T.C.
MİLLÎ EĞİTİM BAKANLIĞI

ELEKTRİK-ELEKTRONİK TEKNOLOJİSİ

MİKRODENETLEYİCİ İLE DİJİTAL İŞLEMLER

523EO0021

Ankara, 2012

- Bu modül, mesleki ve teknik eğitim okul/kurumlarında uygulanan Çerçeve Öğretim Programlarında yer alan yeterlikleri kazandırmaya yönelik olarak öğrencilere rehberlik etmek amacıyla hazırlanmış bireysel öğrenme materyalidir.

- Millî Eğitim Bakanlığınca ücretsiz olarak verilmiştir.

- PARA İLE SATILMAZ.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**523EO0021**
ALAN	**Elektrik Elektronik Teknolojisi**
DAL/MESLEK	**Otomasyon Sistemleri**
MODÜLÜN ADI	**Mikrodenetleyici ile Dijital İşlemler**
MODÜLÜN TANIMI	Dijital işlemler için mikrodenetleyicinin programlaması ve çalıştırılması ile ilgili bilgi ve becerilerin kazandırıldığı öğrenme materyalidir.
SÜRE	40/32
ÖN KOŞUL	Mikrodenetleyici Programlama modülünü tamamlamış olmak
YETERLİK	Mikrodenetleyici ile dijital işlemleri yapmak
MODÜLÜN AMACI	**Genel Amaç** Öğrenci mikrodenetleyici ve çevre birimleri kullanarak dijital işlemler yapabileceksiniz. **Amaçlar** 1. Mikrodenetleyici ile temel seviye dijital işlemleri gerçekleştirebileceksiniz. 2. Mikrodenetleyici ile ileri seviye dijital işlemleri gerçekleştirebileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Atölye **Donanım:** Mikrodenetleyici, mikrodenetleyici eğitim seti veya programlama kartı, programlama yazılımı, haberleşme bağlantı kablosu
ÖLÇME VE DEĞERLENDİRME	Modül içinde yer alan her öğrenme faaliyetinden sonra verilen ölçme araçları ile kendinizi değerlendireceksiniz. Öğretmen modül sonunda ölçme aracı (çoktan seçmeli test, doğru-yanlış testi, boşluk doldurma, eşleştirme vb.) kullanarak modül uygulamaları ile kazandığınız bilgi ve becerileri ölçerek sizi değerlendirecektir.

GİRİŞ

Sevgili Öğrenci,

Günümüzde pek çok iş, yapısında mikro işlemci bulunan makineler ile gerçekleştirilir. Mikroişlemci (CPU) ile kontrol edilen sistemlerde işlemciden başka RAM, BIOS, G/Ç gibi ek birimlere ihtiyaç duyulur. Bu şekilde hem maliyet artar hem de işlemler zorlaşır. Bütün bu olumsuzlukları ortadan kaldıran devre elemanı mikrodenetleyicilerdir. Pek çok firma mikrodenetleyici (MCU-Micro Controller Unit) üretmektedir fakat bir chip(çip) firmasının PIC (Pripheral Interface Controller - Çevre birimlerini kontrol eden ünite) adını verdiği denetleme uygulamalarında geniş yer bulmaktadır.

Bu modüldeki tüm uygulamalarda hem kolay bulunduğu hem ucuz olduğu hem de rahat programlandığı için PIC16F84 kullanılmıştır. Modüldeki uygulamaları takip ederek dijital devreler geliştirebilir ve programlayabilirsiniz. Her uygulamada ayrı bir konu anlatılmaktadır. Uygulamalar basitten karmaşığa doğru sıralanmış ve sizin anlayabileceğiniz sadelikte işlenmiştir.

Modülün amacı, programlama mantığını öğrenerek, elektronik devre uygulamalarını mikrodenetleyiciyle çabuk, doğru ve kolayca çözme yeteneğini kazandırmaktır.

Bu modülün sonunda program yazabilecek, bu programa ait devreyi yapabilecek ayrıca kendi program ve devrelerinizi üretme yeteneğine sahip olabileceksiniz.

ÖĞRENME FAALİYETİ-1

AMAÇ

Mikrodenetleyiciyi ve çevre elemanlarını seçebilecek, dijital işlem için gerekli programı hatasız olarak yazabilecek, programı mikrodenetleyiciye yükleyebilecek ve devreyi hatasız olarak kurup çalıştırabileceksiniz.

ARAŞTIRMA

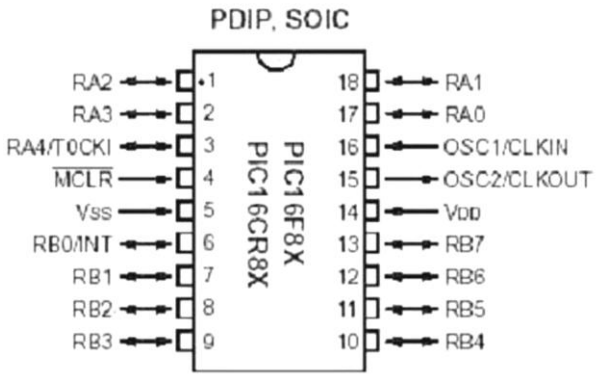
- Zaman gecikme döngüleri hakkında bilgi edinip ve istenilen sürede gecikme programı yazınız.
- Kesme (Interrupt) çeşitlerini öğrenmeli kesme alt programları yazınız.
- Yedi parçalı gösterge ve LCD göstergelerin kullanımı ile ilgili bilgi edininiz.
- Asenkron motorların yapısını ve çalışmasını öğreniniz.

Araştırma işlemleri için Mikrodenetleyici Programlama modülünü gözden geçirirken internet ortamlarından da yararlanabilirsiniz.

1. TEMEL SEVİYE DİJİTAL UYGULAMA DEVRELERİ

Karmaşık lojik fonksiyonlar mikrodenetleyici gibi tek bir entegrede toplamıştır. Genelde denetleyiciler tek başına bir sistem olmayıp başka bir sistemi yönetmek amacıyla kullanılır.

PIC serisi mikro işlemciler bir chip (çip) firması tarafından geliştirilmiş ve PIC adıyla anılmaktadır. CMOS teknolojisiyle üretilmiş olan PIC16F84 çok az enerji harcar. Programlanması ve silinmesi kolay olan EPROM belleğe sahiptir. Uygulamalarda 4 MHz'lik PIC16F84 mikrodenetleyici kullanılmıştır. Şekil 1.1'de PIC'in bacak bağlantıları görülmektedir.



Şekil 1.1: PIC 16F84'ün bacak bağlantısı

PIC 16F84 ile program geliştirmek için aşağıdaki donanım ve yazılımlar gereklidir.

- Metin editörü (Not defteri, word veya MPLAB PFE)
- MPASM assembler derleyici (Assembler dilindeki programları heksadesimal (HEX) koda çevirir.)
- PIC programlayıcı ve yazılımı (Seri port için PICUP programı, paralel port için PPLITE programı)

1.1. Trafik Lambası Uygulama Devresi

PIC 16F84 ile yapılan trafik lambası uygulama devresinde kırmızı, yeşil ve sarı olmak üzere üç adet led bulunmaktadır. Kırmızı ve yeşil ledler 1 dk., sarı led ise 3 sn. yanık kalacaktır. Devre ilk çalıştırıldığında tüm ledler sönük durumda olacaktır. Sistem "START" butonu ile harekete geçirilecek ve devrenin çalışması istenilen anda "RESET" butonu ile kesilecektir. Ledler kırmızı, sarı, yeşil sırasına göre yanacak ve tekrar başa dönecektir.

PIC'in A portu giriş, B portu çıkış olarak kullanılmaktadır. Portlardan hangisinin giriş ve çıkış olacağı kullanıcıya bağlıdır.

Bu devre, dijital entegreleri kullanılarak da basit bir şekilde yapılabilir. Fakat ledlerin yanık kalma sürelerini ayarlamak işin karmaşık kısmı oluşturmaktadır. Bu durumda devreye PIC girer. PIC'e yazılan programdaki küçük değişikliklerle ledlerin yanık kalma sürelerini artırılıp azaltılabilir. Bu süreleri değiştirmek için gecikme döngüleri yazılır.

Tek döngü ile yapılan programın süresi çok kısa olduğundan en az iki döngülü programlar kullanılır. Tablo 1.1 ve 1.2'de verilen zaman gecikme döngüleri farklı uygulamalarda kullanılabilir.

GECİKME			
	MOVLW	**D'255'**	1
	MOVWF	SAYAC1	1
DÖNGÜ1			
	MOVLW	D'255'	255
	MOVWF	SAYAC2	255
DÖNGÜ2			
	DECFSZ	SAYAC2,F	255*255
	GOTO	DONGU2	2*255*255
	DECFSZ	SAYAC1,F	255
	GOTO	DONGU1	2*255
	RETURN		2

Tablo 1.1: İki döngülü zaman gecikme programı

Tablo 1.1'de toplam 196.608 saykıl komut işlenmiştir. 4 MHz PIC için her komut 1µS gecikme sağladığından toplam gecikme süresi, 196608*1µS =196 ms, yani 0,196 saniye eder. Bu da yaklaşık 0,2 saniyeye karşılık gelir.

Tablo 1.2'de toplam 50.070.529 saykıl komut işlenmiştir. Toplam gecikme süresi, 50.070.529*1µS = 50.070.529 mikrosaniye ve yaklaşık 50 saniyedir. Bu süre yaklaşık 1 dakika kabul edilebilir. Dört döngülü zaman gecikme programından yaklaşık 12.767 sn.lik bir gecikme elde edilir. Bu da 212 dakika veya 3,5 saattir.

GECİKME				
	MOVLW	**D'255'**	1	1
	MOVWF	SAYAC1	1	1
DÖNGÜ1				
	MOVLW	D'255'	255	255
	MOVWF	SAYAC2	255	255
DÖNGÜ2				
	MOVLW	D'255'	255*255	650.25
	MOVWF	SAYAC3	255*255	650.25
DÖNGÜ3				
	DECFSZ	SAYAC3,F	255*255*255	165.581.375
	GOTO	DONGU3	2*255*255*255	33.162.750
	DECFSZ	SAYAC2,F	255*255	650.25
	GOTO	DONGU2	2*255*255	130.050
	DECFSZ	SAYAC1,F	255	255
	GOTO	DONGU1	2*255	510
	RETURN		2	2
				50.070.529

Tablo 1.2: Üç döngülü zaman gecikme programı

Sonuç olarak döngülerde kırmızı ile gösterilen değerler değiştirilirek döngü süreleri değiştirilir. Döngüde ilk değer olarak D '255' ile gösterilen kısma 1 ile 255 arasında değişen bir sayı girildiğinde her bir sayıyı minimum süre ile çarparak gecikme süresi bulunabilir.

DÖNGÜ	EN AZ SÜRE	EN ÇOK SÜRE
2	771 mikrosaniye	196,6 mili saniye
3	0,196 saniye	50 saniye
4	50 saniye	3 saat 32 dakika

Mesela 2'li döngüde ilk değeri D '125' girilirse 125X 771 =96,375 mili saniye, 3'lü döngüde ilk değer olarak D '150' girilirse 150X0,196=29,4 saniye ve 4'lü döngüde ilk değer olarak D '80' girilirse 80X50 = 1 saat 6 dakika 6 sn. elde edilir.

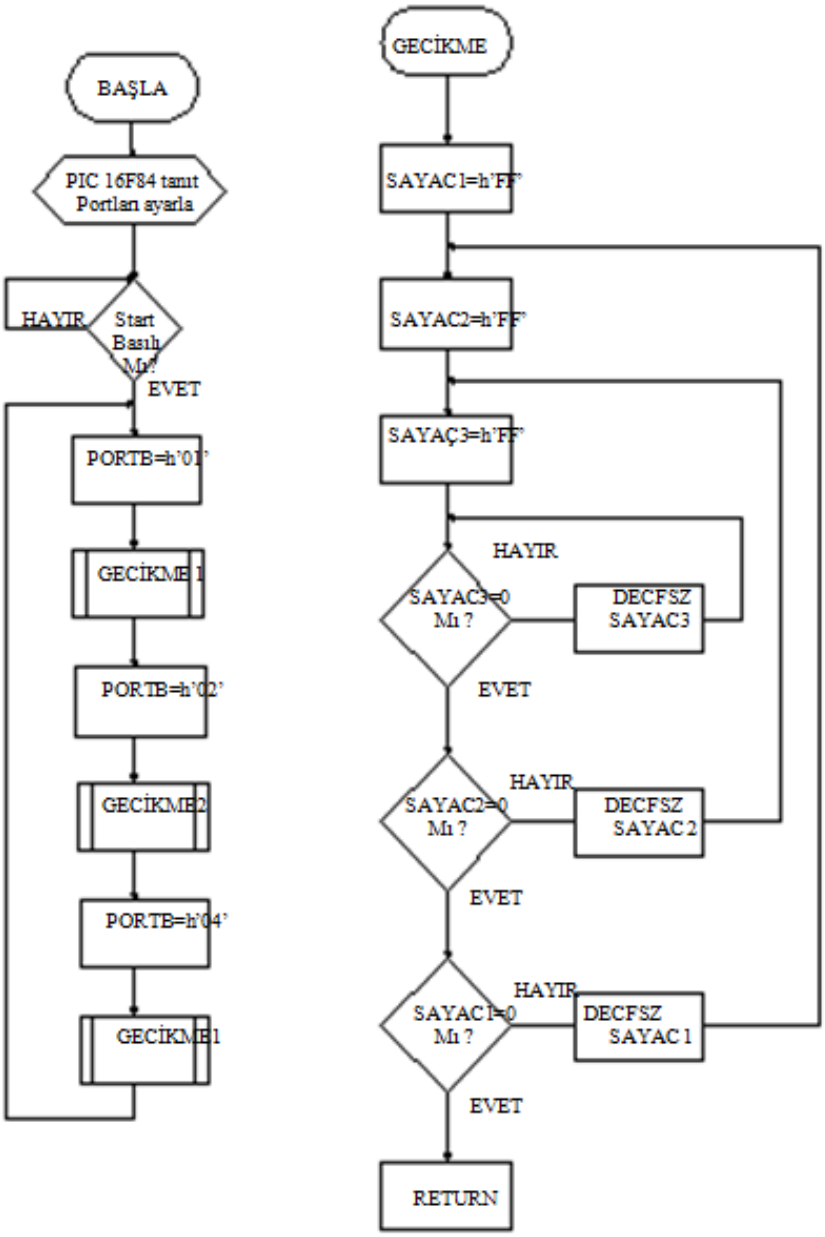
1.1.1. Devrenin Malzemeleri

- PIC16F84 4 MHz mikrodenetleyici
- 4 MHz kristal, buton, led
- C1 = C2 = 22pf
- R1 = R2 = R3 = 330Ω direnç
- R4 = R5 = 10KΩ, R6 = 100Ω

1.1.2. Akış Diyagramı

PIC assembler ile bir program yazmadan önce PIC'e yazılacak programı planlamamız gerekir. Bu planlama işlemi akış diyagramları ile yapılır. Akış diyagramı işlenecek komutların sırasıdır. Uzun ve karmaşık programlarda, akış diyagramları, hangi seviyeden programın ayrılacağını ve hangi komutları yerine getireceğini, sonra tekrar nereye döneceğini göstermesi açısından faydalıdır. Akış diyagramlarının kendine has sembolleri de vardır.

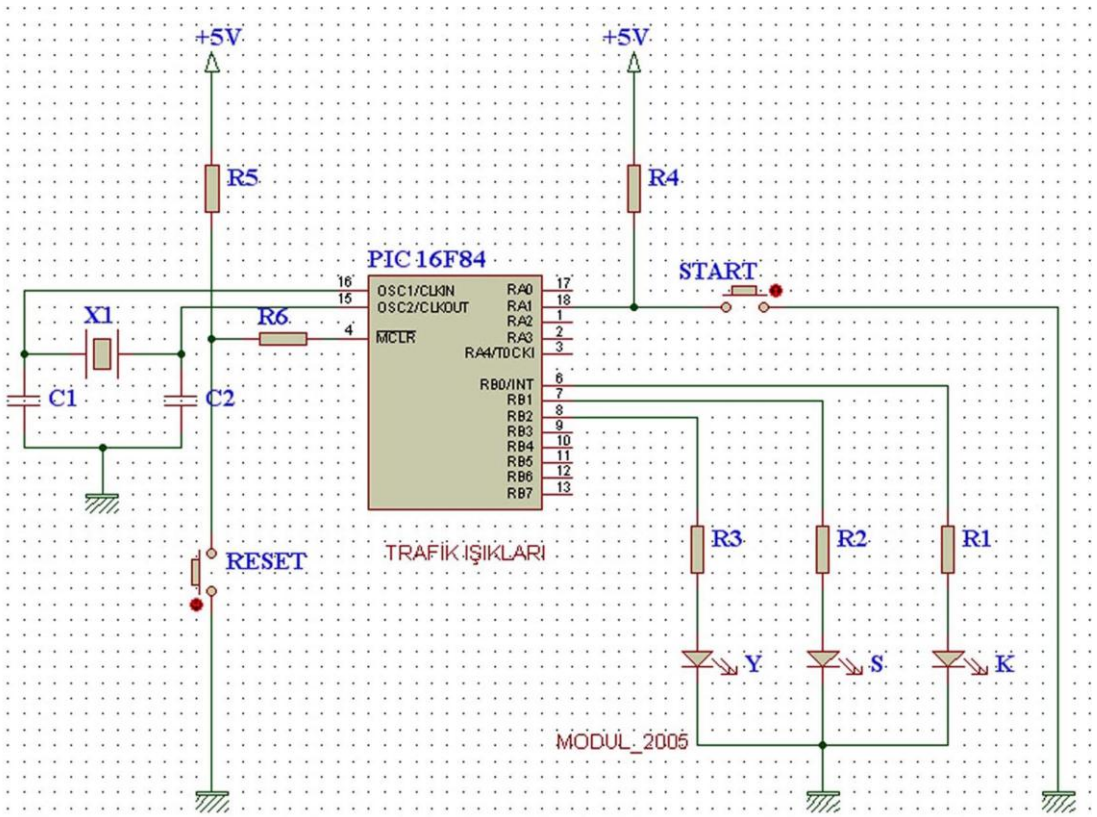
Gecikme 1 ve gecikme 2 alt programlarında yalnızca sayaçlara atanan değerler farklı olduğundan akış diyagramında "GECİKME" adıyla gösterilmiştir.



Tablo 1.3: Trafik lambası akış diyagramı

1.1.3. Devrenin Şeması

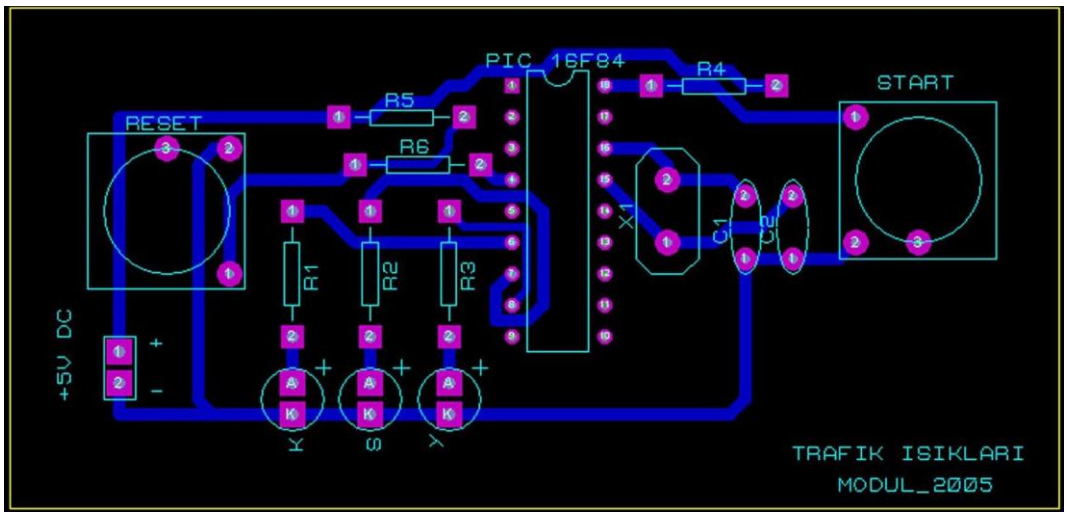
Şekil 1.2'teki devre, proteus programıyla çizilmiştir. Proteus programında devreyi kurabilir, PIC'e hex programı yüklenip devrenin çalışması görülebilir. Ayrıca kurulan devrenin baskı devresi ares programı yardımıyla direkt olarak çıkarılabilir. www.labcenter.co.uk adresinden proteus programın demosunu indirilebilir. Devrenin baskı devresi de Şekil 1.3'te görülmektedir.



Şekil 1.2: Trafik lambası uygulama devresi

Devrede PIC'in besleme bacakları gösterilmemiştir. 14 nu.lı bacak pozitif kaynak (Vcc ya da Vdd), 5 nu.lı bacak ise toprak (GND ya da Vss)'tır.

1.1.4. Baskı Devresi



Şekil 1.3: Trafik lambası uygulama baskı devresi

1.1.5. Devrenin Asm Programı

```
;========TRAFİK LAMBASI UYGULAMA PROGRAMI======S_2005=======

        LIST p=16F84
        INCLUDE "P16F84.INC"
SAYAC1  EQU   H'0C'          ; Gecikme alt programlarında kullanılan değişken
SAYAC2  EQU   H'0D'          ; Gecikme alt programlarında kullanılan değişken
SAYAC3  EQU   H'0E'          ; Gecikme alt programlarında kullanılan değişken

; Portları Ayarla……
        CLRF    PORTB               ; PORTB yi temizle
        BSF     STATUS,5            ; BANK1 e geç
        CLRF    TRISB               ; PORTB çıkış
        MOVLW   H'FF'               ; W <-- H'FF'
        MOVWF   TRISA               ; PORTA giriş
        BCF     STATUS,5            ; BANK0 a geç

Start butonuna basılıncaya kadar bekle……
BUTON
        BTFSC   PORTA,1             ; PORTA'nın 1. biti 0 mı?
        GOTO    BUTON               ; Hayır, tekrar test et

; Program kırmızı, sarı  ve yeşil ledleri sırasıyla yakarak çalışır……
TRAFIK
        MOVLW   H'01'               ; W <-- B'000000001'
        MOVWF   PORTB               ; Kırmızı ledi yak
```

```
        CALL    GECIKME1        ; 60sn. bekle


        MOVLW   H'02'           ; W <-- B'00000010'
        MOVWF   PORTB           ; Sarı  ledi yak
        CALL    GECIKME2        ; 3sn. bekle

        MOVLW   H'04'           ; W <-- B'00000100'
        MOVWF   PORTB           ; Yeşil ledi yak
        CALL    GECIKME1        ; 60sn. bekle

        GOTO    TRAFIK          ; TRAFIK etiketine git.

;=============GECİKME ALT PROGRAMLARI=====================

GECIKME1                        ; 60 sn'lik gecikme alt programı

        MOVLW   d'255'          ; W<--D'255'
        MOVWF   SAYAC1          ; SAYAC1 <-- W

G1
        MOVLW   d'255'          ; W<--D'255'
        MOVWF   SAYAC2          ; SAYAC2 <-- W
G2
        MOVLW   d'255'          ; W<--D'255'
        MOVWF   SAYAC3          ; SAYAC3 <-- W
G3
        DECFSZ  SAYAC3,F        ; Sayac3 bir azalt ve sıfır mı? kontrol et
        GOTO    G3              ; Hayır G3'e git
        DECFSZ  SAYAC2,F        ; Evet. Sayac2 bir azalt ve sıfır mı?
         GOTO   G2              ; Hayır G2'ye git
        DECFSZ  SAYAC1,F        ; Evet. Sayac1 bir azalt ve sıfır mı?
        GOTO    G1              ; Hayır G1'e git
        RETURN

GECIKME2                        ; 3sn'lik gecikme alt programı

        MOVLW   d'15'           ; W<--D'15'
        MOVWF   SAYAC1          ; SAYAC1 <-- W

D1
        MOVLW   d'255'          ; W<--D'15'
        MOVWF   SAYAC2          ; SAYAC1 <-- W
D2
        MOVLW   d'255'          ; W<--D'15'
        MOVWF   SAYAC3          ; SAYAC1 <-- W
```

```
D3
        DECFSZ      SAYAC3,F        ; Sayac3 bir azalt ve sıfır mı? bak
        GOTO        D3              ; Hayır D3'e git
        DECFSZ      SAYAC2,F        ; Sayac2 bir azalt ve sıfır mı?
        GOTO         D2             ; Hayır D2'ye git
        DECFSZ      SAYAC1,F        ; Sayac1 bir azalt ve sıfır mı?
        GOTO        D1              ; Hayır D1'e git

        RETURN
        END
```

1.1.6. Programın Simülasyonu

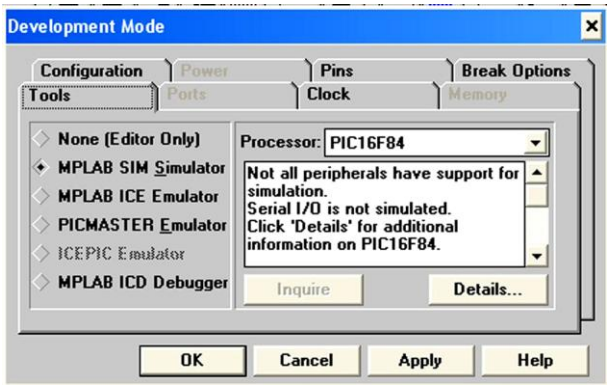
Programın PIC'e yüklenmeden önce çalıştırılması için MPLAB programı kullanılır. MPLAB yazılımı, ilgili firmanın sitesinden veya tanıtım CD'lerinden ücretsiz bulunabilir.

Bu programda bulunan simülatörle program satır satır çalıştırılıp her bir kaydedicinin aldığı değeri izlemek mümkündür. Böylece programın istenilen şekilde çalışıp çalışmadığı kolaylıkla anlaşılır ve hatalar giderilebilir.

1.1.6.1. Programın Kaydedilmesi ve Derlenmesi

MPLAB programı açıldığında konfigürasyonun ayarlanması gerekir. İlk olarak PIC seçilir.

- Option menüsü açılır.
- Development Mode seçeneği üzerine tıklanır. Şekil 1.4'teki pencere açılacaktır.
- MPLAB SIM'i işaretledikten sonra Processor kutusundan 16F84'ü seçilir.
- Program yazmak için File – New seçeneği tıklanır ve açılan pencrede proje ismi yazılıp OK tuşuna basılır.
- Program yazılıp File – Save As seçeneği ile "asm" uzantılı olarak kaydedilir. Programın MPLAB içine yazılması zorunlu değildir. Herhangi bir metin editöründe de ASM uzantılı olarak kaydedilebilir ve MPLAB programında bu dosya açılabilir.



Şekil 1.4: Development mode

- ASM uzantılı dosyanın derlenmesi yani hex uzantılı dosyaya çevrilmesi için önce konfigürasyonunu belirlemek gerekir.
  - Project menüsünden Edit Project seçildiğinde aşağıdaki pencere ekrana gelir.
  - Project Files bölümündeki Add Nodes butonuna basarak açılan pencerede "trafik.asm" adıyla kaydedilen dosya seçilir. Seçilen asm dosyası kaydedilen proje ile aynı klasörde değilse hata verir (Şekil 1.5).

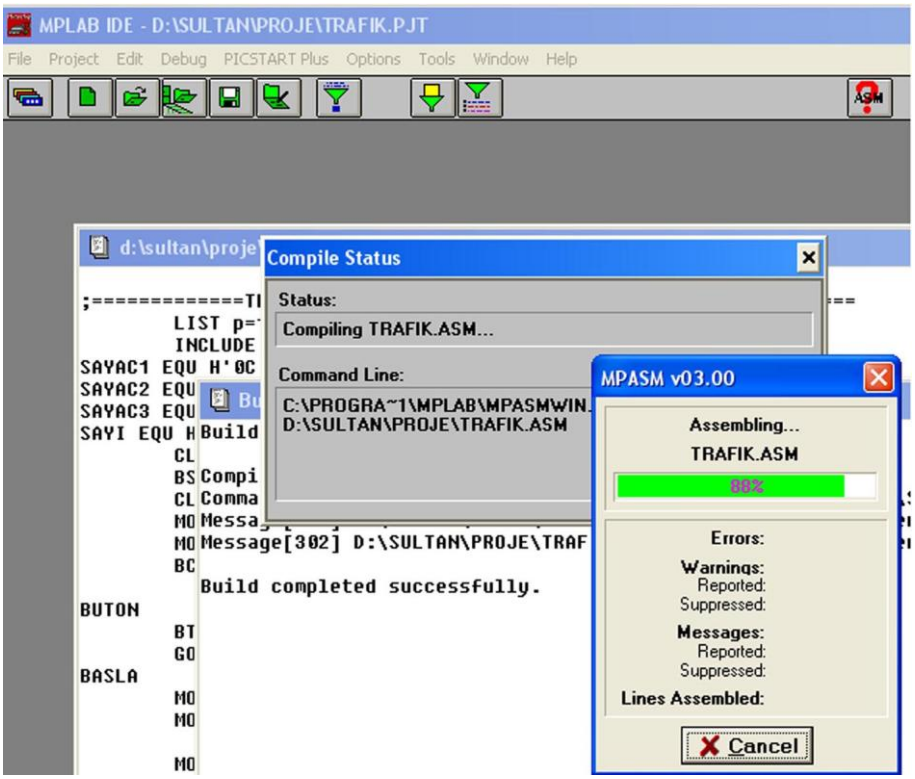
Şekil 1.5: Proje penceresi

- İşlemler bitince OK tuşuna basılıp metin editörüne geri dönülür.

