T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GELENEKSEL MİKRODENETLEYİCİLERLE GELİŞMİŞ BİR ENTEGRE TEST CİHAZININ TASARLANMASI

YÜKSEK LİSANS TEZİ

Murat ATÇI

Enstitü Anabilim Dalı : ELEKTRONİK VE BİLG. EĞT.

Tez Danışmanı : Yrd. Doç. Dr. H.İbrahim ESKİKURT

Haziran 2009

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

GELENEKSEL MİKRODENETLEYİCİLERLE GELİŞMİŞ BİR ENTEGRE TEST CİHAZININ TASARLANMASI

YÜKSEK LİSANS TEZİ

Murat ATÇI

Enstitü Anabilim Dalı : ELEKTRONİK VE BİLG. EĞT.

Bu tez 17 / 06 /2009 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Yrd. Doç. Dr.	Prof. Dr.	Yrd. Doç. Dr.
Halil İbrahim ESKİKURT	Etem KÖKLÜKAYA	Ahmet Turan ÖZCERİT
Jüri Başkanı	Üye	Üye

TEŞEKKÜR

Bu tezde, günümüzde sıklıkla kullanılan, boyut olarak küçük; ancak yaptığı işler açısından büyük avantajlar sağlayan mikrodenetleyicinin genel özellikleri ve donanımsal yapısı anlatılmış ayrıca programlanmasından bahsedilmiştir. Bu çalışmada, dijital entegrelerin sağlamlık testinin hızlı bir şekilde yapılmasını ve seri numaralarının öğrenilmesini sağlamak amacıyla 8051 mikrodenetleyicisi kullanılmıştır.

Bu çalışmanın her aşamasında bana yardımcı olan ve görüşlerini sunan danışmanım Yrd. Doç. Dr. Halil İbrahim ESKİKURT'a, bu noktaya gelmemde bana yardımlarını esirgemeyen aileme ve arkadaşlarım Aydın ALKAN ile Zafer ÖNÜR'e teşekkür ederim.

Murat ATÇI

İÇİNDEKİLER

TEŞEKKÜR……... ii
İÇİNDEKİLER…….. iii
SİMGELER LİSTESİ……..... v
ŞEKİLLER LİSTESİ……. vi
TABLOLAR LİSTESİ……... viii
ÖZET……. ix
SUMMARY……... x

BÖLÜM 1.
GİRİŞ……. 1

BÖLÜM 2.
MİKRODENETLEYİCİLER VE 8051 MİKRODENETLEYİCİSİ
HAKKINDA GENEL BİLGİ…… 2
2.1. 8051 Mikrodenetleyicisinin Genel Yapısı…… 3
2.2. 8051 Mikrodenetleyicisinin C Program Yapısı…… 5

BÖLÜM 3.
FARKLI DİJİTAL ENTEGRE TEST CİHAZLARI……. 6
3.1. Yapılmış Test Cihazları…… 6

BÖLÜM 4.
8051 MİKRODENETLEYİCİLİ DİJİTAL ENTEGRE TEST CİHAZI……... 9
4.1. Transistör Transistör Lojik(TTL)……..... 9
4.2. Tamamlayıcı Metal Oksit Yarı İletken(CMOS)……... 9
4.3. Tasarım Aşamaları…… 11

4.3.1. AT89C51RD2 mikrodenetleyicisi ve geliştirme kartı…… 11
4.3.2. Dijital entegre test cihazı…… 11
4.3.2.1. Devre şemasının tasarımı……... 12
4.3.2.2. Devre baskı şemasının tasarımı……..... 12
4.3.2.3. Devre için program kodunun yazılması.....…….. 16
4.3.3. KEIL µVision programı ile proje oluşturma……..... 16
4.4. Entegre Test Etme……. 26

BÖLÜM 5.
SONUÇLAR VE ÖNERİLER……... 29

KAYNAKLAR…….. 30
EKLER……... 31
ÖZGEÇMİŞ……..…….. 61

SİMGELER LİSTESİ

TTL	: Transistör Transistör Lojik
CMOS	: Tamamlayıcı Metal Oksit Yarı İletken
RST	: Reset

ŞEKİLLER LİSTESİ

Şekil 2.1.	8051 mikrodenetleyicisinin uçları ve entegre olarak görünüşü….	3
Şekil 2.2.	8051 mikrodenetleyicisinin C program yapısı……...	5
Şekil 3.1.	BK Precision 575A dijital entegre test cihazı……	6
Şekil 3.2.	ATMEGA32 mikrodenetleyicisi ile yapılmış dijital entegre test cihazı……..	7
Şekil 3.3.	2 Tane 8951 mkrodenetleyicisi kullanılarak yapılmış olan dijital entegre test cihazı……...	7
Şekil 4.1.	AT89C51RD2 tabanlı mikrodenetleyici geliştirme kartı……...	11
Şekil 4.2.	Entegre test cihazı devresi açık şeması……..	12
Şekil 4.3.	LCD baskı devresi……..	12
Şekil 4.4.	ZIF soket baskı devresi……..	13
Şekil 4.5.	Tuş takımı baskı devresi…….	13
Şekil 4.6.	Cihazın üstten görünüşü…….	14
Şekil 4.7.	Cihazın yandan görünüşü……...	14
Şekil 4.8.	Cihazın içten görünüşü……...	15
Şekil 4.9.	Cihazın çalışır haldeki görünüşü……	15
Şekil 4.10.	Keil µVision'da projenin oluşturulması…….	16
Şekil 4.11.	Oluşturulacak projenin kaydedilmesi…….	17
Şekil 4.12.	Kullanılacak mikrodenetleyici firmasının seçilmesi……..	17
Şekil 4.13.	Atmel mikrodenetleyicisinin seçilmesi……..	18
Şekil 4.14.	Projede kod yazımı için dosya oluşturulması…….	19
Şekil 4.15.	Oluşturulacak dosyanın türünün belirlenmesi……	19
Şekil 4.16.	Dosyanın projeye eklenmesi……..	20

Şekil 4.17. Son ayarların yapılması…….. 20
Şekil 4.18 Projenin derlenmesi……..... 21
Şekil 4.19. Kullanılan mikrodenetleyicinin seçilmesi…….. 22
Şekil 4.20. Seri porttan yükleme için seçimin yapılması……. 22
Şekil 4.21. Portun ve baud hızının seçilmesi…… 23
Şekil 4.22. Hex dosyasının yüklenmesi…… 23
Şekil 4.23. Dosyanın mikrodenetleyiciye yüklenmeye hazır olması……... 24
Şekil 4.24. Dosyanın mikrodenetleyiciye yüklenmesinin tamamlanması…… 24
Şekil 4.25. Uygulamanın akış diyagramı……. 25
Şekil 4.26. 7400 entegresinin blok şeması ve doğruluk tablosu……... 26

TABLOLAR LİSTESİ

Tablo 4.1. CMOS – TTL karşılaştırması……. 10
Tablo 4.2. CMOS – TTL entegrelerin çalışma özellikleri……... 10

ÖZET

Anahtar kelimeler: Mikrodenetleyici, C Programlama Dili, Dijital Entegre Test Cihazı

Elektronik alanında kullanılan mikrodenetleyiciler oldukça geniş bir kullanım alanına sahiptirler. Bir mikrodenetleyici, komple bir bilgisayarın (MİB, hafıza ve giriş - çıkışlar) tek bir tümleşik devre üzerinde üretilmiş halidir. Hafıza birimlerine ve giriş – çıkış uçlarına sahip olmaları sayesinde tek başlarına çalışabildikleri gibi, donanımı oluşturan diğer elektronik devrelerle irtibat kurabilir, uygulamanın gerektirdiği fonksiyonları gerçekleştirebilirler. Mikrodenetleyiciler oldukça küçük boyutludurlar, çok düşük güç tüketimine sahiptirler, düşük maliyetlidirler ve yüksek performansa sahiptirler.

Tezde, mikrodenetleyicinin genel özellikleri, donanımsal yapısı ve programlanması hakkında bilgiler verilmiştir. Yapılan test cihazı, dijital entegrelerin sağlamlık testini ve seri numaralarının öğrenilmesini hızlı bir şekilde gerçekleştirmektedir. Uygulamada AT89C51RD2 mikrodenetleyicisi ve "C" programlama dili kullanılmıştır. Ayrıca Proteus, Keil µVision ve Atmel Flip programlarından faydalanılmıştır.

Bu uygulama sayesinde çok geniş bir kullanım alanına sahip olan dijital entegrelerin hızlı bir şekilde test edilmesi sağlanmaktadır. Böylece zamandan ve maliyetten tasarruf elde edilmesini sağlayacak ayrıca ek çalışmalarla uğraşılmasını da ortadan kaldıracaktır.

DESIGNING ADVANCED INTEGRATED TEST DEVICE WITH TRADITIONAL MICROCONTROLLERS

SUMMARY

Key Words: Microcontroller, C Programming Language, Digital IC Test Device

Microcontrollers used in electronics have a wide range of usage area. A microcontroller is a version of a complete computer (CPU, memory and input-outputs) built on an integrated circuit. Since they have memory and input-output ports they can work themselves and connect to other electronic circuits composing the hardware and execute the functions application requires. Microcontrollers are very small, consume little energy, cost low and have high performance.

This thesis includes general information on microcontrollers, their hardware structure and programming. The constructed test device can easily execute the test on durability of digital integrates and detecting their serial numbers. AT89C51RD2 microcontroller and "C" programming language are used in application. Besides, Proteus, Keil, µVision and Atmel Flip programs are used.

With this application, digital ICs which have wide range of usage area are tested very quickly. This will save time and money and will also save efforts on other studies.

BÖLÜM 1. GİRİŞ

Bir mikrobilgisayarı oluşturan temel bileşenlerin tek bir yonga içerisinde üretilmiş haline mikrodenetleyici denir. Bugün pek çok günlük uygulamada mikrodenetleyiciler kullanılmaktadır. Örneğin; taşıtlar, kamera, cep telefonu, televizyon, radyo, hesap makinesi, çamaşır makinesi, bulaşık makinesi, kombi, fırın, fotokopi makinesi, elektronik oyuncaklar vb.

Dijital elektronik gibi uygulama derslerinde çok kullanılan TTL ve CMOS entegreler bazen sorunlar çıkarabilmektedir. Bunlar kendi yapılarından kaynaklandığı gibi kullanan kişilerden de kaynaklanabilmektedir. Bunların sağlamlığını test etmek bazen zor olabilmektedir. Piyasadaki test cihazları da oldukça pahalı olduğundan böyle bir çalışma yapılmıştır. Geliştirilebilir olduğundan rahat bir kullanım sunmaktadır. 8051 mikrodenetleyicisi kullanılmasının amacı, hazır halde programlanabilir bir kartın olmasındandır. İstenirse diğer mikrodenetleyicilerde kullanılabilir. Kullanıcıya göre değişik özelliklerde eklenebilir.

Tez içinde konu hakkında bilgiler verilmeden önce Bölüm 2'de mikrodenetleyicilerle ilgili kısa bir açıklama yapılmıştır.

3. Bölümde yapılmış olan benzer projeler anlatılmış ve bu işlerle uğraşan şirketlerin yaptıkları test cihazları genel hatları ile anlatılmıştır.

4. Bölümde ise tezde yapılan test cihazının yapım aşaması, kullanılmış olan programlar ve bunların nasıl çalıştığı ile ilgili genel bilgiler anlatılmıştır. Ekler kısmında da yazılmış olan programın kodları verilmiştir.

BÖLÜM 2. MİKRODENETLEYİCİLER VE 8051 MİKRODENETLEYİCİSİ HAKKINDA GENEL BİLGİ

Mikrodenetleyiciler;

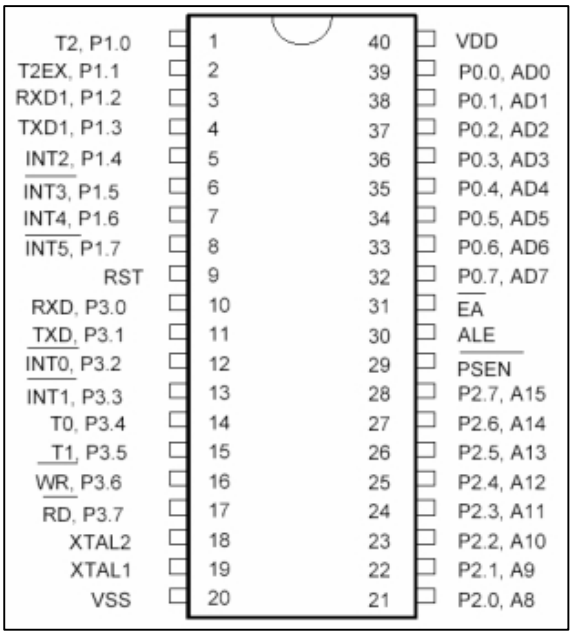
1- Yalnız başlarına çalışabilirler
2- Tek tümdevre elemanıdırlar
3- Sistem kararları genellikle harici sinyallere bağlıdır
4- Elektronik bir cihazın davranışını denetlerler
5- Bir devrenin beyni konumundadırlar
6- Küçük boyutludurlar
7- Düşük güç tüketimine sahiptirler
8- Yüksek performansa sahiptirler

Genel olarak bir mikrodenetleyici şu birimlerden oluşmaktadır;

1- Bir mikroişlemci çekirdeği(CPU)
2- Program ve veri belleği(ROM, RAM)
3- Giriş/Çıkış(I/O) birimleri
4- Saat darbesi üreteçleri
5- Zamanlayıcı/Sayıcı birimleri
6- Kesme kontrol birimi
7- A/D ve D/A (Analog/Dijital ve Dijital/Analog) çeviriciler
8- Darbe genişlik üreteci(PWM)
9- Seri Haberleşme Birimi(UART, RS–232, CAN, I^2C vb.)

2.1. 8051 Mikrodenetleyicisinin Genel Yapısı

İlk üretildikleri günden bugüne kadar geçen zaman zarfında büyük aşamalar kaydeden mikrodenetleyiciler geniş bir kullanım alanına sahip olmuşlardır. İlk olarak Intel tarafından üretilmesine rağmen bugün aralarında Atmel, Dallas, Semiconductors ve Philips'in de bulunduğu birçok üretici firma tarafından da mikrodenetleyiciler üretilmektedir.



Şekil 2.1. 8051 mikrodenetleyicisinin uçları ve entegre olarak görünüşü

Şekilden de görüldüğü gibi 8051 mikrodenetleyicisinin 20 nolu ucu VSS, 40 nolu ucu ise VDD'dir. Bu mikrodenetleyici 5 Voltluk bir gerilimle beslenmektedir. Ancak gelişen teknoloji ile birlikte daha düşük güç harcayan mikrodenetleyicilerde üretilmeye başlanmıştır.

