PIC16F877A ile PWM Tabanlı Frekans Üretimi ve LCD, UART ve Osiloskop Üzerinden İzlenmesi

Serdar YILMAZ[1]

1- Öğr.Gör.Dr.; İstanbul Üniversitesi Teknik Bilimler Meslek Yüksekokulu Elektronik Teknolojisi. serdar.yilmaz@iuc.edu.tr ORCID No: 0000-0002-4316-3737

ÖZET

Bu bölümde, PIC16F877A mikrodenetleyicisi kullanılarak donanımsal PWM tabanlı ayarlanabilir bir frekans üreteci tasarlanmış ve üretilen sinyalin temel parametreleri frekans ve çevrim süresi (duty cycle) hem LCD ekranda hem de UART üzerinden seri portta izlenebilir hâle getirilmiştir. PWM üretimi için PIC16F877A'nın CCP1 modülü etkinleştirilmiş, PWM periyodu Timer2–PR2 kaydı üzerinden ayarlanmış ve duty değeri 10-bit çözünürlükte kontrol edilmiştir. Kullanıcı etkileşimi, RA0–RA3 girişlerine bağlanan butonlar aracılığıyla sağlanmış; buton okuma işlemlerinde sıçrama (debounce) etkisini azaltmaya yönelik yazılımsal gecikme ve bırakma kontrolü uygulanmıştır. Frekans ve duty değerleri, sistem saat frekansı ve Timer2 önbölücü (prescaler) parametreleri dikkate alınarak hesaplanmış; elde edilen değerler 16×2 LCD üzerinde ve 9600 baud seri haberleşme üzerinden terminal çıktısı olarak gösterilmiştir. Çalışma Proteus ortamında simüle edilerek osiloskop/terminal doğrulamaları ile test edilmiş; PR2 ve duty ayarlarına bağlı olarak PWM sinyalinin kararlı biçimde üretildiği ve kullanıcı komutlarına hızlı yanıt verdiği gözlenmiştir. Bu yapı, test sinyali üretimi, motor sürme ve güç elektroniği gibi uygulamalar için düşük maliyetli ve anlaşılır bir temel örnek sunmaktadır.

Anahtar Kelimeler – PIC16F877a, Gömülü Sistemler, PWM, Frekans, Çevrim Süresi

GİRİŞ

Gömülü sistemlerde kare dalga/PWM sinyali üretimi; motor sürme, güç elektroniği, haberleşme ve test sinyali oluşturma gibi pek çok uygulamanın temelini oluşturur. Bu bölümde PIC16F877A mikrodenetleyicisinin CCP1 PWM birimi kullanılarak ayarlanabilir PWM frekansı ve duty-cycle üretilmiş; üretilen parametreler LCD (16×2) ve UART seri port üzerinden anlık olarak izlenmiştir. Uygulama Proteus ortamında simüle edilmiş, CCS C ile yazılım geliştirilmiştir.

PIC16F877A tabanlı bir PWM (Pulse Width Modulation) frekans üreteci tasarlarken iki ana grupta kaynaklar dikkate alınabilir (i) üretici dokümantasyonu (datasheet, referans el kitabı, uygulama notları/teknik bültenler) ve (ii) uygulama/öğretici kaynaklar (derleyici kılavuzları, simülasyon araçları ve ders kitapları). Bu bölümdeki çalışma; PWM'in donanımsal olarak Timer2 + CCP1 birimleri üzerinden üretilmesi, frekansın PR2 ile ayarlanması, duty-cycle'ın 10-bit çözünürlükle kontrol edilmesi ve sonuçların LCD ile UART üzerinden izlenmesi hedefleri doğrultusunda ilgili kaynaklara dayandırılmıştır.

PIC16F877A, eğitim ve prototipleme amaçlı uygulamalarda sık kullanılan bir 8-bit PIC ailesi üyesidir; Proteus gibi devre benzetim ortamları

ile birlikte CCS C derleyicisi kullanıldığında, donanım çevre birimlerini (Timer, CCP/PWM, USART vb.) hızlıca doğrulamak ve sistem davranışını gözlemek mümkün olur (Microchip Technology Inc., 2003:68). Bu tür bir geliştirme akışında CCS'nin donanıma yakın C sözdizimi ve hazır yönergeleri (ör. seri haberleşmeyi tek satırla kuran tanımlar) geliştirme süresini kısaltırken, Proteus'un sanal ölçüm/terminal araçları da çıktının gözlemlenmesini kolaylaştırır (CCS, Inc., 2021:8). PWM üretimi özelinde PIC16F877A'nın CCP1 modülü Timer2 ile birlikte çalışarak ayarlanabilir frekansta PWM üretebilir. PWM periyodu Timer2 önbölücü (prescaler) ve PR2 yazmacı üzerinden belirlenir; ilgili temel bağıntı denklemi *PWM_Period* aşağıdaki Denklem (1)'de olduğu şekilde verilir (Microchip Technology Inc., 2003:68). CCP modülünün Capture/Compare/PWM olmak üzere üç temel çalışma kipine sahip olduğu ve PWM kipinde zaman tabanının Timer2 ile kurulduğu, Microchip'in CCP uygulama notlarında ayrıntılı biçimde açıklanır (Palmer, 1997:1).

