Murat Ertem

Mikroişlemci (2)

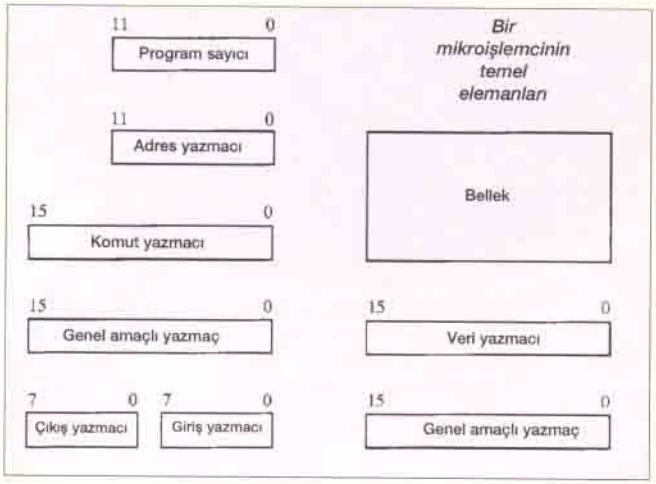
Bilgisayar denince, aklımıza ilk gelen sözcükler arasında yazılım ve donanım yer alır. Bu kelimeler genelde birbirlerinden bağımsız kavramlar olarak düşünülmektedir. Bilgisayar kullanıcılarının birçoğu bilgisayarın içindeki parçalardan çok kendi yazdıkları ya da satın aldıkları programların içeriğiyle ilgilenmektedir. Gerçekten de günümüzde yaygın olarak kullanılan ileri programlama dilleriyle istenilen uygulamanın yazılabilmesi için donanımla ilgili ayrıntılı bilgiye ihtiyaç duyulmamaktadır. Ancak bilgisayarın yapısı incelendiğinde yazılım ve donanımın ayrılmaz bir bütün olduğu görülür. Programlama hangi dilde yapılırsa yapılsın, bilgisayar donanımının daha doğrusu mikroişlemcinin anladığı tek şey makine kodlarıdır. Makine kodlarıysa ikilik sayı sisteminde ifade edilen 1'ler ve 0'lar dizisinden başka birşey değildir. Hiç şüphesiz mikroişlemcinin anladığı komutların her kullanıcı tarafından kodlanması büyük zorluklara neden olmaktadır. Nitekim ilk bilgisayarlarda programlar kartonlar üzerine kodlanırdı. Bugün kullandığımız programlama dilleriyse programların daha kolay yazılmasını sağlamaktadır. Ancak sonuçta bu programlar da mikroişlemcilerin anlayabileceği makine kodlarına çevirmektedir. Sonuçta mikroişlemcinin temelde yapabileceği işlemler makine kodlarıyla ifade edilebilir ve bu işlemleri mikroişlemcinin donanımı belirler.

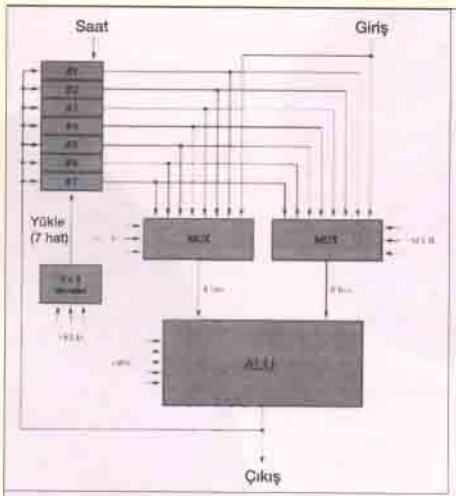
Bir mikroişlemcinin donanımı temel olarak yazmaçlardan (akümülatör ya da "register" da denmektedir) çeşitli göstergeleri belirten flip-floplardan aritmetik işlemleri gerçekleştiren mantık devrelerinden oluşur. Bunlara ek olarak, bir mikroişlemci içinde zamanlama ve kontrol devreleri de yer almaktadır. Mikroişlemcilerin donanımı, yazılımın da karakterini belirlediğinden, bu iki kavramın birarada incelenmesi, birinin diğeriyle olan ilişkisini daha açık ortaya koyacaktır.

