2002 Gazi Üniversitesi Endüstriyel Sanatlar Eğitim Fakültesi Dergisi, Y.10,S.10,S.01-11

# BİLGİSAYAR DONANIMI İŞLEYİŞ İLKELERİNİN ÖĞRETİMİNDE ETKİN BİR YAKLAŞIM: BİZİM BİLGİSAYAR

Doğan ÇALIKOĞLU*

## ÖZET

*Bu makalede, bilgisayar donanımı işleyiş ilkelerinin öğretimi için geliştirilen ve etkinliği kanıtlanan özgün bir yaklaşım anlatılmaktadır. Bu yaklaşımda, mantık geçitleri bilgisi temel alınarak başlanmakta, önce BB/1 (Bizim Bilgisayar, Model-1) dediğimiz bir bilgisayar modelinin tasarımı gerçekleştirilmektedir. Daha sonra bu ilk model üzerinde bir takım iyileştirmeler yapılarak BB/2 elde edilmektedir. Tasarıma konu olan bilgisayar modelinin özelliklerinin gerçeğe uygunluğu hususunda öğrencide tam bir güven hissi uyandırabilmek için, modern bilgisayarların temel işleyiş ilkelerini yeterince yansıtan ve gerçek bir bilgisayar olan PDP/8 ölçü alınmakta ve BB/1'in özellikleri PDP/8'den aşağı kalmayacak şekilde belirlenmektedir. Ayrıca BB/1'in komut kümesinin tamlığı ve bu komutlarla ciddi izlencelerin yazılabilirliği kanıtlanarak bu güven hissi pekiştirilmektedir. Bu arada hem BB/1 izlencelerinin yazılışını ve okunuşunu kolaylaştırmak, hem de öğrenciye "assembler" ilkelerini tanıtmak amacıyla, BBSD ismini verdiğimiz basit fakat yeterince güçlü bir simgesel dil tanımlanmaktadır. BB/1'in donanımını tasarımlarken "Kayıtlık Aktarım Dili" kullanılmaktadır. Ortaya çıkarılan tasarım bir yandan, Kayıtlık Aktarım Dili'ndeki anlatımının PASCAL diline çevrilmesiyle elde edilen bir donanım simülatöründe irdelenmekte, diğer taraftan da geçit devreleri cinsinden nasıl ifade edileceği tam bir netliğe kavuşturulmaktadır.*

**Anahtar Kelimeler**: *Kayıtlık aktarım dili, donanım tasarımı, bilgisayar öğretimi*

# AN EFFECTIVE APPROACH IN TEACHING THE OPERATING PRINCIPLES OF COMPUTER HARDWARE: BİZİM BİLGİSAYAR

## ABSTRACT

*In this paper, an original approach that is developed and proved to be effective, to teach the operating principles of computer hardware, is being described. In this approach at the beginning, a knowledge of logic gates is taken as the basis and the design of a computer model, that we call BB/1 (Bizim Bilgisayar, Model-1) is realized. Later, BB/2 is obtained by making some improvements on the first model. In order to raise a perfect feeling of confidence in the student that the specifications of the computer model subject to design are in accordance with reality, the PDP/8 computer which is a real computer and which sufficiently reflects the basic working principles of the modern computers is taken as a measure and the specifications of BB/1 is determined not to fall below that of PDP/8. In addition, this feeling of confidence is strengthened by proving the completeness of BB/1's instruction set and that serious programs can be written with those instructions. In the meanwhile, both to ease the writing and reading of BB/1 programs, and to introduce the assembler principles to the student, a simple yet sufficiently powerful symbolic language is defined which we call BBSD. "Register Transfer Language" is used in designing the hardware of BB/1. On one side, the design brought about is verified on a hardware simulator obtained by translating the description in Register Transfer Language to PASCAL Language, on the other side, how the design would be expressed in terms of gate circuits, is made completely clear.*

**Key Words**: *Register transfer language, hardware design, teaching computers.*

*Prof.Dr. Gazi Üniversitesi Endüstriyel Sanatlar Eğitim Fakültesi Bilgisayar Eğitimi Bölümü



## 1. GİRİŞ

"**Mantık geçitleri nasıl oluyor da bilgisayarın işlemesini sağlıyor**" sorusu, ülkemizin bilgisayar alanında öğretim yapılan programlarında genellikle yeterince üzerinde durulmayan veya cevabı tam bir netliğe kavuşturulamayan bir sorudur. Bu makalede sunulan çalışmada bu konu ele alınmakta, ve **bilgisayar donanımının işleyiş ilkelerinin, mantık geçitleri düzeyine kadar** öğrenci tarafından tam olarak anlaşılmasını sağlayacak etkin bir yaklaşım ortaya konmaktadır.

