**Chapter 3**

**Bilgisayar Fonksiyonu ve Ara Bağlantısının Üst Düzey Görünümü:**

**Bilgisayar Bileşenleri:**

* Çağdaş bilgisayar tasarımları, John von Neumann tarafından geliştirilen mimariyi esas alır. Bu mimarisi yapısında 3 temel kavram vardır. İlk olarak veri ve emirler tek bir okuma-yazma belleğinde saklanır. Önceki dönemlerde emirler elektronik kartlarda kodludur. Farklı emirler için farklı kartlar kullanılırdı. Günümüzde donanımsal değişiklik yapılmadan sadece yeni emirler yazılarak kodlama yapılabiliyor. İkinci olarak belleğin içeriği, veri tipine bakılmaksızın, adreslenebilir. Üçüncü olarak Program icrası, bir emirden diğerine sırayla (jump gibi atlamayı sağlayan emirler olmadıkça) gerçekleşir.
* **Hardwired Program (Fiziksel Bağlantılı Program):** Kapılardan oluşan elektronik devredir.

**Hardware and Software Approaches (Donanım ve Yazılım Yaklaşımları):**

|  |  |
| --- | --- |
|  |  |
| **Donanımla Programlama** | **Yazılımla Programlama** |

**Yazılım:** Bir dizi kod veya emirlerdir. Donanımın bir kısmı her emiri yorumlar ve kontrol sinyalleri üretir.

* **Ana bileşenleri:**
* **CPU**
* **I/O Bileşenleri**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Memory Address Register (MAR) (Bellek Adres Kaydedicisi):** Bir sonraki okuma veya yazma için bellekteki adresi belirtir.  **Memory Buffer Register (MBR) (Bellek Tampon Kaydedicisi):** Belleğe yazılacak verileri içerir veya bellekten okunan verileri alır.  **I/O Address Register (I/O AR) (G/Ç Adres Kaydedicisi):** Hangi cihazdan okunacağını belirten kaydedicidir.  **I/O Buffer Register (I/O BR) (G/Ç Tampon Kaydedicisi):** Verinin okuma veya yazma yerini tutan kaydedicidir.  **Not:** I/O AR ve I/O BR işlemci ile cihaz arasında arayüzlerdir. | |
|  | |
| **Temel Emir Döngüsü** | |
| Programlar koşulurken hard diskten ana belleğe alınır. Emir, kaba fetch next instruction (bir sonraki emri getirme) aşamasında instruction register’a (emir kaydedicisi) getirilir. Execute instruction (İcra emri) aşamasında emir icra edilir. Execute instruction aşamasında emre birçok kez erişebilir. Bu durum emrin türüne göre değişir. Emrin icrasında bellek referansı gerekebilir. Örneğin LOAD emri varsa bu durumda 1 operand vardır ve 2 bellek erişimi gerekir. Halt (Dur) emrine göre aşamalar tekrar edilir. | |
| **Fetch Cycle (Getirme Döngüsü):** Her emir döngüsünün başında işlemci bellekten bir emri getirir. Program sayıcısı (PC), bir sonraki alınacak komutun adresini tutar. İşlemci, her emrin alınmasından sonra PC'yi sırayla bir sonraki emri getirecek şekilde arttırır. PC’nin ne kadar arttırılacağı belleğin organizasyon yapısına göre değişir. Yani bellek adresi (bellek satırı) 16 bit veya 8 bit olabilir. PC’nin içeriği ne kadar artacağı bu durumdan etkilenir. Bellek adresi 16 bitse 1.PC 1 artar. 2.PC 2 artar. JUMP komutu PC’nin içeriğini günceller. Ardından getirilen emir, emir kaydedicisine (IR) yüklenir. İşlemci emri yorumlar ve gerekli eylemi gerçekleştirir. | |

**Action Categories (İşlem Kategorileri):**

* **Opcode (İşlem Kodu):** Emrin gereğini belirtir.
* **Operand (İşlenen):** Emrin yerine getirilebilmesi için gerekenleri bildirir.