Makine Kodları

Daha önce de belirttiğimiz gibi, makine kodları ikilik sayı sisteminde ifade edilen verilerdir. Bu verilerin yani programın belirli bir yerde saklanması gerekmektedir. Elektronik ortamda bu verilerin saklanabileceği tek yer bellektir. Hangi tür bellek kullanılırsa kullanılsın yapılan temel şey bu verinin hafızadan okunup, içeriğinin belirlenmesidir. Yazılan program bir bütün halindedir. Bu bütünün doğru çalışabilmesi için kendisini oluşturan komutların belli bir sırayla uygulanması gerekir. İşte programın bu düzenini korumak için bir yazmaç kullanılır. Bu yazmaca program sayacı adı verilmektedir. Program sayacı bir sonra yapılacak işlemi belirleyen komutun yerini yani adresini göstermektedir. Program sayacının içeriği VE'nin değillerinden oluşan bir devreden ya da bu devrenin bir entegre olarak sunulduğu elemanlardan geçer ve belirli bir bellek biriminin belli bir yerindeki verinin okunmasını sağlar. Bu sırada diğer verilerin okunması mümkün değildir. Yani her verinin sabit tek bir adresi vardır. İşte program sayacı bir sonra okunması gereken makine kodunun yerini göstermektedir. Her makine kodu bellekten okunduğunda program sayacı otomatik bir sonraki makine kodunu işaret eder. Böylece program, yazıldığı sıraya göre çalışır.

Her mikroişlemcinin anlayabildiği makine kodunun farklı bir yapısı vardır. Ancak temel olarak bir makine kodu birkaç belli bölümden oluşmaktadır. Makine kodunun uzunluğu yani hane sayısı ne olursa olsun, bir bölümü yapılacak işlemi, diğer bölümü de bu işlerde kullanılacak değişkenlerin ne olduğunu gösterir. Örneğin bir makine kodunun uzunluğunun 16 bit olduğunu düşünelim. Bu 16 bitin ilk 4 biti yapılacak işlemi, diğer bitlerse değişkenleri gösterebilir. Ancak 16 bit makine kodu kullanan başka bir bilgisayar daha değişik yapıda bir makine kodu kullanabilir. Örneğin tam ortada yer alan 3 bit yapılacak işlemi, diğer bitler de kullanılacak değişkenleri belirtebilir.





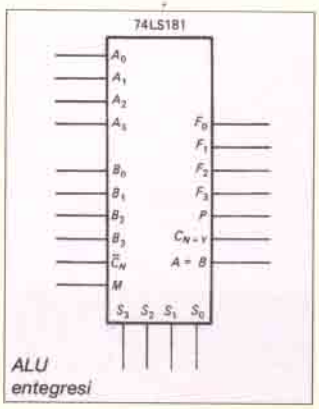
Şekilde en altta makine kodunun formatı görülmekte. OPR ile hangi işlemin yapılacağı belirtilmekte. Bu nedenle komutun OPR kısmı ALU'ya gönderilmekte. SELA ve SELB verileri ise işlemde hangi yazmaçların kullanılacağını; SELB ise işlem sonucunun hangi yazmaca konacağını belirtmekte.

