ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

Dilber Nalan İSTANBULLU

BİLGİSAYAR DESTEKLİ FİZİK DENEYLERİ

FİZİK ANABİLİM DALI

ADANA, 2006

ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR DESTEKLİ FİZİK DENEYLERİ

Dilber Nalan İstanbullu

YÜKSEK LİSANS TEZİ
FİZİK ANABİLİM DALI

Bu tez Tarihinde Aşağıdaki Jüri Üyeleri Tarafından Oybirliği/Oyçokluğu İle Kabul Edilmiştir.

İmza…….	İmza…….....	İmza…….
Doç.Dr. İsa DUMANOĞLU	**Prof.Dr. Gülsen ÖNENGÜT**	**Yard.Doç.Dr. Sami ARICA**
DANIŞMAN	**ÜYE**	**ÜYE**

Bu tez Enstitümüz Fizik Anabilim Dalında hazırlanmıştır.
Kod No:

Prof. Dr. Aziz ERTUNÇ
Enstitü Müdürü

Bu Çalışma Ç.Ü. Bilimsel Araştırma Projeleri Birimi Tarafından Desteklenmiştir.
Proje No: FEF2005YL4

ÖZ
YÜKSEK LİSANS TEZİ

BİLGİSAYAR DESTEKLİ FİZİK DENEYLERİ

Dilber Nalan İSTANBULLU
ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
FİZİK ANABİLİM DALI

Danışman: Doç. Dr. İsa DUMANOĞLU
Yıl: 2006, Sayfa :76
Jüri: Doç. Dr. İsa DUMANOĞLU
Prof. Dr. Gülsen ÖNENGÜT
Yrd.Doç.Dr. Sami ARICA

Fizik deneylerinde bilgisayar kullanımı veri alımını kolaylaştırmakta, hesaplamaların yapımında ve sonuçların yorumlanmasında yardımcı olan grafikleri çizmekte de büyük kolaylık sağlamaktadır**.** Bu çalışmada da lise fizik laboratuarında da kullanılabilecek deneyler geliştirilmiştir. Deneylerin yapımında kullanılacak arabirim sistemi tanıtıldıktan sonra, deneylerin yapılışı ve sonuçlar açıklanmıştır.

Anahtar kelimeler: Arabirim kartı, basit sarkaç, eylemsizlik kütlesi, RC devresi, tek ve çift yarıkta girişim.

ABSTRACT

MSc THESIS

PHYSICS EXPERIMENTS SUPPORTED BY A COMPUTER

Dilber Nalan İSTANBULLU

DEPARTMENT OF PHYSICS

INSTITUDE OF NATURAL AND APPLIED SCIENCES

UNIVERSITY OF ÇUKUROVA

Supervisor: Doçent. Dr. İsa DUMANOĞLU

Year: 2006, Pages: 76

Jury: Assoc. Prof. Dr. İsa DUMANOĞLU

Prof. Dr. Gülsen ÖNENGÜT

Asist. Prof. Dr. Sami ARICA

The use of a computer in physics experiments simplifies getting data. It also gives a great convenience in calculation and in getting graphs that helps interpretation of the results. In this thesis, the experiments that can be used in high school physics laboratories are designed. After the interface system is introduced, procedure and the results of the experiments are explained.

Key Words: Interface card, simple pendulum, inertial mass, RC circuit, interference in single and double slit.

TEŞEKKÜR

Bu tezin oluşum süreci içinde bana yardımcı olan, değerli görüşlerinden yararlandığım değerli hocam Doçent. Dr. İsa DUMANOĞLU'na, hiçbir zaman yardımını esirgemeyen sevgili arkadaşım Mehmet VERGİLİ'ye teşekkür ediyorum.

Ayrıca benden yardımını esirgemeyen, desteğini her zaman hissettiğim sevgili eşim Nasreddin İSTANBULLU'ya teşekkür ederim.

İÇİNDEKİLER **SAYFA**

ÇİZELGELER DİZİNİ **SAYFA**

ŞEKİLLER DİZİNİ **SAYFA**

1.GİRİŞ

Günümüz teknolojisinin ilerlemesiyle ve eğitime verilen önemin artmasıyla, eğitim sorunlarının çözümünde teknolojiden faydalanmak kaçınılmaz olmuştur. Bu teknolojilerden biri de bilgisayardır.

Genel olarak okullarda bulunan tipteki bilgisayarları, fizik öğretim laboratuarlarında kullanılabilecek bir ölçme aletine dönüştürmek mümkündür (Depireux, 1991). Bilgisayar, hafızaya alma, kaydetme, iletişim, veri dizimi gibi birçok işi eş zamanlı olarak yapabilir. Daha sonra verilerin birbirine bağlı değişimlerinin grafiğini çizebilir. Eğer bir kelime işlemci yazılımı varsa, deney raporu hazırlanabilir. Mikrobilgisayar tarafından veri toplama ve kaydı sırasında tasarruf edilen zaman, veri analizinde ve sonuçların geliştirilmesinde kullanılabilir. En önemlisi de bu şekilde çalışan öğrenciler bilim adamlarının araçlarını ve metodlarını öğrenirler (Jacobsen, 1991). Bilgisayara dayalı laboratuar sensörlerini ve yazılımını kullanarak öğrenciler konum, hız, ivme, kuvvet, sıcaklık, ışık şiddeti, ses basıncı, akım ve potansiyel farkı gibi fiziksel büyüklükleri ölçerken grafiğini çizebilmektedirler.Veri toplamanın ve sunmanın kolaylığı, hazırlığı yetersiz öğrencileri bile, bilimsel süreçte aktif katılımcılar yapacak ve kendi sorularını sorup cevaplandıracak şekilde cesaretlendirecektir (Thornton, 1991). Fen eğitimi deneyden teoriye geçiş şeklinde anlatılırsa, öğrencinin fen temelini kavraması daha kolay olacak ve bu yolla verilen fen eğitimi ezberci bir eğitim olmaktan çıkacaktır. Deney düzeneklerinin kurulması, verilerin alınması, hesapların yapılıp grafik çizimi ve yorumlanması oldukça uzun zaman almakta, çoğu zaman bunlar ders saati içinde yapılamamaktadır. Bunların yanında nükleer enerji, radyoaktif bozunma deneyleri gibi laboratuar ortamında yapılması tehlikeli, yüksek maliyetli ve uzun zaman alan deneyler de bilgisayar destekli ortamlarda kolaylıkla yapılabilir.

Bilgisayar kullanarak deney yapılmadan önce bilgisayarda canlandırılabilmekte (simülasyon), beklenen sonuçlar öngörülmeye çalışılmakta ve bulunan deneysel sonuçlarla karşılaştırılabilmektedir.

Bilgisayarların fizik laboratuarlarında kullanım amaçlarından birisi otomatik veri kaydı ve analizi, diğeri de kontroldür. Veri kaydı ve analizi için örnek olarak

simülasyonu, kontrol için ise otomasyonu örnek olarak verebiliriz. Bu amaçlarla fizik laboratuarında bilgisayar kullanımı, laboratuar çalışmalarını daha ilginç bir hale getirmektedir.

Fizik eğitimine az da olsa katkıda bulunmak amacıyla geliştirilen deneylerin bulunduğu bu tezin giriş bölümünde, bilgisayarların fen eğitiminde ve fizik laboratuarında kullanılması ile ilgili genel bilgiler verilmiştir. İkinci bölümde; bilgisayarın fen eğitiminde ve fizik laboratuarında kullanılması ile ilgili önceden yapılan çalışmalar kısaca anlatılmıştır. Üçüncü bölümde bilgisayar hakkında genel bilgiler verildikten sonra veri iletimi ve arabirim sistemi ile ilgili bilgiler verilmiştir. Arabirim sistemi bilgisayarı ölçü aracı olarak kullanabilmemizi sağlar. Dördüncü bölümde geliştirilen deneylerin amacı, teorisi, yapılışı ve deney sonuçları anlatılmıştır. Deneylere ait, C programlama dilinde geliştirilen programlar ise ekler bölümünde sunulmuştur. Beşinci bölümde de sonuçlar yer almaktadır.

2. ÖNCEKİ ÇALIŞMALAR

Bilgisayara dayalı laboratuar araçları ve eğitim araştırmalarına dayalı, dikkatlice tasarlanmış eğitim programları, üniversite ve liselerde geniş bir öğrenci kitlesine fizik kavramlarını öğretmekte kullanılmıştır. Veriler, geleneksel derslerde öğrenilmeyen temel fizik kavramlarının bu yöntemle çok daha iyi öğrenildiğini göstermektedir (Thornton, 1991).

Fizik öğretiminde bilgisayarın bir laboratuar aracı olarak kullanılması 1983 yılından bu yana pek çok araştırmanın konusu olmuştur (Cortini,1992). Giulio Cortini, deneysel fizik öğretiminde bilgisayarın bir laboratuar aracı olarak kullanılmasını ele almıştır. Örnek olarak basit sarkacın hareketini incelemiştir.

Bilgisayarla beraber kullanılan data logger cihazının öğrenci başarısına olan etkisinin araştırıldığı çalışmada deneysel yöntem uygulanmışdır. Trabzon il merkezindeki bir ilköğretim okulunun altıncı sınıflarında öğrenim gören öğrencilerden oluşan deney grubuna (N=23) ohm kanunu data logger cihazı ile kontrol grubuna (N=26) ise manuel aletlerle, voltmetre-ampermetre, ile öğretilmiştir. Çalışmanın sonunda deney grubu öğrencileri kontrol grubuna göre oldukça başarılı olmuştur. Data logger cihazının kullanılması öğrencilerin performanslarını olumlu yönde artırmıştır (Ayvacı, Özsevgeç, Aydın, 2004).

Fizik deneylerinde bilgisayarı kullanarak; veri toplamanın, verilerle işlem yapmanın ve sonuçları grafiklerle yorumlamanın avantajları gösterilmiştir. Konu olarak, fizikte radyoaktif bozunmalarla ilgili deneyler seçilmiş, bu deneylerin bilgisayar kullanmadan İstanbul Teknik Üniversitesi Nükleer Enerji Enstitüsü'nde; bilgisayar kullanarak Marmara Üniversitesi Eğitim Fakültesi Fizik Bölümü'nde yapılması arasındaki fark belirlenmiştir (Çorlu ve Altın, 1999).

Kiel üniversitesinde öğrencilere fizik deneylerini yapabilmeleri için arabirimleri kullanmayı ve bu amaç için program yazmayı öğreten bir fiziğe giriş laboratuarı tasarlanmış ve uygulanmıştır. Bilgisayarların akıllı bir teori ve deney aracı olarak davrandığı ve böylece mikrobilgisayarlardan önce yapılamayan temel laboratuar deneylerini mümkün kıldığı bazı projeler tartışılmıştır (Lincke, 1991).

Bilgisayar destekli fizik eğitimi konusunda yazılan bazı makaleler şunlardır:

Kühnelt (1991), Hasnain (1991), Hacınlıyan ve Tepehan (1991), Ersoy (1991), Campbell (1991), Borkowski (1991), Çakmak (1999).

3. MATERYAL ve METOD

Bu bölümde bilgisayar hakkında ve veri iletimi hakkında genel bilgiler verilecek. Daha sonra bilgisayar destekli deneylerde kullanılacak elemanlar tanıtılacaktır.

3.1. Bilgisayar

Kullanıcıdan aldığı verilerle mantıksal (<, >, =, and, or) ve aritmetiksel (+, -, * , /) işlemleri yapan; yaptığı işlemlerin sonucunu saklayabilen; sakladığı bilgilere istenildiğinde ulaşılabilen elektronik bir makinedir.

Bilgisayarın 4 işlevi bulunur:

- Giriş/Çıkış işlevleri
- Aritmetik işlemler yapabilmesi
- Karşılaştırma(Mantık) işlevleri
- Bilgi depolama ve depolanan bilgiye erişim işlevi

Bilgisayar bu işlevleri yapmak için 2 ana kaynaktan yararlanır:

1. Donanım (hardware)
2. Yazılım (software)

Donanım, bilgisayarın elle tutulur, gözle görülür, mekanik, magnetik veya elektronik olabilen fiziksel parçalarıdır. Bilgisayar kasası, giriş çıkış üniteleri gibi birimler donanımın parçalarıdır.

Yazılım, donanım elemanlarını kontrol etmek veya kolay yoldan bilgisayarınıza veri depolamak, hesap yaptırmak, internet üzerinde gezinmek ve daha bunun gibi bir çok işi yapabilmek için çeşitli programlama dilleri ile yazılan programlardır. İşletim sistemleri (windows, linux), Microsoft Office, İnternet Gezgini, müzik dinleme programları, yazılımlara örnek olarak verilebilir. Yazılımlar birçok programlama dilleri ile yapılabilir. Programlama dillerine örnek olarak C, C++, Pascal, Basic verilebilir. Hazırlanan deneylerde Turbo C editörü ve derleyicisi kullanılacaktır. Yazılımlar programlama dilleriyle yapıldıktan sonra, derleyiciler ile bilgisayarın anlayacağı hale getirilir. Derleyici, programın tamamını okur ve bunu

nesne koduna (object code) dönüştürür. Nesne kodu, programın kaynak kodunun, bilgisayarın doğrudan çalıştırabileceği bir biçime dönüştürülmüş halidir. Nesne kodu, aynı zamanda ikili kod (binary code) ya da makine kodu (machine code) olarak da adlandırılır.

Özel amaçlı kullanılan bilgisayarların yanında yaygın olarak kullanılan bilgisayarlar kişisel bilgisayarlardır (PC). Özel amaçlı bilgisayarlara örnek olarak matbaa işleri için kullanılan Macintosh, personel özlük işlemleri için kullanılan IBM AS400, sunucu (server) olarak kulanılan Sun verilebilir. Bu bilgisayarların mimarileri PC lerden biraz farklıdır. Deneyleri yaparken kullanılan bilgisayar ise PC dir.

3.2. Kişisel Bilgisayar (PC)

PC'ler kişisel kullanım için geliştirilmiş bilgisayarlardır ve günümüz teknolojisi ile çok hızlı veri işleyebilmektedir. İhtiyaca uygun yazılımları kullanarak veya program yazılarak PC ile birçok işi yapabilmek mümkündür.

PC'ler programlanacak her işi yapabilirler. İşletim sistemleri sayesinde kullanımları çok kolay bir hale gelmiştir.

PC' lerin donanım elemanlarını inceleyelim.

Merkezi işlem birimi (CPU): Mikroişlemci olarak da adlandırılır. Bilgisayarın en önemli elemanıdır. Elektronik bir beyin olarak düşünebileceğimiz mikroişlemci bilgisayarın çalışmasını düzenleyen ve programlardaki komutları tek tek işleyen birimdir. Temel olarak mikroişlemcinin yaptığı iş, bitler üzerinde işlem yapmak üzere komutları çalıştırmaktır. Transistöründen yongasına kadar bilgisayarı oluşturan bütün elemanlar emirleri mikroişlemciden alırlar. Bilgisayarda ya da çevre birimlerinde olup biten her şey, mikroişlemci tarafından yollanan sinyallerle gerçekleşir ve denetlenir.

Mikroişlemciler açma kapama anahtarı gibi çalışan milyonlarca transistörden oluşmaktadır. Bu anahtarların programlanma durumuna göre, elektrik sinyalleri bunların üzerinden akar. Bu sinyaller bilgisayarın yaptığı tüm işleri toplama, çıkarma, çarpma, bölme gibi temel matematiksel işlemlere indirir. İşlemci de bu

işlemleri en basit sayma sistemi olan ikilik düzen yani sadece 0 ve 1 sayılarını kullanarak yapar.

Mikroişlemcinin hızı saniyede yapılan işlem ile ölçülür. Hız ölçü birimi olarak Megahertz (MHz) kullanılır.

Mikroişlemcilerin tür ayrımı aynı anda işleyebildiği bit sayısına göre yapılmaktadır. Bugüne kadar üretilmiş olan mikroişlemci türleri;

1) 4 bitlik mikroişlemciler
2) 8 bitlik mikroişlemciler
3) 16 bitlik mikroişlemciler
4) 32 bitlik mikroişlemciler
5) 64 bitlik mikroişlemciler

Deneyleri yaparken kullandığımız mikroişlemci 32 bitliktir. Veriyoluna 32 iletken ile bağlanmıştır ve aynı anda 32 bit uzunluğundaki bir kelimeyi işleyebilir.

Günümüzde kullanılan mikroişlemciler MOTOROLA, INTEL, AMD ve CYRIX'dir. Apple'ın çıkardığı POWER PC makinalarında kullanılmaktadır. IBM uyumlu bilgisayarlarda ise INTEL, AMD ve CYRIX kullanılmaktadır. Bunlardan piyasayı elinde bulunduran ise INTEL'dir. Cyrix çıkardığı birkaç işlemci türünden sonra artık işlemci üretmemeye başlamışdır. AMD ise şu an INTEL ile büyük bir yarış içerisindedir. Son çıkardığı işlemcilerin bazı testlerde INTEL'i geçtiği olmuştur.

Mikroişlemcinin yapısını oluşturan bölümler:

- Kontrol birimi: Bütün komutlar burada işletilir. İşlenen komuta göre mikroişlemci içerisindeki belli bir adresteki veri değiştirilir ya da bir verinin işlemci içindeki başka bir bölüme aktarılması sağlanır.
- İletim yolları: Mikroişlemci ile bilgisayarın diğer birimleri arasındaki bağlantıları sağlayan iletkenlerdir. İletim yolları 3 gruba ayrılır:

1. Veri yolları (Data bus): Bilgisayarın bir bileşeninden diğerine verileri iletmek için kullanılan devrelere veriyolu (bus) adı verilir. İşlemlerde kullandığı veriler de mikroişlemciye veriyolu adı verilen kanallardan gelir. Bu nedenle bilgisayarın performansı, işlemci hızı ile birlikte veriyolu hızına da bağlıdır. Her veriyolunun MHz cinsinden bir saat hızı (frekans değeri) vardir. Hızlı bir veriyolu, verileri daha

hızlı transfer ederek uygulamaların daha hızlı çalışmasını sağlar. Bir veriyolunun kapasitesi de önemlidir çünkü bir seferde ne kadar veri transfer edilebileceğini belirler. Örneğin 16 bit'lik veriyolu bir seferde 16 bit, 32 bit'lik veriyolu 32 bit veri transfer eder. Çeşitli veriyolu standartları vardır. Bunlar ISA, EISA, MCA, VESA ve PCI'dır.

2. Adres yolları (Adress Bus): Verinin tipi ne olursa olsun işlemci bunu doğrudan veri yoluna gönderemez. Önce işlemcinin verinin gideceği yeri belirtmesi gerekir. Bu, adres yolu denen başka bir bağlantı kümesi tarafından gerçekleştirilir

3. Kontrol yolları (Control Bus) : Sistem yolu her bir bileşene erişim olanağı sağlar ve işlemin okuma işlemi mi, yazma işlemi mi olduğuna karar verir.