**Opcode (İşlem Kodu) kategorisinin 4 çeşidi bulunur.**

1. **İşlemci-Bellek:** Veriler işlemciden belleğe veya bellekten işlemciye aktarılır. LOAD-STORE vb. komutlar bu işlemi sağlar.
2. **İşlemci-I / O:** Veriler işlemciden I/O cihazına veya I/O cihazından işlemciye aktarılır. IN-OUT vb komutlar bu işlemi sağlar.
3. **Kontrol:** Bir emir, icra sırasının değiştirileceğini belirtebilir. JUMP vb. komutlar bu işlemi sağlar.
4. **Veri İşleme:** İşlemci, veriler üzerinde bazı aritmetik veya mantıksal işlemler gerçekleştirebilir.

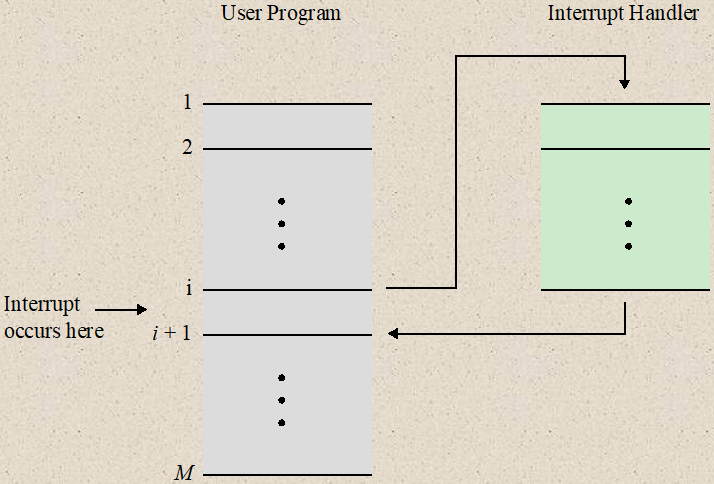
|  |
| --- |
|  |
| **Emir Formatı** |
|  |
| **Dâhili CPU Kaydedicileri** |
|  |
| **Opcodes (İşlem Kodu) Listesi** |
| 16 bit uzunluğundaki emrin 4 bit opcode, 12 bit operandı vardır. farklı emir olabilir. Her emir için farklı bir operand gerektirir. tane farklı adresleme yapılabilir. |

|  |
| --- |
| **Program İcrası Örneği** |
|  |

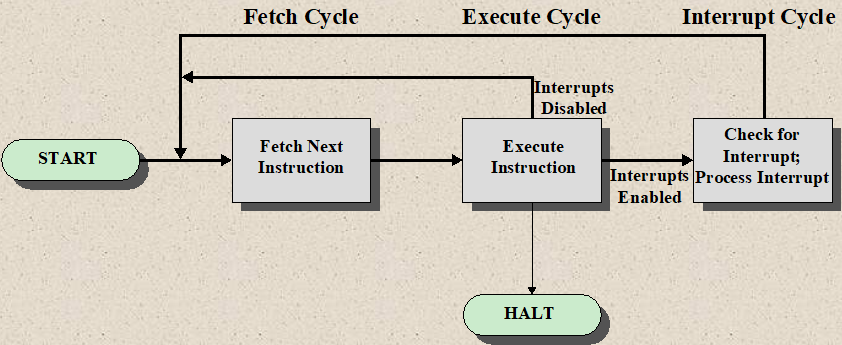
|  |  |
| --- | --- |
|  | |
| **Emir Döngüsü Durum Şeması** | |
|  |  |
| Kesmesiz sistem | |

|  |  |
| --- | --- |
|  |  |
| Kısa I/O beklemeli kesmeli sistem | |

|  |  |
| --- | --- |
|  |  |
| Uzun I/O beklemeli kesmeli sistem | |



Emir icrası, fetch (getirme) ve execute (icra) kısımları kesme gelse de bölünmez. Emir icra edildikten sonra kesme servis programına geçilir ve bitince program kaldığı yerden devam eder. Aynı yerden devam etmesi için emrin içeriği PC’nın içinde saklanır. İlave olarak işlemci AC gibi özel olan kaydedicilerde emrin içeriğini saklayabilir. Bilgisayarın ana belleğinde PC’nin içeriği, işlemci içerisindeki kaydedicilerin tümünün kopyası o yığına itilir. Daha sonra o yığından çekilir ve kesmeden önce devam eden işleme dönüşür. PC haricinde ara değer varsa o da saklanır. Kesme servis programı süreci kendiside kesebilir. Bunlarda yığına itilir



