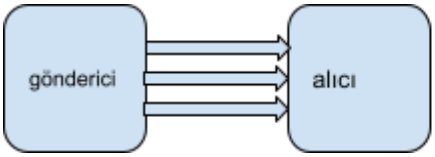
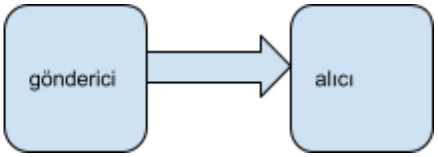
Yol nasıl kullanılır?

Yolun iki ucu arasında iletişimin farklı yolları vardır.

1.) Koşut(-ing paralel) aktarım
2.) Ardışık(-ing serial) aktarım

paralel aktarım çok şeritli yollar gibi düşünülebilir. ardışık aktarım bir yolda tek şerit olması gibi düşünülebilir.





Paralel gönderirsek bant genişliği fazla olur bundan dolayı bir defada daha fazla veri aktarılır. Ama maliyet artmış olur. seri gönderirsek maliyet düşük olur ama bir defada 1 bit gönderilebilir. Paralel aktarım hızlıdır ve daha kısa yollar için uygundur. Seri aktarım ucuz ama yavaştır. ASCII karakter 1 bayttır. Bir klavye kullanırken 1 saniyede en max 20 tuşa basılıyor desek 20 bayt veri aktarmak gerekecek ki bu çok ütopik olarak düşünülmüş bir rakamdır. 20 bayt/sn bilgisayarlar için çok düşük bir veri aktarım hızıdır. Bu gibi aygıtları seri olarak bağlayabiliriz. GPU'da işlem yapacak olursak çok çok daha büyük veri aktarım hızlarına ihtiyaç olacaktır.

Her aygıtın veri gönderim hızı, sıklığı farklıdır bu nedenle farklı arayüzlere ihtiyaç duyarlar. Gönderim hızı farklılıklarından dolayı yol tasarımında cihazları atama yaparken yani hangi cihazın daha önce görevini yerine getireceğine karar verirken bunları göz önüne almak gerekiyor. Hızlı aygıtlar için Kuzey köprüsü, yavaş aygıtlar için Güney köprüsü diye terimler vardır.

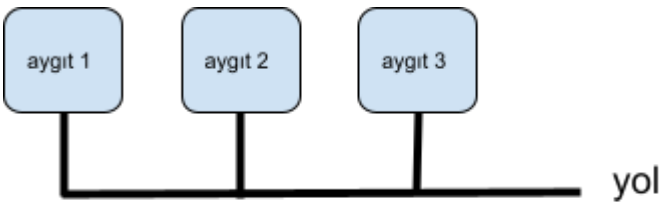
Aygıtlar bir yola bağlandığında yol üzerinde iki tür aygıt olur.

1.) Ana aygıt(-ing. master)
2.) Uydu aygıt(-ing. slave)

Ana aygıt her zaman iletişimi başlatan aygıttır. Bir yolun üzerinde birden fazla ana aygıt bulunabilir. Her zaman iletişimi ana aygıt başlatır ve uydu aygıtlar yanıt verirler.

Yol Atama Yöntemleri

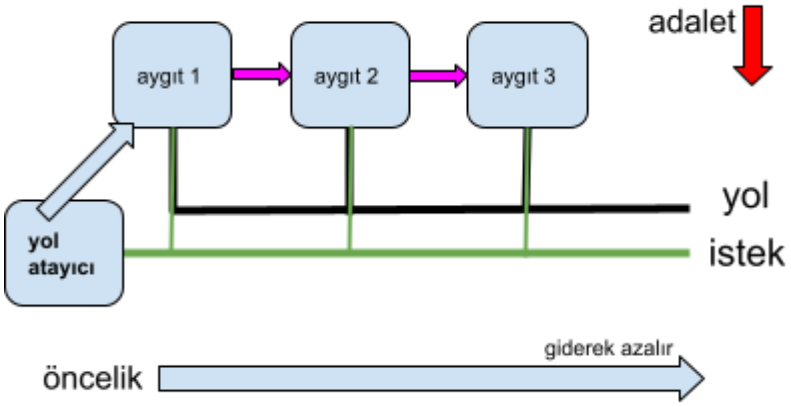
Birden fazla aygıtın paylaştığı bir yol ortaya konuyorsa ilk sorulardan biri ilk hangi aygıt bu yolu alacak. Hangisi önce alacak? Ne kadar bekleyecekler?



