

ARMap-18

ARM UYGULAMALARI DENEY FÖYÜ



Ankara-2021

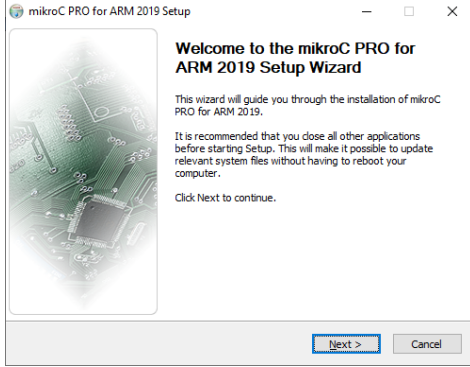
Mehmet Ali DAL

İÇİNDEKİLER

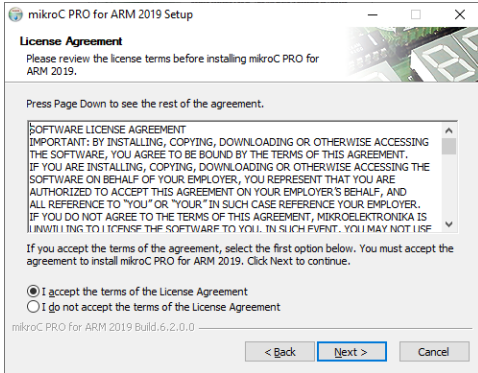
Uygulamalar için Gerekli Ön Hazırlıklar:.....	3
Uygulama 1: Ledleri Yakıp Söndürmek.....	6
Uygulama 2: Anahtarlar ve Ledler.....	9
Uygulama 3: Menü Tuş Takımı.....	12
Uygulama 4: 7 Segment Display Uygulaması.....	17
Uygulama 5: 2x16 Karakter LCD Uygulaması.....	21
Uygulama 6: Grafik LCD Uygulaması.....	25
Uygulama 7: L293D Entegresi ile DC Motor Sürme Uygulaması.....	30
Uygulama 8: Timer ile Yazılımsal PWM Uygulaması.....	34
Uygulama 9: LM35 ile Sıcaklık Ölçümü Uygulaması.....	38
Uygulama 10: Oled Display Uygulaması.....	42
Uygulama 11: Buzzer Uygulaması.....	46
Uygulama 12: Seri İletişim Uygulaması.....	50
Uygulama 13: Bluetooth ile Röle Kontrol Uygulaması.....	54
Uygulama 14: DS18B20 1-Wire Sıcaklık Sensörü Uygulaması.....	57
Uygulama 15: Harici EEPROM Uygulaması.....	60
Uygulama 16: Rezistif Dokunmatik Ekran Uygulaması.....	64
Uygulama 17: mikroBUS – 8x8 Led Matrix Display Uygulaması.....	68
Uygulama 18: mikroBUS-Gsm2Click İle Led Yakma Uygulaması.....	77

Uygulamalar için Gerekli Ön Hazırlıklar:

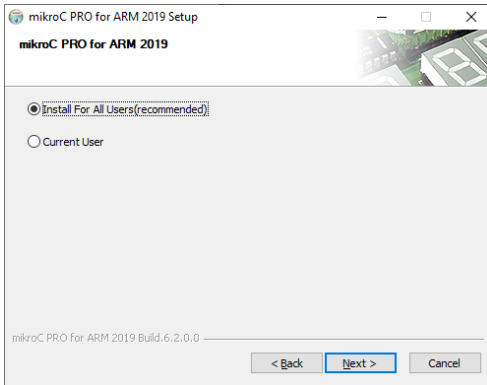
- 1- MikroC Pro for ARM v6.2 derleyicisini [buradan](#) bilgisayarınıza indiriniz.
- 2- Zip dosyası içerisindeki “setup” dosyasını çıkartarak çalıştırınız.
- 3- Resimdeki gibi next butonuna tıklayarak kurulumu devam ediniz.



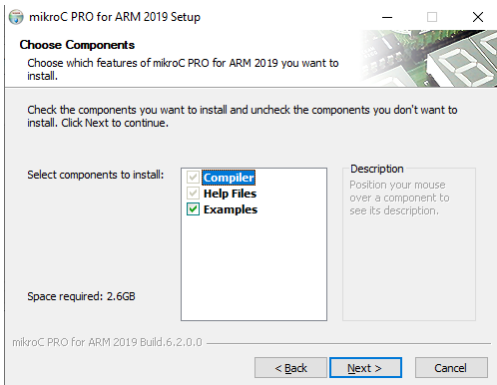
- 4- Resimdeki gibi “I accept” seçeneğini işaretleyerek “Next” butonuna tıklayınız.



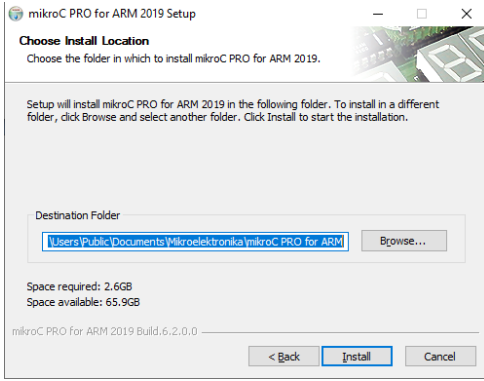
- 5- “Install for all users” seçeneğini işaretleyerek “Next” butonuna tıklayınız.



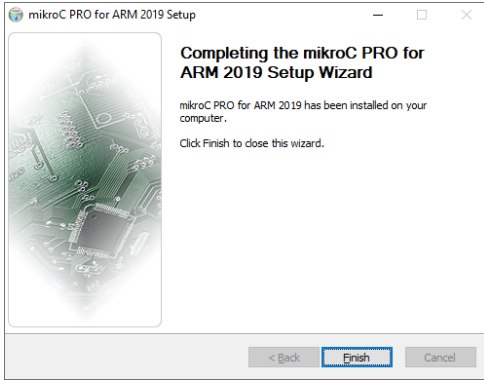
- 6- “Next” butonuna tıklayınız.



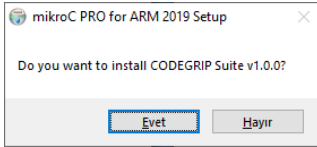
- 7- Aşağıdaki pencereden MikroC pro for ARM derleyicisinin kurulacağı dizini seçerek “Install” butonuna tıklayınız.



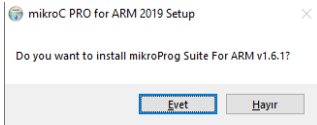
- 8- “Finish” butonuna tıklayınız.



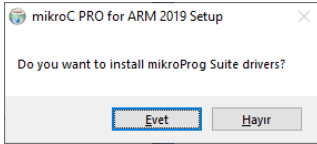
- 9- Eğer CodeGrip programlayıcı aygıtınız var ise “Evet” butonuna yok ise “Hayır” butonuna tıklayınız.



- 10- “mikroprog Suite for ARM” programını “Evet” butonuna tıklayarak yükleyiniz.



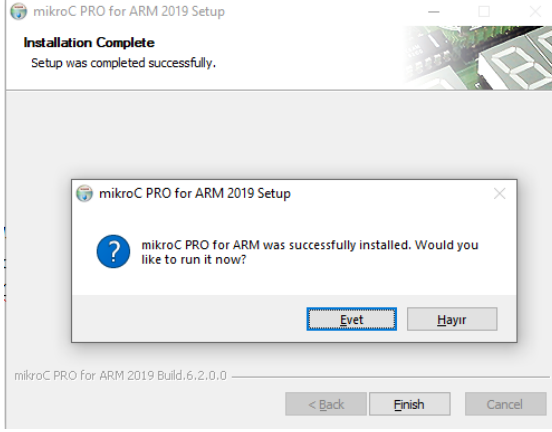
- 11- “Evet” butonuna tıklayarak ST-link driverlarını yükleyiniz.



- 12- Açılan pencerede ST klasörüne girerek “stlink_winusb_install.bat” dosyasını çalıştırınız.

Ad	Değiştirme tarihi	Tür	Boyut
amd64	20.07.2021 10:33	Dosya klasörü	
x86	20.07.2021 10:33	Dosya klasörü	
dpinst_amd64.exe	25.12.2018 20:29	Uygulama	665 KB
dpinst_x86.exe	25.12.2018 20:29	Uygulama	540 KB
stlink_dbg_winusb.inf	25.12.2018 20:29	Kur Bilgileri	4 KB
stlink_VCP.inf	25.12.2018 20:29	Kur Bilgileri	3 KB
stlink_winusb_install.bat	25.12.2018 20:29	Windows Toplu İş ...	1 KB
stlinkdbgwinusb_x64.cat	25.12.2018 20:29	Güvenlik Kataloğu	11 KB
stlinkdbgwinusb_x86.cat	25.12.2018 20:29	Güvenlik Kataloğu	11 KB
stlinkvcp_x64.cat	25.12.2018 20:29	Güvenlik Kataloğu	9 KB
stlinkvcp_x86.cat	25.12.2018 20:29	Güvenlik Kataloğu	9 KB

13- Açılan pencerede “Hayır” butonuna tıklayarak yükleme işlemini sonlandırınız.



14- [Buradan](#) indireceğiniz “*HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch*” isimli frekans ayarlamalarını içeren dosyayı mikroC derleyicisinin kurulu olduğu dizinde (“*Mikroelektronika\mikroC PRO for ARM\Schemes*”) bulunan “*Schemes*” isimli klasöre kopyalayınız.

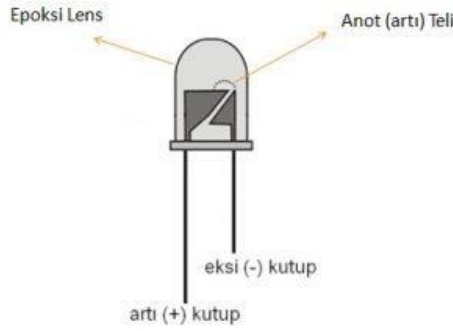
15- Uygulamalarda kullanılan MikroC proje dosyalarını [buradan](#) edinebilirsiniz.

Uygulama 1: Ledleri Yakıp Söndürmek.

Led (Light Emitting Diode) ışık yayan yarı iletken bir devre elemanıdır. Günümüzde çok farklı amaçlarla tasarlanmış onlarca çeşit ve büyüklükte led bulunmaktadır.

Ledlerin genelde 2 adet bacağı bulunur. Bu bacaklardan bir tanesi anot (+) ve katot (-) olarak isimlendirilir. Ledin ışık yayabilmesi için doğru kutuplandırılması ve üzerinden yeterli miktarda akım geçebilmesini sağlayacak şekilde gerilim uygulanması gerekir.

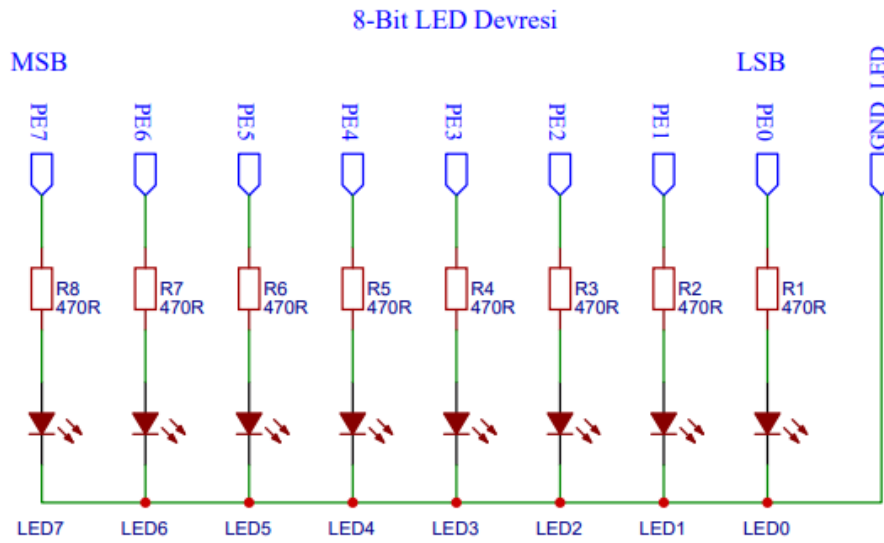
ARMapp-18 deney setinde kullanılan ledler 5mm çapında kırmızı ışık yayan ve ileri yön gerilimleri 1.8V olan ledlerdir.



Şekil 1.1: Led Diyot

Ledlerin çalıştırılmasında dikkat edilecek en önemli nokta akımlarının sınırlandırılmasıdır. Led üzerinden 30mA'dan fazla akım geçerse, kısa sürede bozulur. Ancak, ledler 10mA akım ile çalıştırılırlarsa yeterli ışık şiddetinde uzun süre sorunsuz çalışabilirler.

ARMapp-18 deney setinde bulunan ledlerin olduğu bölüme ait devre şeması şekil 1.2' de görülmektedir.



Şekil 1.2: ARMapp-18 Uygulama setinde bulunan Led modülü şeması.

Bu uygulamada ledler sağdan sola ve soldan sağa sırayla yakılıp söndürülecektir. Uygulama kodları aşağıda görülmektedir.

Uygulama Kodları:

```

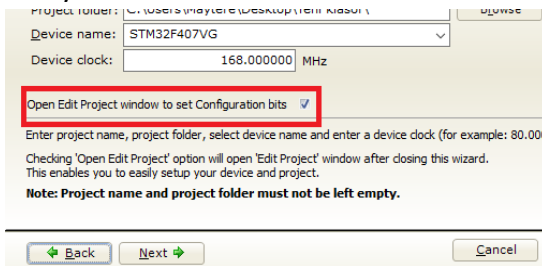
1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //   Karşımşek Efekti Yapan Program   //
4 //   MikroC v6.2 - STM32F407VG       //
5 //   ARMapp-18 DeneY Seti için yazılmıştır //
6 //*****//
7 //   http://elektrovadi.com           //
8 //   http://mikrodunya.wordpress.com  //
9 ////////////////////////////////////////////////////////////////////
10
11 void main()                                // Ana Program Bloğu
12 {
13     GPIO_Digital_Output(&GPIOE_BASE, _GPIO_PINMASK_0| // 'E' Portunun 0-7. pinleri dijital
14                             _GPIO_PINMASK_1|           // çıkış olarak ayarlanıyor.
15                             _GPIO_PINMASK_2|
16                             _GPIO_PINMASK_3|
17                             _GPIO_PINMASK_4|
18                             _GPIO_PINMASK_5|
19                             _GPIO_PINMASK_6|
20                             _GPIO_PINMASK_7);
21
22 while(1)                                    // Sonsuz döngü.
23 {
24     int i=0;                                // int tipinde 'i' isminde yerel değişken tanımlanıyor.
25     char leds=1;                            // char tipinde 'leds' isminde bir değişken tanımlanıyor.
26
27     for(i=0;i<7;i++)                        // 7 kez tekrarlanan for döngüsü bloğu.
28     {
29         GPIOE_ODR=leds;                    // 'E' portuna 'leds' değişkeni yükleniyor.( ilk anda 00000001)
30         leds<<=1;                          // 'leds' değişkeni 1 kez sola öteleniyor. (ilk evrede 00000010)
31         delay_ms(100);                     // 100 ms bekleniyor.
32     }
33     for(i=0;i<7;i++)                        // 7 kez tekrarlanan for döngüsü bloğu.
34     {
35         GPIOE_ODR=leds;                    // 'E' portuna 'leds' değişkeni yükleniyor.( ilk anda 10000000)
36         leds>>=1;                          // 'leds' değişkeni 1 kez sola öteleniyor. (ilk evrede 01000000)
37         delay_ms(100);                     // 100 ms bekleniyor.
38     }
39 }

```



Kod içerisinde “leds” isimli değişkenin içeriği iki adet for döngüsü içerisinde değiştirilerek GPIOE_ODR kaydedicisine aktarılmış ve ledlerin kara şimşek efekti ile yanıp sönmesi sağlanmıştır.

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.



- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.

- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 9- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 10- Pişano anahtarlardan sol taraftakinde bulunan LED anahtarını açarak, ledlerin GND bağlantısını sağlayınız.
- 11-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 12- Programın çalışmasını gözlemleyiniz.

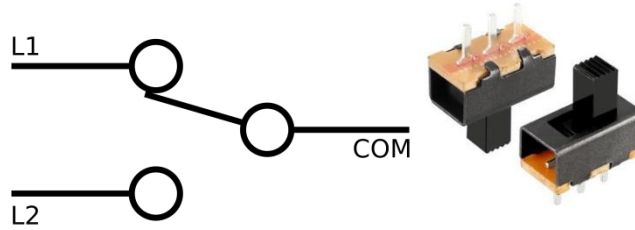
Sorular:

- 1- Ledlerde binary olarak 0-255 arası sayan programı yazınız.
- 2- Ledleri aynı anda iki led yanacak şekilde kara şimşek efekti yapan programı yazınız.

Uygulama 2: Anahtarlar ve Ledler.

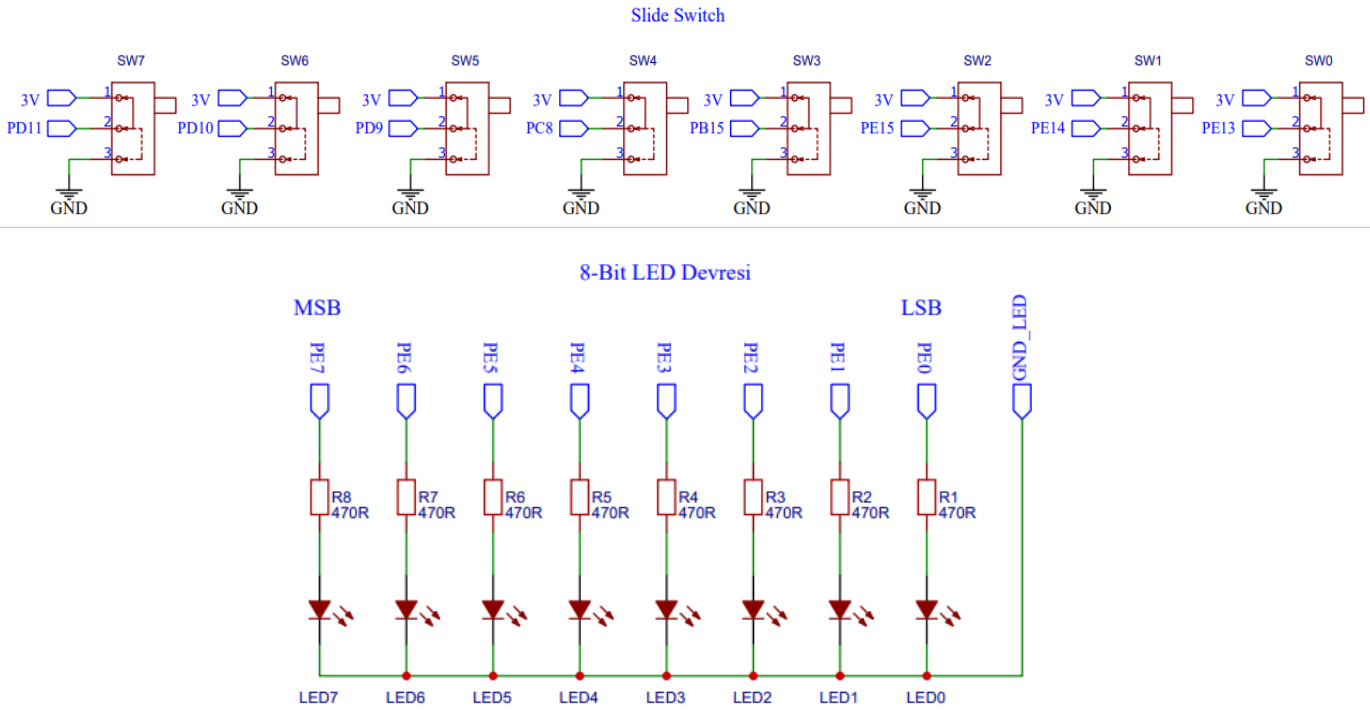
Anahtarlar elektronik devrelerde en sık kullanılan devre elemanlarıdır. Anahtarlar pek çok türde üretilmektedirler. ARMapp-18 uygulama setinde kullanılan anahtar tek kutup iki atımlı bir anahtardır. Bu tip anahtarlar 3 pinli olarak imal edilirler ve "SPDT" olarak adlandırılırlar.

Bu tip anahtarların ortada bulunan pinleri ortak uç, diğer ikisi ise birbirlerinden bağımsız uçlardır. Switch bir konuma alındığında Com ve L1 pinleri, diğer konuma alındığında ise Com ve L2 pinleri kısa devre olmaktadır.



Şekil 2.1: 3 pinli anahtar sembolü ve görüntüsü.

Deney setinde bu şekilde bir anahtar tercih edilmesinin sebebi bir konumda orta uca 3V, diğer konumda orta uca 0V uygulama ihtiyacını karşılayabilmesidir. Şekil 2.2'de bu uygulamada kullanılacak olan ledler ve anahtarlara ait bölümlerin ARMapp-18 deney setindeki devre şemaları görülebilir.



Şekil 2.2: ARMapp-18 Uygulama Setinde Bulunan Anahtar ve Led Modüllerine Ait Şema.

Bu uygulamada anahtarların durumlarına göre kendilerine gelen ledler yakılacaktır. Örneğin SW7 orta ucundan 3V çıkacak konuma alındığında LED7 yanacak, orta ucundan 0V çıkacak konuma alındığında başka bir deyişle orta uç şaseye çekildiğinde LED7 sönecektir.

Uygulama Kodları:

```

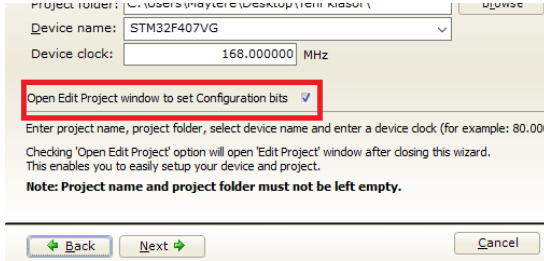
1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // Anahtarların ve Ledlerin kullanımı. //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11 void main() // Ana Program Bloğu
12 {
13     GPIO_Digital_Output(&GPIOE_BASE, _GPIO_PINMASK_0| // 'E' Portunun 0-7. pinleri dijital
14     _GPIO_PINMASK_1| // çıkış olarak ayarlanıyor.
15     _GPIO_PINMASK_2|
16     _GPIO_PINMASK_3|
17     _GPIO_PINMASK_4|
18     _GPIO_PINMASK_5|
19     _GPIO_PINMASK_6|
20     _GPIO_PINMASK_7);
21
22     GPIO_Digital_Input(&GPIOE_BASE, _GPIO_PINMASK_13| // 'E' Portunun 13,14 ve 15. pinleri dijital
23     _GPIO_PINMASK_14| // giriş olarak ayarlanıyor.
24     _GPIO_PINMASK_15);
25
26     GPIO_Digital_Input(&GPIOB_BASE, _GPIO_PINMASK_15); // 'B' Portunun 15. pini dijital
27     // giriş olarak ayarlanıyor.
28
29     GPIO_Digital_Input(&GPIOC_BASE, _GPIO_PINMASK_8 ); // 'C' Portunun 8. pini dijital
30     // giriş olarak ayarlanıyor.
31
32     GPIO_Digital_Input(&GIOD_BASE, _GPIO_PINMASK_9 | // 'D' Portunun 9,10 ve 11. pinleri dijital
33     _GPIO_PINMASK_10| // giriş olarak ayarlanıyor.
34     _GPIO_PINMASK_11);
35
36     while(1) // Sonsuz döngü
37     {
38         char switches; // char tipinde 'switches' isminde bir değişken oluşturuluyor.
39         switches.b0=GPIOE_IDR.B13; // 'E' portunun 13. pininin değeri okunarak
40         // 'switches' değişkenininin 0. bitine aktarılıyor.
41         switches.b1=GPIOE_IDR.B14; // 'E' portunun 14. pininin değeri okunarak
42         // 'switches' değişkenininin 1. bitine aktarılıyor.
43         switches.b2=GPIOE_IDR.B15; // 'E' portunun 15. pininin değeri okunarak
44         // 'switches' değişkenininin 2. bitine aktarılıyor.
45         switches.b3=GPIOB_IDR.B15; // 'B' portunun 15. pininin değeri okunarak
46         // 'switches' değişkenininin 3. bitine aktarılıyor.
47         switches.b4=GPIOC_IDR.B8 ; // 'C' portunun 18. pininin değeri okunarak
48         // 'switches' değişkenininin 4. bitine aktarılıyor.
49         switches.b5=GIOD_IDR.B9 ; // 'D' portunun 9. pininin değeri okunarak
50         // 'switches' değişkenininin 5. bitine aktarılıyor.
51         switches.b6=GIOD_IDR.B10; // 'D' portunun 10. pininin değeri okunarak
52         // 'switches' değişkenininin 6. bitine aktarılıyor.
53         switches.b7=GIOD_IDR.B11; // 'D' portunun 11. pininin değeri okunarak
54         // 'switches' değişkenininin 7. bitine aktarılıyor.
55         GPIOE_ODR=switches; // 'E' portuna switches değişkeni aktarılarak anahtarların
56         // durumu ledlerde gösteriliyor.
57     }
58 }
59
60



```

Kod kısmında switches isimli 8 bitlik bir değişken tanımlanmış ve bu değişkenin bitlerine anahtarların aldığı değerler okunarak aktarılmıştır. Daha sonra E portuna switches değişkeni olduğu gibi aktarılmış ve anahtarların konumuna göre ledlerin yanması sağlanmıştır.

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.



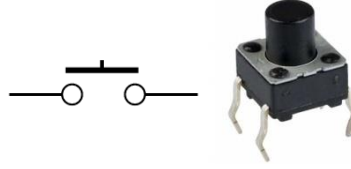
- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 9- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 10- Pişano anahtarlardan sol taraftakinde bulunan LED anahtarını açarak, ledlerin GND bağlantısını sağlayınız.
- 11-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 12- Programın çalışmasını anahtarların konumlarını değiştirerek gözlemleyiniz.

Sorular:

- 1- Anahtarlar şu anda 1 konumuna alındığında ledler yanmaktadır. Ledlerin anahtarlar 0 konumuna alındığında yanmasını sağlayınız.
 - 2- SW7 - LED0
SW6 - LED1
SW5 - LED2
SW4 - LED3
SW3 - LED4
SW2 - LED5
SW1 - LED6
SW0 - LED7
- eşleşmesi ile anahtarların eşleştikleri ledlerin durumlarını kontrol etmelerini sağlayınız.

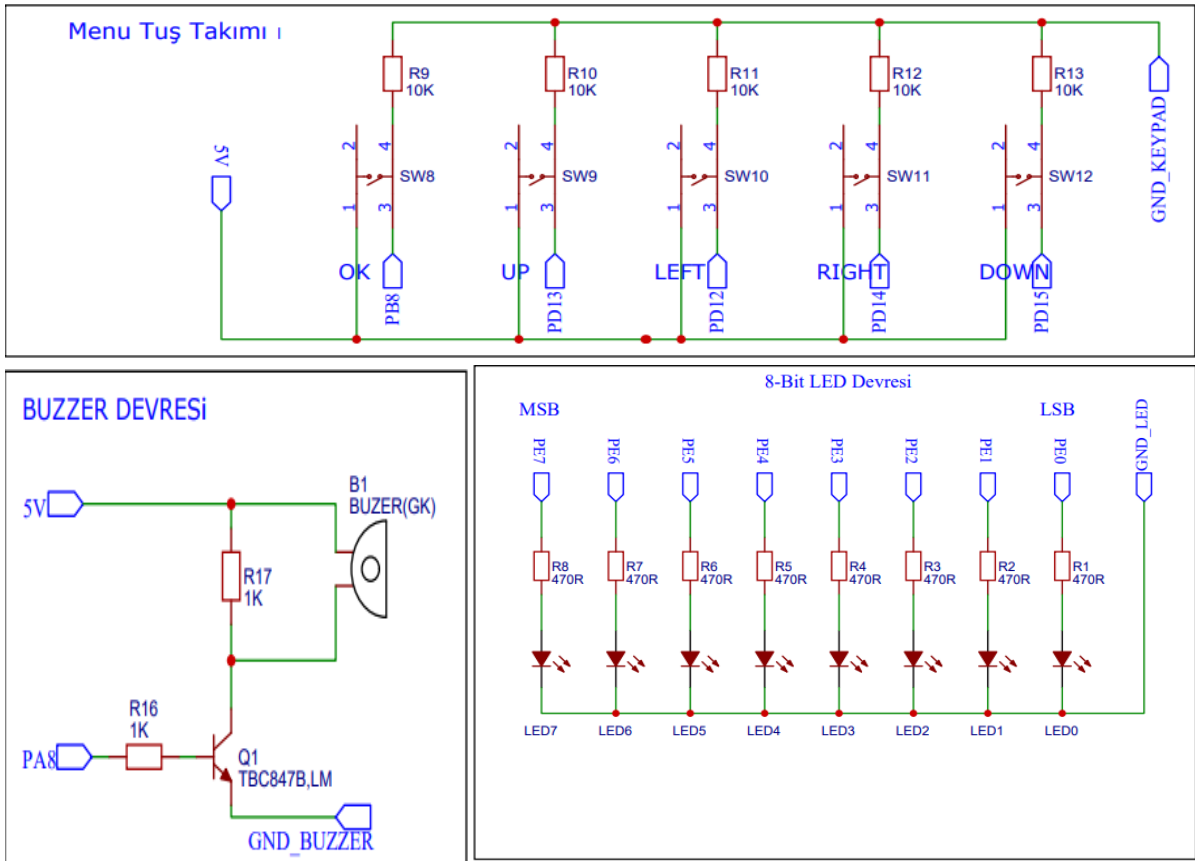
Uygulama 3: Menü Tuş Takımı.

