sağlamaktadır.
- Hareket sensörü, hareket algılayıp dijital giriş vermektedir.
- LDR, analog giriş verip ışık yoğunluğunu algılamaktadır.
- NTC, sıcaklıkla ilgili uygulama yapmamızı kolaylaştırmaktadır.
- Mikrodenetleyici, bütün bu fonksiyonları kontrol etmektedir.

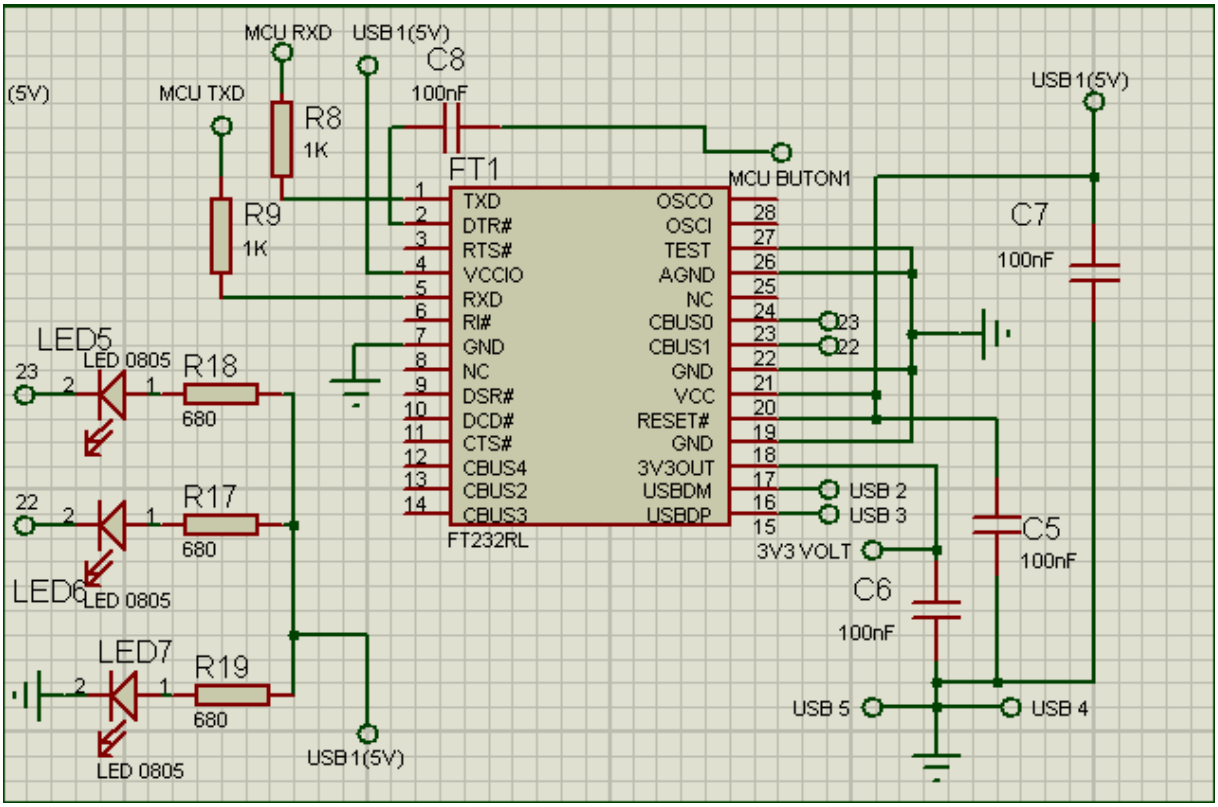
- FT232R demo board ile bilgisayarın seri haberleşmesini sağlamaktadır.

Ayrıca ELAN Demo Board da iki adet anahtar bulunmaktadır. Alt kısımdaki anahtar demo boardun enerjisi ve bilgisayarla bağlantıyı sağlarken, diğer anahtar programlama için kullanılmaktadır. Elan Demo Board'ı iki şekilde programlayabilirsiniz. Birincisi; mikrodenetleyiciyi boarddan çıkarıp programlayıcıya bağlayarak, diğeri ise; demo board ile programlayıcı arasında ISP kablosyuyla bağlantı yaparak programlayabilirsiniz. İkinci programlama şeklinde alttaki anahtarın kapalı ve diğer anahtarın prog konumunda olması gerekmektedir.

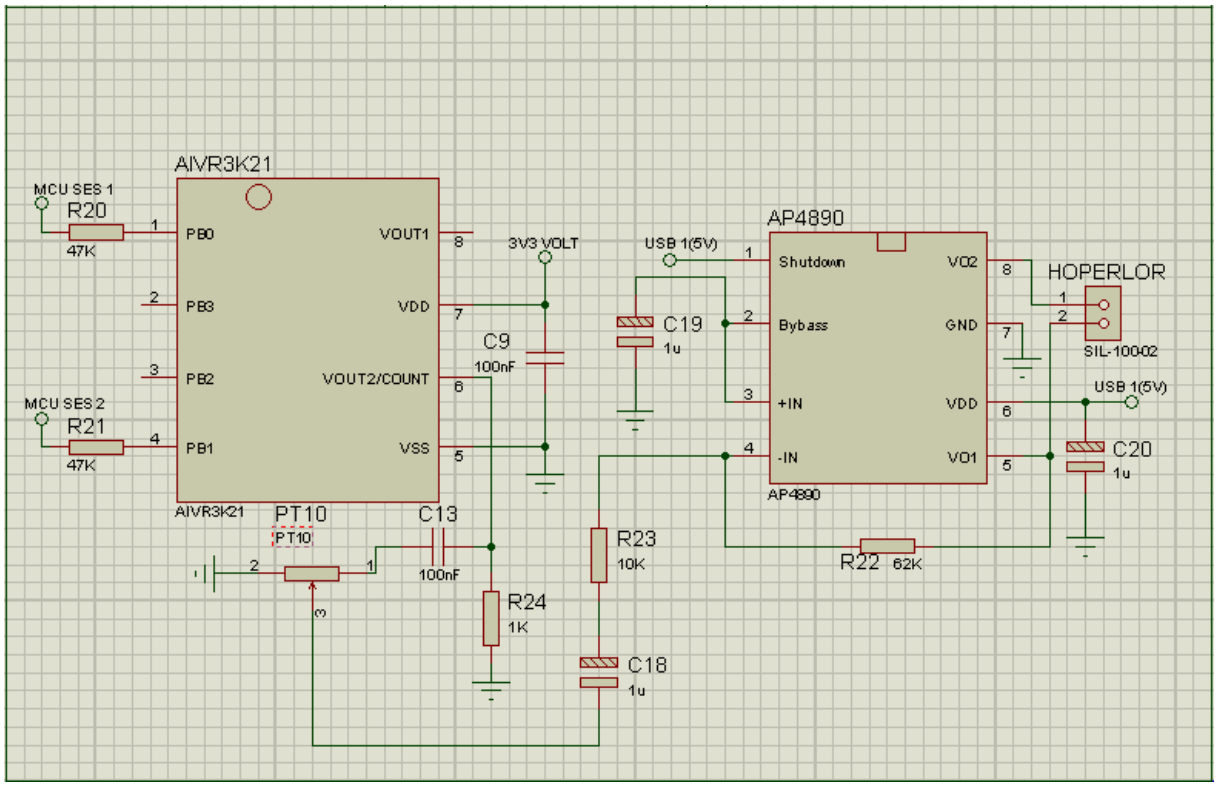
Elan Demo Board'ın bağlantı şeması aşağıdaki resimlerde gösterilmiştir.



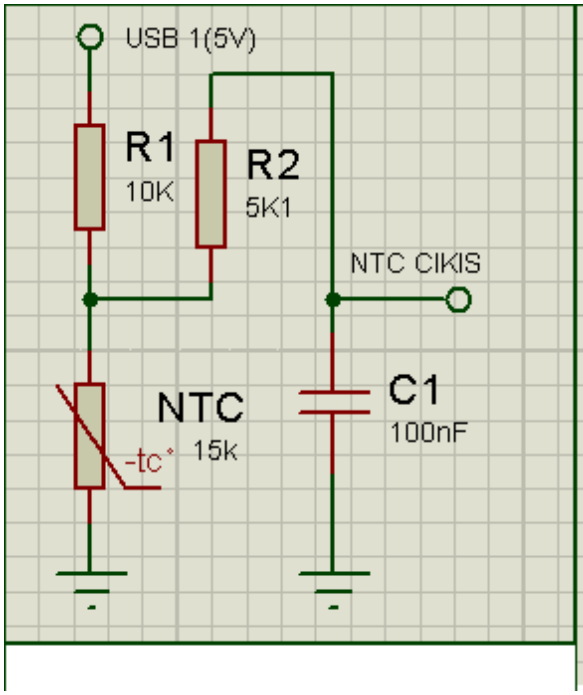
Şekil 1.2 ELAN Mikrodenetleyici Bağlantısı ve Diğer Bağlantılar