8051 mikrodenetleyicisi; $\overline{\text{PSEN}}$, ALE, $\overline{\text{EA}}$, RST gibi kontrol uçlarına sahiptir.

8051 mikrodenetleyicisinde XTAL1 ve XTAL2 olmak üzere 2 tane osilatör girişi vardır. MCS–51 mikrodenetleyicilerinin çoğu 12 MHz'lik kristal frekansına sahiptir.

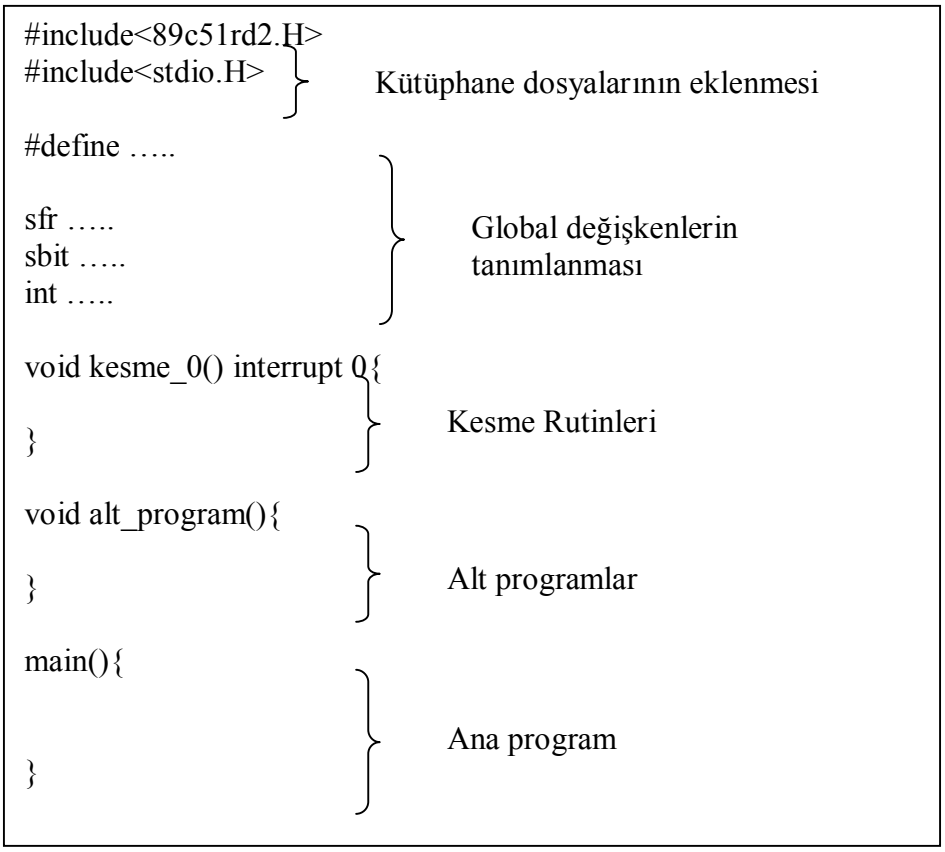
Standart 8051 mikrodenetleyicilerinde 8 bitlik 4 tane Giriş / Çıkış portu vardır. Bunlar; P0, P1, P2 ve P3'tür. Bu portlar çeşitli özelliklere sahiptirler. Port 0, hem genel amaçlı G / Ç portu olarak hem de adres ve veri yolu olarak kullanılabilir. Port 0, genel amaçlı G / Ç portu olarak kullanıldığında pull-up dirençleri ile birlikte kullanılmalıdır.

Port 1, G / Ç portu olarak kullanılmaktadır. Port 2'de Port 0 gibi iki işlevlidir. Birincisi, adresin yüksek değerlikli 8 hattını oluşturur. İkincisi, genel amaçlı G/Ç hattı olarak kullanılır. Port 3'te hem genel amaçlı G / Ç portu olarak kullanılabilir hem de çeşitli alternatif özelliklere sahip uçları sayesinde farklı amaçlar içinde kullanılabilir.

8051 mikrodenetleyicisinde; sayıcı / zamanlayıcı, kesmeler ve seri haberleşme için gerekli kaydediciler bulunmaktadır. Bu kaydedicilerin kullanımı ile ilgili detaylı bilgilere kaynak olarak da gösterilmiş olan 8051 Mikrodenetleyici Uygulamaları kitabından ulaşılabilir.

8051 mikrodenetleyicisine C tabanlı bir dille kod yazılabilir. Tezde kullanılan Keil µVision programı ile bu iş kolaylıkla yapılabilir. Bu program ile ayrıca asm kodları da yazılabilir.

2.2. 8051 Mikrodenetleyicisinin C Program Yapısı



Şekil 2.2. 8051 mikrodenetleyicisinin C program yapısı

BÖLÜM 3. FARKLI DİJİTAL ENTEGRE TEST CİHAZLARI

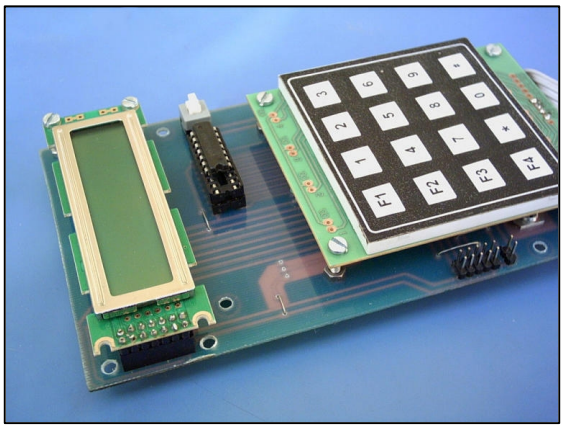
3.1. Yapılmış Test Cihazları

Yapılmış olan bu tezin dışında da B&K Precision isimli şirketin üretmiş olduğu 575A modelli dijital entegre test cihazı vardır. Bu cihazla ilgili detaylı bilgiye B&K Precision şirketinin sitesinden ulaşılabilir. Bunların dışında çeşitli firmaların yaptıkları cihazlarda bulunmaktadır. Bu cihazlar gelişmiş özeliklere sahip olmasına karşın fiyatları da $1000 civarındadır. Bu özelliklerinden dolayı geniş bir kullanım alanına sahiptirler. Yapılmış olan bu tez ise daha çok eğitim amaçlı tasarlanmış ve okullardaki dijital elektronik derslerinde rahatça kullanılabilir, oldukça da düşük maliyete temin edilebilir. Geliştirilmeye de açık olduğundan daha da işlevsel hale getirilebilir.

Şekil 3.1. BK Precision 575A dijital entegre test cihazı

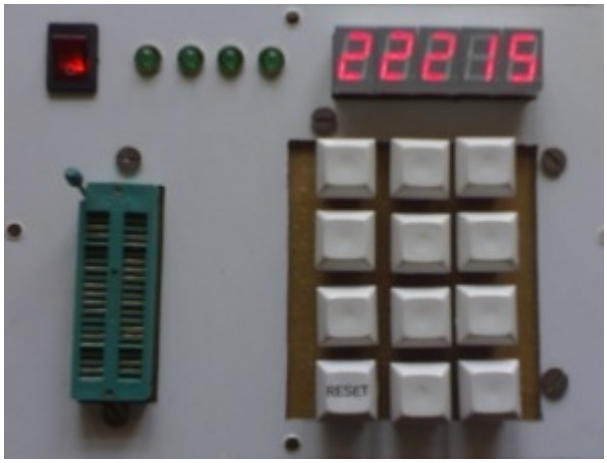
Böyle şirketlerin dışında yapılmış olan projeler de bulunmaktadır. Bunlardan biri olan dijital entegre test cihazı diğer sayfada görülmektedir. Diğerleri gibi aynı işlemi

yapmasına rağmen daha basit bir halde detaya girilmeden basitçe tasarlanmış bir cihazdır. Uygulamada ATMEGA32 mikrodenetleyicisi kullanılmıştır.



Şekil 3.2. ATMEGA32 mikrodenetleyicisi ile yapılmış dijital entegre test cihazı

Yine bir başka projede ise aynı işlemi gerçekleştiren bir cihaz bulunmaktadır. Ancak bu uygulama 2 tane 8951 mikrodenetleyicisinden oluşmaktadır. Bunun amacı port sayısını arttırarak daha fazla pine sahip entegreleri test etme olanağı sağlamaktır.



Şekil 3.3. 2 tane 8951 mikrodenetleyicisi kullanılarak yapılmış olan dijital entegre test cihazı

Bu tezde yapılan uygulamada ise, yapılmış bu projeler dışında entegre arama özelliği vardır. Ayrıca bu cihaz, yeni bir entegre çıktığında ya da cihaz daha işlevsel hale getirildiğinde yeni olan entegre ya da entegrelerin kodları yeniden yazılarak ana

programa eklenir, sonra derlenir ve tekrardan 8051 mikrodenetleyicisine seri port yardımıyla program yüklenir. Kısaca, sadece tek tip mikrodenetleyiciye bağlı kalmadan istenilen ihtiyaca göre devreler yapılabilir. Gelişime açıktır. Günümüzde mikrodenetleyicilerin özellikleri geliştikçe daha işlevli uygulamalar yapılabilir. Kullanılan programlama dili Keil C'dir. Bunun haricinde asm ya da diğer mikrodenetleyicilerin kendi programlama dilleri de kullanılabilir.

BÖLÜM 4. 8051 MİKRODENETLEYİCİLİ DİJİTAL ENTEGRE TEST CİHAZI

4.1. Transistör Transistör Lojik(TTL)

Diyot Transistör Lojik(DTL) entegrelerin gelişmiş şekli olan ve giriş olarak çok elemanlı(emiterli) transistörlerin kullanıldığı TTL entegreler, en yaygın kullanılan dijital entegre grubudur. TTL'de temel eleman 'Ve Değil' kapısıdır.

TTL entegreler 74 ve 54 serileri ile gösterilir. 74 serisi endüstri standardı olarak kullanılır ve 4.75V - 5.25 V çalışma gerilimi ile 0° C -70° C sıcaklık bandında çalışırken, 54 serisi askeri uygulamalarda kullanılır ve 4.5V - 5.5V çalışma gerilimi ile -55° C - +125° C sıcaklık bandında çalışırlar.

4.2. Tamamlayıcı Metal Oksit Yarı İletken(CMOS)

CMOS'ta P ve N kanal MOSFET'ler birlikte kullanılır. CMOS ailesi, P-MOS ve N-MOS kanalın sahip olduğu üstünlükleri aynı devrede toplar. CMOS entegreler, P ve N tipi MOS'lara göre çok daha karmaşık bir yapıya ve daha düşük eleman yoğunluğuna sahiptirler.

40 serisi ve 74C serisi entegreler; 3-15 Volt arasındaki gerilimlerle, 74HC ve 74HCT serileri ise 2V - 6V arasındaki gerilimlerle çalışırlar.

Tablo 4.1. CMOS – TTL karşılaştırması

CMOS	TTL
CMOS, daha iyi paketleme yoğunluğuna sahiptir	TTL, daha fazla yer kaplar
CMOS, büyük bir Fan-Out'a sahiptir.	TTL, daha az girişle çalışabilir
CMOS, daha az güç tüketir.	TTL, daha fazla güç kullanır
CMOS, çok iyi statik hassasiyete sahiptir.	TTL, statik elektrikten dolayı daha az hassastırlar
CMOS, FET(Alan-Etkili Transistör) kullanır	TTL, BJT (Bipolar Transistör) kullanır
CMOS, güç bağlantıları için Vdd ve Vss kullanır	TTL, BJT kullanır.

Tablo 4.2. CMOS – TTL entegrelerin çalışma özellikleri

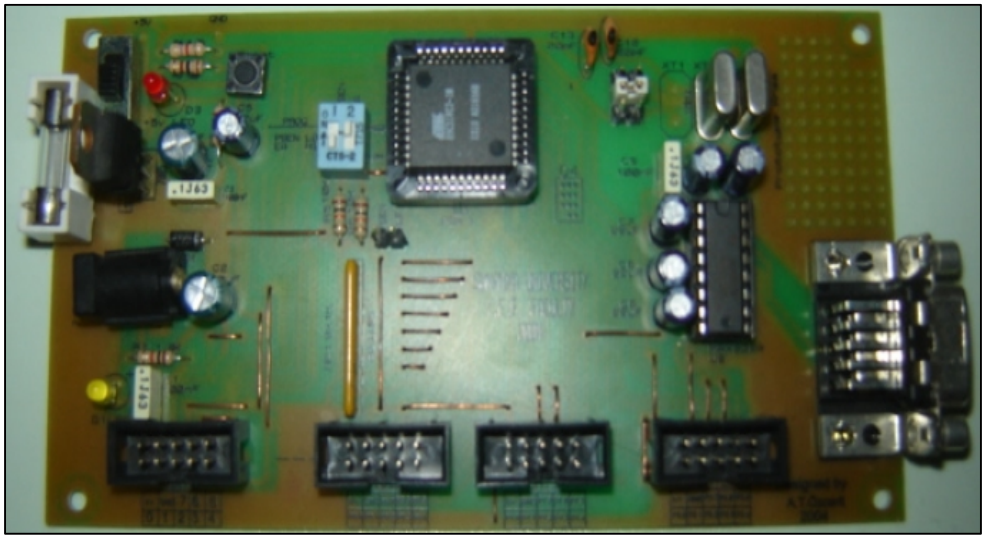
	TTL 74LS	4000 CMOS
Kaynak Voltajı	+5V ± 5%	+3 - +18V
Giriş Akımı	~.25mA	İstemez
Hız	Hızlı, 25MHz	Yavaş, 2MHz
Güç İsrafı	Yüksek, 2mW/gate	Düşük, <.3mW/gate
Fan-Out	10	10

4.3. Tasarım Aşamaları

Yapılmış olan test cihazının hangi aşamalardan geçtiğini, hangi programların kullanıldığını ve mikrodenetleyicinin nasıl programlandığıyla ilgili bilgiler yer almaktadır.

4.3.1. AT89C51RD2 mikrodenetleyicisi ve geliştirme kartı

Tezde kullanılan mikrodenetleyici, AT89C51RD2 mikrodenetleyicisi, ATMEL firması tarafından üretilmektedir.



Şekil 4.1. AT89C51RD2 tabanlı mikrodenetleyici geliştirme kartı

AT89C51RD2 mikrodenetleyicisi, 8 bit 80C51 CMOS mikrodenetleyicisinin yüksek performanslı CMOS Flash sürümüdür.