- Sayıcılar (Counter) : Sayıcılar, işlemi yapılacak komut ve verilerin adreslerini taşıyarak, bilgisayarın çalışması sırasında hangi verinin hangi sıra ile kullanılacağını belirlerler.
- Giriş-çıkış devreleri: Bu devreler mikroişlemcinin, yalnızca giriş ve yalnızca çıkış yapan veya giriş-çıkış yapan birimleri ile bağlantı kurduğu devrelerdir.
- Aritmetik mantık birimi: Mikroişlemcinin, birinci derecede önem taşıyan bir birimidir. Toplama çıkarma gibi basit matematiksel işlemleri yapar.
- Kaydedici (Register): Mikroişlemci, register adı verilen 14 özel alan içerir. Herbiri 16 bit genişliğinde olan bu alanları özel bellek birimleri olarak düşünebiliriz. Mikroişlemci ile bellek ve giriş/çıkış (I/O) kapıları arasındaki bilgi alışverişlerinin çeşitli aşamalarında, bilginin geçici olarak depolanmasını ve bu veriler üzerinde işlem yapılmasını sağlarlar. Mikroişlemci çipinin üzerinde yer aldıklarından yani kontrol biriminin doğrudan bağlandığı bellek birimleri olduğundan registerle yapılan işlemler bellek bölgeleri üzerinde yapılan işlemlere göre çok daha hızlıdır. Farklı komut ve register setlerine sahip olan işlemciler birbirlerinin yazılımlarını çalıştıramazlar. Merkezi işlem birimi belleğe ve giriş/çıkış portlarına iç registerler ile ulaşır. Hafıza herbiri 8 bitlik ve kendi adresi olan milyonlarca registerden oluşur. Bu registerlerde bilgi gerektiğinde ulaşılmak üzere sürekli olarak depolanır.

- Kayar nokta birimi: Matematiksel işlemci olarak da bilinir. Tamsayı olmayan işlemlerden sorumludur.

Bir işlemcinin performansını belirleyenler:

- İşlemci mimarisi: Bir işlemcinin bir saat döngüsünde ne kadar uzunlukta kaç tane komutu aynı anda işleyebildiğini saat hızı ya da önbelleği değil sadece mimarisi belirler. Ortak mimariye sahip olan işlemciler aynı komutları tanımakta ve yazılımları çalıştırabilmektedirler. En meşhur mikroişlemci mimarisi intel'in x86 işlemcisidir. İntel ilk x86 tabanlı işlemcisini 8086 olarak 1978 yılında piyasaya sürdü. Daha sonraki yıllarda yeni nesil x86 tabanlı işlemciler çıkarıldı. 286, 386, 486, Pentium ve Pentium Pro olarak bu kuşakları görebilmekteyiz. Intel 486'dan sonra ürettiği mikroişlemcinin adını Pentium olarak duyurdu ve rakam kullanmaya son verdi. Pentium'dan sonraki işlemci de Pentium Pro olarak adlandırıldı. Pentium II, Celeron, Pentium III, Xeon ve Katmai, Pentium Pro'nun varyasyonlarıdır. İntel'in haricindeki diğer mimariler ise şunlardır: Machintosh'larda bulunan PowerPC, Digital ve Compaq'ın güçlü serverlerinde kullanılan Alpha ailesi, Silicon Graphics'in Mips Rxoo serisi, Hawlett-Packard'ın PARISC'i ve Sun Microsystems'e ait SPARC'tır.
- Saat hızı: İşlemcinin çalışma frekansıdır. Bir işlemcideki bütün elemanlar saat vuruşlarıyla çalışır. Saat hızı bir işlemcinin saniyede ne kadar çevrim yapabileceğini belirler. Her çevrimde işlemcinin ne kadar işlem yapabileceği işlemcinin yapısına göre değişir. Bu saat vuruşları anakart üzerindeki Clock Generator denen yongayla üretilir. Bu yonganın içinde çok hassas kristaller vardır. Bu kristallerin titreşimleri saat vuruşlarını oluşturur. Bir işlemcinin saat hızını sistem hızıyla (FSB, Front Side Bus) işlemcinin çarpanının çarpımı belirler. Sistem hızı fazla yüksek olmasa da işlemci kendi içinde çarpanlarını kullanarak çok daha yüksek hızlara çıkabilir. Örneğin 1.8 GHz hızında çalışan bir Pentium 4 işlemci 18*100 MHz'te çalışır.
- L1/L2 Cache: Çalışmakta olan bir programa ait komutların geçici olarak saklandığı hafızadır. Cache hafızalar, Level1 (L1) ve Level2 (L2) olmak üzere ikiye ayrılırlar. İşlemci ihtiyaç duyduğu komutu ilk önce L1 cache

hafızada arar. Eğer işlemcinin aradığı komut burada yoksa L2 cache hafızaya bakılır.Eğer burada da yoksa sırayla, RAM ve HDD üzerindeki sanal hafıza üzerinde arar. L1 cache hafıza bunlar içerisinde en hızlı olanıdır ve genellikle işlemcinin üzerine imal edilir. L2 cache hafıza ise L1'e göre daha yavaş olmasına rağmen gene de hızı çok yüksektir. Bir kısım işlemcilerde (Celeronların ilk nesillerinde olduğu gibi) L2 cache hafıza bulunmayabilir. Bu durumda L1 cache hafızaya sığmayan komutlar L2 olmadığından doğrudan RAM'a yazılmakta ve işlemcinin performansı düşmektedir.

- Yazılım Uyumluluğu: Bilgisayarların ilk günlerinde herkes kendi yazılımını yazdığı için işlemci mimarisi biraz daha arka plandaydı. Geçen zamanla birlikte yazılımlar da oldukça gelişti ve bugün ise yazılım başlı başına bir sektördür. Günümüzde her ihtiyacımız için oturup kendi yazılımlarımızı hazırlamamız imkansız, bir o kadar da gereksizdir. Belirli bir standartlaşmayla beraber işlemcilerin önemi de arttı. Günümüz PC'leri Intel 80x86 mimarisini kullanır. Bu mimari 70'li yıllardan bugüne kadar gelmiştir, güncel CISC işlemciler hala bu mimariyi kullanır.

Programlar işlemcilere göre değil komut setlerine göre yazılır ve 80x86 mimarisine göre yazılmış bir program hem bir Intel işlemcide hem de bir AMD işlemcide çalışabilir. İşlemcilere özel bazı ek komut setleri olsa da (SSE, 3D Now! gibi) bunlar sadece işlemciye yönelik optimizasyonlardır ve programlar temelde aynıdır. 80x86 mimarisine göre yazılmış 32 bitlik bir program aynı mimarideki 32 bitlik bütün işlemciler tarafından sorunsuzca çalıştırılabilir.

Ham işlemci performansını ifade etmek için MIPS (Million Instructions Per Second, saniyede işlenebilen komut sayısı) ve MFLOPS (MillionFloating Point Operations Per Second, saniyede yapılan kayar nokta hesabı) birimleri kullanılır.

Anakart: Donanım elemanlarının tümünü ve veriyollarını üzerinde bulunduran levhadır. Anakart, üzerinde bulunan ve yonga seti adı verilen entegre devreler ile bilgisayar içindeki elemanlar arasındaki veri akışını denetler. Bilgisayarın bir bileşeninden diğerine verileri iletmek için kullanılan devrelere veriyolu (bus) denir. Sadece iki donanım aygıtını birbirine bağlayan veri yoluna port denir. Çeşitli aygıtları bağlamak için kasanın arkasında yer alan girişler (portlar) doğrudan

anakarta bağlıdır. İşlemci (CPU) bilgisayarın çevre birimleri (klavye, yazıcı, fare v.b.) ile portları kullanarak iletişim kurar. Her port "port adresi" adı verilen özel bir sayı ile işaretlenmiştir. Dışarıdan işlemciye bilgi transferi INPUT, işlemciden dışarıya bilgi transferi OUTPUT olarak isimlendirilir. Kullandığımız kasanın arkasında PS/2 portu adı verilen küçük yuvarlak, 6 pinli bir fare ve bir klavye portu, iki USB portu, iki seri pc (COM) portu, bir paralel (LPT) portu var. Seri portlara genelde harici modemler bağlanır, ama seri port kullanan başka aygıtlar da vardır (yedekleme aygıtları, dijital kameralar gibi). Paralel porta ise yazıcı veya tarayıcı bağlanır. USB portlara neredeyse her tür harici aygıt bağlanabilir. USB'nin özelliği, seri ve paralel portlara göre çok daha hızlı olması ve USB aygıtlar üzerindeki yeni USB portları aracılığı ile ucuca çok sayıda aygıtın zincirleme bağlanabilmesidir. Anakart üzerinde, kasa içinden ulaşılabilen portlar da bulunur. Bunlar genel olarak iki adet IDE portu, bir disket sürücü portu, ana kart ile bütünleşikse SCSI portudur. Bir IDE portuna bağlı kabloya, üzerindeki iki konnektör aracılığıyla iki aygıt bağlanabilir. Bunların dışında, ana kart üzerinde işlemciyi takmak için bir soket bulunur. Soket, yassı dikdörtgen şeklinde, işlemcinin iki düzlem üzerinde (enine ve boyuna) uzanan iğnelerin oturduğu yuvaya verilen addır.

Sabit Disk: Bilgisayarda bulunan verilerin saklandığı yerdir.

Ram: Geçici bellektir. İstenilen bölgesine bilgi depolanabilir, silinebilir, okunabilir, değiştirilebilir. Yalnız elektrik kesintisi veya makineyi kapatma durumunda tüm bilgiler silinir.

Ekran Kartı: Bilgisayar ile monitör arasındaki bağı kuran aygıttır.

Disket Sürücü: Manyetik disklere veri kaydedebilen ve bunların içindeki bilgileri okuyabilen aygıttır. Diğer adı disket sürücüsüdür.

Cdrom Sürücü: Veri depolama birimleridir. CDROM'lar özellikle çok büyük yer kaplayan çoklu ortam (Multimedia) bilgilerini (ses, video, resim, animasyon) içeren yazılımlar için zorunludur.

Diğer Birimler: Bu donanım elemanlarının düzenli bir şekilde yerleştirildiği, güç ve soğutma ihtiyaçlarının karşılandığı yer, kasalardır. Ayrıca klavye gibi bilgisayar kasası dışında bulunup bilgisayara bağlanan birimler de vardır. Bunların dışında, yapacağımız işe göre bilgisayara değişik donanım elemanları da

ekleyebiliriz. Örnek olarak, TV kartı, hoparlör, mikrofon, internete bağlanmak için kullanılan modem, işletim sisteminde daha kolay işlem yapmamızı sağlayan fare verilebilir.

3.3. Mantık Kapıları ve Boolean Mantığı

Bir sistemdeki herhangi bir parça ne işe yararsa yarasın mutlaka mikroişlemciye bağımlı olarak çalışır. Klavyedeki tuşlara her basışımız, yaptığımız her fare hareketi bile bir şekilde işlemciye uğrar. Hangi işlemciyi kullanırsak kullanalım çalışma prensibi aynıdır. Bir işlemci elektriksel sinyalleri "0" ve "1" (ikili sistemle çalışan bilgisayarlarımız için anlamlı olan tek değerler) şeklinde alır ve verilen komuta göre bunları değiştirerek sonucu yine 0'lardan ve 1'lerden oluşan çıktılar halinde verir. Sinyal yollandığı zaman ilgili hatta bulunan voltaj o sinyalin değerini verir. Örneğin 3.3 voltla çalışan bir sistemde 3.3 voltluk bir sinyal "1", 0 voltluk bir sinyal de "0" değerini üretir. İşlemciler aldıkları sinyallere göre karar verip çıktı oluştururlar. Karar verme işlemi her biri en az bir transistörden oluşan mantık kapılarında yapılır. Transistörler, girişlerine uygulanan akım kombinasyolarına göre devreyi açıp kapayabilen ve bu sayede de elektronik bir anahtar görevi gören yarı iletken devre elemanlarıdır. Modern işlemcilerde bu transistörlerden milyonlarca tanesi aynı anda çalışarak çok karmaşık mantık hesaplarını yapabilirler. Mantık kapıları karar verirken (yani akımın geçip geçmeyeceğini belirlerken) Boolean Mantığı'nı kullanırlar. Temel Boolean operatörleri AND (VE), OR (VEYA) ve NOT (DEĞİL) tır. Bu temel operatörlerle birlikte bunların değişik kombinasyonları kullanılır, NAND (VE DEĞİL) gibi. Her mantık işlemi "doğruluk tablosu" adı verilen bir tablo ile karakterize edilir. AND ve OR kapılarının ikisi de iki sinyal alıp onlardan bir sinyal üretir.

VE işleminin iki veya daha fazla giriş değişkeni ve bir çıkış değişkeni vardır. Bir AND kapısının "1" değerini verebilmesi (yani akımı iletebilmesi için) iki girişindeki değerin de "1" olması (yani iki girişinde de akım olması ve ikisinin de yüksek voltajlı olması) gerekir. Aksi takdirde 0 değerini verecek, yani akımı iletmeyecektir. VE kapısı için doğruluk tablosu çizelge 3.1'de verilmiştir.

VEYA işleminin iki veya daha fazla giriş değişkeni, bir çıkış değişkeni vardır. Giriş değişkenlerinden en az birinin değeri "1" ise çıkış değişkeni "1", aksi halde "0" olur.VEYA kapısının doğruluk tablosu çizelge 3.2'de verilmiştir.

NOT kapısı girişindeki değerin tersini çıkışına verir. NOT kapısının doğruluk tablosu çizelge 3.3'de verilmiştir.

Çizelge 3.1. VE kapısının doğruluk tablosu

Giriş 1	Giriş 2	Çıkış
0	0	0
0	1	0
1	0	0
1	1	1

Çizelge 3.2. VEYA kapısının doğruluk tablosu

Giriş 1	Giriş 2	Çıkış
0	0	0
0	1	1
1	0	1
1	1	1

Çizelge 3.3. NOT kapısının doğruluk tablosu

Giriş 1	Çıkış
0	1
1	0

Her girişteki elektrik akışını o girişin transistörü belirler. Bu transistörler devrelerden bağımsız ayrı elemanlar değillerdir. Çok miktarda transistör, yarı iletken bir maddenin (çoğu zaman silikonun) üzerine yerleştirilip kablolar ve dış bağlantılar olmadan birbirine bağlanır. Bu yapılara entegre devre denir ve ancak bu entegre devreler sayesinde karmaşık mikroişlemci tasarımları yapılabilir.

Güncel işlemciler mikroskobik boyuttaki transistörlerin dirençler, kondansatörler ve diyotlarla bir araya getirilmesinden oluşan milyonlarca karmaşık mantık kapısından oluşur. Mantık kapıları entegre devreleri oluştururken entegre devreler de elektronik sistemleri oluşturur.

3.4. Bilgisayarda Veri İletimi

Bir noktadan diğer bir noktaya digital bilgilerin iletilmesine "veri iletimi" denir. Bilgisayar içindeki veri iletimi, aygıtların adresleri üzerinden gerçekleşir. Ses, görüntü gibi analog bilgiler de digital veriler haline dönüştürülerek iletilebilir. Mikroişlemciler verileri “1” ve “0” değerleri üzerinden işler. Bir bit, “0” (0 volt veya toprak) ya da “1” (+5V) mantık değerlerinden birini alır. 1 byte sekiz tane bit’ten oluşur ve bir karakteri temsil eder. Bir baytın bitleri sağdan 0 ile başlayarak numaralanır “0” ve “1” rakamlarının konuma göre aldıkları özel bit değerleri vardır. Bu değerler çizelge 3.4’de verilmiştir.

Çizelge 3.4. Bit değerleri

7.bit	6. bit	5. bit	4. bit	3. bit	2. bit	1. bit	0. bit
2^7=128	2^6=64	2^5=32	2^4=16	2^3=8	2^2=4	2^1=2	2^0=1

Bir standart kod kullanarak her türlü sayı, harf ve klavyedeki tüm karakterler bilgisayarın hafızasında saklanabilir. En yaygın olan kod, ASCII (American Standart Code for Information Interchange) kodudur.

Veri iletimi seri ve paralel olmak üzere iki şekilde yapılır.

3.4.1. Seri İletim

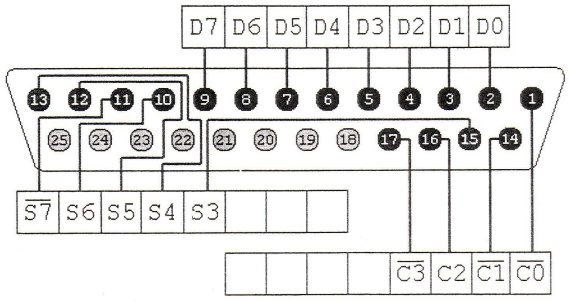
Seri iletimde "1" ve "0" mantık değerleri bir kablo üzerinden ardı sıra iletilir. Seri kablolar paralel kablolara göre daha uzundur. Seri port mantık değerlerini -3 volt ile +25 volt arasında iletebilir.

3.4.2. Paralel İletim

Digital olarak kodlanmış bilginin tüm bitleri aynı anda iletiliyorsa buna "paralel veri iletimi" denir. İletilen bilginin her biti için bir kablo bağlantısı vardır. Örneğin Y harfi 1011001 ile ifade edilir ve bunu göndermek için 8 iletim hattına ihtiyaç vardır . Her hat Y harfinin bir bitini taşır. Bilgisayarlarda mikroişlemci ile harddisk, yazıcı,tarayıcı vb. elemanlar arasındaki kısa mesafelerde paralel iletişim kullanılır.

Paralel port bilgisayarın en kolay programlanabilen portudur. Paralel portun diğer bir adı da printer portudur. Bu port 1 veya 2 adet olabilir ve LPT-l ve LPT-2 kısaltmasıyla gösterilir.Üzerinde 25 pin vardır. Aynı anda 8 bit veri aktarabilir ve

standart TTL voltajı (+5V) ile çalışır. Enerjisini bilgisayardaki anakarttan alır. Veri çıkışı port üzerindeki 2-9 numaralı pinler üzerinden elde edilir. Paralel port üzerinde veri aktarımı için kullanılan bu pinlerin tümüne DATA portu denir. Data portu üzerindeki bu 8 pinin değerleri özel bir durum olmadığı sürece "0" dır. Bu pinlerden istediklerimizi "1" durumuna getirerek istediğimiz çıkışı elde edebiliriz. Paralel port üzerindeki bir data pininin "1" olması, o pinin +5V olması anlamına gelir. Elde edilen düşük amperli bu voltaja ise "lojik voltaj"denir. Bu voltaj, bir cihazı çalıştırabilecek basit bir devreyi çalıştırmak için yeterlidir. Data portuna veri yerleştirebilmek için DATA portunun adresini bilmemiz gerekir. Port üzerinde DATA portundan başka STATUS ve CONTROL portları vardır. Bu portların dışında kalan 18-25 arası pin ise TOPRAK pinleridir. Portun en dışındaki metal kısım da toprak olarak kullanılır. Hangi pinlerin hangi porta ait olduğu şekil 3.1'de gösterilmiştir.