$$PWM_Period = (PR2 + 1) \cdot 4 \cdot T_{OSC} \cdot TMR2_Prescale \quad (1)$$

Duty-cycle ayarı da CCPR1L ile DC1B bitlerinin birlikte kullanılmasıyla 10-bit çözünürlükte yapılır; datasheette PWM duty-cycle'ın yazmaçlara yüklenmesi ve eşdeğer hesap bağıntısı örneklenir (Microchip Technology Inc., 2003:69). CCP'nin PWM kipinde "duty güncellemesini PWM çevrimini bozmadan" yapmaya yönelik tamponlama yaklaşımı ve çevrim içi güncelleme mantığı Microchip teknik notlarında ayrıca vurgulanır (Microchip Technology Inc., 2020:1). Bu bilgiler, butonlarla çevrim zamanı (duty) arttırma/azaltma yapılan uygulamalarda hem çözünürlük seçimini hem de sayısal adım büyüklüğünü kontrol edilmesini sağlar (J. Morton, 2005).

Oluşturulan frekans ve duty bilgilerinin seri porttan izlenmesi için PIC16F877A'nın donanımsal USART birimi yaygın kullanılan yöntemdir. Asenkron seri iletişimde baud hızı SPBRG kaydedicisiyle ve hız bitleri ile ayarlanır (Garbutt, 2003:2). CCS C derleyicisi tarafından seri portun temel kurulumu düzenlenir ve *printf()* ile gönderim yapılır. CCS C derleyici bu, "hesaplanan değerleri düzenli aralıklarla terminale basma" yaklaşımını doğrudan destekler ve uygun makine kodunu oluşturabilir (CCS, Inc., 2021:8). LCD üzerinde izleme için endüstride fiili standart hâline gelmiş HD44780U komut seti ve 4-bit/8-bit veri yolu seçenekleri, gömülü sistemlerde karakter LCD kütüphanesi tarafından kolaylıkla sağlanır (Hitachi, 1999:1). Bu nedenle, PWM parametrelerinin (frekans/duty) aynı anda hem LCD'de hem seri portta gösterilmesi; kullanıcı arayüzü, hata ayıklama ve deneysel doğrulama açısından öğretici bir bütünlük sağlar (Mazidi, McKinlay ve Causey, 2008:1).

Malzeme ve Yöntemler

PIC16F877A'da PWM Üretimi (CCP1 + Timer2)

PIC16F877A üzerinde PWM üretimi için CCP (Capture/Compare/PWM) modülü kullanılır. CCP1'in PWM çıkışı RC2/CCP1 pinindedir. PWM periyodu Timer2 üzerinden belirlenir ve Timer 2 sayıcısının kabiliyet ve kapasitesi altında control edilir.

PWM frekansı PIC mikrodenetleyicisinin sürüldüğü osilatör frekansına bağlı olarak aşağıdaki Denklem (2)'de olduğu gibi yazılabilir.

$$F_{PWM} = \frac{F_{OSC}}{4.\text{Prescaler}.(PR2+1)} \quad (2)$$

F_{osc}=20Mhz
Prescaler =16(*Timer2*: *T2_DIV_BY_16*)
PR2 değeri butonlardan gelen değerdir

Dolayısıyla PR2 arttıkça payda büyür ve frekans azalır.

Duty-Cycle (Görev Oranı) ve Çözünürlük

CCS C'de set_pwm1_duty(duty10) fonksiyonu PWM duty değerini 10-bit ölçekle ayarlar. Ancak Timer2 periyodu (PR2) değiştikçe erişilebilir duty üst sınırı da değişir. Uygulamada bu üst sınır aşağıda verilen Denklem (3)'te olduğu gibi hesaplanabilir.

$$Dutymax=4\cdot(PR2+1)-1 \quad (3)$$

Duty yüzdesi hesaplaması aşağıda verilen Denklem (4)'te olduğu gibi hesaplanabilir.

$$Duty(\%) = \frac{duty10}{Duty_{max}}.100 \quad (4)$$

Bu yapıda PR2 değiştiğinde duty10 sabit kalırsa duty yüzdesi de değişebilir. Uygulama bunu "duty10 değerini PR2'ye göre sınırlandırarak" kararlı çalıştırır. İleri geliştirme olarak frekans değişirken duty yüzdesinin sabit tutulması da eklenebilir.

