Control System Overview

The controller is an essential part of any motion control system. It determines speed, direction, distance and acceleration rate – in fact all the parameters associated with the operation that the motor performs. The output from the controller is connected to the drive's input, either in the form of an analog voltage or as step and direction signals. In addition to controlling one or more motors, many controllers have additional inputs and outputs that allow them to monitor other functions on a machine (see Machine Control, p. A45).

Controllers can take a wide variety of forms. Some examples are listed below.

Standalone – This type of controller operates without data or other control signals from external sources. A standalone unit usually incorporates a keypad for data entry as well as a display, and frequently includes a main power supply. It will also include some form of nonvolatile memory to allow it to store a sequence of operations. Many controllers that need to be programmed from a terminal or computer can, once programmed, also operate in standalone mode.

Bus-based – A bus-based controller is designed to accept data from a host computer using a standard communications bus. Typical bus systems include STD, VME and IBM-PC bus. The controller will usually be a plug-in card that conforms to the standards for the corresponding bus system. For example, a controller operating on the IBM-PC bus resides within the PC, plugging into an expansion slot and functioning as an intelligent peripheral.

PLC-based – A PLC-based indexer is designed to accept data from a PLC in the form of I/O communication. Typically, the I/O information is in BCD format. The BCD information may select a program to execute, a distance to move, a time delay, or any other parameter requiring a number. The PLC is well suited to I/O actuation, but poorly suited to perform complex operations such as math and complicated decision making. The motion control functions are separated from the PLC's processor and thus do not burden its scan time.

X Code-based – X Code is a command language specifically developed for motion control and intended for transmission along an RS-232C link. Controllers using this language either accept real-time commands from a host computer or execute stored sequences that have been previously programmed. The simplicity of RS-232C communication allows the controller to be incorporated into the drive itself, resulting in an integrated indexer/drive package.

X Code Programming

X Code has been designed to allow motion control equipment to be programmed by users with little or no computer experience. Although the language includes more than 150 commands, depending on the product, it is only necessary to learn a small percentage of these to write simple programs.

Most command codes use the initial letter of the function name, which makes them easy to remember. Here are some examples of frequently used commands.

V – velocity in revs/sec
D – distance in steps
A – acceleration rate in revs/sec^2
G – go; start the move
T – time delay in seconds

A typical command string might look like this:

V10 A50 D4000 G T2 G

This would set the velocity to 10 revs/sec, acceleration to 50 revs/sec^2 and distance to 4000 steps. The 4000-step move would be performed twice with a 2-second wait between moves.

Please refer to specifications of X Code products for a list of all the available X Code commands.

Single-axis and Multi-axis Controllers

A single-axis controller can, as the name implies, only control one motor. The controller in an integrated indexer/drive comes into this category. However, such units are frequently used in systems using more than one motor where the operations do not involve precise synchronization between axes.

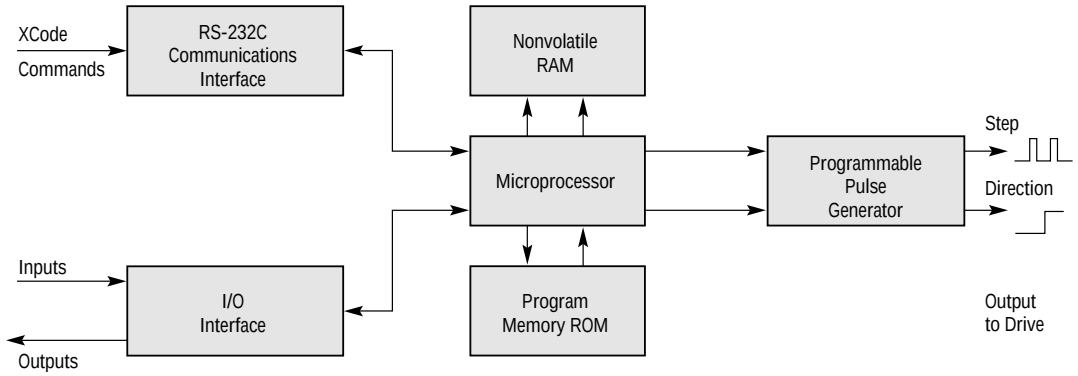
A multi-axis controller is designed to control more than one motor and can very often perform complex operations such as linear or circular interpolation. These operations require accurate synchronization between axes, which is generally easier to achieve with a central controller.

A variant of the multi-axis controller is the multiplexed unit, which can control several motors on a time-shared basis. A printing machine having the machine settings controlled by stepper motors could conveniently use this type of controller when the motors do not need to be moved simultaneously.

Hardware-based Controllers

Control systems designed without the use of a microprocessor have been around for many years and can be very cost-effective in simpler applications. They tend to lack flexibility and are therefore inappropriate where the move parameters are continually changing. For this reason, the hardware-based controller has now given way almost exclusively to systems based on a microprocessor.

Fig. 5.2 Processor-based controller



Processor-based Controllers

The flexibility offered by a microprocessor system makes it a natural choice for motion control. Fig. 5.2 shows the elements of a typical step and direction controller that can operate either in conjunction with a host computer or as a stand-alone unit.

All the control functions are handled by the microprocessor whose operating program is stored in ROM. This program will include an interpreter for the command language, which may be X Code for example.

X Code commands are received from the host computer or terminal via the RS-232C communications interface. These commands are simple statements that contain the required speed, distance and acceleration rate, etc. The processor interprets these commands and uses the information to control the programmable pulse generator. This in turn produces the step and direction signals that will control a stepper or servo drive.

The processor can also switch outputs and interrogate inputs via the I/O interface. Outputs can initiate other machine functions such as punching or cutting, or simply activate drive panel indicators to show the program status. Inputs may come from sources such as operator pushbuttons or directional limit switches.

When the controller is used in a standalone mode, the required motion sequences are programmed from the host and stored in nonvolatile memory (normally battery-backed RAM). These sequences may then be selected and executed from switches via the I/O interface or from a separate machine controller such as a PLC.