Şekil 3.1. Paralel port pinleri (Tuğay 2004).

DATA portuna veri yerleştirmek için öncelikle DATA portunun adresini bilmemiz gerekir. Bu adres Windows altında belirtilen taban adresinin aynısıdır. STATUS portunun adresi taban adresi+1, CONTROL portunun adresi ise taban adresi+2'dir. Taban adresi değerleri onaltılık sayı sisteminde gösterilir. Data portunun adresini belirledikten sonraki işlem veriyi adrese göndermektir. Paralel portu kullanarak giriş-çıkış işlemleri yapabilmemiz için bir programa ihtiyaç duyulur. Eğer biz programımızda bu port adresine bilgi göndermek istersek,

gönderdiğimiz bilgi 2, 3, 4, 5, 6, 7, 8, 9 numaralı pinlerden gönderilecektir. Örnek vererek açıklayacak olursak porta 128 bilgisi gönderildiğinde, sadece 9. pin "1" olur. Başka bir açıdan bakacak olursak, eğer portun 7. 5. ve 3. pinlerinde "1" görmek istiyorsak o zaman programımızdan göndereceğimiz değer, $2^5+2^3+2^1= 42$ olmalıdır.

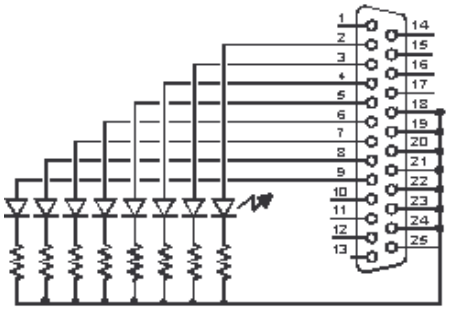
3.4.3. Bit Değerinin Okunması

DATA portuna istediğimiz veriyi kullandığımız programlama dilinin port komutları ile gönderebilir veya okuyabiliriz. Turbo C programlama dilinde bu komutlar veri göndermek için outport, okumak için ise inport'tur. Outport komutunun kullanımı outport (port, gonder) şeklindedir. Burada port, portun adresi, gönder ise gönderilecek veridir. Komutun kullanımını anlamak için led kontrolü uygulaması yapabiliriz. Ek 1'de verilen PORT1.C programını kullanarak paralel porta veri gönderebilir ve okuyabiliriz. Bu programı çalıştırmamız halinde, paralel porta bağlı ledler ile hazırlanmış küçük bir devre yardımı ile paralel porta gönderilen verileri fiziksel olarak izleyebiliriz.

Led'in + ucuna bağlanan direnç, voltajı led'i bozmayacak bir değere düşürür. Led'in uzun bacağı pozitiftir (+). Pozitif uç data pinine negatif uç ise toprağa bağlanır. Bunun için 220 Ohm ile 1KOhm arasında bir direnç kullanılabilir. Her bir led DATA portunun bir biti ile ilişkilendirilmiştir. Led'in bağlı olduğu data pinini "1" yaparak o pinin elektriksel değerini +5 Volt yapmış oluruz ve pine bağlı led yanar. Yapılması gereken hangi ledi yakmak istiyorsanız sadece o ledin bağlı olduğu pin için bit değerini hesaplamak ve bu değeri ekranda göreceğimiz yaz komutuna karşılık yazmaktır. Örnek vermek gerekirse, "1" bilgisini gönderirsek bu sayede TTL yapıdaki olan portumuzun 2 numaralı pininden (D0) +5 V'luk bir gerilim elde ederiz. Bu sayede led yanar. "0" bilgisini gönderirsek portun tüm pinleri "0" lanır yani 0 V geriliminde olur.

Eğer $128(2^7)+64(2^6)+ 32(2^5)+ 16(2^4)+ 8(2^3)+ 4(2^2)+ 2(2^1)+ 1(2^0)=255$ bilgisini gönderirsek tüm data port pinlerinde +5 V elde etmiş oluruz.

Bu devrenin şeması şekil 3.2'de verilmiştir.



Şekil 3.2. Ledler aracılığıyla yazıcı portunun data çıkışındaki verileri görselleştirmeye yarayan devrenin şeması.

Paralel porttan veri girişi için STATUS portuna karşılık gelen 10, 11, 12, 13, 15 numaralı pinlerden 5 bit sayısal giriş yapabiliriz.

Paralel port ile veri girişini anlamak için paralel porta bağladığımız 5 tane buton ile bilgisayara sayısal veri girişi yapacak ve girilen veriyi ekrandan izleyeceğiz. Bunu yaparken kullanım kolaylığı olması için data kablosunun içindeki ince kabloların bağlı olduğu bir kutu kullanıldı.Bu kutunun fotoğrafı şekil 3.3'de verilmiştir.

Şekil 3.3. Yazıcı portunun data kablosunu kullanarak yapılan kutu.

Veri girişi şekil 3.3'de gösterilen S7, S6, S5, S4, S3 pinlerinden yapılır. Bu 5 pinden okunan lojik değer başlangıçta "1" dir. Voltmetre ile bu pinlerdeki voltaj ölçüldüğünde +5 Volt olduğu görülür. Bu pinlere bağlanan butonlar ile pinleri topraklayarak lojik değerlerini "0" yapıp bir nevi veri girişi yapmış oluruz. STATUS portundaki veriyi okumak için inportb(0x379) ; veya inp(0x379) komutu kullanılır. Ek 2'deki PRI_KONT.C programı çalıştırılıp 10, 11, 12, 13, 15 numaralı pinlerden

istenilenleri topraklayarak elde edilecek veriyi ekranda görebiliriz.Bu programı çalıştırdığımızda, ekranda okunan değerler çizelge 3.5'de verilmiştir.

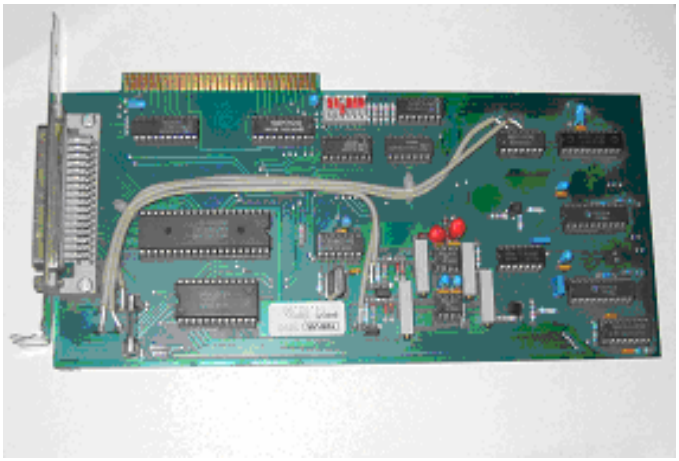
Çizelge 3.5. STATUS portunda topraklanan pinler karşılığında okunan değerler

pin	_	10	11	12	13	15
Okunan değer	127	63	255	95	111	119

Status portundan 5 bit sayısal veri girişi yapıldığında, bu beş pinden hiçbir bağlantı olmadığı durumda okuyacağımız değer 127 olur. 127 değerinin ikilik sistemdeki karşılığı 01111111'dir. Kullanmadığımız S0, S1, S2 değerleri her zaman için "1" dir. S7 pini terslenmiş olduğundan topraklandığında mantıksal değeri "1" olacaktır. Bu durumda bu porttan 11111111 değeri okunacaktır ve bu ikilik sayının onluk sistemde karşılığı 255 olur. S6 pini topraklandığında okunacak değer 00111111, S5 pini topraklandığında 01011111, S4 pini topraklandığında 01101111, S3 pini topraklandığında ise 01110111'dir. Bu ikilik sayıların onluk sistemdeki karşılıkları sırasıyla 63, 95, 111 ve 119'dur.

3.5. Deney Kartı ve Arabirimler

Kullandığımız bilgisayar, 32 bitlik, 128 KB cache belleğe sahip, 550 MHz'lik intel celeron mikroişlemciye sahiptir. Bu bilgisayarın anakartına bağlanan NEVA-PC çok fonkiyonlu kart ile deneyleri kontrol edebiliriz. Bu karta bağlı arabirimlerin port adresleri vardır. Bu port adreslerini kullanarak arabirimlerden aldığımız verileri bilgisayara aktarabiliriz. Şekil 3.4'de kartın fotoğrafı verilmiştir.



Şekil 3.4. NEVA-PC kart.

Bu kartın içeriği:

Analog-Sayısal çevirici kısım (AD670) : Elektronik sistemler "analog" ve "sayısal" olmak üzere ikiye ayrılır. Analog sistemlerde elektrik sinyalleri sürekli değişir ve belli sınırlar içinde her değeri alabilirler. Sayısal sistemlerde ise elektriksel sinyaller olduğu gibi iletilmez. Bu sinyallerin yerine bunlara karşı düşen rakamlar iletilir. Elektronik sistemlerde genel olarak giriş ve çıkış sinyalleri "analog" yapıdadır. Bunların sayısal olarak işlenebilmesi ve iletilebilmesi için Analog/Sayısal Dönüştürücü (ASC) ve Sayısal/Analog Dönüştürücü (DAC) kullanılır. Kullandığımız kartın her biri bir kanal olmak üzere iki tane 8 bitlik analog-sayısal çeviricisi vardır. Bu iki kanal birbirinden bağımsız çalışır. Birinci kanalın (CH1) adresi kart adresi + 4, ikinci kanalın (CH2) adresi ise kart adresi + 8 dir. Kart adresi 100 olduğundan birinci kanalın adresi 0x104, ikinci kanalın adresi de 0x108 olur. Bu adresler entegrelerin elektronik özelliklerinden dolayı sabittir ve yazılım ile değiştirilemez.

Analog sayısal çeviricinin kullanım amacı analog voltajı sayısal değere çevirmektir. ASC'ye dışarıdan girilen analog voltaj sayısal veriye çevrilerek istenirse bunların grafiği çizilebilir veya başka bir hesaplama için de kullanılabilir. ASC portundan alınan veriler 8-bitlik sayılar olduğu için çevirim sonucu elde ettiğimiz sayısal veriler 0-255 arasında değişecektir. Ek 3'de verilen ASC_1.C programı ASC'nin 1.kanalından okunan verileri ekrana yazar. Program çalışırken okunan

verilerin hangi voltajlara karşılık geldiğini görmek için şu yol izlenir: Önce bir sayısal voltmetre ASC'nin CH.1 ve GND'si arasına bağlanır. Bu iki uç arasına 10 K-Ohm luk potansiyometre bağlanır. Potansiyometrenin diğer iki ucunu dağıtıcı kutusunda +12 ve –12 V çıkışlarına bağlanır. Dağıtıcı kutusundan alınan voltajlara dikkat etmek gerekir akım en fazla 100 mA olmalıdır. ASC'nin +10, +25, +/-10, +/-25 V ayarları için ASC_1.C programını çalıştırıp ekranda gördüğünüz 8-bitlik sayılarla voltmetreden okuduğunuz voltajı karşılaştırabiliriz. ASC_1.C ile okunan değerlerin grafiği çizilmek istenirse Ek 4'de verilen ASC_2.C programı buna uygun olacaktır.

Çok hızlı uygulamalar için ASC'yi okuma ve sonucu grafikte gösterme işlemlerini birlikte yapmak doğru değildir. Çünkü grafik programları yavaş programlardır. Verileri hızlı bir biçimde kaydetmenin yolu ASC'de okunan değerleri bir dizide saklamaktır. Bunun için programın başında char a[ ] yazılarak bir dizi tanımlanır. Kaydetme döngüsü şimdi a[x] = inport(asc1) komutundan ibaret olacaktır. Grafik çizme ikinci bir döngü içinde yapılabilir. ASC her inport(asc1) komutu ile yeni bir çevirim yapar. Böylece elde edilen 8 bitlik sayı bir önceki ölçümün sonucudur. Hassas uygulamalarda eski ölçümleri temizlemek için önce bir boş okuma yapılmalıdır. Bu noktaları gözönünde tutarak hızlı veri kaydeden ve daha sonra bu verilerin grafiğini çizen ASC_3.C programı ek 5'de verilmiştir.

Sayısal-Analog çevirici kısım (AD7528): İki tane 8 bitlik çevirici vardır. Adresleri: Birinci kanal için Kart adresi+12, İkinci kanal için, Kart adresi+13'dür.

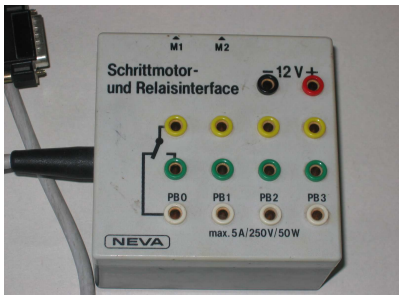
Paralel arabirim (IC8255): 24 adet sayısal hat vardır.Bu hatlar her porta 8 hat olmak üzere üç porta bölüştürülmüştür. Bu 8 hat, 0'dan 7'ye kadar numaralandırılmıştır.

Sayıcı arabirimi (IC8253): Üç adet basamaklandırılabilir 16 bit sayıcı vardır. Bunlardan iki tanesi dış sayıcı şeklinde kullanılabilir veya interrupt kaynağı görevi yapabilirler. Adresler:Birinci için (sayıcı0): Kart adresi+16, İkinci için (sayıcı1): Kart adresi+17, üçüncü için (sayıcı2) : Kart adresi+18'dir.Ayrıca kart üzerinde bir tane 2 MHz kuartz osilatör (sayaç), bir de, interrupt bağlantısı vardır.

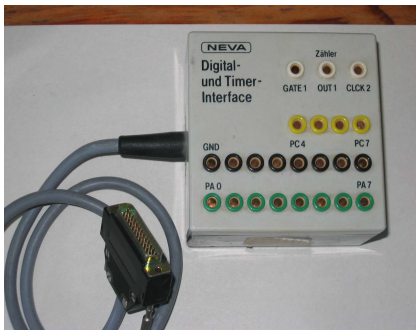
Bu deney kartına bağlanan arabirim elemanları:

Analog sayısal çevirici için iki kanallı bir ön amplifikatör (şekil 3.5), adım motoru ve röle arabirimi (şekil 3.6), sayısal zaman arabirimi (şekil 3.7), dağıtıcı kutusu (şekil 3.8)'dur.

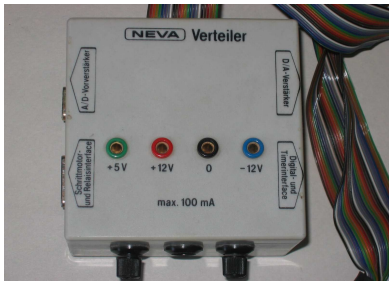
Şekil 3.5. Analog sayısal çevirici için iki kanallı bir ön amplifikatör.



Şekil 3.6. Adım motoru ve röle arabirimi.



Şekil 3.7. Sayısal zaman arabirimi.



Şekil 3.8. Dağıtıcı kutusu.

3.6. Genel Programlama

8255; Port A, Port B, Port C olmak üzere 3 porta sahiptir. Her port, kartın adresine bağlı olarak bir adrese sahiptir. Çizelge 3.6'da port adresleri verilmiştir.

Çizelge 3.6. Port adresleri (Dinçer ve Gürkan, 1999)

Port	Adres
A	Temel adres+0
B	Temel adres+1
C	Temel adres+2

Her port 8 tane hattan (8 bit) oluşmuştur. Bir hat, portu gösteren önek (A, B veya C) ve numarası ile belirtilir. Önek hangi porta ait olduğunu gösterir. Örneğin; B0 Port B'nin ilk hattını, C3 Port C'nin 4.hattını gösterir.

Bir portu giriş veya çıkış olacak şekilde programlayabiliriz. 8255 PC arayüz kartı, üç adımda programlanır. İlk önce, her portun (giriş veya çıkış) oynayacağı rol belirlenir ve ilgili kontrol kelime numarası seçilir. İkinci olarak; kontrol kelime numarası, kontrol kelime adresine, kullanılan programlama dilinin çıkış komutu kullanılarak yazılır. Kullandığımız C programlama dilinde çıkış komutu "outportb" dir. Bir portu giriş veya çıkış olarak programlamak kullanım amacımıza göre değişir. Örneğin; "giriş" portu olarak programlanırsa, PC'yi bir sıcaklık sensörü olarak

kullanabiliriz, "çıkış" portu olarak olarak programlanırsa PC ile bir motoru kontrol edebiliriz. Bu portları çıkış portu olarak kulanabilmek için Ek 6'da verilen P_OKU_YAZ.C programı kullanılabilir.

Üçüncü adım olarak da dış çevre birimlerini kontrol etmek için, çıkış portlarına sinyal gönderilmeli, herhangi bir dış birimi izlemek için ise giriş portlarından sinyal alınmalıdır. Porta veri yazma şu şekilde yapılır: Bir çıkış portuna, portun adresini ve "outportb" komutunu kullanarak veri yazarız.

Her port hat başına bir bit atanmış 8 sayısal hat içerir. Bir hat mantıksal olarak "1" (yani mantıksal seviye "YÜKSEK") ise hattan +5 volt okunacak demektir. Eğer mantıksal olarak "0" (yani mantıksal seviye olarak "DÜŞÜK") ise hattan 0 volt okunur. Bir hattı "YÜKSEK" yapmak için bitini "1" yapmak gerekir. "Düşük" yapmak için ise bitini "0" yapmak gerekir. Örneğin hangi hatlar yüksek yapılmak isteniyorsa, bu hatların bitlerini"1", geri kalanları ise "0" yapmamız gerekir. Çizelge 3.7'de port A'nın sayısal hatlarının ayarlanması için yapılması gerekenler gösterilmiştir.

Çizelge 3.7. Port A'nın sayısal hatlarının ayarlanması

A7	A6	A5	A4	A3	A2	A1	A0
Kapalı	Açık	Kapalı	Açık	Kapalı	Açık	Kapalı	Açık
0	1	0	1	0	1	0	1
0	+5V	0	+5V	0	+5V	0	+5V

Bu ikilik olarak 01010101 ve ondalık olarak 85 dir. Eğer "outportb (PortA,85)" komutunu P_OKU_YAZ.C programına ekleyip çalıştırırsak ve bir AVO metreyle portların hat gerilimlerini ölçersek A0, A2, A4, A6 hatları +5 ve A1, A3, A5, A7 hatları ise 0 V gösterecektir.

