T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR KONTROLLÜ SERBEST DÜŞME HAREKETİ DENEY SETİ TASARIMI ve YAPIMI

YÜKSEK LİSANS TEZİ

Elektronik Öğretmeni Devrim AKGÜN

Enstitü Anabilim Dalı : Elektronik ve Bilgisayar Eğitimi

Tez Danışmanı: Yrd. Doç. Dr. İlyas ÇANKAYA

HAZİRAN 2003

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR KONTROLLÜ SERBEST DÜŞME HAREKETİ DENEY SETİ TASARIMI ve YAPIMI

YÜKSEK LİSANS TEZİ

Elektronik Öğretmeni Devrim AKGÜN

Enstitü Anabilim Dalı : Elektronik ve Bilgisayar Eğitimi

Tez Danışmanı: Yrd. Doç. Dr. İlyas ÇANKAYA

Bu tez 17/07/2003 tarihinde aşağıdaki jüri tarafından Oybirliği/~~Çoğunluğu~~ ile kabul edilmiştir.

Yrd. Doç. Dr. İlyas ÇANKAYA	Doç. Dr. İbrahim OKUR	Yrd. Doç. Dr. Ali Fuat BOZ
Jüri Başkanı	Jüri Üyesi	Jüri Üyesi

TEŞEKKÜR

"Bilgisayar Kontrollü Serbest Düşme Deney Seti Tasarımı ve Yapımı" adlı Yüksek Lisans tezimin hazırlanmasında ilgi ve yardımlarını esirgemeyen hocam Yrd. Doç. Dr. İlyas ÇANKAYA ve Arş.Gör. Fahri VATANSEVER'e saygı ve şükranlarımı sunarım.

HAZİRAN 2003 Devrim AKGÜN

İÇİNDEKİLER

ŞEKİLLER LİSTESİ vi
TABLOLAR LİSTESİ ix
ÖZET x
SUMMARY xi

BÖLÜM 1
GİRİŞ 1
1.1.Giriş 1
1.2. Serbest Bırakılan Cismin Yükseklik-Zaman değişimi 3
1.3. Serbest Bırakılan Cismin Hız-Zaman değişimi 4
1.4. Kullanıcı Ara Yüzünün Tasarımı 4
1.4.1. Delphi görsel programlama dili 5
1.5. Deney Sisteminin Eğitimde Kullanımı 6

BÖLÜM 2
ATIŞ SİSTEMİNİN BİLEŞENLERİ 7
2.1 Giriş 7
2.2 Kullanıcı Arayüzü 8
2.3. Paralel Portun Yapısı ve Sistemde Kullanımı 9
2.4. Adım Motorun Yapısı 10
2.4.1. Sürücü devre dizaynı 12
2.5. Elektromıknatısın Yapısı ve Sistemde Kullanımı 14
2.6. Algılayıcı Sistemin Yapısı ve ve Sistemde Kullanımı 16
2.7. Sonuçlar 19

BÖLÜM 3
KULLANICI ARA YÜZÜNÜN DELPHİ İLE TASARIMI 20
3.1 Giriş 21

3.2. Ana Pencere 22
3.3. Yükseklik Listesi 23
3.3.1. Manuel liste 23
3.3.2. Otomatik liste 27
3.3.3. Listenin kapatılması 29
3.4. Atışların Gerçekleştirilmesi 31
3.4.1. Zamanlama işlemi 31
3.4.2. Adım motorunun kontrolü 35
3.4.3. Yükseklik seviyesinin belirlenmesi 40
3.4.4. Düşme süresinin ölçülmesi 44
3.4.5. Başlangıç konumuna dönüş 47
3.5. Sonuçların Kaydedilmesi 48
3.5.1. Değişken ve tip tanımlamaları 48
3.5.2. Kayıt işlemleri 49
3.6. Atışların Grafiğinin Çizdirilmesi 55
3.7. Sonuçlar 57

BÖLÜM 4

KULLANICI ARA YÜZÜNÜN TANITIMI 58
4.1 Giriş 58
4.2. Ana Pencere 59
4.3. Yükseklik Girişi Pencereleri 59
4.3.1. Otomatik yükseklik girişi 60
4.3.2. Manuel yükseklik girişi 61
4.4. Atış Penceresi 62
4.5. Sonuç Pencereleri 65
4.5.1. Kayıt listesi 66
4.5.2. Grafik 66
4.6. Ayar Pencereleri 72
4.6.1. Yazıcı ayarları 72
4.6.2. Adım motorunun adım hızı ayarı 73
4.6.3 Yükseklik ve adım değeri penceresi 74
4.7. Yardım Penceresi 74

4.8. Sonuçlar..... 76

BÖLÜM 5

SONUÇLAR..... 77

TARTIŞMA VE ÖNERİLER..... 78

KAYNAKLAR..... 79

ÖZGEÇMİŞ 80

ŞEKİLLER LİSTESİ

Şekil 1.1 Kütlesel Kuvvetin Hızlanması....2
Şekil 1.2 Büyük Kütlenin Yanındaki Küçük Kütlenin Hızlanması....2
Şekil 1.3 Yükseklik-Düşme Zamanı Grafiği....3
Şekil 1.4 Hız-Zaman Grafiği....4
Şekil 1.5 Delphi Ekranı....5
Şekil 2.1 Serbest Düşme Deney Setinin Prensip Şeması....7
Şekil 2.2 Paralel Port ve Kontrolündeki Elemanlar....9
Şekil 2.3 Adım Motorunun Yapısı ve Çalışma Prensibi....10
Şekil 2.4 Adım Motorunun Tek ve Çift Kutuplu Modda Çalıştırılması....11
Şekil 2.5 Adım Motorunun Bir Sargısının Sürülmesi....14
Şekil 2.6 Elektromıknatıs ve Port Sürücü Devresi....15
Şekil 2.7 Işık Yayan ve Algılayan Diyotların Yapısı....16
Şekil 2.8 Bilyenin Düşme Anında Algılanması....17
Şekil 2.9 Tek Atımlı Anahtarın Giriş-Çıkış Sinyalleri....18
Şekil 2.10 Algılama Devresi....18
Şekil 3.1. Ana Pencere Seçenekleri....21
Şekil 3.2. 'Mainmenu' Nesnesi....22
Şekil 3.3 Ana Pencere Seçenekleri....22
Şekil 3.4 Ana Pencere Seçeneklerine Ait Alt Programlar....23
Şekil 3.5 Manuel Liste....24
Şekil 3.6 'Onay' Butonuyla Yürütülen Alt Program....25
Şekil 3.7 'Sıfırla' Butonuyla Yürütülen Alt Program....26
Şekil 3.8 Otomatik Liste Penceresi....27
Şekil 3.9 Liste Oluştur Butonuyla Yürütülen Alt Program....28
Şekil 3.10 Pencerenin Kapatılmasıyla Birlikte Yürütülen Algoritma....29
Şekil 3.11 Liste Penceresinin Kapatılmasıyla Yürütülen Alt Program....30
Şekil 3.12 Zamanlayıcı Kontrolünde Yürütülen Algoritma....33
Şekil 3.13 Zamanlayıcı Kontrolünde Yürütülen Alt Program....34

Şekil 3.14 Çift Kutuplu Adım Motorunun İleri ve Geri Döndürülmesi..... 35
Şekil 3.15 Paralel Portla Adım Motorunun Döndürülmesi.....36
Şekil 3.16 Adım Motorunun Döndürülmesinde Kullanılan Algoritma..... 37
Şekil 3.17 Adım Motorunu Kontrol Eden Alt Programlar..... 38
Şekil 3.18 Yükseklik Seviyesi Belirleme Alt programı.....41
Şekil 3.19 Hız Kontrolü Yapan Alt Programı..... 43
Şekil 3.20 Bilyenin Düşme Yüksekliği..... 44
Şekil 3.21 Bilyenin Bırakılmasıyla Birlikte Yürütülen Algoritma..... 45
Şekil 3.22 Düşme Zamanını Ölçen Alt Program..... 46
Şekil 3.23 Bilyenin Alınmasında Kullanılan Alt Program..... 47
Şekil 3.24 Kayıt İşlemleri İçin Dosya Atanması..... 48
Şekil 3.25 Atış Penceresi Kapatılırken Yürütülen Algoritma..... 51
Şekil 3.26 Atış Penceresi Kapatılırken Yürütülen Alt Program..... 52
Şekil 3.27 Kayıt İsimlerini Listeye Yazan Program Kodları.....53
Şekil 3.28 Yükseklik ve Düşme Zamanı Değerlerinin Listelenmesi..... 54
Şekil 3.29 Grafik Formu..... 55
Şekil 4.1 Serbest düşme deney setinin görünümü..... 58
Şekil 4.2 Ana Pencere.....58
Şekil 4.3 Liste Alt Seçenekleri..... 59
Şekil 4.4 Otomatik Liste Penceresi..... 60
Şekil 4.5 Liste Onay Mesajı.....60
Şekil 4.6 Manuel Liste Oluşturma..... 61
Şekil 4.7 Liste Sıfırlama İşlemi Uyarı Mesajı..... 61
Şekil 4.8 Atış Penceresi Liste Uyarısı.....62
Şekil 4.9 Atış Penceresi Başlama Konumu.....63
Şekil 4.10 Atışların Gerçekleştirilmesi.....63
Şekil 4.11 Atışın Gerçekleşmemesi Durumu..... 64
Şekil 4.12 Atışlar Bitmedi İse Pencerenin Kapatılması.....64
Şekil 4.13 Atışların Tamamlanması.....65
Şekil 4.14 Sonuçlar Seçeneği.....65
Şekil 4.15 Kayıt Listeleme Penceresi..... 66
Şekil 4.16 Grafik Penceresi..... 67
Şekil 4.17 Grafik Çizimi İçin Kayıt Listesi..... 67

Şekil 4.18 Yükseklik-Zaman Grafiği..... 68
Şekil 4.19 Yükseklik-Zamanın Karesi Grafiği..... 69
Şekil 4.20 Yükseklik-Zaman Grafiği ve İdeal Grafik Karşılaştırması..... 70
Şekil 4.21 Sütun Şeklinde Üç Boyutlu Grafik..... 71
Şekil 4.22 Grafiğin Resim Olarak Kaydedilmesi..... 71
Şekil 4.23 Yazdırma Butonu..... 72
Şekil 4.24 Ayarlar Seçeneği..... 72
Şekil 4.25 Yazıcı Ayarları Penceresi..... 73
Şekil 4.26 Adım Hızı Ayarı Penceresi..... 73
Şekil 4.27 Yükseklik ve Adım Sayısı Referanslarının Girildiği Pencere..... 74
Şekil 4.28 Yardım Seçeneği..... 74
Şekil 4.29 Yardım Penceresi..... 75

TABLOLAR LİSTESİ

Tablo 2.1 Adım Motorunun Tam ve Yarım Adım Döndürülmesi.....11
Tablo 3.1 116-119 Adımları arası Adım Motoruna Gönderilen Sinyaller 39

ÖZET

Anahtar Kelimeler: Delphi, Adım Motoru, Paralel Port, Optik Algılama

Bu çalışmada, serbest düşme deneyinin gerçekleştirilmesinde kullanılan bir bilgisayar kontrollü sistemin yapısı yazılım ve donanım olarak gerçekleştirilmiştir. Deney sistemi, DELPHI görsel programlama dili ile yazılmış kullanıcı ara yüzü ve bunun kontrolünde çalışan atış düzeneği içermektedir. Serbest düşme deneyinin gerçekleştirildiği atış düzeneğinin yapısı atışlarda kullanılan bilyeyi taşıyan elektromıknatıs, yükseklik seviyesini ayarlayan adım motoru ve bilyeyi düşme noktasında algılayan optik sistemden oluşmuştur. Kullanıcı ara yüzünden girilen atış yüksekliklerine göre, atış düzeneği kontrol edilerek, her bir yükseklik için tekrar sayısınca atışlar gerçekleştirilerek sonuçlar kaydedilebilmektedir. Aynı zamanda elde edilen sonuçlar listelenebilmekte veya grafiksel olarak sunulabilmektedir.

Desing and Implementation of Computer Controlled Experiment System for Free Falling Motion

SUMMARY

Key Words: Delphi, Stepper Motor, Parallel Port, Optic Sensing.

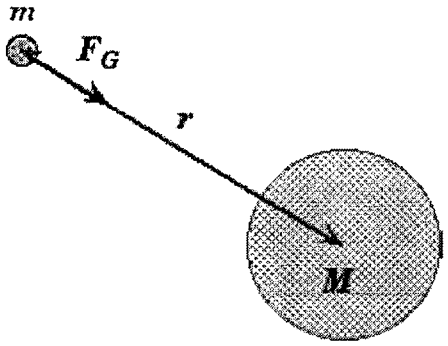
In this study, a computer controlled system which is used in the realization of free falling motion have been implemented as hardware and software. Experimental system includes a user interface that was written using DELPHI and a free fall mechanism works under the control of the user interface. The structure of the free fall mechanism consists of an elctromagnet that carries the ball, a stepper that sets the level of height, and an optic system that senses the ball at finishing point. According to the fall height list that is entered via the user interface, for each height, the falls are repeated for the number of times indicated in the list and the results can be recorded. These results can be listed or showed graphically.

BÖLÜM 1. GİRİŞ

Çoğu bilim adamı açısından modern bilimin babası olarak düşünülen Galileo (1564-1626), serbest düşme üzerine deneyler gerçekleştiren ilk bilim adamıdır. Deneylerin kontrollü bir şekilde gerçekleştirilmesinin önemini vurgulamış ve çeşitli ağırlıklarda olan farklı kütlelerin düşme zamanlarını ölçmek için bir çok deney tasarlanmıştır. Bu deneylerden yola çıkarak, bir nesnenin kütlesinin düşme şeklini etkilemediğinin farkına varmıştır.

Sabit ivmeli doğrusal hareketin örneklerinden biri dünyanın yüzeyine doğru düşmekte olan bir cismin hareketidir. Hava sürtünmesi olmadığı veya ihmal edildiği durumlarda, ağırlıkları, yapıları ve şekilleri ne olursa olsun her cismin dünya yüzeyine doğru aynı ivme ile düştüğü bilinmektedir. Cismin düştüğü yüksekliğin çok büyük olmadığı durumlarda hareket esnasında ivme değişmez. Hava sürtünmesinin olmadığı ve yerden yüksekliğe göre ivmedeki değişimlerin ihmal edilip hesaba katılmadığı harekete serbest düşme hareketi denir. Serbest düşme hareketi yapan bir cisim yalnızca yer çekimi etkisinde düşen bir cisimdir. Bu tanımlama, serbest düşme hareketi yapan cisim hakkında iki önemli noktayı ortaya çıkarır:

- Serbest düşen cisim hava direnciyle karşılaşmaz.
- Yeryüzünde bütün serbest düşen cisimler aşağı doğru yaklaşık olarak $10\,ms^2$ (tam olarak $9.8\,ms^2$) ile hızlanır [1].



$$F_G = G\frac{mM}{r^2} \qquad (1.1)$$

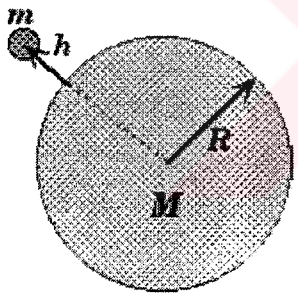
$$F_{net} = F_G$$

$$ma = G\frac{mM}{r^2}$$

$$a = G\frac{M}{r^2} \qquad (1.2)$$

Şekil 1.1. Kütlesel kuvvetin hızlanması

Büyük küresel bir M cisminden, herhangi bir 'r' uzaklığında bulunan daha küçük bir m cismi, Şekil 1.1'de görüldüğü gibi, kütlesel bir kuvvete maruz kalır. Küçük cisme Newton'un ikinci kuralını (F = m.a) uyguladığımızda, eğer etki eden başka kuvvetler yoksa, cismin ivmesi bulunabilir.



$$F_G = G\frac{m_1 m_2}{r^2} = G\frac{mM}{(h+R)^2}$$

$$h << R \Rightarrow F_G = G\frac{mM}{R^2} \qquad (1.3)$$

Şekil 1.2. Büyük kütlenin yanındaki küçük kütlenin hızlanması

M kütleli R yarı çapına sahip büyük kütleli ve yoğunluğu her noktada eşit olan bir cisimden 'h' kadar uzaklıktaki m kütleli cisme uygulanan kütlesel kuvvet denklem 2.3.'teki gibi ifade edilebilir. Bu denklemde küçük kütleli cismin h uzaklığı büyük kütleli cismin yarı çapından oldukça küçük olduğu için ihmal edilmiştir. Eğer hava direnci ihmal edilirse, m kütleli cismin ivmesi denklem 1.4'teki gibi ifade edilebilir.

Öneminden dolayı yer çekiminden kaynaklanan ivme farklı bir sembolle gösterilir;

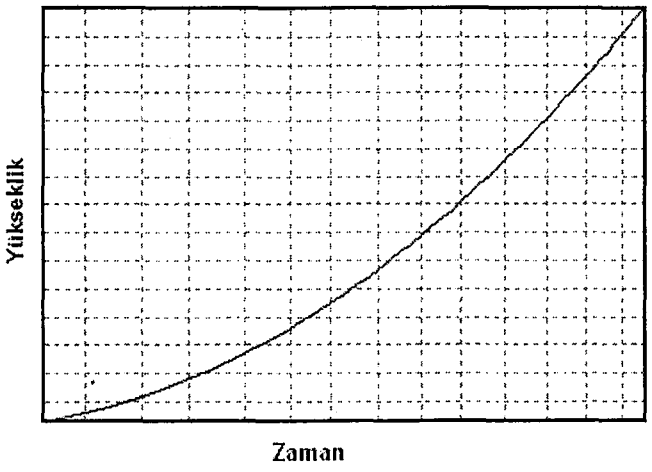
$$g = G\frac{M_E}{R^2_E} \tag{1.4}$$

Denklem 1.4, kuvvet denkleminde sadeleştirilen küçük kütleli cismin kütlesine bağlı değildir. Bu denklem yerçekiminden kaynaklanan ivmenin cismin kütlesinden, yoğunluğundan, hacminden ve hızından bağımsız olarak bütün nesneler üzerinde aynı olduğunu gösterir.

1.1. Serbest Bırakılan Cismin Yükseklik-Zaman Değişim

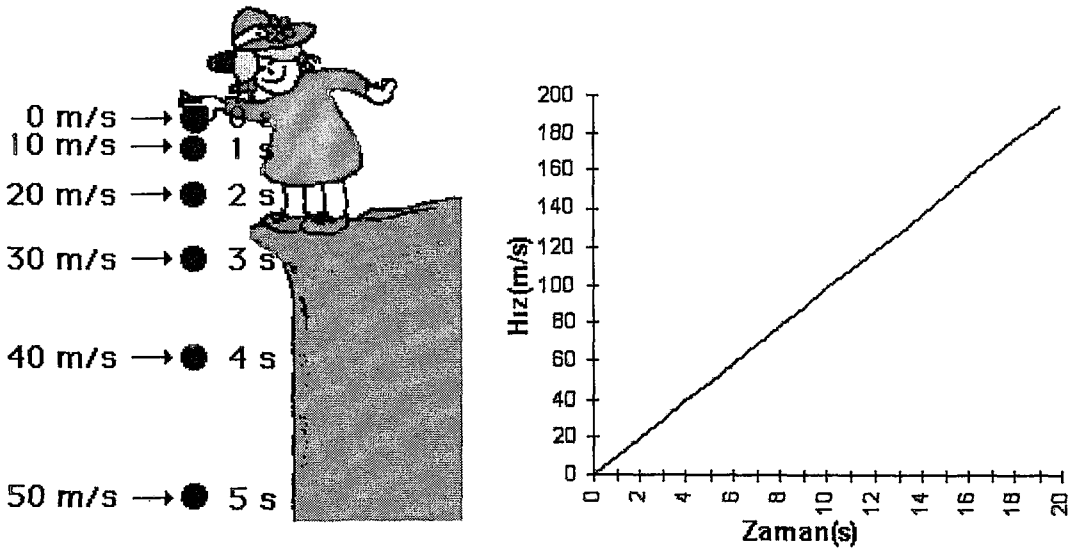
Serbest düşen cisimler yerçekimi ivmesi ile hızlandıkları için, yükseklik-zaman grafikleri eğri biçimdedir. Şekil 1.3'deki grafikte bir cismin yükseklik değerlerine bağlı olarak düşme zamanı değerleri verilmiştir. Denklem 1.5'te yükseklik ve düşme zamanı arasındaki bağıntı verilmiştir.

$$h = \frac{1}{2}gt^2 \tag{1.5}$$



Şekil 1.3. Yükseklik-düşme zamanı grafiği

1.2. Serbest Bırakılan Cismin Hız-Zaman Değişimi



Şekil 1.4. Hız-Zaman grafiği

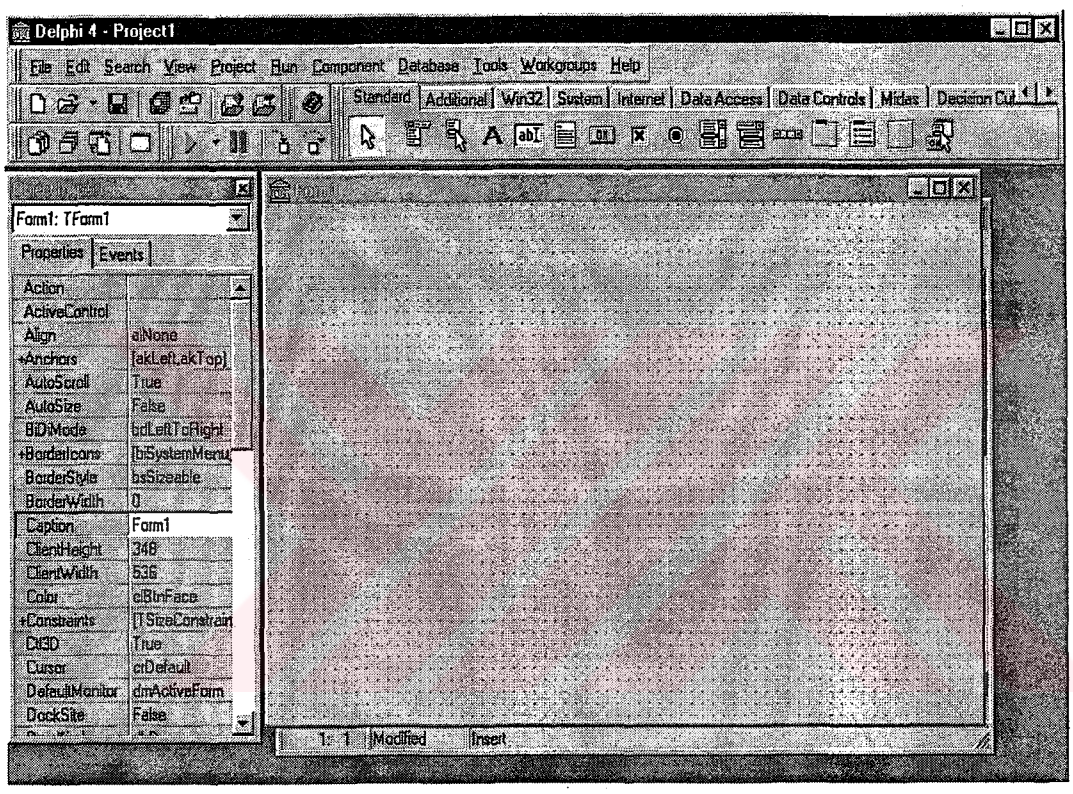
Cismin her sonraki saniyede hız değerinin 10m/s yükseldiği Şekil 1.1. deki hız-zaman değerlerinden görülmektedir [2].

1.3. Kullanıcı Ara Yüzünün Tasarımı

Kullanıcının istekleri doğrultusunda mekanik ve elektronik bileşenlerden oluşan atış sisteminin kontrolünde bir kullanıcı ara yüzüne ihtiyaç vardır. Bu ara yüz kullanıcının yapılmasını istediği işlemleri belirlenmiş olan algoritma doğrultusunda gerçekleştirir. Bu algoritmaların yürütülmesi için bir programlama dili kullanılarak kodlanıp yürütülebilir bilgisayar programına dönüştürülmeleri gerekmektedir. Burada programlama dili olarak 'Delphi' seçilmiştir.

1.3.1. Delphi görsel programlama dili

İlk önce 'Visual Pascal' olarak adlandırılan Delphi, Borland tarafından hazırlamıştır. Bir çok yönü ile pascal programına benzemektedir. Bu yüzden daha önce pascal çalışmış olan programcılar çok rahatlıkla Delphi'yi öğrenebilirler. Yapılması gereken en önemli şey, görsel programlama yapısını öğrenmektir



Şekil 1.5. Delphi ekranı

Bu ekranda aslında üç ayrı pencere görünmektedir. Bunlar "Delphi'nin ana ekranı", "Form" penceresi ve "Object Inspector" penceresi. Bunlardan başka dördüncü bir pencere olan "Kod penceresi", formun altında kalmaktadır. Delphi'nin ana ekranı, tüm bu pencerelere hakimdir. Yani ana ekranın kapanması durumunda diğer pencerelerde kapanır. Fakat form penceresi veya diğer pencerelerden birinin kapanması, ana ekranı etkilememektedir. Dolayısıyla Delphi'nin kapladığı alan, yalnızca ana ekranın kapladığı alan kadardır [3].

1.4. Sistemin Eğitimde Kullanım Amacı

Serbest düşme deneyi gerçekleştirilmesi, yükseklik seviyesinin belirlenerek bırakıldığı andan yere düşene kadar geçen sürenin ölçülmesi şeklindedir. Her seferinde yüksekliğin istenilen seviyede doğru olarak ayarlanması ve cismin bırakıldığı andan düşmesine kadar geçen zamanın ölçülmesinde hataların oluşması kaçınılmazdır. Yükseklik bulma ve süre ölçme işlemlerini otomatik olarak gerçekleştiren bir sistemin kullanılması deneyin gerçekleştirilmesinde oldukça kolaylık sağlar. Bu işlem bilgisayar kontrolünde gerçekleştirileceği için sonuçları grafiksel olarak görmek ve formül yoluyla elde edilen grafikle karşılaştırılabilir. Ayrıca oluşabilen hatalarda önemli ölçüde azaltılır.

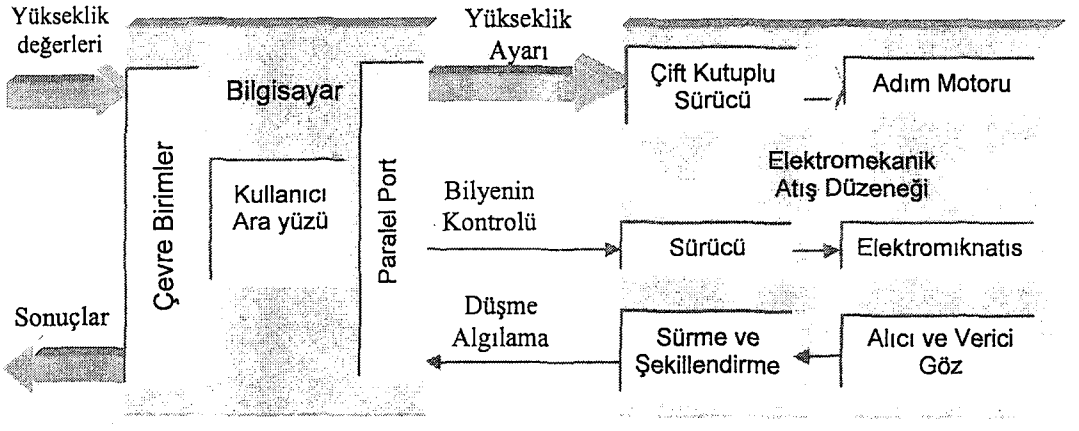
Günümüzde, serbest düşme hareketi fizik dersinin bir konusu olarak incelenmekte ve laboratuar uygulamalarında bununla ilgili deneyler gerçekleştirilmektedir. Tasarımı gerçekleştirilmiş olan deney seti bu deneylerin gerçekleştirilmesinde ve elde edilen sonuçların değerlendirilmesinde kolaylık sağlamaktadır.

