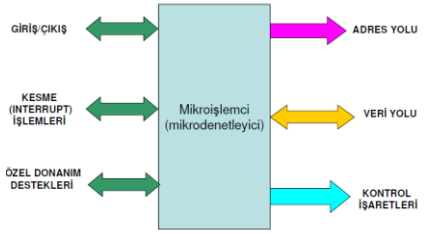
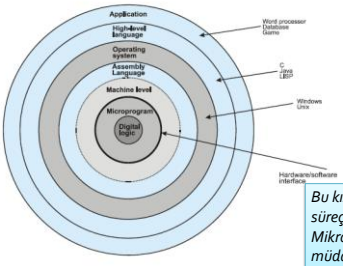
HAFTA 2

MİKROİŞLEMCİLER

Mikroişlemci

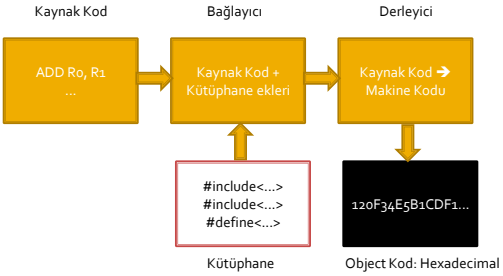


Mikroişlemcili Sistem Katmanları



Bu kısım donanımsal süreçleri gerçeklemektedir. Mikroişlemciyi kullananın müdahalesine açık değildir. Kullanan tüm özelliklerine hakim olmalıdır.

Mikroişlemci Kullanımı



Mikroişlemci

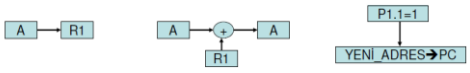
- Mikroişlemci, hafıza elemanları ve giriş-çıkış birimlerinden oluşan donanımsal yapıya mikroişlemcili sistem adı verilmektedir.
- Aslında sistem içindeki tüm bileşenler, mantıksal olarak saklayıcılardan meydana gelmektedir. Bütün süreçler, saklayıcılar üzerindeki bilgilerin birbirleri arasında taşınması veya belirli işlemleri gerçekleştirebilmek (örn. Toplama, çıkarma, kaydırma, ve, veya vb.) için mantık devreleri yoluyla işlem görmesiyle gerçeklenmektedir.
- Assembler'de oluşturulan kodlar, mikroişlemcinin bu "taşıma" veya "mantık devrelerini çalıştırma" süreçlerini gerçekleştirebilmesini sağlamaktadır.

Saklayıcı (Register)

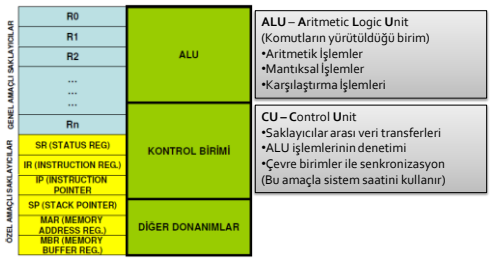
- Mikroişlemcinin üzerinde bulunan saklayıcılara dahili (internal) saklayıcılar, mikroişlemciye bağlanan çevre birimlerinin üzerindeki saklayıcılara ise harici (external) saklayıcı adı verilmektedir.
- Dahili saklayıcılar aritmetik-lojik işlemler ve getir götür işlemleri için kullanılırken, hafıza elemanları üzerindeki saklayıcılar kalıcı veya geçici depolama için kullanılmaktadır. I/O birimlerindeki saklayıcılar ise tamponlama veya çıkış işlemlerinde mikroişlemciden gelen veriyi tutma, giriş işlemlerinde ise dış dünyadan gelen bilgiyi tutma amacıyla kullanılır.
- Mikroişlemcili sistemleri oluşturan saklayıcılar topluluğu ve aralarındaki veri transferlerinin şekli sistem mimarisi olarak isimlendirilir. Sistem içindeki saklayıcıların çeşitleri ve transfer şekilleri, kullanılabilecek komut tipleri, komutların çalışma süresi, mikroişlemci mimarisini belirler.