Bellekten alınan makine kodu belirli bir yazmaca yerleştirilir. Daha sonra bu yazmacın içindeki bilgi MUX, VE'nin değili gibi elemanlarla oluşturulan mantık devrelerinden geçer ve istenen işlemi, istenen değişkenleri kullanarak yerine getirir. Herhangi bir işlem için kullanılan değişkenler farklı elemanlardan alınabilir. Örneğin bu değişkenler bellekten alınabilir. Bunun için kullanılacak değişkenin adresinin verilmesi gerekmektedir. Tıpkı program sayacında olduğu gibi bu adres, belirli elemanlardan geçerek belleğin adres girişlerine ulaşır. Ancak bu adrese ulaşmak için program sayacı kullanılamaz. Çünkü, program sayacı her zaman bir sonraki makine kodunu göstermek zorundadır. Eğer program sayacının içeriği değişirse, programın akışı da değişir ve doğru çalışmaz. Bu nedenle, verilere ulaşmak için ayrı bir yazmaç, adres yazmacı kullanılmaktadır. Adres yazmacı bellekte belirli bir yeri gösterdiğinde buradaki veri, veri yazmacına alınmaktadır. Ancak bazı mikroişlemcilerde veri genel amaçlı yazmaçlara da yerleştirilebilmektedir. Bir mikroişlemcide bu yazmaçlardan birden fazla bulunabilir. Bu yazmaçların içeriği yapılacak bir işlemde değişken olarak kullanılabilir Ayrıca bir işlemde kullanılacak değişkenler giriş çıkış için elemanlarından da alınabilir. Bu giriş çıkış elemanı, bilgisayarın klavyesi, bir disk, bilgisayar ağına bağlanmayı sağlayan elektronik bir devre ya da benzer amaçlar için kullanılan diğer aletler olabilir. Dışarıdan alınan ya da dışarıya iletilen verilerin akışı da giriş çıkış yazmaçları tarafından sağlanmaktadır.

Elemanların Seçimi

En sık kullanılan makine komutları arasında belirli bir yazmacın içeriğini diğer bir yazmaca geçirmek, iki yazmacın içeriğiyle çeşitli aritmetik işlemler gerçekleştirmek, bir yazmacın içeriğini belleğe yazmak ya da bellekteki verileri kullanarak çeşitli aritmetik işlemler yapmak yer alır. Peki, bu işlemlerin doğru şekilde yapılabilmesi için gerekli yazmaçlar ya da bellekler nasıl seçilmektedir? Bu yapıyı anlamak için bu elemanların yapılarını biraz incelemek gerekir.

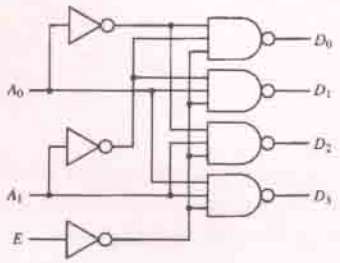
Daha önce belirttiğimiz gibi belleklerde adres girişi ve veri giriş çıkışı yer almaktadır. Bunun yanı sıra, bellek üzerinde işlem yapılmasını sağlayan belirli girişler vardır. Bellek üzerinde işlem yapılabilmesi için, bu girişe uygun değerin verilmesi gerekir. Bu girişe uygun sinyal girildikten sonra adres girişindeki değer bellek içinde istenen 1 baytlık veriye ulaşılmasını sağlar. Örneğin elimizde her biri 4 bayt bilgi taşıyan iki bellek olduğunu düşünelim. Her bir bellek 4 bayt bilgi içerdiğinden, bu belleklerin adres girişleri iki bitten oluşur. Çünkü, iki bitle 0 dan 3'e toplam 4 rakam ifade edilebilir. Ancak mikroişlemcimiz toplam 8 bayt belleğe sahip olacaktır. Yani adres yazmacı üç bitten başka bir değişle üç haneden oluşacaktır. Bu durumda veriler 000, 001, 010, 011, 100, 101, 110 ve 111 adreslerine sahip olacaktır. Bu adreslerden ilk dördü bir bellek, diğer dördüyse diğer bellek üzerinde yer alacaktır. Ayrıca belleklerin çalışmasını sağlayan girişe de 0 sinyali girilmesi gerektiğini düşünelim. Bu durumda, istediğimiz adreslemeyi yapmak için adres yazmacının en soldaki hanesini birinci belleğin çalışmasını sağlayan girişine direkt olarak bağlarız. Aynı hanenin değilini de diğer belleğin çalışmasını sağlayan girişe bağlarız. Geri kalan iki haneyi de her iki belleğin adres girişlerine bağlarız. Böylece adres yazmacında 000, 001, 010 ve 011 değerleri olduğunda en soldaki bit 0 olduğundan birinci bellek çalışacaktır. İkinci belleğin çalışmasını sağlayan girişineyse, 0'ın değili yani 1 ulaşacağından bu bellek çalışmayacaktır. En soldaki bit 1 olduğundaysa, tam ters bir durum oluşacaktır. Tabii günümüzde daha büyük bellekler-