BÖLÜM 2. ATIŞ SİSTEMİNİN BİLEŞENLERİ

2.1. Giriş

Serbest düşme sırasında deneysel ölçümlerin yapılabilmesi için, cismin serbest bırakıldığı yüksekliğin bilinmesi ve bırakıldığı andan yere düşmesine kadar veya belli bir noktaya ulaşana kadar geçen sürenin bilinmesi gerekir. Bunlara ek olarak cismin hacmi, kütlesi ve havayla sürtünme katsayısı gibi değişkenler de eklenebilir.

Yüksekliğin belirlenmesi ve düşme süresinin ölçülmesi değişik şekillerde olabilir. Örneğin cisim metre ile ölçülerek belirlenmiş bir yükseklikten bırakılarak bir kronometre ile düşme süresi ölçülebilir. Fakat bu şekilde yapılan bir deneyde, kronometrenin tam zamanında başlatılıp durdurulması, yüksekliğin yanlış olarak ölçülendirilmesi gibi sorunlardan elde edilecek sonuçların hata değerleri yüksek olur. Ayrıca bir yükseklik-düşme zamanı grafiği oluşturmak için defalarca aynı işlemi tekrarlamak yorucu ve sıkıcı olmaktadır.



Şekil 2.1. Serbest düşme deney setinin prensip şeması

Serbest düşme deney seti, oluşabilecek hataları minimuma indirgemek, deney yapan kişiyi monoton işlemlerden kurtarmak için tasarlanmıştır. Bunun için, serbest düşme işlemini gerçekleştiren, mekanik ve elektronik parçalardan oluşmuş bir sistem ve bu sistemi kontrol eden bir bilgisayar kullanılmıştır. Şekil 2.1'de deney setinin prensip şeması görülmektedir.

Bilgisayar, bir görsel programlama diliyle yazılmış olan ara yüz aracılığıyla yükseklik kontrolünü ve zaman ölçme işlemini mekanik atış düzeneğini kullanarak gerçekleştirir. Mekanik sistemin yapısında serbest bırakılacak cismin yüksekliğini ayarlamak için bir adım motoru ve bunun üzerinde hareket ettiği sonsuz bir dişli bulunmaktadır. Burada sonsuz dişlinin uzunluğu cismin serbest bırakılabileceği maksimum yüksekliği belirler. Cismin adım motorunun belirlediği yükseklikte tutulabilmesi için bir elektromıknatıs kullanılmıştır.

Serbest bırakılan cismin düşme süresinin belirlenmesi için bırakılma ve düşme zamanlarının bilinmesi gerekir. Düşme noktası olarak belirlenmiş olan noktada alıcı ve verici gözler bulunmaktadır. Cismin bu noktaya ulaşması alıcı ve verici gözün iletişiminin kesilmesiyle anlaşılır.

2.2. Kullanıcı Ara Yüzü

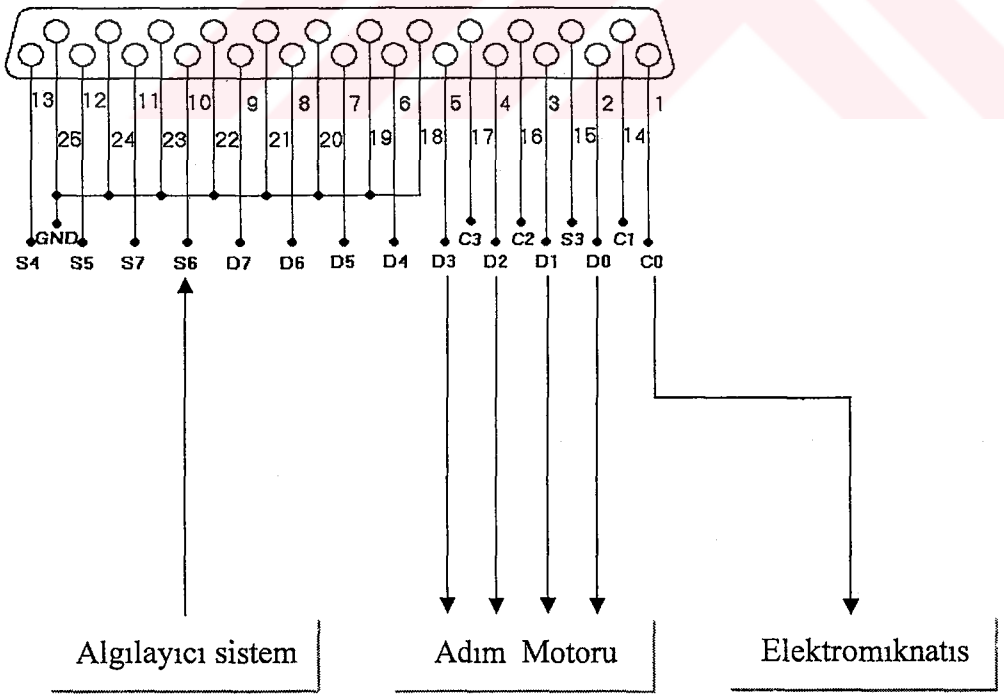
Ara yüz, kullanıcının yapılmasını istediği işlemleri bilgisayarda yürüten bir kontrol programıdır. Bu program atışların yapılması sırasında atış sistemini yazılmış olan çalışma algoritmasına göre kontrol eder. Ara yüz kullanılarak yapılması istenen atışların yükseklik değerleri ve tekrar sayıları girilir ve bu değerlere bağlı olarak ölçülen düşme süreleri liste veya grafik olarak görülür. Hassas ölçümler sonucunda serbest düşme zamanlarına bağlı olarak grafik çizdirilerek, yükseklik - düşme zamanı ilişkisi görsel olarak sunulur. Serbest düşme deney setinin bilgisayar kontrollü olarak tasarlanması, kullanımını kolaylaştırarak kullanıcıyı rutin işlerden kurtarır ve daha kesin sonuçlar elde edilmesini sağlar. Kullanıcı ara yüzü ile ilgili ayrıntılı bilgi Bölüm 3'te verilmiştir.

2.3. Paralel Portun Yapısı ve Sistemde Kullanımı

Bilgisayar dünyasında port, mikroişlemcinin, ya da CPU'nun, diğer bilgisayarlarla veri alışverişinde kullandığı bir sinyal hatları kümesidir Paralel port temel olarak yazıcı bağlantısı için tasarlanmıştır. Her pinin bir görevi vardır. 25 pin DB25 tipi terminal ile gerçekleştirilmiş paralel port soketi üzerinde, yazıcı kontrollerinin gerçekleştirildiği, Veri (Data), Kontrol (Control) ve Durum (Status) isimlerinde üç adet port bulunmaktadır:

Veri, 378H
Durum, 378H+1= 379H
Kontrol, 378H+9= 37AH

Veri portu 8 bitlik çıkış/giriş, Durum portu 5 bitlik giriş, Kontrol portu 4 bitlik çıkış/giriş ucu içermektedir.

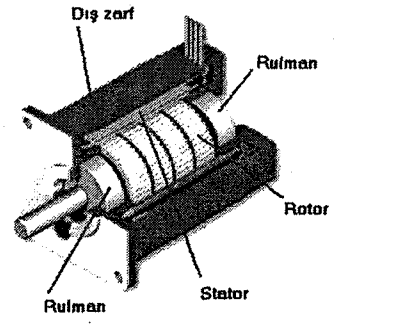


Şekil 2.2. Paralel port ve kontrolündeki elemanlar

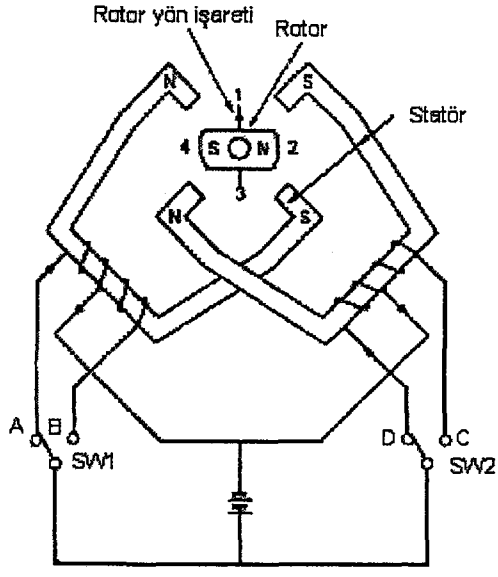
Kullanıcı ara yüzü paralel portu, deney setinin mekanik-elektronik atış sisteminin elemanlarını kontrol etmek ve bilgi almak amacıyla kullanılır. Şekil 2.4'de paralel port ve mekanik-elektronik sistem arasındaki bağlantı şeması görülmektedir. Adım motorunu kontrol için veri kaydedicisinin ilk dört biti olan D2, D3, D4, D5 çıkış uçları kullanılmıştır. Program içerisinde motor kontrol bilgileri veri kaydedicisinin bu bitlerine gönderilir. Elektromıknatıs için kontrol kaydedicisinin kontrol kaydedicinin ilk biti olan C0 ucu kullanılmıştır. Algılayıcı sistemin bilyenin ulaşıp ulaşmadığını belirten durum bilgisi çıkışı ise durum kaydedicinin S6 çıkış ucuna uygulanmıştır [4].

2.4. Adım Motorunun Yapısı

Bütün döner elektrik motorları temel olarak bir stator, bir rotor ve sargılardan oluşur. Geleneksel bir elektrik motoruna enerji verildiği zaman, rotor, güç kesilene kadar kesintisiz olarak döner. Adım motorunun çalışması bundan farklıdır, çünkü motor açıldığı zaman bile şaft (mil), motora bir adım palsı gönderilene kadar hareketsiz kalır. Adım motoru tahrik (çalıştırma) devresi bir adım darbesi aldığı zaman, rotoru belli bir açı kadar, ya da bir adım hareket ettirir ve bir sonraki adım darbesini alana kadar durur.



Enerjilenen fazlar	Rotor konum numarası	Rotor yönü
A, D	1	Saat yönü / Saatin tersi yönü
A, C	2	
B, C	3	
B, D	4	
A, D	1	



Şekil 2.3. Adım motorunun yapısı ve çalışma prensibi

Dolayısıyla maksimum motor yükünün aşılmaması koşuluyla, milin toplam açısal yer değişimi, adım açısı ile verilen darbe sayısının çarpımına eşittir. Mil konumunun, adım darbelerinin sayısıyla doğru orantı içinde olduğunu söylersek bu ilişki daha da basitleşir; çünkü adım motorundaki adım açısı sabittir. Bu sebeplerden adım motorlarında kontrol açık döngülü olabilir. Bu da yer değiştirme transdüserlerine veya karmaşık geri beslemeli kontrol sistemlerine ihtiyaç olmadığı anlamına gelir.

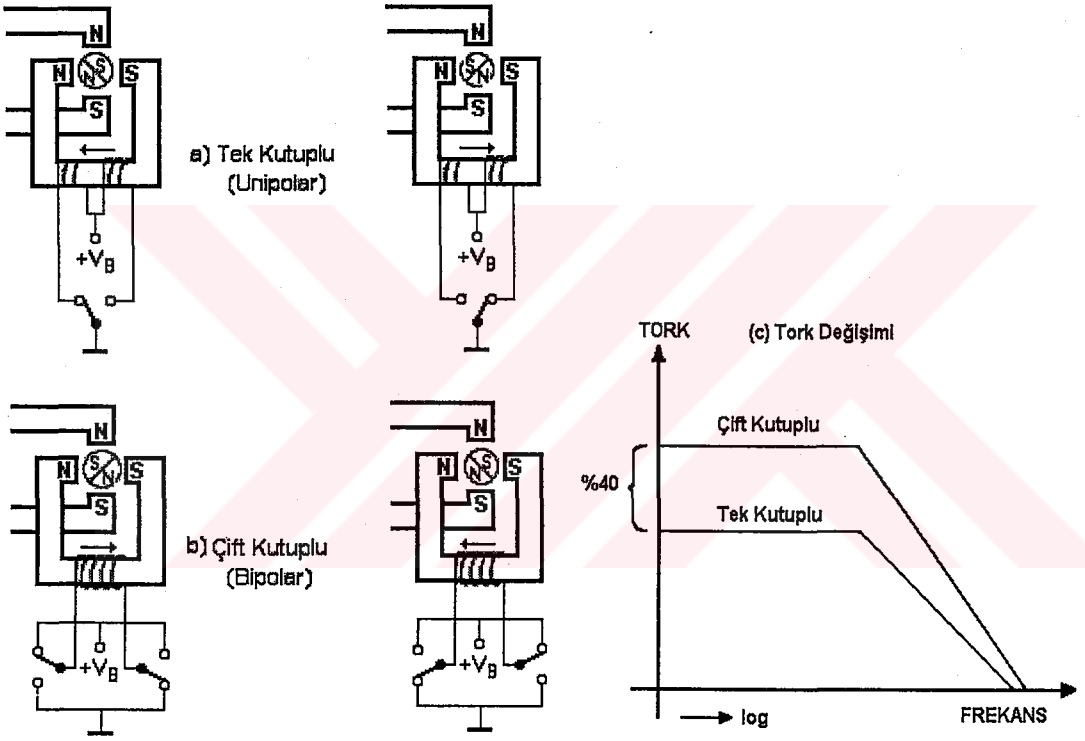
Şekil 2.3'te anahtarlar gösterilen konumlardayken, A ve D fazları enerjilenir, böylece rotor 1 konumuna hizalanır. Bu noktada SW2, C fazı enerjilenecek şekilde değiştirilirse, statorun o kolundaki manyetik alan tersine çevrilir. Manyetik alandaki yön değişikliği, rotoru saatin ibresi yönünde 90° çeker ve rotor 2 konumuna hizalanır. Saatin ibresi yönünde dönüşe devam etmek için SW1, B'ye anahtarlanır, daha sonra SW2 D'ye ve son olarak SW1, geri A'ya anahtarlanarak bir devir tamamlanır. Anahtarlama sırasının tersten alınması halinde rotor saat ibresinin tersi yönde hareket eder. Bu motorun, adım açısının 90° olmasına karşın, anahtarların, seçildiği zaman her iki stator fazını da kapatan üçüncü bir pozisyonu olması halinde yarım adım çalışması da mümkün olacaktır. Tablo 2.1'de tam ve yarım adım çalışmaya ait kontrol sinyalleri verilmiştir.

TAM ADIM					YARIM ADIM				
	Fan					Fan			
Adım	A	B	C	D	Adım	A	B	C	D
1	1	0	0	1	1	1	0	0	1
2	1	0	1	0	2	1	0	0	0
3	0	1	1	0	3	1	0	1	0
4	0	1	0	1	4	0	0	1	0
1	1	0	0	1	5	0	1	1	0
					6	0	1	0	0
					7	0	1	0	1
					8	0	0	0	1
					1	1	0	0	1

Tablo 2.1. Adım motorunun tam ve yarım adım döndürülmesi

2.4.1. Sürücü devre dizaynı

Devreyi tasarlayan kişi açısından adım motorları tek kutuplu ve çift kutuplu olmak üzere iki temel gruba ayrılır. Bobin veya bobinlerdeki akım akış yönü değiştirilerek stator kutuplarının manyetik alanı ters çevrildiğinde motor bir adım hareket eder. Tek kutuplu ve çift kutuplu motor arasındaki fark, manyetik alanın ters çevrilmesi işleminin gerçekleştirilme şeklidir. Şekil 2.4'te bu işlemler prensip olarak görülmektedir.



Şekil 2.4. Adım motorunun tek kutuplu ve çift kutuplu modda çalıştırılması

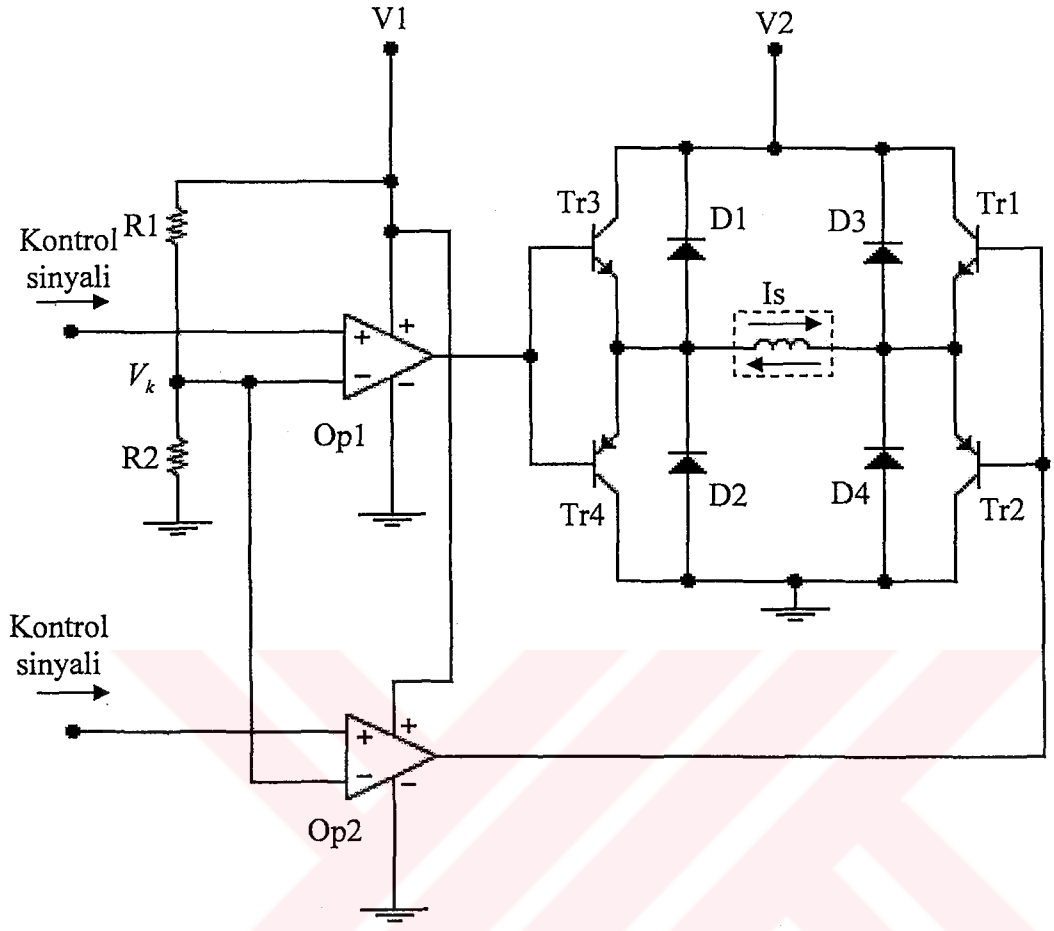
Adım motorunun torku stator sargılarının manyetik alan yoğunluğuyla orantılıdır. Torku artırmak için sargı eklenmesi veya sürme akımının artırılması gerekir. Akım artışına karşı doğal limit, genelde pek önemsenmese de demir çekirdeğin doyum noktasına ulaşmasıdır. Bundan daha önemli olan, stator sargılarındaki güç kaybından dolayı motorun maksimum ısı yüksekliğidir. Bu, sargının ikiye bölünmesinden dolayı bakır direncinin yalnızca yarısına sahip olan tek kutuplu sisteme kıyasla, çift

kutuplu devrenin bir avantajını gösterir. Sargı akımı $\sqrt{2}$ kat artırılabilir ve bu tork üzerinde doğrudan oransal bir artış sağlar. Böylece çift kutuplu çalışan motorlar güç kaybı limitlerinde, aynı çerçeve üzerine inşa edilmiş tek kutuplu motorlara göre Şekil 2.8'de görüldüğü gibi, %40 fazla tork sağlarlar. Eğer yüksek torka ihtiyaç yoksa motor büyüklüğü veya güç kaybı düşürülebilir [5].

Serbest düşme deneyinde, bilyenin istenilen yükseklik seviyesine çıkartılması sırasında adım motoru kullanılır. Adım motorunun dikey eksende aldığı yol adım sayısından belirlenir. Bunun için bir adımda alınan mesafenin bilinmesi gerekir. Böylece adım sayısı ve bir adımda alınan yolun çarpımı adım motorunun gittiği mesafeyi belirtir.

Deneyin toplam süresinin kısaltılması için, adım motorunun istenen yüksekliğe ulaşma ve tekrar başlangıç noktasına dönme süresinin kısaltılması gerekir. Sürenin kısaltılması adım motorunun hızının artırılmasıyla mümkündür. Daha önce de belirtildiği gibi yüksek hız, yüksek torkla sağlanır. Bunun için adım motoru yüksek güçlü seçilmiştir. Motorun verebileceği torkdan tam olarak faydalanabilmek için çift kutuplu sürücü kullanılmıştır. Bunun tek kutuplu sürücüye göre dezavantajı daha kompleks yapıda olmasından kaynaklanır.

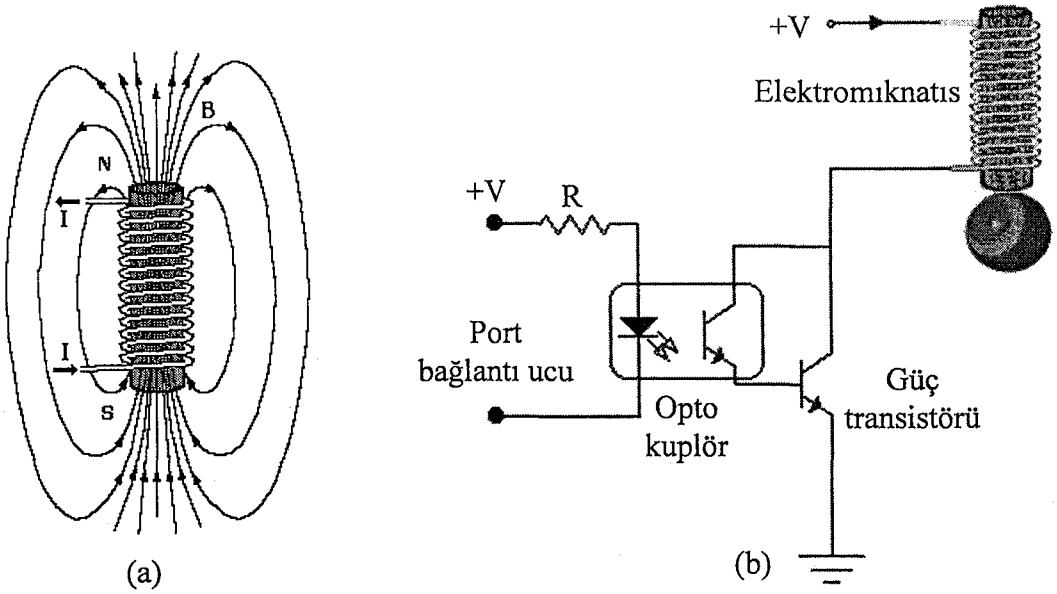
Şekil 2.5' te adım motorunun bir sargısını süren devre verilişitir. Bu devre ile adım motorunun bir sargısı üzerinden iki yönde de akım akıtılabilir. Bunun için bir eşlenik transistör çifti bir opampla sürülmüştür. Opampın çıkışı '1' ise NPN transistör, '0' ise PNP transistör iletime geçer. Tek kaynaklı besleme ile karşılaştırıcı olarak çalışan opamplar güçlendirme işlemi yaparak güç transistörleri için gerekli beyz akımını sağlarlar. Karşılaştırma gerilimi, V_k '1' ve '0' değerlerini ifade eden gerilim değerleri arasındaki etkisiz bölgede seçilir. Diğer sargı için de aynı devre kullanılmıştır.



Şekil 2.5. Adım motorunun bir sargısının sürülmesi

2.5. Elektromıknatısın Yapısı ve Deney Setinde Kullanımı

Bir iletkenin içinden geçen doğru akım, iletken etrafında mıknatıs kutbunda olduğu gibi bir manyetik alan meydana getirir. Biot-Savart'ın incelemelerinden, bir noktadaki alan şiddeti, iletkenden geçen akım şiddetiyle doğru, uzaklıkla ters orantılıdır. Manyetik alan içerisinde bulunan manyetik cisimlere bir kuvvet etki eder. Bu kuvvet manyetik alanın büyüklüğüne bağlı olduğu için manyetik alanın varlığı, manyetik çizgilere yaptığı kuvvetin etkisi ile anlaşılır. Bobin ile manyetik alan oluşumu Şekil 2.6'da verilmiştir [6,7].

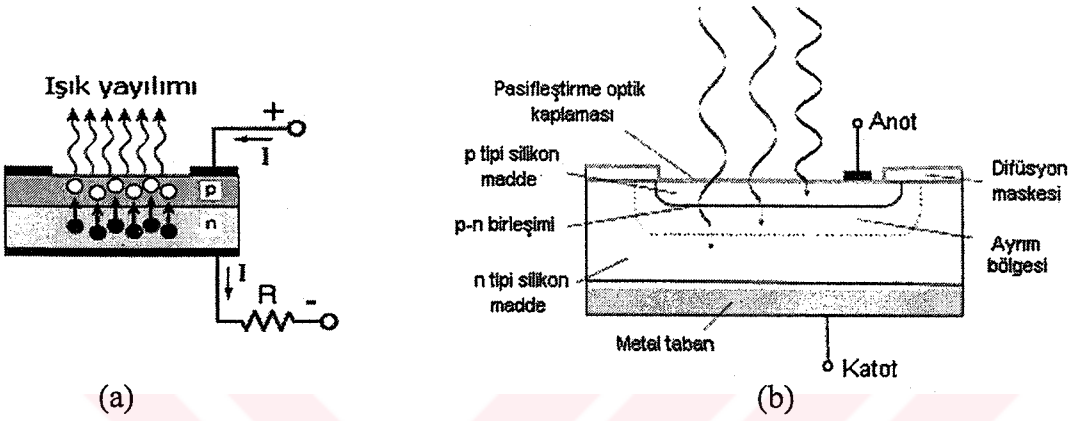


Şekil 2.6. (a) Elektro mıknatıs ve (b) port sürücü devresi

Bilyenin başlangıç noktasından alınması ve istenen yüksekliğe çıkartılıp bırakılması sırasında, cismin alınması ve bırakılması işleminde elektromıknatıs kullanılmaktadır. Elektromıknatıs üzerinden geçen akım portun izin verdiği maksimum akımın çok üstünde olduğu için Şekil 2.6'da görüldüğü gibi bir güç transistörünün kullanımı gerekmektedir. Burada kullanılan opto kuplör yalıtım sağlayarak, güç transistörün bozulması durumunda portun zarar görmesini önler. Port bağlantı ucu program içerisinde mantıksal '0' seviyesinde olduğu durumlarda, opto kuplör içindeki led ışık vererek transistor den emitör- kollektör arası akım akışı sağlar. Bu transistörle darlington bağlantılı güç transistörü beyz akımı alarak elektromıknatıs üzerinden iletime geçer. Akım akışıyla birlikte elektromıknatıs istenen manyetik alanı oluşturur.