4.3.2. Dijital entegre test cihazı

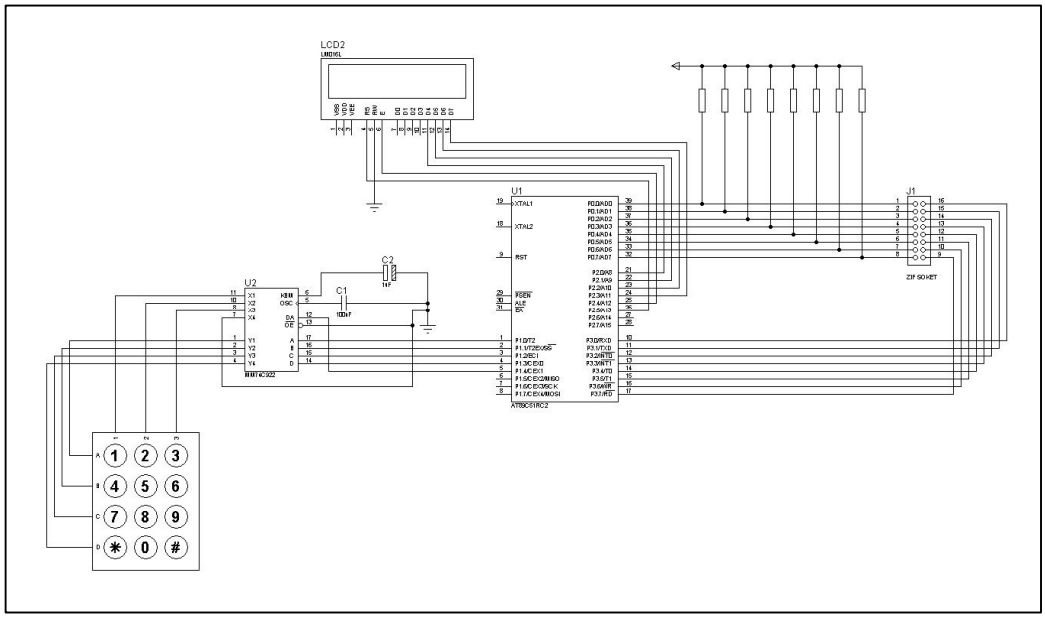
74, 40 45 seri numaralı entegrelerin(gerekli kodlar yazılarak 54 serisi entegrelerde test edilebilir) hem sağlamlığını test eden hem de seri numaralarını öğrenmeye yarayan bu cihaz üç aşamadan oluşmaktadır. Bunlar:

1- Devre şemasının tasarımı

2- Devre baskı şemasının tasarımı

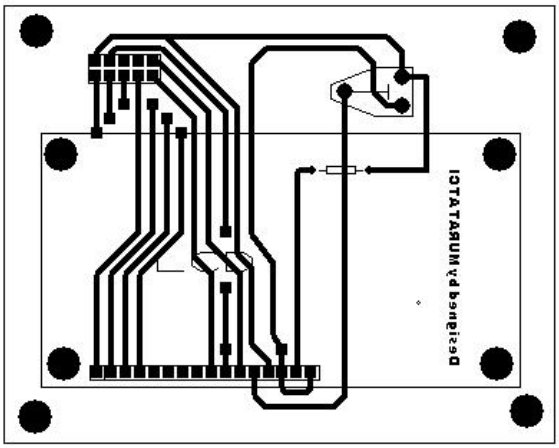
3- Devre için program kodunun yazılması

4.3.2.1. Devre şemasının tasarımı

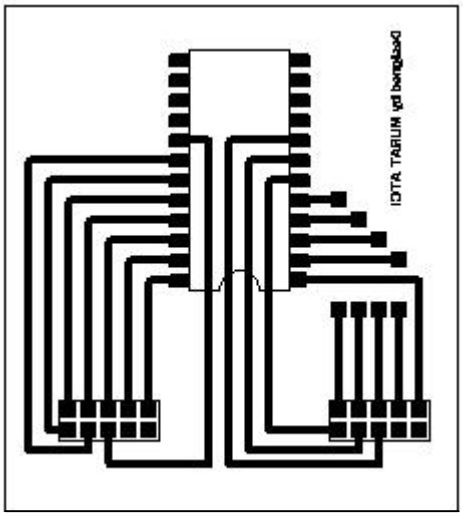


Şekil 4.2. Entegre test cihazı devresi açık şeması

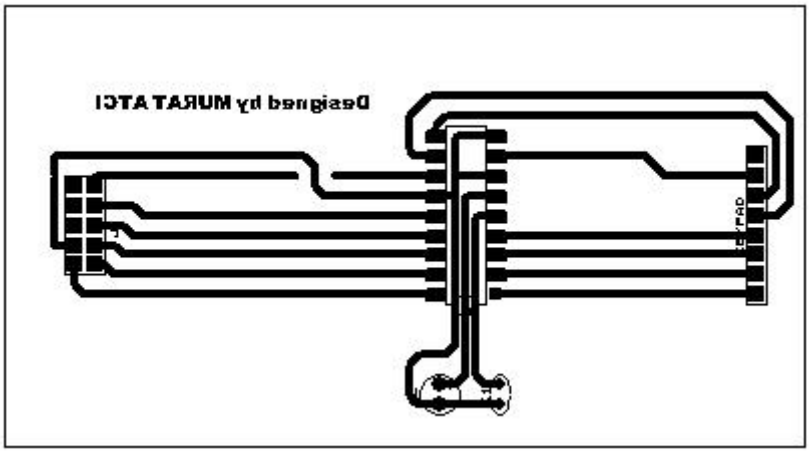
4.3.2.2. Devre baskı şemasının tasarımı



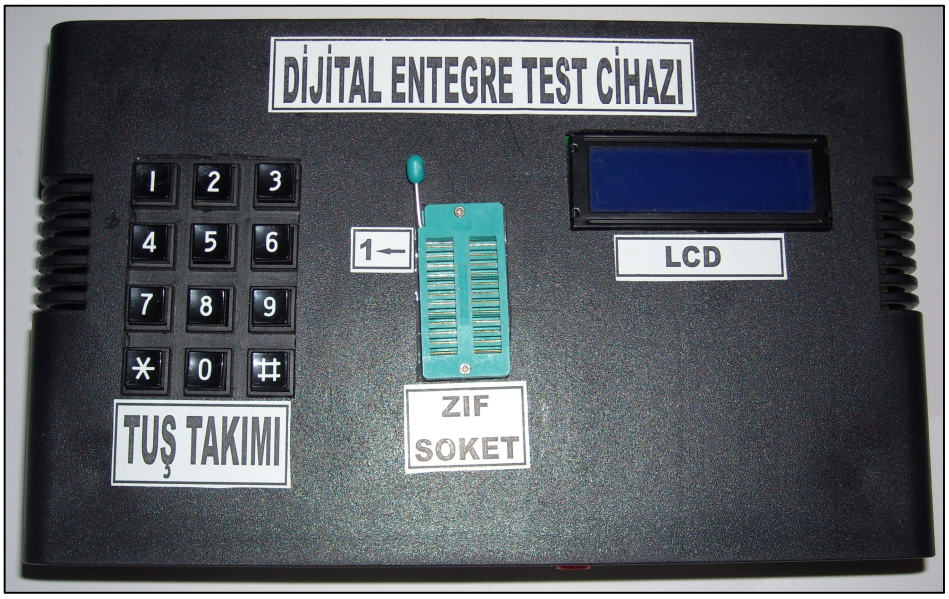
Şekil 4.3. LCD baskı devresi



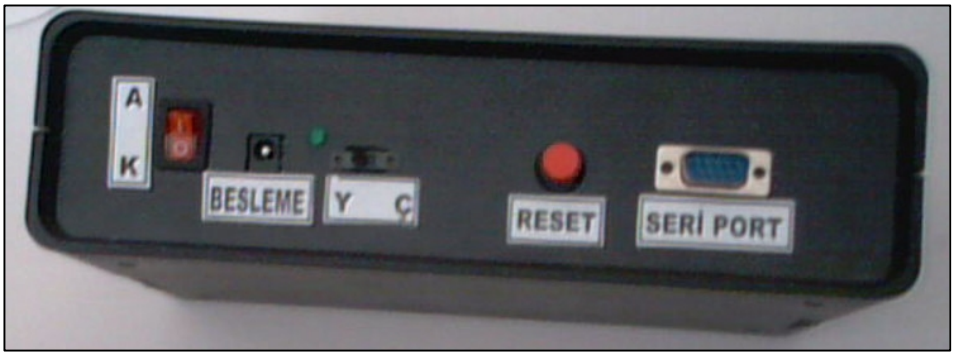
Şekil 4.4. ZIF soket baskı devresi



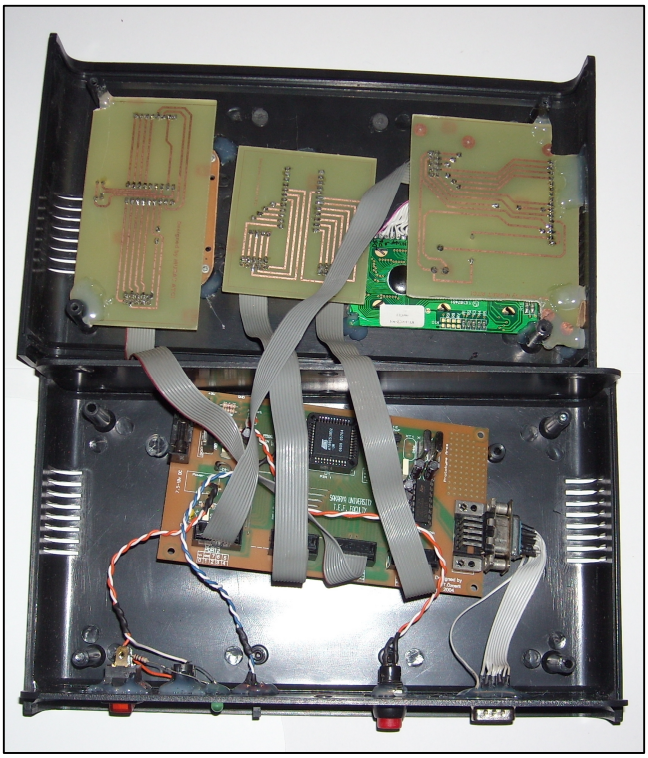
Şekil 4.5. Tuş takımı baskı devresi



Şekil 4.6. Cihazın üstten görünüşü



Şekil 4.7. Cihazın yandan görünüşü



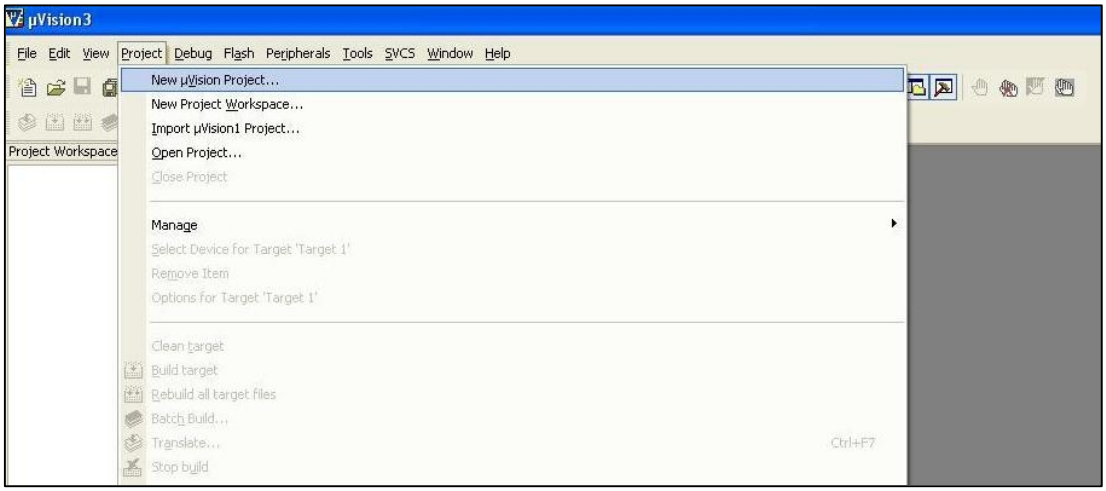
Şekil 4.8. Cihazın içten görünüşü

Şekil 4.9. Cihazın çalışır haldeki görünüşü

4.3.2.3. Devre için program kodunun yazılması

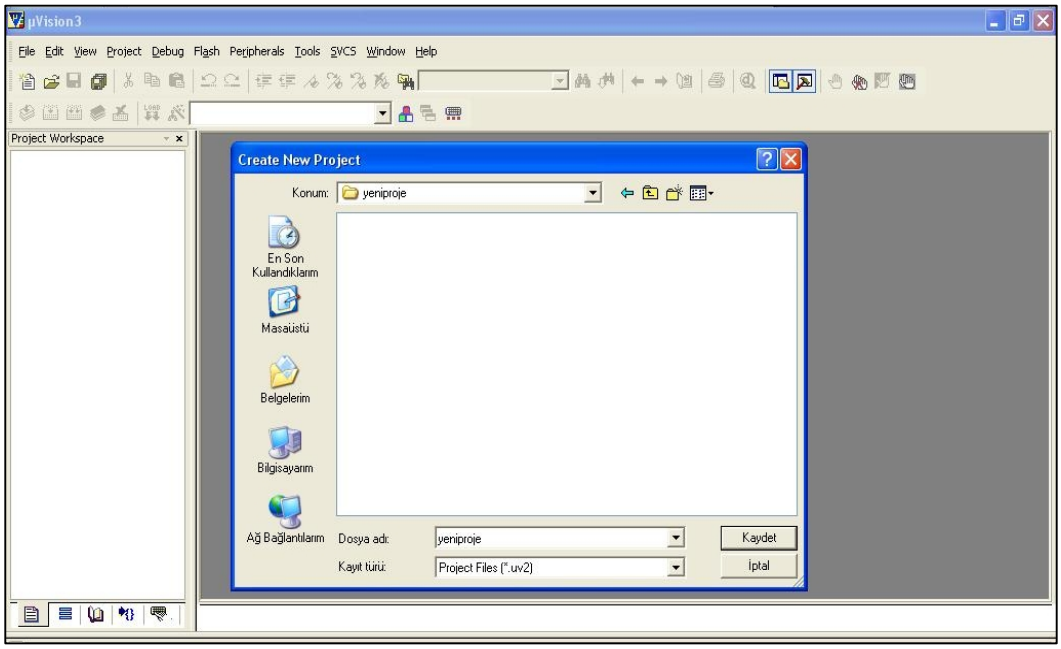
Programın yazılımı Keil µVision programı ile yapıldı. Tanımlanmış her bir entegre için kodlar parçacıklar halinde yazıldı. LCD'de gerekli mesajların görülmesi için ilave kodlar yazıldı.