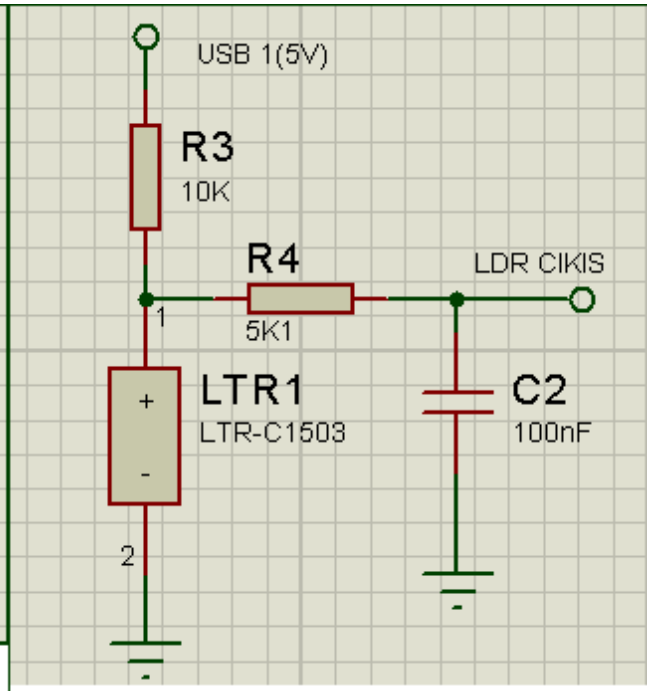


Şekil 1.3 FT232RL Bağlantısı



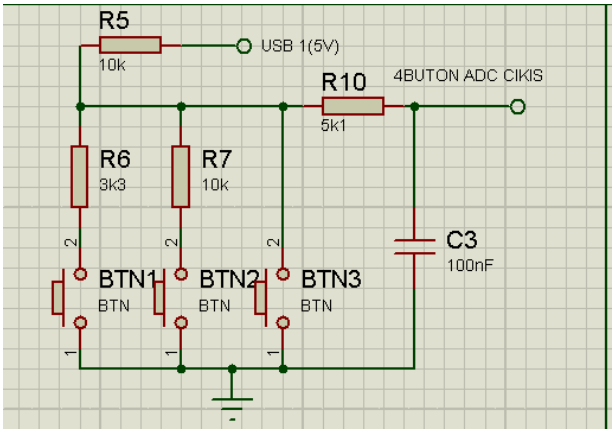
Şekil 1.3 APLUS Voice IC Bağlantısı

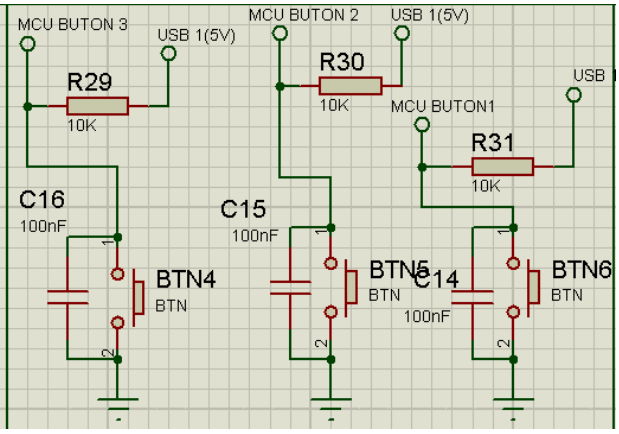




Şekil 1.4 NTC Bağlantısı

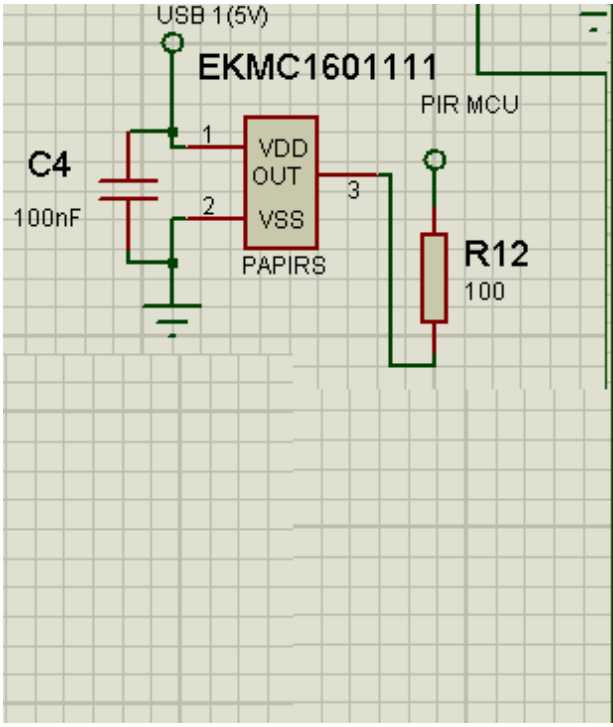
Şekil 1.5 LTR Bağlantısı

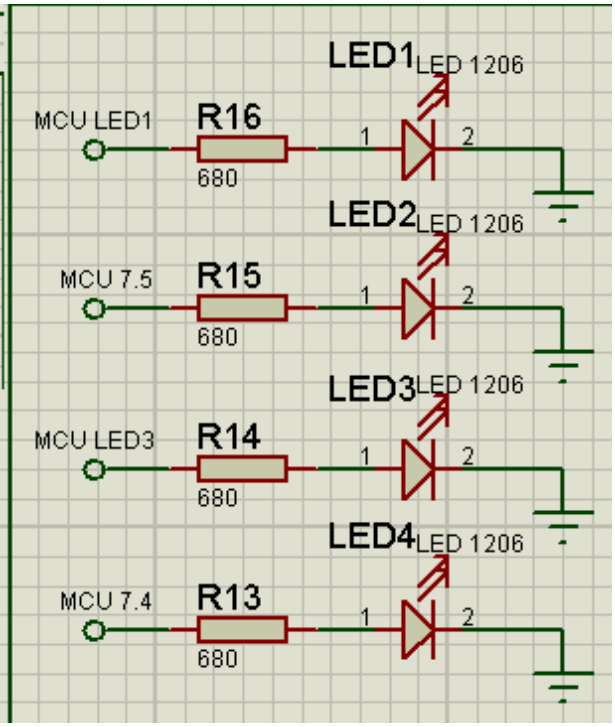




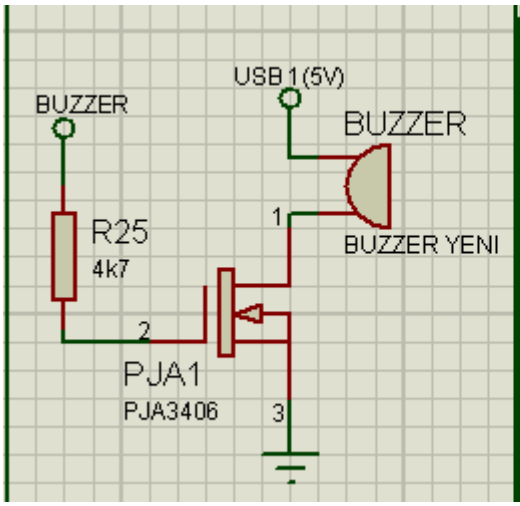
Şekil 1.6 Analog Buton Bağlantısı

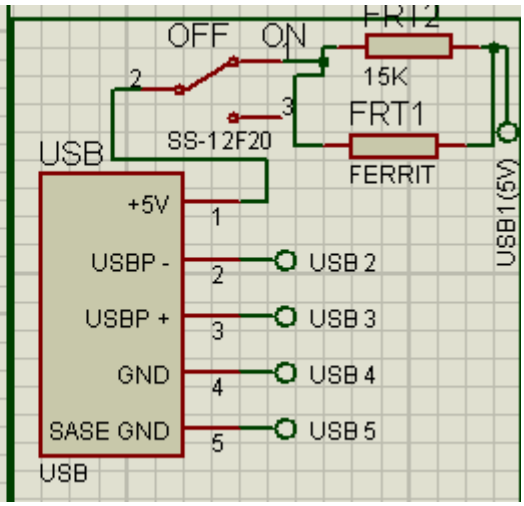
Şekil 1.7 Dijital Buton Bağlantısı



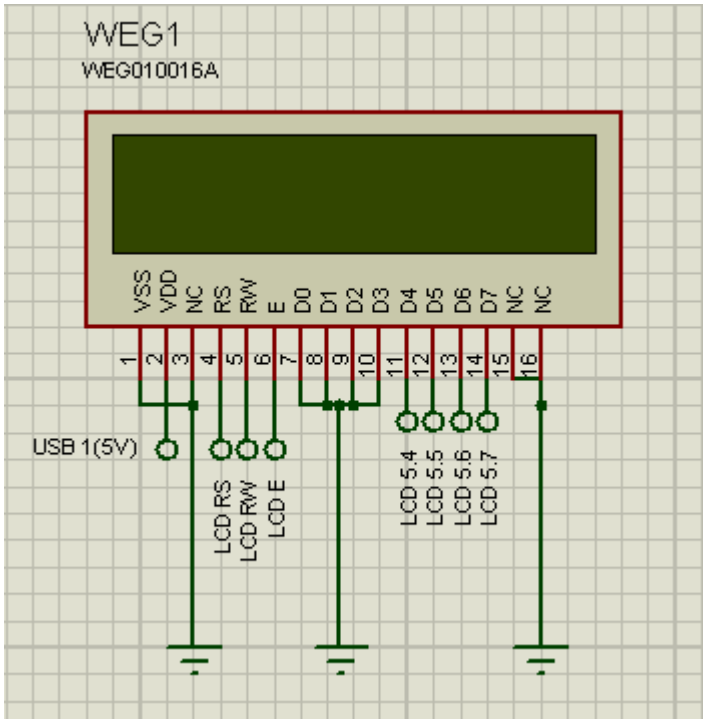


Şekil 1.8 PaPırs Sensör Bağlantısı **Şekil 1.9 LED Bağlantısı**





Şekil 1.10 Buzzer Bağlantısı **Şekil 1.11 USB Bağlantısı**



Şekil 1.12 OLED LCD Bağlantısı

1.1 ELAN 16F668N Mikrodenetleyici

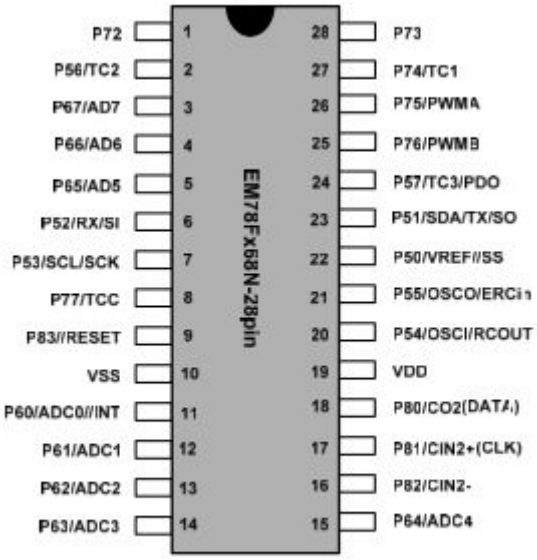


Figure 3-1 28-pin EM78Fx68N

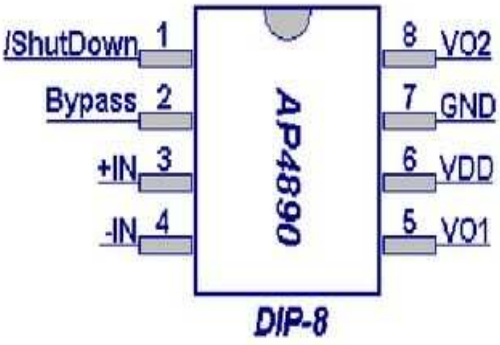
Genel Tanım

EM78F668N 8-bit bir mikrodenetleyicidir ve düşük güçlü, yüksek hızlı CMOS teknolojisiyle geliştirilmiştir. Bu denetleyici üzerinde 8Kx15 bit Elektrik Kontrollü Flash Bellek, 256x8 bit sistemde programlanabilir EEPROM, iki karşılaştırıcı, üç

tane 8-bit zamanlayıcı, bir adet 16-bit zamanlayıcı, iki adet 10-bit PWM, 12-bit çözünürlüklü 8 kanal AD, SPI, UART ve I2C bulundurur.

1.2 APLUS VOICE IC

aIVR3K21 Voice IC



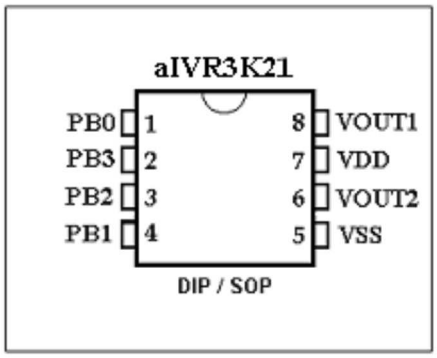
Audio Power Amplifier

1.2.1 AIVR3K31 Voice IC

Özellikleri:

- 4 giriş tetiklemeli anahtar tetikleme modu.
- 13 ses grubunu desteklemesi.
- Her bir kombinasyon için “Level/Edge,Hold/Un-Hold, Retrigger/Non-retrigger” ayarlanmasının yapılması.
- VOUT1 ve VOUT2 den PWM’ye bağlantı.
- VOUT2_COUT pininden DAC’a bağlantı.
- Hiçbir çıkış sinyalini desteklememe.
- 8-bit PCM dekteklemesi, 5-bit uLaw ve 4-bit ADPCM data sıkıştırması.

1.2.2 Pin Bağlantıları



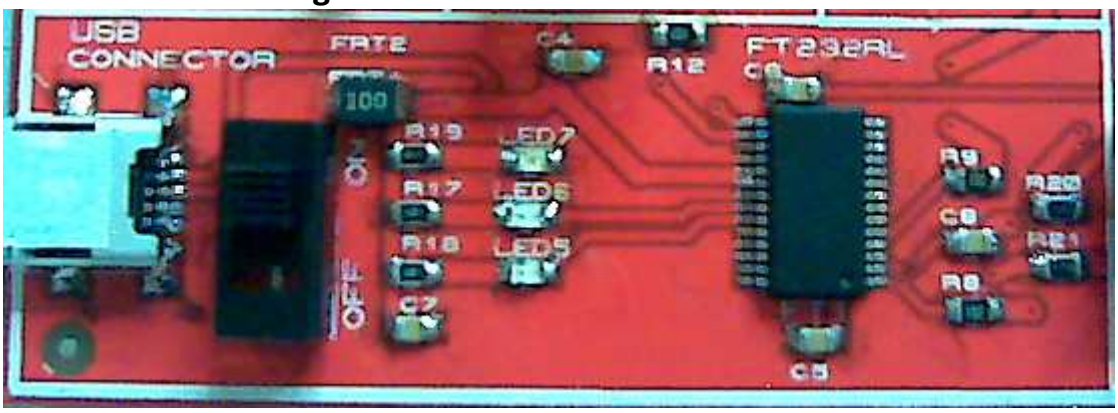
1.2.3 Pin Tanımlamaları

Pin Names	Description
VOUT1	PWM output to drive speaker directly
VOUT2_COUT	PWM output or COUT DAC output select by programmable option
VSS	Power Ground
VDD	Positive Power Supply
PBn	Input trigger pins with 1M Ohm internal pull-down

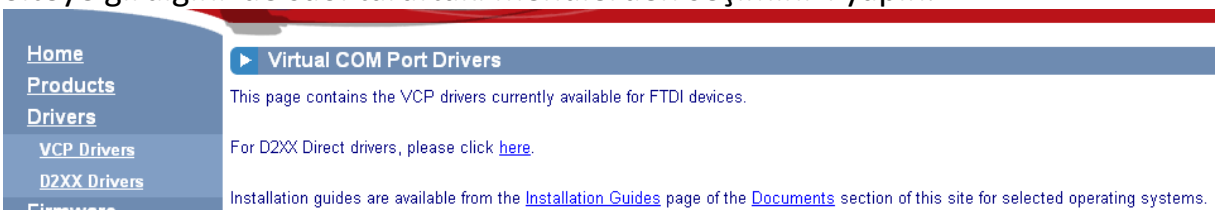
1.3 PaPIRS SENSOR

Series	Current Usage	Operating Voltage (VDC)	Circuit Stability Time	Range	Lens Color	Output	Detection Angle	Profile (mm)*	Lens Diameter (mm)
EKMB 	1μA, 2μA, 6μA	2.3-4.0	1μA/2μA - 25s (typ.)	5m, 12m	White, Black, Pearl White	Digital, Analog	5m: 94° (H), 82° (V)	5m: 14.4	5m: 9.5
			6μA - 30s (max.)				12m: 102° (H), 92° (V)	12m: 20.2	12m:20.7

1.4 FT232RL Bağlantısı ve Kurulumu



USB bağlantısı yapıldıktan sonra, FT232RL'nin bilgisayara tanımlanması gerekmektedir. Bunu yapmak için http://www.ftdichip.com/ linkine tıklayın. Siteye girdiğinizde saol taraftaki menülerden seçiminizi yapın.



Burada Drivers sekmesinden VCP Drivers sekmesini seçin. Daha sonra gelen sayfanın alt kısmında bilgisayarınıza kurmak için uygun olan programı seçin ce bilgisayarınıza indirin.

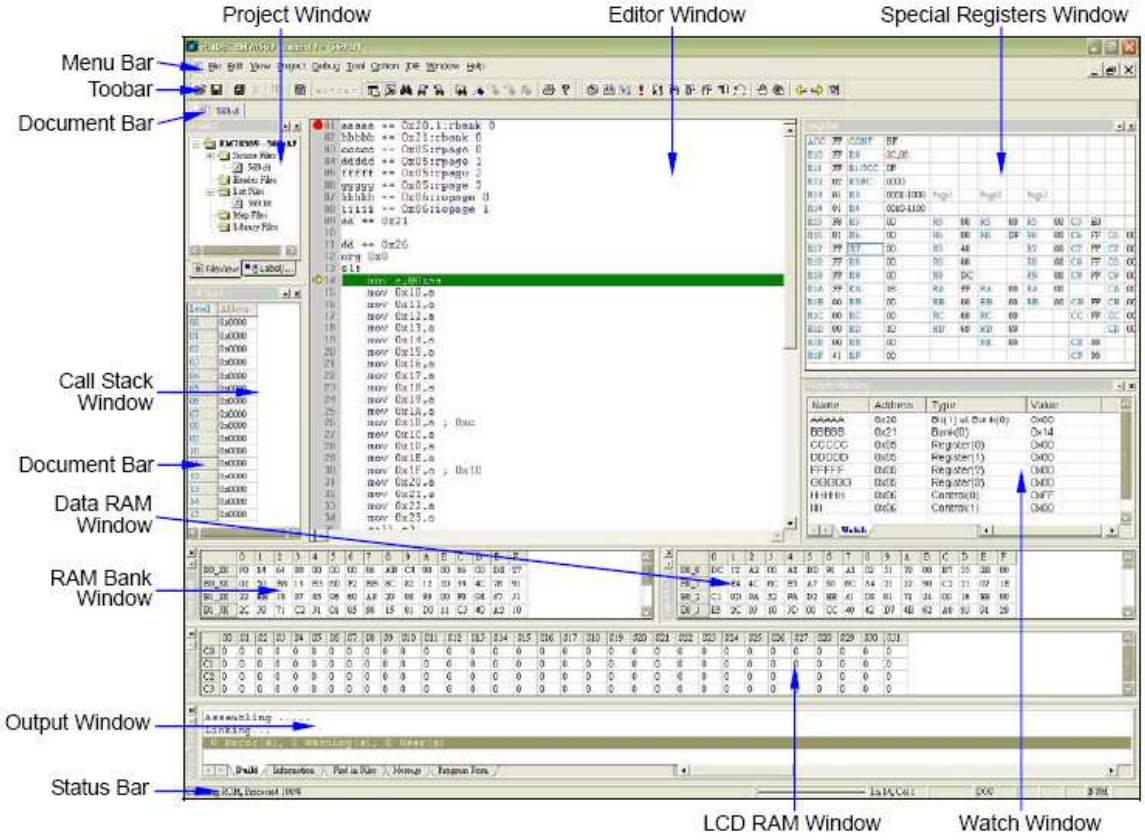
Currently Supported VCP Drivers:

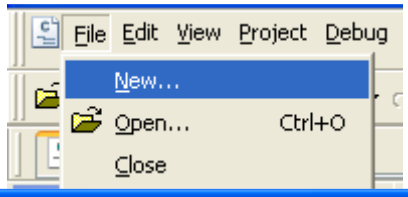
		Processor Architecture							
Operating System	Release Date	x86 (32-bit)	x64 (64-bit)	PPC	ARM	MIPSII	MIPSIV	SH4	Comments
Windows*	2014-09-29	Available as setup executable Contact support1@ftdichip.com if looking to create cusomised drivers		-	-	-	-	-	2.12.00 WHQL Certified Available as setup executable Release Notes
Linux	2009-05-14	1.5.0	1.5.0	-	-	-	-	-	All FTDI devices now supported in Ubuntu 11.10, kernel 3.0.0-19 Refer to TN-101 if you need a custom VCP VID/PID in Linux
Mac OS X	2012-08-10	2.2.18	2.2.18	2.2.18	-	-	-	-	Refer to TN-105 if you need a custom VCP VID/PID in MAC OS
Windows CE 4.2-5.2**	2012-01-06	1.1.0.20	-	-	1.1.0.20	1.1.0.10	1.1.0.10	1.1.0.10	
Windows CE 6.0/7.0	2012-01-06	1.1.0.20 CE 6.0 CAT CE 7.0 CAT	-	-	1.1.0.20 CE 6.0 CAT CE 7.0 CAT	1.1.0.10	1.1.0.10	1.1.0.10	For use of the CAT files supplied for ARM and x86 builds refer to AN_319

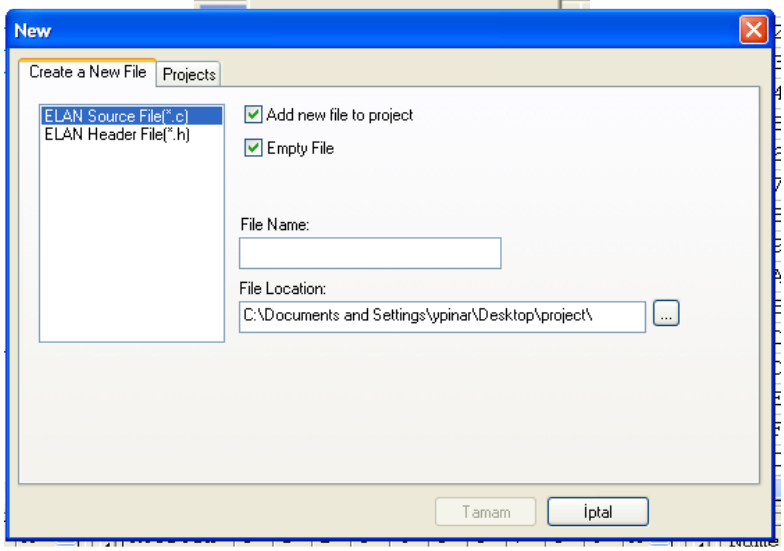
Yazılımı bilgisayarınıza indirdikten sonra kurulumunu yapın.

