- Asynchronous Technologies -

Synchronous vs. Asynchronous Communication

Serial communication sends information sequentially, one bit at a time. This can be accomplished with as few as three wires (or *pins)*: one to *transmit* data, one to *receive* data, and a common *ground* wire. Parallel communication, in contrast, sends a group of bits at a time over multiple pins.

Serial communication requires some form of *synchronization*, so that devices are always aware when data is, or is not, being sent.

Synchronous communication forces two devices to *continuously* stay synchronized, even when *no* data is being sent. This is accomplished by sending **Idle bits** when no **Data bits** are being exchanged.

In contrast, two devices engaging in **asynchronous communication** are *not synchronized* with each other. As a result, the sending device must inform the receiving device when it is *starting* a data stream, and when it is *stopping* a data stream.

This is accomplished using **Start** and **Stop bits.** Asynchronous serial communication usually sends seven or eight data bits between start and stop bits. The sending and receiving devices must *agree* on the number of data bits. An optional **parity bit** allows a basic level of error-checking.

Asynchronous communication introduces additional overhead (due to the start/stop bits), and is **less efficient** than synchronous communication.

Asynchronous communication defines two types of devices:
- **DTE (Data Terminal Equipment) –** end devices, such as a router or workstation.
- **DCE (Data Communication Equipment) –** provides the clocking, such as a modem.

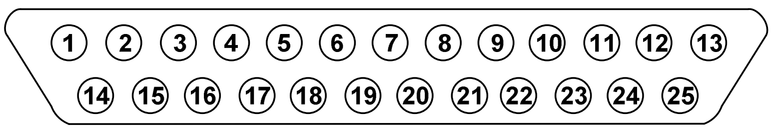
The defined physical standard for connecting DTE and DCE equipment is **EIA/TIA-232** (previously known as **RS-232**).

(Reference: http://www.certiguide.com/aplush/cg_aph_AsynchronousVersusSynchronous.htm; http://www.inetdaemon.com/tutorials/theory/concepts/asynchronous_vs_synchronous.shtml; http://www-scm.tees.ac.uk/users/u0000408/Async/async.htm)

EIA/TIA-232 Pins

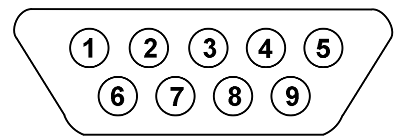
Each pin in an EIA/TIA-232 cable has a designated purpose:

DB25



Pin 1	Ground
Pin 2	TD – Transmit Data
Pin 3	RD – Receive Data
Pin 4	RTS – Request to Send
Pin 5	CTS – Clear to Send
Pin 6	DSR – Data Set Read
Pin 7	Signal Ground
Pin 8	CD – Carrier Detect
Pin 20	DTR – Data Terminal Ready
Pin 22	RI – Ring Indicator

DB9



Pin 1	CD – Carrier Detect
Pin 2	RD – Receive Data
Pin 3	TD – Transmit Data
Pin 4	DTR – Data Terminal Ready
Pin 5	Signal Ground
Pin 6	DSR – Data Set Read
Pin 7	RTS – Request to Send
Pin 8	CTS – Clear to Send
Pin 9	RI – Ring Indicator

The required pins for data transfer are **TD (Transmit Data), RD (Receive Data),** and **Signal Ground.**

The **RTS (Request to Send)** pin is used by the *DTE* to tell the DCE it is ready to receive data. The **CTS (Clear to Send)** is used by the *DCE* to tell the DTE it is ready to receive data.

The **DTR (Data Terminal Ready)** pin informs the DCE that the DTE is powered on and functional. The **DSR (Data Set Ready)** pin informs the DTE that the DCE is powered on and functional.

The **CD (Carrier Detect)** pin indicates that a connection has been made with a remote modem, or that a dial tone is present.

The **RI (Ring Indicator)** pin will indicate that an incoming call is occurring.

A **null modem** cable will flip the *TD* and *RD* pins, the *RTS* and *CTS* pins, and the *DTR* and *DSR* pins - to connect DTE devices back-to-back.

(Reference: http://www.taltech.com/TALtech_web/resources/intro-sc.html; http://www.zytrax.com/tech/layer_1/cables/tech_rs232.htm
CCNP BCRAN Exam Certification Guide, Second Edition. Page 88-89. Cisco Press ISBN: 1587200848)

Modems

As its name implies, a **modem (*mo*dulator/*dem*odulator)** will *modulate* a digital signal onto an analog signal when sending data across the wire. When receiving, the modem will then *demodulate* the analog signal back to digital.

The terms **bit rate** and **baud rate** are often mistakenly used interchangeably when discussing modem *speeds*. Bit rate refers specifically to the amount of bits sent over a period of time, usually measured in **bps (bits-per-second)**. Baud rate, in contrast, refers to the number of **symbols** sent over a period of time. A symbol can be comprised of multiple bits.

Thus, the *bit rate* of a modem may be larger than the *baud rate* of a modem.

Modems may use either *in-band* or *out-of-band* signaling. In-band signaling sends control information on the same 'channel' as the data is being sent. A dialup modem uses in-band signaling.

