T.C.
MİLLÎ EĞİTİM BAKANLIĞI

MEGEP

(MESLEKİ EĞİTİM VE ÖĞRETİM SİSTEMİNİN
GÜÇLENDİRİLMESİ PROJESİ)

ENDÜSTRİYEL OTOMASYON TEKNOLOJİLERİ

MİKRODENETLEYİCİ UYGULAMALARI-1

ANKARA 2007

Milli Eğitim Bakanlığı tarafından geliştirilen modüller;

- Talim ve Terbiye Kurulu Başkanlığının 02.06.2006 tarih ve 269 sayılı Kararı ile onaylanan, Mesleki ve Teknik Eğitim Okul ve Kurumlarında kademeli olarak yaygınlaştırılan 42 alan ve 192 dala ait çerçeve öğretim programlarında amaçlanan mesleki yeterlikleri kazandırmaya yönelik geliştirilmiş öğretim materyalleridir (Ders Notlarıdır).
- Modüller, bireylere mesleki yeterlik kazandırmak ve bireysel öğrenmeye rehberlik etmek amacıyla öğrenme materyali olarak hazırlanmış, denenmek ve geliştirilmek üzere Mesleki ve Teknik Eğitim Okul ve Kurumlarında uygulanmaya başlanmıştır.
- Modüller teknolojik gelişmelere paralel olarak, amaçlanan yeterliği kazandırmak koşulu ile eğitim öğretim sırasında geliştirilebilir ve yapılması önerilen değişiklikler Bakanlıkta ilgili birime bildirilir.
- Örgün ve yaygın eğitim kurumları, işletmeler ve kendi kendine mesleki yeterlik kazanmak isteyen bireyler modüllere internet üzerinden ulaşabilirler.
- Basılmış modüller, eğitim kurumlarında öğrencilere ücretsiz olarak dağıtılır.
- Modüller hiçbir şekilde ticari amaçla kullanılamaz ve ücret karşılığında satılamaz.

İÇİNDEKİLER

AÇIKLAMALAR

KOD	**523EO0371**
ALAN	**Endüstriyel Otomasyon Teknolojileri**
DAL/MESLEK	**Endüstriyel Kontrol Teknisyenliği**
MODÜLÜN ADI	**Mikrodenetleyici Uygulamaları 1**
MODÜLÜN TANIMI	Bu modül ileri seviyede mikrodenetleyici uygulamalarını tanıtan ve öğrencilerin üst düzey mikrodenetleyici sistem tasarımlarını geliştirmelerine yönelik bilgi ve becerilerin verildiği öğrenim materyalidir.
SÜRE	40/32
ÖN KOŞUL	Mikrodenetleyici 6 modülünü almış olmak.
YETERLİK	Mikrodenetleyici ile ileri seviye uygulaması yapmak
MODÜLÜN AMACI	**Genel Amaç** Mikrodenetleyici ile ileri seviye uygulamalarını hatasız olarak yapabileceksiniz. **Amaçlar** **1.** Mikrodenetleyici ile LCD kontrolünü hatasız olarak yapabileceksiniz. **2.** Mikrodenetleyici ile servo motor kontrolünü hatasız olarak yapabileceksiniz. **3.** Mikrodenetleyici ile PWM motor kontrolünü hatasız olarak yapabileceksiniz.
EĞİTİM ÖĞRETİM ORTAMLARI VE DONANIMLARI	**Ortam:** Elektronik Laboratuvarı **Donanım:** Mikrodenetleyici, mikrodenetleyici programlama kartları, mikrodenetleyici uygulama devreleri, elektronik elemanlar, güç kaynağı, osilaskop, sinyal jeneratörü, RF Servo motor, DC motor, LCD, bread bord, el takımları
ÖLÇME VE DEĞERLENDİRME	Her faaliyetin sonunda ölçme soruları ile öğrenme düzeyinizi ölçeceksiniz. Araştırmalarla, grup çalışmaları ve bireysel çalışmalarla öğretmen rehberliğinde ölçme ve değerlendirmeyi gerçekleştirebileceksiniz.

GİRİŞ

Sevgili Öğrenci,

Mikrodenetleyici Uygulamaları-1 Modülü ile LCD kontrolü, RF Servo motor kontrolü ve DC motor kontrolü ile ilgili temel yeterlikleri kazanacaksınız. Kontrol sisteminde, daha önce öğrendiğiniz PIC16F84 mikrodenetleyicisinden çok daha fazla özellikleri bulunan PIC16F877'yi kullanacak ve kontrol programlarını da PIC-C dili ile gerçekleştireceksiniz. Böylece PIC ailesi içerisinde farklı bir mikrodenetleyiciyi ileri seviyede bir programlama dili ile programlamayı öğreneceksiniz. Bu çerçeve altında kontrol sistemlerinde bilgi paylaşımı işlemlerinde önemli bir yere sahip olan LCD göstergenin programlanmasını, enerjiyi harekete çeviren servo ve dc motorların ileri seviye kontrollerini de öğreneceksiniz. Tüm bu yapıların elektronik ve otomasyon dünyasında çok önemli yerleri bulunmaktadır. Endüstriyel Otomasyonun yer aldığı hemen hemen her yerde karşılaşabileceğiniz bu sistemleri öğrenerek ve kontrol uygulaması gerçekleştirerek çok değerli tecrübeler elde edecek ve üst düzey tasarımları gerçeleştirebileceksiniz.

Mikrodenetleyici Uygulamaları-1 Modülü Mikrodenetleyici Uygulamaları Dersi'nin yedinci modülüdür. PIC16F84 mikrodenetleyicisinin temel özelliklerini, temel PIC komutlarını ve temel uygulamaları daha önceki modüllerde öğrenmiştiniz. Bu modülde artık bilgilerinizi biraz daha ileri seviyeye taşıyacaksınız. Hazırlanan tüm öğrenme faaliyetleri birebir uygulamalı bilgileri içermektedir. Bu modül ile farklı yapıda tasarımlar gerçekleştirebilecek ve elektronik bilginizi geliştirebileceksiniz. Bu modülde verilen konuları öğrenirken internetten ve ilgili diğer elektronik kitaplardan da faydalanmanız size değişik bilgiler ve bakış açıları kazandıracaktır.

ÖĞRENME FAALİYETİ-1

AMAÇ

Mikrodenetleyici ile LCD kontrolünü hatasız olarak yapabileceksiniz.

ARAŞTIRMA

- PIC16F877 Mikrodenetleyici hakkında ön araştırma yapınız.
- LCD çeşitleri ve yapıları hakkında ön araştırma yapınız.

1. PIC16F877 İLE LCD KONTROLÜ

1.1. PIC16F877 Mikrodenetleyici

Daha önceki modüllerde uygulama çalışmalarınızı PIC16F84 mikrodenetleyici ile gerçekleştirmiştiniz. Bu modülde ise PIC ailesinin gelişmiş üyelerinden biri olan PIC16F877 ile uygulamalar yaptırılacaktır. Bu nedenle öncelikle PIC16F877'yi tanımak gereklidir. Bu kısımda PIC16F84 ile ortak olan özellikler üzerinde ayrıntılı durulmayacak, aradaki farkları ortaya koyan bir yaklaşım tercih edilecektir.

PIC16F877'de PIC16F84'de olduğu gibi Harvard mimarisi kullanılmıştır. Veri yolu 8 bit genişliğindedir. Aynı anda veri belleğine 8 bit genişliğindeki bu yolla erişilirken; program belleğine program yolu ya da adres yolu denilen 14 bit genişliğindeki diğer bir yolla erişilir. Bunun için PIC 16F877'de de komut kodları 14 bittir. 14 bitlik program belleğinin her bir adresi, bir komut koduna karşılık gelir.

1.1.1. PIC16F877 Mikrodenetleyicinin Özellikleri

- CPU'su azaltılmış komut seti (RISC) temeline dayanır. Programlamada kullanılan 35 komut vardır ve her biri 14 bit uzunluktadır.
- Dallanma komutları iki saykıllık sürede, diğerleri ise bir saykıllık sürede uygulanır.
- İşlem hızı 20 MHz'e kadar arttırılabilir.
- Veri yolu 8 bittir.
- 8 KB flash program belleği vardır ve yaklaşık 1 milyon kez programlanabilir.
- 368 Byte veri belleği (RAM) bulunmaktadır.
- 256 Byte EEPROM veri belleği vardır.
- Port A,B,C,D,E olmak üzere 5 adet giriş çıkış portu bulunur.

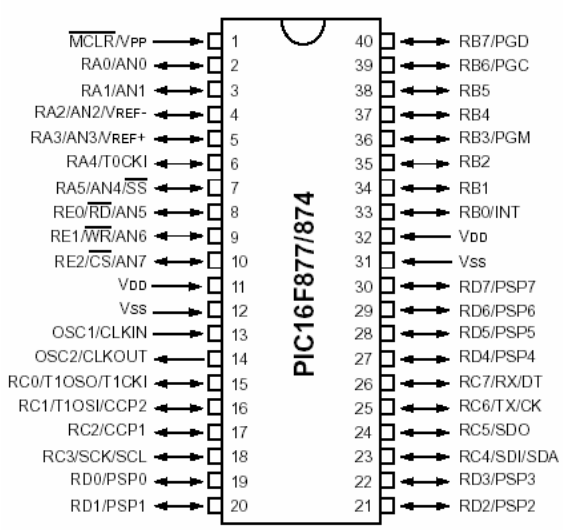
- 54 adet SFR olarak adlandırılan özel işlem yazmacı vardır ve bunlar statik RAM üzerindedir.
- Pin çıkışları PIC 16C73B/74B/76 ve 77 ile uyumludur.
- 14 kaynaktan kesme yapabilir.
- Power-on Reset (Enerji verildiğinde sistemi resetleme özelliği) bulunmaktadır.
- Power-up Timer (Power-up zamanlayıcı) bulunmaktadır.
- Osilatör Start-up Timer (Osilatör başlatma zamanlayıcısı) bulunur.
- Watch-dog Timer (Bekçi köpeği zamanlayıcısı) bulunur.
- Programla kod güvenliğinin sağlanabilmesi özelliği vardır.
- Devre içi Debugger (Hata ayıklamakta kullanılabilecek modül) bulunur.
- Düşük gerilimli ve sadece 2 pinle programlama özelliği vardır.
- Enerji tasarrufu sağlayan uyku modu bulunmaktadır.
- Seçimli osilatör özelliklerine sahiptir.
- 2.0 V – 5.0 V arasında değişen besleme gerilimine sahiptir.
- 25 mA'lik kaynak akımı standardına sahiptir.
- Geniş sıcaklık aralığında çalışabilme özelliği vardır.
- Düşük güçle çalışabilme özelliği vardır.
- TMR0 8 bitlik zamanlayıcısı 8 bit önbölücülü olarak kullanılabilir.
- TMR1 önbölücülü 16 bit zamanlayıcı içeririr. Uyuma modundayken kontrol edilebilir ve değeri arttırılabilir.
- TMR2 8 bitlik zamanlayıcı hem önbölücü hem de son bölücü sabitine sahiptir.
- İki adet Yakalama / Karşılaştırma / PWM modülüne sahiptir. Yakalama ve karşılaştırma 16 bit, PWM ise maksimum 10 bit çözünürlükle yapılabilir.
- 10 bit çok kanallı A/D çeviriciye sahiptir.
- Ana Senkron seri port (MSSP) modülü, SPI (Master mod) ve I^2C (Master Slave) modlarında kullanılabilir.
- Asenkron seri iletişim için USART seri iletişim ara birimine sahiptir.
- Paralel haberleşme için 8 bit genişlikte Paralel Slave Portu bulunur, bu port ile dış RD, WR, CS kontrollerine sahiptir.
- BOR Reset (Brown Out Reset) özelliğine sahiptir.

PIC 16F84 ile PIC 16F877 mikrodenetleyiciler arasındaki farklar Şekil 1.1'de verilmiştir.

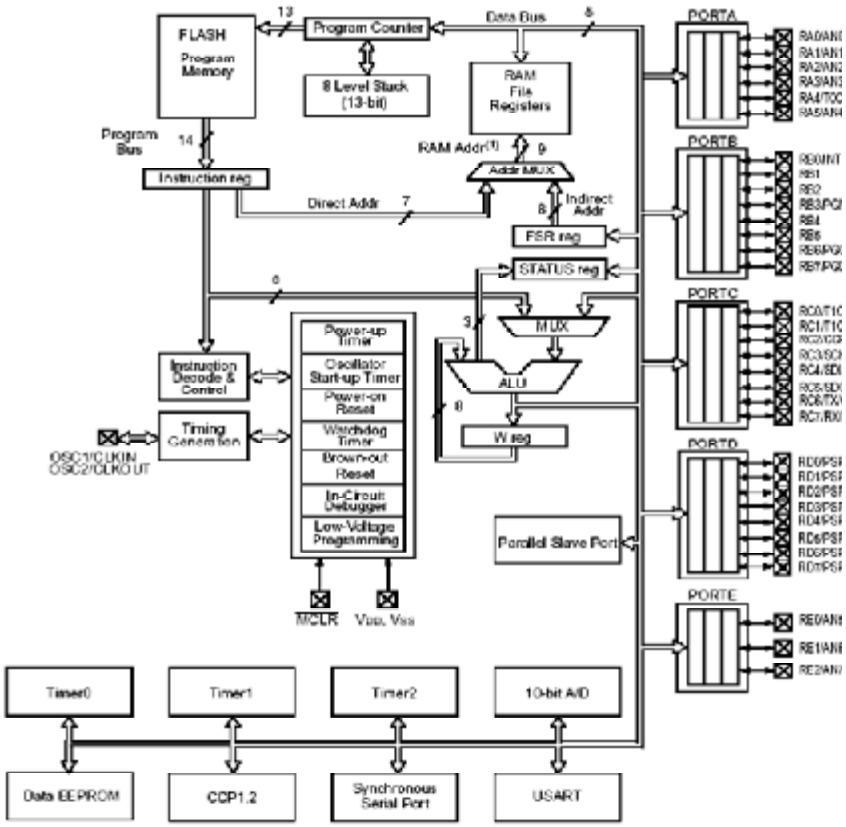
ÖZELLİKLER	PIC16F877	PIC16F84
Çalışma Hızı üst sınırı	20MHz	20MHz
Program Belleği	8 KB Flash ROM	1 KB Flash ROM
EEPROM Belleği	256 Byte	64 Byte
Kullanıcı RAM	368 Byte	68 Byte
Giriş/Çıkış Port Sayısı	33	13
Zamanlayıcı	Timer0, Timer1, Timer2	Timer0
A/D Çevirici	8 Kanal, 10 Bit	Yok
Capture/Compare/PWM	16 Bit Capture, 16 Bit Compare, 10 Bit PWM çözünürlük	Yok
Seri çevresel Arayüz	SPI (Master) ve 12C (Master - Slave) modunda SPI portu ve senkron seri port	Yok
Paralel Uydu Port (PSP)	8 Bit, harici, RD, WR ve CS kontrollü	Yok
USART/SCI	9 Bit adresli	Yok

Şekil 1.1: PIC16F84 ile PIC16F877 arasındaki farklar

1.1.2. PIC16F877 Mikrodenetleyicinin Fiziksel İç ve Dış Yapısı



Şekil 1.2: PIC16F877 Mikrodenetleyici Ayak Bağlantıları



Şekil 1.3: PIC16F877 Mikrodenetleyici iç yapısı

PİN ADI	PİN NO	PİN TİPİ	TAMPON TİPİ	AÇIKLAMALAR
OSC1/CLKIN	13	I	ST/CMOS(4)	Osilatör clock girişi (kristal veya harici kaynak)
OSC2/CLKOUT	14	O	-	Osilatör kristal çıkış ucu
$\overline{\text{MCLR}}$/Vpp	1	I/P	ST	Resetleme girişi / Programlama anında programlama gerilimi girişi (Mikrodenetleyicinin resetlenmesi için bu pin lojik 0 yapılmalıdır.)
RA0/AN0	2	I/O	TTL	Çift yönlü giriş/çıkış.Analog giriş 0.
RA1/AN1	3	I/O	TTL	Çift yönlü giriş/çıkış. Analog giriş 1.
RA2/AN2/V_{REF-}	4	I/O	TTL	Çift yönlü giriş/çıkış.Analog giriş 2. Negatif analog referans gerilimi.
RA3/AN3/V_{REF+}	5	I/O	TTL	Çift yönlü giriş/çıkış.Analog giriş 3. Pozitif analog referans gerilimi.
RA4/ T0CKI	6	I/O	ST	Çift yönlü giriş/çıkış. Bu pin istenirse TMR0 için bir clock girişi olabilir. Çıkış modunda açık drain tipindedir.
RA5/SS/ AN4	7	I/O	TTL	Çift yönlü giriş/çıkış. SSP için slave seçme pini. Analog giriş 4.
RB0/INT	33	I/O	TTL/ST(1)	Çift yönlü giriş/çıkış. Dış kesme girişi olarak seçilebilir.
RB1	34	I/O	TTL	Çift yönlü giriş/çıkış.
RB2	35	I/O	TTL	Çift yönlü giriş/çıkış.
RB3/PGM	36	I/O	TTL	Çift yönlü giriş/çıkış. Düşük gerilimli programlamada da kullanılabilir.
RB4	37	I/O	TTL	Çift yönlü giriş/çıkış. Portb kesme pini.
RB5	38	I/O	TTL	Çift yönlü giriş/çıkış. Portb kesme pini.
RB6/PGC	39	I/O	TTL/ST(2)	Çift yönlü giriş/çıkış. Portb kesme pini. Devre içi hata ayıklama pini. Seri programlamada saat darbesi girişi.
RB7/PGD	40	I/O	TTL/ST(2)	Çift yönlü giriş/çıkış. Portb kesme pini. Devre içi hata ayıklama pini. Seri programlamada veri girişi.

Şekil 1.4a: PIC16F877 Mikrodenetleyici pinlerinin fonksiyonları

PİN ADI	PİN NO	PİN TİPİ	TAMPON TİPİ	AÇIKLAMALAR
RC0/T1OS0/T1CK 1	15	I/O	ST	Çift yönlü giriş/çıkış. Timer1 osc. çıkışı veya Timer1 saat derbe girişi.
RC1/T1OS1/CCP2	16	I/O	ST	Çift yönlü giriş/çıkış. Timer1 osc. girişi. Capture2 girişi. Compare2 çıkışı. PWM2 çıkışı.
RC2/CCP1	17	I/O	ST	Çift yönlü giriş/çıkış. Capture1 girişi. Compare1 çıkışı. PWM1 çıkışı.
RC3/SCK/SCL	18	I/O	ST	Çift yönlü giriş/çıkış. SPI ve I^2 C modları için senkron seri saat darbesi giriş/çıkışı.
RC4/SD1/SDA	23	I/O	ST	Çift yönlü giriş/çıkış. SPI modda SPI veri girişi veya I^2C modda veri girişi ya da çıkışı.
RC5/SDO	24	I/O	ST	Çift yönlü giriş/çıkış. SPI modda SPI veri çıkışı.
RC6/TX/CK	25	I/O	ST	Çift yönlü giriş/çıkış. USART asenkron gönderme veya senkron saat darbesi.
RC7/RX/DT	26	I/O	ST	Çift yönlü giriş/çıkış. USART asenkron alma veya senkron veri.
RD0/PSP0	19	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. Paralel Slave Port 0 (PSP0)
RD1/PSP1	20	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP1.
RD2/PSP2	21	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP2.
RD3/PSP3	22	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP3.
RD4/PSP4	27	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP4.
RD5/PSP5	28	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP5.
RD6/PSP6	29	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP6.
RD7/PSP7	30	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. PSP7.
RE0/RD/AN5	8	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. Analog giriş 5. Paralel slave port için okuma kontrol ucu.
RE1/WR/AN6	9	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. Analog giriş 6. Paralel slave port için yazma kontrol ucu.
RE2/CS/AN7	10	I/O	ST/TTL(3)	Çift yönlü giriş/çıkış. Analog giriş 7. Paralel slave port için seçme kontrol ucu.

Şekil 1.4b: PIC16F877 Mikrodenetleyici pinlerinin fonksiyonları

PİN ADI	PİN NO	PİN TİPİ	TAMPON TİPİ	AÇIKLAMALAR
V_{SS}	12, 31	P		Ground (toprak) ucu
V_{DD}	11, 32	P		Pozitif kaynak ucu

I: Input (Giriş) O: Output (Çıkış) I/O: Input/Output (Giriş/Çıkış) P: Power (Güç)
TTL: TTL Giriş ST: Schimitt Trigger -: Kullanılmıyor
Not: 1. Bu tampon portb dış kesme olarak düzenlendiğinde Schimitt Trigger girişidir.
2. Bu tampon seri programlama modunda Schimitt Trigger girişidir.
3. Bu tampon genel amaçlı giriş-çıkış olarak kullanıldığında Schimitt Trigger girişi ve PSP modunda TTL girişidir.
4. Bu tampon RC osilatör modunda Schimitt Trigger girişidir. Diğer modlarda CMOS girişidir.

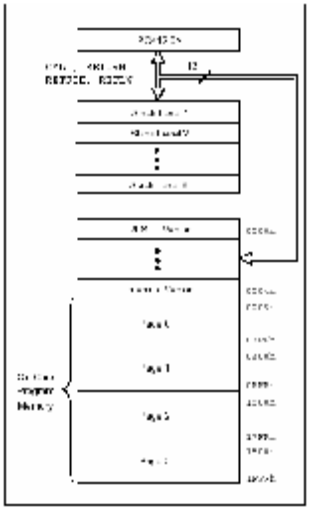
Şekil 1.4c: PIC16F877 Mikrodenetleyici pinlerinin fonksiyonları

1.1.3. PIC16F877 Mikrodenetleyici Hafıza Organizasyonu

PIC16F877 Mikrodenetleyici içerisinde Harvard mimarisine göre tasarlanmış üç adet hafıza bloğu vardır.

1.1.3.1. Program Hafıza Organizasyonu

PIC16F877 içerisinde Şekil 1.3'te görüldüğü gibi 14 bitlik program yoluna bağlı 8KB'lık flash eprom program belleği (8Kx14) bulunmaktadır ve 13 bitlik program sayacı ile adreslenmektedir. Reset vektörü 0000h ve kesme vektörü de 0004h'dir. Şekil 1.5'te program hafıza haritası görülmektedir. Program kodları 0000h ile 1FFFh arasına yazılır. PIC16F877 ilk çalışmaya başladığı anda ya da herhangibir zamanda restlendiği anda program hafızada 0000h adresine gidilir. Kesme sinyali algılandığında ise program sayacı 0004h adresini üretir ve 0004h adresi üzerinden bir kesme alt programı çalışır.



Şekil 1.5: Program hafıza haritası

1.1.3.2. Veri Hafıza Organizasyonu

Veri hafızayı RAM bellek ve Flash Eeprom bellek olmak üzere iki kısımda inceleyebiliriz.

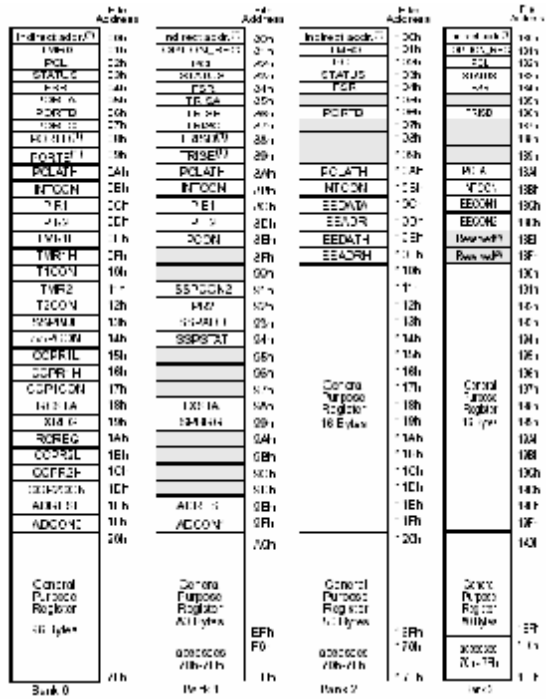
a. Flash Eeprom Bellek

256 Byte kapasiteli flash eeprom veri belleği 8 bitlik veri yoluna bağlıdır. Bu bellek kalıcı olarak veri kaydetmek amaçlı olarak kullanılır. Elektrik enerjisi kesildiğinde bile bu bilgiler bellekte saklanabilir ve daha sonra değiştirilebilir. Normal çalışma sırasında veri eeprom belleğe veri yazılabilir ya da eepromdan veri okunabilir. Bu bellek, diğer veri belleği gibi doğrudan adreslenemez. EEPROM veri belleğine dolaylı erişilir.

Eeprom bellek kullanımı ile ilgili altı adet özel fonksiyon yazmacı (SFR) bulunmaktadır. Bunlar EEDATA, EEDATH, EEADR, EEADRH, EECON1 ve EECON2 yazmaçlarıdır. Bunlarla ilgili bilgi ilerleyen kısımlarda verilecektir.

b. Ram Bellek

368 Byte ram bellek üzerinde genel ve özel amaçlı yazmaçların yer aldığı dört adet saklayıcı kümesi(bank) yer almaktadır. Status yazmacının 5. ve 6. bitleri ile küme seçimi yapılmaktadır. Her bank 128 baytdır ve düşük kelime adreslerinde özel fonksiyon yazmaçları yer almakta iken yüksek adresli değerlerde genel amaçlı yazçmaçlar yer almaktadır. Özel fonksiyon yazmaçlarının bazıları daha hızlı erişim için birden fazla kümede yer almaktadır. Şekil 1.6'da PIC16F877 Mikrodenetleyicisinin RAM haritası görülmektedir.



Şekil 1.6: PIC16F877 RAM Haritası

Açıklamalar:

* Fiziksel bir yazmaç değildir.
1. PIC16F876'da kullanılmayan bir yazmaçtır.
2. Bu yazmaçlar kullanılamaz. Lojik-0 değerinde kalmalıdır.

Şekil 1.6'da görülen GPR bölgesine doğrudan ya da dosya seçim yazmacı (FSR) üzerinden dolaylı olarak ulaşılabilir. Özel fonksiyon yazmaçları mikroişlemci ve çevresel modüller tarafından özel işlemleri kontrol etmek için kullanılır. Bu modülde gerektiğinde uygulamalarda kullanılacak özel fonksiyon saklayıcıları ilgili bilgiler uygulama çalışmalarında verilecektir.

Yazmaçların İşlevleri:

1. PORTA PORTE

Portlar mikrodenetleyicinin dış dünyadan bilgi alması ve kendi dışındaki devrelere veri aktarabilmesi amacıyla kullanılır. PIC16F877'nin beş portu vardır. A portu 6 bit genişliğindedir. B, C, D portları 8 bit, E portu ise 3 bit genişliğindedir.

2. TRISA TRISE

Portların yönünü belirleyen yazmaçlardır. Eğer portların herhangi bir pininden mikrodenetleyici dışına veri gönderilecekse, önce ilgili portun yön yazmacı aynı numaralı biti lojik-0 yapılır. Eğer herhangibir pinden mikrodenetleyiciye veri girilecekse, yine önceden, o portun yön yazmacının aynı numaralı biti lojik-1 yapılır.

3. TMR0, TMR1 ve TMR2

Mikrodenetleyici içinde bulunan zamanlayıcı ve sayaç olarak çalıştırılan bölümü denetleyen yazmaçlardır. TMR1 16 bitlik bir zamanlayıcıdır. TMR1L ve TMR1H olmak üzere iki adet 8 bitlik saklayıcıdan oluşur. TMR0 ve TMR2 8 bitliktir. TMR2 PWM sinyalinini zamanlamasında da kullanılır.

4. T1CON, T2CON ve PR2

Timer1 ve Timer2 zamanlayıcılarını kontrol etmek için kullanılırlar. PR2 Timer2 zamanlayıcısının son sayma değerini diğer bir deyişle 00h'e döndüğü değeri saklar.