2.6. Algılayıcı Sistemin Yapısı ve Kullanımı

Kızılötesi algılayıcı, bir kızıl ötesi verici ve bu kızıl ötesi sinyalleri algılayan bir fotodiyot kullanılarak oluşturulmuştur. Şekil 2.7'de ışık yayan ve algılayan diyotların yapısı görülmektedir.

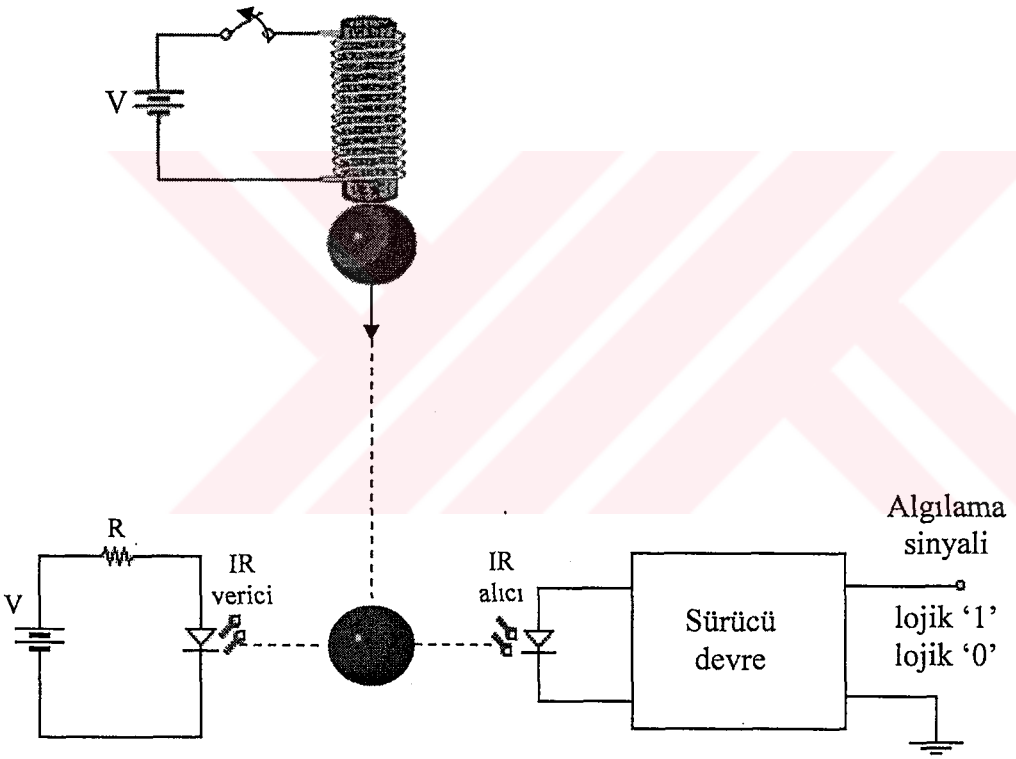


Şekil 2.7 (a) Işık yayan ve (b) algılayan diyotların yapısı

Işık yayan diyotlar (LED), galyum arsenik (GaAs), galyum arsenik fosfat (GaAsP) veya galyum fosfat (GaP) kullanılarak oluşturulan elemanlardır). Silikon ve germanyum ısı ürettikleri ve görülebilir ışık veya kızıl ötesi ışık üretmedikleri için LED yapımına uygun değildir. LED'e doğru yönde gerilim uygulandığında elektronlar ve holler p ve n maddeleri arasındaki aktif bölgeye doğru sürülerek enerji kızıl ötesi veya görülebilir fotonlara dönüştürülür. Elektron-delik çiftleri, elektron volt derecesine enerjisini bırakarak bir fotonun emisyonuyla daha karalı kesin bir duruma geçerler. Görülebilir spektrumun kızımızı uç noktası (700nm), fotonun kuantum enerjisini sağlamak için 1.77eV'luk bir enerjiyi bırakmayı gerektirir .

Foto diyot ışığı algılayarak elektrik akımına dönüştüren bir devre elemanıdır. İlke olarak ışığa maruz bırakılan her P-N eklemi foto diyot olarak çalışır. p-n birleşiminin karakteristiği bilinen birleşimlerden farkı, p maddesinin oldukça ince oluşudur.

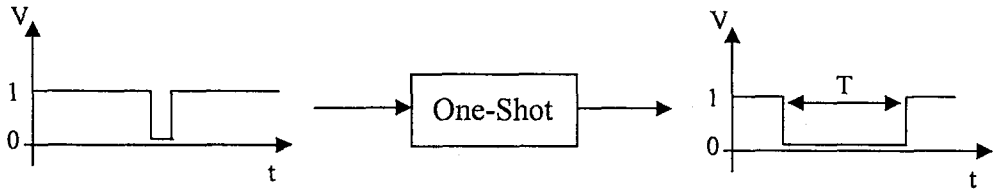
P maddesini inceliği algılanacak olan dalga boyu tarafından belirlenir. P-N birleşim bölgesi ve ayrım bölgesi foto diyotun çalışmasında en önemli kısımlardır. Bu bölgeler delikler ve elektronlar içerir. Yarı iletkenin eşiğinden yüksek enerjili fotonlar emildiğinde elektron-delik çiftleri oluşur. Ayrım bölgesinin karşısındaki büyük elektrik alanı, ayrım bölgesindeki elektronları ve delikleri hızlıca n ve p bölgelere ilerletir. Ayrım bölgesinin dışında üretilen elektronlar ve delikler birleşim bölgesine eriştiklerinde elektrik alanı tarafından karşıya iletilirler. Eğer İki uç elektriksel olarak birleştirilirse bağlantı boyunca bir akım akar ve böylece diyot bir akım kaynağı olarak davranmış olur [7,8] .



Şekil 2.8. Bilyenin düşme anında algılanması

Serbest düşme deney setinde cisim elektromıknatıs yardımıyla çıkarıldığı yükseklikten bırakıldıktan sonra düşme anında kullanıcı ara yüzü tarafından algılanması için kızıl ötesi algılayıcı sistem kullanılır. Şekil 2.8'de görüldüğü gibi algılayıcı sistem bilyenin düşerken izlediği yol üzerine yerleştirilmiştir. Bilye alıcı ve verici arasına ulaştığında algılama sinyali çıkışında mevcut lojik konumun tersi oluşur.

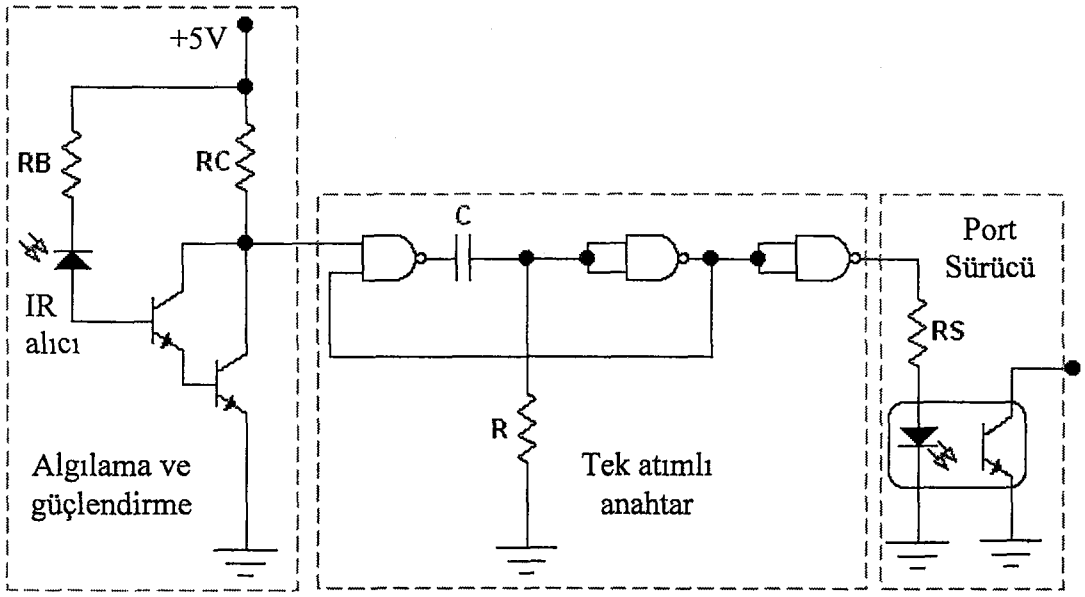
Konum değişimi sırasında algılama işaretini bir süre tutmak ve parazitsiz bir sinyal sağlamak için tek atımlı bir anahtar kullanılmıştır.



Şekil 2.9. Tek atımlı anahtarın giriş-çıkış sinyalleri

Tek atımlı anahtar Şekil 2.9'da görüldüğü gibi girişindeki kısa süreli palse karşılık süresi ayarlanabilen uzunlukta bir pals üretir [9]. Burada pals uzunluğu kullanıcı ara yüzünün algılayıcı sistemi kontrol aralığı olan 1 ms olarak seçilmiştir.

Algılama sinyalini işleyerek port girişine uygulayan devre şeması Şekil 2.10.'da verilmiştir. Algılama ve güçlendirme devresindeki darlington bağlantılı transistörler algılama sinyalini lojik seviyeye dönüştürerek tek atımlı anahtar girişine uygular. R ve C değerleri ile pals süresi belirlenir. Tek atımlı anahtar çıkışına bağlı opto kuplör port girişine bilyenin durumunu belirten sinyali uygular.



Şekil 2.10. Algılama devresi

2.6. Sonuçlar

Bu bölümde deney sisteminin genel yapısının, kullanıcı ara yüzü ve mekanik-elektronik atış sisteminden oluşturulduğu belirtildi. Deney sisteminin tasarımının ağırlıklı olduğu kısım kullanıcı ara yüzü oluşturduğu için, yapısı ve tasarımı hakkındaki detaylı bilgi Bölüm 3'de verilmiştir.

Atışların gerçekleştirildiği sistem adım motoru, elektromıknatıs ve algılama sistemini içermektedir. Adım motoru ve elekromıknatıs birlikte hareket ederek bilyeyi başlangıç noktasından alma ve istenilen yükseklik seviyesinde bırakma işlevini gerçekleştirirler. Algılama sistemi ise bilyenin düşme noktasına ulaşıp ulaşmadığını kontrol eder. Bu elemanlar kullanıcı ara yüzü tarafından bilgisayarın paralel portu aracılığı ile kontrol edilir. Paralel port çıkışının zarar görmesini önlemek ve sinyalleri istenen seviyede güçlendirmek sürücü devreler kullanılmıştır.

BÖLÜM 3. KULLANICI ARA YÜZÜNÜN TASARIMI

3.1. Giriş

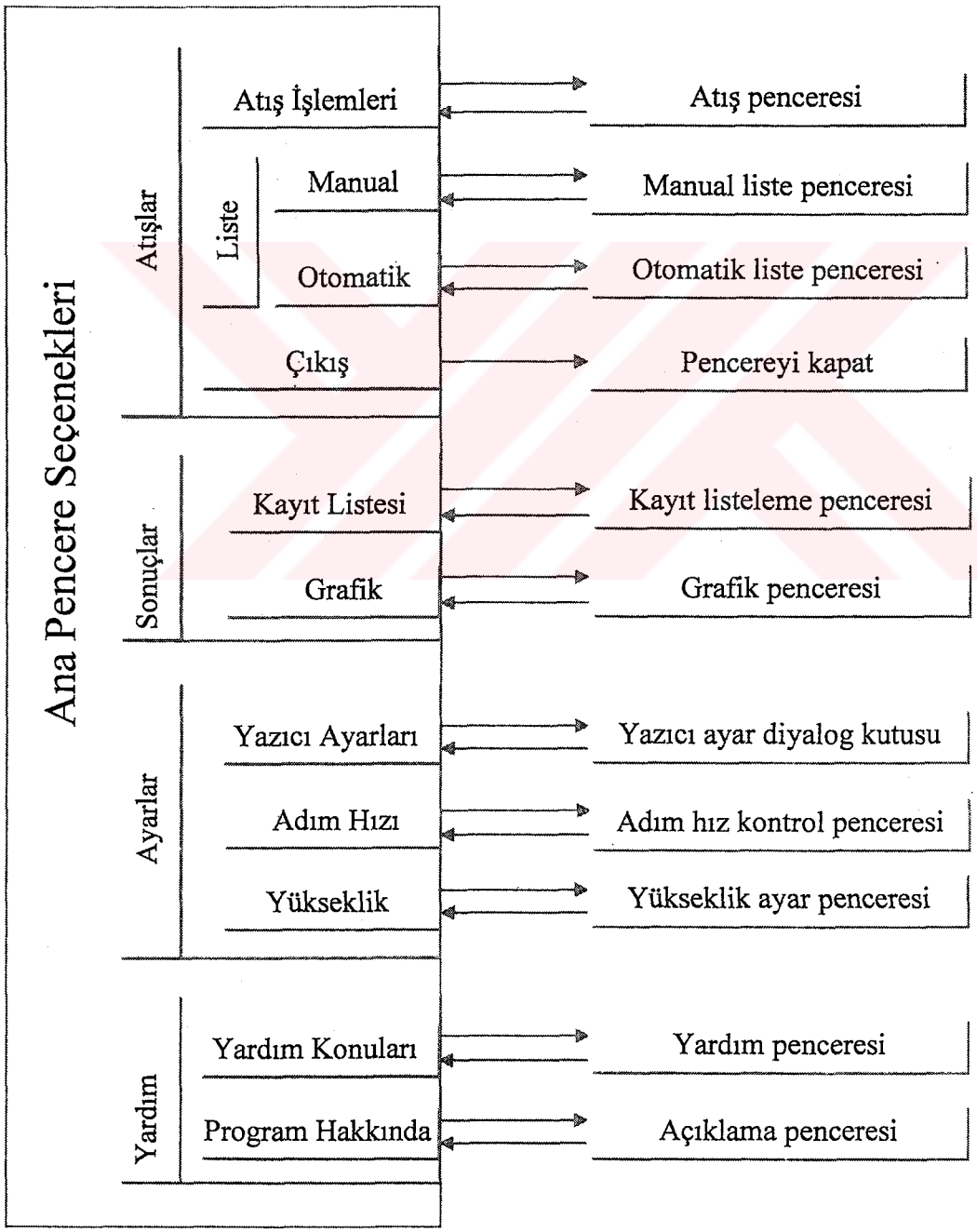
Serbest düşme deney setinin tasarımıyla, girilen yükseklik değerlerinde atışlar gerçekleştirerek, bunlara bağlı düşme zamanı değerlerini ölçüp sonuçları kaydeden ve yükseklik-zaman değerlerini grafik olarak verebilen bir deneysel sistemin gerçekleştirilmesi hedeflenmiştir. Bu işlemlerin gerçekleştirilmesi kullanıcı ara yüzü ve atış sistemi ile sağlanır. Burada atış sistemi mekanik ve elektronik bir yapıda olup, bilyenin istenen yüksekliğe çıkarılmasını ve düşme anında algılanmasını sağlar. Kullanıcı ara yüzü ise bilgisayar tarafından yürütülen görsel bir programdır. Bu ara yüz, atışların yapılması sırasında atış sistemini otomatik olarak kontrol eder. Serbest düşme deney setinin bilgisayar kontrollü olarak tasarlanması, kullanımını kolaylaştırarak kullanıcının zaman kazanmasını ve daha kesin sonuçlar elde etmeye imkan sağlar. Hassas ölçümler sonucunda serbest düşme zamanlarına bağlı olarak grafik çizdirilerek, yükseklik - düşme zamanı ilişkisi görsel olarak sunulur. Serbest düşme deney seti kullanılarak yapılan deneyde, işlemler aşağıdaki gibi sıralanabilir;

- Yükseklik ve tekrar değerleri girilir,
- Her yükseklik değeri için, otomatik olarak tekrar sayısınca atış gerçekleştirilerek kaydedilir,
- Kayıtlar Liste veya grafiksel olarak incelenebilir.

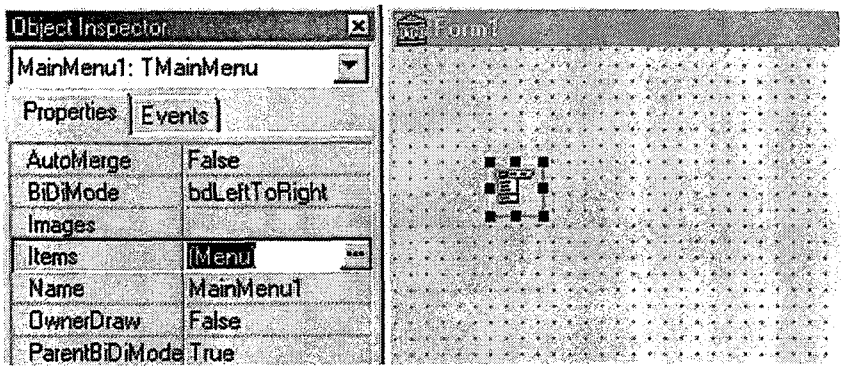
Serbest düşme deneyi işlem basamaklarının bilgisayar kontrolünde çalışan bir ara yüz tarafından gerçekleştirilmesi, ara yüzün bir programlama dili kullanarak yazılmasıyla mümkündür. Projenin gerçekleştirilmesinde "delphi" görsel programlama dili seçilmiştir.

3.2. Ana Pencere

Ana menü, kullanıcının programın kontrolünü sağlamak için yükseklik değerleri girme, atışları gerçekleştirme, grafik çizme, kayıt listeleme gibi işlem seçeneklerine ulaşabildiği penceredir. Program çalıştırıldığında ilk önce bu pencere aktif olur. Daha sonra yapılacak işlemler bu pencerede belirtilen seçenekler kullanılarak gerçekleştirilir.

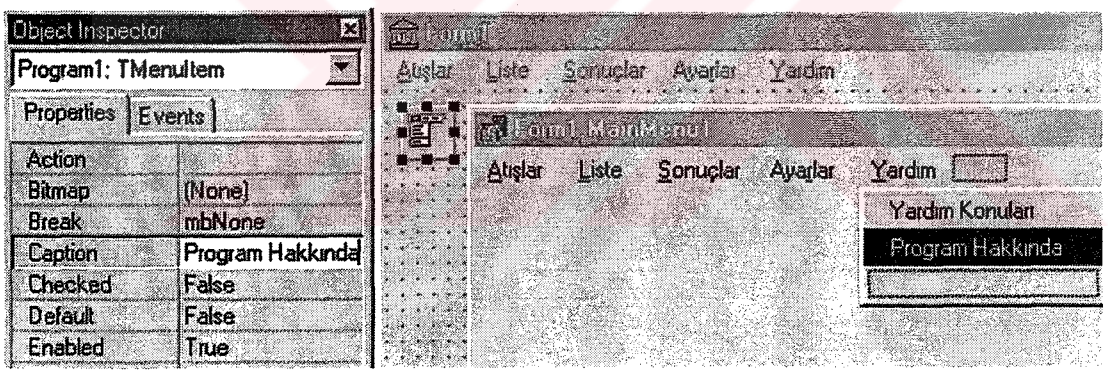


Şekil 3.1. Ana pencere seçenekleri



Şekil 3.2. 'MainMenu' nesnesi

Ana pencere seçenekleri Şekil 3.2' de görülen 'MainMenu' nesnesi kullanılarak oluşturulur. Form üzerine yerleştirilen 'MainMenu' nesnesinin 'Items' özelliği seçildiğinde Şekil 3.3'de görülen pencere çıkar. Buradan pencere üzerinde olması istenen seçenekler 'Caption' özelliği kullanılarak belirtilir.



Şekil 3.3. Ana pencere seçenekleri

Oluşturulan bu seçenekler kendi başlarına bir anlam ifade etmezler. Burada belirtilen seçeneklerin Şekil 3.1' de belirtildiği gibi alt formlarla ilişkilendirilmesi gerekir. Bunun için program tarafından, her hangi bir işlem seçiminde, Şekil 3.4' de görüldüğü gibi seçeneğe bağlı olarak bir alt program yürütülmektedir. Burada açılan 'begin-end' aralığına, yapılacak işleme göre belirtilmiş program kodları yazılır. Bu kodlar genelde ana pencerenin bulunduğu formun saklanması, ve seçenekle ilgili alt formun açılmasıyla ilgilidir. Alt formun açılması 'Show', ana formun saklanması 'Hide' özelliğiyle sağlanır. Alt formdan dönüşte ise alt form 'Close' özelliğiyle kapatılır ve saklanmış olan ana form 'Show' özelliğiyle tekrar gösterilir [3] .

```
procedure TForm1.Secenek1Click(Sender: TObject);
begin
//Seçeneğe ait işlemler
.............
.............
end;

procedure TForm1.Secenek2Click(Sender: TObject);
begin
//Seçeneğe ait işlemler
.............
.............
end;
.
.
```

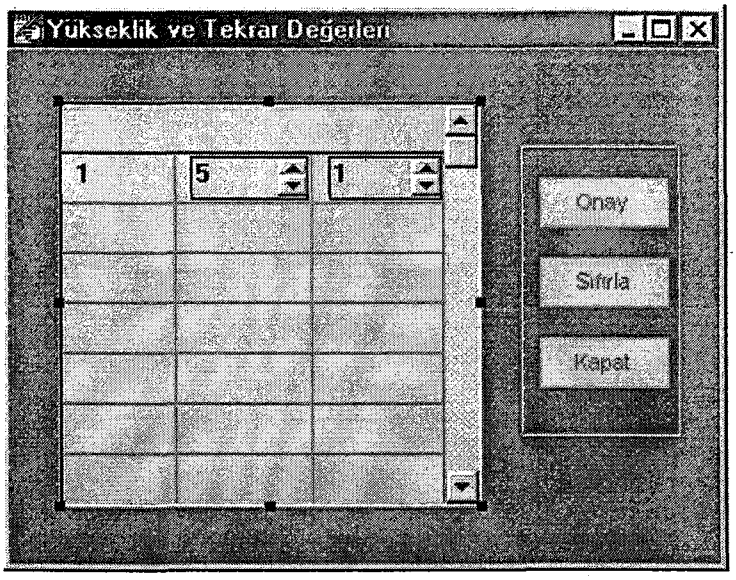
Şekil 3.4. Ana pencere seçeneklerine ait alt program yapısı

3.3. Yükseklik Listesi

Programda, kullanıcının istediği yüksekliklerden atışlar yapabilmesi için, atışlara başlamadan önce yükseklik değerlerinin bir liste olarak belirtmesi gerekir. Bununla birlikte, oluşacak hataları minimuma indirip daha güvenli sonuçlar elde etmek için, her atışın aynı yükseklik değerinde kaç kez tekrarlanacağını belirten tekrar sayısı girişi yapılır. Aynı yükseklik değerinde tekrar sayısınca düşme zamanı alındıktan sonra bu değerlerin aritmetik ortalaması alınır ve o atış yüksekliği için sonuç olarak kaydedilir. Yükseklik listesi her biri ayrı girilmek üzere 'Manuel' ve başlangıç, bitiş ve atış sayısı belirtilmek üzere 'Otomatik' olarak oluşturulabilir.

3.3.1. Manuel liste

Şekil 3.5.'te 'Manuel' listenin formu verilmiştir. Burada yükseklik ve tekrar değerleri girmek için 'SpinEdit' nesneleri kullanılmıştır. Buraya girilen her yükseklik ve tekrar değeri 'Onay' butonu ile 'StringGrid' nesnesinin sıradaki satırına yazdırılır. 'Sıfırla' butonuyla 'StringGrid' nesnesinin hücreleri temizlenir ve değerlerin geçici olarak saklandığı dizileri iptal etmek için bu dizilerin indeksi baştan başlatılır.



