WHITE PAPER

How to choose an Industrial Automation Controller

Choosing the most effective controller requires careful evaluation of multiple requirements.

By Jeff Payne
Product Manager, at AutomationDirect

AUTOMATIONDIRECT.com
The #1 Value in Automation.™

WHITE PAPER

There are many important items to consider when choosing a controller for machine and process automation. Breaking down the equipment's operational needs is a starting point and will help evaluate the range of controllers specified by OEMs or machine builders. Depending on how the equipment fits into the larger manufacturing environment, the automation system can provide a complete solution or just control individual parts.

The controller specified, such as a PLC or PAC, can control a single station, a machine, a process unit, a whole assembly line or an entire plant. If an integrated manufacturing system is being automated, a single large controller using multiple expansion and remote I/O bases communicating via Ethernet can provide end-to-end control. However, another application may require compartmentalizing the automation by breaking the system into multiple, logical sections. In this case, the automation may be split and spread among smaller PLCs or even micro PLCs, depending on the demand and functionality (**Figure 1**).

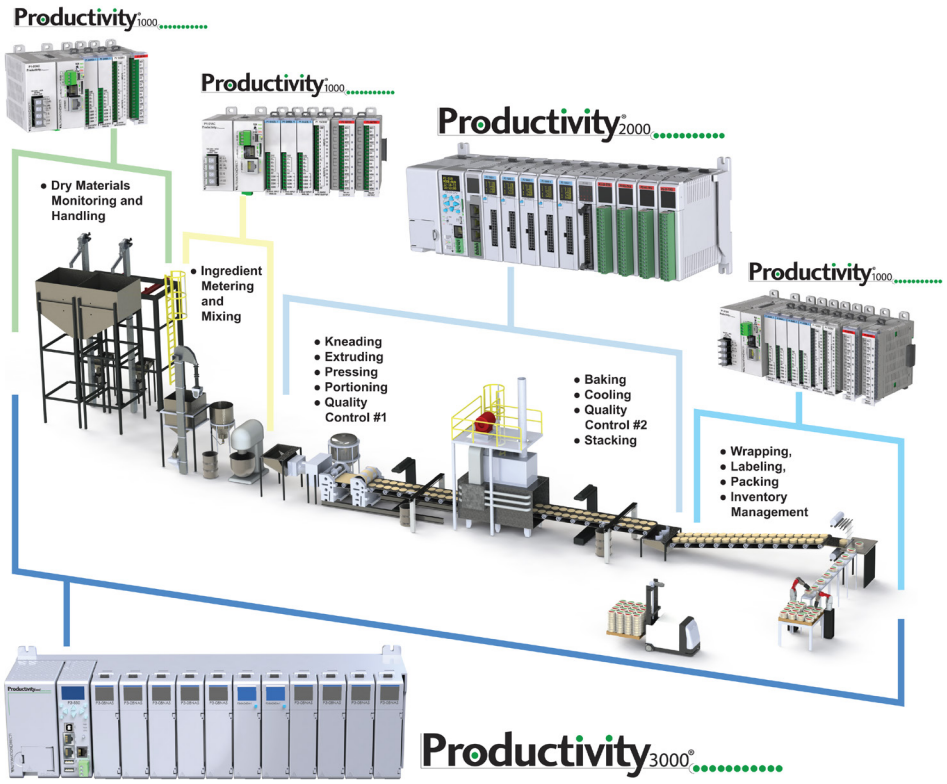

Figure 1: In this block diagram, each module is a machine that can be controlled separately by a smaller PLC, or together by a larger PLC.

Most automation engineers would see this as an irreversible decision as these two choices suggest vastly different platforms, but such does not have to be the case. Some controller families, such as the Productivity Series PLCs, offer several different size options, each using the same programming software (**Figure 2**). The single programming environment provides application flexibility while saving time and money because programs can easily be converted or moved from one PLC to another PLC for compatibility among projects.