5. EEDATA ve EEDATH, EEADR ve EEADRH, EECON1 ve EECON2

Mikrodenetleyici içindeki EEPROM veri belleğine ulaşmakta kullanılır. EEADR yazmacında adres numarası bulunan veri, EEPROM veri belleğinden okunarak EEDATA yazmacına getirilir. EEADR yazmaçı eprom bellekte erişilmek istenen adresi tutar. EEDATA ise yazılmak ya da okunmak istenen veriyi tutar. EEADRH 13 bitlik adresin ve EEDATH 14 bitlik verinin yüksek değerli bitlerini tutmak için kullanılır. EECON1 yazmacı erişimi başlatmak ve ayarlamak için kullanılan bir kontrol yazmacıdır. EECON2 katalog bilgisinde fiziksel olarak kullanılmayan bir yazmaç olarak geçmektedir. Donanımsal olarak yazma işlemi sırasında hatalı yazım işlemlerinden korunma amaçlı olarak kullanılmaktadır.

6. GPR

Genel amaçlı yazmaçların adresleri yukarıdaki çizelgede verilmişti. Programcı buradaki adresleri istediği gibi, kendi değişkenleri için kullanabilir. Bu adreslere isterse programın içinde özel adlar verebilir.

7. Durum Yazmacı

Durum(status) yazmacı, aritmetik ve mantık biriminin(ALU), aritmetik işlem sonucundaki durumunu, merkezi işlem biriminin (CPU) test durumlarını ve veri belleğine ait küme(bank) seçme bitlerini tutar. Herhangi bir yazmaç gibi içeriği okunabilir ve değiştirilebilir. Ancak 3. ve 4. bitleri sadece okunabilir, değiştirilemez.

Eğer, bu yazmacın içeriği CLRF STATUS komutuyla, silinmek istenirse sadece üst üç bit lojik-0 olur. Bu komut sonunda STATUS'un içeriği 000uu1uu değerini alır. Burada u:değişmez (Unchangable) anlamındadır.

7	6	5	4	3	2	1	0
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
R/W	R/W	R	R	R/W	R/W	R/W	R/W

Şekil 1.7: Durum yazmacı

Bit 7: IRP: Yazmaç bank seçme biti (dolaylı adreslemede kullanılır)
0= Bank 0,1 (00h-FFh)
1= Bank 2,3 (100h-1FFh)

Bit 6-5: RP1:RP0: Yazmaç bank seçme biti (doğrudan adreslemede kullanılır)
00= Bank 0
01= Bank1
10= Bank 2
11= Bank 3
Her bir bank 128 byte'tır.

Bit 4: $\overline{TO}$: Time-out (süre aşımı biti)
1= CLRWDT komutuyla veya SLEEP'den güç verme durumuna geçirildiğinde 1 olur.
0= WDT time-out (süre aşımı) işlemi gerçekleşmiş ise 0 olur.

Bit 3: $\overline{PD}$: Power-down (Güç kesme) biti
1= CLRWDT komutu ile veya güç verme durumunda 1 olur.
0= SLEEP komutu çalıştırılınca 0 olur.

Bit 2: Z: Zero (Sonuç sıfır) biti
1= Aritmetik veya lojik işlem sonucu 0 ise bu bit 1 olur.
0= Aritmetik veya lojik işlem sonucu 0 değil ise bu bit 0 olur.

Bit 1: DC: Dijit elde (Digit Carry/Borrow) biti. (ADDWF, ADDLW komutları için.)
1= 8 Bitin düşük öncelikli dörtlüsü, taşarsa bu bit lojik 1 olur.
0= 8 Bitin düşük öncelikli dörtlüsü taşmazsa, bu bit 0 olur.

Bit 0: C: Carry/Borrow biti (ADDWF ve ADDLW komutları için)
1= 7.bitten sonra taşma olursa bu bit 1 olur.
0= 7.bitten sonra taşma olmazsa bu bit 0 olur.

8. Option Register

Option Register, okunabilir ve yazılabilir bir yazmaçtır. Kapsamında TMR0 / WDT zamanlayıcılarının konfigürasyon bitleri, dış kesme (interrupt) denetim bitleri, TMR0

zamanlayıcısı kesme denetim bitleri ve PORTB için yükseğe çekme (pull-up) dirençlerinin kullanılmasını sağlayan bit bulunur.

9. INTCON

INTCON okunabilir ve yazılabilir bir yazmaçtır. Kapsamında TMR0 / WDT yazmacı taşma uyarı bitleri, RB port değişim ve dış kesme (RB0/INT pin interrupt) denetim bitleri, TMR0 kesme denetim bitleri bulunur.

10. PIE1

PIE1 çevresel kesmelerle ilgili bitleri olan bir yazmaçtır. Bir çevresel kesmenin olabilmesi için, PIE1 (INTCON<6>) biti de set edilmelidir.

11. PIR1

PIR1 çevresel kesmelerle ilgili uyarı bitlerini taşıyan yazmaçtır.

12. PIE2

PIE2 yazmacı CCP2 (Capture/Compare/PWM2) çevresel biriminin kesme bitlerini, SSP (Senkron Seri Port) veri yolu çarpışma bitini ve EEPROM yazma kesmesi bitini taşır.

13. PIR2

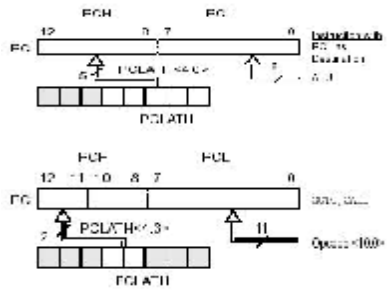
PIR2 çevresel kesmelerle ilgili diğer uyarı bitlerini taşıyan yazmaçtır.

14. PCON

Güç kontrol yazmacı olan PCON, yazılımda ve reset durumlarında kullanılır. Reset durumları; devrenin dışardan MCLR ile, gerilim ya da akımın aşırı düşme ve yükselmesi Brown-Out, Watch Dog Timer ve son olarak Power on reset durumlarında kullanılabilir. BOR biti, Power on reset'te bilinemez. Bir sonraki BOR durumunun öğrenilebilmesi için reset sonrasında lojik-1 yapılmalıdır.

15. PCL ve PCLATH

Program sayacı (PC) olarak adlandırılan adresleme yazmacı 13 bitliktir. Bunun düşük değerlikli byte'ı PCL yazmacından gelir. Üstteki bitler ise PC<12:8> arasındaki 5 bittir, bunlar PCLATH yazmacından alınır. PCL okunabilir ve yazılabilir bir yazmaçtır. Ancak üst bitleri (PCH) doğrudan okunamaz. Dolaylı olarak PCLath yoluyla yazılabilir veya okunabilir.



Şekil 1.8: PC ve PCLATH Yazmaçları

Call komutu, yığının her zaman en tepesine, PCL yazmacının içindeki adres değerini yazar. Return, Retfie ve Retlw komutları ise yığının en tepesindeki elemanın içeriğini PCL'ye aktarır. Saklayıcı küme numaralarının PCLath yazmacından PC'ye aktarılabildiği program yazarken de unutulmamalıdır. PCLATH yazmacının içeriği, alt programa girildikten sonra sabit kalır, bir return ya da retfie benzeri komut gelse de değişmez. Programcı, call veya goto komutlarından önce, PCLATH yazmacını güncellemelidir. PCH daima PCLATH yazmacı yoluyla güncellendiğinden (tersi yapılamaz) alt program veya gidilen kesimin hangi kümede olduğu, aşağıdaki örneğe benzer bir yolla belirtilmelidir.

```
            ORG 0x500
            BCF PCLATH, 4   ;PCLATH yazmacının 4.biti temizlendi
            BSF PCLATH, 3   ;PCLATH'ın 3.biti set edildi, 1.bank geçildi
                            ;(800h-FFFh adres aralığı)
            CALL SUB1_P1    ;1.banktaki alt program çağrıldı
             .
            ORG 0x900       ;Bank 1 (800h-FFFh)
SUB1_P1
             .
                            ; Altprogram 800h ile FFFh aralığına yerleştirildi.
            RETURN          ;Return'den sonra 0.sayfaya (000h-7FFh) dönülecek
```

16. YIĞIN Bölgesi

Yığın bölgesi 8 yazmaçtan oluşur. Her yazmaç 13 bitliktir ve donanımın bir parçasıdır. Veri veya program alanlarında yer almaz. Yığın göstergesi yazılabilir ve okunabilir değildir. Yığın işlemi komutları POP ve PUSH'tur. Her PUSH işleminde yığının en tepesindeki adrese, PC'ın içeriği yüklenir. Her POP işleminde yığının en tepesindeki adres PC'ın içine geri yüklenir.

Yığın, LIFO (Last In First Out) son giren ilk çıkar tekniğiyle çalışır.

17. INDF ve FSR

INDF fiziksel bir yazmaç değildir. Mikrokontrolördeki RAM adresini tutar. INDF'e yazılan her veri, adresi FSR yazmacında bulunan RAM'a yazılır. INDF'ten okunan veriler de adresi FSR'de bulunan RAM'den okunmuştur.

18. SSPBUF, SSPCON, SSPCON2, SSPSTAT ve SSPADD

Senkron seri port alma ve verme sırasında veri SSPBUF yazmacına yazılır. SSPCON ve SSPCON2 ise senkron seri port haberleşmesini kontrol eden yazmaçlardır. Senkron seri portun durum bitleri SSPSTAT yazmacında kayıtlıdır. SSPADD senkron seri port adres yazmacıdır.

Dolaylı adreslemede INDF ile birlikte kullanılır. Mikrodenetleyicinin içindeki RAM adresinde yapılacak işlemlerde, RAM adresini tutar. Bu durumda INDF'ye yazılacak her veri, aslında adresi FSR'de bulunan RAM'e yazılmıştır.

19. CCPR1L, CCPR1H, CCP1CON, CCPR2L, CCPR2H ve CCP2CON

Yakalama-Karşılaştırma-PWM işlemleri ile ilgili yazmaçlardır. Bu işlemler için PIC16F877'de iki ayrı grup bulunmaktadır. CCP1CON ve CCP2CON yazmaçları kontrol bitlerinin bulunduğu yazmaçlardır. CCPR1L ve CCPR2L ilk 8 biti saklarlar. CCPR1H ve CCPR2H yüksek değerli 8 biti saklar.

20. TXSTA, RCSTA, SPBRG, TXREG ve RCREG

Bu yazmaçlar üniversal asenron seri haberleşmede kullanılır. TXSTA seri bilgi vermede durum ve kontrol bitlerinin yer aldığı yazmaçtır. RCSTA Seri bilgi almada durum ve kontrol bitlerinin yer aldığı yazmaçtır. SPRG ise baud hızını belirleyen yazmaçtır. TXREG USART bilgi vermede veri yazmacı olarak kullanılır. RCREG ise USART bilgi almada veri yazmacı olarak kullanılır.

21. ADRESH, ADRESL, ADCON0, ADCON1

ADRESH A/D çevrim sonucunun yüksek değerli 8 bitini, ADRESL ise düşük değerli 8 bitini saklar. ADCON0 ve ADCON1 ise A/D çevrimi kontrol eden saklayıcılardır.

1.2. LCD Yapısı Ve Programlanması

1.2.1. LCD Yapısı

Bu kısımda paralel girişli LCD'nin yapısı ve programlanması verilecektir.

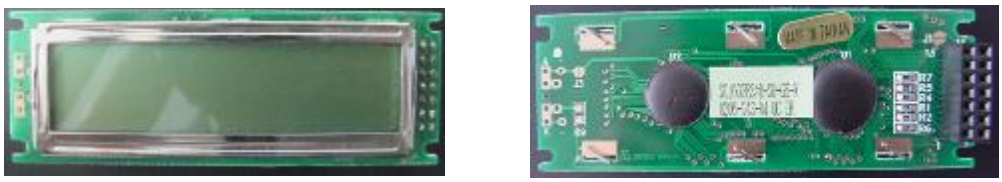
LCD (Liquid Crystal Display)'ler bilginin uygulanış yöntemine göre paralel ve seri girişli olmak üzere iki kısımda incelenebilmektedir. LCD'nin kullanım amacı elektronik sistemlerde bilgilendirme için çeşitli karakterleri ve karakter topluluklarını göstermektir.

LCD içerisindeki hazır karakterler kullanılarak veya özel karakterler hazırlanarak üzerinde anlamlı görüntüler oluşturulabilir.

LCD dijital bilgi üreten herhangi bir sistem ile kontrol edilebilir. Yapılması gereken tek şey uygun komutları katalogda gösterilen zamanlama dilimleri ile göndermektir. LCD bilgisayarla veya mikrodenetleyici ile kontrol edilebilir.

LCD'nin yapısı iki kısımdan meydana gelir. Bunlar yazı ya da grafikleri gösteren sıvı kristal display ve bu displayde karakterleri meydana getiren sürücü işlemcidir. LCD incelendiğinde bir yüzünde LCD diğer yüzünde de sürücü işlemci görülür. İşlemci entegre görünümünde olabileceği gibi siyah epoksi damla şeklinde de olabilir. Şekil 1.9'da HD44780 paralel girişli LCD görülmektedir. HD44780 LCD devresinde yer alan epoksi damla olarak yerleştirilmiş sürücü entegresidir. Bu kısımda piyasada bulunması kolay ve kullanım alanı geniş olduğundan dolayı bu LCD ile ilgili bilgiler verilip uygulama yapılacaktır. HD44780 işlemci LCD'ler için standart olmuş bir işlemcidir. LCD'nin arkasına bakılıp işlemci seri numarası okunabilir ve işlemcinin yapısı kataloglardan öğrenilebilir. Günümüzde birçok firmanın ürettiği LCD, HD44780 ile uyumludur.

LCD'ler satır ve sütun sayısı ile tanımlanır. Satır uzunluklarına (sütun sayılarına) göre 8-16-20-24-32-40 karakterli olarak yapılır. Satır sayısına göre ise 1, 2 ya da daha çok satırlı olabilir. 2x8 denince 2 satırlı 8 sütunlu LCD akla gelmelidir. 2x8 ve 1x16 LCD'de sadece bir işlemci kullanılırken, 2x16 ve 2x20 LCD'lerde işlemcinin yanı sıra bir de hafıza entegresi bulunur. Boyut arttıkça hafıza kapasitesi de artar. Tüm entegreler siyah epoksi damla şeklinde veya entegre şeklinde PCB üzerine yerleştirilir.



(a) Ön görünüş **(b) Arka görünüş**

Şekil 1.9: HD44780 LCD Display

LCD'lerde DDRAM ve CGRAM olmak üzere iki adet RAM hafıza hücresi yer alır. CGRAM karakter üretici RAM bellektir ve 64 Baytlık bir hafızadır. Özel karakterler üretmek için kullanılır. DDRAM ise display veri RAM belleğidir ve 80 Baytlık bir hafızadır. LCD'nin satır ve sütunlarına karakter yazmak için kullanılır. DDRAM içeriği LCD ekranında görülür, fakat görünen veri ancak LCD'nin sütun sayısı ile sınırlıdır. Örneğin 2x16'lık bir LCD'de ancak 16 karakter ekranda görülebilir. Diğer kayıtlı veriler DDRAM'da saklanmaya devam eder. Şekil 1.11'de DDRAM adreslerinin değişimi görülmektedir. LCD'nin içinde bulunan diğer hafıza ise CGROM'dur. Karakter üretici ROM bellek olarak tanımlanır ve 192 Baytlık bir hafızadır. CGROM içerisinde ASCII karakter tablosunda yer alan standart karakterler kayıtlıdır. Toplam karakter sayısı 192'dir. Bunlardan 64 tanesi genel olarak kullanılan ascii karakterleri, 64 tanesi Japon karakterleri ve 32 tanesi de özel yunan alfabesi karakterleridir. Bu karakterlerden genel olarak kullanılan 64 tane standart karakter Şekil 1.10'daki ASCII karakter tablosunda görülmektedir.

KOLON BİTLERİ

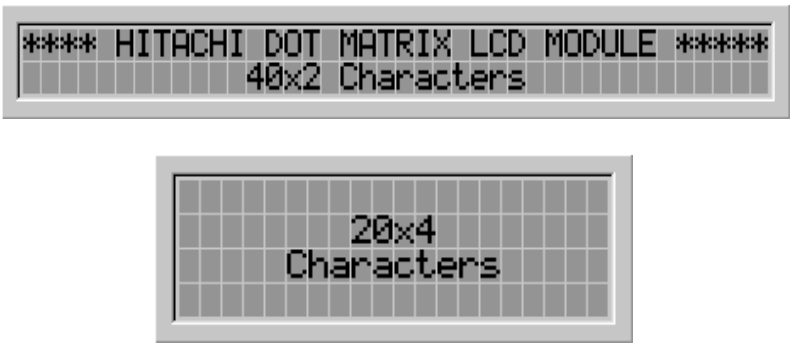
SATIR BİTLERİ

HEX	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Kontrol / Fonksiyon							
HEX								**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
								0	0	0	0	1	1	1	1
								0	0	1	1	0	0	1	1
								0	1	0	1	0	1	0	1
0				0	0	0	0	NUL	DLE	SP	0	@	P		p
1				0	0	0	1	SOH	DC1	!	1	A	Q	a	q
2				0	0	1	0	STX	DC2	“	2	B	R	b	r
3				0	0	1	1	ETX	DC3	#	3	C	S	c	s
4				0	1	0	0	EOT	DC4	$	4	D	T	d	t
5				0	1	0	1	ENQ	NAK	%	5	E	U	e	u
6				0	1	1	0	ACK	SYN	&	6	F	V	f	v
7				0	1	1	1	BEL	ETB	‘	7	G	W	g	w
8				1	0	0	0	BS	CAN	(	8	H	X	h	x
9				1	0	0	1	HT	EM	)	9	I	Y	i	y
A				1	0	1	0	LF	SUB	*	:	J	Z	j	z
B				1	0	1	1	VT	ESC	+	;	K	[	k	{
C				1	1	0	0	FF	FS	,	<	L	\	l	¦
D				1	1	0	1	CR	GS	-	=	M	]	m	}
E				1	1	1	0	SO	RS	.	>	N	^	n	~
F				1	1	1	1	SI	US	/	?	O	_	o	DEL

Şekil 1.10: ASCII Karakter tablosu

1 satırlı LCD DDRAM		
Display	**Uygulama**	
	Karakter durumları	**DDRAM adres**
1*8	00..07	0x00..0x07
1*16	00..15	0x00..0x0F
1*20	00..19	0x00..0x13
1*24	00..23	0x00..0x17
1*32	00..31	0x00..0x1F
1*40	00..39	0x00..0x27
2 satırlı LCD DDRAM		
Display	**Uygulama**	
	Karakter durumları	**DDRAM adres**
2*16	**00..15**	**0x00..0x0F + 0x40..0x4F (*Kullandığımız LCD*)**
2*20	00..19	0x00..0x13 + 0x40..0x53
2*24	00..23	0x00..0x17 + 0x40..0x57
2*32	00..31	0x00..0x1F + 0x40..0x5F
2*40	00..39	0x00..0x27 + 0x40..0x67
4 satırlı LCD DDRAM		
Display	**Uygulama**	
	Karakter durumları	**DDRAM adres**
4*16	00..15	0x00..0x0F + 0x40..0x4F + 0x14..0x23 + 0x54..0x63
4*20	00..19	0x00..0x13 + 0x40..0x53 + 0x14..0x27 + 0x54..0x67
4*40	(00..39) (00..39)	(0x00..0x27 + 0x40..0x67) (0x00..0x27 + 0x40..0x67)

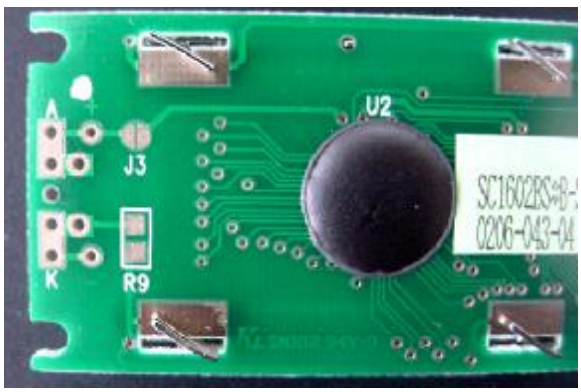
Şekil 1.11: LCD Displayler ve DDRAM Adresleri



Şekil 1.12: 2 Satırlı ve 4 Satırlı LCD'ler

Şekil 1.11'de çeşitli LCD'lerdeki DDRAM adreslerinin satır sayısına göre değişimi görülmektedir. Bu modülde baz alınan LCD'nin DDRAM adresi 1. satır için 0x00..0x0F ve 2. satır için 0x40..0x4F'dir.

LCD'nin dış dünya ile haberleşmesi için pinleri bulunmaktadır. HD44780 uyumlu LCD'de 14 pin bulunmaktadır. Bu pinler 2'şerli 7 sıra halinde olabileceği gibi 14'lü tek sıra olabilir. Bu pinlerin fonksiyonları Şekil 1.14'teki tabloda görülmektedir. LCD 4x40 ise ilave bir pini daha vardır. Ayrıca bazı LCD'lerde ışıklandırma amacıyla 15. ve 16. pinlere de rastlanabilir. 15. ve 16. pinler yoksa LCD devresinde aydınlatma uçlarından da bağlantı yapılabilir. Bu ilave pinlerin karakter gösterilmesinde bir etkisi yoktur. HD44780 LCD ışıklandırma bağlantıları Şekil 1.13'te görülmektedir.



Şekil 1.13: HD44780 LCD displayde ekran ışığı bağlantı uçları

No	Fonksiyonu
1	Vdd (5V gerilim ucu)
2	Vss (Şase bağlantı ucu)
3	Vee (Kontrast gerilimi ucu)
4	RS (Komut/Veri seçim ucu)
5	R/W (Yazma-Okuma ucu)
6	E (Uygulama izin ucu)
7	D0 (0. veri (LSB) ucu)
8	D1 (1. veri ucu)
9	D2 (2. veri ucu)
10	D3 (3. veri ucu)
11	D4 (4. veri ucu)
12	D5 (5. veri ucu)
13	D6 (6. veri ucu)
14	D7 (7. veri (MSB) ucu)

Şekil 1.14: LCD Ayak bağlantıları

Açıklamalar:

a. Vdd ve Vss: LCD Vdd ucuna bağlanan 5V gerilim ile çalışır. Bağlandığı kontrol elemanı ile şase bağlantısı Vss ile mutlaka yapılmalıdır.

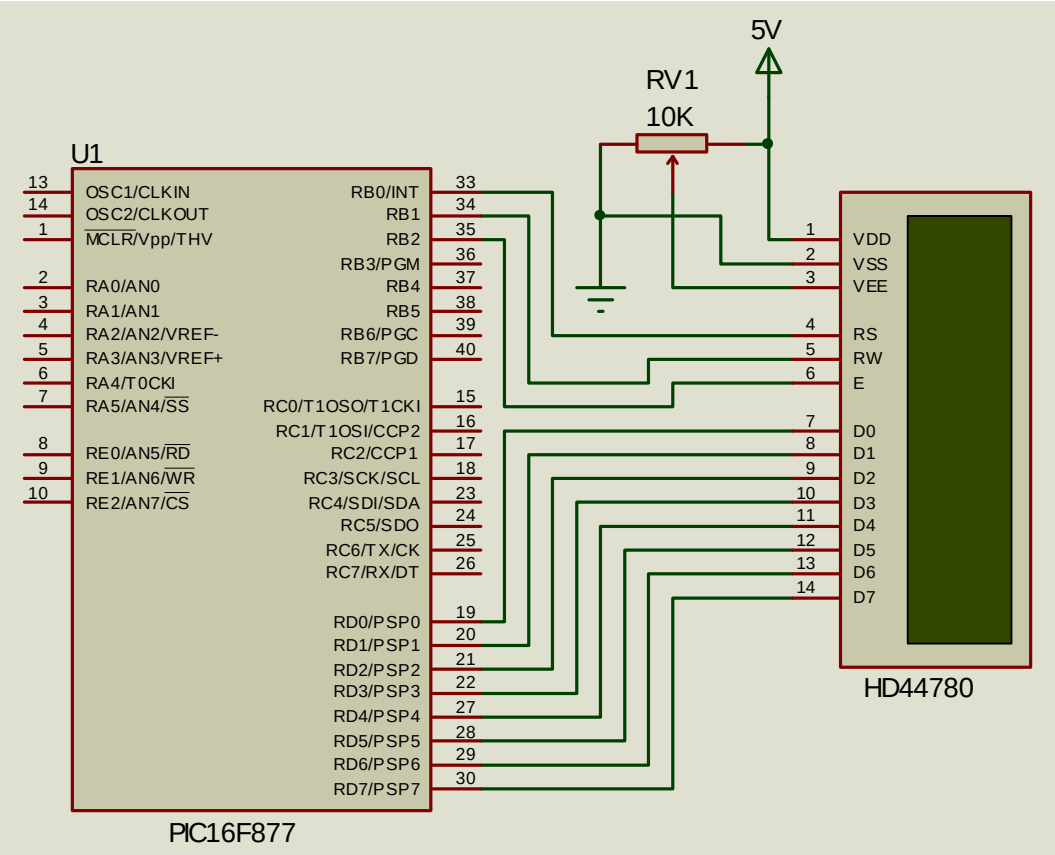
b. Vee: Vee ucu ile 5V arasına 10K değerinde bir trimpot bağlanarak LCD ekranın kontrast seviyesi değiştirilebilir.

c. RS: RS ucu lojik-0 ise komut gönderiliyor, lojik-1 ise veri gönderiliyor anlamındadır. Kontol devresinden RS ucuna önce lojik-0 gönderilir. Sonra işlenecek olan komut veri uçlarına(D0..D7) yazılır. RS ucu lojik-1 yapılarak komutun işlevi ile ilgili veri diğer bir değişle ekranda gösterilecek olan veri aynı uçlara gönderilir.

d. R/W: 0 ise LCD hafıza hücresine veri yazılır. 1 ise hafıza hücresinden veri okunur.

e. E: Gönderilen komutun uygulanması için veya verinin işlenmesi için kontrol devresinden bu uca lojik-1'den 0'a düşen (düşen kenar) bir kare dalga sinyal gönderilir.

f: D0..D7: Komut girişi ve veri giriş-çıkışı için kullanılırlar. 8 bitlik modda tümü kullanılırken 4 bitlik LCD çalışmasında D7..D4 uçları kullanılır.



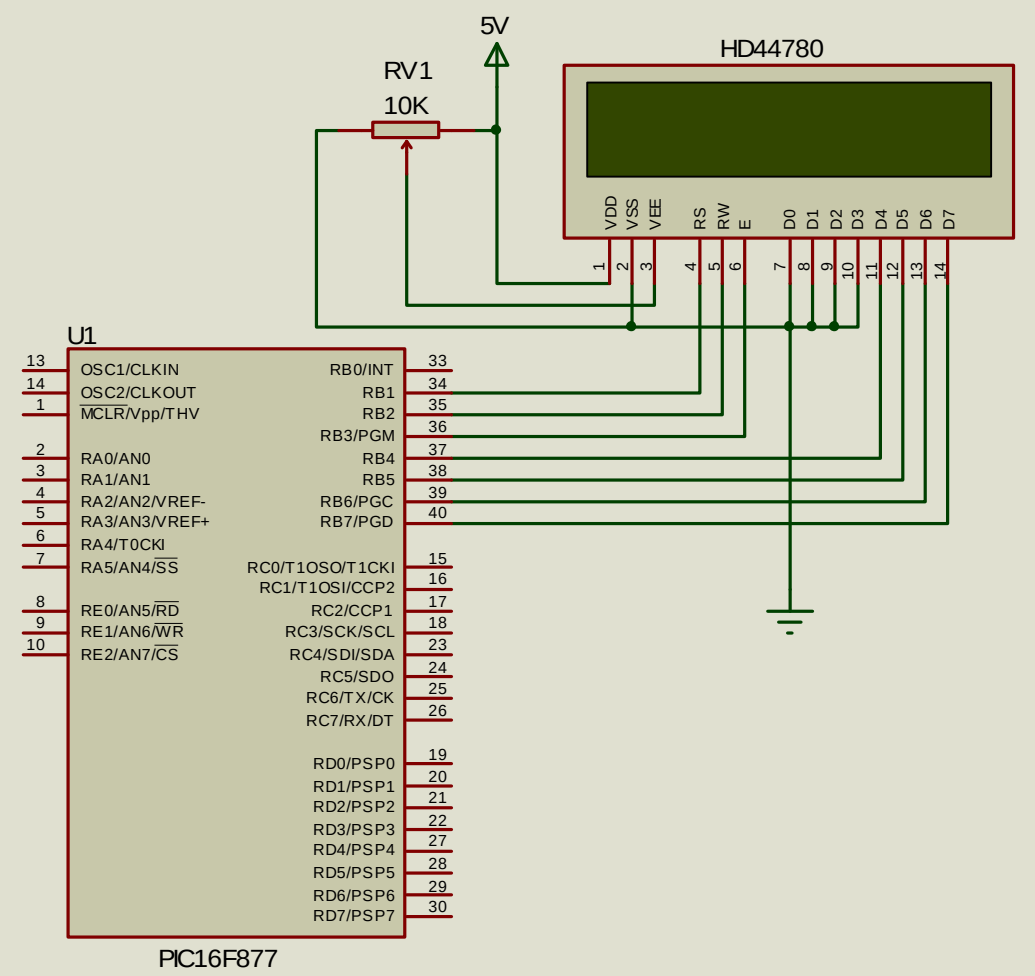
Şekil 1.15: PIC16877-LCD Bağlantısı (8 bit Modu)

Şekil 1.15'te PIC16F877 ile LCD bağlantısı için örnek bir devre görülmektedir. PortB'nin ilk üç biti kontrol bitleri için kullanılmış, PortD ise veri bitleri için kullanılmıştır. RV1 potansiyometresi ile LCD'nin kontrastı ayarlanmaktadır. Bu bağlantı ile LCD 8 bit modda kontrol edilebilir. Ancak yazılan programda program kodları 8 bitlik çalışmaya göre yazılmalıdır.