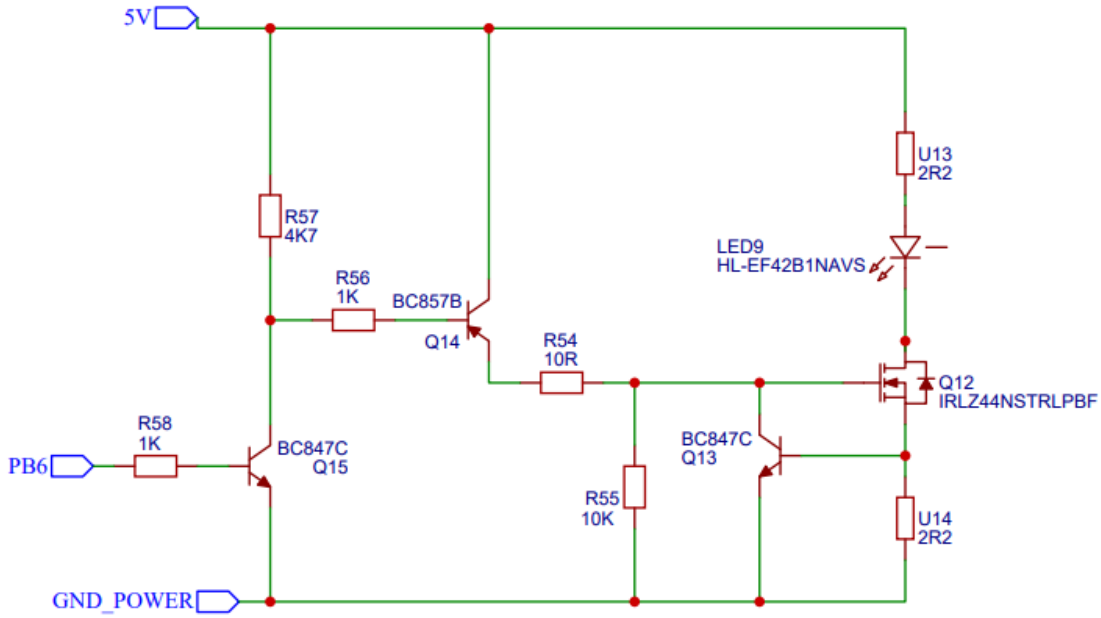
ARMapp-18 ARM Uygulamaları Seti üzerindeki menü tuş takımı, yukarı, aşağı, sol, sağ ve onay tuşu olmak üzere 5 adet push butondan oluşmaktadır. ARMapp-18 deney seti üzerindeki butonlar push tipte 6x6mm boyutlarındadırlar. Bu butonlara basıldığı müddetçe uçları arasında iletim sağlanırken, üzerlerindeki basınç kalktığı anda iletim kesilir.



Şekil 3.1: 6x6 mm Push Buton Sembolü ve Görüntüsü.

Push butonlara basılması ve bırakılması esnasında iletimin sağlanması ve kesilmesi esnasında elektriksel arklar meydana gelir. Butonlar eğer mikrodenetleyiciler ile kullanılıyorlar ise bu anlık arklar hatalı okumalara sebep olabilir.





Şekil 3.2: ARMapp-18 Uygulama Setindeki Buzzer, Power Led, Buton Ve Led Modüllerine Ait Şemalar.

Bu arkların yazılımsal problemler oluşmasını engelleyebilmek amacıyla butona basılma anı algılandığında bir miktar yazılımsal bekleme yapılarak ark oluşumunun bitmesi beklenebilir. Buton arkını engellemenin donanımsal yöntemleri de mevcuttur. En basit donanımsal önlem butona paralel 100nf değerinde bir kondansatör bağlamaktır.

Bu uygulamada sol menü tuşuna basıldığında ledler sola doğru ilerleyerek yanmakta, sağ menü tuşuna basıldığında ledler sağ tarafa doğru sönerek gerilemekte, yukarı menü tuşuna basıldığında power led yanmakta, aşağı menü tuşuna basıldığında power led sönmekte ve ok tuşuna bir kez basıldığında buzzer ötmekte, ikinci kez basıldığında susmaktadır.

Uygulama Kodları:

```

1 ///////////////////////////////////////////////////////////////////
2 //*****//
3 //      Menü buton setinin ve çeşitli //
4 //      elemanların kullanımı. //
5 //      MikroC v6.2 - STM32F407VG //
6 //      ARMapp-18 Deneysel Seti için yazılmıştır //
7 //*****//
8 //      http://elektrovadi.com //
9 //      http://mikrodunya.wordpress.com //
10 ///////////////////////////////////////////////////////////////////
11
12 // Menü butonları tanımlanıyor.
13 #define yukari      GPIOC_IDR.B4 // 'yukarı' butonu PC4 pininin değeri okuma registerına tanımlanıyor.
14 #define asagi      GPIOB_IDR.B14 // 'aşağı' butonu PB14 pininin değeri okuma registerına tanımlanıyor.
15 #define sol        GPIOB_IDR.B12 // 'sol' butonu PB12 pininin değeri okuma registerına tanımlanıyor.
16 #define sag        GPIOB_IDR.B13 // 'sağ' butonu PB13 pininin değeri okuma registerına tanımlanıyor.
17 #define orta       GPIOB_IDR.B8 // 'orta' butonu PB8 pininin değeri okuma registerına tanımlanıyor.
18
19 #define buzzer      GPIOA_ODR.B8 // 'buzzer' pini PA8 pininin çıkış registerına tanımlanıyor.
20 #define powerLed    GPIOB_ODR.B6 // 'powerLed' pini PB6 pininin çıkış registerına tanımlanıyor.
21
22 char ledler=0,buzzerSayac=0; // char tipinde 'ledler' ve 'buzzerSayac' isiminde iki değişken tanımlanıyor.
23
24 void main() // Ana program bloğu.
25 {
26     GPIO_Digital_Input(&GPIOB_BASE,_GPIO_PINMASK_8 | // 'B' Portunun 8,12,13 ve 14. pinleri dijital
27                       _GPIO_PINMASK_12| // giriş olarak ayarlanıyor. (Menü butonları)
28                       _GPIO_PINMASK_13|
29                       _GPIO_PINMASK_14);
30
31     GPIO_Digital_Input(&GPIOC_BASE,_GPIO_PINMASK_4 ); // 'C' Portunun 4. pini dijital giriş olarak ayarlanıyor. (Menü butonu)
32
33     GPIO_Digital_Output(&GPIOA_BASE,_GPIO_PINMASK_8); // 'A' Portunun 8. pini dijital çıkış olarak ayarlanıyor. (Buzzer)
34
35     GPIO_Digital_Output(&GPIOB_BASE,_GPIO_PINMASK_6); // 'B' Portunun 6. pini dijital çıkış olarak ayarlanıyor. (Power Led)
36
37     GPIO_Digital_Output(&GPIOE_BASE,_GPIO_PINMASK_0| // 'E' Portunun 0-7. pinleri dijital çıkış olarak ayarlanıyor.
38                       _GPIO_PINMASK_1|
39                       _GPIO_PINMASK_2|
40                       _GPIO_PINMASK_3|
41                       _GPIO_PINMASK_4|
42                       _GPIO_PINMASK_5|
43                       _GPIO_PINMASK_6|
44                       _GPIO_PINMASK_7);
45
46     buzzer=0; // Buzzer susturuluyor.
47     powerLed=0; // Power Led söndürülüyor.
48     GPIOE_ODR=ledler; // 'ledler' değişkeni 0 olduğu için hepsi söndürülüyor.
49
50     // Sonsuz Döngü
51     while(1)
52     {
53         if(yukari) // Yukarı butonuna basıldıysa ( GPIOC_IDR.B4=1 ise)
54         {
55             powerLed=1; // Power Led'i yak. (GPIOB_ODR.B6=1 yap)
56             delay_ms(100); // Buton arkını önlemek için 100 ms bekle.
57             while(yukari); // Yukarı tuşuna basılı olduğu müddetçe burada bekle.
58             delay_ms(100); // Buton arkını önlemek için 100 ms bekle.
59         }
60
61         if(asagi) // Aşağı butonuna basıldıysa ( GPIOB_IDR.B14=1 ise)
62         {
63             powerLed=0; // Power Ledi söndür. (GPIOB_ODR.B6=0 yap)
64             delay_ms(100); // Buton arkını önlemek için 100 ms bekle.
65             while(asagi); // Aşağı tuşuna basılı olduğu müddetçe burada bekle.
66             delay_ms(100); // Buton arkını önlemek için 100 ms bekle.
67         }
68
69         if(sol) // Sağ butonuna basıldıysa ( GPIOB_IDR.B13=1 ise)
70         {
71             ledler=(ledler<<1)+1; // ledler değişkenini 1 kez sola kaydır ve 1 ekle.
72             GPIOE_ODR=ledler; // E portuna ledler değişkenini yükleyerek istenilen ledlerin yanmasını sağla.
73             delay_ms(100); // Buton arkını önlemek için 100 ms bekle.
74             while(sol); // Sağ tuşuna basılı olduğu müddetçe burada bekle.
75             delay_ms(100); // Buton arkını önlemek için 100 ms bekle.
76         }
77     }
78 }

```

```

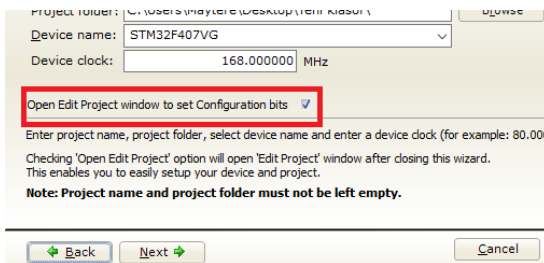
75
76     if(sag)
77     {
78         if(ledler>0)ledler=(ledler-1)>>1;
79         GPIOE_ODR=ledler;
80         delay_ms(100);
81         while(sag);
82         delay_ms(100);
83     }
84
85     if(orta)
86     {
87         if(buzzerSayac==0)
88         {
89             buzzer=1;
90             buzzerSayac++;
91         }
92         else if(buzzerSayac==1)
93         {
94             buzzer=0;
95             buzzerSayac=0;
96         }
97         delay_ms(100);
98         while(orta);
99         delay_ms(100);
100    }
101 }
102 }



```

Kod kısmında dikkat ederseniz butona basıldığında ve bırakıldığında 100 ms'lik beklemler yapılmaktadır. Bu beklemler ark oluşumunu engellemektedir. Diğer önemli bir nokta ise onay butonuna basıldığında buzzerSayac isminde bir değişken arttırılmakta ve bu değişkenin durumuna göre buzzer ötürülüp susturulmaktadır.

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.



- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 9- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 10- Pişano anahtarlardan sol taraftakinde bulunan LED, BUZZER, TUŞ TAKIMI, POWER LED anahtarlarını açarak, bu modüllerin GND bağlantılarını sağlayınız.
- 11-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.

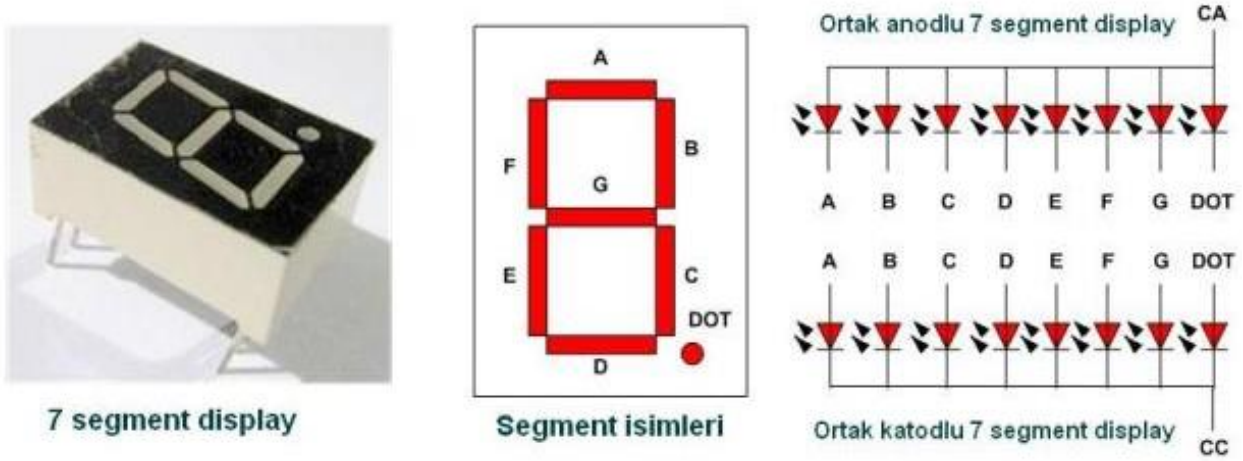
12- Programın çalışmasını menü butonlarına basarak gözlemleyiniz. (**Dikkat: Power led'e doğrudan uzun süre bakmak gözlerde kalıcı hasara sebep olabilir.**)

Sorular:

- 1- "Sol" butonuna basıldıkça ledleri sağdan sola sırayla söndüren, "sağ" butonuna basıldıkça soldan sağa sırasıyla yakan programı yazınız.
- 2- "Onay" butonuna bastıkça ledlerde 0-255 arası binary sayma işlemini gerçekleştiriniz.

Uygulama 4: 7 Segment Display Uygulaması.

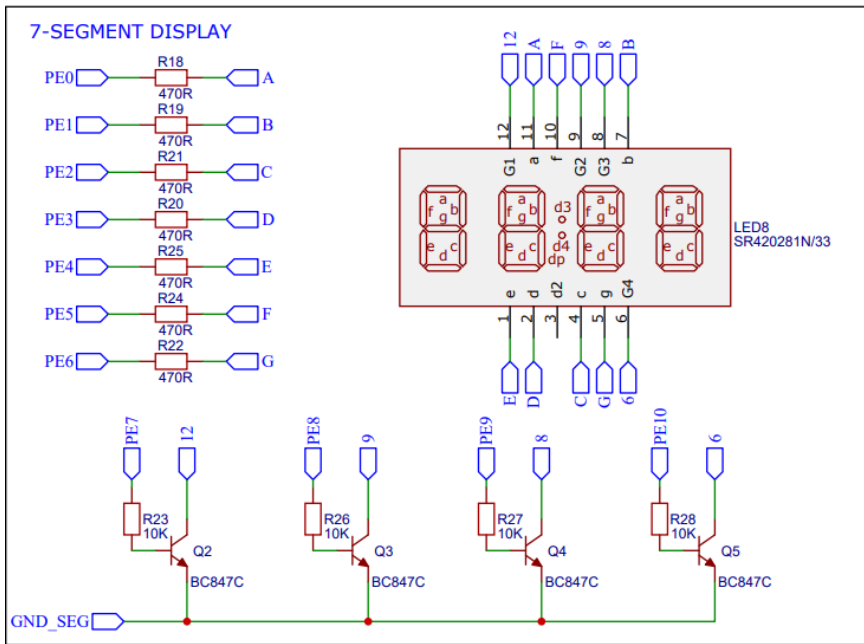
7 Segment Display, asansörlerden, beyaz eşyalara, uydu alıcılarından akaryakıt pompalarına pek çok cihazda gösterge amaçlı olarak kullanılmaktadır. 7 segment displaylerin isimleri 7 parçalı yapılarından kaynaklanmaktadır. Bu parçalar kullanılarak bütün rakamlar ve bazı harfler oluşturulabilmektedir.



Şekil 4.1: 7 Segment Display ve İç Yapısı.

7 segment displayler farklı boyutlarda ve farklı güçlerde üretilmekle birlikte iç yapıları da ortak anod ve ortak katot olmak üzere iki türdür. Küçük boyutlu ve küçük güçle çalışan 7 segment displayler mikrodenetleyiciler ile kullanılacaksa kullanım kolaylığı açısından ortak katotlusunu tercih etmek daha makuldür.

ARMapp-18 uygulama setinde kullanılan 7 segment display 4 haneli ve ortak katotlu bir 7 segment displaydir. 7 segment display modülün devre şeması şekil 4.2' de görülebilir.



Şekil 4.2: ARMapp-18 Uygulama Setindeki 7 Segment Display Modüllerine Ait Şemalar.

ARMapp-18 uygulama setinde kullanılan 7 segment display modülünde her hane birer NPN transistör vasıtası ile kontrol edilmektedir. Hangi hane çalıştırılmak isteniyor ise o haneye ait olan transistör iletime geçirilerek istenilen rakam o hanede gösterilir. Örneğin 1234 sayısını displayde göstermek istersek önce 1 rakamını gösterecek değerleri displayin ABCDEFG pinlerine gönderilir. Daha sonra Q7 transistörünün beyzine gerilim uygulanarak iletime geçirilir (bu esnada diğer transistörler yalıtımda olmalıdır yani beyaz uçlarına 0 gelmelidir). Bu halde 4' lü displayin 1. hanesinde '1' rakamını görürüz. Bu şekilde 5 ms civarında bir bekleme yapılarak 2. Hanede '2' rakamı gösterilir ve yine 5 ms beklenir. Bu işlemler diğer haneler için de gerçekleştirilir ve tekrar başa dönülerek 1. Hanede yine '1' rakamı gösterilir. Bu işlem 5 ms aralıklarla çok hızlı yapıldığı için bir göz yanılması oluşarak bütün rakamlar aynı anda görünür. Bu tekniğe tarama tekniği ismi verilir. Bu sayede bütün displaylerin segment bağlantıları için ayrı ayrı bağlantılar kullanılmaz ve pin sayısından tasarruf edilir.

Uygulama Kodları:

```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // 4 Haneli 7 segment Display kullanımı. //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11
12
13 unsigned int sayilar[10]={0x3F,0x06,0x5B,0x4F,0x66, // 7Segment displayde gösterilecek rakamların kodları 'sayilar'
14 // 0x6D,0x7D,0x07,0x7F,0x6F}; // dizisi olarak tanımlanıyor.
15 unsigned int i=0,j=0; // i ve j isminde iki adet unsigned int tipinde değişken tanımlanıyor
16
17 void display(unsigned int i) // 'display' isminde tanımlanmış ve işaretli tamsayı türünde
18 { // değer alan fonksiyon tanımlanıyor.
19 char binler,yuzler,onlar,birler; // char tipinde basamaklara ait sayı değerleri tanımlanıyor.
20 binler=i/1000; // binler basamağının sayı değeri hesaplanıyor.
21 yuzler=(i/100)%10; // yuzler basamağının sayı değeri hesaplanıyor.
22 onlar=(i/10)%10; // onlar basamağının sayı değeri hesaplanıyor.
23 birler=i%10; // birler basamağının sayı değeri hesaplanıyor.
24
25 GPIOE_ODR=0x0080 | sayilar[binler]; // Binler basamağının sayı değerine karşılık gelen 7 segment kodu
26 delay_ms(2); // diziden çekilerek displayin en soldaki hanesinde gösteriliyor ve
27 // 2 ms beklenecek sayının displayde görülebilmesi sağlanıyor.
28
29 GPIOE_ODR=0x0100 | sayilar[yuzler]; // Yuzler basamağının sayı değerine karşılık gelen 7 segment kodu
30 delay_ms(2); // diziden çekilerek displayin en soldaki hanesinde gösteriliyor ve
31 // 2 ms beklenecek sayının displayde görülebilmesi sağlanıyor.
32
33 GPIOE_ODR=0x0200 | sayilar[onlar]; // Onlar basamağının sayı değerine karşılık gelen 7 segment kodu
34 delay_ms(2); // diziden çekilerek displayin en soldaki hanesinde gösteriliyor ve
35 // 2 ms beklenecek sayının displayde görülebilmesi sağlanıyor.
36
37 GPIOE_ODR=0x0400 | sayilar[birler]; // Birler basamağının sayı değerine karşılık gelen 7 segment kodu
38 delay_ms(2); // diziden çekilerek displayin en soldaki hanesinde gösteriliyor ve
39 // 2 ms beklenecek sayının displayde görülebilmesi sağlanıyor.
40 }
41
42 void main() // Ana program bloğu
43 {
44 GPIO_Digital_Output(&GPIOE_BASE, _GPIO_PINMASK_0| // 'E' portunun 0,1,2,3,4,5,6,7,8,9 ve 10. pinleri
45 // _GPIO_PINMASK_1| // dijital çıkış olarak tanımlanıyor.
46 // _GPIO_PINMASK_2|
47 // _GPIO_PINMASK_3|
48 // _GPIO_PINMASK_4|
49 // _GPIO_PINMASK_5|
50 // _GPIO_PINMASK_6|
51 // _GPIO_PINMASK_7|
52 // _GPIO_PINMASK_8|
53 // _GPIO_PINMASK_9|
54 // _GPIO_PINMASK_10);
55
56 while(1) // Sonsuz döngü.
57 {
58 for(i=0;i<10000;i++) // 0'dan başlayıp 9999 olana kadar sayıları displayde gösteren kod.
59 {
60 for(j=0;j<5;j++)display(i); // aynı sayı 5 kez gösterilerek sayı artışında gecikme sağlanıyor.
61 }
62 }
63 }

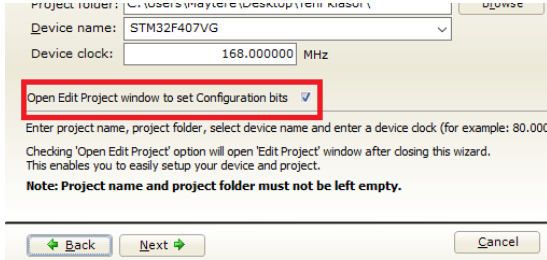
```



ARMapp-18

Kod kısmında dikkat edilecek husus sonsuz döngü içerisinde kullanılan iç içe iki for döngüsünün kullanımınıdır. Bu döngülerden dışarıda olanı 0-9999 arası sayma işlemini yaparken içteki for döngüsü tarama işleminin yapıldığı “display” fonksiyonunu çağırarak for döngüsüdür. İçteki for döngüsünde dikkat edilecek olursa “display” fonksiyonu 5 kez çağırılmakta ve aynı sayı 5 kez görüntülenmektedir. Eğer bu sayı arttırılırsa aynı sayı daha uzun süre kalınacağından sayma hızı düşecektir.

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.



- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 9- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 10- Pişano anahtarlardan sağ taraftakinde bulunan 7 SEGMENT anahtarını açarak, bu modülün GND bağlantısını sağlayınız.
- 11-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 12- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- Bu uygulamada "delay_ms" gibi bir gecikme fonksiyonunun neden kullanılmadığını açıklayınız.
- 2- Menü tuş takımı modülünü kullanarak yukarı butonuna tıkladığında yukarıya doğru sayan, aşağı butonuna basıldığında aşağı doğru sayan programı yazınız.

Uygulama 5: 2x16 Karakter LCD Uygulaması.

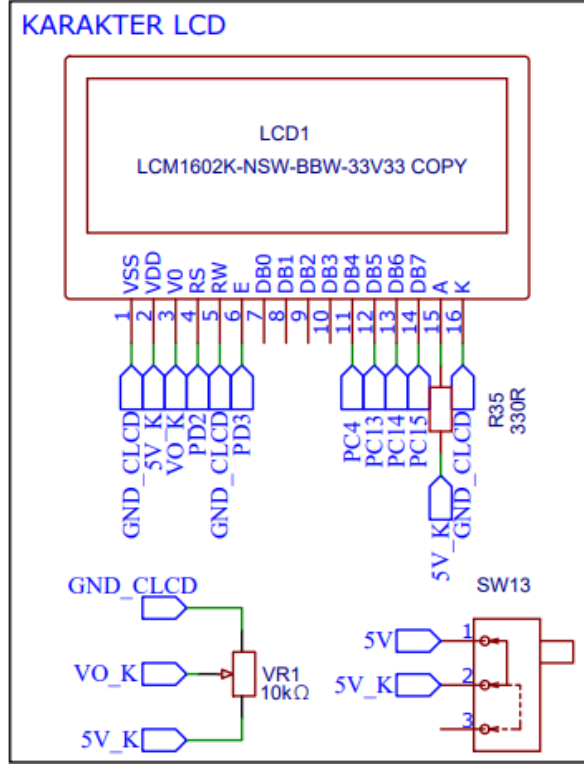
Karakter LCD'ler yazı, rakamlar ve karakterlerin bir arada gösterilmesi gereken yerlerde tercih edilmektedirler. Bu tip karakter LCD'ler 2x16, 4x20, 1x8, 2x8 gibi farklı boyutlarda üretilmektedirler. Örneğin 2x16 tipinde bir karakter LCD' de 2 satır 16 sütun bulunur ve aynı anda toplam 32 adet karakter gösterebilirler.

Karakter LCD'ler üzerinde kendilerine ait bir kontrolcü çip bulunmaktadır ve bu çip harici mikrodenetleyicilerle haberleşerek ekranda istenilen karakterlerin gösterilmesine olanak sağlamaktadır. Bu tip LCD'lerde en sık kullanılan kontrolcü çipi HD44780 isimli alfanümerik LCD kontrolcüsüdür. Şekil 5.1'de 2x16 boyutlarında bir karakter LCD gösterilmektedir.



Şekil 5.1: 2x16 Karakter LCD.

MikroC derleyicisinde HD44780 çipi barındıran karakter LCD'ler için dâhili kütüphane bulunmaktadır. Bu kütüphane kullanılarak karmaşık kodlamaya gerek kalmadan kolayca LCD kullanımı mümkün olmaktadır. Kütüphane fonksiyonları basit ve anlaşılır bir şekilde tasarlanmıştır. Bu kütüphane kullanılarak farklı boyutlarda karakter LCD'leri kullanmak mümkündür. Kütüphane fonksiyonlarıyla ilgili daha fazla bilgi almak için MikroC pro for ARM derleyicisinin yardım dosyalarına göz atılabilir.



Şekil 5.2: ARMapp-18 Uygulama Setindeki Karakter LCD Modülüne Ait Devre Şeması.

ARMapp-18 uygulama setindeki karakter LCD modülü için D portunun düşük 8 biti kullanılmıştır. Ayrıca karakter LCD'nin kontrastını ayarlamak için bir adet potansiyometre de devre üzerine iliştirilmiştir.

Bu uygulamada karakter LCD üzerinde bir mesaj yazılmış ve bu mesaj sağa ve sola kaydırılarak bir gösterim yapılmıştır. Daha sonra bir değişkenin değeri sürekli değiştirilerek ekranda gösterilmiştir.

