BÖLÜM 3

SERİ PORT

3.1 SERİ HABERLEŞMENİN SEBEPLERİ

Seri Port' un ulaşımını ve genel olarak seri haberleşme , Paralel Port'a ve Paralel veri haberleşmesine göre daha zordur. Bir çok kereler , Seri Port'a bir cihaz bağlandığında verinin kullanılabilmesi için seri iletişimin Paralele dönüştürülmesi gerekir. Bunu için bir UART (Universal Asynchronus Receiver Transmitter) tüm devresi kullanılır. Yazılım kısmında ise , Paralel Port' a göre daha çok iş yükü ve kullanılan saklayıcı bulunur. Seri Port veya seri haberleşmenin bütün bu kadar olumsuz yönlerine karşın ne gibi avantajları bulunur.

1. Seri kablolar Paralel Kablolara göre daha uzun olur. Seri Port Lojik 1 seviyesini 3 ile-25 volt ve lojik 0 seviyesini +3 -+25 volt arasında iletir. Buna karşın , Paralel Port' a "0" 0 volt "1" 5V olarak iletir. Bu nedenle seri Port , 50V maksimum voltaj değişim aralığına sahip olmasına karşın , Paralel Port 5V maksimum aralığa sahiptir. Bundan dolayı kabloda oluşan kayıp , seri kablolarda paralel 'e göre çok önemli değildir .

2. Seri iletişimde Paralele göre daha az tel kullanılır. Eğer bu cihaz bilgisayara uzak bir yerde ise , bu cihaza giden , çekirdeğine 3 telli kablo , 19 veya 25 telli kabloya göre çok daha ucuzdur. Bununla beraber, her uçtaki ara birim fiyatı da hesaba katılmalıdır.

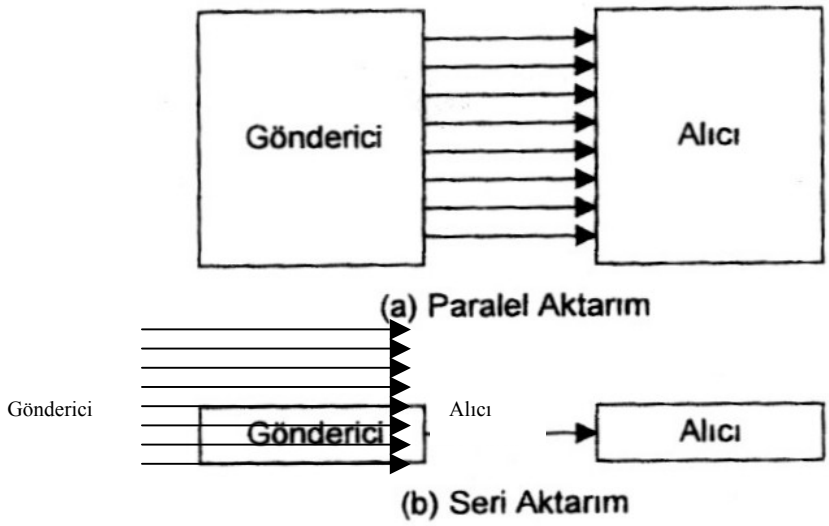
3. Seri haberleşme kullanan kırmızı ötesi (Infra Red) günümüzde çok popüler hale gelmiştir. Bu cihazlarda bir anda ancak bir bit veri seri olarak iletilebilir. Böyle bir haberleşmenin Paralel olması mümkün değildir.

4. Günümüzde , Tümdevresi üzerinde dış dünya ile iletişimde kullanılan , seri haberleşme arabirimi bulunan mikrodenetleyiciler çok yaygınlaşmıştır. Seri haberleşme uçları, mikro denetleyicilerin tümdevre uç sayısını azaltmaktadır. Genelde TxD (Transmit Data) ve RxD (Recive Data) aolarak adlandırılan 2 uç kullanılmaktadır. Buna karşın (-bit veri iletişiminde , 8 uç ve çoğu zaman bir darbe (strobe) ucu gerekir.

3.2 SERİ HABERLEŞMENİN TEMELLERİ

Bir mikroişlemci dış dünyaya (hafıza veya I/0 birimleri)ile genelde 8-bit' lik Parçalarla (8, 16, 32 ve 64 gibi)haberleşir. Şekil 3.1 (a)'da görüldüğü gibi , bu şekilde yapılan veri aktarımı paralel veri aktarımı (transfer) olarak adlandırılır. Bazı durumlarda örneğin PC'nin yazıcı ile haberleşmesinde , veri yolunda 8-bit veri ile paralel haberleşme yapılabilir.

Eğer mesafe uzunsa paralel veri aktarımı pek uygun değildir. Ayrıca 8-bit veri yolu pahalıdır. Bu gibi durumlarda seri haberleşme daha uygun olur. Şekil 3.1 (b)' de görüldüğü gibi yapılan bir veri aktarımı , seri veri aktarımı olarak adlandırılır. Tek bir veri hattının kullanıldığı bu tür haberleşmenin ucuz olmasının yanında , iki farklı şehirde bulunan iki bilgisayarın telefon üzerinde , bu yöntem kullanarak haberleşmesi de mümkün olur.



