Network protocols

Aside from the issues of the physical network (signal types and voltage levels, connector pinouts, cabling, topology, etc.), there needs to be a standardized way in which communication is arbitrated between multiple nodes in a network, even if its as simple as a two-node, point-to-point system. When a node "talks" on the network, it is generating a signal on the network wiring, be it high and low DC voltage levels, some kind of modulated AC carrier wave signal, or even pulses of light in a fiber. Nodes that "listen" are simply measuring that applied signal on the network (from the transmitting node) and passively monitoring it. If two or more nodes "talk" at the same time, however, their output signals may clash (imagine two logic gates trying to apply opposite signal voltages to a single line on a bus!), corrupting the transmitted data.

The standardized method by which nodes are allowed to transmit to the bus or network wiring is called a *protocol*. There are many different protocols for arbitrating the use of a common network between multiple nodes, and I'll cover just a few here. However, its good to be aware of these few, and to understand why some work better for some purposes than others. Usually, a specific protocol is associated with a standardized type of network. This is merely another "layer" to the set of standards which are specified under the titles of various networks.

The International Standards Organization (ISO) has specified a general architecture of network specifications in their DIS7498 model (applicable to most any digital network). Consisting of seven "layers," this outline attempts to categorize all levels of abstraction necessary to communicate digital data.

- **Level 1: Physical** Specifies electrical and mechanical details of communication: wire type, connector design, signal types and levels.
- **Level 2: Data link** Defines formats of messages, how data is to be addressed, and error detection/correction techniques.
- **Level 3: Network** Establishes procedures for encapsulation of data into "packets" for transmission and reception.
- **Level 4: Transport** Among other things, the transport layer defines how complete data files are to be handled over a network.

- **Level 5: Session** Organizes data transfer in terms of beginning and end of a specific transmission. Analogous to *job control* on a multitasking computer operating system.
- **Level 6: Presentation** Includes definitions for character sets, terminal control, and graphics commands so that abstract data can be readily encoded and decoded between communicating devices.
- **Level 7: Application** The end-user standards for generating and/or interpreting communicated data in its final form. In other words, the actual computer programs using the communicated data.

Some established network protocols only cover one or a few of the DIS7498 levels. For example, the widely used RS-232C serial communications protocol really only addresses the first ("physical") layer of this seven-layer model. Other protocols, such as the X-windows graphical client/server system developed at MIT for distributed graphic-user-interface computer systems, cover all seven layers.

Different protocols may use the same physical layer standard. An example of this is the RS-422A and RS-485 protocols, both of which use the same differential-voltage transmitter and receiver circuitry, using the same voltage levels to denote binary 1's and 0's. On a physical level, these two communication protocols are identical. However, on a more abstract level the protocols are different: RS-422A is point-to-point only, while RS-485 supports a bus topology *"multidrop"* with up to 32 addressable nodes.

Perhaps the simplest type of protocol is the one where there is only one transmitter, and all the other nodes are merely receivers. Such is the case for BogusBus, where a single transmitter generates the voltage signals impressed on the network wiring, and one or more receiver units (with 5 lamps each) light up in accord with the transmitter's output. This is always the case with a simplex network: there's only one talker, and everyone else listens!

When we have multiple transmitting nodes, we must orchestrate their transmissions in such a way that they don't conflict with one another. Nodes shouldn't be allowed to talk when another node is talking, so we give each node the ability to "listen" and to refrain from talking until the network is silent. This basic approach is called *Carrier Sense Multiple Access* (CSMA), and there exists a few

variations on this theme. Please note that CSMA is not a standardized protocol in itself, but rather a methodology that certain protocols follow.
One variation is to simply let any node begin to talk as soon as the network is silent. This is analogous to a group of people meeting at a round table: anyone has the ability to start talking, so long as they don't interrupt anyone else. As soon as the last person stops talking, the next person waiting to talk will begin. So, what happens when two or more people start talking at once? In a network, the simultaneous transmission of two or more nodes is called a *collision*. With CSMA/CD (*CSMA/Collision Detection*), the nodes that collide simply reset themselves with a random delay timer circuit, and the first one to finish its time delay tries to talk again. This is the basic protocol for the popular Ethernet network.
Another variation of CSMA is CSMA/BA (*CSMA/Bitwise Arbitration*), where colliding nodes refer to pre-set priority numbers which dictate which one has permission to speak first. In other words, each node has a "rank" which settles any dispute over who gets to start talking first after a collision occurs, much like a group of people where dignitaries and common citizens are mixed. If a collision occurs, the dignitary is generally allowed to speak first and the common person waits afterward.