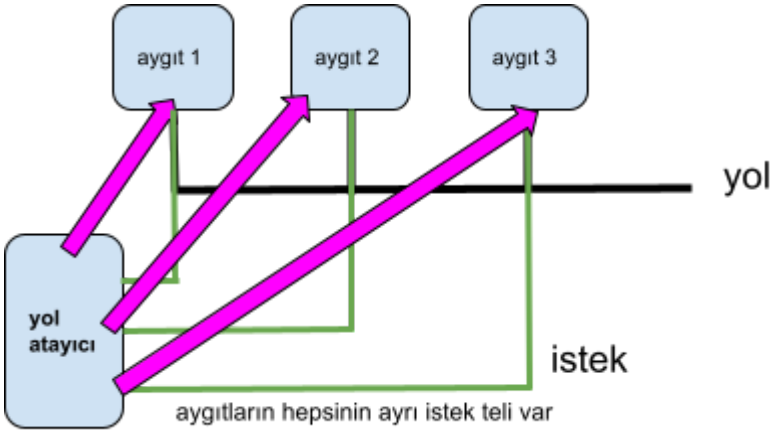
Burada iki şeye bakılır bir **öncelik**, iki **adalet**. Öncelik kaçınılmaz bir şeydir fakat öncelik yüksek tutulursa adalet varsayımı sağlanamayabilir. Yurt dışından büyük bir cumhurbaşkanı geldiğinde yollar onun için kapatılabilir. Bu kapama 4-5 saat gibi uzun bir süre olursa vatandaşlar için sorun olur ve adalet duygusu sarsılabilir. Öncelik vazgeçilmez bir şeydir fakat bunun yanında adaleti de sağlamak gerekir. ***Yol atayıcı:*** yolun başına yolu kimin alacağına karar veren yapıdır. Yukarıdaki verdiğimiz örnekte trafik polisinin olması gibi.

Aygıtlar yol atayıcıya istekte bulunurlar. Yol atayıcı da kullanıp kullanmamalarına karar verecek. Bu işi yapmanın en kolay ve ucuz yöntemi papatya zinciri yol ataması yöntemidir.

Papatya Zinciri Yol Ataması: tüm aygıtlar yan yana dizilip aynı istek teline bağlanırlar. Yol atayıcı bu istek teline bakar. Tek istek teli vardır ve yol atayıcı önceden belirlenmiş bir öncelik sırasında göre atar. Yol atayıcının bildiği tek şey aygıtların ondan yol istemesidir. Yol atayıcı her zaman öncelik sırasında birinci olan aygıta verir, birinci aygıt istemediyse ikinci aygıta verir. birinci istemese bile ilk ona verir daha sonra diğerlerine doğru gider. Ayrıca aygıtların birbirine yol geçirebiliyor olması lazım. Kendisi kullanmayacaksa başkasına paslamasını sağlamak için. Donanımda sabit bir öncelik sırası vardır bu nedenle karmaşıklığı ve maliyeti düşüktür. Sona doğru gittikçe aygıtların yolu alacağının garantisi azalır. Aygıt sayısı sonsuza giderken sonuncu aygıtın alma ihtimali sıfıra yaklaşır. Öncelik sona doğru gidildikçe azalır. Adalet düşüktür. Adaleti sağlayacağına dair garantisi yoktur.



Merkezi Yol Atama: 2.yöntem ise her aygıta ayrı bir istek teli vermektir. Yol atayıcı hepsine ayrı yanıt verebiliyor. Böyle olduğu için yol atıyıcı karmaşıklaşır çünkü her aygıta tel çekilmesi gerekir. Hepsine ayrıca yanıt vermek zorunda kalıyor. Karmaşıklığı ve maliyeti daha yüksek ancak öncelik ve adaletle ilgili daha ayrıntılı düzenlemeler yapılabilir.