2.YAZILIM KLAVUZU

2.1 ELAN eUIDE

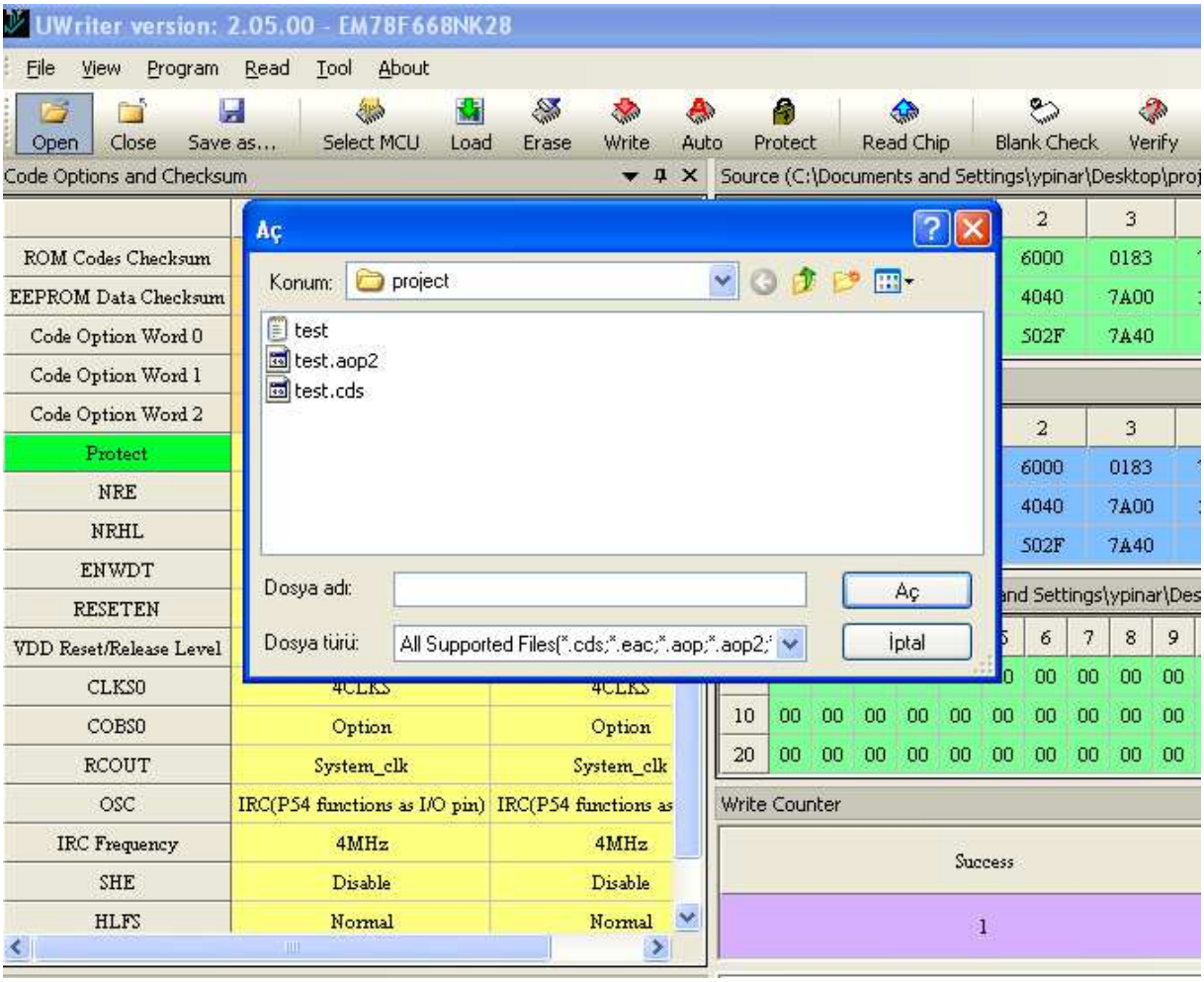




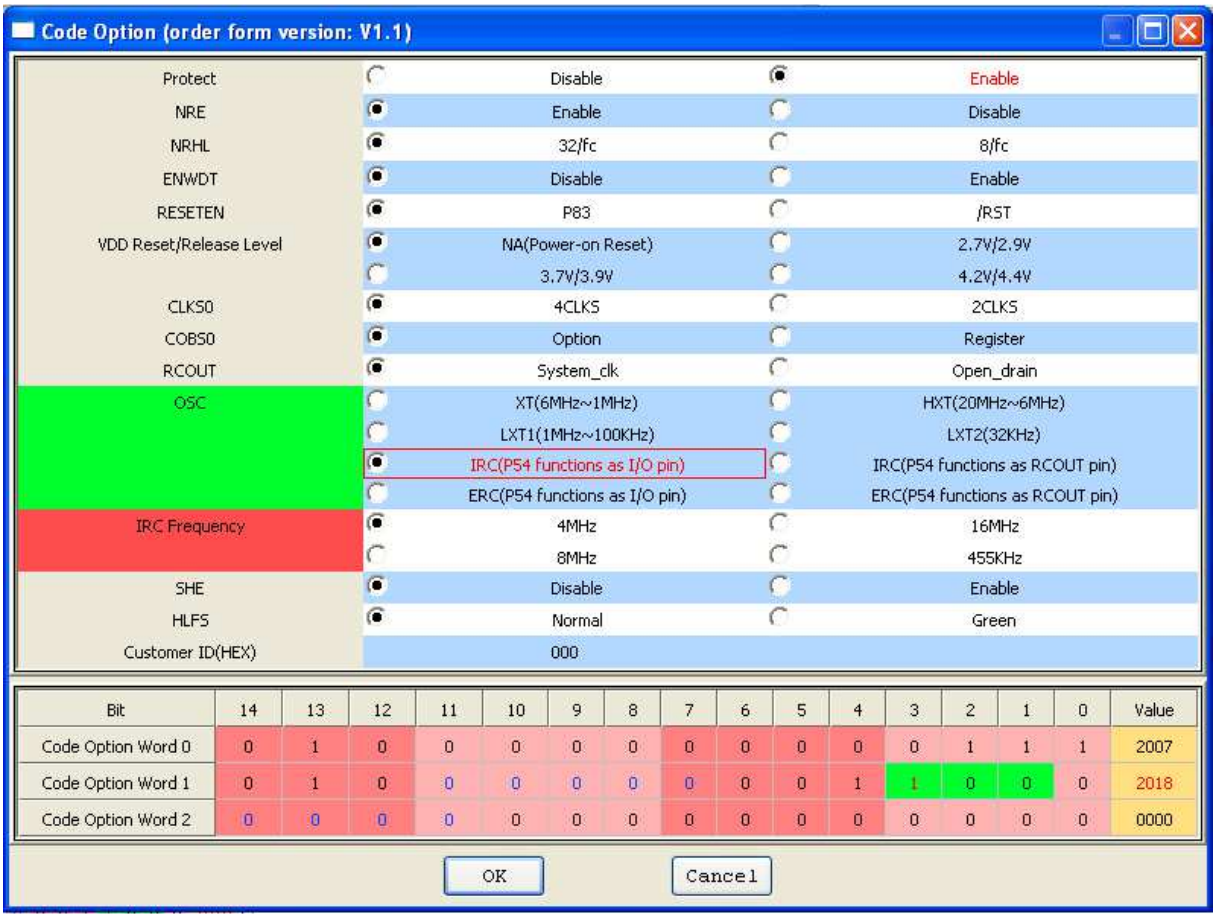


File->New sekmesini seçtikten sonra proje penceresi açılacaktır. İlk önce “Projects” sekmesini seçiyoruz. Projemizin adını yazıp kayıt yerini seçtikten sonra “Tamam” sekmesine basıyoruz. Bu şekilde proje dosyamızı oluşturmuş oluyoruz. Bu işlemden sonra, tekrardan File->New yaparak yeni bir dosya oluşturuyoruz. Dosya uzantısı olarak “.c” seçersek ana programı oluşturmuş oluyoruz. Projemizin ana programına header eklemek istediğimiz zaman aynı işlemleri yapıp “.h” uzantılı dosya oluşturuyoruz. Dosya oluşturma işlemini tamamladıktan sonra “Editor Window” kısmına programı yazıyoruz. Programı test etmek istediğimiz zaman “Build” sekmesine basıp programı koşuyoruz. Programda değişiklik yapıp tekrar koşmak istesek “Rebuild” sekmesine basıyoruz.

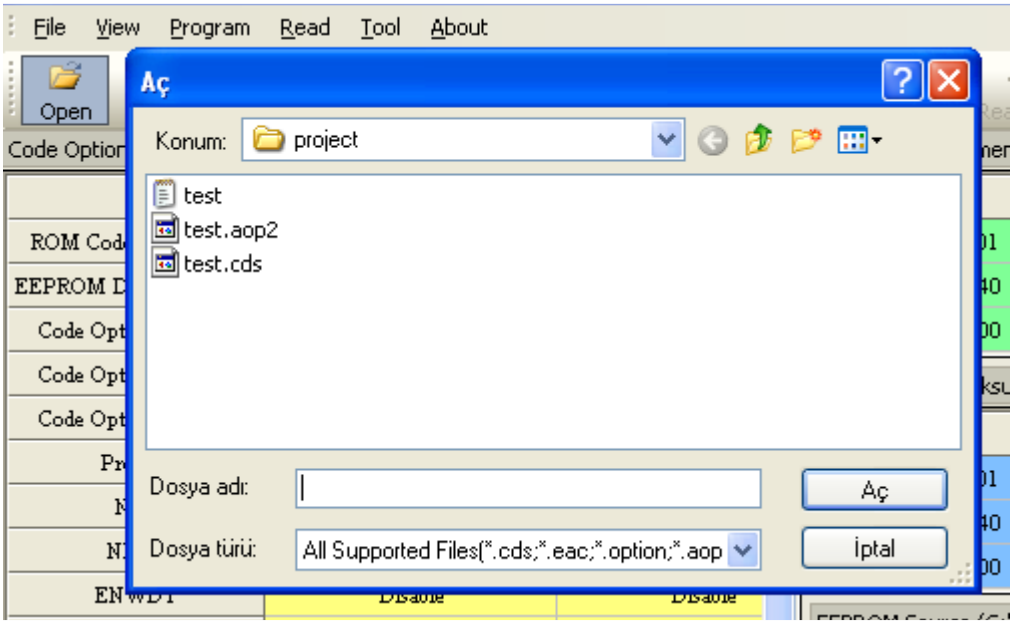
2.2 ELAN UWRİTTER



UWRİTTER programını çalıştırıyoruz. “Open” sekmesine tıkayıp resimdeki pencereyi açıyoruz. Bu pencerede proje dosyamızı seçiyoruz. Proje dosyamızda kayıtlı olan “.cds” uzantılı dosyayı seçip “Aç” sekmesine basıyoruz.

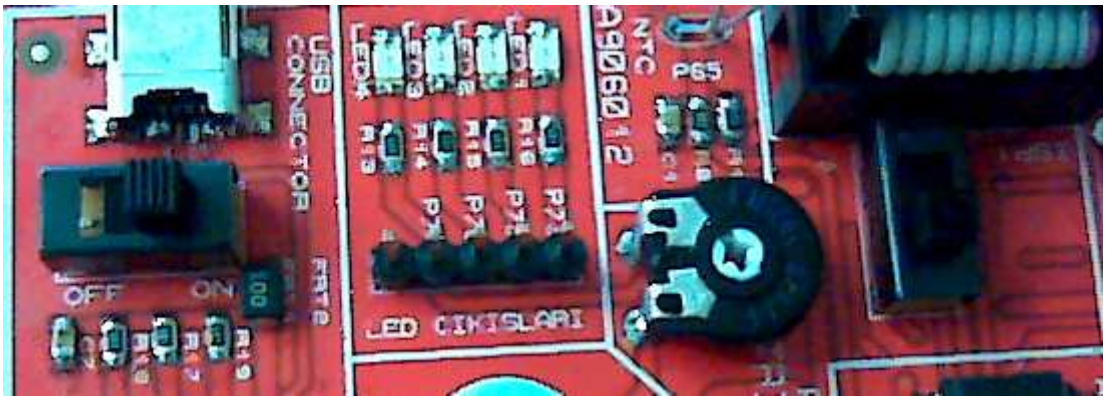


“Aç” sekmesini tıkladığımızda resimdeki pencere açılacaktır. Bu pencerede ayarlamaları yaparken sadece kırmızı yazılı olanları seçiyoruz ve “OK” sekmesine basıyoruz. Bu şekilde “.cds” uzantılı dosyamız “.aop” uzantılı dosyaya çevirilip kaydedilmiş oldu.



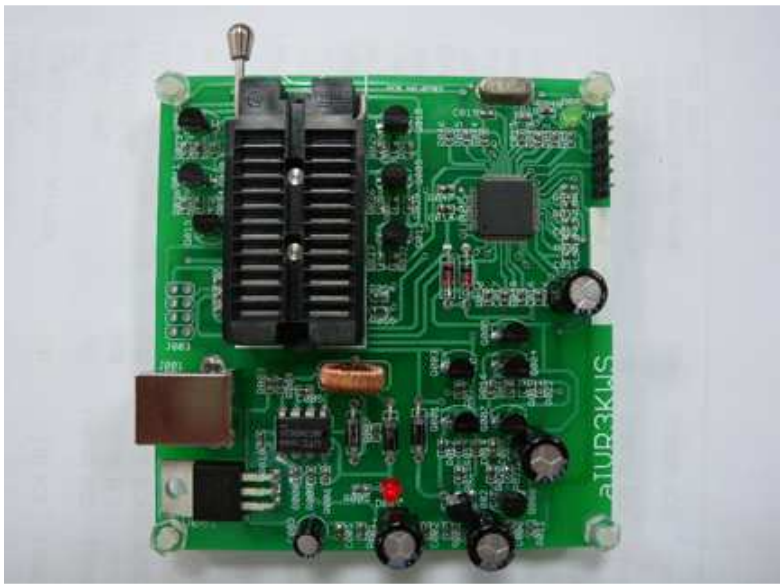
"Open" sekmesine basıyoruz. Açılan pencerede ".aop" uzantılı dosyayı seçip "Aç" sekmesine basıp dosyamızı seçiyoruz.

İşlemleri yaptıktan sonra "Load" sekmesine basıp programlayıcıya yüklenmesini bekliyoruz. Daha sonra "Auto" sekmesine basıp yazdığımız programı mikrodenetleyiciye yüklüyoruz. Programlama yaparken, anahtarların konumlarının doğru olması gerekmektedir. Aksi takdirde UWritter programı hata vermektedir.