Kesmeli giriş çıkışta temel emir döngüsüne, interrupt (kesme) döngüsü eklenir. Her emir sonrası işlemci kesmeyi kontrol eder. İşlemci, kesmeyi aktif ettiyse bu kesme kontrolü gerçekleşir. Kesme aktif değilse bu kontrol yapılmaz.

|  |
| --- |
|  |
| Kesmeli Emir Döngüsü Durum Şeması |

**Çoklu Kesme:**

|  |  |
| --- | --- |
|  |  |
| Sıralı Kesme İşlemi | İç içe Kesme İşlemi |
| Bu durumda tüm cihazlar eşit öncelikli ve kesme sırasına göre cevap veriliyor. Bu günümüzde mümkün değildir. |  |

|  |
| --- |
|  |
| Çoklu Kesme Örneği |

**I / O Fonksiyonu:**

G / Ç modülü, doğrudan işlemci ile veri alışverişi yapabilir. İşlemci, verileri bir G / Ç modülünden okuyabilir veya yazabilir. İşlemci, belirli bir G / Ç modülü tarafından kontrol edilen belirli bir aygıtı tanımlar. İşlemci bir I / O modülüne bellekten okuma veya yazma yetkisi verir, böylece I / O bellek aktarımı işlemciye sorulmadan gerçekleşebilir. G / Ç modülü, değiş tokuşun sorumluluğunu azaltan belleğe okuma veya yazma komutları verir. Bu işleme doğrudan bellek erişimi (DMA) denir.

|  |  |  |
| --- | --- | --- |
| İşlemci (CPU), bellek ve I/O bilgisayarı oluşturan bileşenlerdir. Bu bileşenler birbiriyle sistem yolu (network) sayesinden haberleşir. | | |
|  |  |  |

**Arabağlantı yapısı aşağıdaki transfer tiplerini desteklemelidir:**

* **Bellekten İşlemciye:** İşlemci, bir emir veya bellekten bir veriyi okur.
* **İşlemciden Belleğe:** İşlemci bir veriyi belleğe yazar.
* **I/O’dan İşlemciye:** İşlemci, bir G / Ç aygıtından bir G / Ç modülü aracılığıyla veri okur.
* **İşlemciden I/O’ya:** İşlemci verileri G / Ç aygıtına gönderir.
* **I/O’dan Belleğe veya Bellekten I/O’ya:** Bu tip transferde araç olarak CPU kullanılır. Bir G / Ç modülü, doğrudan bellek erişimi kullanarak işlemciye uğramadan veriyi doğrudan bellek ile değiştirebilir.

**Bus Interconnect (Arabağlantı Yolu):**

İki veya daha fazla cihazı birbirine bağlayan bir iletişim yolu vardır. Temel özelliği paylaşılan bir iletim ortamı olmasıdır. Herhangi bir cihaz tarafından iletilen sinyaller, veri yoluna bağlı diğer tüm cihazlar tarafından alınabilir. İki cihaza aynı anda veri iletilirse, sinyalleri üst üste gelir ve bozulur. Genellikle çoklu iletişim hatlarından oluşur. Her bir hat, 1 ve 0’ı temsil eden sinyalleri iletebilir. Bilgisayar sistemleri, bileşenler arasında bir dizi farklı veri yolu içerir. Büyük bilgisayar bileşenlerini (işlemci, bellek, G / Ç) bağlayan bir veri yoluna sistem yolu denir. En yaygın bilgisayar arabağlantı yapıları, bir veya daha fazla sistem veri yolunun kullanımına dayanmaktadır. Bu sistem genellikle gömülü sistemlerde kullanılır.