Şekil 3.1 Seri ve Paralel veri aktarımı

Seri haberleşmede, gönderici kısmında 8-bit veri , paralel'den seriye çevrilir. Ve daha sonra tek bir hattan karşıya gönderilir. Alıcı seri veriyi Paralele çevirerek 8-bit veriyi oluşturur. Eğer veri telefon hattından iletilecekse , 0 ve 1'ler ses tonuna çevrilir.Bu çevrim modem (modularor/Demulator) olarak adlandırılan bir cihaz tarafından yapılır.

Eğer mesafe ksa ise syısal sinyal modulasyonuna gerk duyulmadan , basit bir hat üzerinden veri iletilir.IBM PC 'de klavye ile anakart haberleşmesi bu şekilde yapılır.Telefon giibi veri Transferi , haberleşme hatlarını kullanır. Bu tür haberleşmede , göndericide 1 ve 0'lar modulator ile ses toonlarına çevrilir;alıcıda ise , bu ses tonları demodulator ile 0 ve 1'lere dönüştürülür.

3.2.1 SENKRON VE ASENKRON HABERLEŞME

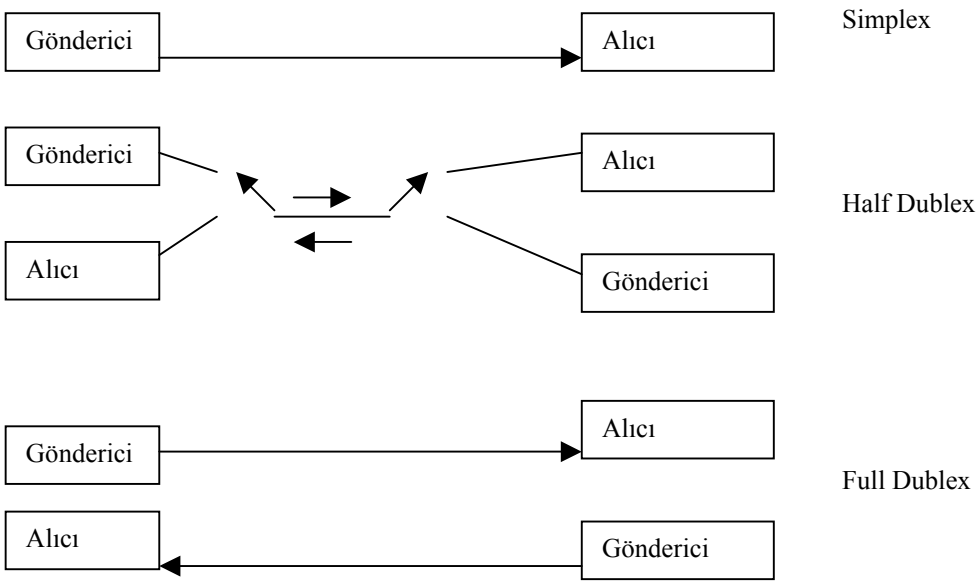
Seri veri haberleşmesinde senkron ve asenkron iki yöntem kullanılır. Senkron haberleşmede bir anda bir veri bloğu (Karakterler dizisi)aktarımı yapılırken , asenkron haberleşmede bir anda sadece bir byte iletilir.

Her iki haberleşme yöntemi yazılım ile gerçekleştirilebilir. Bununla beraber , bu şekildeki haberleşmede programlar uzun ve zor olacağı için , genelde seri veri

iletiminde özel tümdevreler kullanılır. Bu tümdevreler yaygın şekilde , UART (Universal Synchronus Asynnchronus Receiver Transmitter) olarak adlanndırılır. IBM PC'de kullanılan Com port'u, daha ilerki blümlerde detaylı olarak sunulacak olan , 8250 UART'ını veya günümüzde daha gelişmiş sürümlerini kullanmaktadır. 8251 ise bir USART tümdevresidir.

3.2.2 SERİ VERİ AKTARIM KANALLARI

Seri veri iletişim tek yönlü oluyorsa , PC'den yazıcıya olduğu gibi , bu veri iletimi simplex olarak adlandırılır. Veri hem gönderiliyor hem alınabiliyorsa bu yönteme dublex denir.Bu iletişimde eğer veri bir anda sadece bir bit aktarılabiliyorsa half dublex haberleşmede , bir hat gönderme ve bir hat ta alma için olmak üzere , toplam iki tane hat kullanılır. Seri veri aktarımındaki kullanılan kanal durumuna göre yapılan haberleşme çeşitleri şekil 3.2'de görülmektedir.



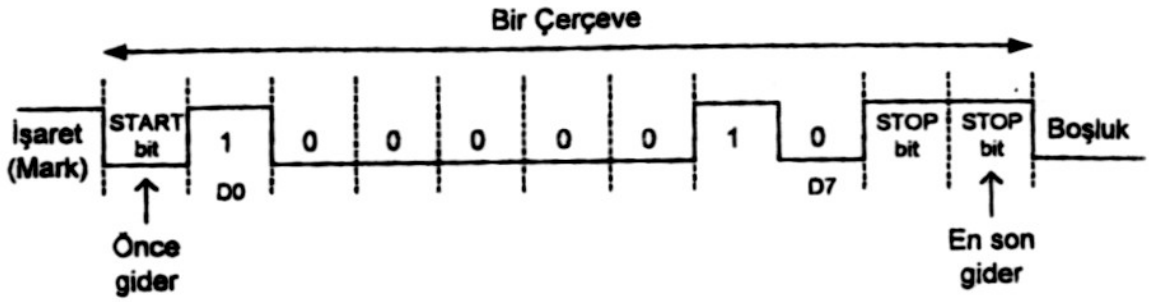
Şekil 3.2 .Seri veri aktarım yöntemleri .