Üniversitelerimizin bilgisayar alanında öğretim yapılan programlarında mantık tasarımı konusu çerçevesinde temel sayısal devreler (geçit devreleri) okutulmakta, daha sonra ise bilgisayar organizasyonu konusuna geçilerek bilgisayarın donanımsal yapısı öğretilmektedir. Bu arada "**mantık geçitleri nasıl oluyor da bilgisayarın işleyişini gerçekleştiriyor**" şeklinde ifade ettiğimiz çok önemli bir bağlantı sorusu bulunmaktadır. Bu soruya yeterli bir karşılık verilmediği takdirde bilgisayar alanındaki öğretim programlarında mantık tasarımı konusunun tuttuğu yeri öğrenciye tam olarak izah etmek mümkün olamamaktadır. Gerek kataloglardan, gerek kişisel temaslardan, gerekse de çeşitli üniversitelerin mezunlarından edinilen bilgilerden, yukarıda ifade ettiğimiz bağlantı sorusunun ele alınışında üniversitelerimiz genelinde benimsenen yaklaşımların yeterince etkin olmadıkları anlaşılmaktadır.

Bu makalede, yukarıda ifade edilen soruna bir çözüm olarak, bilgisayar donanımı işleyiş ilkelerinin öğretimi için geliştirilen ve etkinliği kanıtlanan özgün bir yaklaşım anlatılmaktadır. Bu yaklaşımın ana özellikleri şunlardır:

1. Başlangıçta mantık geçitleri bilgisi, yeterli bir temel olarak alınmaktadır.
2. BB/1 (Bizim Bilgisayar, Model-1) dediğimiz bir bilgisayar modelinin tasarımı gerçekleştirilmektedir.
3. Tasarımda, gerçek bir bilgisayar olan ve modern bilgisayarların temel ilkelerini yeterince yansıtan PDP/8 ölçü alınmakta ve PDP/8'den aşağı kalmamasına dikkat edilerek, BB/1'in özellikleri belirlenmektedir. Böylece tasarıma konu olan bilgisayar modelinin özelliklerinin gerçeğe uygunluğu hususunda öğrencide tam bir güven hissi uyanmaktadır.
4. BB/1'in komut kümesinin tamlığı ve bu komutlarla ciddi izlencelerin yazılabilirliği örneklerle kanıtlanarak bu güven hissi pekiştirilmektedir.
5. Bu arada hem BB/1 izlencelerinin yazılışını ve okunuşunu kolaylaştırmak, hem de öğrenciye "assembler" ilkelerini tanıtmak amacıyla, basit fakat yeterince güçlü bir simgesel dil olan **BBSD** (**B**izim **B**ilgisayarın **S**imgesel **D**ili) tanımlanmaktadır.
6. BB/1'in donanımının tasarımında "Kayıtlık Aktarım Dili" kullanılmaktadır.
7. Ortaya çıkarılan tasarım, onun Kayıtlık Aktarım Dili'ndeki anlatımının, PASCAL diline çevrilmesiyle elde ettiğimiz bir donanım simülatöründe irdelenmektedir.

8. Simülatörün salt mantıktan ziyade, doğrudan donanımı simüle etmesi, donanımda düşünülecek değişikliklerin doğrudan simülatöre yansıtılabilmesini ve kolayca denenebilmesini sağlamaktadır.

9. BB/1 tasarımı üzerinde bir takım iyileştirmeler yapılarak "BB/2" elde edilmektedir. Bu arada, hem model yükseltiş kavramı işlenmekte, hem de simülatörün doğrudan donanımı simüle etmesi özelliğinin yararı gösterilmektedir.

10. Bir taraftan bu simülatör sayesinde tasarım elle tutulur hale getirilirken, diğer taraftan da söz konusu tasarım geçit devreleri düzeyine kadar ayrıntılandırılmakta ve geçit devreleri cinsinden şemalar çıkarılarak, bilgisayar donanımı işleyiş ilkelerinin baştan sona kadar tam bir netlik içinde kavranması sağlanmaktadır.