Şekil 1.16'da PIC16F877 mikrodenetleyicinin 4 bitlik moda LCD'yi kontrol ettiği örnek bir devre görülmektedir. Bu devrede LCD'nin 4 bit modda bağlantısı için sadece PortB kullanılmıştır. Böylece mikrodenetleyicinin diğer portları farklı amaçlar için kullanılabilir.

1.2.2. LCD Programlama

1.2.2.1. HD44780 Komut Seti



Şekil 1.16: PIC16877-LCD Bağlantısı (4 bit Modu)

PIC-C ile LCD'yi kontrol etmek ve veri alışverişi yapmak çok kolaylaştırılmıştır. PIC Assembly dili ile çok karmaşık olan programlar PIC-C ile başlık ve sürücü dosyaları sayesinde çok daha kısa ve anlaşılır biçimde yazılabilir. Bu kısımda LCD'nin çalışma mantığını anlamak amacıyla kataloglarda gösterilen fonksiyonlardan bahsedilecektir. Şekil 1.17'de LCD komutlarının toplu olarak gösterildiği tablo görülmektedir.

KOMUT	KOD										
	RS	R/$\overline{W}$	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	**Açıklama**
Ekran temizle	0	0	0	0	0	0	0	0	0	1	Tüm ekran temizlenir.
Başa Dön	0	0	0	0	0	0	0	0	1	*	Kursör ekran başına döner.
Giriş Kurulumu	0	0	0	0	0	0	0	1	I/D	S	Kursör hareket yönünü ve display kayma özelliğini belirtir.
Display Aç/Kapa	0	0	0	0	0	0	1	D	C	B	Display ve Kursör on/off, karekter karartma yapar.
Kursör/Display Kaydır	0	0	0	0	0	1	S/C	R/L	*	*	S/C kursör hareketi ve display kaymasını, R/L kayma yönü nüayarlar.
Fonksiyon Kurulumu	0	0	0	0	1	DL	N	F	*	*	Data uzunluğu, display satır sayısı, karekter fontunu ayarlar.
CGRAM Adresi kur	0	0	0	1	CGRAM adresi						CGRAM adresi kurar.
DDRAM Adresi kur	0	0	1	DDRAM adresi							DDRAM adresi kurar.
Meşgul bayrağı ve adresi oku	0	1	BF	CGRAM veya DDRAM adresi							BF ve adres sayıcının içeriğini okur.
CG/DD RAM adresine veri yaz	1	0	YAZILAN VERİ								CGRAM veya DDRAM adresine veri yazar.
CG/DD RAM adresinden veri oku	1	1	OKUNAN VERİ								CGRAM veya DDRAM'dan veri okur.

Şekil 1.17: HD44780 LCD Komut seti

Tablo ile ilgili açıklamalar:

- DDRAM: Display Data RAM
- CGRAM: Karakter Üretici RAM
- BF=1 ise komut kabul edilmez.LCD meşguldur. BF=0 ise komut kabul edilir.
- N=0 ise 1 satır, N=1 ise 2 satır kullanılır.
- * Önemsiz giriş 1 ya da 0
- F=1 5x10 karakter fontu, F=0 5x7 karakter fontu
- I/D=0 Kursör pozisyonunu 1 azalt, I/D=1 Kursör pozisyonunu 1 arttır.
- S=0 ise display kaymaz, S=1 ise display kayar.

- C=0 ise kursör kapalıdır, C=1 ise kursör açıktır.
- R/L=0 ise sola kayma gerçekleşir, R/L=1 ise sağa kayma gerçekleşir.
- DL=0 ise LCD 4 bit modda çalışır, DL=1 ise LCD 8 bit modda çalışır.
- S/C=0 ise kursör hareket eder, S/C=1 ise display hareket eder.
- B=0 ise kursör karartma kapalı, B=1 ise kursör karartma açık.
- LCD'nin yukarıdaki komutları işlemesi için ilgili uçlara dijital bilgi uygulandıktan sonra E ucundan 1'den 0'a düşen bir izin darbesi uygulanmalıdır. Bu sinyal uygulanmadan LCD komutu işleyemez.

Komutların Ayrıntılı Açıklamaları

a. Display temizle

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

Tüm ekranı temizler ve kursörü 0. adres olan başlangıç adresine getirir. Display Data RAM'deki tüm adreslere boşluk koduna karşılık gelen 20H yazılır. DDRAM için adres sayacı 0 yapılır. Eğer kaydırılmışsa displayi orijinal konumuna getirir. Herhangi bir karakter displayde görülmez ve kursör displayin başına gelir.

Komut uygulama zamanı: 82µs-1.64ms (8bit modda)
120µs- 4.9ms (4bit modda)

b. Başa Dön

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	*

Kursörü displayin başına getirir. DDRAM için adres sayacını 0 yapar. DDRAM içeriğini etkilemez. * etkisiz giriş anlamındadır. Display eğer kaydırılmış ise orijinal durumuna getirir.

Komut uygulama zamanı: 40µs-1.64 ms (8bit modda)
120µs-4.8 ms (4bit modda)

c. Giriş Kur

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

DDRAM adreslerinin artışına ya da azalışına göre kursör hareket yönünü ayarlar. Bu çalışmalar veri yazma ve okuma sırasında icra edilir. Bir karakter kodu DDRAM'e yazıldığıda ya da karakter kodu DDRAM'den okunduğunda;

I/D=1 olduğunda DDRAM adresi 1 arttırılır ve kursör sağa hareket eder.
I/D=0 olduğunda DDRAM adresi 1 azaltılır ve kursör sola hareket eder.

Yukarıda bahsedilen çalışma CGRAM'den okuma ve CGRAM'e yazma sırasında da geçerlidir.

S=0 ise kursörün sağa ya da sola kaymasına izin verilir. S=1 yapıldığında kursörün sağa hareketi sırasında, displayi karakter sayısı kadar komple sola kaydırır.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

d. Display Aç/Kapa

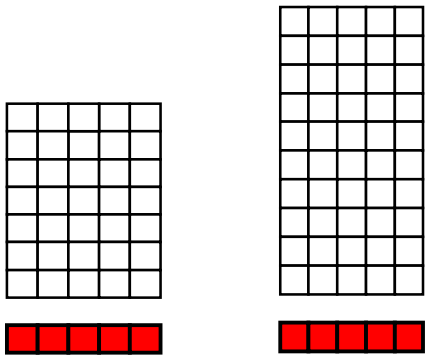
RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

Displayi ve kursörü açıp kapatır, kursörün görünümünü ayarlar.

D=1 ise display açılır. D=0 ise display kapanır. Kapalıyken display verisi enerji olduğu sürece DDRAM'de saklanır.

C=1 ise kursör ekranda görüntülenir. C=0 ise kursör ekranda görüntülenmez.

B=1 ise kursör ile gösterilen tüm noktalar yanıp söner. B=0 ise sadece kursör ekranda görülür. Kursör 5x7 dot karakterde 8. satırda, 5x10 dot karakterde 11. satırda görüntülenir.



Şekil 1.18: Kursör durumları

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

e. Kursör ve Display Kaydır

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	*	*

DDRAM içeriğini değiştirmeden kursorü hareket ettirir ve displayi sağa veya sola kaydırır. Bu fonksiyon display için araştırma ve düzeltme işlemleri için kullanılır. 2 satırlı

displayde kursör 1. satırın 40. sütununu geçtikten sonra 2. satıra geçer. 1. ve 2. satırlar aynı anda kayar.

S/C	R/L	
0	0	Kursör sola kayar. Adres sayacı 1 azalır.
0	1	Kursör sağa kayar. Adres sayacı 1 artar.
1	0	Tüm display sola kayar. Kursör display ile birlikte kayar.
1	1	Tüm display sağa kayar. Kursör display ile birlikte kayar.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

f. Fonksiyon Kur

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	*	*

Veri uzunluğunu, display satır sayısını ve karakter fontunu ayarlar.

DL: Veri uzunluğunu ayarlar. DL=1 ise gönderilen veya alınan veri 8 bit uzunluğundadır (DB7-DB0). DL=0 ise gönderilen veya alınan veri 4 bit uzunluğundadır (DB7-DB4). 4 bit veri uzunluğu seçilirse verinin önce üst dört biti sonra da alt dört biti gönderilmelidir.

N: Display satır sayısını belirtir.
F: Karakter fontunu ayarlar.

Meşgul bayrağı ve adres okumanın dışında herhangi bir komut uygulamadan önce programın başında fonksiyon kurulumu yerine getirilmelidir.

N	F	Satır	Font
0	0	1	5x 7 nokta
0	1	1	5x10 nokta
1	*	2	5x 7 nokta

5x10 noktalı karakter fontu 2 satır displayde kullanılmaz.

5x10 noktayı LCD'nin desteklemesi gereklidir. Desteklemiyorsa 5x7 nokta ayarlanmalıdır.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

g. CGRAM Adresi Kur

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	A	A	A	A	A	A

CGRAM adresini kurar. CGRAM verisi bu ayarlamadan sonra gönderilir. Adres sayacında 6 bitlik CGRAM adresi oluşur. Veri bundan sonra CGRAM için mikroişlemciden okunur veya mikroişlemciye yazılır.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

h. DDRAM Adresi Kur

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	A	A	A	A	A	A	A

DDRAM adresini kurar. DDRAM verisi bu ayarlamadan sonra gönderilir. Adres sayacında 6 bitlik DDRAM adresi oluşur. Veri bundan sonra DDRAM için mikroişlemciden okunur veya mikroişlemciye yazılır.

1 Satırlı LCD'de
AAAAAAA adresi 00H - 4FH arasında iken,

2x12 Satırlı LCD'de
1. satır için AAAAAAA 00H – 0FH arasındadır.
2. satır için AAAAAAA 40H – 4FH arasındadır.

Mikrodenetleyiciden adres kurma komutu gönderirken DB7 biti lojik 1 olduğu için adres kurma komutu 1.satır için 80H ile 8FH arasındadır. 2. satır için bu aralık C0H ile CFH arasındadır. 0FH ile 40H aralığı LCD ekranında görüntülenemez.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

i. Meşgul Bayrağı ve Adresi Oku

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	BF	A	A	A	A	A	A	A

LCD içerisinde bir çalışmanın olduğunu ifade eden meşgul bayrağını okur. Diğer görevi de adres sayacının içeriğini okumaktır. BF=1 okunuyorsa LCD'nin meşgul olduğu anlaşılır. BF=0 olmadan yeni bir komut LCD tarafından kabul edilmez. BF durumu bir sonraki komut gönderilmeden önce kontrol edilmelidir.

Aynı zamanda 7 bitlik adres sayacı değeri bu komut ile okunur. Adres sayacı hem CGRAM hem de DDRAM adresleri için kullanılır. O andaki kullanımı bir önceki komuta bağlıdır.

Komut uygulama zamanı: 0-1µs (Her iki modda)

j. CG/DD RAM Adresine Veri Yaz

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	D	D	D	D	D	D	D	D

DDRAM veya CGRAM'a 8 bitlik veri yazar. CGRAM veya DDRAM ile ilgili bu işlemler daha önce yapılan tanımlama ve adres kurma ile belirlenir. Örneğin yazma işleminden sonra adresin 1 artması veya azalması giriş kur komutuna göre gerçekleşir. Giriş kurma aynı zamanda display kaymasını da belirler.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

k. CG/DD RAM'den Veri Oku

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	D	D	D	D	D	D	D	D

CGRAM veya DDRAM'dan 8 bitlik formatta veri okur. Daha önceki komut uygulamaları ile CGRAM'ın ya da DDRAM'ın okunacağı belirlenir. Okuma komutuna başlamadan önce ya CGRAM ya da DDRAM için adres kurma komutu uygulanmalıdır. Eğer bu yapılmazsa ilk okuma verisi iptal olacaktır. Okumadan sonra giriş moduna bağlı olarak adres otomatik olarak 1 arttırılır veya azaltılır.

Komut uygulama zamanı: 40µs (8bit modda)
120µs (4bit modda)

1.2.2.2. LCD'yi Kullanıma Hazırlama

HD44780 LCD 8 bitlik ve 4 bitlik moda göre çalışabilir. PIC16F877 ile LCD kontrolü gerçekleştirmek istenirse, 8 pin veri için(D0'dan D7'ye) ve 3 pin de kontrol için olmak üzere toplam 11 pin kullanılmalıdır. Üst düzey uygulamalarda mikrodenetleyicinin bacak sayısı yetmeyebileceğinden 8 bit modunu kullanmak uygun olmayabilir. Bu nedenle LCD'yi 4 bit modunda kullanmakta fayda vardır. 4 bit ile kontrolde 4 pin veri için(D4'den D7'ye), 3 pin de kontrol için olmak üzere sadece 7 pin kullanılması yeterli olacaktır.

Her iki çalışma modunda da LCD üzerinde karakterler göstermeden önce yapılması gereken birtakım işlemler vardır. Bunlar sırasıyla aşağıda verilmiştir.

a. Enerjinin Gelmesi ve Resetleme

HD44780 LCD'ye enerji uygulandığı zaman dahili reset devresi çalışır. Enerji verildikten yaklaşık 10 msn sonra meşgul bayrağı (BF) lojik-1 olur. BF bayrağı ilk kullanıma hazırlama işlemlerinin sonuna kadar otomatik olarak meşgul durumda kalır. Programda yapılması gereken ilk kullanıma hazırlama temel işlemleri aşağıda verilmiştir.

LCD'ye enerji geldiği zaman mikrodenetleyici programının ilk başlangıçta en az 40 ms beklenmesinde fayda vardır. Kullanılan gerilim kaynağı yeterli akım veremiyorsa ya da kalitesiz ise sistemde hata oluşabilir. Gerilim kaynağı lojik yükselme zamanı 10 msn'den daha fazla ise LCD iç reset devresinin çalışmasında problem çıkabilir. Bu nedenle gerilim kaynağı özenle seçilmelidir. Portların giriş çıkış işlemleri için kurulumu sırasında belirli bir zaman geçtiğinden dolayı port kurulumu için bekleme yapmakta fayda vardır.

Programda ilk işlem olarak LCD fonksiyon kurma komutunun yüksek dört biti ile resetlenmelidir. Bu amaçla 8 bitlik çalışma modunda arka arkaya 3 defa 30h komutu gönderilir. 4 bitlik modda ise arka arkaya 3 defa 20h komutu gönderilir. Meşgul bayrağı kullanılmıyorsa her komut arasında 1 ms'lik gecikme yapılmalıdır.

b. Display Temizleme

Yazılan programda display temizle komutu ile ekran temizleme işlemi gerçekleşir.

c. Fonksiyon Kurulumu

Fonksiyon kurma komutu ile veri uzunluğu, display satır sayısı ve karakter fontu ayarlanır.

d. Displayin Açılması

Display Aç/Kapa komutu ile display açılır.

e. Giriş Modu Kurulumu

Giriş Kurulumu komutu ile kursörün sağa veya sola hareketi ve displayin kayma izin ayarı yapılır.

f. Adres Kurulumu

LCD'ye veri yazılmak istenen adres, adres kurma komutu ile LCD'ye gönderilmelidir.

Bütün bu temel hazırlık işlemlerinden sonra displaye yazma işlemi veya okuma işlemi gerçekleştirilebilir.

1.2.2.3. LCD Kontrolüde Dikkat Edilecek Hususlar

Mikrodenetleyici ile LCD kontrolünde birtakım unsurlara dikkat edilmelidir. RS, R/W ve E uçlarına uygun lojik değerler uygulanmalıdır.

- LCD'ye veri veya komut gönderilecekse R/W ucu lojik-0 olmalı, LCD'den bir veri okunacaksa R/W ucu lojik-1 yapılmalıdır.

- Eğer komut gönderilecek ise önce RS ucu lojik-0 yapılmalı sonra komut gönderilmelidir. Veri gönderilecek ise RS ucu lojik-1 yapılmalıdır.

- Bir komutun veya verinin LCD içerisinde işleme konması için E ucuna düşen kenarlı bir izin darbesi uygulanmalıdır.

- Komut uygulama zamanlarına dikkat edilmelidir. Bir komut uygulandıktan sonra daha önce belirtilen komut icra süreleri kadar bekleme yapmak gereklidir. En kesin çözüm LCD'nin meşgul olup olmadığını kontrol etmektir. BF meşgul bayrağı daha önce anlatıldığı gibi kontrol edilerek bu gerçekleşir. Böylece en kısa zaman gecikmesi elde edilir.

1.2.2.4. PICC Dili ile LCD Programlama

CCS C derleyici yaması bilgisayara yüklenip MPLAB programına bağlandığında PICC dili ile program yazılıp derlenebilir. Tabi istenirse başka programlar ve yamaları kullanılabilir. Bilgisayardaki PICC klasörü incelendiğinde iki alt klasör göze çarpar. Bunlar "Devices" ve "Drivers" klasörleridir.

"Devices" klasörü içerisinde mikrodenetleyicilerin donanım tanımlarının yer aldığı başlık dosyaları yer almaktadır. Bu dosyaların uzantısı .h ile biter. Programda PIC16F877 entegresi ile ilgili donanım kullanılacağından dolayı "#include <16F877.h>" komutu programın en başında kullanılmalıdır. Aslında 16F877.h dosyası C dili ile yazılmıştır. İncelendiğinde PIC16F877 mikrodenetleyicisinin donanım kullanımına yönelik ifadeler ve fonksiyonlar göze çarpar. 16F877 başlık dosyası programın başında tanımlandığında başlık dosyasında tanımlanan fonksiyonlar ve ifadeler yazılan PIC-C programında kullanılabilir.

➡ 16F877.h başlık dosyasını inceleyiniz.

"Drivers" klasörü içerisinde çeşitli elektronik yapıların sürücü dosyaları yer almaktadır. Burada yer alan dosyaların uzantıları .C ile biter. Bu klasörde LCD'ye özel LCD.C dosyası vardır ve bu dosya içerisinde LCD kulanımı için fonksiyonlar yer almaktadır. Eğer LCD kullanılmak isteniyorsa LCD.C sürücü dosyası programın başında tanımlanmalıdır. Bu amaçla "#include <LCD.C>" komutu kullanılır. Aşağıda CCS C derleyicisi ilk kurulduğunda LCD.C dosyasının içeriğinden bir kısım görülmektedir. Aşağıda görüldüğü gibi, sürücünün en başında sürücü tanımlandığında kullanılabilecek fonksiyonlar gösterilmiştir.

➡ LCD.C sürücü dosyasını inceleyiniz.

```
////                        LCDD.C                                    ////
////              Driver for common LCD modules                       ////
////                                                                  ////
////  lcd_init()   Must be called before any other function.          ////
////                                                                  ////
////  lcd_putc(c)  Will display c on the next position of the LCD.    ////
////                The following have special meaning:               ////
////                 \f  Clear display                                ////
////                 \n  Go to start of second line                  ////
////                 \b  Move back one position                       ////
////                                                                  ////
////  lcd_gotoxy(x,y) Set write position on LCD (upper left is 1,1)   ////
////                                                                  ////
////  lcd_getc(x,y)   Returns character at position x,y on LCD        ////
////                                                                  ////
// As defined in the following structure the pin connection is as follows:
//     D0  enable
//     D1  rs
//     D2  rw
//     D4  D4
//     D5  D5
//     D6  D6
//     D7  D7
//
//   LCD pins D0-D3 are not used and PIC D3 is not used.
// Un-comment the following define to use port B

//#define use_portb_lcd TRUE

struct lcd_pin_map {                // This structure is overlayed
        BOOLEAN enable;             // on to an I/O port to gain
        BOOLEAN rs;                 // access to the LCD pins.
        BOOLEAN rw;                 // The bits are allocated from
        BOOLEAN unused;             // low order up.  ENABLE will
        int     data : 4;           // be pin B0.
      } lcd;
```

⬅

LCD.C sürücüsü HD44780 temelli tüm LCD'ler için kullanılabilen bir dosyadır. İlk başlangıç değeri 4 bitlik çalışma modu için uygundur. Fakat bu haliyle programda kullanılmaz. Öncelikle kullanılacak donanıma uyup uymadığı incelenmelidir. Bu haliyle LCD'nin PortD'ye bağlanması gereklidir. Ancak bu modülde örnek olarak şekil 1.16'daki devreye göre programlar yazılacaktır. Bu devrede LCD PortB'ye bağlanmıştır. Bu nedenle LCD.C dosyası öncelikle açılmalı, yukarıda ok ile gösterilen kısımda gerekli donanım değişiklikleri yapılmalı ve dosyanın orjinalliğini bozmamak için farklı adla kaydedilmelidir.

Yukarıdaki tanımlamalarda öncelikle portb kullanımı aktif hale getirilmelidir. Bu modülde 4 bitlik uygulama devresinde LCD PortB'ye bağlı olduğundan aşağıdaki düzeltme gerçekleştirilir.

```
//#define use_portb_lcd TRUE        satırı

#define use_portb_lcd TRUE          yapılır.
```

Yukarıdaki komut topluluğunda lcd_pin_map yapısına bakıldığında 0.bit (D0) izin ucu, D1 rs ucu, D2 ise rw ucu olarak tanımlanmıştır. PortB'nin 3. biti (D3) kullanım dışıdır(unused). Lcd.data 4 bit olarak yapı içersinde belirtilmiş ve böylece 4 bitlik çalışma tanımlanmıştır. Öncelikle burada yapılan donanım yapısı değiştirilmelidir. LCD.C dosyasında aşağıdaki değişiklik yapılıp LCD2.C olarak kaydedilmelidir.

```
struct lcd_pin_map {
        BOOLEAN unused;
        BOOLEAN rs;
        BOOLEAN rw;
        BOOLEAN enable;
        int     data : 4;
      } lcd;
```

Bu modülde kullanılan LCD kontrol devresine göre PB.0 kullanılmamaktadır. Bu nedenle yukarıdaki yapıda "unused" değişken adı ile ifade edilmiştir. Sırasıyla PB.1 rs ucuna, PB.2 rw ucuna, PB3'de E izin ucuna bağlanmışlardır. PB.4 ile PB.7 bitleri 4 bitlik data için kullanılmaktadır.

LCD.C sürücüsü kullanılarak aşağıdaki hazır fonksiyonlar PIC-C programı içerisinde kullanılabilir. Sürücü programın başında tanımlanmazsa program derlemesi sırasında hata meydana gelecektir. Hazır fonksiyonlar ve görevleri aşağıda verilmiştir.

1. lcd_init()

LCD'yi hazırlamak için mutlaka öncelikle kullanılmalıdır.

2. lcd_putc(c)

LCD ekranında istenilen koordinata "c" ile ifade edilen karakteri veya karakter dizisini yazar. Bu fonksiyon ile yardımcı anahtarlar kullanılabilir.

\f : Ekranı temizler ve kursörü 1. satırın 1.sütununa alır.
\n : Kursörü 2. satırın başına getirir.
\b : Kursörü 1 adım geriye hareket ettirir.

3. lcd_gotoxy(x,y)

LCD ekranında yazma koordinatını ayarlar. X sütun numarasını, Y ise satır numarasını temsil eder. LCD ekranında sol üst köşenin koordinatı (1,1)'dir.

4. lcd_getc(x,y)

(x,y) ile ifade edilen koordinattaki karakteri fonksiyon üzerinden döndürür. Diğer bir ifade ile istenen koordinattaki karakteri okur.

Aşağıda Şekil 1.16'daki devre baz alınarak LCD.C sürücüsü ve hazır fonksiyonlar kullanılmadan gerçekleştirilen bir LCD kontrol programı görülmektedir.

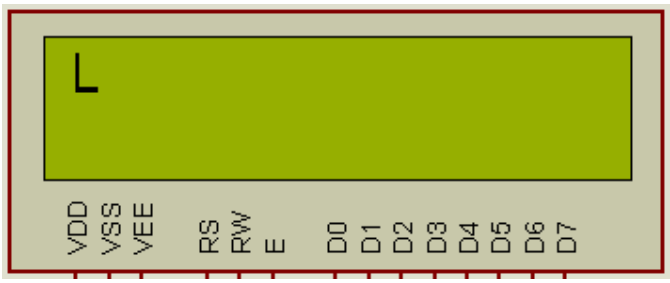
Örnek 1:

```
//LCD KONTROL PROGRAMI-LCD1_1.C
//LCD ekranında 80H adresine L harfi yazdırma
//PB0-X / PB1-RS / PB2-RW / PB3-E / PB4...PB7 - 4 bit data

#include <16F877.h>
#use delay(clock=20000000)      //Devrede 20Mhz'lik krsital kullnılıyor
#fuses hs,NOWDT,NOPROTECT,NOLVP
#byte port_b=6                  // port_b nin adresi

#define RS    PIN_B1
#define RW  PIN_B2
#define E     PIN_B3
#define PB4  PIN_B4
#define PB5  PIN_B5
#define PB6  PIN_B6
#define PB7  PIN_B7

io_setout()
   {
     set_tris_b(0x00);          // port_b = çıkış
   }

komutyaz()
   {
    Output_Low(RS);             // RS=0 Komut geliyor.
    Output_Low(RW);             // R/W=0 Yazma modu
   }

veriyaz()
   {
     Output_High(RS);           // RS=1 veri geliyor.
     Output_Low(RW);            // R/W=0 Yazma modu
    }



uyg()
```

```
   {
     Output_High(E);  // Uygulamadan önce portların kurulması için 1 ms ile beklenir
     delay_ms(1);       // Kullanılmazsa LCD çalışması kitlenmektedir.
     Output_Low(E);
     delay_ms(1);
   }

 clrdsply()
   {
     komutyaz();
     port_b=0x00;
     uyg();
     port_b=0x10;         //01H ekran temizleme komutu gönderiliyor.
     uyg();
    }

dsplay_on()
   {
     komutyaz();
     port_b=0x00;
     uyg();
     port_b=0xC0;     //0CH Display açma komutu gönderiliyor.
     uyg();
    }

lcdreset()
    { komutyaz();
      port_b=0x20;    // 2 değeri porb7-4'e gönderiliyor.
      uyg();
      port_b=0x20;
      uyg();
      port_b=0x20;   //4 bit modda resetleme için 3 defa 20h PB7-4'e uygulanıyor.
      uyg();
    }

lcdkur()
   {
     komutyaz();
     port_b=0x20;  // 2 değeri porb7-4'e gönderiliyor.
     uyg();
     port_b=0x80;  // 8 değeri porb7-4'e gönderiliyor.
     uyg();             // 28h komutu uygulandı. 4 bit modu 2 satır 5*7 fontu  kuruldu.
  }

adreskur()
   {
     komutyaz();
```

```
    port_b=0x80;
    uyg();                  // 80H PortB'den gönderiliyor.
    port_b=0x00;           // DDRAM adresi 00h olarak kuruluyor. 1.satır 1.sütun
    uyg();
   }

verigonder()
  {
    veriyaz();
    port_b=0x42;       // RS'nin 1'de kalması sağlanıyor.
    uyg();              // Bu nedenle 42 ve C2 verileri gönderiliyor.
    port_b=0xC2;
    uyg();             // 4CH verisi gönderildi.
  }                    // DDRAM adresi üzerinden L harfi(4Ch) 1.satır 1.sütuna yazıldı.

giriskur()
  {
    komutyaz();
    port_b=0x00;
    uyg();
    port_b=0x60;
    uyg();             // 06H PortB'den gönderiliyor. Giriş modu sağa hareket
  }

void main()
  {
    io_setout();
    lcdreset();
    clrdsply();
    lcdkur();
    dsplay_on();
    giriskur();
    adreskur();
    verigonder();
  }
```

Örnek-1'deki program Şekil 1.16'daki devre üzerinde çalıştırıldığında LCD ekranında Şekil 1.19'daki çıktı elde edilir.