- Dosyayı derlemek için Project araç çubuğundan Şekil 1.6'da görülen simgeye tıklanır ya da Project menüsünden "Built All" seçeneği seçilir.

Şekil 1.6: Derleme butonu

- Kaynak kod dosyası Şekil 1.7'de gösterildiği gibi derlenir. Eğer hata yoksa "Built completed succesfully" mesajı ile işlem tamamlanır.



Şekil 1.7: Derleme adımları

1.1.6.2. MPSIM PIC Simülatörü

Derleyicinin görevi, sadece program komutlarının kuralına göre yazılıp yazılmadığını denetlemek ve hexadesimal kodlara çevirme işlemini yapmaktır. Eğer mantıksal hatalar yapıldıysa bunlar derlemede ortaya çıkmaz ve hata olarak kabul edilmez.

MPLAB programının en önemli özelliği de mantıksal hataları aramayı kolaylaştırıcı MPSIM programını içermesidir. MPSIM'i kullanmak için Debug araç çubuğu ekrana getirilir.

Run: Simülasyonu başlatır.

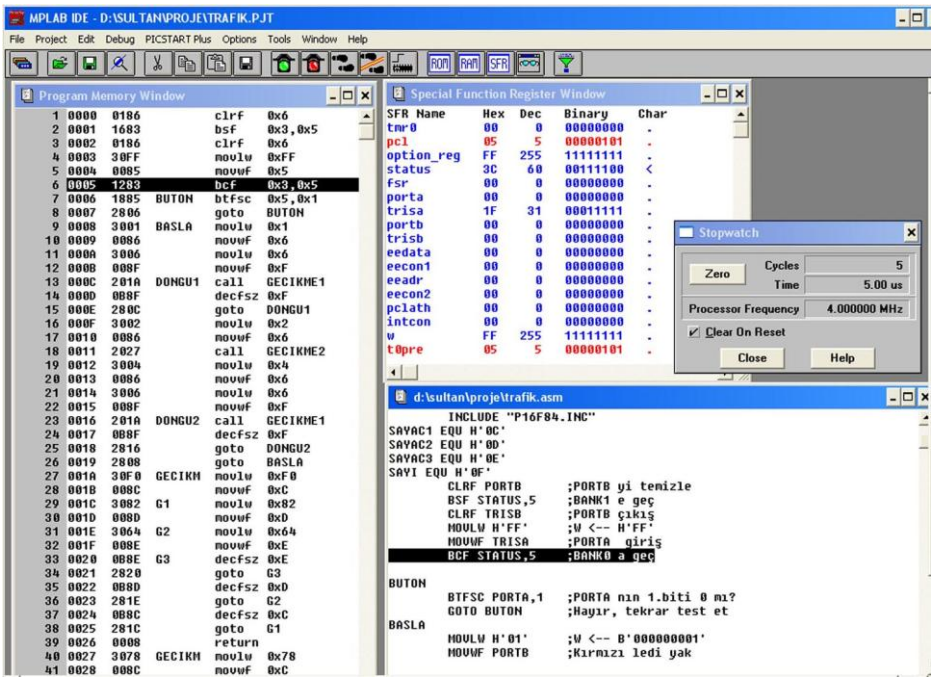
Halt Processor: Simülasyonu durdurur. Run butonuna tıklandığında program kaldığı yerden devam eder.

Step: Programı adım adım çalıştırır. Butona her basışta bir komut ilerletir.

Reset: Simülasyonu tümüyle durdurup PIC'i resetler.

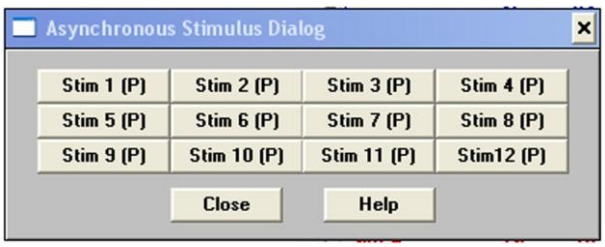
Simülasyona başlamadan önce

- Windows menüsünden "Program Memory" seçilerek programın assembly ve hex kodlarını listeleyen pencere açılır.
- Windows menüsünden "Special Function Register" seçerek program içerisinde kullandığımız registerleri bulunduğu pencre açılır.
- Simülasyon süresince bize gerekli olan ve devamlı açık kalması gereken Windows menüsünden "Stopwatch" penceresi açılır.



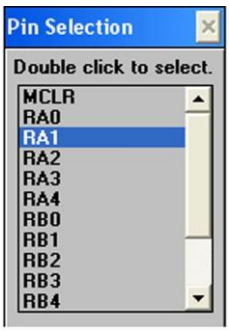
Şekil 1.12: Simülatörün çalışma adımları

- Butona her basışta sonraki komutun devreye girdiğini ve hangi komut çalışıyorsa o komutun yer aldığı satırın seçili olduğu Şekil 1.12'de görülebilir.
- Devrede "START" butonuna basıldıktan sonra devre çalışmaya başlamaktadır. Simülasyon "STEP" butonuna her basışta butona basılıyormuş gibi işlem yapar. Buton tanımlamak için "Debug" menüsünden "Simulator Stimulus" seçeneğinden "Asynchronous Stimulus"u seçilen Şekil 1.13'teki pencere açılır.



Şekil 1.13: Debug diyalog penceresi

- Bu pencerede "Stim1-12" tuşlarına port giriş/çıkış uçlarına bağlı olan butonlar atanabilir. Bu girişleri high, low ya da toggle yapmak mümkündür.
- "Stim 1" butonuna RA1 butonuna atamak için sağ tuşa tıklanır."Assign Pin…" seçilir ve Şekil 1.14 penceresinden RA1 seçilir. Böylece START butonu simülasyona dâhil edilir.

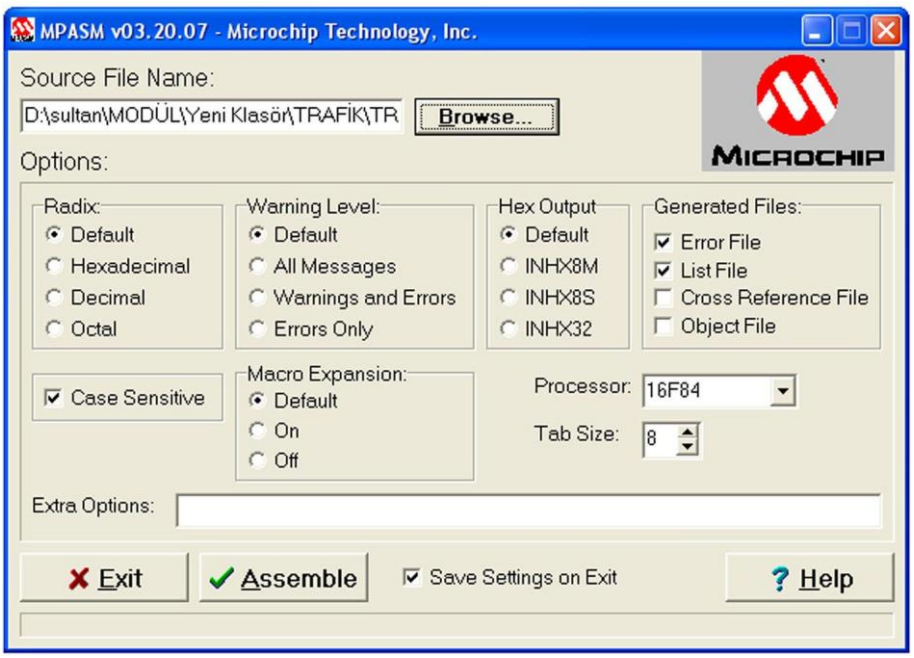


Şekil 1.14: Pin seçimi

- "Debug" menüsünden "Run" ve "Animate" komutu seçilir. Bu komutla program hızlı bir şekilde çalıştırılır.
- Program içerisinde kullanılan tüm registerlerin içeriği "Special Function Register" penceresinden izlenebilir.

1.1.6.3. Programın PIC'e Yazdırılması

PIC'e program yazmak için asm kodunun derlenip hex dosyasına çevrilmesi gerekir. Bunun için MPLAB programı ya da MPASM programı kullanılır. Şekil 1.15'te programın çalışma ekranı görülmektedir.



Şekil 1.15: MPSAM programı

- Radix bölümünde "Hexadecimal" seçilir.
- Warning bölümünde "Warnings and Errors" seçilir.
- Hex output bölümünde "INHX8M" seçilir.
- Generated Files bölümünde "Error File" , "List File" seçilir.
- Case Sensitive kutusu onaylanır.
- Macro Expansions bölümünde "Default" seçilir.
- Processor penceresinden "16F84" seçilir.
- Tab size penceresine "8" yazılır.
- Browse butonuna tıklanıp derlenecek dosya seçilir.
- Programı derlemek için "Assemble" butonuna tıklanır.
- Programda hata yoksa "Assemly successful" mesajı alınır.
- Programda hata varsa " Errors Found" mesajı alınır. Hatalar ERR uzantılı dosyaya kaydedilir.

Programı PIC'e yazmak için PIC programlayıcıya ve PIC programlayıcı yazılımına ihtiyaç olacaktır. PIC'leri programlamak için çok çeşitli elektronik devreler kullanmaktadır. Kullanılan karta uygun olarak da yazılım kullanmak gerekmektedir.

1.2. Merdiven Otomatiği Uygulama Devresi

Dört girişli merdiven otomatiğinde hangi girişten sinyal verilirse verilsin bir çıkış 1 dk. süreyle aktif olacak, lambaları kumanda etmek için transistör ve röle kullanılacak, 1 dk. sonunda ise lambalar sönecektir. Sistem dört girişli olduğundan dört tane buton kullanılacaktır. Hangi butona basılırsa basılsın bütün lambalar yanacak ve 1 dk. sonunda

sönecektir. Merdiven otomatiği turn –off (gecikmeli sönen) olarak çalışan zamanlayıcı devresidir.

Zamanlayıcı devreleri bir sistemi önceden belirlenmiş bir zaman sonunda, zamana bağlı fonksiyonları yerine getiren devrelerdir. Bu tip zamanlayıcı devreleri birçok alanda kullanılmaktadır.

En basit zaman kontrolü iki transistörlü bir multivibratörle yapılabilir. Zaman ayarı bir potansiyometre ile yapılır. Bu tip zamanlayıcılar merdiven otomatiği devrelerinde kullanılır. Bu tip bir multivibratörün en büyük mahzuru zamanın hassas olmamasıdır. Zaman kat sayısı τ = RC bağlıdır. Her iki eleman da ısıya bağlı olarak değiştiklerinden zamanda ortam ısısıyla değişir. Bir başka mahzur da, sürenin uzaması için yüksek değerli kondansatör veya direnç kullanmakla akımın azalması ve kondansatörün hiç dolmamasıdır.

Biraz daha kararlı ve basit zamanlayıcı 555 entegresidir. 555, içinde opamp ve zamanlayıcı için gerekli elemanları barındıran 8 bacaklı bir entegredir. Çok küçük zaman birimlerinden dakika ölçekli zaman aralıklarına kadar transistörlü multivibratörden daha güvenle kullanılabilir. Ama hassasiyet konusunda gene RC devresinin ısıl kararlılığı kadar güvenilirdir. Kararlılıktan anlamamız gereken her çalıştığında, 1 dakikalık tasarlanmış bir zamanlayıcının hep bir dakika sonunda işlevini yerine getirmesidir.

Kesin bir hassasiyet gerektiren işlerde örneğin, verici ve alıcı devrelerinde kullanılmak üzere kristal kontrollü osilatörler geliştirilmiştir. Quartz kristaller rezonans frekansına göre kesilmiş ve iki metal plaka arasına hapsedilmiş piezoelektrik etkisiyle çalışan elemanlardır. Kristale basınç uygulandığında titreşim yaparak gerilim üretir veya değişken gerilim uygulandığında titreşim yaparak dış ortama basınç uygular işte bu özelliğe "piezoelektrik" denir.

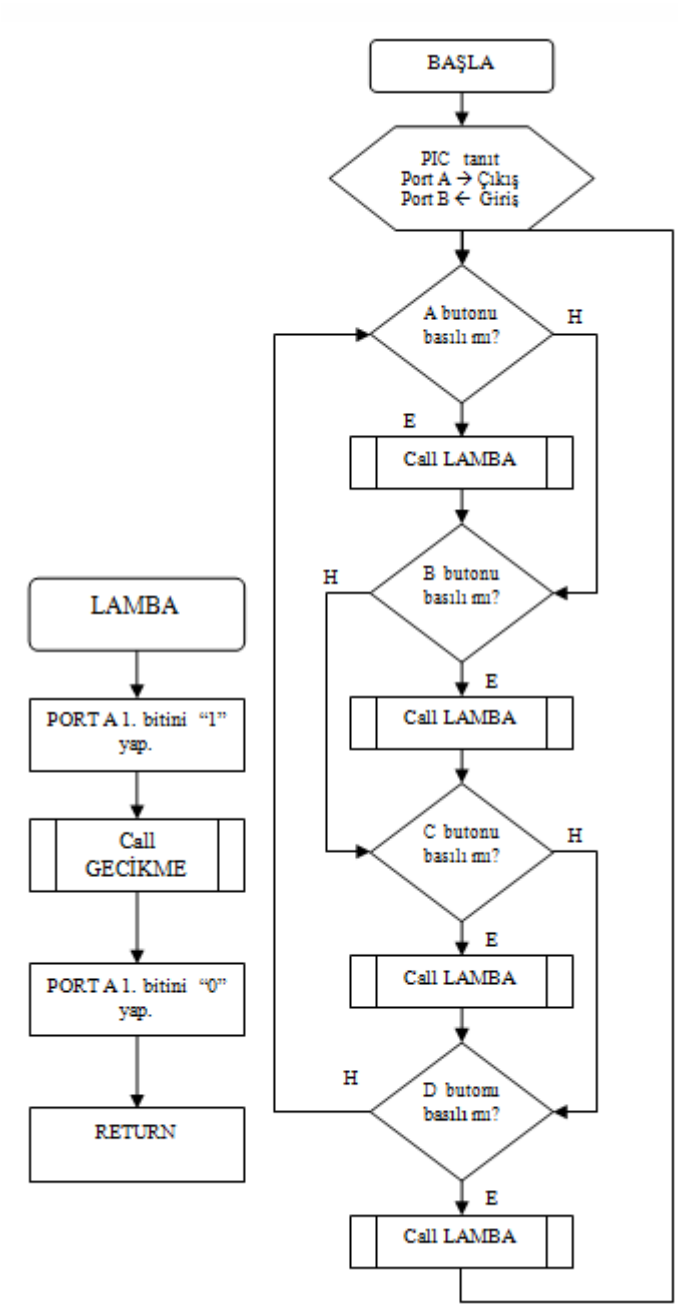
Bir kristal ile kurulmuş osilatörün ısıl kararlılığı bir RC osilatörüne göre çok daha fazladır. Multivibratör ve 555 osilatörlerinin salınım frekansı bir potansiyometre yardımıyla değiştirilebilir fakat kristalin frekansını kristalin kesildiği değer dışında başka salınıma ayarlamak mümkün değildir. Kristal kontrollü bir zamanlayıcı tasarlanmak istendiğinde bu osilatörün çıkışı dijital programlanabilir bölücülerle bölünür. Bu bölme işlemi titreşimlerin frekansını bölme oranı kadar azaltır, bu da darbe genişliğinin zamanının uzamasına sebep olur. Pratikte yeterli bölücü kullanım ile 1 saat, 1 gün, 1 yıl ve ötesi kadar zaman gecikmesi veya başka bir deyişle zamanlama programı elde edilir.

1.2.1. Devrenin Malzemeleri

- PIC16F84 4MHz mikrodenetleyici
- 4MHz kristal
- 4 x Buton
- C1 = C2 = 22pf
- R1 = R2 = R3 = R4 = 10KΩ direnç
- R5 = 1KΩ, R7 = 100Ω
- D1 = 1N4148
- Q1 = BC548
- 12V RÖLE

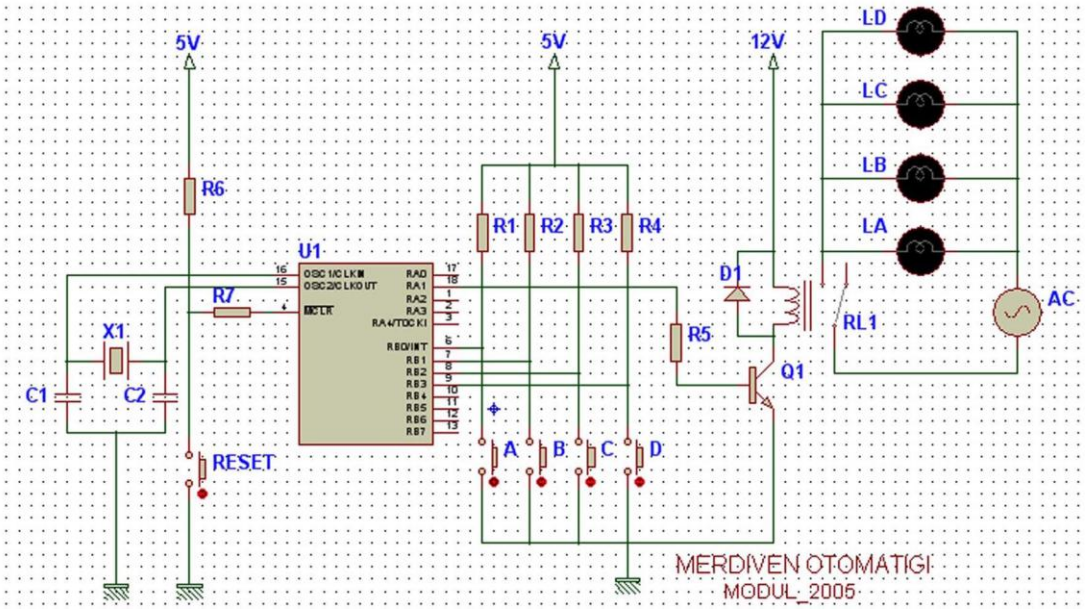
1.2.2. Akış Diyagramı

Ana programda butonlar basılıp basılmadığı konrol edilir. Herhangi bir butona basıldığında "LAMBA" alt programı çağrılır. Lambalar 1 dk. yanıp söndükten sonra ana programa dönülür ve butona basılıp basılmadığı kontrol edilir.



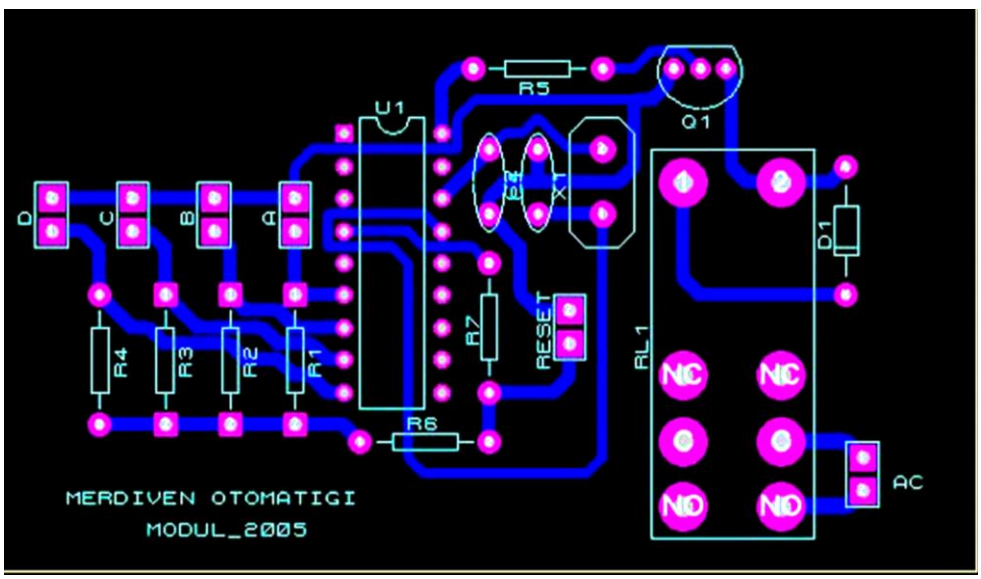
Tablo 1.4: Merdiven otomatiği akış diyagramı

1.2.3. Devre Şeması



Şekil 1.16: Merdiven otomatiği devresi

1.2.4. Baskı Devresi



Şekil 1.17: Merdiven otomatiği baskı devresi

1.2.5. Devrenin Asm Programı

```
;=======MERDİVEN OTOMATİĞİ UYGULAMA PROGRAMI=====S_2005=====
        LIST P=16F84
        INCLUDE "P16F84.INC"

        SAYAC1  EQU  H '0C'          ; Gecikme alt programı değişkeni
        SAYAC2  EQU  H'0D'           ; Gecikme alt programı değişkeni
        SAYAC3  EQU  H'0E'           ; Gecikme alt programı değişkeni

        BSF     STATUS, 5            ; BANK1'e geç
        CLRF    TRISA                ; PORTA çıkış
        MOVLW   H'FF'                ; W <-- H'FF'
        MOVWF   TRISB                ; PORTB    giriş
        BCF     STATUS,5             ; BANK0'ya geç
        CLRF    PORTA                ; PORTA temizle

A1                                   ; Butonları kontrol et.
        BTFSC    PORTB,0
        GOTO    B1                   ; A butonuna basılı mı?
        CALL    LAMBA
B1
        BTFSC   PORTB,1
        GOTO    C1                   ; B butonuna basılı mı?
        CALL    LAMBA
C1
        BTFSC    PORTB,2
        GOTO    D1                   ; C butonuna basılı mı?
        CALL    LAMBA
D1
        BTFSC   PORTB,3
        GOTO    A1                   ; D butonuna basılı mı?

DONGU                                ; Sonsuz döngü
        GOTO    DONGU

;============LAMBA ALT PROGRAMI==============

LAMBA
        BSF     PORTA,1              ; PortA çıkışını aktif yap
        CALL    GECIKME
        BCF     PORTA,1              ; PortA çıkışını pasif yap

        RETURN
;=============GECİKME ALT PROGRAMI=====================
```

```
GECİKME                         ; 60 sn lik gecikme alt programı

        MOVLW   d'255'          ; W<--D'255'
        MOVWF   SAYAC1          ; SAYAC1 <-- W

G1
        MOVLW   d'255'          ; W<--D'255'
        MOVWF   SAYAC2          ; SAYAC2 <-- W
G2
        MOVLW   d'255'          ; W<--D'255'
        MOVWF   SAYAC3          ; SAYAC3 <-- W
G3
        DECFSZ  SAYAC3,F
        GOTO    G3
        DECFSZ  SAYAC2,F
        GOTO    G2
        DECFSZ  SAYAC1,F
        GOTO    G1
        RETURN

        END
```

1.3. Dört Aboneli Numaratör Uygulama Devresi

1.3.1. LCD 'nin Yapısı ve Çalışması

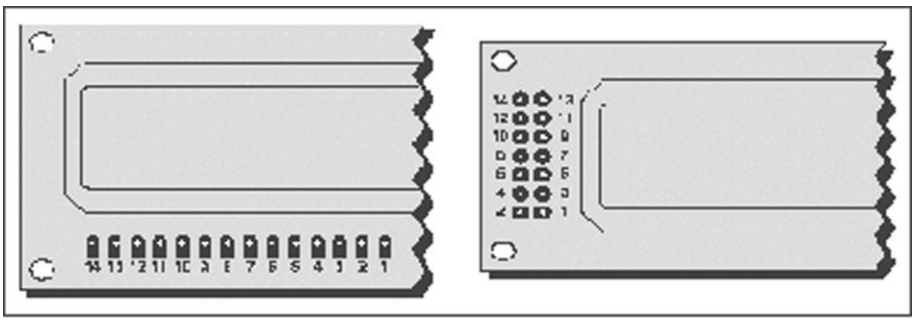
1.3.1.1. Yapısı

7 parçalı gösterge ile sayıların dışında diğer karakterleri, harfleri, işaretleri elde etmek imkânsızdır. Bundan dolayı karakterleri gösterecek yeni bir hücre yapısına yani LCD'ye ihtiyaç vardır. LCD'ler kendi içlerinde çeşitlilikler göstermesine rağmen temel olarak nokta matrisli ve grafik ekranlı olarak ikiye ayrılır. Kullanım olarak grafik ekran LCD'ler ile daha fazla işlem yapılabilmesine rağmen kullanımda nokta matrisli LCD'ler kolaylık sağlamaktadır.

LCD'ye veri yazmak için ekranı kontrol eden işlemcilere ihtiyaç duyulur. Üreticiler çoğunlukla HITACHI HD44780 ve bir firmanın1602D entegrelerini LCD sürücü entegresi olarak kullanmaktadır. LCD göstergeler 1x8, 1x16, 2x16, 4x20, 1x40, 2x40 satır/karakter olarak üretilmektedir. Modüldeki uygulama devrelerinde 2x16 LCD kullanılmıştır.

Şekil 1.18: 2x40 LCD

Standart LCD modüllerinde 8 veri hattı, 3 kontrol hattı ve 3 güç hattı içeren 14 tane pin vardır. Bazı LCD modüllerinde 16 pin bulunur. Bağlantılar, genelde 7 pinlik iki sıra, ya da 14 pinlik tek sıra şeklinde yapılmıştır. Her iki yerleştirme Şekil 1.19'da görülmektedir.



Şekil 1.19: LCD'nin bacak yapıları

Displaylerin çoğunda pinler, LCD' nin baskı devresi üzerinde numaralandırılmıştır. Eğer numaralandırılmamışsa PIN 1'in yerini bulmak oldukça kolaydır. Bu pinin toprağa bağlanmasından dolayı sıklıkla toprağa bir bağlantı yolu vardır ve bir yerlerden genellikle metal çerçeveye bağlanmıştır. Bu bağlantılardan her birinin fonksiyonu Tablo 1.5'te gösterilmiştir.

LCD Pinleri	**İşlevleri**
1	Toprak (Vss- Ground)
2	Besleme (Vcc ya da Vdd +5 Volt)
3	Aydınlatma (VO ya da Vee- Kontrast)
4	Kaydedici Seçici (RS - Register Select)
5	Oku / Yaz (R/W - Read / Write)
6	Yetki (E - Enable)
7-14	Veri Girişleri (D0-D7 - DATA)
15*	LCD Panel ışığı (+5 Volt)
16*	Toprak (Ground)

Tablo 1.5: LCD pin isimleri ve fonksiyonları

- **VO (Kontrast):** Displayin kontrastını (parlaklık) ayarlamaya yarayan bir kontrol ucudur. 1 K'lık bir direnç üzerinden toprağa bağlanır. Buradaki direncin değerini artırarak kontrastı düşürülebilir, azaltarak yükseltilebilir. 1 K'lık ayarlı direnç bağlayarak isteğe bağlı olarak değiştirilebilir.
- **RS (Register Select):** RS girişi "lojik 0" olduğunda LCD'ye gönderilen veri komut olarak algılanır. Lojik 1 olduğunda veri karakter veya sayı olarak algılanır.
- **R/W (Read/Write):** Bu giriş "lojik 1" olduğunda LCD'den veri okuma, "lojik 0" olduğunda LCD'ye veri yazma işlemi yapılır.

- **E (Enable):** Bu giriş aktif yapılınca mikrodenetleyiciden LCD'ye veri gönderilebilir. E bacağının lojik 1'den lojik 0'a geçişi ile LCD'ye veri transferi yapılır. Lojik 0'dan lojik 1'e geçmesi ile LCD'den veri okunur.
- **D0-D7 (Data):** LCD'nin veri girişleridir. LCD'ye 8 bitlik veya iki 4 bitlik veri transferi yapılabilir. 4 bit modda sadece üst 4 bit kullanılır.

1.3.1.2. LCD' nin Çalışması

Tablo 1.6'da standart LCD panelin komut kaydedicisinin fonksiyonları gösterilmiştir. Aşağıdaki komutları kullanabilmek için önce komut kaydedicisi (RS=0) seçilmelidir.