Devre Elemanları ve Devre Şeması

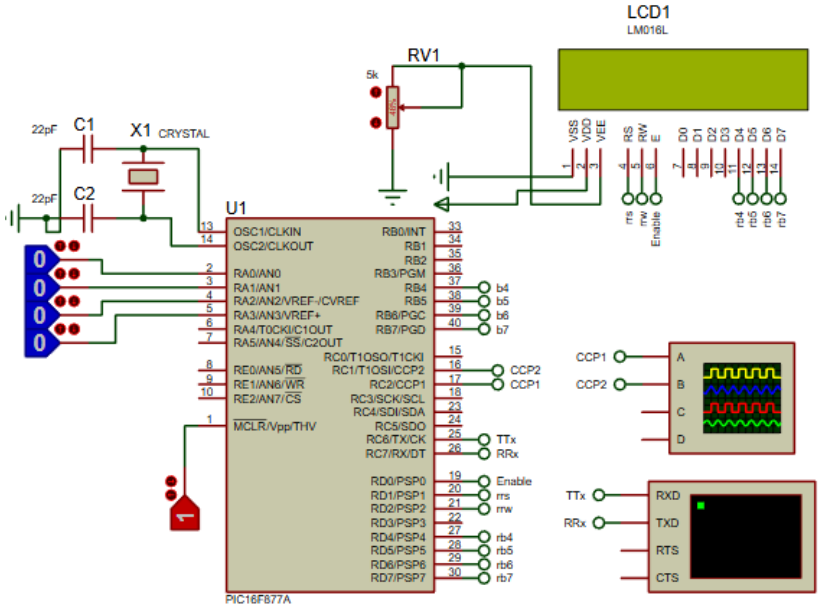
Devrenin kurulumu Proteus programı üzerinde gerçekleştirilmiştir. Devre elemanları gerçek hayatta olan karşılıkları bulunmaktadır. Bunlardan

bazıları görsel açıdan gerçek devre elamanlarından farklılık gösterebilir. Devre elemanları ve kullanım amaçları Tablo 1'de verilmektedir.

Tablo 1: Proteus Programı ve Seçilen Devre Elemanları

Malzeme	Açıklama
PIC16F877A	Mikrodenetleyici
20 MHz kristal + 2 adet kapasitör (22–33 pF)	Osilatör Elemanları
16×2 LCD (LM016)	LCD Ekran
4 adet buton (RA0–RA3 girişleri)	Giriş Anahtarları
Pull-up/pull-down dirençler	Gerçek Uygulama için Direnç
Virtual Terminal (UART izleme için)	Proteus Sanal Terminal
Osiloskop / Frequency Counter (PWM çıkışını doğrulamak için)	Proteus Sanal
LOGICSTATE	Dijital Anahtar (0-1)

Devreye ait şema, Şekil1'de verilmektedir. Devre üzerinde bulunan elemanlar doğrudan bağlantı veya etiket yöntem ile PIC16F877A mikrodenetleyici elemanına bağlanmıştır.



Şekil 1: PIC16F877 Proteus Devre Şeması

LOGICSTATE devre anahtarları mikrodenetleyiciye doğrudan bağlanmış ve bağlantılar RA0-RA3 pinlerinde görülmektedir. Etiket bağlantıları ise osiloskop, sanal seri port elemanı ve LCD'nin D portuna yapılan düzenlemelerinde kullanılmıştır.

B portunda RB4-RB7 etiketlenmiş olmakla birlikte herhangi bir eşleştirilme yapılmamıştır.

Proteus ortamında kurulan donanım tasarımında merkez bileşen olarak **PIC16F877A** seçilmiş ve mikrodenetleyicinin kararlı çalışması için saat kaynağı **20 MHz kristal** ile sağlanmıştır. Kristalin her iki ucuna uygun değerli kondansatörler eklenerek osilatör devresi tamamlanmış, MCLR bağlantısı gerçek fiziksel bir devre üzerinde uygulanacak ise pin girişine 10 kΩ civarında bir direnç ile +5 V'a çekilerek reset hattının gürültüye karşı güvenilir olması sağlanmalıdır. Devre genelinde +5 V besleme ve ortak toprak (GND) bağlantıları ortak olarak oluşturulmalıdır. Proteus devresinde LOGICSTATE devre elemanı ile HIGH (1) ve LOW (0) kolaylıkla sağlanabilmektedir. Şekil 1'de de PIC16F877A 1 numaralı ayak bağlantısında görülmektedir. Simülasyonda olası kararsızlıkları azaltmak amacıyla besleme hatlarında bypass/filtreleme yaklaşımı (pratikte 100 nF decoupling gibi) gözetilmiştir. PWM sinyalinin doğrulanabilmesi için PIC'in **CCP1 çıkışı olan RC2** pini osiloskop sanal ölçüm cihazlarına bağlanarak üretilen dalga şekli ve frekans değişimi anlık olarak gözlemlenebilir hâle getirilmiştir.