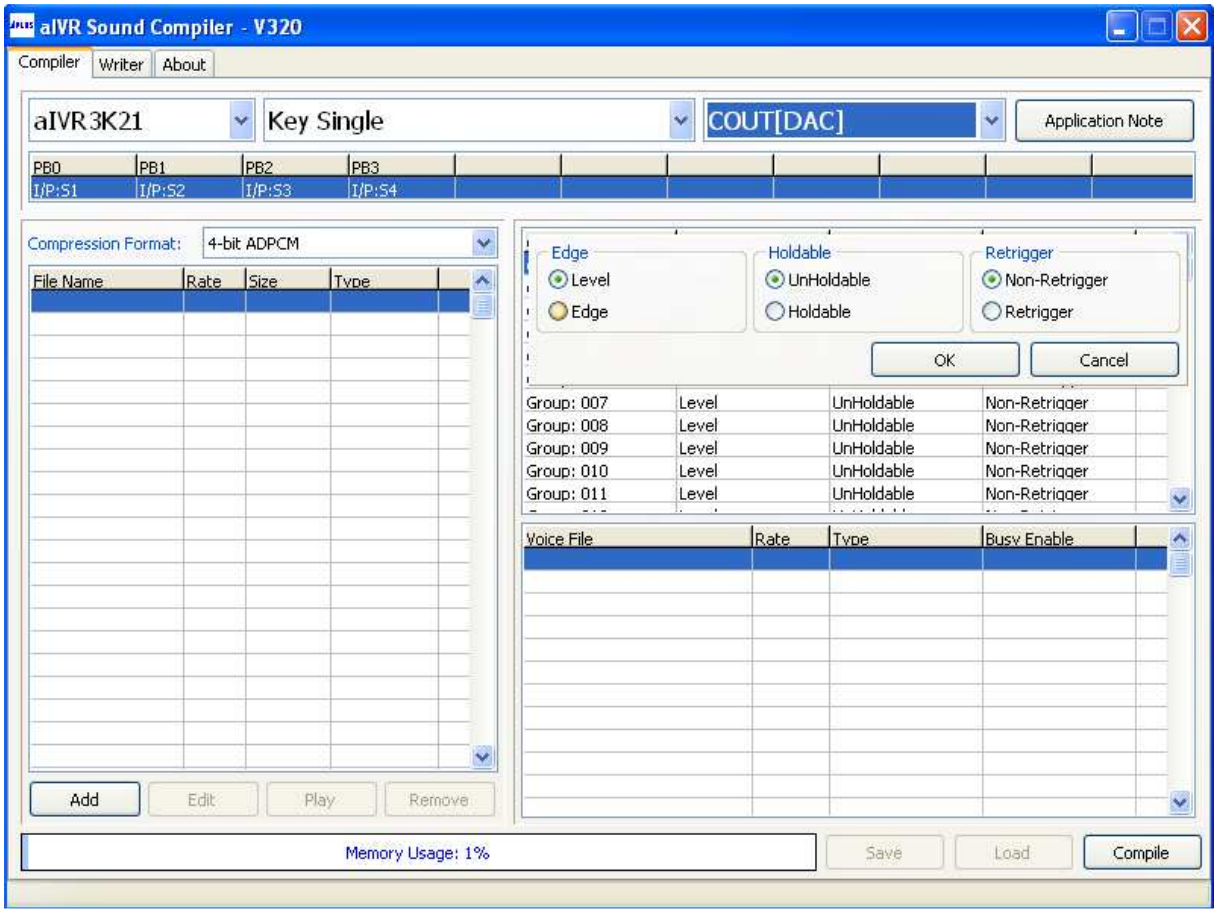
Şekil 2.1 ELAN Demo Board Anahtarlama

2.3 APLUS aIVR Sound Compiler



Şekil 2.3.1 aIVR3KWS Programlayıcı

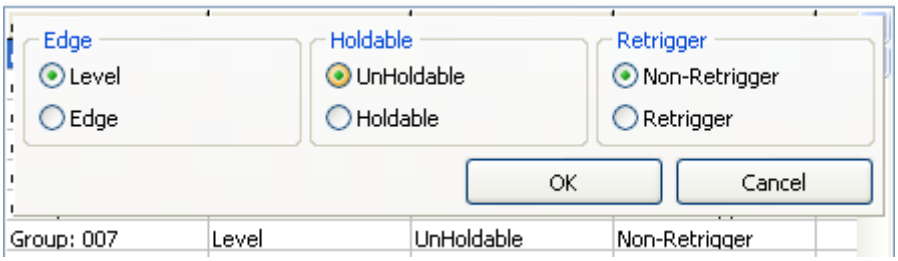
aIVR3KWS Programlayıcısını bilgisayara bağlıyoruz. Bağlantıyı yaptıktan sonra aIVR Sound Compiler programını çalıştırıyoruz. Programlayıcıyı bilgisayara bağlamadığınız zaman program çalışmayacaktır. Program çalışınca ekrana bir arayüz gelecektir ve arayüzde programlama ayarları yapılacaktır.



Şekil 2.3.2 aIVR Sound Compiler Arayüz

Kullanmak istediğimiz ürünün seçimini yapıyoruz. Üst sekmelerden kullandığımız ürüne göre diğer ayarlamaları yapıyoruz. Ses entegremizi DAC çıkışına bağlayacaksak “COUT” u, PWM ye bağlayacaksak “PWM” seçilmelidir. Biz AIVR3K21 kullandığımızdan ayarlarımız ekrandaki gibi olacaktır. Bu işlemleri yaptıktan sonra alt kısımda “Add” komutuna bastığımızda “mute” ve “wave” eklentileri gelmektedir. “Wave” ses dosyasını eklememizi, “mute” bekleme süresi eklememizi sağlar. Bu eklediğimiz ses dosyalarımız 8-bitlik formatta olmaktadır. “Compression Format”tan bu ses dosyalarını 5-bitlik yada 4-bitlik olarak sıkıştırabiliyoruz. Eğer eklediğimiz ses dosyasını kaldırmak istiyorsak, ses dosyasına bir kere tıklayıp “remove” sekmesine basıyoruz. Bu işlemlerden sonra gruplarımızı belirliyoruz. Gruplara ses dosyalarımızı ekliyoruz. Bunun için eklediğimiz ses dosyasının üzerine çift tıklamamız yeterlidir. Entegremiz dijital çalışmaktadır. Bu şekilde grup tablolarını inceleyip ayarlamanızı yapabilirsiniz.

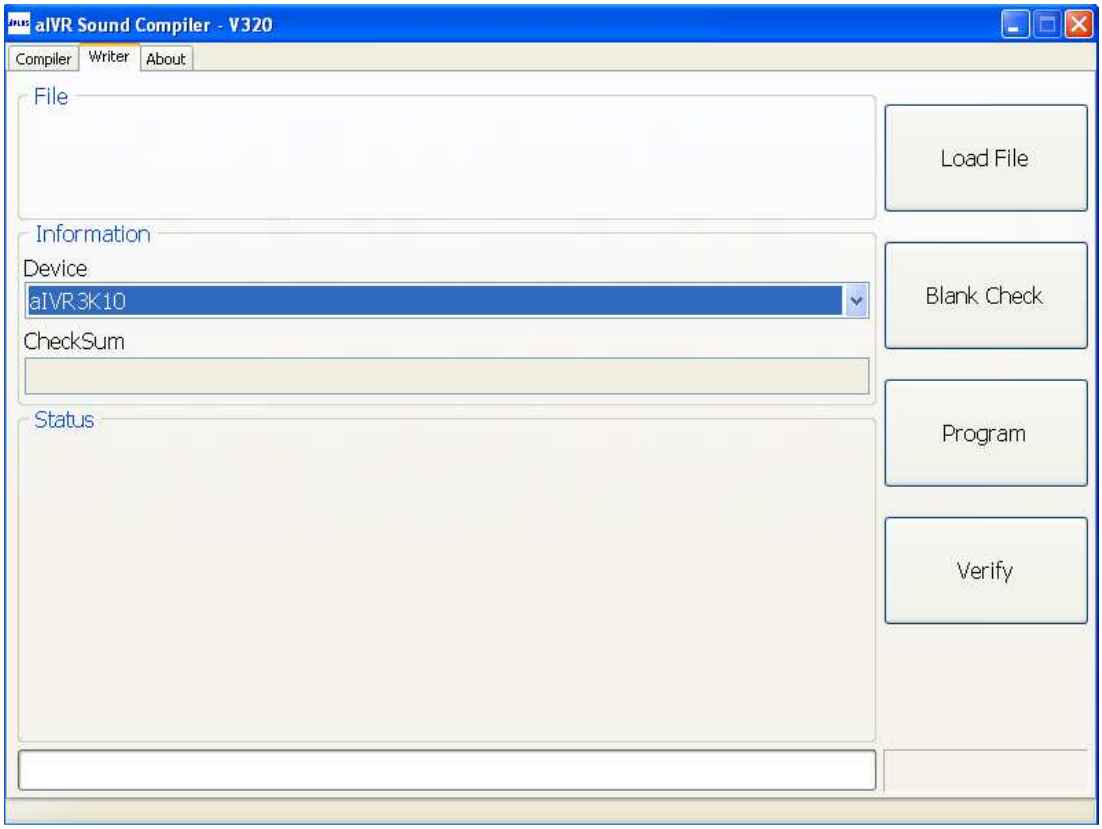
Ayrıca grupların üzerine sağ tıklayıp “edit” e bastığımızda ayarlama kısmı gelmektedir. Bu ayarlama sekmesinin ayarlamaları şu şekildedir.



Şekil 2.3.3 Grup Düzenlemesi

Level: Kanal aktif olduğu sürece ses tekrar eder.
Edge: Kanal aktif olduğu sürece sadece bir defa çalar. Bitirdikten sonra yeniden giriş verilmesini bekler.
UnHoldable: Girişine kısa bir pulse verildiğinde ses bitene kadar çalar.
Holdable: Girişe pulse verildikçe sesi çalar. Pulse kesildiğinde sesi keser.
Non-Retrigger: Girişten gelen ardışık pulseleri dikkate almaz.
Retrigger: Girişe gelen her pulsede yeniden başlatır.

Bu işlelmeri yaptıktan sonra “compile” sekmesine basarak compile işlemini yapıyoruz ve “writter” kısmına geçiyoruz.



Şekil 2.3.4 Writter

"Device" kısmından bağladığımız entegreyi seçiyoruz. "Load" sekmesine basarak compile ettiğimiz dosyayı yüklüyoruz. Programı yüklemeden önce "Blank Check" butonuna basmakta fayda var. Bu buton programlayıcıdaki entegrenin doğru bağlanıp bağlanmadığını test eder. Bu işlem başarıyla gerçekleştikten sonra "Program" sekmesine basıp programımızı yüklüyoruz.

3. UART Haberleşmesi

3.1 Uart Haberleşme Registerları

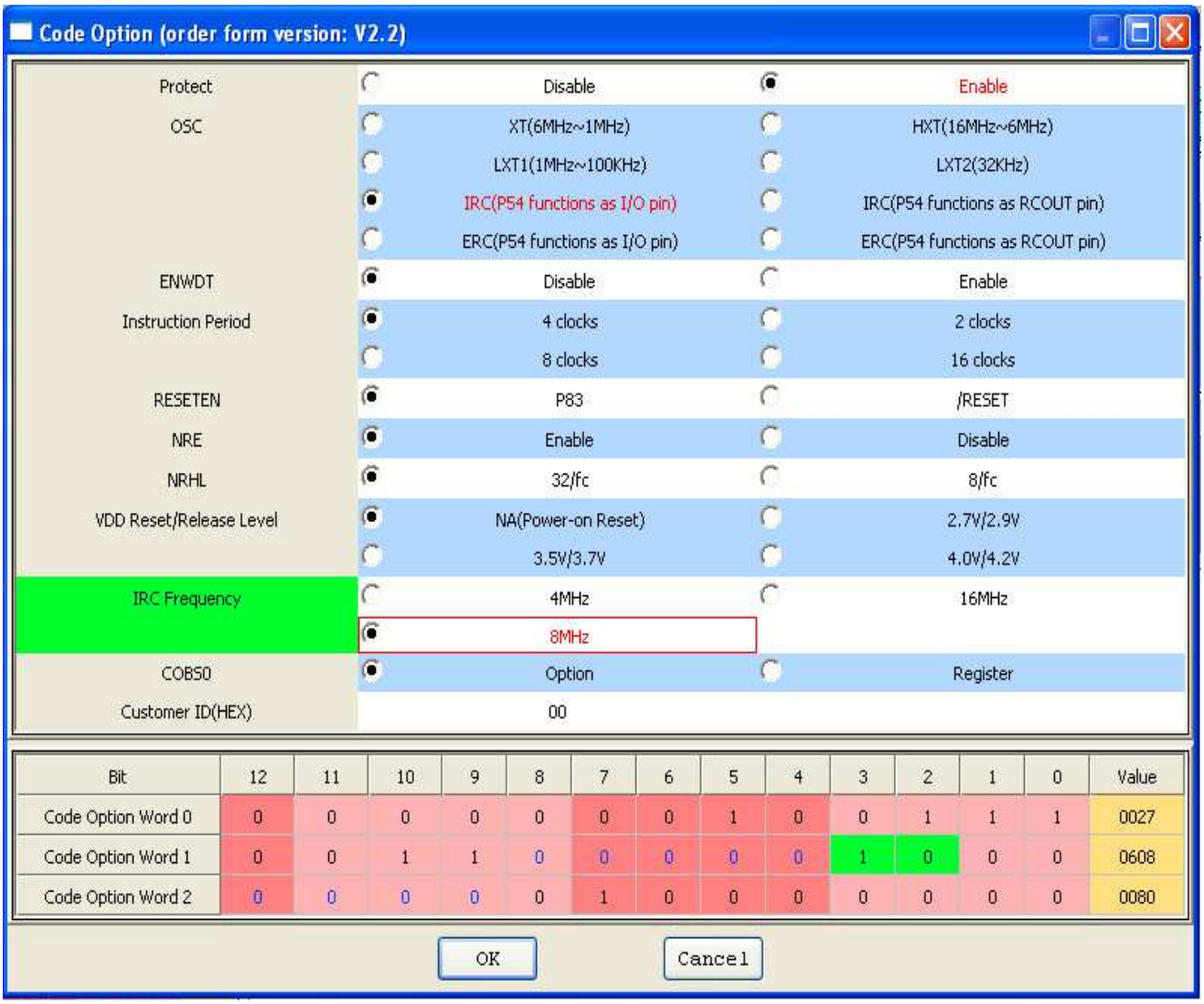
Registers for UART Circuit

R_Bank	Addr	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Bank 0	0×32	URCR1	URTD8	UMODE1	UMODE0	BRATE2	BRATE1	BRATE0	UTBE	TXE
			R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
Bank 0	0×33	URCR2	0	0	SBIM1	SBIM0	UINVEN	0	0	0
					R/W	R/W	R/W			
Bank 0	0×34	URS	URRD8	EVEN	PRE	PRERR	OVERR	FMERR	URBF	RXE
			R	R/W	R/W	R	R	R	R	R/W
Bank 0	0×35	URRD	URRD7	URRD6	URRD5	URRD4	URRD3	URRD2	URRD1	URRD0
			R	R	R	R	R	R	R	R
Bank 0	0×36	URTD	URTD7	URTD6	URTD5	URTD4	URTD3	URTD2	URTD1	URTD0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bank 0	0×0D	ISR2	CMP2IF	CMP1IF	TC3IF	TC2IF	TC1IF	UERRIF	RBFF	TBEF
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bank 0	0×1D	IMR2	CMP2IE	CMP1IE	TC3IE	TC2IE	TC1IE	UERRIE	URIE	UTIE
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Şekil 3.1.1 UART Register Tablosu

URCR1 ve URCR2 haberleşme ayarlarının yapıldığı registerlerdir. URS registerından "Parity" ayarlaması yapılmaktadır. Uart haberleşmede; haberleşme başlamadan önce başlama biti "0" gönderilir. Haberleşme yapıldıktan sonra da bitiş biti "1" gönderilir. Parity bit ise bitiş bitinden önce gönderilmektedir. Eğer Parity biti göndermek istemiyorsak, bunu kontrol eden registerdeki kısmı "0" yaparak etkisizleştirebiliriz. TXE vericiyi aktif hale getirirken, RXE de alıcı kısmı aktif hale getirmektedir. Göndereceğimiz datayı URTD registerına yazıyoruz. Aldığımız data ise otomatik olarak URRD registırına kaydedilmektedir. Uart ile ilgili ayrıntılı bilgi için mikrodenetleyicinin datasheetini ve örnek programramı inceleyebilirsiniz.

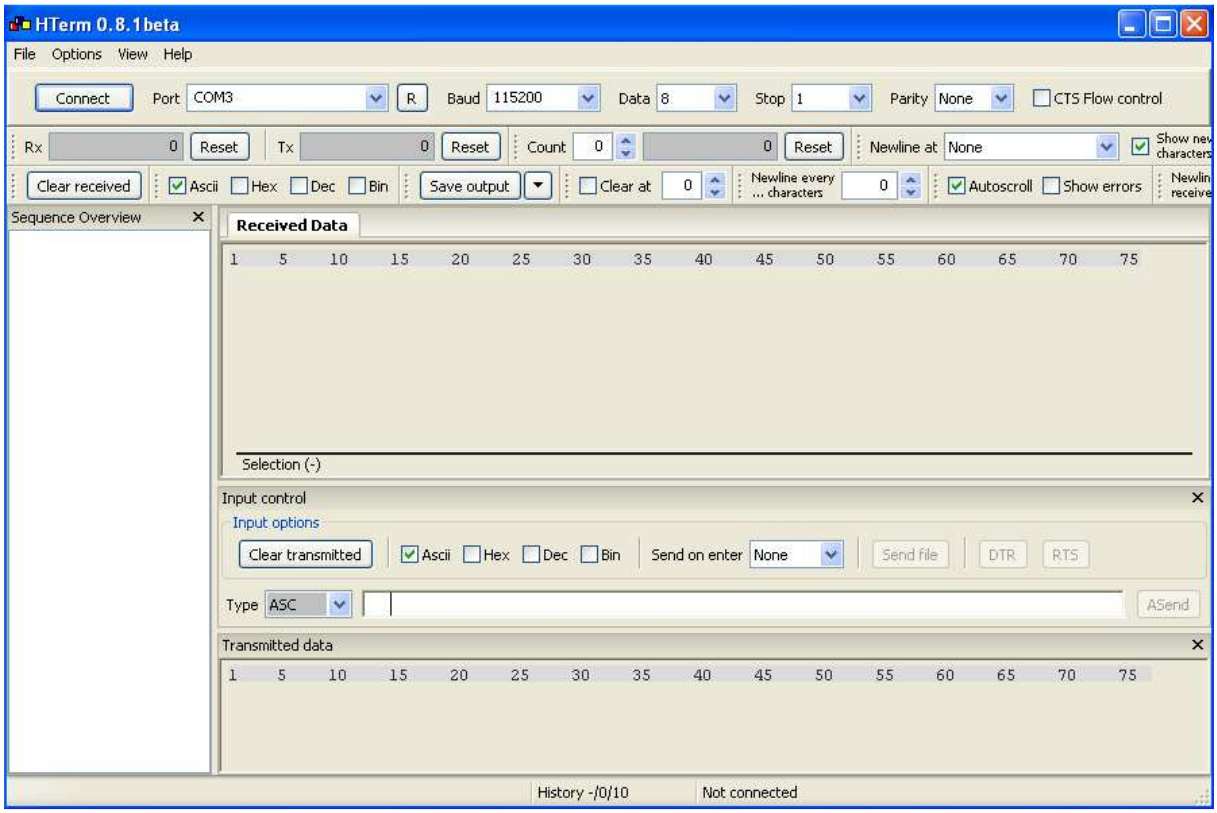
UART haberleşmesinde programı yazdıktan sonra uWRİTTER programından mikrodenetleyiciye yükleme yapacağınız zaman yaptığımız normal ayarlamaya ek olarak “IRC Frequency”i 8MHz olarak seçmelisiniz.