Bir porttan okuma yapmak için "inputb " komutu kullanılır. 8 sayısal hattan +5V uygulananın biti "1", şasiye bağlanan hattın biti "0" olur.Örneğin B0, B1, B4,

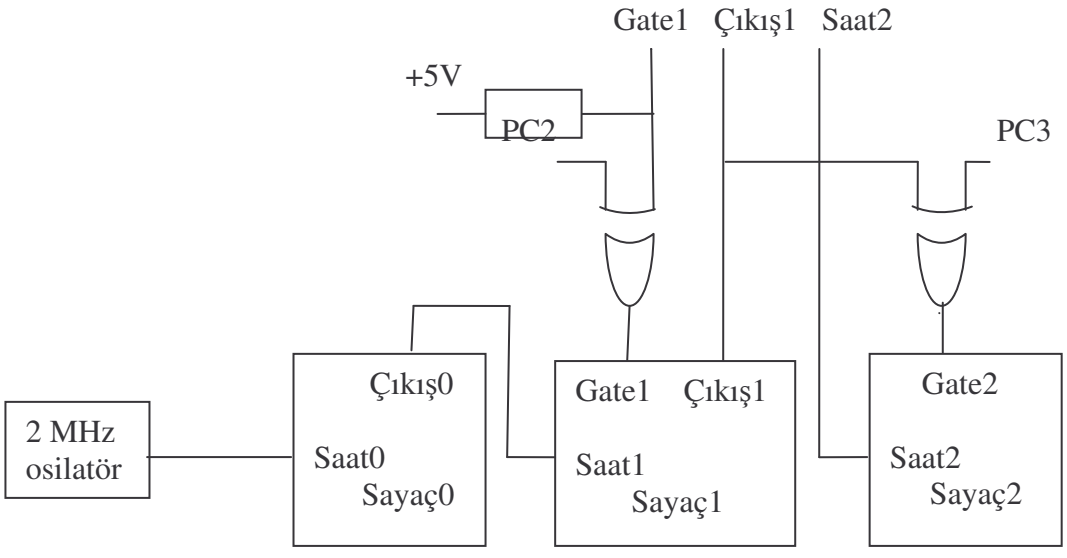
B6'ya +5 V uyguladığımızı ve B2, B3, B5, B7'yi şasiye bağladığımızı düşünelim.Ek 7'de verilen PORT_OKU.C programını yazıp çalıştıralım. PORTB, "83 " değerini (ikilik olarak 01010011) göstermelidir. PORT C'de A ve B gibi 8 adet sayısal hat içerir ve PORT C'yi 4'lü gruplar halinde ikiye bölebiliriz. Bu alt gruplara "Port C Alt" (C0-C3 hatları) ve "PORT C Üst" (C4-C7 hatları) adı verilir. Bu hatları giriş veya çıkış hatları yapmamız mümkündür. Çizelge 3.8'de kontrol kelimesi (dirreg) numarası (mode) seçimi gösterilmiştir.

Çizelge 3.8. Mode 0 programlama seçiminde giriş ve çıkış konfigürasyonları

PORT A	PORT B	PORT C ALT	PORT C ÜST	KONTROL KELİMESİ ONDALIK/ONALTILIK
Çıkış	Çıkış	Çıkış	Çıkış	128 / 80
Çıkış	Çıkış	Giriş	Çıkış	129 / 81
Çıkış	Giriş	Çıkış	Çıkış	130 / 82
Çıkış	Giriş	Giriş	Çıkış	131 / 83
Çıkış	Çıkış	Çıkış	Giriş	136 / 88
Giriş	Çıkış	Çıkış	Çıkış	144 / 90
Çıkış	Çıkış	Giriş	Giriş	137 / 89
Çıkış	Giriş	Çıkış	Giriş	138 / 8A
Çıkış	Giriş	Giriş	Giriş	139 / 8B
Giriş	Çıkış	Giriş	Çıkış	145 / 91
Giriş	Giriş	Çıkış	Çıkış	146 / 92
Giriş	Giriş	Giriş	Çıkış	147 / 93
Giriş	Çıkış	Çıkış	Giriş	152 / 98
Giriş	Çıkış	Giriş	Giriş	153 / 99
Giriş	Giriş	Çıkış	Giriş	154 / 9A
Giriş	Giriş	Giriş	Giriş	155 / 9B

3.7. Zaman Ölçümü

Sayıcı 0 ve sayıcı 1 sayısal-zaman arabirimine şekil 3.9'da görüldüğü gibi bağlanmıştır. Osilatör 2MHz ile 0 numaralı sayacı besler. Bu sayacın kaydedicisinde (register) bir sayı vardır (örneğin 32+78*256=20000). Önce düşük bayt 32, sonra yüksek bayt 78 yüklenir. Sayaç 20000 den aşağıya doğru saymaya başlar. Sonuç 0 olduğunda, 1 numaralı sayaca bir puls gönderir. Frekans 2 MHz olduğundan 1 nolu sayaç her 0.01 s'de bir puls alır. Bir nolu sayaç başlangıçta 255+255*256=65535 ile yüklenmiştir. Bu sayı eğer Gate 1 voltajı yüksekse, 100 Hz lik bir hızla azalır. Bu Gate'in durumu sayısal zaman arabirimindeki GATE 1 konnektörü ile belirlenir. Bu konnektör sayaca bir XOR gate'i ile bağlandığından PC'nin durumuna (yani Port C de 0 veya 4 olmasına) bağlı olarak GATE 1 konnektörünün açık veya kapalı olması sayacın Gate 1'inde yüksek voltaj olmasına neden olur. Sayaçların (IC8253) çalışma modları Counter Register'a yazılan çeşitli baytlarla değiştirilebilir. Bu sayaçları kullanarak gerçek zamanda veri kaydı alınabilir.



Şekil 3.9. Zaman ölçümü.

4. ARAŞTIRMA ve BULGULAR

Bu bölümde, bu çalışmada geliştirilen deneylerin yapılışı ve sonuçları anlatılacaktır.

4.1. Basit Sarkaç

4.1.1. Amaç

Sarkacın genliğinin ve periyodunun deneysel olarak ölçülerek tespit edilmesi.

4.1.2. Araç ve Gereçler

1) Sarkaç topu ve ip
2) Optik kapı
3) Açı ölçer
4) Üreteç
5) Yazıcı portunun data kablosu kullanarak yapılan kutu
6) Sayısal zaman arabirimi
7) Dağıtıcı kutusu

4.1.3. Teori

Basit sarkacın periyodu küçük açılar için iyi bilinen $2\pi\sqrt{\ell/g}$ formülü ile verilir. Tam çözüm ise eliptik bir integraldir. Periyot bu eliptik integral cinsinden 4.1.1'de verilmiştir.

$$T = 2\pi\sqrt{\frac{\ell}{g}}\int \frac{da}{\left[\sin^2\left(A/2 - \sin^2\left(a/2\right)\right)\right]^{1/2}} \qquad (4.1.1)$$

Burada A genlik açısıdır. Bu integral A için seriye açılırsa aşağıdaki eşitlik bulunur.

$$T = 2\pi\sqrt{\ell/g}\left\{1 + (1/2)^2 \sin^2(A/2) + [(1.3)/(2.4)]^2 \sin^4(A/2) + ...\right\} \qquad (4.1.2)$$

Standart metodda, bir kızılötesi ışın sarkaç ipi tarafından her bir tam salınımda iki kez kesilir ve periyod ölçülür. Salınımın sadece periyodunu değil, genliğini de bulmak için sistemi sarkacın denge konumundan kaydırdık. Şekil 4.1'de C noktası, B merkezi konumuna göre D uzunluğu kadar ötelenmiş haliyle kızılötesi ışının yeni konumunu gösterir. O eksenini ve C ışınını içeren düzlem, sarkacın O eksenini içeren düşey düzlem ile sabit bir α açısı yapar. OB uzunluğunu L olarak gösterirsek (sarkaç uzunluğu olan ℓ den farklıdır) α açısı 4.1.3 ile verilir.

$$\alpha = \tan^{-1}\left(\frac{D}{L}\right) \qquad (4.1.3)$$

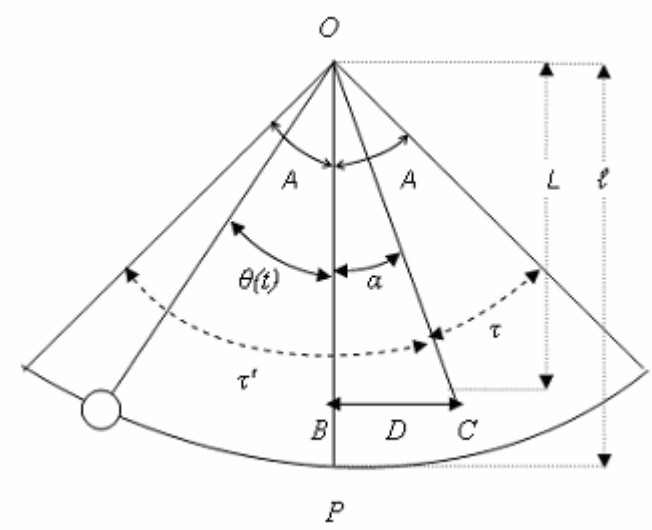
Böylelikle deney düzeneği kullanılarak τ ve τ' ölçülür. Burada $\tau' + \tau = T$ 'dir. Küçük salınımların yaklaşımından, sahip olduğumuz bağıntılardan birisi 4.1.4'dür.

$$\theta(t) = A\cos\left(\frac{2\pi}{T}t\right) \qquad (4.1.4)$$

A genliğini bulmak için kullanılacak denklem 4.1.5'dir.

$$A = \frac{\alpha}{\cos(\pi\tau/T)} \qquad (4.1.5)$$

Buradaki τ ölçülen zaman aralıklarından kısa olanıdır (Cortini,1992).

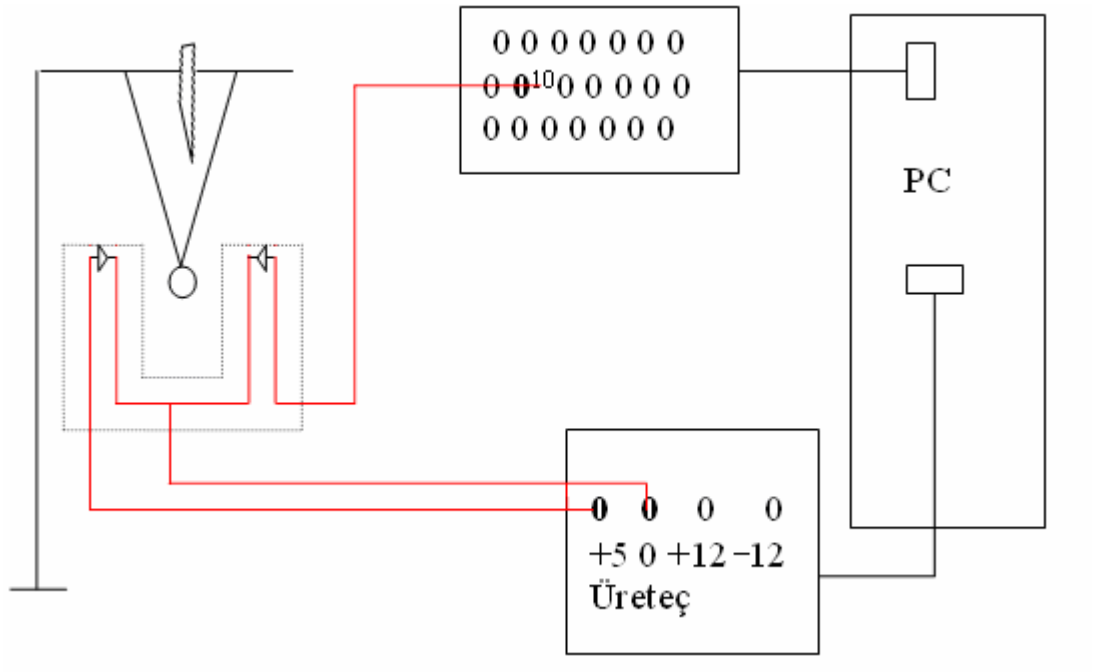


Şekil 4.1. Genliği ve periyodu bulmak için kullanılan parametreleri gösteren geometrik çizim (Cortini,1992).

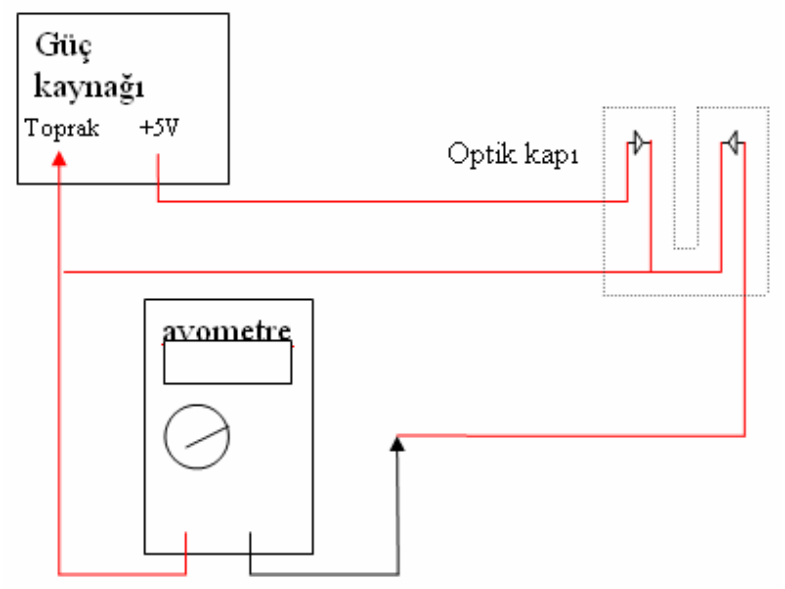
4.1.4.Deneyin Yapılışı

Bu deneyi iki farklı yöntem kullanarak yapacağız.Birinci yöntemde şekil 4.2'de gösterilen deney düzeneği kullanarak veriler STATUS portundan bilgisayara aktarılır.

Optik kapı şekil 4.1'de gösterildiği gibi *C* noktasına yerleştirilir. Sarkaç salınım yaparken Ek 8'de verilen SARKAC_PRINTER.C programı çalıştırılarak *T* periyodunun ölçülmesi sağlanır. Programda *C* programlama dilinin clock fonksiyonunu kullanarak zaman ölçülür. Optik kapı açıkken bilgisayarda okunan sinyal değeri 127 dir. Sarkaç kapının önünden geçerken, optik kapı kapandığında yazıcı portunun 10. pini topraklanır ve okunan değer 63 olur. Deneyde kullandığımız optik kapının alıcı LED çıkış kablosundan okunacak voltaj değerini voltmetre kullanarak öğrenebiliriz. Bunun için şekil 4.3'deki devreyi kuralım. Optik kapı kapalı iken alıcı LED'den gelen çıkış kablosunun bağlı olduğu pinin değeri sıfır olur. Bu, deney düzeneğinde okunan lojik değerin "1" olduğu duruma karşılık gelir. Kapı açıkken data kablosu topraklanır ve voltmetrede +5V okunur. Bu, deney düzeneğinde 10. pin'de okunan lojik değerin "0" olduğu duruma eşdeğerdir.

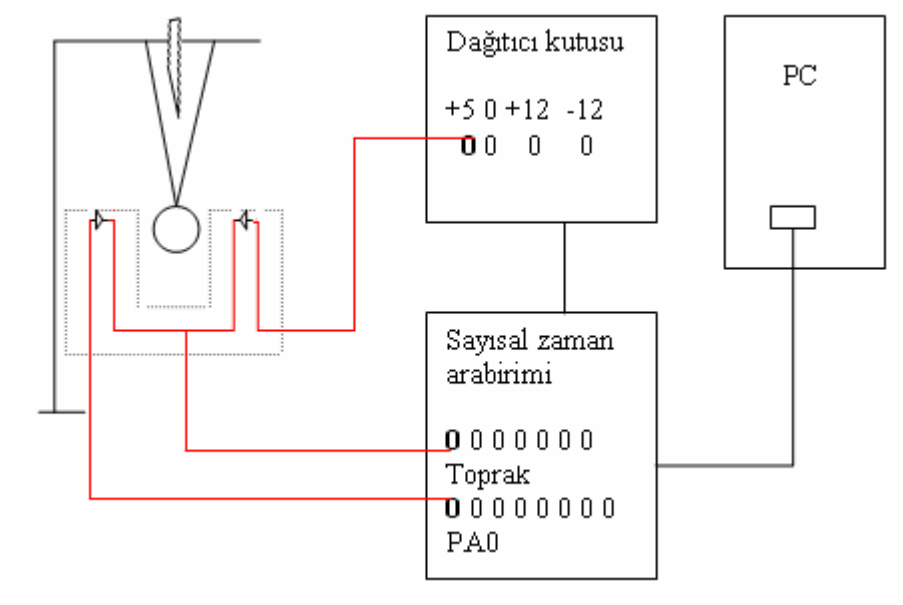


Şekil 4.2. STATUS portundan veri girişi yaparak basit sarkacın periyodunu ölçebilmek için kurulan deney düzeneği.



Şekil 4.3. Optik kapının alıcı LED kablosundan okunacak voltaj değerini voltmetre kullanarak öğrenebilmek için kurulacak devre.

İkinci yöntemle ölçüm yaparken, sayısal zaman arabirimi, dağıtıcı kutusu ve deney kartı kullanılacaktır. Bu deneye ait olan deney düzeneği şekil 4.4'de verilmiştir. Ek 9'da verilen SARKAC_PortA.C programı sarkaç periyodunu ve genliğini verecektir. Optik kapı yine şekil 4.1'de belirtilen *C* noktasına yerleştirilir.



Şekil 4.4. Sayısal arabirim ve dağıtıcı kutusu kullanılarak kurulan deney düzeneği.

4.1.5. Deneyin Sonucu

Birinci yöntem uygulandığında ölçülen sonuçlar çizelge 4.1'de belirtilmiştir. Bu deney sırasında D=20 cm ve L=64,5 cm olarak kullanılmıştır.

İkinci yöntem uygulandığında bulunan aynı D ve L değerleri için ölçülen periyot ve genlik değerleri çizelge 4.2'de verilmiştir.

Sarkacın ip boyu 67,5 cm'dir. Bunu kullanarak periyodun 27°'lik genlik için teorik değeri hesaplandığında 1,671 s bulunur. 4.1.4 eşitliğinde bu periyod değeri kullanılarak τ'nun teorik değeri bulunur. Bu değer 0,467'dir. Tabloda okunan genlik değerleri ile açı ölçerden okunan açı değerleri birbirine çok yakındır.