|  |
| --- |
| **Veri Yolu:**  Sistem modülleri arasında veri taşımak için bir yol sağlayan veri hatlarına denir. 32, 64, 128 veya daha fazla ayrı hattan oluşabilir. Hat sayısı, veri yolunun genişliği olarak adlandırılır. Hat sayısı, bir seferde kaç bitin aktarılabileceğini belirler. Veri yolunun genişliği, genel sistem performansını belirlemede önemli bir faktördür. |
| **Adres Yolu:**  Veri yolundaki verilerin kaynağını veya hedefini belirtmek (yazma veya okuma) için kullanılır. İşlemci, bellekten bir veri kelimesini okumak isterse, istenen kelimenin adresini adres hatlarına koyar. Genişliği sistemin maksimum olası bellek kapasitesini belirler. Ayrıca G / Ç bağlantı noktalarını ele almak için kullanılır. Yüksek sıra bitleri, veri yolu üzerinde belirli bir modülü seçmek için kullanılır ve alt sıra bitleri, modül içindeki bir bellek konumunu veya G / Ç bağlantı noktasını seçer. |
| **Kontrol Yolu:**  Veri ve adres hatlarının erişimini ve kullanımını kontrol etmek için kullanılır. Veri ve adres hatları tüm bileşenler tarafından paylaşıldığından, bunların kullanımını kontrol etmek için bir araç olmalıdır. Kontrol sinyalleri, sistem modülleri arasında hem komut hem zamanlama bilgilerini iletir. Zamanlama sinyalleri, verilerin geçerliliğini ve adres bilgisini gösterir. Komut sinyalleri gerçekleştirilecek işlemleri belirler. Adres üretebilen CPU’dur. Ancak DMA olursa |
|  |

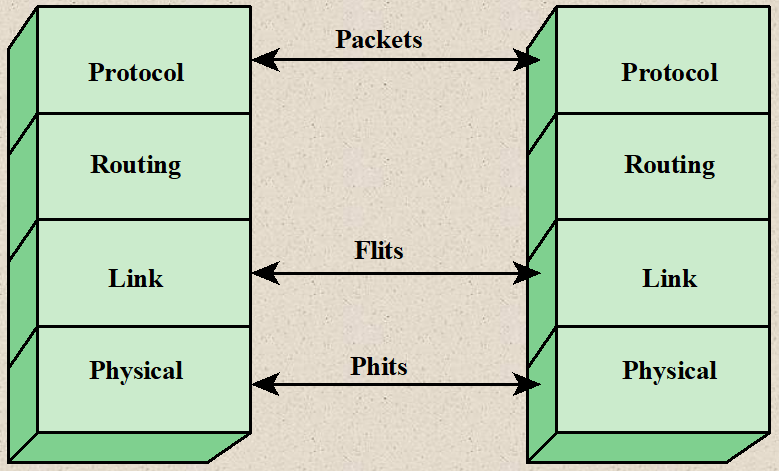
**Point-to-Point Interconnect (Noktadan Noktaya Arabağlantı):**

Arabağlantı yoluna göre daha yüksek hızlı veri iletimi yapılır. Paylaşılan, onay alınması gereken gecikmeyi arttıran bir ortam yoktur. Her model çifti arasında ayrı bağlantı vardır ve çift yönlüdür. Sistemin ölçeklenebilirliği çok iyidir. Sistemde fazla I/O, çekirdek vardır. Bileşen sayısı arttığında arabağlantı yolu adapte olamaz. Noktadan noktaya arabağlantıda ayrı ayrı iletişime geçildiğinde ölçeklenebilir.

