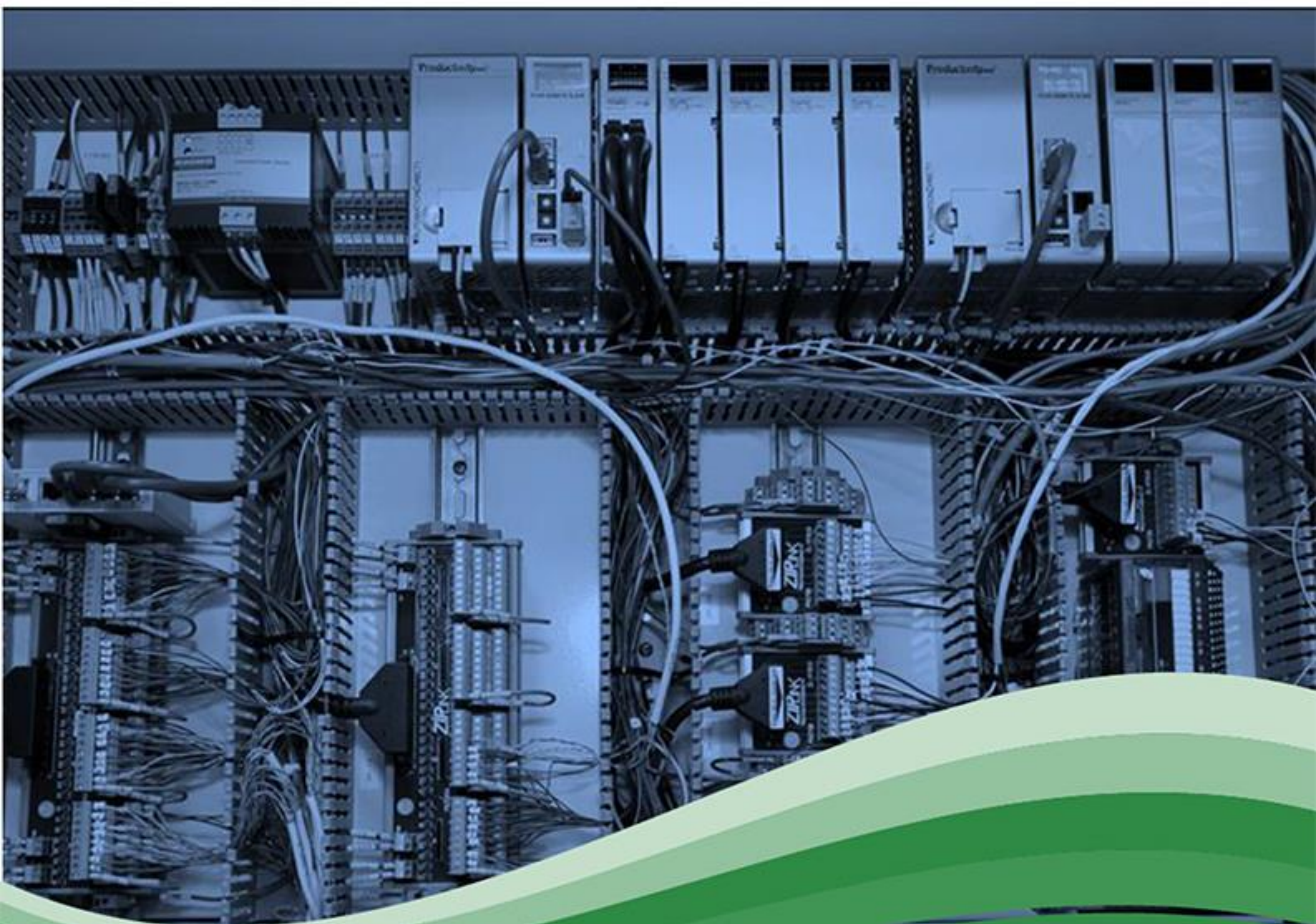
PLC Handbook

Practical Guide to Programmable Logic Controllers



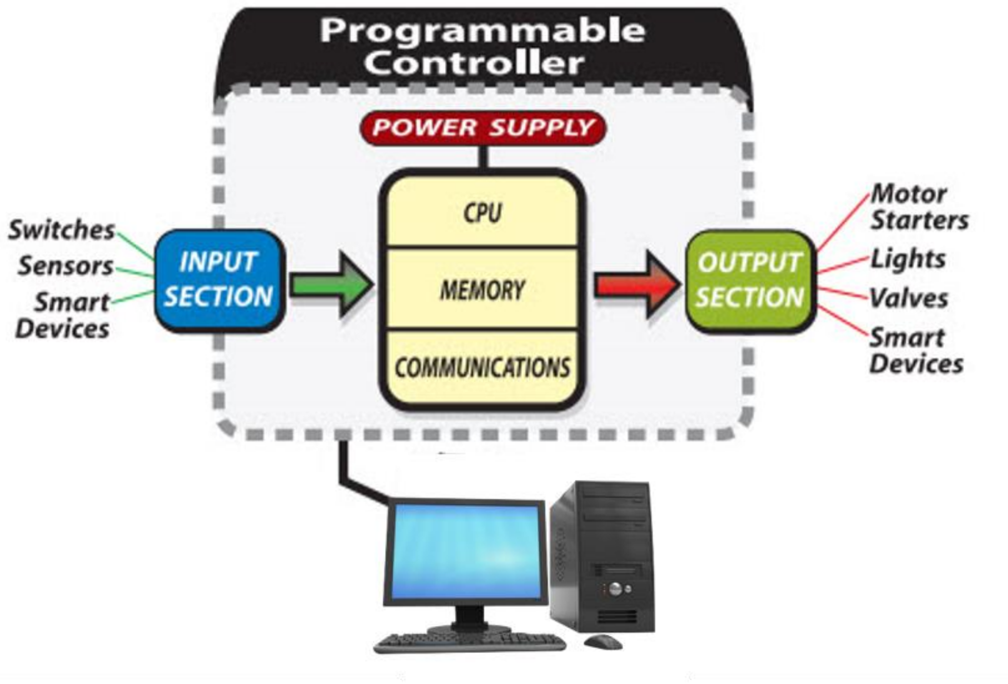
AUTOMATIONDIRECT.com
the #1 value in automation

Contents

Chapter 1

What is a PLC…

Programmable Logic Controllers (PLC) are often defined as miniature industrial computers that contain hardware and software used to perform control functions. More specifically, a PLC would be used for the automation of industrial electromechanical processes, such as control of machinery on factory assembly lines, amusement rides, or food processing. They are designed for multiple arrangements of digital and analog inputs and outputs with extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. A PLC will consist of two basic sections: the central processing unit (CPU) and the Input/Output (I/O) interface system.



The CPU controls all system activity primarily through its processor and memory system. The CPU consists of a microprocessor, memory chip and other integrated circuits to control logic, monitoring and communications. The CPU has different operating modes. In programming mode the CPU will accept changes to the downloaded logic from a PC. When the CPU is placed in run mode it will execute the program and operate the process. Input data from connected field devices (e.g., switches, sensors, etc.) is processed, and then the CPU “executes” or performs the control program that has been stored in its memory system. Since a PLC is a dedicated controller it will process this one program over and over again. The time it takes for one cycle through the program is called scan time and happens very quickly (in the range of 1/1000th of a second, depending on your program). The memory in the CPU stores the program while also holding the status of the I/O and providing a means to store values.

The input/output system is physically connected to field devices and provides the interface between the CPU and its information providers (inputs) and controllable devices (outputs). After the CPU processes the input data (input scan), it will then make any needed output changes after executing the user program (output scan). There are four basic steps in the operation of all PLCs: Input Scan, Program Scan, Output Scan, and Housekeeping. These steps continually take place in a repeating loop.

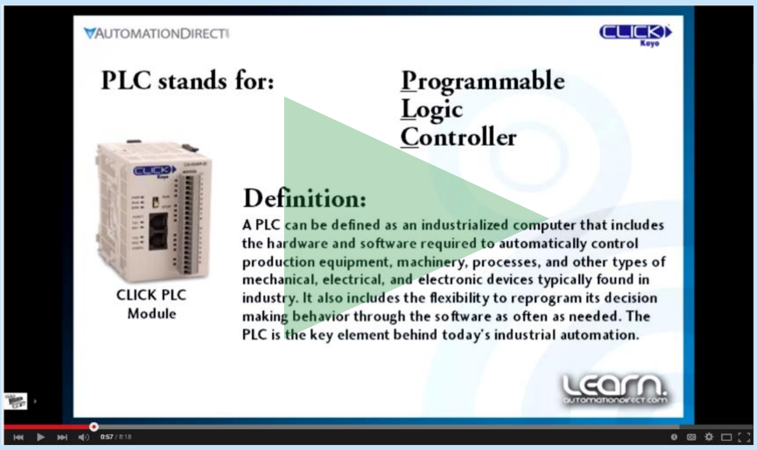
- **Input Scan** – Detects the state of all input devices that are connected to the PLC
- **Program Scan** – Executes the user created program logic
- **Output Scan** – Energizes or de-energizes all output devices that are connected to the PLC
- **Housekeeping** – Includes communicating with programming devices and performing internal diagnostics

Typical PLCs have a wide range of I/O modules available to accommodate all kinds of sensors and output devices. For example, discrete input modules can be used to detect object presence or events with devices such as proximity or photoelectric sensors, limit switches and pushbuttons. Discrete output modules can control “ON/OFF” loads such as motors, lights, and solenoid valves. Analog input modules can accept signals from process instrumentation such as flow, pressure, temperature and level transmitters. These modules can interpret the signal and present a value within a range determined by the devices’ electrical specifications. Analog outputs will command loads that require a varying control signal, such as panel meters, variable frequency drives or analog flow valves. Many PLCs also offer specialized modules such as high-speed I/O or motion control, and serial or Ethernet communications.