Şekil 3.5. Manuel Liste

Yükseklik ve tekrar değerleri girişi, Şekil 3.5'te görüldüğü gibi, iki adet 'SpinEdit' nesnesi ile yapılır. Üzerindeki ok tuşları, SpinEdit'in belirttiği değeri bir artırır veya azaltır. Bu nesnenin 'Value' özelliğiyle, gösterdiği değer, tamsayıyı belirten 'Integer' olarak okunarak işlemler içinde kullanılabilir. Böylece istenilen değer ayarlanarak listeye 'Onay' butonu ile aktarılabilir. Her değer girişi ile birlikte atış no bir artarak bir sonraki atış değerini belirtir

'Onay' butonuna basıldığında Şekil 3.6.da görülen alt program yürütülerek girilen yükseklik ve tekrar değeri 'AtisNo2' indeksi ile belirtilen dizi elemanına kaydedilir ve bir sonraki dizi elemanına giriş yapılabilmesi için 'AtisNo' değişkeni bir artırılır.

```
// Genel Değişkenler
var
  Form2: TForm2;
  AtisNo2    : Integer = 0;
  Yükseklik2: array[1..150] of Integer;
  Tekrar2    : array[1..10]  of Integer;

.....................................

procedure TForm2.OnayClick(Sender: TObject);
begin
Inc(AtisNo2); // dizi index değerini bir artır

Yukseklik2[AtisNo2]:= SpinEdit1.Value;//Yüksekliği kaydet
Tekrar2[AtisNo2]   := SpinEdit2.Value;//Tekrar sayısını kaydet

Edit1.Text := IntToStr(AtisNo2 + 1);//Sonraki atış no'yu göster

//StringGrid'in satır sayısı aşıldıysa satır sayısını artır
if AtisNo2>5 then
   begin
   StringGrid1.RowCount := AtisNo2;
   StringGrid1.TopRow   := AtisNo2 - 5;
   end;

//Girilen değerleri ve atış numarasını  listeye yaz
StringGrid1.Cells[0,AtisNo2+1]:=IntToStr(AtisNo2);
StringGrid1.Cells[1,AtisNo2+1]:=IntToStr(Yukseklik2 [AtisNo2]);
StringGrid1.Cells[2,AtisNo2+1]:=IntToStr(Tekrar2[AtisNo2]);

end;
```

Şekil 3.6. 'Onay' butonuyla yürütülen alt program

Kayıt işlemi için tanımlanan dizi geçici olarak kullanılır. Girilen yükseklik ve tekrar değerleri 'StringGrid' nesnesi liste olarak kullanılarak her defasında bir alt satıra yazılır. 5'ten fazla değer girildiğinde, yani ekranda görülen satır sayısını aştığında, listenin satır sayısını artırmak için 'RowCount' özelliği kullanılır. 'AtisNo2' indeksinin belirttiği değer girilen atış sayısını belirttiği için 'AtisNo2' değerini listenin satır sayısına eşitlemek bu sorunu çözmektedir. Ayrıca girilen son satırın listede gözükmesi için satırların her yükseklik ve tekrar girişinde bir satır yukarıya kaydırılması gerekir. Bunun için 'StringGrid' nesnesinin 'TopRow' özelliği kullanılır. Bu özellik en üstteki satırın belirtilmesini sağlar. Satır sayısı 5'i geçtiğinde en üstteki satır toplam satır sayısının 5 eksiğidir. Hücrelere yazı yazdırmada 'Cells[x,y]' özelliği kullanılır. Burada 'x' sütun, 'y' de satır numarasını

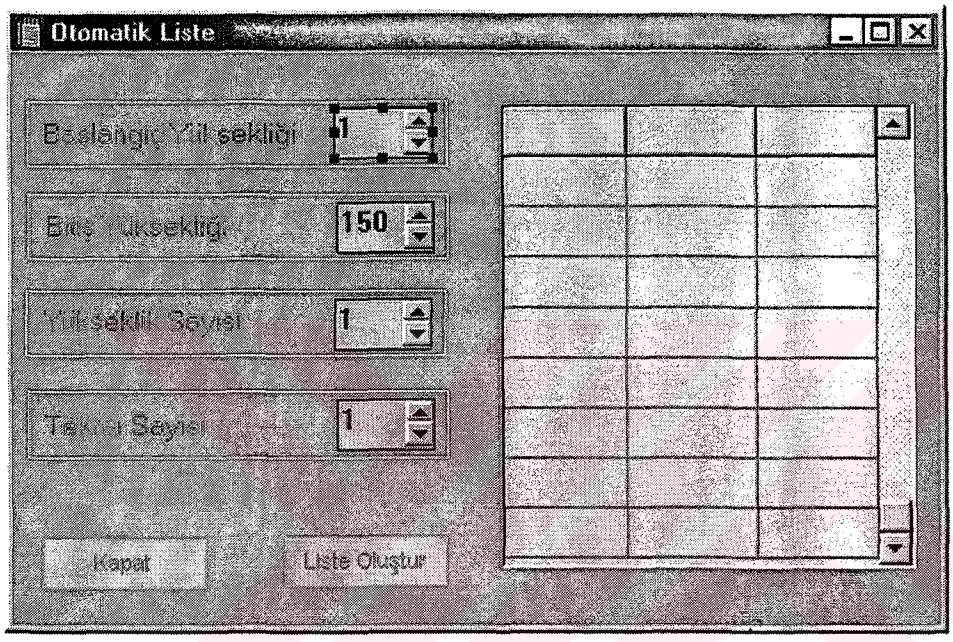
gösterir. Listeye girilen değerler değiştirilmek istendiğinde veya hatalı girildiğinde, 'Sıfırla' butonu ile liste sıfırlanabilir. Şekil 3.7'de 'Sıfırla' butonuyla yürütülen alt program görülmektedir. Bu butona basıldığında, uyarı mesajı penceresi ile tekrar onay alınır. Onay alındıktan sonra 'StringGrid' nesnesinin hücreleri 'for ... do' döngüsü kullanılarak, o ana kadar girilen değer sayısınca satır silinir. Yeni girilecek atış değerleri için kullanılan 'AtısNo2' indeksi sıfırlanır ve listenin aktif satırı 1. satır yapılır. Ayrıca 'Atış no', 'Yükseklik' ve 'Tekrar' değerleri başlangıç değerlerine kurulur.

```
procedure TForm2.SifirlaClick(Sender: TObject);
var i:integer;
begin

// Listede atış değerleri varmı?
if AtisNo2>0 then
   begin

   //Değerler silinsin mi?
   if messageDlg('Değerler Sıfırlansın mı?',
   mtConfirmation,[mbYes,mbNo],0)=mrYes then
      begin

      //listenin hücrelerini sil
      for i:=2 to AtisNo2+1 do
            begin
            StringGrid1.Cells[0,i] := '';
            StringGrid1.Cells[1,i] := '';
            StringGrid1.Cells[2,i] := '';
            end;

      //Yeni liste girişi için değişkenleri kur
      AtisNo2 := 0;
      StringGrid1.TopRow := 1;
      Edit1.Text:=IntToStr(1);
      SpinEdit1.Value:=5;
      SpinEdit2.Value:=1;
      end;
   end;

end;
```

Şekil 3.7. 'Sıfırla' butonuyla yürütülen alt program

3.3.2. Otomatik liste

Yükseklik listesini, belirtilen başlangıç ve bitiş yükseklik aralığında ve belirtilen sayıda eşit aralıkta yükseklik değerleri elde edilmesiyle oluşturulur. Şekil 3.8'de görüldüğü gibi, başlangıç ve bitiş yüksekliklerini, bu değerler arasındaki yükseklik ve tekrar sayılarını belirtme işlemleri için dört adet 'SpinEdit' nesnesi kullanılmıştır.



Şekil 3.8. Otomatik liste penceresi

Girilen değerlerin listelenmesinde yine 'StringGrid' nesnesi kullanılmıştır. 'Liste Oluştur' butonuyla, girilen değerlere göre elde edilen sonuçlar hücrelere, her bir satıra bir yükseklik değeri gelecek şekilde liste oluşturulur. Yeni bir liste oluşturulduğunda yükseklik ve tekrar değerleri dizi indeksi baştan başlatılarak aynı dizi üzerine yazılmasıyla eski liste değerleri iptal edilir.

Şekil 3.9'daki program kodlarından görüldüğü gibi, 'Listeyi_Sil' alt programıyla yükseklik değerleri listeye aktarılmadan önce, varsa eski değerler temizlenir. Daha sonra belirtilen bitiş yüksekliğinden, başlangıç yüksekliği çıkartılıp yükseklik sayısına bölünerek yükseklik aralığı belirlenir. Programda başlangıç değerini 'SpinEdit2', bitiş değerini 'SpinEdit2' ve yükseklik sayısını 'SpinEdit3' belirtmektedir. Yükseklik değerleri 'StringGrid' nesnesinde listeleneceği için satır

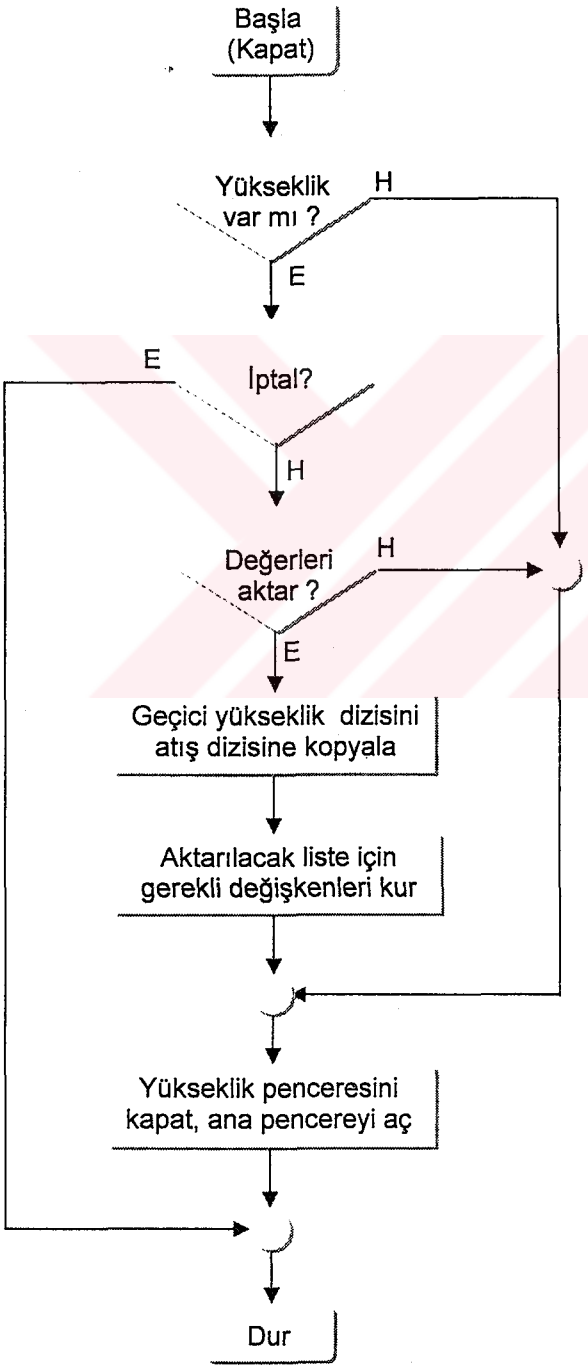
sayısı yükseklik değeri sayısına eşitlenir. Yükseklik değerlerinin geçici diziye aktarılması için bir 'for...do' döngüsü açılmıştır. Döngünün her bir tekrarında yükseklik değeri hesaplanan aralık değerince artarak geçici diziye aktarılır ve listeye yazılır.

```
procedure TForm3.ListeOlusturClick(Sender: TObject);
begin

Listeyi_Sil(StringGrid1); //Listeyi temizle

// Yükseklik değerleri arası aralığı bul
Aralik:=Round((SpinEdit1.Value-SpinEdit2.Value)/SpinEdit3.Value);

Sayi:=SpinEdit3.Value; // Yükseklik sayısını değişkene aktar

// Listenin satır sayısını ayarla
if SpinEdit3.Value > 8 then
   begin
   StringGrid1.RowCount:= Sayi+2;
   StringGrid1.TopRow  := Sayi-6;
   end;

Yukseklik:=0;


for i:=1 to Sayi do // Yükseklik sayısınca döngüyü tekrar et

    begin

    Yukseklik:=Yukseklik+Aralik;//Yüksekliği bir aralık artır

    Yukseklik2[i]:=Yukseklik; // Yüksekliği diziye aktar

    Tekrar2[i]    := SpinEdit4.Value;//Tekrar sayısını diziyeaktar

    // Belirtilen değerleri listeye aktar
    StringGrid1.Cells[0,i] := '  '+IntToStr(i);
    StringGrid1.Cells[1,i] := '  '+IntToStr(Yukseklik2 [i]);
    StringGrid1.Cells[2,i] := '  '+IntToStr(Tekrar2[i]);

    end;
end;
```

Şekil 3.9. 'Liste Oluştur' butonuyla yürütülen alt program

3.3.3. Listenin kapatılması

Otomatik veya Manuel liste formları kapatıldığı zaman Şekil 3.10'de görülen algoritma yürütülür. Listeye girilmiş yükseklik değeri yoksa, yükseklik penceresi kapatılarak ana penecere açılır.



Şekil 3.10. Pencerenin kapatılmasıyla birlikte yürütülen algoritma

Yükseklik değerleri girilmişse bir mesaj penceresi açılır. Bu penceredeki seçeneklerden 'İptal' seçilirse, kapatma işlemi iptal edilerek form kapatılmaz. 'Hayır' seçilirse form kapatılır fakat geçici olarak dizilere kaydedilmiş değerler, atışlar gerekleştirilirken kullanılan diziye kaydedilmez. 'Evet' seçildiğinde ise geçici dizilere kaydedilen değerler atış sırasında kullanılan dizilere aktarılır. Bu aktarma işlemi aynı indeksli elemanların birbirine eşitlenip, 'for...do' döngüsü kullanılarak atış sayısınca döngünün tekrar etmesiyle sağlanır. Bu işlem tamamlandığında Şekil 3.11' deki işlemlerin program kodlarındanda görüldüğü gibi bazı kurma işlemleri yapılır.

```
procedure TForm2.KapatClick(Sender: TObject);
label Cikis;
var i:Word;
    S:Byte;

begin
//Hehangibir değer girilmemişse soru sormadan çık
if AtisNo2=0 then  goto Cikis;
S:=MessageDlg('Değerler atış listesine aktarılsın
mı?',mtConfirmation,
   [mbYes, mbNo,mbCancel],0);

if not(S=2)then // İptal seçilmediyse
      begin
      if   S=6  then // Evet seçildiyse
            begin
            //Geçici diziyi aktar
            for i:=1 to AtisNo2 do
                  begin
                  Tekrar[i]    := Tekrar2[i];
                  Yukseklik[i]:= Yukseklik2[i];
                  end;

            //Kurma işlemleri
            Sec:=1;
            ListeyiSil:= True;
            AtisNo := AtisNo2;
            Atis :=1;
            TekrarNo:=1;
            end;

      Cikis:
      Form1.Show;
      Form2.Close;
      end;
end;
```

Şekil 3.11. Pencerenin kapatılmasıyla yürütülen alt program

3.4. Atışların Gerçekleştirilmesi

Yükseklik değerleri ve tekrar sayıları girildikten sonra yapılacak işlem atışların gerçekleştirilmesidir. Bir bilye ile serbest düşme atışının gerçekleştirilmesi için yapılması gereken işlemler;

- Bilyenin başlangıç konumundan alınması,
- Bilyenin atış yapılmak istenen yüksekliğe çıkartılması,
- Bilye serbest bırakılarak, düşene kadar geçen sürenin ölçülümü,
- Hareketli mekanizmanın başlangıç konumuna getirilmesi, şeklinde sıralanabilir.

Bilyenin, programın çalışmasına göre istenildiği zaman başlangıç konumundan alınıp istenilen yükseklikten bırakılması hareketli bir sistemle sağlanır. Bu hareketli sistem, adım motoru ve elektromıknatısın bir metal levha ile bağlantısından oluşur. Bilyenin tutulması veya serbest bırakılması için, hareketli sistem üzerinde bir elektromıknatıs yerleştirilmiştir. Böylece bilyenin manyetik alan vasıtasıyla tutularak hareketli sistem ile birlikte hareket etmesi sağlanır. Adım motorunun görevi, programın belirttiği adım sayısına göre sonsuz dişli üzerinde hareket ederek, elektromıknatıs yardımıyla bilyeyi istenilen yüksekliğe çıkarmaktır.

Atış düzeneğinde, hareketli sistemden başka bilyeyi düşme anında algılayan, alıcı ve verici gözlerden oluşturulmuş bir algılayıcı sistem bulunmaktadır. Algılayıcı sistem ile, bilyenin serbest bırakılmasından itibaren, port kullanılarak uygun olan eşit zaman aralıklarıyla (1 ms) kontrol edilerek topun düşme zamanı belirlenir.

3.4.1. Zamanlama işlemi

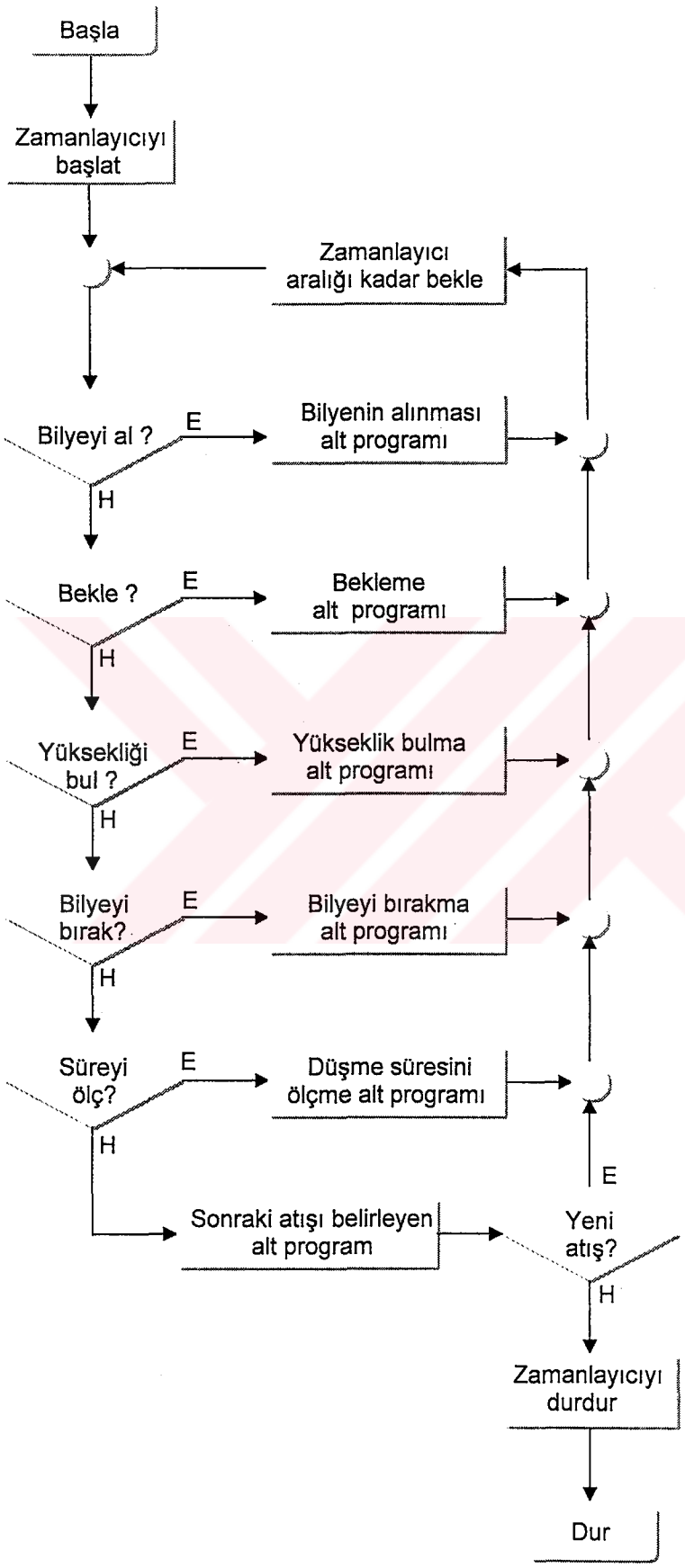
Serbest düşme deneyinin yapılmasında işlemlerin yürütülmesi için yazılmış programın büyük bir kısmı zamanlayıcı (timer) kontrolünde çalışan alt program tarafından yürütülür. Zamanlayıcı kontrolündeki alt program, değeri belirtilmiş olan sabit aralıklarla aktif edilerek içinde yazılı algoritma yürütülür.

Zamanlayıcının kullanılmasıyla iki önemli sorun ortadan kalkar. Birincisi, adım motorunun her adım için gereken bekleme süresi belirlenir. Diğeri ise, bilyenin serbest bırakılmasından düşene kadar algılayıcı sistemin sabit aralıklarla kontrol edilmesi için gerekli aralık süresini belirler.

Adım motoru, gücünün belirlediği çalışma frekansı üzerinde çalışamadığı için adım atma frekansı sınırlıdır. Saniyedeki adım sayısını belirten bu frekans değerinin kontrolü, zamanlayıcı tarafından kontrol edilen her adımdaki bekleme zamanı değiştirilerek sağlanır. Bu bekleme süresi başlangıç ve bitiş noktalarına yaklaşıldığında değiştirilerek, adım motorunun birden hızlanması veya durmasında meydana gelebilecek hataları önlemiş olur.

Belirlenen yüksekliğe ulaşmak için gereken adım sayısı ve her adımdaki bekleme süresinin çarpımı bize harcanan toplam zamanı verir. Bu yüzden, toplam zamanı kısaltmak iki için seçenek vardır; Birincisi, her adımda gidilen yükseklik uzunluğunu artırmaktır. Bunun için adım motorunun miline bağlı olan dişlinin yarıçapı büyültülerek, her adımda alınan yol artırılır ve böylece gereken adım sayısı azaltılarak süre kısaltılır. Diğeri ise zamanlayıcı ile belirlenen adım süresini kısaltmaktır. Programda hız kontrolü için zamanlayıcının belirlediği aralık değiştirilir.

Şekil 3.12' de verilen algoritma, zamanlayıcının aktif olmasıyla birlikte, yürütülen alt programdaki işlemleri özetlemektedir. Bu alt program, her zamanlayıcı aralığında bir kontrol edilerek, yine program akışının belirlediği bir alt program yürütülür.



Şekil 3.12 Zamanlayıcı kontrolünde yürütülen algoritma

Atışların gerçekleştirilmesinde zamanlayıcı kontrolünde çalışan algoritma genel olarak yedi ana işlem yürütür:

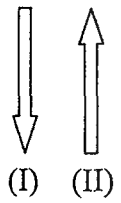
1. Hareketli mekanizma başlangıç konumuna getirilir ve elektromıknatıs enerjilendirilir,
2. Başlangıç konumunda bir süre bekleyerek elektromıknatısın oluşturduğu manyetik alan yardımıyla bilye tutulur,
3. Adım motoru, sıradaki yükseklik değerine göre hesaplanan adım sayısınca adım atar,
4. İstenilen yüksekliğe ulaşıldığı anda, program tarafından düşme süresi ölçümü için gerekli kurma işlemleri yapılarak bilye serbest bırakılır,
5. Bilye düşmeye başladığı andan itibaren algılayıcı sistem, bilye ulaşana kadar, sabit aralıklarla kontrol edilir,
6. Bilye ulaşmadıysa aynı atışın tekrarlanması için kullanıcının tekrar atışı başlatmasıyla 1. işlemden tekrar başlanır,
7. Atışlar bittiyse hareketli mekanizma başlangıç konumuna getirilir, bitmediyse tekrar aynı işlemleri yürütmek için 1. İşlemden başlanır.

```
procedure TForm1.STimer1Timer(Sender: TObject);
begin
if        ZamanOlc     then Zaman_Olc(STimer1,StringGrid1,Canvas)
else if   BilyeAl      then Bilye_Al(STimer1,Canvas)
else if   Bekle        then Bekle_(STimer1,Canvas)
else if   KonumBul     then Konum_Bul(Canvas,STimer1)
else if   BilyeBirak   then Bilye_Birak(STimer1,Canvas)
else  Sonraki_Atis(STimer1,Canvas,Liste,Ayarlar)
end;
```

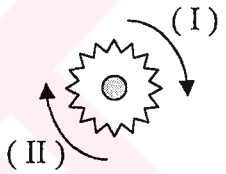
Şekil 3.13 Zamanlayıcı kontrolünde yürütülen alt program

3.4.2. Adım motorunun kontrolü

Adım motorunun program tarafından döndürülmesi, daha önce de bahsedildiği gibi bilgisayarın paralel portu ve bu portun çıkışının güçlendirildiği motor sürücü devre kullanılarak gerçekleştirilir. Adım motorunun kontrolü için paralel portun 'data' kaydedicisi kullanılır. 8 bitlik bu kaydedicinin 4 biti kullanılarak adım motorunun döndürülmesi için gerekli ikilik bilgi Şekil 3.14' te görüldüğü gibi üretilebilir. Adım motorunu bir yönde döndürmek için dört farklı sayı değerini belli bir sıraya göre kullanmak gerekir. Aynı sıra sondan geri doğru izlendiğinde, dönüş zıt yönde gerçekleşir.



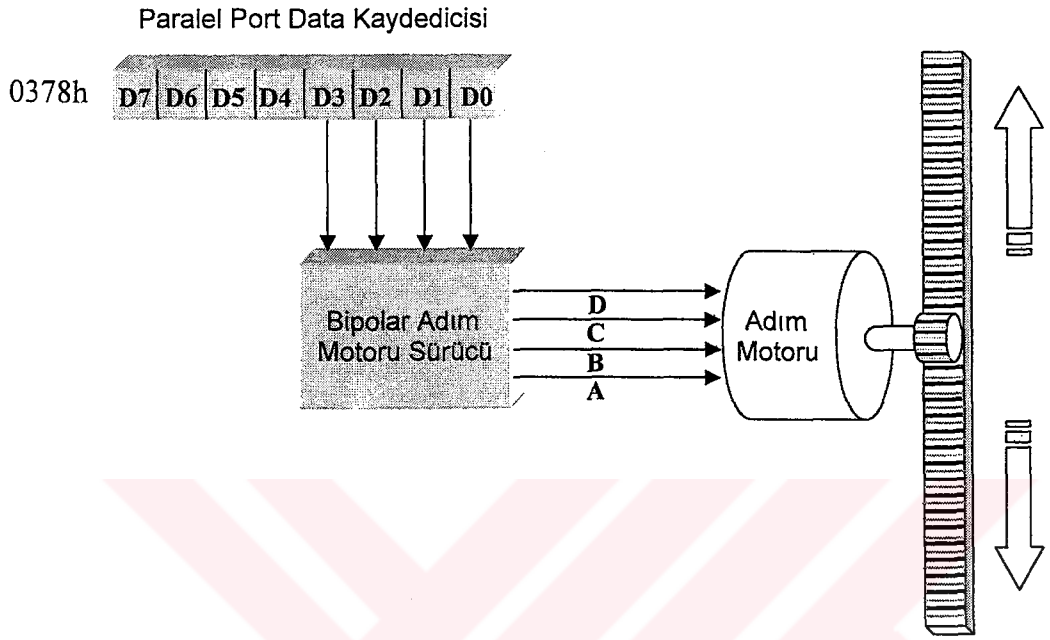
	D3	D2	D1	D0
1	1	0	1	0
2	0	1	1	0
3	0	1	0	1
4	1	0	0	1
1	1	0	1	0



Şekil 3.14 Çift kutuplu adım motorunun ileri ve geri döndürülmesi

Adım motorunun döndürülmesi için paralel port çıkışında elde edilen sinyallerin motor tarafından kullanılabilir seviyede güçlendirilmesi gerekir. Bunun için adım motorunun yapısına göre, tek kutuplu veya çift kutuplu sürücü devre kullanılır. Burada, Şekil 3.14 de ki sinyallerden de görüldüğü gibi, çift kutuplu (bipolar) adım motoru kullanıldığı için buna uygun bir sürücü tasarlanmıştır.

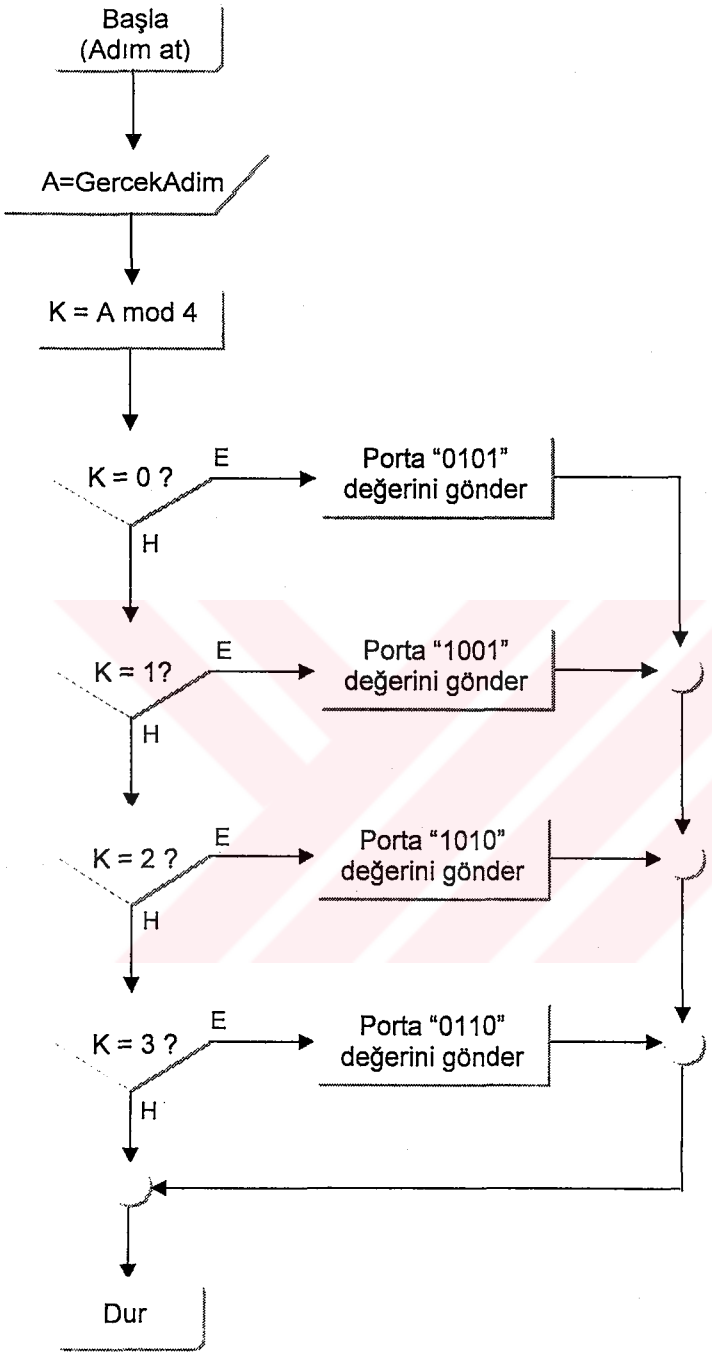
Şekil 3.15'te adım motoru, sürücü devre ve port arasındaki bağlantının prensip şeması görülmektedir. Paralel portun data kaydedicisinden alınan 4 bitlik bilgi sürücü devre tarafından güçlendirilerek adım motoruna uygulanır.