- ID= 1 gösterge yazdıktan sonra kursörü 1 artır. S= 1 gösterge kayar.
- ID= 0 gösterge yazdıktan sonra kursörü 1 azalt. S= 0 gösterge kaymaz.
- BF göstergenin hazır olup olmadığını gösteren bayraktır.

  - BF = 1 gösterge meşgul, komut kabul edilmez.
  - BF = 0 komut kabul edilir.

Standart LCD Panel Komut Kaydedicisi Fonksiyonları									
İşlem	D7	D6	D5	D4	D3	D2	D1	D0	Açıklama
Ekranı Sil	0	0	0	0	0	0	0	1	Ekranı siler.
Ekran seçenekleri	0	0	0	0	1	D	C	B	(D) GÖSTERGE AÇ (1) / KAPA (0) (C) Kürsör AÇ (1) / KAPA (0) (B) Yanıp Sönme AÇ (1) / KAPA (0)
Fonksiyon ayarları	0	0	1	D	N	0	X	X	(D) Veri girişi 8-bit (1) / 4-bit (0) (N) Çift satır (1) / Tek satır (0)
CGRAM Adres ayarı (ilk satırın adresleri)	0	1	C5	C4	C3	C2	C1	C0	(C5-C0) CGRAM'e gidecek veriler.
DDRAM Adres ayarı (ikinci satırın adresleri)	1	D6	D5	D4	D3	D2	D1	D0	(D6-D0) DDRAM'e gidecek veriler.
Busy Flag / Adres sayaç okuma	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Meşgul bayrağı (BF) RW=1, RS="0" iken D7 pininden okunur. D6-D0 dan adres counter verisi okunur.
Cursor Shift	0	0	0	1	SC	RL	X	X	Shift (1) / Cursor (0) S/C kursör hareketi ve gösterge kayması. Right (1) / Left (0) R/L kayma yönü

Entry Mode Set	0	0	0	0	0	1	I D	S	Increment (1), Decrement (0) Shift (1) on (0) off Kursörün hareket yönü (I/D) ve göstergenin kayma yönü (S)

Tablo 1.6: LCD gösterge komutları

- LCD göstergeler (display) 8-bit ve 4-bit olarak çalışır. 4094, 74164 gibi kaydıran kaydediciler (shift register) entegreleri kullanılarak göstergeler 1-bit olarak da çalıştırılır. 8-bit ile çalıştığında 11 tane bağlantı (8 tane veri/komut, RS, RW, E), 4- bit ile çalıştığında 7 tane bağlantı ile göstergeyi denetlemek gerekir. LCD'den herhangi bir veri okunmadığında RW bacağı lojik 0'a bağlanır.
- LCD göstergeye komut veya veri gönderirken her komut veya veri arasında bir miktar beklememiz (gecikme döngüsü) gerekmektedir. BF bayrağı kullanılırsa bu bekleme daha kolaylıkla yapılabilir.
- Bu uygulama devresinde LCD 4-bit modda kullanılmıştır. LCD'ye güç kaynağı bağlanınca LCD içerisindeki elektronik devre reset yapar. Fakat bazı durumlarda bu reset devresi tam olarak çalışmayabilir ve LCD'yi programla reset yapmak gerekir. Aşağıda LCD'yi reset yapmak için gerekli adımlar verilmiştir.

  - İlk olarak 15 ms beklenir.
  - LCD'ye H'30' komutu 8-bit olarak gönderilir.
  - 5 ms beklenir.
  - Yine LCD'ye H'30' komutu 8-bit olarak gönderilir.
  - 160 µs beklenir.
  - 3. defa LCD'ye H'30' komutu 8-bit olarak gönderilir.
  - 160 µs beklenir (Bu noktada BF bayrağı kullanılabilir).

- LCD resetlendikten sonra aşağıdaki adımlar isteğe göre LCD'yi ayarlanabilir.

	RS	**RW**	**D7**	**D6**	**D5**	**D4**
• Function Set (4-bit ile çalışmayı belirle)	0	0	0	0	1	0
• Function Set (4-bit çalışma ve satır sayısı)	0	0	0	0	1	0
	0	0	N	0	*	*
• Display On/Off Control	0	0	0	0	0	0
(Gösterge ve Kursörü aç)	0	0	1	1	1	0
• Entry Mode Set	0	0	0	0	0	0
(Artan kursör ve gösterge kaymaz)	0	0	0	1	1	0

- Bu noktadan itibaren veriyi istenilen satıra ve sütuna yazılabilir. 1. satırın başlangıç adresi h'80' dir. 1. satırda 1X16 LCD'de 16 sütun bulunur. 2. satırın başlangıç adresi de h'C0'dır. LCD resetlendiğinde 1. satırın 1. sütunundan verileri yazmaya başlar. Tablo 1.7'de LCD'nin adresleri bulunur.

80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF

Tablo 1.7: LCD adresleri

- **Busy flag (Meşgül Bayrağı) test**: LCD işlem yaparken BF biti lojik-1 olur. Bu durumda başka bir komut göndermek hatalara sebep olur. Bunu engellemenin iki yolu vardır: Gecikme döngüsü kullanmak ya da BF bitini test etmektir. BF bitini test etmek için çıkış olan pinler önce giriş yapılacak daha sonra veri gönderebilmek için tekrar çıkış yapılacaktır.

  - RS = 0 RW = 0 E= 1 → 0 komut yazma
  - RS = 0 RW = 1 E= 1 → 0 komut okuma (BF test)
  - RS = 1 RW = 0 E= 1 → 0 veri yazma

Numaratör devresinde 4 abone bulunmaktadır. Bu abonelerden hangisi çağrı yaparsa LCD'de abonenin kendisine ait olan kısmında abonenin ismi yazılırken bir led yanar ve bir buzzerden sesli uyarı yapılır. Buzzer ve led yaklaşık 30 sn. aktif olur. Aboneler aynı anda çağrı yaparsa çağrı yapan tüm abonelerin ismi LCD'de gösterilir**.** Aboneler belli bir süre sonra çağrı yaparsa LCD silinir ve son çağrı yapan abonenin ismi göstergede kalır.

LCD'nin ilk satırına "ABONE 1" ve "ABONE 3" yazdırılır. İkinci satırına ise "ABONE 2 " ve "ABONE 4" yazdırılır.

1.3.2. Devrenin Malzemeleri

- PIC16F84 4MHz mikrodenetleyici
- 4MHz kristal , 4xbuton, led
- C1 = C2 = 22pf
- R2 = R3 = 1KΩ direnç
- R1 = R4 = R5 = R6 = R7 = 10KΩ
- RV1 = 1K potansiyometre
- Q1 = BC237 , Q2 = BD135
- 12V Röle , D1= 1N4148

Programda öncelikle portlar ayarlanır. Port A çıkış, Port B yüksek bitler giriş, düşük bitler çıkış olarak ayarlanmıştır. Akış diyagramında kullanılan alt programlar ve görevleri şunlardır:

- **LCD_AYAR:** LCD'yi kullanmadan önce bir defa çağrılır.
- **LCD_RESET:** LCD ilk çalıştırıldığında resetlenmelidir. Bunun için H'00' bilgisi LCD'ye 3 defa 8 bit olarak gönderilir.

- **FUNCTION_SET:** LCD 4 bit olarak kullanıldığından LCD'ye H'02'komutu ardından da H'28' komutu 4 bit olarak gönderilir. Bunun için yüksek ve düşük bitler bölünerek gönderilir. H'02' ve H'08' şeklinde gönderilir.
- **DISPLAY_ON:** Display açık, kursör ve yanıp sönme kapalı olduğundan (Kullanıcıya göre değişebilir). LCD'ye H'0C' bilgisi 4 bit olarak gönderilir.
- **ENTRY_MODE:** Bir karakter yazıldıktan sonra adresin otomatik olarak artması için ve kursörün kayması iptal edilmesi için LCD'ye H'06' bilgisi 4 bit olarak gönderilir.
- **LCD_KOMUT:** LCD'ye komut yazdırılırken bu alt program çağrılır. Komut yazılırken RS = 0, RW = 0 ve E = 1→ 0'a değişmelidir.
  - BF_TEST: Bu alt program kullanılmadığında LCD'ye veri ve komut yazarken gecikme döngüleri kullanılmalıdır. BF (Busy Flag) LCD işlem yaparken lojik 1 olur. BF bitini okumak için LCD'ye bağlı olan port A uçları giriş olarak değiştirilir. Okuma işlemi için RS = 0, RW = 1 ve E = 1→0 olmalıdır. BF biti LCD'nin D7 bacağından okunur. Okunan değer "1" ise LCD meşgul demektir ve BF tekrar test edilir. Okunan değer "0" olduğunda LCD meşgul değildir ve program alt programın çağrıldığı yerden devam eder.

- **MESAJ 1:** Programda dört ayrı kullanıcı olduğundan dört ayrı mesaj bulunur. Hepsi aynı olduğundan yalnızca biri açıklanmıştır. Mesaj alt programı çalıştığından dolayı A1 butonu basılı durumdadır. İlk önce Buzzer ve led aktif yapılır. (lojik 1)
  - *LCD_SIL:* LCD' ye H'01'bilgisi gönderilerek LCD silinir ve kursör 1.satırın başına alınır.
  - *SATIR_1W:* 1.satırın başlangıç adresi H'80'dir. Kursörü 1. satırda istenilen sütuna götürür.
  - *SATIR_2W:* 2.satırın başlangıç adresi H'C0' dır. Kursörü 2. satırda istenilen sütuna götürür.

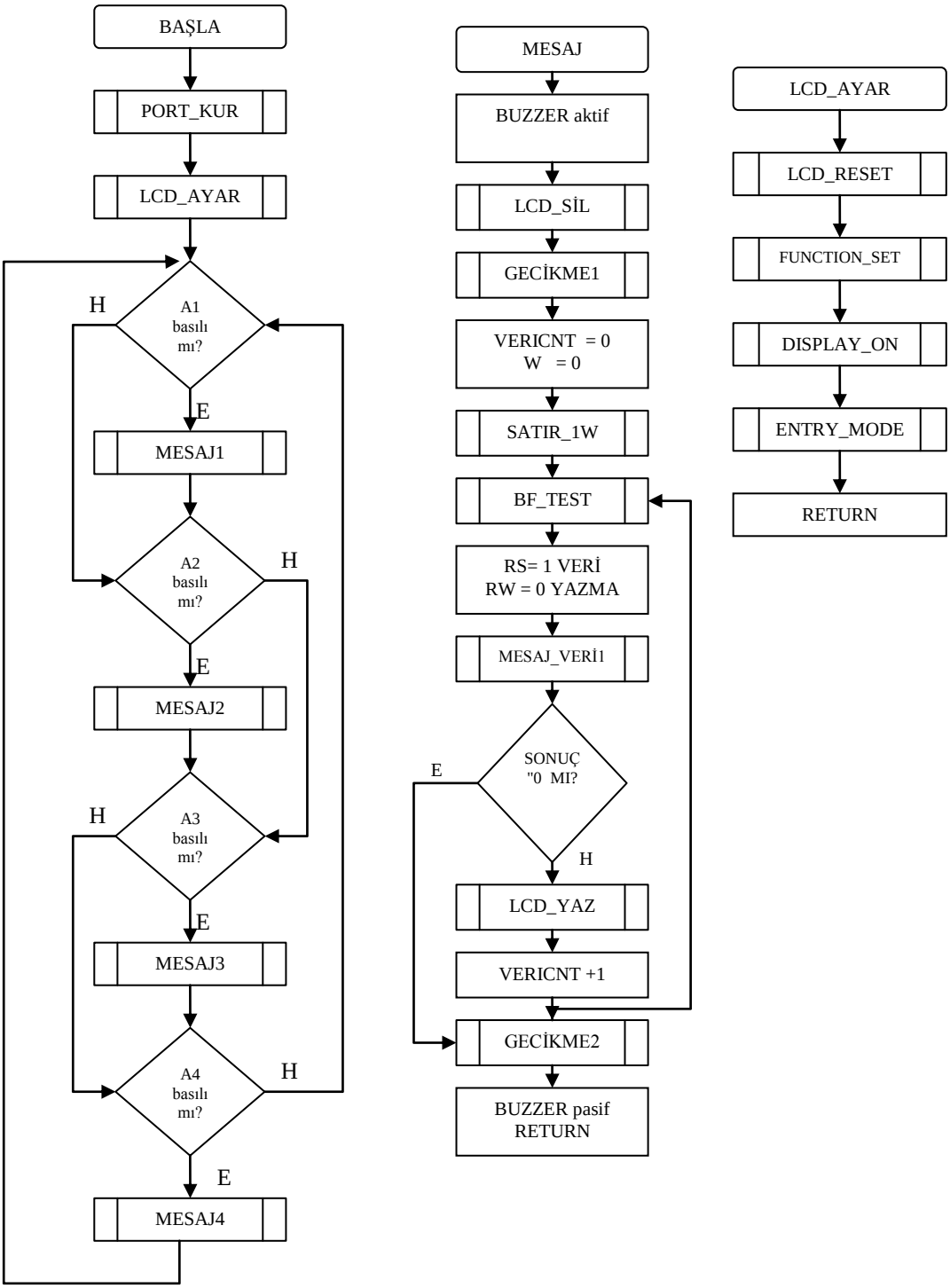
LCD'de mesajı yazmak için satır ve sütun belirlendikten sonra BF biti test edilir.

- *MESAJ_VERISI1:* Ekrana yazdırılacak olan mesaj için bu alt program çağrılır. W registerindeki değer kaç ise o satırdaki bilgi W register ile geri gönderilir. Sayı verisi göndermek için hexadesimal kodu yazılır. Örneğin, H'34' yazıldığında ekranda "4" görünür.

Gönderilen değer "0 " ise alt programdan çıkılır. "0" değilse LCD'ye veri gönderilmeye devam edilir.

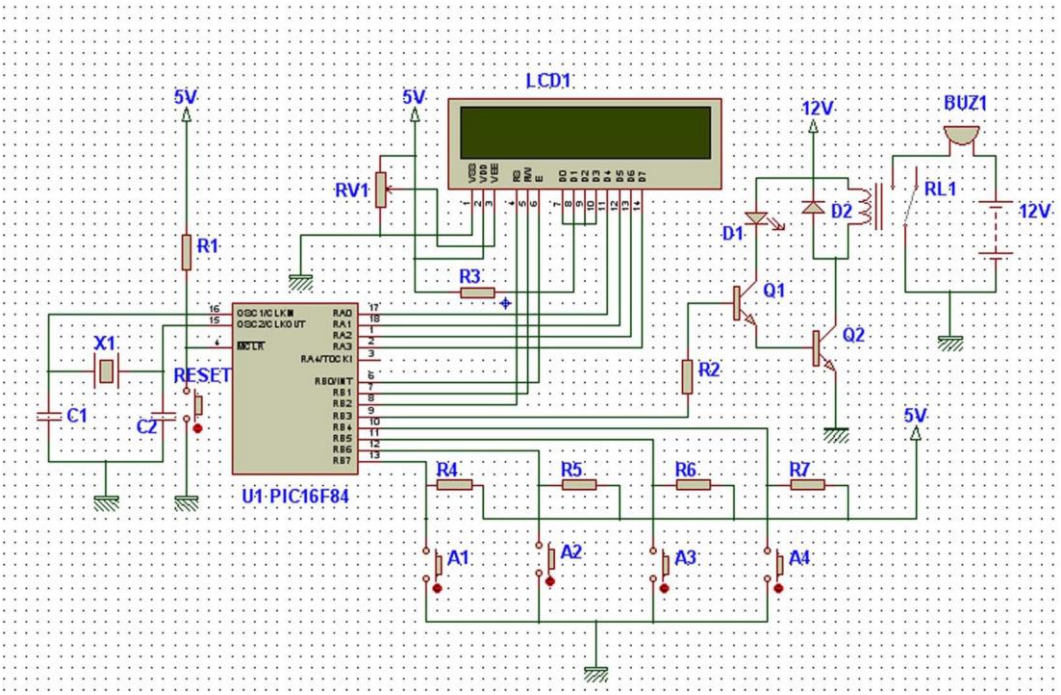
- *LCD_YAZ:* W registerine yüklenen veri LCD'ye 4 bit olarak gönderilir. (Yüksek ve düşük bitler SWAP komutuyla yer değiştirilir. Düşük bitler sıfırlanır. Yüksek bitler LCD'ye gönderilir. Yüksek ve düşük bitler tekrar yer değiştirilir. Yüksek bitler sıfırlanır ve düşük bitler LCD'ye gönderilir.)
- Bunun için RS = 1 , RW=1 ve E = 1→ 0 olarak değiştirilir.

1.3.3. Akış Diyagramı



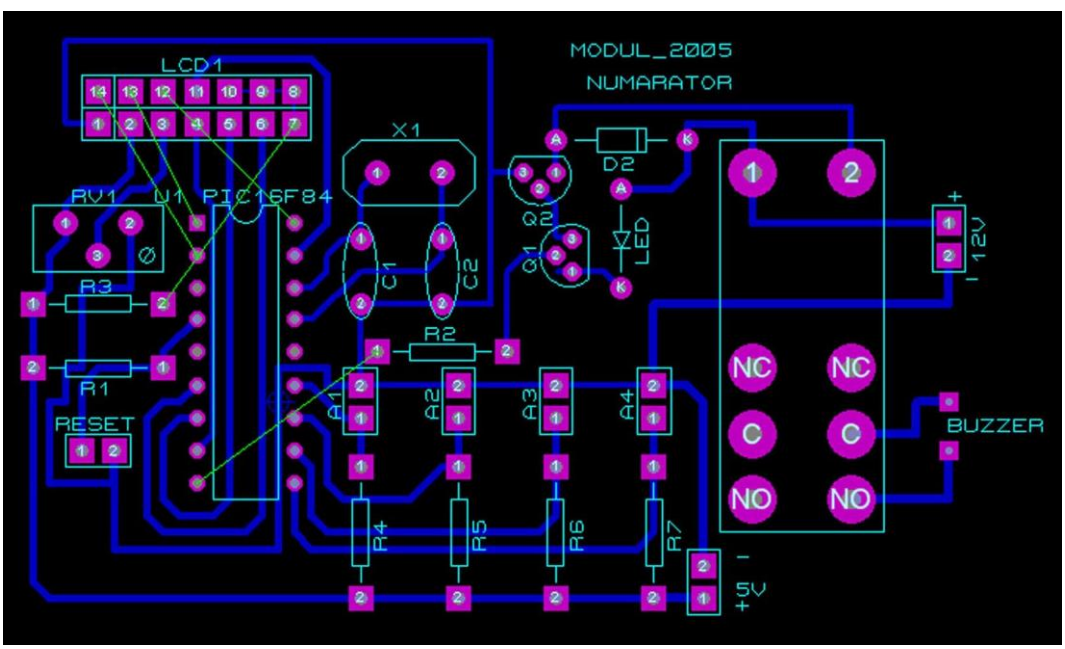
Tablo 1.8: Numaratör akış diyagramı

1.3.4. Devrenin Şeması



Şekil 1.20: Numaratör uygulama devresi

1.3.5. Baskı Devresi



Şekil 1.21: Numaratör baskı devresi

1.3.6. Devrenin Asm Programı

Diğer programlardan farklı olarak DEFINE komutu kullanılmıştır. Bu komut bit adreslemesini tanımlamayı sağlar.

```
#DEFINE   RS   PORTA , 2  →   RS   EQU   2
 BSF      RS                →   BSF    PORTA, 2
```

```
;=====NUMARATÖR DEVRESİ UYGULAMA PROGRAMI======S_2005=====

            LIST P=16F84
            INCLUDE "P16F84.INC"
#DEFINE     RS          PORTB,2
#DEFINE     RW          PORTB,1
#DEFINE     EN          PORTB,0
SAYAC1      EQU         H'11'           :Gecikme döngüsü değişkeni
SAYAC2      EQU         H'12'           : Gecikme döngüsü değişkeni
SAYAC3      EQU         H'13'           : Gecikme döngüsü değişkeni
VERICNT     EQU         H'14'
TEMP        EQU         H'15'
TEMP0       EQU         H'16'
TEMP1       EQU         H'17'
CX          EQU         H'18'

BASLA;.....

            CALL        PORT_KUR        : Portları kur
            CALL        LCD_AYAR        :LCD ilk kullanım için ayarla
TUS_ARA
            BCF         PORTB,3         :Buzzer ve led lojik 0
            BTFSS       PORTB,7         :A1 butonuna basılı mı?
            CALL        MESAJ1          :MESAJ1'e git
            BTFSS       PORTB,6         :A2 butonuna basılı mı?
            CALL        MESAJ2          :MESAJ2'ye git
            BTFSS       PORTB,5         :A3 butonuna basılı mı?
            CALL        MESAJ3          :MESAJ3'e git
            BTFSS       PORTB,4         :A4 butonuna basılı mı?
            CALL        MESAJ4          :MESAJ4'e git
            GOTO        TUS_ARA         :Butona basılı mı tekrar kontrol et?
TEKRAR
            GOTO        TEKRAR
```

```
PORT_KUR;.....
            MOVLW    H'F0'
            TRIS     PORTB          : PORTB → RBH (Giriş), RBL (Çıkış)
            MOVLW    H'00'
            TRIS     PORTA          : PORTA → Çıkış
            CLRF     VERICNT        : VERICNT ← 0
            RETURN

LCD_AYAR ;.....
            CALL     LCD_RESET      : LCD Resetle
            CALL     FUNCTION_SET   : 4 bit mod ve iki satır aktif
            CALL     DISPLAY_ON     : Ekranı aç ve kursörü kapat
            CALL     ENTRY_MODE     : Kürsör 1 artan mod
            RETURN

LCD_RESET;.....
            CALL     GECIKME1
            MOVLW    H'03'
            MOVWF    CX
RESET       MOVLW    H'00'
            MOVWF    PORTA
            CALL     LCD_KOMUT
            DECFSZ   CX,1           :CX ← CX -1
            GOTO     RESET
            RETURN

FUNCTION_SET;.....
            MOVLW    H'02'          : 4 bit mod
            MOVWF    PORTA
            CALL     LCD_KOMUT
            MOVLW    H'02'          : 4 bit mod
            MOVWF    PORTA
            CALL     LCD_KOMUT
            MOVLW    H'08'          : 2 satır
            MOVWF    PORTA
            CALL     LCD_KOMUT
            RETURN
```

```
DISPLAY_ON;.....
            MOVLW    H'00'
            MOVWF    PORTA
            CALL     LCD_KOMUT
            MOVLW    H'0C'          : LCD on
            MOVWF    PORTA
            CALL     LCD_KOMUT
            RETURN

ENTRY_MODE;.....
            MOVLW    H'00'
            MOVWF    PORTA
            CALL     LCD_KOMUT
            MOVLW    H'06'          : Kursör 1 artan mod
            MOVWF    PORTA
            CALL     LCD_KOMUT
            RETURN
LCD_SIL;.....

            MOVLW    H'00'
            MOVWF    PORTA
            CALL     LCD_KOMUT
            MOVLW    H'01'          : Ekranı temizle, kursör 1. satır 1.sütunda
            MOVWF    PORTA
            CALL     LCD_KOMUT
            RETURN
LCD_YAZ;.....
            MOVWF    TEMP           : TEMP ← W
            SWAPF     TEMP,F        : Üst 4 bit ile alt 4 biti yer değiştir
            MOVF     TEMP,W         : Üst 4 biti al
            MOVWF    PORTA          : LCD'ye gönder
            BSF      EN             : E ← 1
            NOP                     : Bekle
            BCF      EN             : E ← 0
            CALL     BF_TEST        : Meşgul bayrağını test et
            BSF      RS             : RS → 1 Veri
            BCF      RW             : RW → 0 Yaz
            SWAPF    TEMP,F         : Yazılacak verinin tekrar üst 4 biti
            MOVF     TEMP,W         : alt 4 bitini yer değiştir
            MOVWF    PORTA          : LCD ye gönder
            BSF      EN             : E ← 1
            NOP                     : Bekle
            BCF      EN             : E ← 0
            RETURN
```

```
LCD_KOMUT;.....
            CALL    BF_TEST         : BF test et
            BCF     RS              : RS → 0 Komut
            BCF     RW              : RW → 0 Yaz
            BSF     EN              : E ← 1
            NOP                     : Bekle
            BCF     EN              : E ← 0
            RETURN

BF_TEST;.....
            BSF     STATUS,5        :Bank 1'e geç
            MOVLW   H'FF'
            MOVWF   TRISA           : PORTA ← Giriş
            BCF     STATUS,5        : Bank 0 geç
            BCF     RS              : Veri
            BSF     RW              : LCD Oku
            BSF     EN              : E← 1
            MOVF    PORTA,W         : LCD'den gelen bilgileri W  aktar
            BCF     EN              : E ← 0
            ANDLW   H'F0'           : Üst 4 biti sıfırla
            MOVWF   TEMP1           : TEMP1'de sakla
            SWAPF   TEMP1,F         : Alt 4 bit ile üst 4 biti yer değiştir
            BSF     EN              : E ← 1
            GOTO    $+1             : Bir alt satıra git
            BCF     EN              : E ← 0
            BTFSC   TEMP1,7         : BF = 1 mi?
            GOTO    BF_TEST         : Evet , BF'yi tekrar test et
            BSF     STATUS,5        : Hayır
            CLRF    TRISA           : PORTA ←  Çıkış
            BCF     STATUS,5
            RETURN

SATIR1W;.....

            MOVWF   TEMP0           : TEMP0 ← W
            MOVLW   H'08'           : LCD 1. satırı aktif
            MOVWF   PORTA
            CALL    LCD_KOMUT
            MOVF    TEMP0,W
            MOVWF   PORTA           : 1. satırda W sütununa git
            CALL    LCD_KOMUT
            RETURN
```

```
SATIR2W;.....

            MOVWF   TEMP0               : TEMP0 ← W
            MOVLW   H'0C'               : LCD 2. satırı aktif
            MOVWF   PORTA
            CALL    LCD_KOMUT
            MOVF    TEMP0,W
            MOVWF   PORTA               : 2. satırda W sütununa git
            CALL    LCD_KOMUT
            RETURN

MESAJ1;.....
            BSF     PORTB,3             :Buzzer aktif
            CALL    LCD_SIL             : LCD temizle
            CALL    GECIKME1            : Bekle
            MOVLW   H '00'
            MOVWF   VERICNT             :VERICNT ← 0
            MOVLW   H '00'
            CALL    SATIR1W             : 1.satırın 1. sütununa git
M1          CALL    BF_TEST             : BF test et
            BSF     RS                  : RS → 1 Veri
            BCF     RW                  : RW → 0  Yaz
            MOVF    VERICNT,W           : W ← VERICNT
            CALL    MESAJ_VERISI1       : Veriyi al
            IORLW   0                   : 0 ile test et
            BTFSC   STATUS,2            : Sonuç 0 mı?
            GOTO    MESAJ1_SON          : Evet  MESAJ1_SON etiketine git
            CALL    LCD_YAZ             : Hayır veriyi LCD gönder
            INCF    VERICNT,F           : Bir sonraki veriyi adresle
            GOTO    M1                  : Veri yazma işlemine devam et
MESAJ1_SON
            CALL    GECIKME2            : Buzzer ve led için gecikme
            BCF     PORTB,3             : Buzzer pasif
            RETURN
```

```
MESAJ2;.....
            BSF     PORTB,3             :Buzzer aktif
            CALL    LCD_SIL             : LCD temizle
            CALL    GECIKME1            : Bekle
            MOVLW   H '00'
            MOVWF   VERICNT             :VERICNT ← 0
            MOVLW   H '00'
            CALL    SATIR2W             : 2.satırın 1. sütununa git
M2          CALL    BF_TEST             : BF test et
            BSF     RS                  : RS → 1 Veri
            BCF     RW                  : RW → 0  Yaz
            MOVF    VERICNT,W           : W ← VERICNT
            CALL    MESAJ_VERISI2       : Veriyi al
            IORLW   0                   : 0 ile test et
            BTFSC   STATUS,2            : Sonuç 0 mı?
            GOTO    MESAJ2_SON          : Evet  MESAJ2_SON etiketine git
            CALL    LCD_YAZ             : Hayır veriyi LCD gönder
            INCF    VERICNT,F           : Bir sonraki veriyi adresle
            GOTO    M2                  : Veri yazma işlemine devam et
MESAJ2_SON
            CALL    GECIKME2            : Buzzer ve led için gecikme
            BCF     PORTB,3             : Buzzer pasif
            RETURN

MESAJ3;.....
            BSF     PORTB,3             : Buzzer aktif
            CALL    LCD_SIL             : LCD temizle
            CALL    GECIKME1            : Bekle
            MOVLW   H '00'
            MOVWF   VERICNT             : VERICNT ← 0
            MOVLW   H '09'
            CALL    SATIR1W             : 1.satırın 9. sütununa git
M3          CALL    BF_TEST             : BF test et
            BSF     RS                  : RS → 1 Veri
            BCF     RW                  : RW → 0  Yaz
            MOVF    VERICNT,W           : W ← VERICNT
            CALL    MESAJ_VERISI3       : Veriyi al
            IORLW   0                   : 0 ile test et
            BTFSC   STATUS,2            : Sonuç 0 mı?
            GOTO    MESAJ3_SON          : Evet  MESAJ3_SON etiketine git
            CALL    LCD_YAZ             : Hayır veriyi LCD gönder
            INCF    VERICNT,F           : Bir sonraki veriyi adresle
            GOTO    M3                  : Veri yazma işlemine devam et
```

```
MESAJ3_SON
            CALL    GECIKME2            : Buzzer ve led için gecikme
            BCF     PORTB,3             : Buzzer pasif
            RETURN
MESAJ4;.....................................................................................................
            BSF     PORTB,3             : Buzzer aktif
            CALL    LCD_SIL             : LCD temizle
            CALL    GECIKME1            : Bekle
            MOVLW   H '00'
            MOVWF   VERICNT             VERICNT ← 0
            MOVLW   H '09'
            CALL    SATIR2W             : 2.satırın 9. sütununa git
M4          CALL    BF_TEST             : BF test et
            BSF     RS                  : RS → 1 Veri
            BCF     RW                  : RW → 0  Yaz
            MOVF    VERICNT,W           : W ← VERICNT
            CALL    MESAJ_VERISI4       : Veriyi al
            IORLW   0                   : 0 ile test et
            BTFSC   STATUS,2            : Sonuç 0 mı?
            GOTO    MESAJ4_SON          : Evet  MESAJ4_SON etiketine git
            CALL    LCD_YAZ             : Hayır veriyi LCD'ye gönder
            INCF    VERICNT,F           : Bir sonraki veriyi adresle
            GOTO    M4                  : Veri yazma işlemine devam et
MESAJ4_SON
            CALL    GECIKME2            : Buzzer ve led için gecikme
            BCF     PORTB,3             : Buzzer pasif
            RETURN

MESAJ_VERISI1;.....................................................
             ADDWF   PCL,F              : PCL = PCL + W ile veriyi adresle
              RETLW 'A'
              RETLW 'B'
              RETLW 'O'
              RETLW 'N'
              RETLW 'E'
              RETLW ' '
              RETLW H'31'
              RETLW 0
              MOVF   PCL,W              : İstenilen karakteri W registerine al
              RETURN
```

```
MESAJ_VERISI2;.....
                ADDWF   PCL,F           : PCL = PCL + W ile veriyi adresle
                RETLW 'A'
                RETLW 'B'
                RETLW 'O'
                RETLW 'N'
                RETLW 'E'
                RETLW ' '
                RETLW H'32'
                RETLW 0
                MOVF    PCL,W           : İstenilen karakteri W registerine al
                RETURN

MESAJ_VERISI3;.....
              ADDWF     PCL,F           : PCL = PCL + W ile veriyi adresle
                RETLW 'A'
                RETLW 'B'
                RETLW 'O'
                RETLW 'N'
                RETLW 'E'
                RETLW ' '
                RETLW H'33'
                RETLW 0
              MOVF      PCL,W           : İstenilen karakteri W registerine al
              RETURN

MESAJ_VERISI4;.....
              ADDWF     PCL,F           : PCL = PCL + W ile veriyi adresle
                RETLW 'A'
                RETLW 'B'
                RETLW 'O'
                RETLW 'N'
                RETLW 'E'
                RETLW ' '
                RETLW H'34'
                RETLW 0
              MOVF      PCL,W           : İstenilen karakteri W registerine al
              RETURN
```

```
GECIKME1;.....
          MOVLW   D'65'
          MOVWF   SAYAC1          : 50 ms'lik gecikme alt programı
G1        MOVLW   D'255'
          MOVWF   SAYAC2
G2        DECFSZ  SAYAC2,F
          GOTO    G2
          DECFSZ  SAYAC1,F
          GOTO    G1
          RETURN


GECIKME2;.....
          MOVLW   D'155'
          MOVWF   SAYAC1          : 30 sn'lik gecikme alt programı
D1        MOVLW   D'255'
          MOVWF   SAYAC2
D2        MOVLW   D'255'
          MOVWF   SAYAC3
D3        DECFSZ  SAYAC3,F
          GOTO    D3
          DECFSZ  SAYAC2,F
          GOTO    D2
          DECFSZ  SAYAC1,F
          GOTO    D1
          RETURN
          END
```