3.2.3 ASENKRON SERİ HABERLEŞME

Seri veri haberleşmesinde , veri gönderen ve alan uçların belli kurallarına göre haberleşmesi gerekir.Protookol olarak adlandırılan bu kurallar, verinin nasıl paketleneceği , bir karakterdeki bit sayısı ve verinin ne zaman başlayıp biteceği gibi bilgileri belirler.

Asenkron seri veri haberleşmesi, karakter-tabanlı iletişimlerde yaygın olarak kullanılır. Asenkron yöntemde , her gönderilecek karakter , başlama(START) ve bitime (STOP) bit'lerinin arasına yerleştirilir.Bu işlem çerçeveleme (framing) olarak adlandırılır.

Asenkron haberleşmesindeki veri çerçeveleme (framing) olarak adlandırılır. Asenkron haberleşmesindeki veri çerçeveleme işleminde , ASCII karakterler bir START ile bir STOP bit'i arasında paketlenir. Start bit'i her zaman bir bit'tir , fakat STOP bit'i bir veya iki bit'tir. Şekil 3.3 'te görüldüğü gibi , START bit her zaman lojik 0(düşük), Stop bit lojik 1'dir(yüksek).Verilen ASCII 'A', ikili olarak 01000001, START bit ile 2 STOP bit arasında çerçevelenmiştir. Seri iletişimde önce en ddüşük değerli bit D0(LSB) dışarı gönderilir.



Şekil 3.3 ASCII'A' (41H) Karakterinin çerçevesi.

Şekil 3.3 te solda görüldüğü gibi, seri veri aktarımının yapılmadığı durumda, sinyal seviyesi 1 dir(yüksek) ve bu durum işaret (mark) olarak adlandırılır. Lojik 0(düşük) ise boşluktur(space). Veri iletimi bir start biti ile başlar ve bunu takiben D0 (LSB) daha sonra D7 ye (MSB) kadar geri kalan bitler ve son olarak "A" karakterinin sonunu belirten iki Stop Biti gelir.

Asenkron seri haberleşmede kullanılan tüm devreler ve modemler 5, 6, 7 ve 8 bit veri uzunlukları için programlanabilir. Veri, uzunluğuna ek olarak bir veya iki tane STOP biti kullanılır. Eski sistemlerde ASCII karakterler 7 bit idi. Yeni uzatılmış ASCII karakterler yüzünden bir ASCII karakter için 8 bit gerekir. Bazı ASCII olmayan klavyeler 5 veya 6 bit klarakterleri kullanır. Bazı eski sistemlerde, veri alan cihazın yavaşlığından dolayı , cihaza, kendini ayarlamaya yeterli zaman vermek için iki stop bit kullanıldı. Günümüzde modern PC'lerde genellikle 1 STOP bit kullanılır.

Bir seri haberleşmesinde, iletilen verinin dışında fazladan kullanılan bitler 1 fazladan zaman(overhead) ve yük oluşturur.

Örneğin , ASCII karakterin iletiminde, eğer iki STOP bit'i kullanılıyorsa, her bir 8-bit ASCII kodu için toplam 11-bit iletilir. Yani her 8-bit için, iletişim hattından fazladan 3-bit veya %30 ek yük bulunur.

Bazı durumlarda, veri bütünlüğünü korumak için karakter byte' nın eşlik(Parity) bit' i veri çerçevesine eklenir. Yani her karakter için(sisteme bağlı olarak 7-bit veya 8-bit) START ve STOP bitlerine ek olarak bir tek eşlik bit' i eklenir. Eşlik bit' i tek (odd) veya

çift (even) olur. Tek- eşlik bit durumunda , eşlik bit'i dahil, veri bit'lerinin sayısı bir tek sayıdır. Örneğin , ASCII "A" karakteri ikili 0100 0001 olarak 0 çift eşlik bit'ine sahiptir. UART tümdevreleri daha sonra görüleceği gibi, tek , çift veya eşliksiz.(no-parity) olarak programlanabilir. Eğer bir sistem eşlik bit'i gerektiriyorsa , eşlik bit'i verideki en değerli bit'ten (MSB) sonra gönderilir. Bunun STOP bit'i takip eder.