Kullanıcı etkileşimi için **RA0–RA3** pinlerine dört adet buton bağlanmış; bu butonlar frekansın (PR2 üzerinden) ve duty-cycle'ın (10-bit duty değeri üzerinden) artırılıp azaltılmasını sağlamaktadır. PortA'da dahili pull-up bulunmadığından, girişlerin "floating" kalmaması için buton hatlarına uygun **LOGICSTATE** anahtarlar eklenerek kararlı lojik seviye elde edilmiştir. Üretilen frekans ve duty bilgilerinin izlenmesi amacıyla **16×2 karakter LCD** 4-bit modda PortD'ye bağlanmış ve kontrast ayarı için LCD'nin V_{EE} pinine bir potansiyometre eklenmiştir. Ayrıca seri port üzerinden veri izlemesi için PIC'in **RC6 (TX)** ve **RC7 (RX)** pinleri Proteus **Virtual Terminal** bileşenine bağlanarak 9600 baud hızında UART çıktıları simülasyon sırasında doğrulanmıştır. Bu donanım kurgusu, PWM üretimi ile kullanıcı kontrolünü aynı sistem üzerinde birleştirirken, LCD ve seri port sayesinde ayarların anlık olarak takip edilmesine olanak tanımaktadır.

Yazılım Tasarımı

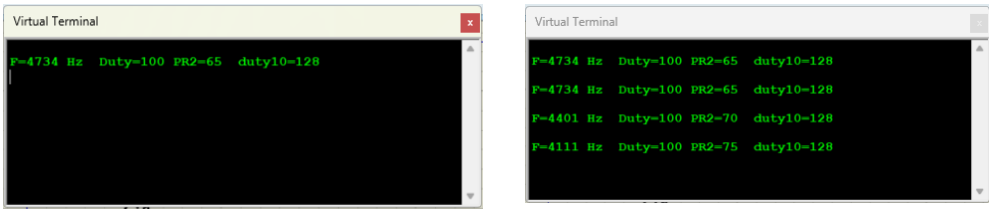
Yazılım tasarımında, PIC16F877A üzerinde PWM üretimi ve izleme işlevlerinin kararlı ve anlaşılır biçimde yürütülmesi için modüler bir yapı benimsenmiştir. Uygulama başlangıcında PortA üzerindeki buton girişlerinin doğru okunabilmesi amacıyla analog işlevler kapatılarak pinler dijital moda alınmış, ilgili yönlendirmeler TRIS ayarlarıyla tanımlanmıştır. PWM üretimi donanımsal olarak CCP1 modülü üzerinden gerçekleştirilmiş; Timer2, seçilen önbölücü (prescaler) değeri ile birlikte PR2 yazmacı (register) kullanılarak PWM periyodunu/frekansını belirleyecek şekilde yapılandırılmıştır. Frekans ayarı, kullanıcı butonlarıyla PR2'nin belirli adımlarla artırılıp azaltılması üzerinden yapılırken, duty-cycle kontrolü 10-bit çözünürlüklü duty değeri (duty10) değiştirilerek sağlanmıştır. PR2 değeri değiştiğinde erişilebilir duty

aralığı da değiştiğinden, duty değerinin geçerli üst sınıra taşmaması için PR2'ye bağlı bir sınırlandırma uygulanmış ve PWM güncellemesi tek bir noktadan yürütülmüştür.

Kullanıcı girişleri RA0–RA3 pinlerinden okunmuş ve buton sıçramalarının (debounce) PWM parametrelerinde hatalı artış/azalışa yol açmaması için kısa gecikme ve "buton bırakılana kadar bekleme" yaklaşımı kod içerisinde kullanılmıştır. Diğer taraftan Proteus simülasyon üzerinde donanım kurulurken kullanılan LOGICSTATE anahtarları olası sıçrama etkilerini ortadan kaldırmaktadır. Parametre değişikliği algılandığında PWM yapılandırması yeniden uygulanmış ve sistem durumu güncellenmiştir. İzleme amacıyla hesaplanan frekans ve duty bilgileri hem 16×2 LCD ekrana hem de UART üzerinden seri porta yazdırılmıştır. CCS C sürümünde printf() format ayrıştırmasının bazı durumlarda tek satırda çok sayıda değişkenle sorun çıkarabildiği göz önünde bulundurularak, seri port çıktısı pratikte daha güvenilir olacak şekilde gerektiğinde birden fazla printf() çağrısına bölünmüştür. Bu yazılım yaklaşımı, frekans ve duty ayarlarının kullanıcı komutlarına hızlı tepki vermesini sağlarken; ekran ve terminal çıktıları sayesinde simülasyon aşamasında sistemin davranışının kolayca doğrulanmasına imkân tanımaktadır.

