Motion Control Networks

Chuck Lewin, President & CEO of Performance Motion Devices

The trend toward distributed motion control and motion networks is driven by a desire to reduce wiring, lower cost, and increase reliability. When launching a development project to tap into these benefits, you will face choices regarding network buses, protocols, and a host of technical issues. In the end it is the architecture and physical configuration of your machine that will dictate which motion network you select and how you distribute the control problem. The cost savings and flexibility offered by motion networks can be substantial, but the key is knowing what kind of solution will work best in your control application.

Architecture

Architecture means the structure and organization of the control problem. Broadly speaking there are flat motion control applications, where a number of motors all must be controlled more or less equally by the central PC (we use PC here to mean the software program that controls the overall flow of the machine but this could be a microprocessor, or even a PLC), and there are hierarchical applications where the axes are clustered into 2, 3, or more functional axes. Figures 1 and 2 show this.

An example of a flat motion control problem is a printing press with multiple servo-controlled spools. In this application timing is critical, and the central controller, usually a PC or PLC, must drive all axes in synchrony. Typical commands in such a system are "move axis #1 to position X, move axis #2 to position Y," etc.

An example of a hierarchical motion control application is a semiconductor wafer handling system that has a central robot (4 axes), a wafer aligner (3 axes), and a valve controller (1 or 2 axes). In this architecture the network typically connects the local robot or valve controllers to a central PC, but the actual motion control is local to the robot, aligner, or valve. Thus the overall machine controller doesn't give commands such as "move robot axis #2 to position 12345," it gives commands such as "extend robot arm" which the local robot controller interprets and executes.

Time to be responsive

When using a motion network, try to anticipate the kinds of signaling that will be required in your application. Does the behavior of the motion depend on the status of signals located on another part of the machine? Will you place sensors, and other non-motion controlled actuators, such as relays, on the network bus? How quickly does the motion have to shut down if an error occurs? Depending on the answers to these questions you may be able to place some of the systems on a motion network, none of them, or all of them. The answers to these questions will, at a minimum, influence the type of network you choose.

Mechanical configuration

Another important consideration regarding how, and how much, you can use a network-based approach, is the mechanical organization of your machine. This issue addresses questions such as "How will the machine be serviced if electronics are physically distributed throughout the machine?" The cost reduction anticipated due to reduction in wiring must be compared against the cost of servicing the whole system in the field. Although the traditional card rack that the technician services may be a mess of wires, there is something to be said for

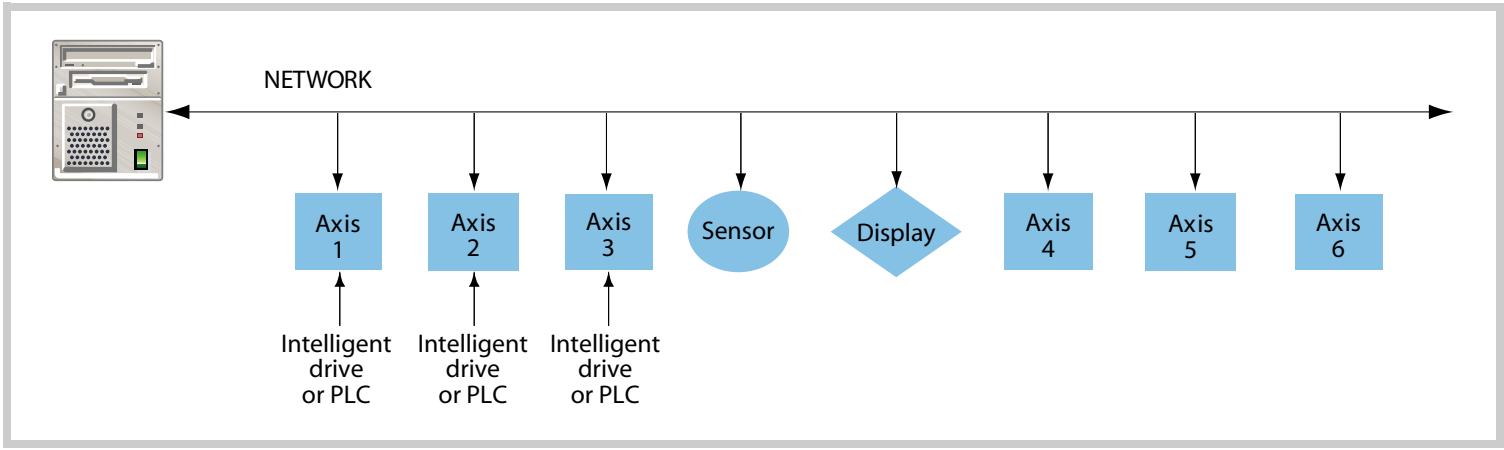


Figure 1. In a **flat motion control application**, all motors are directly controlled by the central PC.