The greatest benefit of automating with a Programmable Logic Controller is the ability to repeat or change and replicate the operation or process while collecting and communicating vital information. Those making the buying decisions for Programmable Controller applications can have very different needs. Cost, power, speed, and communication are a few of the many considerations when choosing the right PLC for the job.

Learn More… Watch The Video!

What Is a PLC?



AUTOMATIONDIRECT

PLC stands for:

Programmable
Logic
Controller

CLICK PLC
Module

Definition:

A PLC can be defined as an industrialized computer that includes the hardware and software required to automatically control production equipment, machinery, processes, and other types of mechanical, electrical, and electronic devices typically found in industry. It also includes the flexibility to reprogram its decision making behavior through the software as often as needed. The PLC is the key element behind today's industrial automation.

learn. automationdirect.com

0:57 / 8:18

www.N2ADC.com/what-is

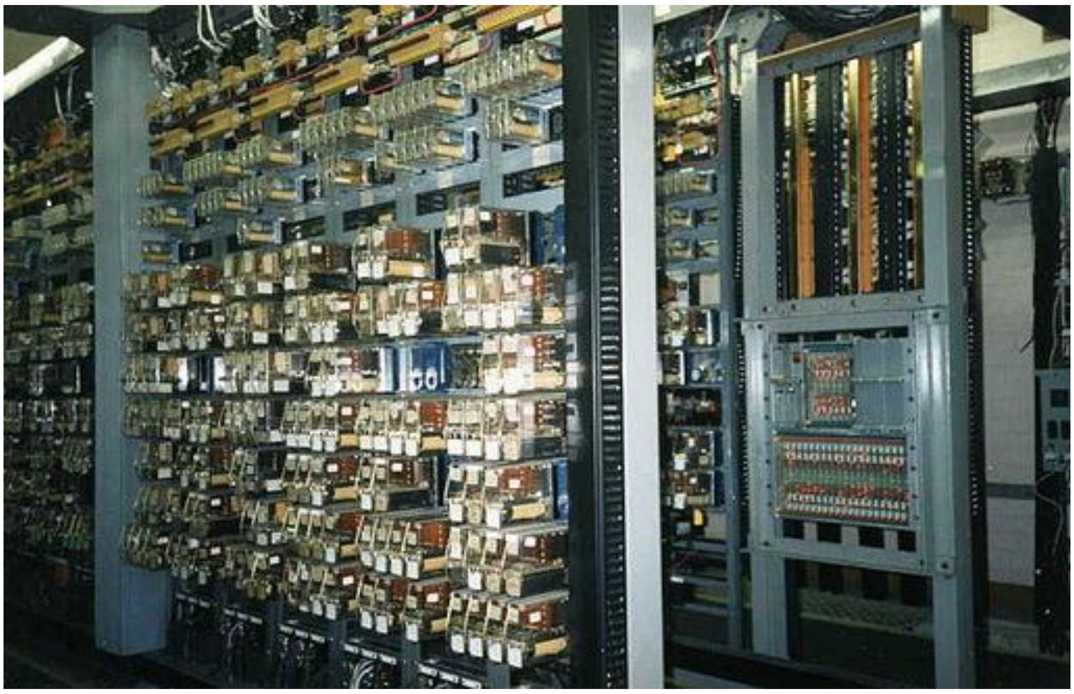
Chapter 2

History of the PLC

The PLC or Programmable Logic Controller has revolutionized the automation industry. Today PLCs can be found in everything from factory equipment to vending machines, but prior to New Year's Day 1968 the programmable controller didn't even exist. Instead what existed was a unique set of challenges that needed a solution. In order to understand the history of the PLC we must first take some time to understand the problems that existed before programmable controllers.

Before the Programmable Controller

Before the days of the PLC the only way to control machinery was through the use of relays. Relays work by utilizing a coil that, when energized, creates a magnetic force to effectively pull a switch to the ON or OFF position. When the relay is de-energized, the switch releases and returns the device to its standard ON or OFF position. So, for example, if I wanted to control whether a motor was ON or OFF, I could attach a relay between the power source and the motor. Then I can control whether the motor is getting power by either energizing or de-energizing the relay. Without power, of course, the motor would not run, thus I am controlling the motor. This type of relay is known as a power relay. There could be several motors in one factory that need to be controlled, so what do you do? You add lots of power relays. So factories started to amass electrical cabinets full of power relays. But wait, what switches the coil in the power relays ON and OFF before the power relay turns the motor ON, and what if I want to control that? What do you do? More relays. These relays are known as control relays because they control the relays that control the switch that turns the motor ON and OFF. I could keep going, but I think you get the picture of how machines were controlled pre-PLC, and, more importantly, I think you start to see some of the problems with this system of electromechanical control via relays.



Courtesy of Signalhead via Wikimedia Commons

The Problem with Relays

Think about modern factories, and how many motors and ON/OFF power switches you would need to control just one machine. Then add on all the control relays you need and what you get is… Yes, machine control, but you also get a logistical nightmare. All these relays had to be hardwired in a very specific order for the machine to work properly, and heaven forbid if one relay would have an issue, the system as a whole would not work. Troubleshooting would take hours, and because coils would fail and contacts would wear out, there was need for lots of troubleshooting. These machines had to follow a strict maintenance schedule and they took up a lot of space. Then what if you wanted to change something? You would basically have to redo the entire system. It soon became clear that there were problems installing and maintaining these large relay control systems.

Let's hear from a controls designer in the thick of things in the early '70s -

"Upon graduating from technical college in 1970, I began working as a controls designer, automating metal working machinery and equipment with industrial relays, pneumatic plunger timers, and electro-mechanical counters. Also included were fuses, control transformers, motor starters, overload relays, pushbuttons, selector switches, limit switches, rotary drum sequencers, pilot lights, solenoid valves, etc.

The relay based control systems I created included anywhere from 50 to well over 100 relays. The electrical enclosures to house the controls would typically be six feet wide by four feet high, mounted near the machinery. Picture lots of wires bundled and laced together, connecting the relays, timers, counters, terminals, and other components, all nice and tidy. Then picture after a few months or years the same wiring, after many engineering changes and troubleshooting, being out of the wire duct or unlaced; in many cases wires were added in a crisscross point to point pattern to take the shortest route and amount of time to make the change. We referred to the condition of these control enclosures as a rat's nest; reliability suffered, along with an increase in difficulty during troubleshooting, or making additional operational engineering changes."

Tom, Controls Designer

Birth of the PLC Solution

So what was the solution? I am sure this is the exact question that engineers at the Hydra-Matic division of General Motors were struggling with every day. Fortunately, at that time, the concept of computer control had started to make its way into conversations at large corporations such as GM. According to Dick Morley, the undisputed father of the PLC, "The programmable controller was detailed on New Year's Day, 1968."

The popular forum PLCDEV.com outlines a list of requirements that GM engineers put out for a "standard machine controller." It is this request that Dick Morley and his company, Bedford and Associates, were responding to when the first PLC was envisioned. Besides replacing the relay system, the requirements listed by GM for this controller included:

1. A solid-state system that was flexible like a computer but priced competitively with a like kind relay logic system.
2. Easily maintained and programmed in line with the already accepted relay ladder logic way of doing things.
3. It had to work in an industrial environment with all its dirt, moisture, electromagnetism and vibration.
4. It had to be modular in form to allow for easy exchange of components and expandability.

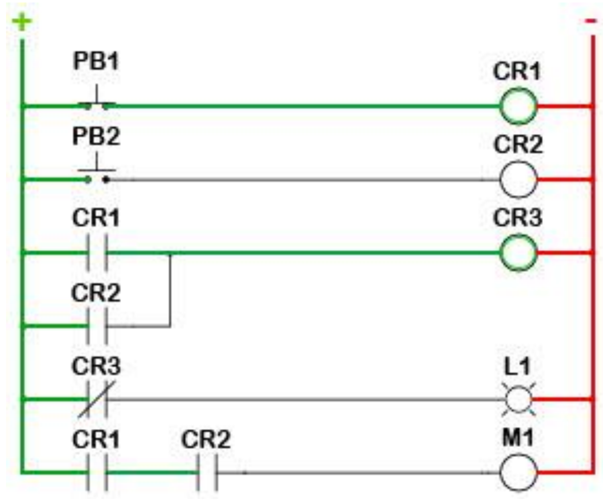


Image of Dick Morley
Courtesy of AutomationWorld.com

The programming look of the PLC required that it be easily understood and used by maintenance electricians and plant engineers. As relay-based control systems evolved and became more complicated, the use of physical component location wiring diagrams also evolved into the relay logic being shown in a ladder fashion. The control power hot wire would be the left rail, with the control power neutral as the right rail. The various relay contacts, pushbuttons, selector switches, limit switches, relay coils, motor starter coils, solenoid valves, etc., shown in their logical order would form the ladder's rungs. It was requested that the PLC be programmed in this Ladder Logic fashion.

As Dick Morley laments in his memoirs , the process from idea to actual controller wasn't all smooth sailing.