Çizelge 4.1. STATUS portu kullanılarak bulunan ölçüm sonuçları

τ	τ'	T (periyod) (s)	A (genlik) (^{o})
0,494	1,154	1,648	29,241
0,439	1,154	1,592	26,553
0,440	1,154	1,594	26,553
0,440	1,209	1,649	25,691
0,385	1,209	1,594	23,681

Çizelge 4.2. Sayısal arabirim ve dağıtıcı kutusu kullanılarak bulunan ölçüm sonuçları

τ	τ'	T (periyod) (s)	A (genlik) (°)
0,500	1,161	1,661	29,130
0,499	1,170	1,669	28,890
0,490	1,169	1,659	28,443
0.488	1,180	1,668	28,200
0,480	1,180	1,660	27,870

4.2. Eğik Düzlem

4.2.1. Amaç

Eğik düzlemde hareket eden cismin ivmesinin ve eğik düzlem ile cisim arasındaki sürtünme katsayısının bulunması.

4.2.2. Araç ve Gereçler

1) Eğik düzlem
2) Eğik düzlemde kayan cisim
3) 7 adet ışık kapısı
4) Sayısal-zaman arabirimi
5) Güç kaynağı

4.2.3. Teori

Newton'un ikinci yasası'na göre bir cismin ivmesi o cisme uygulanan net kuvvetle doğru orantılı, kütlesi ile ters orantılıdır. Cisme etki eden net kuvvet, kütle ve ivmenin çarpımına eşittir. Newton'un ikinci yasası eşitlik 4.2.1'de verilmiştir.

$$F_{net} = ma \tag{4.2.1}$$

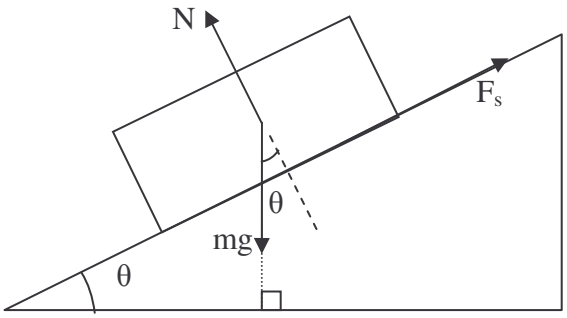
m kütleli cisim eğim açısı θ olan sürtünmeli eğik düzlem üzerinde kayarken cisme etkiyen kuvvetler şekil 4.5'de gösterilmiştir. Kütleyi aşağıya doğru hareket ettiren net kuvvet 4.2.2 bağıntısında verilmiştir.

$$F_{net} = mg\sin\theta - kmg\cos\theta \tag{4.2.2}$$

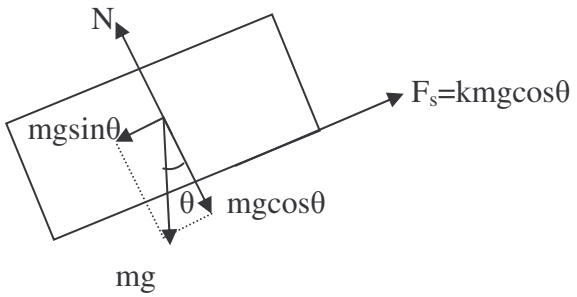
Bu denklemde verilen k kinetik sürtünme katsayısıdır. Cismin ivmesini ve cisimle yüzey arasındaki sürtünme katsayısını veren bağıntılar sırasıyla 4.2.3 ve 4.2.4'dür.

$$a = g\left(\sin\theta - k\cos\theta\right) \tag{4.2.3}$$

$$k = \frac{\left(g\sin\theta - a\right)}{g\cos\theta} \tag{4.2.4}$$



Şekil 4.5. Sabit bir eğik düzlem üzerinde kayan cisme etki eden kuvvetler.



Şekil 4.6. Eğik düzlemde kayan cisim için serbest cisim diagramı.

4.2.4. Deneyin Yapılışı

Deneyi yapmak için üzerinde 7 tane ışık kapısı bulunan bir eğik düzlem tasarlanmıştır. Verici ve alıcıların negatif uçları birleştirilir ve sayısal zaman arabirimindeki toprağa bağlanır. Vericilerin pozitif uçları da birleştirilir ve 220 ohm luk bir dirence bağlanır. Bu direncin diğer ucu da dağıtıcı kutusundaki +5V'a bağlanır. Alıcıların pozitif uçları PA0 dan başlayarak sırasıyla sayısal zaman arabirimindeki girişlere bağlanır. Işık kapılarının hepsi açıkken alıcıların bağlı olduğu pinin mantıksal değeri "1" olur. Kapandığında ise "0" olur. Eğik düzlem üzerinde 7 tane kapı bulunduğu için Port A'nın 7 tane pini kullanılmaktadır.

Kullanılmayan 8. pinin mantıksal değeri "0" dır. Çizelge 4.3'de kapıların bağlı olduğu port pini ve kapandıklarında okunan sayısal veriler yazılmıştır.

Çizelge 4.3. Eğik düzlemde kapıların bağlı olduğu pinler ve kapandıklarında okunan değerler

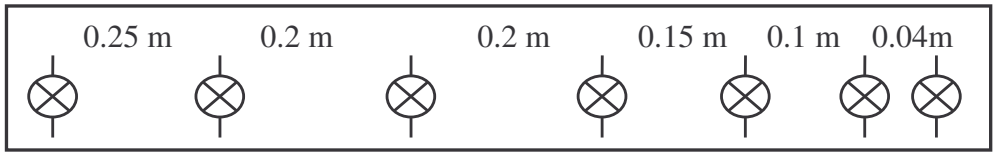
Port pinleri	PA6	PA5	PA4	PA3	PA2	PA1	PAO
Kapı numarası	7	6	5	4	3	2	1
Kapandığında okunan sayısal veri	63	95	111	119	123	125	126

Kapıların tümü açıkken 7 pin için de mantıksal değer 1 olduğundan, bit değerleri toplamı olan 127 sayısal verisini okuruz. Kapanan kapı için bit değeri 0, kalan kapıların ait olduğu pinlerin bit değerlerini toplayarak tablodaki değerler bulunmuştur. Örneğin 4. kapı kapandığında 8 bitlik veri 01101111 olarak alınır. Bit değerleri toplandığında 0+64+32+16+0+4+2+1=119 bulunur.

Ek 10'da verilen EGIK_DUZLEM.C programı çalıştırılır. Eğik düzlemin yatayla yaptığı açı ölçülerek programa kaydedilir. 1.5x2.5x3.0 cm boyutlarındaki demir blok eğik düzlem üzerinde kaymaya bırakılır. Cisim ilk kapıyı kapattığı anda zaman ölçümü başlatılır. Diğer kapıların kapanma zamanları da ölçülür. Bu ölçüm eğik düzlem boyunca sürdürülür. Bulunan her zaman aralığı bir öncekine eklenerek kaydedilir. Daha sonra bu ardışık zamanların farkları alınarak cismin aralıkları geçiş süresi hesaplanır. Kapılar arası mesafeler kullanılarak cismin aralıkları geçerken sahip olduğu ortalama hızlar bulunur. Cismin ardışık aralıklardaki ortalama hızlarının farkı bulunur. Daha önce bulunan zamanlar kullanılarak cismin aralıkların orta noktasını geçiş zamanları bulunur. Ortalama hızların farkı, bu zaman değerlerine bölünerek ivme hesaplanır, daha sonra bu ivme değerlerinin ortalaması bulunur. 4.2.4 bağıntısı kullanılarak sürtünme katsayısı k hesaplanır. Bu deney 20, 30 ve 40

derece için yapılmıştır. İlk aralıkta bulunan değerler anlamsız olduğundan hesaplarda kullanılmamıştır.

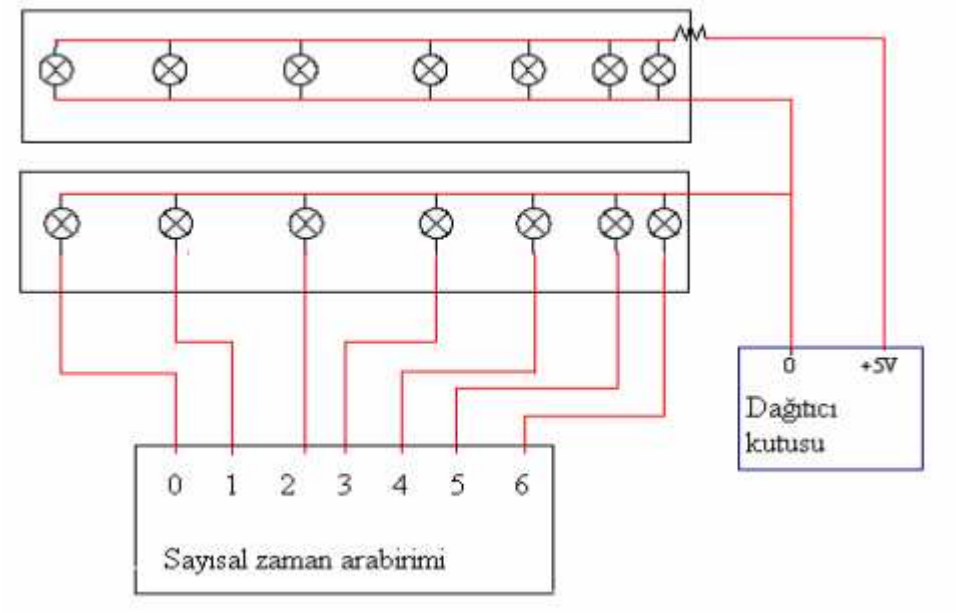
Optik kapıların arasındaki mesafeler şekil 4.7'de verilmiştir.



Şekil 4.7. Optik kapıların eğik düzlem üzerine yerleşimi.

4.2.5. Deneyin Sonucu

Şekil 4.8'deki deney düzeneğini kullanarak yapılan deneyde belirlenen açı (α) değerleri için kullanılan aralıklar (Δx) ile bulunan süre (t), hız (v) ve ivme (a) değerleri çizelge 4.4'de, ortalama ivme (a_{ort}) ve sürtünme katsayıları (k) çizelge 4.5'de verilmiştir.



Şekil 4.8. Eğik düzlem deney düzeneği.

Çizelge 4.4. Eğik düzlemde kullanılan aralıklar ve bulunan süre, hız, ivme değerleri

α (°)	Δx (m)	t (s)	V (m/s)
25	0,10	2,29	0,53
	0,15	2,47	0,83
	0,20	2,65	1,14
	0,20	2,81	1,42
	0,25	2,95	1,73
30	0,10	3,21	0,64
	0,15	3,36	1,01
	0,20	3,50	1,38
	0,20	3,64	1,68
	0,25	3,76	1,98
40	0,10	1,85	0,76
	0,15	1,97	1,20
	0,20	2,10	1,63
	0,20	2,21	2,01
	0,25	2,31	2,38

Çizelge 4.5. Eğik düzlem deneyinde bulunan ivme, ortalama ivme, sürtünme katsayısı değerleri

α (°)	a (m/s^2)	a_{ort} (m/s^2)	k
25	1,63 1,73 1,79 2,17	1,83	0,214
30	2,44 2,55 2,23 2,44	2,42	0,219
40	3,44 3,48 3,41 3,65	3,49	0,219

Tahta-metal sürtünme katsayısı teorik olarak 0,2-0,6 aralığındadır. Deneyde kullanılan açılardan biri olan 30° için ivmenin beklenen değeri 4.2.4 bağıntısı kullanılarak hesaplanırsa, 0,192-3,206 aralığında olduğu görülür. Deneysel olrak bulunan sonuç da beklenen değer aralığındadır. Aynı zamanda ivme, açının artışından etkilenmiş ve artmış, sürtünme katsayısı da birbirine çok yakın değerlerde çıkmıştır. Bu da beklenen bir sonuçtur.

4.3. Eylemsizlik Kütlesi

4.3.1. Amaç

Eylemsizlik kütlesini ölçmek. Eylemsizlik kütlesi ile çekim kütlesini karşılaştırmak.

4.3.2. Araç ve Gereçler

1) Eylemsizlik terazisi
2) 5 adet silindirik kütle
3) Dağıtıcı kutusu
4) Sayısal zaman arabirimi

4.3.3. Teori

Eylemsizlik kütlesi, bir cismin eylemsizliğinin ölçüsüdür. Başka bir deyişle, bir kuvvet uygulandığında cismin var olan durumunu değiştirmeye karşı olan direncidir.

Eylemsizlik terazisi, cisimlere yatay düzlemde titreşim hareketi yaptırır. Kütleleri bilinen cisimler için hareket periyotları tespit edilerek karşılaştırma metoduyla bilinmeyen kütleler bulunabilir. Bir cismin bu metodla bulunan kütlesine eylemsizlik kütlesi denir. Bu metodla kütle ölçümü sırasında cisimlerin eylemsizliklerinden dolayı hareket durumlarındaki değişmelere karşı gösterdikleri dirençler karşılaştırılır.

Eylemsizlik terazisinde kütlelerle titreşim periyotları arasındaki ilişki, $\frac{m_1}{m_2} = \frac{T_1^{\,2}}{T_2^{\,2}}$ şeklinde ifade edilir (Şahan,1999). Burada m_1 ve m_2 farklı kütleler ve T_1 ve T_2 bu kütlelere ait titreşim periyotlarıdır. Eylemsizlik terazisi ile kütle ölçümü yapılırken, kütleleri bilinen cisimlerin terazi yardımıyla titreşim periyotları ölçülür. Periyot ve kütle değerleri ile bir grafik çizilir. Kütlesi bilinmeyen cismin de titreşim periyodu ölçülerek grafikte yerine konulup grafik üzerinde bu periyot değerine karşılık gelen kütle değeri işaretlenir. Böylece cismin kütlesi bulunmuş olur. Deney sırasında hep aynı terazi kullanılacağından terazinin kendi kütlesinin diğer bütün kütlelere etkisi aynı olur. Bundan dolayı terazinin kütlesinin deney sonuçlarına bir etkisi yoktur.

4.3.4. Deneyin Yapılışı

Tartı yardımıyla beş adet silindirik cismin kütlelerini ölçerek programda kaydederiz. Şekil 4.9'daki düzeneği kurdukdan sonra ek 11'de verilen KÜTLE.C programı çalıştırılır. Bu program yardımıyla terazinin kefelerine yerleştireceğimiz 1, 2, 3, 4 ve 5 adet silindirik kütlenin ve teraziyi boşaltarak, bir kefeye koyduğumuz bilinmeyen kütlenin 20 salınım yapması için geçen süreyi optik kapı yardımıyla ölçeriz, bu zaman değerlerini kullanarak periyotları ve periyotların karelerini hesaplarız, Şekil 4.10'da görüldüğü gibi sırasıyla 1, 2, 3, 4 ve 5 adet kütleye karşılık gelen periyodun karelerinin yatay eksende, kütle değerlerin ise düşey eksende bulunduğu bir grafik çizilir. Son olarak da grafiğin eğiminden yararlanarak bilinmeyen kütlenin değerini hesaplarız.

4.3.5. Deneyin Sonucu

Deneyin sonucunda kütle değerlerine karşılık okunan periyod kare (T^2) değerleri çizelge 4.6'da belirtilmiştir. *m*' bilinmeyen kütle değerini belirtmektedir.

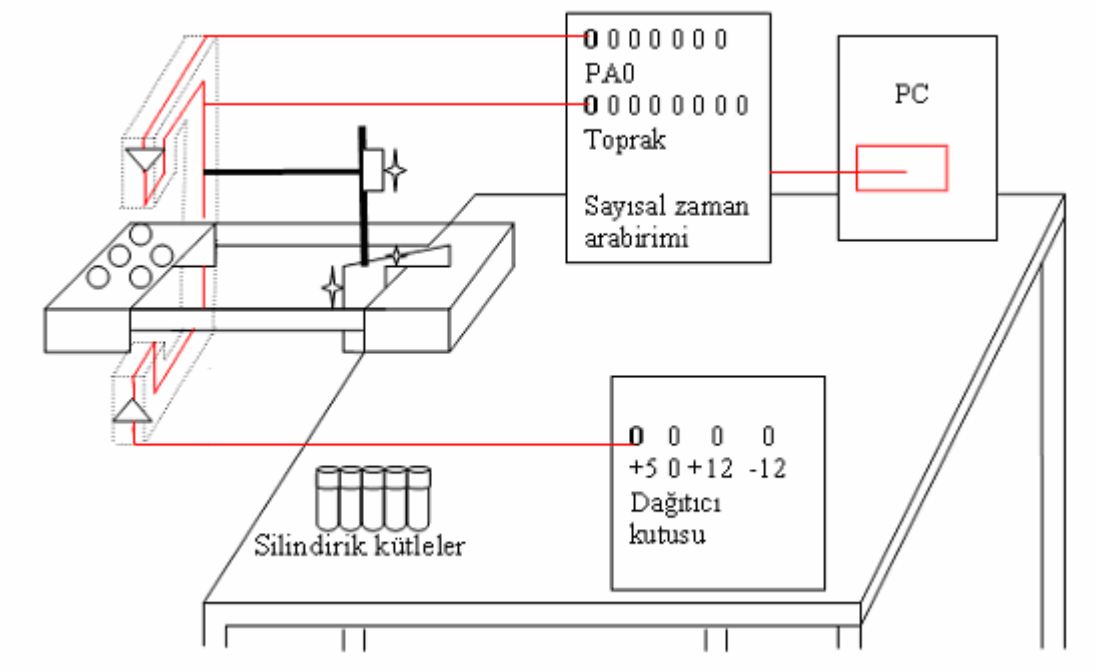
Tablodaki değerleri kullanarak çizilen grafik şekil 4.9'da verilmiştir. Bilinmeyen kütle için ölçülen periyod kare değeri ile eğim çarpıldığında bulunan

değer m' kütle değeri olur. Bu değerin bulunuşunda kullanılan bağıntı 4.3.1'de verilmiştir.

$$m' = eğim * T^2 \qquad (4.3.1)$$

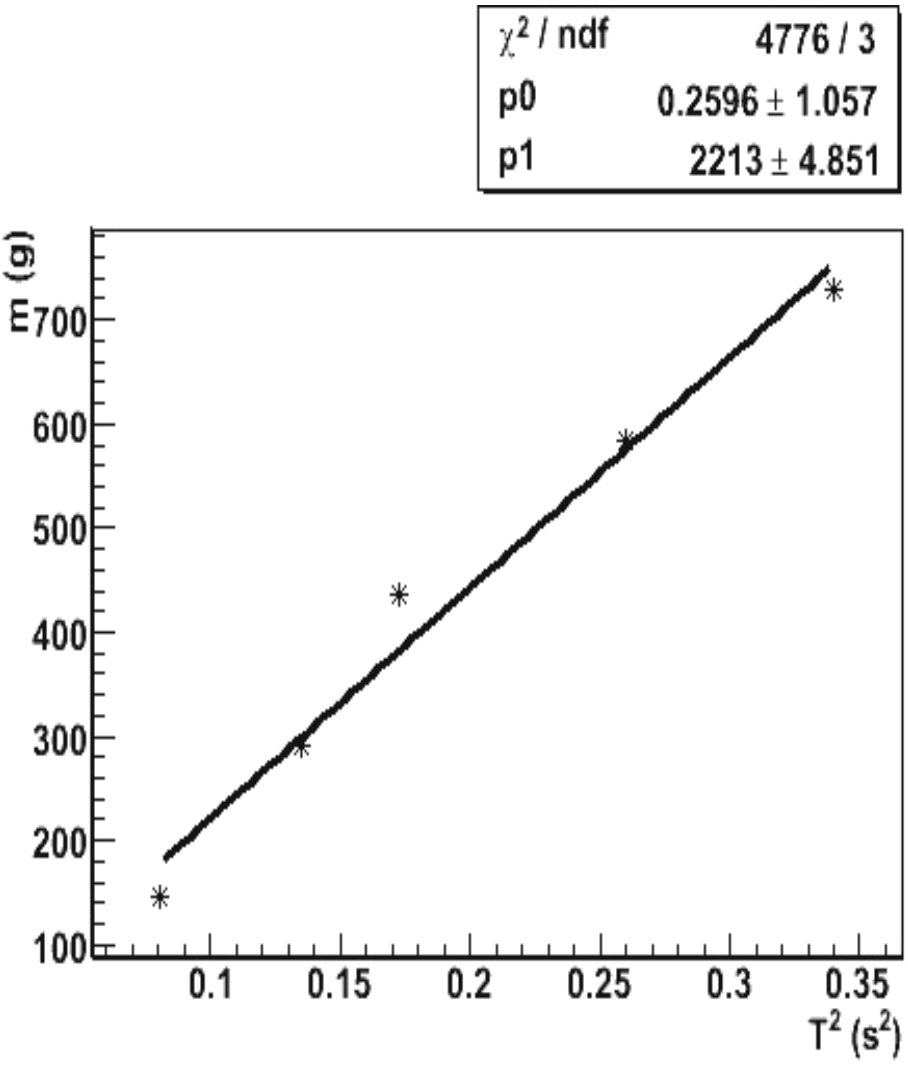
Çizelge 4.6. Kullanılan kütle değerlerine karşılık ölçülen periyod ve periyod kare değerleri

m (g)	T (s)	T^2 (s^2)
146	0,284	0,080
291	0,367	0,135
437	0,416	0,172
584	0,510	0,260
729	0,583	0,340
m'	0,367	0,134



Şekil 4.9. Eylemsizlik kütlesinin bulunması için kurulan deney düzeneği.