3.2.4 VERİ AKTARIM HIZI

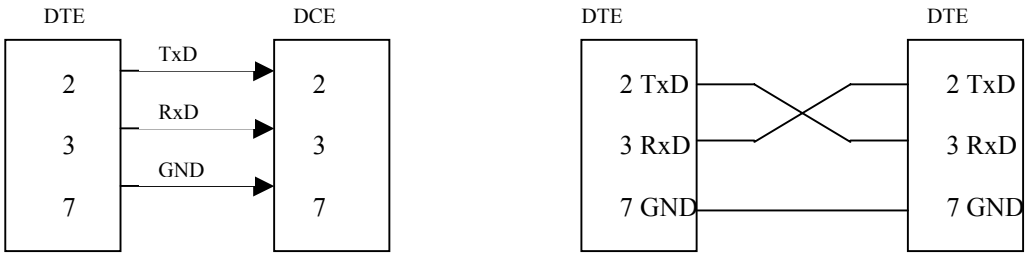
Seri veri haberleşmesinde veri aktarım hızı saniyedeki bit sayısı(BPS-bit Per second) olarak belirtilir. Veri aktarım hızını belirtmede diğer yaygın olarak kullanılan İngilizce terim **baud rate**'dir . Bununla beraber bu iki ifadenin bir birine eşit olması gerekmez. Çünkü, baud rate bir modem terminolojisidir ve saniyede sinyaldeki değişim olarak tanımlanır. Modemlerde bazı durumlarda bir sinyal değişimi ile bir çok veri bit'i transfer edilir. İletişim teli düşünüldüğünde , bps ve baud rate aynıdır. Bu yüzden kitapta ikisi karşılıklı değişmeli olarak kullanılmaktadır.

Bir bilgisayarın seri veri transfer hızı haberleşme Port'larına bağlıdır. Örneğin eski IBM PC/XT 100 ile 9600 bps hızlarında veri transfer edebilmekteydi .Bununla beraber. Hızlı PC'ler 19200 BPS gibi yüksek hızlara çıkabilmektedir. Asenkron veri haberleşmesinde , baud rate genellikle 100000 bps ile sınırlıdır.

3.2.5.VERİ HABERLEŞME SINIFLARI

Seri haberleşmeyi kullanan cihazlar iki sınıfa ayrılmaktadır. Bunlar DTE (Data Terminal Equipment) ve DCE(Data Communication Equipment) olarak adlandırılır. DTE, bilgisayar ve ya terminal gibi veri gönderen veya alan cihazlardır. Buna karşın DCE , modem ve yazıcı gibi veri aktaran cihazlardır. Daha sonraki bölümde Tablo 3.1'de gösterilen sinyaller DTE yönünden tanımlanmıştır.

DTE ve DCE arasındaki en basit bağlantı , Şekil 3.4 (a)'da görüldüğü gibi en az üç tane uç , TxD, RxD ve toprak gerektirir. İki PC gibi , iki DTE cihazı arasındaki minumum bağlantıda , 2 ve 3 numaralı uçlar, Şekil 3.4 (b)'deki gibi , karşılıklı çapraz bağlanır.



(a) DTE – DCE bağlantısı (b) DTE-DTE bağlantısı

Şekil 3.4 DTE –DCE ve DTE- DTE bağlantıları.

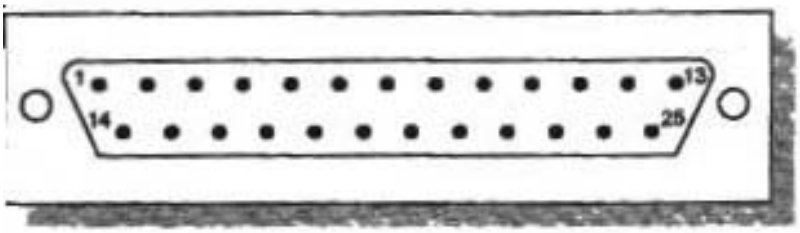
3.3 RS232 STANDARDI

Değişik üreticiler tarafından yapılmış veri haberleşme cihazlarının uyumluluğunu sağlamak amacıyla , EIA (Elektronik İndustries Association) tarafından RS232olarak adlandırıldı .Daha sonraları 1965 'te RS232B ve 1969 da RS232C standartları ilan edildi. Kitapta bu standart için kısaca RS232 kullanılacaktır .

Günümüzde RS232 en yaygın kullanılan seri I/O arabirim standardıdır. Bu standart TTL lojik ailesinden çok önceleri belirlendiği için , giriş ve çıkış voltaj seviyeleri TTL uyumlu değildir. RS232 'de lojik 1-3V ile –25 volt arasında, Lojik 0+3V ile +25V arasında tanımlanır. –3v ile +25 v arası tanımsızdır. Bu yüzden herhangi RS232 cihazını , bir mikroişlemci –tabanlı sisteme bağlamak için , MC1488, MC 1489 veya TSc232 gibi voltaj çeviriciler kullanılır. Bu tümdevreler hat sürücüler /alıcıları (Line driver /receiver) olarak adlandırılır.

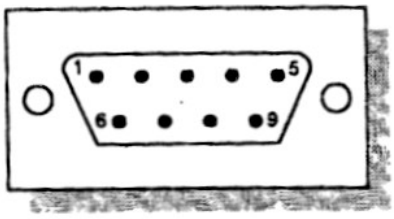
3.3.1.RS232 SİNYALLERİ

RS232 için , Şekil 3.5'te görülen , bir DB-25 konnnektörü ile erişilen toplam 25 uç tanımllanmıştır. Bu uçların hepsinde sinyal bulunmaz. Kullanılan konnektörü belirlemede , DB25P(Plug) erkek konnektör, DB25(sovket) dişi konnektör için kullanılır.