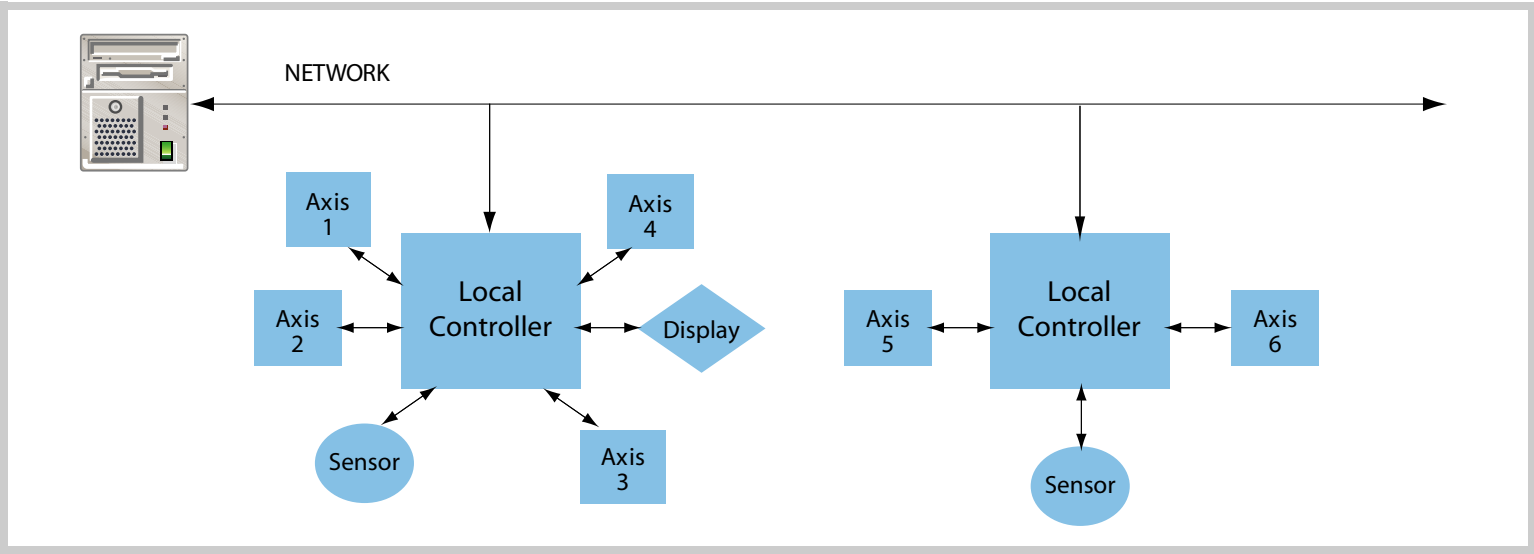


Figure 2. Axes in a **hierarchical motion control applilcation** are grouped together and controlled locally.

keeping everything under-one-roof. Serviceability and lifetime ownership cost issues strongly affect control system design choices.

Remember also that distributing the control by placing amplifiers near the motors may not always be feasible for weight, heat, or other environmental reasons. The traditional control rack cabinet can be air conditioned and insulated from the machine environment relatively easily but is often not possible if the controls are distributed.

Motion network buses: It's a jungle out there

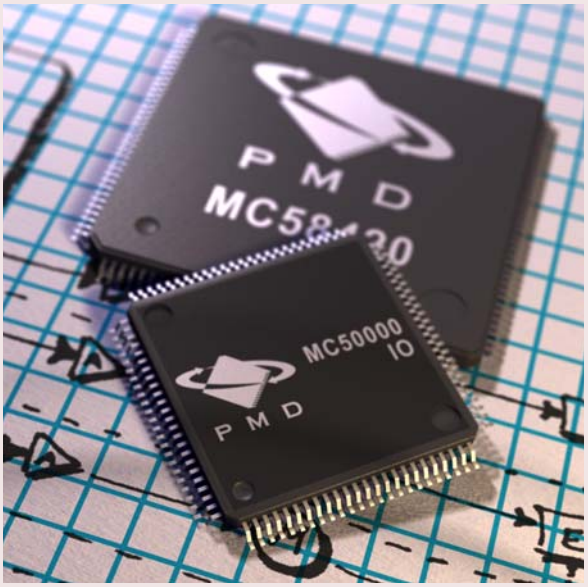
Now that we have looked at some of the characteristics we should be aware of in our machine application, let's look at the networks that are actually available, and discuss some overall concepts related to network bus selection.