Şekil 1.19: Örnek-1 için LCD görüntüsü

Aşağıdaki örnekte ise LCDdrv.C adlı bir sürücü oluşturulmuş ve programda bu sürücü kullanılarak 4Fh adresine L harfi yazdırılmıştır.

Örnek 2:

```
//LCD KONTROL PROGRAM-sürücü dosyalı
//LCD ekranında 2. satır 16. sütuna (4Fh) L harfi yazdırma
//PB0-X / PB1-RS / PB2-RW / PB3-E / PB4...PB7 - 4 bit data

#include <LCDdrv.c>              //Gerekli tüm tanımlamalar sürücüde mevcuttur
#fuses hs,NOWDT,NOPROTECT,NOLVP

void main()
{   io_setout();
    clrdsply();
    lcdreset();
    lcdkur();
    dsplay_on();
    giriskur();
    adreskur();
    verigonder();
  }
```

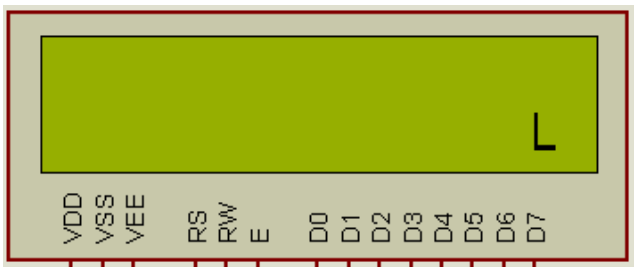
LCDdrv.c sürücü programı aşağıda görülmektedir.

```
//LCD SÜRÜCÜ-LCDdrv.c
//PB0-X / PB1-RS / PB2-RW / PB3-E / PB4...PB7 - 4 bit data

#include <16F877.h>
#use delay(clock=20000000)
#byte port_b=6                  // port_b nin adresi
#define RS PIN_B1
#define RW PIN_B2
#define E PIN_B3
#define PB4 PIN_B4
#define PB5 PIN_B5
#define PB6 PIN_B6
#define PB7 PIN_B7

io_setout()
   {
     set_tris_b(0x00);          // port_b = çıkış
   }

komutyaz()
   {
    Output_Low(RS);             // RS=0 Komut geliyor.
```

```
     Output_Low(RW);             // R/W=0 Yazma modu
    }

veriyaz()
    {Output_High(RS);            // RS=1 veri geliyor.
     Output_Low(RW);             // R/W=0 Yazma modu
      }

uyg()
    {Output_High(E);             //Portların kurulması buradaki 1 ms ile bekleme
     delay_ms(1);
     Output_Low(E);
     delay_ms(1);
    }

clrdsply()
    {komutyaz();
     port_b=0x00;
     uyg();
     port_b=0x10;           //01H ekran temizleme komutu
     uyg();
     }

dsplay_on()
    {komutyaz();
     port_b=0x00;
     uyg();
     port_b=0xC0;           //0CH Display açma komutu
     uyg();
    }

lcdreset()
   { komutyaz();
      port_b=0x20;           // 2 değeri porb7-4'te
      uyg();
      port_b=0x20;
      uyg();
      port_b=0x20;   //4 bit modda resetleme için 3 defa arka arkaya 20h
      uyg();
      }

lcdkur()
   { komutyaz();
      port_b=0x20;  // 2 değeri porb7-4'te
      uyg();
      port_b=0x80;  // 28h 4 bit modu 2 satır 5*7 fontu
      uyg();
```

```
    }

adreskur()
   { komutyaz();
     port_b=0xC0;
     uyg();                  // DDRAM adresi 4Fh 2.satır 16.sütun
     port_b=0xF0;
     uyg();
    }

verigonder()
   { veriyaz();
     port_b=0x42;
     uyg();
     port_b=0xC2;
     uyg();                  // DDRAM adresine L harfi(4Ch) 2.satır 16. sütuna yazılıyor.
   }

giriskur()
   {komutyaz();
    port_b=0x00;
    uyg();
    port_b=0x60;
    uyg();                   //06H PortB'den gönderiliyor. Giriş modu sağa hareket
   }
```

Örnek-2 programı ile sürücü dosyası mantığı da anlaşılmaktadır. Ana LCD kontrol programında include komutu ile sürücü dosyası tanımlanırsa sürücü programı ile ana program birlikte derlenir. Böylece sürücü programdaki fonksiyonlar ana programda kullanılabilirler. Aynı çalışma şekli Lcd.c sürücüsü için de geçerlidir. Örnek-2'deki programın çıktısı Şekil 1.20'de görülmektedir.



Şekil 1.20: Örnek-2 için LCD görüntüsü

Aşağıdaki örnekte PIC-C klasöründe oluşturulan LCD2.c sürücüsü kullanılarak 80h adresine L harfi yazan program görülmektedir.

Aynı çalışmayı gerçekleştiren program örnek-1'de sürücü kullanılmadan daha önce yazılmıştı. Örnek-1 ve Örnek-3'ü karşılaştırınız.

Örnek-3

```
//Lcd'ye Lcd2.c sürücüsü ile L harfinin yazılması//
#include <16F877.h>
#use delay(clock=20000000)
#include <LCD2.C>
#fuses hs,NOWDT,NOPROTECT,NOLVP
void main()
{
  lcd_init();                //LCD'yi ilk kullanıma hazırlıyor.
  lcd_putc("\fL");        //Ekran temizlenip L harfi 1.satır 1.sütuna yazılıyor.
}
```

Aşağıdaki örnekte ise 2 satır kullanılmış ve "LCD" kelimesi aşağı ve yukarı sürekli hareket ettirilmiştir.

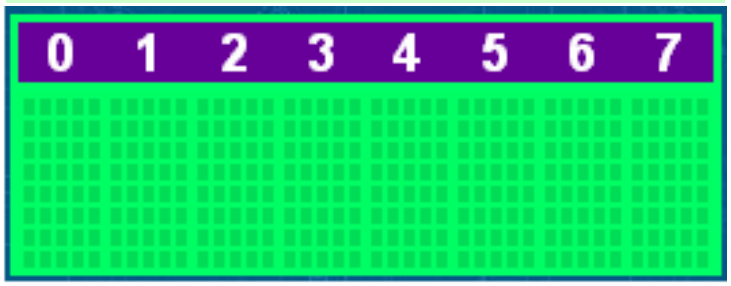
Örnek-4

```
//Yazının yukarı ve aşağı sürekli yazılması ve hareket ettirilmesi
#include <16F877.h>
#use delay(clock=20000000)
#include <LCD2.C>
#fuses hs,NOWDT,NOPROTECT,NOLVP
void main()
  { int a=1,b=1;
    lcd_init();                  // LCD'yi ilk kullanıma hazırlıyor.
   While(1)                     //Sonsuz döngü
     {
        lcd_putc("LCD");  // LCD'ye "LCD" ifadesini yazıyor.
       delay_ms(200);     // 200ms gecikme
        lcd_putc("\f");        // LCD temizleniyor
        lcd_gotoxy(1,2);    // 2.satır 1. sütuna geçiliyor.
        lcd_putc("LCD");
        delay_ms(200);
        lcd_putc("\f");      //Ekran temizlenip 1. satıra geçiriliyor.
      }
     }
```

1.2.2.5. LCD'de Karakter Oluşturma

LCD'nin CGRAM belleği kullanılarak Şekil 1.10'da görülen ASCII tablosundaki karakterlerin dışında özel karakterler gerçekleştirilebilir. Örneğin Türkçe karakterler ASCII tablosuna göre LCD'ye yazdırılamaz. Aynı şekilde yukarı ok, dörtgen, ağaç figürü gibi karakterler için hazır bir kod mevcut değildir. Bunlar gibi özel karakterleri LCD ekranına getirmek için CGRAM kullanılır. Ancak LCD enerjisi kesildiğinde CGRAM'da oluşturulan karakter de silinir. CGRAM'in yapısı ve kullanımı Şekil 1.21 ile anlaşılabilir.

Komut	40	48	50	58	60	68	70	78
Adres	0	8	10	18	20	28	30	38



Şekil 1.21: CGRAM Yapısı

CGRAM'de toplam 8 adet özel karakter oluşturulabilir. Bu amaçla yukarıda 0-7 aralığında gösterilen 8 adet karakter kümesi kullanılır. Her kümenin bir adresi vardır ve CGRAM'deki karakter bu adresle DDRAM'e aktarılır ve ekranda gösterilir. Yukarıdaki şekilde küme adreslerine ulaşmak için gerekli komutlar görülmektedir. Örneğin 4. kümeye ulaşmak için 60H komutu kullanılmalıdır. Daha önce anlatıldığı gibi DB6 biti CGRAM kurma komutunu bildirir.

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	A	A	A	A	A	A

Yukarıda görüldüğü gibi aslında adres sayacında 6 bitlik CGRAM adresi oluşur. Örneğin 60H komutu ile CGRAM adresi 20H (32D) yapılır. Kümedeki 1. piksel satırının adresi küme adresinin kendisidir. Bundan sonra her piksel satırının adresi birer birer artar. Örneğin 0. kümenin ilk satır adresi 0h ve bu adrese ulaşmak için gereken komut da 40H'dir. Sonraki adresler 1H, 2H,3H,4H,5H,6H ve 7H'dir. 7H adresli piksel satırına ulaşmak için gereken komut ise 47H'dir. CGRAM adresinin kurulmasından sonra karakter verileri tek tek her satıra yerleştirilir. Temel karakter hücrelerinin gerçekte, sekiz piksel yüksekliğinde ve beş piksel genişliğinde olduğu görülebilir, ancak karakterlerin çoğu için sadece üst yedi satır kullanır. Alttaki sıra, genellikle alt çizgi imleci (kursör) için kullanılır. Her karakter sadece beş piksel genişliğinde olduğundan, sadece D0'dan D4'e kadar olan veri bitleri kullanılır, sol taraftaki D7, D6 ve D5 bitleri ihmal edilir. CGRAM kullanımı için gereken işlem basamakları aşağıda sırasıyla verilmiştir.

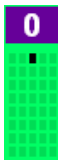
1. İlk kurulum işlemleri sırasında giriş modu karakter oluşturma işlemine uygun olarak kurulur. Çünkü I/D=1 ise CGRAM adresi 1 artar, 0 ise 1 azalır.
2. Öncelikle özel karakteri oluşturmak için seçilen kümedeki piksel satırlarında hangi piksellerin dolu(1) hangilerinin boş(0) olacağına kağıt üzerinde karar verilir ve piksel satır verileri çıkartılır.
3. CGRAM'deki küme adresine ulaşmak için uygun komut LCD'ye gönderilir.
4. R/W=0 ve RS=1 yapılır.
5. CGRAM'e ilk piksel satırı verisi gönderilir. Bu işlemden sonra CGRAM adresi otomatik olarak bir artar. (Giriş modu I/D=0 ise 1 azalır)
6. RS=0 yapılır ve DDRAM adresi kurulur.

7. 4 bit modda D7-D4 arası, 8 bit modda ise D2-D0 arası bitlere, hangi kümedeki karakter ekrana yazdırılacak ise o bilgi girilir. Örneğin 0. küme için 000 bilgisi en düşük üç bite yazılır.
8. RS=1 yapılır. 8 bit modda bir kere E izin darbesi, 4 bit modda ise 2 defa E izin darbesi uygulanır.

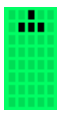
Bu işlemlerden sonra özel olarak üretilen karakter LCD ekranında görüntülenir.

Aşağıda CGRAM'de yukarı ok karakterinin oluşturulması ve LCD ekranında gösterilmesi ile ilgili örnek verilmiştir. Programın sadece daha önceki programlardan farkı yazılmıştır. "---" ile belirtilen yerlere gerekli olan diğer fonksiyonlar da yazılmalıdır. Programda dsplay_on() fonksiyonunda kursör aktif yapılmıştır. 0x0C yerine 0x0E komutu uygulanmıştır.

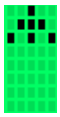
```
//LCD KONTROL PROGRAMI- CGRAM karakteri oluşturma
//PB0-X / PB1-RS / PB2-RW / PB3-E / PB4...PB7 - 4 bit data
---
---
cgram()
  {  komutyaz();
     port_b=0x40;
     uyg();
     port_b=0x00;          //CGRAM adresi 40h yapıldı.
     uyg();                //0. kümenin adresi ile 0. piksel satırına ulaşılıyor.

     veriyaz();            //R/W=0 ve RS=1 yapıldı.Veri yazılacak.


     port_b=0x02;          // 0. piksel satırı 04h
     uyg();                // **Alt dört bite 2 yazılmalı.
     port_b=0x42;          //  RS=1 kalmalı
     uyg();

     port_b=0x02;          //1. piksel satırı 0Eh
     uyg();
     port_b=0xE2;
     uyg();

     port_b=0x12;          //2. piksel satırı 15h
     uyg();
     port_b=0x52;
     uyg();

     port_b=0x02;          //3. piksel satırı 04h
     uyg();
     port_b=0x42;
     uyg();
```



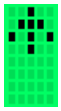
00100



01110

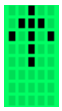


10101

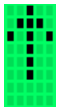


00100

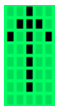
```
      port_b=0x02;          //4. piksel satırı 04h
      uyg();
      port_b=0x42;
      uyg();

      port_b=0x02;          //5. piksel satırı  04h
      uyg();
      port_b=0x42;
      uyg();

      port_b=0x02;          //6. piksel satırı  04h
      uyg();
      port_b=0x42;
      uyg();

      port_b=0x02;          //7. piksel satırı 00h
      uyg();
      port_b=0x02;
      uyg();
           //  7. piksel satırı kursör kullanımı için boş bırakıldı.
     //   CGRAM'a 5x7 fontunda yukarı ok karakteri yazılmış oldu.
  }
---
---
---
---
verigonder()
    { port_b=0x00;          //D7-D4'e 0000 yazıldı. Böylece 0. küme ayarlanmış oldu
      veriyaz();            //CGRAM okunup DDRAM'a yazılacak.
      uyg();                //4 bit mod çalışması için 2 defa E darbesi
      uyg();
    }
---
---
---
//ANA PROGRAM
void main()
{   io_setout();
    lcdreset();
    clrdsply();
    lcdkur();
    dsplay_on();
    giriskur();
    cgram();
    adreskur();
    verigonder();
    }
```



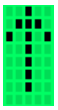
00100



00100

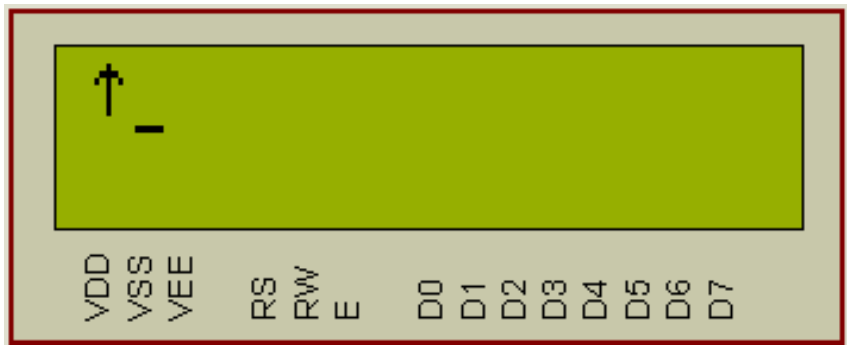


00100



00000

Programın yazılması ve çalıştırılması ile elde edilen görüntü aşağıda verilmiştir. Görüldüğü gibi kursör özel karakterin yazılması ile sağa kaymıştır.



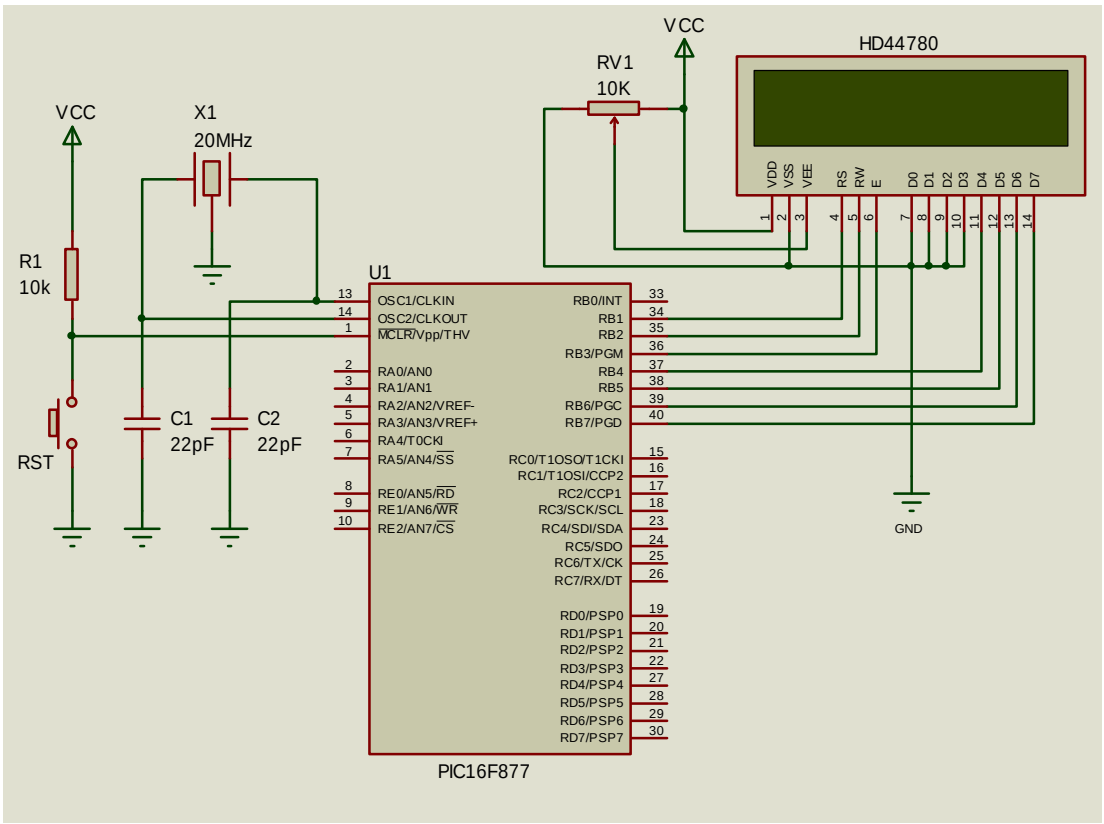
Şekil 1.22: LCD'ye Ok karakterinin yazdırılması

➡ Aynı programı, Ç,Ş gibi Türkçe karakterleri CGRAM'da oluşturup yeniden yazınız.

UYGULAMA FAALİYETİ

Kullanılan Malzeme ve Araç-Gereç

1. 1 adet PIC16F877
2. 1 adet 40 bacaklı entegre soketi
3. 1 adet HD44780 işlemcili LCD
4. 1 adet LCD için uygun 14'lü soket
5. 1 adet 20 MHz kristal
6. 2 adet 22pF kondansatör
7. 10KOhm direnç ve 10 KOhm trimpot
8. 1 adet buTon
9. PIC programlayıcı devresi
10. 5V DC gerilim kaynağı
11. 1 adet deney bordu
12. Yeterli sayıda zil teli
13. Yankeski ve kargaburun



Şekil 1.23: Öğrenme Faaliyeti-1 uygulama devresi

Şekil 1.24: Öğreneme Faaliyeti-1 uygulama çıktısı

İŞLEM BASAMAKLARI	ÖNERİLER
Ø Şekil 1.23'teki devre için gereken malzemeleri ve araç gereci hazırlayınız.	Ø 5V gerilim kaynağı kaliteli bir kaynak olmalıdır.
Ø PIC16F877'nin ve HD44780 LCD'nin katalog bilgilerini inceleyiniz.	Ø Entegrenin ve LCD'nin çalışmasını, karakteristik özelliklerini inceleyiniz
Ø Devreyi kurunuz.	Ø Enerji vermeyiniz. Devrenizi önce öğretmeninize kontrol ettiriniz.
Ø Şekil 1.24'teki görüntüyü LCD ekranında veren programı MPLAB editöründe LCD sürücüsü kullanarak yazınız.	Ø PIC-C sürücüsünü uygulama devresine uygun hale getirmelisiniz. Ø Görüntü koordinatları (1,1)'dir.
Ø PIC16F877 entegresine programı yükleyiniz.	Ø Entegrenin bacaklarına sokete çıkarıp takarken dikkat ediniz.
Ø Programı Şekil 1.23'teki devrede çalıştırınız.	Ø LCD'deki yazının istediğiniz koordinata geldiğinden emin olunuz.
Ø Aynı programı LCD sürücüsü kullanmadan yazınız.	Ø Daha önce anlatılanlardan yola çıkarak orijinal fikirler geliştiriniz.
Ø PIC16F877 entegresine programı yükleyiniz.	Ø Entegrenin bacaklarına sokete çıkarıp takarken dikkat ediniz.
Ø Programı Şekil 1.23'teki devrede çalıştırınız.	Ø LCD'deki yazının istediğiniz koordinata geldiğinden emin olunuz.
Ø Sürücü kullanmadan yazdığınız programda değişiklik yaparak aynı yazıyı 2. satırda yazdırınız.	Ø Sütun numarası sabit kalmalıdır.
Ø Sürücü kullanmadan yazdığınız programdaki yazıyı 1 sn aralıklarla 1. ve 2. satır arasında sürekli hareket ettiren değişikliği yapınız.	Ø Yazı 1 sn 1. satırda 1 sn 2. satırda kalacaktır. Sürekli bir döngü gerçekleştirilecektir.

Şekil 1.25: Öğrenme Faaliyeti-1 uygulama işlem basamakları

ÖLÇME VE DEĞERLENDİRME

OBJEKTİF TESTLER (ÖLÇME SORULARI)

1. PIC16F877 mikrodenetleyicinin program belleği kaç KB'dır?
A) 8 B) 16 C) 24 D) 32

2. PIC16F877 mikrodenetleyicide kaç adet port bulunmaktadır?
A) 3 B) 4 C) 5 D) 6

3. PIC16F877'de bank seçimi için hangi yazmaç kullanılır?
A) Option B) Intcon C) Status D) PIR1

4. LCD'de dışarıdaki elektronik birimlerle haberleşen, bu birimlerin gönderdiği kodları uygulayan birime ne ad verilir?
A) DDRAM B) CGRAM C) Mikroişlemci D) Sıvı kristal ekran

5. LCD ekranında görüntülenecek karakterin kayıtlı olduğu bellek hangisidir?
A) CGROM B) CGRAM C) Mikroişlemci D) DDRAM

6. Aşağıdakilerden hangisi ASCII tablosu verilerini saklar?
A) CGRAM B) DDRAM C) CGROM D) DDROM

7. LCD'den veri okumak için aşağıdakilerden hangisi kesinlikle yapılmalıdır?
A) RS=0, E=1 B) RS=1, E=1
C) RS=0, R/W=0 D) RS=1, R/W=1

8. LCD'yi resetlemek için aşağıdakilerden hangisi 4 bitlik moda 3 kere uygulanmalıdır?
A) 5H B) 10H C) 15H D) 20H

9. LCD.C sürücüsü kullanıldığında aşağıdakilerden hangisi LCD'yi ilk kullanıma hazırlar?
A) Kur() B) Lcd_lur() C) Lcd_init() D) Dsply_kur()

10. Lcd_gotoxy(10,2) komutu ile LCD ekranında hangi fiziksel adrese ulaşılır?
A) 12 B) 49 C) 4A D) CA

DEĞERLENDİRME

Cevaplarınızı cevap anahtarlarıyla karşılaştırınız. Yanlış cevaplarınız için faaliyetin ilgili konularını tekrar ediniz.

ÖĞRENME FAALİYETİ-2

AMAÇ

Mikrodenetleyici ile servo motor kontrolünü hatasız olarak yapabileceksiniz.

ARAŞTIRMA

Ø Servo motor çeşitleri ve yapıları hakkında ön araştırma yapınız.

2. SERVO MOTOR KONTROLÜ

Otomasyon sistemlerinde üretimin otomatik olarak gerçekleşmesi için elektrik enerjisinin harekete çevrilmesi gerekmektedir. Hareket elde etmek için kullanılan en önemli elemanlar motorlardır. Daha önceki modüllerde doğru akım motoru ve adım motorunu öğrenmiştiniz. Bu öğrenme faaliyetine başlamadan önce bu motorların yapısını incelemenizde fayda vardır. Doğru akım motoru ve adım motoru ile hassas açılara dönme hareketi yapmak zordur. Özellikle robot sistemlerinde belirli açılarla dönme hareketi yapılmalıdır. Bu nedenle doğru akım motorları geliştirilmiş ve "servo motor" ortaya çıkmıştır. Servo-motorlar, verilen girişe göre istenen açısal konuma gelen motorlardır. Günümüzde ac veya dc gerilimle çalışabilen, çeşitli güçlerde ve ebatlarda servo motorlar mevcuttur. Servo motorların çok çeşitli uygulamalarda kullanılmasını nedenleri;

- Güvenilir olması
- Yüksek tork değerlerine sahip olması
- Konumlamada yüksek doğruluk göstermesi
- Kolay kurulum ve bağlantı özelliği
- Kontrol kolaylığı
- Ekonomik oluşu

olarak verilebilir.

Bu öğrenme faaliyetinde literatürde "RF Servo Motor" ya da "RC Servo Motor" olarak da geçen DC Servo Motorun temel yapısı incelenecektir.

2.1. Servo Motorun Yapısı

Sahip olduğu bir mil üzerinden gücünü dışarıya aktarabilen mekanizmaya "Servo" adı verilir. Bu mekanizmaya özel olarak tasarlanmış motor birimi ve dişli kutusunun ilave edilmesiyle servo motorlar üretilirler. Hobi amaçlı kullanılan küçük çaplı robotlarda ve oyuncak arabalarda kullanılan servo motorlar için "RF(RC) Servo Motor" kavramı kullanılır.

Uzaktan kumandalı uçak, araba vs gibi hobi araçlarının yönlendirme mekanizmalarının harekete geçirilmesi, radyo kontrollü gemiler ve kuklalar diğer kullanım alanlarının başında gelir.

Mil hareket edebildiği aralıkta belirli bir pozisyona, servoya gönderilen kodlu bir sinyale uygun olarak getirilebilir. Kodlu sinyal servonun giriş ucunda bulunduğu sürece konum aynen korunacaktır. Sinyaldeki değişimlere göre milin açısı da değişecektir.

Şekil 2.1: Servo motorlar

Servo motor robot kontrolünde son derece kullanılışlı bir motordur. Şekil 2.1'de görüldüğü gibi küçük boyutlu olmasına rağmen çok güçlüdür ve çok az akım çeker. Bunun yanı sıra yüksek döndürme momentlerine(tork) sahiptirler. Servo-motorlar 4,7 ile 7 Volt besleme gerilimleri arasında çalışabilirler, kontrol sinyali gerilimi TTL seviyesinde olmalıdır. Servo motorlar yüksek güç ve uygun sinyal tepkileri için genellikle 5V'a yakın gerilimlerde çalıştırılır.

Şekil 2.2'de servo motor iç yapısında bulunan parçalar görülmektedir. Elektronik kumanda devresi, geri besleme potansiyometresi, dc motor, dişli grubu ve dişli kutusu servo motoru meydana getirir. DC motor çift mıknatıslı bir stratora ve fırçalı rotora sahiptir. Motor mili belirli bir dönme oranına sahip dişli sistemine bağlıdır. Böylece yüksek bir tork değerine ulaşılır. Dişli sisteminin çıkışında geri besleme potansiyometresi mil konumunu elektronik kumanda devresine iletir. Kumanda devresinin görevi, mil konumunun Ton süresi ile ifade edilen açıya gelmesi için motoru döndürmek ve açı noktasında durdurmaktadır.