Şekil 3.5. RS232 DP-25P erkek konnektör.

Modem bilgisayarda bütün 25 uca gerek olmadığı için , IBM şekil 3.6'da görülen, DP-9 seri I/O standardını tanımlamıştır.



Şekil 3.6 IBM PC-9 uçlu erkek konnektör.

Tablo 3.1 ‘de IBM PC 9-uçlu konnektör’de bulunan RS232 sinyalleri verilmiştir.

Tablo 3.1 9 ve 25 uçlu konnektörler içinRS232 sinyalleri		
1	8	Data Carrier Detect(DCD)
2	3	Receive Data (RxD)
3	2	Transmite Data (TxD)
4	20	Data terminal Ready (DTR)
5	7	Signal Ground(GND)
6	6	Data set Ready(DSR)
7	4	Request to Send(RTS)
8	5	Clear To Send(CTS)
9	22	Ring Indicator(RI)

3.3.2. RS232 senkronizasyon sinyalleri

İki cihaz arasında güvenli veri iletişimin olabilmesi için , veri aktarımın düzenlenmesi gerekir.Bu amaçla seri veri haberleşmesinde senkronizasyon sinyalleri kullanılır. Bu sinyal çok yaygın olarak kullanıldıkları için , bilinmeleri herhangi UART tümdevresi veya modem cihazının çalışmada çok önemlidir. Bu senkronizasyon sinyalleri aşağıda tanıtılmaktadır.

1. DTR (Data Terminal Ready): DTE(PC COM port’u) çıkış modem giriş ucudur. Bu hat modem’e , terminalin (Bir PC com Port’u , dolayısıyla UART) veri iletimi için hazır olduğunu belirtir. COM port’unda bir problem var ise , bu sinyal aktif yapılmaz.

2. DSR (Data Set Ready): DTE (PC COM port’u) giriş , modem çıkış ucudur. Bu hat terminele(Bir PC COM port’u , UART), modem’in veri iletişimi için hazır olduğunu belirtir. Eğer modem herhangi bir nedenden dolayı telefon hattına bağlanmıyorsa , bu sinyaller pasif yapılarak , veri göndermek veya almak için modem’in hazır olmadığı PC’ye belirtir.

3. RTS (Reques To Send): DTE (PC gibi) göndereceği verisi olduğunda, bu uçla modeme haber verir.

4. CTS (Clear To Send): RTS sinyaline cevap olarak , modem alacağı veri için yeri olduğunda , bu sinyali DTE’ye (PC UART’ına) göndererek veri almaya hazır oolduğunu belirtir.

5. DCD (Data Carrier Detect): Modem , telefon hattından bir taşıyıcıyı (carrier) belirlediğinde , bu hattı akktif yaparak DTE ‘ye (PC) bu durumu haber verir.

6. RI (Ring Indigator): Telefon çaldığıında , modem DTE’ye(PC) bu sinyalle haber verir. Altı Senkronizasyon sinyalinden en az kullanılanı bu sinyaldir. Çünkü telefona modem cevap verir. Eğer telefona PC’nin cevap vermesi isteniyorsa bu sinyal kullanılabilir.

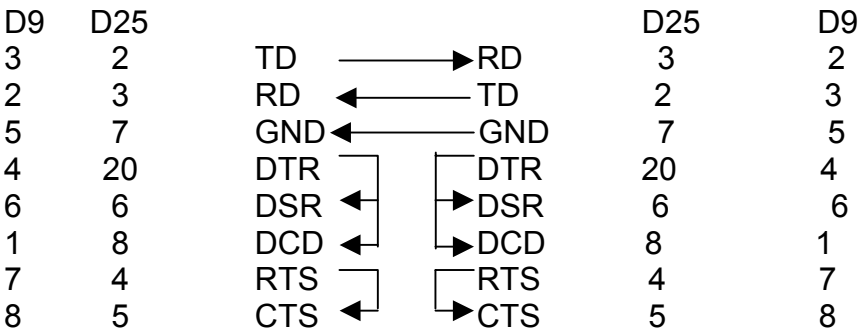
Yukarıdaki tanımlamalardan sonra, PC ve modem haberleşmesi şu şekilde özetlenebilir. DTR ve DSR sinyalleri , PC ve modem tarafından çalışmakta ve hazır olma durumlarını belirtmede kullanılır. Buna karşın , RTS ve CTS veri akışını kontrol etmede kullanılır. PC veri göndermek istediğinde , RTS hattını aktif yapmakta , buna cevaben , modem veri kabul etmek için hazır ise (veri için yeri var ise), CTS sinyalini gönderir. Eğer yer yok ise , modem CTS'yi aktif yapmaz. Bu durumda PC'nin tekrar aynı şekilde denemesi gerekir.

3.3.3. BOŞ MODEM BAĞLANTISI

Boş (null) modem bağlantısı iki DTE cihazını bağlamada kullanılır. Bu yöntem Zmodem veya Xmodem protokolü gibi bir protokol kullanarak, ağ haberleşmesi veya bilgisayarlar arası dosya aktarımında kullanılan ucuz bir yoldur. Bu yöntem ayrıca bir çok mikroişlemci geliştirme sistemlerinde kullanılır.