Seri Port ve LCD Çıktısı

Seri port ve LCD çıktısı bölümünde amaç, PWM ile üretilen sinyalin frekans ve duty-cycle bilgilerinin kullanıcı tarafından anlık olarak izlenebilmesini sağlamaktır. Bu doğrultuda PIC16F877A'nın donanımsal USART birimi 9600 baud hızında asenkron haberleşmeye uygun şekilde yapılandırılmış; TX hattı RC6, RX hattı RC7 pinleri üzerinden Proteus Virtual Terminal bileşenine bağlanarak sistemin ürettiği durum bilgileri terminal ekranında metin olarak görüntülenmiştir.

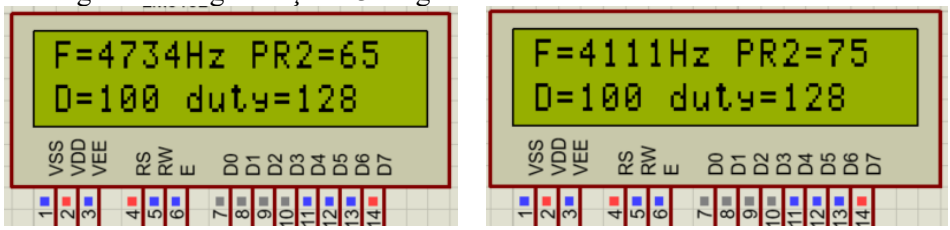


Şekil 2: Sanal (Virtual) Seri Terminal Görselleri

RA0- RA1 Butonları ile oluşturulan frekans değişimleri sanal seri terminal görselleri Şekil 2'de ve LCD görselleri Şekil 3'te verilmiştir.

Aynı zamanda kullanıcı arayüzünü güçlendirmek için 16×2 karakter LCD, 4-bit modda sürülmüş ve frekans ile duty değerleri iki satır halinde ekrana yazdırılmıştır. Böylece kullanıcı, butonlarla yaptığı her ayar

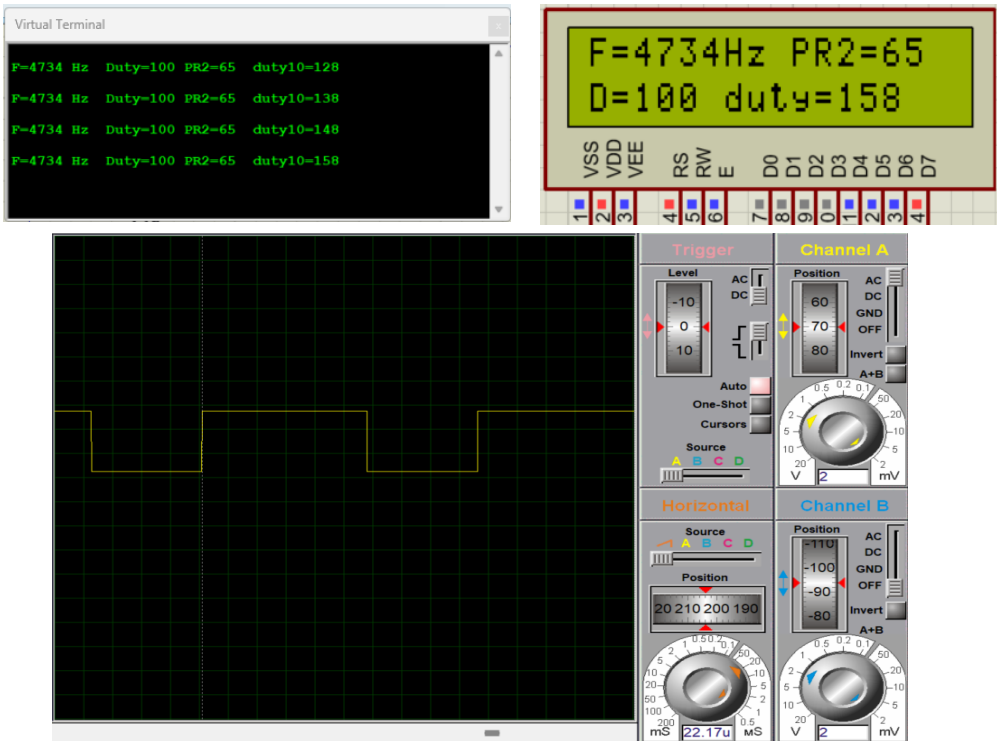
değişikliğinin sonucunu hem LCD üzerinde özet biçimde hem de seri port üzerinden daha ayrıntılı bir kayıt/izleme çıktısı olarak takip edebilmiştir. LCD ekrana gönderilen görsel Şekil 3'de gösterilmektedir.