## 2. PDP/8 BİLGİSAYARI VE BAŞLICA ÖZELLİKLERİ

Bilgisayar donanımı ilkeleri gibi bir konunun genel boyutlarda ve sırf teorik olarak ele alınıp etkin bir tarzda öğretilmesi mümkün değildir. Mutlak surette belli bir sistemin seçimi gerekmektedir. Seçilen sistem gerçeğe uygun ve yeterince işlevsel olmakla birlikte, gereğinden fazla karmaşık da olmamalıdır. Bu itibarla yeni modellerden birinin esas alınması pedagojik açıdan uygun değildir. Ele alınacak tasarım için, bütün bu değerlendirimler çerçevesinde bir örnek araştırılmıştır. Sonuçta, Digital Equipment Corporation yapımı PDP/8 bilgisayarının, işlevsellikten taviz vermemek şartıyla pedagojik rötuşlardan geçirilmiş bir şekli, "Bizim Bilgisayar" ismi altında benimsenmiştir.

PDP/8 bilgisayarı, tasarımının nisbi sadeliğine rağmen, simgesel boyutlarda ileri teknikleri andıran özelliklere sahiptir. Burada sunulan yaklaşımın geliştirilmesinde, benzer bir yöntem ortaya koyan Mano'dan da yararlanılmıştır (Mano, 1982). Ayrıntılara geçmeden önce, PDP/8 bilgisayarının başlıca özelliklerinden söz etmekte yarar vardır.

PDP/8, 1960'lı senelerde Digital Equipment Corporation firması tarafından piyasaya sürülen ve Dünyada mini-bilgisayarlar sınıfında önemli yer tutmuş olan bir bilgisayardır. Türkiye'ye ilk 1969 yılında girmiş ve ODTÜ Elektrik Mühendisliği Bölümü Bilgisayar Laboratuarına kurulmuştur. Ana hafızası, her biri 12 bitlik olan 4K kelimeden oluşmaktadır. Bütün aritmetik, mantık ve kaydırmaca işlemleri, "birikeç" denen 12 bitlik bir kayıtlıkta yapılmaktadır.(Şekil 1)


Şekil 1: PDP/8 bilgisayarının ana hafızası ve temel kayıtlıkları

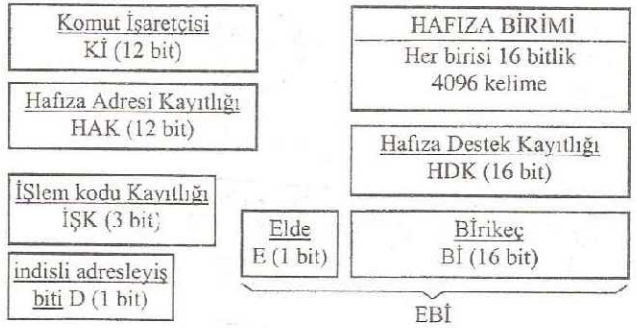


Aritmetik işlem komutları, 12 bitlik işaretsiz tamsayılar içindir. Bu komutlar, bir adet topla komutu, bir adet **"2-tamlayanını al"** komutu ve kaydırmaca komutlarından ibarettir. Programcılar işaretli tamsayıları temsil etmek için 2-tamlayan aritmetiğini kullanmaktadırlar. Böylece, işaret değiştiriş ve çıkartış mümkün olmaktadır. Çarpım, kaydırarak toplayışla, bölüm, kaydırarak çıkartışla elde edilmektedir. İki kelimelik veya daha uzun sayılarla işlemler, arada elde bitini kullanarak kelime kelime gerçekleştirilmektedir. PDP/8, bir-adresli bir bilgisayardır ve komutları birer kelimeliktir. Toplam altı adet hafıza adresleyen komutu vardır. Bu komutlarda üç bit, işlem kodu olarak kullanılmaktadır. Geriye kalan dokuz bit ise, bir hedef adres belirlemekte kullanılmaktadır. Ancak bu dokuz bit, 4K'lık hafızayı doğrudan adreslemeye yetmediği için, indisli adresleyiş yöntemi ile "paging" yani sayfalandırış yöntemine müracaat edilmektedir. İndisli adresleyiş için özel bir indis kayıtlığı bulunmayıp, hafıza yerlerinin her biri bu amaçla kullanılabilmektedir.