First of all it is important to distinguish between dedicated motion buses, such as SERCOS, which are often proprietary and generally expensive, from buses which *can* be used in motion applications but which are also used in other industries such as CANbus and Ethernet. In this article we will mention dedicated motion buses, but focus mainly on the general-purpose buses. This distinction is important, because if you are going to use a dedicated motion bus, unless you happen to work for a motion control vendor, you will most likely just purchase a complete system and interact with it as a user. Buses such as SERCOS are not easily extended to handle non-motion sensors or actuators, and thus make poor general-purpose networks.

While we are looking at networks for motion control, what about the protocol that we will execute on these networks? Well, if network buses are many and varied, motion protocols are few

A relatively recent development in the field of motion networks is the availability of motion processors that directly support a network node connection. Motion processors are dedicated ICs that process motion language commands such as "move to this location" or "start the motion after this signal goes active" but have special hardware that lets them directly interface to DC servo, brushless DC, or step motors.

Products such as the MC58000 Series from Performance Motion Devices fall into this category. They provide all the usual motion features such as profile generation, servo loop closure, and commutation, but add a complete CANbus network connection that make it easy to construct low cost multi-axis motion networks.



Network Name	Comments	Typical Speed Range
RS-485	Low cost, easy to use. Slow.	< 1Mbit/sec
CANbus	Low cost, easy to use. Available at chip as well as module level	.5 - 5Mbit/sec
Ethernet	Not deterministic, but fast. Widely adopted.	10Mbit/sec - 1Gbit/sec
FireWire (IEEE 1394)	High speed and deterministic. Has length limitation.	400Mbit/sec
Profibus	Good all-around device control bus, but popular in Europe, not in the US	12 Mbit/sec
USB 2.0	High speed, low cost PC peripheral interconnect. Not common as motion network	12 Mbit/sec - 480Mbit/sec

and far between. What higher-level protocols do exist typically do not cater to motion. Most popular are DeviceNet and CanOpen, which are protocols that are hosted on the CANbus network. Both of these higher-level networks are well defined, so that today it is possible to buy ready-made sensors and components for CanOpen or DeviceNet. A fully accepted, real-time standard for motion control over these buses does not yet exist however, so many users who choose DeviceNet build custom motion extensions, or accept the performance reduction inherent in supporting these higher level network layers.

Your motion network

So with those introductions under our belt, lets get down to brass tacks and start naming names. Here are brief summaries of the most popular buses used for motion networking today. These networks are Ethernet, CANbus, RS485, and FireWire. Rest assured that there are many other choices available, in particular Profibus (popular in Europe), Foundation field bus, USB (the reigning PC interconnect bus-of-choice), SERCOS, LonWorks, and others. But for various reasons, these buses are not in the mainstream of today's general-purpose designs, with the possible exception of Profibus in Europe.

Ethernet is not deterministic in its native, full protocol mode, but can be made more deterministic by stripping some of the higher levels away. Ethernet has speed ranges from 10MB/sec to 1 gigabit per second and is sold in very high volumes, so costs are low for chips and hardware. For motion applications, Ethernet is commonly used in Semiconductor Capital Equipment industry, and other high value, high performance products.

CANbus started as an interconnect system for the automobile world. It evolved to be a popular device-level interconnect bus for general industrial use, and many motion users have adopted it due to its robustness and ease of use. DeviceNet and CanOpen are protocols on top of CANbus that are sometimes used to gain access to standard, off-the-shelf sensors and actuators. CANbus is used in a wide range of industries including medical automation, packaging, liquid dispensing, general automation, and others.

RS-485 and its related cousins RS-422, and RS-232 are surprisingly popular for use in communicating between motion modules. Many of the existing all-in-one "integrated motors" now on the market use RS-485. What they lack in protocol sophistication and automatic error checking, they make up for in simplicity and low cost. RS-485 and similar serial buses are used in industries whenever performance requirements are modest, and cost sensitivity is high.