**Quick Path Interconnect (QPI) (Hızlı Yol Arabağlantısı):**

|  |
| --- |
|  |
| * İletim sistemlerinde bulunan uzlaşma ihtiyacını ortadan kaldıran diğer bileşenlerle doğrudan çift yönlü bağlantılar vardır. * Bu işlemci türlerinde arabağlantılar, veri yolu düzenlemelerinde bulunan kontrol sinyallerinin basit kullanımı yerine katmanlı bir protokol mimarisi kullanır. * Veriler, her biri control headers (kontrol başlıkları) ve error control codes (hata kontrol kodlarını) içeren paketlerle birbiriyle haberleşirler. Yani synı anda birden çok veri arasında eş zamanlı olarak veri alışverişi olur ve bu yolda dar boğaz olmaz. Her bileşen haberleşmesinde bileşenler protokol yığınına koşmaktadır. * Memory bus (hafıza yolu)’da veriler paketlenme yapılmadan bit bit iletilirler. Her çekirdeğin kendi belleği vardır. Ancak sistemin ortak paylaştığı bellek de olabilirdi. * Paket transferinin sistem yolundan farkı, veri gönderilecekse bit bit gönderilir. Stream moduyla gönderilirse bitler bitene kadar gönderilir. Paket moduyla gönderilirse bitleri çerçeve oluşturarak paket yapılıp karşıya gönderilir. Örneğin 10 bit gönderilmek istenirse 10 bitin önüne ve arkasına ilave bitler koyularak bir çerçeve oluşturulup paket iletimi sağlanır. QPI, paket yöntemiyle ve memory bus, stream yöntemiyle veri iletimi sağlar. |

**Quick Path Interconnect Layers (Hızlı Yol Arabağlantısı Katmanları):**



|  |
| --- |
|  |
| **Physical Interface of the Intel Quick Path Interconnect (Intel Hızlı Yol Arabağlantısının Fiziksel Arayüzü)** |
| Quick Path Interconnect Layers (Hızlı Yol Arabağlantısı Katmanları)’ın fiziksel katmanıdır. Fizksel katman port demektir. Oklar A bileşeni ile B bileşeni arasındaki tellerdir. Teller linkleri (bağlantı) temsil ederler. Şekle bakarsak A’dan B’ye 44 iletken tel vardır. 40 tane iletken tel bit iletimi yapar. 40 tane telden 20 tane bit iletilebilir. Çünkü 1 tel bit iletimi sağlarken diğer bir tane telde onun etrafına sarılarak korumayı sağlar. Geriye kalan 2 tane tel clock (saat) biti görevini yapar. Yani genel olarak A’dan B’ye veya B’den A’ya paralel olarak 20 bit iletimi yapılabilir. Bu iletilen bitlere phits denir. |

**Örnek:**

|  |  |
| --- | --- |
|  |  |
| * A çekirdeğinin belleğindeki 160 bayt protocol (iletişim kuralı) katmanında paketlere bölünüyor. Her paket 16 bayta bölünsün. Yani 80 bitlik paketler oluşturuldu. * Bu 80 bit alt katman olan routing (rotalama)’e iletiliyor. Bu katmanda routing tablosu (Aşağıda gösterilmiştir.) oluşturuluyor. Routing katmanındaki paket bir alt katman olan link (bağlantı) katmanına iletiliyor. İletim şu şekilde gerçekleşir. A çekirdeği C çekirdeğine paketi göndermeden önce istekte bulunur. C çekirdeği gönderebileceğini dedikten sonra A çekirdeği paketi gönderir. Bu olaya akış kontrolü denir. A çekirdeğinden C çekirdeğine 10 paket gönderilebiliyor. Bu değer tampon boyutuyla belirlenir. İçerdeki sayaç bunun bilgisini tutar. * Link katmanında 16 bayt iki parçaya bölünüyor (8 bayt+8 bayt). Yani 80 bit iki parçaya bölünüyor. Bu parçalara flits denir. 80 bitlik fllits’in 72 bit’i veri veya protokol mesajıdır. 8 biti CRS kodudur. İletim esnasında bitler bozulabilir. Alıcıda da aynı katmanlı yapı vardır. 80 bit karşı tarafın link katmanında 80 bit olarak açılacaktır. Bozulup bozulmadığını anlamak için CRS kodu denilen değer olarak hesaplanır. Eğer C çekirdeğinin tamponu dolduysa link katmanından fiziksel katmana iletim olmaz. Veriler işlendikçe tamponda yer açılır. * Bir alt katman olan fiziksel katmana 20 bit gönderilebilir. 4x20 bit bitince diğer 8 baytlık (80 bitlik) flits fiziksel katmana gönderilir. 80 bitlik flits’i 72+8’e ayırır. Ardından fiziksel katmanda elde edilen 8 bitlik CRS kodu ile önceden link’e gelen 8 bitlik CRS kodu karşılaştırılır. Eğer aynı ise bilgi bozulmamıştır. Eğer aynı değilse başka bir protokol devreye girip C’den A’ya ekstra protokol içeren flits yollar ve paketi tekrar ister. | |