"The initial machine, which was never delivered, only had 125 words of memory, and speed was not a criteria as mentioned earlier. You can imagine what happened! First, we immediately ran out of memory, and second, the machine was much too slow to perform any function anywhere near the relay response time. Relay response times exist on the order of 1/60th of a second, and the topology formed by many cabinets full of relays transformed to code is significantly more than 125 words. We expanded the memory to 1K and thence to 4K. At 4K, it stood the test of time for quite a while."

Tom, our controls designer, recounts, "*My experience in creating relay-based control systems, at that time, put me in the perfect position to be one of the first control system designers to use some of the very first programmable controllers to replace relay-based control systems. My first experience with a PLC happened to be with one of Bedford Associates competitor's solid state devices. The unit was programmed with a suitcase-sized programming device that required setting the instruction type and line address and then pressing a button to burn a fuse link open in a memory chip to set the logic path. Once the programming was completed and tested, the PLC was able to perform the machine cycle operation in a very reliable manner. Unfortunately the PLC card rack was open in the rear with a mixture of 24 VDC and 120 VAC power and signals. It didn't take much for an electrician checking signals during troubleshooting to accidently short the 120 VAC to the 24 VDC and take out the entire PLC system. Being the first use of a PLC in a large corporation, the failure doomed the use of PLCs at this manufacturing facility for a couple of years."*

Eventually Dick Morely spun off a new company named Modicon and started to sell those first PLCs, the Modicon 084 (named because it was prototype #84). It was the Modicon 084 that was presented to GM to meet its criteria for its "standard machine controller." Modicon started to sell the 084 with very limited success. As Dick Morley puts it, "Our sales in the first four years were abysmal." But nevertheless the company continued to learn and develop. Eventually, Modicon would bring to life the controller that would change the industry forever, the Modicon 184. Dick Morley writes this about the 184:

"The thing that made the Modicon Company and the programmable controller really take off was not the 084, but the 184. The 184 was done in design cycle by Michael Greenberg, one of the best engineers I have ever met. He, and Lee Rousseau, president and marketer, came up with a specification and a design that revolutionized the automation business. They built the 184 over the objections of yours truly. I was a purist and felt that all those bells and whistles and stuff weren't "pure", and somehow they were contaminating my "glorious design", Dead wrong again, Morley! They were specifically right on! The 184 was a walloping success, and it—not the 084, not the invention of the programmable controller—but a product designed to meet the needs of the marketplace and the customer, called the 184, took off and made Modicon and the programmable controller the company and industry it is today."

Image Courtesy of RepairZone.com

The PLC in its teenage years

The first PLCs had the ability to work with input and output signals, relay coil/contact internal logic, timers and counters. Timers and counters made use of word size internal registers, so it wasn't too long before simple four-function math became available. The PLC continued to evolve with the addition of one-shots, analog input and output signals, enhanced timers and counters, floating point math, drum sequencers and mathematic functions. Having built-in PID (Proportional-Integral-Derivative) functionality was a huge advantage for PLCs being used in the process industry. Common sets of instructions evolved into fill-in-the-blank data boxes that have made programming more efficient. The ability to use meaningful Tag Names in place of non-descriptive labels has allowed the end user to more clearly define their application, and the ability to import/export the Tag Names to other devices eliminates errors that result when entering information into each device by hand.

As the functionality of the PLC evolved, programming devices and communications also saw rapid growth. The first programming devices were dedicated, but unfortunately the size of suitcases. Later, handheld programming devices came into the picture, but soon were replaced with proprietary programming software running on a personal computer. AutomationDirect's DirectSOFT, developed by Host Engineering, was the first Windows-based PLC programming software package. Having a PC communicating with a PLC provided the ability to not only program, but also allowed easier testing and troubleshooting. Communications started with the MODBUS protocol using RS-232 serial communications. The addition of various automation protocols communicating over RS-485, DeviceNet, Profibus, and other serial communication architectures have followed. The use of serial communications and the various PLC protocols also allowed PLCs to be networked with other PLCs, motor drives, and human to machine interfaces (HMI). Most recently Ethernet and protocols such as EtherNet/IP (for Industrial Protocol) have gained tremendous popularity.

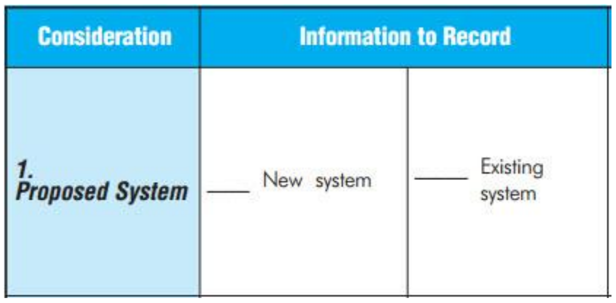
Sources
http://www.barn.org/FILES/historyofplc.html
http://www.machine-information-systems.com/PLC_History.html
http://www.plcs.net/chapters/history2.htm
http://www.controldesign.com/articles/2005/264/
http://www.plcdev.com/plc_timeline, http://www.plcdev.com/the_birth_of_the_plc

Chapter 3

How to Choose a Controller

Choosing the most effective controller for your application depends on a number of factors. The worksheet below serves as a checklist of things to consider when determining programmable controller requirements. It lists the most important areas to consider when choosing a system, as well as provides space for recording determinations of your system needs. To print a copy, here is the PDF version of the Worksheet for Choosing a Controller.

Step 1:

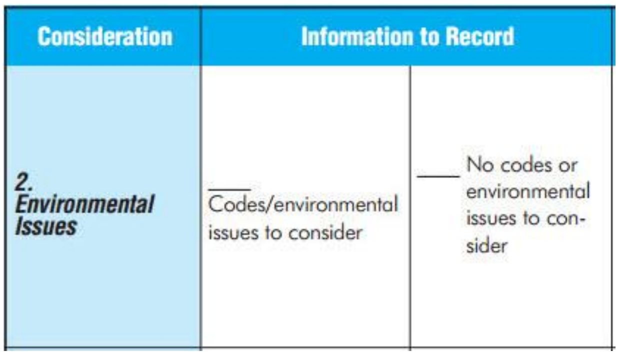


Consideration	Information to Record	
1. Proposed System	___ New system	___ Existing system

Determine whether your system is new or existing: Will your system be installed from scratch or are there existing products already installed? The rest of your system will need to be compatible with new components.

Why this is important: Certain controller products may not be compatible with others. Making sure your existing products are compatible with any new products you are researching will save you time and money. Check appropriate entry.

Step 2:



Consideration	Information to Record	
2. Environmental Issues	___ Codes/environmental issues to consider	___ No codes or environmental issues to consider

Consider any environmental issues that will affect your application (temperature, dust, vibration, codes specific to your facility, etc.).

Why this is important: Certain environments may affect the operation of a controller. For example, typical controllers have an operating temperature of 0-55 degrees Celsius (32-130 degrees F). If your application will include any extreme environmental conditions, or you have specific codes at your facility that must be met, you will need to either research products that meet those specifications or design the installation to meet requirements. Check appropriate entry.

Step 3:

Consideration	Information to Record	
3. Discrete Devices	_____ Total inputs: _____ AC _____ DC	_____ Total outputs: _____ AC _____ DC

Determine how many discrete devices your system will have. Which types (AC, DC, etc.) are needed?

Why this is important: The number and type of devices your system will include is directly linked to the amount of I/O that will be necessary for your system. You will need to choose a controller that supports your I/O count requirements and has modules that support your signal types. Enter quantities and type based on corresponding field devices.

Step 4:

Consideration	Information to Record	
4. Analog Devices	_____ Total inputs: ____ Voltage ____ Current ____ Thermo ____ RTD	_____ Total outputs: ____ Voltage ____ Current

Determine how many analog devices your system will have. Which types (voltage, current, temperature, etc.) are needed?

Why this is important: The number and type of devices your system will include is directly linked to the amount of I/O that will be necessary for your system. You will need to choose a controller that supports your I/O count requirements and has modules that support your signal types. Enter quantities and type based on corresponding field devices.

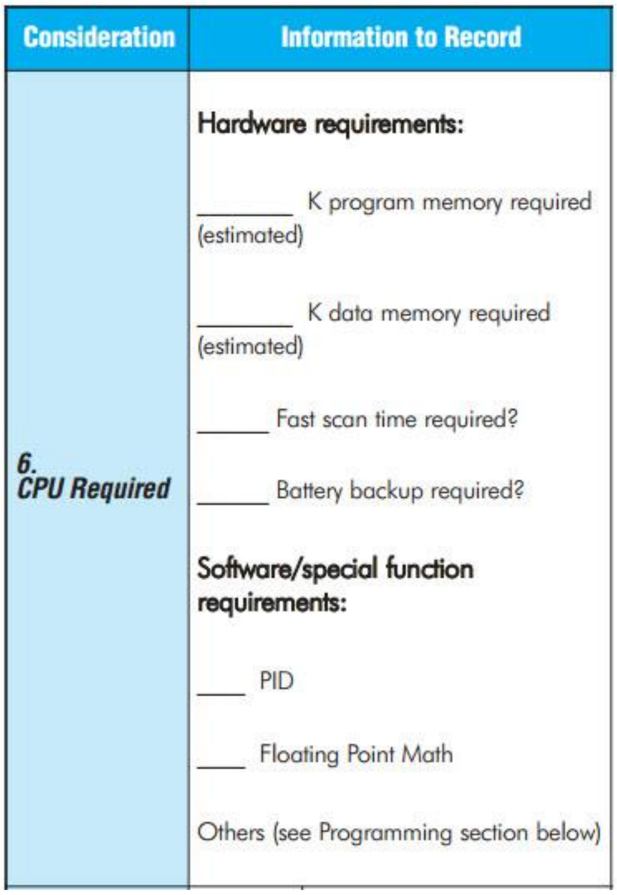
Step 5:

Consideration	Information to Record
5. Specialty Modules or Features (application-specific)	_____ High speed counter _____ Positioning _____ Servo/stepper _____ BASIC programming _____ Real-time clock _____ Others (list)

Determine whether your system will require any specialty features: Will your application require high-speed counting or positioning? What about a real-time clock or other specialty feature?