Şekil 3.1.2 UART Haberleşmesi uWRİTTER Ayarları

Bu ayarlamayı yaptıktan sonra “OK” tuşuna basıp programınızın “aop” uzantılı dosyasını oluşturmuş olursunuz. Daha sonra tekrar “open” sekmesinden “aop” uzantılı dosyayı seçip “load” ve daha sonra “aotu” sekmelerinden programı mikrodenetleyiciye yükleyebilirsiniz.

3.2 HTerm



HTerm, mikrodenetleyici ile bilgisayar arasında haberleşmeyi sağlayan bir haberleşme arayüzüdür. Programı açtığınızda yukarıdaki gibi bir ekran gelmektedir. Buradan boardın bağlı olduğu portu seçip ayarları yaptıktan sonra "connect" tuşuna bastığınızda bağlantı gerçekleşecektir. Ayarlamaları şu şekilde yapmanız gerekmektedir.

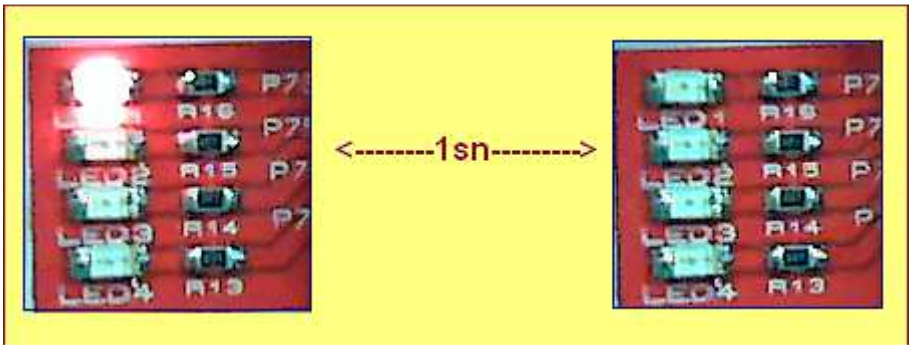
- Uart registerınızda ayarladığınız Baud Rate seçiniz.
- Göndereceğiniz yada alacağınız data boyutunu seçiniz. Örneğin; 7-bit, 8-bit
- Stop bitini seçiniz ve eğer ayarladıysanız Parity bitini seçiniz ayarlı değilse "none" de bırakınız.

Bu ayarlamalar yapıldıktan sonra, aldığınız dataları hangi formatta görmek istiyorsanız "clear received" butonunun yanındaki kutucuklardan bu seçimiyapabilirsiniz. Alınan datalar "received data" penceresine yazılmaktadır. Bunları silmek için "clear received" butonuna basmalısınız. Aynı ayarlamaları data göndermek için de yapabilirsiniz. Göndermek için seçtiğiniz datanın tipi belirledikten sonra "type" nin yanındaki boşluğa datayı yazıp "ASend" butonuna basarak datayı gönderebilirsiniz. Gönderilen datalar "Transmitted Data" kısmına yazılmaktadır.

4. Örnek Yazılımlar

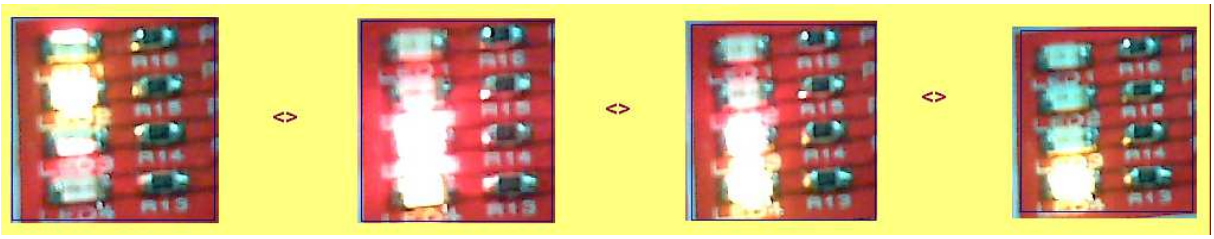
Yazdığımız örnek programlarda “EM78F668N.h” headerı bulunmaktadır. Bu program, ek olarak dosyalarda bulunmaktadır. Projeleri oluşturduğunuz klasörün içine bu header dosyasını kopyalayıp projenize header olarak ekleyebilirsiniz. Ayrıca bu isimle bir header dosyası oluşturarak bu dosyanın içindekileri oraya kopyalayabilirsiniz.

4.1 Led Yak-Söndür




```
//led yakıp söndürme programı
#include "EM78F668N.h"
//mikroişlemci ayarları
#define DISI()  _asm{disi}
#define ENI()   _asm{eni}
#define WDTC()  _asm{wdtc}
#define NOP()   _asm{nop}
#define SLEP()  _asm{slep}
//P73 pininin tanımlanması
#define LED   P73
//delay'ın alt fonksiyonu (bu fonksiyonu header olarak da yazabilirsiniz.)
void DelayMs(unsigned char x) {
        unsigned char say,i;
        for(say=0; say<x; say++) {
                for(i=0; i<100; i++) {
                        NOP();
                }
        }
}
void main(void)
{
        P7CR=0b10000100; // port  control register  (1:INPUT, 0:OUTPUT)
        while(1)  {  //while(1) programı sonsuz döngüye sokar.
                LED=1;  //led yak
                DelayMs(1000);  //bekle(100ms)
                LED=0; //led söndür
                DelayMs(1000);
        }
}
```

4.2 Kayan Led



```
//kayan led programı
#include "EM78F668N.h"
//mikroişlemci ayarları
#define DISI() _asm{disi}
#define ENI()  _asm{eni}
#define WDTC() _asm{wdtc}
#define NOP()  _asm{nop}
#define SLEP() _asm{slep}
//led pinlerinin tanımlanması
#define LED1   P73
#define LED2   P75
#define LED3   P76
#define LED4   P74
//delay'ın alt fonksiyonu (bu fonksiyonu header olarak da yazabilirsiniz.)
void DelayMs(unsigned char x) {
        unsigned char say,i;
        for(say=0; say<x; say++) {
                for(i=0; i<100; i++) {
NOP();   }   }  }
void main(void)
{
        P7CR=0b10000100;  // port  control register  (1:INPUT, 0:OUTPUT)
        while(1) {  //while(1) programı sonsuz döngüye sokar.
                LED4=0;  //led söndür
                LED1=1;  //led yak
                DelayMs(100);  //bekle(100ms)
                LED1=0;
                LED2=1;
                DelayMs(100);
                LED2=0;
                LED3=1;
                DelayMs(100);
                LED3=0;
                LED4=1;
                DelayMs(100);
        }
}
```

4.3 Buton ile Led Kontrolü

```
//button ile led kontrolü
#include "EM78F668N.h"
//mikroişlemci ayarları
#define DISI() _asm{disi}
#define ENI()  _asm{eni}
#define WDTC() _asm{wdtc}
#define NOP()  _asm{nop}
#define SLEP() _asm{slep}
//button ve led pininin tanımlanması
#define BUTTON P60
#define LED1   P73
//delay fonksiyonu (bu fonksiyonu header olarak da yazabilirsiniz.)
void DelayMs(unsigned char x) {
	unsigned char say,i;
	for(say=0; say<x; say++) {
		for(i=0; i<100; i++) {
NOP(); } } }
void main(void){
	P7CR=0b11110111; // port  control register  (1:INPUT, 0:OUTPUT)
	P6CR=0b00000001;
	while(1) {
		if(BUTTON==0) {
		LED1=1;}
		else {
			LED1=0;
		}
	}
}
```

4.4 Buton ile Sıralı Led Yakma

```
//button ile sıralı led yakma
#include "EM78F668N.h"
//mikroişlemci ayarları
#define DISI() _asm{disi}
#define ENI()  _asm{eni}
#define WDTC() _asm{wdtc}
#define NOP()  _asm{nop}
#define SLEP() _asm{slep}
//button pini ve led pinlerinin tanımlanması
#define BUTTON1 P60
#define LED1   P73
#define LED2   P75
#define LED3   P76
#define LED4   P74
//delay fonksiyonu (bu fonksiyonu header olarak da yazabilirsiniz.)
void DelayMs(unsigned char x) {
	unsigned char say,i;
	for(say=0; say<x; say++) {
		for(i=0; i<100; i++) {
NOP(); } } }
int s=0;
void main(void)
```

```
{
        P7CR=0b10000100; // port  control register  (1:INPUT, 0:OUTPUT)
        P6CR=0b00000001;
        while(1) {
                DelayMs(200);
                if(BUTTON1==0 && s==0) {
                        LED4=0;
                        LED1=1;
                s++; }    //s=s+1;
                else if(BUTTON1==0 && s==1) {
                        LED1=0;
                        LED2=1;
                s++; }
                else if(BUTTON1==0 && s==2) {
                        LED2=0;
                        LED3=1;
                s++; }
                else if(BUTTON1==0 && s==3) {
                        LED3=0;
                        LED4=1;
                        s=0;
                }
        }
}
```

4.5 Analog Trimpot ve Buzzer


```
//trimpottan analog değer okuma ve  buzzer
#include "EM78F668N.h"
//mikrodenetleyici  ayarları
#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}
//buzzer'ın pin tanımlaması
#define BUZZER                          P64
//trimpot ayarlarının yapılması
#define AD_TRIMPOT    0b00100001  //port6 ADC olarak kullanılmaktadır. Son 3 bit bağlı olan pini
//göstermektedir.(001-61. Pin anlamına gelmektedir.)
//5. Bit ADPD'yi kontrol eder ve bu bitin 1 olması ADC'yi çalıştırır.
//alt fonsiyonların tanımlanamsı
void AD_init(void);
void init(void);
```

```
unsigned int AD_OKU(unsigned int);
unsigned int deger=0;
//alt fonsiyonların tanımlanaması
void DelayMs(unsigned char x)
{
	unsigned char say,i;
	//WDTC();
	for(say=0;say<x;say++){
		for(i=0;i<100;i++){
		NOP(); }
	}
}
//ADC başlangıç fonksiyonu
void AD_init()
{
	ADICL = 0b00000010;  //analog input select register
}
//Başlangıç ayarlarının yapılması
void init()
{
	DISI();
	WDTC();
	OMCR=0b10111110;		//mode select
	P6CR=0b11100011;
	AD_init();
}
unsigned int AD_OKU( unsigned int CH )
{

	ADCR1=CH; //AD  control register

	ADRUN=1; //AD conversation starts
	while(ADRUN==1);

	return ADDH;
}
void main()
{
	init();
	while(1)
	{
		unsigned int deger=0;
		deger=AD_OKU(AD_TRIMPOT);  //trimpottan değer okuma
		if(deger >= 125)
		{
			BUZZER=1;
			DelayMs(deger);
			BUZZER=0;
			DelayMs(deger);
		}
		else
		BUZZER=0;
```

```
        }
}
```

4.6 Analog LDR

```
//Analog LDR
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}

#define BUZZER                          P64
#define LED1        P73
#define LED2        P75
#define LED3        P76
#define LED4        P74
#define AD_LDR          0b00100110

void AD_init(void);
void init(void);
unsigned int AD_OKU(unsigned int);
unsigned int deger=0;

void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                NOP(); }
        }
}
void AD_init()
{
        ADICL = 0b01000000;  //analog input select register
}
void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P6CR=0b11100011;
        P7CR=0b10000111;
        AD_init();
}
unsigned int AD_OKU( unsigned int CH )
```

```
{
        ADCR1=CH;
        ADRUN=1;
        while(ADRUN==1);
        return ADDH;
}
void main()  {
        init();
        while(1){
                unsigned int deger=0;
                deger=AD_OKU(AD_LDR);
                LED1=1; LED2=1; LED3=1; LED4=1;
                if(deger>100) {
                        LED1=0; LED2=0; LED3=0; LED4=0;
                        BUZZER=1;
                        DelayMs(deger);
                        BUZZER=0;
                DelayMs(deger);}
        }
}
```

4.7 Analog NTC


```
//Analog NTC
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}

#define BUZZER                          P64
#define LED1        P73
#define LED2        P75
#define LED3        P76
#define LED4        P74
#define AD_LDR          0b00100101

void AD_init(void);
void init(void);
unsigned int AD_OKU(unsigned int);
unsigned int deger=0;

void DelayMs(unsigned char x)
```

```
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                NOP(); }
        }
}
void AD_init()
{
        ADICL = 0b00100000;
}
void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P6CR=0b11100011;
        P7CR=0b10000111;
        AD_init();
}
unsigned int AD_OKU( unsigned int CH )
{
        ADCR1=CH;
        ADRUN=1;
        while(ADRUN==1);
        return ADDH;
}
void main()  {
        init();
        while(1){
                unsigned int deger=0;
                deger=AD_OKU(AD_LDR);
                LED1=1; LED2=1; LED3=1; LED4=1;
                if(deger>100) {
                        LED1=0; LED2=0; LED3=0; LED4=0;
                        BUZZER=1;
                        DelayMs(deger);
                        BUZZER=0;
                DelayMs(deger);}
        }
}
```

4.8 Papırs Sensor


```
//PaPIRS  Sensor
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}

#define BUZZER                          P64
#define LED1          P73
#define LED2          P75
#define LED3          P76
#define LED4          P74
#define PIR          P72

void init(void);
unsigned int deger=0;

void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                NOP(); }
        }
}
void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P6CR=0b11100011;
        P7CR=0b10000100;
}
void main() {
        init();
        while(1)  {
                unsigned int deger=0;
```

```
                if(PIR==1) {
                        LED1=1; LED2=1; LED3=1; LED4=1;
                        BUZZER=1;
                        DelayMs(100);
                        BUZZER=0;
                DelayMs(100); }
                else {
                        LED1=0; LED2=0; LED3=0; LED4=0;
                }

        }
}
```

4.9 Trimpot ile PWM Değişimi

```
//Trimpot ile PWM Degisimi
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   _nop_()   _asm{nop}
#define   SLEP()   _asm{slep}

#define AD_TRIMPOT    0b00100001
void AD_init(void);
void init(void);
unsigned int AD_OKU(unsigned int);
void AD_init()
{
        ADICL = 0b11100010;  //analog input select register

}
void init()
{

        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P6CR=0b11100011;
        AD_init();
}
unsigned int AD_OKU( unsigned int CH )
{
        ADCR1=CH;
        ADRUN=1;                                        //
        while(ADRUN==1);                        //
        return ADDH;
}
void main()  {
        init();
        while(1) {
                unsigned int deger=0,deger2=0;
```

```
                deger=AD_OKU(AD_TRIMPOT);
                deger2=deger/5;
                WDTC();
                DISI();
                TASS=1;
                PRDxL=0X00;             //PERIYOT LOW
                TAPRDH=0X34; //PERIYOT HIGHT
                DTxL=0X00;              //DUTY LOW
                TADTH=deger2;//DUTY HIGH
                TIMEN=0X01;             //Timer A enable
                TACR=0;
                PWMER=0X01;             //PWM A ENABLE
        }
}
```