Uygulama Kodları:

```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //      2x16 Karakter LCD Kullanımı      //
4 //      MikroC v6.2 - STM32F407VG      //
5 //      ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 //      http://elektrovadi.com      //
8 //      http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11
12 sbit LCD_RS at GPIO_ODR.B2;           // RS pini PD2 pinine bağlı.
13 sbit LCD_EN at GPIO_ODR.B3;           // EN pini PD3 pinine bağlı.
14 sbit LCD_D4 at GPIO_ODR.B4;           // D4 pini PD4 pinine bağlı.
15 sbit LCD_D5 at GPIO_ODR.B5;           // D5 pini PD5 pinine bağlı.
16 sbit LCD_D6 at GPIO_ODR.B6;           // D6 pini PD6 pinine bağlı.
17 sbit LCD_D7 at GPIO_ODR.B7;           // D7 pini PD7 pinine bağlı.
18
19 char dizi[16]="ARMapp-18 STM32";       // 16 karakterlik dizi içerisinde bir text tanımlanıyor.
20 int i=0;                               // int tipinde 'i' isminde değişken tanımlanıyor.
21 char txt[7];                           // int veri tipinden yazıya çevrim işleminde kullanılacak tampon dizi.
22
23 void main()                             // Ana program bloğu
24 {
25     Lcd_Init();                          // LCD kurulumu yapılıyor.
26     delay_ms(200);                       // LCD kurulumunun hatasız sonuçlanması için 200 ms bekleniyor.
27     Lcd_Cmd(_LCD_CLEAR);                 // LCD ekranı temizleniyor.
28     Lcd_Cmd(_LCD_CURSOR_OFF);           // LCD imleci kapatılıyor.
29     Lcd_Out(1,1,"MikroC");               // LCD'nin 1. satır 1. sütunundan itibaren 'MikroC' yazısı yazdırılıyor.
30
31     for(i=0;i<15;i++)                    // LCD'nin 2. satırına dizi'nin bütün karakterleri teker teker yazılıyor.
32     {
33         Lcd_Chr(2,(i+1),dizi[i]);
34     }
35     delay_ms(3000);                      // 3 saniye bekleniyor.
36
37     for(i=0;i<3;i++)                     // LCD ekrandaki tüm görüntü 3 kez sola kaldırılıyor.
38     {
39         Lcd_Cmd(_LCD_SHIFT_LEFT);
40         delay_ms(1000);
41     }
42     for(i=0;i<6;i++)                     // LCD ekrandaki tüm görüntü 6 kez sağa kaldırılıyor.
43     {
44         Lcd_Cmd(_LCD_SHIFT_RIGHT);
45         delay_ms(1000);
46     }
47     for(i=0;i<3;i++)                     // LCD ekrandaki tüm görüntü 3 kez sola kaldırılıyor.
48     {
49         Lcd_Cmd(_LCD_SHIFT_LEFT);
50         delay_ms(1000);
51     }
52     delay_ms(3000);                      // 3 kez bekleniyor.
53
54     Lcd_Cmd(_LCD_CLEAR);                 // LCD temizleniyor.
55     Lcd_Out(1,1,"Sayi=");               // 1. satırın 1. sütununa 'Sayi=' yazdırılıyor.
56     for(i=-9999;i<10000;i++)            // -9999'dan 10000'e kadar olan sayılar LCD'de yazdırılıyor.
57     {
58         inttostr(i,txt);                 // int tipindeki i değişkeni yazıya dönüştürülerek txt isimli diziye aktarılıyor.
59         Lcd_Out(1,6,txt);                 // LCD'nin 1. satır 6. sütununa txt dizisinin içeriği yazdırılıyor.
60         delay_ms(50);                    // 50 ms bekleniyor.
61     }
62 }

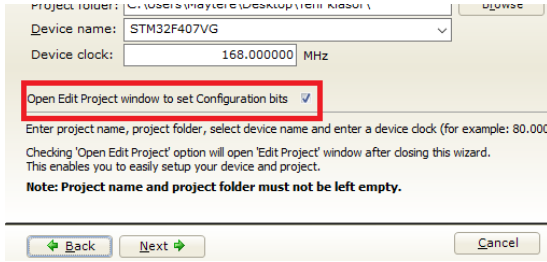
```

Sayısal değerlerin LCD gibi ekranlarda gösterilebilmesi için öncelikle yazıya yani karakter dizisine çevrilmesi gerekmektedir. MikroC derleyicisinde bu işlem için de bir kütüphane mevcuttur. Bu kütüphaneye "Conversions" (dönüşümler) ismi verilmiştir. Bu uygulamaya ait örnek kod kullanılırken de hem LCD kütüphanesi hem de Conversions kütüphanesi kullanılmıştır.

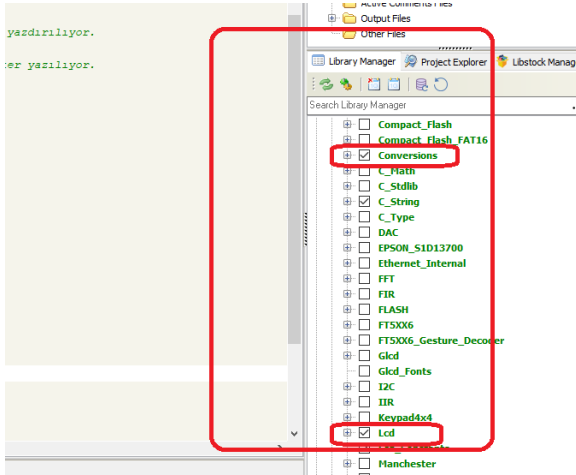
İşlem Basamakları:



- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.

- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.



- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “Conversions” vs “LCD” kütüphanelerini seçiniz.



- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMap-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Pişano anahtarlardan sol taraftakinde bulunan K.LCD anahtarını ve sağ taraftakinde bulunan GLCD anahtarını açarak, bu modüllerin GND bağlantısını sağlayınız. Karakter LCD modülünün bitişiğinde bulunan anahtarı “ON” konumuna, Grafik LCD modülünün bitişiğinde bulunan anahtarı OFF konumuna getiriniz.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- 1. Satıra adınızı ve soyadınızı, 2. Satıra okul numaranızı yazan programı yazınız.
- 2- Menü tuş takımı modülünü kullanarak yukarı butonuna tıkladıkça LCD’ de yukarıya doğru sayan, aşağı butonuna bastıkça aşağı doğru sayan programı yazınız.

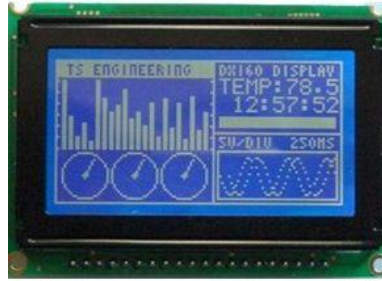
Uygulama 6: Grafik LCD Uygulaması.

Grafik LCD'ler harfler, rakamlar ve karakterler göstermenin yanı sıra tek renkli resimler göstermek amacıyla kullanılırlar. İstenilen resimler, geometrik şekiller ve grafikler belirli kurallara uygun şekilde biçimlendirilerek bu tip ekranlarda istenilen amaca yönelik biçimde gösterilebilirler.

Grafik LCD ekranlar 128x64, 240x128, 240x64 gibi farklı çözünürlüklerde üretilmektedirler. ARMapp-18 uygulama setinde, piyasada en çok kullanılan grafik LCD kontrolcülerinden olan KS0108 kontrolcü çipe sahip 128x64 çözünürlüğünde bir grafik LCD kullanılmıştır.

Grafik LCD'ler harfler, rakamlar ve karakterler göstermenin yanı sıra tek renkli resimler göstermek amacıyla kullanılırlar. İstenilen resimler, geometrik şekiller ve grafikler belirli kurallara uygun şekilde biçimlendirilerek bu tip ekranlarda istenilen amaca yönelik biçimde gösterilebilirler.

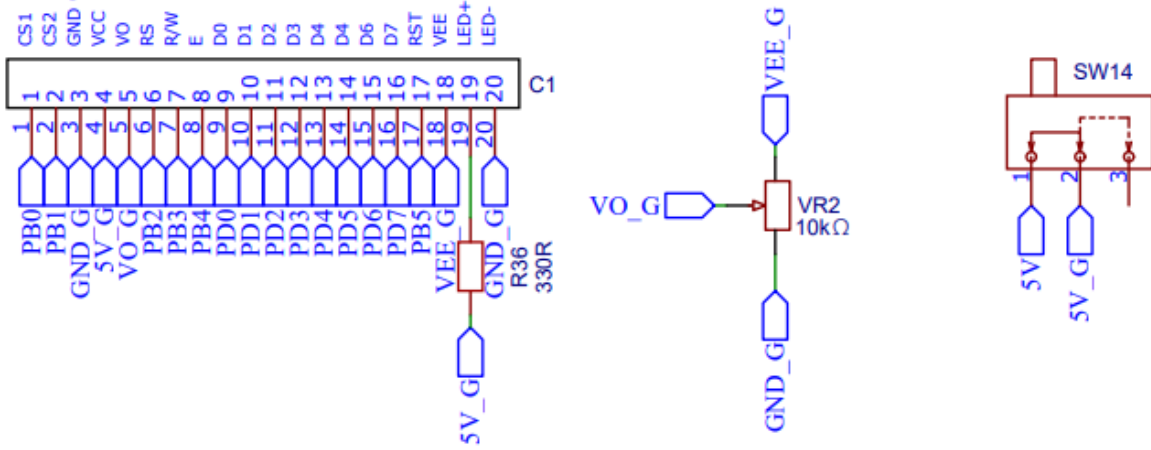
Grafik LCD ekranlar 128x64, 240x128, 240x64 gibi farklı çözünürlüklerde üretilmektedirler. ARMapp-18 uygulama setinde, piyasada en çok kullanılan grafik LCD kontrolcülerinden olan KS0108 kontrolcü çipe sahip 128x64 çözünürlüğünde bir grafik LCD kullanılmıştır.



Şekil 6.1: 128x64 Grafik LCD.

MikroC pro for ARM derleyicisinde KS0107/KS0108 kontrolcü çiplere ait bir kütüphane dâhili olarak bulunmaktadır. Ayrıca tek renkli bmp dosyalarını ".c" uzantılı dosyalara dönüştürmek için de bir araç bulunmaktadır. Bu araca "Tools" sekmesi altında bulunan "GLCD Bitmap Editor" seçeneğine tıklayarak ulaşılabilir. Bu araçta "Load BMP" tuşuna tıklanarak 128x64 çözünürlüğüne tek renkli bir ".bmp" uzantılı resim dosyası seçilerek ".c" uzantılı bir dosyaya dönüştürülebilir. Bu şekilde resim dosyaları mikrodenetleyici kullanılarak grafik LCD ekran da gösterilebilir.

GRAFİK LCD



Şekil 6.2: ARMapp-18 Uygulama setindeki Grafik LCD modülüne ait devre şeması.

ARMapp-18 uygulamaları setinde kullanılan Grafik LCD modülüne ait devre şeması şekil 6.2’de görülmektedir. Data pinleri için D portu, kontrol pinleri için ise B portuna ait pinler kullanılmıştır.

Bu uygulamada Grafik LCD üzerinde öncelikle çeşitli fontlar kullanılarak bir mesaj yazdırılmış, daha sonra çeşitli geometrik şekiller çizdirilmiş ve son olarak ta bir resim gösterilmiştir.

Uygulama Kodları:

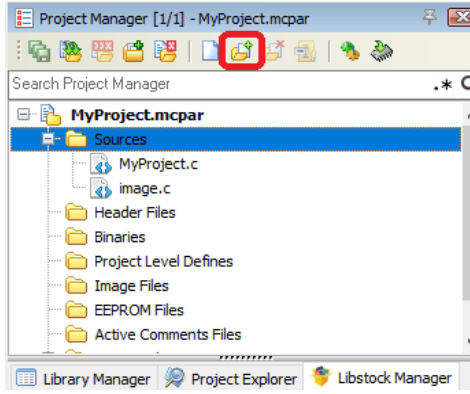
```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //      Grafik LCD Kullanımı      //
4 //      MikroC v6.2 - STM32F407VG //
5 //      ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 //      http://elektrovadi.com      //
8 //      http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10 extern const code char world[1024]; // Harici olarak tanımlanan world isimli resim dizisi.
11 unsigned long GLCD_DataPort_Input at GPIOID_IDR; // Grafik LCD data pinleri
12 unsigned long GLCD_DataPort_Output at GPIOID_ODR; // 0.-7. bitlere bağlanıyor.
13
14 sbit GLCD_CS2 at GPIOB_ODR.B0; // Kontrol pinleri
15 sbit GLCD_CS1 at GPIOB_ODR.B1; // bağlantıları
16 sbit GLCD_RS at GPIOB_ODR.B2;
17 sbit GLCD_RW at GPIOB_ODR.B3;
18 sbit GLCD_EN at GPIOB_ODR.B4;
19 sbit GLCD_RST at GPIOB_ODR.B5;
20
21 void main() // Ana program boğu.
22 {
23   Glcd_Init(); // GLCD kurulumu yapılıyor.
24   Glcd_Fill(0); // Grafik LCD temizleniyor.
25   Glcd_Set_Font(Font_Glcd_System5x7,5,7,32); // 5x7 Font seçildi.
26   Glcd_Write_Text("ARMapp-18",35,0,1); // 0. sayfaya yazdırılıyor.
27   Glcd_Set_Font(Font_Glcd_Character8x7,8,7,32); // 8x7 font seçildi.
28   Glcd_Write_Text("STM32F407VG",12,2,1); // 2. sayfaya yazdırılıyor.
29   Glcd_Set_Font(Font_Glcd_System3x5,3,5,32); // 3x5 font seçildi.
30   Glcd_Write_Text("ARM UYGULAMALARI SETI ",22,4,1); // 4. sayfaya yazdırılıyor.
31   Glcd_Set_Font(FontSystem5x7_v2,5,7,32); // 5x7 v2 font seçildi.
32   Glcd_Write_Text("MikroC",45,6,1); // 6. sayfaya yazdırılıyor.
33   delay_ms(6000); // 6 saniye bekleniyor.
34   Glcd_Fill(0); // Glcd temizleniyor.
35   Glcd_Line(0,0,127,63,1); // Sol üst köşeden sağ alt köşeye çizgi çiziliyor.
36   Glcd_Line(127,0,0,63,1); // Sağ üst köşeden sol alt köşeye çizgi çiziliyor.
37   delay_ms(3000); // 3 saniye bekleniyor.
38   Glcd_Circle(63,31,25,2); // 25 piksel yarıçaplı çember çiziliyor.
39   delay_ms(3000); // 3 saniye bekleniyor.
40   Glcd_Circle_Fill(31,31,18,2); // iki adet 18 piksel yarıçaplı
41   Glcd_Circle_Fill(95,31,18,2); // daire çizdiriliyor.
42   delay_ms(3000); // 3 saniye bekleniyor.
43   Glcd_Rectangle(0,0,20,20,1); // Kare çizdiriliyor.
44   Glcd_Rectangle_Round_Edges_Fill(107,43,127,63,5,2); // Yuvarlak köşeli içi dolu dörtgençizdiriliyor.
45   delay_ms(3000); // 3 saniye bekleniyor.
46   Glcd_Box(107,0,127,20,1); // İçi dolu kare çizdiriliyor.
47   Glcd_Rectangle_Round_Edges(0,43,20,63,5,2); // Yuvarlak köşeli dörtgen çizdiriliyor.
48   delay_ms(3000); // 3 saniye bekleniyor.
49   Glcd_Fill(0); // Glcd temizleniyor.
50   Glcd_PartialImage(60,30,30,30,128,64,world); // Resimin belli bir kısmı bastırılıyor.
51   delay_ms(3000); // 3 saniye bekleniyor.
52   Glcd_Fill(0); // Glcd temizleniyor.
53   Glcd_Image(world); // Resim ekrana bastırılıyor.
54 }

```

ARMapp-18

Grafik LCD’ de gösterilecek resme ait c dizisinin (“GLCD Bitmap Editor” ile oluşturulan) yeni bir “.c” uzantılı bir dosya oluşturularak ve istenilen isimle kaydedilerek programa dahil edilmesi gerekmektedir. Dosyayı programa dahil etmek için sağ üst kısımda yer alan “Project Manager” alanından proje dosyası altındaki sources klasörü seçilir ve dosya ekle butonuna (şekil 6.3) tıklanarak daha önce “.c” uzantılı kaydedilen dosya seçilir.



Şekil 6.3: Project Manager ile projeye dosya eklenmesi.

Harici resim dosyası:

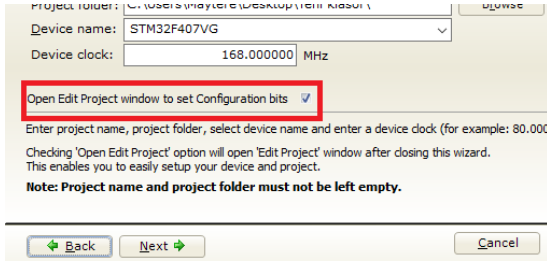
```



1 // -----
2 // GLCD Picture name: world.bmp
3 // GLCD Model: KS0108 128x64
4 // -----
5
6 const code char world[1024] = {
7 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
8 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
9 255, 255, 255, 255, 255, 255, 127, 63, 63, 159, 143, 15, 15, 7, 71, 71, 67, 51, 99, 123, 251,
10 251, 251, 249, 251, 255, 255, 251, 247, 247, 183, 231, 199, 15, 31, 15, 63, 63, 127, 127, 127,
11 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
12 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
13 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
14 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
15 255, 255, 255, 255, 255, 127, 63, 143, 135, 3, 129, 192, 0, 132, 226, 251, 253, 187, 153, 205,
16 196, 198, 195, 193, 224, 225, 240, 249, 255, 255, 239, 255, 255, 255, 255, 207, 189, 127, 24, 1,
17 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 7, 15, 95, 255, 255, 255, 255, 255,
18 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
19 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
20 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
21 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 254, 255, 254,
22 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 63, 190, 30, 127, 255, 255, 39, 131, 193, 248, 36,
23 16, 0, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
24 0, 0, 0, 0, 0, 3, 7, 127, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
25 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
26 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
27 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 31, 255, 255, 255,
28 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 143, 143, 143, 205, 193,
29 225, 224, 224, 208, 160, 192, 0, 0, 128, 0, 112, 48, 56, 48, 32, 0, 0, 0, 0, 0,
30 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 224, 192, 192, 199, 255, 255, 255, 255,
31 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
32 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
33 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
34 248, 3, 127, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 31, 15, 7, 7, 1, 1, 0, 0,
35 0, 0, 0, 0, 0, 7, 7, 14, 15, 7, 6, 15, 15, 14, 6, 14, 96, 192, 128, 0, 0,
36 0, 12, 8, 16, 8, 8, 216, 248, 252, 252, 248, 248, 224, 192, 140, 191, 255, 255, 254, 245, 255,
37 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
38 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
39 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
40 255, 255, 255, 255, 248, 225, 135, 223, 255, 255, 255, 255, 255, 248, 224, 192, 128, 128, 128,
41 0, 128, 128, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3,
42 6, 12, 16, 24, 136, 228, 254, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
43 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
44 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
45 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
46 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
47 255, 255, 255, 255, 255, 255, 254, 254, 240, 192, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
48 0, 0, 48, 248, 254, 126, 63, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
49 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
50 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
51 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
52 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
53 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 252, 248, 240, 224, 224, 224, 224, 224, 240, 240,
54 248, 248, 254, 254, 255, 252, 254, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
55 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
56 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255,
57 };

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.



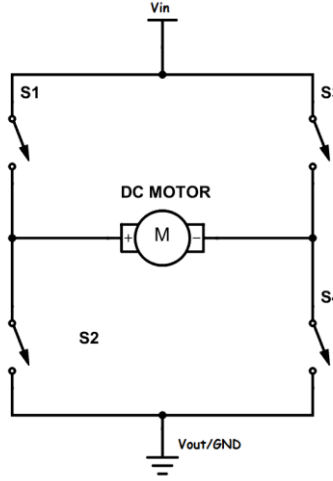
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “GLCD” kütüphanesini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- “GLCD Bitmap Editor” aracını kullanarak 128x64 boyutlarında bir tek renkli bmp resim dosyasını c dizisine çeviriniz ve kopyalayarak yeni bir dosya olarak proje klasörüne kaydediniz.
- 11- “Project Manager” kullanarak oluşturduğunuz resim datası içeren “.c” dosyasını projeye dâhil ediniz.
- 12- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 13- Piyano anahtarlardan sol taraftakinde bulunan K.LCD anahtarını ve sağ taraftakinde bulunan GLCD anahtarını açarak, bu modüllerin GND bağlantısını sağlayınız. Grafik LCD modülünün bitişiğinde bulunan anahtar “ON” konumuna, Karakter LCD modülünün bitişiğinde bulunan anahtar OFF konumuna getiriniz.
- 14-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 15- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- Grafik LCD’de farklı fontlar kullanarak isminizi, soy isminizi ve okul numaranızı yazınız.
- 2- Grafik LCD’de bir yazıyı ve resmi aynı anda gösterebilmek için yapılması gerekenleri araştırınız.

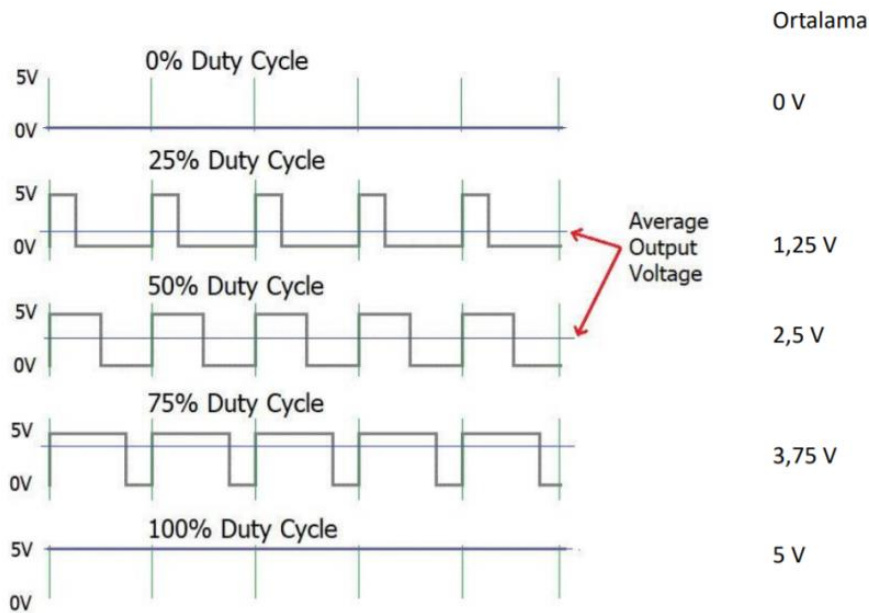
Uygulama 7: L293D Entegresi ile DC Motor Sürme Uygulaması.

Elektromekanik cihazların büyük bir bölümünde elektrik enerjisini hareket enerjisine çevirebilmek için DC motorlara ihtiyaç duyulmaktadır. Çoğu durumda bu motorların yön ve hız kontrollerinin de yapılabilmesi gerekmektedir. Bu tarz işlemleri gerçekleştirebilmek için temel olarak 4 adet elektronik anahtardan oluşan H-köprü devrelerine ihtiyaç duyulur. Şekil 7.1' de basit bir H köprü devresi görülmektedir.



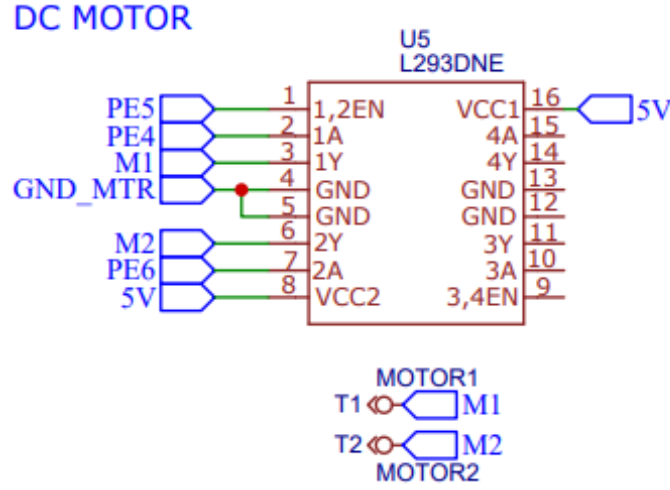
Şekil 7.1: Temel H-Köprü Devresi.

Şekil 7.1'deki H köprü devresinde motorun bir yöne dönmesi için S1 ve S4 anahtarlarının kapalı, S2 ve S3 anahtarlarının açık olması, diğer yöne dönmesi için ise S2 ve S3 anahtarlarının kapalı olması, S1 ve S4 anahtarlarının açık olması gerekmektedir. Yön kontrolü bu şekilde yapılırken hız kontrolünün mikrodenetleyici ile sağlanabilmesi için PWM (Pulse Width Modulation – Darbe Genişlik Modülasyonu) tekniğinin kullanılması gerekmektedir. PWM sinyali anahtarların girişlerine uygulanarak anahtarların belirli sürelerde sürekli açılıp kapanarak motorun uçlarına uygulanan ortalama voltaj değiştirilmiş olur. Bu şekilde motorun hızı ayarlanır. Şekil 7.2' de PWM sinyalinin görev saykılına göre ortalama voltajın değişimi görülmektedir.



Şekil 7.2: PWM sinyali.

ARMapp-18 uygulama setinde kullanılan L293D motor sürücüsü içerisinde bir H köprü devresini hazır olarak bulunduran oldukça kullanışlı bir motor sürücü entegresidir. ARMapp-18 uygulama setindeki DC motor sürücü devresine ait şema Şekil 7.3' te görülmektedir.



Şekil 7.3: ARMapp-18 Uygulama setindeki DC Motor modülüne ait devre şeması.

L93D motor sürücüsü 2 kanallı bir motor sürücüdür. Yani aynı anda 2 farklı DC motor kontrol edilebilmektedir. Fakat ARMapp-18 uygulama setinde bu kanallardan bir tanesi kullanılmıştır. Şemada M1 ve M2 ile gösterilen pinler DC motor bağlantı uçlarıdır.

MikroC pro for ARM derleyicisinde PWM sinyali üretebilmek için bir kütüphane bulunmaktadır ve STM32 mikrodenetleyicisinde PWM kullanımı için gerekli olan karmaşık register ayarlamaları çok basite indirgenerek birkaç fonksiyon ile kolayca yapılabilir hale getirilmiştir.

Bu uygulamada DC motor hızı artırılarak önce bir yöne daha sonra durdurulup tekrar hızlandırılarak ters yöne döndürülmektedir.

Uygulama Kodları:

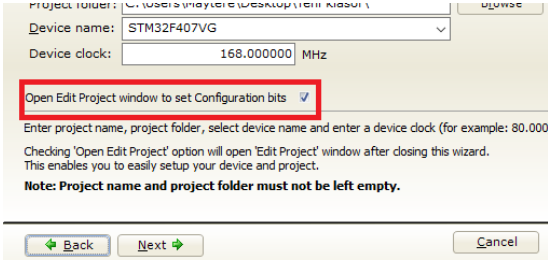
```