ALU entegresi

Kod Çözücü

Kod çözücüler (decoder), dijital devrelerde adreslemede ya da elamen seçimini sağlayan devrelerde kullanılmaktadır. Kod çözücüler, n tane girişleri 2^n çıkışları olan elemanlardır. Amaçları girişlerindeki değere karşılık gelen çıkışa diğerlerinden farklı bir sinyal göndermektir. Şekilde görüldüğü gibi dört tane VE'nin değiline kod çözücünün her iki girişinden de değerler gönderilmektedir. Ancak her birinin girişine farklı değerler gitmektedir. VE'nin değili işlemi girişlerinin hepsi bir olduğunda çıkışına sıfır, diğer durumlarda da bir değerini vermektedir. Örneğin D_0 çıkışındaki VE'nin değiline A_0'ın değili, A_1'in değili ve E'nin değili girmektedir. Bu nedenle A_0, A_1 ve E sıfıra eşit olduğu zaman D_0 çıkışında sıfır görülecektir. Dikkat edilecek olursa kod çözücüdeki bütün VE'nin değil işlemlerine E'nin değili girmektedir. Burda E kod çözücünün çalışıp çalışmayacağını belirlemektedir.

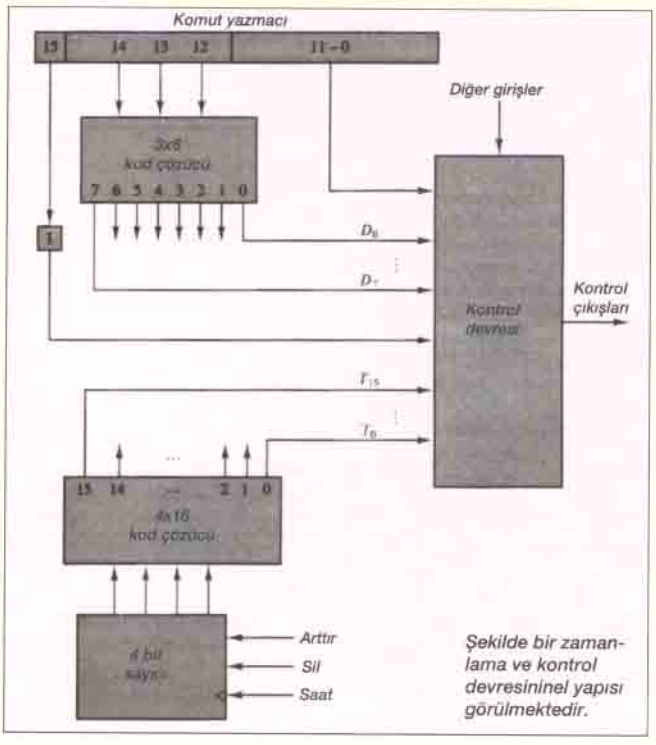


E	A_1	A_0	D_0	D_1	D_2	D_3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	×	×	1	1	1	1

