



**DELTA TAU**  
Data Systems, Inc.

## **New Connectivity Strategies in Motion Control**

by Curtis S. Wilson  
Delta Tau Data Systems, Inc.

### *Abstract*

Evolving technology in high-speed serial links has dramatically changed the connectivity options available to integrators of motion control systems. Substantial investment in the technologies underlying office networks and the Internet has essentially eliminated the traditional cost and bandwidth penalties associated with serial communications. Serial networks and buses can now replace both backplane buses and discrete field wiring. These serial links have several origins: from the personal computer world come USB, Ethernet, and FireWire; from the industrial I/O world come Fieldbuses such as DeviceNet and Profibus; from the servo world come motion rings like MACRO and SERCOS.

These new possibilities mean that there motion system designers can get the capabilities of backplane communications without worrying about mechanical fit issues. Additionally, they can eliminate the need to bring all of the field wiring directly back to the central computer. Selecting the appropriate serial bus or network depends on many factors: what devices are to be connected, what update rates are required, what degree of determinism is needed, and the capabilities of each link.

With these new capabilities, the entire structure of a motion controller can be re-evaluated. An example of the new thinking can be found in Delta Tau's UMAC (Universal Motion and Automation Controller). The UMAC consists of a series of 3U-format Euro-cards that can slide in and out of an enclosed rack, communicating over an internal bus. It can communicate to a host computer or PLC over a variety of serial links, including USB, Ethernet, or a Fieldbus. It can serve as either a central "master" or a remote "slave" on a MACRO ring. Field wiring can be brought directly to breakout connectors on the 3U boards.

## *Introduction*

One of the most difficult issues in designing a motion control system is deciding on a connectivity strategy. New technologies have opened up a world of possibilities, but made the decision harder, both due to the sheer number of choices, and to the difficulty in properly comparing competing approaches. This paper provides context and a basis for comparing the different possibilities.

## *Avenues of Communications*

The traditional comparison of physical styles of electronic communications links set up a hierarchy of levels, often listed as follows:

1. Between registers on a single IC
2. Between ICs on a single circuit board (through board traces)
3. Between circuit boards in the same box (through backplane bus)
4. Between boxes in the same room or building (through wire links)
5. More remote communications (modem or wide-area-networks)

The rule of thumb in comparing these links was that for each level more remote in the communications, the cost would be about an order of magnitude more for a unit of bandwidth. This meant that devices requiring high-bandwidth communications had to be in close proximity to each other. Wire communications could only be used in low-bandwidth situations.

However, in recent years this situation has changed significantly. Below we will look at developments in communications on various fronts and how they have affected motion controllers.

## *Traditional Wire Links*

Traditional wire communications links such as RS-232 and RS-422 have been around for a long time. The earliest motion controllers typically used one of these communications links, whether just for configuration in a standalone application, or for active communications in the actual application. These links are still valid in many applications where the required bandwidth of communications is not great. If the motion controller can store entire sequences, the communications can be reduced to sequence selection, start, stop, and “how are you doing?” However, the fact that most implementations cannot exceed 38.4 kbits/sec, and almost none exceed 115.2 kbits/sec, inhibits the use of these links in any application requiring significant data rates. Traditional CNC controllers with RS-232 for a “direct numerical control” (DNC) link would choke to a stop if many quick moves needed to be done.

## *Backplane Buses*

Backplane buses developed seriously in the 1980’s and hit their stride in motion control applications in the 1990’s, with ISA, VME, and STD buses predominant at first, followed by the PCI bus. A key virtue of these buses was and is the very simple method of addressing different devices on the bus: with only a little extra logic, the same processor doing the computation could directly address the different devices on the bus. No additional processor is required to manage the backplane-bus communications, unlike most of the wire links.

The industrial control world made a Faustian bargain with the consumer and office product market to use these backplanes. The industrial world counted on these higher-volume markets to bring unit costs down, but this tied industrial users into the very different dynamics of these markets. In the high volume markets, continuity of product is not very important compared to getting to market quickly with the newest features and the highest speeds. Consequently, industrial users, who prize long availability of a product line, have been buffeted by the almost monthly changes in personal-computer products.

Some of the thorniest issues have been in form factor. A user of a motion-control backplane-expansion board may find suddenly that his personal computer no longer accommodates his board – tall motherboard components may be in the way, or expansion slots reduced in size and/or number.

Special industrial buses such as VME, or even industrial PCs, avoided many of these problems, but also lost many of the cost advantages of the high-volume office market. By the end of the 90's, there was widespread dissatisfaction with the use of backplane-expansion motion-control boards.

#### *Computer Wire Buses and Networks*

Still, even in the office computer market itself, a lot of work went on to reduce the cost and increase the speed of wire communications. There were several driving factors. The most important of these was the need to connect individual computers on a network, first in a single network, then between networks (the Internet). The predominant technology here is Ethernet.