WHITE PAPER

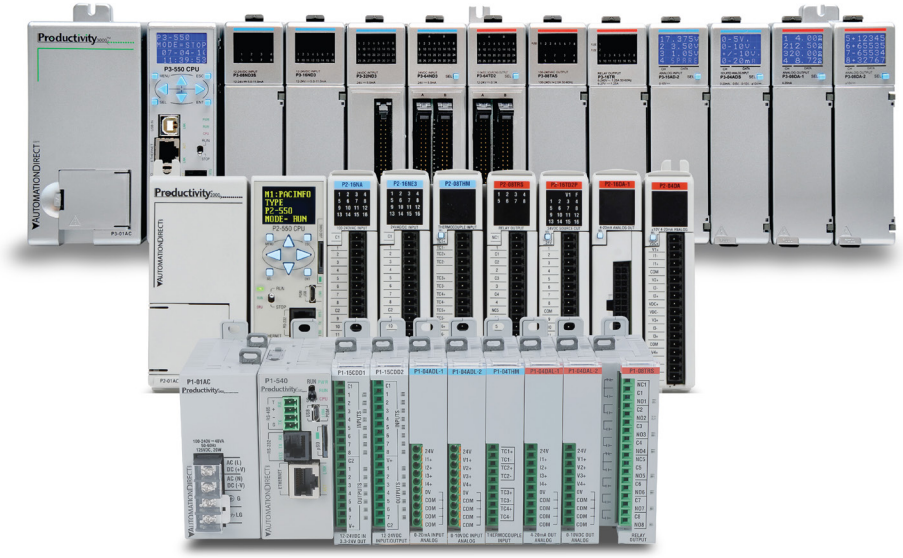

Figure 2: These AutomationDirect Productivity Series 1000, 2000 and 3000 PLCs are different size controllers, but each uses the same programming software.

The difficult part can be deciding whether to run a single program on a large PLC, or to deploy the same project on multiple smaller PLCs, each only executing the parts of the program needed to run the specific subsystem.

To help decide which is the best controller to use in your application, the Table provides a list of factors to consider. It's more complex than simply picking a PLC, PAC or PC-based controller—size, capabilities and functions all enter into the discussion.

Table: Factors to consider when choosing a controller

- Automation - new or existing system
- Environmental issues
- Discrete devices
- Analog devices
- Loop control
- Specialty modules or features
- I/O locations (local and remote)
- Communication
- Programming

Whether the system is new or existing often dictates many of the critical factors for selection. If there are products already installed, it's a good practice to make the new system compatible with them. Some controller products are not compatible with others, even from the same manufacturer.

If extreme environmental conditions exist, ambient temperature limits can be a big issue. A typical controller has an operating temperature range of 0 - 55 °C (30 – 130°F), but actual conditions on the plant floor or specific codes in force at the facility may demand a design to a tougher standard.

Number, Types and Location of I/O

With some of the system-level items out of the way, defining the I/O count and field device types is next on the list. It is good practice to list all the discrete inputs and outputs on a spreadsheet—and to define each type such as analog sensor, digital sensor, solenoid, actuator, control valve and so on. Include the signal type, power requirement, communication protocol and other considerations.

The number of I/O points and types defined has a big effect on the control platform selected. A common mistake is to select a controller able to handle immediate needs but without room for future expansion. Including room to accommodate an extra 20 percent I/O can prevent major difficulties down the road. At the same time, be aware that some controllers limit certain types of I/O, especially analog and specialty I/O such as high-speed inputs or outputs. These may be just as problematic a constraint.

The I/O spreadsheet should also list function and signal level for all the analog devices needed. This includes individual totals for voltage loop, current loop, thermocouple and RTD inputs—with summary totals for voltage and current outputs. The controller specifications must be checked to ensure the total number of analog inputs and outputs are supported, as well as the signal levels.

Specialty inputs and outputs, or intelligent modules, must also be broken down and listed in the I/O spreadsheet. Specialty items include real-time clock, high-speed counter, high-speed output, positioning, servo/stepper motors and others.