1.4. Basketbol Skorbord Uygulama Devresi

Basketbol skorbord devresinde iki takım için iki ayrı buton grubu bulunacaktır. Butonlarla skor artırılıp azaltılabilecek ve sonuçlar 2x16 LCD'de gösterilecektir.

Takımlar A ve B olarak isimlendirilir. "A " LCD'nin 1. satır 4. sütununa, "B" ise 1. satır 11. sütununa yazılır. A takımının skoru 2.satırın 3.sütunundan itibaren B takımının skoru ise 2. satırın 10. sütunundan itaberen yazdırılır.

LCD'ye karakter yazdırılırken karakterin onaltılık kodu kullanılır. Basketbolda skor en fazla 3 basamaklı bir sayıdan oluşur. Bu sayıyı LCD'ye göndermek için sayı basamaklarına ayrılır ve karakterler tek tek gönderilir. Örneğin, 123 sayısı "1" sol karakter, "2" orta karakter ve "3" sağ karakter olarak bölünür. Önce sol karakter, sonra orta karakter ve son olarak da sağ karakter LCD'ye gönderilir. Sayıları artırmak ve azaltmak için aşağıdaki sıra takip edilir.

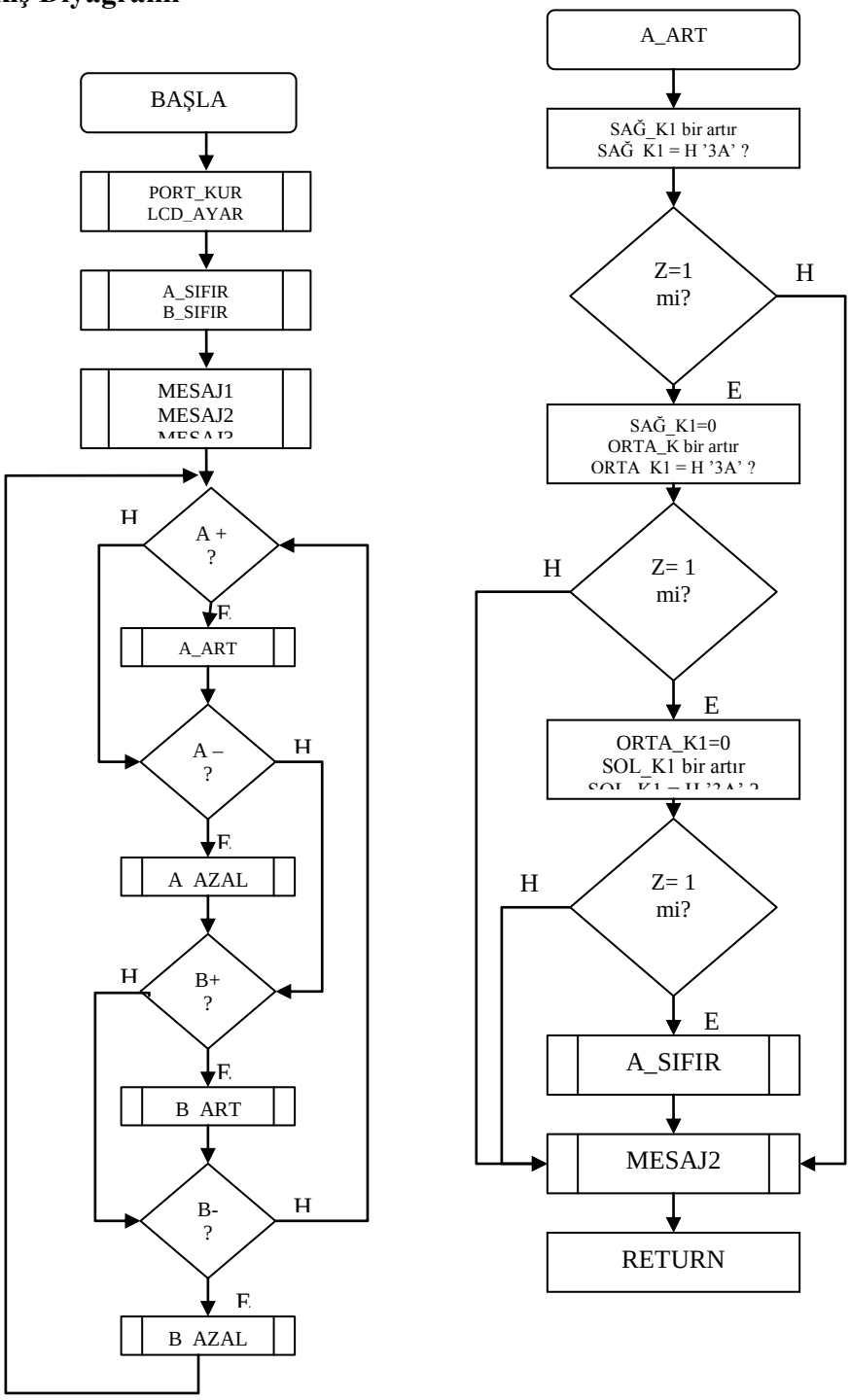
Sayıyı artırmak için,

- Sağ karakter 1 artırılır. Onluk sayı sisteminde en son karakter 9 (H'39') dur. Sağ karakterin 9'u aşıp aşmadığı kontrol edilir. Bunun için sağ karakter H'3A' sayısıyla karşılaştırılır.
- Sağ karakter 9'u aştığında orta karakter 1 artırılır. Orta karakter en fazla 9'a kadar artırılır. 9'u aştığında sol karakter artırılır.
- 999 sayısına ulaşılırsa sayı karakterleri sıfırlanır.
- Örneğin 099 sayısı artırıldığında,
  - Sağ karakter 9'u aştığı için 0'a eşitlenir ve orta karakter bir artırılır.
  - Orta karakter 9'u aştığı için 0'a eşitlenir ve sol karakter bir artırılır.
  - Sol karakter=1, orta karakter=0 ve sağ karakter= 0 olur.

Sayıyı azaltmak için

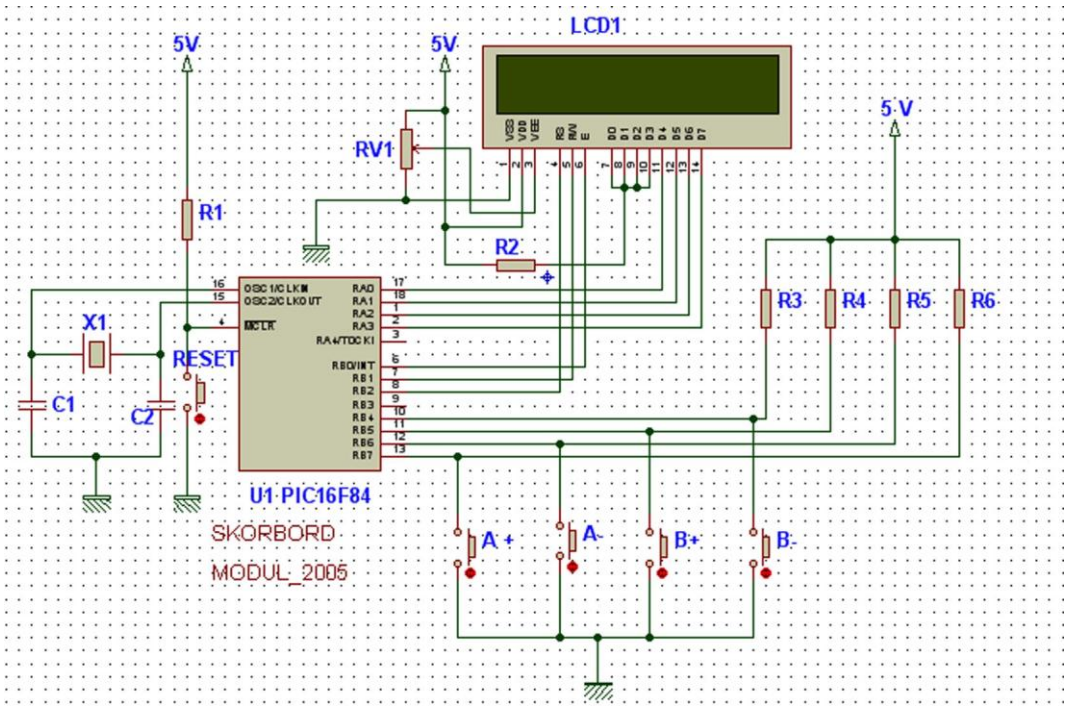
- Sağ karakterin "0" olup olmadığı kontrol edilir (h'30'). Sıfır değilse sayı bir azaltılır. Sıfıra eşitse orta karakterin 0'a eşit olup olmadığı kontrol edilir.
- Orta karakter sıfır değilse sol karakter 1 azaltılır ve sağ karakter 9'a eşitlenir.
- Orta karakter sıfır ise sol karakterin 0'a eşit olup olmadığına bakılır. Sıfıra eşitse sayıda hiçbir değişiklik yapılmaz.
- Sıfıra eşit değilse sol karakter 1 azaltılır. Orta ve sağ karakter 9'a eşitlenir.
- Örneğin 100 sayısı azaltıldığında,
  - Sağ karakterin 0'a eşit olup olmadığı kontrol edilir. Sıfıra eşit orta karaktere geçilir.
  - Orta karakter kontrol edilir. Sıfıra eşit olduğundan sol karaktere geçilir.
  - Sol karakter sıfıra eşit olamadığından 1 azaltılır. Sağ ve orta karakter "9" yapılır. Sonuçta sayı 099 olur.

1.4.1. Akış Diyagramı



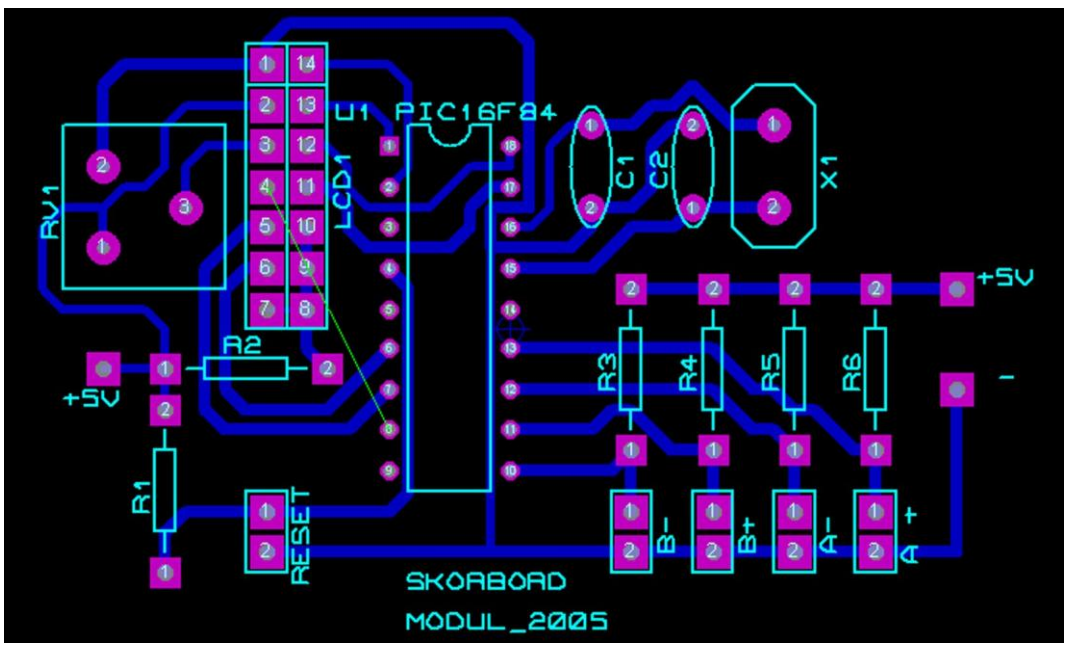
Tablo 1.9: Skorbord akış diyagramı

1.4.2. Devre Şeması



Şekil 1.22: Basketbol skorbord devresi

1.4.3. Baskı Devresi



Şekil 1.23: Basketbol skorbord baskı devresi

1.4.4. Devrenin Malzemeleri

- PIC16F84 4 MHz mikrodenetleyici
- 4MHz kristal, 4xbuton, led
- C1 = C2 = 22pf
- R2 = 1KΩ direnç
- R1 = R3 = R4 = R5 = R6 = 10KΩ
- RV1 = 1K potansiyometre

1.4.5. Devrenin Asm Programı

Programda değişken tanımlamaları CBLOCK – ENDC komutuyla yapılmıştır. EQU komutu kullanarak bir değere eşitlenen herhangi bir değişkenin değeri program içerisinde değiştirilemez. Bu da programın istediğiniz gibi çalışmasını engeller. CBLOCK –ENDC komutu bir grup değişkene değer vermek için kullanılır. CBLOCK'tan sonra yazılan adres değeri sırasıyla tüm değişkenlere birer artarak atanır.

```
;=============SKORBOARD UYGULAMA PROGRAMI======S_2005========

            LIST            P=16F84
            INCLUDE    "P16F84.INC"
#DEFINE RS   PORTB,2
#DEFINE RW   PORTB,1
#DEFINE EN   PORTB,0

CBLOCK H'10'            : Değişkenleri tanımla
            CX
            TEMP
            TEMP0
            TEMP1
            SOL_K1
            ORTA_K1
            SAG_K1
            SOL_K2
            ORTA_K2
            SAG_K2
            SAYAC1
            SAYAC2
ENDC
```

```
BAŞLA;.....
            CALL    PORT_KUR        : Portları ayarla
            CALL    LCD_AYAR        : LCD ilk kullanım için ayarla
            CALL    A_SIFIR         : A takımının değişkenlerini sıfırla
            CALL    B_SIFIR         : B takımının değişkenlerini sıfırla
            CALL    MESAJ1          : 1. satıra A ve B yaz
            CALL    MESAJ2          : 2. satıra takım A'nın ilk puanını yaz
            CALL    MESAJ3          : 2. satıra  takım B'nin ilk puanını yaz
BUTON
            BTFSS   PORTB,7         : Tuşları kontrol et
            CALL    A_ART
            BTFSS   PORTB,6
            CALL    A_AZAL
            BTFSS   PORTB,5
            CALL    B_ART
            BTFSS   PORTB,4
            CALL    B_AZAL
            GOTO    BUTON

PORT_KUR;.....
            BSF     STATUS,5
            CLRF    TRISA
            MOVLW   H'F0'
            MOVWF   TRISB
            BCF     STATUS,5
            RETURN

LCD_AYAR ;.....
            CALL    LCD_RESET       : LCD Resetle
            CALL    FUNCTION_SET    : 4 bit mod ve iki satır aktif
            CALL    DISPLAY_ON      : Ekranı aç ve kursörü kapat
            CALL    ENTRY_MODE      : Kursör 1 artan mod
            RETURN

LCD_RESET;.....
            CALL    GECIKME1
            MOVLW   H'03'
            MOVWF   CX
RESET       MOVLW   H'00'
            MOVWF   PORTA
            CALL    LCD_KOMUT
            DECFSZ  CX,1            : CX ← CX -1
            GOTO    RESET
            RETURN
```

```
FUNCTION_SET;.....
        MOVLW   H'02'           : 4 bit mod
        MOVWF   PORTA
        CALL    LCD_KOMUT
        MOVLW   H'02'           : 4 bit mod
        MOVWF   PORTA
        CALL    LCD_KOMUT
        MOVLW   H'08'           : 2 satır
        MOVWF   PORTA
        CALL    LCD_KOMUT
        RETURN

DISPLAY_ON;.....
        MOVLW    H'00'
        MOVWF    PORTA
        CALL     LCD_KOMUT
        MOVLW    H'0C'          : LCD on
        MOVWF    PORTA
        CALL     LCD_KOMUT
        RETURN

ENTRY_MODE;.....
        MOVLW   H'00'
        MOVWF   PORTA
        CALL    LCD_KOMUT
        MOVLW   H'06'           : Kursör 1 artan mod
        MOVWF   PORTA
        CALL    LCD_KOMUT
        RETURN

LCD_SIL;.....

        MOVLW   H'00'
        MOVWF   PORTA
        CALL    LCD_KOMUT
        MOVLW   H'01'           : Ekranı temizle, kürsör 1.satır 1.sütunda
        MOVWF   PORTA
        CALL    LCD_KOMUT
        RETURN
```

```
LCD_YAZ;.....
        MOVWF   TEMP            : TEMP ← W
        SWAPF    TEMP,F         : Üst 4 bit ile alt 4 biti yer değiştir.
        MOVF    TEMP,W          : Üst 4 biti al
        MOVWF   PORTA           : LCD'ye gönder.
        BSF     EN              : E ← 1
        NOP                     : Bekle
        BCF     EN              : E ← 0
        CALL    BF_TEST         : Meşgul bayrağını test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        SWAPF   TEMP,F          : Yazılacak verinin tekrar üst 4 biti
        MOVF    TEMP,W          : alt 4 bitini yer değiştir
        MOVWF   PORTA           : LCD'ye gönder
        BSF     EN              : E ← 1
        NOP                     : Bekle
        BCF     EN              : E ← 0
        RETURN

LCD_KOMUT;.....
        CALL    BF_TEST         : BF test et
        BCF     RS              : RS → 0 Komut
        BCF     RW              : RW → 0 Yaz
        BSF     EN              : E ← 1
        NOP                     : Bekle
        BCF     EN              : E ← 0
        RETURN

BF_TEST;.....
        BSF     STATUS,5        : Bank 1'e geç
        MOVLW   H'FF'
        MOVWF   TRISA           : PORTA ← Giriş
        BCF     STATUS,5        : Bank 0'a geç
        BCF     RS              : Veri
        BSF     RW              : LCD Oku
        BSF     EN              : E← 1
        MOVF    PORTA,W         : LCD'den gelen bilgileri W  aktar.
        BCF     EN              : E ← 0
        ANDLW   H'F0'           : Üst 4 biti sıfırla
        MOVWF   TEMP1           : TEMP1'de sakla
        SWAPF   TEMP1,F         : Alt 4 bit ile üst 4 biti yer değiştir.
        BSF     EN              : E ← 1
        GOTO    $+1             : Bir alt satıra git
        BCF     EN              : E ← 0
        BTFSC   TEMP1,7         : BF = 1 mi?
        GOTO    BF_TEST         : Evet, BF'yi tekrar test et
```

```
        BSF     STATUS,5        : Hayır
        CLRF    TRISA           : PORTA ← Çıkış
        BCF     STATUS,5
        RETURN

SATIR1W;.....

        MOVWF   TEMP0           : TEMP0 ← W
        MOVLW   H'08'           : LCD 1. satırı aktif
        MOVWF   PORTA
        CALL    LCD_KOMUT
        MOVF    TEMP0,W
        MOVWF   PORTA           : 1. satırda W sütununa git
        CALL    LCD_KOMUT
        RETURN

SATIR2W;.....

        MOVWF   TEMP0           : TEMP0 ← W
        MOVLW   H'0C'           : LCD 2. satırı aktif
        MOVWF   PORTA
        CALL    LCD_KOMUT
        MOVF    TEMP0,W
        MOVWF   PORTA           : 2. satırda W sütununa git
        CALL    LCD_KOMUT
        RETURN
A_SIFIR;.....
        MOVLW   H'30'           : 1. sayının karakterlerine H'30'
        MOVWF   SOL_K1          : yani  "0 " yükle
        MOVWF   ORTA_K1
        MOVWF   SAG_K1
        RETURN

B_SIFIR;.....
        MOVLW   H'30'           : 2. sayının karakterlerine H'30'
        MOVWF   SOL_K2          : yani  "0 " yükle
        MOVWF   ORTA_K2
        MOVWF   SAG_K2
        RETURN
```

```
MESAJ1…….
        MOVLW   H'04'           : LCD'nin 1. satırın 4. sütununa git.
        CALL    SATIR1W
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVLW   H'41'           : Ekrana  " A" yaz
        CALL    LCD_YAZ
        MOVLW   H'0B'           : LCD'nin 1.satır 11. sütununa git.
        CALL    SATIR1W
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVLW   H'42'           : Ekrana "B" yaz
        CALL    LCD_YAZ
        RETURN

MESAJ2……..
        MOVLW   H'03'           : LCD'nin 2. satır 3. sütununa git
        CALL    SATIR2W
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVF    SOL_K1,W        : W← SOL_K1
        CALL    LCD_YAZ         : ilk sayının sol karakterini ekrana yaz
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVF    ORTA_K1,W       : İlk sayının orta karakterini ekrana yaz
        CALL    LCD_YAZ
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVF    SAG_K1,W        : İlk sayının sağ  karakterini ekrana yaz
        CALL    LCD_YAZ
        RETURN
```

```
MESAJ3……..
        MOVLW   H'0A'           : LCD'nin 2. satır 10. sütununa git.
        CALL    SATIR2W
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVF    SOL_K2,W        : W← SOL_K1
        CALL    LCD_YAZ         : ikinci sayının sol karakterini ekrana yaz.
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVF    ORTA_K2,W       : ikinci sayının orta karakterini yaz.
        CALL    LCD_YAZ
        CALL    BF_TEST         : BF test et
        BSF     RS              : RS → 1 Veri
        BCF     RW              : RW → 0 Yaz
        MOVF    SAG_K2,W        : İkinci sayının sağ  karakterini yaz.
        CALL    LCD_YAZ
        RETURN

A_ART;.....
        CALL    TIMER1          : Gecikme
        INCF    SAG_K1          : Sağ karakteri 1 artır.
        MOVLW   H'3A'           : Sağ karakter H'3A sayısına eşit mi?
        XORWF   SAG_K1,W
        BTFSS   STATUS,2
        GOTO    SON1            : Hayır devam et.
        MOVLW   H'30'           : Evet  sağ karakteri sıfırla.
        MOVWF   SAG_K1
        INCF    ORTA_K1,F       : Orta karakteri 1 artır.
        MOVLW   H'3A'           : Orta karakter H'3A' sayısına eşit mi?
        XORWF   ORTA_K1,W
        BTFSS   STATUS,2
        GOTO    SON1            : Hayır devam et.
        MOVLW   H'30'           : Evet orta karakteri sıfırla.
        MOVWF   ORTA_K1
        INCF    SOL_K1,F        : Sol  karakteri 1 artır.
        MOVLW   H'3A'           : Sol karakter H'3A' sayısına eşit mi?
        XORWF   SOL_K1,W
        BTFSS   STATUS,2
        GOTO    SON1            : Hayır devam et.
        CALL    A_SIFIR         : Evet tüm karakterleri sıfırla.
SON1    CALL    MESAJ2          : Karakterleri LCD'ye yazdır.
        RETURN
```

```
A_AZAL;.....................................................................
        CALL    TIMER1          : Gecikme
        MOVLW   H'30'           : Sağ karakter sıfıra H'30' eşit mi?
        XORWF   SAG_K1,W
        BTFSC   STATUS,2
        GOTO    BIR             : Evet BIR etiketine git
        DECF    SAG_K1          : Hayır sağ karakteri 1 azalt
        GOTO    SON2            : LCD'ye yazdır.
BIR     MOVLW   H'30'
        XORWF   ORTA_K1,W       : Orta karakter sıfıra eşit mi?
        BTFSC   STATUS,2
        GOTO     IKI            : Evet IKI etiketine git
        DECF    ORTA_K1         : Hayır orta karakteri 1 azalt
        MOVLW   H'39'           : Sağ karakteri "9" yap.
        MOVWF   SAG_K1
        GOTO    SON2            : LCD'ye yazdır.
IKI     MOVLW    H'30'
        XORWF   SOL_K1,W        : Sol karakter sıfıra eşit mi?
        BTFSC   STATUS,2
        GOTO    SON2            : Evet SON2  etiketine git
        DECF    SOL_K1          : Hayır sol karakteri 1 azalt
        MOVLW   H'39'           : Sağ ve orta karaktere "9"
        MOVWF   SAG_K1          : sayısını yükle
        MOVWF   ORTA_K1
SON2    CALL    MESAJ2          : Karakterleri LCD'ye yazdır.
        RETURN

B_ART;.....................................................................................................
        CALL    TIMER1          : Gecikme
        INCF    SAG_K2          : Sağ karakteri 1 artır
        MOVLW   H'3A'           :Sağ karakter H'3A' sayısına eşit mi?
        XORWF   SAG_K2,W
        BTFSS   STATUS,2
        GOTO    SON3            : Hayır devam et.
        MOVLW   H'30'           : Evet  sağ karakteri sıfırla.
        MOVWF   SAG_K2
        INCF    ORTA_K2,F       : Orta karakteri 1 artır
        MOVLW   H'3A'           : Orta karakter H'3A' sayısına eşit mi?
        XORWF   ORTA_K2,W
        BTFSS   STATUS,2
        GOTO    SON3            : Hayır devam et.
        MOVLW   H'30'           : Evet orta karakteri sıfırla.
        MOVWF   ORTA_K2
        INCF    SOL_K2,F        : Sol  karakteri 1 arttır.
        MOVLW   H'3A'           : Sol karakter H'3A' sayısına eşit mi?
        XORWF   SOL_K2,W
        BTFSS   STATUS,2
```

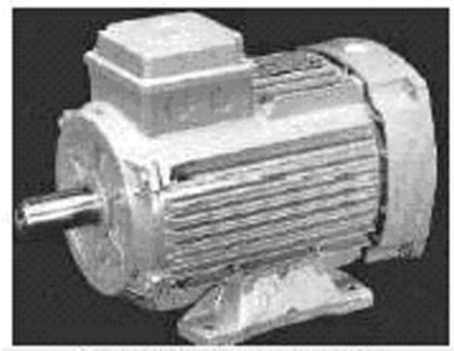
```
            GOTO    SON3            : Hayır devam et.
            CALL    B_SIFIR         : Evet tüm karakterleri sıfırla.
SON3        CALL    MESAJ3          : Karakterleri LCD'ye yazdır.
            RETURN

B_AZAL;.....
            CALL    TIMER1          : Gecikme
            MOVLW   H'30'           : Sağ karakter sıfıra H'30' eşit mi?
            XORWF   SAG_K2,W
            BTFSC   STATUS,2
            GOTO    UC              : Evet UC etiketine git
            DECF    SAG_K2          : Hayır sağ karakteri 1 azalt
            GOTO    SON4            : LCD'ye yazdır
UC          MOVLW   H'30'
            XORWF   ORTA_K2,W       : Orta karakter sıfıra eşit mi?
            BTFSC   STATUS,2
            GOTO     DORT           : Evet DORT etiketine git.
            DECF    ORTA_K2         : Hayır orta karakteri 1 azalt.
            MOVLW   H'39'           : Sağ karakteri "9" yap.
            MOVWF   SAG_K2
            GOTO    SON4            : LCD'ye yazdır.
DORT        MOVLW    H'30'
            XORWF   SOL_K2,W        : Sol karakter sıfıra eşit mi?
            BTFSC   STATUS,2
            GOTO    SON4            : Evet SON4  etiketine git.
            DECF    SOL_K2          : Hayır sol karakteri 1 azalt.
            MOVLW   H'39'           : Sağ ve orta karaktere "9"
            MOVWF   SAG_K2          : sayısını yükle
            MOVWF   ORTA_K2
SON4        CALL    MESAJ3          : Karakterleri LCD ye yazdır
            RETURN

GECIKME1……
            MOVLW   D'60'
            MOVWF   SAYAC1
A1          MOVLW   D'50'
            MOVWF   SAYAC2
A2          DECFSZ  SAYAC2,F
            GOTO    A2
            DECFSZ  SAYAC1,F
            GOTO    A1
            RETURN

TIMER1…….
            MOVLW    D'15'
            MOVWF   SAYAC1
T1          MOVLW   D'25'
```

```
        MOVWF   SAYAC2
T2      DECFSZ  SAYAC2,F
        GOTO    T2
        DECFSZ  SAYAC1,F
        GOTO    T1
        RETURN
        END
```

1.5. Asenkron Motorun Yıldız Üçgen Çalışması

1.5.1. Asenkron Motorun Yapısı ve Çalışması

Sanayi tesislerinde elektrik enerjisini dairesel harekete çevirebilmek için motorlar kullanılır. Endüstride asenkron motorların (endüksiyon motorları) kullanım oranı çok yüksek olup % 90'lar civarındadır.