Şekil 3: LCD Ekran Değerleri

Derleyici programda printf() format ayrıştırmasının zaman zaman tek satırda çok sayıda değişken kullanıldığında hata üretebildiği gözlemlendiğinden, seri port çıktıları uygulamada daha güvenilir sonuç vermesi için gerektiğinde birden fazla printf() çağrısına bölünmüş; LCD tarafında ise ekranın yenilenmesiyle güncel değerlerin her değişimde tutarlı şekilde gösterilmesi sağlanmıştır.

Şekil 4'te Sanal seri terminal ekranı, LCD ve Osiloskop ekranları birlikte verilmiştir. Sinyal frekansının 4734Hz olduğu görülmektedir.



Şekil 4: Sanal Seri Terminal, LCD Ekran ve Osiloskop Frekans Değişim Görselleri

Kod içinde PWM frekansı **PR2 [**int8 pr2_val = 65;] kod üzerinde verilen başlangıç değeri 65 üzerinden (bkz. Ek A: Kod) Denklem (2) kullanılarak "PR2+1=65+1=66" değeri yerine yazılırsa

$$F_{PWM} = \frac{F_{OSC}}{4.Prescaler.(PR2+1)} = \frac{20Mhz}{4224} = 4734Hz$$

olarak hesaplanır.

Ayrıca osiloskop ekranındaki sinyale ait olan değer dikkate alınırsa, yaklaşık olarak 9,5 kareden oluşan sinyalin 22,17µs olduğu hesaplanır. Sinyalin periyodu;

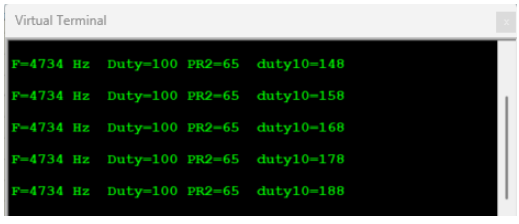
$$T_S = (9{,}5).(22{,}17\mu) = 210{,}615\mu s$$

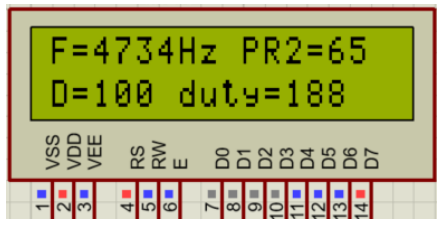
olarak hesaplanır.

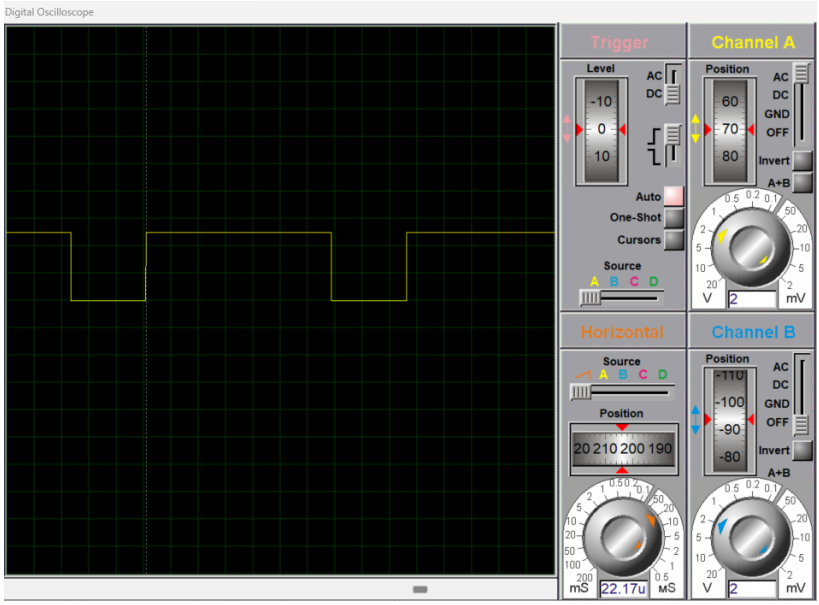
$$F_{PWM} = \frac{1}{T_S} = 4747Hz$$

olarak hesaplanır.
Sonuçlar incelendiğinde frekans değerinin başarılı bir şekilde sanal seri terminal, LCD ekran ve osiloskop üzerinden alındığı görülmektedir.

PWM sinyale ait çevrim süresi dijital karşılığı 128-188 değerleri değişimi nihayetinde ve osiloskop ekranın görselleri Şekil 5'te verilmektedir.