4.3.3. KEIL µVision programı ile proje oluşturma



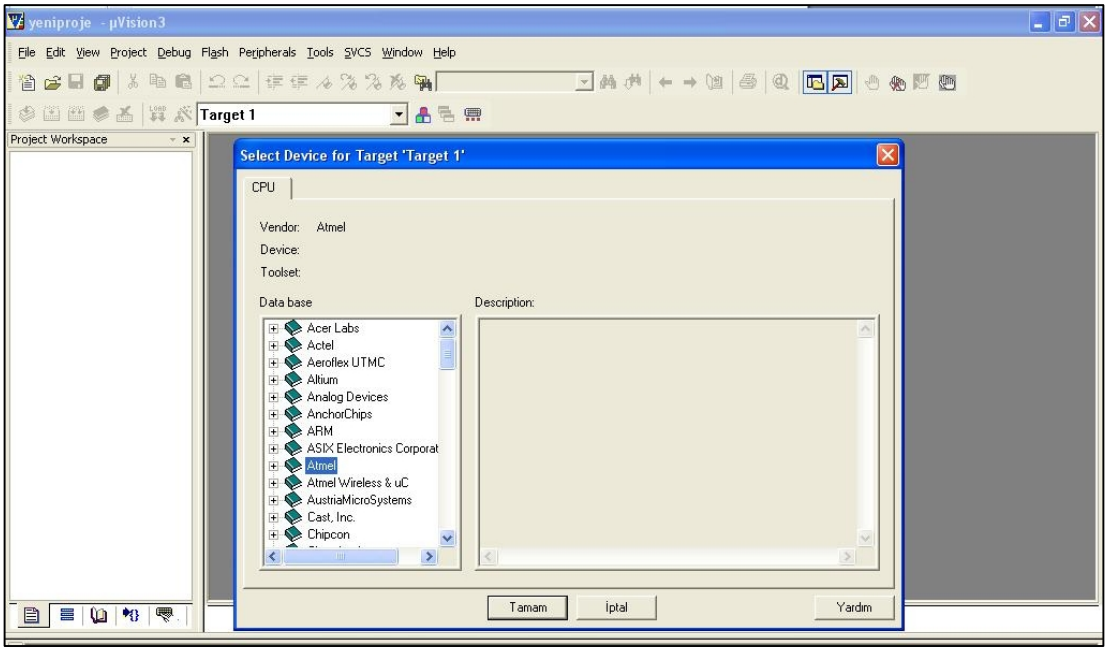
Şekil 4.10. Keil µVision'da projenin oluşturulması

Yazılan kodların derlenmesini ve ilgili *.hex dosyasının oluşturulması için gerekli aşamalar şunlardır: İlk olarak Project menüsünden New µVision Project seçilir.



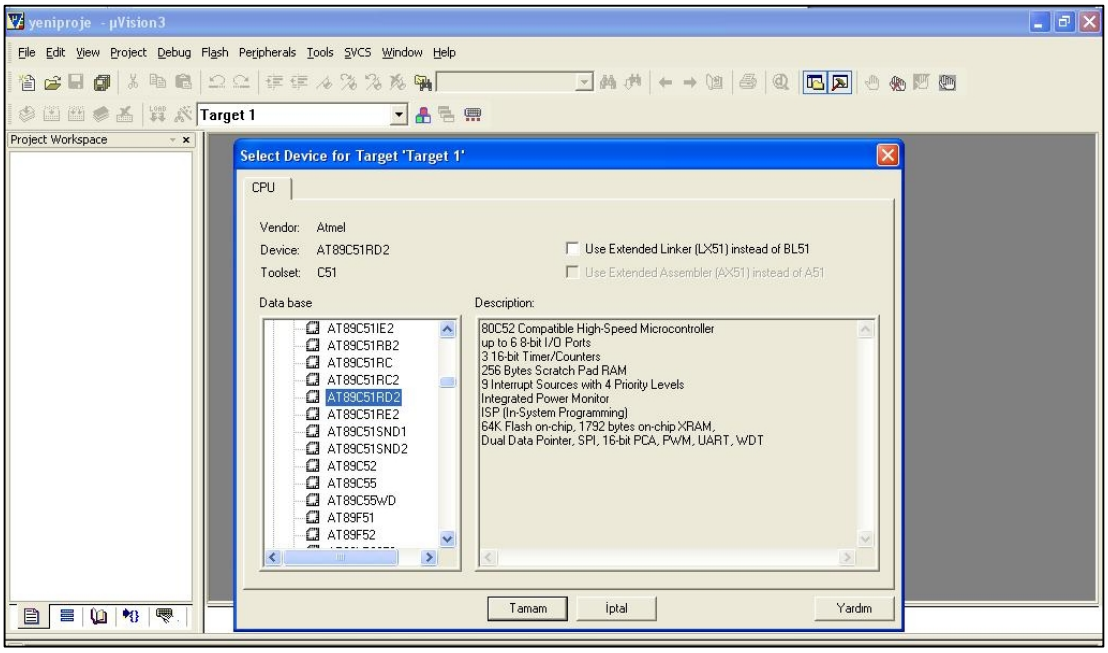
Şekil 4.11. Oluşturulacak projenin kaydedilmesi

Burada, oluşturulacak projenin yeri belirlenir ve kaydedilir. Genellikle sabit diskin C biriminde oluşturulur.



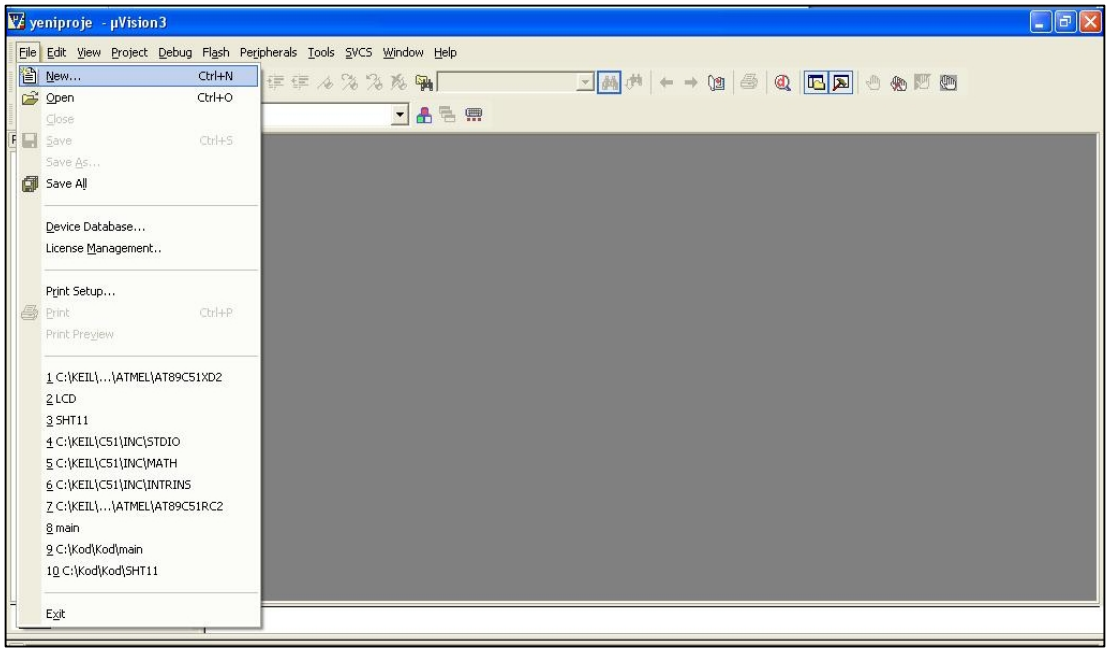
Şekil 4.12. Kullanılacak mikrodenetleyici firmasının seçilmesi

Proje oluşturulduktan sonra mikrodenetleyici firmalarının isim listesi çıkar. Buradan projede kullanılacak olan mikrodenetleyicinin firması seçilir. Kullanılacak olan mikrodenetleyici ATMEL'in AT89C51RD2 mikrodenetleyicisidir.

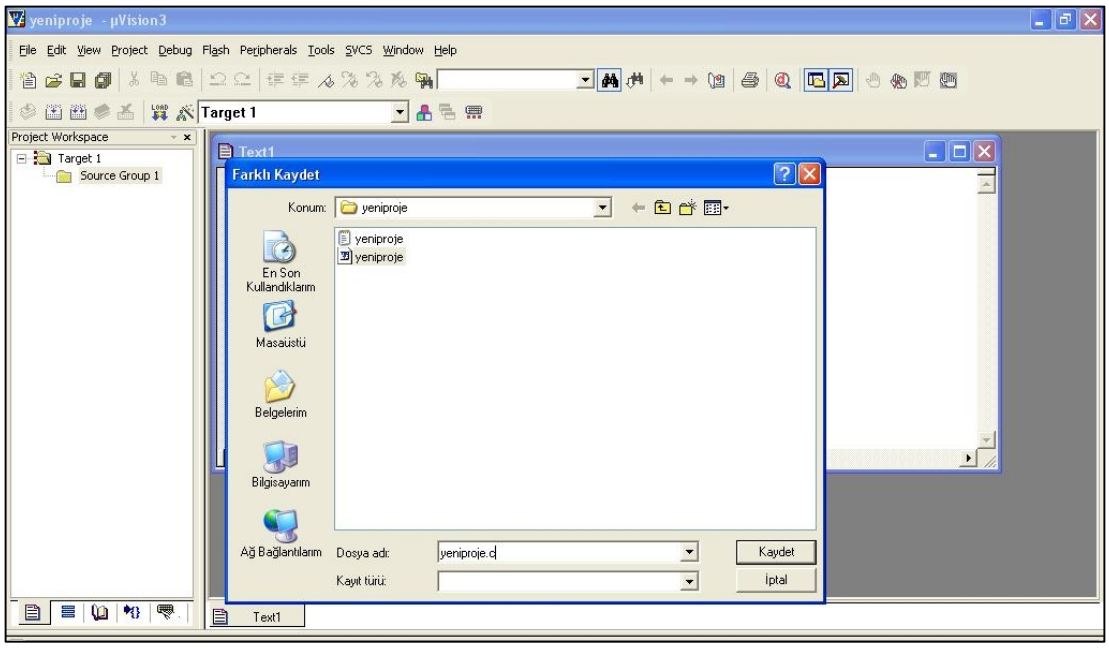


Şekil 4.13. Atmel mikrodenetleyicisinin seçilmesi

Çıkan ekranda projede kullanılacak olan mikrodenetleyici seçilir. Ardından Tamam denilerek ekrandan çıkılır.



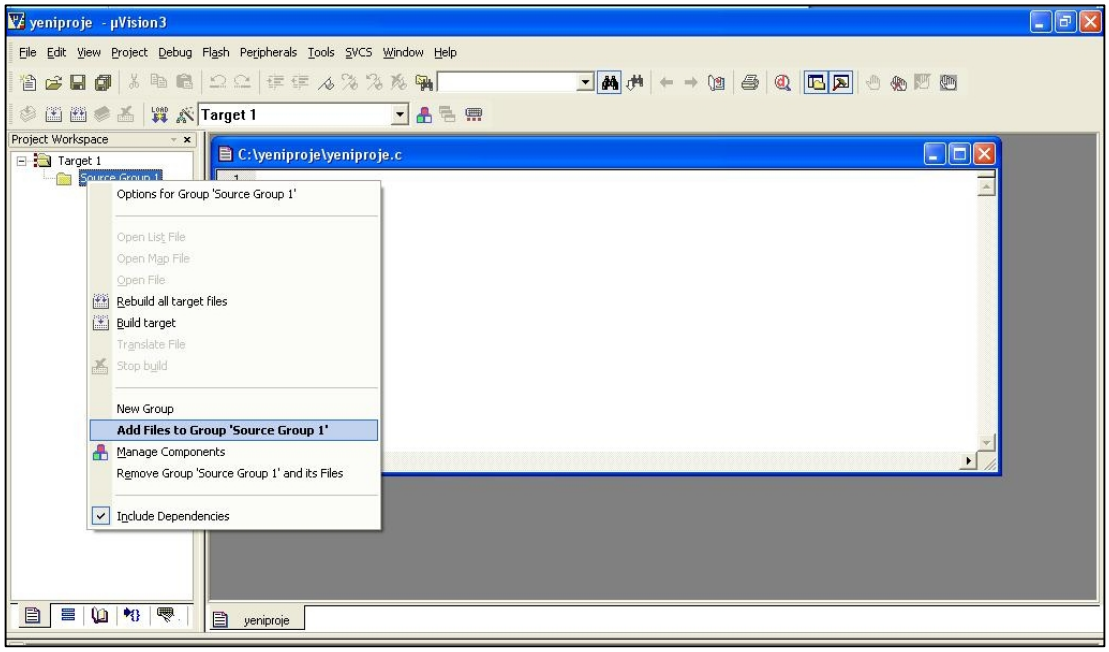
Şekil 4.14. Projede kod yazımı için dosya oluşturulması



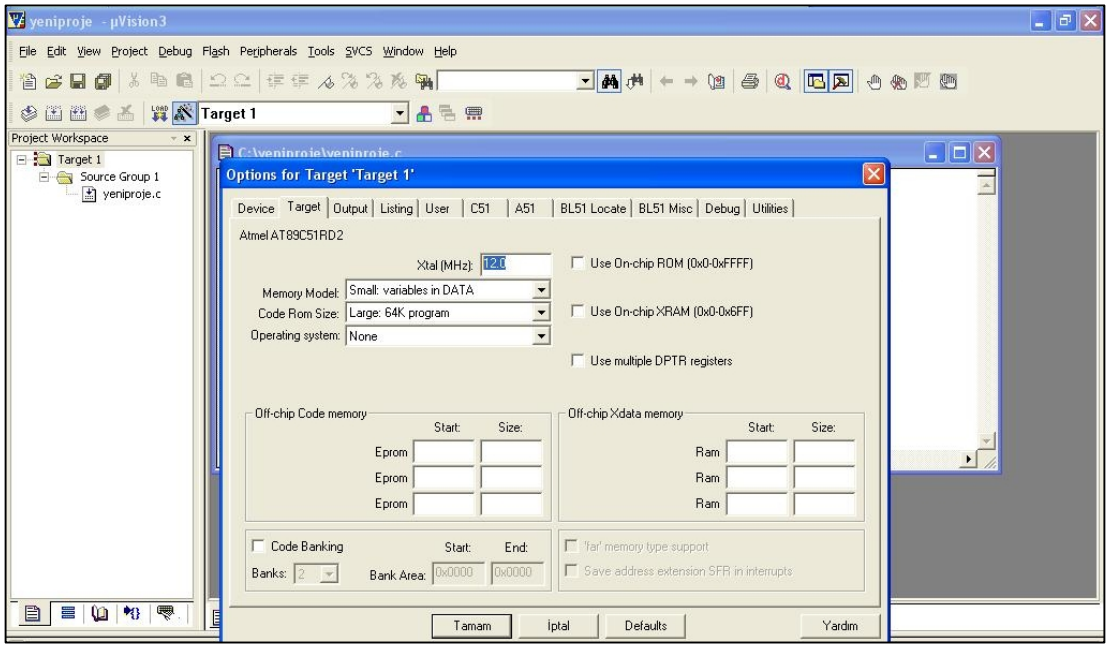
Şekil 4.15. Oluşturulacak dosyanın türünün belirlenmesi

File menüsünden New sekmesi seçilerek yeni bir dosya oluşturulur. İlk oluşturulan projenin genel bir çerçevesiydi. Bu ise kod yazımı için gerekli olan kısımdır.