Mikroişlemci İşlevi

- Saklayıcılar arası veri transferi
- Komutlar ile bir durumdan başka bir duruma geçiş
- Sınama işlemleri ve program yönlendirme

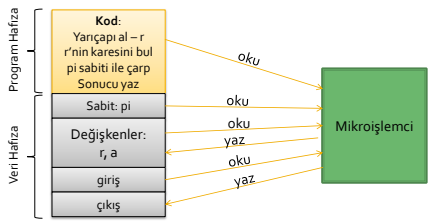


Mikroişlemci İç Mimarisi

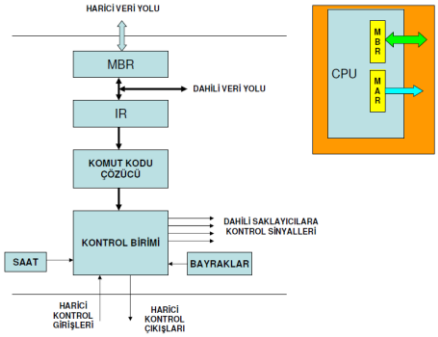


İşlemci-Harici Hafıza

Dairenin alanını hesaplayan bir kodu işlemcide çalıştırabilmek için



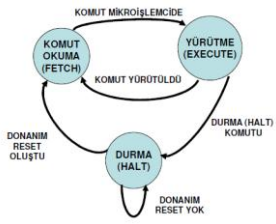
Mikroişlemci Dahili Saklayıcıları



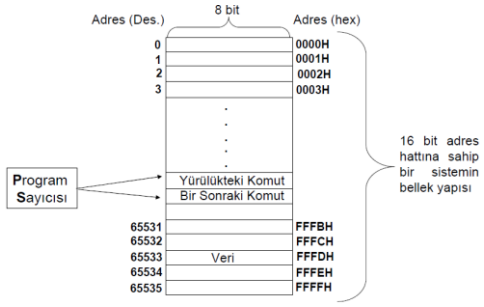
Mikroişlemcinin çalışması

- Bir mikroişlemcinin çalışması esnasında kontrol birimi tarafından yerine getirilen iki temel işlem vardır.
  - Komut Okuma (**FETCH**): Mikroişlemcinin program hafızadan (MBR ile) bir işlem kodu alıp komut saklayıcısına (IR) getirmesi işlemidir.
  - Komut Yürütme (**EXECUTE**): Komut saklayıcısına gelen komut ile hangi işlemin yapılacağı, yani hangi mantık devrelerinin aktif hale getirileceği komut kod çözücü tarafından belirlenir. Gerekli işaret üretimleri kontrol devresi tarafından gerçekleştirilir. Eğer işlemin gerçekleştirilebilmesi için veri veya verilere ihtiyaç varsa (örn. toplama), işlemin ihtiyaç duyduğu veriler, veri hafızasından getirilerek komutun yürütülmesine devam edilir.

Mikroişlemcinin Çalışması



Bellek



RTL – Register Transfer Language

- Bu dil, bilginin bellek ve mikroişlemcinin dahili saklayıcıları üzerinde nasıl erişildiğini ve işlem gördüğünüanlatmak amacıyla kullanılan bir gösterim şeklidir.
- [X]; X'iniçeriği
- ← ; Veri transferini belirtir. Sol kısım hedef, Sağ kısım kaynak
- [MAR]←[PC] ; PC'nin içeriği MAR saklayıcısının içerisine aktarılır.
- [3]←[5] ; 5. konumun içeriği 3. konuma aktarılır. (x=y; gibi)
- [PC]←[PC]+1 ; Program sayıcısının içeriğini 1 arttır.
- [M(x)] ; Belleğin x konumunun içeriği
- [M(20)]←[PC]; Belleğin x konumunun içeriği PC'nin içeriği ile yüklenir.

Komut Okuma (FETCH)

- CPU'nun bir komutu yürütmeden önce, bu komutu bellekten getirmesi gerekir.
- Program Counter(PC), bellekte yürütülecek bir sonraki komutun adresini içerir.
- Komutun getirilmesi için PC'nin içeriği Memory Address Register (MAR)'a aktarılır.
  - [MAR]←[PC]
- Ardından PC'nin içeriği bir arttırılır.
  - [PC]←[PC+1]
- MAR, yazma çevrimi veya okuma çevrimi sırasından üzerinde işlem yapılacak bellek konumunun adresini barındırır.

Komut Okuma (FETCH)

- Bu aşamada MAR, PC'nin artımdan bir önceki içeriğini yansıtır.
- Okuma çevrimi gerçekleştirildiğinde MAR ile belirlenmiş olan bellek konumunun içeriği okunup içeriği MBR(MemoryBufferRegister'e) aktarılır.
  - [MBR]←[M([MAR])]
- MBR okuma çevrimi sırasında bellekten okunan verilerin geçici olarak tutulduğu; yazma çevrimi sırasında da belleğe yazılacak verilerin geçici olarak tutulduğu bir saklayıcıdır.
- MBR'nin içeriği daha sonra IR (InstructionRegister) olarak adlandırılan saklayıcıya aktarılır.
  - [IR]←[MBR]

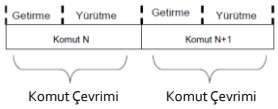
Komut Okuma (FETCH)

- IR'yegetirilen veri 2 parçadan oluşur. Bunlar;

- Op-code(İşlem Kodu):CPU'ya hangi işlemin gerçekleştirileceğini söyler.
- Operand(İşlenen): Op-codetarafından kullanıcak verinin adresini belirtir.
- Mikroişlemci içindeki komutlar birden fazla operanda sahip olabileceği gibi hiçbir operandı olmayan komutlarda mevcuttur.
- ControlUnit (**CU**), IR'dekiOp-code'ualır ve gerekli kontrol sinyallerini üretir.