Şekil 3.7'de boş modem bağlantısı için , bir örnek verilmiştir. Görüldüğü gibi sadece 3 hat (TD, RD ve GND) gerekir. Bu verilen bağlantı ile amaçlanan , bilgisayarın diğer bir bilgisayardan ziyade , sanki bir modem ile konuşuyor gibi düşünmesini sağlamaktır. TD ve RD veri hatları , karşılıklı çaprazlama bağlanmış ve topraklar ortaklanmıştır. Her iki makine de DTR sinyalleri aktif yapıldığında , DSR ve DCD hemen aktif olur. Bu nokta da , bilgisayar bağlı olduğu bir sanal modem'in hazır olduğunu ve bu modem'in DCD sinyalini aldığı düşünür.Her iki bilgisayar aynı hızda haberleştiği için kontrolüne gerek yoktur.. Bu yüzden RTS ve CTS sinyalleri her iki makinede birbirine bağlanmıştır. Bir bilgisayar verisini göndermek istediğinde , RTS sinyalini aktif yapar. Bu sinyal CTS'ye geri bağlı olduğundan , gönderme için tamam cevabını hemen alır ve bu işlemi yerine getirir.

RI sinyali her iki uçta bağlanmıştır. Bu sinyal telefon hattında çalma işlemini bilgisayara bildirir. Telefon hhattına bağlı bir modem olmadığı için bu hhat bağlantısız bırakılmıştır.



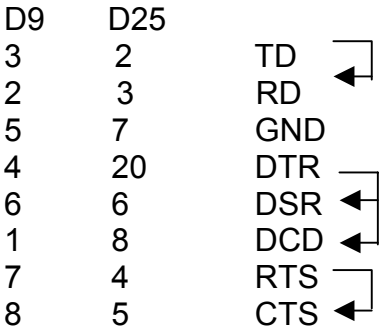
Şekil 3.7. Boş modem bağlantısı.

3.3.4.GERİ BESLEME BAĞLANTISI

Geri besleme (loop –back) bağlantısı , özellikle seri RS232 haberleşme programları yazarken , geliştirme aşamasındaki test'lerde çok faydalıdır. Bu bağlantı da , Şekil 3.8'de görüldüğü gibi , almma ve göndermme hatları birbirine bağlanmıştır. Seri olarak dışarı gönder, len veri yine aynı port'tan içeri alınır.

3.3.5.DTE/DCE HIZLARI

Daha öncede belirtildiği gibi , tipik bir veri terminal cihazı (DTE) bir bilgisayar ve tipik bir veri iletişim cihazı (DCE) bir modemdir . Genellikle DTE'den DCE'ye ve DCE'den DCE'ye olan veri hızları söz konusu olur. DTE ile DCE arasındaki hız, bilgisayar ile modem arsındaki hızdır ve bazen terminal hızı olarak belirtilir. Bu hız DCE-DCE hhızından fazla oolmalıdır. DCE-DCE arası, nodemler arası bir bağdır ve bu aradaki veri hızı bazen hat(line) hızı olarak belirtilir.



Şekil 19.8. Geri besleme bağlantısı.

Günümüzde 28.8 K ve 56K gibi hızlardaki modemler yaygındır. Yani, bu modemlere göre , iki DCE arası hızın, 28.8 K, 33.6K ve 56K olması beklenir. Yüksek hızdaki bir modem için , DTE –DCE hızının yaklaşık 115200 BPS (16550A UART'ının maksimum hızı) olması gerekir. Bu nokta genellikle pek çok kişinin hataya düştükleri hususustur. Kullanılan hhaberleşme programları DTE-DCE ayarlarını alır. Böylece 9.6 K BPS, 14.4K BPS gibi hızlar görülür ve bunların modem hızı oldukları düşünülür.

Günümüzde modemler veri sıkıştıırma ve açma fonksiyonlarına sahiptir. Sıkıştırma oranları olarak 1:4 ve hatta daha yüksek oranlar yaygındır. Bu 1:4 ooranı bir metin dosyası(text fiele) için tipik bir değerdir. Eğer bir metin dosyası 28.8 K(DCE-DCE) hızında aktarılıyor ve kullanılan modem sıkıştırma yapıyor ise , gerçekte bilgisayarlar 115.2 K hızında veri aktarımı yapılır. Aynı zamanda DTE-DCE hızı mmoodem'in bağlantı hızından çok daha fazla olmalıdır.

Bazı modemler üzerinde 1:8 maksimum sıkıştırma sağlayabileceği yazılır. Eğer bu modem 33.6 K ise , modem ile UART arasında , maksimum 268800 BPS veri aktarım hızı sağlanır. Bununla beraber , eğer bir bilgisayarın UART'ı en fazla 115200

BPS hızına sahip 16550A ise, modemimizin fazla performansından yararlanamayız. Eğer maksimum 230400 BPS hızı sağlayan 16C650 UART'ı kullanır ise bu problem olmaz.