Çıkan ekranda, daha önceden oluşturulmuş projenin ismi verilir ve uzantısı .c olacak şekilde kaydedilir. Asm kodları ile yazılacaksa dosyanın uzantısı .asm olarak kaydedilir.

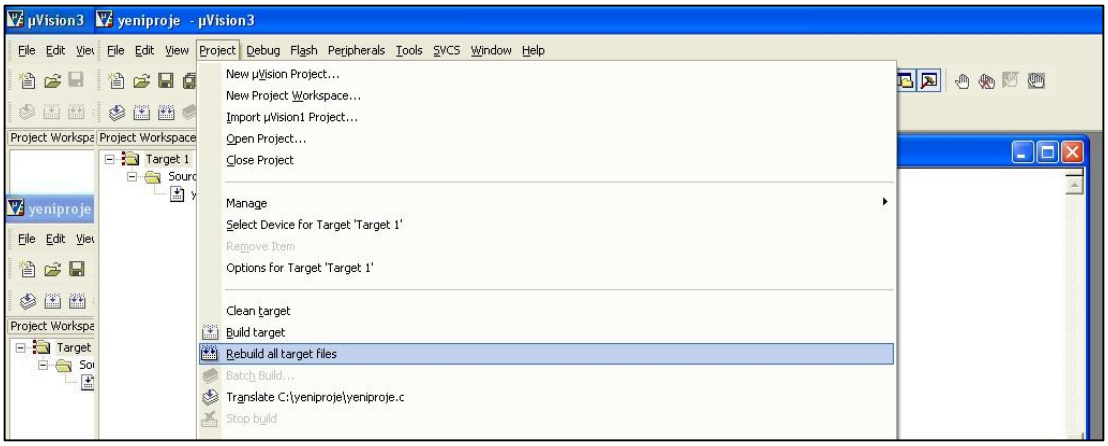


Şekil 4.16. Dosyanın projeye eklenmesi



Şekil 4.17. Son ayarların yapılması

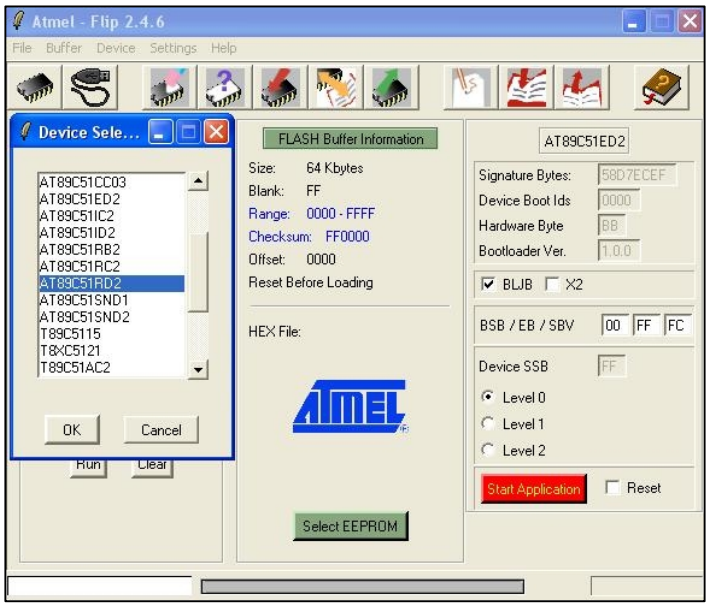
Burada ise Target1 yazan yerdeki kısmın yanında bulunan sekme tıklanarak resimdeki ekran gelir. Bu ekranda Target sekmesinden mikrodenetleyicinin çalışma frekansı seçilir. Output sekmesinden ise yazılmış olan kodun hex dosyasının üretilmesi için Create Hex File'ın yanındaki kutucuk işaretlenir. Tamam denilerek çıkılır.



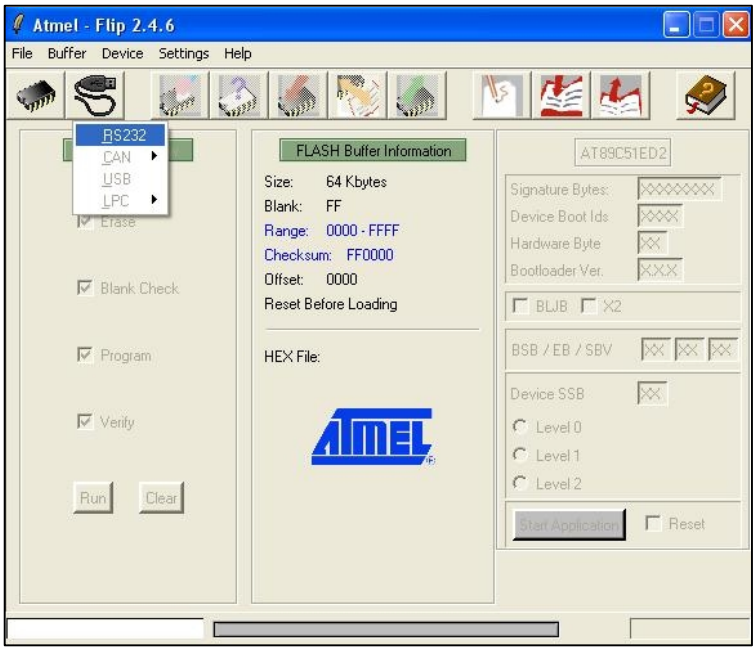
Şekil 4.18. Projenin derlenmesi

Son olarak Project menüsünden Build All Target Files seçilir ve proje derlenir. Hata yoksa projenin hex dosyası üretilir. Eğer hata varsa gerekli uyarılar dikkate alınarak düzeltmeler yapılır. Yeniden çalıştırılır.

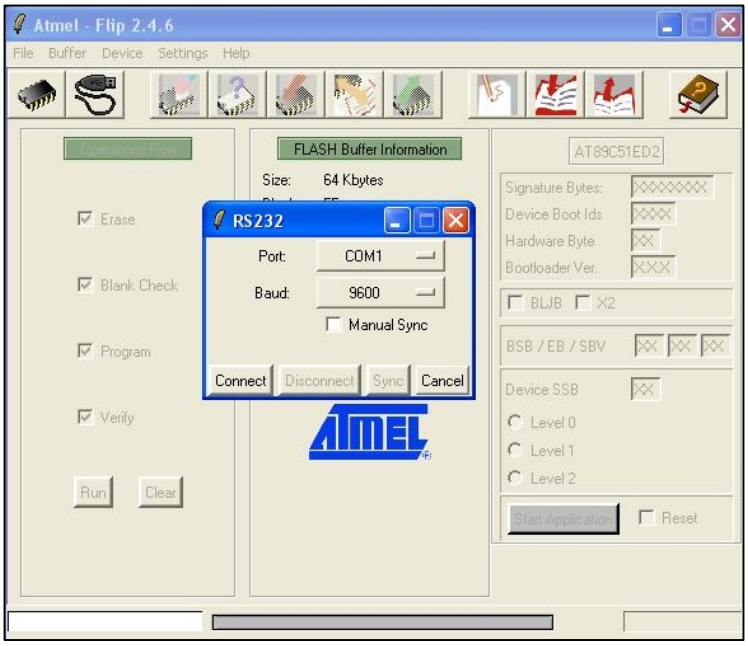
Daha sonra oluşturulmuş olan hex dosyasının mikrodenetleyiciye aktarılması için Atmel Flip programı kullanılır. Bu program sayesinde hex dosyaları kolaylıkla mikrodenetleyiciye yüklenebilir.



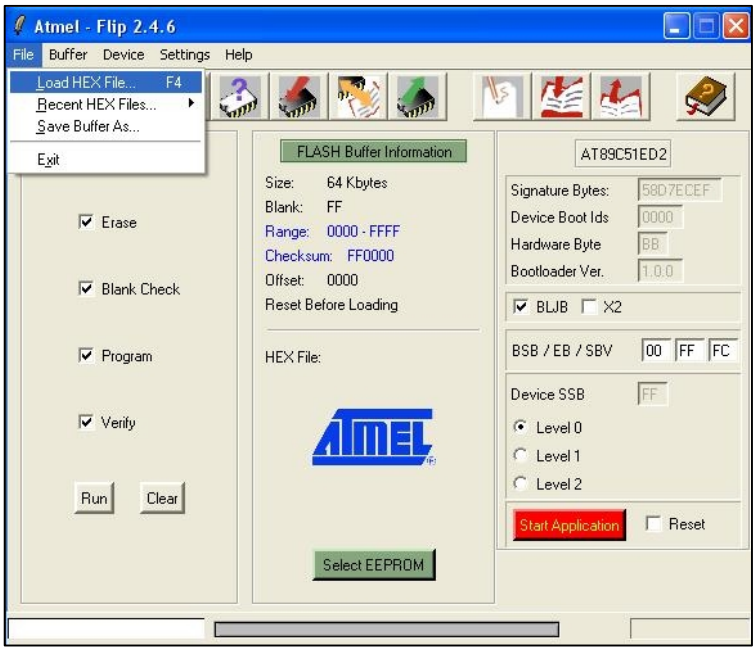
Şekil 4.19. Kullanılan mikrodenetleyicinin seçilmesi



Şekil 4.20. Seri porttan yükleme için seçimin yapılması

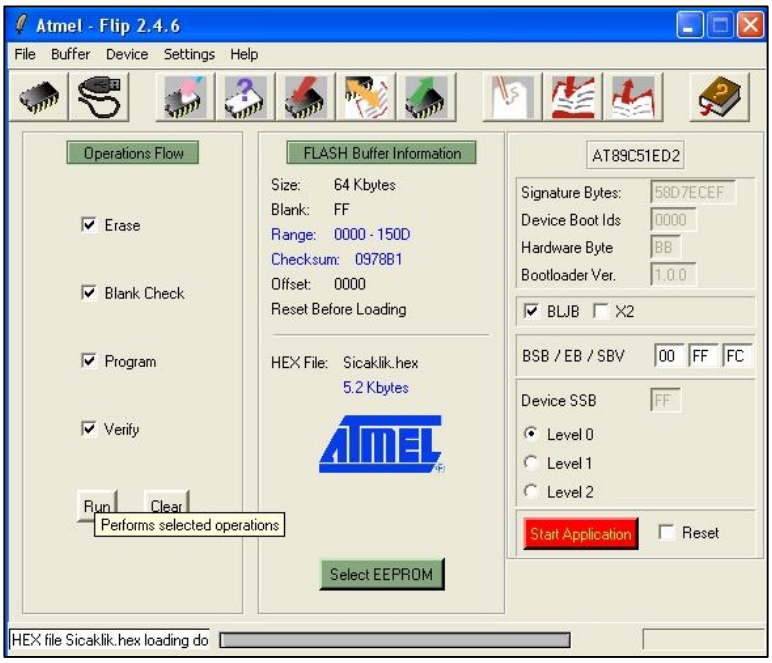


Şekil 4.21. Portun ve baud hızının seçilmesi

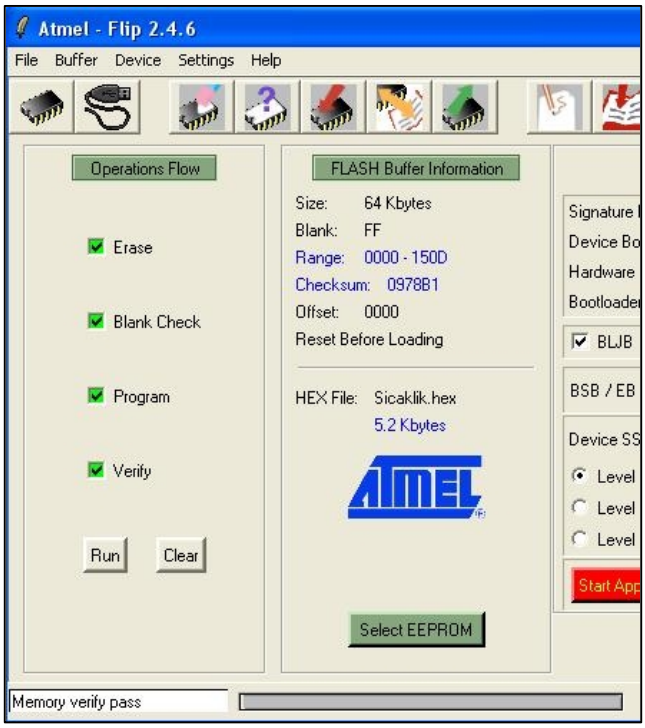


Şekil 4.22. Hex dosyasının yüklenmesi

Keil programı ile oluşturulmuş olan hex dosyası seçilerek programa eklenir.