Şekil 3.15 Paralel portla adım motorunun döndürülmesi

Kullanıcı ara yüzü için yazılan program akışı içerisinde adım motorunun bulunduğu konumu belirtmek için tanımlanmış olan 'GercekAdim' değişkeni kullanılır. Bir atış yapmak için istenen yüksekliğe ulaşılması veya atış yapıldıktan sonra geri dönülmesi sırasında her adımda 'GercekAdim' değişkeni artırılır veya azaltılır. Yani 'GercekAdim' değişkeninin değeri bulunulan konumu belirtir. Bu değişkenden faydalanarak adım motoru istenilen yönde ve adım sayısında hareket ettirilir.

Şekil 3.16'da verilen algoritma adım motorunun döndürülmesinde takip edilen algoritmayı göstermektedir. Her bir adım atma işleminde 'GercekAdim' değişkeninin dörde bölümünden kalan sayı kullanılarak adım atma işlemi için gereken ifade gönderilir. 'GercekAdim' değişkeninin artması sırasında 0,1,2,3, azalması sırasında ise 3,2,1,0 şeklinde bir değişim elde edileceği için adım motorunun iki yöne de dönmesi sağlanmış olur.



Şekil 3.16 Adım motorunun döndürülmesinde kullanılan algoritma

Adım motorunu kontrol eden 'Adim_At' ve 'Port_Out_Adim' alt programları Şekil 3.17'de görülmektedir. 'Adim_At' alt programı 'GercekAdim' değişkenini kullanarak, çıkışa gönderilecek bilgiyi belirler. 'Port_Out_Adim' alt programı ise belirlenen bu bilgiyi, assembler komutları kullanarak paralel port çıkışlarına aktarır.

```
procedure Port_Cikis_Adim(Step:byte);
begin
     asm
     push dx
     mov dx,0378h
     mov al,Step
     out dx,al
     pop
     end;
end;

procedure Adim_At;
begin

case (GercekAdim mod 4) of
      0: begin Port_ Cikis _Adim(5) ;end;
      1: begin Port_ Cikis _Adim(9) ;end;
      2: begin Port_ Cikis _Adim(10);end;
      3: begin Port_ Cikis _Adim(6) ;end;
      end;

end;
```

Şekil 3.17 Adım motorunu kontrol eden alt programlar

'Adim_At' alt programı içinde, adım motorunun döndürülmesi için gerekli olan dört farklı sayı değeri, 'Port_Cikis_Adim' alt programına gönderilecek biçimde 'case...of' yapısı içinde adım motorunun dönüşünü sağlayacak şekilde sıralanmıştır. Adım motorunun konumu 'GercekAdim' değişkeni ile belirleneceği için 'case...of' yapısında seçme değişkeni olarak kullanılır. Şekil 3.16 da, bu işlemin prensip algoritması verilmiştir. Seçme işlemi sırayla porta gönderilecek dört sayı arasında yapılacağından, bu işlem için 0,1,2,3 olmak üzere dört etiket belirtilmiştir. Bunun için 'GercekAdim' değişkeninin 4'e bölümünden kalan sayı seçme değişkeni olarak kullanılır. Böylece, 'GercekAdim' değişkeninin bulunduğu adım değerine göre 0-3 arası bir sayı elde edilerek, paralel porta gitmesi gereken değer 'Port_Cikis_Adim' alt programı kullanılarak gönderilir.

Tablo 3.1'te örnek olarak 119-126 adımları arası değişkenlerin durumu gösterilmiştir. Görüldüğü gibi, adım değerini belirten 'GercekAdim' değişkeninin bulunduğu değer 4'e bölünerek seçme bilgisi elde edilmiş ve bu bilgiyle seçilen değer porta gönderilmiştir. Burada, motor sürücü çıkışı terslenmiş olarak elde edildiği için, adım motorunu döndürmek için gerekli sayılar terslenmiş olarak belirtilmiştir.

Adım Değeri GercekAdim	Dörde Bölümden Kalan GercekAdim mod 4	Paralel Porta Gönderilen Bilgi		Motor Sürücü Çıkışı (Terslenmiş) D C B A
		Onluk	İkilik	
119	3	6	00000110	1 0 01
120	0	5	00000101	1 0 1 0
121	1	9	00001001	0 1 10
122	2	10	00001010	0 1 0 1
123	3	6	00000110	1 0 0 1
124	0	5	00000101	1 0 1 0
125	1	9	00001001	0 1 1 0
126	2	10	00001010	0 1 0 1

Tablo 3.1 119-126 adımları arasında adım motoruna gönderilen sinyaller

3.4.3. Yükseklik seviyesinin belirlenmesi

Adım motorunun istenilen yükseklik değerine ulaşması için belirtilen yükseklik, mekanik sistemin bir parçası olan sonsuz dişlinin belirlediği uzunluk sınırının altında olmalıdır. Örneğin, sonsuz dişlinin uzunluğu 150 cm ise, maksimum yükseklik değeri bu değer ile sınırlanır.

Adım motoru istenen yüksekliğe ulaşırken adım hesabını kullanır. Bunun için maksimum adım sayısı ve gidilecek yükseklik değeri hesaba katılır. Burada oluşacak hatanın minimum olması, maksimum yükseklik seviyesi için gereken adım sayısının doğru olarak belirtilmesine bağlıdır.

Atışların gerçekleştirilmesi sırasında her yükseklik seviyesi için atılacak adım sayısı denklem (3.1) de belirtildiği gibi belirlenir.

$$AS = MAS \times \frac{GY}{MY} \qquad (3.1)$$

AS : İstenen yükseklik seviyesine ulaşmak için gereken adım sayısını,

MAS: Gidilebilecek maksimum adım sayısını,

GY : Gidilecek yükseklik seviyesi,

MY : Gidilebilecek maksimum yüksekliği belirtir.

Program içerisinde, yükseklik değerine bağlı olarak atılacak adım sayısı belirlendikten sonra yapılan işlem; bilyenin bulunduğu konumdan alınması ve belirtilen yükseklik değerine götürülmesidir. Burada dikkat edilmesi gereken önemli bir nokta, bilye başlangıç konumundan alındıktan sonra, istenilen yükseklik seviyesine ulaşması sırasındaki adım sayma işlemine algılama çizgisini geçtiği andan itibaren başlanmasıdır.

Şekil 3.18 de, adım motorunu, istenilen yükseklik seviyesine ulaşmak için kontrol eden alt program görülmektedir. Yükseklik belirleme işlemi için adım motorunun bulunduğu konum ve ulaşılacak konumun bilinmesi gerekir. Ulaşılacak konum, atış sırasına göre daha önce girilmiş olan yükseklik listesinden belirlenir. Ancak yükseklik değerini, adım sayısı olarak belirtmek gerekir. Bunun için Hedef_Adim fonksiyonu kullanılmıştır.

```
//Genel olarak tanımlanmış sabitler
const
  Uzunluk : Word = 150;//Yükselik
  SonAdim : Word = 1590;// en yüksek adım sayısı değeri

.............

//Bulunulan yükseklik
procedure Konum;
begin
SimdikiYukseklik:=Round(Uzunluk * SimdikiAdim / SonAdim);
end;

.............

procedure Konum_Bul(Canvas:TCanvas;STimer1:TSuperTimer);

//Hedefe kadar gidilecek adım sayısı
function   Hedef_Adım( x :Word) :Word;
begin HedefAdım := Round( SonAdim * x /Uzunluk );end;

begin

//Hedefe ulaşıldı mı?
if  SimdikiAdim < Hedef_Adim(HedefYukselik)then
    begin
    Inc(GercekAdim); //Adım değerini artır

    // Bilye başlangıcı geçtiyse yüksekliği artır
    if not(Port_In) then Inc(SimdikiAdim);

    Konum; //Konumu hesapla

    Hız_ayarla(STimer1); //Adım hızını ayarla

    Adim_At; //Adım at

    end
else
    //Bilyeyi bırak
    begin BilyeBirak:=True; KonumBul := False; end;

end;
```

Şekil 3.18. Yükseklik seviyesi belirleme alt programı

Hedef_Adim fonksiyonuyla yürütülen işlem, 3.1 formülü ile verilmiş olan adım hesaplama fonksiyonudur. Burada 'const' sabit tanımlama ifadesiyle, 16 bit uzunlığunda olan işaretsiz 'Word' tipi ile 'Uzunluk' ve 'SonAdim' sabitleri belirtilmiştir. Burada, 'Uzunluk' en büyük yükseklik değerini, 'SonAdim' ise bu değere ulaşmak için gereken adım sayısını belirtmektedir. 'SonAdim' sabitinin gösterdiği değeri bulmak için adım motorunun başlangıçtan son yükseklik noktasına ulaşana kadar attığı adım bir program aracılığıyla sayılır.

Maksimum yükseklik değerinin yükseklik hesaplama 'Hedef_Adim' fonksiyonunda kullanılmasıyla birlikte konum bulma hatası en aza indirilmiş olur. Çünkü ulaşılacak yükseklik değerleri için adım sayısı, bu sabit değerden kullanılarak hesaplanılacaktır. Eğer, örneğin 1cm'lik değer için adım sayısı sabit olarak belirlenip, bu sabite göre yükseklik bulunsaydı, bu sabitin içerdiği hata her 1cm'de katlanarak çoğalacaktı. Dolayısıyla, düşme zamanı ölçümü sonuçlarının içereceği hata daha büyük olacaktı.

Alt program içindeki değişkenlerden 'SimdikiAdım' değişkeni, bilye başlangıç konumundan alınıp algılama çizgisini geçtikten sonra adım sayacı olarak kullanılır. Bu sayacın değeri, ulaşılacak yüksekliğin adım sayısıyla her adım da bir 'if..then' yapısı kullanılarak karşılaştırılır. Karşılaştırılan yükseklik değerinin adım sayısı daha önce de belirtildiği gibi 'Hedef_Adim' fonksiyonuyla bulunur. Programdan da görüldüğü gibi bu fonksiyonun değeri, ulaşılacak yükseklik değerini belirten 'HedefYukseklik' değişkenine bağlı elde edilir.

Adım motoruna gönderilen sinyalleri belirleyen 'GercekAdim' değişkeni, başlangıçtan yükseklik noktasına ulaşılana kadar artırılarak adım motorunun yukarı yönde hareketini sağlar. Aynı şekilde, bilye serbest bırakıldıktan sonra son kaldığı yükseklik değerinden geri doğru azalarak, aşağı yönde hareketi sağlar. Başlangıç noktasına gelindiğinde bu değer '0' olur. Eğe sıfır değilse, konum bulma veya geri dönüşte bir hata oluşmuş demektir. Bu durumda, adım motoru başlangıca ulaştığı halde, konumu belirten 'GercekAdim' değişkeni bunu işaret etmediği için adim motoru dönmeye devam ederek mekanik sisteme zarar verebilir.

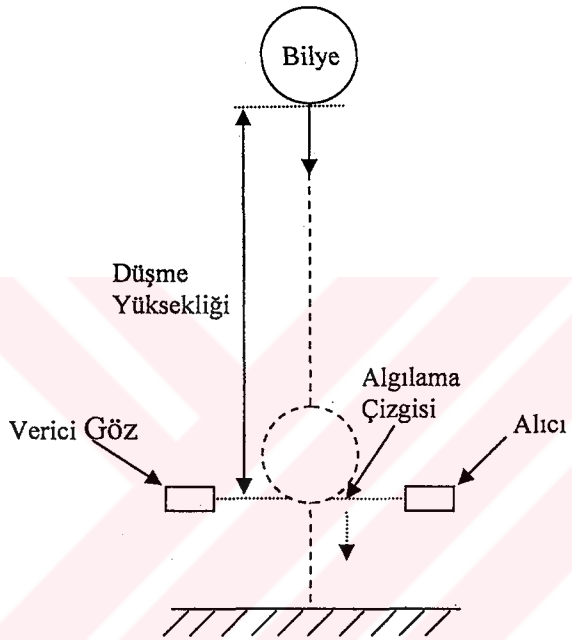
Konum bulma işlemi sırasında, adım motorunun ani kalkış ve durma hareketini önleme amacıyla Şekil 3.19' da verilen hız kontrol alt programı kullanılmıştır. Adım motorunun adım atma hızı zamanlayıcının aralık değerine (Interval) bağlı olduğu için, bu değer değiştirilerek dönüş hızı değiştirilir. Bu işlem, programdan görüldüğü gibi başlangıç ve bitişe yakın yüksekliklerde yapılır.

```
procedure Hız_Kontrol(STimer1: TSuperTimer);
begin

// Başlangıçla algılama çizigisi arasındaysa
if Port_In then STimer1.Interval:=Interval+10

// Yükseklik 5cm'den küçükse
else if  SimdikiYukseklik < 5 then
     STimer1.Interval:=Interval+10-2* SimdikiYukseklik

// Hedef yüksekliğe 5cm'den yakınsa
else if  Abs(SimdikiYukseklik-HedefYukseklik) < 5 then
     STimer1.Interval:=Interval+12-
                  2*Abs(SimdikiYukseklik-HedefYukseklik)

// Diğer durumlarda
else STimer1.Interval:=Interval;
end;
```

Şekil 3.19. Hız kontrolü yapan alt programı

3.4.4. Düşme süresinin ölçülmesi

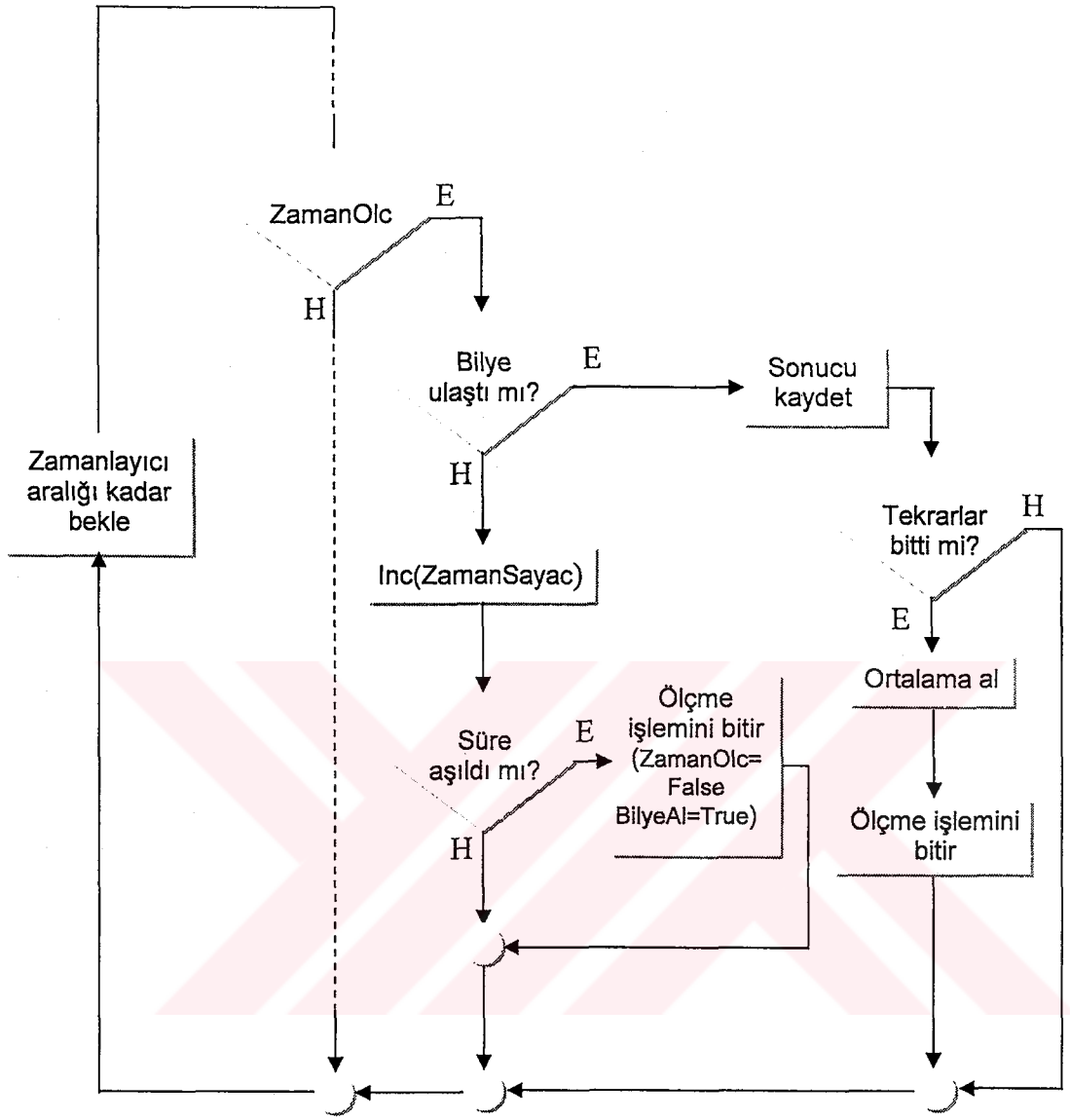
Bilye belirli bir düşme noktasına göre istenen yükseklikten bırakıldığı andan düşme noktasına ulaşana kadar geçen zaman, o yükseklik aralığı için bize düşme zamanını verir. Mekanik düzenek içerisinde, bilyenin düşme anında algılanma işlemi algılayıcı sistem ile sağlandığından, düşme noktası olarak algılayıcı sistemin bulunduğu nokta kabul edilir.



Şekil 3.20 Bilyenin düşme yüksekliği

Şekil 3.20' de gösterildiği gibi, bilye düşme anında algılayıcı sistemin oluşturduğu algılama çizgisinden geçer. Bilye algılama çizgisine geldiğinde, yani alıcı ve verici göz arasındaki iletişim kesildiği anda, düşme noktasına ulaştığı kabul edilir. Bu nedenle, yükseklik hesaplanırken algılama çizgisinin bulunduğu yer başlangıç noktası olarak alınır.

Şekil 3.21'de bilyenin serbest bırakılmasından itibaren yürütülen alt program verilmiştir. Zaman ölçme programının aktif olması için 'Boolean' olarak belirtilen 'ZamanOlc' değişkeninin 'True' değerini alması gerekir. Böylece Şekil 3.13 deki zamanlayıcı tarafından yürütülen alt programda her zamanlayıcı aralığında bu alt program çağırılarak yürütülür.



Şekil 3.21. Bilyenin bırakılmasıyla birlikte yürütülen algoritma

Bilye bırakıldığı andan itibaren, 1ms olan her zamanlayıcı aralığında bilyenin ulaşıp ulaşmadığı kontrol edilir. Port kullanılarak yapılan kontrol işlemi 'Port_Algila' fonksiyonu ile sağlanır. Bilye algılandıysa 'True', diğer durumda 'False' değeri bu fonksiyona atanır.

Algoritma yürütülürken eğer bilye ulaşmadıysa süreyi tutan değişken olan 'ZamanSayac' değişkeni 'Inc' komutu ile bir artırılır. Dolayısıyla bu 1ms'lik süre geçmesi anlamına gelir. Eğer 500 ms olarak belirlenen süre aşıldıysa atış yapılmadı kabul edilir ve zaman ölçme işlemi iptal edilerek, aynı atışın tekrarlanması için adım motoru başlangıç konumuna getirilir.

```
function Port_Algila:Boolean;
begin
     asm
        push dx
        mov dx,0379h
        in al,dx
        and al,040h
        mov Bit,al
        pop dx
     end;
if Bit=0 then Port_Algila:= True
else Port_Algila := False;
end;
.....................................

procedure Zaman_Olc(STimer1: TSuperTimer;
                    StringGrid1:TStringGrid; Canvas:TCanvas);
label Cikis;
var
   i      : Word;
   Toplam : Real;
Begin

//Bilye algılandıysa süreyi kaydet ve yaz
if Port_ Algila then
   begin
   // Düşme süresini dizi elemanına kaydet
   Sonuc[TekrarNo] := ZamanSayac/10 ;

   // Tekrarlar tamamlandıysa ortalama değeri bul
   if  TekrarNo = Tekrar[YapilanAtis] then
       begin
       Toplam := 0;
       // Toplam değeri bul
       for i:=1 to TekrarNo do
            Toplam := Toplam + Sonuc[i];

       // Toplam değeri tekrar sayısına böl
       OrtAtis [SimdikiAdım]:=Toplam/TekrarNo;
       end;

   Liste(StringGrid1); //Sonucu atış listesine aktar
   ZamanOlc:= False;
   end;
Inc(ZamanSayac);// süreyi 1 ms artır

// Atış gerçekleşmediyse tekrarla
if ZamanSayac > 500 then
   begin
   ZamanOlc:= False; // Ölçme işlemini bitir
   BilyeAl:=True; // Başlangıç konumuna git
   Display(Canvas,3);//Atış gerçekleşmedi uyarısı ver
   end;

Cikis:
end;
```

Şekil 3.22. Düşme zamanını ölçen alt program

Eğer bilye algılandıysa kontrol aralıklarını sayan 'ZamanSayac' değişkeninin değeri sonuç olarak atış sırasına göre dizi elemanına kaydedilir. Tekrar sayısı birden fazla ise diğer tekrarlar tamamlandıktan sonra bunların ortalaması alınıp sonuç olarak kaydedilir. Kaydedilen sonuç değerler listeye yazdırılarak ölçme işlemi sonlandırılır. Şekil 3.22' de yürütülen algoritmanın program kodları verilmiştir.

3.4.5. Hareketli sistemin başlangıç konuma Alınması

Hareketli sistemin tekrar başlangıç konumuna alınması için Şekil 3.23'de görülen alt program yürütülür. Bilyenin alınmasından istenen yükseklik noktasına ulaşmasına kadar her adımda bir artırılarak adım motorunu kontrol eden 'GercekAdim' değişkeninin, kaldığı sayıdan itibaren her adımda bir azaltılmasıyla adım motoru başlangıç noktasına doğru hareket ettirilir. Bu işlem azaltılan değişken sıfırdan büyük olduğu sürece devam eder. Daha sonra adım motorunun enerjisi kesilerek, bilyenin alınması için mıknatıs enerjilendirilir.

```
procedure Bilye_Al(STimer1: TSuperTimer;Canvas:TCanvas);
begin
// Başlangıç konumuna ulaşılmadıysa bir adım aşağı git
if GercekAdim > 0 then
   begin
   Dec(GercekAdim); //Bir adım azalt

   if not(Port_Algila) then Dec(SimdikiAdım);
   Hiz_Ayarla(STimer1);//Konuma göre hız belirle
   Adim_At;// Bir adım at
   end

else
   begin

   Port_Cikis_Adim(0); // Adım motorunun enerjisini kes
   Port_Cikis_Miknatis(1); // Elektromıknatısı enerjilendir
   Bekle:=True; // Bekleme konumunu ayarla
   BilyeAl := False; // Mevcut konumu iptal et
   STimer1.Interval:=Interval; // Zamanlayıcı aralığını kur

   // Atış gerçekleşmemişse bekle
   if ZamanSayac>500 then STimer1.Enabled:=False
   else Display(Canvas,2);
   end;
end;
```

Şekil 3.23. Bilyenin alınmasında kullanılan alt program

3.5. Sonuçların Kaydedilmesi

Serbest düşme deneyinin gerçekleştirilmesi sonunda elde edilen düşme zamanı değerleri, yeni bir atış listesi oluşturulana kadar ve program çalıştığı sürece tanımlanmış olan sonuç dizisinde saklanır. Sonuçları istendiği zaman tekrar grafik veya liste olarak inceleyebilmek için sonuçların kaydedilmesi ihtiyacı doğar. Bunun için bir kayıt dosyası oluşturulur ve okuma yazma işlemleri bu dosya üzerinde yapılır. Kayıt işlemlerinde önceki bölümde bahsedilen dosyalama komutları kullanılır.

3.5.1 Değişken ve tip tanımlamaları

```
      .
      .
// Kayıtla ilgili tip tanımlamaları
type
    Kayitlar=Record
            Isim      : String[10];
            Uzunluk   : Byte;
            Yukseklik : Array[1..150] of Byte;
            Sonuc     : Array[1..150] of Real;
    end;

//genel değişken tanımlamaları
var
  Form1 : TForm1;
  Atis  : Kayitlar;
  Kayit : File of Kayitlar;

//diğer tanımlamalar
      .
      .
procedure TForm1.FormCreate(Sender: TObject);
begin

// kayitlar için dosya ata
AssignFile(Kayit,'c:\Bilgi\Bilgi.dat');
{$I-} Reset(Kayit); {$I+}
if IOResult<>0 then ReWrite(Kayit);

//diğer işlemler
      .
      .
end;
```

Şekil 3.24. Kayıt işlemleri için dosya atanması

Şekil 3.24 de kayıt işlemi için tanımlanan tip ve değişkenler ve dosya atanması ile ilgili program kodları görülmektedir. Yükseklik ve sonuç değerlerini saklamak için değikenlerden ikisi Sonuc ve Yukseklik olmak üzere 150 elemanlı dizi olarak tanımlanmıştır. Yukseklik dizisi, yükseklik değerleri tamsayı değerlerden oluştuğu için 'Integer' olarak, 'Sonuc' dizisi ondalıklı değerleri saklayacağı için 'Real' olarak tanımlanmıştır. Bu dizilere kaydedilen eleman sayısını saklamak için 'Byte' olarak Uzunluk değişkeni tanımlanmış Yapılan kayıtlara bir isimlendirmek için 10 karakterlik bir 'Isim' değişkeni 'String' olarak tanımlanmıştır. Sözü geçen bu dört değişken 'Kayitlar' adı altında olmak üzere tip olarak tanımlanmıştır. Yine oluşturulan bu tip kullanılarak 'Atis' değişkeni tanımlanmıştır. Dolayısıyla; 'Atis.Isim', 'Atis.Uzunluk', 'Atis.Yukseklik' ve 'Atis.Sonuc' olarak kayıt işlemlerinde kullanılmak üzere değişkenler atanmıştır.