Out-of-band signaling dedicates a separate channel for control information. ISDN BRI contains data (Bearer or "B") channels and one control (Delta or "D") channel, and is thus an example of out-of-band signaling.

Modems must perform call setup (called **handshaking**), before data can be transferred. This often takes several seconds.

Various modem standards include (by no means a comprehensive list):
- V.32 – max speed of 9,600 (newer revisions capable of 19,200 bps)
- V.34 – max speed of 28,000 bps
- V.90 – max speed of 56,000 bps
- V.92 – max speed of 56,000 bps (with quicker handshake than V.90)

A **win-modem** (sometimes referred to as a **soft-modem**) offloads functions normally controlled by hardware into software.

(Reference: http://www.webopedia.com/quick_ref/dialup_modem_standards.asp; http://en.wikipedia.org/wiki/Modem)

Modem AT Commands

Modems can be controlled using the command line, or via an initialization string. These commands are usually based on the Hayes modem command set, though some modems use proprietary command strings.

The **AT (attention)** string informs the modem that a command will follow. Common (standard) AT commands include:

ATA0	Answers incoming call
ATD*xxx…*	Dials the specified number
ATD9W*xxx…*	Dials 9, waits for a second dial tone, then dials the specified number
ATE0	Disables local echo
ATH0	Hangs up
ATL*x*	Adjusts the volume, where *x* can be a value of 0 - 3
ATM0	Disables the speaker
ATZ	Resets the modem

Additional AT modem commands are available in the extended command set, referenced with an ampersand (&). An example would include:

AT&F	Reverts modem to factory defaults

To configure a modem connected to an asynchronous interface on a Cisco device, **reverse telnet** must be used. Instead of connecting to a remote device, reverse telnet connects to a *local* interface, referenced by port number.

Reverse telnet can only be used if an asynchronous interface exists, and an *IP address* is configured on at least one interface:

Router(config)# *interface loopback 0*
Router(config-if)# *ip address 10.10.10.10*

To connect to a modem off of the *first* asynchronous interface:

Router(config)# *telnet 10.10.10.10 2001*

The port numbers for character-mode access to asynchronous interfaces will range from 2000-2999.

(Reference: http://docs.kde.org/stable/en/kdenetwork/kppp/appendix-hayes-commands.html)

Configuring Asynchronous Connections on a Router

Cisco Routers can have three types of serial interfaces:

- Synchronous interfaces
- Asynchronous/synchronous (A/S) interfaces
- Asynchronous interfaces

An asynchronous/synchronous (A/S) interface operates **synchronously by default**. To force the interface to operate asynchronously:

Router(config)# *interface serial0/4*
Router(config-if)# *physical-layer async*

Other asynchronous interface commands include:

Router(config)# *ip local pool MYPOOL 10.1.1.100 10.1.1.124*

Router(config)# *int serial0/4*
Router(config-if)# *ip unnumbered fastethernet0/0*
Router(config-if)# *encapsulation ppp*
Router(config-if)# *ppp authentication chap*
Router(config-if)# *async mode interactive*
Router(config-if)# *peer default ip address pool MYPOOL*

The *ip unnumbered* command allows the *serial0/4* interface to use the configured IP address on *fastethernet0/0.*

Hosts connecting into this asynchronous interface will receive an *ip address* from the range specified in the *pool* named *MYPOOL.*

The *async mode interactive* command provides EXEC mode access to any host that establishes a connection.

(Reference: CCNP BCRAN Exam Certification Guide, Second Edition. Page 96. Cisco Press ISBN: 1587200848)

Configuring Asynchronous Connections on a Router (continued)

Physical-layer configuration is completed on the *line*, and not the *interface*:

Router(config)# *line 4*
Router(config-line)# *login local*
Router(config-line)# *autoselect during-login*
Router(config-line)# *autoselect ppp*
Router(config-line)# *modem inout*
Router(config-line)# *modem autoconfigure type use_sportster*
Router(config-line)# *stopbits 1*
Router(config-line)# *txspeed 56000*
Router(config-line)# *rxspeed 56000*
Router(config-line)# *flowcontrol hardware*
Router(config-line)# *transport input all*

The *login local* command will force the router to examine its local username and password database to authenticate users accessing this asynchronous line.

The *autoselect during-login* and *autoselect ppp* commands automatically start the PPP process and prompt the connecting host for login credentials.

The *modem inout* command enables incoming and outgoing communication.

The *modem autoconfigure* command specifies the appropriate initialization parameters for the modem, based on the *modemcap database*. To view entries in the modemcap database:

Router# *show modemcap*

The router can attempt to auto-discover the modem type:

Router(config)# *line 4*
Router(config-line)# *modem autoconfigure discovery*

Transmit and receive speeds are set using the *txspeed* and *rxspeed* commands. The *speed* command will set both simultaneously.

The *flowcontrol hardware* command indicates that the RTS and CTS pins will be utilized for flow control.

The *transport input* command specifies the protocols that can gain interactive access to the Cisco device (SSH, Telnet, etc.).

(Reference: CCNP BCRAN Exam Certification Guide, Second Edition. Page 98. Cisco Press ISBN: 1587200848)

* * *