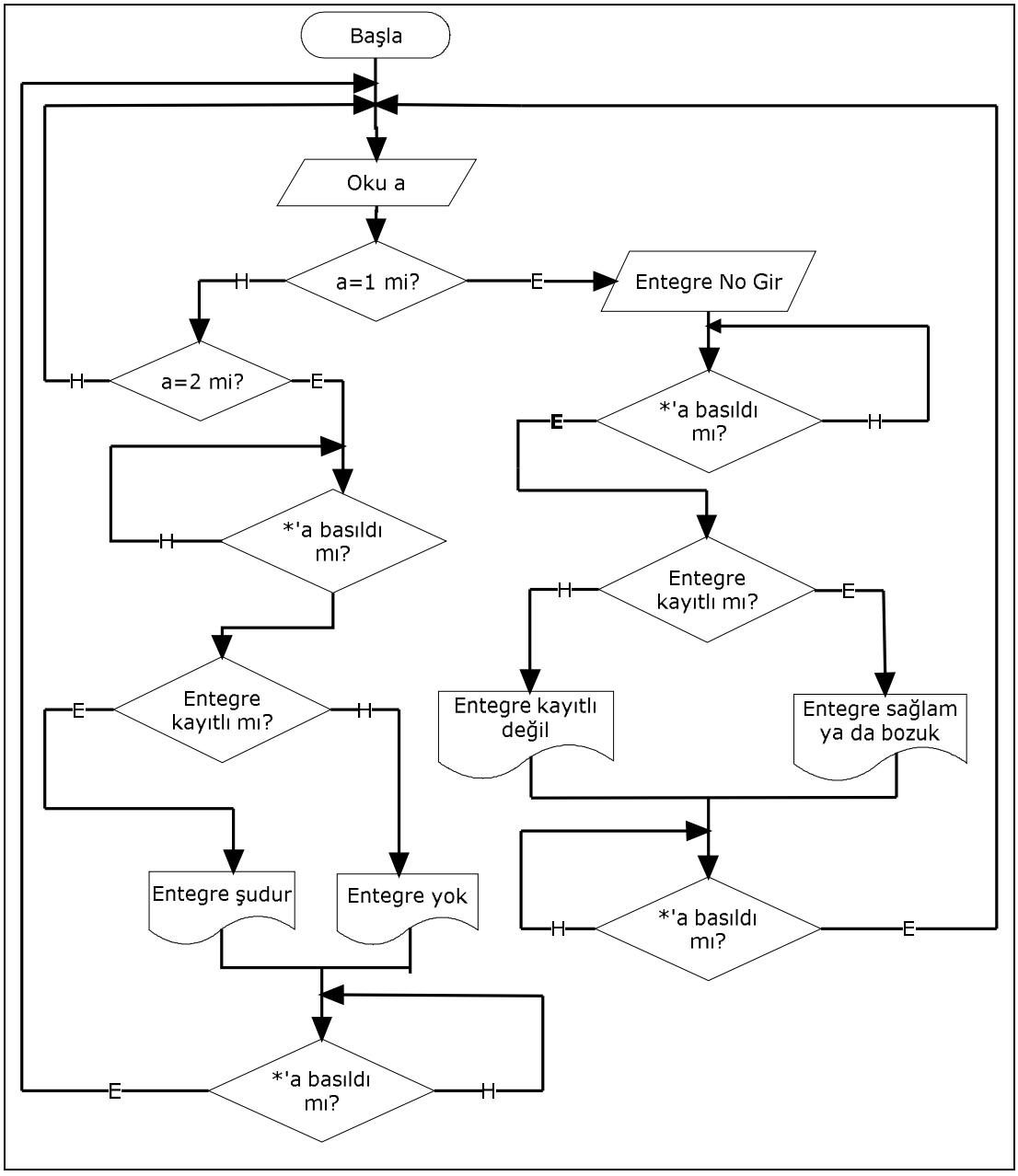


Şekil 4.23. Dosyanın mikrodenetleyiciye yüklenmeye hazır olması



Şekil 4.24. Dosyanın mikrodenetleyiciye yüklenmesinin tamamlanması

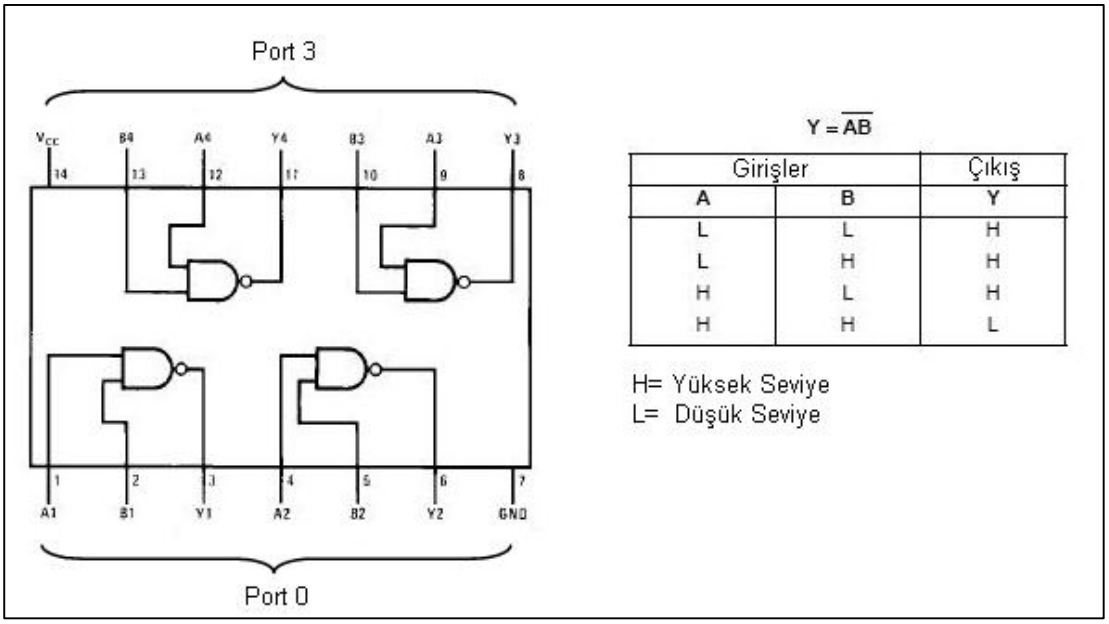
Resimdeki gibi kutucuklar yeşil ise dosya sorunsuz bir şekilde mikrodenetleyiciye aktarılmıştır ve program çalıştırılmaya hazırdır. 8051 kartında gerekli ayarlamalar yapıldıktan sonra program, ilgili görevini yerine getirmeye başlayacaktır.



Şekil 4.25. Uygulamanın akış diyagramı

4.4. Entegre Test Etme

Cihazdaki tuş takımından 1 tuşuna basılarak entegrenin sağlamlığı, 2 tuşuna basılarak ise entegrenin seri numarası öğrenilir. Onaylama tuşu ise *(yıldız)'dır. Eğer entegre numarası yanlış yazılmışsa silmek için de #(diyez ya da kare) tuşuna basılır.



Şekil 4.26. 7400 entegresinin blok şeması ve doğruluk tablosu

Yukarıda bulunan 7400 entegresinin blok şeması ve doğruluk tablosu görülmektedir. Örnek olarak bu entegre için kodlar aşağıdaki gibi yazılmıştır. Ekler kısmında ise bütün kodlar eklenmemiştir.

```
git7400();
        if(c1==4){
                c1=0;
                LCD_clear();
                LCD_row1(); LCD_puts("7400");
        }
        else{
                c1=0;
                a++;
        }
}
```

Programın en başında değişken tanımlamaları yapıldıktan sonra ilgili entegrenin kodları yukarıdaki gibi yazılır. Bu kısın entegre numarasını öğrenmek için kullanılan bölümdür. İlk önce 7400 entegresinin bulunduğu global fonksiyona gidilir ve ilgili sağlamalar yapıldıktan sonra geri dönülür. Eğer yapılan sağlamada sayaç değeri 4 ise entegre numarası 7400 olarak lcd de gösterilir.. Eğer sayaç 4 değilse diğer fonksiyonlara geçer ve hiçbirisi sağlamazsa entegre numarası yok der.

```
case 7400:
        git7400();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
```

Bu kısım ise entegrenin sağlam mı bozuk mu olduğunu anlamak için kullanılır. Tuş takımından 7400 seri numarası girildikten sonra *(yıldız) tuşuna basılır ve test etme işlemi başlar. Yine ilgili fonksiyona gider ve sayaç değeri 4'e eşitse entegre sağlam, değilse entegre bozuk mesajını lcd ye gönderir.

```
void git7400(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=0;P3_4=1;P3_5=0;P0_0=1;P0_1=0;P0_3=1;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=1;P3_4=0;P3_5=1;P0_0=0;P0_1=1;P0_3=0;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
```

```
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_4=0;P3_5=0;P0_0=0;P0_1=0;P0_3=0;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){

                c1++;
        }
}
```

Bu bölümde ise tanımlanmış entegre için giriş değerlerine yüksek ya da düşük seviyeli sinyaller verilerek çıkışları karşılaştırılır. Elde edilen değerler bir sayaca aktarılır ve sonra ilgili fonksiyonu gerçekleştirmek için o fonksiyona geri döner.

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Bu tezde 8051 mikrodenetleyicisi kullanarak yapılan dijital entegre test cihazı sayesinde entegrelerin sağlamlık ve seri numara bilgilerini öğrenmek pratik hale gelmektedir. Entegrelerin durumlarını öğrenmek için çeşitli devre düzenekleri kurmaya gerek yoktur. Piyasada da çeşitleri bulunmakta ancak yapılan bu cihaz sayesinde gerekli tanımlamalar yapılarak yeni programlar mikrodenetleyiciye yüklenebilmektedir. Bu işlem cihaz üzerindeki seri port sayesinde olmaktadır. Eğitim kurumlarında yaygın olarak kullanılabilirler. Özellikle dijital elektronik derslerinde yapılan kapı uygulamaları, sayıcı uygulamaları gibi çalışmalarda kullanılan entegreler rahatlıkla test edilebilir. Böylece zamandan ve maliyetten tasarruf sağlanabilir.

Yapılan uygulama, geliştirilmeye müsait bir yapıya sahiptir. Sadece 8051 mikrodenetleyicisiyle değil diğer mikrodenetleyicilerle(PIC vb.) de yapılabilir. Devredeki port sayısı arttırılarak daha fazla bacak sayısına sahip entegrelerde sağlamlık ve seri numara öğrenme işlemine tabi tutulabilir. Bilgilerin depolanmasında hafıza sorunu yaşamamak için kullanılan mikrodenetleyicinin 2 katı belleğe sahip AT89C51RE2 serisi kullanılabilir. Hafıza ile algoritmalar farklı yerlerde saklanabilirler. Örneğin; eprom, eeprom gibi. Simülasyon ortamında da devrenin çalışması gözlenebildiğinden, program gerçek ortama aktarılmadan önce gerekli ayarlamalar yapılabilir. Cihaz bir adaptör sayesinde beslenmektedir. Ancak gerekli ayarlamalar yapılarak pille çalıştırılması da sağlanabilir.

KAYNAKLAR

[1] ÖZCERİT, A. T., ÇAKIROĞLU, M., BAYILMIŞ, C., 8051 Mikrodenetleyici Uygulamaları, Papatya Yayıncılık, İstanbul, Ekim 2005.

[2] EKİZ, H., Mantık Devreleri, Değişim Yayınları, Adapazarı, Ocak 2003.

[3] TAŞBAŞI, G. M., İleri C Programlama, Altaş Yayıncılık, İstanbul, Eylül 2003.

[4] http://rabbit.eng.miami.edu/info/datasheets, Mart 2009.

[5] http://www.kpsec.freeuk.com/compon.htm, Mart 2009.

[6] http://www.datasheetcatalog.com, Şubat 2009.

[7] http://www.datasheetarchive.com, Şubat 2009.

[8] http://www.alldatasheet.com, Şubat 2009.

[9] http://electrofriends.com/projects/microcontrollers/digital-ic-tester, Şubat 2009.

[10] http://www.samengstrom.com/nxl/9571/cmos_gates_page.en.html, Mart 2009.

[11] http://amir8797.blogfa.com/post-26.aspx, Nisan 2009.

[12] http://www.thesatya.com/ic.html, Mart 2009.

[13] http://www.keil.com, Ekim 2008.

[14] http://www.labcenter.co.uk/index.cfm, Ekim 2008.

[15] http://www.atmel.com, Ekim 2008.

[16] http://tr.wikipedia.org/wiki/8051, Mart 2009.

[17] http://www.bkprecision.com/products/, Nisan 2009.

[18] PAÇ, M. R., PIC1.pdf, Temmuz 2003.