|  |  |
| --- | --- |
| **Routing (Rotalama) Tablosu** | |
| C | C |
| D | D |
| B | C-D |

|  |
| --- |
|  |
| **Quick Path Interconnect Multi Lane Distribution (Hızlı Yol Arabağlantısı Çok Şeritli Dağıtım)** |
| Fiziksel katmana gelen 80 bit bir donanımla ayrılıyor. Tek bir flit için bu sütunlardan 4 tane vardır. Fiziksel katmanda bölünen 20 bitlik verilerde CRC kodu özel olarak ayrılmaz. |

**Örnek:**

|  |  |
| --- | --- |
| Paket🡪 20 bayt | Flit🡪 10 bayt |
| Phit🡪2,5 bayt | 1 pakette 2 flit vardır. |
| 1 flit 4 aşamada gerçekleşir. | Toplam 8 paket (160 bayt) olduğu için 8x8=64 aşamada gerçekleşir. |

**Peripheral Component Interconnect (PCI) (Çevresel Bileşen Arabağlantısı):**

|  |  |
| --- | --- |
|  | PCI, bus (sistem yolu) teknolojisini kullanır. Yüksek hızlı değildir. I/O cihazlarının yüksek hızını sistem yolundan dolayı destekleyemez ve dışarıdan aldığını CPU’ya hızlı iletemez. O nedenle Peripheral Component Interconnect Express (PCIe) (Çevresel Bileşen Arabağlantısı Hızlı) devreye girer. PCIe noktadan noktaya arabağlantısını kullanır. Daha yüksek hızlarda cihazın veri okuyup yazabilmesini sağlar. QPI’daki gibi katmanlı yapı kullanımı vardır. Ancak 3 katmana sahiptir. Routing katmanı yoktur. Çünkü hedef zaten tek olduğu için roting’e ihtiyaç yoktur. |

|  |
| --- |
|  |
| **Peripheral Component Interconnect Express (PCIe) (Çevresel Bileşen Arabağlantısı Hızlı) Protokol Katmanları** |

|  |  |
| --- | --- |
|  | * Cihaz çekirdeğe gidecek veya cihaz DMA ile belleğe erişecektir. Arada bazı Chipset (yonga), switch, PCI bridge gibi donanımlar vardır. * Chipset PCIe’i kullanarak iletilecek verinin tamponlanmasını sağlar. Çekirdekteki bilgi cihaza gidecekse gönderilen hızla, alıcı cihaz bilgiyi alabilecek mi, işleyebilecek mi, gibi sorunlarını çözüp hız uyumunu sağlar. * Çekirdeğin cihaza gönderebileceği çok farklı bilgi vardır. Örneğin çekirdek Ethernet kartına veri gönderebilir. Günümüzde veriler sınıflara ayrılıyor. Bunlar gerçek zamanlı ve gerçek zamanlı olmayanlar olarak ikiye ayrılıyor. Gerçek zamanlı olan daha hızlıdır. Bir video (video stream) açılıyor birde FTP (bilgisayardan bilgisayara dosya aktarımı) yapılıyor. Bunların ikisi de Ethernet kartında yapılır. Switch gelen verilere bakıyor ve yüksek öncelikli olanlara etiket konularak daha yüksek öncelik kazanılmasını sağlıyor. Yani video stream verisine etiket koyulur ve cihaza video stream yazılıyor. Böylece normal metin dosyasının transferinden ayrılıyor. * PCI Bridge’de PCIe’yi desteklemeyen cihaz varsa ve sistem destekliyorsa cihazın kullanabilmesini sağlar. |