4.10 Analog Buton

```
//butondan analog okuma
#include "EM78F668N.h"
#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}
#define BUZZER      P64
#define LED1        P73
#define LED2        P75
#define LED3        P76
#define LED4        P74
#define AD_BUTTON     0b00100111

void AD_init(void);
void init(void);
unsigned int AD_OKU(unsigned int);
unsigned int deger=0;

void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                NOP(); }
        }
}

void AD_init()
{
        ADICL = 0b10000000;
}

void init()
{
```

```
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P6CR=0b11100011;
        P7CR=0b10000111;
        AD_init();
}
unsigned int AD_OKU( unsigned int CH )
{
        ADCR1=CH;
        ADRUN=1;
        while(ADRUN==1);
        return ADDH;
}
void main()
{
        init();
        while(1)
        {
                unsigned int deger=0;
                deger=AD_OKU(AD_BUTTON);
                if (( deger >= 58 ) & ( deger <= 70 ) )  {
                        DelayMs(100);
                LED2=0;LED3=0;LED1=1;LED4=0; }
                else if (( deger >= 117 ) & ( deger <= 133))  {
                        DelayMs(100);
                LED1=0;LED3=0;LED2=1;LED4=0; }
                else if (( deger >= 0 ) & ( deger <= 25))  {
                        LED1=0;LED2=0;LED3=0;LED4=0;
                        BUZZER=1;
                        DelayMs(100);
                BUZZER=0; }
                else  {
                LED1=1;LED2=1;LED3=1;LED4=1; }
        }
}
```

4.11 Oled OZDİSAN


```
//OLED OZDISAN
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   _nop_()   _asm{nop}
#define   SLEP()   _asm{slep}

#define LCD_RS                P50
#define LCD_RW                P82
#define LCD_EN                P53
#define LCD_D4                P54
#define LCD_D5                P55
#define LCD_D6                P56
#define LCD_D7                P57

void WriteIns(unsigned char);
void WriteCmd(unsigned char);
void WriteData(unsigned char);
void Fullon(void);
void Initial_OLED(void);
void CheckBusy(void);

void init(void);
void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                _nop_(); }
        }
}

int i;
```

```
void main()
{
        init();
        Initial_OLED();
        OZDISAN();  //alt programa girer
}

void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P5CR=0b00000100;
        P8CR=0b00001000;
}
void CheckBusy(void)
{
        bit busy_f;
        LCD_D4=1;
        LCD_D5=1;
        LCD_D6=1;
        LCD_D7=1;
        LCD_RS = 0;
        LCD_RW = 1;
        P5CR=0b10000100;
        do
        {
                LCD_EN = 1;
                busy_f = LCD_D7;
                LCD_EN = 0;
                LCD_EN = 1; //dummy read
                LCD_EN = 0;
                _nop_();
        }while(busy_f);
        P5CR=0b00000100;
}
void WriteCmd(unsigned char cmd)
{
        unsigned char hIns=cmd,lIns=cmd;

        LCD_RS = 0;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hIns & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hIns & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hIns & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hIns & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
```

```
        if ((lIns<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lIns<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lIns<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lIns<<4) & 0x80) LCD_D7=1;else LCD_D7=0;


        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
        CheckBusy();

}
void WriteData(unsigned char dat)
{
        unsigned char hDat=dat,lDat=dat,ara=0,ara2=0;

        LCD_RS = 1;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hDat & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hDat & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hDat & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hDat & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        if ((lDat<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lDat<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lDat<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lDat<<4) & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        CheckBusy();

}
void Fullon(void)
{
        unsigned char i;

        //First line address
        WriteCmd(0x40);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
        //Second line address
        WriteCmd(0x41);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
```

```
}
void WriteIns(unsigned char instruction)
{
        LCD_RS = 0;
        LCD_EN = 0;
        LCD_RW = 0;
        if (instruction & 0x10) LCD_D4=1;else LCD_D4=0;
        if (instruction & 0x20) LCD_D5=1;else LCD_D5=0;
        if (instruction & 0x40) LCD_D6=1;else LCD_D6=0;
        if (instruction & 0x80) LCD_D7=1;else LCD_D7=0;
        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
}


void Initial_OLED(void)
{

        /*need to set five "0x00" cmds*/
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x20);//function set //do it only once
        WriteCmd(0x28);//function set
        WriteCmd(0x08);//display off
        WriteCmd(0x06);//entry mode set
        WriteCmd(0x1f);//Graphic mode and internel power on (have to turn on the internel power
to get the best brightness)
        WriteCmd(0x01);//clear display
        WriteCmd(0x02);
        WriteCmd(0x0c);//display on
}
//OZDISAN alt programı
void OZDISAN(void)
{
        unsigned int i;
        WriteCmd(0x40);
        for(i=0;i<=9;i++) WriteData(0x00);
        WriteData(0x00);
        WriteData(0x10);  //Logo
        WriteData(0x30);
        WriteData(0x70);
        WriteData(0x70);
        WriteData(0x70);
        WriteData(0x7f);
        WriteData(0x7e);
        WriteData(0x3c);
        WriteData(0x00);
```

```
WriteData(0x3c);
WriteData(0x7e);
WriteData(0x7f);
WriteData(0x70);
WriteData(0x70);
WriteData(0x70);
WriteData(0x30);
WriteData(0x10);
WriteData(0x00);

WriteData(0x70); //Ö
WriteData(0x78);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x78);
WriteData(0x70);
WriteData(0x00);

WriteData(0x0c); //Z
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x4c);
WriteData(0x6c);
WriteData(0x3c);
WriteData(0x1c);
WriteData(0x1c);
WriteData(0x00);

WriteData(0x7c); //D
WriteData(0x7c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x78);
WriteData(0x70);
WriteData(0x00);

WriteData(0x0c);  //i
WriteData(0x7d);
WriteData(0x7d);
WriteData(0x0c);
WriteData(0x00);

WriteData(0x18);  //s
WriteData(0x3c);
WriteData(0x6c);
```

```
WriteData(0x4c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x00);

WriteData(0xb8);  //a
WriteData(0xbc);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0x0c);
WriteData(0xbc);
WriteData(0xb8);
WriteData(0x00);

WriteData(0x7c);  //n
WriteData(0x7c);
WriteData(0x30);
WriteData(0x60);
WriteData(0x40);
WriteData(0x00);
WriteData(0x00);
WriteData(0x7c);
WriteData(0x7c);

for(i=0;i<=9;i++) WriteData(0x00);

WriteCmd(0x41);
for(i=0;i<=9;i++) WriteData(0x00);
WriteData(0x04);
WriteData(0x06);
WriteData(0x07);
WriteData(0x07);
WriteData(0x07);
WriteData(0x7f);
WriteData(0x3f);
WriteData(0x1e);
WriteData(0x00);

WriteData(0x1e);
WriteData(0x3f);
WriteData(0x7f);
WriteData(0x07);
WriteData(0x07);
WriteData(0x07);
WriteData(0x06);
WriteData(0x04);
WriteData(0x00);

WriteData(0x07);   //ö alt
```

```
WriteData(0x0f);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x0f);
WriteData(0x07);
WriteData(0x00);

WriteData(0x00);
WriteData(0x1c);
WriteData(0x1e);
WriteData(0x1b);
WriteData(0x19);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x00); //z alt

WriteData(0x1f);
WriteData(0x1f);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x0f);
WriteData(0x07);
WriteData(0x00); //d alt

WriteData(0x18);  //i   alt
WriteData(0x1f);
WriteData(0x1f);
WriteData(0x18);
WriteData(0x00);

WriteData(0x18);  //s alt
WriteData(0x18);
WriteData(0x18);
WriteData(0x18);
WriteData(0x19);
WriteData(0x1b);
WriteData(0x1e);
WriteData(0x0c);
WriteData(0x00);

WriteData(0x1f);  //a alt
WriteData(0x1f);
WriteData(0x03);
WriteData(0x03);
```

```
        WriteData(0x03);
        WriteData(0x03);
        WriteData(0x1f);
        WriteData(0x1f);
        WriteData(0x00);

        WriteData(0x1f);  //n  alt
        WriteData(0x1f);
        WriteData(0x00);
        WriteData(0x00);
        WriteData(0x00);
        WriteData(0x01);
        WriteData(0x03);
        WriteData(0x1f);
        WriteData(0x1f);
}
```

4.12 Oled MERHABA DUNYA


```
//OLED "MERHABA DUNYYA"
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   _nop_()   _asm{nop}
#define   SLEP()   _asm{slep}

#define LCD_RS          P50
#define LCD_RW          P82
#define LCD_EN          P53
#define LCD_D4          P54
#define LCD_D5          P55
#define LCD_D6          P56
#define LCD_D7          P57

void WriteIns(unsigned char);
void WriteData(unsigned char);
void Initial_OLED(void);
```

```
void CheckBusy(void);

void init(void);
void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                _nop_(); }
        }
}
void main()
{
        init();
        Initial_OLED();
        MERHABADUNYA();
}
void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P5CR=0b00000100;
        P8CR=0b00001000;
}




void CheckBusy(void)
{
        bit busy_f;
        LCD_D4=1;
        LCD_D5=1;
        LCD_D6=1;
        LCD_D7=1;
        LCD_RS = 0;
        LCD_RW = 1;
        P5CR=0b10000100;
        do
        {
                LCD_EN = 1;
                busy_f = LCD_D7;
                LCD_EN = 0;
                LCD_EN = 1; //dummy read
                LCD_EN = 0;
                _nop_();
        }while(busy_f);
        P5CR=0b00000100;
```

```
}
void WriteCmd(unsigned char cmd)
{
        unsigned char hIns=cmd,lIns=cmd;
        LCD_RS = 0;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hIns & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hIns & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hIns & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hIns & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us

        if ((lIns<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lIns<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lIns<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lIns<<4) & 0x80) LCD_D7=1;else LCD_D7=0;


        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
        CheckBusy();
}



void WriteData(unsigned char dat)
{
        unsigned char hDat=dat,lDat=dat,ara=0,ara2=0;

        LCD_RS = 1;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hDat & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hDat & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hDat & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hDat & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        if ((lDat<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lDat<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lDat<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lDat<<4) & 0x80) LCD_D7=1;else LCD_D7=0;
```

```
        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        CheckBusy();

}
void WriteIns(unsigned char instruction)
{
        LCD_RS = 0;
        LCD_EN = 0;
        LCD_RW = 0;
        if (instruction & 0x10) LCD_D4=1;else LCD_D4=0;
        if (instruction & 0x20) LCD_D5=1;else LCD_D5=0;
        if (instruction & 0x40) LCD_D6=1;else LCD_D6=0;
        if (instruction & 0x80) LCD_D7=1;else LCD_D7=0;
        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
}
void Initial_OLED(void)
{
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x20);//function set //do it only once
        WriteCmd(0x28);//function set
        WriteCmd(0x08);//display off
        WriteCmd(0x06);//entry mode set
        WriteCmd(0x1f);//Graphic mode and internel power on (have to turn on the internel power
to get the best brightness)
        WriteCmd(0x01);//clear display
        WriteCmd(0x02);
        WriteCmd(0x0c);//display on
}
void MERHABADUNYA(void) {
        //MERHABADUNYA üst
        unsigned int i;
        for(i=0; i<=11; i++) {
                WriteData(0x00);
        }

        WriteData(0xF0); //M UST
        WriteData(0x60);
        WriteData(0xC0);
        WriteData(0x60);
        WriteData(0xF0);
        WriteData(0x00);

        WriteData(0xF0); //E UST
```

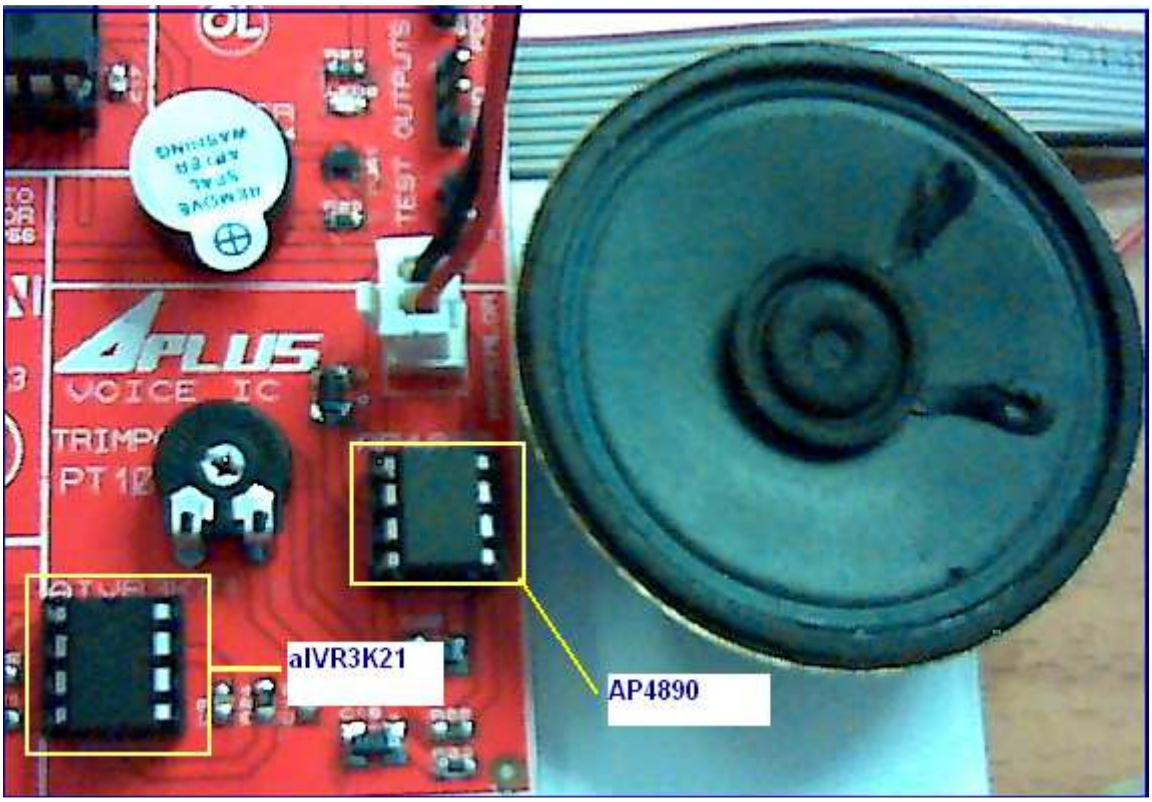
```
WriteData(0x90);
WriteData(0x90);
WriteData(0x90);
WriteData(0x10);
WriteData(0x00);

WriteData(0xF0);
WriteData(0x90);  //R UST
WriteData(0x90);
WriteData(0x90);
WriteData(0x60);
WriteData(0x00);

WriteData(0xF0);  //H UST
WriteData(0x80);
WriteData(0x80);
WriteData(0x80);
WriteData(0xF0);
WriteData(0x00);

WriteData(0xE0);  //A UST
WriteData(0x10);
WriteData(0x10);
WriteData(0x10);
WriteData(0xE0);
WriteData(0x00);

WriteData(0xF0);  //B UST
WriteData(0x90);
WriteData(0x90);
WriteData(0x90);
WriteData(0x60);
WriteData(0x00);

WriteData(0xE0);  //A UST
WriteData(0x10);
WriteData(0x10);
WriteData(0x10);
WriteData(0xE0);
WriteData(0x00);

WriteData(0x00);
WriteData(0x00);
WriteData(0x00);
WriteData(0x00);

WriteData(0xF0);  //D UST
WriteData(0x10);
WriteData(0x10);
WriteData(0x60);
WriteData(0xC0);
WriteData(0x00);
```

```
WriteData(0xE0);  //Ü ÜST
WriteData(0x10);
WriteData(0x00);
WriteData(0x10);
WriteData(0xE0);
WriteData(0x00);

WriteData(0xF0);  //N UST
WriteData(0x40);
WriteData(0x80);
WriteData(0x00);
WriteData(0xF0);
WriteData(0x00);

WriteData(0xF0);  //N UST
WriteData(0x00);
WriteData(0x00);
WriteData(0x00);
WriteData(0xF0);
WriteData(0x00);

WriteData(0xE0);  //A UST
WriteData(0x10);
WriteData(0x10);
WriteData(0x10);
WriteData(0xE0);
WriteData(0x00);

for(i=0; i<=23; i++) {
        WriteData(0x00);
}

//ALT KISIM
WriteData(0x07); //M ALT
WriteData(0x00);
WriteData(0x01);
WriteData(0x00);
WriteData(0x07);
WriteData(0x00);

WriteData(0x07);  //E ALT
WriteData(0x04);
WriteData(0x04);
WriteData(0x04);
WriteData(0x04);
WriteData(0x00);

WriteData(0x07);  //R ALT
WriteData(0x00);
WriteData(0x01);
WriteData(0x02);
```

```
WriteData(0x04);
WriteData(0x00);

WriteData(0x07);  //H ALT
WriteData(0x00);
WriteData(0x00);
WriteData(0x00);
WriteData(0x07);
WriteData(0x00);

WriteData(0x07); //A ALT
WriteData(0x01);
WriteData(0x01);
WriteData(0x01);
WriteData(0x07);
WriteData(0x00);

WriteData(0x07);  //B ALT
WriteData(0x04);
WriteData(0x04);
WriteData(0x04);
WriteData(0x03);
WriteData(0x00);

WriteData(0x07); //A ALT
WriteData(0x01);
WriteData(0x01);
WriteData(0x01);
WriteData(0x07);
WriteData(0x00);

WriteData(0x00);
WriteData(0x00);
WriteData(0x00);
WriteData(0x00);

WriteData(0x07);  //D ALT
WriteData(0x04);
WriteData(0x04);
WriteData(0x06);
WriteData(0x03);
WriteData(0x00);

WriteData(0x07);  //Ü ALT
WriteData(0x04);
WriteData(0x04);
WriteData(0x04);
WriteData(0x07);
WriteData(0x00);

WriteData(0x07);  //N ALT
WriteData(0x00);
```

```
        WriteData(0x00);
        WriteData(0x01);
        WriteData(0x07);
        WriteData(0x00);

        WriteData(0x00);  //Y ALT
        WriteData(0x01);
        WriteData(0x06);
        WriteData(0x01);
        WriteData(0x00);
        WriteData(0x00);

        WriteData(0x07); //A ALT
        WriteData(0x01);
        WriteData(0x01);
        WriteData(0x01);
        WriteData(0x07);
        WriteData(0x00);
}
```