Resim 1.1: Asenkron motor

1.5.1.1. Asenkron Motorların Üstünlükleri

- Sürekli bakım istemez.
- Yük altında devir sayıları çok değişmez.
- Elektronik devreyle devir sayısı kolayca ayarlanabilir.
- Fiyatları diğer motorlara oranla ucuzdur.
- Çalışma anında ark (kıvılcım) oluşturmaz.
- Bir ve üç fazlı olarak üretilebilir.

1.5.1.2. Asenkron Motor Çeşitleri

Asenkron motorlar faz sayısına göre iki çeşittir:

- Bir fazlı asenkron motorlar: Küçük ve güçlüdür. Çamaşır makinesi, pompa, buzdolabı gibi cihazlarda kullanılır.
- Üç fazlı asenkron motorlar: Sanayide çok yaygın olarak kullanılan motor çeşididir.

Asenkron motorlar rotorlarının yapısına göre iki çeşittir:

- Rotoru kısa devre çubuklu asenkron motorlar
- Rotoru sargılı (bilezikli) asenkron motorlar

Asenkron motorlar genel olarak stator ve rotor olmak üzere iki kısımdan yapılmıştır.

1.5.1.3. Stator

Asenkron motorun duran bölümüdür. 0,4-0,8 mm kalınlığında bir tarafı yalıtılmış sacların özel kalıplarda paketlenmesiyle üretilir. (Resim 1.2) Stator, motorun en önemli parçasıdır. Bu parçanın iç kısmında emaye izoleli bakır telden yapılan sarımlar bulunur. Sarımların görevi AC enerji uygulandığında manyetik alan oluşturarak rotorun dönmesini sağlamaktır.

Resim 1.2: Stator

1.5.1.4. Rotor

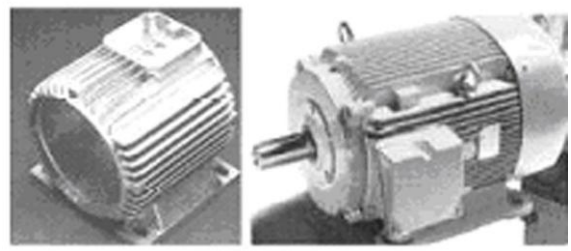
Asenkron motorun dönen bölümüdür. Rotor, ince çelik sacların üst üste paketlenmesiyle oluşturulmuştur. Bu elemanın stator manyetik alanının etkisiyle ikinci bir manyetik alan oluşturabilmesi için gövdesi üzerine açılan oyuklara alüminyum çubuklar ya da sargılar konulmuştur. Rotor çeşitleri şunlardır:

- Alüminyum çubuklu (sincap kafesli) rotor
- Sargılı (bilezikli) rotor

Resim 1.3: Rotor

1.5.1.5. Gövde

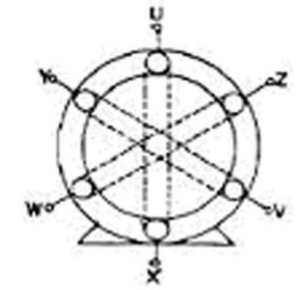
Asenkron motorların gövdesi soğutmanın çabuk olması için çıkıntılı (kanatçıklı) olarak üretilir. Resim 1.4' te motor gövdeleri görülmektedir.



Resim 1.4: Motor gövdeleri

1.5.1.6. Üç Fazlı Asenkron Motorların Çalışma İlkesi

R-S-T fazları motorun statorunda bulunan sargılara uygulandığında döner bir manyetik alan oluşur. Statordaki manyetik alanın dönüş sayısı şebekenin frekansı ve sargıların kutup sayısına göre değişir. Statorda oluşan döner alan rotorun çubuklarını (ya da sarımlarını) etkiler ve bu çubuklardan akım dolaşmaya başlar. Rotordan geçen akım, ikinci bir alan oluşturur. Statorun alanıyla rotorun alanı birbirini itip çekerek dönüşü başlatır.



Şekil 1.24: Üç fazın stator sargıları

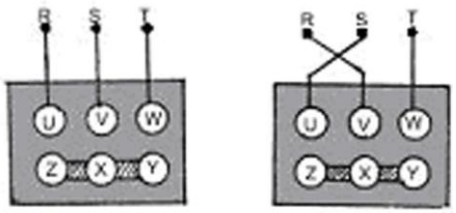
Üç fazlı asenkron motorların statoruna yapılan sarımların uçları klemens kutusuna (bağlantı terminali) çıkarılır. Klemens kutusunda bulunan harflerin anlamları şunlardır:

- R Fazı : Giriş U, Çıkış X
- S Fazı : Giriş V, Çıkış Y
- T Fazı : Giriş W, Çıkış Z harfleriyle gösterilir.

Resim 1.5: Motor klemens kutusunda yıldız – üçgen bağlantısı

Üç fazlı asenkron motorların klemens kutusunda altı adet uç bulunur. Bu uçlar motorun gücü göz önüne alınarak yıldız ya da üçgen şeklinde bağlandıktan sonra R-S-T ile besleme yapılır. Yıldız ya da üçgen bağlantısı yapılmamış üç fazlı asenkron motor asla çalışmaz.

Üç fazlı asenkron motorların devir yönünü değiştirmek son derece kolaydır. Motora uygulanan R-S-T fazlarından herhangi ikisinin yeri Şekil 1.25'te görüldüğü gibi değiştirildiğinde stator sargılarının oluşturduğu manyetik alanın dönüş yönü değişir ve rotor önceki dönüş yönünün tersinde hareket etmeye başlar.



Şekil 1.25: Üç fazlı asenkron motorun devir yönünün değiştirilmesi

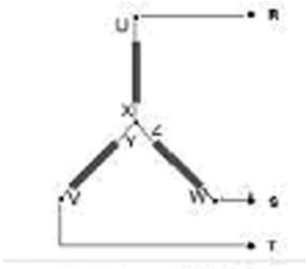
1.5.1.7. Üç Fazlı Asenkron Motora Yol Verme

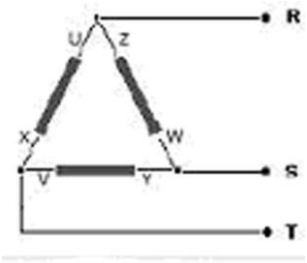
Üç fazlı asenkron motorlar ilk hareket anında normal akımlarının 6-10 katı fazla akım çeker. Bu aşırı akım küçük güçlü bir motorda şebekeye pek bir zarar vermez. Ancak gücü 4 kW'tan büyük olan bir motorun ilk anda 6-10 kat fazla akım çekerek çalışmaya başlaması birçok olumsuz etki (tesisisin geriliminin kısa süreli olarak anormal derecede düşmesi, hatların aşırı yüklenmesi vb.) ortaya çıkarır. İşte bu sebeple 4 kW'tan büyük güçteki motorları ilk kalkış anında az akım çekerek çalıştırmak gerekir. Günümüzde büyük güçlü motorların ilk kalkış akımını kabul edilebilir düzeye indirebilmek için çeşitli yöntemler kullanılmaktadır. Düşük kalkınma akımıyla çalıştırmada kullanılan bazı yöntemler şöyle sıralanabilir:

- Dirençle yol verme
- Oto transformatörüyle yol verme
- Yıldız / üçgen yol verme

Uygulamada en çok kullanılan yöntem yıldız / üçgen yol vermedir. Stator sargıları önce yıldız şeklinde bağlanır. Bu sayede 380 volta dayanacak şekilde üretilmiş sargılara 220 volt uygulanmış olacağından motor düşük akım çekerek çalışmaya başlar. 2 – 4 saniye sonra yıldız bağlantısı kaldırılıp üçgen bağlantıya geçilir.

Üç fazlı asenkron motorlarda uygulanan yıldız bağlama U-V-W / X-Y-Z klemens uçları birbirine köprülendiğinde yıldız bağlantı yapılmış olunur.





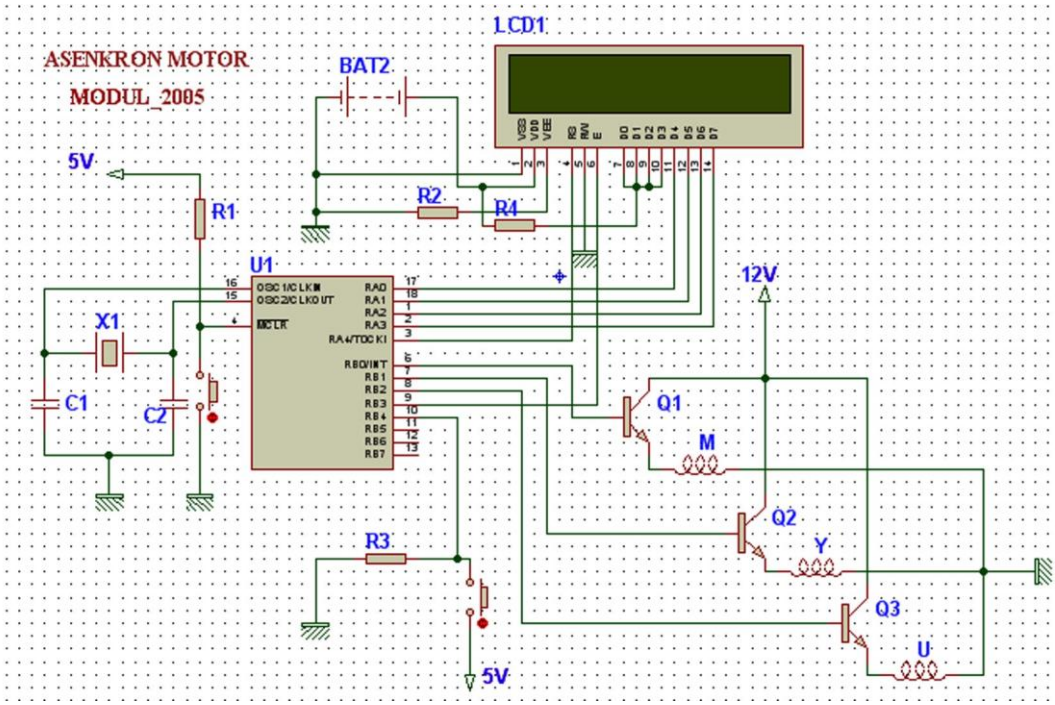
Şekil 1.26: Yıldız bağlama **Şekil 1.27: Üçgen bağlama**

Yıldız bağlanarak çalıştırılması gereken bir motor yanlışlıkla üçgen bağlanarak çalıştırılacak olursa sargılara 380 volt uygulanmış olacağından motor yanar.

Üçgen bağlamada stator sargıları birbirine seri bağlanır. Üçgen bağlı olarak çalışacak şekilde üretilmiş motorların sarımları 380 volta dayanacak şekilde üretilmiştir. Üçgen bağlanması gereken motor yanlışlıkla yıldız bağlanarak çalıştırılırsa motor yanmaz. Ancak motor düşük verimde çalışır.

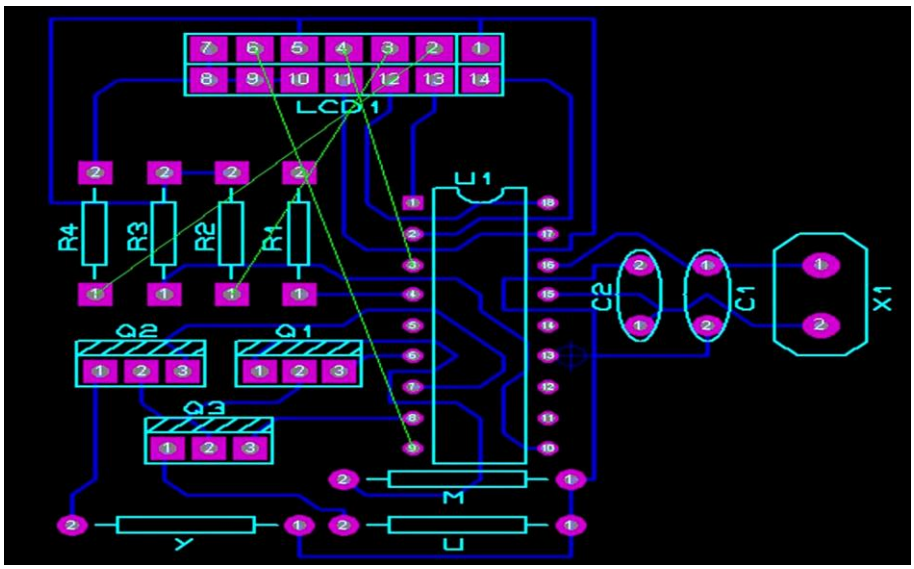
Pic ile yapılan asenkron motor uygulama devresin sistem "BAŞLA" butonu ile kontrol edilecektir. Üç çıkış yıldız üçgen çalışma tekniğine göre kontrol edilecek, yıldız çalışma ve üçgen çalışma süresi 2x16 LCD'de görüntülenecektir. Motoru durdurmak için "RESET" butonuna basılacaktır. Motor önce yıldız çalışma ile çalıştırılacak LCD ekranında "yıldız çalışma" yazacak bundan sonra üçgen çalışmaya geçilecek ve ekranda "üçgen çalışma" yazısı görüntülenecektir.

1.5.2. Devrenin Şeması



Şekil 1.28: Asenkron motor uygulama devresi

1.5.3. Baskı Devresi



Şekil 1.29: Asenkron motor baskı devresi

1.5.4. Devrenin Malzemeleri

- PIC16F84 4MHz mikrodenetleyici
- 4MHz kristal, 4xbuton, led
- C1 = C2 = 22pf
- R2 = R4 = 1KΩ direnç
- R1 = R3 = 10KΩ
- 12 V Röle X 3