Tablodaki T^2 değerini ve grafikte ölçülen eğim değerini 4.3.1'de yerine koyacak olursak bulunan sonuç (2213*0,134)=297 g olur. Bu cismin kütlesinin tartıdaki ölçüm sonucu (çekim kütlesi) 293 g'dır. Bu değer ile eylemsizlik terazisi kullanılarak bulunan değer birbirine çok yakındır. Bu nedenle eylemsizlik kütlesi ile çekim kütlesi eşittir diyebiliriz.



Şekil 4.10. Kütlenin periyodun karesiyle değişimi. Yukarıdaki kutuda P_0 parametresi doğrunun x eksenini kestiği noktayı, P_1 parametresi ise eğimi belirtir.

4.4. Tek Ve Çift Yarıkta Girişim

4.4.1. Amaç

Tek yarıkta kırınım ve çift yarıkta girişim sonucu oluşan girişim saçaklarını grafik olarak bilgisayar ekranında göstermek.

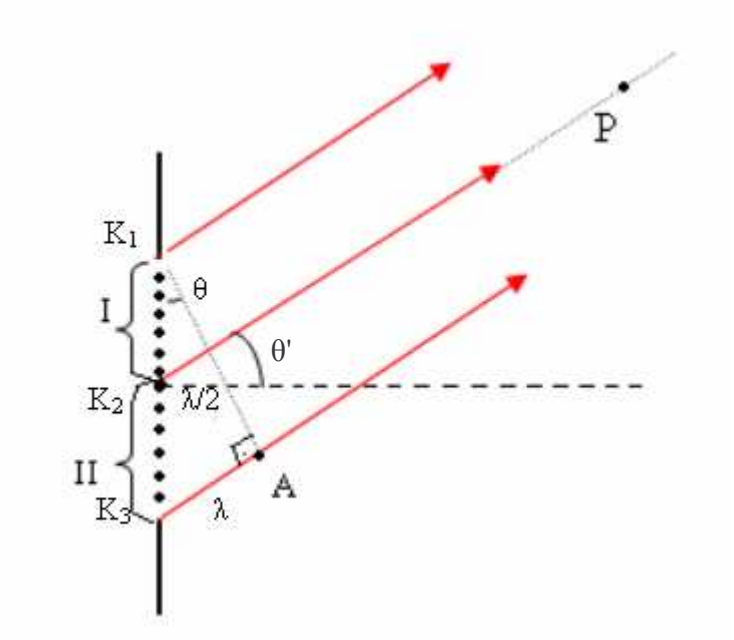
4.4.2. Araç ve Gereçler

1) Helyum-Neon lazer
2) Fototransistör
3) Potansiyometre
4) Üzerinde tek ve çift yarık bulunan siyah cam
5) Lazer sehpası
6) Direnç
7) A/D çevirici arabirimi

4.4.3. Teori

TEK YARIKTA GİRİŞİM (KIRINIM): Kırınım ışığın dalga modelini destekleyen bir olaydır. Şekil 4.11'deki tek yarığa paralel ışık demeti düşerse, Huygens prensibine göre, yarıkta her bir nokta aynı fazda ışık yayan kaynaklar gibi davranır ve aynı fazda titreşen kaynaklardan çıkan dalgalar, yarığın orta dikmesi üzerindeki noktalara eşit yollar alarak varır. Böylece birçok kaynaktan çıkarak merkez doğrusu üzerindeki noktalarda üst üste binen dalga tepeleri maksimum olacak ve merkez doğrusu boyunca aydınlık saçak meydana gelecektir. Şekil 4.11'de girişim noktaları, yarığın genişliğiyle kıyaslandığında çok uzaklarda olduğundan bu noktaya gelen ışık dalgaları birbirine paralel doğrultuda gidiyormuş gibi kabul edilir. Bu durumda yol farkı $|AK_3|$ kadar olacaktır. $|AK_3|$ yol farkı λ'ya eşit olunca K_1 ve K_2 yol farkı $\lambda/2$ olur. Bu dalgalar P noktasına ulaştıklarında birinin tepesiyle

diğerinin çukuru karşılaşacağından, birbirinin etkisini yok eder. *P* noktasında karanlık saçak meydana gelir.



Şekil 4.11. Tek yarıkta girişim. I. ve II. bölgeden $\lambda/2$ yol farkıyla gelen dalga çiftleri birbirlerinin etkisini yok eder.

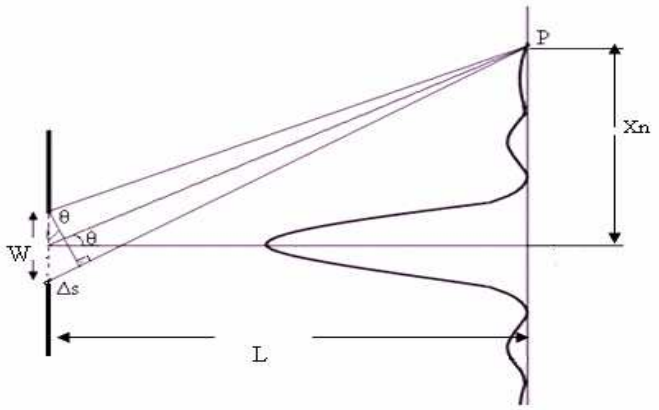
Tek yarığın ortasındaki K_2 noktasının her iki yanında eşit sayıda n tane nokta kaynağın sıralandığını düşünelim. K_1 ve K_2'den sonraki I. kaynaklar arasındaki yol farkı da $\lambda/2$'dir. Bu durumda 2., 3. ve n. kaynak çiftlerinden çıkan dalgaların yol farkları da $\lambda/2$'dir.

Sonuç olarak yarıktan *P* noktasına gelen ışık dalgaları, ikişer ikişer birbirlerinin etkisini yok eder ve *P* noktası karanlık olur.

Merkezi aydınlık saçak ve diğer saçaklar şekil 4.12'de verilmiştir.

Şekil 4.12. Tek yarıkta girişim sonucu oluşan aydınlık ve karanlık saçaklar.

Aydınlık ve karanlık saçakların oluşma koşullarını çıkarabilmek için şekil 4.13'ü inceleyelim.



Şekil 4.13. Tek yarıkta girişim düzeneğinde aydınlık ve karanlık şartlarının hesaplanmasında kullanılan uzunluklar.

Şekildeki P herhangi bir girişim deseninin ekran üzerindeki yeri, X_n bu noktanın merkez doğrusuna olan uzaklığıdır. Yarık genişliği W, P'nin merkez doğrusuna göre açısal yeri θ'dır. θ açısı çok küçük olduğundan sin θ=tanθ alınabilir. Yarığın kenarlarından çıkıp P'ye ulaşan ışınların yol farkı Δs ile diğer parametreler arasındaki bağıntı 4.4.1'de belirtildiği gibidir.

$$\frac{\Delta s}{W} = \frac{X_n}{L} \tag{4.4.1}$$

Yol farkı ile dalga boyu arasında 4.4.2 bağıntısında belirtilen bir eşitlik varsa P noktası karanlık saçak üzerinde, 4.4.3'deki eşitlik varsa P noktası aydınlık saçak üzerindedir. Bu bağıntılarda n tamsayı olup, P noktasının merkez doğrusundan itibaren kaçıncı saçak üzerinde olduğunu belirtir.

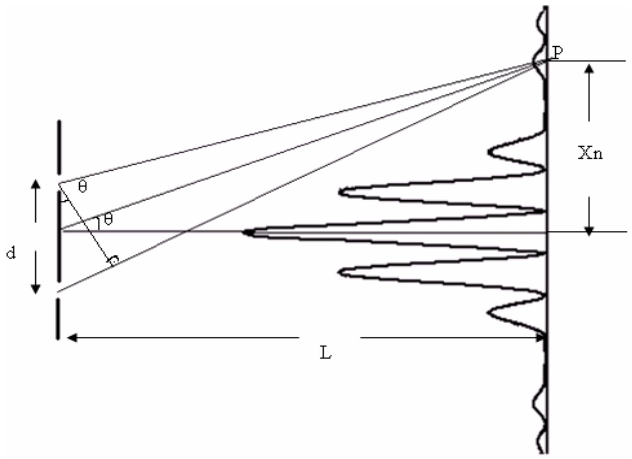
$$\Delta s = n\lambda, \quad n=1,2,3,.... \tag{4.4.2}$$

$$\Delta s = (n+1/2)\lambda, \quad n=1,2,3,.... \tag{4.4.3}$$

ÇİFT YARIKTA GİRİŞİM: Lazer yarıklara eşit uzaklığa konulduğunda, lazerden çıkan dalgalar, yarıklara eşit yollar alarak varırlar. Böylece bu yarıklar aynı fazda çalışan iki kaynak gibi davranır. Kaynaklardan çıkan ışık dalgaları, merkez doğrusuna eşit uzaklıkta olduğundan, perdede merkez doğrusu üzerinde birbirini kuvvetlendirmesi ile aydınlık saçak oluşur. Bu saçağa merkezi aydınlık saçak denir. Merkezi aydınlık saçağın sağında ve solunda simetrik olarak onu takip eden karanlık ve aydınlık saçaklar meydana gelir. Merkezi aydınlık saçak ve diğer saçaklar şekil 4.14'de verilmiştir.

Şekil 4.14. Çift yarıkta girişim sonucu oluşan aydınlık ve karanlık saçaklar.

Aydınlık ve karanlık saçakların oluşma koşullarını çıkarabilmek için şekil 4.15'deki düzeneği inceleyelim.



Şekil 4.15. Çift yarıkta girişim düzeneğinde aydınlık ve karanlık şartlarının hesaplanmasında kullanılan uzunluklar.

θ açısı çok küçük olduğundan $\sin\theta = \tan\theta$ alınabilir. Şekildeki P herhangi bir girişim deseninin ekran üzerindeki yeri, X_n de bu noktanın merkez doğrusuna olan uzaklığıdır. Yarıklar arasındaki uzaklık d, P noktasını kaynakların orta noktasına

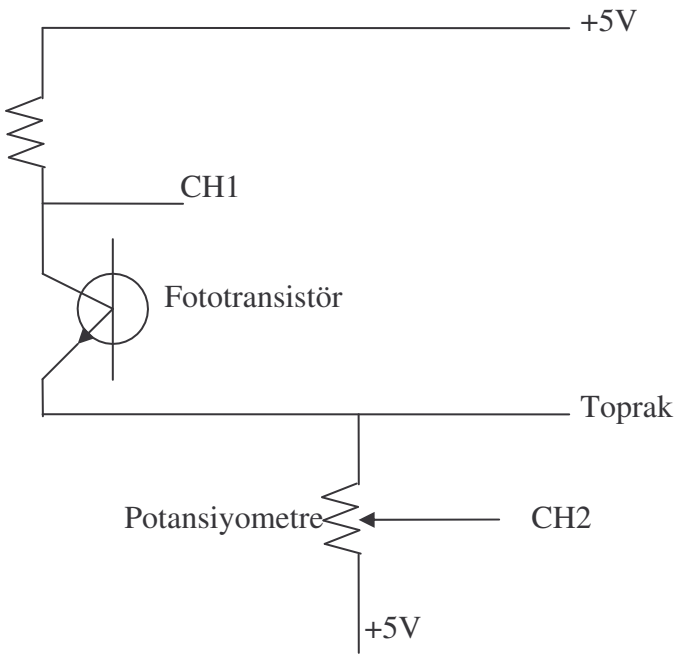
birleştiren doğru ile merkez doğrusu arasındaki açı θ, yarıklardan çıkıp P'ye ulaşan ışınların yol farkı Δs arasındaki bağıntı 4.4.1'de belirtildiği gibidir. Yol farkı ile dalga boyu arasında 4.4.4 bağıntısında belirtilen bir eşitlik varsa P noktası karanlık saçak üzerinde, 4.4.5 eşitliği varsa P noktası aydınlık saçak üzerindedir. Bu bağıntılarda n tamsayı olup, P noktasının merkez doğrusundan itibaren kaçıncı saçak üzerinde olduğunu belirtir.

$$\Delta s=(n\text{-}1/2)\lambda, \quad n=1,2,3,.... \tag{4.4.4}$$

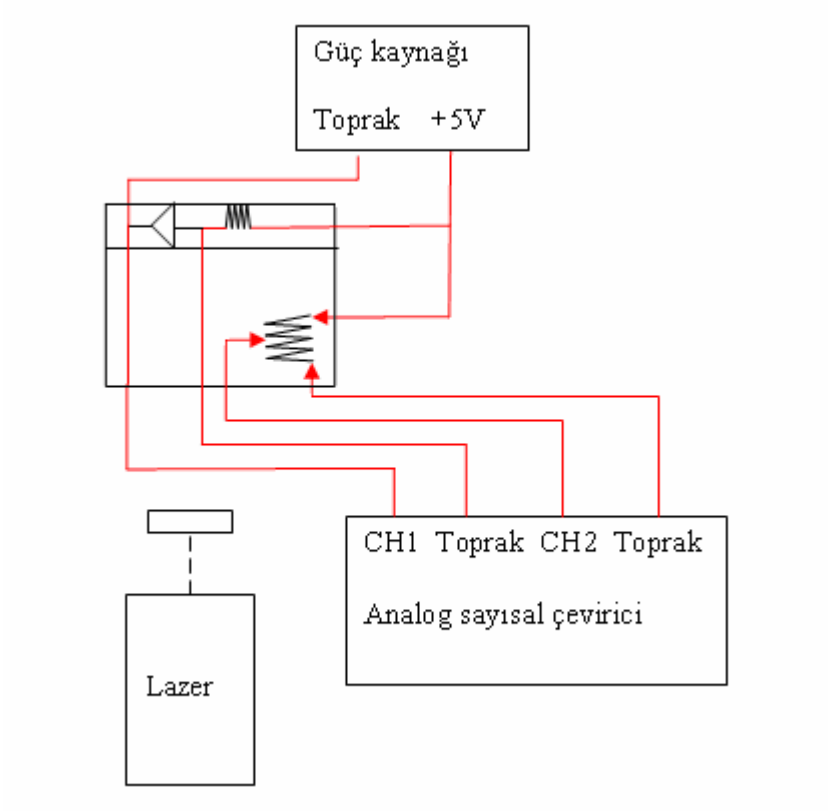
$$\Delta s=n\lambda, \quad n=1,2,3,.... \tag{4.4.5}$$

4.4.4. Deneyin Yapılışı

Fototransistör, direnç, ve A/D çevirici ile kurulan devrenin şeması şekil 4.16'da verilmiştir. Devre şeması da şekil 4.17'de verilmiştir.



Şekil 4.16. Tek ve çift yarıkta girişim deneyi için kullanılan devre şeması.

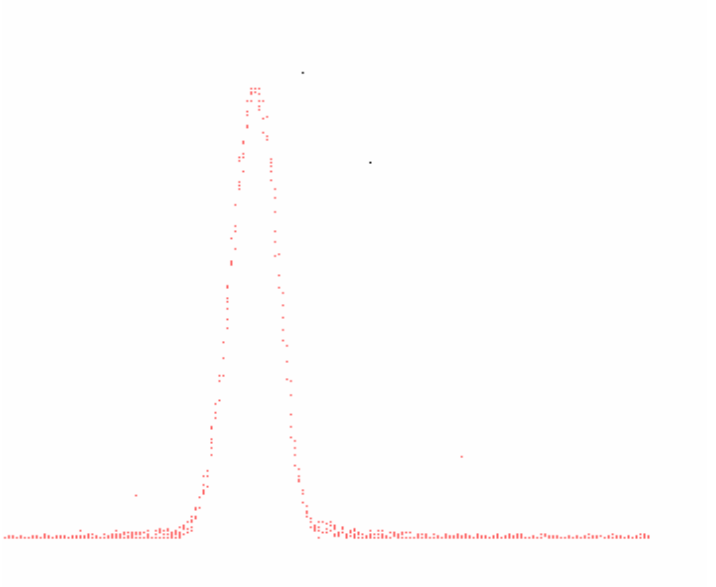


Şekil 4.17. Tek ve çift yarıkta girişim deneyi için kullanılan deney düzeneği.