Kayıt için dosya atama işlemi programın çalıştırılmasıyla birlikte gerçekleşir. Kayıt dosyasının yeri ve adı 'AssignFile(Kayit, 'c:\bilgi\bilgi.dat')' olarak belirtilmiştir. Burada 'Kayit', 'Kayitlar' tipine bağlı olarak oluşturulmuş bir dosya değişkenidir.

3.5.2. Kayıt işlemi

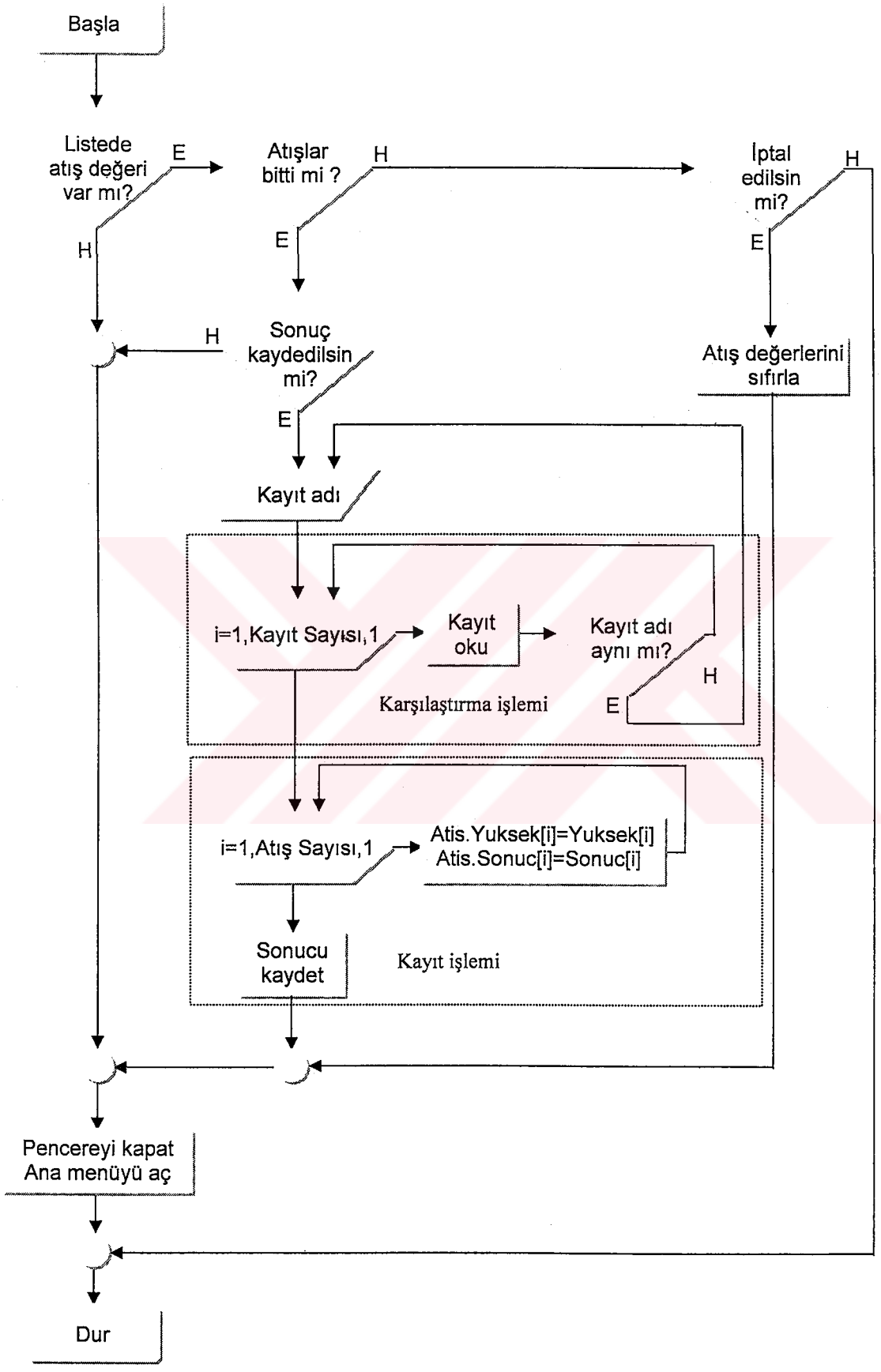
Kayıt işlemi, belirtilen atışlar yapılarak serbest düşme deneyi tamamlandığında, pencere kapatılırken gerçekleştirilir. Kaydedilen değerler atış yükseklikleri ve bunlara ait düşme zamanı değerleridir. Ayrıca bu değerlerin oluşturduğu dizilerin uzunluğunu belirten değer kaydedilir.

Şekil 3.25'de atışların gerçekleştirildiği pencerenin kapatılmasıyla birlikte yürütülen algoritma verilmiştir. İlk önce atış listesinde yükseklik değeri olup olmadığı kontrol edilir. Yükseklik değeri olmaması durumunda herhangi bir işlem yapılmadan form kapatılarak ana pencereye dönülür.

Eğer listeden bir veya daha fazla yükseklik değeri varsa bu durumda, atış işlemleri bitmemiş ise, iptal edilip edilmeyeceği sorulur. İptal edilirse, atış değerleri sıfırlanarak pencereden çıkılır. Diğer durumda herhangi bir işlem yapılmadan algoritmadan sonlanarak, pencereyi kapatma işlemi iptal edilmiş olur.

Listede yükseklik değerleri varsa ve bunlarla ilgili atışlar bitmişse bu durumda 'kayıt yapılsın mı?' diye sorulur. Kayıt yapılmazsa pencere kapatılır ve ana pencere açılır. Eğer kayıt yapılırsa bu durumda kayıt adı girilmesi için çıkan pencereye kayıt adı girilir. Bundan sonraki adım aynı isimde kayıt olup olmadığının kontrolüdür. Girilen isimle dosyadaki isim kayıtları karşılaştırılır. Bunun için 'for...do' döngüsü kullanılır. Bu döngünün üst sınırı kayıt dosyasındaki kayıt sayısı kadardır.

Eğer aynı isimde kayıt varsa yeni bir isim girişi yapılır. Yoksa, yükseklik ve düşme zamanı değerleri geçici olarak kaydedilmiş oldukları diziden, kayıt tipi diziye 'for...do' döngüsüyle aktarılır. Buradaysa döngünün üst sınırını listedeki yükseklik değerleri sayısı belirler. Aktarma işlemi bittiğinde kayıt işlemi gerçekleştirilerek pencere kapatılır.



Şekil 3.25. Atış penceresi kapatılırken yürütülen algoritma

```
procedure TForm7.KapatClick(Sender: TObject);
label Tekrar,Cikis,Kapat;
var
   İsim : String[10];
   k    : Byte;
begin

// Atışlar bitmemişse uyar
if AtisNo = 0 then goto Exit;

else if YapilanAtis <= AtisNo then
   begin
   if messageDlg('Atışlar iptal edilsin mi?',
   mtWarning,[mbYes,mbNo],0)=mrNo then  goto Exit
   else YapilanAtis:=0;
   end
else
    begin
    if MessageDlg('Atışlar kaydedilsin mi?',
    mtWarning,[mbOk],0)=mbNo then goto Kapat;

    Tekrar:
    İsim:=InputBox('Kayıt Adı','Kayıt adını giriniz','adsız');
    //Aynı isim varsa uyar
    if FileSize(Dosya)>0 then
       begin
       for k:=0 to FileSize(Dosya)-1 do
           begin
           Seek(Dosya,k);Read(Dosya,Atis);

           if Atis.İsim:=İsim then//İsimleri karşılaştır
              begin
              MessageDlg('Farklı isimgirin',mtWarning,[mbOk],0);
              goto Tekrar;
              end;
          end;
       end;

       // Kayıt işlemini yap
       Atis.İsim:=İsim;
       Atis.Uzunluk := AtisNo;

       for k:=1 to Atis.Uzunluk do
           begin
           Atis.Yukseklik[k] := Yukseklik[k];
           Atis.Sonuc[k]     := OrtAtis[k];
           end;
           Seek(Dosya,FileSize(Dosya));Write(Dosya,Atis);
       end;
Kapat:
Form1.Show;
Form7.Close;
Cikis:
end;
```

Şekil 3.26. Atış penceresi kapatılırken yürütülen alt program

Yapılmış olan serbest düşme deneylerinin kaydedilmiş sonuçları, kayıt dosyasından okunarak listelenebilir. Bunun için dosyada toplam kaç tane kayıt olduğu okunur ve bu değer sayısınca dosyadan kayıtlar okutularak kayıtların isimleri liste şeklinde 'ListBox' nesnesinde sıralanır. Bu işlemi gerçekleştiren program kodları Şekil 3.27'de verilmiştir.

```
procedure TForm1.KayitListesi1Click(Sender: TObject);
var i:Byte;
begin

Form6.ListBox1.Clear;//Listeyi temizle

//Dosyadaki kayıt sayısınca döğüyü tekrar et
for i:=1 to FileSize(Dosya)-1 do
    begin

    //belirtilen indeksteki kaydı bul ve oku
    Seek(Dosya,i);Read(Dosya,Atis);

    //Kayıt ismini listeye yaz
    Form6.ListBox1.Items.Add(Atis.isim);
    end;

Form6.Show;
Form1.Hide;
end;
```

Şekil 3.27. Kayıt isimlerini listeye yazan program kodları

Listelenen kayıt isimlerinden yükseklik ve düşme zamanı değerlerini görmek için sonuçlar ayrı bir listede tekrar listelenir. Bunun için 'ListBox' kutusunda listelenmiş olan kayıt isimlerinden içeriği görülmek istenenin seçilmesi gerekir. Herhangi bir seçme işlemiyle birlikte, 'ListBox' nesnesinin 'OnClick' özelliğinden faydalanılarak Şekil 3.28'daki alt program yürütülür. Görüldüğü gibi önceki listenin satır uzunluğuna bağlı olarak, listeleme işlemi için kullanılan 'StringGrid' nesnesinin satırları temizlenir. 'ListBox' nesnesindeki kayıt isimlerinin listesi dosyadaki kayıt sırasını belirttiği ve dosyadaki kayıt sayısı kadar olduğu için, seçili satırın no suna bakılarak istenilen kayıt dosyadan okutulur. Seçili satırın kayıt no su, 'ItemIndex' özelliği ile seçili kaydın sırasından belirlenir.

Kayıt listeleme işlemi özetlenirse; formun açılmasıyla birlikte 'ListBox' nesnesinde kayıt isimleri listelenir, buradan istenilen kaydın seçilmesiyle birlikte eski liste silinir, seçilen kaydın bulunduğu sıra no ile kayıt dosyasından listelenmek istenen kayıt okutularak 'StringGrid' üzerinde listelenir.

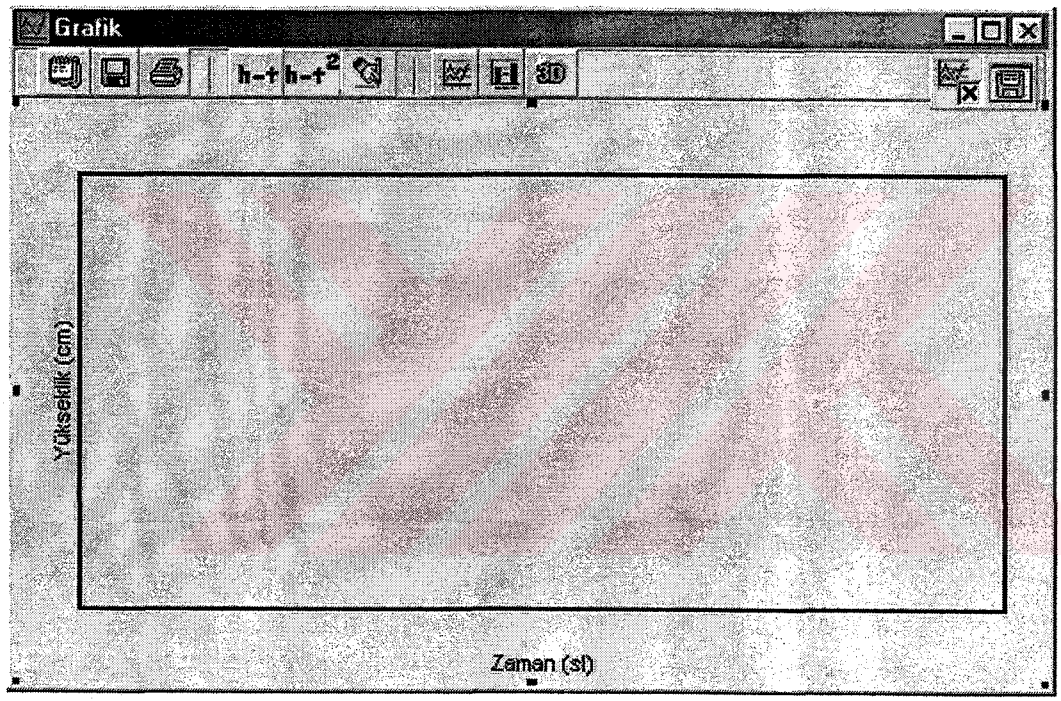
Dosyadaki kaydedilmiş kayıtlar, dizi olarak yükseklik ve düşme zamanı değerlerini ve bu dizilerin uzunluğunu belirten bir tamsayı değer içerir. Bu tamsayı değer, 'Uzunluk' olarak kodlanmış olup, atışlar yapılmadan önce girilmesi gereken yükseklik listesindeki yükseklik değerleri sayısını belirtir. Listeleme işleminde, yükseklik ve düşme zamanı değerlerini yazdırmak için kullanılan döngüde sınır değeri olarak bu uzunluk değeri kullanılır.

```
procedure TForm6.ListBox1Click(Sender: TObject);
var i:Byte;

begin
//Listeyi temizle
for i:=1 to S do
    begin
    StringGrid1.Cells[0,i]:='';
    StringGrid1.Cells[1,i]:='';
    StringGrid1.Cells[2,i]:='';
    end;

// Listede seçili olan kaydı oku
Seek(Folder,Listbox1.ItemIndex);Read(Dosya,Atis);

if Atis.Uzunluk>9 then StringGrid1.RowCount:=Atis.Uzunluk;

S:= Atis.Uzunluk;
for i:=1 to Atis.Uzunluk do
    begin
    StringGrid1.Cells[0,i]:= IntToStr(i);
    StringGrid1.Cells[1,i]:= IntToStr(Atis.Yukseklik[i]);
    StringGrid1.Cells[2,i]:= FloatToStr(Atis.Sonuc[i]);
    end;

end;
```

Şekil 3.28. Yükseklik ve düşme zamanı değerlerinin listelenmesi

3.6. Atışların Grafiğinin Çizdirilmesi

Yapılan serbest düşme deneyi sonunda elde edilen sayısal değerleri grafik olarak görüntülemek, deneyin sonuçlarının yorumlanmasına görsel olarak katkıda bulunur. Bu işlemi gerçekleştirmek için ayrı bir form kullanılarak, form üzerine Şekil 3.29.'da görüldüğü gibi grafik çizdirme işlemlerinde kullanılan 'Chart' nesnesi yerleştirilmiştir. 'Chart' nesnesini kontrol etmek için 'ChartView' isimli ayrıca bir bileşen kullanılmıştır [12].



Şekil 3.29. Grafik Formu

Panel üzerine çeşitli işlevleri yerine getiren butonlar yerleştirilmiştir. Bunlardan üzerinde liste resmi olan, mevcut kayıtları listeleyerek hangi kaydın grafiği çizilecekse seçim işlemi yapılır. Bu butona basıldığında özel bir bileşen olan 'ChartView' nesnesinin 'Execute' özelliği aktif olarak kayıtları listeler.

Çizdirilen grafiğin kaydedilmesi için 'SaveDialog', yazdırma işlemi için 'PrintDialog' bileşeni kullanılır.

Yükseklik zaman grafiği çizdirilirken, seçilmiş olan kayıt adına ait yükseklik ve düşme zamanı değerleri okutularak diziye aktarılır. Grafiğe dizi aktarma işlemi Şekil 3.30'daki genel grafik çizdirme yapısını oluşturan alt programla gerçekleştirilir. Bu alt programla, kayıt listesindeki bütün kayıtlar grafiğe aktarılır. Çizdirilmek istenen kayıtların aktarılabilmesi için grafik seçimi sırasında seçilmiş olan grafiklerin kayıt numaraları bir diziye aktarılarak, grafik çizimi yapılırken bu diziye göz önünde bulundurulur.

```
...
var
   Dizi1 : TChartSeries;
...


procedure Grafik_Ciz(ChartView1: TChartView;Chart1: TChart);
var i,j:Byte;

begin

// Grafiği temizle
ChartView1.MasterChart.SeriesList.Clear;

for i:=0 to (FileSize(Dosya)-1) do

    begin

    //Grafik dizisi oluştur
    Dizi1 := TLineSeries.Create(Chart1);

    // Kaydı oku
    Seek(Dosya,i);Read(Dosya,Atis);

    //Kayıt uzunluğu kadar tekrarla
    for j:=1 to Atis.Uzunluk do

    //Yükseklik ve zaman değerini diziye ekle
    Dizi1.AddXY(Atis.Sonuc[j],Atis.Yukseklik[j],
                        FloatToStr(Atis.Sonuc[j],clBlue);

    //Diziyi grafiğe ekle
    Chart1.AddSeries(Dizi1);
    end;

end;
```

Şekil 3.30. Grafik çiziminde kullanılan temel alt program

3.7. Sonuçlar

Bu bölümde kullanıcı ara yüzünün 'Delphi' görsel programlama dili ile tasarımı genel hatlarıyla açıklanmıştır. Serbest düşme deneyinin gerçekleştirilmesinde izlenen işlem basamakları belirlenerek bunları gerçekleştirmek için tasarlanan algoritmalar program kodları olarak ifade edilmiştir. Kullanıcı ara yüzünün ana penceresine yerleştirilen işlem seçenekleri, yükseklik girme, atışları gerçekleştirme, sonuçları görme ve diğer yardımcı pencerelere ulaşma imkanı sağlar.

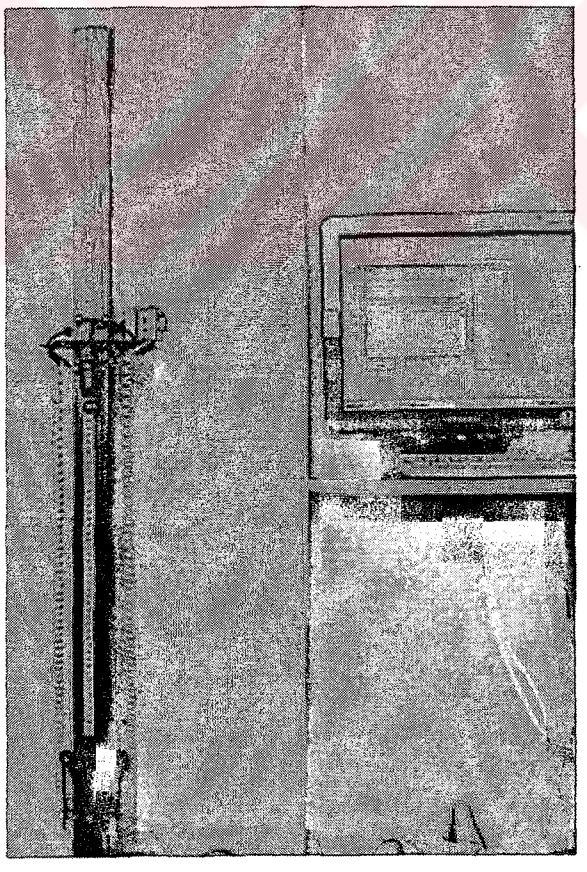
Deneyin gerçekleştirilmesi için atış yapılacak yükseklik değerleri ve tekrar sayıları liste olarak kullanıcı tarafından girilir. Program, bir atış için gereken işlemleri her bir yükseklik için ayrı ayrı gerçekleştirilir. Atışlar tamamlandığında elde edilen sonuçlar ve gerçekleştirdiği yükseklik değerleri dizi olarak kaydedilir. Bu diziler istendiğinde tekrar kayıt dosyasından okutulup liste şeklinde veya grafiksel olarak incelenebilir.

Kullanıcı ara yüzü serbest düşme deneylerini gerçek paralel portu kullanarak elektromekanik sistemle haberleşir. Program içerisinde her bir elemanın paralel port üzerinde bağlanmış olduğu çıkış bitleri ile kontrol işlemleri gerçekleştirilir. Bir serbest düşme deneyi gerçekleştirilirken bilyenin başlangıç konumundan alınması, program içinde elektromıknatısı ifade eden bit içeriğinin elektromıknatısı aktif edecek lojik konuma alınmasıdır. Bilyenin başlangıç konumundan alındıktan sonra istenen yükseklik seviyesine çıkartılmasında kullanılan adım motorunun kontrolü, bu motorun döndürülmesi için gerekli sinyali belirten dört bitlik verinin uygun şekilde program tarafından değiştirilmesiyle sağlanır. Bilyenin düşme noktasında algılanması ise, port üzerinde belirlenmiş olan giriş bitine, algılayıcı sistemden gelen ve bilyenin düşme noktasında olup olmadığını belirten tek bitlik veri ile sağlanır. Bilyenin bırakılmasıyla birlikte, program algoritması tarafından bu algılama sinyalini 1 ms'lik eşit zaman aralıklarıyla kontrol edilir. Her kontrolde zamanı belirten sayaç program tarafından bir tarafından bir artırılır. Böylece düşme zamanına kadar yapılan kontrol sayısı sayaca kaydedilerek, mili saniye cinsinden düşme zamanını verir.

BÖLÜM 4. KULLANICI ARA YÜZÜNÜN TANITIMI

4.1. Giriş

Kullanıcı ara yüzü, serbest düşme deneyini gerçekleştirmede kullanıcının gerekli kontrol işlemlerini yapabilmesi için dizayn edilmiş görsel bir bilgisayar programıdır. Şekil 4.1.'den görüldüğü gibi, mekanik ve elektronik parçalardan oluşan atış düzeneği, bu ara yüz tarafından kontrol edilir. Bu bölümde, serbest düşme deneyini sistematik hale getirerek sonuçların kullanıcı tarafından değerlendirilmesini kolaylaştıracak kullanıcı ara yüzünün kullanımına yönelik açıklamalar verilecektir.



Şekil 4.1. Serbest düşme deney setinin görünümü

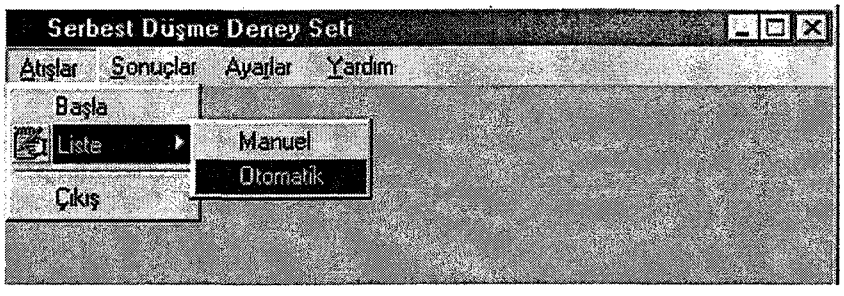
4.2. Ana Pencere

Kullanıcı ara yüzünün kullanımına yönelik işlemleri gerçekleştirmesi için seçeneklerin sunulduğu penceredir. Şekil 4.2.' de görüldüğü gibi ana pencere üzerinde, Atışlar, Sonuçlar, Ayarlar ve Yardım olmak üzere dört ana seçenek verilmiştir.

Şekil 4.2. Ana Pencere

4.3. Yükseklik Girişi Pencereleri

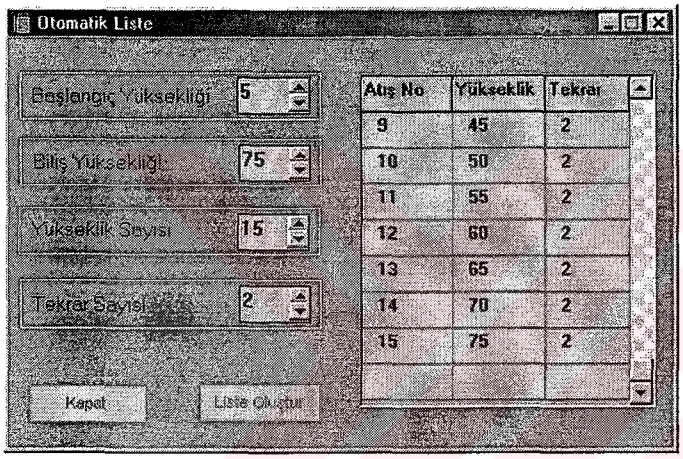
Serbest düşme deneyinin hangi yükseklik değerlerinde kaç kez yapılacağını belirtmek için kullanılan pencerelerdir. Aynı yükseklik değerinde birden fazla atış yapılmasıyla elde edilecek sonuç daha güvenli olacağı için tekrar sayısı girişi için bir seçenek de sunulmuştur. Şekil 4.3.' de görüldüğü gibi yükseklik listesi, 'Otomatik' ve 'Manuel' olmak üzere iki şekilde oluşturulabilir.



Şekil 4.3. Liste Alt Seçenekleri

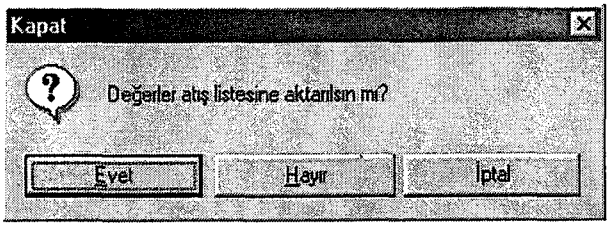
4.3.1. Otomatik yükseklik girişi penceresi

Otomatik liste belirtilen yükseklik aralığında, belirtilen sayıda yükseklik değerinden , otomatik olarak liste oluşturur. Şekil 4.4' de otomatik liste oluşturmak için kullanılan pencere ve oluşturulmuş örnek bir liste görülmektedir. Başlangıç değeri 5 cm, bitiş değeri 75 cm, bu aralıktaki yükseklik sayısı 15 ve her yükseklikteki tekrar sayısı 2 olarak belirtildikten sonra 'Liste Oluştur' butonu ile yükseklik değerleri liste olarak sunulur. Eğer yeni değerler belirtilip 'Liste Oluştur' butonuna basılırsa eski değerler iptal olarak yeni değerler geçerli olacaktır.



Şekil 4.4. Otomatik liste penceresi

Liste oluşturulduğunda listelenen yükseklik değerlerinin atışlarda kullanılması için pencere kapatılırken onay verilir. Şekil 4.5' de onay işlemi sırasında aktif olan pencere verilmiştir. Evet seçeneğiyle yükseklik değerleri atışlarda kullanılmak üzere atış penceresindeki listeye aktarılır. Hayır seçilirse herhangi bir işlem yapılmadan ana pencereye geçilir. İptal ile pencereyi kapatma işlemi iptal edilmiş olur. Bu işlem manuel listenin kapatılmasında da geçerlidir.