1.5.5. Devrenin Asm Programı

```
;===============ASENKRON MOTOR UYGULAMA PROGRAMI===========

            LIST P = 16F84

            INCLUDE  "P16F84.INC"
            RADİX       DEC

; Değişkenler……..
CBLOCK     H'12'
S0
SYS_WSAV
SYS_SSAV
TR1
TR1_HI
ZV1
ZV1_FD
SYS_TMP1
SYS_TMP2
SYS_TMP3
SYS_TMP4
SYS_TMP5
SYS_TMP6
SYS_TMP9
SYS_TMP10
LT1_FD
S1
LT2_FD
LT3_FD
LT4_FD
MF1
MF1_HI
MF1_FD
TR2
TR2_HI
LT5_FD
```

```
RAM
ENDC

            ORG 0

            CLRF STATUS
            GOTO PORT_KUR
            NOP
            NOP

            MOVWF   SYS_WSAV
            SWAPF   STATUS,W
            CLRF    STATUS              : Bank 0
            MOVWF   SYS_SSAV
            BCF     INTCON,T0IF
            INCFSZ  TR1,F
            GOTO    SYS_INT_LABEL_0
            INCF    TR1_HI,F

SYS_INT_LABEL_0;…….

            INCFSZ  MF1,F
            GOTO    SYS_INT_LABEL_1
            INCF    MF1_HI,F

SYS_INT_LABEL_1;……

            INCFSZ  TR2,F
            GOTO    SYS_INT_LABEL_2
            INCF    TR2_HI,F

SYS_INT_LABEL_2;……..

EXITINT

            SWAPF   SYS_SSAV,W
            MOVWF   STATUS
            SWAPF   SYS_WSAV,F
            SWAPF   SYS_WSAV,W
            RETFIE

LCD_AYAR;…….
            BCF     STATUS,RP0          : Bank 0
            BCF     PORTA,4             : RS → 0
LCD_LB1_L1
            CALL    LCD_LB1_OUT4
            DECFSZ  SYS_TMP1,F
```

```
        GOTO    LCD_LB1_L1

        BCF     STATUS,RP0      : Bank 0
        MOVF    PORTA,W
        ANDLW   H'F0'
        IORLW   B'0011'         : Function Set
        MOVWF   PORTA

        BCF     STATUS,RP0      : Bank 0
        BSF     PORTB,3         : Yaz
        BCF     PORTB,3

        BSF     SYS_TMP1,5      : Bekle
LCD_LB1_L2;……

        CALL    LCD_LB1_OUT4
        DECFSZ  SYS_TMP1,F
        GOTO    LCD_LB1_L2
        BCF     STATUS,RP0      : Bank 0
        BSF     PORTB,3         : Yaz
        BCF     PORTB,3
        CALL    LCD_LB1_OUT4
        BCF     STATUS,RP0      : Bank 0
        BSF     PORTB,3         : Yaz
        BCF     PORTB,3
        CALL    LCD_LB1_OUT4
        BCF     STATUS,RP0      : Bank 0
        BCF     PORTA,0         : 4-Bit

        BCF     STATUS,RP0      : Bank 0
        BSF     PORTB,3         : Yaz
        BCF     PORTB,3
        CALL    LCD_LB1_OUT4

        MOVLW   B'00101000'     : Function Set 4-Bit, 2 satır
        CALL    LCD_LB1_OUT

        MOVLW   B'00001100'     : Display on, Cursor off, Blink off
        CALL    LCD_LB1_OUT

        MOVLW   B'00000110'     : Entrymode set
        CALL    LCD_LB1_OUT

LCD_LB1_CLR;…….
        BCF     STATUS,RP0      : Bank 0
        BCF     PORTA,4
        MOVLW   B'00000001'     : Display temizle
```

```
            CALL    LCD_LB1_OUT

            BSF     SYS_TMP1,5          : Bekle
LCD_LB1_L3;……
            CALL    LCD_LB1_OUT4
            DECFSZ  SYS_TMP1,F
            GOTO    LCD_LB1_L3
            GOTO    LCD_LB1_OUT4

LCD_LB1_GOTOXY;…….
            BCF     STATUS,RP0          : Bank 0
            BCF     PORTA,4
            MOVLW   H'40'
            BTFSC   SYS_TMP2,0          : Ypos
            ADDWF   SYS_TMP1,F
            BSF     SYS_TMP1,7          : dd-ram seç
            GOTO    LCD_LB1_OUT1

LCD_LB1_WRITESTR;……..
            BCF     STATUS,RP0          : Bank 0
            BSF     PORTA,4             : dd-ram seç
LCD_LB1_WRITESTR1;……..
            CALL    LCD_LB1_CHAROUT
            BSF     INTCON,GIE
            ANDLW   H'FF'               : last is zero
            BTFSC   STATUS,Z
            RETURN

            MOVWF   SYS_TMP1
            CALL    LCD_LB1_OUT1
            INCFSZ  SYS_TMP4,F
            GOTO    LCD_LB1_WRITESTR1
            GOTO    LCD_LB1_WRITESTR1

LCD_LB1_CHAROUT;……
            BCF     INTCON,GIE
            MOVFW   SYS_TMP5
            MOVWF   PCLATH
            MOVFW   SYS_TMP4
            MOVWF   PCL

LCD_LB1_DEZO10000;……..

            MOVLW   LOW 10000
            MOVWF   SYS_TMP1
            MOVLW   HIGH 10000
            MOVWF   SYS_TMP2
```

```
        CALL    LCD_LB1_CONVERTDEZ
        MOVF    SYS_TMP9,F
        BTFSS   STATUS,Z
        GOTO    LCD_LB1_DEZO1000
        CALL    LCD_LB1_CHECKNULL

LCD_LB1_DEZO1000;…….

        MOVLW   LOW 1000
        MOVWF   SYS_TMP1
        MOVLW   HIGH 1000
        MOVWF   SYS_TMP2
        CALL    LCD_LB1_CONVERTDEZ
        MOVF    SYS_TMP9,F
        BTFSS   STATUS,Z
        GOTO    LCD_LB1_DEZO100
        CALL    LCD_LB1_CHECKNULL

LCD_LB1_DEZO100;……..

        MOVLW   LOW 100
        MOVWF   SYS_TMP1
        CLRF    SYS_TMP2
        CALL    LCD_LB1_CONVERTDEZ
        MOVF    SYS_TMP9,F
        BTFSS   STATUS,Z
        GOTO    LCD_LB1_DEZO10
        CALL    LCD_LB1_CHECKNULL

LCD_LB1_DEZO10;……..
        MOVLW   LOW 10
        MOVWF   SYS_TMP1
        CLRF    SYS_TMP2
        CALL    LCD_LB1_CONVERTDEZ
        MOVF    SYS_TMP9,F
        BTFSS   STATUS,Z
        GOTO    LCD_LB1_DEZO1
        CALL    LCD_LB1_CHECKNULL

LCD_LB1_DEZO1;……..
        MOVLW   48
        ADDWF   SYS_TMP4,W
        GOTO    LCD_LB1_WRITECHAR
```

```
LCD_LB1_CONVERTDEZ;…….
        MOVF    SYS_TMP9,F
        BTFSS   STATUS,Z
        DECF    SYS_TMP9,F
        CLRF    SYS_TMP6

LCD_LB1_CONVERTDEZ2;……..
        INCF    SYS_TMP6,F
        MOVFW   SYS_TMP1
        SUBWF   SYS_TMP4,F
        MOVFW   SYS_TMP2
        BTFSS   STATUS,C
        INCFSZ  SYS_TMP2,W
        SUBWF   SYS_TMP5,F
        BTFSC   STATUS,C
        GOTO    LCD_LB1_CONVERTDEZ2
        MOVFW   SYS_TMP2
        ADDWF   SYS_TMP5,F
        MOVFW   SYS_TMP1
        ADDWF   SYS_TMP4,F
        BTFSC   STATUS,C
        INCF    SYS_TMP5,F
        DECF    SYS_TMP6,W
        BTFSS   STATUS,Z
        BSF     SYS_TMP10,0
        RETURN


LCD_LB1_CONVERTHEX;……
        ANDLW   15
        MOVWF   SYS_TMP1
        MOVLW   48
        ADDWF   SYS_TMP1,F
        MOVLW   58
        SUBWF   SYS_TMP1,W
        BTFSS   STATUS,C
        GOTO    $+3
        MOVLW   7
        ADDWF   SYS_TMP1,F
        MOVFW   SYS_TMP1
```

```
LCD_LB1_WRITECHAR;……
        BCF     STATUS,RP0      : Bank 0
        BSF     PORTA,4         : dd-ram

LCD_LB1_OUT;……
        MOVWF   SYS_TMP1        : store

LCD_LB1_OUT1;……
        CLRF    SYS_TMP3        : LCD yaz

LCD_LB1_OUT2;…….
        SWAPF   SYS_TMP1,W
LCD_LB1_OUT3;……
        ANDLW   H'0F'
        MOVWF   SYS_TMP2
        BCF     STATUS,RP0      : Bank 0
        MOVFW   PORTA
        ANDLW   H'F0'
        IORWF   SYS_TMP2,W
        BCF     STATUS,RP0      : Bank 0
        MOVWF   PORTA           : out

        BCF     STATUS,RP0      : Bank 0
        BSF     PORTB,3         : E → 1
        BCF     PORTB,3         : E →0

        BTFSC   SYS_TMP3,3      : LCD hazır mı ?
        GOTO    LCD_LB1_OUT4
        BSF     SYS_TMP3,3
        MOVFW   SYS_TMP1
        GOTO    LCD_LB1_OUT3
LCD_LB1_OUT4;……..
LCD_LB1_OUTV;……..
        DECFSZ  SYS_TMP3,F
        GOTO    LCD_LB1_OUTV
        RETURN

LCD_LB1_CHECKNULL;…….
        BTFSS   STATUS,Z
        DECF    SYS_TMP9,F
        MOVWF   SYS_TMP1
        BTFSS   STATUS,Z
        BSF     SYS_TMP10,0
        MOVLW   48
        BTFSC   SYS_TMP10,0
        ADDWF   SYS_TMP1,f
```

```
            MOVLW   32
            BTFSS   SYS_TMP10,0
            ADDWF   SYS_TMP1,F
            MOVFW   SYS_TMP1
            GOTO    LCD_LB1_WRITECHAR

PORT_KUR ;……

            MOVLW   B'00100000'         : TMR0 sayıcısı aktif
            MOVWF   INTCON
            BCF     STATUS,RP0          : Bank 0 a geç
            MOVLW   B'00000000'
            MOVWF   PORTA               : PortA temizle
            MOVLW   B'00000000'
            MOVWF   PORTB               : PortB temizle
            MOVLW   143
            BSF     STATUS,RP0          ; Bank 1
            MOVWF   OPTION_REG          : WDT kesmesi geçerli
            MOVLW   B'11100000'         : 1→ Giriş, 0→ Çıkış
            MOVWF   TRISA
            MOVLW   B'11110000'
            MOVWF   TRISB
; Değişkenleri Temizle

            MOVLW   12
            MOVWF   FSR                 : Sayı dizisinin başlangıç adresini
CR1;……
            CLRF    INDF                : sırası ile
            INCF    FSR,F               : sıfırla
            MOVf    FSR,W
            SUBLW    80                 : FSR adresi 80 mi?
            BTFSS   STATUS,Z
            GOTO    CR1                 : Hayır temizlemeye devam et
            MOVLW   B'00010000'         : Evet S1 = 10
            MOVWF   S1
            CALL LCD_AYAR
            BCF     STATUS,RP0          : Bank 0
            CLRF    TMR0
            BCF     INTCON,T0IF
            BSF     INTCON,GIE
MAIN
```

```
LABEL_UG1_1;……..
            BTFSS       S0,1
            GOTO        LABEL_UG1_FALSE
LABEL_UG1_2;……..
            BTFSC       S0,2
            GOTO        LABEL_UG1_FALSE

LABEL_UG1_TRUE;……
            BSF         S0,0
            GOTO        LABEL_UG1_ENDE
LABEL_UG1_FALSE;…….
            BCF         S0,0
LABEL_UG1_ENDE
LABEL_UG2_1;……
            BTFSC       S0,0
            GOTO        LABEL_UG2_TRUE
LABEL_UG2_2;……
            BCF         STATUS,RP0          ; Bank 0
            BTFSC       PORTB,5
            GOTO        LABEL_UG2_TRUE

LABEL_UG2_FALSE;…….
            BCF         S0,1
            GOTO        LABEL_UG2_ENDE
LABEL_UG2_TRUE;…….
            BSF         S0,1
LABEL_UG2_ENDE
LABEL_UG3_1;……
            BTFSS       S0,1
            GOTO        LABEL_UG3_FALSE

LABEL_UG3_TRUE;…….
            BCF         STATUS,RP0          : Bank 0
            BSF         PORTB,0
            GOTO        LABEL_UG3_ENDE
LABEL_UG3_FALSE;……
            BCF         STATUS,RP0          : Bank 0
            BCF         PORTB,0
LABEL_UG3_ENDE;……

            MOVLW       7
            SUBWF       TR1_HI,W
            BTFSS       STATUS,Z
            GOTO        LABEL_TR1_CHECK
            MOVLW       161
            SUBWF       TR1,W
```

```
LABEL_TR1_CHECK;……
        BTFSS   STATUS,C
        GOTO    LABEL_TR1_ENDE

LABEL_TR1_RESET;……..
        MOVLW   7
        SUBWF   TR1_HI,F
        MOVLW   161
        SUBWF   TR1,F
        BTFSS   STATUS,C
        DECF    TR1_HI,F

LABEL_TR1_SET;……
        BTFSC   S0,3
        GOTO    LABEL_TR1_CLR
        BSF     S0,3
        GOTO    LABEL_TR1_ENDE
LABEL_TR1_CLR;……..
        BCF     S0,3
LABEL_TR1_ENDE;……

        BTFSS   S0,2
        GOTO    LABEL_ZV1_COUNT
        CLRF    ZV1
        GOTO    LABEL_ZV1_SETOUT

LABEL_ZV1_COUNT;……..

        BTFSS   S0,6
        GOTO    LABEL_ZV1_SETFD
        BTFSC   ZV1_FD,0
        GOTO    LABEL_ZV1_SETFD
        BSF     ZV1_FD,0
        BTFSC   S0,7
        GOTO    LABEL_ZV1_BACKWARD
        INCF    ZV1,F
        GOTO    LABEL_ZV1_SETOUT

LABEL_ZV1_BACKWARD;……..
        DECF    ZV1,F

LABEL_ZV1_SETOUT
LABEL_ZV1_SETFD;……..
        BTFSC   S0,6
        GOTO    LABEL_ZV1_EXIT
        BCF     ZV1_FD,0
```

```
LABEL_ZV1_EXIT
LABEL_UG5_1;……..
        BTFSS   S0,1
        GOTO    LABEL_UG5_FALSE

LABEL_UG5_2;…….
        BTFSS   S0,3
        GOTO    LABEL_UG5_FALSE

LABEL_UG5_TRUE;……
        BSF     S0,6
        GOTO    LABEL_UG5_ENDE

LABEL_UG5_FALSE;…….
        BCF     S0,6

LABEL_UG5_ENDE;……..
        BTFSC   LT1_FD,0
        GOTO    LABEL_LT1_SETFD
        BTFSS   S1,1
        GOTO    LABEL_LT1_SETFD
        MOVLW   13
        MOVWF   SYS_TMP1
        MOVLW   1
        MOVWF   SYS_TMP2

        CALL    LCD_LB1_GOTOXY
        MOVFW   ZV1
        MOVWF   SYS_TMP4
        CLRF    SYS_TMP5
        BSF     SYS_TMP10,0
        MOVLW   2
        MOVWF   SYS_TMP9
        CALL    LCD_LB1_DEZO100

LABEL_LT1_SETFD;……..
        BTFSS   S1,1
        GOTO    LABEL_LT1_SETFD2
        BSF     LT1_FD,0
        GOTO    LABEL_LT1_END

LABEL_LT1_SETFD2;……..
        BCF     LT1_FD,0

LABEL_LT1_END
LABEL_LG5_1;…….
        BTFSS   S0,1
```

```
            GOTO    LABEL_LG5_FALSE

LABEL_LG5_2;……
            BTFSC   S0,4
            GOTO    LABEL_LG5_FALSE

LABEL_LG5_TRUE;……
            BSF     S1,0
            GOTO    LABEL_LG5_ENDE

LABEL_LG5_FALSE;……..
            BCF     S1,0

LABEL_LG5_ENDE
LABEL_LG6_1;……..
            BTFSS   S1,0
            GOTO    LABEL_LG6_FALSE

LABEL_LG6_TRUE;…….
            BCF     STATUS,RP0      : Bank 0
            BSF     PORTB,1
            GOTO    LABEL_LG6_ENDE

LABEL_LG6_FALSE;……
            BCF     STATUS,RP0      : Bank 0
            BCF     PORTB,1

LABEL_LG6_ENDE;……
            MOVLW   5
            SUBWF   ZV1,W

LABEL_IF1_CHECK;……
            BTFSS   STATUS,C
            GOTO    LABEL_IF1_FALSE

LABEL_IF1_TRUE;……
            BSF     S0,5
            GOTO    LABEL_IF1_EXIT

LABEL_IF1_FALSE;……..
            BCF     S0,5

LABEL_IF1_EXIT;……
            BTFSC   LT2_FD,0
            GOTO    LABEL_LT2_SETFD
            BTFSS   S1,4
            GOTO    LABEL_LT2_SETFD
```

```
        MOVLW   0
        MOVWF   SYS_TMP1
        MOVLW   0
        MOVWF   SYS_TMP2
        CALL    LCD_LB1_GOTOXY
        MOVLW   LOW TAB_LT2
        MOVWF   SYS_TMP4
        MOVLW   HIGH TAB_LT2
        MOVWF   SYS_TMP5
        CALL    LCD_LB1_WRITESTR

LABEL_LT2_SETFD;……..
        BTFSS   S1,4
        GOTO    LABEL_LT2_SETFD2
        BSF     LT2_FD,0
        GOTO    LABEL_LT2_END

LABEL_LT2_SETFD2;……..
        BCF     LT2_FD,0

LABEL_LT2_END;……..

        BTFSC   LT3_FD,0
        GOTO    LABEL_LT3_SETFD
        BTFSS   S1,0
        GOTO    LABEL_LT3_SETFD
        MOVLW   0
        MOVWF   SYS_TMP1
        MOVLW   1
        MOVWF   SYS_TMP2
        CALL    LCD_LB1_GOTOXY
        MOVLW   LOW TAB_LT3
        MOVWF   SYS_TMP4
        MOVLW   HIGH TAB_LT3
        MOVWF   SYS_TMP5
        CALL    LCD_LB1_WRITESTR

LABEL_LT3_SETFD;……
        BTFSS   S1,0
        GOTO    LABEL_LT3_SETFD2
        BSF     LT3_FD,0
        GOTO    LABEL_LT3_END

LABEL_LT3_SETFD2;……
        BCF     LT3_FD,0

LABEL_LT3_END
```

```
LABEL_LG7_1;……
        BTFSS   S1,0
        GOTO    LABEL_LG7_FALSE

LABEL_LG7_2;…….
        BTFSS   S0,3
        GOTO    LABEL_LG7_FALSE

LABEL_LG7_TRUE;……..
        BSF     S1,1
        GOTO    LABEL_LG7_ENDE

LABEL_LG7_FALSE;……..
        BCF     S1,1

LABEL_LG7_ENDE;…….
        BTFSC   LT4_FD,0
        GOTO    LABEL_LT4_SETFD
        BTFSS   S1,2
        GOTO    LABEL_LT4_SETFD
        MOVLW   0
        MOVWF   SYS_TMP1
        MOVLW   1
        MOVWF   SYS_TMP2
        CALL    LCD_LB1_GOTOXY
        MOVLW   LOW TAB_LT4
        MOVWF   SYS_TMP4
        MOVLW   HIGH TAB_LT4
        MOVWF   SYS_TMP5
        CALL    LCD_LB1_WRITESTR
```

```
LABEL_LT4_SETFD;……
        BTFSS   S1,2
        GOTO    LABEL_LT4_SETFD2
        BSF     LT4_FD,0
        GOTO    LABEL_LT4_END

LABEL_LT4_SETFD2;……..
        BCF     LT4_FD,0

LABEL_LT4_END;…….
        BTFSC   MF1_FD,1
        GOTO    LABEL_MF1_CHKTIME
        BTFSC   MF1_FD,0
        GOTO    LABEL_MF1_SETFD
        BTFSS   S0,4
        GOTO    LABEL_MF1_SETFD

LABEL_MF1_RESET;…….
        CLRF    MF1
        CLRF    MF1_HI
        BSF     MF1_FD,1
        BSF     S1,2

LABEL_MF1_CHKTIME;……..
        MOVLW   0
        SUBWF   MF1_HI,W
        BTFSS   STATUS,Z
        GOTO    LABEL_MF1_CHECK
        MOVLW   8
        SUBWF   MF1,W

LABEL_MF1_CHECK;……
        BTFSS   STATUS,C
        GOTO    LABEL_MF1_SETFD

LABEL_MF1_TO;……..
        BCF     MF1_FD,1
        BCF     S1,2

LABEL_MF1_SETFD;……
        BTFSS   S0,4
        GOTO    LABEL_MF1_CLR
        BSF     MF1_FD,0
        GOTO    LABEL_MF1_ENDE

LABEL_MF1_CLR;……..
```

```
        BCF     MF1_FD,0

LABEL_MF1_ENDE;……
        MOVLW   30
        SUBWF   TR2_HI,W
        BTFSS   STATUS,Z
        GOTO    LABEL_TR2_CHECK
        MOVLW   132
        SUBWF   TR2,W

LABEL_TR2_CHECK;……..
        BTFSS   STATUS,C
        GOTO    LABEL_TR2_ENDE

LABEL_TR2_RESET;……..
        MOVLW   30
        SUBWF   TR2_HI,F
        MOVLW   132
        SUBWF   TR2,F
        BTFSS   STATUS,C
        DECF    TR2_HI,F

LABEL_TR2_SET;……..
        BTFSC   S1,3
        GOTO    LABEL_TR2_CLR
        BSF     S1,3
        GOTO    LABEL_TR2_ENDE

LABEL_TR2_CLR;……
        BCF     S1,3

LABEL_TR2_ENDE;…….
        BTFSC   LT5_FD,0
        GOTO    LABEL_LT5_SETFD
        BTFSS   S1,3
        GOTO    LABEL_LT5_SETFD
        MOVLW   0
        MOVWF   SYS_TMP1
        MOVLW   0
        MOVWF   SYS_TMP2
        CALL    LCD_LB1_GOTOXY
        MOVLW   LOW TAB_LT5
        MOVWF   SYS_TMP4
        MOVLW   HIGH TAB_LT5
        MOVWF   SYS_TMP5
        CALL    LCD_LB1_WRITESTR
```

```
LABEL_LT5_SETFD;……
        BTFSS   S1,3
        GOTO    LABEL_LT5_SETFD2
        BSF     LT5_FD,0
        GOTO    LABEL_LT5_END

LABEL_LT5_SETFD2;……
        BCF     LT5_FD,0

LABEL_LT5_END
LABEL_LG8_1;…….
        BTFSS   S1,3
        GOTO    LABEL_LG8_FALSE

LABEL_LG8_TRUE;……..
        BCF     S1,4
        GOTO    LABEL_LG8_ENDE

LABEL_LG8_FALSE;……
        BSF     S1,4

LABEL_LG8_ENDE
LABEL_LG9_1;…….
        BTFSS   S0,4
        GOTO    LABEL_LG9_FALSE

LABEL_LG9_2;……..
        BTFSC   S1,6
        GOTO    LABEL_LG9_FALSE

LABEL_LG9_TRUE;……
        BSF     S1,5
        GOTO    LABEL_LG9_ENDE

LABEL_LG9_FALSE;……..
        BCF     S1,5

LABEL_LG9_ENDE
LABEL_LG10_1;…….
        BTFSC   S1,5
        GOTO    LABEL_LG10_TRUE
LABEL_LG10_2;……..
        BTFSC   S0,5
        GOTO    LABEL_LG10_TRUE

LABEL_LG10_FALSE;…….
        BCF     S0,4
```

```
            GOTO        LABEL_LG10_ENDE

LABEL_LG10_TRUE;……..
            BSF         S0,4

LABEL_LG10_ENDE
LABEL_LG11_1;…….
            BTFSS       S0,4
            GOTO        LABEL_LG11_FALSE

LABEL_LG11_TRUE;……..
            BCF         STATUS,RP0          : Bank 0
            BSF         PORTB,2
            GOTO        LABEL_LG11_ENDE

LABEL_LG11_FALSE;…….
            BCF         STATUS,RP0          : Bank 0
            BCF         PORTB,2

LABEL_LG11_ENDE;……
            GOTO MAIN
TAB_LT2;……..
            RETLW 65                : A
            RETLW 115               : s
            RETLW 101               : e
            RETLW 110               : n
            RETLW 107               : k
            RETLW 114               : r
            RETLW 111               : o
            RETLW 110               : n
            RETLW 32                :
            RETLW 109               : m
            RETLW 111               : o
            RETLW 116               : t
            RETLW 111               : o
            RETLW 114               : r
            RETLW 32                :
            RETLW 32                :
            RETLW 0                 : end
TAB_LT3;……
            RETLW 89                : Y
            RETLW 105               : i
            RETLW 108               : l
            RETLW 100               : d
            RETLW 105               : i
            RETLW 122               : z
            RETLW 32                :
```

```
                RETLW 67            : C
                RETLW 97            : a
                RETLW 108           : l
                RETLW 115           : s
                RETLW 58            : :
                RETLW 0             : end

TAB_LT4;……
                RETLW 117           : u
                RETLW 99            : c
                RETLW 103           : g
                RETLW 101           : e
                RETLW 110           : n
                RETLW 32            :
                RETLW 99            : c
                RETLW 97            : a
                RETLW 108           : l
                RETLW 105           : i
                RETLW 115           : s
                RETLW 109           : m
                RETLW 97            : a
                RETLW 32            :
                RETLW 32            :
                RETLW 0             : end

TAB_LT5;……
                RETLW 121           : y
                RETLW 105           : i
                RETLW 108           : l
                RETLW 100           : d
                RETLW 105           : i
                RETLW 122           : z
                RETLW 32            :
                RETLW 117           : u
                RETLW 99            : c
                RETLW 103           : g
                RETLW 101           : e
                RETLW 110           : n
                RETLW 32            :
                RETLW 99            : c
                RETLW 97            : a
                RETLW 108           : l
                RETLW 0             : end
                END
```

UYGULAMA FAALİYETİ

Programlama işlemini yapınız.

İşlem Basamakları	Öneriler
➢ Kurulacak sistem için ihtiyaçları (devre elemanlarını) tespit ediniz. ➢ İhtiyacınızı karşılayacak mikrodenetleyiciyi seçiniz. ➢ Sisteminin çalışması için gerekli programı yazınız. ➢ Programı mikrodenetleyiciye yükleyiniz. ➢ Çevre elemanları ile devreyi kurunuz.	➢ Mikrodenetleyici olarak PIC 16F84 kullanınız. ➢ Programı yazdıktan sonra MPLAB ile deneyiniz. ➢ Programı mikrodenetleyiciye yüklerken kullanılan PIC programlayıcıya uygun yazılım kullanınız. ➢ Devreyi Proteus programında çalıştırarak deneyiniz.
➢ Devre için gerekli giriş elemanları ve özelliklerini belirleyiniz. ➢ Devre için gerekli çıkış elemanları ve özelliklerini tespit ediniz. ➢ Devrenin baskı devre şemasını çıkartınız. ➢ Devre elemanlarının ve mikrodenetleyicinin montajını yapınız.	➢ Kullandığınız devre elemanlarının özelliklerini internetten araştırınız. ➢ Baskı devreyi, devre şemasını proteus programından "ares" programına aktrarak çıkarınız. ➢ Devrenin montajını yapmadan önce bredboard üzerinde kurarak deneyiniz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet,** kazanamadıklarınız için **Hayır** kutucuklarına (X) işareti koyarak öğrendiklerinizi kontrol ediniz.

	Değerlendirme Ölçütleri	Evet	Hayır
1.	Kurulacak sistem için ihtiyaçları tespit ettiniz mi?		
2.	İhtiyacı karşılayacak mikrodenetleyiciyi seçtiniz mi?		
3.	Sistemin mikrodenetleyici programını yazdınız mı?		
4.	Programı mikrodenetleyiciye yüklediniz mi?		
5.	Çevre elemanları ile devreyi kurdunuz mu?		

DEĞERLENDİRME

Değerlendirme sonunda "Hayır" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "Evet" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlede boş bırakılan yere getirilecek bilgilerin bulunduğu seçeneği işaretleyiniz.

1. INTCON kaydedicisi ……uygulamalarında kullanılır. Boşluğa aşağıdakilerden hangisi gelmelidir?
   A) Giriş
   B) Çıkış
   C) Kesme
   D) Reset

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

2. Status registerinde bulunan RP0 ve RP1 bitlerinin görevi aşağıdakilerden hangisidir?
   A) Kesmenin geçerli olması için kullanılır.
   B) İşlemlerde taşma meydana geldiğinde kullanılır.
   C) İşlemlerin sonucu sıfır olduğunda kullanılır.
   D) Bank değiştirmek için kullanılır.

3. PIC16F84'te kesme meydana geldiğinde program hangi adrese atlamaktadır?
   A) H'00'
   B) H'02'
   C) H'04'
   D) H'06'

4. Aşağıdaki seçeneklerden hangisi PIC'i programlamak için gerekli değildir?
   A) MPSAM ASM derleyici
   B) PIC programlayıcı
   C) MPLAB PFE
   D) PROTEUS programı

5. LCD'ye veri yazmak için E girişine hangi kenar tetiklemeli sinyal uygulanmalıdır?
   A) Düşen kenar
   B) Yükselen kenar
   C) Pozitif seviye
   D) Negatif seviye

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki öğrenme faaliyetine geçiniz.

ÖĞRENME FAALİYETİ-2

AMAÇ

Mikrodenetleyiciyi ve çevre elemanlarını seçebilecek, dijital işlem için gerekli programı hatasız olarak yazabilecek, programı mikrodenetleyiciye yükleyebilecek ve devreyi hatasız olarak kurup çalıştırabileceksiniz.

ARAŞTIRMA

- Red röleler hakkında bilgi edinin.
- Kesme (Interrupt) çeşitlerini öğrenmeli kesme alt programları yazınız.
- Asenkron motorların yapısını ve çalışmasını öğreniniz.

Araştırma işlemleri için Mikrodenetleyici Programlama modülünü gözden geçirirken internet ortamlarından da yararlanabilirsiniz.

2. İLERİ SEVİYE DİJİTAL UYGULAMA DEVRELERİ

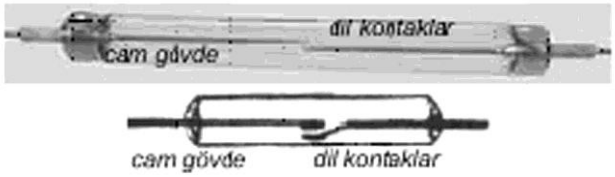
2.1. Dört Girişli Kapı Pencere Alarm Devresi

Dört girişli olan sistemde kapı ve pencerelere yerleştirilen anahtarlar kapalı (OFF) konumunda kabul edilecektir. Anahtarlardan biri konum değiştirdiğinde (kapı veya pencere açıldığında) sistem bir led ve buzer ile çıkış verecek, "DUR" butonuna basılıncaya kadar sesli ve ışıklı ikaz devam edecektir.

Devreye enerji uygulandıktan 1 dk. sonra alarm aktif olmaktadır. Bu sürede içeride bulananlar dışarı çıkabilir. Dışarıdan içeriye girebilmek için kapının dışına bir reed röle yerleştirilir. Bu anahtar kapandığında alarm 1 dk. pasif olur ve bu süre içinde sistem kapatılır. 1 dk. süre kullanıcıya göre değiştirilebilir.

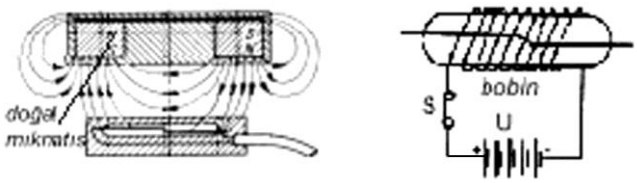
- Reed (Dil Kontaklı) Röleler

Cam gövde içine konmuş minik kontaklara sahip elemanlara reed röle denir. Reed rölelerde havası alınmış şeffaf cam ya da başka bir maddeden yapılmış olan muhafaza içinde bulunan demir-nikel alaşımı mini kontakların konumu sabit mıknatıs ya da elektromıknatısla değiştirilir. Resim 2.1'de cam gövdeli reed rölelerin yapısı verilmiştir.



Resim 2.1: Reed röle

Reed rölelerde kontakların konumu Resim 2.2'de görüldüğü gibi doğal mıknatısla ya da elektromıknatıs ile değiştirilebilmektedir. Reed rölelerin kontaklarının çekme ve bırakma zamanı 0,5 mili saniye, çalışma sayısı ise 1-2 milyon adet dolayındadır.



Resim 2.2: Reed rölenin konum değiştirmesi

- Kesmeler (Interrupt)

Kesme bir mikrokontrolörün dışarıda olan herhangi bir olaya anında müdahele etmesini sağlamaktadır. Kesme olunca mikrokontrolör yapmakta olduğu işi bırakır ve H'04' adresindeki kesme alt programına gider. Kesmeden RETFIE (RETurn From IntErrupt) komutu ile dönülür ve program kaldığı yerden devam eder. Kesme dışarıdan veya içeriden olabilir. Dışarıdan gelen kesmelerin zamanı bilinmemektedir. İçeriden gelen kesmeler ise genellikle mikrokontrolörün sayaç devrelerine bağlı gelmektedir.

PIC 16F84 mikrokontrolörde dört çeşit kesme bulunur.

- RBO / INT bacağından gelen dış (haricî) kesme
- Port B (4,5,6,7) bacaklarında lojik seviye değişikliğinden oluşan dış kesme
- TMR0 sayıcısında oluşan zaman aşım iç kesmesi
- EEPROM belleğe yazıldıktan sonra oluşan iç kesme

Herhangi bir kesmenin geçerli olması için

- INTCON registeri GIE = lojik 1 yapılınca tüm kesmeler aktif olur.
- Hangi kesmenin geçerli olması isteniyorsa o kesme biti lojik 1 yapılmalıdır. Kesmeler INTCON registerinin aşağıdaki bitleri ile kontrol edilir.
  - INTE: RBO / INT dış kesmesini aktif yapma bayrağı
  - RBIE: RB4 – RB7 bacaklarında değişiklik kesmesini aktif yapma bayrağı
  - T0IE: TMR0 sayıcı kesmesini aktif yapma bayrağı
  - EEIE: EPROM belleğe yazma işlemi tamamlama kesmesi

Kesme meydana gelince o kesmeye ait INTCON registerinde bulunan bayrak aktif olur.

- INTF: Lojik 1 ise RBO / INT kesmesi var.
- RBIF: Lojik 1 ise RB4 – RB7 kesmesi var.
- T0IF: Lojik 1 ise TMR0 sayıcı kesmesi var.
- EEIF: Lojik 1 ise EEPROM kesmesi var.

Alarm devresinde dört giriş RB4 – RB7 bacaklarına bağlanımıştır. Alarm aktif olduktan sonra bu girişlerden birinde herhangi bir değişiklik olduğunda "KESME" alt programı çalıştırılır ve buzzer çalışır. Kesme oluştuktan sonra ana programa geri dönülmez. Ana programa dönmek için "DUR" butonuna basılması gerekir.

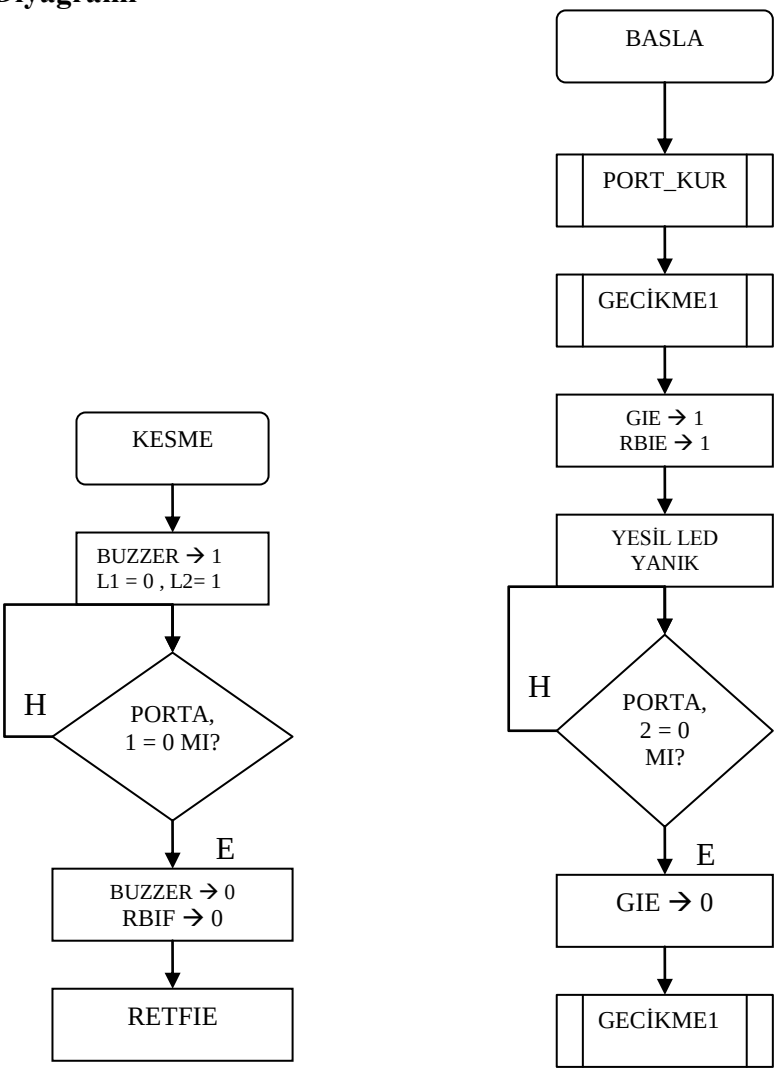
RB4 – RB7 değişiklik kesmesinin oluşması için aşağıdakiler sırasıyla yapılır.

- INTCON – GIE bayrağı "lojik 1"
- INTCON - RBIE bayrağı "lojik 1"
- OPTION registerin 6. biti (INTEDG) ile dışarıdan gelen sinyalin kenar tetiklemesi ayarlanır. (Lojik 1 → yükselen kenar, Lojik → 0 düşen kenar)
- Kesme oluştuğunda RBIF bayrağı "lojik 1" olur. Kesmelerin geçerli olması için kesme alt programında "lojik 0 " yapılır.