Doğruluk tablosu

den birçoğu biraraya gelerek onlarca megabaytlık hafızalar oluşturulmaktadır. Bu durumda hangi belleğin ne zaman çalışacağını belirlemek için VE'nin değiliyle oluşturulan devreler ya da entegreler kullanılmaktadır. Makine kodlarının gerçekleştirdikleri işlemlerde kullanılan önemli diğer elemanlar da yazmaçlardır. İşlemcilerin doğru şekilde yapılabilmesi için uygun yazmaçların seçilmesi gerekmektedir. Her yazmacın üstünde yükle, temizle gibi girişler bulunmaktadır. Örneğin bir yazmaca herhangi bir veri yükleneceği zaman, yükle girişine uygun sinyal girilmelidir. Temizle girişine uygun değer verildiğindeyse, yazmaç 0'la doldurulmaktadır. Yazmaçların bu girişlerinin nasıl kullanıldığını bir örnekler açıklayalım. Mikroişlemcimizde 8 tane yazmaç olduğunu düşünelim. Bu durumda yazmaçların birbirinden ayırt edilebilmesi için 2^3 sekize eşit olduğundan, 3 bit yeterli olmaktadır. Bu durumda birinci yazmacın 000 değeri ile gösterildiğini düşünelim. Ayrıca makine kodunda yapılacak işlemin de 3 bitle gösterildiğini ve 111 değerinin herhangi bir yazmaca birşey yükleme işlemini gösterdiğini düşünelim. Bu durumda birinci yazmaca birşey yükleyeceğimiz zaman makine kodunun gösterdiği işlem 111 ve kullanılacak değişken de 000 olmalıdır. Ayrıca yazmaca birşey yüklemek için yükle girişine 1 girmek gerektiğini varsayalım. Demek ki işlem 111, değişken 000 olduğunda yükle girişine 1 girilmeli. Bunun için işlem bitleriyle, değişken bitlerinin değilini, VE işleminden geçirip yükle girişine bağlamamız gerekmektedir. VE işleminin çıkışının 1 olması için girişlerinin hepsinin 1 olması gerekmektedir. Bu örnekte işlem bitleri zaten 1'dir. Değişken bitlerinin de değilini aldığımızdan, VE işlemine 1 değerleri girilmektedir. Böylece birinci yazmaca herhangi birşey yüklemek için işlem bitlerinin 1, değişken bitlerinin de 0 olması gerekmektedir.

Mikroişlemcilerin yaptığı temel şey verilerin bir yerden alınıp başka bir yere yazılması ya da veriler üzerinde çeşitli işlemlerin gerçekleştirilmesidir. Sonuçta esas olan veriye ulaşmaktır. Bu nedenle bütün yazmaçların ve belleğin veri giriş çıkışları birbirine bağlıdır. Bu yapıya Türkçe'de yol (İngilizce "bus") denmektedir. Ancak bütün veri giriş çıkışları da belli bir sıraya göre çalışmalıdır. Örneğin bir yazmaca hafızadan bir şey yüklerken diğer yazmaçların veriye ulaşması gerekmektedir. Bunun için de sadece o işlem için kullanılacak yazmaçların ya da belleğin giriş çıkışlarının aktif duruma getirilmesi gerekmektedir. Bu da belirli devrelerin kullanılmasıyla mümkündür.



Şekilde bir zamanlama ve kontrol devresininel yapısı görülmektedir.

ALU

Geçen sayıda da değindiğimiz gibi ikilik sayı sistemindeki aritmetik işlemler mantık işlemleri kullanılarak gerçekleştirilebilir. Mikroişlemcilerde de aritmetik işlemleri gerçekleştirmek için oluşturulmuş mantık devreleri bulunmaktadır. Bu devrelerin tümü ALU (Arithmetic Logic Unit) altında toplanmıştır. Makine kodunda belirtilen işleme ve kullanılacak değişkenlere göre ALU'ya uygun sinyaller gönderilmektedir. Böylece ALU, istenilen elemanlar üzerinde makine kodunda belirten işlemi gerçekleştirmektedir. ALU aritmetik işlemleri yazmaçlar üzerinde gerçekleştirir. Bu yüzden ALU'nun uzunluğu, genel amaçlı yazmaçların uzunluğuna yani bit sayısına eşit iki girişi bulunmaktadır. Makine koduna göre gerekli yazmaçların içeriği ALU'nun girişlerine aktarılmaktadır. Bu da istenilen yazmaçların aktif hale getirilmesiyle sağlanmaktadır. Makine kodunda belirtilen işlem de ikilik sayı sisteminde bir sayıdır. Bu değerler çeşitli devrelerden geçtiğinde ALU içindeki gerekli elemanlarının çalışmasını sağlamaktadır. Böylece istenilen aritmetik işlem gerçekleştirilmektedir.