Why this is important: Specialty functions are not necessarily available in a controller CPU or in standard I/O modules. Understanding the special functions your system may perform will help you determine whether or not you will need to purchase additional specialty modules. Check all features required.

Step 6:



Consideration	Information to Record
6. CPU Required	**Hardware requirements:** ________ K program memory required (estimated) ________ K data memory required (estimated) ______ Fast scan time required? ______ Battery backup required? **Software/special function requirements:** ____ PID ____ Floating Point Math Others (see Programming section below)

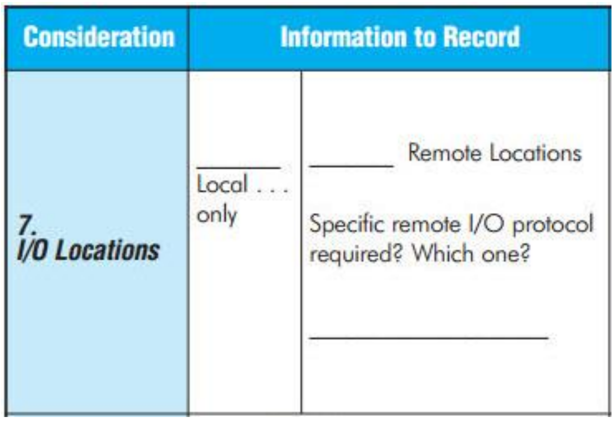
Determine the type of CPU you will need: How much memory will your system require? How many devices will your system have (determines data memory)? How large is your program, and what types of instructions will your program include (determines program memory)? How fast a scan time do you need?

Why this is important: Data memory refers to the amount of memory needed for dynamic data manipulation and storage in the system. For example, counter and timer instructions typically use data memory to store setpoints, current values, and other internal flags. If the application requires historical data retention, such as measured device values over a long period of time, the size of the data tables required may determine the CPU model you choose. Program memory is the amount of memory needed to store the sequence of program instructions that have been selected to perform the application. Each type of instruction requires a specific amount of program memory, typically defined in a programming manual. Applications that are basically sequential in nature can rely on the I/O device rule of thumb to estimate program memory (five words of memory for each I/O device); complex applications will be more difficult to judge.

If scan time is important in your application, consider the CPU processor speed as well as instruction execution speed. Some CPUs are faster at boolean logic but slower with data handling instructions.

If special functions such as PID are required, the CPU you select may make those functions easier to perform. For program memory required, follow this rule of thumb: 5 words of program memory for each discrete device and 25 words for each analog device. Check or calculate all requirements that apply.

Step 7:



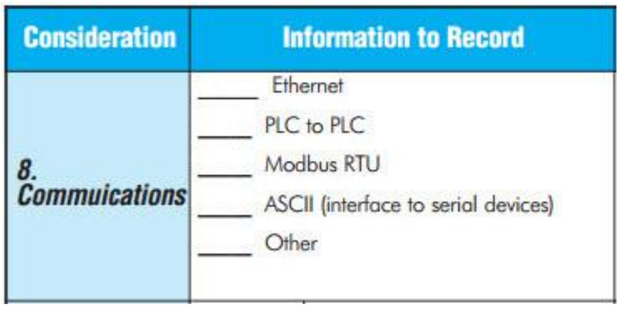
Consideration	Information to Record	
7. I/O Locations	____ Local . . . only	____ Remote Locations Specific remote I/O protocol required? Which one? ____

Determine where your I/O will be located: Will your system require only local I/O, or both local and remote I/O locations?

Why this is important: If subsystems will be needed at long distances from the CPU, you will need a controller that supports remote I/O. You will also have to determine if the remote distances and speeds supported will be adequate for your application. Serial and Ethernet-based I/O hardware are two typical choices available for most systems. This I/O may also be referred to as distributed I/O, and may require a particular protocol, such as Modbus.

Enter number of physical locations needed, and if/what specific protocol may be required.

Step 8:

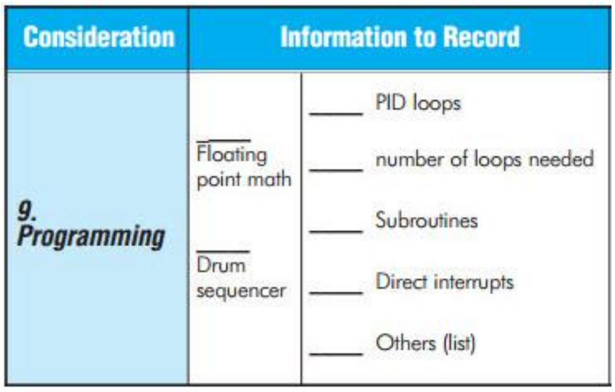


Consideration	Information to Record
8. Commuications	____ Ethernet ____ PLC to PLC ____ Modbus RTU ____ ASCII (interface to serial devices) ____ Other

Determine your communication requirements: Will your system be communicating to other networks, systems or field devices?

Why this is important: Communication ports (other than the programming port) are not always included with a controller. Knowing your system communication requirements will help you choose a CPU that supports your communication requirements, or additional communication modules if necessary. Check any/all communications functions required.

Step 9:



Consideration	Information to Record	
9. Programming	____ Floating point math ____ Drum sequencer	____ PID loops ____ number of loops needed ____ Subroutines ____ Direct interrupts ____ Others (list)

Determine your programming requirements: Does your application require only traditional programming instructions, or are special instructions necessary?

Why this is important: Certain controllers may not support every type of instruction. You will need to choose a model that supports all instructions that you may need for a specific application. For example, built-in PID functions are much easier to use than writing your own code to perform closed-loop process control. Typical instructions such as timers, counters, etc., are available in most controllers; note any other special instructions required here. Check any/all programming functions required.

Chapter 4

PLC Hardware

What controls you? No, I'm not talking about your boss, your spouse or the government. Instead, think of your body as an industrial control system. Now, what controls you? Hopefully, you said your brain, although the heart is an acceptable answer sometimes. But for now, let's focus on the brain and the nervous system as well. Your body receives signals from your senses or nerves that travel through the spinal cord to the brain. Well, a PLC system operates the same way, with a brain (the CPU), a spine (the base or backplane), and the senses (I/O modules). Now, let's take a closer look at each and discuss functions, options and considerations.

The Base-ics

We'll start with the spine. In a PLC system, you have different options when it comes to connecting I/O modules to the CPU. Some systems have a fixed style where the CPU comes with a fixed set of I/O points already installed and dedicated connections already determined.

Other PLC systems have a stackable style where the CPU and I/O modules are separate, but come with connectors built in that are used to attach the components together. These connections form one continuous data bus throughout the system. This internal data bus is often referred to as a backplane.

Another option with PLC hardware is the modular base configuration. With this style of PLC, a separate base unit that holds the modular components is needed. Each module will seat into a slot on the base unit that is used to connect the module to the backplane already installed in the base. These bases come with different numbers of slots and some with power supplies built in. Typically, the first two leftmost slots are dedicated for the power supply and CPU.

One question to keep in mind with internal PLC connections is, “How easy are they to change?” In other words, after all of the modules are installed, how easy is it to get them out? With the fixed style, you are just that, fixed, so no help there. Stackable PLC components can be changed, but you have to break the backplane connection for all the modules downstream of the module you wish to remove. That could be an issue. The modular base configuration is by far the easiest to change. Simply slide the module out and slide the new one in. Some of these PLCs even offer hot swapping, which means that the module change can be done with the PLC powered and with no interruption to the control process.

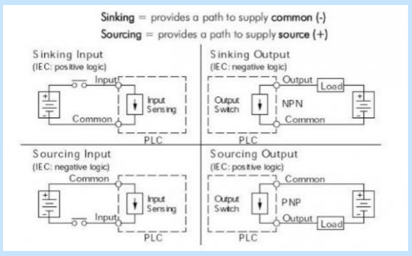
The Ins and Outs

Now, let’s look at the senses, the information providers, or, in the PLC’s case, the I/O modules. The I/O modules, and their respective end devices, allow the PLC to know and affect the current state of the process being controlled. There are many types of input and output modules available, but they can all be classified as analog, discrete or specialty I/O.

Discrete I/O is the simplest of the bunch and provides the PLC with ON/OFF control. Used with both AC and DC voltage ranges, they provide the CPU with a yes/no, true/false indication and allow simple full ON or full OFF responses. On the input side, you would use discrete I/O for simple questions like, “Is the box there?”, “Is the tank full?”, “Can I start this motor?” These input signals are provided by devices such as photoeyes, proximity switches, E-stop pushbuttons, float switches, etc. For discrete outputs, your command choices are either ON or OFF with nothing in between and are commonly used for stack lights, alarms, relays, solenoids, etc.