1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // L293D Motor Sürücü Uygulaması //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMap-18 Denei Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10 #define IN1 GPIOE_ODR.B4 // L293D entegresinin IN1 pini PE4 pinine baęlı
11 #define IN2 GPIOE_ODR.B6 // L293D entegresinin IN2 pini PE6 pinine baęlı
12
13 unsigned int dutyCycle; // unsigned int tipinde dutyCycle isimli deęişken tanımlanıyor.
14 unsigned int i=0; // unsigned int tipinde i isimli deęişken tanımlanıyor.
15
16 void ileri() // ileri isimli fonksiyon tanımlanıyor.
17 {
18 IN1=1; // IN1 girişine '1' veriliyor.
19 IN2=0; // IN2 girişine '0' veriliyor.
20 }
21
22 void geri() // geri isimli fonksiyon tanımlanıyor.
23 {
24 IN1=0; // IN1 girişine '0' veriliyor.
25 IN2=1; // IN2 girişine '1' veriliyor.
26 }
27
28 void kurulum() // kurulum isimli fonksiyon tanımlanıyor.
29 {
30 GPIO_Digital_Output(&GPIOE_BASE, _GPIO_PINMASK_4|_GPIO_PINMASK_6); // E portunun 4. ve 6. pinlerini dijital çıkış olarak ayarla.
31 IN1=0; // IN1 pinine 0 gönder.
32 IN2=0; // IN2 pinine 0 gönder.
33
34 dutyCycle=PWM_TIM9_Init(5000); // Timer 9 5KHz PWM üretecek şekilde ayarlanıyor.
35
36 PWM_TIM9_Set_Duty((dutyCycle/1000)*500, // PWM terslenmemiş biçimde ve %50 duty oranında
37 _PWM_NON_INVERTED,_PWM_CHANNEL1 // Kanal 1 çıkış verecek şekilde ayarlanıyor.
38 );
39
40 PWM_TIM9_Start(_PWM_CHANNEL1,&GPIO_MODULE_TIM9_CH1_PE5); // PE5 pininden çıkış alınacak şekilde PWM başlatılıyor.
41 }
42
43 void main() // Ana program bloęu
44 {
45 kurulum(); // kurulum alt programı çağırılıyor.
46 while(1) // Sonsuz döngü.
47 {
48 ileri(); // ileri fonksiyonu çağırılıyor.
49 for(i=300;i<1000;i++) // PWM değeri %30 dan başlayarak %100 olana kadar
50 { // 10 ms aralıklarla arttırılıyor.
51 PWM_TIM9_Set_Duty((dutyCycle/1000)*i,_PWM_NON_INVERTED,_PWM_CHANNEL1);
52 delay_ms(10);
53 }
54
55 delay_ms(1000); // 1 saniye bekleniyor.
56
57 PWM_TIM9_Set_Duty((dutyCycle/1000)*0,_PWM_NON_INVERTED,_PWM_CHANNEL1); // PWM duty saykılı 0 olarak ayarlanarak motor durduruluyor.
58 ////////////////////////////////////////////////////////////////////
59 geri(); // geri fonksiyonu çağırılıyor.
60 for(i=300;i<1000;i++) // PWM değeri %30 dan başlayarak %100 olana kadar
61 { // 10 ms aralıklarla arttırılıyor.
62 PWM_TIM9_Set_Duty((dutyCycle/1000)*i,_PWM_NON_INVERTED,_PWM_CHANNEL1);
63 delay_ms(10);
64 }
65
66 delay_ms(1000); // 1 saniye bekleniyor.
67
68 PWM_TIM9_Set_Duty((dutyCycle/1000)*0,_PWM_NON_INVERTED,_PWM_CHANNEL1); // PWM duty saykılı 0 olarak ayarlanarak motor durduruluyor.
69 }
70 }

```


İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHz) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.



- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemi tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “PWM” kütüphanesini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Piyano anahtarlardan sağ taraftakinde bulunan DC Motor anahtarını açarak, bu modüllerin GND bağlantısını sağlayınız.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- Motorun hızını arttırmak ve azaltmak için menü tuş takımında bulunan sağ ve sol butonlarını kullanınız. Sağ butona bastıkça motor hızını arttıran, sol butona bastıkça motor hızını azaltan, yukarı tuşuna basınca motoru bir yöne, aşağı tuşa basınca motoru diğer yöne döndüren, orta tuşa bir kez basıldığında motoru çalıştırıp, bir kez daha basıldığında durduran programı yazınız.
- 2- DC Motorlarda elektriksel frenleme kavramını araştırınız.

Uygulama 8: Timer ile Yazılımsal PWM Uygulaması.

Timerlar mikrodenetleyicilerde bulunması gereken hayati modüllerdendir. Kritik zamanlama işlemlerinin yapılabilmesine olanak sağlarlar. STM32 serisi mikrodenetleyicilerde farklı özelliklerde birden fazla timer modülü bulunmaktadır. STM32F407VG mikrodenetleyicisinde toplam 14 adet timer bulunmaktadır. Bu timerlara ait tablo şekil 8.1’de görülebilir.

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/compare channels	Complementary output	Max interface clock (MHz)	Max timer clock (MHz)
Advanced-control	TIM1, TIM8	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	84	168
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	84	168
	TIM10, TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	84	168
	TIM12	16-bit	Up	Any integer between 1 and 65536	No	2	No	42	84
	TIM13, TIM14	16-bit	Up	Any integer between 1 and 65536	No	1	No	42	84
Basic	TIM6, TIM7	16-bit	Up	Any integer between 1 and 65536	Yes	0	No	42	84

Şekil 8.1: STM32F4 Timer Modülleri ve Özellikleri.

Bu uygulamada 32 bit uzunluğunda sayıcıya sahip Timer2 kullanılmıştır. Timer2’nin periyot hesaplaması;

$$T_{timer} = \frac{Timer\ Clock}{(Prescaler + 1) \times (ARR + 1)}$$

formülü ile hesaplanır. Bu formülde Timer2’nin beslendiği saat kaynağı frekansı (84 MHz), Prescaler kaydedicisine eklenen değerin bir fazlasının ve otomatik yenilenen değer kaydedicisine yüklenen değerin bir fazlasının çarpımlarına bölünerek elde edilen sonuç bize Timer2’nin kesme üreteceği süreyi vermektedir.

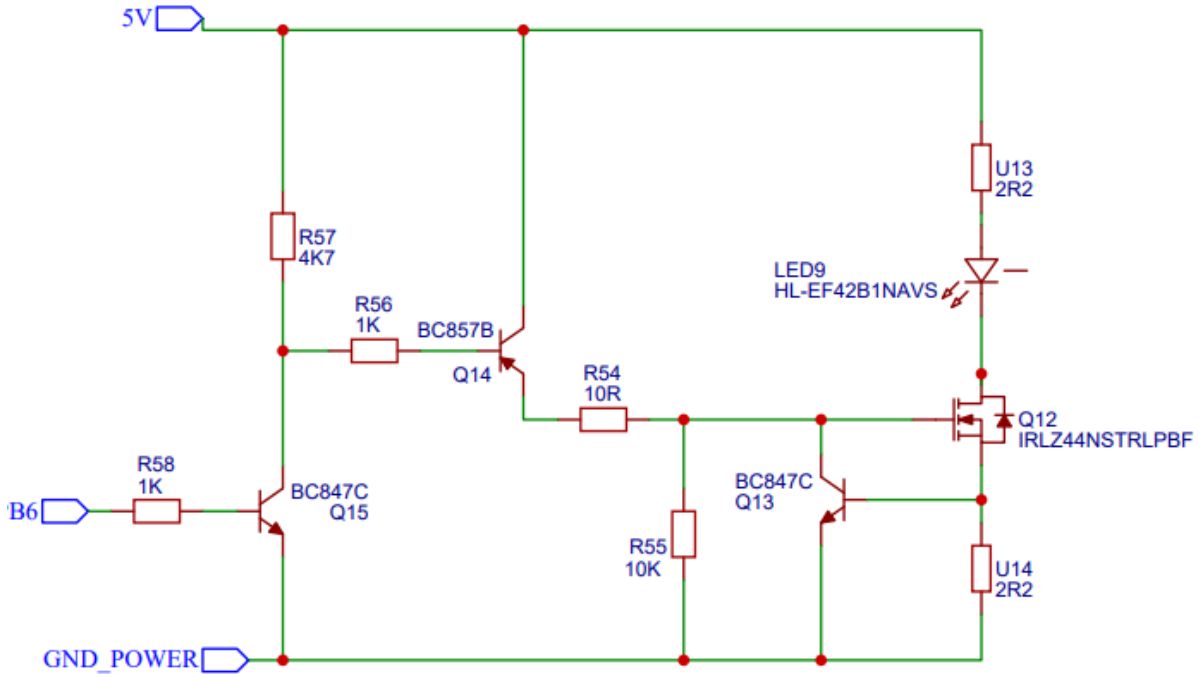
Bizim kodlarımızda yaptığımız ayarlamalar sonucunda

$$f_{timer} = \frac{84 \text{ MHz}}{(0 + 1) \times (83 + 1)} = 1 \mu s$$

işlemi yapılır ise timer kesmesinin oluşacağı periyot 1us olarak bulunur.

Bu uygulamada 1 us'lik çözünürlükte 1024 adımlık (yaklaşık 1Khz) bir yazılımsal PWM oluşturularak aynı zamanda hem PA15 pinine sinyal çıkışı verilmiş hem de ARMapp-18 uygulama seti üzerindeki Power LED parlaklığı değiştirilmiştir.

Şekil 8.2'de ARMapp-18 deney seti üzerinde bulunan Power Led modülünün şeması görülmektedir.



Şekil 8.2: ARMapp-18 Uygulama setindeki Power Led modülüne ait devre şeması.

Şemaya dikkat edilirse power led bir N kanal MOSFET ile sürülmektedir. Bu MOSFET'in source ucu ile GND arasında 2.2 Ω değerinde bir direnç bağlanmıştır. Bu direncin görevi üzerindeki gerilim 0.7 V üzerine çıkınca BC847 transistörünü iletime geçirerek MOSFET'i kesime götürmek böylece power led üzerindeki akımın 300-350 ma üzerine çıkmasını engelleyerek akım sınırlaması yapmaktır. Q12 ve Q11 transistörleri ise MOSFET'in gate ucuna uygulanacak STM32F407 mikrodenetleyicisinden çıkan 3V'luk sinyali 5V seviyesine çıkarmak amacıyla kullanılmışlardır. Power ledlerde akım sınırlamak önemlidir çünkü iç dirençleri sıcaklıklarıyla birlikte azalmakta, iç dirençleri azaldıkça üzerlerinden daha fazla akım geçmekte bu da daha fazla ısınmalarına sebep olmaktadır.

Bu uygulamada oluşturulan yazılımsal PWM sinyali power led ile gözlemlenebileceği gibi PA15 pininden de bir osiloskop yardımı ile gözlemlenebilir.

Ayrıca uygulamada Kare Sinyal modülündeki potansiyometre yardımı ile PWM sinyalinin görev sayıklı değiştirilmektedir. Bunun mümkün olabilmesi için STM32F407 mikrodenetleyicisinde dâhili olarak bulunan analog dijital çevirici modülü kullanılmaktadır. Potansiyometrenin orta ucundaki voltaj PC5 pininden ölçülür ve 0-4095 arasında değerlendirilir. Oluşturduğumuz PWM sinyali 0-1023 arasında değerler alacağı için ADC'den gelen sinyal 4'e bölünmektedir.

Uygulama Kodları:

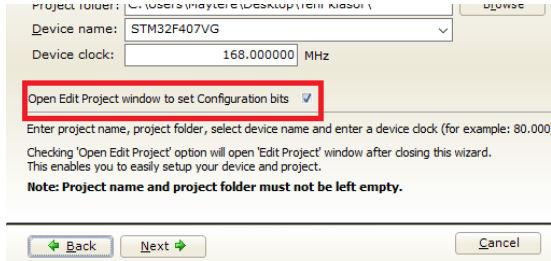
```



1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //      Kare Dalga Oluşturma Uygulaması //
4 //      MikroC v6.2 - STM32F407VG      //
5 //      ARMapp-18 Deneysel Seti için yazılmıştır //
6 //*****//
7 //      http://elektrovadi.com        //
8 //      http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10 int pwmSayac=0; // int tipinde pwmSayac isminde değişken tanımlanıyor.
11 int pwmDuty=0; // int tipinde pwmDuty isminde değişken tanımlanıyor.
12
13 void InitTimer2(){ // Timer2'nin kurulumunun yapıldığı fonksiyon.
14     RCC_APB1ENR.TIM2EN = 1; // TIMER2 clock kaynağı aktif ediliyor.
15     TIM2_CR1.CEN = 0; // TIMER2 sayıcısı pasif hale getiriliyor.
16     TIM2_PSC = 0; // TIMER2 prescaler değeri 0 olarak ayarlanıyor.
17     TIM2_ARR = 83; // TIMER2 otomatik yükleme değeri 83 olarak ayarlanıyor.
18     NVIC_IntEnable(IVT_INT_TIM2); // TIMER2 Kesmesine izin veriliyor.
19     TIM2_DIER.UIE = 1; // TIMER2 güncelleme kesmesine izin veriliyor.
20     TIM2_CR1.CEN = 1; // TIMER2 sayıcısı aktif ediliyor.
21 }
22
23 void main() // Ana program bloğu
24 {
25     GPIO_Digital_Output(&GPIOA_BASE, _GPIO_PINMASK_15); // PA15 pini dijital çıkış olarak ayarlanıyor.
26     GPIO_Digital_Output(&GPIOB_BASE, _GPIO_PINMASK_6); // PB6 pini dijital çıkış olarak ayarlanıyor. (Power Led)
27     InitTimer2(); // TIMER2 kurulum fonksiyonu çağırılıyor.
28     ADC1_Init(); // ADC1 kuruluyor.
29     ADC_Set_Input_Channel(_ADC_CHANNEL_15); // PC5 pini ADC girişi olarak ayarlanıyor.
30
31     while(1) // Sonsuz döngü
32     {
33         pwmDuty=ADC1_Get_Sample(15); // PC5 pinindeki analog değer (0-4095) arası okunuyor
34         pwmDuty/=4; // ve pwmDuty değişkenine aktarılıyor.
35     } // 0-1024 arasında ayar yapabilmek için pwmDuty değeri 4'e bölünüyor.
36 }
37
38 void Timer2_interrupt() iv IVT_INT_TIM2 { // 1us'lik timer kesme alt programı.
39     TIM2_SR.UIF = 0; // Timer kesme bayrağı temizleniyor.
40     pwmSayac++; // pwmSayac değişkeni bir arttırılıyor.
41     if(pwmSayac>1024 && pwmDuty>10) // Eğer pwmSayac değeri 1024'e ulaştıysa ve pwmDuty değeri 10'dan büyükse
42     { // (Burada pwmDuty değerinin 10'dan büyük olma şartı ADC'nin 0
43         GPIOA_ODR.B15=1; // değerine düşmemesinden kaynaklanmaktadır.
44         GPIOB_ODR.B6=1; // Kare dalga çıkışlarını HIGH konuma getir.
45         pwmSayac=0; // pwmSayac değişkeni 0 yapılarak baştan sayması sağlanıyor.
46     }
47
48     if(pwmSayac>pwmDuty) // Eğer pwmSayac değişkeni pot ile ayarlanan değerden
49     { // büyükse pwm çıkışlarını LOW yap.
50         GPIOA_ODR.B15=0;
51         GPIOB_ODR.B6=0;
52     }
53 }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.



- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan "Library Manager" sekmesinden "ADC" kütüphanesini seçiniz.
- 9- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Piyano anahtarlardan sol taraftakinde bulunan POWER LED anahtarını ve sağ taraftakinde bulunan KARE DALGA anahtarını açarak, bu modüllerin GND bağlantısını sağlayınız.
- 12-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- Kare Dalga modülündeki potansiyometre konumunu değiştirerek programın çalışmasını gözlemleyiniz. (Dikkat: Power led'e doğrudan uzun süre bakmak gözlerde kalıcı hasara sebep olabilir.)

Sorular:

- 1- Timer2 çözünürlüğünü 0.5us olarak ayarlayarak yazılımsal PWM sinyalini 0-2048 arasında değişecek şekilde ayarlayınız.
- 2- Aynı işlemi Timer1 ile de gerçekleştiriniz.

Uygulama 9: LM35 ile Sıcaklık Ölçümü Uygulaması.

Mikrodenetleyiciler tamamen dijital yapılar olmakla birlikte bazı uygulamalarda sıcaklık, ışık şiddeti, hacim, ağırlık gibi analog verilerle de çalışmak gerekebilmektedir. Bu gibi durumlarda analog-dijital çeviricilere (ADC) ihtiyaç duyulur. Analog-Dijital çeviriciler, analog voltaj değerlerini sayısal değerlere çevirirler. ADC modüllerinin bit uzunlukları ne kadar yüksek ise çözünürlükleri o kadar düşük olur. Örneğin 10 bitlik bir ADC ölçtüğü voltaj aralığını 1024 eşit parçaya böler. Aşağıdaki formül ADC çözünürlüğünü ifade etmektedir.

$$ADC_{res} = \frac{V_{max} - V_{min}}{2^n}$$

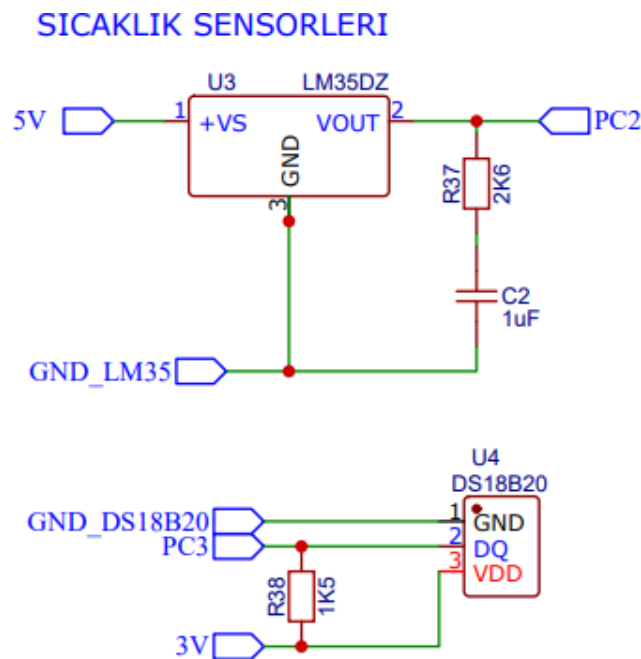
Burada “Vmax” ölçülecek maksimum voltaj, “Vmin” ölçülecek minimum voltaj, “n” ise ADC bit uzunluğudur.

Örneğin ARMapp-18 deney setinde kullanılan STM32F4 discovery kitinde ADC modülünün referans voltaj aralığı 0-3V değerlerindedir. STM32F407VG mikrodenetleyicisinin ADC bit uzunluğu 12 bit olduğu için çözünürlük;

$$\frac{3V - 0V}{2^{12}} = 0,732421875 \text{ mV}$$

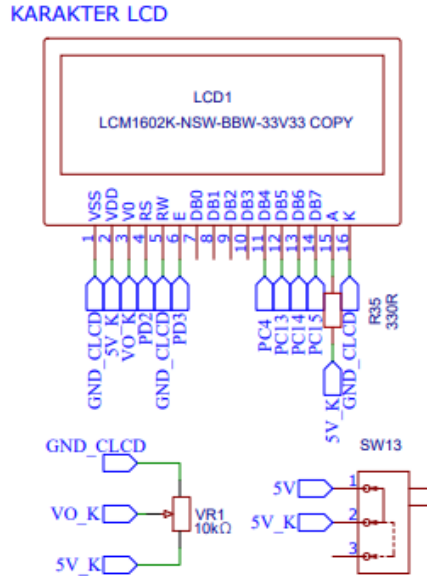
olarak hesaplanır.

LM35 sıcaklık sensörü -55°C ile 150°C arasında ölçüm yapabilen ve 10-mV/°C değerinde çıkış veren bir devre elemanıdır. Şekil 9.1’de ARMapp-18 deney setinde bulunan LM35 sıcaklık sensörü modülüne ait devre şeması görülmektedir.



Şekil 9.1: ARMapp-18 Uygulama setindeki LM35 Sıcaklık Sensörü modülüne ait devre şeması.

Bu uygulamada LM35 sıcaklık sensöründen okunan veri önce voltaj değerine daha sonra da sıcaklık değerine dönüştürülerek karakter LCD'de gösterilmektedir. LM35 sıcaklık sensöründen okunan veri 0-4095 arasında olan sayısal bir değerdir. Bu sayısal değeri voltaj değerine dönüştürebilmek için öncelikle ADC çözünürlüğü ile (mV olarak) ADC'den okunan sayısal değer çarpılır. Sonuç bize ölçülen voltaj değerini verecektir. Hesaplanan bu voltaj değerini de, LM35 her 1°C başına 10mV değerinde voltaj ürettiği için, 10mV değerine bölerek sıcaklık bilgisine ulaşabiliriz.



Şekil 9.2: ARMapp-18 Uygulama setindeki Karakter LCD modülüne ait devre şeması.

Uygulama Kodları:

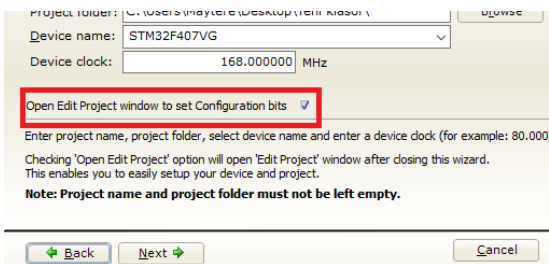
```



1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // LM35 Sıcaklık Sensörü ve LCD Uygulaması //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11 sbit LCD_RS at GPIOD_ODR.B2; // RS pini PD2 pinine bağlı.
12 sbit LCD_EN at GPIOD_ODR.B3; // EN pini PD3 pinine bağlı.
13 sbit LCD_D4 at GPIOD_ODR.B4; // D4 pini PD4 pinine bağlı.
14 sbit LCD_D5 at GPIOD_ODR.B5; // D5 pini PD5 pinine bağlı.
15 sbit LCD_D6 at GPIOD_ODR.B6; // D6 pini PD6 pinine bağlı.
16 sbit LCD_D7 at GPIOD_ODR.B7; // D7 pini PD7 pinine bağlı.
17
18 unsigned int okunan; // unsigned int tipinde okunan isimli değişken tanımlanıyor.
19 float voltaj; // float tipinde voltaj isimli değişken tanımlanıyor.
20 float sicaklik; // float tipinde sicaklik isimli değişken tanımlanıyor.
21 char txt[15]; // txt isminde 15 karakterli char tipinde dizi tanımlanıyor.
22
23
24 void main() // ana program bloğu
25 {
26 ADC1_Init(); // ADC kurulumu yapılıyor.
27 ADC_Set_Input_Channel(_ADC_CHANNEL_12); // ADC girişi 12. kanal olarak ayarlanıyor. (PC2 pini)
28 Lcd_Init(); // LCD kurulumu tamamlanıyor.
29 Lcd_Cmd(_LCD_CLEAR); // LCD temizleniyor.
30 Lcd_Cmd(_LCD_CURSOR_OFF); // LCD imleci tamamlanıyor.
31 Lcd_Out(1,1,"Sicaklik="); // LCD'nin 1. satır 1. sütununa "Sicaklik=" yazılıyor.
32 while(1) // Sonsuz döngü
33 {
34 okunan=ADC1_Read(12); // ADC'nin 12. kanalından okuma yapılıyor. (PC2)
35 voltaj=(3300.0/4095)*okunan; // Okunan ADC değeri mV cinsinden voltaja çeviriliyor.
36 sicaklik=voltaj/10; // miliVolt değeri sıcaklık değerine dönüştürülüyor.
37 FloatToStr(sicaklik,txt); // Float tipindeki sıcaklık değeri LCD'de yazdırılmak üzere
38 Lcd_Out(2,1,txt); // String'e dönüştürülüyor ve LCD'de gösteriliyor.
39 Lcd_Chr_CP(223); // Derece sembolü (°) LCD ye yazdırılıyor.
40 Lcd_Chr_CP('C'); // Sıcaklık Celsius tipinde olduğu için ekrana C yazdırılıyor.
41 delay_ms(100); // 100 ms bekleniyor.
42 }
43 }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.



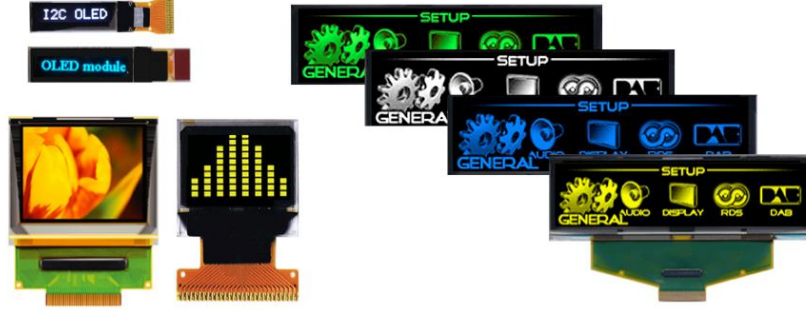
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “ADC” ve “LCD” kütüphanelerini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 14- Pişano anahtarlardan sol tarafta bulunan K.LCD anahtarını ve sağ tarafta bulunan GLCD anahtarını açarak, bu modüllerin GND bağlantısını sağlayınız. Karakter LCD modülünün bitişğinde bulunan anahtarı “ON” konumuna, Grafik LCD modülünün bitişğinde bulunan anahtarı OFF konumuna getiriniz.
- 11-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 12- LM35 sıcaklık sensörüne parmağınızla dokunarak sıcaklık değişimini karakter LCD’ de gözlemleyiniz.

Sorular:

- 1- Karakter LCD’de gözlemlenen sıcaklık değeri çok oynaktır. Bu değeri daha stabil hale getirmek için kullanılabilir yazılımsal filtreleri araştırınız.
- 2- Medyan filtre kullanarak ölçülen sıcaklık değerini filtreleyiniz.

Uygulama 10: Oled Display Uygulaması.

OLED (Organik Işık Yayan Diyotlar), iki iletken arasında bir dizi organik ince film yerleştirilerek yapılan düz ışık yayan bir teknolojidir. Elektrik akımı uygulandığında parlak bir ışık yayılır. OLED'ler, arkadan aydınlatma gerektirmeyen ve LCD ekranlardan daha ince ve daha verimli olan (arka ışık gerektirenlere göre) yayıcı ekranlardır. Monochrome OLED ekran modülleri, ekran üzerindeki yanan piksellere karşı son derece koyu bir arka plana sahip olup bu sebeple çok yüksek kontrast oranına sahiptir. ARMapp-18 uygulama setinde kullanılan tek renkli 128x64 piksel OLED ekran modülü, 175° 'ye kadar çok geniş bir görüntüleme açısına sahiptir.

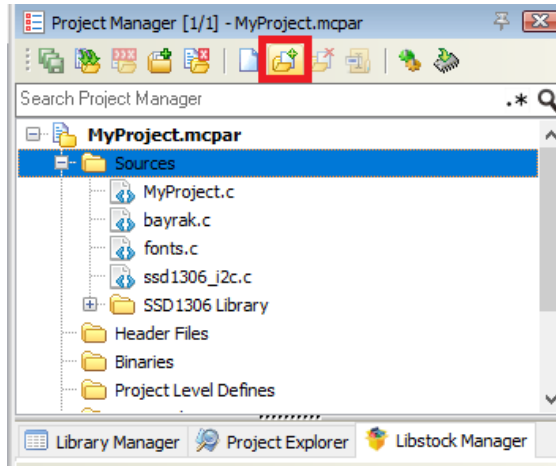


Şekil 10.1: Çeşitli Oled ekran çeşitleri.

Oled ekranlar da çalıştırılabilmek için sürücü çiplere ihtiyaç duyarlar. ARMapp-18 uygulama setinde bulunan oled ekranın sürücü çipi SSD1306 isimli kontrolcüdür.

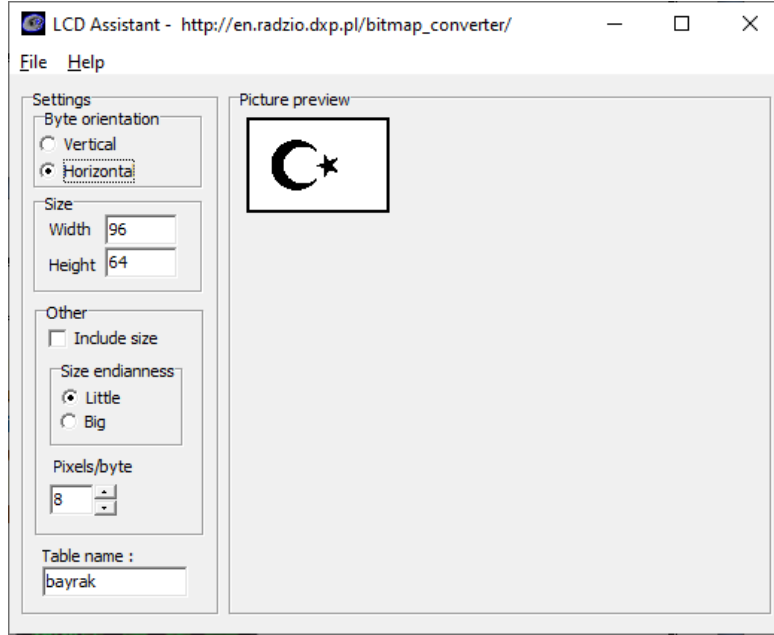
MikroC içerisinde dâhili olarak herhangi bir oled ekran kütüphanesi bulunmamaktadır. Bu sebeple SSD1306 kontrolcü çipine ait datasheet kullanılarak yeni bir kütüphane yazılabilir veya hâlihazırda var olan bir kütüphane STM32F407 mikrodenetleyicisine ve MikroC' ye uyarlanabilir. Bu uygulamada MikroC için uyarlanmış bir kütüphane kullanılacaktır.

Kütüphane dosyaları ".h" uzantılı header dosyaları ve ".c" uzantılı fonksiyonları barındıran dosyalardan oluşmaktadır. MikroC'de ".c" uzantılı dosyaları projeye dahil etmek için "Project Manager" kısmından "Sources" klasörü seçilerek kullanılacak bütün dosyalar projeye dahil edilmelidir. SSD1306 kütüphanesindeki fonksiyon prototiplerini barındıran "ssd1306_i2c.h" dosyası da projeye "#include" yönergesi kullanılarak ana program dosyasının başında projeye dâhil edilmelidir.



Şekil 10.2: MikroC'de Project Manager ile kütüphane dosyalarının eklenmesi.

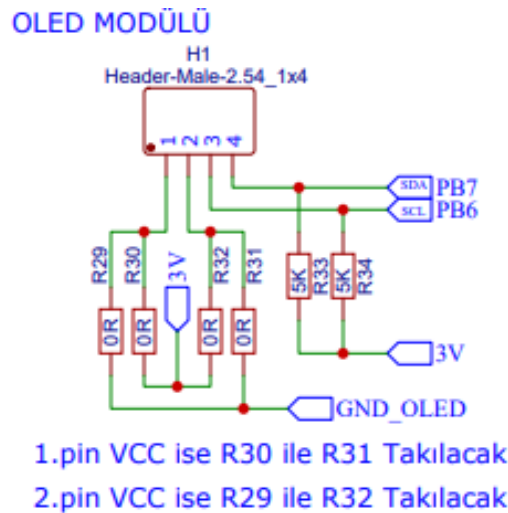
SSD1306 OLED ekranda resim gösterebilmek için ise öncelikle resim dosyasının byte dizisine dönüştürülmesi gerekmektedir. Şekil 10.3' te LCD Assistant programının ekran görüntüsü görülebilmektedir.



Şekil 10.3: LCD Assistant programı.

Bu programda "File" menüsü altından "Load Image" seçeneği kullanılarak tek renkli bmp formatında resim dosyası programa yüklenebilir. Daha sonra "Byte orientation" kısmından "Horizontal" seçeneği işaretlenmelidir. Dizimize "Table name" kısmından isim verebiliriz. Son olarak yine "File" menüsünden "Save output" seçeneği kullanılarak oluşturulan dizimizin içinde bulunduğu dosya ".c" uzantısıyla isimlendirilerek istenilen yere kaydedilir. Bu uygulamada dizi ismi programın sol alt kısmındaki kutucuktan "bayrakResmi" olarak adlandırılmış ve proje klasörüne kaydedilmiştir.

ARMap-18 uygulama setinde kullanılan SSD1306 OLED ekran modülüne ait devre şeması şekil 10.4' te görülmektedir.



Şekil 10.4: ARMap-18 Uygulama setindeki SSD1306 Oled Ekran modülüne ait devre şeması.

Uygulama Kodları:

```

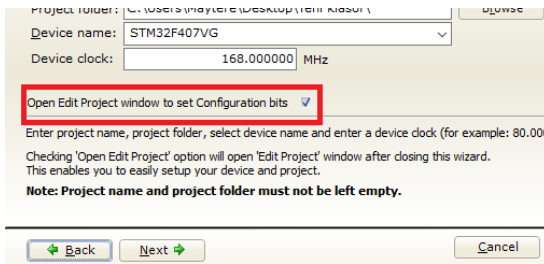
1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //   SSD1306 Oled Display Uygulaması //
4 //   MikroC v6.2 - STM32F407VG //
5 //   ARMap-18 Deney Seti için yazılmıştır //
6 //*****//
7 //   http://elektrovadi.com //
8 //   http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10 #include "ssd1306_i2c.h" // Kütüphane fonksiyonlarını içeren header dosyası projeye dahil ediliyor.
11 char* text = "SSD1306 Test Programı"; // text isminde bir karakter dizisi (string) tanımlanıyor.
12 char i=0,y=0,x=0; // i,y ve x isminde char tipinde değişkenler tanımlanıyor.
13 extern const unsigned char bayrakResmi[]; // bayrakResmi ismindeki harici değişken programa tanıtılıyor.
14 void main() // ana program bloğu
15 {
16   SSD1306_begin(SSD1306_SWITCHCAPVCC, SSD1306_I2C_ADDRESS); // ssd1306 kurulumu tamamlanıyor.
17   SSD1306_clearDisplay(); // Lcd temizleniyor.
18   SSD1306_GotoXY(35, 10); // imleç x=35, y=10 konumuna getiriliyor.
19   SSD1306_Puts ("beti", &Font_16x26, 1); // 16x26 boyutlarındaki font ile ekrana koyu renkli yazı yazdırılıyor.
20   SSD1306_GotoXY(10, 40); // imleç x=10, y=40 konumuna getiriliyor.
21   SSD1306_Puts ("ELEKTRONIK", &Font_11x18, 1); // 11x18 boyutlarındaki font ile ekrana koyu renkli yazı yazdırılıyor.
22   SSD1306_display(); // tampon belleğe yazılan veriler ekranda gösteriliyor.
23   delay_ms(3000); // 3 saniye bekleniyor.
24
25   SSD1306_clearDisplay(); // Lcd temizleniyor.
26   SSD1306_GotoXY(x, y); // Bütün yazı karakterleri sırasıyla ekrana yazdırılıyor.
27   for(i='!';i<='~';i++)
28   {
29     if(x>0 && x%18==0) // yan yana 18 (7x18=126) karakter geldiği zaman alt satıra geçen şart.
30     {
31       x=0;
32       y+=10;
33       SSD1306_GotoXY(0, y);
34     }
35     SSD1306_Putc(i,&Font_7x10,1); // Karakterler sırasıyla yazdırılıyor.
36     x++; // satırda kaçınıcı karakterde olduğumuzu tutan değişken.
37   }
38   SSD1306_display(); // tampon belleğe yazılan veriler ekranda gösteriliyor.
39   delay_ms(5000); // 5 saniye bekleniyor.
40
41   SSD1306_clearDisplay(); // Lcd temizleniyor.
42   SSD1306_DrawLine(20, 10, 100, 50, WHITE); // 20,10 konumundan 100,50 konumuna çizgi çiziliyor.
43   SSD1306_DrawCircle(64,32,20,WHITE); // 64,32 konumuna 20 piksel çapında çember çiziliyor.
44   SSD1306_DrawRectangle(0,0,127,63,WHITE); // 0,0 - 127,63 konumları köşeleri olan dikdörtgen çiziliyor.
45   SSD1306_DrawTriangle(64,0,10,58,120,40,WHITE); // Köşeleri 64,0 - 10,58 - 120,40 olan üçgen çiziliyor.
46   SSD1306_display(); // tampon belleğe yazılan veriler ekranda gösteriliyor.
47   delay_ms(3000); // 3 saniye bekleniyor.
48
49   SSD1306_clearDisplay(); // Lcd temizleniyor.
50   SSD1306_DrawFilledTriangle(64,0,10,58,120,40,WHITE); // Köşeleri 64,0 - 10,58 - 120,40 olan içi dolu üçgen çiziliyor.
51   SSD1306_DrawFilledRectangle(50,15,30,30,BLACK); // Köşeleri 50,15 -30,30 olan rengi ters çevrilmiş dikdörtgen çiziliyor.
52   SSD1306_DrawFilledCircle(65, 30, 15, WHITE); // merkezi 65,30 olan 15 piksel çaplı daire çiziliyor.
53   SSD1306_display(); // tampon belleğe yazılan veriler ekranda gösteriliyor.
54   delay_ms(3000); // 3 saniye bekleniyor.
55
56   SSD1306_clearDisplay(); // Lcd temizleniyor.
57   SSD1306_DrawBitmap(0, 0, bayrakResmi, 96, 63, 1); // Bayrak resmi 0,0 konumundan 96,63 konumları arasına basılıyor.
58   SSD1306_display(); // tampon belleğe yazılan veriler ekranda gösteriliyor.
59   delay_ms(1000); // 1 saniye bekleniyor.
60   SSD1306_StartScrollRight(0, 7); // Ekrandaki page lerin tamamı 1 saniye boyunca sağa kaydırılıyor.
61   delay_ms(1000);
62   SSD1306_StartScrollLeft(0,7); // Ekrandaki page lerin tamamı 1 saniye boyunca sola kaydırılıyor.
63   delay_ms(1000);
64   SSD1306_StartScrollDiagRight(0, 7); // Ekrandaki page lerin tamamı 3 saniye boyunca çarpaz sağa kaydırılıyor.
65   delay_ms(3000);
66   SSD1306_StartScrollDiagLeft(0, 7); // Ekrandaki page lerin tamamı 3 saniye boyunca çarpaz sola kaydırılıyor.
67   delay_ms(3000);
68   SSD1306_stopscroll(); // Ekranın kaydırılması durduruluyor.
69   delay_ms(1500); // 1.5 saniye bekleniyor.
70   SSD1306_Dim(1); // Ekran parlaklığı dimleniyor.. (1-dim, 0 - normal)
71   delay_ms(1500); // 1.5 saniye bekleniyor.
72
73   while(1) // Ekran sürekli 500 ms kapatılıyor, 1500 ms açık tutuluyor.
74   {
75     SSD1306_OFF();
76     delay_ms(500);
77     SSD1306_ON();
78     delay_ms(1500);
79   }
80 }



```

Bu uygulamada öncelikle ekrana farklı fontlarda bir yazı yazdırılmış, daha sonra kütüphanede mevcut olan bütün karakterler ekrana yazdırılmış, geometrik şekiller çizdirilmiş ve son olarak ekrana 96x64 piksel boyutlarında bir bayrak resmi bastırılmıştır.

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.



- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan "Library Manager" sekmesinden "PWM" kütüphanesini seçiniz.
- 9- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Pişano anahtarlardan sol tarafta bulunan OLED anahtarını açarak, bu modülün GND bağlantısını sağlayınız.
- 12-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- LM35 sıcaklık sensöründen sıcaklık verisini okuyarak bir dikdörtgen içerisinde Oled ekranda gösteriniz.
- 2- Belirlediğiniz bir logoyu "LCD Assistant" programında dönüştürerek ekranda gösteriniz.

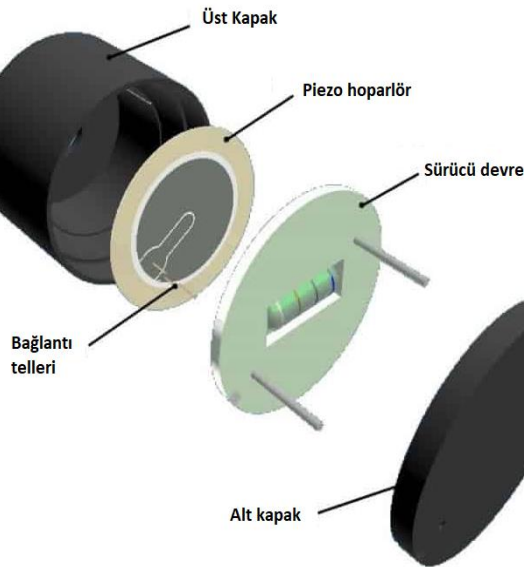
Uygulama 11: Buzzer Uygulaması

Elektronik cihaz uygulamalarında sıklıkla sesli uyarılara yer verilmektedir. Sesli uyarıları sağlamanın en pratik yolu bir buzzer kullanmaktır. Buzzerlar farklı boyutlarda ve tiplerde üretilmektedirler. Şekil 11.1’de buzzer çeşitleri görülmektedir.



Şekil 11.1: Buzzer çeşitleri.

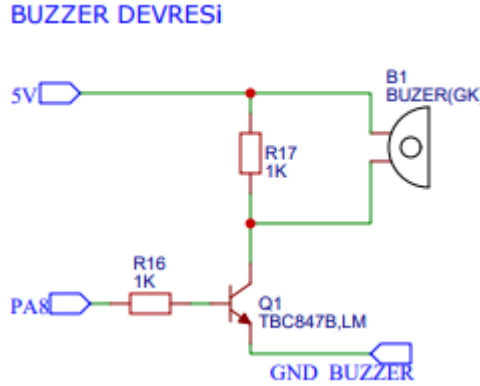
Buzzer yapısında bir adet sürücü devre ve bir adet piezo hoparlör barındırır. Sürücü devre uçlarına uygulanan gerilim ile piezo hoparlörü rezonans frekansında titreştirecek bir sinyal oluşturur. Piezo buzzer titreşerek ses sinyalini oluşturur.



Şekil 11.2: Buzzerın yapısı.

ARMapp-18 uygulama setinde kullanılan buzzer 5V ile çalışan 12mm boyutunda bir buzzerdir.

Bu uygulamada buzzer uçlarına belli frekanslarda voltaj uygulanarak oluşturacağı ses sinyalinin frekansı değiştirilecek ve farklı tonlarda sesler elde edilecektir. MikroC pro for ARM derleyicisinde ton üretmek için "Sound Library" isminde bir kütüphane bulunmaktadır. Uygulamada bu kütüphane kullanılarak İstiklal Marşı'na ait notalar çaldırılacaktır. Şekil 11.3'te ARMapp-18 Uygulama setinde bulunan buzzer modülünün devre şeması görülmektedir.



Şekil 11.3: ARMapp-18 Uygulama setinde bulunan Buzzer modülüne ait devre şeması.

Uygulama Kodları:

```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // Buzzer ile İstiklal Marşı Uygulaması //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11 // Notalar tanımlanıyor.
12 int a, ad, b, c, cd, d, dd, e, f, fd, g, gd, a2, ad2, b_2, c2, cd2, d_2, dd2, e2, f2, fd2, g2, gd2, a3;
13
14 void main()
15 {
16     a=440*4; // buradaki çarpım değeri değiştirilerek
17     ad=466*4; // notaların frekansları değiştirilebilir.
18     b=494*4; // nota frekansları Hz cinsinden belirleniyor.
19     c=523*4;
20     cd=554*4;
21     d=587*4;
22     dd=622*4;
23     e=659*4;
24     f = 698*4;
25     fd = 740*4;
26     g = 784*4;
27     gd = 830*4;
28     a2 = 880*4;
29     ad2 = 932*4;
30     b_2 = 988*4;
31     c2 = 1046*4;
32     cd2 = 1108*4;
33     d_2 = 1174*4;
34     dd2 = 1244*4;
35     e2 = 1318*4;
36     f2 = 1396*4;



```

```

37  td2 = 1480*4;
38  g2 = 1568*4;
39  gd2 = 1660*4;
40  a3 = 1760*4;
41
42  Sound_Init(&GPIOA_ODR, 8);           // PA8 pini ses çıkışı olarak ayarlanı
43
44  while(1)                             // Sonsuz döngü.
45  {
46  //Korkma Sönmez Bu Şafak
47      Sound_Play(c, 800);               // c notası 800 ms boyunca oynatılıyor.
48      Sound_Play(f, 800);
49      Sound_Play(g, 800);
50      Sound_Play(gd, 800);
51      Sound_Play(e, 400);
52      Sound_Play(g, 200);
53      Sound_Play(f, 1600);
54      delay_ms(300);
55
56  //Larda Yüzden Al Sancak
57      Sound_Play(f, 800);
58      Sound_Play(ad2, 800);
59      Sound_Play(c2, 800);
60      Sound_Play(cd2, 800);
61      Sound_Play(a2, 400);
62      Sound_Play(c2, 200);
63      Sound_Play(ad2, 1600);
64
65  //Sönmeden Yurdumun Üstünde Tüten En Son Ocak O Be
66      Sound_Play(c2, 200);
67      Sound_Play(ad2, 200);
68      Sound_Play(c2, 200);
69      Sound_Play(g, 400);
70      delay_ms(100);
71      Sound_Play(g, 400);
72      Sound_Play(ad, 200);
73      Sound_Play(gd, 400);
74      Sound_Play(e, 200);
75      Sound_Play(f, 400);
76      Sound_Play(g, 200);
77      Sound_Play(gd,400);
78      Sound_Play(ad, 200);
79      Sound_Play(c2, 400);
80      Sound_Play( cd2, 200);
81      Sound_Play( dd2, 400);
82      Sound_Play( f2, 200);
83      Sound_Play( dd2, 400);
84
85  //Nim Milletimin
86      Sound_Play( dd, 200);
87      Sound_Play( d, 200);
88      Sound_Play( dd, 200);
89      Sound_Play( c2, 800);
90      Sound_Play( ad, 800);
91      Sound_Play( gd, 1600);

```


İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “Sound” kütüphanesini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Piyano anahtarlardan sol taraftakinde bulunan BUZZER anahtarını açarak, bu modülün GND bağlantısını sağlayınız.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- Programın çalışmasını gözlemleyiniz.

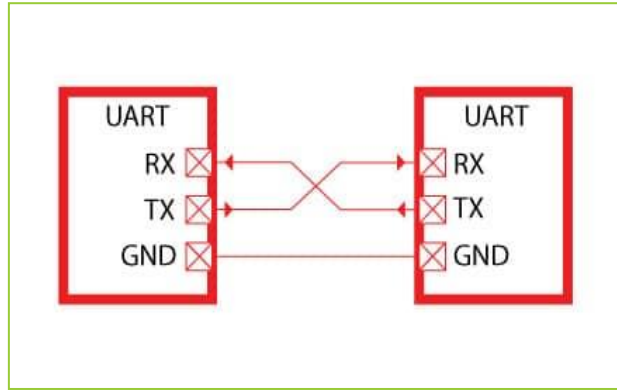
Sorular:

- 1- Kare Sinyal Modülündeki potansiyometreyi kullanarak ses frekansını değiştiren uygulamayı yapınız.
- 2- Menü butonlarını kullanarak basit bir piyano uygulaması gerçekleştiriniz.

Uygulama 12: Seri İletişim Uygulaması

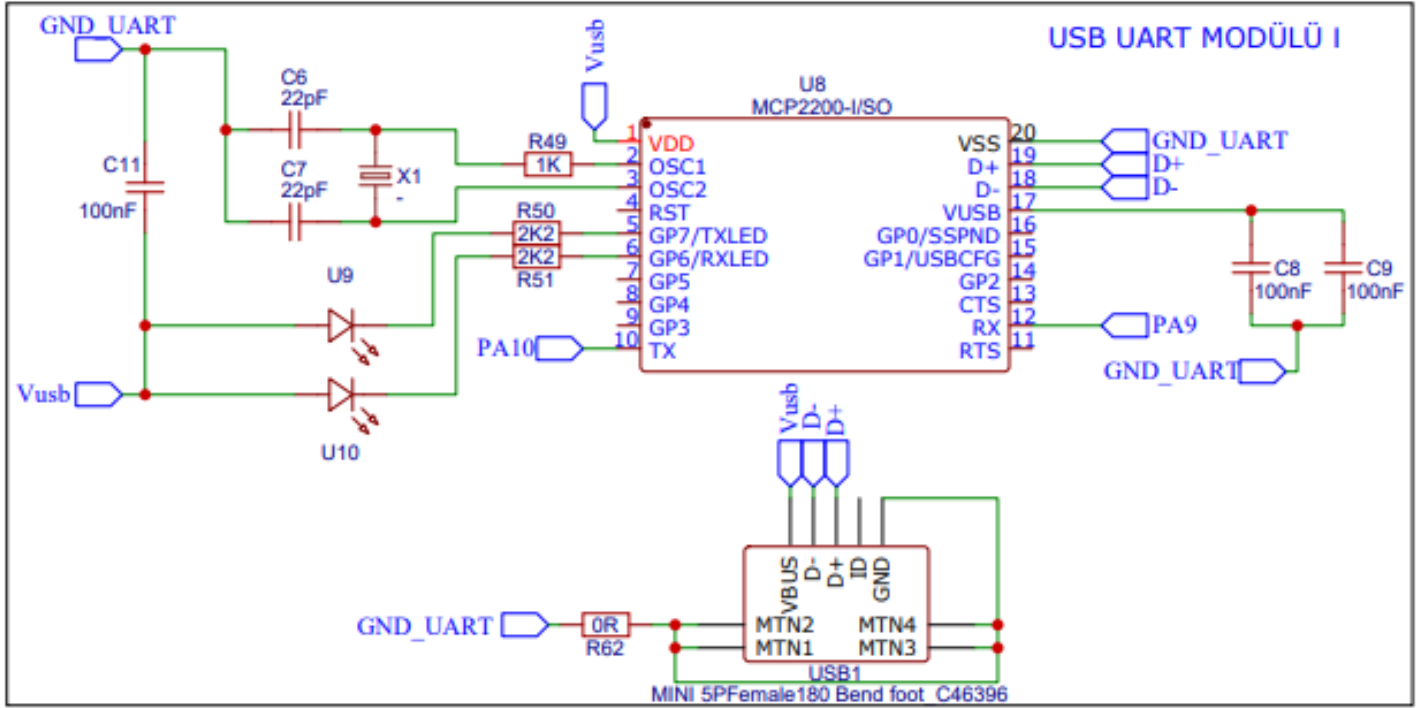
Seri iletişim verilerin bir hat üzerinden bit dizileri halinde sırasıyla gönderme işlemiyle gerçekleştirilen iletişim türlerine genel olarak verilen isimdir. CAN, ETHERNET, I2C, SPI, RS232, USB, 1-Wire gibi seri iletişim protokolleri bulunmaktadır.

Bu uygulamada STM32F407VG mikrodenetleyicisinde bulunan UART (universal asynchronous receiver-transmitter – evrensel asenkron alıcı-verici) donanımı kullanılarak bir seri iletişim uygulaması gerçekleştirilecektir. Bilgisayar üzerinden ARMapp-18 uygulama kartına gelen veriler olduğu gibi bilgisayara geri gönderilecek ve karakter LCD’de gösterilecektir. UART ile seri iletişim 3 kablo ile mümkündür. Bunlardan iki adedi haberleşecek iki birim arasındaki TX ve RX hatlarıdır ve bu hatlar birbirlerine çapraz bağlanır (TX->Rx). 3. Hat ise haberleşecek iki sistem arasındaki voltaj dengesini sağlamak adına yapılan iki tarafın GND hatlarının birbirlerine olan bağlantısıdır.



Şekil 12.1: UART İletişim.

ARMapp-18 uygulama setinde bilgisayar ve STM32F407VG mikrodenetleyicisi arasındaki UART iletişimi sağlamak için MCP2200 isiminde bir USB-UART dönüştürücü çip kullanılmıştır. Bu çip USB ile UART arasında veri dönüşümü yaparak bilgisayarın USB portlarının UART ile seri iletişim amaçlı kullanılmasını sağlar. Şekil 12.2’ de ARMapp-18 uygulama setinde kullanılan USB-UART modülüne ait devre şeması görülmektedir.



Şekil 12.2: ARMap-18 Uygulama setinde bulunan USB-UART modülüne ait devre şeması.

STM32F407 mikrodenetleyicisinde bulunan UART modüllerine ait pinler Şekil 12.3'te, tabloda görülebilmektedir.

U(S)ARTx	Pins pack 1		Pins pack 2		Pins pack 3	
	TX	RX	TX	RX	TX	RX
USART1	PA9	PA10	PB6	PB7		
USART2	PA2	PA3	PD5	PD6		
USART3	PB10	PB11	PC10	PC11	PD8	PD9
UART4	PA0	PA1	PC10	PC11		
UART5	PC12	PD2				
USART6	PC6	PC7				

Şekil 12.3: STM32F407 UART modüllerine ait pinler.

Bu uygulamada UART1 modülü kullanılmıştır. MikroC pro for ARM derleyicisinde bulunan UART kütüphanesinde UART1 modülü seçildiğinde varsayılan olarak PA9 ve PA10 pinleri kullanılmaktadır.


Uygulama Kodları:


```

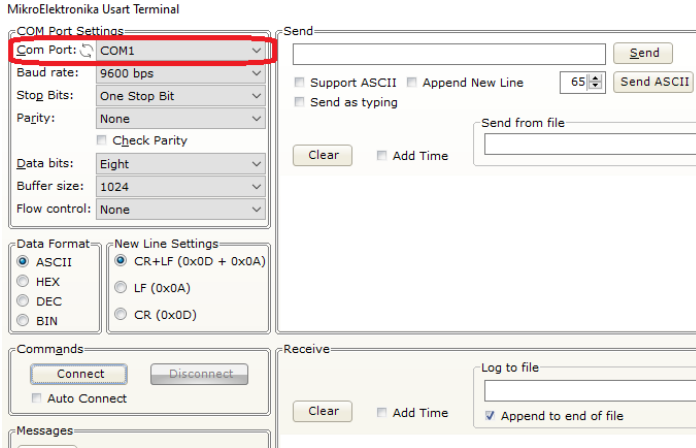
1  //////////////////////////////////////
2  //*****//
3  //      UART iletişim Uygulaması      //
4  //      MikroC v6.2 - STM32F407VG      //
5  //      ARMapp-18 Deney Seti için yazılmıştır //
6  //*****//
7  //      http://elektrovadi.com        //
8  //      http://mikrodunya.wordpress.com //
9  //////////////////////////////////////
10
11 sbit LCD_RS at GPIOD_ODR.B2; // RS pini PD2 pinine bağlı.
12 sbit LCD_EN at GPIOD_ODR.B3; // EN pini PD3 pinine bağlı.
13 sbit LCD_D4 at GPIOD_ODR.B4; // D4 pini PD4 pinine bağlı.
14 sbit LCD_D5 at GPIOD_ODR.B5; // D5 pini PD5 pinine bağlı.
15 sbit LCD_D6 at GPIOD_ODR.B6; // D6 pini PD6 pinine bağlı.
16 sbit LCD_D7 at GPIOD_ODR.B7; // D7 pini PD7 pinine bağlı.
17
18 void main()
19 {
20     char x=0,gelen;           // char tipinde x ve gelen isiminde değişkenler tanımlanıyor.
21     Lcd_Init();              // LCD kurulumu yapılıyor.
22     Lcd_Cmd(_LCD_CLEAR);     // LCD temizleniyor.
23     Lcd_Cmd(_LCD_CURSOR_OFF); // İmleç kapatılıyor.
24     UART1_Init(9600);        // Seri iletişim PA9 ve PA10 pinleri kullanılarak olarak kuruluyor.
25
26     while(1)                 // Sonsuz döngü
27     {
28         if(UART1_Data_Ready()) // Eğer UART üzerinden veri geldiyse.
29         {
30             gelen=UART1_Read(); // UART verisini oku ve gelen isimli değişkene aktar.
31             UART1_Write(gelen); // gelen isimli veriyi UART üzerinden geri gönder.
32             Lcd_Chr_CP(gelen);  // LCD'de imlecin olduğu pozisyona gelen verisini karakter olarak bas.
33         }
34     }
35 }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.
- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan "Library" sekmesinden "UART" kütüphanesini seçiniz.
- 9- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.

- 11- Piyano anahtarlarından sol taraftakinde bulunan K.LCD ve sağ tarafta bulunan USB-UART anahtarlarını açarak, bu modüllerin GND bağlantısını sağlayınız.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- ARMapp-18 setindeki USB-UART modülü ve PC arasına mini USB kablosu takınız.
- 14- “Tools” sekmesinden “Usart Terminal” seçeneğini tıklayarak terminal aracını çalıştırınız. Çıkan ekranda bağlantı portunu ve bağlantı hızını (uygulamada 9600 bps.) seçerek “Connect” butonuna tıklayınız. Send kısmındaki text kutusuna göndermek istediğiniz mesajı yazıp send butonuna tıklayınız.



- 15- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- UART iletişimi kullanarak ‘y’ karakteri gönderildiğinde power ledi yakan , ‘s’ karakteri gönderildiğinde power ledi söndüren programı yazınız.