Zamanlama ve Kontrol Devresi

Bu noktaya kadar anlattıklarımızı toparlayacak olursak mikroişlemcilerin çalışma prensibini şöyle sıralayabiliriz: İlk olarak program sayacının gösterdiği adresdeki makine kodu alınır. Artık program sayacı bir sonraki makine kodunun yerini göstermektedir. Daha sonra hafızadan okunmuş olan kod incelenir. Böylece kullanılacak değişkenler öğrenilir. Daha sonra da istenilen işlem yerine getirilir. Bir sonraki makine kodu için aynı işlem tekrarlanır. Burdan da anlaşıldığı üzere bu işlemler belirli bir sırada gerçekleştirilmelidir ve sadece gerekli elemanlar aktif hale getirilmelidir. İşte bütün bu işlemler zamanlama ve kontrol devrelerinin görevidir.

Geçen sayıda değindiğimiz gibi dijital devrelerde elemanların eşzamanlı çalışması için bir saat bulunmaktadır. Kuvars kristalinden elde edilen saat sinyali bir kare dalgadır. Zamanlama devresine saat sinyali ulaştığında devredeki bir sayıcı çalışır. Bu sayıcının çıkışı o anda hangi evrenin gerçekleşeceğini belirler. Burada evre olarak kastedilen, makine kodunun okunması, kodun incelenmesi ve kodda belirtilen işlemin gerçekleştirilmesinden her biridir. Hiç kuşkusuz yapılan bütün işlemler aynı sürede gerçekleştirilememektedir. Bu yüzden bu sayıcı bazen gerekli değerle yüklenmelidir. Örneğin makine kodunun belirttiği herhangi bir işlem gerçekleştikten sonra bu sayıcı sıfırlanmalıdır.

Zamanlama devresi hangi evrede olduğunu belirledikten sonra kontrol devresi hangi elemanların çalışır duruma geçeceğini ya da elemanların hangi değerleri alacağını belirler. Bunun için kontrol devresi zamanlama devresindeki sayıcının değerini, makine kodunu ve diğer göstergeleri kullanarak çeşitli kontrol sinyalleri oluşturur. Bu kontrol sinyalleri, VE'nin değili gibi mantık işlemleri, MUX ve diğer entegre elemanların kullanılmasıyla oluşturulan dijital devrelerden elde edilen çıkışlardır. Bu sinyaller gerekli elemanların çalışır hale gelmesini sağlar ve istenilen fonksiyonları yerine getirir.

Bu yazıda mikroişlemcilerin çalışmasının temel prensiplerini inceledik. Bu yüzden mikroişlemcilerin gerçekleştirdiği işlemlerin sadece birkaçına değinmiş olduk. Ancak bu basit örnekler için gerekli devreleri oluşturmak bile, birçok karmaşık devrenin kullanılmasını gerektirmektedir. Günümüzde kullanılan işlemcilerin performansı düşünüldüğünde, ne kadar karmaşık bir yapıya sahip oldukları rahatça görülür. Fakat günümüzde geliştirilen yarı iletken teknolojisi bu devreleri küçük bir hacimde toplayabilmektedir. 486 tipi bir işlemci 8 santimetrekarelik 3 mm kalınlığındaki bir seramik plaka üzerinde 1,2 milyondan fazla transistör içermektedir. Bu da ne kadar karmaşık bir devreye sahip olduğunun göstergesidir.

Kaynaklar
Mano M. Morris, *Computer Architecture*, Prentice-Hall, 1993.
Hall V. Douglas, *Microprocessors and Interfacing*, McGraw Hill, 1986.

Düzeltme: 347. sayı, Elektronik Dünyası, sayfa 91 sütun 4'de, Diğer Elemanlar başlığı altında yer alan cümle "...MUX bir çıkışı, 2^n tane girişi ve n tane kontrolü olan bir elemandır. Basit anlamda bir MUX, 2^n tane..." olarak düzeltilecektir.