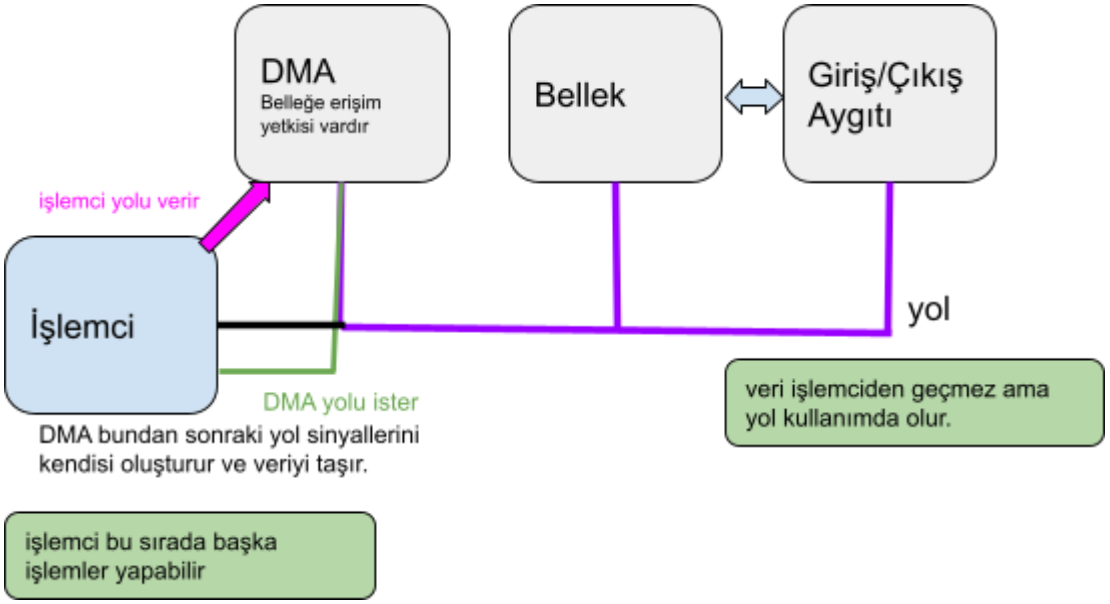


Doğrudan Bellek Erişimi(Direct Memory Access(DMA))

- Yüksek boyutlu verinin aktarımı için bellek ve işlemci arasında kullanılır.
- Bu sayede işlemci aradan çıkarılır.

İşlemci giriş/çıkışlar aygıtları ile sürekli olarak haberleşiyor. Giriş/çıkış aygıtları ile haberleşirken her türlü giriş/çıkış işlemi işlemci içerisinden geçiyor. İşlemci bellek erişimini de kendisi yaptığı için bu giriş/çıkış aygıtları bir şekilde belleğe yazmaya kalktığında eskiden hep işlemciden geçiyordu. Bundan dolayı giriş/çıkış aygıtlarının belleğe erişimi sırasında bir

gecikme yaşanıyordu. Hem işlemci meşgul ediliyor hem de gecikme yaşanıyordu. Bunun önüne geçmek için araya DMA eklenmiş. DMA yol üzerinde ana aygıt gibi davranır. İşlemci doğrudan DMA'ya talimat verir. İşlemci der ki sen bu doğrudan bellek erişimini yapacak yongasın şu adresten başla şu adrese kadar bu kadar baytı kopyala işin bitince bana haber ver. DMA bu şekilde giriş/çıkış aygıtından belleğe verileri aktarıyor. Böylece işlemci üzerindeki yük azalıyor.