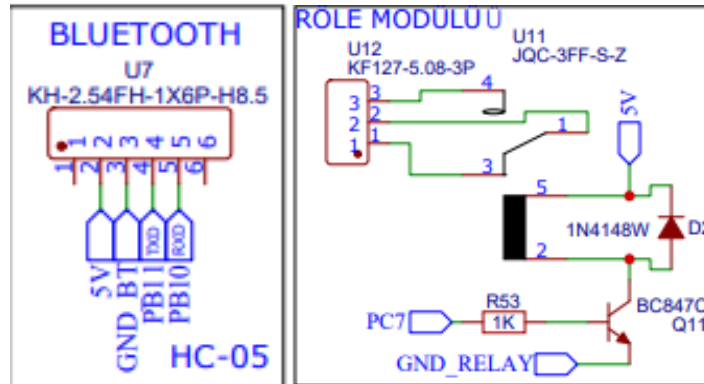
Uygulama 13: Bluetooth ile Röle Kontrol Uygulaması

Bluetooth bilgisayar, cep telefonu, tablet v.b cihazlarının birbirlerine kablosuz olarak bağlanabilmeleri amacıyla kullanılan RF (radyo frekans) teknolojisine verilen isimdir. Günümüzde mikrodenetleyiciler ile UART üzerinden bağlanan bluetooth modülleri birçok firma tarafından üretilmektedir. ARMapp-18 uygulama setinde HC-05 isimli bluetooth modülü kullanılmaktadır. Bu modül şekil 14.1’de görülebilir.



Şekil 13.1: HC-05 Bluetooth Modülü.

HC-05 Bluetooth modülü ile mikrodenetleyici bağlantısı yapılırken temel olarak 4 pin kullanılır. Bunlar VCC, GND, RX ve TX pinleridir. Bu dört pin kullanılarak mikrodenetleyici ile bağlantısı sağlanan HC-05 bluetooth modülü enerji verildiği anda diğer bluetooth donanımına sahip cihazlar ile eşleştirilmeye ve iletişim kurmaya hazırdır. Şekil 13.2’ de ARMapp-18 uygulama setinde bulunan bluetooth modülüne ait devre şeması görülmektedir.



Şekil 13.2: ARMapp-18 Uygulama Setinde Bulunan Bluetooth ve Röle Modüllerine Ait Devre Şeması.

Bu uygulamada STM32F407VG mikrodenetleyicisinin PB10 ve PB11 pinleri kullanılmıştır. Bu pinler STM32F407’nin UART modülünde donanımsal olarak desteklenmelerine rağmen bu uygulamada yazılımsal UART Kütüphanesi kullanılarak bu kütüphaneye de değinilmek istenilmiştir.

HC-05 Bluetooth modülüne bir terminal programı (android telefonlarda bluetooth terminal isimli program, bilgisayarlarda ise putty isimli program veya MikroC’de bulunan USART Terminal kullanılabilir) üzerinden bağlanılıp veri akışı sağlanabilir. Bir bilgisayarlardan bluetooth modül ile iletişim kurmak için bir

com port kullanılır. Bluetooth özelliklerinden hangi com portun kullanıldığına bakılıp terminal programı üzerinden iletişim kurulabilir.

Uygulama Kodları:



```

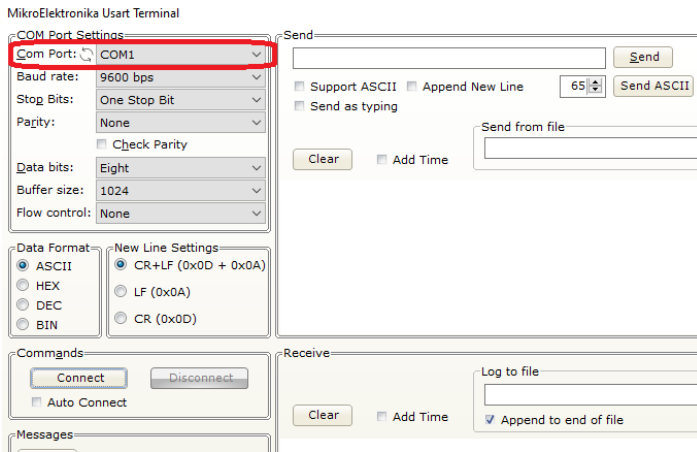
1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // Bluetooth ile Röle Kontrol Uygulaması //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11 #define relay GPIOC_ODR.B7 // PC7 pini 'relay' olarak tanımlanıyor.
12 const char *ron="Relay is ON\r\n"; // 'ron' isminde karakter dizisi tanımlanıyor ve içerisine mesaj kaydediliyor.
13 const char *roff="Relay is OFF\r\n"; // 'roff' isminde karakter dizisi tanımlanıyor ve içerisine mesaj kaydediliyor.
14
15 void main() // ana program bloğu
16 {
17     char incomingByte, error=1, i=0; // programda kullanılacak bir takım değişkenler tanımlanıyor.
18     char incomingArray[3]; // gelen veriyi tutacak 3 byte'lık dizi tanımlanıyor.
19     Soft_UART_Init(&GPIOB_ODR, 11, 10, 9600, 0); // B portunun 11. ve 10. pinleri yazılımsal uart pinleri olarak tanımlanıyor.
20     GPIO_Digital_Output(&GPIOC_BASE, GPIO_PINMASK_7); // PC7 pini dijital çıkış olarak tanımlanıyor.
21     relay=0; // Röle pini 0'a çekilerek röle açılıyor.
22
23     while(1) // Sonsuz döngü
24     {
25         do // Yazılımsal UART üzerinden veri gelene kadar bekleyen kısım.
26         | incomingByte = Soft_UART_Read(&error); // Eğer veri geldiyse veriyi incomingByte değişkenine aktar.
27         while(error);
28
29         incomingArray[i]=incomingByte; // veri geldikçe incomingArray dizisinin elemanlarına sırasıyla verileri kaydet.
30         if(incomingArray[0]!='R')i++; // eğer ilk gelen veri 'R' ise i değişkenini 1 arttır.
31
32         if(i==3) // Eğer 3 adet veri geldiyse.
33         { // (Mesajları senkronize edebilmek için.)
34             i=0; // veri sayacını 0 la.
35             if( incomingArray[0]=='R' // Eğer gelen veriler sırasıyla RON ise
36                 && incomingArray[1]=='O'
37                 && incomingArray[2]=='N')
38             {
39                 char j=0,length;
40                 relay=1; // relay pinini 1 yaparak röleyi çektir.
41                 length=strlen(ron); // 'ron' dizisinin eleman sayısını al ve length değişkenine aktar.
42                 for(j=0;j<length;j++)Soft_UART_Write(ron[j]); // bluetooth üzerinden ron dizisinin elemanlarını sırasıyla gönder.
43             }
44
45             if( incomingArray[0]=='R' // Eğer gelen veriler sırasıyla ROF ise
46                 && incomingArray[1]=='O'
47                 && incomingArray[2]=='F')
48             {
49                 char j=0,length;
50                 relay=0; // relay pinini 0 yaparak röleyi bıraktır.
51                 length=strlen(roff); // 'roff' dizisinin eleman sayısını al ve length değişkenine aktar.
52                 for(j=0;j<length;j++)Soft_UART_Write(roff[j]); // bluetooth üzerinden roff dizisinin elemanlarını sırasıyla gönder.
53             }
54         }
55     }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.

- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan "Library" sekmesinden "C_String" ve "Software UART" kütüphanelerini seçiniz.
- 9- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Pi-yano anahtarlardan sol taraftakinde bulunan RÖLE ve sağ tarafta bulunan BLUETOOTH anahtarlarını açarak, bu modüllerin GND bağlantısını sağlayınız.
- 12-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- HC-05 Bluetooth modül ile bilgisayar veya cep telefonu ile "1234" şifresini kullanarak eşleşiniz.
- 14- Eğer bilgisayar ile bluetooth iletişim kuracaksınız "Tools" sekmesinden "Usart Terminal" seçeneğini tıklayarak terminal aracını çalıştırınız. Çıkan ekranda bluetooth'a ait bağlantı portunu ve istediğiniz bağlantı hızını seçerek "Connect" butonuna tıklayınız. Send kısmındaki text kutusuna RON veya ROF mesajını yazıp send butonuna tıklayınız.



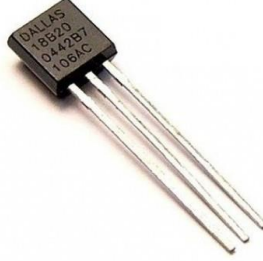
- 15- Eğer telefon ile iletişim sağlayacaksınız kullanacağınız terminal programından HC-05 bluetooth modül ile eşleşerek RON veya ROF mesajı gönderiniz.
- 16- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- Bluetooth üzerinden göndereceğiniz mesajlar ile ARMapp-18 uygulama setindeki 8 adet LED'i ayrı ayrı kontrol ediniz.

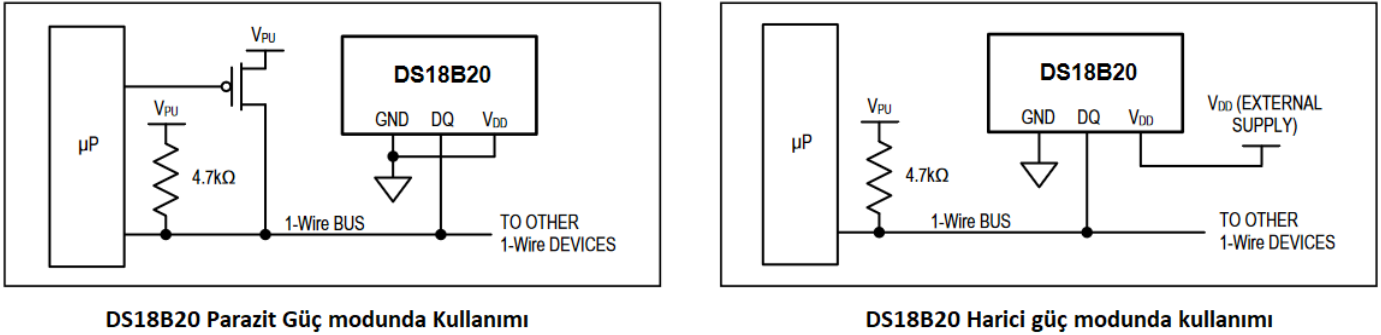
Uygulama 14: DS18B20 1-Wire Sıcaklık Sensörü Uygulaması

DS18B20 9 bit- 12 bit aralığında ayarlanabilir çıkış verebilen , 55°C / +125°C aralığında ölçüm yapabilen ve tek hat üzerinden (1-Wire) veri iletişimi kurabilen bir sıcaklık sensörüdür.



Şekil 14.1: DS18B20 Sıcaklık sensörü.

Bu sensör Vcc, Data ve GND olmak üzere 3 pine sahip olmakla birlikte parazit güç özelliği sayesinde sadece data ve GND pinleri kullanılarak da çalıştırılabilmektedir. 3 pinli çalışma ve 2 pinli parazitik çalışma arasında data pinine uygulanacak sinyaller arasında farklılıklar mevcuttur.

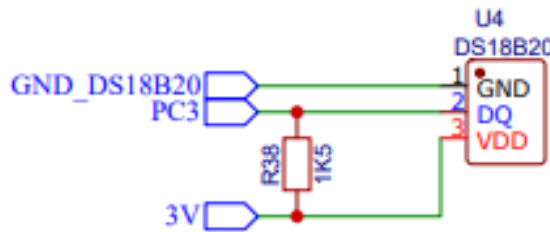


DS18B20 Parazit Güç modunda Kullanımı

DS18B20 Harici güç modunda kullanımı

Şekil 14.1: DS18B20 güç modları ve mikrodenetleyici ile bağlantıları.

ARMapp-18 uygulama setinde DS18B20 harici güç modunda kullanılmaktadır. Şekil 14.2’de ARMapp-18 uygulama setindeki DS18B20 modülünün şeması görülmektedir.



Şekil 14.2: ARMapp-18 Uygulama setinde bulunan DS18B20 modülüne ait devre şeması.

Bu uygulamada DS18B20 sıcaklık sensöründen okunan sıcaklık verisi ARMapp-18 uygulama setindeki USB-UART modülü kullanılarak bilgisayara gönderilmektedir. Bu veriler MikroC pro for ARM derleyicisinde tools sekmesi altında bulunan “USART Terminal” programı aracılığı ile görüntülenebilir.

Uygulama Kodları:



```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //OneWire kütüphanesi ile DS18B20 kullanımı//
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11 #define buzzer GPIOA_ODR.B8 // PA8 pini buzzer olarak tanımlanıyor.
12 int tempmsb, tempmsb; // tempmsb ve tempmsb olarak iki adet değişken tanımlanıyor.
13 float sıcaklik; // ondalıklı sayı tipinde sıcaklik isminde bir değişken tanımlanıyor.
14 char txt[15]; // sıcaklik değişkenini stringe çevirirken kullanılacak karakter dizisi.
15 char dizi[8]; // 8 elemanlı bir dizi tanımlanıyor.
16
17 void main() // ana program bloğu
18 {
19 GPIO_Digital_Output(&GPIOA_BASE, _GPIO_PINMASK_8); // PA8 bini (buzzer) dijital çıkış olarak tanımlanıyor.
20 buzzer=0; // buzzer susturuluyor.
21 UART1_Init(9600); // UART1 kuruluyor. ( PA9 ve PA10 )
22
23 Ow_Reset(&GPIOC_ODR, 3); // DS18B20 'ye PC3 pininden reset sinyali gönderiliyor.
24 Ow_Write(&GPIOC_ODR, 3, 0xCC); // SKIP_ROM komutu gönderiliyor.
25 Ow_Write(&GPIOC_ODR, 3, 0x4E); // SCRATCHPAD'e yaz komutu gönderiliyor.
26 Ow_Write(&GPIOC_ODR, 3, 0x00); // TH=0
27 Ow_Write(&GPIOC_ODR, 3, 0x00); // TL=0
28 Ow_Write(&GPIOC_ODR, 3, 0x7F); // Config=0x7F ( Çözünürlük 12 bit olarak ayarlanıyor)
29
30 Ow_Reset(&GPIOC_ODR, 3); // DS18B20 'ye PC3 pininden reset sinyali gönderiliyor.
31 Ow_Write(&GPIOC_ODR, 3, 0xCC); // SKIP_ROM komutu gönderiliyor.
32 Ow_Write(&GPIOC_ODR, 3, 0x48); // SCRATCHPAD üzerindeki veriler EEPROM'a yazılıyor.
33 GPIOC_ODR.B3=1; // Veriler EEPROM'a yazılırken data hattı 100 ms
34 delay_ms(100); // boyunca HIGH konumda tutuluyor.
35
36 while(1) // Sonsuz döngü.
37 {
38 Ow_Reset(&GPIOC_ODR, 3); // DS18B20 'ye PC3 pininden reset sinyali gönderiliyor.
39 Ow_Write(&GPIOC_ODR, 3, 0xCC); // SKIP_ROM Komutu
40 Ow_Write(&GPIOC_ODR, 3, 0x44); // CONVERT_T Komutu (Sıcaklık dönüşümü yapılıyor.)
41
42 Delay_ms(800); //Dönüşüm için bekleniyor.
43
44 Ow_Reset(&GPIOC_ODR, 3);
45 Ow_Write(&GPIOC_ODR, 3, 0xCC); // SKIP_ROM Komutu
46 Ow_Write(&GPIOC_ODR, 3, 0xBE); // READ_SCRATCHPAD Komutu
47
48 tempmsb = Ow_Read(&GPIOC_ODR, 3); // Sıcaklık bilgisinin LSB kısmı okunuyor.
49 tempmsb = Ow_Read(&GPIOC_ODR, 3); // Sıcaklık bilgisinin MSB kısmı okunuyor.
50
51 sıcaklik=((tempmsb<<8)+tempmsb)*0.0625; // LSB ve MSB birleştirilerek okunan veri sıcaklık olarak hesaplanıyor.
52 floattostr(sıcaklik,txt); // Sıcaklık değeri String veri tipine dönüştürülerek txt dizisine aktarılıyor.
53 UART1_Write_Text(txt); // txt dizisi UART üzerinden gönderiliyor.
54 UART1_Write_Text(" °C\r\n"); // verinin sonun Celsius derece bilgisi ve alt satıra göndermek için
55 // carriage return verisi gönderiliyor.
56 if(sıcaklik>50)buzzer=1; // Sıcaklık 50°C üzerindeyse buzzer ötsün.
57 else buzzer=0; // değilse buzzer sussun.
58 }
59 }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.

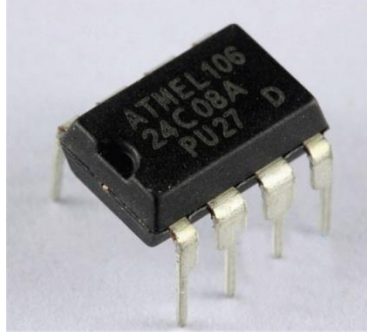
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “Conversions”, “C_String”, “One_Wire” ve “UART” kütüphanelerini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Pişano anahtarlardan sağ taraftakinde bulunan USB-UART ve DS18B20 anahtarlarını açarak, bu modüllerin GND bağlantılarını sağlayınız.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- ARMapp-18 setindeki USB-UART modülü ve PC arasına mini USB kablosu takınız.
- 14- MikroC pro for ARM derleyicisinde bulunan “Tools” sekmesindeki “USART Terminal” programını çalıştırarak 9600 baud hızında ARMapp-18 uygulama setine bağlanınız.
- 15- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- DS18B20 sıcaklık sensörünün dahili düşük sıcaklık ve yüksek sıcaklık alarmlarını +10°C ve +40°C değerlerine ayarlayarak bu değerlere sıcaklık geldiğinde Buzzer öttüren programı yazınız.

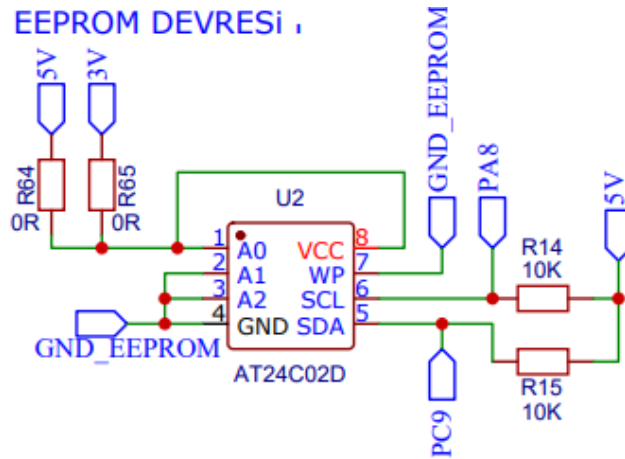
Uygulama 15: Harici EEPROM Uygulaması.

EEPROM' lar (**Electronically Erasable Programmable Read-Only Memory**) küçük boyutlu verilerin kalıcı olarak saklanabilmesi için kullanılan elektronik çiplerdir. EEPROM'lar enerjileri kesilse bile üzerlerine yazılan verileri korurlar. Bu sebeple kalıcı hafızanın gerekli olduğu projelerde sıklıkla kullanılırlar. Modern bilgisayarlarda şimdiye dek kullanılan CMOS kalıcı BIOS bellek teknolojisi (CMOS nonvolatile BIOS memory) ile yer değiştirmiştir. Örneğin; kişisel bilgisayarlarda bu yongalar BIOS kodlarını ve sistem ayarlarını saklamak için kullanılır. Günümüzde piyasada 128 bit – 2Mbit aralığında veri saklayabilen EEPROM'lar rahatlıkla bulunabilmektedir.



Şekil 15.1: ARMapp-18 Uygulama setinde bulunan EEPROM modülüne ait devre şeması.

ARMapp-18 uygulama setinde Microchip 24C08A isimli EEPROM kullanılmaktadır. Bu EEPROM en az 1.000.000 kez veri yazma, 100 yıl veri koruma ve >3000V ESD koruması sunmaktadır. 24C08A EEPROM'unda A2 pini donanımsal adresini belirlemek için kullanılır. Şekil 15.2'de ARMapp-18 uygulama setindeki EEPROM modülüne ait devre şeması görülebilir. 24C08A I2c iletişim protokolü ile haberleştirilebilir.



Şekil 15.2: 24C08A isimli EEPROM çipinin adreslenmesi.

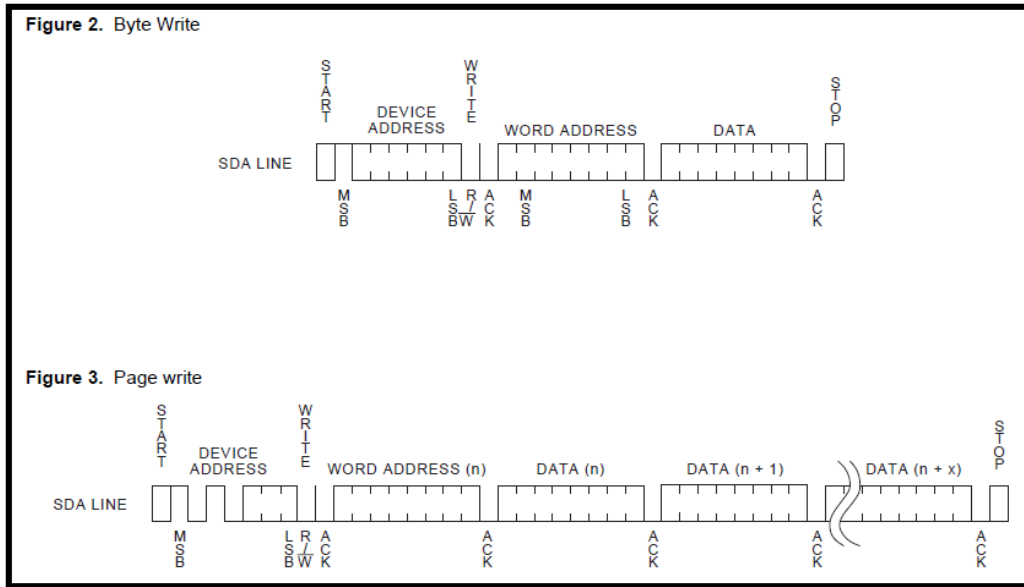
A1 ve A0 pinleri kullanılmaz (N.C.). ARMapp-18 uygulama setinde A2 pini GND'ye bağlanmıştır böylece EEPROM adresi şekil 15.2' deki gibi oluşmuştur.

1	0	1	0	A ₂	P1	P0	R/W
---	---	---	---	----------------	----	----	-----

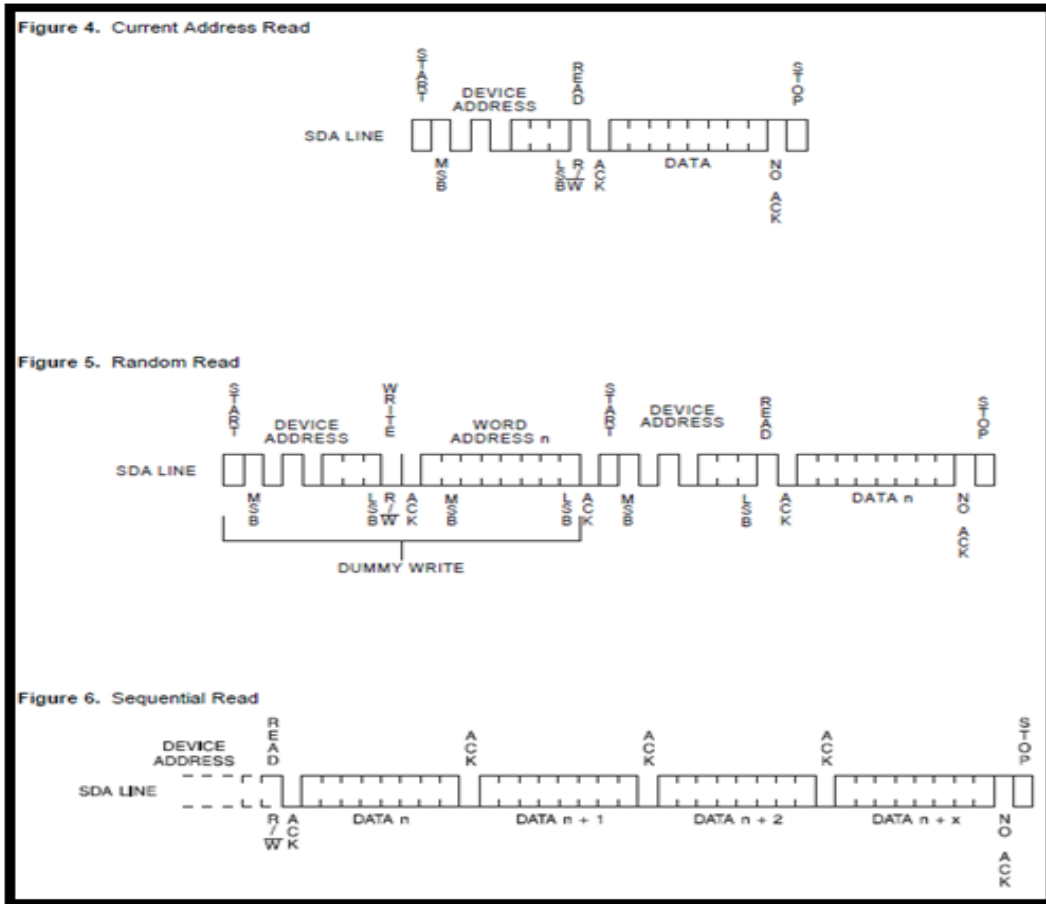
Şekil 15.3: 24C08A isimli EEPROM çipinin adreslenmesi.

P1 ve P0 isimli bitlere page adresleri gelmektedir. 24c08A' da her biri 4 byte veri saklayabilen 256 page bulunmaktadır. Toplamda 1024 byte veri saklayabilir. Byte adresleri 0'dan başlayıp 1023'e kadar 10 bit olarak uzanmaktadır. Bu 10 bitlik adres verisinin üst 2 biti P1 ve P0 bitlerine yazılmalıdır.

Şekil 15.3' te 24C08A EEPROM'a veri yazma modları görülmektedir. Üstte tek byte veri yazma diyagramı, altta ise page write diyagramı görülmektedir. Page write yapılacaksa 24C08A EEPROM'unda bir defada en fazla 16 byte veri yazılabileceği unutulmamalıdır.



Şekil 15.4: 24C08A isimli EEPROM veri yazma diyagramları.



Şekil 15.5: 24C08A isimli EEPROM veri okuma diyagramları.

Şekil 15.4' te 24C08A 'dan veri okuma diyagramları görülmektedir. En üstteki diyagramda adres sayacının o anki konumundan veri okuma, ortadaki belirli bir adresten veri okuma, en altta ise art arda veri okuma diyagramı görülmektedir.

Art arda veri okunacak ise ilk verinin adresi bilgisi girilir, daha sonraki verilerin adresleri adres sayacı otomatik olarak birer artarak okunur.

Bu uygulamada öncelikle 16 byte uzunluğunda bir veri EEPROM'a 0 adresinden başlanarak yazılmıştır. Yazım işlemi "page write" şeklinde, veri okuma işlemi ise "sequential read" şeklinde yapılmış, okunan veriler UART üzerinden bilgisayara gönderilmiştir.