EKLER

EK A. Program Kaynak Kodu

```
#include <at89c51xd2.h>

#include <stdio.h>
#include "lcd.h"

int tusoku();
int test(int );
int testa();
void gecikme();
void gecikme1();
void clock1();
void clock2();
void clock3();
void clock4();
void clock5();
void clock6();
void msj_saglam();
void msj_bozuk();
void git7400();
void git7401();
void git7402();
void git7404();
void git7407();
void git7408();
void git7410();
void git7432();
void git7436();
void git7447();
void git7448();
void git7449();
void git7476();
void git74138();
void git4000();
void git4001();
void git4002();
void git4010();
void git4011();
```

```
void git4055();
void git4502();
void git4516();

#define A       P1_0
#define B       P1_1
#define C       P1_2
#define D       P1_3
#define buton P1_4
int c1=0;

main(){
        LCD_init();
        LCD_row1(); LCD_puts("DIJITAL ENTEGRE");
        LCD_row2(); LCD_puts("  TEST CIHAZI ");
        gecikme();

while(1){

        unsigned int bas1,bas2,bas3,bas4,bas5,bas6,top=0,basla,k=0;
        char char_cikis[10];

        LCD_clear();
        LCD_row1(); LCD_puts("1- TEST ETME");
        LCD_row2(); LCD_puts("2- ENTEGRE NO");
        gecikme();
////////////////////////////////////////////////////////////////////////////
geri0:bas1=tusoku();
if(bas1==0 || bas1==3 || bas1==4 || bas1==5 || bas1==6 || bas1==7 || bas1==8 ||
  bas1==9 || bas1==10 || bas1==11){
  goto geri0;
  }
if(bas1==2){
//      P0=0x00; P3=0x00;
        LCD_clear();
        LCD_row1(); LCD_puts("ENTEGREYI TAKIN");
        LCD_row2(); LCD_puts("VE *'a BASIN");

        if(bas1==11){
                LCD_row2(); LCD_puts(" ");
                goto geri0;
        }
         else{
         top=bas1;
         sprintf(char_cikis,"%d",bas1);
}

xyz:bas1=tusoku();
        if(bas1==10){
```

```
                //P0=0xFF; P3=0xFF;
                LCD_clear();
                LCD_row1(); LCD_puts("ENTEGRE");
            LCD_row2(); LCD_puts("TEST EDILIYOR");
                gecikme();
                testa();
                goto son1;
        }
        else{
        goto xyz;
        }
son1:basla=tusoku();
        if(basla!=10)
        goto son1;

}

if(bas1==1){
        //P0=0x00; P3=0x00;
        LCD_clear();
        LCD_row1(); LCD_puts("ENTEGRE NO GIR");
        LCD_row2(); LCD_puts("VE *'a BASIN");

////////////////////////////////////////////////////////
geri1:bas1=tusoku();
if(bas1==10 || bas1==11){
        goto geri1;
}
        else{
                if(bas1==0){
                        goto geri1;
                }
                else{
                top=bas1;
                sprintf(char_cikis,"%d",bas1);
                LCD_clear();
                LCD_row1(); LCD_puts("ENTEGRE NO");
        LCD_row2(); LCD_puts(char_cikis);
                }
        }

////////////////////////////////////////////////////////
geri2:bas2=tusoku();
        if(bas2==10){
                goto geri2;
}
        if(bas2==11){
                LCD_row2(); LCD_puts(" ");
                goto geri1;
```

```
}
        else{
                top=(bas1*10)+bas2;
                sprintf(char_cikis,"%d",bas2);
        LCD_row2_1(); LCD_puts(char_cikis);
        }
//////////////////////////////////////////////////////////////////
geri3:bas3=tusoku();
        if(bas3==10){
                goto geri3;
}
        if(bas3==11){
                LCD_row2_1();LCD_puts(" ");
                goto geri2;
}
        else{
                top=(bas1*100)+(bas2*10)+bas3;
                sprintf(char_cikis,"%d",bas3);
        LCD_row2_2(); LCD_puts(char_cikis);
        }
//////////////////////////////////////////////////////////////////////
geri4:bas4=tusoku();
        if(bas4==10){
                goto geri4;
}
        if(bas4==11){
                LCD_row2_2();LCD_puts(" ");
                goto geri3;
}
        else{
                top=(bas1*1000)+(bas2*100)+(bas3*10)+bas4;
                sprintf(char_cikis,"%d",bas4);
        LCD_row2_3(); LCD_puts(char_cikis);
        }
//////////////////////////////////////////////////////////////////////////
geri5:bas5=tusoku();
        if(bas5==11){
                LCD_row2_3();LCD_puts(" ");
                goto geri4;
}
        if(bas5==10){
                LCD_clear();
                LCD_row1(); LCD_puts("ENTEGRE");
                LCD_row2(); LCD_puts("TEST EDILIYOR");
                gecikme();
                test(top);
                goto son;
}
        else{
```

```
                top=(bas1*10000)+(bas2*1000)+(bas3*100)+(bas4*10)+bas5;
                sprintf(char_cikis,"%d",bas5);
        LCD_row2_4(); LCD_puts(char_cikis);
        }
////////////////////////////////////////////////////////////////////////////
geri6:bas6=tusoku();
        if(bas6==10){
                LCD_clear();
                LCD_row1(); LCD_puts("ENTEGRE");
                LCD_row2(); LCD_puts("TEST EDILIYOR");
                gecikme();
                test(top);
                goto son;
                }
        if(bas6==11){
                LCD_row2_4();LCD_puts(" ");
                goto geri5;
                }
                else{
                goto geri6;
                }
        son:basla=tusoku();
                if(basla!=10)
                goto son;
        }
  }
 }

//========================== Tus Takimindan sayilarin
okunmasi============================
int tusoku(){
char a=0;
int cikis;
P1=0xFF;

while(a==0){

if(!buton){
        while(!buton);

if(A==0 && B==0 && C==0 && D==0){
                cikis=1; gecikme();
                a=1;
}
      if(a==1)break;

if(A==1 && B==0 && C==0 && D==0){
                cikis=2; gecikme();
                a=1;
```

```
}
                if(a==1)break;

if(A==0 && B==1 && C==0 && D==0){
                cikis=3; gecikme();
                a=1;
}
                if(a==1)break;

if(A==0 && B==0 && C==1 && D==0){
                cikis=4; gecikme();
                a=1;
}
                if(a==1)break;

if(A==1 && B==0 && C==1 && D==0){
                cikis=5; gecikme();
                a=1;
}
                if(a==1)break;

if(A==0 && B==1 && C==1 && D==0){
                cikis=6; gecikme();
                a=1;
}
                if(a==1)break;

if(A==0 && B==0 && C==0 && D==1){
                cikis=7; gecikme();
                a=1;
}
                if(a==1)break;

if(A==1 && B==0 && C==0 && D==1){
                cikis=8; gecikme();
                a=1;
}
                if(a==1)break;

if(A==0 && B==1 && C==0 && D==1){
                cikis=9; gecikme();
                a=1;
}
                if(a==1)break;

if(A==0 && B==0 && C==1 && D==1){
                cikis=10; gecikme();
                a=1;
}
```

```
                    if(a==1)break;

if(A==1 && B==0 && C==1 && D==1){
                    cikis=0; gecikme();
                    a=1;
}
                    if(a==1)break;

if(A==0 && B==1 && C==1 && D==1){
                    cikis=11; gecikme();
                    a=1;
}
                    if(a==1)break;
          }
}
      return cikis;
}

void clock1(){
          P0_0=1;
          }
void clock2(){
          P0_0=0;
          }
void clock3(){
          P0_5=1;
}
void clock4(){
          P0_5=0;
}
void clock5(){
          P3_1=1;
}
void clock6(){
          P3_1=0;
}

void msj_saglam(){
          LCD_clear();
          LCD_row1(); LCD_puts("ENTEGRE SAGLAM");
}
void msj_bozuk(){
          LCD_clear();
          LCD_row1(); LCD_puts("ENTEGRE BOZUK");
}

void gecikme(){
unsigned int k;
          for(k=0;k<5000;k++);
```

```
}
void gecikme1(){
unsigned int k;
        for(k=0;k<45000;k++);
}
//============================ Entegre Bilgileri
==================================

int testa(){
        int a=0;

        if(a==0){
                git7400();
                if(c1==4){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7400-7403");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==1){
                git7401();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7401");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==2){
                git7402();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("Asagidakilerden");
                        LCD_row2(); LCD_puts("Birisi");
                        gecikme1();
                        LCD_clear();
                        LCD_row1(); LCD_puts("7402");
                        }
                else{
                        c1=0;
                        a++;
```

```
                }
}
        if(a==3){
                git7404();
                if(c1==2){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7404-7405-7406");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==4){
                git7407();
                if(c1==2){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7407");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==5){
                git7408();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7408-7409 ");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==6){
                git7410();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7410");
                        }
                else{
                        c1=0;
                        a++;
                }
```

```
}
        if(a==10){
                git7436();
                if(c1==4){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7436");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==12){
                git7447();
                if(c1==6){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7446-7447");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==14){
                git7432();
                if(c1==4){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7432");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==19){
                git7448();
                if(c1==4){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("7448");
                        }
                else{
                        c1=0;
                        a++;
                }
}
```

```
if(a==20){
		git7449();
		if(c1==4){
			c1=0;
			LCD_clear();
			LCD_row1(); LCD_puts("7449");
			}
		else{
			c1=0;
			a++;
		}
}
	if(a==23){
		git7476();
		if(c1==4){
			c1=0;
			LCD_clear();
			LCD_row1(); LCD_puts("7476");
			}
		else{
			c1=0;
			a++;
		}
}
	if(a==26){
		git74138();
		if(c1==9){
			c1=0;
			LCD_clear();
			LCD_row1(); LCD_puts("74138");
			}
		else{
			c1=0;
			a++;
		}
}

	if(a==28){
		git4000();
		if(c1==3){
			c1=0;
			LCD_clear();
			LCD_row1(); LCD_puts("4000");
			}
		else{
			c1=0;
			a++;
		}
}
```

```
        if(a==29){
                git4001();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4001");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==30){
                git4002();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4002");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==33){
                git4010();
                if(c1==2){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4010");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==34){
                git4011();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4011");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==37){
```

```
                git4055();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4055");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==38){
                git4502();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4502");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==40){
                git4516();
                if(c1==3){
                        c1=0;
                        LCD_clear();
                        LCD_row1(); LCD_puts("4516");
                        }
                else{
                        c1=0;
                        a++;
                }
}
        if(a==41){
                LCD_clear();
                LCD_row1(); LCD_puts("ENTEGRE YOK");
        }
}

//////////////////////////////////////////////////////////////////////////
int test(int top){
        switch(top){

        case 7400:
                git7400();
                if(c1==4){
                        c1=0;
```

```
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }

case 7401:
        git7401();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }

case 7402:
        git7402();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }

case 7403:
        git7400();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }

case 7404:
        git7404();
        if(c1==2){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
```

```
        }
case 7405:
        git7404();
        if(c1==2){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7406:
        git7404();
        if(c1==2){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7407:
        git7407();
        if(c1==2){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7408:
        git7408();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }

case 7409:
        git7408();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
        else{
```

```
                c1=0;
                msj_bozuk();break;
        }

case 7410:
        git7410();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7432:
        git7432();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7436:
        git7436();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7446:
        git7447();
        if(c1==6){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7447:
        git7447();
        if(c1==6){
                c1=0;
                msj_saglam();break;
```

```
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7448:
        git7448();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7449:
        git7449();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 7476:
        git7476();
        if(c1==4){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 74138:
        git74138();
        if(c1==9){
                c1=0;
                msj_saglam();break;
        }
        else{
                c1=0;
                msj_bozuk();break;
        }
case 4000:
        git4000();
        if(c1==3){
                c1=0;
```

```
                msj_saglam();break;
        }
         else{
                c1=0;
                msj_bozuk();break;
        }
case 4001:
        git4001();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
         else{
                c1=0;
                msj_bozuk();break;
        }
case 4002:
        git4002();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
         else{
                c1=0;
                msj_bozuk();break;
        }
case 4010:
        git4010();
        if(c1==2){
                c1=0;
                msj_saglam();break;
        }
         else{
                c1=0;
                msj_bozuk();break;
        }
case 4011:
        git4011();
        if(c1==3){
                c1=0;
                msj_saglam();break;
        }
         else{
                 c1=0;
                msj_bozuk();break;
        }
case 4055:
        git4055();
        if(c1==3){
```

```
                        c1=0;
                        msj_saglam();break;
                }
                 else{
                 c1=0;
                        msj_bozuk();break;
                }
        case 4502:
                git4502();
                if(c1==3){
                        c1=0;
                        msj_saglam();break;
                }
                 else{
                 c1=0;
                        msj_bozuk();break;
                }
        case 4516:
                git4516();
                if(c1==3){
                        c1=0;
                        msj_saglam();break;