Dışarıdaki kontrol devresinden teorik olarak 1 ile 2 milisaniye arasında Ton süresi değişen bir kontrol sinyali her 20 milisaniyede bir servoya gönderilir. Ton süresi 1,5 ms olduğunda servo motor mili başlangıç pozisyonundadır. Ton süresi 1 milisaniye ise motor

mili başlangıç pozisyonuna göre 90 derece bir yöne, 2 milisaniye ise -90 derece zıt yöne konumlanır. 1-2 ms arasında herhangibir zamanlama ile de -90 ve +90 derecelik açıların arasındaki herhangibir açıya konumlanabilir. Ancak bu açı aralığı pratikte 1-2 ms olmayabilir. Servo kontrol devresi tasarımı için öncelikle kataloglara başvurulmalı ve motor milinin konumlanacağı yön ve süreler denenmelidir. Uygun zamanlama sinyalleri servoya uygulandığında, içerideki elektronik kumanda devresi, ilk önce gelen darbelerin darbe genişliğini ölçer, daha sonra geri besleme potansiyometresinin konumuna bakar ve kendi darbe osilatörünün darbe genişliği gelen darbelerle eşitleninceye kadar motoru hareket ettirir.

Servo motor içerisinde yer alan elektronik kumanda devresi ve geri besleme potansiyometresi çok önemli yapılardır. Şekil 2.2'de potansiyometre motorun hemen yanında görülmektedir. Geri besleme potansiyometresi lineer değişen kaliteli bir potansiyometredir ve mile doğrudan bağlıdır. Mille birlikte dönme hareketi yapar. Böylece direnç değeri değişir. Potansiyometrenin kullanım amacı milin pozisyonunun doğru değerde olup olmadığını anlamaktır. Mil gereken açıya gelmişse potansiyometre bunu algılar ve motor durur. Kontrol devresine gelen dönüş açısı bilgisi yanlış ise motor doğru açıyı bulana kadar döndürülür.

Servomotor milinin taradığı açı teorik olarak maksimum 180°'dir. Bazı servo motor modellerinde maksimum tarama açısı 210° ye kadar artabilmektedir. Maksimuım tarama açısı kataloglardan ve deneme yanılma yoluyla bulunabilir. Servo motorların iç yapısında yer alan mekanik sınırlama nedeniyle servo motor mili bu süpürme aralığının dışına çıkamaz.

Şekil 2.2: Servo motor iç yapısı

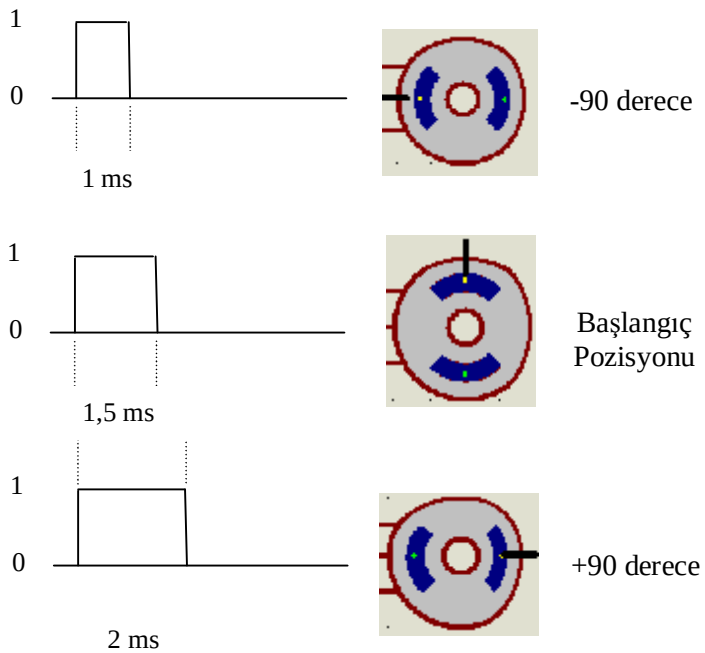
Şekil 2.3'te 3 adet kablo görülmektedir. Bunlar servo motorun dışarıdaki elektronik sistemlerle bağlantısını sağlarlar. Bunlardan kırmızı olan +5V gerilim için, siyah olan şase bağlantısı için ve yeşil olan da servo kontrol siyali için kullanılır. Bazı servolarda kontrol sinyali için kullanılan kablonun rengi beyaz veya turuncu olabilmektedir.

Şekil 2.3: Servo motor bağlantı kabloları

Elektronik kumanda devresince motora uygulanan güç katedilmesi gereken mesafe ile doğru orantılıdır. Eğer motor milinin geniş bir açıyı katetmesi gerekiyorsa motor tam hızda çalıştırılır. Eğer küçük bir açıda dönme gerekiyorsa motor hızı kumanda devresince düşürülür. Bu çalışma için kumanda devresinin kullandığı kontrol yöntemi oransal kontrol yöntemidir.

Oransal kontrolün gerçekleşmesi için bir kontrol sinyaline ihtiyaç vardır. Bu kontrol sinyali daha önce anlatıldığı gibi servo motora milin konumlanması gereken açıyı bildirir. Dışarıda mikroişlemci tabanlı bir sistem örneğin PIC16F877 servo motorun kontrol bilgisini çok rahat üretebilir. Milin konumlanacağı açı, bu bağlantı üzerinden gönderilen bir sinyalin lojik-1 seviyesinde kalma süresi (Ton) ile belirlenir. Ton süresi "darbe genişlik modülasyonu (PWM)" ile ayarlanır.

Yatay eksen referans alındığında, servo motor milinin 90°'lik açı pozisyonuna nötr pozisyon da denilebilir ve açı zamanlaması hesabında 0 noktası olarak alınır. Nötr pozisyonun solunda kalan açılar saat yönünün tersine artacak şekilde negatif, sağında kalan açılar da saat yönünde artacak şekilde pozitif olarak düşünülebilir. Nötr pozisyon için gereken Ton süresinin 1,5 ms olduğu varsayılırsa, eğer Ton süresi 1.5 ms den kısa süreli ise mil konumu -90° ye doğru, 1.5ms den uzun ise mil +90° ye doğru ilerleyecektir. Ton süresi 1 ms ise servo mili başlangıç pozisyonuna göre -90 dereceye, 2 ms ise +90 dereceye konumlanır. Bu düşünce şekli servo motor milinin konumlanacağı açıları anlayabilmek açısından tasarımcıya yarar sağlar. Bu anlatılanlardan farklı düşünce şekilleri de üretilebilir. Şekil 2.4'te örnek olarak bir servo motorun kontrol sinyallerine göre ulaştığı konumları göstermektedir.



Şekil 2.4: Kontrol sinyallerinin servo motora etkisi

Şekil 2.4'te görüldüğü gibi Ton süresi milin konumlanacağı açıyı belirler. Ton süreleri ve mil sapma yönleri üreticiden üreticiye farklılık gösterebilir. Ancak servo motorların zamanlama prensipleri aynıdır. Servo motorlu bir sistem tasarlanmak istendiğinde katalog bilgilerine başvurulmalı ve servo karakteristikleri ön uygulamalarla belirlenmelidir.

2.2. Sürücü Devreleri

Servo motoru kontrol etmek için besleme gerilimi ve bir kontrol sinyalinin gerektiğini daha önce belirtmiştik. Servo motoru kontrol etmek için elektronik bir devre tasarlanmalıdır. Bu elektronik devrenin mikrodenetleyici tabanlı olmasında fayda vardır. Bir servo motoru sürmek için tasarlanan elektronik devrenin güç kısmı "servo sürücü" olarak adlandırılır. Endüstriyel robotlarda ve diğer nümerik olarak kontrol edilen ekipmanlarda servo sürücüler geniş ölçüde kullanılır. Servo sürücünün kullanım amacı yüksek güç gerektiren servo motorlara gerekli akım ve gerilimi sağlamaktır. Servo sürücüye servo motorun hareketi için uygun kontrol sinyalleri uygulanır. PWM sinyali uygulanan servo sürücülere PWM servo sürücüler adı da verilir. PWM modülasyonu ile transistör veya mosfetler anahtarlanarak servo motora daha yüksek güç aktarılabilir.

Transistorlü devrelerde, özellikle transistorlerin lineer çalışma bölgesinde ısı şeklinde ortaya çıkan bir güç harcanır. Bu nedenle güç kaybı meydana gelir. Transistörlerdeki güç kaybını azaltmak ve servo sürücülerin verimini iyileştirrnek için DC konvertörlerde olduğu gibi darbe genişlik modulasyon tekniği kullanılır. Bir PWM sürücüde, transistör ya doyum bölgesinde yada kesim bölgesindedir. Böylece kaybedilen güç azalır ve servo motora daha yüksek güç aktarılır.

Servo motor kontrol sinyalinin uygulanması ile hareket ettiğinde hareketini gerçekleştirmek için besleme kaynağından akım çeker. Kontrol devresi ve servo motor aynı besleme kaynağından akım çektiğinde istenmeyen parazit sinyaller ve gerilim dalgalanmaları oluşabilir. Bu durum eğer güç kaynağı kalitesiz ise mikrodenetleyicinin çalışmasını etkiler. Düşük güçlü servo motorlar mikrodenetleyici portuna bir led diyot gibi doğrudan bağlanabilirler. Ancak karmaşık sistem tasarımnlarında servomotorun mikrodenetleyici ile aynı kaynağı kullanması özellikle motorun dönüş hareketleri sırasında program çalışmasını etkileyebilir.

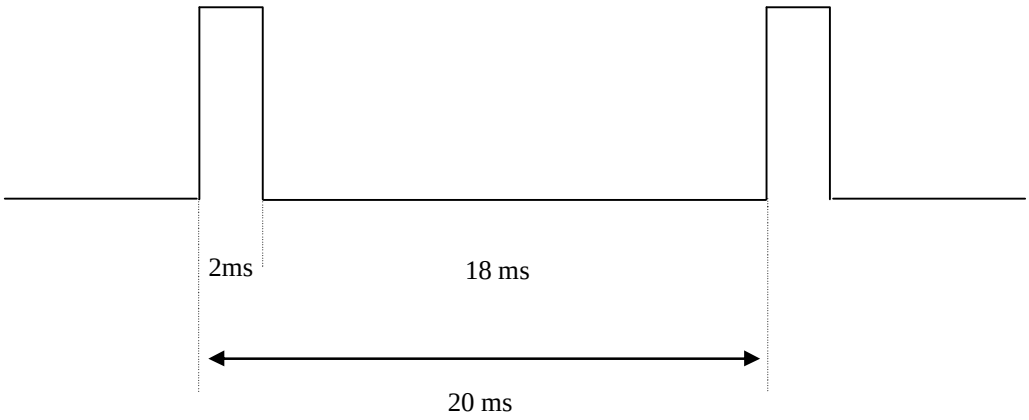
Servomotorun gücünü arttırmak için kondansatör kullanılabilir. Kondansatör deşarj akımı servo motora takviye yaparak gücünü arttırır. En etkin çözüm servo motor ile mikrodenetleyici arasında tampon bir eleman kullanmaktır. Örneğin şekil 2.6'da görüldüğü gibi mosfet kullanılarak kontrol entegresinin ve motorun besleme kaynakları ayrılabilir.

Sürücü tasarımında diğer önemli bir nokta da servomotorların boyutları ve özelliklerinin farklı olabileceği gerçeğidir. Bu bağlamda kataloglara başvurularak karakteristik özelliklerine ve kablo renklerine dikkat etmek gerekir.

Yazılan programda ya da sayısal çıkış üreten devrede servonun kontrol girişine uygun bir kare dalga sinyal tasarlanır. Eğer bir mikrodenetleyici programı yazılıyorsa +90 ve -90 derecelik açılar için gereken Ton zaman değerleri arasında bir lojik-1 süresi ayarlanmalıdır.

Her lojik-1 darbesi arasında 10-18 ms lojik-0 ile bekleme yapılmalıdır. Aslında her Ton süresi arasında 10 ms'lik Toff süreleri servo motorun hareket etmesi için yeterli olacaktır.

Servo motora kontrol sinyali uygulanmazsa mil boşta kalır ve el ile bile döndürülebilir. Servonun konumunu koruması için sürekli olarak kontrol sinyalinin uygulanması gerekir. Ton süresi değiştiğinde servo motor mili yeni açı konumuna hareket eder. Aşağıda periyotu 20 ms olan bir kontrol sinyali görülmektedir.



Şekil 2.5: Örnek kontrol sinyali

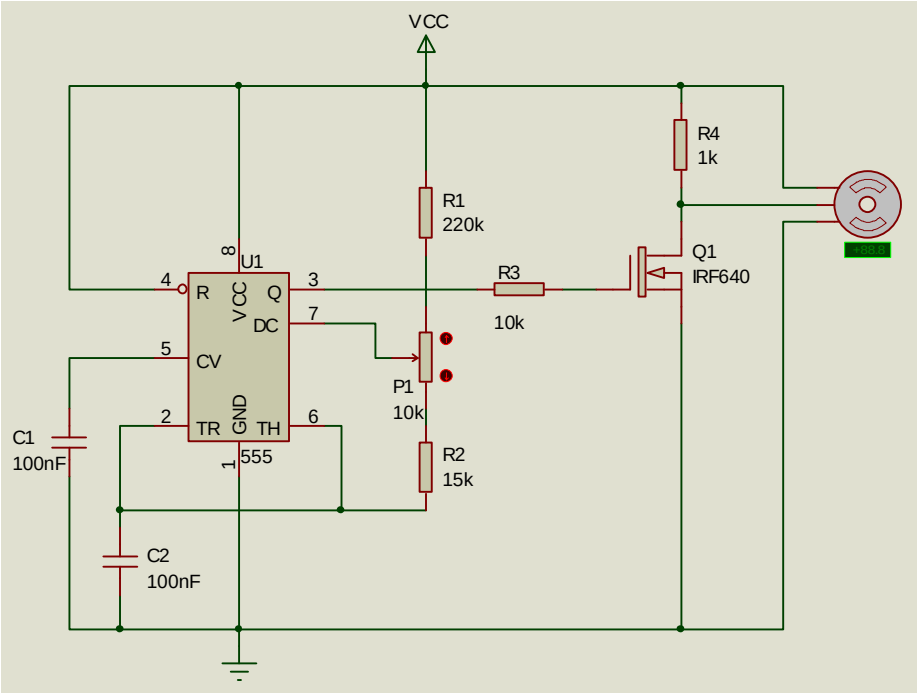
Şekil 2.5'te servo motor milini teorik olarak 90 derecelik konuma getiren kontrol sinyali görülmektedir. Sinyal şeklinde görüldüğü gibi Ton darbesi 18 ms aralıklarla sürekli verilmektedir. Süreklilik sağlanmazsa mil boşta kalır. Eğer mile bağlı bir robot kolu varsa ağırlık nedeniyle mil konum değiştirebilir ve istenmeyen durumlarla karşılaşılabilir.

Daha önce belirtildiği gibi servo motor mili tam tur atamaz. Servo motorun içerisinde mekanik olarak sınırlama yapılmıştır. Servo motoru şekil 2.5'te görüldüğü gibi 2 ms Ton süreli sinyaller ve 10-18 ms'lik bekleme aralıklarını tekrarlayarak konumlayabiliriz. Daha sonra servo motoru bir uç konumdan diğer uç konuma tam tersi yönde konumlandırmak için de 1 msn. sinyal ve 10-18 ms'lik bekleme sürelerinden oluşan bir dizi sinyal gerekir.

Kontrol sinyali arka arkaya uygulandığı zaman şaftın yeni konumuna ulaşması için bir süre geçer. Servo motorun iç devresinin kontrol sinyaline karşı olan tepki süresi de önemlidir. Sinyali üreten devrenin tasarımında tepki süresi ve yeni konuma ulaşma süresi de göz önüne alınmalıdır.

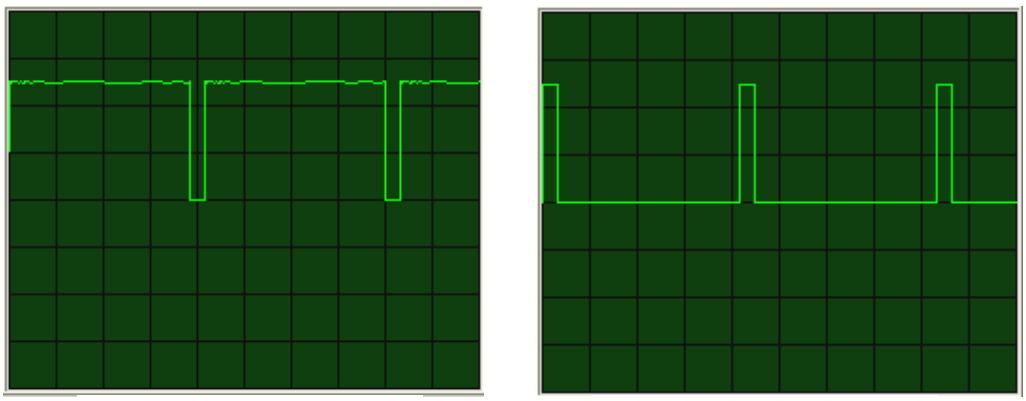
2.1.1. Servo Motorun 555 Entegresi ile Kontrolü

Konum bilgisini veren kontrol sinyalini üretmek için çeşitli devreler tasarlanabilir. Darbe üreteci olarak sinyal jeneratörü, özel tasarlanmış osilatörler veya dijital devreler kullanılabilir. Şekil 2.6'da 555 osilatör entegresi ile yapılan bir sürücü devresi görülmektedir.



Şekil 2.6: 555 Entegresi ile yapılan sürücü devresi

Devrede 555 çıkışında PWM sinyali elde edilmektedir. P1 potansiyometresi ile Ton süresi ayarlanır. Servomotor doğrudan 555 çıkışına bağlanamaz. Doğrudan bağlandığında entegre çalışmasında kararsızlıklar meydana gelebilir. Servo motorun verimli bir şekilde sürülebilmesi ve kararsızlıkları önleyerek kaliteli bir kontrol sinyali elde edilmesi için mosfet kullanılır. Mosfet kapı girişine gelen kare dalga sinyal ile mosfet ms bazında açılıp kapatılır. Mosfetin tampon olarak kullanıldığı bu devrede servo motor Vcc kaynağından ihtiyacı olan akımı çeker.



a. Mosfet giriş sinyali **b. Mosfet çıkış sinyali**

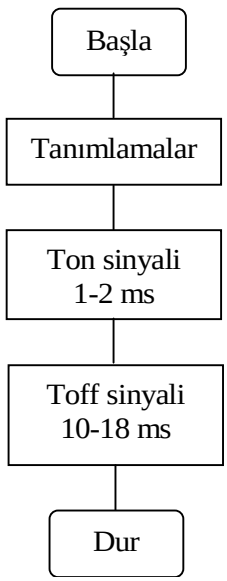
Şekil 2.7: 555 Servo sürücü devresi sinyalleri
Time/Div=5ms Volt/Div=2V

Şekil 2.7'deki sinyallerde görüldüğü gibi mosfet çıkışındaki sinyalin Ton zamanı yaklaşık 1,5 ms'dir. Tepe değeri de 5V'tur. Servo motor mili başlangıç pozisyonuna gelir. Burada mil hareketsiz bekler. Ton zamanı değişirse yeni konumuna dönüş yapar. Sinyal şekillerinde görüldüğü gibi mosfet çıkışında mosfet giriş sinyalinin 180 derece faz farklı versiyonu elde edilmiştir. Mosfet giriş sinyali lojik-1 iken mosfet iletime geçer. Akaç ucu lojik-0 (şase) potansiyeline gelir. Giriş sinyali lojik-0 iken mosfet yalıtımdadır. Bu durumda akaç ucu lojik-1 potansiyeline ulaşır. Şekil 2.7'de görüldüğü gibi mosfet uzun süreler iletimde kalmaktadır. Bu durumda mosfet daha fazla güç harcayacaktır. Mosfetin iletimde kalma süresini kısaltmak için servo kontrol sinyali çıkışı mosfetin kaynak ucundan alınabilir. Bu durumda çıkış ile giriş aynı fazlı olur ve mosfet sadece servo motorun iletimde kalması gereken zamanlarda güç harcar. Bu amaçla kullanılabilecek mosfet bağlantı örneği şekil 2.9'da görülmektedir.

2.1.2. Servo Motorun PIC16F877 Mikrodenetleyici ile Kontrolü

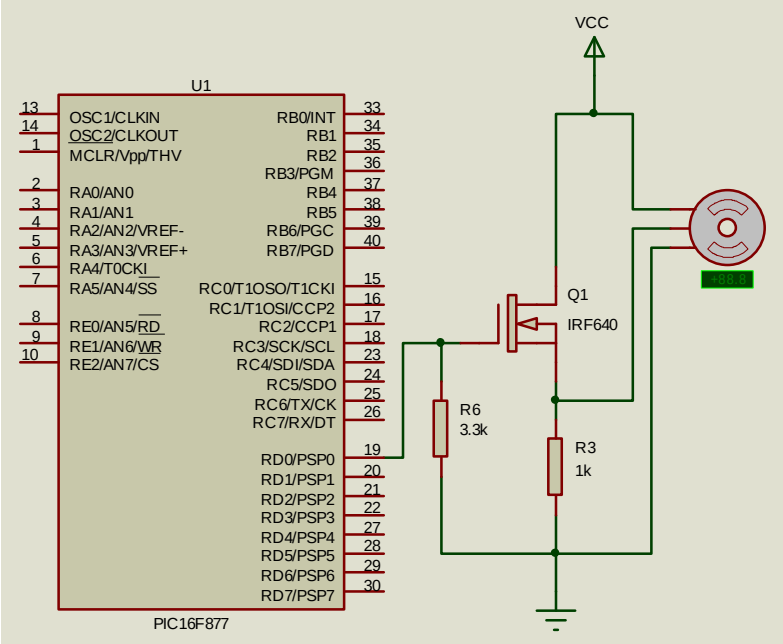
PIC16F877 mikrodenetleyici kullanarak çok kaliteli zamanlama sinyalleri elde edilebilir ve servo motor hassas açılarla hareket ettirilebilir. Bu bağlamda program içerisinde PIC-C dilinin avantajarını kullanmak gerekir. Örneğin 1 ms'lik bir zamanlama için "delay_ms(1)" komutunu kullanmak yeterlidir.

Şekil 2.8'de servo motoru kontrol etmek için gereken akış diyagramı görülmektedir. PIC16F877'yi ilk kullanıma hazırladıktan sonra servonun konumlanacağı açıyı bildiren zamanlama sinyali oluşturulur. Bu sinyal alt program yardımıyla da oluşturulabilir. Öncelikle servo motorun kontrol ucuna bağlı port biti lojik-1 yapılır. Kullanılan servo motorun karakteristiğine göre Ton süresi ayarlanır ve bu süre kadar bekleme yapılır.Böylece kontrol ucuna açı kontrol bilgisi ulaşmış olur. Lojik-1 seviyesindeki beklemeden sonra PIC ucu lojik-0 yapılır. Aynı şekilde 10-18 ms lojik-0'da bekleme yapılır. Bu işlemler sürekli tekrarlanırsa motor mili boşta kalmadan sürekli açı pozisyonunu korur.



Şekil 2.8: Sürücü sinyali akış diyagramı

Şekil 2.9'da PIC16F877'li bir servo sürücü devresi örneği görülmektedir. Örnek devrede görüldüğü gibi PIC ile servo motor arasına n kanal bir mosfet bağlanmıştır. Servo motor dönüş hareketi yapmaya başladığında gerilim kaynağından akım çeker. Akım çekilmesi esnasında gerilim kaynağındaki dalgalanma mikrodenetleyiciyi etkiler. Mosfet kullanılmadan da 5V'luk servo motorlar doğrudan mikrodenetleyici portuna bağlanarak dönme hareketi yapabilir. Ancak mosfet gibi arada bir tampon eleman kullanıp servo motor için ikinci bir besleme kaynağı kullanmak sürücü devrenin ve tüm sistemin çok daha kaliteli ve güvenilir olmasını sağlayacaktır. Özellikle servo motor bir yükü hareket ettirdiği zaman daha fazla akıma ve dolayısıyla güce ihtiyaç duyacağından harici besleme kaynağı yararlı olacaktır. Harici besleme kaynağının kullanılıp kullanılmayacağı yapılan sisteme göre tasarımcının karar vereceği bir problemdir. Şekil 2.9'da mosfet ve servo motor için ikinci besleme kaynağı(Vcc) kullanılmıştır.



Şekil 2.9'daki devrede mosfet kullanıldığından ve çıkış kaynak (source) ucundan alındığından zamanlama sinyalleri mikrodenetleyici çıkışı ile aynı fazlı olarak üretilmiştir. Örnek-1'de basit bir servo motor programı görülmektedir.

```
//SERVO MOTOR KONTROL PROGRAMI- ORNEK-1
//PD0: Servo kontrol sinyali için kullanılacak

#include <16F877.h>
#use delay(clock=20000000)
#fuses HS,NOWDT,NOPROTECT,NOLVP
#byte port_d=8                          //portd'nin adresi
#define CNT PIN_D0
io_setout()
   {
```

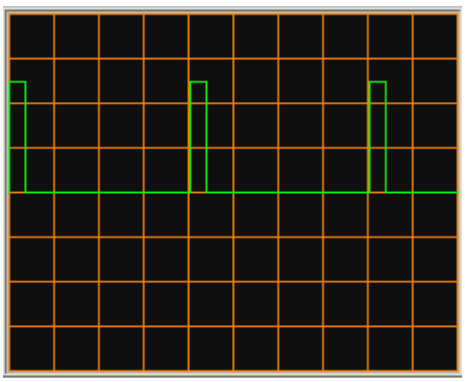
```
        set_tris_d(0x00);                  // port_d = çıkış
   }

void main()
{io_setout();
     while(1)
     {
      Output_High(CNT);
      delay_us(1750);                      // Lojik-1 süresi 1,75 ms
      Output_Low(CNT);
      delay_us(18250);                     // Lojik-0 süresi 18,25 ms
     }
}                                          // Periyot 20 ms
```

Şekil 2.10'da N kanal mosfet çıkışı görülmektedir. Mosfet çıkışı ile mikrodenetleyici çıkışı aynı fazlıdır. Mosfet kapı girişine lojik-1 seviyeli sinyal geldiğinde mosfet iletime geçer. Kaynaktaki R3 direncinin üst ucu lojik-1 olur. Böylece giriş lojik-1 iken çıkış da lojik-1'dir. Tam tersi bir şekilde kapı girişine lojik-0 seviyeli sinyal geldiğinde mosfet yalıtımdadır. Bu durumda mosfet yalıtımda olduğu için R3 direncinin üst ucu şase potansiyelindedir, diğer bir deyişle lojik-0'dır. Sonuç olarak girişdeki zamanlama sinyalinin aynısı çıkışta elde edilir. Mosfet sayesinde servo motor kendisi için gereken akımı gerilim kaynağından çeker.

Mosfet çıkışındaki sinyalin Ton süresi 1,75 ms, Toff süresi de 18,25 ms'dir. Mosfet çıkışındaki sinyalin periyotu ise;

T=Ton+Toff formülüne göre
T=1,75+18,25=20 ms bulunur.

Zamanlama sinyaline göre servo motor 1,75 ms ile ifade edilen açıya doğru dönüş yapar ve bu açıda bekler.



Şekil 2.10: Servo sürücü devresi mosfet çıkışı

Time/div=5 ms Volt/div=2V

➡ Servo motor milinin nötr pozisyonu 1,5 ms, + 90 derece için Ton süresi 2 ms ve - 90 derece için Ton süresi 1 ms ise 1,75 ms'lik Ton süresi yaklaşık kaç derecelik açıya karşılık gelir?