Günümüz ölçüleriyle gülünç derecede kısıtlı görünen bu boyutlara rağmen PDP/8, FORTRAN izlencelerini bir geçişte derleyebilen bir FORTRAN derleyicisine sahiptir. PDP/8'in yeterince işlevsel olduğu, üzerinde diferansiyel denklemler çözülmesi, komut kümesi üzerinde geliştirmeler yapılması, Analog sistem denetimi, veri toplama, genel sistem denetimi, mikrobilgisayar geliştirme, mikrobilgisayar ödevlerinin otomatik değerlendirilmesi gibi çalışmalarda kullanılmasıyla kanıtlanmıştır. (ÇALIKOĞLU, D., 1970, 1975, 1978, 1979, 1980a, 1980b, 1980c, 1982a, 1982b, 1983)

## 3. BİZİM BİLGİSAYAR'IN ÖZELLİKLERİ

Bizim Bilgisayar'ın ilk modeline BB/1 demekteyiz (Şekil 2). PDP/8'e göre temel fark, kelime uzunluğunun 12 yerine 16 bit olmasıdır. BB/1 komutları, PDP/8'inkilere paraleldir. Her komut bir kelimedir ve bütün önemli PDP/8 komutlarının bir eşdeğeri vardır. Kelime uzunluğunun 16 bite çıkarılması sayesinde, işlem kodu ve indisli adresleyiş biti için dört bit ayrıldıktan sonra geriye kalan 12 bit ile 4K hafızanın tamamının doğrudan adreslenmesi mümkün olmaktadır.




Şekil 2 : BB/1 bilgisayarının ana hafızası ve temel kayıtlıkları

PDP/8'deki komut tasarımına paralel olarak BB/1'de, Hafıza adresleyen komutlar, Kayıtlık komutları ve Giriş-çıkış komutları olarak üç türlü komut vardır.

Hafıza adresleyen komutlar şunlardır:

**VE** Hedef hafıza yerindeki kelimeyi Bİ'e **VE**'le
**TOP** Hedef hafıza yerindeki kelimeyi (Elde E olarak) Bİ'e **TOP**la
**YÜK** Hedef hafıza yerindeki kelimeyi Bİ'e **YÜK**le
**SAK** Bİ'yi hedef hafıza yerine **SAK**la
**SAP** Hedef hafıza adresine **SAP**
**DÜS** **D**önmek **Ü**zere **S**ap (Altyordamlara gitmek için)
**ASS** **A**rtır, **S**ıfırsa **S**ek

Her hafıza adresleyen komut kelimesinde, indisli adresleyişi belirtmek için bir bit, işlem kodu için ise üç bit bulunur. Geri kalan oniki bit adrestir. Burada, PDP/8'deki altı komut kapsanmakta, fazladan da **YÜK** komutu bulunmaktadır.

Kayıtlık komutları şunlardır:

| | | | |
|---|---|---|---|
| SLB | Bİ'yi sil | ART | Bİ'yi bir artır |
| SLE | E'yi sil | BAS | Bİ artıysa sek |
| TMB | Bİ'yi tamla | BES | Bİ eksiyse sek |
| TME | E'yi tamla | BSS | Bİ sıfırsa sek |
| SAD | E ve Bİ'yi sağa döndür | ESS | E sıfırsa sek |
| SOD | E ve Bİ'yi sola döndür | DUR | MİB'i durdur. |

(sek = sıradaki komutu atla.)

Giriş-çıkış komutları:

OKU Bİ'ye bir bayt oku
YAZ Bİ'den bir bayt yaz
GBS Giriş bayrağı 1 ise sek.
ÇBS Çıkış bayrağı 1 ise sek.
KEV Kesintiye imkan Var
KEY Kesintiye imkan Yok

## 4. BB SİMGESEL DİLİ: BBSD

BB/1'in komutlarıyla ciddi izlencelerin yazılabilirliğini meydana çıkaracak örnekleri yazmakta ve onları okumakta kolaylık sağlamak için, kuralları belli bir simgesel dil gereklidir. Bu amaçla, BB Simgesel Dili (kısaca BBSD) ismini verdiğimiz basit fakat yeterince güçlü bir simgesel dil ortaya konmuştur. Aşağıda görülen BBSD'nin tanımı, bir sayfaya sığacak kısalıkta olmakla birlikte lüzumlu gördüğümüz bütün özellikleri içermektedir. Bu simgesel dil vesilesiyle, öğrencilere "assembler" ilkelerinden söz etmek de mümkün olmaktadır.