The second factor was the desire to connect outside peripherals to the computer with high-speed links. While several technologies vied for supremacy here, Universal Serial Bus (USB) became the most predominant. Apple's "FireWire" interface was designed for higher speeds, with digital video applications in mind.

Once again, the volume of the office market justified large investments to drive down the unit costs of these links. The result has been a virtual elimination of the cost/performance penalty for wire communications compared to backplane communications. And because of the physical nature of wire links, the form-factor issues that have plagued backplane communications are not

#### *Industrial Fieldbuses*

From the industrial world came a different class of technologies of wire communications. The first class is called "Fieldbus". These systems were designed to simplify and reduce the wiring between controllers and a large set of sensors and actuators, whether in a factory or an end product. Traditional systems had connected all of these remote devices to the controllers with parallel wiring, often resulting in large and long wiring bundles.

A typical Fieldbus would be implemented with serial communications on a "multi-drop" cable, allowing many devices to be daisy-chained on a slender cable. Each device would require some intelligence to handle issues of addressing (often "plug-and-play"), protocol, and arbitration. Most of the Fieldbus designs were optimized for discrete I/O devices.

The most popular of the Fieldbus formats have been CANbus, out of North America, and Profibus, out of Europe. There are several software protocols using the basic CANbus format, the most popular of which are DeviceNet and CANopen.

### *Motion Control Rings*

A couple of wire links have been design especially for controller-to-drive interface. These are distinguished by having built-in features for cyclic (repeated) bi-directional data transfer, and “hard real-time” determinism.

The first of these was SERCOS, developed in Germany. Originally running at 2 Mbits/sec, and more recently at 4 and 8 Mbits/sec, it uses an optical-fiber ring topology to connect the controller to its drives.

Because the cyclic update frequency of the SERCOS ring is not high enough in general to close loops across the ring, SERCOS drives must have the capability to perform at least fine command-position interpolation, plus to close all servo loops and perform any required commutation. This requires sophisticated and relatively expensive drives, plus a complex protocol to handle all of the gains and modes used.

A newer motion ring is MACRO (Motion And Control Ring Optical), developed in the U.S. By employing a very high data rate of 125 Mbits/sec, it permits a ring update fast enough to close any loops required over the ring. This in term permits simple, inexpensive remote nodes, and a very simple communications protocol.

### *Implications*

Now that some of the new communications technologies developed in the 1990’s have matured, what are their implications for motion control systems? It is clear that the heyday of the motion controller as a backplane-expansion card with all the field wiring brought directly back to it is past, for two reasons. The first reason is the trouble in finding a consistent source of inexpensive computers that can physically accommodate a given expansion card. The second reason is the cost, complexity, and unreliability of bringing so many conductors directly back to the main computer.

### *Use of Wire Controller-to-Drive Links*

What will replace this? There look to be two main trends here (for systems keeping a PC as part of the control system). The first trend is to keep at least a minimal version of the controller as a backplane expansion card, but to simplify the wiring to some sort of serial wire link. All of the above-mentioned types of wire links have been used for this purpose, with varying degrees of performance.

Because of the low update rate and lack of determinism in most Fieldbus designs, these links are mainly used to give high-level commands to smart and independent servo drives. This can work well when the motion does not have to be tightly coordinated between axes, but can falter if coordination is required. Several major users of motion-control technology have started down this road, implementing Fieldbus links for simple and uncoordinated axes, and then finding that the approach could not be generalized to coordinated axes. Even when coordination requirements were not tight, this approach can be limiting due to delays in handshaking between tasks – e.g. making sure Motor 7 has finished its move before Motor 4 can start its next sequence.

However, Fieldbus interfaces are increasingly being applied for general-purpose I/O control in motion-control systems. Here the update rates and determinism requirements are not that great.

Some of the computer wire buses are also being used for controller-to-drive links. With higher bit rates and some possibilities for deterministic update rates, these provide more possibilities for tightly coordinated control. Still, these have some serious limits.

USB(1) has a length limitation of 5 meters, with unisolated electrical transmission only. While the raw bit rate of 12.5 Mbits/sec is quite good, the complex collision avoidance algorithms make the effective bit rate at most 20% of that, or 2.5 Mbits/sec. Achieving synchronous transmissions, especially bi-directionally, is difficult, and cannot be done at a frequency greater than 1 kHz. The second-generation USB2 promises much higher bit rates, and an 8 kHz “heartbeat”, but other limitations remain. It should be remembered that the intent of this interface was to connect low-power computer peripherals.

Ethernet can have high bit rates (10 Mbits/sec or 100 Mbits/sec), and can go substantial distances, but its common implementations were not designed with hard real-time applications in mind.