Komut Okuma (FETCH)

- Programın yürütümüsırasında, mikroişlemci, her komutu sırayla bellekten getirir, komutla belirtilen işi belirler ve bu işi yapar (yürütür). Bu yüzden mikroişlemci 2 fazlıolarak çalışır.

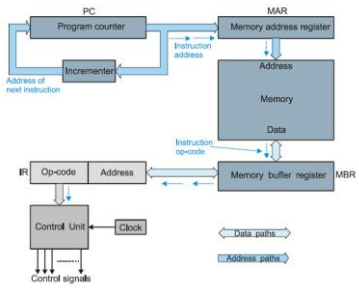


Komut Okuma (FETCH)

FETCH[MAR] ←***[PC]***; PC'nin içeriği MAR'akopyalanır.
[PC] ←***[PC]+1***; PC'nin içeriği 1 arttırılır.
[MBR] ←***[M ([MAR])]***; Komut bellekten okunur.
[IR] ←***[MBR]***; Komut IR'ye aktarılır.
CU←***[IR (op-code)]***; Op-code kontrol birimine aktarılır.

- Getirme fazında, komut bellekten okunur ve kontrol birimi (**CU**) tarafından kodu çözülür.
- Yürütme fazında kontrol birimi, komutun yürütülmesi için gerekli olan sinyalleri üretir.

Komut Okuma (FETCH)



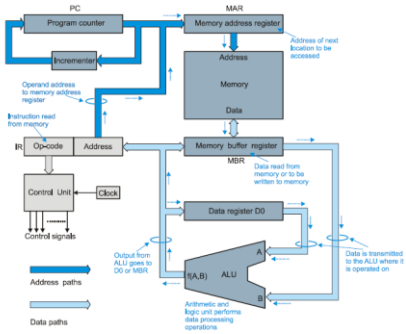
ADD işlemi

Örn: P = A + Ro işleminin gerçekleştirilmesi

…
ADD A, Ro; Bellekteki R konumunun içeriğini Akümülatöre ekle.
…

ADD A, Ro; satırının RTL ile ifadesi aşağıdaki gibidir;
FETCH[MAR] ←[PC]; PC'nin içeriği MAR'a kopyalanır.
[PC] ←[PC]+ 1; PC'nin içeriği 1 arttırılır.
[MBR] ←[M ([MAR])]; Komut bellekten okunur
[IR] ←[MBR]; Komut IR'ye aktarılır.
CU←[IR (op-code)]; Op-code kontrol birimine aktarılır.
ADD[MAR] ←[IR(address)]; Operand MAR'a aktarılır.
[MBR] ←[M([MAR])]; A konumunu bellekten oku
[ALU] ←[MBR];[ALU] ←[Ro]; Ekleme işlemini gerçekleştirir.
[Ro] ←[ALU]; Sonuç Ro'a yazılır.

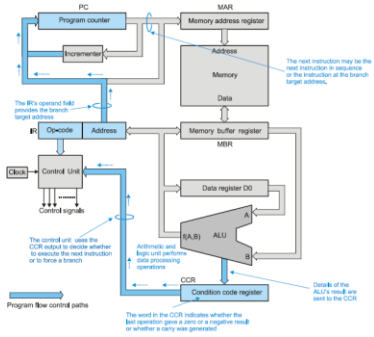
ADD işlemi



Şartlı Komutların Yürütülmesi

- Dallanma (Branch) ve Sıçrama (Jump) CPU'yu komutların normal icra sırasının dışına çıkmasına zorlayan komutlardır. Branchlar belli bir şarta göre gerçekleştirilirken, Jump'lar için şart söz konusu değildir. (IF ….ELSE….) veya (GOTO)
- ConditionCodeRegister (CCR), her bir komutun işletilmesinden sonra ALU'nun sonuç değerine göre şekil alır.
- Branchkomutu yürütüleceğinde CU(ControlUnit) CCR'nino anki durumunu sorgular. Sorgulama sonucuna göre, yakomutların sırayla işletilmesine devam edilir yada bellek üzerinde farklı bir bölgeye dallanılarak komutları yürütülür.

Şartlı Komutların Yürütülmesi



Şartlı Komutlar