One thing to consider with discrete I/O is whether you need a sink, source or relay configuration. With sinking inputs/outputs, the PLC will provide the reference voltage (typically 0V) when completing the circuit. Sourcing inputs/outputs are the opposite and the PLC will provide the source voltage, be it 12VDC, 24VDC, 240VAC, etc. Relay types don’t provide either. They function just as a relay contact would, using an external source and connecting it to a load once activated. You can read more about sink/source concepts in this blog.

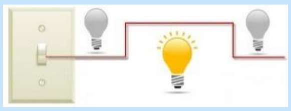
Sinking and Sourcing Concepts



Make the right choice the first time when selecting the type of I/O points for your application by reviewing these sinking and sourcing concepts.

Visit: http://N2ADC.com/jhll2

Understanding Discrete and Analog I/O

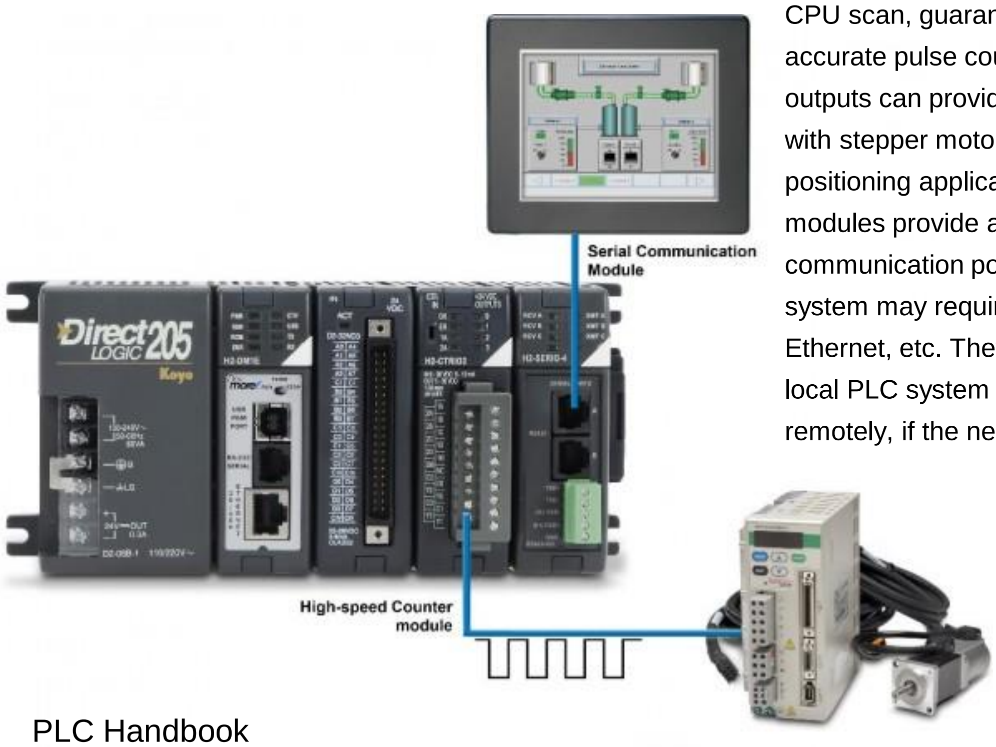


Shed some light on the differences between discrete and analog I/O for your PLC applications

Visit: N2ADC.com/ngipf

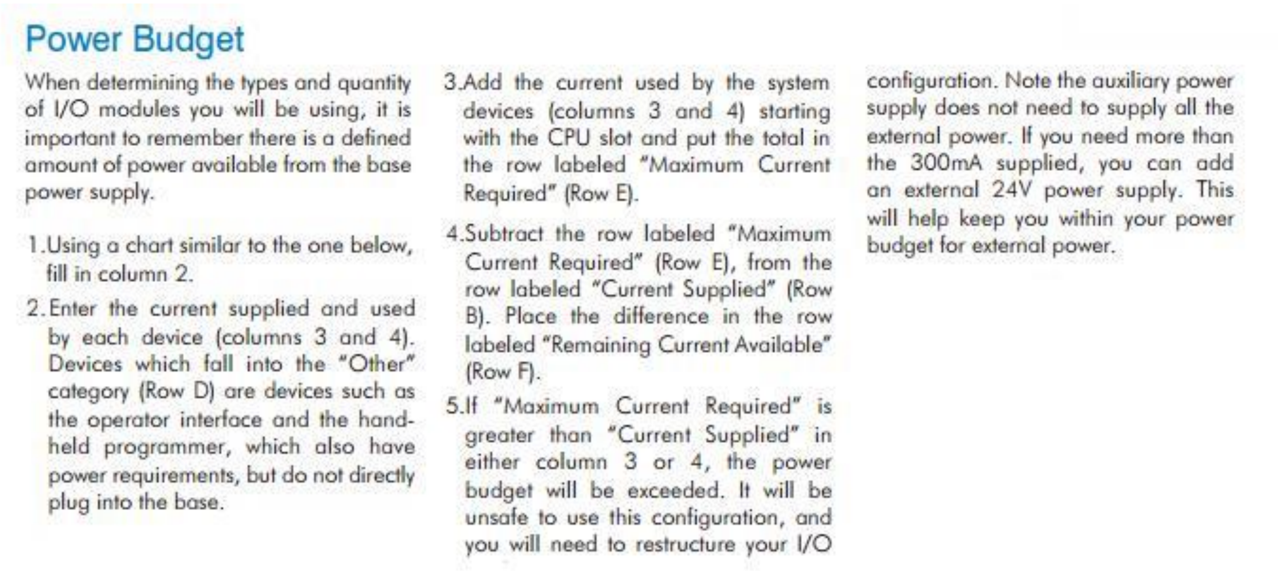
Next up is analog. Analog I/O deals with the gray area between full ON and full OFF that discrete I/O ignores. It provides the PLC with the data it needs for precision control of a process. If you want to know the exact liquid level in a tank or you want to open a valve 1/3 of the way, analog is the answer. Analog signals come in a variety of ranges including: 0-20mA, 4-20mA, 0-10V, etc. RTD and thermocouple modules are two analog modules that specialize in converting low voltage signals from temperature probes into usable data. One important factor to remember with analog modules is the resolution they provide. The higher the resolution, the higher the accuracy of the input measurement or output response. To read more about discrete and analog I/O, refer to this blog.

That leaves us with specialty I/O. This type of I/O includes special functions like high-speed and communication. High-speed modules are needed when the input/output data is comprised of high-frequency pulses. These modules can track input data, such as encoder signals, independent of the CPU scan, guaranteeing a more accurate pulse count. And high-speed outputs can provide precision control with stepper motors used in motion or positioning applications. Communication modules provide additional communication ports/protocols that a system may require: RS232, RS485, Ethernet, etc. They can also allow a local PLC system to be expanded remotely, if the need arises.



PLC Power

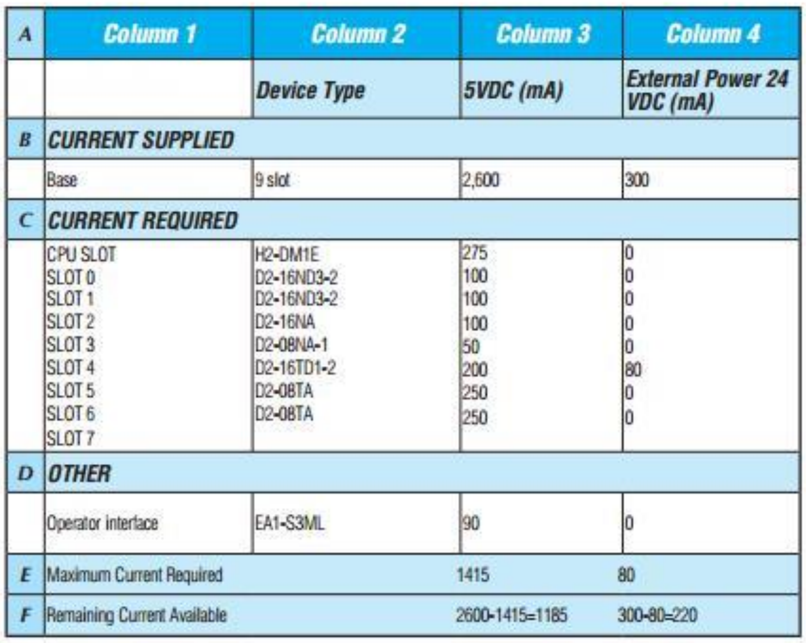
Now before we touch on the brain of the system, I'd like to detour a bit and discuss the Power Supply. As mentioned earlier, the PLC power supply can be included with the base or it can be a separate unit. They come in a variety of voltage ranges, including 12-24VDC and 110/220 VAC, and supply a limited amount of amperage. It is very important that the power supply chosen is capable of supplying the CPU and I/O modules with the power they need. To ensure this, perform a power budget analysis and calculate the required current for your application. An example is shown below.



Power Budget

When determining the types and quantity of I/O modules you will be using, it is important to remember there is a defined amount of power available from the base power supply.