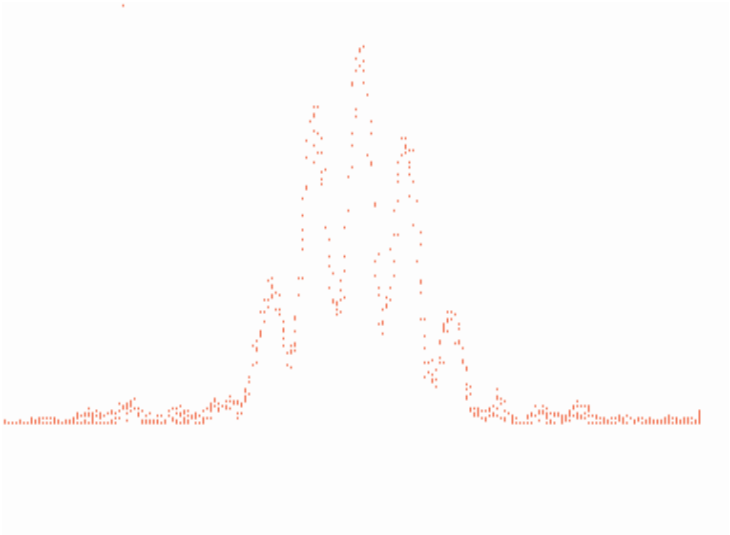
Lazerin karşısına, üzerinde tek ve çift yarıklar bulunan siyah cam yerleştirilir. Kullanılan tek yarığın genişliği 0,08 mm, çift yarığın merkezden merkeze aralık genişliği 0,250 mm' dir.Yarıkların arkasına yatayda hareket etmesi sağlanan fototransistör, potansiyometre ve A/D çevirici arabirimi vardır. Fototransistör, bir bilgisayar yazıcısının içinde bulunan raylara monte edilmiş ve düşük devirle hareket eden bir motorla yatayda hareket etmesi sağlanmıştır. Lazerden çıkan ışık cam üzerindeki yarıktan geçerek arkada bir girişim deseni oluşturur. Potansiyometrenin hareketli ucuna bağlı olan fototransistör girişim deseni üzerinde hareket ettirildiğinde, bu harekete bağlı olarak potansiyometrenin direnci değişir. Dirence bağlı olarak değişen potansiyometre voltajı, fototransistörün desen üzerindeki konumunu, fototransistörün akımı ise o konumdaki ışık şiddetini belirler. A/D çeviricinin 1.kanalı +100mV, 2.kanalında +5V kullanılır. Deney yapılırken kullanılan program ek 12'de verilen OPTIK. C programıdır.

4.4.5. Deneyin Sonucu

Tek ve çift yarıkla yapılan deneyin sonucunda şekil 4.18 ve 4.19'daki grafikler elde edilmiştir. Potansiyometrenin direncindeki değişme karşılık gelen konum grafikte *x* ekseni olarak, fototransistörün direncindeki değişime karşılık gelen ışık şiddeti ise grafikte *y* ekseni olarak gösterilmiştir.



Şekil 4.18. Tek yarıkta girişim için elde edilen girişim deseni.



Şekil 4.19. Çift yarıkta girişim için elde edilen girişim deseni.

4.5. RC Devresinde Zaman Sabitinin Bulunması

4.5.1 Amaç

RC devresinde zaman sabitinin bulunması ve bu zaman sabitinden ve bilinen bir direnç değerinden faydalanarak bilinmeyen sığa değerini bulmak.

4.5.2 Araç ve Gereçler

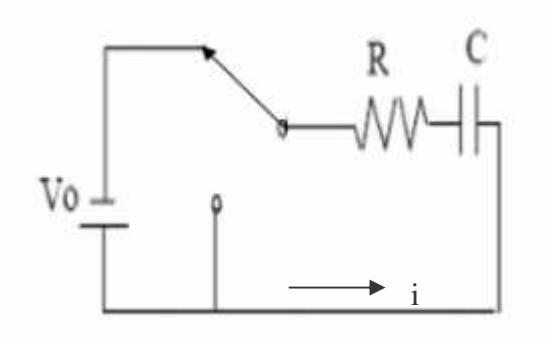
1) Direnç (47300, 4750, 624, 1271, 1500, 5000, 2700Ω)
2) Kondansatör(6.8, 2200, 22, 1000, 47μF)
3) Güç kaynağı
4) Anahtar
5) Analog-sayısal çevirici

4.5.3. Teori

Yapısında elektrolit adı verilen kimyasal emdirilmiş kağıt bulunan kutuplandırılmış kondansatörlere elektrolitik kondansatör denir. Yalnızca doğru akım devresine bağlanırlar ve artı kutbu artıya, - kutbu eksiye bağlanır. Kondansatörde depolanan elektrik yükü (q), yüklü iletkenler arasındaki V potansiyel farkı ile doğru orantılıdır. q ile V arasındaki orantı katsayısına kondansatörün sığası denir ve C ile gösterilir. Birimi faraddır. C, q, V arasındaki bağıntı 4.5.1'de verilmiştir.

$$C = \frac{q}{V} \tag{4.5.1}$$

Doğru akım üretecine bağlanan kondansatörün yüklenmesi için şekil 4.20'deki devre kurulur.



Şekil 4.20. Kondansatörün yüklenmesi.

Kondansatörün içinden yük geçmese de böyle bir devreden akım geçebilir. *RC* devresine bağlı olan üreteç, zamanla değişen bir akım oluşturur. Kondansatördeki yük q olduğu anda 4.5.2 bağıntısını yazabiliriz.

$$V_0 = iR + \frac{q}{C} \tag{4.5.2}$$

Bu bağıntıya göre direnç üzerindeki potansiyel düşüşü ile sığası C olan kondansatörün uçları arasındaki potansiyel farkının toplamı uygulanan voltaja eşittir.

$$i = \frac{dq}{dt}$$

eşitliği 4.5.2 de yerine konulursa

$$V_0 = R\frac{dq}{dt} + \frac{q}{C}$$

bulunur. Bu denklem çözüldüğünde kondansatörün yükünün zamana bağlı denklemi

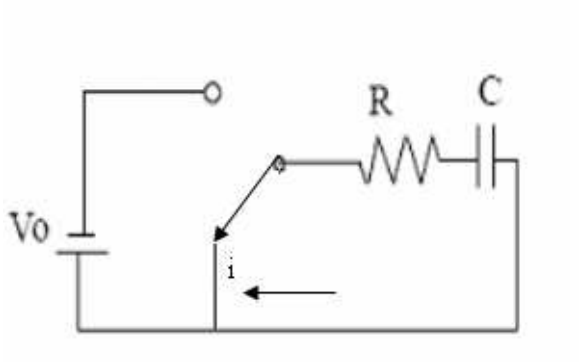
$$q(t) = V_0 C\left(1 - e^{-t/RC}\right) \tag{4.5.3}$$

bulunur. Bu bağıntıdaki $V_0 C$ değeri kondansatördeki maksimum yük değeridir.

Kondansatörün uçları arasındaki voltaj dolum süresince artar, dolum sonunda besleme voltajına ulaşır ve sabit kalır. Bu gerilimi veren bağıntı 4.5.4 olur.

$$V_C(t) = V_0\left(1 - e^{-t/RC}\right) \tag{4.5.4}$$

Devre şekil 4.21'deki duruma getirilirse kondansatörün yükü direnç üzerinden boşalmaya başlar.



Şekil 4.21. Kondansatörün boşalma devresi.

Boşalma sırasında yükün zamana bağlı denklemini bulabilmek için üreteç devreden çıkarıldığından 4.5.2 bağıntısındaki V_0 yerine sıfır yazarız.Eşitliğimiz yeniden yazıldığında,

$$R\frac{dq}{dt} + \frac{q}{C} = 0$$

olur.

Bu denklem çözüldüğünde kondansatörün yükünün zamana bağlı denklemi

$$q(t) = V_0 C e^{-t/RC} \tag{4.5.5}$$

olur.

Kondansatörün uçlarındaki gerilimin zamana göre değişimini veren bağıntı ise 4.5.6 da belirtildiği gibidir.

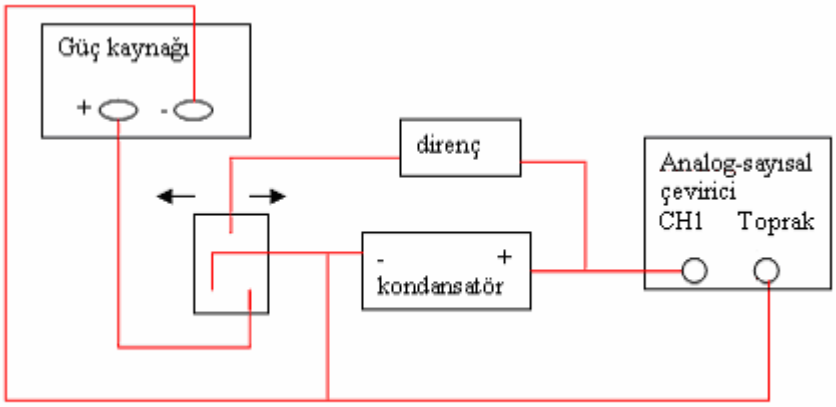
$$V_C(t) = V_0 e^{-t/RC} \tag{4.5.6}$$

Kondansatörün boşalması sırasında uçları arasındaki voltajın ilk değerinin $1/e$=0,378'ine düştüğü zamana zaman sabiti denir. τ sembolü ile gösterilir. Birimi zaman birimidir.

4.5.4. Deneyin Yapılışı

Anahtar, direnç, kondansatör, analog-sayısal çevirici, güç kaynağı kullanarak şekil 4.22'deki düzenek kurulur.

Kondansatörün dolması için kullanılacak besleme voltajı +5V olduğundan analog-sayısal çeviricinin 1. kanalı +5V' a ayarlanmalıdır. Böylece kondansatör +5V' a ulaştığında analog-sayısal çeviriciden okunan değer 255 olur. Ek 13'de verilen RC.C programı çalıştırılır. Anahtar dolma konumuna getirilerek kondansatörün dolması sağlanır. Ekranda 255 sayısı görüldüğünde anahtar boşalma konumuna getirilir. Kondansatör boşalmaya başlar.



Şekil 4.22. Bilinmeyen kondansatörün değerinin bulunması için kullanılan deney düzeneği.

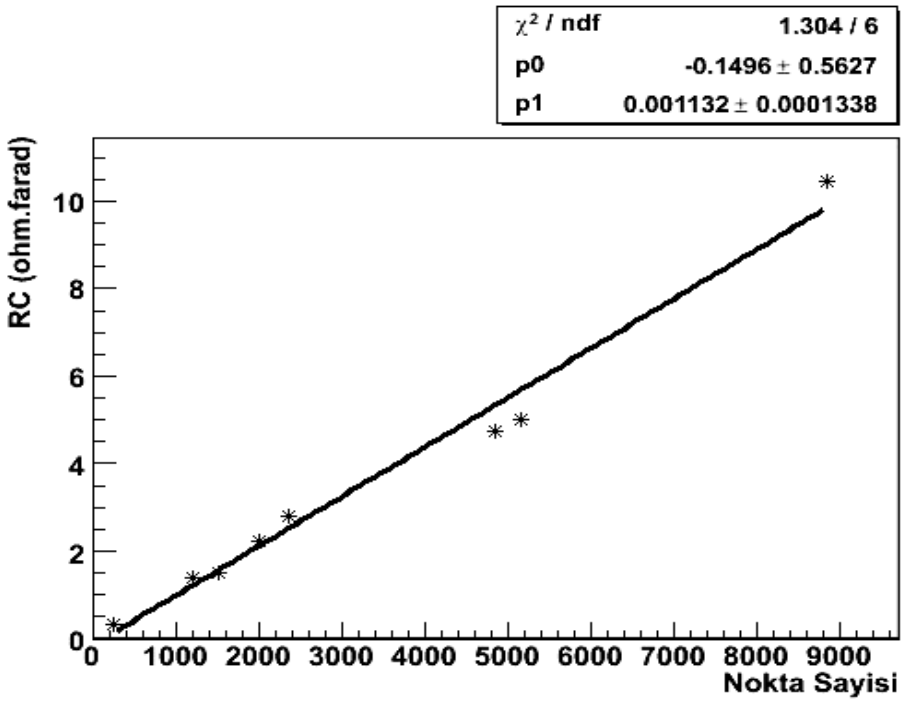
Kullanılacak olan kondansatör ve direnç değerleri ile bulunan *RC* değerleri çizelge 4.7'de verilmiştir. Voltajın 0,378'ine düştüğü ana kadar kaydedilen nokta sayıları yatay eksende, bu nokta sayılarının ait olduğu *RC* çarpımları düşey eksende olacak şekilde grafik çizilir. Bu grafiği çizmek için çizelge 4.7'deki ilk sekiz değer kullanılır. Bu grafiğin eğimi bulunarak *RC* değerleri ile nokta sayıları arasında bir katsayı bulunur. Bilinmeyen *RC* değerine ait nokta sayısını bu katsayı ile çarptığımızda *RC* değerine ulaşabiliriz. Sonucu *R* değerine bölerek kondansatörün sığasını buluruz.

4.5.5. Deneyin Sonucu:

Tablodaki sekiz değeri kullanarak çizilen grafik şekil 4.23'de gösterildiği gibidir. Bu grafiğin eğimi 0,001132'dir. Bu değerin 5097 ile çarpımı 5,76 değerini verir. Bu sonucu kullandığımız *R* değerine böldüğümüzde, 2130 µf değeri bulunur. Kullanılan kondansatörün sığası 2200 µf'dır. Sonuç gerçek değere oldukça yakındır.

Çizelge 4.7. Deneyde kullanılan direnç, kondansatör değerleri ve bulunan nokta sayıları

R (Ω)	*C* (µf)	*RC* (s)	Nokta sayısı
47300	6,8	0,322	252
624	2200	1,373	1202
1500	1000	1,500	1518
4750	470	2,233	2006
1271	2200	2,796	2368
4750	1000	4,750	4842
5000	1000	5	5148
4750	2200	10,450	8835
2700	-	-	5097



Şekil 4.23. RC değerinin nokta sayısına göre değişimi.

5. SONUÇLAR ve ÖNERİLER

Basit sarkaç deneyi yapılırken iki yöntem kullanılmıştır. Birinci yöntemde veri girişi paralel port ile yapılmıştır. Zaman ölçümünde TurboC++ derleyicisinin zaman fonksiyonu kullanılmıştır. Diğerinde ise arabirim kartı üzerindeki quartz osilatörü ile zaman ölçümü yapılmıştır. Sonuçlar incelendiğinde, ikinci yöntemle yapılan ölçümün daha hassas olduğu görülmüştür. Bunun nedeni bilgisayar ara birimi ile birlikte kullanılan kartın içerisinde yer alan quartz kristel ile okunan zamanların hassasiyetinin daha iyi olmasıdır.

Eğik düzlem deneyinde ise arabirim kartı kullanılarak yapılan deney sonucunda sürtünme katsayısı ve ivme değerlerinin teorik sonuçlara çok yakın olduğu görülmüştür.

Eylemsizlik kütlesinin hesaplanması için yapılan deney sonucunda öngörülen sonuçlara ulaşılmıştır. Bu deneyde kronometre kullanarak yapılacak zaman ölçümünden daha hassas bir ölçüm yapıldığı gözlenmiştir. Bilgisayar ile yapılan ölçümler reflekslere dayalı hatalardan kaynaklanan ölçüm hatalarını oldukça azaltmaktadır. Ayrıca grafik çizimi ve eğim hesaplamasında da bilgisayarın çok kolaylık sağladığı gözlenmiştir.

RC devresinde zaman sabitinin bulunduğu deneyde yine verilerin alınması ve grafik çizimi arabirim kartı kullanarak yapılmıştır. Zaman sabitinin doğru olarak bulunduğu rahatlıkla söylenebilir. Zaman sabitini ve bilinen direncin değerini kullanarak hesaplanan sığa değerinin gerçek değerine çok yakın olduğu görülmüştür.

Analog sayısal çevirici kullanarak yapılan girişim deneylerinde de ilgili bölümdeki grafiklerden de görülebileceği gibi beklenen sonuçlara ulaşılmıştır.

Bilgisayarın fizik deneylerinde kullanılması beklenildiği gibi zaman kazandırmış ve ölçümler daha hassasiyetle yapılarak doğru sonuçlara ulaşılmıştır. Bu yöntemin liselerde de kullanılması fizik derslerinin verimini arttıracaktır.

Bu deneyleri yapmak için çok gelişmiş bilgisayarlara ihtiyaç yoktur. Çok hızlı bilgisayarlar kullanılmadan da veri alımı ve analizi mümkündür. Bugün bir doküman hazırlamak için dahi kullanmadığımız bilgisayarlar bile bu işler için kullanılabilir. Bu deneyleri okullarımızda da yapabilmek için evlerde kullanılmamak üzere kaldırılmış

olan bilgisayarların liselerimize bağışlanması yeterli olacaktır. Uygun yazılım ve arabirimleme ile bu bilgisayarlar birer ölçü aleti olarak kullanılabilir.

KAYNAKLAR

AYVACI, H.Ş, ÖZSEVGEÇ, T., ve AYDIN, M., 2004. Data Loger Cihazının Ohm Kanunu Üzerindeki Pilot Uygulaması. The Turkish Online Journal of Educational Technology, 3(3)

BOLAT, M., 1997. İlkelerle Fizik Testleri. Feryal Matbaacılık, Ankara, 312s.

BORKOWSKI, J. D., 1991. Computer Models In Teaching Physics.Doğa-TR. J. of Physics, 15: 165-180.

CAMPBELL, J. A., 1991. Symbolic computing For Computer-Aided Education In Physics. Doğa-TR. J. of Physics, 15: 181-195.

CORTINI, G., 1992. The Use of A Computer As A Laboratory Instrument In Teaching Experimental Physics. Physics Education, 27: 159-162.

ÇAKMAK, O., 1999. Fen Eğitiminin Yeni Boyutu: Bilgisayar-Multimedya-İnternet Destekli Eğitim. D.E.Ü Buca Eğitim Fakültesi Dergisi Özel Sayı, 11: 116-125.

ÇORLU, M. A., ALTIN, K., 1999. Bilgisayar Destekli Deney Ortamı ve Uygulamaları: Radyoaktif Bozunma Deney Verilerinin Bilgisayarda Toplanması ve Değerlendirilmesi. D.E.Ü Buca Eğitim Fakültesi Dergisi Özel Sayı, 10: 242-249.

DEPIREUX, J., 1991. Simple Interfacing and Use of A Computer As A Laboratory Instrument. Doğa-TR. J. of Physics, 15: 196-202.