**BBSD'nin tanımı**:

1. Sayılar 16 tabanında ifade edilirler.
2. Her satırda en çok bir hafıza yeri muhtevasının (komut veya veri) ifadesi olabilir.
3. Makine komutlarının hafızada denk geleceği yerin tayini için, sıradaki makine komutlarının hangi yerden itibaren dizilmesi istendiği, bir YER yardımcı komutuyla belirtilir. Ör. YER 200
4. Makine komutlarının sayısal kodları yerine simgeleri yazılabilir.
5. Bir hafıza yerinin muhtevası bir simge olarak, veya işaretli/işaretsiz bir 16'lık sayı olarak yazılabilir. Bu sayı eksi işaretli olduğunda, esas muhteva, o sayının 2-tamlayanı olur. İşaretsiz 16'lık sayıları simgelerden ayırt edebilmek için önlerine bir = işareti konur. Örneğin: BABA, bir simgedir, =BABA ise 16'lık bir sayıdır.
6. Bir satıra birden fazla komut, 16'lık sayı veya simge, aralarında boşluk bırakılmak suretiyle yazıldığında o satıra tekabül eden muhteva, bunların 16 bitlik karşılıklarının VEYA'lanmasıyla elde edilen sonuçtur.
7. Her hangi bir satırın solunda, bir harfle başlayan, virgülle veya iki-nokta-üst-üste ile biten ve arada boşluk bulunmayan bir harf-rakam dizgisi, o satıra denk gelen adres değerine sahip bir yafta (simgesel adres tanımı) dır. Böyle bir yafta ile tanımlanan bir simgesel adres, hafıza adresleyen komutlarda kullanılabilir.
   Örnek:
   ```
   Y1: SOD
       ---
       SAP Y1
   ```
8. Simgesel programa açıklayıcı ifadeler ilave edebilmek için / işareti kullanılır. Bu işaretin sağında kalan şekilcikler makine diline çevirişte dikkate alınmaz.
9. Hafıza adresleyen komutlarda hatıradan önce bulunan bir D harfi, indisli adresleyişin olduğunu belirtir.
10. Simgesel programın sonunu belirtmek için SON yardımcı komutu kullanılır.

BBSD dilinde yazılmış küçük bir izlence örneği:

```
/ BU İZLENCE, İKİ (İŞARETLİ) TAMSAYININ BÜYÜK OLANINI
/ BİRİKEÇE KOYDUKTAN SONRA DURUR.
YER 200
     YÜK  B
     TMB    / B'NİN 2-TAMLAYANINI ...
     ART    / OLUŞTUR.
     TOP  A / Bİ=A-B
     BES
     SAP  Y1 /A BÜYÜK (VEYA B'YE EŞİT)
     YÜK  B  /B BÜYÜK
     SAP  Y2
Y1:  YÜK  A
Y2:  DUR
YER 0F0
A:   =A2
B:   =8C
     SON
```

## 5. BB/1'DE ALTYORDAM YAPISI

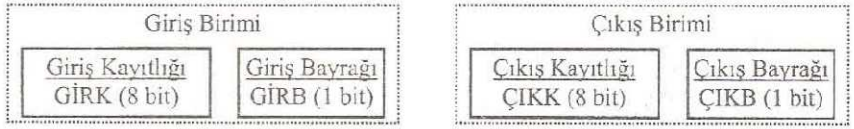
Basit altyordamlar için genel bir şekil, aşağıda görülmektedir. Somutluk açısından, altyordamın adresi 100 olarak, çağrının yapıldığı adres de 200 olarak verilmektedir.

```
/ "ALTYOR" İSİMLİ ALTYORDAM.
YER 100        / Altyordamın bulunduğu yer.
  ALTYOR: =0/ "DÜS" komutunun dönüş adresini saklaması için ayrılan yer.
  K1:          / "DÜS" komutunun hemen ardından ifa edilecek olan ilk altyordam komutu.
    Altyordamın diğer komutları.
  D SAP ALTYOR      / Altyordamdan geriye dönüş için kullanılan komut.
    Altyordama ait yerel veriler.
```

```
/ ALTYORDAM ÇAĞRISI YAPILAN YER.
YER 200
ÇAĞRI:  DÜS   ALTYOR
DEVAM: ---                    / "DÜS" KOMUTUNUN ALTINDAKİ KOMUT.
```

## 6. BB/1'DE GİRİŞ-ÇIKIŞ VE PROGRAM KESİNTİSİ

BB/1'de giriş birimi klavye, çıkış birimi de yazıcı olarak kabul edilmektedir. Bu birimlerle veri alış verişi, yapmak için şu kayıtlıklar kullanılmaktadır:




BB/1'de yalnız bir adet kesinti talep girişimiz vardır ve adına KEST denmektedir. KEST'e bayrak çıkışları bir VEYA geçiti ile bağlıdır. Kesinti izin denetimi için KYV (Kesintiye Yol Ver) isimli bir ikili vardır. Bir kesinti talebi, eğer ve ancak, KYV=1 ise kesinti doğurabilmektedir.