1. Using a chart similar to the one below, fill in column 2.
2. Enter the current supplied and used by each device (columns 3 and 4). Devices which fall into the "Other" category (Row D) are devices such as the operator interface and the hand-held programmer, which also have power requirements, but do not directly plug into the base.
3. Add the current used by the system devices (columns 3 and 4) starting with the CPU slot and put the total in the row labeled "Maximum Current Required" (Row E).
4. Subtract the row labeled "Maximum Current Required" (Row E), from the row labeled "Current Supplied" (Row B). Place the difference in the row labeled "Remaining Current Available" (Row F).
5. If "Maximum Current Required" is greater than "Current Supplied" in either column 3 or 4, the power budget will be exceeded. It will be unsafe to use this configuration, and you will need to restructure your I/O configuration. Note the auxiliary power supply does not need to supply all the external power. If you need more than the 300mA supplied, you can add an external 24V power supply. This will help keep you within your power budget for external power.



A	Column 1	Column 2	Column 3	Column 4
		Device Type	5VDC (mA)	External Power 24 VDC (mA)
B	CURRENT SUPPLIED			
	Base	9 slot	2,600	300
C	CURRENT REQUIRED			
	CPU SLOT SLOT 0 SLOT 1 SLOT 2 SLOT 3 SLOT 4 SLOT 5 SLOT 6 SLOT 7	H2-DM1E D2-16ND3-2 D2-16ND3-2 D2-16NA D2-08NA-1 D2-16TD1-2 D2-08TA D2-08TA	275 100 100 100 50 200 250 250	0 0 0 0 0 80 0 0
D	OTHER			
	Operator interface	EA1-S3ML	90	0
E	Maximum Current Required		1415	80
F	Remaining Current Available		2600-1415=1185	300-80=220

The CPU

So we have the connections made and the external data, now we need to know what to do with it. That is the job of the CPU (central processing unit). The CPU contains a microprocessor, memory storage and other integrated circuits that are used to execute the control program, store logic data, and communicate to external devices. So what should the CPU hardware include? Well, for starters, the CPU will need some way to communicate and this is typically done using Ethernet, serial or USB communication ports. Serial ports are important since many existing networks use these standards but Ethernet has become the favored method to communicate in today's industrial applications. USB is a recent addition and is extremely useful when connecting to a CPU to program or monitor its logic. Just as important as the ports available are the protocols the CPU can support: EtherNet/IP, Modbus TCP, etc… Some PLCs use proprietary communication and others use open standards. Either way, make sure the CPU you select has the communication capabilities you need. For a detailed look into PLC communication protocols, see the topic titled "PLC Communications - Coming of Age".

Another important aspect of CPU hardware is the memory size. Your CPU needs enough storage space to handle the amount of tasks you are going to assign and it doesn't hurt to have a little extra for future needs. CPU memory has expanded with the times and some CPUs today can have 50MB or more of memory available to the user. The large memory capacity in these CPUs allows for almost limitless programming, ample space for program documentation and impressively quick scan times. Removable memory card support is another feature that is popular today. Adding an additional 1 to 32GB of memory storage, these memory cards allow for extensive data logging and easy program downloads, without a PC. Other hardware related options available for CPUs include a battery backup, built-in I/O and status indicators/OLED message displays.

One Last Thing…

Now that we have discussed the individual hardware components of a PLC, there are a few other hardware considerations I'd like to mention for the PLC as a whole.

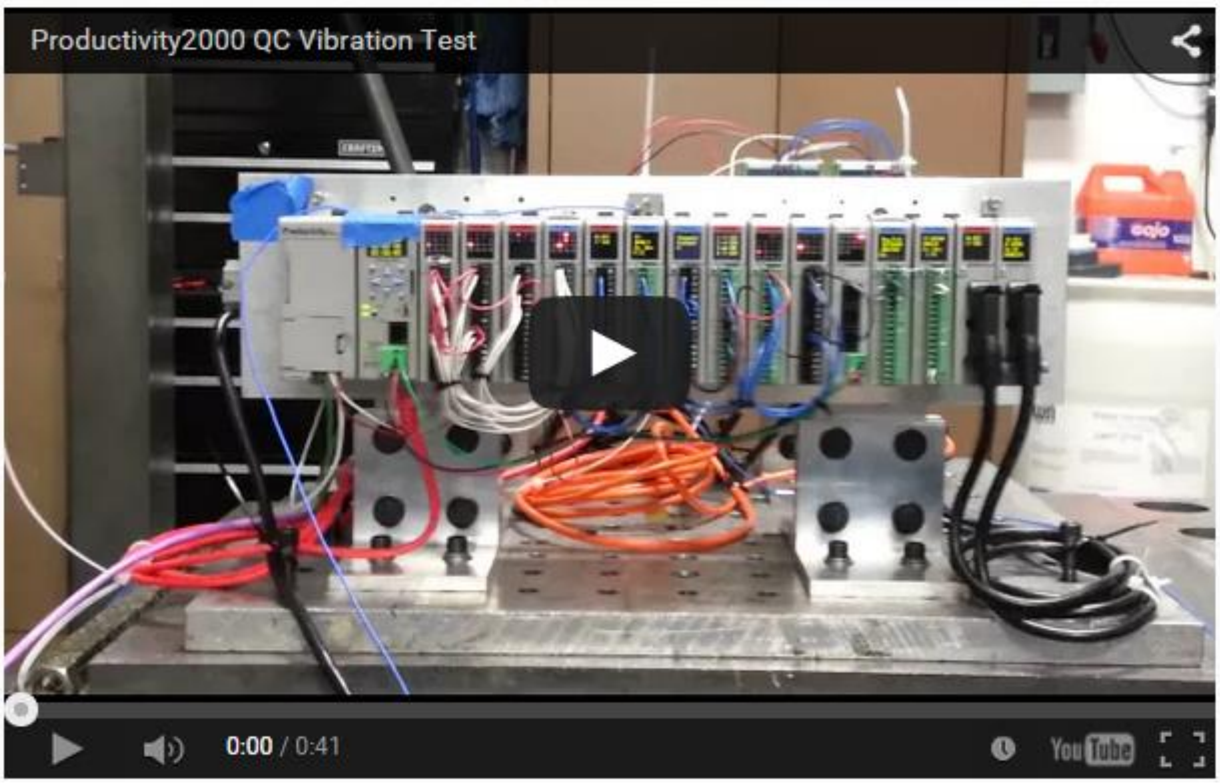
- Real Estate – size can be an issue with PLC installations and if it is a concern, look into compact or slim form factor designs. Also look for high I/O density. Check the maximum number of points per module and modules per base. More I/O points per module equals less modules and therefore more space.

- Wiring Options – screw type and spring clamp connectors are most common with I/O modules. Wiring connection systems like the ZipLinks wiring solution may also be available.

- Mounting Options – See what options are available and which you require: DIN rail, panel mount or both.

- Environmental Certifications – make sure the hardware you are using is certified for the environment it will reside in. For instance, hazardous locations and may require extra precautions.

- Quality Control Certifications – see what kind of testing was done to verify the hardware's durability. Extreme temperature, electric shock and vibration testing are a few examples. By the way, if you have ever wondered what vibration testing was all about, wonder no more. See the Productivity2000's vibration test in action below or click here:



Chapter 5

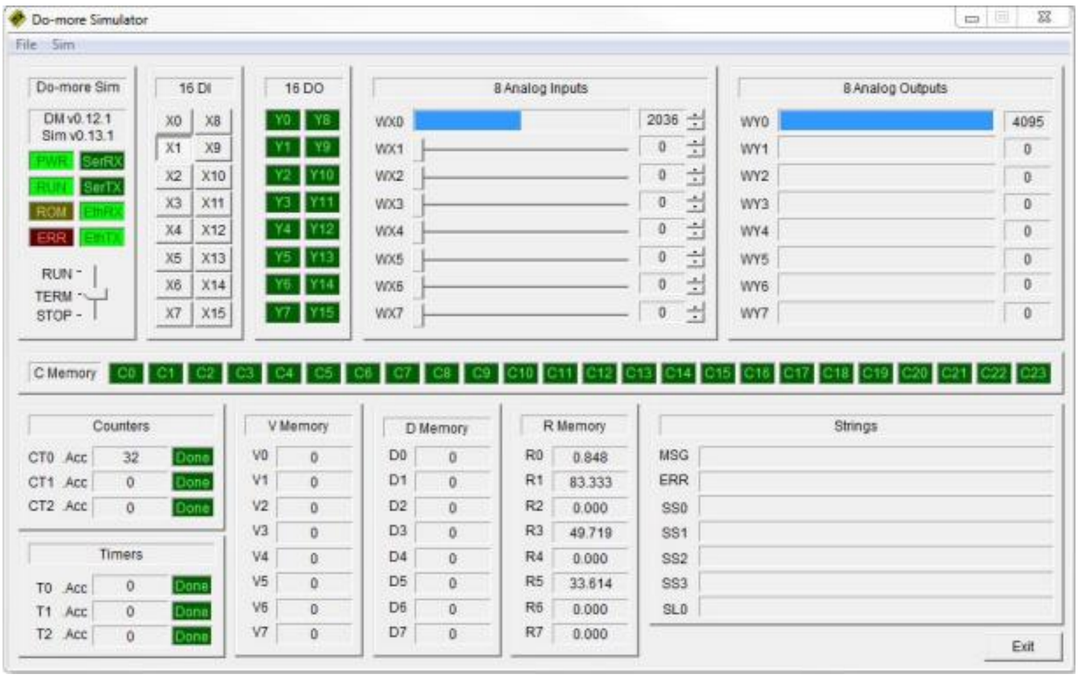
PLC Software

One of the biggest factors in any PLC buying decision is the capability of the programming software. How easy is it to use? What does it offer? PLC hardware is only half of the equation, and who wants to buy a controller only to find out the software is lacking the basics? Been there, done that! So what should you expect or want in a PLC programming software?