FireWire (also called IEEE 1394) was developed by Apple Computer for use in video processing. It has attracted motion-world attention because of its high speed and determinism. FireWire has been adopted by a number of motion vendors as the "ideal" motion bus but users who are building their own network-based control system do not commonly use it. FireWire has a length limitation of 10 ft (depending on data rate, and which can be extended with repeaters) and if its use in the PC world declines, it is unclear how much ongoing support there will be from major chip vendors. FireWire is commonly used in highly synchronized applications such as machine tools.

Pick a bus, any bus

For most users who are designing their own machine, bus choice really comes down to three choices: RS-485 (yes, the old standby), CANbus, and Ethernet. Firewire and USB have enough problems associated with them, and enough of a learning curve, that there is very little activity at the user level in constructing systems using these buses. Again, if you are going to purchase a proprietary motion system that uses a network bus, then some of these considerations don't really matter. But if you are going

to construct a machine control system from an open, expandable, bus, those are the choices that you will most likely select from.

CANbus is probably the most widely used bus for "inside-the-machine" networking. CANbus is faster than RS-485, but slower than Ethernet. What matters is whether it is adequate for the application at hand. And for a large number of users, it is.

Ethernet is the bus of the present, and the future. What it lacks in determinacy, it makes up for in speed. Its biggest drawback is the complexity of protocol layers. This may be fine when communicating with multi-axis robots or intelligent controllers, but becomes a nuisance when the distributed module is supposed to be simple and cheap. Still, Ethernet is pushing further and further into the box. New microprocessors are being released which can perform real time I/O tasks as well as host a complete Ethernet protocol stack.

Four applications

Lets briefly take a look at four common motion applications to see how their controls might use, or not use, networks. While not an all-inclusive list, these applications are intended to represent a fairly broad spectrum of industries and products that use motion control. The four applications will be called *conveyer belt*, *standalone, local controller*, and *synchronized multi-axis*.

Conveyer-belt. Typical applications: packaging, factory floor automation, continuous flow processing, medical liquids analyzers.

This application involves motion that occurs over a relatively large distance. There tends to be a variety of actuators used including servo motors, cutters, valves, etc. In this application the primary motion elements tend to be standalone drives or intelligent amplifiers which control a single motor, and which are loosely linked to other sensors and actuators. Sub-millisecond synchronization is generally not required here, and the interactivity tends to be localized and autonomous. That is, "move when this local sensor goes active" etc. In this kind of application the network dramatically reduces the cost of wiring because of the large number of axes, and the relatively large distances involved. This application would be well served either by CANbus or one of the higher-level protocols CanOpen and DeviceNet. Ethernet is also an option here, and if Ethernet-based off-the-shelf actuators can be purchased, this option may be an equal if not preferable approach.

Standalone machine. Typical applications: semiconductor and medical robotics, pointing systems, scientific instruments.

This application describes a machine that is relatively small (usually from a few cubic feet in size to no more than closet-sized.) In this application there is a serious question as to whether networking is even warranted, because distribution of electronics throughout the machine often causes more servicing problems than it solves, despite the total reduction in number of wires. If networking can be used to an advantage, then one of the device networks such as CANbus could be used. To communicate to the "outside" world, any number of networks can be used, but Ethernet would probably be the best choice.

Hybrid controller. Typical applications: Chemical Processing, Pharmaceuticals, Semiconductor Equipment, some general automation, some medical systems

These applications include both standalone and conveyor-belt type control elements, blending the need to communicate between intelligent modules and the need to provide direct control of device-level actuators and sensors. Ethernet, CANbus, or RS485 can work in this application, depending on the bandwidth requirement.

Synchronized multi-axis. Typical applications: Machine tools, plotters, measuring machines.

This application entails systems that perform highly synchronized multi-axis motion. For this application SERCOS may be a good choice, and for the more adventurous, FireWire. Be aware that you may find that a traditional multi-axis card-based centralized controller is every bit as cost effective and reliable as a distributed controller for this type of application

Conclusion

When making motion network choices you should carefully consider the configuration of your control architecture, your timing and latency response requirements, and your machine servicing issues. There is no one-size-fits-all approach to motion control networks, so try to make choices that will allow expansion and re-organization of the controller in the future.

Performance Motion Devices, Inc

55 Old Bedford Road
Lincoln, MA 01773
e-mail: info@pmdcorp.com

www.pmdcorp.com

About Performance Motion Devices

Performance Motion Devices (PMD) is the recognized world leader in motion control ICs, cards, and modules. Dedicated to providing cost-effective, high performance motion systems to OEM customers, PMD utilizes extensive in-house expertise to minimize time-to-market and maximize customer satisfaction.