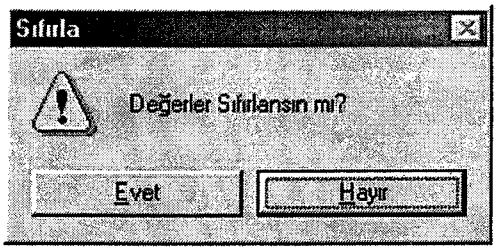
Şekil 4.5. Liste onay mesajı

4.3.2. Manuel yükseklik girişi penceresi

Şekil 4.6' de görülen, yükseklik girişinin her yükseklik için ayrı yapıldığı penceredir. Yükseklik değerlerinin sıralandığı listenin ilk satırında yükseklik ve tekrar değerlerini belirtmek için ayrı iki pencere kullanılmıştır.

Şekil 4.6. Manuel liste oluşturma

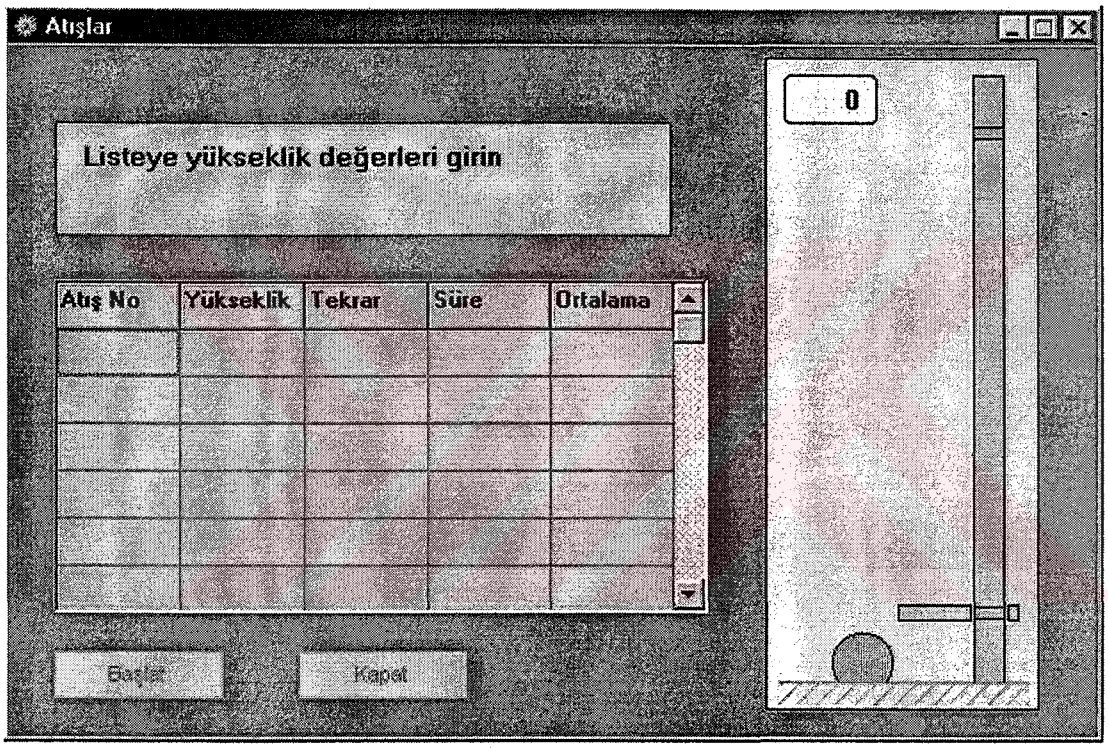
Pencere üzerinde 'Onay', 'Sıfırla' ve 'Kapat' olmak üzere butonlar bulunmaktadır. 'Onay' butonuyla, belirtilen değerler listenin 'Atış No' ile belirtilen elemanına kaydedilir. 'Sıfırla' butonu ile listedeki değerler iptal edilip yeni değerler girilmek istendiği zaman liste sıfırlanabilir. Butona basıldığında Şekil 4.7.'da ki uyarı mesajında 'Evet' seçilirse verilirse, sıfırlama işlemi yapılır, 'Hayır' ile seçilirse işlem iptal edilir ve liste aynı durumda kalır.



Şekil 4.7. Liste sıfırlama işlemi uyarı mesajı

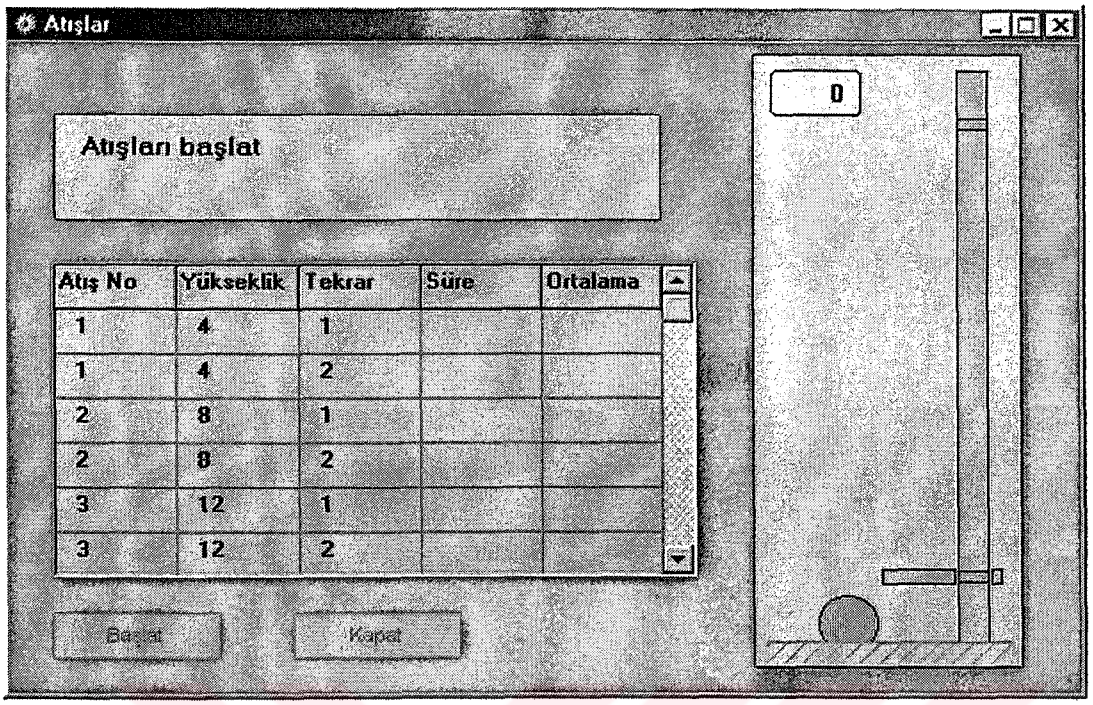
4.4. Atış Penceresi

Şekil 4.8'de görülen serbest düşme deneyinin gerçekleştirildiği penceredir. Şekil 4.3. de verilen şekilde atışlar seçeneğinin altında bulunan 'Başlat' seçeneğiyle bu pencereye ulaşılır. Pencere üzerinde yükseklik ve düşme zamanı değerlerinin yazıldığı bir liste, atış işlemlerinin durumunu gösteren bir ekran ve atış sistemin durumunu belirten bir animasyon bulunmaktadır.



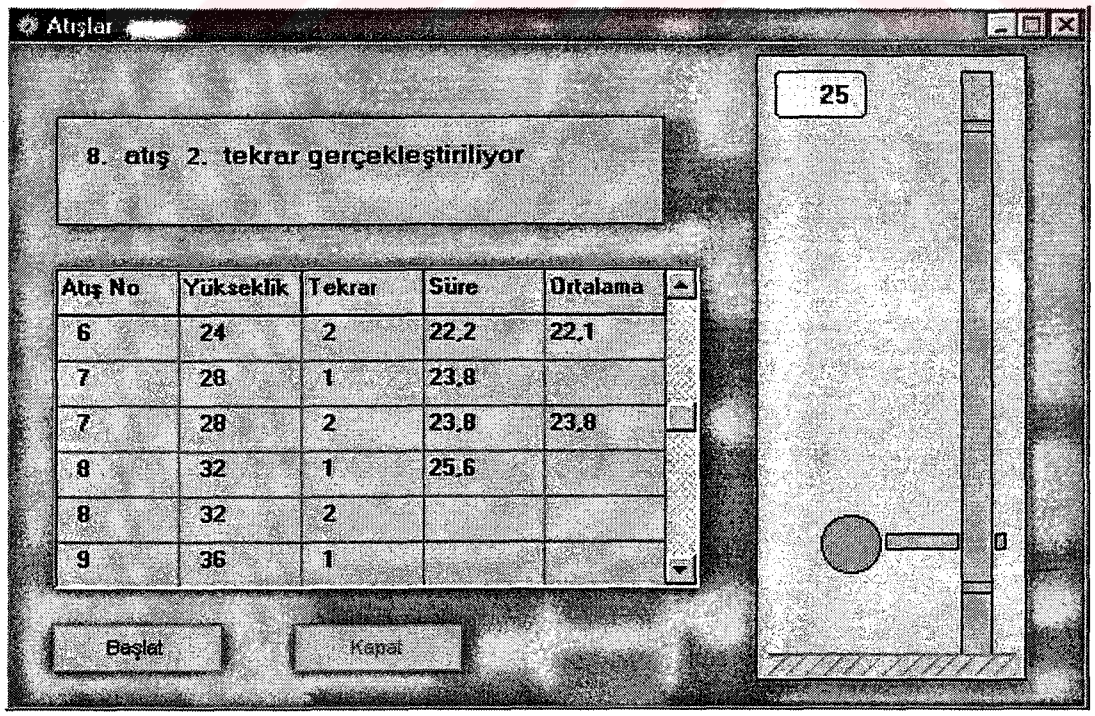
Şekil 4.8. Atış penceresi liste uyarısı

Eğer 'Manuel' veya 'Otomatik' liste oluşturulmadıysa herhangi bir işlem gerçekleştirilmez. Sadece listeye yükseklik değerleri girilmesi uyarısı verilir.Yükseklik değerleri girildikten sonra atış listesi açıldığında Şekil 4.9' da görüldüğü gibi atışlar başlatılmaya hazır konuma gelir. Oluşturulan listede girilen her yükseklik değeri, tekrar sayısınca atış yapılmak üzere listede alt alta sıralanır.



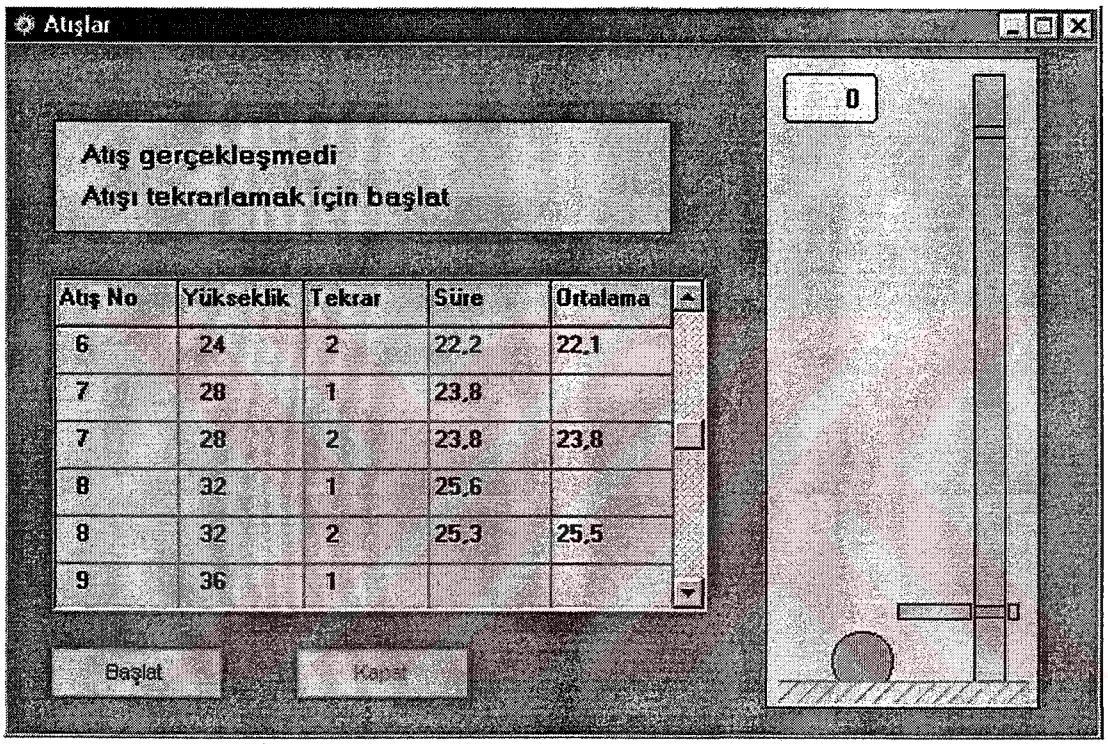
Şekil 4.9. Atışlar penceresi başlama konumu

Atışlar 'Başlat' butonuna basıldığında Şekil 4.10'daki gibi atış numarasına göre sırayla gerçekleştirilir.



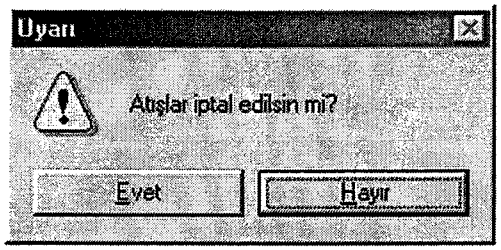
Şekil 4.10. Atışların gerçekleştirilmesi

Atış işlemleri, mekanik-elektronik atış sistemi bilgisayarın paralel portu aracılığıyla kontrol edilerek yapılır. Dolayısıyla bağlantının kopması, atış sisteminde oluşabilecek bir arıza veya atış sisteminin enerjisi sağlanmamışsa atışlar gerçekleşmeyecektir. Bu durumda ara yüz üzerindeki durum penceresinde Şekil 4.11'de görülen uyarı verilecektir. Aynı atışın tekrarlanması için sistem çalışır duruma getirildikten sonra tekrar 'Başlat' butonuna basılması gerekir.

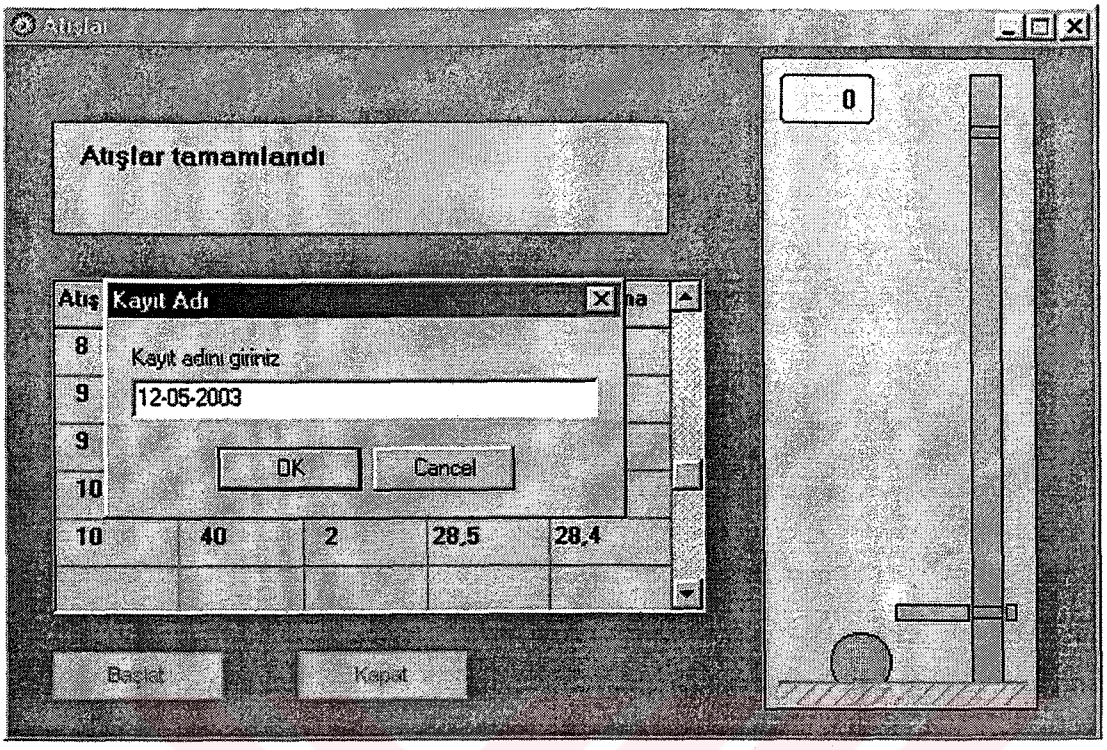


Şekil 4.11. Atışın gerçekleşmemesi durumu

Atışların gerçekleştirilmesi sırasında pencere kapatılırsa Şekil 4.12'de verilen uyarı mesajı çıkar. Evet seçilirse atışlar iptal edilerek pencere kapatılır. Hayır seçilirse pencere mevcut durumunu korur.



Şekil 4.12. Atışlar bitmedi ise pencerenin kapatılması

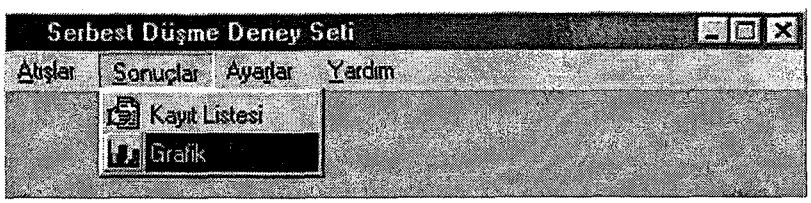


Şekil 4.13. Atışların tamamlanması

Eğer atışlar tamamlandıysa atış sonuçlarının kaydı için Şekil 4.13' de görülen pencere çıkar. Kayıt adı harf, rakam veya diğer semboller olabilir. Girilen kayıt adı daha önceki kayıtlarda varsa, farklı isim girilmesi için uyarı mesajı verilir. Kayıt yapılması istenmezse 'Cancel' butonuyla kayıt işlemi yapılmadan pencere kapatılır.

4.5. Sonuç Pencereleri

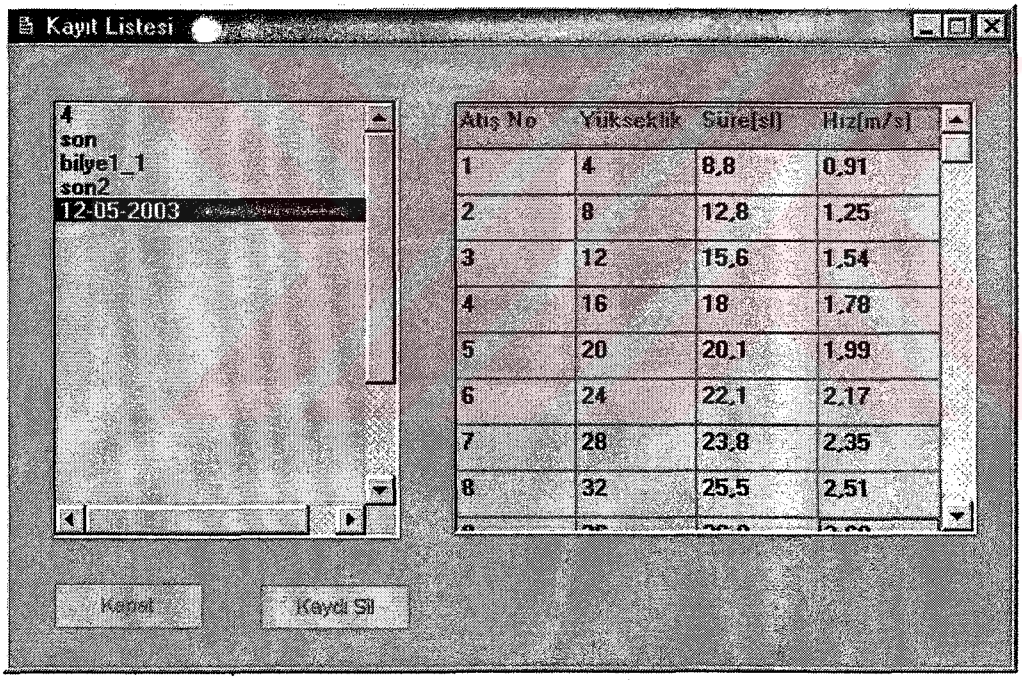
Atış işlemleri sonucunda kaydedilmiş olan sonuç değerlerin Şekil 4.14'de görüldüğü gibi liste ve grafik olmak üzere görüntülendiği pencerelerdir.



Şekil 4.14. Sonuçlar seçeneği

4.5.1. Kayıt listesi

Şekil 4.15'de görülen kayıtların listelendiği penceredir. Gerçekleştirilmiş olan kayıtlar, eğer kaydedildi ise kayıt listesinde kayıt adı yer alır. Mouse ile seçilen kayıt liste yanındaki pencerede kayıt adına ait değerler liste olarak verilir. Değerlerin listelendiği pencerede, atış sırasını belirten 'Atış No', atışın yüksekliğini belirten 'Yükseklik', bu yüksekliğe ait düşme zamanını salise cinsinden gösteren 'Süre' ve bilyenin düşme noktasında ulaşmış olduğu hızı gösteren 'Hız' etiketleri bulunmaktadır. Buradaki hız değeri atış yüksekliği ve düşme zamanı değerlerinden formül yoluyla hesaplanır.



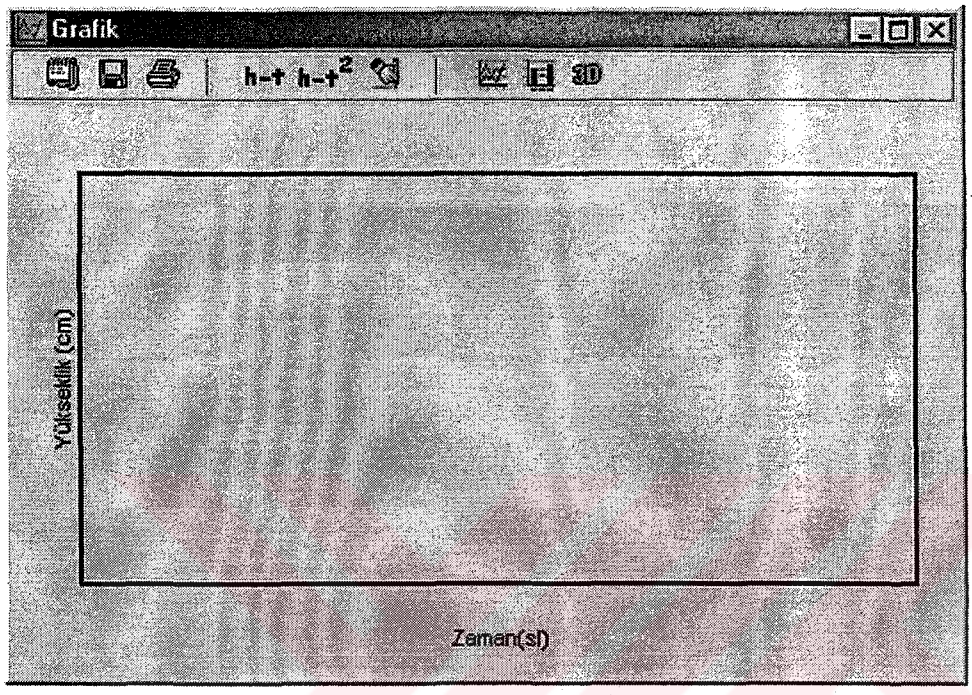
Atış No	Yükseklik	Süre[sl]	Hız[m/s]
1	4	8,8	0,91
2	8	12,8	1,25
3	12	15,6	1,54
4	16	18	1,78
5	20	20,1	1,99
6	24	22,1	2,17
7	28	23,8	2,35
8	32	25,5	2,51

Şekil 4.15. Kayıt listeleme penceresi

4.5.2. Grafik

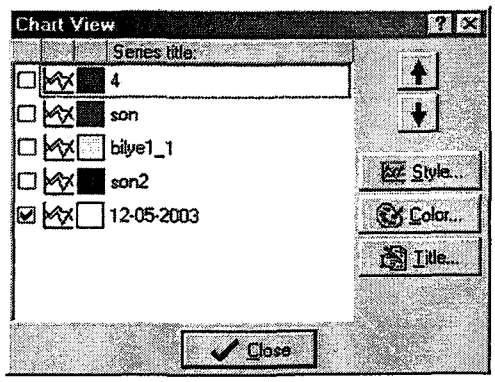
Sonuç değerlerin grafik olarak sunulduğu, programın en önemli penceresidir. Serbest düşme deneyi sonunda her bir yükseklik değerine karşı bir düşme zamanı elde edilir. Grafik, yükseklik seviyesi ve ona ait düşme zamanı değerleri kullanılarak

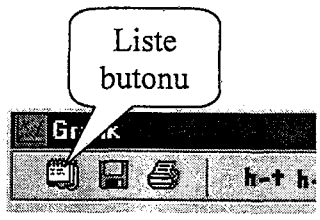
çizilir. Yükseklik-zaman grafiğinden başka hız-zaman grafiği de çizilebilir. Bilyenin düşme anındaki hızı yükseklik ve zaman değerinden hesaplanabilir. Şekil 4.16' de söz edilen çizimleri gerçekleştirildiği grafik penceresi görülmektedir.



Şekil 4.16. Grafik penceresi

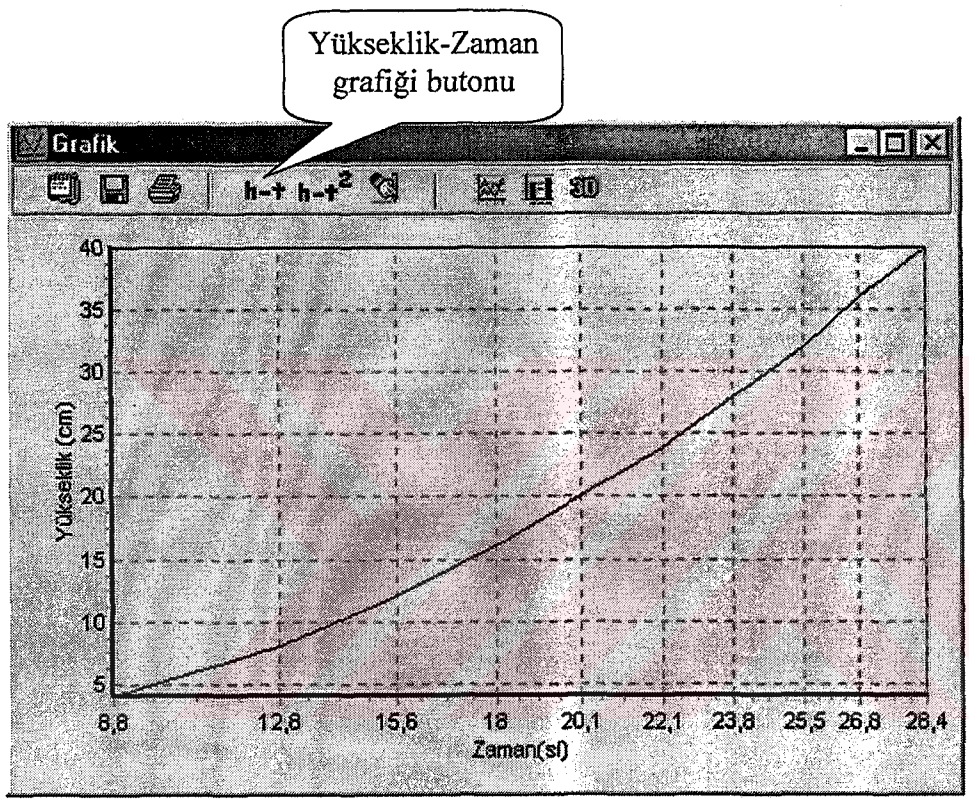
Serbest düşme deneylerinden alınmış sonuçların kayıt isimlerinin grafik çizimi için listelendiği penceredir. Şekil 4.17'de görüldüğü gibi grafik olarak görüntülenmek istenen kayıt listeden seçilir.