4.13 Buton ile Ses Kontrolü




```
//Button ile ses kontrolü
#include "EM78F668N.h"
//mikrodenetleyici ayarları
#define DISI() _asm{disi}
#define ENI()  _asm{eni}
#define WDTC() _asm{wdtc}
#define NOP()  _asm{nop}
#define SLEP() _asm{slep}
#define BUTTON1 P60
#define LED1   P73
#define LED2   P75
#define LED3   P76
#define LED4   P74
#define SES1   P62
#define SES2   P63
//delay fonksiyonu (bu fonksiyonu header olarak da yazabilirsiniz.)
void DelayMs(unsigned char x) {
        unsigned char say,i;
        for(say=0; say<x; say++) {
                for(i=0; i<100; i++) {
NOP(); } } }
int s=0;
void main(void)
{
        P7CR=0b10000100; // port  control register  (1:INPUT, 0:OUTPUT)
        P6CR=0b00001001;
        while(1) {
                DelayMs(300);
```

```
                if(BUTTON1==0 && s==0) {
                        P6CR=0b00001001;
                        SES1=1;
                        LED3=0;
                        LED1=1;
                s++; }    //s=s+1;
                else if(BUTTON1==0 && s==1) {
                        P6CR=0b00000101;
                        SES2=1;
                        LED1=0;
                        LED2=1;
                s++; }
                else if(BUTTON1==0 && s==2) {
                        P6CR=0b00000001;
                        SES1=1;
                        SES2=1;
                        LED2=0;
                        LED3=1;
                s=0; }
        }
}
```

4.14 Oled Harf

/* bu program harflerin tanımlı olduğu programdır. C uzantılı dosya olarak kullanılmaktadır. Oled ile oluşturduğunuz herhangi bir programda eğer harf kütüphanesine ihtiyacınız varsa mevcut c dosyanıza ek olarak yeni bir c dosyası oluşturup bu programı oraya kopyalayabilirsiniz. */

```
void A_(void)                                                   //(A)
{
        WriteData(0x7e);
        WriteData(0x11);
        WriteData(0x11);
        WriteData(0x11);
        WriteData(0x7e);
        WriteData(0x00);
}

void B_(void)                                                   //(B)
{
        WriteData(0x7F);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x36);
        WriteData(0x00);
}

void C_(void)                                                   //(C)
{
        WriteData(0x3E);
```

```
        WriteData(0x41);
        WriteData(0x41);
        WriteData(0x41);
        WriteData(0x22);
        WriteData(0x00);
}

void D_(void)                                   //(D)
{
        WriteData(0x7F);
        WriteData(0x41);
        WriteData(0x41);
        WriteData(0x22);
        WriteData(0x1c);
        WriteData(0x00);
}

void E_(void)                                   //(E)
{
        WriteData(0x7F);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x41);
        WriteData(0x00);
}

void F_(void)                                   //(F)
{
        WriteData(0x7F);
        WriteData(0x09);
        WriteData(0x09);
        WriteData(0x09);
        WriteData(0x01);
        WriteData(0x00);

}

void G_(void)                                   //(G)
{
        WriteData(0x3e);
        WriteData(0x41);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x7a);
        WriteData(0x00);
}

void H_(void)                                   //(H)
{
        WriteData(0x7F);
        WriteData(0x08);
```

```
        WriteData(0x08);
        WriteData(0x08);
        WriteData(0x7F);
        WriteData(0x00);
}

void I_(void)                                   //(I)
{
        WriteData(0x00);
        WriteData(0x41);
        WriteData(0x7F);
        WriteData(0x41);
        WriteData(0x00);
        WriteData(0x00);
}

void J_(void)                                   //(J)
{
        WriteData(0x20);
        WriteData(0x40);
        WriteData(0x41);
        WriteData(0x3f);
        WriteData(0x01);
        WriteData(0x00);
}

void K_(void)                                   //(K)
{
        WriteData(0x7f);
        WriteData(0x08);
        WriteData(0x14);
        WriteData(0x22);
        WriteData(0x41);
        WriteData(0x00);

}

void L_(void)                                   //(L)
{
        WriteData(0x7F);
        WriteData(0x40);
        WriteData(0x40);
        WriteData(0x40);
        WriteData(0x40);
        WriteData(0x00);
}

void M_(void)                                   //(M)
{
        WriteData(0x7F);
        WriteData(0x02);
        WriteData(0x0c);
```

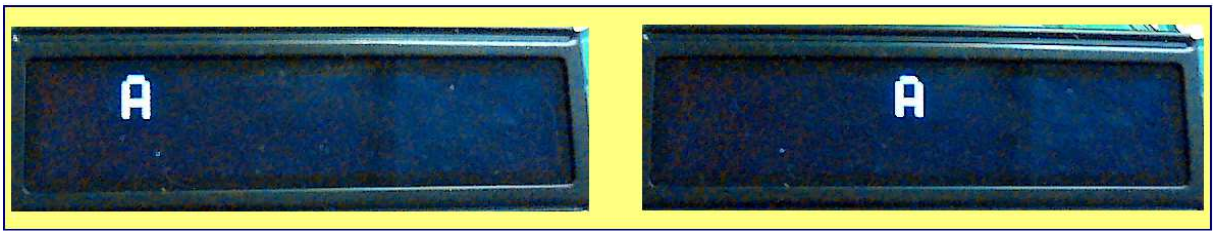
```
        WriteData(0x02);
        WriteData(0x7f);
        WriteData(0x00);
}

void N_(void)                                   //(N)
{
        WriteData(0x7F);
        WriteData(0x04);
        WriteData(0x08);
        WriteData(0x10);
        WriteData(0x7f);
        WriteData(0x00);
}

void O_(void)                                   //(O)
{
        WriteData(0x3e);
        WriteData(0x41);
        WriteData(0x41);
        WriteData(0x41);
        WriteData(0x3e);
        WriteData(0x00);
}

void P_(void)                                   //(P)
{
        WriteData(0x7f);
        WriteData(0x09);
        WriteData(0x09);
        WriteData(0x09);
        WriteData(0x06);
        WriteData(0x00);
}

void R_(void)                                   //(R)
{
        WriteData(0x7f);
        WriteData(0x09);
        WriteData(0x19);
        WriteData(0x29);
        WriteData(0x46);
        WriteData(0x00);
}

void S_(void)                                   //(S)
{
        WriteData(0x46);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x49);
        WriteData(0x31);
```

```
        WriteData(0x00);
}

void T_(void)                                   //(T)
{
        WriteData(0x01);
        WriteData(0x01);
        WriteData(0x7f);
        WriteData(0x01);
        WriteData(0x01);
        WriteData(0x00);
}

void U_(void)                                   //(U)
{
        WriteData(0x3f);
        WriteData(0x40);
        WriteData(0x40);
        WriteData(0x40);
        WriteData(0x3f);
        WriteData(0x00);
}

void V_(void)                                   //(V)
{
        WriteData(0x1F);
        WriteData(0x20);
        WriteData(0x40);
        WriteData(0x20);
        WriteData(0x1f);
        WriteData(0x00);
}

void Y_(void)                                   //(Y)
{
        WriteData(0x07);
        WriteData(0x08);
        WriteData(0x70);
        WriteData(0x08);
        WriteData(0x07);
        WriteData(0x00);
}

void Z_(void)                                   //(Z)
{
        WriteData(0x61);
        WriteData(0x51);
        WriteData(0x49);
        WriteData(0x45);
        WriteData(0x43);
        WriteData(0x00);
}
```

```
void CC_(void)                                          //(Ç)
{
        WriteData(0x3e);
        WriteData(0x41);
        WriteData(0xc1);
        WriteData(0x41);
        WriteData(0x22);
        WriteData(0x00);
}

void II_(void)                                          //(İ)
{
        WriteData(0x00);
        WriteData(0x44);
        WriteData(0x7d);
        WriteData(0x44);
        WriteData(0x00);
        WriteData(0x00);
}

void OO_(void)                                          //(Ö)
{
        WriteData(0x38);
        WriteData(0x45);
        WriteData(0x44);
        WriteData(0x45);
        WriteData(0x38);
        WriteData(0x00);
}

void SS_(void)                                          //(Ş)
{
        WriteData(0x46);
        WriteData(0xc9);
        WriteData(0xc9);
        WriteData(0x49);
        WriteData(0x31);
        WriteData(0x00);
}

void UU_(void)                                          //(Ü)
{
        WriteData(0x3c);
        WriteData(0x41);
        WriteData(0x40);
        WriteData(0x41);
        WriteData(0x3c);
        WriteData(0x00);
}
```

/* bu programı yükledikten sonra sadece programınızın main kısmında bu alt programları çağırabilirsiniz. Mesela void main { UU_(); } yazarsanız ekrana ü harfini yazmış olursunuz. */

4.15 Oled Buton ile Harf Hareketi




```
//Button ile OLED LCD Harf Hareket ettirme
//Buuton 1-sağa hareket, Button2 Yukarı aşağı hareket
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   _nop_()   _asm{nop}
#define   SLEP()   _asm{slep}

#define LCD_RS          P50
#define LCD_RW          P82
#define LCD_EN          P53
#define LCD_D4          P54
#define LCD_D5          P55
#define LCD_D6          P56
#define LCD_D7          P57
#define BUTTON1    P60
#define BUTTON2    P77
#define BUTTON3    P83


void WriteIns(unsigned char);
void WriteCmd(unsigned char);
void WriteData(unsigned char);
void Fullon(void);
void Initial_OLED(void);
void CheckBusy(void);

void init(void);
void DelayMs(unsigned char x)
{
        unsigned char say,i;
```

```
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                _nop_(); }
        }
}

int i=0;

void main()

{
        P6CR=0b11100011;
        P7CR=0b10000100;
        P8CR=0b00001000;
        init();
        Initial_OLED();
        A_();
        while(1){
                DelayMs(500); //hızlı ise arttırarak yavaşlatabilirsiniz.
                if(BUTTON1==0) {
                        WriteCmd(0x01); //LCD yi temizle
                        A_(); //A yaz
                }
                else if(BUTTON2==0 & i==0) {
                        WriteCmd(0x41);  //Alt satıra geç
                        i++;
                }
                else if(BUTTON2==0 & i==1) {
                        WriteCmd(0x40);  //üst satıra geç
                        i=0;
                }

        }



}

void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P5CR=0b00000100;
```

```
        P8CR=0b00001000;
}
void CheckBusy(void)
{
        bit busy_f;
        LCD_D4=1;
        LCD_D5=1;
        LCD_D6=1;
        LCD_D7=1;
        LCD_RS = 0;
        LCD_RW = 1;
        P5CR=0b10000100;
        do
        {
                LCD_EN = 1;
                busy_f = LCD_D7;
                LCD_EN = 0;
                LCD_EN = 1; //dummy read
                LCD_EN = 0;
                _nop_();
        }while(busy_f);
        P5CR=0b00000100;
}
void WriteCmd(unsigned char cmd)
{
        unsigned char hIns=cmd,lIns=cmd;

        LCD_RS = 0;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hIns & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hIns & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hIns & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hIns & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us

        if ((lIns<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lIns<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lIns<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lIns<<4) & 0x80) LCD_D7=1;else LCD_D7=0;
```

```
        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
        CheckBusy();

}
void WriteData(unsigned char dat)
{
        unsigned char hDat=dat,lDat=dat,ara=0,ara2=0;

        LCD_RS = 1;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hDat & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hDat & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hDat & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hDat & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        if ((lDat<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lDat<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lDat<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lDat<<4) & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        CheckBusy();

}
void Fullon(void)
{
        unsigned char i;

        //First line address
        WriteCmd(0x40);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
        //Second line address
        WriteCmd(0x41);//Y
        for(i = 0; i<100;i++)
```

```
        WriteData(0xff);
}
void WriteIns(unsigned char instruction)
{
        LCD_RS = 0;
        LCD_EN = 0;
        LCD_RW = 0;
        if (instruction & 0x10) LCD_D4=1;else LCD_D4=0;
        if (instruction & 0x20) LCD_D5=1;else LCD_D5=0;
        if (instruction & 0x40) LCD_D6=1;else LCD_D6=0;
        if (instruction & 0x80) LCD_D7=1;else LCD_D7=0;
        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
}


void Initial_OLED(void)
{

        /*need to set five "0x00" cmds*/
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x20);//set DDRAM Address
        WriteCmd(0x28);//set CGRAM Address
        WriteCmd(0x08);//display off
        WriteCmd(0x06);//entry mode set
        WriteCmd(0x1f);//Graphic mode and internel power on (have to turn on the internel power
to get the best brightness)
        WriteCmd(0x01);//clear display
        WriteCmd(0x02);
        WriteCmd(0x0c);//display on
}
```

/*Ana programımıza bir ek program daha yapmamız gerekmektedir. Eğer dosyamız proje dosyamız içerisinde varsa “Source Files”’e sağ tıklayıp “Add files” dedikten sonra dosyayı ekleyebiliriz. Ama mevcut değilse, “File” sekmesine basarak yeni bir c uzantılı dosya oluşruracağız. Bu dosyaya “harf” ismini vermemizde fayda var. Çünkü harflerimiz bu dosyada yüklü olacaktır. Boş sayfayı görünce harf dodlarını kopyalayıp, oraya yapıştırıyoruz ve “Rebuild All” ‘a basıyoruz. */