Since each user has his/her own needs and preferences when programming a PLC, that's a hard question to answer. However, here are some thoughts on features and tools that we believe are helpful and that you'll want to consider in your next PLC software package.

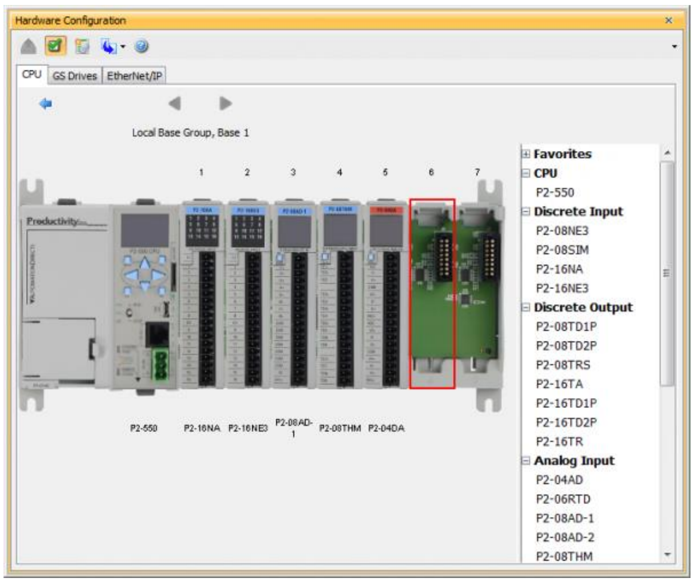
Built-in Simulator

A built-in simulator is extremely useful in PLC project development. The simulator provides a virtual PLC that you can use to test your logic without downloading to an actual PLC. A good simulator will emulate analog and discrete I/O and provide access to logic elements such as timers, counters, control bits, etc.


Do-more Simulator
File Sim
Do-more Sim
DM v0.12.1
Sim v0.13.1
PWR SerRX RUN SerTX ROM EthRX ERR EthTX
RUN TERM STOP
16 DI
X0 X8 X1 X9 X2 X10 X3 X11 X4 X12 X5 X13 X6 X14 X7 X15
16 DO
Y0 Y8 Y1 Y9 Y2 Y10 Y3 Y11 Y4 Y12 Y5 Y13 Y6 Y14 Y7 Y15
8 Analog Inputs
WX0 2036
WX1 0
WX2 0
WX3 0
WX4 0
WX5 0
WX6 0
WX7 0
8 Analog Outputs
WY0 4095
WY1 0
WY2 0
WY3 0
WY4 0
WY5 0
WY6 0
WY7 0
C Memory
C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 C13 C14 C15 C16 C17 C18 C19 C20 C21 C22 C23
Counters
CT0 Acc 32 Done
CT1 Acc 0 Done
CT2 Acc 0 Done
Timers
T0 Acc 0 Done
T1 Acc 0 Done
T2 Acc 0 Done
V Memory
V0 0 V1 0 V2 0 V3 0 V4 0 V5 0 V6 0 V7 0
D Memory
D0 0 D1 0 D2 0 D3 0 D4 0 D5 0 D6 0 D7 0
R Memory
R0 0.848 R1 83.333 R2 0.000 R3 49.719 R4 0.000 R5 33.614 R6 0.000 R7 0.000
Strings
MSG ERR SS0 SS1 SS2 SS3 SL0
Exit

Hot Swapping and Run Time Transfers

Once you have your project up and running, system modifications can become much harder. Imagine having to schedule a shutdown at your plant in order to replace an I/O module or change a rung of code. If your software allows for hot swapping and run time edits, then don't worry. Hot swapping means replacing system modules while the system is HOT or powered, and run time transfers allow the user to transfer project edits into the CPU without stopping the CPU scan. These two software features are important because they prevent costly shutdowns and production losses whenever hardware and/or software changes are needed.

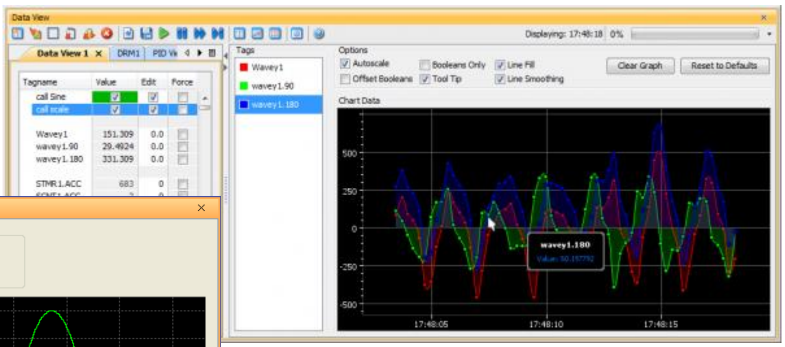


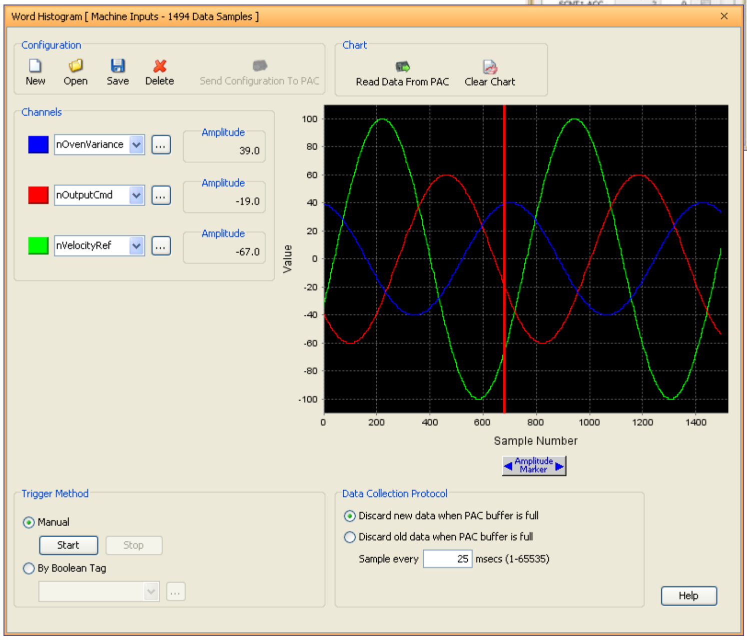
Auto Discovery

A huge timesaver! When software offers automatic hardware detection, everybody wins! With this feature, in order to configure the hardware setup of your PLC, all you do is install each I/O module in the base and power it up. The software will automatically discover the installed modules and assign physical I/O tags based on each module's position in the base. From there you are ready to program with the auto-configured settings or you can reconfigure the setup and assign new tags manually. Some software can also auto detect other connected devices, such as VFDs, to make your job even easier.

Data View and Histograms

Data View windows allow users to monitor and manipulate PLC logic values in real time. Although most PLC software will have a way to do this, there are some advanced capabilities, graphical trend charts for instance, that you may want to keep an eye out for. Also, bit and word histograms allow you to sample and plot logic values over time. These can be set up to sample data as often as you choose and can be very helpful when troubleshooting.





Security

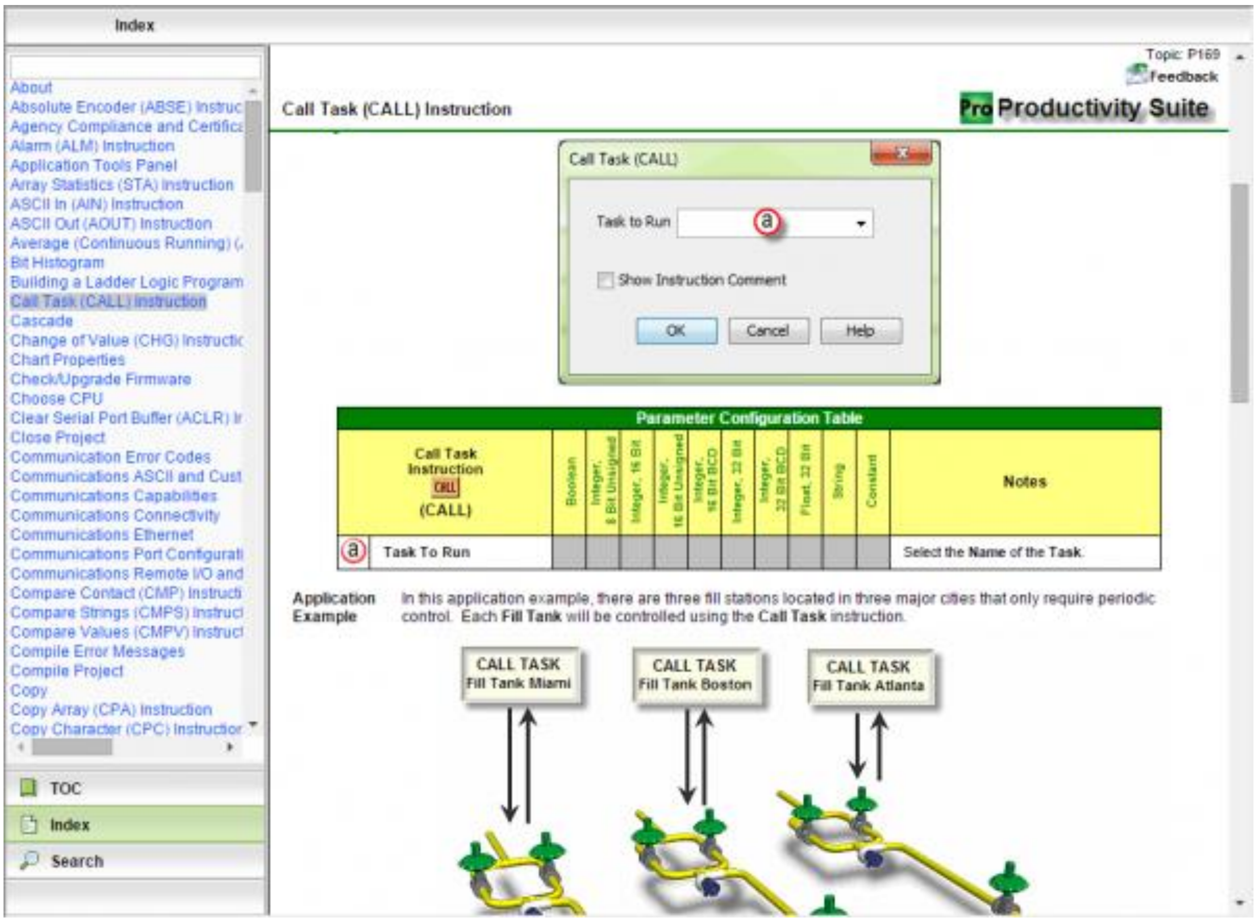
Software user accounts allow you to control who has access to your PLC and what they are allowed to do with that access. System security is an important feature that good PLC software should have.