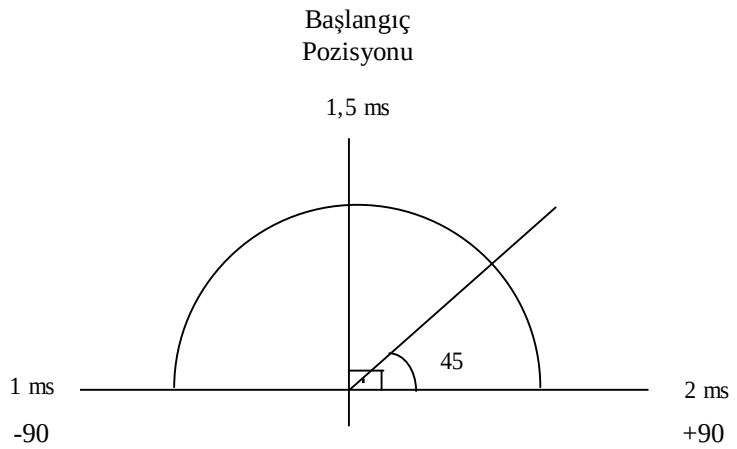
Soruda verilen karakteristiğe sahip servo motorda 1 derecelik dönüş için gereken Ton süresini yaklaşık olarak bulabiliriz. Buna göre;

Başlangıç pozisyonuna 0 derece dersek, 0 dereceden +90 dereceye kadar olan zaman farkı= 2ms-1,5ms=0,5 ms (500 ms) bulunur.

Derece başına düşen zaman=500 ms /90=5,55 ms 'dir.

Servo motorun +45 derecelik konuma gitmesini istiyoruz. +45 derece için gereken Ton zaman farkını bulmak için;

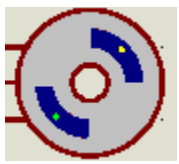
Ton=45.(5,55 ms)=250 ms bulunur.



Şekil 2.11: Servo motor açı ve zaman durumları

Bulunan zaman farkı programda belirlenecek Ton süresi için kullanılacaktır. Motor mili 1,5 ms'lik başlangıç konumundan saat yönünde + 45 derece ilerleyecekse 1,5 ms'nin üzerine 250 ms eklenir. Örnek-1 programında 45 derecelik açı için gereken zamanlama tasarlanmıştır. 45 derece için gereken Ton süresi

Ton=1,5ms+0,25 ms=1,75 ms bulunur. Şekil 2.9'daki mosfet çıkışındaki sinyalin Ton süresinin 1,75 ms olması bu yüzdendir.

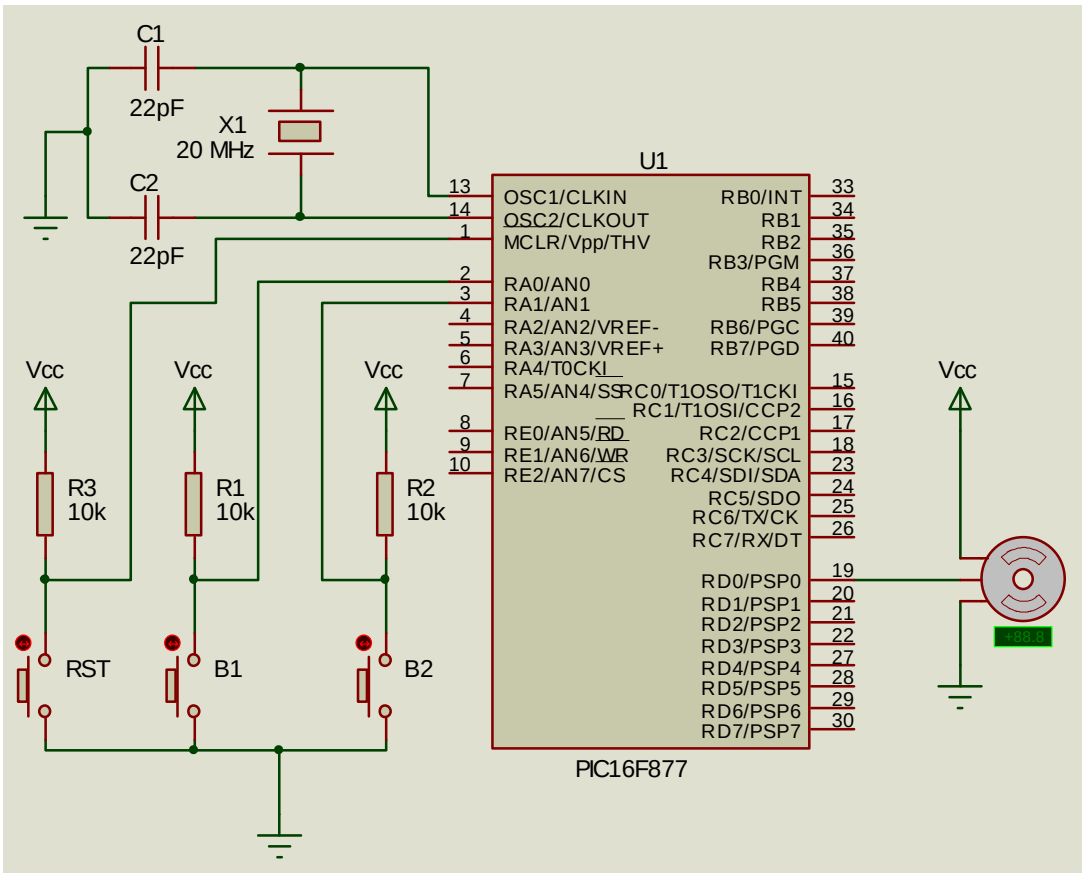


Şekil 2.12: Servo motorun 45 dereceye konumlanması

UYGULAMA FAALİYETİ

Kullanılan Malzeme ve Araç-Gereç

1. 1 adet PIC16F877
2. 1 adet 40 bacaklı entegre soketi
3. 1 adet RC Servo Motor
4. 1 adet 20 MHz kristal
5. 2 adet 22pF kondansatör
6. 3x10KOhm dirençler
7. 3 adet buTon
8. PIC programlayıcı devresi
9. 5V DC gerilim kaynağı
10. 1 adet deney bordu
11. Yeterli sayıda zil teli
12. Yankeski ve kargaburun



Şekil 2.13: Öğrenme Faaliyeti-2 uygulama devresi

Devrenin Çalışması:

Devreye ilk enerji verildiğinde servo motor başlangıç pozisyonunda olacaktır. Bu aşamadan sonra B1 buTonuna basıldığında motor mili -90 dereceye gelecek, B2 buTonuna basıldığında ise +90 dereceye dönecektir. Çalışma sürekli +90 ve -90 derece arasında gerçekleşecektir.

İşlem Basamakları	Öneriler
Ø Şekil 2.13'teki devre için gereken malzemeleri ve araç gereci hazırlayınız.	Ø 5V gerilim kaynağı kaliteli bir kaynak olmalıdır.
Ø PIC16F877 mikrodenetleyicinin ve kullandığınız servo motorun katalog bilgilerini inceleyiniz.	Ø Çalışmalarını ve karakteristik özelliklerini inceleyiniz
Ø Servo motorun 0, +90, -90, +max ve-max açıları için gereken Ton sinyallerini deneme yanılma yaparak bulunuz.	Ø Butonları göz önüne almadan servoyu doğrudan kontrol eden basit bir programla açı sürelerini bulabilirsiniz.
Ø Devreyi kurunuz.	Ø Enerji vermeyiniz. Devrenizi önce öğretmeninize kontrol ettiriniz.
Ø Devrenin çalışması için gereken programı yazınız.	Ø Akış şemasını oluşturmanız size kolaylık sağlayacaktır.
Ø PIC16F877 entegresine programı yükleyiniz.	Ø Entegrenin bacaklarına sokete çıkarıp takarken dikkat ediniz.
Ø Programı Şekil 2.13'teki devrede çalıştırınız.	Ø Servo motorun başlangıç açısını kontrol ediniz. Motor miline düz bir çubuk takmanız açı kontrolünü kolaylaştıracaktır.
Ø Başlangıç değerindeki kontol sinyalini osilaskop ile ölçünüz. Servo motorun pozisyonunu çiziniz.	Ø Başlangıç noktasınındaki motor milini yatay eksene dik olacak pozisyona getirmeniz açıları takip etmenizi kolaylaştırır.
Ø B1 butonuna basınız. Motor milinin yeni konumundaki osilaskop görüntüsünü ve servo milinin pozisyonunu çiziniz.	Ø Motor milinin yeni pozisyonunu başlangıç noktasını referans alarak çiziniz.
Ø B2 butonuna basınız. Motor milinin yeni konumundaki osilaskop görüntüsünü ve servo milinin pozisyonunu çiziniz.	Ø Motor milinin yeni pozisyonunu başlangıç noktasını referans alarak çiziniz.
Ø RST butonuna basınız ve motor milinin durumunu gözleyiniz ve çiziniz.	Ø RST butonuna basmadan önce motorun konumunun ne olabileceğini düşününüz.
* +max saat yönünde motor milinin ulaştığı maksimum açı değeridir. – max motor milinin saat yönünün tersine ulaştığı maksimum açı değeridir.	

Şekil 2.14: Öğrenme Faaliyeti-2 uygulama işlem basamakları

ÖLÇME VE DEĞERLENDİRME

OBJEKTİF TESTLER (ÖLÇME SORULARI)

1. İstenilen bir açıya doğrudan konumlanabilen motor türü aşağıdakilerden hangisidir?
A) Step motor B) DC motor C) AC motor D)Servo motor

2. Sahip olduğu bir mil üzerinden gücünü dışarıya aktarabilen elemana ne ad verilir?
A) Kam B) Servo C) Çark D) Şaft

3. Servo motora uygulanan sinyalin Ton süresinin değişmesi sonuç olarak aşağıdakilerden hangisini doğrudan değiştirir?
A) Açı B) Hız C) İvme D) Tork

4. Servo motor iç yapısında tork değerini arttıran yapıya ne ad verilir?
A) Mıknatıs B) Mil C) Dişli kutusu D) Rotor

5. Servo motorun dönme açısı hangi eleman ile algılanır?
A) Kumanda devresi B) Geri besleme potansiyometresi
C) Motor mili D) Strator

6. Aşağıdakilerden hangisi motor milinin istenenen açı değerine ayarlar?
A) Rotor B) Geri besleme potansiyometresi
C) Motor bobini D) Kumanda devresi

7. Kontrol sinyali Ton süresi 1,25 ms ise motor milinin katedeceği açı teorik olarak aşağıdakilerden hangisidir?
A) -30 B) -45 C) +30 D) +45

8. +60 derecelik açı hareketi için Ton süresi teorik olarak ne olmalıdır?
A) 1,5 ms B) 1,6 ms C) 1,8 ms D) 2 ms

9. Periyot sabit kalmak şartıyla Ton süresinin değişmesi ile elde edilen sinyal modülasyon tekniğine ne ad verilir?
A) FM B) AM C) PCM D) PWM

10. Bir servo motoru sürmek için tasarlanan elektronik devrenin güç kısmına ne ad verilir?
A) Kumanda devresi B) Güç kaynağı C) Servo dişlisi D) Servo sürücü

DEĞERLENDİRME

Cevaplarınızı cevap anahtarlarıyla karşılaştırınız. Yanlış cevaplarınız için faaliyetin ilgili konularını tekrar ediniz.

ÖĞRENME FAALİYETİ-3

AMAÇ

Mikrodenetleyici ile PWM motor kontrolünü hatasız olarak yapabileceksiniz.

ARAŞTIRMA

- Doğru akım motor çeşitleri ve yapıları hakkında ön araştırma yapınız.
- PWM'in hangi sistemlerde kullanıldığını araştırınız.

3. PWM KONTROLÜ

3.1. PWM Teorisi

Bu faaliyette dc motor, PIC16F877'de bulunan PWM modülü ile kontrol edilecektir.

Otomasyon sistemlerinde PWM tekniğinin kullanıldığı birçok uygulama mevcuttur. Bunlardan en yaygın olanı da dc gerilim değerinin değiştirilmesi işlemidir. Dc gerilimin değeri "dc-dc konverter" adı verilen sistemler ile değiştirilebilmektedir. Bu sistemlerde dc gerilimin seviye dönüşümü için bir veya daha fazla transistör, mosfet vb. gibi yarı iletken anahtar kullanılır. Ortalama çıkış gerilimi anahtarlama elemanının Ton ve Toff zamanı ayarlanarak yapılır. Anahtarlamalı güç kaynaklarında gerilim regülasyonunu sağlamak için yapılan bu işleme "Darbe Genişlik Modülasyonu"(PWM) adı verilir. PWM sinyali üreten bir devreye de "PWM Modülatörü" adı verilir.

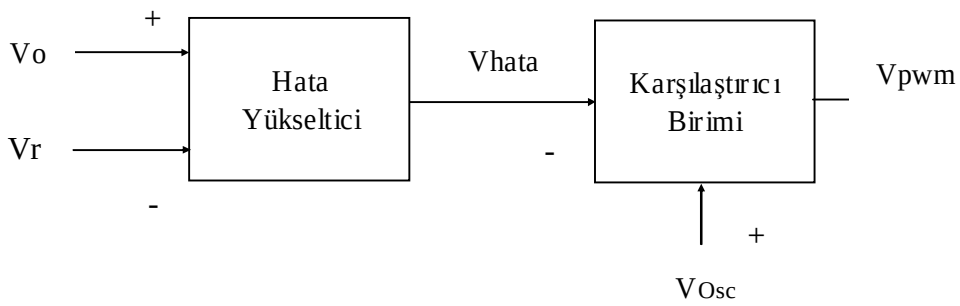
Kare dalga bir sinyalin periyotu için;

$$T=Ton+Toff$$

eşitliği kullanılır. Bu formülde "Ton" ifadesi kare dalga sinyalin lojik-1 değerinde kalma süresi, "Toff" ise kare dalga sinyalin lojik-0 değerinde kalma süresidir.

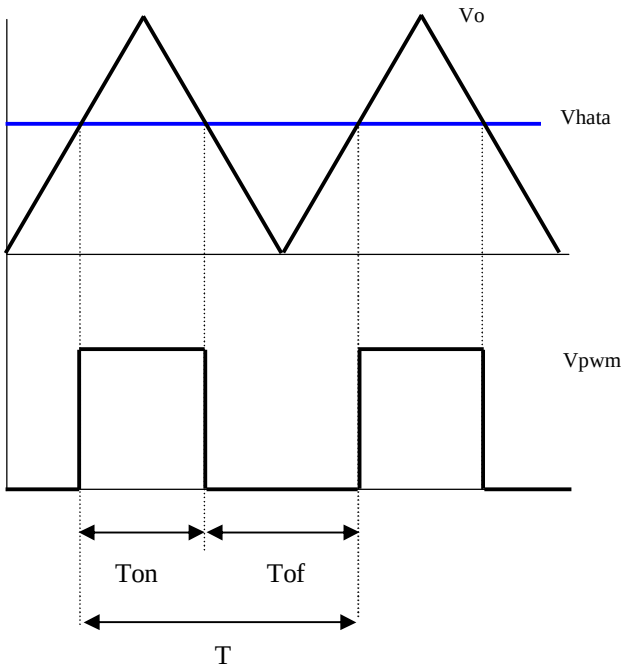
Darbe genişlik modülasyonunda çıkış gerilimi izlenir ve sabit bir gerilim ile karşılaştırılır. Şekil 3.1'de görüldüğü gibi Vo çıkış gerilimi ile Vr referans gerilimi arasındaki fark alınarak yükseltilir. Vo çıkış gerilimi, fark yükselticinin pozitif girişine uygulandığı için çıkış gerilimi arttığında hata gerilimi artacak, çıkış gerilimi azaldığında ise hata gerilimi azalacaktır. Vhata gerilimi ile bir üçgen dalga sinyali opampla yapılan karşılaştırıcı ile kıyaslanır. Karşılaştırıcı çıkışında modüleli kare dalga sinyal elde edilir. Hata gerilimi karşılaştırıcının negatif girişine verilmişse, hata gerilimi artınca darbe genişliği azalacak, hata gerilimi azalınca darbe genişliği artacaktır. Böylece gerilim regülasyonu sağlanacaktır. Bu durum şekil 3.2'deki sinyal şekillerinde görülmektedir.

Darbe genişlik modülasyonunda; $D = \frac{Ton}{T}$ oranı önemli bir büyüklüktür. Burada "D" İngilizce olarak "Duty Ratio" olarak geçer. Gerilimin belirli bir oranı çıkışa aktarıldığından "D" büyüklüğü için "aktarma oranı" kavramı kullanılabilir. Aktarma oranı PWM için çok önemli bir parametredir. Aktarma oranı kare dalga sinyalin lojik-1 zamanının (Ton) periyoda (T) oranıdır. Normalde kullanılan kare dalga sinyallerin aktarma oranı 0,5'tir. PWM tekniği ile modüle edilen kare dalganın aktarma oranı 0 ile 1 arasındadır.



Şekil 3.1: PWM Modülatörü blok şeması

Şekil 3.1'de donanımsal olarak bir pwm sinyalinin nasıl elde edilebileceği görülmektedir. Vhata geriliminin artması ya da azalması ile Vpwm sinyalinin Ton süresi değiştirilebilir. Şekil 3.2'de çıkış sinyalinin durumu görülmektedir. Sinyal şekillerinde görüldüğü gibi üçgen dalga sinyalin Vhata geriliminden büyük olduğu yerlerde çıkış pozitif, küçük olduğu yerlerde de çıkış negatifdir. Buna göre Vhata seviyesi değiştikçe Ton süresi de değişir.



Şekil 3.2: PWM Modülatörü dalga

Darbe genişlik modülasyonu iki farklı yöntem ile yapılabilir.

1. Sabit frekansta Ton süresi değiştirilerek
2. Sabit zamanda değişken frekansta çalışarak

Yukarıda anlatılan çalışma 1. yönteme örnek olarak verilebilir. Bu yöntem çok iyi bir regülasyon sağlamak bakımından dc-dc konverterlerde tercih edilir. Çünkü değişken frekansta regülasyon yapmak çok zordur. Daha karmaşık sistemler gerektirir. İkinci yöntemimiz tristörlü ve triyaklı güç kontrol sistemlerinde kullanılır. Frekansın değişken olması nedeniyle bir dc-dc konverterde dalgacık gerilimlerini yok etmek çok zordur. Her frekansta farklı filtre düzenleri gerekecektir.

Sabit frekanstaki pwm anahtarlamada üçgen dalganın tepe değeri ve frekansı sabit olmalıdır. Bu tip bir devrede çalışma frekansı üçgen dalganın frekansına eşittir. Bu frekans birkaç KHz'den birkaç MHz'e kadar olabilmektedir. Üçgen dalganın tepe değeri kaynak gerilimine bağlıdır. Devrenin tipine göre +Vcc,-Vcc ve +Vcc,0 olarak değişebilir. Bu durumu karşılaştırıcının besleme türü belirler. Simetrik kaynak kullanılıyorsa ilk durum, tek çıkışlı kaynak kullanılıyorsa ikinci durum meydana gelir.

Darbe genişlik modülasyonu dc-dc konverterlerde kullanılan en önemli işlemdir. Regülasyonun can damarıdır ve devrelerin korunması için gerekli gerilim düzenlemesini yapar. Ayrıca otomasyon sistemlerinde daha önce bahsedildiği gibi servo motor ve dc motor kontrolünde çok geniş bir şekilde kullanılmaktadır. Günümüzde darbe genişlik modülasyonu temel devre mantığı ile yapılabilir ya da entegre devre olarak hazır PWM sistemleri de bulunabilir. PWM sinyali mikrodenetleyicilerle de program kodları yardımıyla elde edilmektedir. Aşağıda dc motorun hız kontrolüne yönelik bir yaklaşımla, mikrodenetleyicili kontrol çerçevesinde PWM tekniğinden bahsedilmeye devam edilecektir.

3.2. PIC16F877'DE PWM Özelliği

2. öğrenme faaliyetinde servo motor kontrolü için PIC16F877 mikrodenetleyici ile pwm sinyali üretilmişti. Daha önceki program uygulamalarında pwm sinyali üretmek için herhangibir port pini belirli süreler ile lojik-1 ve lojik-0 yapılmıştı. Bu programlar dikkatli bir şekilde incelenirse, özellikle mikrodenetleyici ile birden fazla iş yapılması durumu göz önüne alındığında, mikrodenetleyicinin sürekli meşgul durumda olduğu ve ikinci bir iş yapıldığında pwm sinyalinin bundan etkileneceği anlaşılır. PWM üretiminin de yer aldığı kompleks sistemlerde mikrodenetleyicilerin çok fonksiyonluluk özelliğinden etkili bir şekilde yararlanılırsa, sistemin verimliliği ve güvenirliği artarken karmaşıklığı ve maliyeti azalır. Bu bağlamda, PIC16F877 mikrodenetleyici iç yapısında birçok elektronik modül vardır ve konumuzla ilgili olduğundan burada PWM modülünün kullanılmasına değinilecektir. PWM modülü sayesinde ana programın çalışmasını meşgul etmeden otomatik olarak bağımsız bir hattan PWM sinyali elde edilebilir. PIC-C dili ile PWM sinyali üretmek hazır fonksiyonlar sayesinde oldukça kolaylaştırılmıştır. Bu konuya girmeden önce PWM için kullanılan yazmaçlara göz atalım.

3.2.1. PWM için Kullanılan Yazmaçlar

PIC16F877 mikrodenetleyici ile PWM sinyali üretmek için TMR2, PR2, CCPR1, CCPR2, CCP1CON ve CCP2CON yazmaçları kullanılır. Bu kısımda ilgili yazmaçlar PWM sinyalinin üretimine yönelik olarak sunulacaktır.

3.2.1.1. CCPR1 ve CCPR2

CCPR1 ve CCPR2 yazmaçları 16 bit yakalama, 16 bit karşılaştırma ve PWM aktarma oranı belirleme görevlerini gerçekleştirir. CCPR1 ve CCPR2 yazmaçları aynı şekilde çalışırlar. Birbirinden bağımsız iki ayrı aktarma oranı bu yazmaçlarla elde edilebilir. CCPR1 ve CCPR2 yazmaçları CCPR1L, CCPR1H ve CCPR2L, CCPR2H olmak üzere iki adet 8 bitlik yazmaçtan oluşur.

Not: CCPR1(2) ifadesi CCPR1 ve CCPR2 yazmaçlarını ifade etmektedir.

3.2.1.2. CCP1CON ve CCP2CON

Bu yazmaçlar CCPR1(2) yazmaçlarını kontrol ederler. Yazmaçtaki bitler ve görevleri aşağıda verilmiştir.

7	6	5	4	3	2	1	0
-	-	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0
		R/W	R/W	R/W	R/W	R/W	R/W

Şekil 3.3: CCP1CON1 ve CCP2CON Yazmaçları

Bit 7-6: Kullanılmayan bitlerdir.

Bit 5-4: Sadece PWM için kullanılırlar. PWM aktarma saykılının en düşük iki bitidir. Geriye kalan en büyük 8 bit CCPR1(2)L yazmacında bulunur. Aktarma saykılı PWM sinyalinin Ton olduğu süredir.

Bit 3-0: CCPR1(2) mod seçim bitleridir.

0000: PWM sinyali iptal
11xx : PWM moduna geçilir.

3.2.1.3. PR2

Timer2'nin periyot değeri 8 bitlik PR2 yazmaçına yazılır. Bu yazmaç yazılabilen ve okunabilen bir yazmaçtır. Mikrodenetleyici resetlendiğinde PR2'nin aldığı değer FF'dir. Timer2 zamanlayıcısı kullanılacaksa PR2 yazmacına 00h ile FFh arasında bir değer

yazılmalıdır. Yazılan bu değere göre PWM sinyalinin periyotu ve frekansı belirlenir. Ayrıca PWM sinyalinin Ton süresinin belirlenmesinde de doğrudan etkilidir.

3.2.1.4. TMR2

Timer2 zamanlayıcısı 8 bitlik ön bölücü ve son bölücü değerlerine sahip bir modüldür. CCP modülü PWM amaçlı kullanıldığında zaman değerlerine doğrudan etki eder. Yazılabilir ve okunabilir bir yazmaçtır. Mikrodenetleyicinin resetlenmesi ile sıfırlanır. Timer2 modülünün zamanlama süresinde T2CON yazmacı etkindir. Bu yazmacın kontrolünde, giriş osilatör frekansı (Fosc/4) Timer2 için 1:1, 1:4 veya 1:16 ön bölücü seçeneklerinden biri ile işleme sokulur.

Timer2 modülü TMR2 ve daha önce bahsedilen PR2 yazmacını içerir. TMR2'de sayma ya da zamanlama değeri, PR2'de ise daha önce belirtildiği gibi zamanlama periyodunu saklanır. TMR2'nin değeri sayma(zamanlama) darbeleri ile 00h'den başlar PR2 değerine ulaşır. PR2 değerinden sonra 00h değerine döner. Zamanlama ya da sayma değeri TMR2 içeriği okunarak anlaşılabilir.

Timer2 modülünün iki çıkışı vardır. Bunlardan birincisi TMR2 çıkışı, diğeri de TMR2 kesme çıkışıdır. Kesme çıkışı için timer2 modülü 4 bitlik son bölücü değerine sahiptir. Son bölücü değeri 1:1'den 1:16'ya kadar olabilir. TMR2 kesmesi ile PIR1 yazmacının TMR2IF biti lojik-1 değerini alarak etkilenir.

Ön bölücü ve son bölücü işlemleri aslında Timer2 modülündeki bu iş için özelleştirilmiş sayıcılarla gerçekleştirilir. Bu sayıcılar aşağıdaki olaylar gerçekleştiğinde sıfırlanır.

- TMR2 yazmacına yazma
- T2CON yazmacına yazma
- POR, MCLR, WDT Reset ve BOR gibi herhangi bir resetleme işleminde

TMR2 yazmacının değeri aynı zamanda SSP modülü ile haberleşme sırasında kaydırma darbesi için kullanılır. Aşağıda ön bölücü ve son bölücü değerlerinin daha iyi kavranması için T2CON yazmacı verilmiştir. Timer2 modülü T2CON.2 biti sıfırlanarak kapatılabilir ve böylece güç harcaması azalabilir.

3.2.1.5. T2CON

7	6	5	4	3	2	1	0
-	Toutps3	Toutps2	Toutps1	Toutps0	Tmr2on	T2Ckps1	T2Ckps0
		R/W	R/W	R/W	R/W	R/W	R/W

Şekil 3.4: T2CON Yazmaçı

Bit 7: Kullanılmayan bittir.

Bit 6-3: Timer2 çıkış son bölücü seçim bitleridir.

b6 b5 b4 b3
0 0 0 0 = 1:1 son bölücü
0 0 0 1 = 1:2 son bölücü
:…:…:…:……:…
1 1 1 1 = 1:16 son bölücü

Bit 2: Timer2'nin çalışmasını başlatır.

1: Timer2 başlar.
0 : Timer2 durur.

Bit 1-0: Timer2 için osilatör ön bölücü seçim bitleridir.

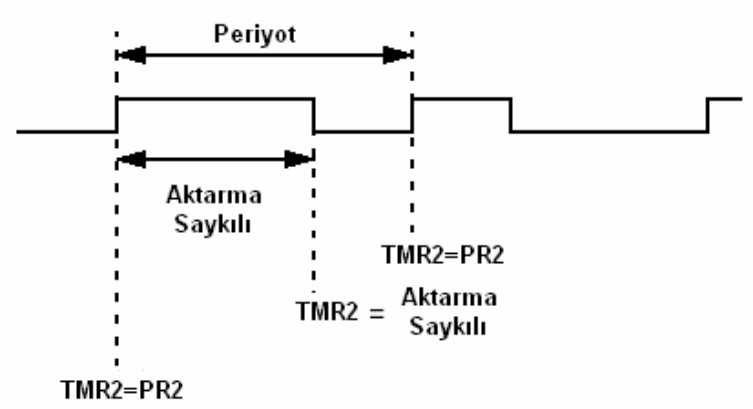
00: Ön bölücü değeri 1
01: Ön bölücü değeri 4
1x: Ön bölücü değeri 16

3.2.2. PWM Sinyalinin Donanımsal Üretimi

Darbe genişlik modülasyonunda CCP1 veya CCP2 ucundan 10 bit çözünürlükte PWM çıkışı elde edilir. CCP1 ucu aynı zamanda PortC.2 olduğundan bu ucun çıkış yapılabilmesi için TrisC.2 biti lojik-0 yapılmalıdır. Aynı koşul CCP2 için de geçerlidir.