Yukarıda bahsedilen yüksek oranlar bazı durumlarda elde edilmez. Çünkü modemlerin üzerinde olan sıkıştırma oranları maksimumdur. Örneğin sıkıştırılmış bir dosya gönderilir ise , modem bu doosyayı sıkıştıırmak için çok zaman harcar. Böyle bir durumda modem'in veri sıkıştırma özelliğinin kapatıılması daha uygundur. Ayrıca , bazı dosyalar diğerlerinden daha kolay sıkışır. Bunun sonucu daha yüksek sıkıştırma oranlarına sahip olunur.

3.3.6 VERİ AKIŞ KONTROLÜ

DTE-DCE arsındaki hız DCE-DCE hızından çok fazla ise , PC'nin gönderdiği veri modem'deki buffer alanını kısa zamanda doldurduğundan bir veri akış kontrolü gerkir. Akış kontrolü için yazılım ve donanım olmak üzere iki yöntem vardır.

Yazılım akış kontrolünde , bazı yerlerde Xon|Xoff olarak belirtilen iki karakter kullanılır. Xon olarak normalde ASCII 17 ve Xoff olarak ASCII 19 karakteri kullanılır. Modem'in için sınırlı bir allaan olduğundan , bilgisayar bu alanı doldurduğunda, modem Xoff karakteri göndererek bilgisayara daha fazla veri göndermesini belirtir. Modem veri için yeri olduğunda bilgisayara bir Xon karakteri gönderir. Bunun sonucunda bilgisayar daha fazla veri gönderir.Bu çeşit akış kontrolünün avantajı , veri iletimin yapıldığı TD/RD hatlarına ek olarak başka hatlara gerek olmasıdır. Bununla beraber, yavaş hızlarda olan haberleşmede , bu tür karşılıklı el sıkışma senkronizasyonu , iletişimi daha da yavaşlatır.

Donanım veri akış kontrolü , aynı zamanda RTS/CTS akış kontrolü olarak ta bilinir. Bu yöntemde yazılım yöntemindeki veri hatlarından aktarılan fazladan karakterler yerine , seri kablodaki iki hat kullanılır.Bu şekilde yapılan bir akış kontrolü iletişim hızını yavaşlatmaz . Bilgisayar veri göndermek istendiğinde RTS(Reguest to Sent) hattını aktif yapar .Eğer modemde veri için yer var ise , CTS (Clear to Sent) hattını aktif yapar ve bilgisayar veri göndermeye başlar. Eğer modem yere sahip değıl ise CTS sinyalini göndermez .

3.3.7 DİĞER STANDARTLAR

Bir RS232 kablosunun uzunluğu arasında , sinyalde kapasitif yük artmaktadır. Bunun sonucu , yüksek veri hızları güvensiz olmaktadır. Eğer kablo uzunluğu 5 feet veya daha az olursa , RS232 ile 10000 BPS ile veri aktarım hızı elde edilebilir. Eri hızını ve kablo uzunluğunu arttırmak için RS232 'nin elektriksel özellikleri yeniden tanımlanmalıdır. Bunun sonucu , RS422 ve RS423 gibi yeni standartlar çıkmıştır. Tablo 3.2 RS232 ile bu standartların karşılaştırılmasını göstermektedir.

Tablo 3.2. RS232'nin RS422 ve RS423 ile karşılaştırılması.			
	RS232	**RS422**	**RS423**
Maks.Kablo Uzunluğu (m) Maksimum Hız (baud)	15 20K	1200 10M/12 m 1M/ 120 m 100K/1200 m	1200 100K/9 m 10K/90 m 1K/1200 m

3.4 PC SERİ PORT'U

Bir PC Seri Port'u , daha sonra anlatılacak olan, bir UART tümdevresi kullanılır. DOS ve BIOS , 8250 gibi bir UART tümdevresinin detaylarından programcıları kurtararak , IBM PC'nin Seri Port'una kolay erişim sağlar. PC'deki INT 14 kesmesi seri veri iletişimde kullanılabilir. DOS kesmesinin yanında Seri Port'ların veri uzunluklarını belirleme ve hızlarını ayarlama gibi fonksiyonları , DOS'ta MODE komutuyla programlanabilir.

3.4.1.SERİ PORT ADRESLERİ

Bir IBM PC, COM olarak belirtilen 4 tane taneye kadar Seri Port'a sahiptir. Bu port'lar 1, 2, 3 ve 4 (BIOS numaraları 0, 1, 2, 3) olarak numaralanmıştır. PC açıldığında POST(Power-On Self –Test) esnasında , 4 COM adresleri görülmektedir. Her UART için atanan I/O adresi 16-bit olduğundan, UART başına 2 byte gerekir.

Tablo 3.3.PC Seri Port adres atamaları	
COM Taban Adresi	**İsim**
0000:0400 0000:0402 0000:0404 0000:0406	COM1 COM2 COM3 COM4

Tablo 3.4'te PC seri Port standart I/O adres adres atamaları gösterilmektedir. Bu adresler Birçok bilgisayar için geçerlidir.