                }
                 else{
                 c1=0;
                        msj_bozuk();break;
                }
        default:
        LCD_clear();
        LCD_row1(); LCD_puts("YANLIS ENTEGRE");
                LCD_row2(); LCD_puts("NUMARASI");
}
}
void git7400(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=0;P3_4=1;P3_5=0;P0_0=1;P0_1=0;P0_3=1;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
```

```
        P3_1=0;P3_2=1;P3_4=0;P3_5=1;P0_0=0;P0_1=1;P0_3=0;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_4=0;P3_5=0;P0_0=0;P0_1=0;P0_3=0;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
}
void git7401(){
        P0=0xFF;P3=0xFF;
        P3_2=0;P3_3=1;P3_5=0;P3_6=1;P0_1=0;P0_2=1;P0_4=0;P0_5=1;P0_6=0;P
3_0=1;
        if(P3_1==1 & P3_4==1 & P0_0==1 & P0_3==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=1;P3_3=1;P3_5=1;P3_6=1;P0_1=1;P0_2=1;P0_4=1;P0_5=1;P0_6=0;P
3_0=1;
        if(P3_1==0 & P3_4==0 & P0_0==0 & P0_3==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=1;P3_3=0;P3_5=1;P3_6=0;P0_1=1;P0_2=0;P0_4=1;P0_5=0;P0_6=0;P
3_0=1;
        if(P3_1==1 & P3_4==1 & P0_0==1 & P0_3==1){
                c1++;
        }
}
void git7402(){
        P0=0xFF;P3=0xFF;
        P3_2=0;P3_3=1;P3_5=0;P3_6=1;P0_1=0;P0_2=1;P0_4=0;P0_5=1;P0_6=0;P
3_0=1;
        if(P3_1==0 & P3_4==0 & P0_0==0 & P0_3==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=0;P3_3=0;P3_5=0;P3_6=0;P0_1=0;P0_2=0;P0_4=0;P0_5=0;P0_6=0;P
3_0=1;
        if(P3_1==1 & P3_4==1 & P0_0==1 & P0_3==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=1;P3_3=0;P3_5=1;P3_6=0;P0_1=1;P0_2=0;P0_4=1;P0_5=0;P0_6=0;P
3_0=1;
        if(P3_1==0 & P3_4==0 & P0_0==0 & P0_3==0){
```

```
                c1++;
        }
}
void git7404(){
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_3=0;P3_5=0;P0_0=0;P0_2=0;P0_4=0;P0_6=0;P3_0=1;
        if(P3_2==1 & P3_4==1 &  P3_6==1 & P0_1==1 & P0_3==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_3=1;P3_5=1;P0_0=1;P0_2=1;P0_4=1;P0_6=0;P3_0=1;
        if(P3_2==0 & P3_4==0 &  P3_6==0 & P0_1==0 & P0_3==0 & P0_5==0){
                c1++;
        }
}
void git7407(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_3=1;P3_5=1;P0_0=1;P0_2=1;P0_4=1;P0_6=1;P3_0=1;
        if(P3_2==1 & P3_4==1 &  P3_6==1 & P0_1==1 & P0_3==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_3=0;P3_5=0;P0_0=0;P0_2=0;P0_4=0;P0_6=0;P3_0=1;
        if(P3_2==0 & P3_4==0 &  P3_6==0 & P0_1==0 & P0_3==0 & P0_5==0){
                c1++;
        }
}
void git7408(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=0;P3_4=1;P3_5=0;P0_0=1;P0_1=0;P0_3=1;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=1;P3_4=0;P3_5=1;P0_0=0;P0_1=1;P0_3=0;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
}
void git7410(){
        P0=0xFF;P3=0xFF;
```

```
        P3_1=1;P0_0=0;P0_1=0;P0_2=1;P0_3=0;P0_4=0;P3_3=1;P3_4=0;P3_5=0;P
0_6=0;P3_0=1;
        if(P3_2==1 & P3_6==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P0_0=0;P0_1=0;P0_2=0;P0_3=0;P0_4=0;P3_3=0;P3_4=0;P3_5=0;P
0_6=0;P3_0=1;
        if(P3_2==1 & P3_6==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P0_0=1;P0_1=1;P0_2=1;P0_3=1;P0_4=1;P3_3=1;P3_4=1;P3_5=1;P
0_6=0;P3_0=1;
        if(P3_2==0 & P3_6==0 & P0_5==0){
                c1++;
        }
}
void git7432(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=1;P3_4=0;P3_5=1;P0_0=0;P0_1=1;P0_3=0;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_4=0;P3_5=0;P0_0=0;P0_1=0;P0_3=0;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
}
void git7436(){
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=1;P3_4=0;P3_5=1;P0_0=0;P0_1=1;P0_3=0;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
```

```
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_4=0;P3_5=0;P0_0=0;P0_1=0;P0_3=0;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_6==1 & P0_2==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_4=1;P3_5=1;P0_0=1;P0_1=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_6==0 & P0_2==0 & P0_5==0){
                c1++;
        }
}
void git7447(){
        P0=0xFF;P3=0xFF;
        P0_6=1;P0_0=0;P0_1=0;P0_5=0;P0_2=1;P0_3=1;P0_4=1;P0_7=0;P3_0=1;//
siralama a,b,c,d,lt,bi,rbi,gnd,vcc
        if(P3_3==1 & P3_4==0 & P3_5==0 & P3_6==1 & P3_7==1 & P3_1==1 &
P3_2==1){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=0;P0_1=0;P0_5=0;P0_2=0;P0_3=1;P0_4=0;P0_7=0;P3_0=1;//
siralama a,b,c,d,lt,bi,rbi,gnd,vcc
        if(P3_3==0 & P3_4==0 & P3_5==0 & P3_6==0 & P3_7==0 & P3_1==0 &
P3_2==0){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=0;P0_1=0;P0_5=0;P0_2=1;P0_3=0;P0_4=0;P0_7=0;P3_0=1;//
siralama a,b,c,d,lt,bi,rbi,gnd,vcc
        if(P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 & P3_7==1 & P3_1==1 &
P3_2==1){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=0;P0_1=0;P0_5=1;P0_2=1;P0_3=1;P0_4=1;P0_7=0;P3_0=1;//
8
        if(P3_3==0 & P3_4==0 & P3_5==0 & P3_6==0 & P3_7==0 & P3_1==0 &
P3_2==0){
                c1++;
```

```
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=0;P0_1=0;P0_5=0;P0_2=1;P0_3=1;P0_4=1;P0_7=0;P3_0=1;
        //0
        if(P3_3==0 & P3_4==0 & P3_5==0 & P3_6==0 & P3_7==0 & P3_1==0 &
P3_2==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=1;P0_1=1;P0_5=0;P0_2=1;P0_3=1;P0_4=1;P0_7=0;P3_0=1;
        //6
        if(P3_3==1 & P3_4==1 & P3_5==0 & P3_6==0 & P3_7==0 & P3_1==0 &
P3_2==0){
                c1++;
        }
}
void git7448(){
        P0=0xFF;P3=0xFF;
        P0_6=1;P0_0=0;P0_1=0;P0_5=0;P0_2=1;P0_3=1;P0_4=1;P0_7=0;P3_0=1;//
siralama a,b,c,d,lt,bi,rbi,gnd,vcc
        if(P3_3==0 & P3_4==1 & P3_5==1 & P3_6==0 & P3_7==0 & P3_1==0 &
P3_2==0){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=1;P0_1=1;P0_5=0;P0_2=1;P0_3=1;P0_4=1;P0_7=0;P3_0=1;
        //6
        if(P3_3==0 & P3_4==0 & P3_5==1 & P3_6==1 & P3_7==1 & P3_1==1 &
P3_2==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=0;P0_1=0;P0_5=0;P0_2=0;P0_3=1;P0_4=0;P0_7=0;P3_0=1;//
siralama a,b,c,d,lt,bi,rbi,gnd,vcc
        if(P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 & P3_7==1 & P3_1==1 &
P3_2==1){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_6=0;P0_0=0;P0_1=0;P0_5=0;P0_2=1;P0_3=0;P0_4=0;P0_7=0;P3_0=1;//
siralama a,b,c,d,lt,bi,rbi,gnd,vcc
        if(P3_3==0 & P3_4==0 & P3_5==0 & P3_6==0 & P3_7==0 & P3_1==0 &
P3_2==0){//a,b,c,d,e,f,g
                c1++;
        }
}
void git7449(){
        P0=0xFF;P3=0xFF;
```

```
        P0_4=0;P0_0=0;P0_1=0;P0_3=0;P0_2=1;P0_6=0;P3_0=1;//siralama
a,b,c,d,bi,gnd,vcc
        if(P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 & P0_5==1 & P3_1==1 &
P3_2==0){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_4=1;P0_0=0;P0_1=1;P0_3=0;P0_2=1;P0_6=0;P3_0=1;//siralama
a,b,c,d,bi,gnd,vcc
        if(P3_3==1 & P3_4==0 & P3_5==1 & P3_6==1 & P0_5==0 & P3_1==1 &
P3_2==1){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_4=0;P0_0=1;P0_1=0;P0_3=1;P0_2=1;P0_6=0;P3_0=1;//siralama
a,b,c,d,bi,gnd,vcc
        if(P3_3==0 & P3_4==0 & P3_5==0 & P3_6==1 & P0_5==1 & P3_1==0 &
P3_2==1){//a,b,c,d,e,f,g
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_4=0;P0_0=0;P0_1=0;P0_3=0;P0_2=0;P0_6=0;P3_0=1;//siralama
a,b,c,d,bi,gnd,vcc
        if(P3_3==0 & P3_4==0 & P3_5==0 & P3_6==0 & P0_5==0 & P3_1==0 &
P3_2==0){//a,b,c,d,e,f,g
                c1++;
        }
}
void git7476(){
        P0=0xFF;P3=0xFF; //
        //P0_1=0;P0_2=1;P0_0=0;P0_3=0;P3_0=0;P0_6=0;P0_7=1;P0_5=0;P3_7=0;
P3_4=0;P3_3=0;P0_4=1;
        P0_1=0;P0_2=1;P0_3=0;P3_0=0;P0_6=0;P0_7=1;P3_7=0;P3_4=0;P3_3=0;P
0_4=1;
        clock1();       clock3();
        clock2();       clock4();
        if(P3_1==1 & P3_2==0 & P3_5==1 & P3_6==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        //P0_1=1;P0_2=0;P0_0=1;P0_3=1;P3_0=1;P0_6=1;P0_7=0;P0_5=1;P3_7=1;
P3_4=1;P3_3=0;P0_4=1;
        P0_1=1;P0_2=0;P0_3=1;P3_0=1;P0_6=1;P0_7=0;P3_7=1;P3_4=1;P3_3=0;P
0_4=1;
        clock1();       clock3();
        clock2();       clock4();
        if(P3_1==0 & P3_2==1 & P3_5==0 & P3_6==1){
                c1++;
        }
```

```
        P0=0xFF;P3=0xFF;
        //P0_1=0;P0_2=0;P0_0=0;P0_3=0;P3_0=0;P0_6=0;P0_7=0;P0_5=0;P3_7=0;
P3_4=0;P3_3=0;P0_4=1;
        P0_1=1;P0_2=1;P0_3=1;P3_0=0;P0_6=1;P0_7=1;P3_7=1;P3_4=0;P3_3=0;P
0_4=1;
        clock1();       clock3();
        clock2();       clock4();
        if(P3_1==1 & P3_2==0 & P3_5==1 & P3_6==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_1=1;P0_2=1;P0_3=0;P3_0=1;P0_6=1;P0_7=1;P3_7=0;P3_4=1;P3_3=0;P
0_4=1;
        clock1();       clock3();
        clock2();       clock4();
        if(P3_1==0 & P3_2==1 & P3_5==0 & P3_6==1){
                c1++;
        }
}
void git74138(){
        P0=0xFF;P3=0xFF;
        P0_0=0;P0_1=0;P0_2=0;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;       //0
        if(P3_1==0 & P3_2==1 & P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 &
P3_7==1 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=1;P0_1=1;P0_2=1;P0_3=1;P0_4=1;P0_5=0;P0_7=0;P3_0=1;//boşver
        if(P3_1==1 & P3_2==1 & P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 &
P3_7==1 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=1;P0_1=0;P0_2=0;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;//1
        if(P3_1==1 & P3_2==0 & P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 &
P3_7==1 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=0;P0_1=1;P0_2=0;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;//2
        if(P3_1==1 & P3_2==1 & P3_3==0 & P3_4==1 & P3_5==1 & P3_6==1 &
P3_7==1 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=1;P0_1=1;P0_2=0;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;//3
        if(P3_1==1 & P3_2==1 & P3_3==1 & P3_4==0 & P3_5==1 & P3_6==1 &
P3_7==1 & P0_6==1){
                c1++;
```

```
        }
        P0=0xFF;P3=0xFF;
        P0_0=0;P0_1=0;P0_2=1;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;//4
        if(P3_1==1 & P3_2==1 & P3_3==1 & P3_4==1 & P3_5==0 & P3_6==1 &
P3_7==1 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=1;P0_1=0;P0_2=1;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;
        if(P3_1==1 & P3_2==1 & P3_3==1 & P3_4==1 & P3_5==1 & P3_6==0 &
P3_7==1 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=0;P0_1=1;P0_2=1;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;
        if(P3_1==1 & P3_2==1 & P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 &
P3_7==0 & P0_6==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_0=1;P0_1=1;P0_2=1;P0_3=0;P0_4=0;P0_5=1;P0_7=0;P3_0=1;
        if(P3_1==1 & P3_2==1 & P3_3==1 & P3_4==1 & P3_5==1 & P3_6==1 &
P3_7==1 & P0_6==0){
                c1++;
        }
}
void git4000(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_3=1;P3_6=1;P0_2=1;P0_3=1;P0_4=1;P0_6=0;P3_0=1;
        if(P3_4==0 & P3_5==0 & P0_5==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_3=0;P3_6=0;P0_2=0;P0_3=0;P0_4=0;P0_6=0;P3_0=1;
        if(P3_4==1 & P3_5==1 & P0_5==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_3=0;P3_6=1;P0_2=0;P0_3=0;P0_4=0;P0_6=0;P3_0=1;
        if(P3_4==1 & P3_5==0 & P0_5==1){
                c1++;
        }
}
void git4001(){
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_5=1;P3_6=1;P0_0=1;P0_1=1;P0_4=1;P0_5=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_4==0 & P0_2==0 & P0_3==0){
                c1++;
```

```
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=1;P3_5=0;P3_6=1;P0_0=0;P0_1=1;P0_4=0;P0_5=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_4==0 & P0_2==0 & P0_3==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_5=0;P3_6=0;P0_0=0;P0_1=0;P0_4=0;P0_5=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_4==1 & P0_2==1 & P0_3==1){
                c1++;
        }
}
void git4002(){
        P0=0xFF;P3=0xFF;
        P3_2=1;P3_3=1;P3_4=1;P3_5=1;P0_1=1;P0_2=1;P0_3=1;P0_4=1;P0_6=0;P
3_0=1;
        if(P3_1==0 & P0_0==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=0;P3_3=0;P3_4=0;P3_5=0;P0_1=0;P0_2=0;P0_3=0;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_1==1 & P0_0==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=0;P3_3=1;P3_4=1;P3_5=0;P0_1=0;P0_2=1;P0_3=1;P0_4=0;P0_6=0;P
3_0=1;
        if(P3_1==0 & P0_0==0){
                c1++;
        }
}
void git4010(){
        P0=0xFF;P3=0xFF;
        P3_2=1;P3_5=0;P3_7=1;P0_2=0;P0_4=1;P0_6=0;P0_7=0;P0_0=1;P3_0=1;
        if(P3_1==1 & P3_4==0 & P3_6==1 & P0_1==0 & P0_3==1 & P0_5==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_2=0;P3_5=1;P3_7=0;P0_2=1;P0_4=0;P0_6=1;P0_7=0;P0_0=1;P3_0=1;
        if(P3_1==0 & P3_4==1 & P3_6==0 & P0_1==1 & P0_3==0 & P0_5==1){
                c1++;
        }
}
void git4011(){
        P0=0xFF;P3=0xFF;
```

```
        P3_1=1;P3_2=0;P3_5=1;P3_6=0;P0_0=1;P0_1=0;P0_4=1;P0_5=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_4==1 & P0_2==1 & P0_3==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_2=1;P3_5=1;P3_6=1;P0_0=1;P0_1=1;P0_4=1;P0_5=1;P0_6=0;P
3_0=1;
        if(P3_3==0 & P3_4==0 & P0_2==0 & P0_3==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_2=0;P3_5=0;P3_6=0;P0_0=0;P0_1=0;P0_4=0;P0_5=0;P0_6=0;P
3_0=1;
        if(P3_3==1 & P3_4==1 & P0_2==1 & P0_3==1){
                c1++;
        }
}
void git4055(){
        P0=0xFF;P3=0xFF;
        P0_3=1;P0_1=0;P0_2=0;P0_4=0;P0_6=-1;P0_7=0;P3_0=1;
        if(P3_7==1 & P3_6==1 &  P3_5==1 &  P3_4==1 &  P3_3==1 &  P3_1==1
& P3_2==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_3=1;P0_1=1;P0_2=1;P0_4=1;P0_6=-1;P0_7=0;P3_0=1;
        if(P3_7==0 & P3_6==0 &  P3_5==0 &  P3_4==0 &  P3_3==0 &  P3_1==0
& P3_2==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P0_3=0;P0_1=0;P0_2=0;P0_4=0;P0_6=-1;P0_7=0;P3_0=1;
        if(P3_7==1 & P3_6==1 &  P3_5==1 &  P3_4==1 &  P3_3==1 &  P3_1==1
& P3_2==0){
                c1++;
        }
}
void git4502(){
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_3=1;P3_6=0;P0_0=1;P0_5=0;P0_2=1;P3_4=1;P0_3=0;P0_7=0;P
3_0=1;
        if(P3_2==0 & P3_5==0 &  P3_7==0 &  P0_1==0 &  P0_6==0 &  P0_4==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=0;P3_3=0;P3_6=0;P0_0=0;P0_5=0;P0_2=0;P3_4=0;P0_3=0;P0_7=0;P
3_0=1;
        if(P3_2==1 & P3_5==1 &  P3_7==1 &  P0_1==1 &  P0_6==1 &  P0_4==1){
```

```
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_1=1;P3_3=1;P3_6=1;P0_0=1;P0_5=1;P0_2=1;P3_4=0;P0_3=0;P0_7=0;P
3_0=1;
        if(P3_2==0 & P3_5==0 &  P3_7==0 &  P0_1==0 &  P0_6==0 &  P0_4==0){
                c1++;
        }
}
void git4516(){
        P0=0xFF;P3=0xFF;
        P3_7=0;P0_0=1;P3_6=1;P0_4=1;P0_3=1;P3_4=1;P3_3=1;P0_2=1;P0_7=0;P
3_0=1;
        clock6();       clock5();
        if(P0_5==1 & P3_5==1 &  P3_2==1 &  P0_1==1){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_7=1;P0_0=0;P3_6=0;P0_4=0;P0_7=0;P3_0=1;
        clock6();       clock5();
        if(P0_5==0 & P3_5==0 &  P3_2==0 &  P0_1==0){
                c1++;
        }
        P0=0xFF;P3=0xFF;
        P3_7=0;P0_0=1;P3_6=0;P0_4=0;P0_3=0;P3_4=0;P3_3=0;P0_2=0;P0_7=0;P
3_0=1;
        clock6();       clock5();
        if(P0_5==0 & P3_5==0 &  P3_2==0 &  P0_1==0){
                c1++;
        }
}
```

ÖZGEÇMİŞ

Murat ATÇI, 1985'te Adapazarı'nda doğdu. İlkokul ve ortaokul öğrenimini Sakarya Namık Kemal İlköğretim Okulu'nda tamamlamıştır. Lise öğrenimini Sakarya Fatih Anadolu Meslek Lisesi Elektronik bölümünde tamamladıktan sonra 2003 yılında Sakarya Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Bölümü Elektronik Öğretmenliği programına girmiştir. 2007 yılında lisans eğitimini tamamladıktan sonra aynı yıl Sakarya Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi Ana Bilim dalında yüksek lisans eğitimine başlamıştır. Halen Taşdelen İMKB Teknik ve Endüstri Meslek Lisesi'nde Elektronik Öğretmeni olarak görev yapmaktadır.