Bir PWM çıkışı iki büyüklük ile ifade edilir. Bunlardan birincisi taban zaman (periyot,Ton+Toff) diğeri de çıkışın lojik-1 değerinde kaldığı süre olan aktarma saykılıdır (Ton). PWM frekansı periyotun tersidir (1/T).

Şekil 3.5'te PWM modundaki sinyal şekli ve TMR2 ve PR2 yazmaçlarının bu sinyaldeki görevleri görülmektedir.



Şekil 3.5: PWM Modunda zamanlama sinyalleri

3.2.2.1. PWM Sinyalinin Periyotu

PWM periyotu PR2 yazmacı ile belirlenir. Periyot aşağıdaki formül ile hesaplanır;

Tpwm = [(PR2) + 1] • 4 • Tosc • (Tmr2 ön bölücü değeri)

Fpwm=1/Tpwm

Formülü ile de PWM frekansı hesaplanabilir. Tmr2 son skala değeri Pwm frekansının belirlenmesinde kullanılmaz.

Şekil 3.4'e göre TMR2 değeri PR2 değerine eşit olduğunda, bir sonraki artım saykılında aşağıdaki olaylar meydana gelir:

- TMR2 temizlenir
- CCP1 biti lojik-1 olur. (PWM aktarma oranı %0 ise CCP1 lojik-0'da kalır)
- PWM aktarma saykılı CCPR1L'den CCPR1H'e aktarılır.

3.2.2.2. PWM Sinyalinin Aktarma Saykılı(Ton)

PWM Ton zamanı CCPR1L yazmacı ve CCP1CON yazmacının 5. ve 4. bitlerine yazarak belirlenir. 10 bit çözünürlük ile bunu gerçekleştirmek mümkündür. CCPR1L yüksek 8 biti, CCP1CON ise düşük 2 biti saklar.

Ton =(CCPR1L:CCP1CON<5:4>) • TOSC • (TMR2 ön bölücü değeri)

Akarma oranı da daha önce ifade edildiği gibi;

D=Ton/T formülü ile bulunabilir.

CCPR1L ve CCPCON(5:4) bitlerine herhangi bir zaman diliminde yazılabilir. Fakat aktarma saykılı değeri, periyot tamamlanmadıkça diğer bir deyişle PR2 ile TMR2 arasında eşleşme meydana gelmedikçe CCPR1H'ye aktarılmaz. PWM modunda CCPR1H yazmacı sadece okunabilir özelliktedir.

Maksimum PWM çözünürlüğü aşağıdaki formül ile hesaplanır.

$$\text{Çözünürlük} = \frac{\log(\frac{Fosc}{Fpwm})}{\log(2)} \text{ bit}$$

Pwm aktarma saykılı periyottan uzun olursa CCP1 pini sıfırlanmayacaktır.

3.2.2.3. PWM Çalışmasının Kurulumu

CCP modülünü PWM moduna ayarlamak için aşağıdaki işlem basamakları gerçekleştirilir.

- PR2 yazmacı ile periyot değeri ayarlanır.
- CCPR1L ve CCP1CON(5:4) bitleri ile PWM aktarma saykılı (Ton süresi) ayarlanır.
- Trisc.2 biti sıfırlanarak CCP1 pini çıkış yapılır.
- TMR2 ön bölücü değeri ayarlanır ve T2CON yazmacı ile Timer2 aktif hale getirilir.
- CCP1CON modülünde PWM çalışması için çözünürlük ayarlanır.

PWM frekansı	**1,22 kHz**	**4,88 kHz**	**19,53 kHz**	**78,12 kHz**	**156,3 kHz**	**208,3 kHz**
Tmr2 Ön bölücü (1,4,16)	16	4	1	1	1	1
PR2 değeri	0xFF	0xFF	0xFF	0x3F	0x1F	0x17
Max çözünürlük	10	10	10	8	7	5,5

Şekil 3.6: Örnek-20 MHz'lik Osilatör Frekansında PWM Frekansları ve Çözünürlükler

PIC16F877 mikrodenetleyicide PWM ve Timer2 çalışmasına etki eden diğer yazmaçlar Intcon, PIR1, PIR2, PIE1 ve PIE2'dir. Bu yazmaçlar PWM modülünün ve Timer2'nin çalışmasına etki ederler. Assembly dili ile PWM sinyali üretilecekse bu yazmaçlara da uygun bitler yazılmalıdır. Bu kısımda PIC-C hazır fonksiyonları kullanılacağından bu yazmaçlar hakkında bilgi verilmeyecektir. PIC-C fonksiyonlarının kullanımı ve bu fonksiyonlardaki hesaplamalar yukarıdaki hesaplamalarla bağlantılı bir şekilde işlenecektir.

3.2.2.4. PWM için Kullanılan PIC-C Komutları

PIC16F877 Mikrodenetleyicide PIC-C dili kullanarak PWM sinyali üretmek için 3 fonksiyonu kullanmak yeterlidir.

a. SET_PWM1_DUTY() [SET_PWM2_DUTY()]

Aktarma saykılını(Ton) kurmak için 10 bitlik değer bu fonksiyon ile belirtilir. Düşük değerli bitler gerekli değilse 8 bitlik değerde bu fonksiyon ile kullanılabilir. Sadece PIC16F877 gibi CCP/PWM modülü içeren mikrodenetleyiciler için kullanılabilir. Herhangibir sürücü veya başlık dosyası gerektirmez.

Kullanımı: set_pwm1_duty(X)
set_pwm2_duty(X)

X değeri için 8 veya 16 bit genişliğinde bir sabit veya değişken kullanılabilir. Değer 10 bitlik bir sayı ise bu sayı için kullanılan değişken veya sabit en az 16 bitlik olmalıdır. Fonksiyonun dönüş değeri yoktur. Değer 8 bitlik bir sayı ise, 10 biti elde etmek için donanım tarafından lsb bitlerinde 2 adet lojik-0 biti ile kaydırma yapılır. 10 bitlik değer bir periyot

içerisindeki PWM sinyalinin lojik-1'de kaldığı zaman miktarını saptamak için kullanılır. Ton zamanı (Aktarma saykılı) aşağıdaki formül ile bulunur.

$$Ton = X \frac{4}{Fosc} t2div$$

Fosc: Kristal veya rezonatör ile belirlenen osilatör frekansıdır.

t2div: Timer2 ön bölücü değeridir. "setup_timer2()" fonksiyonu ile belirlenir. Bu fonksiyon hakkında ilerleyen kısımlarda bilgi verilmiştir.

b. SET_CCP1() [SET_CCP2()]

CCP modülünün çalışma modunu ayarlar.

Kullanımı: setup_ccp1 (mod)
setup_ccp2 (mod)

"mod" bir sabittir. Her mikrodenetleyicinin başlık dosyasında geçerli sabitler tanımlanmıştır. Mod değeri için kullanılan sabit ifadeler PWM modu için aşağıda verilmiştir.

"CCP_OFF" : CCP modülünü pasif yapar.
"CCP_PWM": Darbe genişlik modülatörünü (PWM) aktif yapar.

PWM modunda CCP modülü PWM sinyali üretir.

c. SETUP_TIMER_2()

Timer2 yazmacını PWM sinyali için uygun değere ayarlar. Timer2 yazmacına sahip mikrodenetleyiciler için kullanılabilir.

Kullanımı: setup_timer_2 (mod, periyot, son bölücü)

Bu fonksiyonda mod, T2_DISABLED, T2_DIV_BY_1, T2_DIV_BY_4, T2_DIV_BY_16 ifadelerinden biridir. Mod osilatör saat darbesinin kaça bölüneceğini bildirir.

Periyot: Fonksiyon periyot katsayısıdır. Saat darbesi değerinin sıfırlandığı zamanı belirleyen 0-255 arasındaki int tamsayı tipinde bir değişkendir.

Son bölücü: Bir kesme sinyalinden önce zamanlayıcının kaç defa sıfırlanacağını belirleyen 1-16 arasında bir sayıdır. Örneğin bu sayı 1 olursa zamanlayıcı 1 kere, 2 olursa 2 kere resetlenir.

Aşağıda PWM sinyalinin hesaplanması ile ilgili bir örnek verilmiştir.

Örnek-1:

```
setup_ccp1(CCP_PWM);                    //CCP modülünü PWM için kurar.
setup_timer_2(T2_DIV_BY_4, 140, 0);  //periyot 140, Osc 4'e böl, son bölücü 0
set_pwm1_duty(70);        //%50 aktarma oranı, yukarıda periyot 140
                          //70 140'ın yarısı olduğu için D=%50 ayarlanır.
```

Aşağıdaki örnekte 996 Hz ve 2,475KHz frekanslarında iki ayrı ses Tonunun PWM modunda elde edildiği bir program örneği görülmektedir.

Örnek-2:

```
void ses(sec)
{
setup_ccp1(CCP_PWM);      //CCP1 modülünü PWM moduna ayarla
if(sec==2)
{
setup_timer_2(T2_DIV_BY_4, 100, 0);
set_pwm1_duty(50);                    // %50 aktarma oranı
delay_ms(1000);                       //1 saniye gecikme
                                      // PWM sinyali üretilmeye devam eder
}
else
{
setup_timer_2(T2_DIV_BY_4, 250, 0);
set_pwm1_duty(125);                   // %50 aktarma oranı
delay_ms(500);                        //500 ms gecikme
}
set_pwm1_duty(0);         //PWM sinyali kesilir. Çıkış sürekli lojik-0 olur.
}                         // Bu durumda ses Tonu elde edilemez.
```

Programda frekanslar

setup_timer_2(T2_DIV_BY_4, 100, 0);

fonksiyonu ile belirlenmiştir.

Aktarma oranı da aşağıdaki formül ile belirlenmiştir.

set_pwm1_duty(50);

Dikkat edilirse *set_pwm1_duty(50);* fonksiyonundaki 50 değeri s*etup_timer_2(T2_DIV_BY_4, 100, 0);* fonksiyonundaki 100 değerinin yarısıdır. Bu nedenle D oranı %50'dir.

Programda y değerine göre bir ses Tonu belirli bir süre üretilmiş ve sonra ses kesilmiştir. Ses kesilmiş olmasına rağmen 0 aktarma oranlı PWM sinyalinin üretilmesine devam edilir.

Aşağıdaki örnek PWM çalışıyorken ana programda bir ledin kontrol edildiği bir programdır. PWM çalışması PIC'i meşgul etmez. Ana programda PWM dışında başka çalışmalar gerçekleştirilebilir.

Örnek-3

```
////                          Örnek-3                                    ////
////  Programda aktarma oranı program içinde belirleniyor.               ////

#include <16F877.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)
#define in input(PIN_D0)
#define out PIN_B0
byte port_b=6,port_d=8;

io_set()

 {
  set_tris_b(0x00);
  set_tris_d(0xff);
 }

pwmkur()

 {
  setup_ccp1(CCP_PWM);                  // CCP1 modülü PWM moduna kuruluyor.
  setup_timer_2(T2_DIV_BY_16, 100, 1);  //Periyot, mod, son bölücü ayarlanıyor.
 }

pwm()
 {
   set_pwm1_duty(50);                   // %50 Aktarma
 }

main()
{
  io_set();
  pwmkur();                             // PWM kuruluyor.
  pwm();                                // CCP1 pininden PWM sinyali alınmaya başlar.

  while(1)
    {
```

```
      if(in==0) output_high(out);          //PD0'daki buTona basılırsa PB0'daki led ışık verir.
      else output_low(out);                //PWM çıkışı bu çalışmadan etkilenmez.
   }
}
```

PWM komutları aşağıda belirtildiği gibi çalışır.

1. setup_ccp1(CCP_PWM);

Bu komut ile CCP1 modülü PWM moduna ayarlanır.

2. setup_timer_2(T2_DIV_BY_16, 100, 1);

Bu komut satırında periyot 100, mod ise T2_DIV_BY_16'dır. 1 değeri son bölücü değeridir ve kesme sinyalinden önce timer2 1 kere sıfırlanır.

Bu fonksiyona göre PWM periyotu ve frekansı aşağıdaki formül ile hesaplanır.

T=(1/clock)*4*t2div*(period+1)

Programda kristal frekansı 20 MHz ve fonksiyon periyot katsayısı=100'dür. Buna göre;

Periyot (T);

(1/20000000)*4*16*101= 323,2 us olarak bulunur. Frekans ise;

F=1/T=1/323,2us= 3,09 KHz bulunur.

3. set_pwm1_duty(50);

%50 aktarma oranlı PWM sinyalinin Ton süresini ayarlar. Bu süre;

$$Ton = X\frac{4}{Fosc}t2div$$

formülü ile hesaplanır. Fonksiyonda X=50 ve t2div=16'dır. Buna göre;

Ton=50x4x(1/20000000)X16=160 us bulunur.

$$\text{PWM Çözünürlüğü}=\frac{\log(\frac{Fosc}{Fpwm})}{\log(2)}\text{ bit'dir.}$$

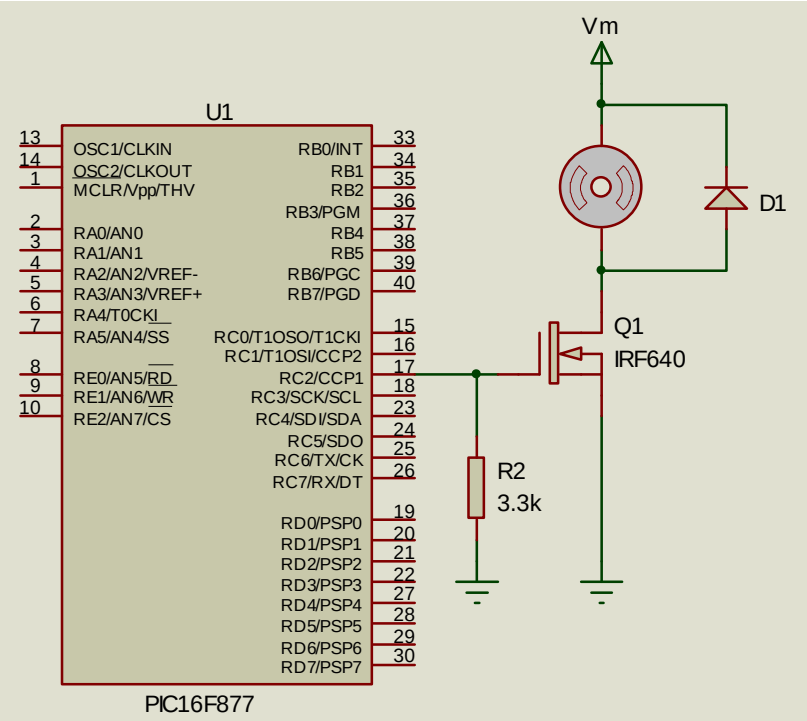
Buna göre;

Pwm çözünürlüğünü “R” ile ifade edersek

$$R = \frac{\log(\frac{20x10^{6}}{3{,}09x10^{3}})}{\log(2)} = \frac{\log(6{,}47x10^{3})}{0{,}3} = \frac{3{,}81}{0{,}3} = 12{,}7 \text{ bit bulunur.}$$

3.3. PIC16F877 ile Dc Motor Kontrolü

Daha önceki mikrodenetleyici modüllerinde doğru akım motorunun kontrolü ile ilgili bilgiler verilmişti. Bu kısımda aynı bilgiler tekrar verilmeyecek, sadece PIC16F877 PWM modülü ve PIC-C dili kullanılarak bir doğru akım motorunun hızının nasıl değiştirileceği işlenecektir.



Şekil 3.7: PIC16F877 ile DC Motor sürücü devresi bağlantısı

Şekil 3.7’de dc motor sürücü devresi ile PIC16F877 mikrodenetleyicinin bağlantısı görülmektedir. Mikrodenetleyici programı ile üretilen PWM sinyali RC2/CCP1 ucundan alınarak N-Kanal Mosfetin kapı girişine uygulanır. Mosfet PWM sinyalinin Ton sürelerinde iletimde Toff sürelerinde ise yalıtımdadır. Mosfetin iletimde olduğu sürelerde Motor içerisinden bir akım geçer. Motorun dönme hareketi sırasındaki hızı ve devir sayısı Vm gerilimi ve aktarma oranına bağlıdır. Bu bağlamda motor için ortalama gerilim değerinden

söz edilebilir. Mosfet sadece Ton zamanlarında iletimde olduğundan, mosfetin akaç-kaynak arası gerilimi ihmal edilirse oluşacak ortalama gerilim değeri aşağıdaki formülle bulunabilir;

$$Vort = DxVm$$

Bu formülde D aktarma oranı, Vm ise motora uygulanan kaynak gerilimidir.

Örnek-4:

PIC16F877 mikrodenetleyici tarafından üretilen PWM sinyalinin Ton süresi 2 ms, Toff süresi ise 8 ms'dir. Motor kaynak gerilimi 12V ise motoru hareket ettiren ortalama gerilimi bulunuz.

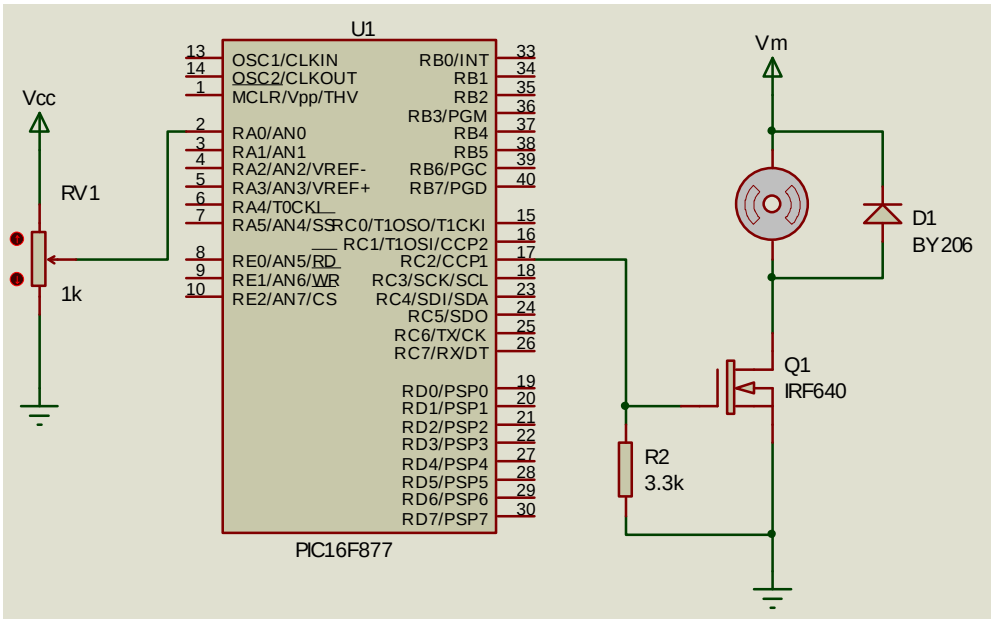
$Vort = DxVm$ formülüne göre;

$$Vort = \frac{2ms}{10ms} x12V = \frac{12V}{5} = 2,4V \text{ bulunur.}$$

Motor 12V'luk bir motor ise muhtemelen bu PWM sinyali ile dönme hareketi yapamayacaktır. Ton süresi yavaş yavaş arttırıldığında motora uygulanan ortalama gerilim de artacak ve motor hareket etmeye başlayacaktır.

Örnek-3'deki program Şekil 3.7'deki devrede DC motoru hareket ettirmek için kullanılabilir. Bu örnekte PWM sinyali aktarma oranı %50 olduğu için ortalama motor gerilimi de Vort=0,5x12V=6 V bulunur.

Şekil 3.8'de PIC16F877 analog dijital çevirici modülü girişine bağlanan bir potansiyometre ile analog bir değer girilmiş ve bu değer dijitale çevrilmiştir. Analog değerin dijital karşılığı da PWM sinyalinin Ton süresini belirlemek için kullanılmıştır. Böylece potansiyometreyi çevirerek DC motorun hız kontrolü yapılmış olur.



Şekil 3.8: Analog gerilim ile DC motorun PWM kontrolü

Örnek-5'te Analog değere göre PWM sinyali oluşturan program görülmektedir.

Örnek-5:

```
//////////////////////////////////////////////////////////////////////
////                  ÖRNEK-5 PROGRAMI            ////
////  Program Analog girişi okur ve bunun dijital değerini aktarma oranı için kullanır.
////  Pwm çalışıyorken  analog giriş sürekli okunur. Analog değer değiştikçe Ton süresi
değişir.

#include <16F877.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)

void main()
{ byte X;
  setup_ccp1(CCP_PWM);                    // CCP1 PWM moduna ayarlanıyor
  setup_timer_2(T2_DIV_BY_16, 127, 1);   //    Periyot (1/clock)*4*t2div*(period+1)
  setup_port_a(ALL_ANALOG);              //    Fosc=20000000 ve periyot katsayısı=127
  setup_adc(adc_clock_internal);         //     (1/20000000)*4*16*128= 409,6 us or 2,44 khz
  set_adc_channel( 0 );

  while( TRUE )
  {
    X=read_adc();
    set_pwm1_duty(X);       //    X*4*(1/clock)*t2div
  }

}
```

"setup_port_a(ALL_ANALOG)": PortA Analog giriş moduna sokuluyor.

"setup_adc(adc_clock_internal)": A/D çevrim için osilatör frekansı kullanılacak anlamındadır.

"set_adc_channel(0)": PortA.0 pini analog giriş için kullanılacak

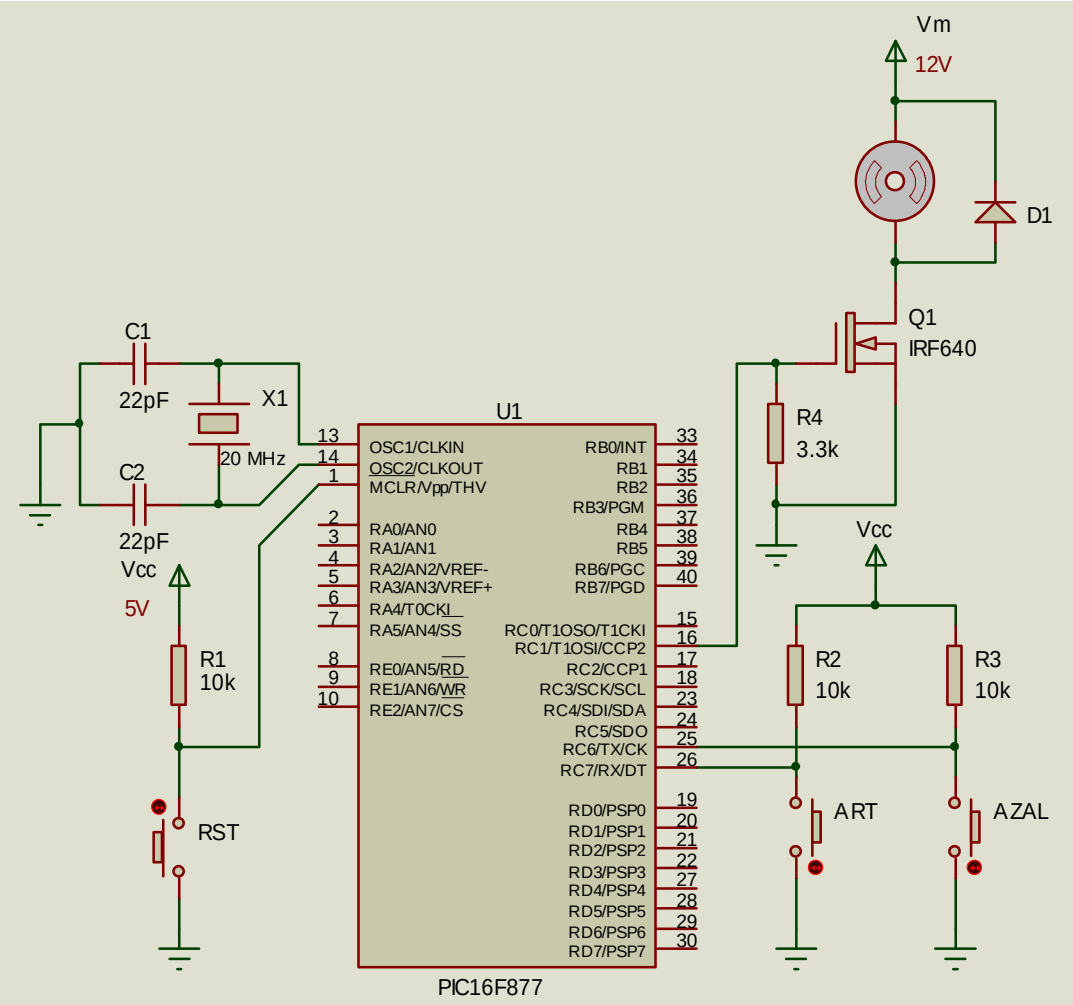
"X=read_adc()": Analog değer dijitale çevrilir. Bu fonksiyon üzerinden dijital değer döner ve bu değer X değişkenine atanır.

Program çalıştığında PA.0'daki potansiyometrenin değerinin değiştirilmesi ile PWM sinyalinin Ton süresi değişir ve motorun hızı artırılıp azaltılabilir.

UYGULAMA FAALİYETİ

Kullanılan Malzeme ve Araç-Gereç

1. 1 adet PIC16F877 ve Entegre Soketi
2. 1 adet DC Motor (Çalışma gerilimi 12V)
3. 1 adet 20 MHz kristal
4. 1 adet IRF640 Mosfet
5. 1 adet BY206 Diyot
6. 2 adet 22pF kondansatör
7. 3x10K ve 1 x3.3K Ohm dirençler
8. 3 adet buTon
9. PIC programlayıcı devresi
10. 5V DC gerilim kaynağı



Şekil 3.9: Öğrenme Faaliyeti-3 uygulama devresi

Devrenin Çalışması:

Devreye ilk enerji verildiğinde PWM sinyalinin frekansı 10 KHz ve aktarma oranı %50'dir. Motor bu sinyale göre hareket edecektir. ART buTonuna bastıkça set_pwm2_duty(Ton) formülündeki Ton katsayısı 5'er 5'er artacak, AZAL buTonuna basınca Ton katsayısı 5'er 5'er azalacaktır. Aktarma oranı %0 ile %100 arasında değişecektir. Çalışma sonsuz döngü ile devam edecektir.

İşlem Basamakları	Öneriler
Ø Şekil 3.9'daki devre için gereken malzemeleri ve araç-gereci hazırlayınız.	Ø Gerilim kaynakları kaliteli olmalıdır.
Ø PIC16F877 mikrodenetleyicinin, Mosfetin ve kullandığınız motorun katalog bilgilerini inceleyiniz.	Ø Çalışmalarını ve karakteristik özelliklerini inceleyiniz
Ø PWM sinyali elde etmek için CCP2 modülünü nasıl ayarlamanız gerektiğini düşününüz.	Ø CCP1 modülünün kullanıldığı örnekleri inceleyiniz.
Ø 10 KHz frekansını elde etmek için gereken t2div ve periyot değerlerini bulunuz.	Ø Daha önce anlatılan formüllerden faydalanınız.
Ø Şekil 3.11'deki tabloda gereken değerleri hesaplayınız.	Ø D oranı %0'dan %100 olana kadar gereken değerleri bulmalısınız.
Ø Devreyi kurunuz.	Ø Enerji vermeyiniz. Devrenizi önce öğretmeninize kontrol ettiriniz.
Ø Devrenin çalışması için gereken programı yazınız.	Ø Akış şemasını oluşturmanız size kolaylık sağlayacaktır.
Ø PIC16F877 entegresine programı yükleyiniz.	Ø Entegrenin bacaklarına sokete çıkarıp takarken dikkat ediniz.
Ø Programı Şekil 3.9'daki devrede çalıştırınız.	Ø DC motorun hızına dikkat ediniz.
Ø Başlangıç değerindeki PWM sinyalini osilaskop ile ölçünüz.	Ø Ton, Toff ilişkisini izleyiniz.
Ø ART buTonuna ard arda basınız ve motorun çalışmasını gözleyiniz.	Ø BuTonlarda titreşim (chattering) kontrolü yapmalısınız.
Ø DC motor hızı maksimum değerine ulaştığında osilaskop ile PWM sinyalini ölçünüz.	Ø ART buTonunun etki etmediği hızda ölçüm yapınız. PD.0'a bir led bağlayarak %100 D oranını anlayabilirsiniz.
Ø AZAL buTonuna ard arda basınız ve motorun çalışmasını gözleyiniz.	Ø BuTonlarda titreşim (chattering) kontrolü yapmalısınız.
Ø Motor durduğunda oluşan PWM sinyalini ölçünüz.	Ø AZAL buTonunun etki etmediği an olan motorun hareketsiz kaldığı durumda ölçüm yapınız. PD.1'e bir led bağlayarak %0 D oranını anlayabilirsiniz.