2.1.1. Devrenin Malzemeleri

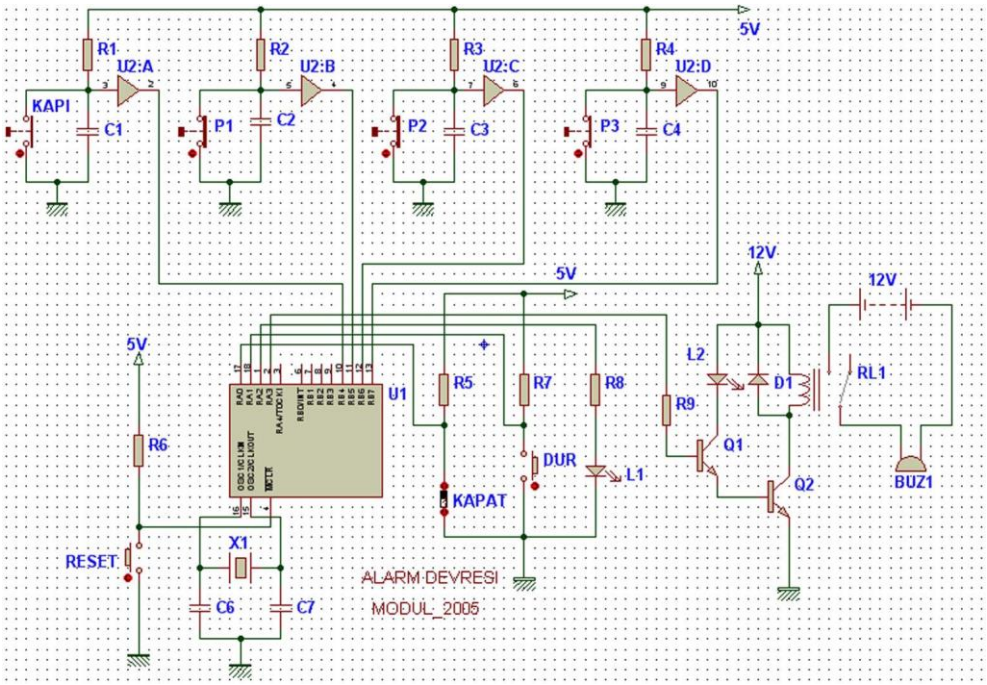
- PIC16F84 4MHz mikrodenetleyici
- 4MHz kristal, 2xbuton, led
- U2= 7407 veya 4050, Reed Röle
- C6 = C7 = 22pf
- C1= C2= C3= C4= 100nF
- R8= 330Ω, R9= 1 K direnç
- R5= R6= R7= 10 K
- R1 = R2 = R2 = R4 = 100 KΩ
- Q1 = BC237 , Q2 = BD135
- 12V Röle , D1= 1N4148

2.1.2. Akış Diyagramı



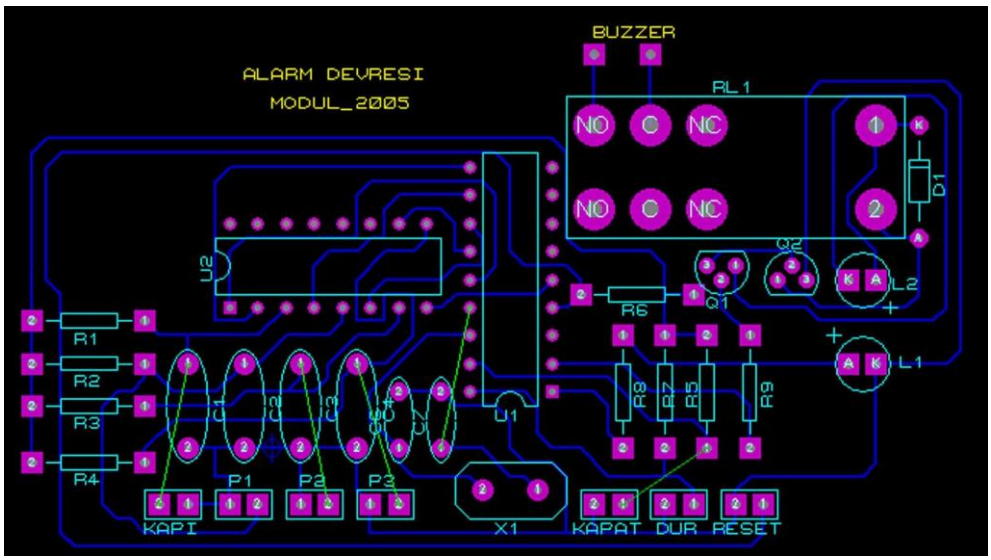
Tablo 2.1: Alarm devresi akış diyagramı

2.1.3. Devrenin Şeması



Şekil 2.1: Alarm devresi

2.1.4. Baskı Devresi



Şekil 2.2: Alarm baskı devresi

2.1.5. Devrenin Asm Programı

Devreye enerji uygulandığında, yaklaşık 60 sn. sonra alarm aktif olur. Alarmın aktif olduğu yeşil ledin (L1) yanmasıyla anlaşılır. Kapı ya da pencerelerden biri açıldığında RB4-RB7 port girişlerinde değişlik meydana geldiğinden kesme aktif olur. Bu durumda yeşil led söner, buzzer çalışır ve kırmızı led (L2) yanar.

Alarmı kapatmamak için DUR butonuna basılır ve program çalışmasına ILK etiketinden itibaren devam eder.

```
;=========ALARM DEVRESİ UYGULAMA PROGRAMI======S_2005========

LIST P=16F84
INCLUDE "P16F84.INC"
SAYAC1   EQU H'0C'
SAYAC2   EQU H'0D'                    : Gecikme alt programı için sayaç
SAYAC3   EQU H'0E'

         ORG      H'00'
         GOTO     BASLA
         ORG      H'04'
         GOTO     KESME

BASLA;…….

         CALL     PORT_KUR            : Portları ayarla
ILK      CALL     GECIKME1
         BSF      INTCON , RBIE       : RBIE → 1
         BSF      INTCON, GIE         : GIE → 1
         BSF      PORTA,2             : L1 (yeşil led) yanık, alarm aktif
BUTON    BTFSC    PORTA,0             :  KAPAT    butonuna basılı mı?
         GOTO     BUTON               : Hayır , kesme bekle
         BCF      INTCON,GIE          : Evet GIE → 0 kesmeler pasif
         CALL     GECIKME1            : 1dk. gecikme
         GOTO     ILK

PORT_KUR;…….

         BSF      STATUS,5            : Bank 1'e geç
         MOVLW    H'FF'
         MOVWF    TRISB               : PortB giriş
         MOVLW    H'03'
         MOVWF    TRISA
         BCF      STATUS,5            : Bank 0'a geç
         RETURN
```

```
GECIKME1;……..
                                        :60 sn'lik gecikme sağlar.
        MOVLW   D'255'
        MOVWF   SAYAC1
G1      MOVLW   D'255'          : W ← D'255'
        MOVWF   SAYAC2          : SAYAC2 ← W
G2
        MOVLW   D'255'          : W ← D'255'
        MOVWF   SAYAC3          : SAYAC3 ← W
G3
        DECFSZ  SAYAC3,F        : SAYAC3= SAYAC3 - 1
        GOTO    G3
        DECFSZ  SAYAC2,F        : SAYAC2= SAYAC2 - 1
        GOTO    G2
        DECFSZ  SAYAC1,F        : SAYAC1= SAYAC1 - 1
        GOTO    G1
        RETURN

KESME;……..

        BSF     PORTA,3         : Buzzer ve kırmızı led (L2) aktif
        BCF     PORTA,2         : Yeşil ledi söndür.
DON     BTFSC   PORTA,1         : DUR butonuna basılı mı?
        GOTO    DON             : Hayır, kesme programında kal
        BCF     PORTA,3         : Evet
        BCF     INTCON,RBIF     : RBIF → 0 bayrağını sıfırla
        RETFIE                  : kesmeden geri dön
        END
```

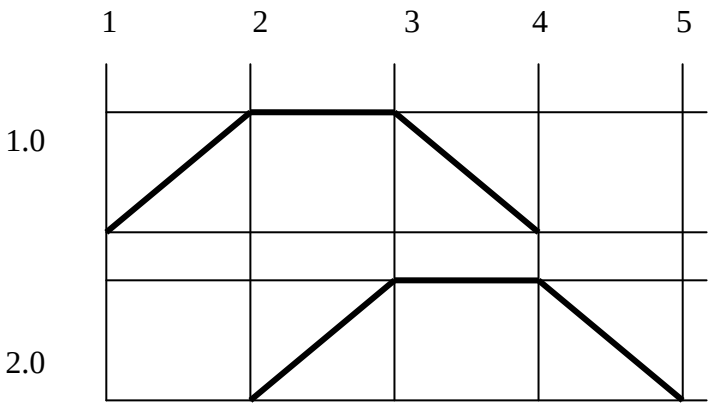
2.2. Elektropnomatik Sistemin PIC ile Kumandası

Pnömatik ve elektrik-teknolojisinin bir arada kullanılması endüstriyel otomasyon uygulamalarında önemli rol oynar. Sanayide hızlı, ucuz ve kaliteli üretim sağlar.

Pic ile uygulanacak olan sistemde, iki silindir, parçaları besleme sütunundan taşıyıcı bir sisteme nakleder. Bir düğmeye basıldığında 1.0 silindiri ileri doğru hareket eder. Besleme sütunundan bir parçayı iter. Bu parçayı 2.0 silindiri için konumlandırır. 2.0 silindiri parçayı taşıyıcı sisteme iter. Bundan sonra ilk olarak birinci, ardından ikinci silindir geri gelir. İki silindirin de ileri ve geri geliş hızlarının ayarlanabilir olması gerekir. Ayrıca iki silindirin de öndeki ve arkadaki son konumlarının algılanması gerekir.

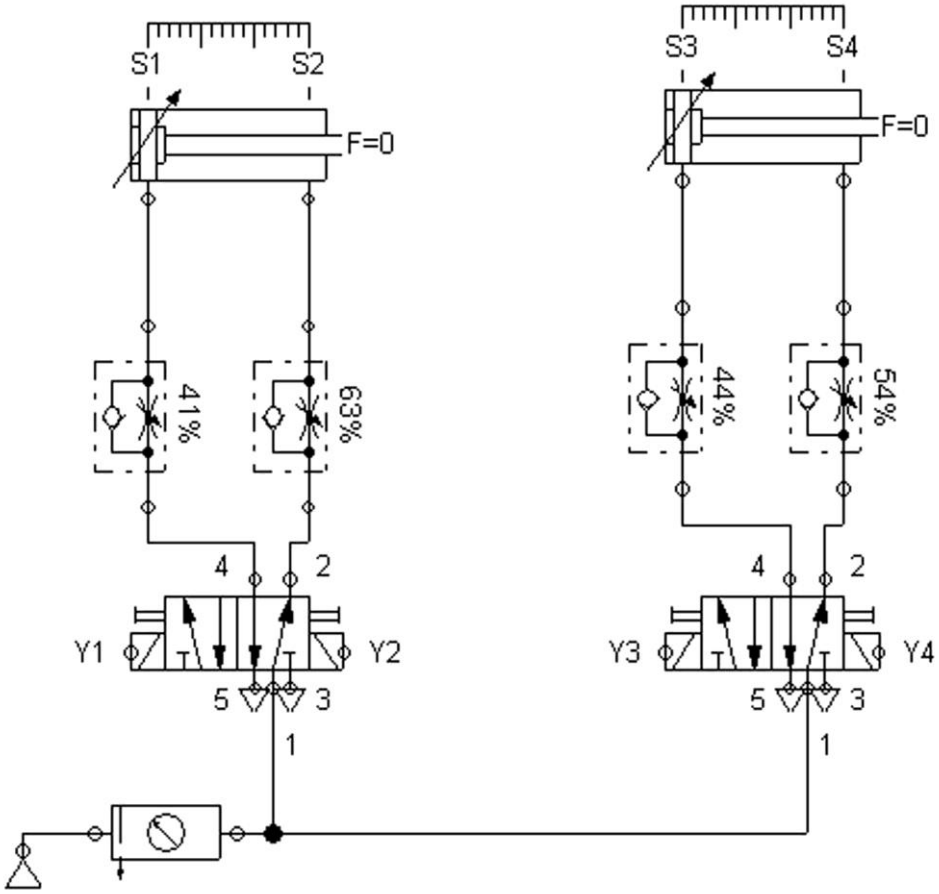
Silindirin hareket akışı manyetik anahtarla kontrol edilir. Sistem çalıştırıldığında aşağıdaki sıra ile silindirler hareket eder.

- 1.0 silindiri geride - S1 anahtarı kapalı, 2.0 silindiri geride ve S3 anahtarı kapalıdır.
- "S" anahtarına basıldığında 1.0 silindiri ileri hareket eder. S2 anahtarı kapanır.
- S2 kapandığında 2.0 silindiri ileri doğru hareket eder. S4 silindiri kapanır.
- S4 kapandığında 1.0 silindiri geriye doğru hareket eder. S1 anahtarı kapanır.
- S1 kapandığında 2.0 silindiri geriye doğru hareket eder. S3 anahtarı kapanır.
- Sistem tekrar başa döner. İşlem sırasının her adımı yol – adım şemasında gösterilir.



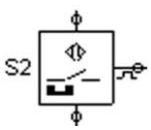
Tablo 2.2: Yol – adım şeması

- 1- 2 aralığında Y1 rölesi enerjilenir. 1.0 ileri hareket eder.
- 2 - 3 aralığında Y3 enerjilenir 2.0 ileri hareket eder. Y1 enerjili ve 1.0 ileridedir.
- 3 - 4 aralığında Y2 enerjilenir. 1.0 geri hareket eder. Y3 enerjili ve 2.0 ileridedir.
- 4 – 5 aralığında Y4 enerjilenir. 2.0 geri hareket eder.



Şekil 2.3: Elektropnömatik devre şeması

Manyetik algılayıcılar (üç hatlı algılayıcılar) doğrudan silindir gövdesi üzerine yerleştirilir. Silindir anahtarı geçtiğinde manyetik alan oluşur ve anahtarın çıkışında bir sinyal elde edilir.



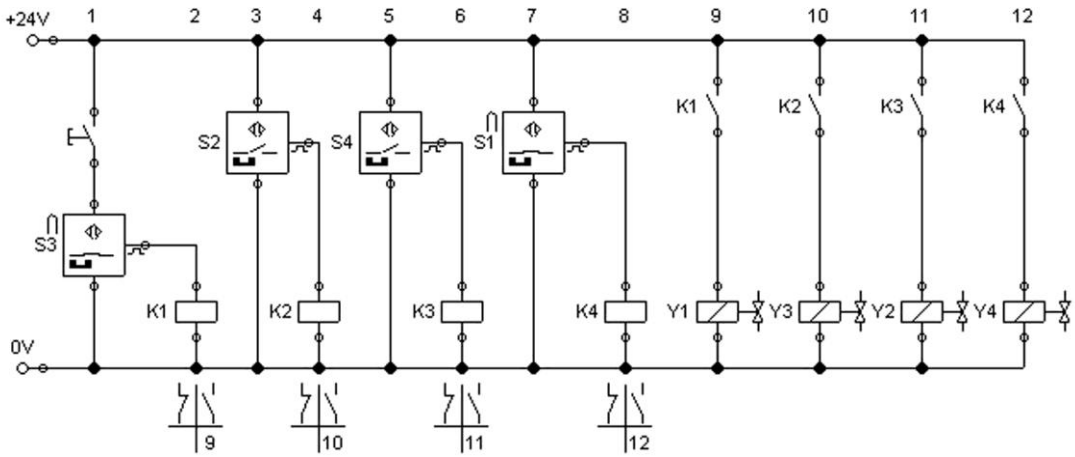
Şekil 2.4: Manyetik algılayıcılar

Devrede S1, S2, S3 ve S4 ile gösterilen girişlere sisteme yerleştirilen manyetik anahtarların çıkışları bağlanır. Çıkış gerilimi yüksek olduğundan 5V seviyesine düşürülmelidir. Y1 ve Y2 röleleri 1. silindiri, Y3 ve Y4 röleleri 2. silindiri kontrol eder.

Port A giriş, Port B çıkış olarak yönlendirilmiştir. Silindirleri kontrol etmek için PortB çıkışlarına Tablo 2.3'teki değerler uygulanır.

GERİ2	İLERİ2	GERİ1	İLERİ1	HAREKET
Y4	Y3	Y2	Y1	
RB7	RB6	RB5	RB4	
1	0	1	0	1.0 ve 2.0 silindirleri geride (h'A0')
1	0	0	1	1.0 ileriye , 2.0 geride (h'90')
0	1	0	1	1.0 ileride, 2.0 ileriye (h'50')
0	1	1	0	1.0 geriye, 2.0 ileride (h'60')
1	0	1	0	1.0 geride, 2.0 geriye (h'A0')

Tablo 2.3: Port B çıkışlarına uygulanan değerler

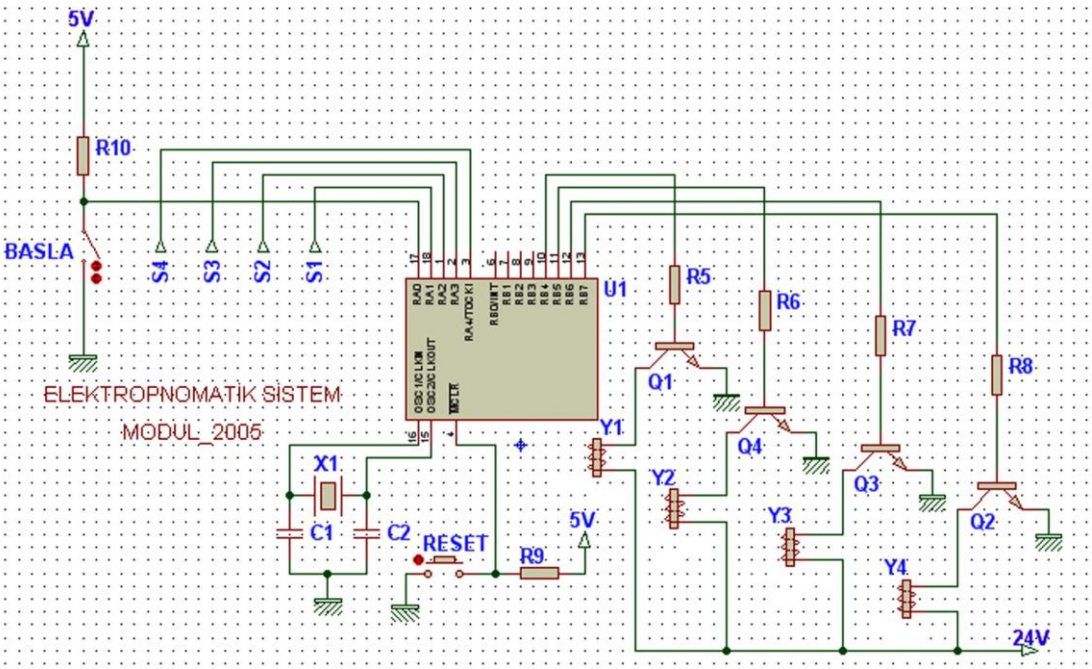


Şekil 2.5: Sistemin elektriksel bağlantı şeması

2.2.1. Devrenin Malzemeleri

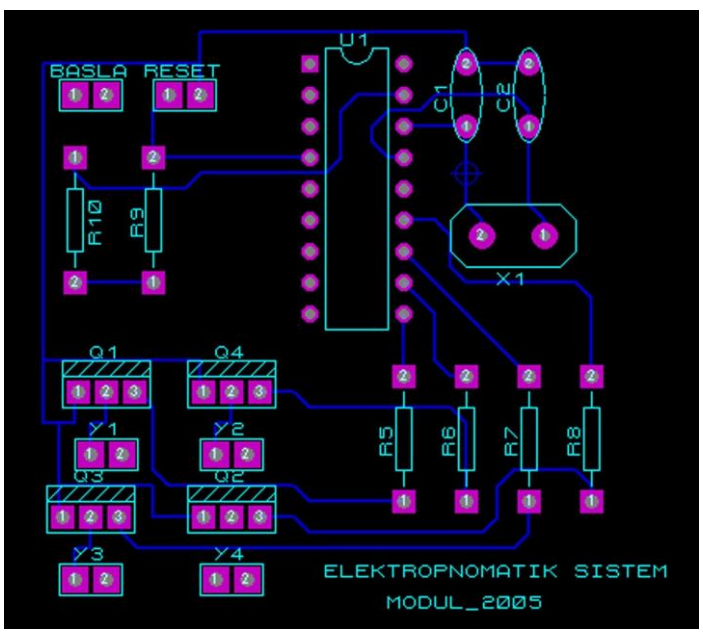
- PIC16F84 4MHz mikrodenetleyici
- 4MHz kristal, 2xbuton
- C1 = C2 = 22pf
- R5= R6= R7= R8= 1 K
- R9 = R10 = 10 KΩ
- Q1= Q2 = Q3 = Q4 = BD135

2.2.2. Devrenin Şeması



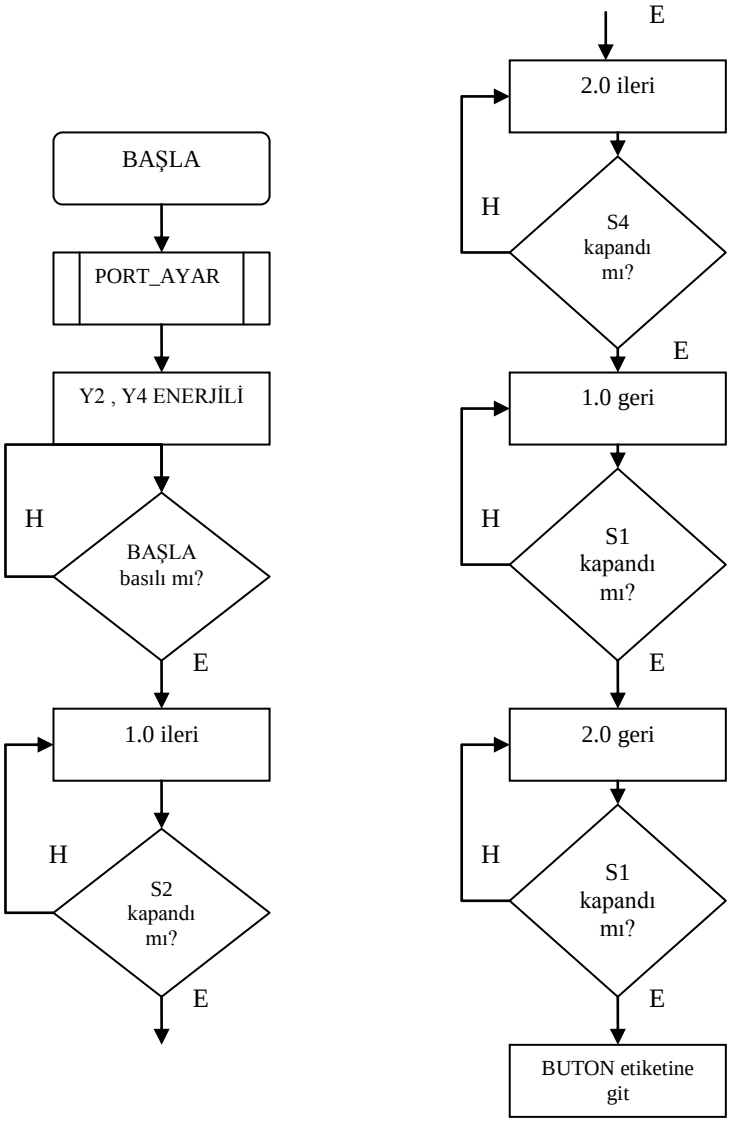
Şekil 2.6: Elektropnomatik sistemin PİC ile uygulama devresi

2.2.3. Baskı Devresi



Resim 2.7: Elektropnomatik sistemin baskı devresi

2.2.4. Devrenin Akış Diyagramı



Tablo 2.4: Elektropnomatik sistemin akış diyagramı

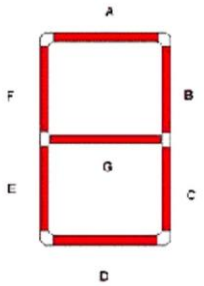
2.2.5. Devrenin Asm Programı

```
;=========ELEKTROPNOMATIK SISTEM PROGRAMI======S_2005========

LIST      P=16F84
INCLUDE   "P16F84.INC"

          ORG       H'00'
          GOTO      BASLA
BASLA
          CALL      PORT_AYAR
          MOVL      W H'A0'          : 1.0 ve 2.0 silindirleri geride
          MOVWF     PORTB
BUTON     BTFSC     PORTA,0
          GOTO      BUTON            : START butonuna basılı mı?

S2        MOVLW     H'90'            : 1.0 silindirini ileri hareket ettir.
          MOVWF     PORTB
          BTFSS     PORTA,2          : 1.0 silindiri ileride mi?
          GOTO      S2               : HAYIR 1.0 ileri- Y1 enerjili
S4        MOVLW     H'50'
          MOVWF     PORTB            : 2.0 silindirini ileri hareket ettir.
          BTFSS     PORTA,4          : 2.0 silindiri ileride mi?
          GOTO      S4               : HAYIR Y1 ve Y4 enerjili
S1        MOVLW     H'60'            : 1.0 silindirini geri hareket ettir
          MOVWF     PORTB            : 2.0 silindiri ileride

          BTFSS     PORTA,1          : 1.0 silindiri geride mi?
          GOTO      S1               : HAYIR Y2 ve Y4 enerjili
S3        MOVLW     H'A0'
          MOVWF     PORTB

          BTFSS     PORTA,3
          GOTO      S3
          GOTO      BUTON

PORT_AYAR
          BSF       STATUS,5
          MOVLW     H'FF'
          MOVWF     TRISA
          CLRF      TRISB
          BCF       STATUS,5
          RETURN
          END
```

2.3. Programlanabilir Zamanlayıcı

Bu programlanabilir sayıcı ev içi, ticarî ve endüstriyel uygulamalar için kullanılışlıdır. Ayarlanan zaman sonunda açma-kapama yapmaktadır. Devrenin zaman periyodu programla değiştirildiğinden oldukça işlevselidir.

Devre saat, dakika ve saniye olarak ayarlanır, ardından "BAŞLA" butonuna basılır ve 1sn. aralıkla geriye doğru saymaya başlar. Sayma işlemi bittiğinde bir çıkış aktif olur ve led yanar. Led yerine bir transistör ve röle bağlanarak herhangi bir kontrol yapılabilir. Devrede gösterge olarak 6 X 7 segment ortak anotlu display kullanılmıştır. Şekil 2.8'de 7 parçalı göstergenin yapısı gösterilmiştir.



Şekil 2.8: 7 parçalı gösterge

7 parçalı göstergeler ortak anot ve ortak katot olmak üzere iki çeşittir. PIC ile doğrudan denetlendiğinde, 7 bacak PortB'ye bağlanır. Port B'den gelen bilgilerin, göstergede oluşturduğu şekil, Port B'ye aktrılan bilgiye bağlıdır. Örneğin, segmentte 0 olması için, A, B, C, D, E, F girişlerinin ortak anot için "0" , ortak katot için "1" olması gerekir. Gösterge ortak katotlu olduğunda Port B çıkışları "00111111" olmalıdır. Bu atama ikilik olarak yapılabileceği gibi onluk veya onaltılık olarak da yapılabilir. Yukarıdaki ikilik sayının karşılığı D "63" , H "3F" tir. Yani onluk 63 sayısı ikilik olarak ifade edildiğinde, karşılığı B "00111111"dir. Bu da Port B'de "0" gösterir. Sonuçta Port B'ye atanan onaltılık veya onluk sayısının ikilik karşılığıdır.

Onluk	Onaltılık	İkilik	7 Parçalı Gösterim
63	3F	00111111	0
6	06	00000110	1
91	5B	01011011	2
79	4F	01001111	3
102	66	01100110	4
109	6D	01101101	5
125	7D	01111101	6
7	07	00000111	7
127	7F	01111111	8
111	6F	01101111	9

Tablo 2.5: 7 parçalı ortak katotlu gösterge çevrim tablosu

Tablo 2.5'te çevrim tablosuna onaltılık veya onluk olarak yazdığımız sayı, ikilik karşılığına çevrilerek PIC portlarından göstergeye aktarılır ve bir görüntü oluşur.

Göstergeye veri göndermek için programda CALL – RETLW komut ikilisi kullanılır. "RETLW" komutu "RETURN" komutu gibi bu da alt programdan ana programa dönüş için kullanılır. Bu komut ana programa dönüşte karşısında yazılan değeri Write kaydedciye yazar. Bunun için program counter (PC) kullanılır.

PC'ler PIC'lerde kullanılan bir kaydedicidir. PIC belleğinin değerine göre 10 ila 14 bitlik olabilir. 10 bit yani 210 = 1024 byte olduğundan ve PIC16F84'ün de 1 Kbyte'lık bir program belleği olduğundan bunu adreslemek için sayıcıya 10-bit yeterlidir.

PIC16F84'in PC'si 13 bittir. İlk 8 biti PCL'dir ve bununla 256 byte adreslenir. Diğer 5 biti PCH'tir ve bunun doğrudan okunup yazılması mümkün değildir, bu taraf ancak "PLATCH" kaydedicisi ile kullanılır. Programda PC olarak tüm kaydedici bitleri kullanılırsa "PLATCH" kaydedicisinin üst 2 bitini sıfırlamak gerekir. Bu sebeple dar kapasiteli kullanımlarda PC yerine PCL yazılarak bunun önüne geçilir. (ADDWF PCL, F)

Devrede kullanılan göstergeyi sürmek için 7447 entegresi kullanılmıştır. 7447 BCD kodunu 7 parçalı göstergeye dönüştürür. 7447 aktif sıfırdır ve ortak anotlu göstergede kullanılır kullanılır. Örneğin, displayde "0" yazdırmak için girişine "0000" uygulanırken çıkışında "0000001" elde edilir. Bu şekilde G ledi dışında tüm ledler yanar.