Uygulama Kodları:



```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 // Harici EEPROM yazma ve okuma uygulması //
4 // MikroC v6.2 - STM32F407VG //
5 // ARMapp-18 Deney Seti için yazılmıştır //
6 //*****//
7 // http://elektrovadi.com //
8 // http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10
11 sbit Soft_I2C_Scl_Output at GPIOA_ODR.B8; // Yazılımsal I2C bağlantıları tanımlanıyor.
12 sbit Soft_I2C_Scl_Input at GPIOA_IDR.B8;
13 sbit Soft_I2C_Sda_Output at GPIOC_ODR.B9;
14 sbit Soft_I2C_Sda_Input at GPIOC_IDR.B9;
15
16 char dizi[]="ARMapp-18 MikroC"; // 13 karakterlik dizi isminde bir dizi tanımlanıyor
17 // ve içerisine program boyunca kullanılacak mesaj kaydediliyor.
18 char tmp[16]; // tmp isimli tampon amaçlı boş dizi tanımlanıyor.
19 char i; // char tipinde 'i' isimli değişken tanımlanıyor.
20 void main() // ana program bloğu.
21 {
22 UART1_Init(9600); // UART1 kuruluyor (PA9 TX-PA10 RX)
23 Soft_I2C_Init(); // Yazılımsal I2C kuruluyor.
24 Soft_I2C_Start(); // Yazılımsal I2C başlatılıyor.
25 Soft_I2C_Write(0xA0); // EEPROM adresi 0b10100000=0xA0
26 Soft_I2C_Write(0x00); // Veri yazılmaya başlanacak adres 0x0000'dan başlıyor.
27 for(i=0;i<16;i++)Soft_I2C_Write(dizi[i]); // Sırasıyla 8 byte veri EEPROM'a yazdırılıyor.
28 Soft_I2C_Stop(); // Yazılımsal I2C iletişimi sonlandırılıyor.
29
30 delay_ms(100); // 100 ms bekleniyor.
31
32 Soft_I2C_Start(); // Yazılımsal I2C başlatılıyor.
33 Soft_I2C_Write(0xA0); // EEPROM adresi 0b10100000=0xA0
34 Soft_I2C_Write(0x00); // Veri okunmaya başlanacak adres 0x0000'dan başlıyor.
35 Soft_I2C_Start(); // Yazılımsal I2C tekrar başlatılıyor.
36 Soft_I2C_Write(0xA1); // EEPROM adresi 0b10100001=0xA1 (1010001R)
37 for(i=0;i<16;i++)tmp[i]=Soft_I2C_Read(1); // 8 byte boyunca her seferinde ACK gönderilerek veri okunuyor.
38 tmp[i]=Soft_I2C_Read(0); // son byte okunurken ACK gönderilmiyor. Okunan veriler her seferinde
39 // tmp dizisine kaydediliyor.
40 Soft_I2C_Stop(); // yazılımsal I2C durduruluyor.
41
42 for(i=0;i<16;i++)UART1_Write(tmp[i]); // tmp isimli dizi UART üzerinden gönderiliyor.
43 }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.
- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.

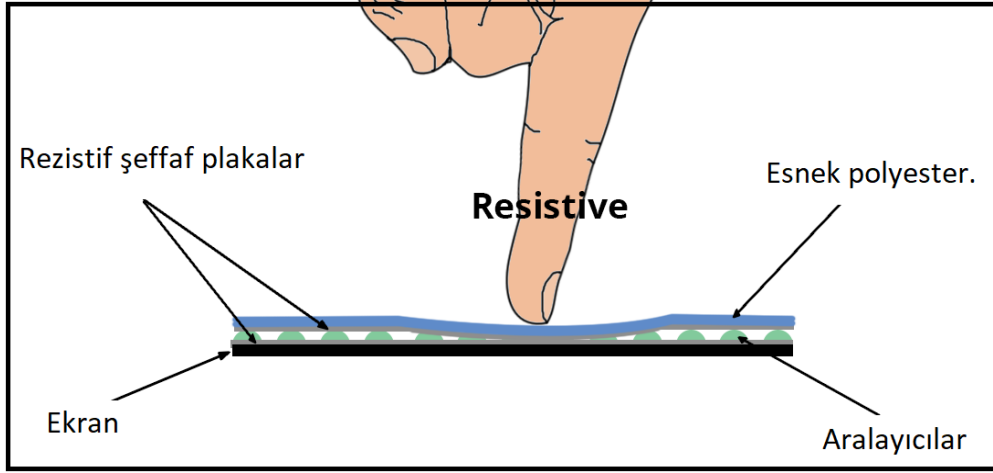
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “Software I2C” ve “UART” kütüphanelerini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Pişano anahtarlardan sağ tarafta bulunan USB-UART ve EEPROM anahtarlarını açarak, bu modüllerin GND bağlantılarını sağlayınız.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- ARMapp-18 setindeki USB-UART modülü ve PC arasına mini USB kablosu takınız.
- 14- MikroC pro for ARM derleyicisinde bulunan “Tools” sekmesindeki “USART Terminal” programını çalıştırarak 9600 baud hızında ARMapp-18 uygulama setine bağlanınız.
- 15- Programın çalışmasını gözlemleyiniz.

Sorular:

- 1- 24C08A’da DS18B20 sıcaklık sensöründen okunan sıcaklık değerlerinin tam sayı kısımlarını alacak şekilde 100 adet veri yazdırınız ve daha sonra bunları UART üzerinden PC’ye gönderiniz.
- 2- Bir program yazarak potansiyometre değerini LCD ekranda gösterip EEPROM’a kaydeden bir program yazınız. Daha sonra bir başka program yazarak EEPROM’a kaydedilen potansiyometre değerini okuyup bu değeri LCD ekranda yazdıracak programı yazınız. Önce ilk programı yükleyip potansiyometre değerini ekranda gösteriniz. Daha sonra ARMapp-18 uygulama setinin enerjisini kesip potansiyometre değerini değiştirerek ikinci programı yükleyiniz. Ekranda gösterilen değerin aynı olup olmadığını karşılaştırınız.

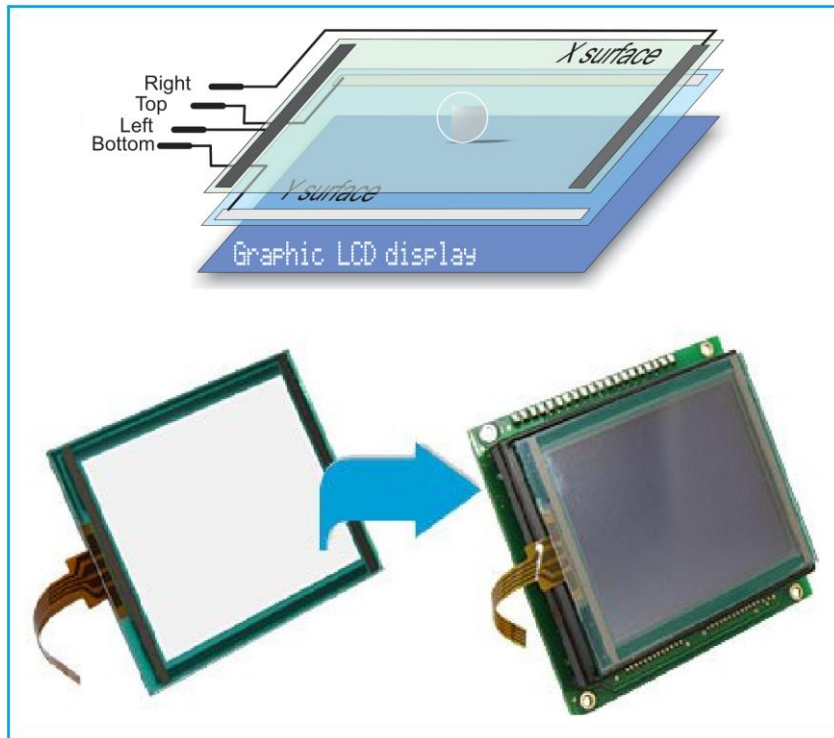
Uygulama 16: Rezistif Dokunmatik Ekran Uygulaması.

Günümüzde insan makine ara yüzünü oluşturan elemanlardan olan tuş ve butonların yerini dokunmatik paneller almakta ve giderek yaygınlaşmaktadır. Dokunmatik paneller hem alandan tasarruf edilmesini sağlamakta hem de daha anlaşılır ve detaylı görseller sunarak cihazların özelliklerine kullanıcının daha kolay erişebilmesini sağlamaktadır. Dokunmatik paneller rezistif ve kapasitif olmak üzere ikiye ayrılırlar. Rezistif paneller iki ayrı rezistif plakanın arasına bir boşluk bırakılarak, panele dokunulduğunda bu iki rezistif plakanın birbirlerine değmesi sonucu uçlarından ölçülecek direnç değerine göre koordinatın belirlenmesi esasına göre çalışır.



Şekil 16.1: Rezistif dokunmatik panellerin yapısı.

ARMapp-18 uygulama setinde kullanılan dokunmatik panel ve grafik LCD şekil 16.2' de görülmektedir. Bu dokunmatik panel Right-Left, Up-Down olmak üzere 4 pine sahiptir.

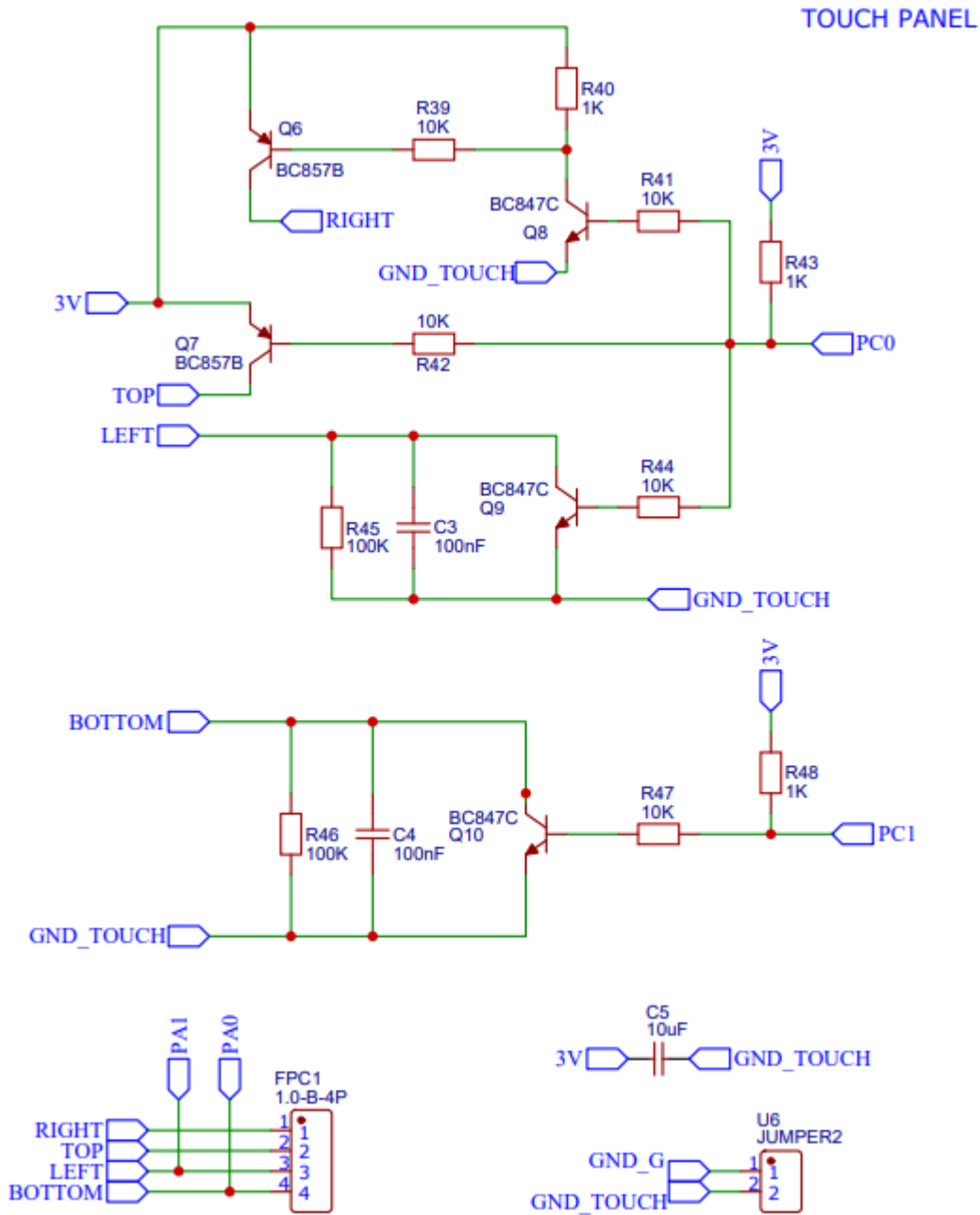


Şekil 16.2: ARMapp-18 Uygulama setinde kullanılan dokunmatik panel.

Rezistif dokunmatik panelin nasıl çalıştığı ile ilgili detaylı bilgilere [buradan](#) ve [buradan](#) ulaşabilirsiniz.

Bu uygulamada ekrana iki adet kare şeklinde buton çizdirilmiş ve bu butonların olduğu koordinatların sınırlarına dokunularak ekranın üst tarafına yazdırılan koordinat değerleri kaydedilmiştir. Eğer tıklanan alan bu koordinat değerleri içerisinde ise işlem yaptırılmıştır. Sol taraftaki butona tıkladığında power led yakılmış, sağ taraftakine tıkladığında söndürülmüştür.

Uygulamaya ait devre şeması şekil 16.3'te görülebilir.



Şekil 16.3: ARMap-18 Uygulama Setinde Kullanılan Dokunmatik Panel Modülüne Ait Devre Şeması.



Uygulama Kodları:

```

1 ////////////////////////////////////////////////////////////////////
2 //*****//
3 //      Dokunmatik Panel uygulması //
4 //      MikroC v6.2 - STM32F407VG   //
5 //      ARMapp-18 DeneY Seti için yazılmıştır //
6 //*****//
7 //      http://elektrovadi.com      //
8 //      http://mikrodunya.wordpress.com //
9 ////////////////////////////////////////////////////////////////////
10 char txt[7];
11
12 unsigned long GLCD_DataPort_Input at GPIOID_IDR; // Grafik LCD data pinleri
13 unsigned long GLCD_DataPort_Output at GPIOID_ODR; // 0.-7. bitlere bağlanıyor.
14
15 sbit GLCD_CS2 at GPIOB_ODR.B0; // Kontrol pinleri
16 sbit GLCD_CS1 at GPIOB_ODR.B1; // bağlantıları
17 sbit GLCD_RS at GPIOB_ODR.B2;
18 sbit GLCD_RW at GPIOB_ODR.B3;
19 sbit GLCD_EN at GPIOB_ODR.B4;
20 sbit GLCD_RST at GPIOB_ODR.B5;
21
22
23 unsigned int x_coord, y_coord;
24
25 unsigned int GetX() { // X eksenindeki analog değeri okunuyor.
26     GPIOC_ODR.B0 = 1; // DRIVEA = 1 Sol ve sağ eksenler sürülüyor, üst eksen açık bırakılıyor.
27     GPIOC_ODR.B1 = 0; // DRIVEB = 0 Alt eksen açık bırakılıyor.
28     Delay_ms(5);
29     return ADC1_read(0); // Alt eksen pininden X eksenine ait değeri okunuyor.
30 }
31 unsigned int GetY() { // Y eksenindeki analog değeri okunuyor.
32     GPIOC_ODR.B0 = 0; // DRIVEA = 0 Sol ve sağ eksenler kapatılıyor. Üst ve alt eksen sürülüyor.
33     GPIOC_ODR.B1 = 1; // DRIVEB = 1
34     Delay_ms(5);
35     return ADC1_read(1); // Sol eksen pininden Y eksenine ait değeri okunuyor.
36 }
37
38
39 void main()
40 {
41     ADC_Set_Input_Channel(_ADC_CHANNEL_0 | _ADC_CHANNEL_1); // PA0 ve PA1 analog giriş olarak belirleniyor.
42     ADC1_Init(); // ADC1 kurulumu yapılıyor.
43
44     GPIO_Digital_Output(&GPIOC_BASE, _GPIO_PINMASK_0 // Dokunmatik panel sürme pinleri çıkış olarak ayarlanıyor.
45                       | _GPIO_PINMASK_1);
46     GPIO_Digital_Output(&GPIOB_BASE, _GPIO_PINMASK_6); // Power Led pini çıkış olarak ayarlanır.
47
48     Glcd_Init(); // GLCD kurulumu yapılıyor.
49     Glcd_Fill(0); // GLCD temizleniyor.
50     Glcd_Set_Font(Font_Glcd_System5x7,5,7,32); // 5x7 boyutundaki font seçiliyor.
51     Glcd_Write_Text("DOKUNMATIK UYGULAMASI",1,7,1); // Ekran tasarımı yapılıyor.
52     Glcd_Rectangle(8,16,60,48,1);
53     Glcd_Rectangle(68,16,120,48,1);
54     Glcd_Box(10,18,58,46,1);
55     Glcd_Box(70,18,118,46,1);
56
57     Glcd_Write_Text("LED",25,3,0);
58     Glcd_Write_Text("ON",27,4,0);
59
60     Glcd_Write_Text("LED",85,3,0);
61     Glcd_Write_Text("OFF",85,4,0);
62
63     Glcd_Write_Text("x=",6,0,1); // x ve y koordinat değeri ekranda gösterilecek.
64     Glcd_Write_Text("y=",64,0,1);
65     delay_ms(500);
66     while (1)
67     {
68         x_coord = GetX(); // x değeri ölçülüyor.
69         y_coord = GetY(); // y değeri ölçülüyor.
70         inttostr(x_coord,txt); // x değeri stringe çevriliyor.
71         Glcd_Write_Text(txt,18,0,1); // x değeri ekranda gösteriliyor.
72         inttostr(y_coord,txt); // y değeri stringe çevriliyor.
73         Glcd_Write_Text(txt,75,0,1); // y değeri ekranda gösteriliyor.
74
75         if ((x_coord >= 350) && (x_coord <= 1850) && (y_coord >= 1140) && (y_coord <= 2900))
76         { // eğer belirlenen koordinatlar arasına tıklanırsa
77             GPIOB_ODR.B6=1; // power led'i yak.
78         }
79
80         if ((x_coord >= 2050) && (x_coord <= 3450) && (y_coord >= 1140) && (y_coord <= 2900))
81         { // eğer belirlenen koordinatlar arasına tıklanırsa
82             GPIOB_ODR.B6=0; // power led'i söndür.
83         }
84     }
85 }

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- "Project" sekmesinden "New Project" seçeneğine tıklayınız.
- 3- Açılan ekrandan "Standart Project" seçilerek "Next" butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve "Open Edit Project window to set Configuration bits" seçeneğini işaretleyerek "Next" butonuna tıklayınız.
- 5- Açılan pencerede "Finish" butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden "HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch" dosyasını seçiniz.
- 7- "OK" butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan "Library Manager" sekmesinden "ADC", "Conversions" ve "GLCD" kütüphanelerini seçiniz.
- 9- Açılan ".c" uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 16- Pişano anahtarlardan sol taraftakinde bulunan K.LCD anahtarını, "POWER LED" anahtarını ve sağ taraftakinde bulunan GLCD anahtarını, açarak, bu modüllerin GND bağlantısını sağlayınız. Grafik LCD modülünün bitişiğinde bulunan anahtarı "ON" konumuna, Karakter LCD modülünün bitişiğinde bulunan anahtarı OFF konumuna getiriniz.
- 11-  "Build and Program" ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 12- Dokunmatik panele tıklayarak power ledin durumunu gözlemleyiniz. (**Dikkat: Power led'e doğrudan uzun süre bakmak gözlerde kalıcı hasara sebep olabilir.**)

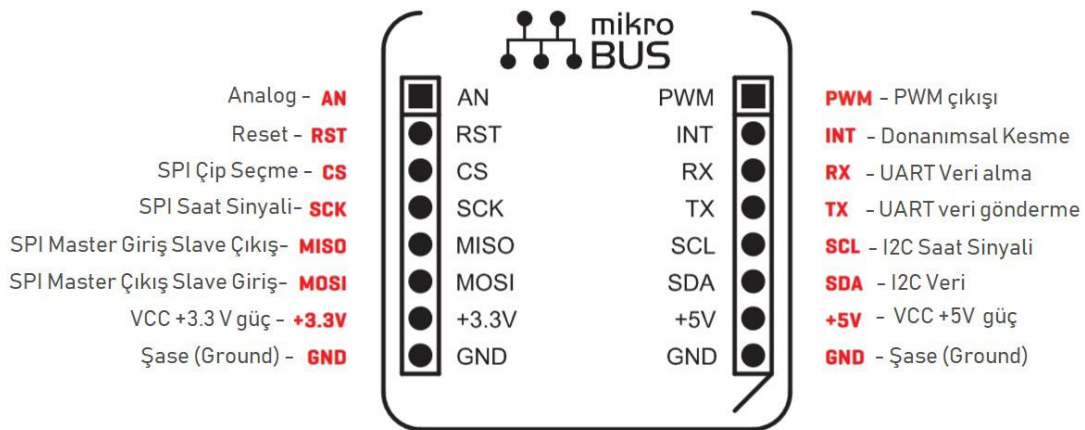
Sorular:

- 1- Ekranda 4 adet buton tanımlayarak LED0, LED1, LED2 ve LED3'ü açıp kapatan programı yazınız. (Butona ilk tıklamada led yanacak, ikinci tıklamada sönecektir.)

Uygulama 17: mikroBUS – 8x8 Led Matrix Display Uygulaması.

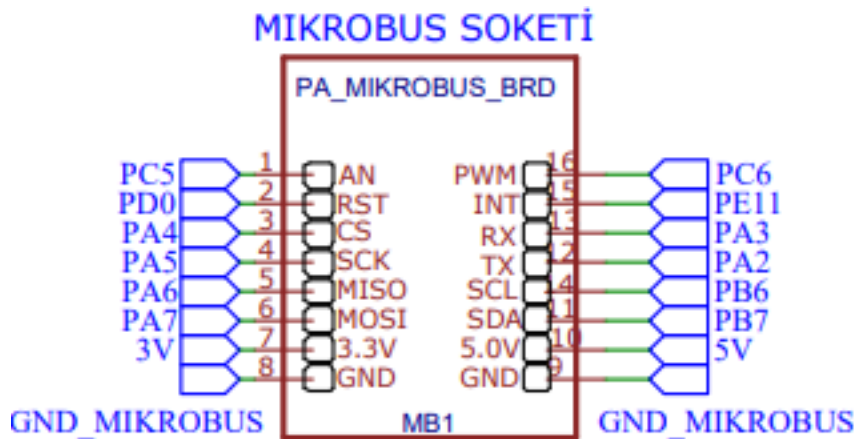
mikroBUS standardı, mikrodenetleyicileri veya mikroişlemcileri (ana kartlar) entegre devreler ve modüller (ek kartlar) ile bağlamak için kullanılan anakart soketlerini ve eklenti kartlarını tanımlar. Standart, mikroBUS pin çıkışı, iletişim ve güç pinlerinin fiziksel yerleşimini, kullanılan pinleri, ek kartların boyutu ve şeklini, mikroBUS soketinin anakart üzerindeki konumunu ve son olarak hem ek kartlar hem de soketler için serigrafik kurallarını belirler.

mikroBUS' in amacı, her biri tek bir sensör, alıcı verici, ekran, kodlayıcı, motor sürücüsü, bağlantı portu veya başka herhangi bir elektronik modül veya entegre devre taşıyan çok sayıda standart kompakt eklenti kartıyla kolay donanım genişletilebilirliğini sağlamaktır. MikroElektronika firması tarafından oluşturulan mikroBUS açık kaynak bir standarttır ve tasarım şartları yerine getirildiği sürece herkes donanım tasarımında mikroBUS' ı uygulayabilir.



Şekil 18.1: mikroBus Standardı.

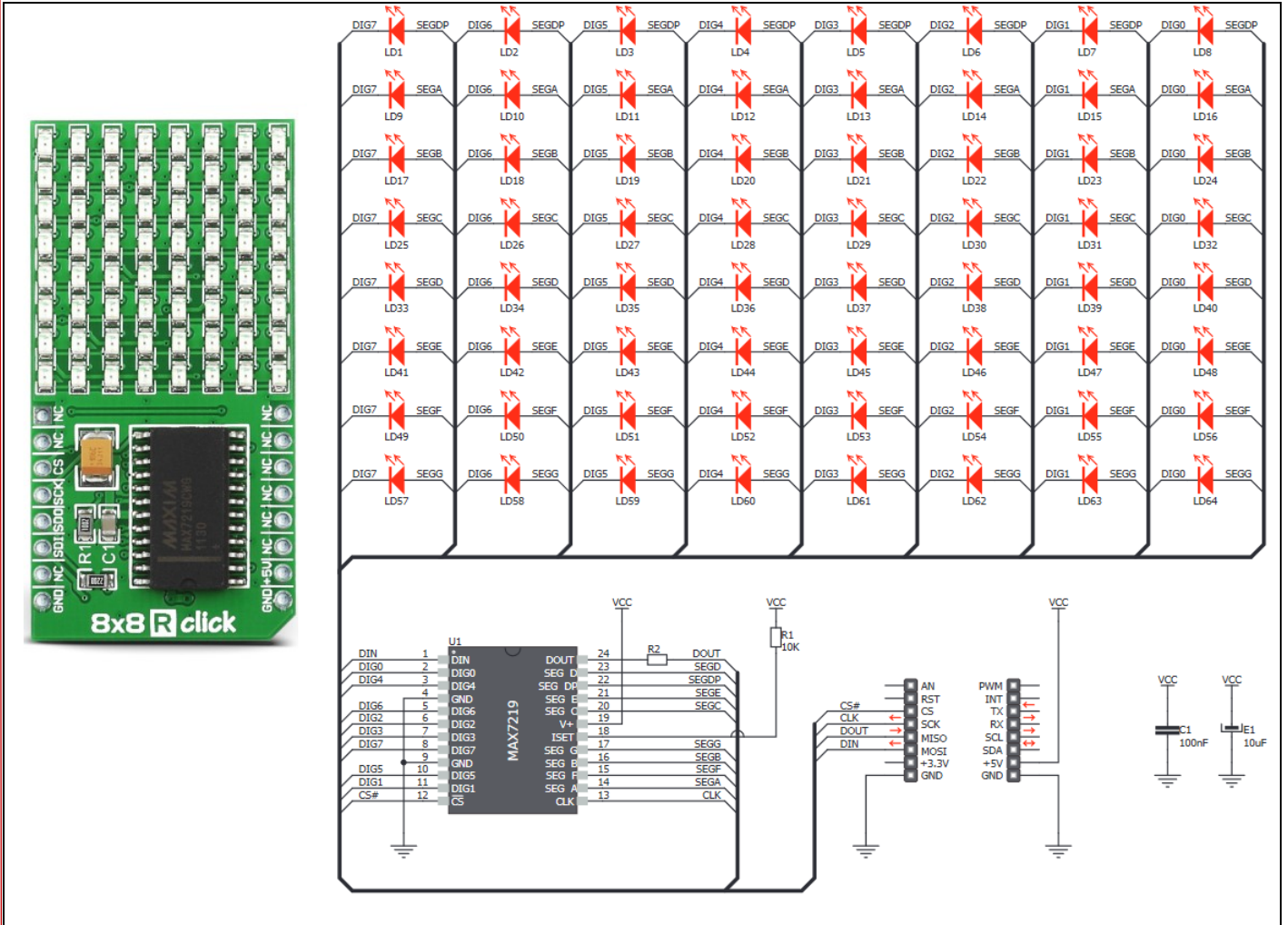
Mikroelektronika firması mikroBUS soketlerine uyumlu "click board" ismi verilen eklenti kartları tasarlayıp üretmektedir. Click kartları WiFi, LoRa, BLE, GSM, GPS, OLED, LED, ADC, röle, konuşma tanıma, biyosensörler, hareket algılayıcılar, çevre algılayıcılar gibi geniş bir yelpazede ürün barındırmaktadır. Ağustos 2021 itibarıyla 1047 adet click kartı bulunmaktadır.



Şekil 18.2: ARMapp-18 uygulama setinde bulunan mikroBus soketine ait devre şeması.

Click kartları mikroBUS soketine uyumludur. Her click mikroBUS soketinin farklı pinlerini kullanabilir. Mesela doğalgaz seviyesi ölçümü yapan bir click kartı analog pini ve besleme pinlerini kullanırken, buzzer barındıran bir click kart INT pinini ve besleme pinlerini kullanabilir. Şekil 18.2' de ARMapp-18 uygulama setinde bulunan mikroBUS soketine ait devre şeması görülebilir.

Bu uygulamada mikroBUS soketine mikroelektronika firmasının 8x8R Click isimli kartı takılarak kullanılmıştır. Şekil 18.3' te bu kart ve devre şeması görülebilir.



Şekil 18.3: 8x8R Click Kartı ve devre şeması.

Bu kart üzerinde MAX7219 entegresi bulunmaktadır ve bu entegre SPI protokolü kullanılarak iletişim sağlamaktadır. Bu entegre ile 8x8 led matris oluşturulurken satırlar yukarıdan aşağı sırasıyla DP, A, B, C, D, E, F, G, sütunlar ise soldan sağa DIG7, DIG6, DIG5, DIG4, DIG3, DIG2, DIG1 ve DIG0 olarak bağlanmıştır. Uygulamada oluşturulan 2 boyutlu dizilerin (matrixlerin) sırasıyla her satırının önce 7. elemanları, sonra 6. elemanlar vs. olacak şekilde MAX7219 entegresine gönderilir.

Bu uygulamada öncelikle displaydeki bütün ledler yakılarak parlaklıkları değiştirilmiş daha sonra da animasyonlar oynatılmıştır. Son olarak menü tuşları ile istenilen piksele gidilerek onay tuşuna basıldığında o pikselin yakılması yani menü tuşlarıyla desen oluşturm işlemi yapılmıştır. Onay tuşuna basılı tutularak displaye çizilen desen temizlenebilir.