KEST ile ilgili donanım mantığı:


Kesintiye bakan yordam, hafızanın en başında (yer0000'da) bir altyordam şeklindedir. Buraya sapış ve dönüş, aynı DÜS komutuyla olduğu gibi yapılmaktadır.



BB/1'in, kesinti güdümlü giriş-çıkış'ı gerçekleştirmekteki elverişliliğini tartmak için şu problem ortaya konmaktadır:

"Giriş ve çıkış birimlerinin ortalama hızlarının aynı olduğunu, fakat anlık hızlarının birbirine göre değiştiğini varsayınız. Giriş biriminden gelen verileri, kesinti güdümlü giriş-çıkış tekniğinden yararlanarak, zamanı en iyi değerlendiren bir şekilde çıkış birimine aktaran bir izlence yazınız."

Bu problemin çözümünde, hafızanın bir bölgesi destek olarak kullanılmaktadır. Başlangıç değerleri gerektiği şekilde ayarlandıktan sonra kesinti beklenmektedir. Kesintiye bakan yordam, önce bayrakları yoklayarak kesinti nedenini bulmaktadır. Kesintinin giriş biriminden gelmesi, GİRK'de taze veri olduğunu göstermektedir. Hal böyle olduğunda, bu veri destek alanına aktarılmaktadır. Kesintinin çıkış biriminden gelmesi ise, ÇIKK'ın sıradaki veriyi kabul etmeye hazır olduğu anlamına gelmektedir. Böyle olduğunda, destek alanındaki işaretçilere bakılmakta ve sırada taze veri varsa, ÇIKK'a aktarılmaktadır.

Sözü edilen bu izlence, (destek alanı hariç) kesintiye bakan yordamıyla beraber BB/1 hafızasında 50 kelimeye sığmaktadır. BBSD dilindeki anlatımı ise kağıt üzerinde bir sayfadan ibarettir.

## 7. BB/1'İN İŞLEYİŞİ

BB/1'in işleyişinde zaman dayanağını bir sistem saat üreteci oluşturmaktadır,. Her darbe bir zamanı, $z_0$, $z_1$, $z_2$, $z_3$ şeklindeki her dört zaman bir komut devrini oluşturmaktadır. Genel olarak her komutun ifası, birincisi "getir devri", ikincisi "ifa devri" olmak üzere iki komut devrinde gerçekleşmektedir. İndisli adresleyiş olduğunda, bu iki devir arasına bir "adres devri" ilave edilmektedir. Kesinti vukuunda, o anda ifa edilmekte olan komutun "ifa devri"nden sonra, fazladan bir "kesinti devri" gerçekleştirilmektedir. Bu kesinti devri ile, kesintiye bakan yordama geçilmesi sağlanmaktadır. Devirler, $d_0$, $d_1$, $d_2$, $d_3$ olarak belirtilmektedir. Zaman ve devir işaretleri, uygun sayaçların çıkışlarına bağlanan kod çözücülerden elde edilmektedir.

BB/1'in işleyişinin mantıksal tasarımında, kayıtlık aktarım dili denen bir resmi dil kullanılmaktadır. Bu dildeki deyimler şu genel biçimde olmaktadırlar:

(Mantık ifadesi): (mikroişlem), (mikroişlem), ..., (mikroişlem)

Örneğin,

$d_0z_1$: HDK ← H, Kİ ← Kİ + 1

Bu deyimde, $d_0z_1$ şartı sağlandığı anda, HDK ← H ve Kİ ← Kİ + 1 mikroişlemlerinin gerçekleşeceği ifade edilmektedir. Yani, "getir devri"ndeki $z_1$ zamanında, bir yandan (HAK'ın gösterdiği adresteki) hafıza kelimesinin HDK'ya aktarılacağı, diğer yandan da, komut işaretcisinin bir artırılacağı bildirilmektedir.