4.16 Oled Button Harf Seçimi


```
/*Bu programda button1 ile harf seçerken buton2 ile harf seçimimizi yaptıktan sonra da sağa gider.
Eğer sola da gitmesini istiyorsanız if(BUTTON3==0) { yatay=yatay-0x06; i=0; } kodunu ekleyebilirsiniz.
*/
//Button harf secimi,
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   _nop_()   _asm{nop}
#define   SLEP()   _asm{slep}

#define LCD_RS          P50
#define LCD_RW          P82
#define LCD_EN          P53
#define LCD_D4          P54
#define LCD_D5          P55
#define LCD_D6          P56
#define LCD_D7          P57
#define BUTTON1    P60
#define BUTTON2    P77
#define BUTTON3    P83


void WriteIns(unsigned char);
void WriteCmd(unsigned char);
void WriteData(unsigned char);
void Fullon(void);
void Initial_OLED(void);
void CheckBusy(void);

void init(void);
void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                _nop_(); }
        }
}

int i=0;
int s=0;
```

```
int yatay=0x80;  //0x80 lcd nin ilk satırını temsil etmektedir. Bu sayıyı arttırararak lcd satur hareketi
sağlanır
void main(){
        init();
        Initial_OLED();
        while(1) {
                DelayMs(500); //butonun algılaması için bekliyoruz
                if(BUTTON1==0 & i==0) {
                        WriteCmd(yatay);
                        A_();
                i++; }
                else if(BUTTON1==0 & i==1) {
                        WriteCmd(yatay);
                        B_();
                i++; }
                else if(BUTTON1==0 & i==2) {
                        WriteCmd(yatay);
                        C_();
                i++; }
                else if(BUTTON1==0 & i==3) {
                        WriteCmd(yatay);
                        CC_();
                i++; }
                else if(BUTTON1==0 & i==4) {
                        WriteCmd(yatay);
                        D_();
                i++; }
                else if(BUTTON1==0 & i==5) {
                        WriteCmd(yatay);
                        E_();
                i++; }
                else if(BUTTON1==0 & i==6) {
                        WriteCmd(yatay);
                        F_();
                i++; }
                else if(BUTTON1==0 & i==7) {
                        WriteCmd(yatay);
                        G_();
                i++; }
                else if(BUTTON1==0 & i==8) {
                        WriteCmd(yatay);
                        H_();
                i++; }
                else if(BUTTON1==0 & i==9) {
                        WriteCmd(yatay);
                        I_();
                i++; }
                else if(BUTTON1==0 & i==10) {
                        WriteCmd(yatay);
                        II_();
                i++; }
                else if(BUTTON1==0 & i==11) {
```

```
            WriteCmd(yatay);
            J_();
i++; }
else if(BUTTON1==0 & i==12) {
            WriteCmd(yatay);
            K_();
i++; }
else if(BUTTON1==0 & i==13) {
            WriteCmd(yatay);
            L_();
i++; }
else if(BUTTON1==0 & i==14) {
            WriteCmd(yatay);
            M_();
i++; }
else if(BUTTON1==0 & i==15) {
            WriteCmd(yatay);
            N_();
i++; }
else if(BUTTON1==0 & i==16) {
            WriteCmd(yatay);
            O_();
i++; }
else if(BUTTON1==0 & i==17) {
            WriteCmd(yatay);
            OO_();
i++; }
else if(BUTTON1==0 & i==18) {
            WriteCmd(yatay);
            P_();
i++; }
else if(BUTTON1==0 & i==19) {
            WriteCmd(yatay);
            R_();
i++; }
else if(BUTTON1==0 & i==20) {
            WriteCmd(yatay);
            S_();
i++; }
else if(BUTTON1==0 & i==21) {
            WriteCmd(yatay);
            SS_();
i++; }
else if(BUTTON1==0 & i==22) {
            WriteCmd(yatay);
            T_();
i++; }
else if(BUTTON1==0 & i==23) {
            WriteCmd(yatay);
            U_();
i++; }
else if(BUTTON1==0 & i==24) {
```

```
                WriteCmd(yatay);
                UU_();
        i++; }
        else if(BUTTON1==0 & i==25) {
                WriteCmd(yatay);
                V_();
        i++; }
        else if(BUTTON1==0 & i==26) {
                WriteCmd(yatay);
                Y_();
        i++; }
        else if(BUTTON1==0 & i==27) {
                WriteCmd(yatay);
                Z_();
        i=0; }
        else if(BUTTON2==0) {
                yatay=yatay+0x06; // Harfler 5 karakter ve bir boşluktan oluştuğundan 6
karakter olarak hareket ediyoruz.
        i=0;}
    }
}

void init()
{
    DISI();
    WDTC();
    OMCR=0b10111110;        //mode select
    P5CR=0b00000100;
    P8CR=0b00001000;
    P6CR=0b11100011;
    P7CR=0b10000100;

}
void CheckBusy(void)
{
    bit busy_f;
    LCD_D4=1;
    LCD_D5=1;
    LCD_D6=1;
    LCD_D7=1;
    LCD_RS = 0;
    LCD_RW = 1;
    P5CR=0b10000100;
    do
    {
        LCD_EN = 1;
        busy_f = LCD_D7;
        LCD_EN = 0;
        LCD_EN = 1; //dummy read
        LCD_EN = 0;
        _nop_();
    }while(busy_f);
```

```
        P5CR=0b00000100;
}
void WriteCmd(unsigned char cmd)
{
        unsigned char hIns=cmd,lIns=cmd;

        LCD_RS = 0;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hIns & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hIns & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hIns & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hIns & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us

        if ((lIns<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lIns<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lIns<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lIns<<4) & 0x80) LCD_D7=1;else LCD_D7=0;


        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
        CheckBusy();

}
void WriteData(unsigned char dat)
{
        unsigned char hDat=dat,lDat=dat,ara=0,ara2=0;

        LCD_RS = 1;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hDat & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hDat & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hDat & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hDat & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        if ((lDat<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lDat<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lDat<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lDat<<4) & 0x80) LCD_D7=1;else LCD_D7=0;
```

```
        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        CheckBusy();

}
void Fullon(void)
{
        unsigned char i;

        //First line address
        WriteCmd(0x40);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
        //Second line address
        WriteCmd(0x41);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
}
void WriteIns(unsigned char instruction)
{
        LCD_RS = 0;
        LCD_EN = 0;
        LCD_RW = 0;
        if (instruction & 0x10) LCD_D4=1;else LCD_D4=0;
        if (instruction & 0x20) LCD_D5=1;else LCD_D5=0;
        if (instruction & 0x40) LCD_D6=1;else LCD_D6=0;
        if (instruction & 0x80) LCD_D7=1;else LCD_D7=0;
        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
}


void Initial_OLED(void)
{

        /*need to set five "0x00" cmds*/
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x20);//set DDRAM Address
        WriteCmd(0x28);//set CGRAM Address
        WriteCmd(0x08);//display off
        WriteCmd(0x06);//entry mode set
        WriteCmd(0x1f);//Graphic mode and internel power on (have to turn on the internel power
to get the best brightness)
        WriteCmd(0x01);//clear display
        WriteCmd(0x02);
        WriteCmd(0x0c);//display on
```

}

/* lcd ayarlarını datasheeti inceleyerek değiştirebilirsiniz. WriteCmd(0x80) ve bu hexa sayıdan daha büyük sayılar. Xkoordinatı boyunca konumu, WriteCmd(0x40) ve WriteCmd(0x41) y koordinatı boyunca konumu gösterir. X ekseninde 100 karakter konumu varken y ekseninde 2 konum vardır. Çünkü LCD 100*16 karakterdir ve veriler 8-bit olarak yazdırılmaktadır. Yukarıdaki programa ek olarak harf.c dosyasını eklemeniz gerekmektedir. Eğer böyle bir dosya yoksa yeni bir dosya açıp aşağıdaki programı oraya kopyalayın. Programa ek olarak "EM78F668N.h" header dosyasını eklemeniz gerekmektedir. Bu dosyada mikrodenetleyici ile ilgili ayarlar bulunmaktadır. Bu dosyayı eklemek için Header Files'e sağ tıklayın ve dosyayı ekleyin. Eğer mevcut değilse bu isimle yeni bir dosya oluşturun ve hedef belgenin içindekileri oraya kopyalayın. */

4.17 LDR UART Data

//LDR den alınan datayı seri port uart haberleşmesi ile gönderiyoruz

```
#include "EM78F668N.h"
#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}

#define LED1 P73
#define TX   P51
#define RX   P52
#define AD_LDR          0b00100110  //analog pin P66, analog pinler Port60 ta bulunmaktadır.

void UART_init(void);
unsigned int AD_OKU(unsigned int);
void AD_init(void);

void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<100;i++){
                NOP(); }
        }
}
```

```
void main(void) {
        UART_init();  //UART register ayarları
        AD_init();     //ADC register ayarları
        P5CR=0b00000100;
        P7CR=0b11110111;

        while(1) {

                URTD=AD_OKU(AD_LDR);
                URS=0b11000000;
                ISR2=0b00000000;
                URCR1=0b10100011;
                LED1=0;
        }
}
void UART_init() {
        URCR1=0b10100010;
        URCR2=0b00100001;
        URS=0b11000000;
        IMR2=0b00000111;
        ISR2=0b00000000;

}
void AD_init()
{
        ADICL = 0b01000000;
}
unsigned int AD_OKU( unsigned int CH )
{
        ADCR1=CH;
        ADRUN=1;
        while(ADRUN==1);
        return ADDH;
}
```

4.18 UART RX

```
//uart haberleşme gönderilen datayı test etme, ledin yanıp yanmadığını kontrol ederek test
#include "EM78F668N.h"
#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   NOP()    _asm{nop}
#define   SLEP()   _asm{slep}



#define LED1 P73
#define TX   P51
#define RX   P52

void UART_init(void);
```

```
void DelayMs(unsigned char x)
{
	unsigned char say,i;
	//WDTC();
	for(say=0;say<x;say++){
		for(i=0;i<100;i++){
		NOP(); }
	}
}



void main(void) {
	UART_init();   //UART register ayarları
	P5CR=0b00000100;
	P7CR=0b11110111;

	while(1) {


		URS=0b11000001;
		ISR2=0b00000000;
		URCR1=0b10100010;
		LED1=0;
		if(URRD==0b10101010) {
LED1=1;}}}
void UART_init() {
	URCR1=0b10100010;
	URCR2=0b00100001;
	URS=0b11000000;
	IMR2=0b00000111;
	ISR2=0b00000000;
}
```

4.19 Grafik Oled

```
#include "EM78F668N.h"

#define   DISI()   _asm{disi}
#define   ENI()    _asm{eni}
#define   WDTC()   _asm{wdtc}
#define   _nop_()   _asm{nop}
#define   SLEP()   _asm{slep}

#define LCD_RS                P50
#define LCD_RW                P82
#define LCD_EN                P53
#define LCD_D4                P54
#define LCD_D5                P55
#define LCD_D6                P56
#define LCD_D7                P57

void WriteIns(unsigned char);
void WriteCmd(unsigned char);
void WriteData(unsigned char);
void Fullon(void);
void Initial_OLED(void);
void CheckBusy(void);

void init(void);
void DelayMs(unsigned char x)
{
        unsigned char say,i;
        //WDTC();
        for(say=0;say<x;say++){
                for(i=0;i<200;i++){
                _nop_(); }
        }
}
void main()
{
        init();
        Initial_OLED();
        aglayanyuz();
        WriteData(0x00);
        ev();
```

```
        WriteData(0x00);
        araba();
        WriteData(0x00);
        gulenyuz();
}
void init()
{
        DISI();
        WDTC();
        OMCR=0b10111110;        //mode select
        P5CR=0b00000100;
        P8CR=0b00001000;
}
void CheckBusy(void)
{
        bit busy_f;
        LCD_D4=1;
        LCD_D5=1;
        LCD_D6=1;
        LCD_D7=1;
        LCD_RS = 0;
        LCD_RW = 1;
        P5CR=0b10000100;
        do
        {
                LCD_EN = 1;
                busy_f = LCD_D7;
                LCD_EN = 0;
                LCD_EN = 1; //dummy read
                LCD_EN = 0;
                _nop_();
        }while(busy_f);
        P5CR=0b00000100;
}
void WriteCmd(unsigned char cmd)
{
        unsigned char hIns=cmd,lIns=cmd;

        LCD_RS = 0;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hIns & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hIns & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hIns & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hIns & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us

        if ((lIns<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
```

```
        if ((lIns<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lIns<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lIns<<4) & 0x80) LCD_D7=1;else LCD_D7=0;


        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
        CheckBusy();

}
void WriteData(unsigned char dat)
{
        unsigned char hDat=dat,lDat=dat,ara=0,ara2=0;

        LCD_RS = 1;
        LCD_RW = 0;
        LCD_EN = 0;

        if (hDat & 0x10) LCD_D4=1;else LCD_D4=0;
        if (hDat & 0x20) LCD_D5=1;else LCD_D5=0;
        if (hDat & 0x40) LCD_D6=1;else LCD_D6=0;
        if (hDat & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        if ((lDat<<4) & 0x10) LCD_D4=1;else LCD_D4=0;
        if ((lDat<<4) & 0x20) LCD_D5=1;else LCD_D5=0;
        if ((lDat<<4) & 0x40) LCD_D6=1;else LCD_D6=0;
        if ((lDat<<4) & 0x80) LCD_D7=1;else LCD_D7=0;

        LCD_EN = 1;
        _nop_();
        LCD_EN = 0;
        CheckBusy();

}
void Fullon(void)
{
        unsigned char i;

        //First line address
        WriteCmd(0x40);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
        //Second line address
        WriteCmd(0x41);//Y
        for(i = 0; i<100;i++)
        WriteData(0xff);
}
void WriteIns(unsigned char instruction)
```

```
{
        LCD_RS = 0;
        LCD_EN = 0;
        LCD_RW = 0;
        if (instruction & 0x10) LCD_D4=1;else LCD_D4=0;
        if (instruction & 0x20) LCD_D5=1;else LCD_D5=0;
        if (instruction & 0x40) LCD_D6=1;else LCD_D6=0;
        if (instruction & 0x80) LCD_D7=1;else LCD_D7=0;
        LCD_EN = 1;         //1us
        _nop_();                //1us
        LCD_EN = 0;             //1us
}


void Initial_OLED(void)
{

        /*need to set five "0x00" cmds*/
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x00);
        WriteIns(0x20);//function set //do it only once
        WriteCmd(0x28);//function set
        WriteCmd(0x08);//display off
        WriteCmd(0x06);//entry mode set
        WriteCmd(0x1f);//Graphic mode and internel power on (have to turn on the internel power
to get the best brightness)
        WriteCmd(0x01);//clear display
        WriteCmd(0x02);
        WriteCmd(0x0c);//display on
}
/*Grafikleri harf gibi farklı dosyada saklayabilirsiniz. Kullanacağınız zaman programın mevcut c
dosyasına yeni bir c dosyası ekleyerek oraya yazabilirsiniz. Bu şekilde daha kulay kullanım sağlanmış
olur. */

void gulenyuz(void) {
unsigned int i;
        WriteData(0x18);
        WriteData(0x7E);
        WriteData(0xFF);
        WriteData(0xFB);
        WriteData(0xEB);
        WriteData(0xDB);
        WriteData(0xBB);
for(i=0; i<5; i++) {
        WriteData(0xBF);
        }
        WriteData(0xDB);
        WriteData(0xEB);
        WriteData(0xFB);
```

```
        WriteData(0xFF);
        WriteData(0x7E);
        WriteData(0x18);



}
void aglayanyuz(void) {
unsigned int i;
        WriteData(0x18);
        WriteData(0x7E);
        WriteData(0xFF);
        WriteData(0xFB);
        WriteData(0xBB);
        WriteData(0xDB);
        WriteData(0xEB);


for(i=0; i<5; i++) {
        WriteData(0xEF);
        }
  WriteData(0xEB);
        WriteData(0xDB);
        WriteData(0xBB);
        WriteData(0xFB);
        WriteData(0xFF);
        WriteData(0x7E);
        WriteData(0x18);
        }

void ev(void) {
WriteData(0x20);
WriteData(0xF0);
WriteData(0xF8);
WriteData(0xDC);
WriteData(0xFE);
WriteData(0xBF);
WriteData(0xFE);
WriteData(0xDC);
WriteData(0xF8);
WriteData(0xF0);
WriteData(0x20);
}

void araba(void) {

WriteData(0x38);
WriteData(0x38);
WriteData(0x38);
WriteData(0x7C);
WriteData(0xFE);
WriteData(0xFF);
```

```
WriteData(0xFF);
WriteData(0x73);
WriteData(0x33);
WriteData(0x33);
WriteData(0x3F);
WriteData(0x3F);
WriteData(0x33);
WriteData(0x33);
WriteData(0x73);
WriteData(0xFF);
WriteData(0xFF);
WriteData(0xFE);
WriteData(0x7C);
WriteData(0x38);
WriteData(0x38);
WriteData(0x38);
}
```