Tablo 3.4 PC seri Port I/0 adres atamaları		
İsim	**Adres**	**IRQ**
COM1	3F8h	4
COM2	2F8h	3
COM3	3E8h	4
COM4	2E8h	3

Bir port'a atanmış olan adresi öğrenmek için değişik yöntemler bulunur. Örneğin, sistem ayarlarına bakılarak bu öğrenilebilir. Aşağıda DOS debug programı ile 0040:0000 adresine bakılarak COM atamaları öğrenilmektedir.

```
>debug
-d 0040:0000 L8
0040:0008    F8  03   F8   02   00   00   00   00
```

Yukarıda verilen örnekte , COM1 3F8h ve COM2 2F8h atanmış olup COM3 ve COM4'te bir atama yapılmıştır. Aşağıda verilen örnekte C programı ile , Seri portlar için BIOS tarafından atanan adresleri akuyabileceğimiz gösterilmektedir.

```
#incllude  <stdio.h>
#include  <dos.h>

void main(void)

{
    unsigned int far    *ptr_addr;
    unsigned int address;
    int   i;

   ptr _addr =  (unsigned int far   *)   0x00000400;
     for      ( i=0;    i<4;     i++)
   {
        address = *ptr_addr;
        if (address= 0)
        printf("COM%d için  port bulunaamadı   \n", i+1);
        else
            printf("COM%d için atanan adres %Xh\n", i+1,  address);
         *ptr_addr++;
   }
}
```

Yuakarıda verilen C programı , daha önce debug programı ile Seri Port adresleri bulunan bilgisayarda çalıştırıldığında aşağıdaki çıkış elde edilir.

```
COM1 için atanan adres 3F8h
COM2 için atanan adres 2F8h
COM3 için port bulunamadı
COM4 için port bulunamadı
```

3.4.2 UART İLE SERİ HABERLEŞME

Seri haberleşme programı yazarken iki yöntem bulunur. Birinci yöntemde UART' ın durumu okunarak , verinin hazır olup olmadığı anlaşılır. Bu yöntem ingilizce polling olarak adlandırılır. Diğer yöntemde , UART ile PC arasındaki haberleşmede bir

kesme üretilir;bunun sonucu kesme hizmet programında veri işlenir. İlk yöntemdeki UART' ın durumunun okunması yavaş bir işlemdir ve CPU'yu çok meşgul eder. Bu yöntemle en fazla 34.8 K BPS veya yeni pentium Pro ve daha üst işlemcilerde biraz daha yüksek hızlar elde edilir. Kesme proogramı kullanan diğer yöntemde düşük hızlı makinelerde bile 115.2K BPS hızları kolay şekilde desteklenir.

Daha önce Bölüm 19.6'da Asembly dili kullanarak iki PC'nin haberleşmesi için basit bir program örneği verilmişti. O programda seri haberleşmede BIOS INT 14 kesmesi kullanılıyordu. Aşağıda verilen iki farklı C programı ile , benzeri şekilde iki PC birbiriyle konuşmaktadır. Bir PC' den klavye yolu ile girilen karakkterler diğer PC' nin ekranına gönderilir. İlk program UART' ın okunması (polling) yöntemi kullanılır. İkinci programda kesme alt programı ile veri işlemi yapılır. Program kodunun açıklamalarında , yapılanlar izah edilmektedir.

```
/*seri_port.c
*Polling yöntemi ile UART çalışması
*/
#include  <dos .h>
#include <stdio.h>
#include  <conio.h>

#define PORT1 0x3F8
/* Seri Port Taban adresleri */
/* COM1   0x3F8 */
/* COM2   0x2F8  */
/* COM3   0x3E8 */
/* COM4   0x2E8 */

void main (void)
{
     int   c;
     int ch;

     outportb(PORT1  +1  , 0);    /* Kesmeler pasif */
     /* PORT1  -  Haberleşme ayarlari   */
     outportb(PORT1    +3  , 0x80);   /*DLAB = 1   */
     outportb(PORT1   +0 , 0x03);
/* Baud Rate Ayeri  -   Divisor Latch Low Byte  */
/*Defult   0x03  =   38.400 BPS  */
/* 0x01  = 115.200 BPS  */
/* 0x02 =  56, 700 BPS */
/*  0x06 =  19, 200 BPS */
/* 0x0C =   9600 BPS  */
/*0x18  =   4, 800  BPS  */
/* 0x30  =  2, 400 BPS  */
```

```
 outporb (PORT1  +1 , 0x00);
/* 8-Bit,  NO parity, 1 Stop Bit  */

outporb(PORT1  +3  , 0x03);
outportb(PORT1  +4,  0x0B);
/*Turn on DTR, RTS, and  OUT2 */

printf("\nSeri Haberleşme Programi\n"
printf("Cikmak için ESC tuşu gerekiyor...\n");

do
{  c= inportb(PORT1  +5);  / *Line Status Register */
     /* Karakterlerin alınıp alınmadığını kontrol et */
if   (c & 1)
{    ch = importb(PORT1);  /* Veri Hazır, okku  */
     printf( "%c, ch);              /* ve ekrana yaz...*/
}

if (kbhit()) {ch = getch ();  /*Tuşa basıldıysa oku */
outportb(PORT1, ch);}       /* ve seri port'a gönder  */

}  While (ch !=27);  /*ESC(ASC 27) basıldığında cık  */

}
```