Bu suretle, kayıtlıklar üzerindeki ilkel işlemler olan mikroişlemlerden hangisinin ne zaman vuku bulması gerektiği tam olarak tanımlanabilmektedir. Geçit devreleri düzeyindeki mantıksal tasarım açısından bu deyimin anlamı, $d_0$ ve $z_1$ işaretlerinin bir VE geçidinde

birleştirilmesiyle elde edilen $d_0z_1$ işaretinin, bir tarafta HDK'nın hafızadan veri alan girişine yol verici olarak bağlanması, diğer taraftan ise Kİ'nin "bir artır" girişine bağlanması gerektiğidir. Söz konusu girişlere aynı mikroişlemleri gerçekleştirmesi gereken başka işaretler de geldiği takdirde, bunlar önce bir VEYA geçidinde toplanacak, ondan sonra gereken kayıtlık girişlerine uygulanacaktır.

## 8. BB/1 İÇİN BİR DONANIM SİMÜLATÖRÜ

BB/1'in işleyişinin mantıksal tasarımını irdelemek için "kayıtlık aktarım dili"ndeki anlatımı PASCAL diline çevrilerek bir simülatör üretilmiştir. Bu simülatörde BB/1 hafızası,

```
Const Azami_adr=4095; Var HAFIZA :Array[0..Azami_adr] of word;
```

şeklinde, 16 bitlik kayıtlıklar `word`, 8 bitlik kayıtlıklar `byte`, bayraklar ise `bit`, olarak tanımlanmaktadır.

Herhangi bir BB/1 izlencesinin yürütülebilmesi için onun öncelikle "Yukle" şeklinde çağırılan bir "procedure" altyordamı vasıtasıyla "HAFIZA" isimli diziye, yerleştirilmesi gerekmektedir. Daha sonra `Yurut(adres)` şeklinde çağırılan bir "procedure" altyordamı vasıtasıyla, hafızadaki BB/1 izlencesinin belirtilen "`adres`"ten itibaren yürütülmesine başlanmaktadır.

BB/1 izlenceleri simülatörde yürütülürken kayıtlıklarındaki değerler izlenebilmektedir. Bu amaçla simülatör, seçime bağlı olarak, her "zaman adımı" bitiminde, her devir bitiminde, veya her komut bitiminde duraklatılabilmektedir.

Yukarıda örnek olarak verilmiş bulunan kayıtlık aktarım dili deyimi, simülatördeki `Yurut` altyordamı içinde şöyle yer almaktadır:

```
if d_[0]*Z[1]=1 then
              Begin HDK:=HAFIZA[HAK]; KI:= (KI+1) MOD (Azami_adr+1) end;
```

`Yurut` altyordamının genel mantığı şöyledir:

1- Kayıtlık ilk değerlerini ata,
2- Simüle edilecek devire ve zamana göre kod çözücü çıkış değerlerini belirle,
3- Kayıtlık aktarım dili satırlarını tara, şartları geçerli olan mikroişlemleri gerçekleştir,
4- Kayıtlık değerlerine bakılacaksa durakla ve değerleri sergile.
5- Sıradaki zamana geç,
6- İşleyiş devam edecek ise 2 no lu satıra git, yoksa çık.

## 9. BB/2: BİR ÜST MODEL

Tasarımlanan BB/1 bilgisayarı, biraz daha geliştirilmiş ve adına BB/2 denmiştir. BB/2'de iki adet yenilik yer almaktadır:



1. Kesinti düzenindeki kesinti talebi kaynakları ayrı ayrı izne bağlanmaktadır. Bu maksatla, her bayrak için, GBİ (Giriş Bayrağı İzni) ve ÇBİ (Çıkış Bayrağı İzni) şeklinde birer "izin" ikilisi getirilmektedir ve KEST girişi KEST= GBİ· GİRB+ÇBİ· ÇIKB mantığıyla yeniden düzenlemektedir. Böylece, bayraklardan istenmeyenlerin kesinti talebinde bulunmaları önlenebilmektedir. Bu yeni ikilileri izlence denetimine almak için de komut kümesi genişletilmektedir.

2. Kayıtlık komutlarında kullanılan mikroişlemlerin zamanları yeniden düzenlenmektedir. Böylece birden fazla işlem aynı komutun içine programlanabilmekte ve bu da komut kümesini daha çok zenginleştirmektedir. Örneğin, "önce Bİ'yi tamla, sonra bir artır" gibi 2-tamlayan aritmetiğinin temel bir işlemini gerçekleştiren yeni bir komut kazanılmaktadır.