The specialty functionality needed for an application may not be supported by a controller. Don't assume every controller can tell the time, or has advanced or even simple motion control functions. Understanding the application requirements and controller capabilities is a must to ensure all features needed now and in the foreseeable future are available.

The physical location of the I/O terminals with respect to the field devices should also be carefully defined and added to the spreadsheet. This modular breakdown will help lay out the local and remote I/O needs, and what real-time communication protocols may be required. Some installations keep things local, whereas others rely heavily on remote I/O, or a combination of both.

If there are long distances between the controller and subsystems, remote I/O is a good option instead of long wire runs to individual field devices. The communication method and speeds supported must be adequate for the application. Serial and Ethernet-based I/O are just some of the options. Industrial Ethernet protocols such as EtherNet/IP are popular, along with various versions of Modbus and others.

It's Time to Communicate

In addition to distributed I/O, communication among multiple PLCs, peripheral devices and enterprise-level systems may be required. The extent of these communication

needs must be determined early in the process with the expectation that whatever they are now, they will only get more complex going forward. Some controllers may only have one or two communication ports, one of which may be used for programming only. The controllers may also not support the most popular protocols, or a specific protocol needed for a critical application.

Communication to other systems, HMIs and field devices via industrial Ethernet or serial communication needs to be defined. With the rapid growth of Internet of Things applications, more comm ports and communication options are always better. Make sure there are one or two extra Ethernet ports, a serial port, a USB port and other configurable options available (**Figure 3**).

Figure 3: AutomationDirect Productivity Series CPUs offer the ability to communicate via industrial Ethernet, serial and USB connections.

Specify which Ethernet protocols—such as EtherNet/IP, Modbus TCP and others, along with serial and ASCII protocols—are needed. This will help select a controller able to support current and future requirements.

Hardware Requirements

Some hardware requirements to consider are the amount of memory, scan-time speed and battery backup. The controller will need sufficient system memory to support both data and program requirements.

Determining how many devices need to be supported by the system helps with the required data memory estimates. Data memory is used for both variable storage, and for dynamic data manipulation. Preset setpoints, accumulated time/counts and other internal flags in timers and counters are examples of data memory users.

The need to store historical data in the controller can call for a much larger data table size. Careful detail of data logging requirements, access methods to get to the data, and interfaces to HMI/SCADA and historian databases should be specified. Networking, protocol and the memory needs are all import requirements for connection to the Industrial Internet of Things.

The program size and the types of instructions used also affect program memory needs. Larger programs with many sequences, sophisticated control functions and fault logic may increase memory needs. Estimating controller memory needs based on the number of program rungs and data files may be possible, but some controllers have tag-name based programming, while others have fixed but expandable data

tables of different types. Some controllers also store documentation in the controller program memory.

Different program instructions have different memory needs, usually noted in the programming manual. The amount of memory use by programs and data tables varies widely between controllers. A useful rule of thumb suggests five to 100 words of memory for each discrete I/O device, and 25 to 500 words of memory for analog I/O, but complex applications make it difficult to estimate. A better way is to develop some preliminary code for a portion of the application and check actual memory usage.

Fast cycle times on a machine need all the help they can get from the controller. Often a fast controller scan time is a requirement. The controller CPU speed and instruction execution speed are both factors as a controller may have faster Boolean logic, yet be slower when executing data handling instructions.

Software Requirements

While the software platform and programing methods are often a matter of personal choice, functional requirements are not. The availability of PID loops, floating-point math, drum sequencing, program interrupts and subroutines must be considered in the selection process.

Some controllers don't support all program instructions necessary for a specific application. An example of this is PID loop function. It's much easier to use built-in PID instructions, if available, instead of writing custom code to support closed-loop process control needs. The number of PID loops required is often underestimated, so the application and controller support should both be checked. Look carefully at all programming functions required.

Drums, sequencers and real-time clocks are other programming needs may also be necessary for a successful control system and application. Many other factors may enter into the discussion, but performing a thorough analysis of the points presented here will be a good start to selecting the right controller for your application.