Gönderdiğimiz veri 6 göstergenin sadece birinde görüntülenir. Bunun için gösterge seçmek gerekir. Ortak anotlu göstergede "1", ortak katotlu displayde ise "0 "uygulanarak seçme yapılır. Göstergenin 6 girişinden hangisi "1" olursa o gösterge seçilir. Gösterge seçmek için

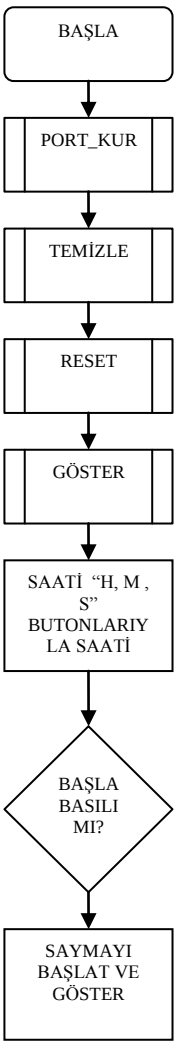
- Seçme işleminde 6 girişin tamamı PIC'in portlarına bağlanır. "1" bilgisi tüm girişlere kaydırılarak gönderilir. Tarama süresi çok kısa olduğundan göstergedeki tüm ledlerde aynı anda çıkış elde edilir.
- Port sayısı yeterli değilse seçme işleminde sayıcı ya da kaydıran kaydedici entegreleri kullanılır. Devremizde 4017'nin 10'lu sayıcı entegresi kullanılmıştır. Sayıcıya saat sinyali uygulandığında çıkışındaki veriyi kaydırır ve sayma işlemini gerçekleştirir. Entegrenin aktif olması için Enable girişi "0" olmalıdır. Göstergeye veri yazma işleminde sayıcı entegresi resetlenmelidir. Çünkü entegrenin tüm çıkışları kullanılmamıştır.

Devrede üç ayrı ayar butonu bulunmaktadır. Bunlar:

- H butonu saati ayarlamak için kullanılır. Saat 0 – 24 olarak ayarlanmıştır.
- M butonu dakikayı ayarlamak için kullanılır.
- S butonu ise saniyeyi ayarlamak için kullanılır.

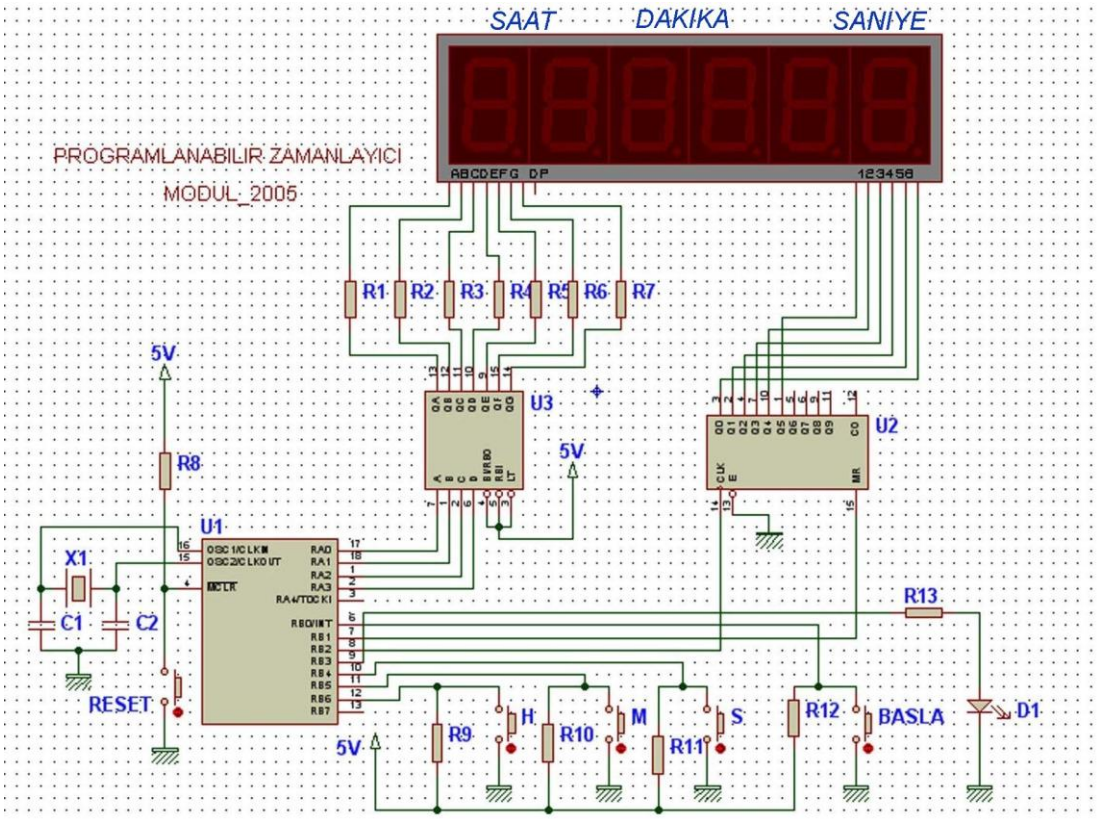
2.3.1. Devrenin Malzemeleri ve Akış Diyagramı

- PIC16F84 4MHz mikrodenetleyici
- 4MHz kristal, 5xbuton , C1 = C2 = 22pf
- 6x 7-parçalı gösterge, LED
- R1 = R2 = R3 = R4 = R5= R6 = R7 = R13 = 330 Ω
- R8 = R9 = R10 = R11= R12 = 10 KΩ
- 7447 ve 4017 entegreleri



Tablo 2.6: Sayıcı akış diyagramı

2.3.2. Devrenin Şeması



Şekil 2.9: Zamanlayıcı devresi

2.3.3. Devrenin Asm Programı

```
                LIST        P=16F84
                INCLUDE     "P16F84.INC"
I EQU H'0C'
STATUS_Y EQU H'0D'
W_Y EQU H'0E'
SAYI EQU H'10'
SN EQU H'1A'
COUNT EQU H'1C'
SAYAC1 EQU H'20'
SAYAC2 EQU H'21'
                ORG         H'00'
                GOTO        BASLA
                ORG         H'04'
                GOTO        KESME
BASLA;.....................................................................................................
                BCF         PORTB,3
                CLRWDT
```

```
            CALL    PORT_KUR        : Portları kur.
            CALL    TEMIZLE         : Register içeriklerini sıfırla.
            CALL    RESET           : Entegreyi sıfırla.
            CALL    GOSTER          : Display'lere yaz.

TUS_ARA     BTFSS   PORTB,4
            CALL    SANIYE
            BTFSS   PORTB,5
            CALL    DAKIKA
            BTFSS   PORTB,6
            CALL    SAAT
            BTFSS   PORTB,0
            GOTO    SAY
            CALL    GOSTER
            GOTO    TUS_ARA
SAY
            BSF     STATUS,5        : Bank 1 seç.
            MOVLW   B'10000100'     : TMR0 seçili
            MOVWF   OPTION_REG      : Prescaler d <-- 32
            BCF     STATUS, 5       : Bank 0 seç.
            CALL    GOSTER
            GOTO    SAY

PORT_KUR;.....
            BSF     STATUS,5        : Bank 1 seç.
            MOVLW   H'00'           : Port A çıkış
            MOVWF   TRISA
            MOVLW   H'F8'
            MOVWF   TRISB
            MOVLW   B'10100000'     : GIE --> 1, T0IE -->1
            MOVWF   INTCON
            BCF     STATUS,5        : Bank 0 seç.
            RETURN

TEMIZLE;.....
            MOVLW   SAYI            : İlk register adresini tut.
            MOVWF   FSR
            MOVLW   H'0F'           : 15 adet değişkeni
            MOVWF   I               : sıra ile
SIFIR       CLRF    INDF            : sıfırla
            INCF    FSR
            DECFSZ  I
            GOTO    SIFIR
            RETURN
```

```
GOSTER;.....
        CALL    RESET           : Display reset
        MOVLW   6               : sayı dizisinin boyutu
        MOVWF   I               ;i <-- 4
        MOVLW   SAYI            : sayı dizisinin başlangıç adresini
        MOVWF   FSR             : FSR'e yaz.
TARA    MOVF    INDF ,W         : dizi elemanını W'e yaz.
        MOVWF   PORTA           : Port A'dan gönder.
        CALL    GECIKME         : Bir süre bekle.
        INCF    FSR             : Dizinin bir sonraki elemanına ulaş.
        CALL    PALS            : Bir sonraki dijiti seç.
        DECFSZ  I               : Dizinin son elemanına ulaşıldı mı?
        GOTO    TARA            : Hayır taramaya devam et.
        RETURN                  : Evet alt programdan çık.

GECIKME;.....
        MOVLW   H'FF'           : COUNT <-- H'FF'
        MOVWF   COUNT
DON     DECFSZ  COUNT,F         : COUNT=COUNT-1, C=0 MI?
        GOTO    DON             : Hayır. DON etiketine git.
        RETURN                  : Evet. Alt programdan çık.

PALS;.....

        BSF     PORTB,2         : CLK -->1
        BCF     PORTB,2         : CLK -->0
        RETURN                  : Düşen kenar tetikleme.

RESET;.....
        BSF     PORTB, 1        : RESET --> 1 aktif
        BCF     PORTB, 1        : RESET --> 0
        CLRWDT
        RETURN

SANIYE;.....

        CALL    GECIKME1        : Bir süre bekle.
        INCF    SAYI            : SAYI = SAYI+1
        MOVLW   D'10'           : W --> D'10'
        SUBWF   SAYI,W
        BTFSS   STATUS,2        : SAYI[0]= 10?
        GOTO    SANIYE_SON      : Hayır. Saniye_son'a git.
        CLRF    SAYI            : Evet. SAYI[0]=0
        INCF    SAYI+1          : SAYI[1]= SAYI[1]+1
        MOVLW   D'6'            : W-->6
        SUBWF   SAYI+1,W
        BTFSS   STATUS,2        : SAYI[1] = 6?
```

```
            GOTO    SANIYE_SON      : Hayır. Saniye_son'a git
            CLRF    SAYI+1          : Evet. SAYI[1]=0
SANIYE_SON
            RETURN

DAKIKA;……..

            CALL    GECIKME1        : Bir süre bekle
            INCF    SAYI+2          : SAYI[2] = SAYI[2]+1
            MOVLW   D'10'           : W --> D'10'
            SUBWF   SAYI+2,W
            BTFSS   STATUS,2        : SAYI[2]= 10?
            GOTO    DAKIKA_SON      : Hayır. Dakika_son'a git
            CLRF    SAYI+2          : Evet. SAYI[2]=0
            INCF    SAYI+3          : SAYI[3]= SAYI[3]+1
            MOVLW   D'6'            : W-->6
            SUBWF   SAYI+3,W
            BTFSS   STATUS,2        : SAYI[3] = 6?
            GOTO    DAKIKA_SON      : Hayır.Dakika_son'a git
            CLRF    SAYI+3          : Evet. SAYI[3]=0
DAKIKA_SON
            RETURN

SAAT;……..
            CALL    GECIKME1        : Bir süre bekle
            INCF    SAYI+4          : SAYI[4] = SAYI[4]+1
            MOVLW   D'2'            : W --> D'2'
            SUBWF   SAYI+5,W
            BTFSC   STATUS,2        : SAYI[5]= 2?
            GOTO    S2              : Evet. S2'ye git
            MOVLW   D'10'           : W --> D'10'
            SUBWF   SAYI+4,W
            BTFSS   STATUS,2        : SAYI[4]= 10?
            GOTO    SAAT_SON        : Hayır Saat_son'a git
            CLRF    SAYI+4          : Evet. SAYI[4]=0
            INCF    SAYI+5          : SAYI[5] = SAYI[5]+1
            GOTO    SAAT_SON
S2          MOVLW   D'4'            : W --> D'4'
            SUBWF   SAYI+4,W
            BTFSS   STATUS,2        : SAYI[4]= 4?
            GOTO    SAAT_SON        : Hayır Saat_son'a git.
            CLRF    SAYI+4          : Evet. SAYI[4]=0
            CLRF    SAYI+5          : Evet. SAYI[5]=0
SAAT_SON
            RETURN
```

```
GECIKME1;.....
          MOVLW     H'50'
          MOVWF     SAYAC1
NEXT2     MOVLW     H'FF'
          MOVWF     SAYAC2
NEXT1     DECFSZ    SAYAC2,F
          GOTO      NEXT1
          DECFSZ    SAYAC1,F
          GOTO      NEXT2
          RETURN

KESME;.....
          MOVWF     W_Y              : W_Y <-- W
          MOVF      STATUS,W
          MOVWF     STATUS_Y         :_STATUS <--- STATUS
          CALL      TIMER            : TIMER alt programını çağır.
          MOVF      STATUS_Y,W
          MOVWF     STATUS           : STATUS değerini geri yükle.
          MOVF      W_Y,W            : W değerini geri yükle.
          RETFIE

TIMER;.....
          BCF       INTCON,T0IF      : T0IF bayrağını sıfırla.
          INCF      SN               : SN= SN + 1
          MOVLW     D'125'
          SUBWF     SN,W             : 1sn ayarı
          BTFSS     STATUS,2         : Z= 1 mi?
          GOTO      TIMER_SON        : Hayır
          CLRF      SN               : Evet sn=0
          CALL      AZALT            : Displaydeki sayıyı azalt.
TIMER_SON
          RETURN

AZALT;.....
          MOVLW     D'0'             : W  → D'0'
          SUBWF     SAYI,W
          BTFSC     STATUS,2         : SAYI[0]= 0?

          GOTO      BIR              : Evet BIR etiketine git.
          DECF      SAYI             : SAYI[0]=SAYI[0]-1
          GOTO      AZALT_SON
BIR       MOVLW     D'0'             : W→D' 0'
          SUBWF     SAYI+1,W
          BTFSC     STATUS,2         : SAYI[1]= 0?
          GOTO      IKI              : Evet IKI etiketine git
          DECF      SAYI+1           : SAYI[1]=SAYI[1]-1
          MOVLW     D'9'             : W --> D'9'
```

```
        MOVWF   SAYI            : SAYI[0]=9
        GOTO    AZALT_SON

IKI     MOVLW   D'0'            : W→ D'0'
        SUBWF   SAYI+2,W
        BTFSC   STATUS,2        : SAYI[2]= 0?
        GOTO    UC
        DECF    SAYI+2          : SAYI[2]=SAYI[2]-1
        MOVLW   D'5'
        MOVWF   SAYI+1          : SAYI[1]=5
        MOVLW   D'9'            : SAYI[0]=9
        MOVWF   SAYI
        GOTO    AZALT_SON

UC      MOVLW   D'0'            : W→ D'0'
        SUBWF   SAYI+3,W
        BTFSC   STATUS,2        : SAYI[3]= 0?
        GOTO    DORT
        DECF    SAYI+3          : SAYI[3]=SAYI[3]-1
        MOVLW   D'9'
        MOVWF   SAYI+2          : SAYI[2]=9
        MOVLW   D'5'            : SAYI[1]=5
        MOVWF   SAYI+1          : SAYI[0]=9
        MOVLW   D'9'
        MOVWF   SAYI
        GOTO    AZALT_SON

DORT    MOVLW   D'0'            : W → D'0'
        SUBWF   SAYI+4,W
        BTFSC   STATUS,2        : SAYI[4]= 0?
        GOTO    BES
        DECF    SAYI+4          : SAYI[4]=SAYI[4]-1
        MOVLW   D'9'
        MOVWF   SAYI+2
        MOVWF   SAYI
        MOVLW   D'5'
        MOVWF   SAYI+1
        MOVWF   SAYI+3
        GOTO    AZALT_SON

BES     MOVLW   D'0'
        SUBWF   SAYI+5,W
        BTFSC   STATUS,2
        GOTO    ALTI
        DECF    SAYI+5
        MOVLW   D'9'
        MOVWF   SAYI+4
```

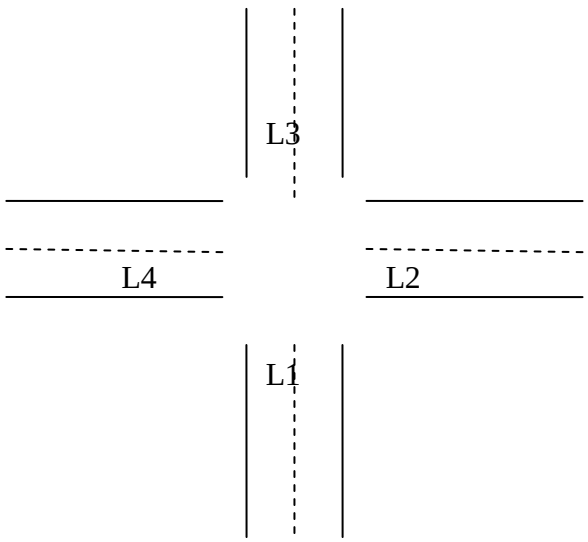
```
            MOVWF   SAYI+2
            MOVWF    SAYI
            MOVLW   D'5'
            MOVWF   SAYI+1
            MOVWF    SAYI+3
            GOTO    AZALT_SON
ALTI
            BSF      PORTB,3
            BCF      INTCON,T0IE
            GOTO     BASLA
AZALT_SON
            RETURN
END
```

2.4. Proje Uygulaması

Proje uygulamasında grup çalışması yapılacak ve her bir grup farklı bir proje konusu tespit edecektir. Aşağıda örnek projeler verilmiştir.

2.4.1. Proje 1

- Dört Kavşaklı Yolun Trafik Lamba Uygulaması



Üstteki şekilde görüldüğü gibi 4 ayrı lamba grubu bulunmaktadır. L1- L2 - L3 ve L4 gruplarında sarı, kırmızı ve yeşil lambalar bulunmaktadır. Lambaların yanık kalma süreleri ve hangi yolun önce başlayacağı programcıya bırakılmıştır. Sistem "BASLA" butonu ile başlayacaktır.

- Devrenin akış diyagramı çıkarılır.
- ASM programı yazılır.
- Devre için gerekli malzemeler ayarlanır.
- Devre şeması çizilir.
- Baskı devresi çıkarılır.
- PIC programlanır.
- Devre elemanlarının montajı yapılarak devre çalıştırılır.

2.4.2. Dijital Saat Uygulaması

Saat 7 parçalı gösterge ve LCD gösterge ile yapılacaktır. Saat ayarlanabilir olacak. Saat, dakika ve saniye ayarı için ayrı butonlar bulunacak. Ayar yapıldıktan sonra saatin çalışması "BAŞLA" butonu ile başlatılacaktır.

- Devrenin akış diyagramı çıkarılır.
- ASM programı yazılır.
- Devre için gerekli malzemeler ayarlanır.
- Devre şeması çizilir.
- Baskı devresi çıkarılır.
- PIC programlanır.
- Devre elemanlarının montajı yapılarak devre çalıştırılır.

2.4.3. Termometre Uygulaması

Sıcaklık LM35DZ sensörü ile ölçülecek, analog sinyal dijital sinyale dönüştürülecek ve LCD ekranında gösterilecektir. LCD'nin ilk satırında "SICAKLIK" yazdırılacak ve ikinci satırında ise ölçülen sıcaklığın değeri yazdırılacaktır.

- Devrenin akış diyagramı çıkarılır.
- ASM programı yazılır.
- Devre için gerekli malzemeler ayarlanır.
- Devre şeması çizilir.
- Baskı devresi çıkarılır.
- PIC programlanır.
- Devre elemanlarının montajı yapılarak devre çalıştırılır.

2.4.4. Bipolar Adım Motor Denetimi

Mikrodenetleyici ile bipolar adım motorunu döndürmektedir. Sistemde 2 buton kullanılmıştır. Butonlardan biri motorun hızını artırmakta diğeri ise motorun hızını azaltmaktadır.

- Devrenin akış diyagramı çıkarılır.

- ASM programı yazılır.
- Devre için gerekli malzemeler ayarlanıe.
- Devre şeması çizilir.
- Baskı devresi çıkarılır.
- PIC programlanır.
- Devre elemanlarının montajı yapılarak devre çalıştırılır.

2.4.5. Şifreli Kilit Uygulaması

Şifrenin yazımı için devrede 16 ve 12'lik tuş takımı kullanılacaktır. Şifre programcı tarafından programda belirlenecektir. Şifre yanlış girildiğinde tekrar şifre girilebilecektir. Şifre doğru girildiğinde kapıya bağlı olan elektronik kilit açılacaktır. Bu kilit bir transistör ve role ile kontrol edilebilir.

- Devrenin akış diyagramı çıkarılır.
- ASM programı yazılır.
- Devre için gerekli malzemeler ayarlanır.
- Devre şeması çizilir.
- Baskı devresi çıkarılır.
- PIC programlanır.
- Devre elemanlarının montajı yapılarak devre çalıştırılır.

UYGULAMA FAALİYETİ

Sistemin yazılımını programlayıp mikrodenetleyici ile çalıştırınız.

İşlem Basamakları	Öneriler
➢ Kurulacak sistem için ihtiyaçları (devre elemanlarını) tespit ediniz. ➢ İhtiyacınızı karşılayacak mikrodenetleyiciyi seçiniz. ➢ Sisteminin çalışması için gerekli programı yazınız. ➢ Programı mikrodenetleyiciye yükleyiniz. ➢ Çevre elemanları ile devreyi kurunuz.	➢ Mikrodenetleyici olarak PIC 16F84 kullanınız. ➢ Programı yazdıktan sonra MPLAB ile deneyiniz. ➢ Programı mikrodenetleyiciye yüklerken kullanılan PIC programlayıcıya uygun yazılım kullanınız. ➢ Devreyi Proteus programında çalıştırarak deneyiniz.
➢ Devre için gerekli giriş elemanları ve özelliklerini belirleyiniz. ➢ Devre için gerekli çıkış elemanları ve özelliklerini tespit ediniz. ➢ Devrenin baskı devre şemasını çıkartınız. ➢ Devre elemanlarının ve mikrodenetleyicinin montajını yapınız.	➢ Kullandığınız devre elemanlarının özelliklerini internetten araştırınız. ➢ Baskı devreyi, devre şemasını "Proteus" programından "Ares" programına aktrarak çıkarınız. ➢ Devrenin montajını yapmadan önce bredboard üzerinde kurarak deneyiniz.

KONTROL LİSTESİ

Bu faaliyet kapsamında aşağıda listelenen davranışlardan kazandığınız beceriler için **Evet,** kazanamadıklarınız için **Hayır** kutucuklarına (X) işareti koyarak öğrendiklerinizi kontrol ediniz.

Değerlendirme Ölçütleri		Evet	Hayır
1.	Kurulacak sistem için ihtiyaçları tespit ettiniz mi?		
2.	İhtiyacı karşılayacak mikrodenetleyiciyi seçtiniz mi?		
3.	Sistemin mikrodenetleyici programın yazdınız mı?		
4.	Programı mikrodenetleyiciye yüklediniz mi?		
5.	Çevre elemanları ile devreyi kurdunuz mu?		

DEĞERLENDİRME

Değerlendirme sonunda "Hayır" şeklindeki cevaplarınızı bir daha gözden geçiriniz. Kendinizi yeterli görmüyorsanız öğrenme faaliyetini tekrar ediniz. Bütün cevaplarınız "Evet" ise "Ölçme ve Değerlendirme"ye geçiniz.

ÖLÇME VE DEĞERLENDİRME

Aşağıdaki cümlelerde boş bırakılan yerlere getirilecek bilgilerin bulunduğu seçeneği işaretleyiniz.

1. LCD'ye veri yazmak için E girişine hangi kenar tetiklemeli sinyal uygulanmalıdır?
   A) Düşen kenar
   B) Yükselen kenar
   C) Pozitif seviye
   D) Negatif seviye

2. DECFSZ, CX komutunun görevi nedir?
   A) CX değişkenini bir artır, eğer sıfırsa bir alt komuta git.
   B) CX değişkenini bir artır.
   C) CX değişkenini bir azalt, eğer sıfırsa bir alt komuta git.
   D) CX değişkenini bir azalt.

3. MPSAM programının görevi nedir?
   A) HEX uzantılı dosyayı, ASM uzantılı dosyaya çevirir.
   B) ASM uzantılı dosyayı, HEX uzantılı dosyaya çevirir.
   C) HEX uzantılı dosyayı PIC'e yazdırır.
   D) ASM uzantılı dosyayı PIC'e yazdırır.

4. LCD'ye komut yazdırmak için RW, RS, E komutlarının değeri aşağıdakilerden hangisidir?

	RW	RS	E
A)	0	0	1→0
B)	1	1	1→0
C)	0	0	0→1
D)	1	0	1→0

5. PORTB'nin bacaklarını giriş – çıkış – giriş – çıkış – çıkış – giriş – giriş – çıkış olarak ayarlamak için aşağıdaki girişlerden hangisi uygulanmalıdır?

	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
A)	1	0	1	0	0	0	0	1
B)	1	1	1	1	0	0	0	1
C)	0	1	0	1	0	1	1	0
D)	1	0	1	0	0	1	1	0

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyete geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise "Modül Değerlendirme"ye geçiniz.

MODÜL DEĞERLENDİRME

Aşağıdaki cümlede boş bırakılan yere getirilecek bilgilerin bulunduğu seçeneği işaretleyiniz.

1. INTCON kaydedicisi ……uygulamalarında kullanılır. Boşluğa aşağıdakilerden hangisi gelmelidir?
A) Giriş
B) Çıkış
C) Kesme
D) Reset

2. Bit test etmek için kullanılan komut…….. dir. Boşluğa aşağıdakilerden hangisi gelmelidir?
A) BTFSS
B) IORLW
C) DECFSZ
D) SUBWF

Aşağıdaki soruları dikkatlice okuyunuz ve doğru seçeneği işaretleyiniz.

3. STATUS registerinde bulunan RP0 ve RP1 bitlerinin görevi aşağıdakilerden hangisidir?
A) Kesmenin geçerli olması için kullanılır.
B) İşlemlerde taşma meydana geldiğinde kullanılır.
C) İşlemlerin sonucu sıfır olduğunda kullanılır.
D) Bank değiştirmek için kullanılır.

4. PIC16F84'te kesme meydana geldiğinde program hangi adrese atlamaktadır?
A) H'00'
B) H'02'
C) H'04'
D) H'06'

5. Aşağıdaki seçeneklerden hangisi PIC'i programlamak için gerekli değildir?
A) MPSAM ASM derleyici
B) PIC programlayıcı
C) MPLAB PFE
D) PROTEUS programı

6. LCD'ye veri yazmak için E girişine hangi kenar tetiklemeli sinyal uygulanmalıdır?
A) Düşen kenar
B) Yükselen kenar
C) Pozitif seviye
D) Negatif seviye

7. DECFSZ, CX komutunun görevi nedir?

A) CX değişkenini bir artır, eğer sıfırsa bir alt komuta git.
B) CX değişkenini bir artır.
C) CX değişkenini bir azalt, eğer sıfırsa bir alt komuta git.
D) CX değişkenini bir azalt.

8. MPSAM programının görevi nedir?

A) HEX uzantılı dosyayı, ASM uzantılı dosyaya çevirir.
B) ASM uzantılı dosyayı, HEX uzantılı dosyaya çevirir.
C) HEX uzantılı dosyayı PIC'e yazdırır.
D) ASM uzantılı dosyayı PIC'e yazdırır.

9. LCD'ye komut yazdırmak için RW, RS, E komutlarının değeri aşağıdakilerden hangisidir?

	RW	RS	E
A)	0	0	1→0
B)	1	1	1→0
C)	0	0	0→1
D)	1	0	1→0

10. PORTB'nin bacaklarını giriş – çıkış – giriş – çıkış – çıkış – giriş – giriş – çıkış olarak ayarlamak için aşağıdaki girişlerden hangisi uygulanmalıdır?

	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
A)	1	0	1	0	0	0	0	1
B)	1	1	1	1	0	0	0	1
C)	0	1	0	1	0	1	1	0
D)	1	0	1	0	0	1	1	0

DEĞERLENDİRME

Cevaplarınızı cevap anahtarıyla karşılaştırınız. Yanlış cevap verdiğiniz ya da cevap verirken tereddüt ettiğiniz sorularla ilgili konuları faaliyetlere geri dönerek tekrarlayınız. Cevaplarınızın tümü doğru ise bir sonraki modüle geçmek için öğretmeninize başvurunuz.

CEVAP ANAHTARLARI

ÖĞRENME FAALİYETİ-1'İN CEVAP ANAHTARI

1	**C**
2	**D**
3	**C**
4	**D**
5	**A**

ÖĞRENME FAALİYETİ-2'NİN CEVAP ANAHTARI

1	**A**
2	**C**
3	**B**
4	**B**
5	**D**

MODÜL DEĞERLENDİRMENİN CEVAP ANAHTARI

1	**C**
2	**A**
3	**D**
4	**C**
5	**D**
6	**A**
7	**C**
8	**B**
9	**B**
10	**D**

KAYNAKÇA

- ALTINBAŞAK Orhan, Mikrodenetleyiciler ve PIC Programlama, İstanbul, 2000.
- Doğan İBRAHIM, Hamit I. MUSTAFA, **PIC Programlama ve İleri PIC Projeleri**, İstanbul, 2004.
- KARAKAŞ Hakan, **İleri PIC 16F84 Uygulamaları – 1**, İstanbul ,2002.
- TOPALOĞLU Nurettin, Salih GÖRGÜNOĞLU, **Mikroişlemciler ve Mikrodenetleyiciler**, Ankara 2003.