Uygulama Kodları:

```

1 ///////////////////////////////////////////////////////////////////
2 //*****//
3 // 8x8R Click Matrix Led Display Uygulaması//
4 //      MikroC v6.2 - STM32F407VG      //
5 // ARMap-18 Deney Seti için yazılmıştır //
6 //*****//
7 //      http://elektrovadi.com      //
8 //      http://mikrodunya.wordpress.com //
9 ///////////////////////////////////////////////////////////////////
10
11 #define Max7219_NoOp          0x00
12 #define Max7219_Dig0         0x01
13 #define Max7219_Dig1         0x02
14 #define Max7219_Dig2         0x03
15 #define Max7219_Dig3         0x04
16 #define Max7219_Dig4         0x05
17 #define Max7219_Dig5         0x06
18 #define Max7219_Dig6         0x07
19 #define Max7219_Dig7         0x08
20 #define Max7219_Decode       0x09
21 #define Max7219_Intensity    0x0A
22 #define Max7219_ScanLimit    0x0B
23 #define Max7219_ShutDown     0x0C
24 #define Max7219_Disptest     0x0F
25
26 #define CS      GPIOA_ODR.B4
27
28 #define yukari  GPIOC_IDR.B4
29 #define asagi   GPIOB_IDR.B14
30 #define sol     GPIOB_IDR.B12
31 #define sag     GPIOB_IDR.B13
32 #define ok      GPIOB_IDR.B8
33
34
35 char animasyon1[8][8]=
36 {
37     {1,1,1,1,1,1,1,1},
38     {0,0,0,0,0,0,0,0},
39     {0,0,0,0,0,0,0,0},
40     {0,0,0,0,0,0,0,0},
41     {0,0,0,0,0,0,0,0},
42     {0,0,0,0,0,0,0,0},
43     {0,0,0,0,0,0,0,0},
44     {1,1,1,1,1,1,1,1}
45 };
46
47 char animasyon2[8][8]=
48 {
49     {1,1,1,1,1,1,1,1},
50     {0,0,0,0,0,0,0,0},
51     {0,0,0,0,0,0,0,0},
52     {1,0,0,0,0,0,0,0},
53     {1,0,0,0,0,0,0,0},
54     {0,0,0,0,0,0,0,0},
55     {0,0,0,0,0,0,0,0},
56     {1,1,1,1,1,1,1,1}
57 };

```

```
58
59 char animasyon3[8][8]=
60 {
61     {1,1,1,1,1,1,1,1},
62     {0,0,0,0,0,0,0,0},
63     {0,0,0,0,0,0,0,0},
64     {1,1,0,0,0,0,0,0},
65     {1,1,0,0,0,0,0,0},
66     {0,0,0,0,0,0,0,0},
67     {0,0,0,0,0,0,0,0},
68     {1,1,1,1,1,1,1,1}
69 };
70
71 char animasyon4[8][8]=
72 {
73     {1,1,1,1,1,1,1,1},
74     {0,0,0,0,0,0,0,0},
75     {0,0,0,0,0,0,0,0},
76     {0,1,1,0,0,0,0,0},
77     {0,1,1,0,0,0,0,0},
78     {0,0,0,0,0,0,0,0},
79     {0,0,0,0,0,0,0,0},
80     {1,1,1,1,1,1,1,1}
81 };
82
83 char animasyon5[8][8]=
84 {
85     {1,1,1,1,1,1,1,1},
86     {0,0,0,0,0,0,0,0},
87     {0,0,0,0,0,0,0,0},
88     {0,0,1,1,0,0,0,0},
89     {0,0,1,1,0,0,0,0},
90     {0,0,0,0,0,0,0,0},
91     {0,0,0,0,0,0,0,0},
92     {1,1,1,1,1,1,1,1}
93 };
```

```
94
95 char animasyon6[8][8]=
96 {
97     {1,1,1,1,1,1,1,1},
98     {0,0,0,0,0,0,0,0},
99     {0,0,0,0,0,0,0,0},
100    {0,0,0,1,1,0,0,0},
101    {0,0,0,1,1,0,0,0},
102    {0,0,0,0,0,0,0,0},
103    {0,0,0,0,0,0,0,0},
104    {1,1,1,1,1,1,1,1}
105 };
106
107 char animasyon7[8][8]=
108 {
109     {1,1,1,1,1,1,1,1},
110     {0,0,0,0,0,0,0,0},
111     {0,0,0,0,0,0,0,0},
112     {0,0,0,0,1,1,0,0},
113     {0,0,0,0,1,1,0,0},
114     {0,0,0,0,0,0,0,0},
115     {0,0,0,0,0,0,0,0},
116     {1,1,1,1,1,1,1,1}
117 };
118
119 char animasyon8[8][8]=
120 {
121     {1,1,1,1,1,1,1,1},
122     {0,0,0,0,0,0,0,0},
123     {0,0,0,0,0,0,0,0},
124     {0,0,0,0,0,1,1,0},
125     {0,0,0,0,0,1,1,0},
126     {0,0,0,0,0,0,0,0},
127     {0,0,0,0,0,0,0,0},
128     {1,1,1,1,1,1,1,1}
129 };
130
```



```

131 char animasyon9[8][8]=
132 {
133     {1,1,1,1,1,1,1,1},
134     {0,0,0,0,0,0,0,0},
135     {0,0,0,0,0,0,0,0},
136     {0,0,0,0,0,0,1,1},
137     {0,0,0,0,0,0,1,1},
138     {0,0,0,0,0,0,0,0},
139     {0,0,0,0,0,0,0,0},
140     {1,1,1,1,1,1,1,1}
141 };
142
143 char animasyon10[8][8]=
144 {
145     {1,1,1,1,1,1,1,1},
146     {0,0,0,0,0,0,0,0},
147     {0,0,0,0,0,0,0,0},
148     {0,0,0,0,0,0,0,1},
149     {0,0,0,0,0,0,0,1},
150     {0,0,0,0,0,0,0,0},
151     {0,0,0,0,0,0,0,0},
152     {1,1,1,1,1,1,1,1}
153 };
154
155
156
157 char matrix_8x8[8][8];
158 char matrixTMP[8][8];
159 //*****
160 void max7219_komut(char adres, char veri) // max7219'a komut gönderen fonksiyon.
161 {
162     CS=0; // CS pini 0'a çekilerek çip iletişime hazırlanıyor.
163     Delay_Cyc(100); // 100 saykıl bekleniyor.
164     SPI1_Write(adres); // Register adresi gönderiliyor.
165     SPI1_Write(veri); // veri gönderiliyor.
166     Delay_Cyc(100); // 100 saykıl bekleniyor.
167     CS=1; // CS pini 1'e çekilerek çip iletişimi sonlandırılıyor.
168 }
.....
169 //*****
170 void max7219_goster(char mtrx[8][8]) // displaye görüntü basan fonksiyon.
171 {
172     char x,y; // x,y değişkenleri tanımlanıyor.
173     char digit; // digit değişkeni tanımlanıyor.
174     for(x=0;x<8;x++) // satırlara teker teker sağ üstten başlanarak
175     { // matrixin (2 boyutlu dizinin) elemanları gönderiliyor.
176         digit=0; // bu fonksiyonu anlayabilmek için 8x8 click şemasına ve
177         for(y=0;y<8;y++) // MAX7219 datasheetine bakmak gerekir.
178         {
179             digit<<=1;
180             digit|=mtrx[y][7-x];
181         }
182         max7219_komut(x+1,digit);
183     }
184 }
185 //*****
186 void max7219_pixelOn(char x, char y) // istenilen konumdaki pixeli yakan fonksiyon.
187 {
188     matrix_8x8[y][x]=1; // matrix_8x8 dizisinin istenilen konumu 1 yapılıyor.
189     max7219_goster(matrix_8x8); // matrix_8x8 dizisi displayde gösteriliyor.
190 }
191 //*****
192 void max7219_pixelOff( char x, char y) // istenilen konumdaki pixel söndüren fonksiyon.
193 {
194     matrix_8x8[y][x]=0; // matrix_8x8 dizisinin istenilen konumu 0 yapılıyor.
195     max7219_goster(matrix_8x8); // matrix_8x8 dizisi displayde gösteriliyor.
196 }
197 //*****
198 void max7219_temizle() // display temizleniyor.
199 {
200     char x,y; // x ve y değişkenleri tanımlanıyor.
201     for(x=0;x<8;x++) // display için kullanılan değişkenlerin bütün
202     { // elemanları sıfırlanıyor.
203         for(y=0;y<8;y++)
204         {
205             matrixTMP[x][y]=0;
206             matrix_8x8[x][y]=0;
207         }
208     }
209     max7219_goster(matrix_8x8); // display temizleniyor.
210 }

```

```

211 //*****
212 void max7219_doldur() // bütün displaydeki pixeller yakılıyor.
213 {
214     char x,y;
215     for(x=0;x<8;x++)
216     {
217         for(y=0;y<8;y++)
218         {
219             max7219_pixelOn(x,y);
220         }
221     }
222 }
223 //*****
224 void guncelle() // Menü tuşlarıyla istenilen pixellerin yakılabilmesi için
225 // tuşlarla gezinirken bir önceki display durumunun kaybolmaması
226 // için matrixTMP dizisine kopyalanıyor.
227 {
228     char x,y;
229     for(x=0;x<8;x++)
230     {
231         for(y=0;y<8;y++)
232         {
233             matrixTMP[x][y]=matrix_8x8[x][y];
234         }
235     }
236 //*****
237
238 void main()
239 {
240     char x,y,i,j;
241     GPIO_Digital_Output(&GPIOA_BASE, _GPIO_PINMASK_4); // MikroBus CS pini çıkış olarak ayarlanıyor.
242
243     GPIO_Digital_Input(&GPIOB_BASE, _GPIO_PINMASK_8 // Menü butonları giriş olarak ayarlanıyor.
244                       | _GPIO_PINMASK_12
245                       | _GPIO_PINMASK_13
246                       | _GPIO_PINMASK_14);
247
248     GPIO_Digital_Input(&GPIOC_BASE, _GPIO_PINMASK_4);
249
250     SPI1_Init(); // SPI1 kuruluyor.
251     //SPI1_Init_Advanced(_SPI_FCLK_DIV16, _SPI_MASTER | _SPI_8_BIT | _SPI_CLK_IDLE_LOW | _SPI_FIRST_CLK_EDGE_TRANSITION | _SPI
252
253     max7219_komut(Max7219_ShutDown, 0x01); // max7219 shutdown registerına 0x01 değeri gönderilerek normal çalışmay
254     max7219_komut(Max7219_DispTest, 0x00); // Test registerına 0x00 gönderilerek normal işlem moduna alınıyor.
255     max7219_komut(Max7219_Decode, 0x00); // Decode mod iptal ediliyor. (BCD Decode)
256     max7219_komut(Max7219_ScanLimit,0x07); // 7 digit taranacak
257     max7219_komut(Max7219_Intensity,0x01); // Led parlaklığı en aza indiriliyor. (0x00-0x0f arası ayarlanabilir.)
258
259     //////////////////////////////////////
260     max7219_doldur(); // 8x8 matrixin bütün ledleri yakılıyor.
261     for(j=0;j<5;j++) // 5 kez parlaklık değiştiriliyor.
262     {
263         for(i=0;i<15;i++)
264         {
265             max7219_komut(Max7219_Intensity,i);
266             delay_ms(100);
267         }
268     }
269     max7219_komut(Max7219_Intensity,1); // parlaklık en alt seviyeye alınıyor.
270     max7219_temizle(); // display temizleniyor.
271     //////////////////////////////////////
272     for(x=0;x<8;x++) // yatay olarak her satırda pikseller sırasıyla yakılıyor.
273     {
274         for(y=0;y<8;y++)
275         {
276             max7219_pixelOn(y,x);
277             delay_ms(50);
278         }
279     }
280     for(x=0;x<8;x++) // yatay olarak her satırda pikseller sırasıyla söndürülüyor.
281     {
282         for(y=0;y<8;y++)
283         {
284             max7219_pixelOff(x,y);
285             delay_ms(50);
286         }
287     }
288     //////////////////////////////////////

```

```

289 for(i=0;i<20;i++) // Displayde programın en üst kısmında tanımlanan
290 { // animasyonlar oynatılıyor.
291     max7219_goster(animasyon1);
292     delay_ms(35);
293     max7219_goster(animasyon2);
294     delay_ms(35);
295     max7219_goster(animasyon3);
296     delay_ms(35);
297     max7219_goster(animasyon4);
298     delay_ms(35);
299     max7219_goster(animasyon5);
300     delay_ms(35);
301     max7219_goster(animasyon6);
302     delay_ms(35);
303     max7219_goster(animasyon7);
304     delay_ms(35);
305     max7219_goster(animasyon8);
306     delay_ms(35);
307     max7219_goster(animasyon9);
308     delay_ms(35);
309     max7219_goster(animasyon10);
310     delay_ms(35);
311     max7219_goster(animasyon9);
312     delay_ms(35);
313     max7219_goster(animasyon8);
314     delay_ms(35);
315     max7219_goster(animasyon7);
316     delay_ms(35);
317     max7219_goster(animasyon6);
318     delay_ms(35);
319     max7219_goster(animasyon5);
320     delay_ms(35);
321     max7219_goster(animasyon4);
322     delay_ms(35);
323     max7219_goster(animasyon3);
324     delay_ms(35);
325     max7219_goster(animasyon2);
326     delay_ms(35);
327 }
328 max7219_temizle(); // Display temizleniyor.
329 ////////////////////////////////////////////////////////////////////
330 x=4;y=4; // Displayin (4,4) konumundaki pixel matrixTMP isimli 2 boyutlu dizide yakılıyor.
331 matrixTMP[x][y]=1;
332 max7219_goster(matrixTMP); // matrixTMP isimli dizi displayde gösteriliyor.
333 while(1) // sonsuz döngü
334 {
335     if(ok) // Eğer ok tuşuna basıldıysa
336     {
337         char sayac=0; // sayacı sıfırla.
338         matrix_8x8[y][x]=1; // matrix_8x8 isimli dizinin istenilen konumunu 1 yap.
339         delay_ms(100); // 100 ms bekle
340         while(ok) // ok tuşuna basılı tutulursa displayi temizleyen algoritam
341         {
342             sayac++; // sayac arttırılıyor.
343             delay_ms(100); // 100 ms bekleniyor.
344             if(sayac>10) // sayac 10'dan büyükse
345             {
346                 sayac=0; // sayacı sıfırla
347                 max7219_temizle(); // displayi temizle
348                 matrixTMP[y][x]=1; // o anki mevcut konumdaki pixeli matrixTMP dizisinde 1 yap.
349                 guncelle(); // display güncelleniyor.
350                 max7219_goster(matrixTMP); // displayde matrixTMP dizisini göster.
351             }
352         }
353     }
354
355     if(yukari) // yukarı tuşuna basılırsa
356     {
357         matrixTMP[y][x]=0; // o anki konumdaki pikseli söndür.
358         if(y>0)y--; // y sıfırdan büyükse y konumunu bir azalt. (0,0) kınumu sol üst köşedir. (7,7) konumu sağ alt köşedir.
359         matrixTMP[y][x]=1; // matrixTMP dizisinin o anki konumunu 1 yap.
360         guncelle(); // matrixTMP dizisini bir önceki haliyle güncelle.
361         max7219_goster(matrixTMP); // matrixTMP dizisini displayde göster.
362         delay_ms(200); // 200 ms bekle.
363     }
364
365     if(asagi) // aşağı tuşuna basınca
366     {
367         matrixTMP[y][x]=0; // o anki konumdaki pikseli söndür.
368         if(y<7)y++; // y yediden küçükse y konumunu bir arttır.
369         matrixTMP[y][x]=1; // matrixTMP dizisinin o anki konumunu 1 yap.
370         guncelle(); // matrixTMP dizisini bir önceki haliyle güncelle.
371         max7219_goster(matrixTMP); // matrixTMP dizisini displayde göster.
372         delay_ms(200); // 200 ms bekle.
373     }



```

```

374
375     if(sol) // sol tuşuna basınca
376     {
377         matrixTMP[y][x]=0; // o anki konumdaki pikseli söndür.
378         if(x>0)x--; // x sıfırdan büyükse x konumunu bir azalt. (bir pixel sola gel)
379         matrixTMP[y][x]=1; // matrixTMP dizisinin o anki konumunu 1 yap.
380         guncelle(); // matrixTMP dizisini bir önceki haliyle güncelle.
381         max7219_goster(matrixTMP); // matrixTMP dizisini displayde göster.
382         delay_ms(200); // 200 ms bekle.
383     }
384
385     if(sag) // sag tuşuna basınca
386     {
387         matrixTMP[y][x]=0; // o anki konumdaki pikseli söndür.
388         if(x<7)x++; // x yediden küçükse x konumunu bir arttır.
389         matrixTMP[y][x]=1; // matrixTMP dizisinin o anki konumunu 1 yap.
390         guncelle(); // matrixTMP dizisini bir önceki haliyle güncelle.
391         max7219_goster(matrixTMP); // matrixTMP dizisini displayde göster.
392         delay_ms(200); // 200 ms bekle.
393     }
394 }
395 }
396 //*****

```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “SPI” kütüphanesini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve  ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Piyano anahtarlardan sol taraftakinde bulunan “TUŞ TAKIMI” anahtarını ve “mikroBUS” anahtarını, açarak, bu modüllerin GND bağlantısını sağlayınız.
- 12-  “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz.
- 13- Programın çalışmasını gözlemleyiniz. Animasyonlar bittikten sonra menü tuşlarıyla ekranda desen oluşturunuz.

Sorular:

0-9 arası rakamları 1 er saniye aralıklarla ekranda yazdıran programı yazınız.

Uygulama 18: mikroBUS-Gsm2Click ile Led Yakma Uygulaması

MikroBus standardı, mikrodenetleyicileri veya mikroişlemcileri entegre devreler ve modüller ile bağlamak için kullanılan anakart soketlerini ve eklenti kartlarını tanımlar. MikroElektronika Firması tarafından oluşturulan mikroBUS açık kaynak bir standarttır ve tasarım şartları yerine getirildiği sürece herkes donanım tasarımında mikroBUS'ı uygulayabilir. Bu modülün detayları Uygulama 18 başlığı altında anlatılmıştır. Bu uygulama kapsamında aynı üretici tarafından mikroBUS soketine uygun olarak üretilmiş, mikrodenetleyici projelerine hücresel bağlantı yetenekleri eklemek için kullanılan, Quectel M95 4G GSM modülünü içeren GSM2Click kartı incelenecektir. Bu kart mikrodenetleyici sistemlerine SMS gönderme, arama yapma, veri iletimi gibi hücresel ağ işlevlerini eklemek için kullanılır. Bu uygulamada özetle; GSM2Click kartı kullanılarak UART2 modülü ile (RX(PA3)-TX(PA2)) mikrodenetleyiciye bir telefon hattı aracılığı ile 8 bitlik değer gönderilecek, ardından bu değere göre (PE7-PE0) pinlerine bağlı olan LED'ler yanacaktır.

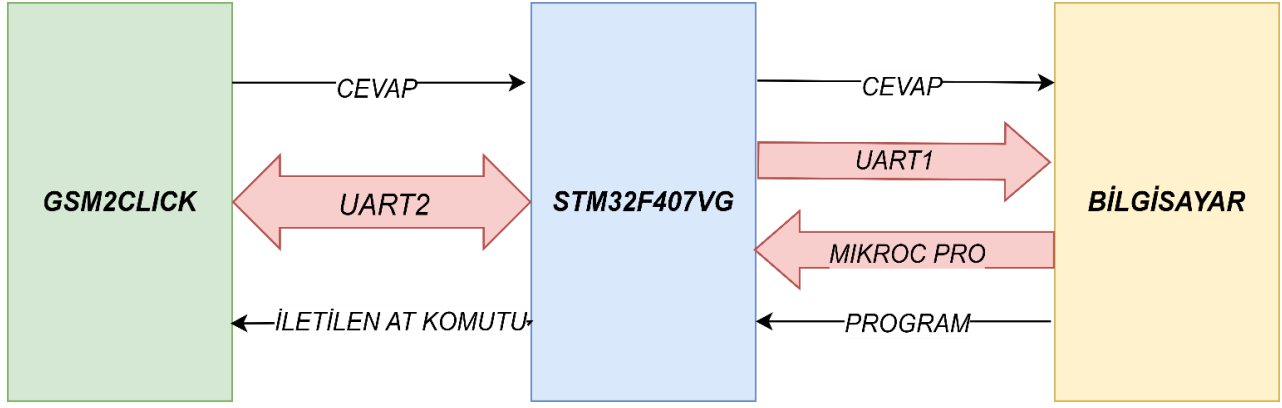


STM32F4707VG ile haberleşebilmesi için kartı mikroBUS soketine yerleştirip Piyano anahtarlardan mikroBUS ve UART(UART1 üzerinden kartın yanıtlarını bilgisayar ortamında izleyebilmek için) açılması yeterlidir.

Quectel M95 GSM modülünün çalışma prensibi genel olarak UART üzerinden bir "AT KOMUTU" alıp bu kodu modül içerisinde uygun olarak çözüp ona uygun olarak cevap vermesidir. Örnek olarak Mikrodenetleyiciden Modüle "AT<CR><LF>" mesajı gönderilirse modül "OK" yanıtını iletacaktır. Başka bir örnek olarak "AT+CNMI=1,2,0,0,0\r\n" mesajı iletilirse, modül SMS'leri doğrudan terminale iletir ve SMS içeriği görüntülenebilir. Daha fazla AT komutu ve detayları "Quectel M95 AT Command Manual" içeriğinde görüntülenebilir.

ARMapp-18

Bu uygulamada, STM ve GSM2Click UART2 üzerinden haberleşirken anlık olarak GSM2Click'in cevaplarını görebilmek, programlarken oluşabilecek hataları çözmek ve daha iyi bir çalışma sürdürmek için gereklidir. Bu sebeple mikrodenetleyiciye iletilen mesajı UART1 modülü(RX(PA9)-TX(PA10))üzerinden bilgisayara iletmek ve sürekli olarak GSM modülünün cevaplarını izlemek, bu alandaki çalışmalar için daha faydalı olacaktır. Bilgisayar üzerinden mesajı izlemek için gerekli kodlar yazıldıktan ve UART1 modülü bilgisayara bağlandıktan sonra "hercules" gibi bir port izleyicisi veya mikroc PRO'nun USART terminali kullanılabilir. Anlatılanlar, daha açıklayıcı olması amacıyla **Şekil19.2'**de gösterilmiştir.



Şekil19.2- Uygulama19 Blok Şeması

C kodu yazılmış olan uygulamada kodun doğru çalışabilmesi için modüle SMS üzerinden sadece 8 bitlik bir sayı gönderilmelidir. Gönderilen sayıya göre E portuna bağlı LED'ler yanacaktır.

Uygulama Kodları:

```
#include <built_in.h> // Gecikme fonksiyonları için
#include <string.h> // String işlemleri için

char a[100];
int index = 0;
int extractedBits = 0b0;
int previous_value = 0b0;
int i;
int j;

void clear_buffer() { //Buffer kod içinde sürekli sıfırlandığı için bir kere fonksiyon tanımlandı.
    for ( i = 0; i < 100; i++) {
        a[i] = '0';
    }
    index = 0;
}

void main() {
    clear_buffer();

    GPIO_Digital_Output(&GPIOE_BASE, _GPIO_PINMASK_0
    | _GPIO_PINMASK_1
    | _GPIO_PINMASK_2
    | _GPIO_PINMASK_3
    | _GPIO_PINMASK_4
    | _GPIO_PINMASK_5
    | _GPIO_PINMASK_6
    | _GPIO_PINMASK_7); // E PORTU ÇIKIŞ OLARAK TANIMLANDI:

    UART2_Init(115200); //UART BAUD RATE
    UART1_Init(115200); //UART BAUD RATE
    delay_ms(1500);
    delay_ms(1000);
    UART2_Write_Text("AT+CMGF=1\r\n"); //Modülü "pdu" modundan "text" moduna geçiren AT komutu.
    delay_ms(400);
    UART2_Write_Text("ATE0\r\n");
    delay_ms(400);
    UART2_Write_Text ("AT+CNMI=1,2,0,0,0\r\n");

    while (1) {
        if (UART2_Data_Ready()) {
            a[index] = UART2_Read();
            UART1_Write(a[index]); //UART de okunan değeri yani GSM modülünün yanıtını anlık olarak UART1 üzerinden yazdırıp bilgisayarda görüntülemek için.

            // Dizide \r\n gördüğünde okumayı bırak
            if (index > 0 && a[index-1] == '\r' && a[index] == '\n') {
                // \r\n tespit edildiğinde işleme başla

                extractedBits = 0; // extractedBits'i sıfırla

                for ( i = 0; i < 8; i++) { // for döngüsü
                    extractedBits <<= 1; // extractedBits'i bir bit sola kaydır
                    if (a[i] == '1') {
                        extractedBits |= 1; // Eğer karakter '1' ise, extractedBits'in en sağındaki biti 1 yap
                    } //Bu for döngüsünde kısaca girilecek sms üzerinden girilecek 8bitlik sayı char tipinden int tipine dönüştürüldü.
                }
                //Burada previous_value'ya atılma sebebi sms birdaha gönderildiği zaman extractedBit'in sıfırdan başlaması gerektiridir.
                previous_value = extractedBits; // extractedBits'i previous_value'ya ata
                extractedBits = 0; // extractedBits'i sıfırla
                clear_buffer(); //Buffer'ı temizle
                index = 0; // index'i sıfırla, yeni veri için hazırla
            } else {
                index++;
            }
        }

        GPIOE_ODR = previous_value; // LED'lerin durumunu güncelle
    }
}
```

İşlem Basamakları:

- 1- MikroC Pro for ARM programını çalıştırınız.
- 2- “Project” sekmesinden “New Project” seçeneğine tıklayınız.
- 3- Açılan ekrandan “Standart Project” seçilerek “Next” butonuna tıklayınız.
- 4- Açılan pencereden proje ismi, projenin kaydedileceği dizin, kullanılacak mikrodenetleyiciyi (STM32F407VG) ve mikrodenetleyici frekansını (168.000000 MHZ) belirleyerek ve “Open Edit Project window to set Configuration bits” seçeneğini işaretleyerek “Next” butonuna tıklayınız.
- 5- Açılan pencerede “Finish” butonuna tıklayarak proje oluşturma aşamasını bitiriniz.
- 6- Açılan pencerede sağ tarafta bulunan Load Scheme butonuna tıklayınız ve açılan pencereden “HSI 16 MHz_PLL 168 MHz Clock Ayarı.cfgsch” dosyasını seçiniz.
- 7- “OK” butonuna tıklayarak proje oluşturma işlemini tamamlayınız.
- 8- Derleyici ekranının sağ alt kısmında bulunan “Library Manager” sekmesinden “UART” ve “ C-String” kütüphanelerini seçiniz.
- 9- Açılan “.c” uzantılı sayfada kodları yazınız ve ikonuna tıklayarak projeyi kaydediniz.
- 10- ARMapp-18 setini USB kablo kullanarak bilgisayara bağlayınız.
- 11- Piyano anahtarlardan sağ taraftakinde bulunan mikroBUS ve UART anahtarını açarak, bu modüllerin GND bağlantısını sağlayınız.
- 12- GSM2Click mikrobus soketine takıldıktan, UART1 modülü bilgisayar ile bağlandıktan sonra bilgisayar üzerinde hercules programında UART1 in bağlandığı port izlenir.Uart terminali için birden fazla program mevcuttur. Hercules programı yerine başka programlar veya mikroc PRO USART Terminali de kullanılabilir.
- 13- “Build and Program” ikonuna tıklanarak veya (Ctrl+F11) tuşlarına aynı anda basarak yazılan programı derleyip STM32F407VG mikrodenetleyicisine yükleyiniz. 13- Programın çalışmasını gözlemleyiniz.

Program çalıştıktan sonra 8 bitlik bir tam sayı değeri, ilgili numaraya sms olarak iletildiğinde bu sms terminal üzerinde görüntülenecektir. Eş zamanlı olarak bu 8 bitlik değere göre PE7-PE0 arasındaki ledler yanacaktır.

Sorular:

- 1-Herhangi bir SMS geldiğinde buzzer'ın 1 saniye çalıştığı programı yazınız.