## 10. SONUÇ

"**Mantık geçitleri nasıl oluyor da bilgisayarın işleyişini gerçekleştiriyor**" sorusunun, ülkemizde bilgisayar alanında öğretim yapılan programlarda okuyan öğrencilerin zihinlerinde açıklığa kavuşturulması önemlidir. Bu amaca ulaşabilmek için, bu makalede özgün bir yaklaşım ortaya konmuştur.

Temel olarak mantık geçitleri bilgisi yeterli kabul edilen bu yaklaşımda, gerçek bir bilgisayar örneğinden yola çıkılarak bazı pedagojik rötuşlarla BB/1 ve BB/2 dediğimiz bilgisayar modellerinin tasarımı gerçekleştirilmektedir. Tasarımda kayıtlık aktarım dili kullanılmaktadır. Bu dilde anlatılan tasarımın hem anlaşılması, hem simüle edilmesi, hem de mantık devrelerinin elde edilmesi kolay olmaktadır. Bu kolaylıkları deneyen öğrenciler kazandıkları güven duygusuyla, bilgisayar yapabilecek bir seviyeye geldiklerini ifade etmektedirler. Bu da uygulanan yaklaşımın etkinliğinin bir göstergesi olarak değerlendirilmektedir. Bu arada öğrenci, bilgisayarlarla ilgili farklı konulara da açılmaktadır.

Burada anlatılan yaklaşım esnasında, BB/1-2 için basit fakat yeterince güçlü bir simgesel dil olsun istenmiştir. Bu ihtiyaç karşısında yapılan bir çalışmayla, bir yan katkı olarak BBSD dilinin bir sayfalık bir tanımı elde edilmiştir.

## KAYNAKLAR:


BARBACCI, M.R. (1975) "A Comparison of Register Transfer Languages for Describing Computers and Digital Systems," IEEE Trans. On Computers, Vol. C-24 (Şubat 1975), s.137-150.

ÇALIKOĞLU, D. (1970) Design and Implemantation of a Hybrid Computer Linkage System, M.Sc. Thesis, ODTÜ, Elektrik Müh. Böl., Mayıs 1970.

ÇALIKOĞLU, D. (1975) "Bilgisayarın Yürüyüş Bozukluklarını Tanılamak İçin Veri Toplamada Kullanılması", I.Uluslararası Biyomühendislik Toplantısı 12-14 Mayıs 1975, ODTÜ, Ankara.

ÇALIKOĞLU, D. (1978) "Bir Mikro-Hesaplayıcı Sistemi", TÜBİTAK VI. Bilim Kongresi, 24-28 Ekim 1977, İzmir, Mühendislik Araştırma Grubu Tebliğleri (Elektrik Seksiyonu), TÜBİTAK, Ankara, 1978, s.165-171.

ÇALIKOĞLU, D. (1979) Mikrohesaplayıcılı Kontrol Sistemi, TÜBİTAK Projesi Kesin Raporu, No. MAG-G-456, TÜBİTAK, Temmuz 1979.

ÇALIKOĞLU, D. (1980a) "Mikrohesaplayıcılı Kontrol Sistemi", TÜBİTAK Doğa Bilim Dergisi, Seri-B, Cilt 4, Sayı 1, 1980, s.27-32.

ÇALIKOĞLU, D. (1980b) "Minicomputer Evaluated Microcomputer Programming", System Approach for Development. Third IFAC/IFIP/IFORS Conference, 24-27 Nov. 1980, Rabat, Morocco, Pergamon Press, s.457-460.

ÇALIKOĞLU, D. (1980c) "PDP/8 in Microcomputer Development Projects", 1980 DECUS Europe Symposium, 16-19 Eylül 1980. RAI Congress Center, Amsterdam, Hollanda, Meeting Program and Mini-Papers, s.137.

ÇALIKOĞLU, D. (1982a) "Microcomputer Development Interface" International Meeting on Development Aids for Microprocessors Systems, Liege, Oct.25th and 26th 1982, sayfa 3.1-3.4.

ÇALIKOĞLU, D. (1982b) PDP/8 İçin Bir Zamanlayıcı, Yayın No. 82-1, ODTÜ, Elektrik Mühendisliği Bölümü, Ankara, Temmuz 1982.

ÇALIKOĞLU, D. (1983) Mikrobilgisayarın Minibilgisayarla İşletimi, TüBİTAK Projesi Kesin Raporu No. MAG-538, Şubat 1983, Ankara.

MANO, M.M. (1982), *Computer System Architecture*, Prentice Hall (2nd Edition)