In either of the two examples above (CSMA/CD and CSMA/BA), we assumed that any node could initiate a conversation so long as the network was silent. This is referred to as the "unsolicited" mode of communication. There is a variation called "solicited" mode for either CSMA/CD or CSMA/BA where the initial transmission is only allowed to occur when a designated master node requests (solicits) a reply. Collision detection (CD) or bitwise arbitration (BA) applies only to post-collision arbitration as multiple nodes respond to the master device's request.

An entirely different strategy for node communication is the *Master/Slave* protocol, where a single master device allots time slots for all the other nodes on the network to transmit, and schedules these time slots so that multiple nodes *cannot* collide. The master device addresses each node by name, one at a time, letting that node talk for a certain amount of time. When it is finished, the master addresses the next node, and so on, and so on.
Yet another strategy is the *Token-Passing* protocol, where each node gets a turn to talk (one at a time), and then grants permission for the next node to talk when its done. Permission to talk is passed around from node to node as each one hands off the "token" to the next in sequential order. The token itself is not a physical thing: it is a series of binary 1's and 0's broadcast on the network, carrying a specific address of the next node permitted to talk. Although token-passing protocol is often

associated with ring-topology networks, it is not restricted to any topology in particular. And when this protocol is implemented in a ring network, the sequence of token passing does not have to follow the physical connection sequence of the ring.

Just as with topologies, multiple protocols may be joined together over different segments of a heterogeneous network, for maximum benefit. For instance, a dedicated Master/Slave network connecting instruments together on the manufacturing plant floor may be linked through a gateway device to an Ethernet network which links multiple desktop computer workstations together, one of those computer workstations acting as a gateway to link the data to an FDDI fiber network back to the plant's mainframe computer. Each network type, topology, and protocol serves different needs and applications best, but through gateway devices, they can all share the same data.

It is also possible to blend multiple protocol strategies into a new hybrid within a single network type. Such is the case for Foundation Fieldbus, which combines Master/Slave with a form of token-passing. A Link Active Scheduler (LAS) device sends scheduled "Compel Data" (CD) commands to query slave devices on the Fieldbus for time-critical information. In this regard, Fieldbus is a Master/Slave protocol. However, when there's time between CD queries, the LAS sends out "tokens" to each of the other devices on the Fieldbus, one at a time, giving them opportunity to transmit any unscheduled data. When those devices are done transmitting their information, they return the token back to the LAS. The LAS also probes for new devices on the Fieldbus with a "Probe Node" (PN) message, which is expected to produce a "Probe Response" (PR) back to the LAS. The responses of devices back to the LAS, whether by PR message or returned token, dictate their standing on a "Live List" database which the LAS maintains. Proper operation of the LAS device is absolutely critical to the functioning of the Fieldbus, so there are provisions for redundant LAS operation by assigning "Link Master" status to some of the nodes, empowering them to become alternate Link Active Schedulers if the operating LAS fails.

Other data communications protocols exist, but these are the most popular. I had the opportunity to work on an old (circa 1975) industrial control system made by Honeywell where a master device called the *Highway Traffic Director*, or HTD, arbitrated all network communications. What made this network interesting is that the signal sent from the HTD to all slave devices for permitting transmission

was *not* communicated on the network wiring itself, but rather on sets of individual twisted-pair cables connecting the HTD with each slave device. Devices on the network were then divided into two categories: those nodes connected to the HTD which were allowed to initiate transmission, and those nodes not connected to the HTD which could only transmit in response to a query sent by one of the former nodes. *Primitive* and *slow* are the only fitting adjectives for this communication network scheme, but it functioned adequately for its time.

Source: http://www.allaboutcircuits.com/vol_4/chpt_14/7.html