Şekil 3.10: Öğrenme Faaliyeti-3 uygulama işlem basamakları

Fosc	Div	Periyot Katsayısı	Ton Katsayısı	T (us)	F (KHz)	D	Ton(us)
20 MHZ			0		10		
20 MHZ			5		10		
20 MHZ			10		10		
20 MHZ			15		10		
20 MHZ			20		10		
20 MHZ			25		10		
20 MHZ			30		10		

Şekil 3.11: Öğrenme Faaliyeti-3 uygulama sonuç tablosu

Osiloskop Görüntüleri

Hesaplamalar

1. Başlangıç Durumu (D=%50)

2. Maksimum Hız Durumu (D=%100)

3. Minimum Hız Durumu (D=%0)

ÖLÇME VE DEĞERLENDİRME

OBJEKTİF TESTLER (ÖLÇME SORULARI)

1. Bir kare dalganın periyotu sabit kalmak şartıyla Ton süresinin değiştirilmesi ile tanımlanan modülasyon şekli aşağıdakilerden hangisidir?
A) FM B) GM C) PWM D) PM

2. Aşağıdaki motorlardan hangisinin hızı PWM ile kontrol edilebilir?
A) Step motor B) DC motor C) AC motor D)Servo motor

3. Bir kare dalga sinyalin Ton süresi 2 ms, Toff süresi de 8 ms ise bu sinyalin aktarma oranı aşağıdakilerden hangisidir?
A) 0,1 B) 0,2 C) 0,25 D) 0,4

4. Mikrodenetleyici PWM modülü kullanılmadan ana programda yazılan komutlarla PWM sinyali elde etme işlemindeki en büyük sakınca nedir?
A) Hızın azalması B) Programın karmaşıklaşması
C) Hafızada daha fazla yer kaplaması D) Programı meşgul etmesi

5. Aşağıdaki modüllerden hangisi ile PWM sinyali üretilir?
A) TMR1 B) CCP1 C) TRISC D) MSSP

6. PIC16F877'de PWM sinyalini elde etmek için aşağıdaki birimlerden hangisinin doğrudan ya da dolaylı etkisi yoktur?
A) USART B) CCPR2 C) TMR2 D) PR2

7. Timer2 periyot katsaysı değeri aşağıdaki yazmaçlardan hangisinde saklanır?
A) PIR1 B) PIR2 C) TMR1 D) PR2

8. 16F877'de üretilen PWM sinyali maksimum kaç bit olabilir?
A) 1 B) 5 C) 10 D) 20

9. PWM modülünü aktif yapmak için aşağıdaki komutlardan hangisi kullanılır?
A) setup_ccp1(CCP_PWM) B) setup_ccp1(PWM_ON)
C) set_pwm1_on D) setup_ccp1_PWM

10. Aşağıdaki komutlardan hangisi aktarma oranını 0 yapar?
A) set_pwm_duty(0) B) set_pwm1_duty(0)
C) set_pwm_0 D) Pwm_Ton(0)

DEĞERLENDİRME

Cevaplarınızı cevap anahtarlarıyla karşılaştırınız. Yanlış cevaplarınız için faaliyetin ilgili konularını tekrar ediniz.

MODÜL DEĞERLENDİRME

A. OBJEKTİF TESTLER (ÖLÇME SORULARI)

1. PIC16F877 mikrodenetleyicinin veri belleği kaç B'dır?
A) 8 B) 38 C) 68 D) 368

2. PIC16F877 mikrodenetleyicide PWM sinyali hangi porttan veya portlardan elde edilebilir.?
A) PortA B) PortB C) PortC D) Hepsi

3. PIC16F877'de kesme kontrolü için hangi yazmaç kullanılır?
A) Intcon B) PR2 C) PIR2 D) Status

4. LCD'de ekrana yazılacak olan veri aşağıdaki birimlerden hangisi içerisinde kayılıdır?
A) CGRAM B) DDRAM C) CPU D) CGROM

5. LCD ekranında istenilen koordinata bir karakter veya karakter dizisini yazan komut aşağıdakilerden hangisidir?
A) Lcd_putc() B) Lcd_kur() C) Lcd_init() D) Dsply_yaz()

6. Aşağıdaki motor türlerinden hangisinde açı geri beslemesi bir potansiyometre ile yapılır?
A) Step motor B) DC motor C) AC motor D) Servo motor

7. Aşağıdakilerden hangisi servomotorda tork değerini artırır?
A) Kam B) Potansiyometre C) Çark Grubu D) Mil

8. Şekil 3.9'daki devreye göre bir dc motor kontrol sinyalinde Ton süresi 1,25 ms, Periyot 5 ms ve motor besleme gerilimi 24V ise, bu koşullar altındaki motor uçları arasındaki gerilim kaç volttur?
A) 3 V B) 5 V C) 6 V D) 8 V

9. Aşağıdaki modüllerden hangisinin PWM sinyali üzerinde bir etkisi yoktur?
A) Timer1 B) CCP1 C) TRISC D) Timer2

10. PWM modülünde periyot katsayısı değerini 50, t2div değerini 4 ve son bölücü değerini 1 yapan komut aşağıdakilerden hangisidir?
A) setup_timer_2(T2_DIV_BY_50,1,4)
B) setup_timer_2(T2_DIV_BY_4,1,50)
C) setup_timer_2(T2_DIV_BY_4,50,1)
D) setup_timer_2(T2_DIV_BY_1,4,50)

B. UYGULAMALI TEST

Modül sonu yaptığınız uygulamayı aşağıdaki kotrol listesini değerlendirerek kendinizi ölçünüz.

AÇIKLAMA: Aşağıda listelenen ölçütleri uyguladıysanız EVET sütununa, uygulamadıysanız HAYIR sütununa X işareti yazınız.		
Değerlendirme Ölçütleri	**DEĞERLENDİRME**	
	Evet	**Hayır**
Ø PIC16F877 mikrodenetleyiciyi istenen sisteme uygun olarak kullanabildin mi?		
Ø LCD-Mikrodenetleyici bağlantısını doğru olarak yapabildin mi?		
Ø LCD'yi istenen amaca uygun olarak sürücü kullanmadan programlayabildin mi?		
Ø LCD'yi istenen amaca uygun olarak sürücü ile programlayabildin mi?		
Ø LCD sürücüsü yazabildin mi?		
Ø PIC16F877-Servo motor bağlantısını sürücü devresi ile yapabildin mi?		
Ø Servo motoru istenen açı değerine getirebildin mi?		
Ø Sürücü kullanmadan PWM sinyali elde edebildin mi?		
Ø PIC16F877'de PWM modülünü kullanabildin mi?		
Ø DC motor kontrolünü PWM modülü ile gerçekleştirebildin mi?		

DEĞERLENDİRME

Hayır cevaplarınız var ise ilgili uygulama faaliyetini tekrar ediniz. Cevaplarınızın tümü evet ise bir sonraki modüle geçebilirsiniz.

CEVAP ANAHTARLARI

1. Öğrenme Faaliyeti -1 Cevap Anahtarları

1.1. Uygulama Faaliyeti -1 Cevap Anahtarı

a. LCD2.C sürücüsü kullanarak yazılması gereken örnek program

```
#include <16F877.h>
#use delay(clock=20000000)
#include <LCD2.C>
#fuses hs,NOWDT,NOPROTECT,NOLVP

void main()
{
  lcd_init();
  lcd_putc("\fLCD PROGRAMLAMA");
}
```

b. LCD2.C sürücüsü kullanmadan olması gereken örnek program

```
// Lcd'ye LCD PROGRAMLAMA yazan program
//PB0-X / PB1-RS / PB2-RW / PB3-E / PB4...PB7 - 4 bit data

#include <16F877.h>
#include<string.h>
#use delay(clock=20000000)
#fuses hs,NOWDT,NOPROTECT,NOLVP
#byte port_b=6              // port_b nin adresi
#byte port_d=8
#define RS PIN_B1
#define RW PIN_B2
#define E PIN_B3
#define PB4 PIN_B4
#define PB5 PIN_B5
#define PB6 PIN_B6
#define PB7 PIN_B7
char str[20]="LCD PROGRAMLAMA";
char i,j,gec;

io_setout()
   {
      set_tris_b(0x00);              // port_b = çıkış
   }
```

```
komutyaz()
   {
   Output_Low(RS);        //RS=0 Komut geliyor.
   Output_Low(RW);        //R/W=0 Yazma modu
    }

veriyaz()
   {Output_Low(E);
    Output_High(RS);       //RS=1 veri geliyor.
    Output_Low(RW);        //R/W=0 Yazma modu
  }

uyg()
  { Output_High(E);
    delay_ms(1);
    Output_Low(E);
    delay_ms(1);
   }

clrdsply()
  {komutyaz();
    port_b=0x00;
    uyg();
    port_b=0x10;         //01H ekran temizleme komutu gönderildi
    uyg();
   }

dsplay_on()
  {komutyaz();
   port_b=0x00;
    uyg();
   port_b=0xC0;       //0CH Display açma komutu gönderiliyor.
    uyg();
   }

lcdreset()
  { komutyaz();
     port_b=0x20;    // 2 değeri porb7-4'e gönderiliyor.
     uyg();
     port_b=0x20;
     uyg();
     port_b=0x20;
     uyg();
     }

lcdkur()
```

```
  { komutyaz();
    port_b=0x20;  // 2 değeri porb7-4'e gönderiliyor.
    uyg();
    port_b=0x00;  // 20h 4 bit modu 1 satır 5*7 fontu kuruluyor.
    uyg();
   }

adreskur()
   { komutyaz();
    port_b=0x80;
    uyg();                    //DDRAM adresi 00h olarak kuruluyor. 1.satır 1.sütun
    port_b=0x00;
    uyg();
   }

verigonder(char ch)
   { veriyaz();
    gec=ch;
    gec=gec&0xf0;
    gec=gec|0x02;
    port_b=gec;
    uyg();

    #ASM
    swapf ch,1
    #ENDASM

    gec=ch;
    gec=gec&0xF0;
    gec=gec|0x02;
    port_b=gec;
    uyg();
   }

giriskur()
   {komutyaz();
    port_b=0x00;
    uyg();
    port_b=0x60;
    uyg();          //06H PortB'den gönderiliyor. Giriş modu sağa hareket
   }

void main()
  { io_setout();
    lcdreset();
    clrdsply();
    lcdkur();
```

```
     dsplay_on();
     giriskur();
     adreskur();
     j=strlen(str);
    for(i=0;i<j;i++) verigonder(str[i]);

 }
```

c. Yazının 2. satırda görünmesi için;

```
adreskur()
    { komutyaz();
      port_b=0xC0;
      uyg();                    //DDRAM adresi C0h olarak kuruluyor. 2.satır 1.sütun
      port_b=0x00;
      uyg();
     }
```

yapılmalıdır.

d. Yazının 1 sn aralıklarla sürekli olarak 1. ve 2. satırda görünmesi için b şıkkında yazılan programda aşağıdaki değişiklikler yapılmalıdır.

```
---
---
lcdkur()
  { komutyaz();
     port_b=0x20;  // 2 değeri porb7-4'e gönderiliyor.
     uyg();
     port_b=0x80;  // 28h 4 bit modu 2 satır 5*7 fontu kuruluyor.
     uyg();
     }
---
---
adreskur(int c)
    { komutyaz();
      port_b=c;
      uyg();              //DDRAM adresi c değişkenine göre belirleniyor
     port_b=0x00;
      uyg();
     }
---
---
void main()
  {io_setout();
    lcdreset();
    clrdsply();
    lcdkur();
```

```
    dsplay_on();
    giriskur();

    while(1)
    {
    adreskur(0x80);                    //1. satırın adresi gönderiliyor
    j=strlen(str);
    for(i=0;i<j;i++) verigonder(str[i]);

    delay_ms(1000);
    clrdsply();

    adreskur(0xC0);                    //2.satırın adresi gönderiliyor
    j=strlen(str);
    for(i=0;i<j;i++) verigonder(str[i]);

    delay_ms(1000);
    clrdsply();
    }
}
```

1.2. Ölçme Soruları Cevap Anahtarı

SORU	CEVAP	SORU	CEVAP
1	a	**6**	c
2	c	**7**	d
3	c	**8**	d
4	c	**9**	c
5	d	**10**	b

2. Öğrenme Faaliyeti -2 Cevap Anahtarları

2.1. Uygulama Faaliyeti -1 Cevap Anahtarı

1.

Servo motor katalog bilgileri.

S3003 Standard		
FUTM0031		
Volts	**Torque**	**Speed**
4.8V	44.4 oz-in.	0.23 sec/60°
6.0V	56.9 oz-in.	0.16 sec/60°
Dimensions		**Weight**
1.6 x 0.8 x 1.4 in.		1.3 oz.

2. Servo motorun 0, +90, -90, +max ve-max açıları için gereken Ton sinyalleri:

Uygulama çalışmasında piyasada bulunabilen Futaba S3003 servo motoru kullanılmıştır. Uygulama devresinde servoya doğrudan kontrol sinyalleri uygulayarak yapılan denemelere göre;

0 derece için 1,29 ms
+90 derece için 0,31 ms
-90 derece için 2,25 ms
+ max için 0,2 ms
- max için 2,33 ms bulunmuştur.

Servo motor milinin hareket aralığı 180 dereceden büyükür. Tahminen 200 derece civardır.

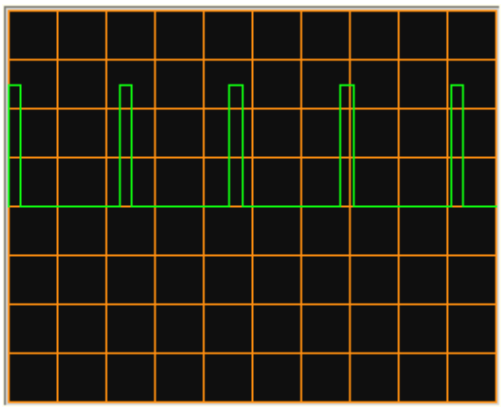
3. Servo motor kontrol programı aşağıda verilmiştir.

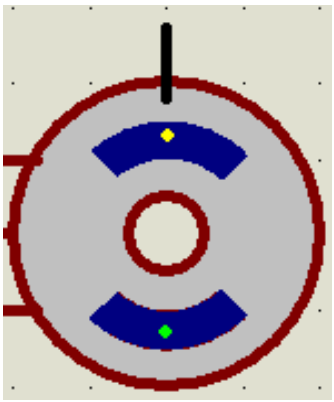
```
//SERVO MOTOR KONTROL PROGRAMI-Uygulama faaliyeti
//PD0: Servo kontrol sinyali için kullanılacak
//B1 ve B2 konum kontrolü için kullanılacak
//B1 -90'a B2 +90'a hareket sağlayacak.

#include <16F877.h>
#use delay(clock=20000000)
#fuses hs,NOWDT,NOPROTECT,NOLVP
#byte port_d=8                    //portd'nin adresi
#define CNT PIN_D0
#define B1 input(PIN_A0)
#define B2 input(PIN_A1)

io_setout()
    {
   set_tris_d(0x00);                 // port_d = çıkış
    }

init()
 {while(1)
   {
    Output_High(CNT);          //Servomotor mili başlangıç pozisyonunda / 0 derece
    delay_us(1290);
    Output_Low(CNT);
    delay_us(10000);
     if((B1==0)|(B2==0)) break;
   }
 }

move1()
 {while(1)
   {
    Output_High(CNT);           //+90 derecede
    delay_us(310);
    Output_Low(CNT);
    delay_us(10000);
    if(B1==0) break;
    }
 }

move2()
 {while(1)
   {
    Output_High(CNT);            //- 90 derece
    delay_us(2250);
```

```
    Output_Low(CNT);
    delay_us(10000);
    if(B2==0) break;
    }
  }

void main()
 {
    io_setout();
    init();
dongu:  move1();
    move2();
    goto dongu;
 }
```

3.

Başlangıç pozisyonundaki kontrol sinyali ve motor durumu





Başlangıç değerindeki Kontrol sinyali ve Servo Mil Durumu

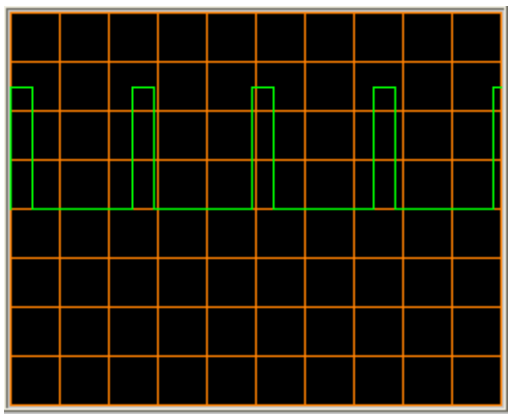
Motor mili konumu 0 derece / Volt/div=2V Time/div=5ms

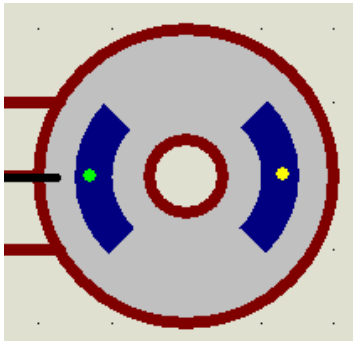
Şekil: Öğrenme Faaliyeti-2 Uygulama Sinyali-1

Ton=1,29 ms // Toff=10 ms

4.

B1 buTonuna basıldığında elde edilen kontrol sinyali ve motor durumu





B1 BuTonuna Basıldığında Oluşan Kontrol Sinyali ve Servo Mil Durumu

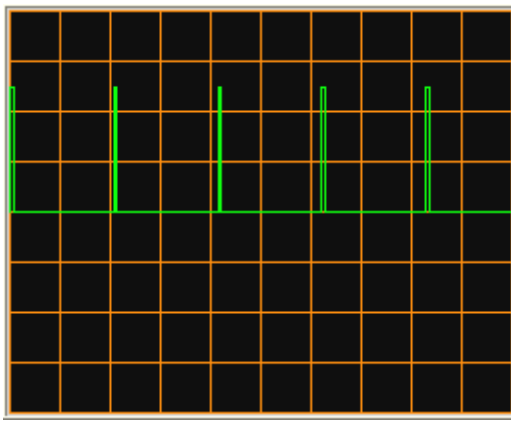
Motor mili konumu -90 derece / Volt/div=2V Time/div=5ms

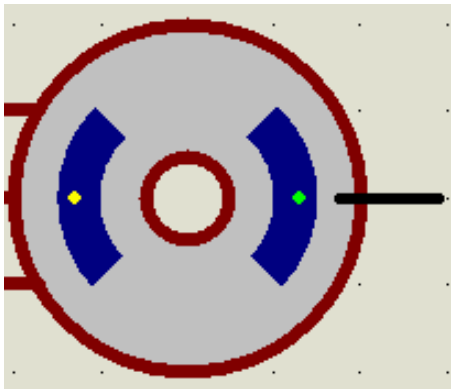
Şekil: Öğrenme Faaliyeti-2 Uygulama Sinyali-2

Ton=2,25 ms // Toff=10 ms

5.

B2 buTonuna basıldığında elde edilen kontrol sinyali ve motor durumu





B2 BuTonuna Basıldığında Oluşan Kontrol Sinyali ve Servo Mil

Motor mili konumu +90 derece / Volt/div=2V Time/div=5ms

Şekil: Öğrenme Faaliyeti-2 Uygulama Sinyali-3

Ton=0,31 ms // Toff=10 ms

6.

RST buTonuna basıldığında mikrodenetleyici resetlenir ve motor mili başlangıç konumuna geri döner. 3 nolu cevap anahtarında ilgili şekiller görülmektedir.

2.2. Ölçme Soruları Cevap Anahtarı

SORU	CEVAP	SORU	CEVAP
1	d	**6**	d
2	b	**7**	b
3	a	**8**	c
4	c	**9**	d
5	b	**10**	d

3. Öğrenme Faaliyeti -3 Cevap Anahtarları

3.1. Uygulama Faaliyeti -1 Cevap Anahtarı

1. Mosfet N-Kanallı çalışma frekansı yüksek bir mosfetttir. Motor çalışma gerilimi 12 V'dur.

2.CCP2 modülü tıpkı CCP1 modülü gibi kullanılır. Aşağıdaki komutlarda 1 yerine 2 rakamı yazılması yeterlidir.

```
setup_ccp2(CCP_PWM);
set_pwm2_duty(Ton);
```

3. Periyot aşağıdaki gibi bulunur.

(1/Fosc)*4*t2div*(periyot katsayısı+1) formülüne göre;

Fosc=20000000 ve periyot katsayısı=30 alındığında

(1/20000000)*4*16*31= 99,2 us bulunur.

F=(1/T) formülünden de F=10,08 khz

setup_timer_2(T2_DIV_BY_16,30,1);

komutunda t2div=16 periyot katsayısı=30'dur.

4. Tablo değerleri aşağıda verilmiştir.

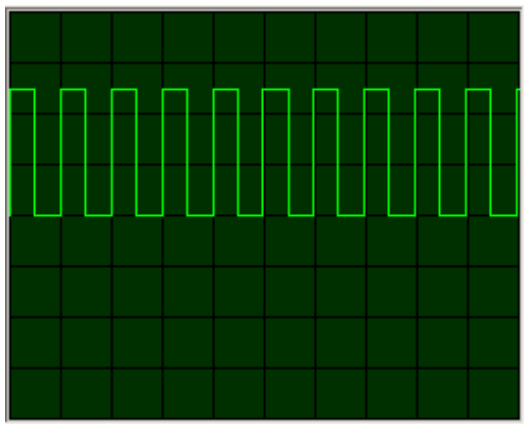
Fosc	Div	Periyot Katsayısı	Ton Katsayısı	T (us)	F (KHz)	D	Ton(us)
20 MHz	16	30	0	99,2	10,08065	0,000	0,00
20 MHz	16	30	5	99,2	10,08065	0,167	16,53
20 MHz	16	30	10	99,2	10,08065	0,333	33,07
20 MHz	16	30	15	99,2	10,08065	0,500	49,60
20 MHz	16	30	20	99,2	10,08065	0,667	66,13
20 MHz	16	30	25	99,2	10,08065	0,833	82,67
20 MHz	16	30	30	99,2	10,08065	1,000	99,20

5. Devrenin istenen şekilde çalışması için gereken program aşağıda verilmiştir.

```
////////////////////////////////////////////////////////////////////////
////              PWM Uygulama programı                ////
//// Programın başında PWM aktarma oranı %50 ////
////  ART buTonuna basıldığında PWM oranı %5'er artacak, AZAL buTonuna basıldığında PWM oranı %5'er azalacak
//// Pwm çalışması %0 ile %100 arasında değişecek

#include <16F877.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)
#define art input(PIN_C7)
#define azal input(PIN_C6)
byte port_c=7;
int Ton,T;

io_set()
 {
   set_tris_c(0xF0);      //7. ve 6. bitler giriş, 1. bit çıkış
  }

pwmkur()
 {
   setup_ccp2(CCP_PWM);   // CCP2'yi PWM moduna ayarla
   setup_timer_2(T2_DIV_BY_16,30,1);
  }

up()
 {
  delay_ms(10);
   while(1)
   {
     if(art==1){Ton=Ton+5;break;}          //BuTon titreşiminin geçmesi bekleniyor.
```

```
      else continue;
     }
    delay_ms(10);
 }

dwn()
{ delay_ms(10);
   while(1)
    {
      if(azal==1){Ton=Ton-5;break;}        //BuTon titreşiminin geçmesi bekleniyor
      else continue;
    }
    delay_ms(10);
}

pwm()
 {
    set_pwm2_duty(Ton);           //PWM sinyali CCP2'de oluşturuluyor.
 }

void main()
{io_set();
 pwmkur();
 T=30;
 Ton=15;                          //D=%50 başlangıç değeri
 pwm();
  while(1)
  {
   if((art==0)&(Ton!=30)) up();
   if((azal==0)&(Ton!=0)) dwn();
   pwm();
  }
}
```

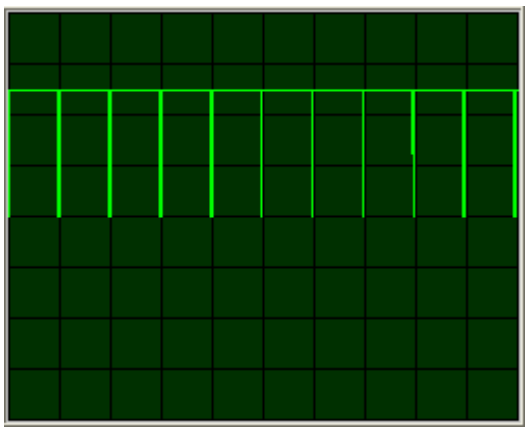
6. Başlangıç değerindeki osiloskop görüntüsü



D=%50 iken Osiloskop Görüntüsü
Volt/div=2V Time/div=100us
Şekil: Öğrenme Faaliyeti-3 Uygulama Sinyal Görüntüsü-1

7. Art buTonuna bastıkça Ton zamanının arttığı ve motorun hızlandığı gözlenir.

8. Dc motor maksimum hıza ulaştığında aşağıdaki osiloskop görüntüsü elde edilmiştir.

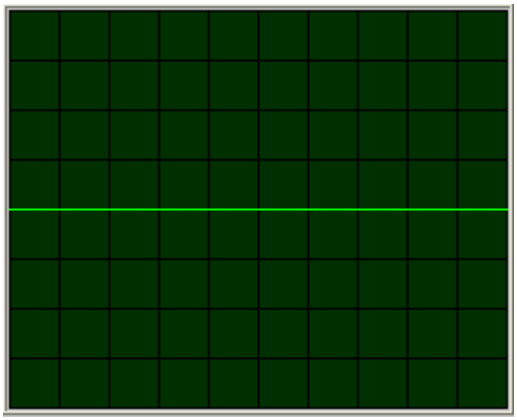


D=%100 iken Osiloskop Görüntüsü
Volt/div=2V Time/div=100us
Şekil: Öğrenme Faaliyeti-3 Uygulama Sinyal Görüntüsü-2

Şekilde görüldüğü gibi D=100 oranı hesaplamlara göre gerçekleştiği halde pratikte çok az bir Toff süresi görülmüştür. Toff süresi yaklaşık 0,3 us ölçülmüştür. Periyot katsayısı değeri 30'un biraz daha üstüne çıktığında tam olarak %100 D oranı elde edilmiştir.

9. Azal buTonuna bastıkça Ton zamanının azaldığı ve motorun yavaşladığı gözlenir. Belirli bir noktadan sonra motor durur.

10. Aşağıda motor durduğunda oluşan PWM sinyali görülmektedir.



D=%0 iken Osiloskop Görüntüsü

Volt/div=2V Time/div=100us

Şekil: Öğrenme Faaliyeti-3 Uygulama Sinyal Görüntüsü-3

Görüldüğü gibi sinyal sürekli lojik-0'dır. Aslında hala PWM çalışması devam etmektedir.

3.2. Ölçme Soruları Cevap Anahtarı

SORU	CEVAP	SORU	CEVAP
1	c	**6**	a
2	b	**7**	d
3	b	**8**	c
4	d	**9**	a
5	b	**10**	b

4. Modül Değerlendirme Cevap Anahtarı

4.1. Ölçme Soruları Cevap Anahtarı

SORU	CEVAP	SORU	CEVAP
1	d	**6**	d
2	d	**7**	c
3	a	**8**	c
4	b	**9**	a
5	a	**10**	c

KAYNAKÇA

- AKTAŞ İsmail, **Anahtarlamalı Güç Kaynakları Bitirme Projesi**, İstanbul, 2000.
- YUZAWA Shuichi, Gürcan BILDIR, **Mekatronik Atölyesi 11.Sınıf Endüstriyel Otomasyon Teknolojileri Proje Kitabı**, İzmir, 2005.
- www.alldatasheet.com
- www.futaba-rc.com