FireWire has seen several (non-compatible) implementations as controller-to-drive links. It has high raw bit rates of 200 Mbits/sec and 400 Mbits/sec, and an 8 kHz synchronous transfer mechanism. However, it also has complex collision-avoidance mechanisms that greatly reduce its effective bit rate to about 20% or the raw rate, leaving unspectacular performance. One prominent implementation can control at most 8 axes, at a 2 kHz update rate. Also the distance between nodes is limited to 4.5 meters, and there is no electrical isolation. Finally, the synchronous transfers necessary in a servo loop can only happen at 8 kHz; no flexibility is provided.

Of course, the links designed especially for motion control – SERCOS and MACRO – fare better in this regard. They can go much greater distances (SERCOS up to 75m between nodes, MACRO up to 3000m between nodes), and their optical fiber media automatically provide complete electrical isolation. They are designed around a bi-directional synchronous cyclic data-transfer mechanism essential for smooth operation of difficult profiles and for passing data. They have more variable cycle update rates than most other mechanisms.

However, due to SERCOS’ relatively low communications rates, in many applications loops must be closed in the drive. For this reason, all SERCOS drives must support sophisticated positioning functions, even if they are not used in a given application, driving up the cost and complexity of the drives, and the difficulty of setup.

MACRO’s extremely high data rate permits a very fast loop update rate. This in turn permits all loops to be closed in the central controller, and a very simple protocol to be implemented.

The processor at the remote slave node (if required) just has to copy data between the ring registers and the machine-interface I/O registers, and to monitor the ring for failures. These processes are simple enough that it is possible to perform them in dedicated ASIC circuitry if flexible configurability is not required.

MACRO thus provides the best of both worlds: highly distributed hardware to simplify the wiring, but highly centralized software to simplify inter-task communications. Designers have been pursuing distributed hardware solutions for many years to simplify system wiring; many have (mistakenly) believed that this requires highly distributed software, which in turn requires complex, high-level protocols to communicate between tasks.

The vast majority of designers prefer the simplicity of centralized software when possible; a link such as MACRO permits it. This provides the capability for easy implementation of tasks that would be difficult, if not impossible in a distributed system: control laws that are cross-coupled between axes; adaptive control schemes that are dependent on the states of multiple axes, and the like.

#### *Use of Wire Computer-to-Controller Links*

A second strategy in the design of motion control systems is to make the “long wire” the connection between the host computer and the motion controller. With the new fast wire links now available and cost-effective, this is proving a very popular approach. More options are available here, because the required bandwidth and determinism requirements of this link are generally not so high. Of course, to use this approach optimally, it is usually best not just to take a former backplane-bus expansion card, put a wire link on it, and move it out of the central computer (though in some cases this can be appropriate).

Now that the necessity for having the control board fit in the backplane bus is gone, a lot of former constraints disappear. The expansion card virtually requires cable links to “breakout boards” because the field wiring cannot feasibly be brought directly to or inside the computer. With the controller outside the computer, a better strategy for the field wiring can be found.

As examples of the new strategies, Delta Tau has introduced its new QMAC™ and UMAC™ controllers. The QMAC is a “boxed” controller for standard applications of up to 4 axes. While capable of standalone operation, it can provide RS-232, USB, and Ethernet communications links to a host computer.

Internally, the QMAC consists of two circuit boards: a control board and a connector board. The control board has a Euro-card format and is mounted to the side of the box. The connector board mounts to the control board at right angles (from the top, the two boards form an “L” shape) with a simple snap-in connection, with its field-wiring connectors protruding through the front of the box. For the manufacturer, this design yields quick, simple, and inexpensive assembly. For the user, it provides easy access to system wiring.

The UMAC controller provides a modular solution for larger and more varied applications. It consists of a series of 3U-format Euro-cards (100mm x 160mm) that can slide in and out of a standard Euro-rack, communicating with each other over a common backplane. It can either be configured as a full controller, using a Turbo PMAC CPU board as one of the 3U boards (creating a “UMAC Turbo”), or as a remote slave node on a MACRO link, using a MACRO interface/CPU board instead.

The UMAC can provide RS-232/422, USB, and Ethernet communications links to a host computer. In addition, it can act as a Fieldbus slave so it can be commanded from a host PLC on that PLC’s Fieldbus network. (It can also act as a Fieldbus master to

In both cases, the other 3U-format cards have both interface circuitry and the breakout connectors for direct connection of field wiring. A wide variety of these interface/breakout boards is available, both for motion I/O (analog and digital) and general-purpose I/O (analog and digital).

Internally, communications over a backplane still makes sense here. The update rates are in the thousands, or even tens of thousands of Hertz; the distances are not great, and it would be expensive or cumbersome to process everything through wire-link processors.