DİNÇER, G., GÜRKAN, S., 1999. PC Tabanlı Kontrol ve Otomasyon, Bileşim Yayıncılık A.Ş, İstanbul,112s.

ERSOY, Y., 1991. The Computer Asisted Laerning/Teaching Project. Doğa-TR. J. of Physics, 15: 203-211.

FINDLAY, D., 1993. Microprocessor Projects In A Physics Laboratory. European Journal Of Physics, 6: 65-71.

HACINLIYAN, A., TEPEHAN, G., 1991. Language Problem In Secondary School Instruction Involving Computers. Doğa-TR. J. of Physics, 15: 217-224.

JACOBSEN, E., 1991. Mikrobilgisayarların Fen Derslerinde Kullanılmasına Dünyadaki Yaklaşımlar. Doğa-TR. J. of Physics, 15: 152-164.

HASNAIN, A.F., 1991. Computer Simulation and Modelling In Teaching Physics. Doğa-TR. J. of Physics, 15: 212-216.

KÜNHELT, H., 1991. Computers In The Physics Classroom Beyond The Analytic Solution. Doğa-TR. J. of Physics, 15: 228-235.

LINCKE, R., 1991. An Interfacing and Programming Course In The Introductory Physics Laboratory of Kiel University. Doğa-TR. J. of Physics, 15: 236-240.

_____, 1991. Physics Projects with Microcomputers. Doğa-TR. J. of Physics, 15: 241-253.

MACLEOD, A. M., 1984. A Microprocessor Laboratory In A Physics Department. European Journal Of Physics, 5: 1-5.

OSBORNE, J., 1991. Issues In Software Design and Evaluation. Doğa-TR. J. of Physics, 15: 279-289.

STAUDENMAİER, H. M., 1991, Use of Computers In Physics Education. Doğa-TR. J. of Physics, 15: 296-315.

ŞAHAN, B. Y., 1999, Fizik Laboratuar Deneyleri. Sürat Yayınları, İstanbul, 194s.

THORNTON, R. J., 1991.Using The Microcomputer-Based Laboratory To Improve Student Conceptual Understanding In Physics. Doğa-TR. J. of Physics, 15: 316-335.

TUĞAY, G., 2004. Elektronik Hobi. Melisa Matbaacılık, İstanbul, 246s.

ÖZGEÇMİŞ

1973 yılında Şanlıurfa'da doğdum. İlk, orta ve lise öğrenimimi İzmit'te yaptıktan sonra 1989'da Marmara Üniversitesi Atatürk Eğitim Fakültesi Fizik Öğretmenliği'ni kazandım.1994 yılında Osmaniye Anadolu Lisesi'nde fizik öğretmenliğine başladım. 1998 yılında Adana'ya Seyhan Rotary Anadolu Lisesi'ne geldim, halen bu okulda görev yapmaktayım.

EKLER

EK 1: PORT_1.C

```
//Paralel portun data çıkışına veri gönderip oradaki değerleri okur.
#include <stdio.h>
#include <dos.h>
#include <conio.h>
void oku(void);  //oku fonksiyonunu tanımlar
void yaz(void);
const port=0x378; //portun adresi
unsigned char gonder,al; //8 bitlik tamsayı değişken tanımlar
char c;//karakter değişken tanımlar
main()
{
  do {
     clrscr();
     printf("1 - OKU\n");
     printf("2 - YAZ\n");
     printf("3 - CIKIS\n");
     c = getch();
     switch(c) {
           case '1' : oku(); break;
           case '2' : yaz(); break;
           case '3' : exit(0);
     }
  } while (1);
}
void yaz(void)
{
  clrscr();
  printf("Gonderilecek veri :");
  scanf("%d", &gonder);
  outport(port, gonder);
  c = getch();
}
void oku(void)
{
  clrscr();
  do {
     al = inportb(port); //asc den değer okur ve a1 değişkenine atar.
     printf("Okunan Veri  %d\n", al);
     c = getch();
  } while (c != 'q'); //klavyeden q harfi girilene kadar işleme devam eder.
}
```

EK 2: PRI_KONT.C

```
//Status portundan topraklanan pinlerin ürettiği sayısal değerleri okuyan program.
#include <stdio.h>
#include <dos.h>
#include <conio.h>
const port=0x379;
main()
{
  unsigned char al;
  clrscr();
  do{
     al=inportb(port);
      printf("okunan veri %d\n",al);
   }while(al!=63);
}
```

EK 3: ASC_1.C

```
// ASC portunun birinci kanalındaki 8 bitlik sayısal değerleri okur.
#include <dos.h>
#include <stdio.h>
const asc1=0x104;
main()
{
  unsigned char y;
  do{
     y=inport(asc1); /*asc den okuma*/
     printf("%d\n",y); /*değeri ekrana yazma*/
    delay(50); /*50 ms bekleme*/
  }while(1);
}
```

EK4: ASC_2.C

```
// ASC portunun 1 kanalından okunan 8 bitlik sayısal değerlerin grafiğini çizer.
#include <graphics.h>
#include <dos.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <conio.h>
const asc1=0x104;
main() {
  unsigned char y;
  int gd, gm,x, hata;
  gd=DETECT;
  initgraph(&gd, &gm, "c:\\tc\\bgi\\"); //grafik moduna geçiş
  hata=graphresult();
  if (hata !=grOk)
  {
       printf("grafik hatasi: %s\n", grapherrormsg(hata));
       getch();
       exit(1);
  }
  line(0, 0,  0, 500);
  line(0,0, 679,0);
  line(0,479,679,479);
  line(635,0,635,479);
  for(x=0; x<679; x=x+1)
 {
    y=inport(asc1);
    putpixel(x, 240-y, 15);
    delay(30);
  }
  getch();
  closegraph();
}
```

EK 5: ASC_3.C

```
//ASC portunun birinci kanalından çok kısa zaman aralığında gerçekleşen fiziksel bir
//olaydan alınan 8 bitlik veriler önce bir diziye kaydedilip daha sonra grafiği
//çiziliyor.
#include <graphics.h>
#include <dos.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <conio.h>
const asc1=0x104;
main(){
  unsigned char a[450];
  int gd, gm,x, hata;
  gd=DETECT;
  initgraph(&gd, &gm, "c:\\tc\\bgi\\"); //grafik moduna geçiş
  hata=graphresult();
  if (hata !=grOk){
     printf("grafik hatasi: %s\n", grapherrormsg(hata));
     getch();
     exit(1);
  }
  line(0, 0,  0, 500);
  line(0,0, 679,0);
  line(0,479,679,479);
  line(635,0,635,479);
  for(x=0; x<400; x=x+1){
     a[x]=inport(asc1);
     delay(0);
  }
  for(x=0; x<400; x=x+1){
    putpixel(x, 240-a[x], 15);
    delay(0);
  }
  getch();
  closegraph();
}
```

EK 6:P_OKU_YAZ.C

```
// Bütün portları çıkış yapar.
#include <dos.h>
#include <stdio.h>
void main
{
     int TemelAdres;
     int PortA, PortB, Port C, KontrolKelimeAdresi;
     TemelAdres=608: /*4.atlama pozisyonu 608'e karşılık gelir.*/
      PortA=TemelAdres ;
      PortB=TemelAdres+1 ;
      KontrolKelimeAdresi = TemelAdres+3 ;
      outportb(KontrolKelimeAdresi, 128) ;  /*bütün portları çıkış yapar.*/
}
```

EK 7 : PORT_OKU.C

```
//Porttan okuma yapar.
#include <dos.h>
void main{
  int PortB ;
  outportb(KontrolKelimeAdresi,155) ; /*Bütün portlar Mode 0 giriş olarak
ayarlandı*/
  PortB_Veri=inportb(PortB) ;
  printf(˝%d\n˝,PortB_Veri) ;
}
```

EK 8:SARKAC_PRINTER.C

```
//Paralel port ile basit sarkacın periyodunu ölçer.
#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
const port=0x379; //paralel port status adresi
unsigned char  al;
int oku(int);
main() {
      clock_t start, end,start2,end2,start3,end3;
      float to,A,T,to_ussu;
      double z,d,l;
      printf("d değini yazınız");
      scanf("%lf",&d);
      printf("l değerini yazınız");
      scanf("%lf",&l);
      do{
            oku(127);
            start = clock();
            oku(63);
            end = clock();   // saat duruyor
            oku(127);
            start2 = clock();  // saat tekrar çalışmaya başlıyor
            oku(63);
            end2 = clock();   // saat duruyor
            to=(end2 - start2) / CLK_TCK;
            to_ussu=(end - start) / CLK_TCK;
            T=to+to_ussu; // periyod
            printf("T= %f\n",T);
            printf("to= %f\n",to);
            printf("to'= %f\n",to_ussu);
             z=d/l;
             A=atan(z)*57.2/(cos(3.14*to/T));
            printf("Genlik= %lf\n ",fabs(A));
      }while(1);
       getch();
       return 0;
 }
 int oku(int x) {
   do{
            al=inportb(port);
    }while(x!=al);
}
```

EK 9: SARKAC_PortA.C

```
// Çok fonksiyonlu kartı kullanarak basit sarkacın periyodunu ve genliğini bulma.
#include<stdio.h>
#include<dos.h>
#include<conio.h>
#include <math.h>
const card=0x100;
const mode=0x98;
main(){
        int porta,portc,dirreg,n;
        int counter0,counter1,counterreg;
        unsigned char a,b;
        float zaman1[100], zaman2,zaman3[100];
        float A,T,to,to_ussu;
        float z,d,l;
        printf("d değerini yazınız");
        scanf("%f",&d);
        printf("l değerini yazınız");
        scanf("%f",&l);
        porta = card;
        portc = card+2;
        dirreg =card+3;
        counter0=card+16;
        counter1=card+17;
        counterreg=card+19;
        outport(dirreg,mode);
        clrscr();
        outport(portc,4),
        outport(counterreg,0+48+6+0);
        outport(counter0,208);
        outport(counter0,7);
        outport(counterreg,64+48+0+0);
        outport(counter1,255);
        outport(counter1,255);
        while(inport(porta)==0){}
        n=0;
        do{
            outport(portc,0);
            while(inport(porta)==1)  {} // sarkaç optik kapının önünden geçiyor
            printf("%d\n",inport(porta));
            while(inport(porta)==0)  {} // sarkaç optik kapıdan çıktı
            printf("%d\n",inport(porta));
            outport(portc,4);
            a=inport(counter1);
            b=inport(counter1);
            zaman1[n]=(65535.0-b*256.0-a)/1000;
```

```
  zaman3[n]=zaman1[n]-zaman2;
  printf("n=%d  to or  to'=%8.2f\n",n,zaman3[n]);
  zaman2=zaman1[n];
  if(n>=6){
      to=zaman3[n-1];
      to_ussu=zaman3[n];
      T=to+to_ussu;
       z=d/l;
      A=atan(z)*57.2/(cos(3.14*to/T));
      printf("%lf\n" , T);
      printf("%lf\n",fabs(A));
  }
  n++;
}while(1);
```

EK 10: EGIK_DUZLEM.C

```
// Eğik düzlemde kayan cismin ivmesini ve cisimle yüzey arasındaki sürtünme
//katsayısını bulur.
#include<stdio.h>
#include<conio.h>
#include<dos.h>
#include <math.h>
void egik_kont();
const card=0x100;
const mode=0x98;
int porta,portc,dirreg,n;
int counter0,counter1,counterreg;
float g;
unsigned char a[6],b[6];
float t[6],tm[6],v[6],acc[6],ivme;
int check[6]={125,123,119,111,95,63};
float yol[5]={0.1,0.15,0.2,0.2,0.25};
double aci,k;
main() {
      porta=card;
      portc=card+2;
      dirreg=card+3;
      counter0=card+16;
      counter1=card+17;
      counterreg=card+19;
      outportb(dirreg,mode);
      outport(portc,4);
      outport(counterreg,0+48+6+0);
      outport(counter0,200);
      outport(counter0,0);
      outport(counterreg,64+48+0+0);
      outport(counter1,255);
      outport(counter1,255);
     clrscr();
      for(n=0;n<7;n++){
            a[n]=0;
            b[n]=0;
            tm[n]=0;
            t[n]=0;
            v[n]=0;
            acc[n]=0;
      }

      do{
          a[0]=inport(porta);
          printf("%d\n",a[0]);
```

```
    }while(!kbhit());
    while(inport(porta)!=127){}
    outport(portc,0);
    a[0]=inport(counter1);
    b[0]=inport(counter1);
    for(n=0;n<=5;n++) {
        while(inport(porta)!=check[n]){}
        a[n]=inport(counter1);
        b[n]=inport(counter1);
    }
    outport(portc,4);
    for(n=0;n<=5;n++) {
        t[n]=(65535.0-b[n]*256.0-a[n])/10000.;
        printf("%6.3f*** \n",t[n]);
    }
    for(n=0;n<5;n++) {
        tm[n]=(t[n+1]+t[n])/2.;
        printf("%5.2f s\n",tm[n]);
    }
    for(n=0;n<5;n++) {
        v[n]=yol[n]/(t[n+1]-t[n]);
        printf("%5.2f m/s \n",v[n]);
    }
    g=0;
    for(n=0;n<4;n++)  {
        acc[n]=(v[n+1]-v[n])/(tm[n+1]-tm[n]);
        printf("%5.2f m/s^2 \n",acc[n]);
        g=g+acc[n];
    }
    printf("a=%5.2f m/s^2\n",g/4);
    k=(9.81*sin(35*3.14/180)-(g/4) )/9.81*cos(35*3.14/180);
    printf("k=%f",k);
    getch();
}
```

EK 11: KUTLE.C

```
// Eylemsizlik kütlesini hesaplar.
#include<stdio.h>
#include <graphics.h>
#include<dos.h>
#include<conio.h>
const card=0x100;
const mode=0x98;
void  main()
{
  int porta,portc,dirreg,i,x;
  int counter0,counter1,counterreg;
  unsigned char a,b;
  float zaman1[200], zaman2,zaman3,per,per2[6],per3,egim,bkutle;
  float m[]={146.,291.,437.,584.,729.};
  int gd, gm,y;
  porta = card;
  portc = card+2;
  dirreg =card+3;
  counter0=card+16;
  counter1=card+17;
  counterreg=card+19;
  outport(dirreg,mode);
  clrscr();
  outport(portc,4),
  outport(counterreg,0+48+6+0);
  outport(counter0,208);
  outport(counter0,7);
  outport(counterreg,64+48+0+0);
  outport(counter1,255);
  outport(counter1,255);
  while(inport(porta)==0){}
  x=0;
  do{
       i=0;
       do{
         outport(portc,0);
         while(inport(porta)==1)  {}
         while(inport(porta)==0)  {}
         outport(portc,4);
         a=inport(counter1);
         b=inport(counter1);
         zaman1[i]=(65535.0-b*256.0-a)/1000;
         printf("zaman:%8.2f\n",zaman1[i]);
         i++;
       }while(i!=21);
```

```
        zaman2=zaman1[20]-zaman1[1];
        per=zaman2/20.;
        printf("fark= %f   periyod=%f  periyod2=%f\n" , zaman2,per,per*per);
        getch();
        per2[x]=per*per;
        x++;
   }while(x!=6);
   for(y=0;y<6;y++){
      printf("%f\n",per2[y]);
   }
   getch();
}
```

EK 12: OPTIC.C

```
//Tek ve çift yarıkta girişim deneyinde fototransistörün ve direncin uçlarındaki
//voltajları asc'den okuyup birbirlerine karşı grafiklerini çizer.
#include <graphics.h>
#include <dos.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <conio.h>
const asc1=0x104;
const asc2=0x108;
main()
{
    unsigned char a,b;
    int gd, gm,x, hata;

    gd=DETECT;
    initgraph(&gd, &gm, "c:\\tc\\bgi\\"); //grafik moduna geçiş
    hata=graphresult();
    if (hata !=grOk)
    {
        printf("grafik hatasi: %s\n", grapherrormsg(hata));
        getch();
        exit(1);
    }
    line(0, 0,  0, 500);
    line(0,0, 679,0);
    line(0,479,679,479);
    line(635,0,635,479);
     x=0;
    do{
        a=inport(asc1); // asc nin 1. kanalından veri okunuyor
        b=inport(asc2); // asc nin 2. kanalından veri okunuyor
        putpixel(b*2, 400-a, 3); //
        delay(20); // 20 mili saniye bekliyor
         x++;
     }while(!kbhit()); // bir tuşa basana kadar döngü çalışıyor
}
```

EK 13: RC.C

```
// RC devresinde kondansatör boşalırken  her kondansatörün 1 mili saniye aralıkla bir
//sinyal gönderip buna göre kondansatörün üzerindeki voltajın ilk değerinin 0.378
//katına düşene kadar bu sinyalin kaç defa alındığını bulan program.
#include <dos.h>
#include <stdio.h>
#include <graphics.h>
#include <conio.h>
const asc1=0x104;
void tetik();
void doldumu();
main()
{
   unsigned char y,z;
   int i,k,m,n[10];
   int a,b;
   float R[10],C[10],egim,r,rc,c;
   for(a=0; a<9; a++){
      printf("%d. R degerini girin\n",a);
      scanf("%f",&R[a]);
      printf("%d. C degerini girin\n",a);
      scanf("%f",&C[a]);
   }
   printf("bilinen R degerini girin\n");
   scanf("%f",&r);
   printf("R ve C degerlerini girdiniz\n\n LÜTFEN İLK GİRİLEN R ve C degerini
DENEY DUZENEG˜NE BAGLAYINIZ\n");
   printf("Ve Bir Tusa Basiniz\n");
   getch();
   for(k=0; k<10; k++){
     doldumu(); // kondansatörün dolması bekleniyor
     tetik(); // tetik şartı bekleniyor
     i=0;
      do{
         y=inport(asc1);
         delay(1);
         i=i+1;
      }while(y>96); // asc den alınan sayısal değer 96 olana kadar döngü çalışıyor
      printf("nokta sayisi= %d\n\n",i);
      n[k]=i;
      printf("Diger R ve C yi Deney Duzenegine Baglayiniz Dolmasini Bekleyiniz\n
Ve Bir Tusa Basiniz !!!\n");
  getch();
 }
 for(m=0;m<10;m++) {
    printf("%d\n",n[m]);
```

```
  }
   getch();
}
////////////////////////////Tetiklememe fonksiyonu/////////////////////////////////////////
void tetik()
{
  unsigned char y;
  y=inport(asc1);
  do{
       y=inport(asc1);
       delay(1);
       printf("tetik sarti bekleniyor %d\n",y);
    } while(y>250);
}
////////////////////////////// Kondansatörün dolduğunu anlama ////////////////////////////////
void doldumu()
{
  unsigned char y;
  y=inport(asc1);
  printf("kondansator doluyor \n");
  do{
       y=inport(asc1);
       delay(1);
  } while(y<254);
  printf("kondansator doldu \n");
}
```