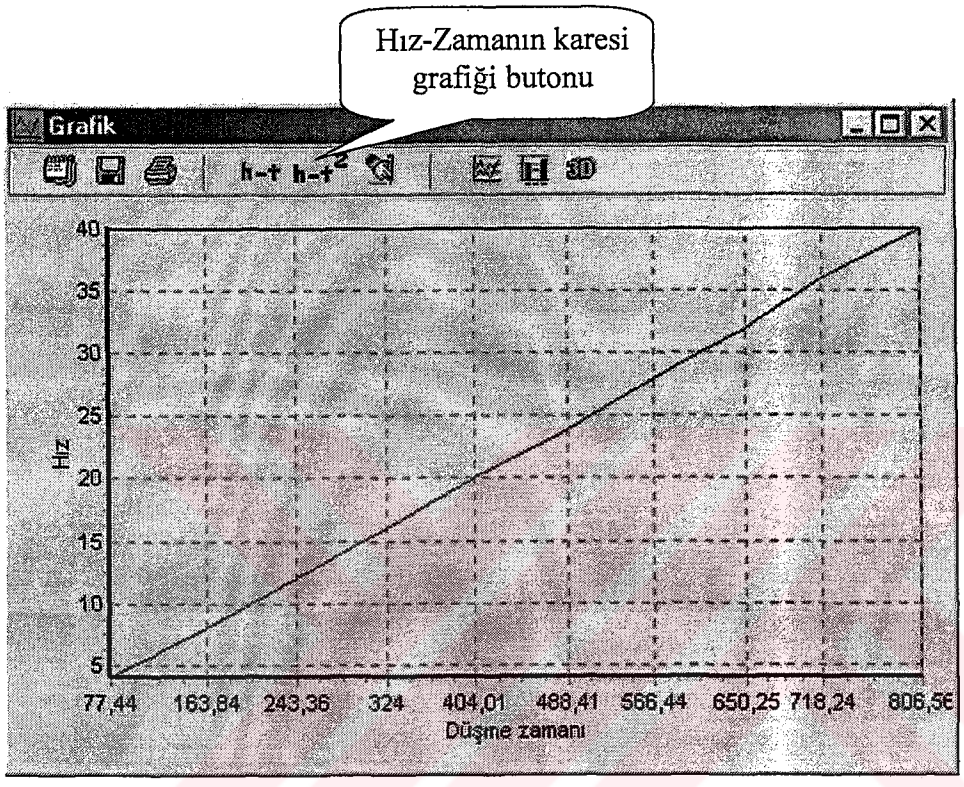
Şekil 4.17. Grafik çizimi için kayıt listesi

Şekil 4.18'de görüldüğü gibi seçilen kaydın dikey eksende yükseklik değişkeni, yatay eksende zaman değişkeninin görüntülendiği grafik çizimidir. Listeden seçilen kayıtlar otomatik olarak çizgi şeklinde görüntülenir. Değişiklik yapıldığı zaman tekrar bu buton kullanılarak tekrar çizgi olarak görüntülenebilir.



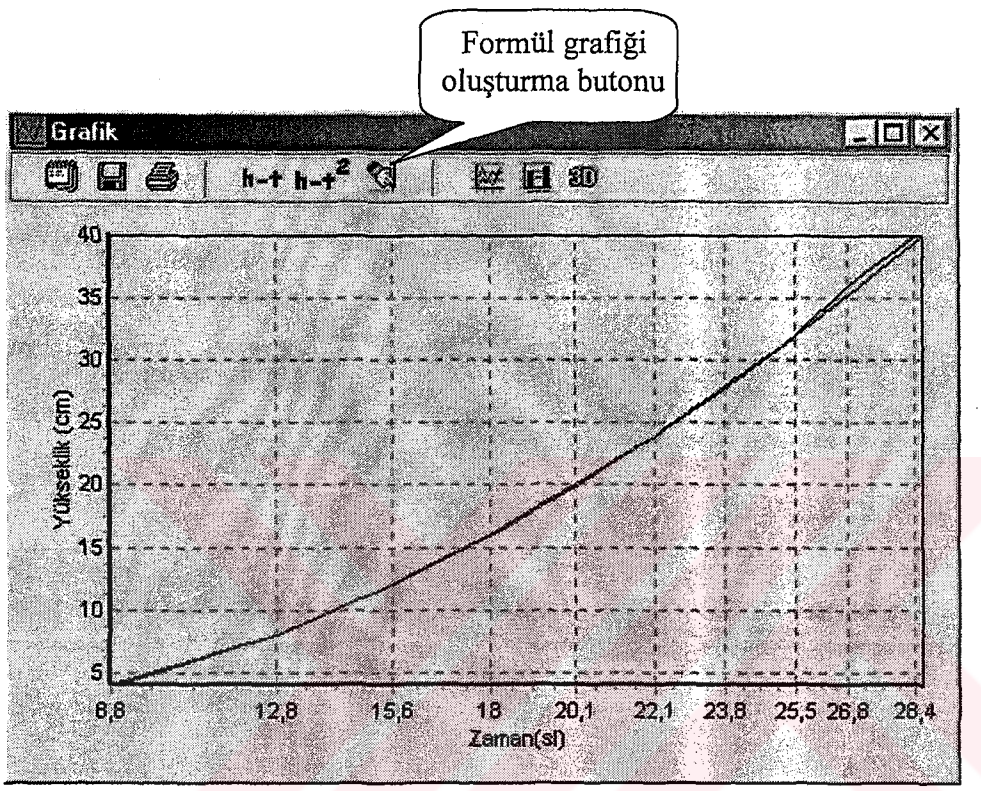
Şekil 4.18. Yükseklik-Zaman grafiği

Şekil 4.19'da görüldüğü gibi, grafik olarak görüntülenmekte olan kaydın dikey eksende yükseklik değişkeni, yatay eksende zamanın karesinin görüntülendiği grafik çizimidir.



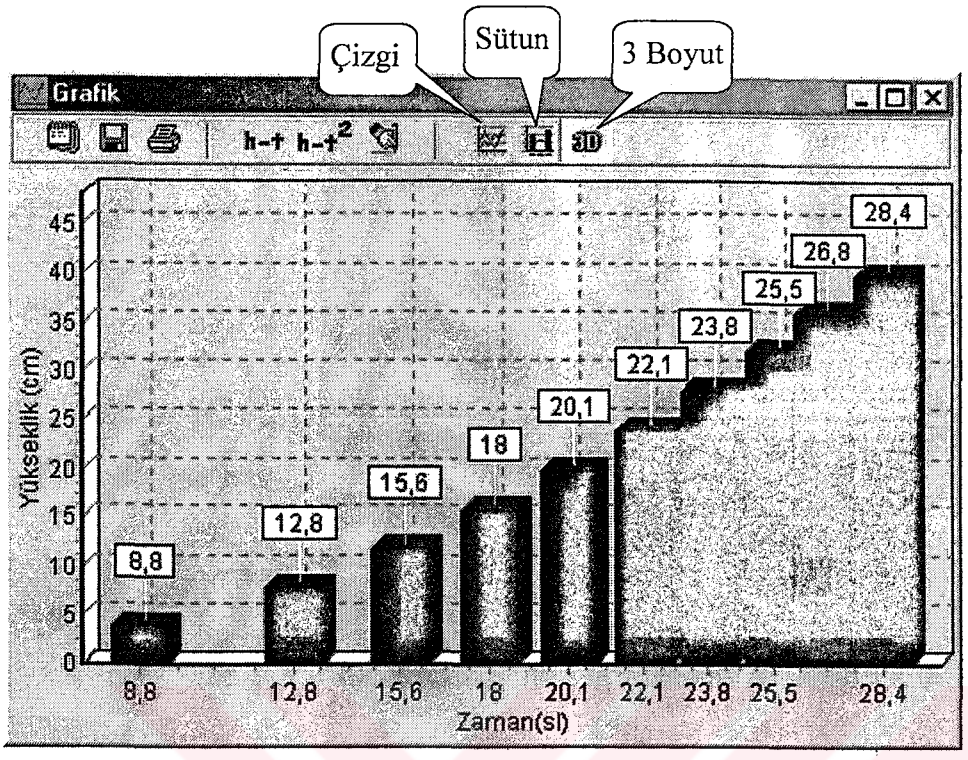
Şekil 4.19. Yükseklik-Zamanın karesi grafiği

Grafik olarak çizdirilmiş olan sonuç değerlerin formül yoluyla hesaplanarak grafik olarak çizdirildiği fonksiyondur. Şekil 4.20' de görüldüğü gibi, deneysel olarak hesaplanmış sonuçların formül yoluyla hesaplanan sonuçlarla kıyaslaması yapılır.



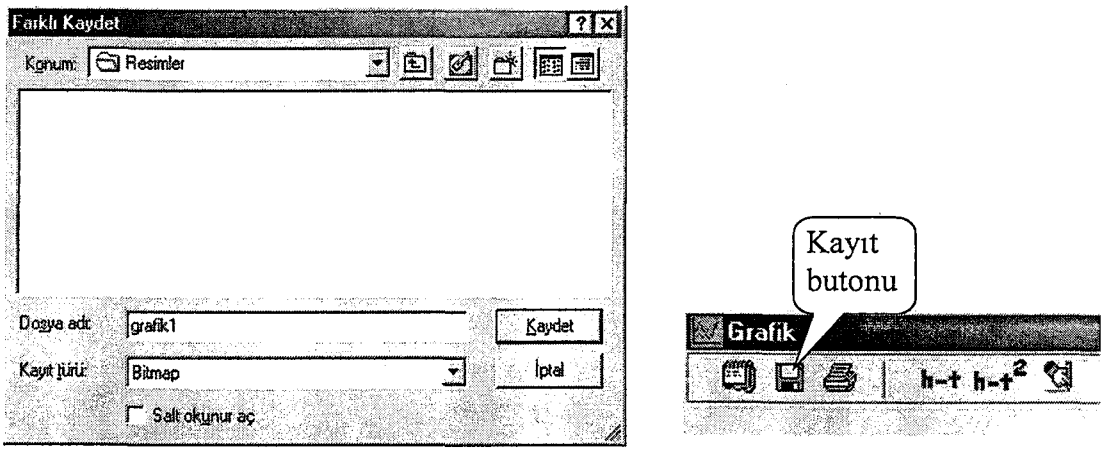
Şekil 4.20. Yükseklik-Zaman grafiği ve ideal grafik karşılaştırması

Görüntülenmekte olan grafiğin stilini değiştirmek için kullanılan butonlar Şekil 4.21'de verilmiştir. Bu butonları kullanarak grafiği görünümünü sütun veya çizgi şeklinde üç boyutlu veya 2 boyutlu olarak değiştirebiliriz.



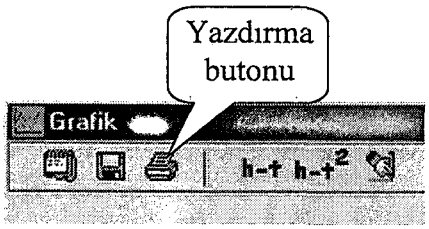
Şekil 4.21. Sütun şeklinde üç boyutlu grafik

Grafik resim olarak kaydedilmek istenirse, Şekil 4.21'de görülen kayıt butonuyla kaydetme işlemi gerçekleştirilebilir.



Şekil 4.22. Grafiğin resim olarak kaydedilmesi

Grafik penceresinde çizdirilmiş olan kaydın grafiği Şekil 4.23'de görülen yazdırma butonu kullanılarak yazdırılabilir.



Şekil 4.23. Yazdırma butonu

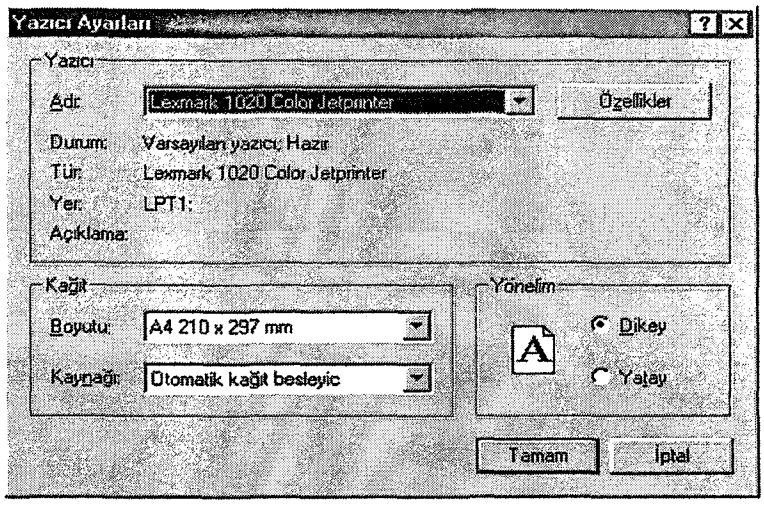
4.6. Ayar Pencereleri

Yazıcı ile ilgili ayarlar, adım motorunun dönüş hızı ve yükseklik seviyesi ayarları Şekil 4.24'de görüldüğü gibi, ana pencere üzerindeki 'Ayarlar' seçeneğinden yapılabilir.

Şekil 4.24. Ayarlar seçeneği

4.6.1. Yazıcı ayarları

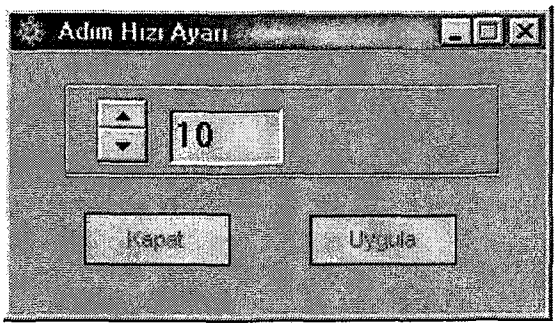
Yazıcının çıktı özelliklerinin ayarlandığı penceredir. Herhangi bir yazıcı yüklü değilse hata mesajı verir.



Şekil 4.25. Yazıcı ayarları penceresi

4.6.2. Adım motoru adım hız Ayarı

Bilyenin başlangıç noktasından alınıp istenilen yüksekliğe ulaştırılmasında ve başlangıç noktasına geri dönüşte hareketi sağlayan adım motorunun dönüş hızının ayarlandığı penceredir. Dönüş hızı bir adımda geçen bekleme süresinin değiştirilmesi ile belirlenir.



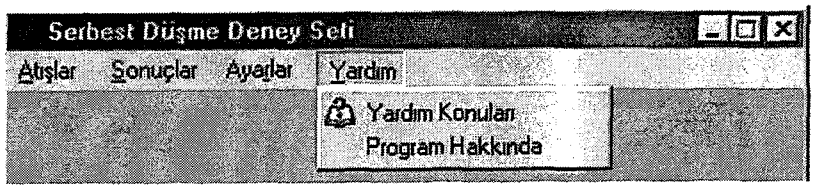
Şekil 4.26. Adım hızı ayarı penceresi

4.6.3. Yükseklik ve adım değeri penceresi

Program içerisinde maksimum yükseklik değerini belirten sabit 100 cm olarak atanmıştır. Atış deneylerinin gerçekleştirildiği düzeneğin yükseklik seviyesi artırılmak istendiğinde bu sabitin değeri değiştirilebilir. Bu işlem Şekil 4.27'de verilmiş olan pencere üzerinden yapılır. Yükseklik değerinden başka, belirtilmiş olan yüksekliğin adım motoru ile kaç adımda tamamlandığını belirten sabit değerin de belirtilmesi gerekir.

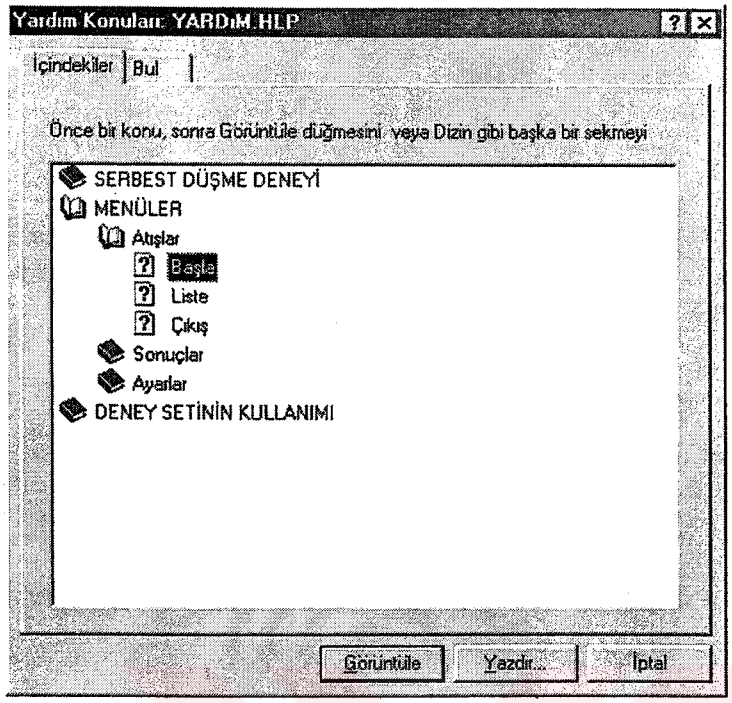
Şekil 4.27. Yükseklik ve adım sayısı referanslarının girildiği pencere

4.7. Yardım Penceresi



Şekil 4.28. Yardım seçeneği

Bu pencerede programın ve mekanik sistemin kullanımına ilişkin açıklamalar verilmiştir. Şekil 4.28'de görüldüğü gibi ana pencere üzerindeki 'Yardım' isimli seçenekten yardım penceresine ulaşılır.



Şekil 4.29. Yardım penceresi

Yardım penceresi Şekil 4.29'da görüldüğü gibi, serbest düşme deneyi, kullanıcı ara yüzünün pencereleri ve deney setinin kullanımına yönelik bilgiler içermektedir.

4.8. Sonuçlar

Bu bölümde tasarımı yapılmış olan ara yüzünün tanıtımı yapılarak pencereler ve görevleri hakkında açıklamalar verilmiştir. Bu açıklamalardan anlaşıldığı gibi serbest düşme deneyi gerçekleştirilmesi için yükseklik ve tekrar değerlerinin liste şeklinde ara yüz üzerinden girilmesi gerekir. Yükseklik listesi manuel olarak oluşturulursa istenen yükseklik seviyeleri tek tek girilir. Otomatik liste tercih edilirse başlangıç ve bitiş noktaları belirtilerek bu aralıkta kaç yükseklik seviyesi istendiği belirtilir.

Atışların geçekleştirilmesi için 'Atış penceresi' kullanılır. Bu pencere açıldığında girilmiş olan yükseklik değerleri liste olarak sıralanır. Atışlar başlatıldığında düşme süreleri ilgili atışın karşısına yazdırılır. Mekanik sistemdeki bir arıza nedeniyle atış gerçekleşmezse atış işlemi kullanıcı tekrar başlatıncaya kadar durur. Atışlar tamamlandı ise bir kayıt adı ile kaydedilerek pencere kapatılır.

Deneylerden elde edilen sonuçlar liste veya grafiksel olarak kullanıcıya sunulur. Bunun için ara yüz üzerindeki sonuçlar seçeneği kullanılır. Liste üzerinde kayıt isimleri ve seçilen kayda ait yükseklik ve sonuç değerleri sıralanır. Grafiksel gösterimde ise seçilen kayıt adlarına ait yükseklik ve düşme zamanlarından elde edilen grafikler sunulur. İdeal grafikle çizdirilen grafiğin kıyaslaması yapılabilir. Grafik çizgi veya sütun şeklinde ve istenirse üç boyutlu olarak çizdirilebilir.

Atış siteminin yüksekliği ile ilgili ayarlar, adım motorunu hızı ve yazıcı çıktı ayarları ana p encere ü zerindeki 'Ayarlar S eçeneği' ile gerçekleştirilir. Yükseklik ayarı atış sisteminin boyutları değiştirildiğinde eni yükseklik ve bu yüksekliğe ulaşmak için gereken adım sayısı belirtilir. Kullanılan bilgisayarın çalışma hızının düşük olması adım motorunu dönüşü sırasında hatalı çalışmaya sebep olabilir. Adım motorunun dönüş hızı uygun değere ayarlanabilir.

Programla ilgili açıklamalara ve oluşabilecek hatalara 'Yardım' penceresi üzerinden ulaşılır.

BÖLÜM 5. SONUÇLAR

Günümüzde serbest düşme hareketi fizik dersinin bir konusu olarak incelenmekte ve laboratuar uygulamalarında bununla ilgili deneyler gerçekleştirilmektedir. Bu araştırmalardan, yer yüzüne doğru hareket eden cismin hava sürtünmesi ihmal edilirse, şekli, yapısı ve kütlesi ne olursa olsun sabit ivmeyle düştüğü sonucuna ulaşılmıştır.Yapılan bu çalışmada serbest düşme deneyinin gerçekleştirilmesine ve elde edilen sonuçların grafiksel olarak sunulmasına yönelik bir bilgisayar kontrollü deney seti tasarımı yapılmıştır.

Serbest düşme deneyinde ölçümlerin yapılabilmesi için cismin serbest bırakıldığı yükseklik seviyesinin ve bırakıldığı andan yere düşmesine kadar geçen sürenin bilinmesi gerekir. Serbest düşme deney seti bu işlemleri, bilgisayar kontrollü bir atış sistemi tarafından gerçekleştirmektedir. Kullanıcı, deney sistemini çalıştırıldığında deneyi gerçekleştirmek için, atış yapmak istediği yükseklik değerlerini girerek bir yükseklik atış listesi oluşturur. Atışlar geçekleştirildikten sonra elde edilen düşme zamanları kaydedilir. Bu kayıtlar liste veya grafiksel olarak kullanıcıya sunulur.

Deney sisteminin yapısı yapılan tez çalışması açısından, programlama kısmı ve elektronik kısmının tasarımı olmak üzere iki şekilde değerlendirilmiştir. Programlama kısmı ara yüzün bir görsel programlama dili kullanılarak yazılması işlemini kapsamaktadır. Burada kullanılan 'Delphi' görsel programlama dili ile deney sisteminin istenen şekilde çalışması için tasarlanan olan algoritmalar kullanılarak ara yüz yazılmıştır. Elektronik kısmı ise üç ana elemandan oluşur. Bunlar, yükseklik ayarında kullanılan adım motoru, bilyenin tutulmasında kullanılan elektromıknatıs, bilyeyi düşme noktasında algılayan algılama sistemidir.

Sonuç olarak tasarımı gerçekleştirilmiş olan deney seti bu deneylerin gerçekleştirilmesinde ve elde edilen sonuçların değerlendirilmesinde kolaylık sağlamaktadır.

TARTIŞMA VE ÖNERİLER

Serbest düşme deney setinin, istenen yükseklik değerlerine bağlı olarak düşme zamanı değerlerini minimum hatayla ölçmesi ve elde edilen sonuçları kullanıcıya görsel olarak sunması amaçlanmıştır. Bununla birlikte gerçekleştirilmiş olan deney seti bazı dezavantajlara sahiptir. Tez çalışması içinde ayrıntılı olarak anlatıldığı gibi serbest düşme deney seti yapı olarak bilgisayar ve bunun kontrolünde çalışan elektromekanik atış sisteminden oluşmaktadır. Bu nedenle oluşan hataları bilgisayardan kaynaklanan hatalar ve elektromekanik sistemden kaynaklanan hatalar olmak üzere sınıflandırabiliriz.

Bilgisayardan kaynaklanan hatalar zamanlama ile ilişkilidir. Serbest düşme deneyinde bilyenin düşmesi ile birlikte, bilye algılanana kadar paralel port, 1 ms'lik sabit zaman aralıklarıyla kontrol edilir. Günümüzdeki bilgisayarlar bu hıza rahatlıkla izin vermektedir. Fakat deney sırasında ara yüz programından başka programların çalıştırılması zamanlama işleminde hatalara yol açarak düşme zamanının yanlış ölçülmesine sebep olur. Aynı şekilde, yükseklik seviyesi belirlenirken adım motoruna gönderilen dönüş sinyallerinin adım motoruna gönderilme hızı bilgisayarın çalışma hızı ile sınırlanmaktadır.

Elektromekanik sistemden kaynaklanan hatalardan birisi elektromıknatıstan kaynaklanan ölçme hatasıdır. Bu ölçme hatası elektromıknatısın enerjisi kesildiği halde bir süre bilyeyi bırakmamasından kaynaklanmaktadır. Diğeri ise algılayıcı sistemden kaynaklanan hatalardır. Bu hata bilyenin düşme anında algılama çizgisinin kenarından geçmesinden kaynaklanır. Bu ise atış sisteminin düz bir zemine yerleştirilmemiş olmasıdır. Bu nedenle atışlar gerçekleştirilirken sistemin eğik durmamasına dikkat edilir.

.

KAYNAKLAR

[1] **YALÇIN**, Cengiz, "Fiziğin Temelleri', Savaş yayınları, Ankara, 1985,

[2] cosmology.berkeley.edu,

[3] **VATANSEVER**, Fahri, "Borland Delphi ile görsel programlama", 2000,

[4] **AXELSON**, Jan, "Her Yönüyle Paralel Port", Bileşim yayıncılık, İstanbul, 2000,

[5] **SAX**, H., "Stepper Motor Driving", SGS Thomson Microelectronics, 1995,

[6] **GÜRDAL**, Osman, "Elektromanyetik Alan Teorisi", Nobel, Ankara 2000,

[7] **NAVE**, R., "Light Emmiting Diode Structure", hyperphysics.phy-astr.gsu.edu,

[8] **ANTON**, Kruger, "Active / Passive Components", www.chipcenter.com,

[9] **EKİZ**, Hüseyin, "Sayısal Elektronik", Değişim yayınları, Adapazarı 2000,

[10] **CHARLES**, Calvert, "Delphi", Sistem Yayıncılık, İstanbul, 1997,

[11] www.torry.net/listf.htm,

[12] **ALEKSEY**, Kuznetsov, "ChartView Components for Delphi32" www.utilmind.com, 1999.

ÖZGEÇMİŞ

1979 Edirne doğumludur. İlk, orta ve lise öğrenimlerini Edirne'de tamamlamıştır. Üç yıl Edirne Endüstri Meslek Lisesinde öğrenim gördükten sonra, 1998 yılında Sakarya Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Elektronik Öğretmenliği Programına girmiştir. 2001 yılında öğrenimini tamamlayarak aynı yıl Sakarya Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi Bölümünde Yüksek Lisans Eğitimine başlamıştır, halen öğrenci olarak eğitimine devam etmektedir.