Şekil 5: Sanal Seri Terminal, LCD Ekran ve Osiloskop Duty Cycle Değşim Görselleri

Simülasyon / Doğrulama Sonuçları

Simülasyon ve doğrulama aşamasında geliştirilen sistem Proteus ortamında çalıştırılarak PWM sinyalinin hem dalga biçimi hem de sayısal parametreleri doğrulanmıştır. Öncelikle CCP1 çıkışı olan RC2 pini sanal osiloskop bağlanmış; Timer2 önbölücü (prescaler) ve PR2 değerine bağlı olarak PWM frekansının beklenen şekilde değiştiği gözlemlenmiştir. RA0 ve RA1 butonları ile PR2 belirli adımlarla artırılıp azaltıldığında, frekansın PR2 ile ters orantılı biçimde değişmesi (PR2 arttıkça frekansın düşmesi, PR2 azaldıkça frekansın artması) hem ölçüm cihazlarında hem de yazılımın hesapladığı değerlerde tutarlı sonuçlar vermiştir. Benzer şekilde RA2 ve RA3 butonları ile 10-bit duty değeri artırılıp azaltıldığında, osiloskop üzerinde darbe genişliğinin değiştiği ve duty-cycle oranının kullanıcı komutlarına kararlı biçimde yanıt verdiği doğrulanmıştır. Elde edilen frekans ve duty bilgileri aynı anda LCD üzerinde iki satır halinde görüntülenmiş; seri port üzerinden Virtual Terminal'e gönderilen çıktılar sayesinde yapılan ayarların zaman içinde takibi ve hata ayıklama kolaylaşmıştır. Ayrıca buton okuma işlemlerinde uygulanan yazılımsal debounce yaklaşımı sayesinde, buton sıçramalarının parametrelerde istenmeyen çoklu artış/azalışlara yol açmadığı; PWM güncellemelerinin ise Timer2 ve CCP1 yapılandırması yeniden uygulanarak sorunsuz biçimde gerçekleştirildiği görülmüştür. Bu doğrulamalar sonucunda, önerilen donanım ve yazılım bütünlüğünün Proteus simülasyonunda kararlı çalıştığı, PWM üretiminin hedeflenen şekilde kontrol

edilebilir olduğu ve izleme arayüzlerinin (LCD ve UART) güvenilir biçimde güncel değerleri sunduğu tespit edilmiştir.

SONUÇ

Bu çalışma kapsamında PIC16F877A mikrodenetleyicisi kullanılarak Timer2 ve CCP1 birimleri üzerinden donanımsal PWM sinyali üretilmiş, PWM'in frekans ve duty-cycle parametreleri butonlar aracılığıyla ayarlanabilir hâle getirilmiş ve elde edilen değerler hem 16×2 LCD ekranda hem de UART üzerinden seri portta eşzamanlı olarak izlenebilmiştir. Proteus ortamında gerçekleştirilen doğrulamalar, PR2 değerine bağlı frekans değişiminin beklenen yönde gerçekleştiğini, 10-bit duty ayarı ile darbe genişliğinin kararlı biçimde kontrol edilebildiğini ve kullanıcı girişlerine hızlı yanıt veren güvenilir bir yapı elde edildiğini göstermiştir. Ayrıca yazılımın modüler kurgusu (PWM uygulama ve durum yazdırma işlevlerinin ayrıştırılması) bakım kolaylığı ve genişletilebilirlik açısından önemli bir avantaj sağlamıştır. Sunulan çalışmada PWM sinyalinin frekans değerinin ve çevrim süresi olan duty cycle değiştirilebilir olması dikkate değer bir kazanım sunmaktadır.

Gelecekteki çalışmalar için sistemin işlevselliğini artıracak çeşitli geliştirme önerileri bulunmaktadır. Öncelikle PR2 değiştiğinde duty yüzdesinin sabit kalmasını sağlamak amacıyla duty değerinin PR2'ye göre yeniden ölçeklenmesi ve "yüzde bazlı" bir kontrol mantığı eklenebilir. Kullanıcı arayüzü tarafında, yalnızca artır/azalt butonları yerine keypad veya enkoder kullanılarak frekans ve duty değerlerinin doğrudan sayısal girilmesi sağlanabilir; aynı zamanda seçilen değerlerin EEPROM'a kaydedilip cihaz yeniden başlatıldığında otomatik geri yüklenmesi kullanıcı deneyimini iyileştirecektir. Doğrulama yönünde ise CCP capture veya zamanlayıcı tabanlı ölçüm ile gerçek frekansın ölçülerek hesaplanan değerle karşılaştırılması, sistemin ölçüm-belirsizlik analizini güçlendirebilir. Son olarak, uygulama alanına göre PWM çıkışının filtrelenmesi, sürücü katı eklenmesi (MOSFET sürme vb.) ve elektromanyetik girişim/koruma önlemlerinin incelenmesi gibi adımlar, tasarımın endüstriyel koşullara uyarlanmasını destekleyecektir.