İşlemcinin Aygıtlarla İletişim Yolları

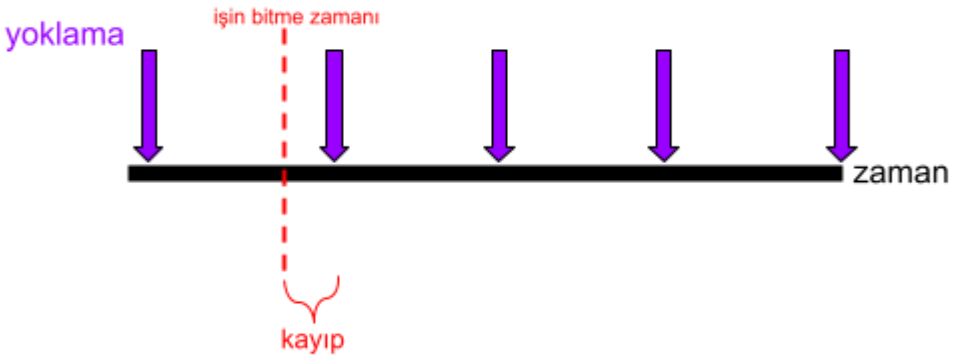
1.) Özel giriş çıkış buyrukları(*intel 8086 daki in ve out buyrukları*)
   - başarımı daha yüksek olabilir
   - fazladan donanım gereklidir.
   - esnek bir yapı değildir. haberleşmeyi sınırlar.

2.) Bellek eşlemeli(memory mapped I/O): veri belleğe yazılır yani giriş/çıkış aygıtına verilecek talimat belleğe yazılıyor, aygıt da sürekli olarak belleği izliyor ve oraya yazılan veriyi emir olarak kabul ediyor. Bunun yararı daha sonra aygıt ile ilgili değişiklik yapıldığında aygıta verilen emirler değiştirilebiliyor. Çünkü bellek üzerinden haberleşiliyor, daha esnek işlem yapılmasını sağlıyor.
   - ayrı yola ihtiyaç yok
   - bellek işlemleri ile aynı yolu kullanır. Bundan dolayı sistemi yavaşlatma riski oluşur.

Bilgisayar mimarisinde genelde ara yol bulmaya çalışır. Kendi sisteminizde neye ihtiyacınız varsa, bellek üzerinden giriş/çıkış yapacaksanız bellek eşlemeli giriş/çıkış kullanabilirsiniz.

Diyelim ki yazıcıya yazma talimatı gönderdik ve kağıt sıkıştı. İşlemcinin bundan nasıl haberi olacak?

1.) Yoklama (-ing. Polling) yöntemi
   - bir giriş/çıkış yazmacı eklenir
   - belirli aralıklarla bu yazmaçlar kontrol edilir
   - denetimi işlemci yapar. Ne zaman gidip bakacağına işlemci karar verir.
   - zaman kaybı oluşabiliyor



2.) Kesme(-ing. interrupt) yöntemi

Donanımın kesme atabilir olması ve işlemcinin de kesme alabilir olması gerekir. Giriş/çıkış aygıtı işini bitirdiğinde benim işim bitti diye haber verir işlemciye, işlemci de bu haberi alır ne yapması gerekiyorsa yapar. Sadece işlem bitti demeyebilir, fare ile tıklama yapıldı ve bu şekilde bir kesme de atılabilir. Bilinmeyen zamanlarda gelebilir. İşlemci açısından kötü tarafı denetim işlemcileri değil, hangi anda kesme geleceğini bilmiyor ama işlemciye bir kesme geldiğinde işini gücünü bırakıp onunla ilgilenmesi gerekir. Babanın bulaşık yıkayıp çocuğunun tuvalette olması örneği.

- zaman kaybı yok
- zaman kaybı yok ama işi gücü bırakıp gitmesi gerektiği için (o anda) hızlıca bir hareket etmek ve tepki vermek gerekir.

Bazı bilgisayar sistemleri kesmeyi almamayı tercih edebilir. Maskable interact. Kesmeyi maskeleyebilir. Örneğin intel işlemcilerde bayrak bitleri vardır(interrupt flag). onu sıfırlayıp kesme almamayı seçebilirsiniz. Yalnız bazı kesmeler vardır bunları almamayı seçemezsiniz. Yani bu kesmeyi alıp yanıt vermek zorundasınız.