Ek A: Kod

```
#include <16F877A.h>
#fuses HS,NOWDT,NOLVP,NOPROTECT,PUT
#use delay(clock=20000000)
#use rs232(baud=9600, xmit=PIN_C6, rcv=PIN_C7)
// Sende olan lcd.c buna ihtiyaç duyuyor:
#define use_portd_lcd TRUE
#include <lcd.c>
#define FOSC_HZ     20000000UL
#define T2_PRESCALE 16UL // T2_DIV_BY_16
int8  pr2_val = 65;       // PR2: frekans kontrolü
int16 duty10  = 128;      // 10-bit duty
unsigned int32 pwm_freq_hz(int8 pr2)
{// Fpwm = Fosc / (4 * prescale * (PR2+1))
  return (unsigned int32)(FOSC_HZ / (4UL * T2_PRESCALE * ((unsigned int32)pr2 + 1)));}
int16 duty_max_for_pr2(int8 pr2)
{// max duty = 4*(PR2+1)-1
  return (int16)(4U * ((int16)pr2 + 1) - 1);}
void apply_pwm(void)
{  int16 dmax = duty_max_for_pr2(pr2_val);
  if(duty10 > dmax) duty10 = dmax;
  setup_ccp1(CCP_PWM);
  setup_timer_2(T2_DIV_BY_16, pr2_val, 1);
  set_pwm1_duty(duty10);}
void print_status(void)
{  unsigned int32 f = pwm_freq_hz(pr2_val);
  int16 dmax = duty_max_for_pr2(pr2_val);
  int16 duty_pct = (int16)(((unsigned int32)duty10 * 100UL) / (unsigned int32)dmax);
 // LCD (16x2 kısa yaz)
  printf(lcd_putc, "\fF=%luHz PR2=%u", f, pr2_val);
  printf(lcd_putc, "\nD=%lu duty=%lu", duty_pct, (unsigned int32)duty10);
  // UART
  printf("\r\nF=%lu Hz Duty=%lu", f, duty_pct);
  printf(" PR2=%u  duty10=%lu \r\n", pr2_val, (unsigned int32)duty10);}
void main()
{ // RA0-RA3 buton -> dijital yap
  setup_adc_ports(NO_ANALOGS);
  setup_adc(ADC_OFF);
  set_tris_a(0x0F);  // RA0..RA3 giriş
  set_tris_c(0x80);  // RC7 RX giriş
  set_tris_d(0x00);  // LCD PortD çıkış
  lcd_init();
  apply_pwm();
  print_status();
  while(TRUE)
  {  int1 changed = 0;
    if(input(PIN_A0))  {  delay_ms(20); if(input(PIN_A0)) { while(input(PIN_A0));
      if(pr2_val < 250) pr2_val += 5; if(pr2_val > 250) pr2_val = 250; changed=1; } }
    if(input(PIN_A1))  {  delay_ms(20); if(input(PIN_A1)) { while(input(PIN_A1));
      if(pr2_val > 65) pr2_val -= 5; if(pr2_val < 65) pr2_val = 65; changed=1; } }
    if(input(PIN_A2))  {  delay_ms(20); if(input(PIN_A2)) { while(input(PIN_A2));
      duty10 += 10; changed=1; } }
    if(input(PIN_A3))  {  delay_ms(20); if(input(PIN_A3)) { while(input(PIN_A3));
      if(duty10 >= 10) duty10 -= 10; changed=1; } }
    if(changed)
    {  apply_pwm();
  print_status();  }  delay_ms(30);  }}
```

REFERANSLAR

CCS, Inc. (2021). CCS C Compiler: User Manual (Kasım 2021). CCS, Inc.

Garbutt, M. (2003). Asynchronous communications with the PICmicro USART (Application Note AN774, DS00774A). Microchip Technology Inc.

Hitachi. (1999). HD44780U (LCD-II): Dot matrix liquid crystal display controller/driver (ADE-207-272(Z), Rev. 0.0).

Labcenter Electronics. (t.y.). Serial communication between Proteus VSM and physical serial port
Mazidi, M. A., McKinlay, R. D. ve Causey, D. (2008). PIC microcontroller and embedded systems: Using assembly and C for PIC18. Pearson Prentice Hall.
Microchip Technology Inc. (1997). PICmicro Mid-Range MCU Family Reference Manual (DS33023A). Microchip Technology Inc.
Microchip Technology Inc. (2003). PIC16F87XA data sheet (DS39582B). Microchip Technology Inc.
Microchip Technology Inc. (2020). Using the CCP Module (Technical Brief TB3275, DS90003275A). Microchip Technology Inc.
Morton, J. (2005). The PIC microcontroller: Your personal introductory course (3. bs.). Newnes/Elsevier.
Palmer, M. (1997). Using the CCP Module(s) (Application Note AN594, DS00594B). Microchip Technology Inc.
Predko, M. (2007). Programming and customizing the PIC microcontroller (3. bs.). McGraw-Hill Education/TAB.
J. Morton, The PIC Microcontroller: Your Personal Introductory Course, 3rd ed., Newnes/Elsevier.