Search and Cross Reference

Three PLC software functions you will use often are the Search, Search and Replace, and Cross Reference. These functions will tell you if an address or variable is being used, where it is located, and allow you to easily make changes, if needed. One thing to consider is how well the PLC software performs these functions and if it offers the usability you need. See what kind of searches are allowed (tag, address, comment, instruction or partial) and what filtering options are available.

Help Files

Good PLC programming software should have good help files. Embedded help files are a great asset when programming in a new environment or when troubleshooting. Help files should provide information in a way that is easy to comprehend with visuals, screenshots and application examples. They should also include the ability to search for items and be printable.



About
Absolute Encoder (ABSE) Instruc
Agency Compliance and Certific
Alarm (ALM) Instruction
Application Tools Panel
Array Statistics (STA) Instruction
ASCII In (AIN) Instruction
ASCII Out (AOUT) Instruction
Average (Continuous Running) (
Bit Histogram
Building a Ladder Logic Program
Call Task (CALL) Instruction
Cascade
Change of Value (CHG) Instructic
Chart Properties
Check/Upgrade Firmware
Choose CPU
Clear Serial Port Buffer (ACLR) Ir
Close Project
Communication Error Codes
Communications ASCII and Cust
Communications Capabilities
Communications Connectivity
Communications Ethernet
Communications Port Configurati
Communications Remote I/O and
Compare Contact (CMP) Instructi
Compare Strings (CMPS) Instruct
Compare Values (CMPV) Instruct
Compile Error Messages
Compile Project
Copy
Copy Array (CPA) Instruction
Copy Character (CPC) Instructior

Call Task (CALL) Instruction

Parameter Configuration Table

	Call Task Instruction (CALL)	Boolean	Integer, 8 Bit Unsigned	Integer, 16 Bit	Integer, 16 Bit Unsigned	Integer, 16 Bit BCD	Integer, 32 Bit	Integer, 32 Bit BCD	Float, 32 Bit	String	Constant	Notes
a	Task To Run											Select the **Name** of the Task.

Application Example In this application example, there are three fill stations located in three major cities that only require periodic control. Each **Fill Tank** will be controlled using the **Call Task** instruction.

Connectivity

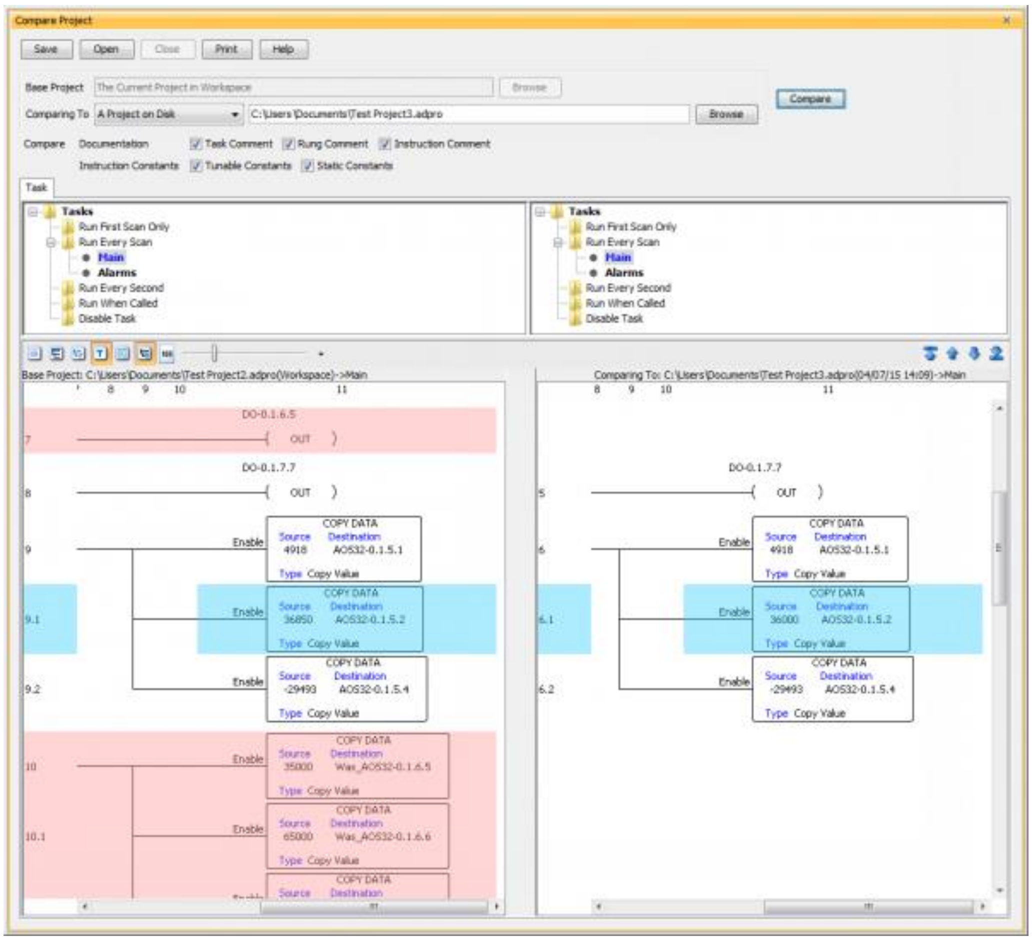
Another thing to consider is what connection options are available when downloading or working online. A lot of PLCs today offer plug-and-play USB connections for programming, which are fast and easy to use but require direct connections between the PC and CPU. With networked PLCs, Ethernet is typically the connection method (unless you are using a serial network) but you may want to verify if any additional communication software is required with these connections. That software may cause unwanted headaches if it can’t recognize connected controllers. One last option you may find helpful when it comes to downloading is USB project transfers. See if your PLC allows projects to be downloaded via an onboard USB port. This can be important if you have remote PLCs that aren’t easily accessible with a PC.

Customizable Layouts

No matter what the PLC software has to offer, none of it matters if it isn’t user friendly. Layouts that are fully customizable with dockability, display options, view options, accessibility options, and text options allow you to personalize your programming experience to fit your needs.

Project Compare

“Why is this message telling me my project is different from the one on the CPU?!” We’ve all been there. That’s what makes the project compare feature so valuable. It allows you to perform project-to-project comparisons and get detailed information on the differences. This can be very useful when going online or when trying to remember what you changed in the last revision.


Compare Project

Base Project: The Current Project in Workspace

Comparing To: A Project on Disk — C:\Users\Documents\Test Project3.adpro

Compare Documentation: [✓ Task Comment] [✓ Rung Comment] [✓ Instruction Comment]

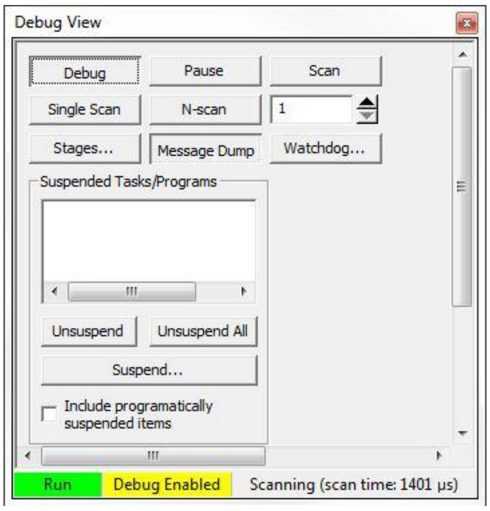
Instruction Constants: [✓ Tunable Constants] [✓ Static Constants]

Base Project: C:\Users\Documents\Test Project2.adpro(Workspace)->Main

Comparing To: C:\Users\Documents\Test Project3.adpro(04/07/15 14:09)->Main

Debugging Tools

Debugging tools assist with locating and repairing bugs in your code by allowing you to slowly step through the execution of your logic. Step from one rung to the next, execute multiple rungs, run one complete scan or pause the execution all together. Some debugging tools also allow you to suspend selected tasks or routines and force watchdog errors.

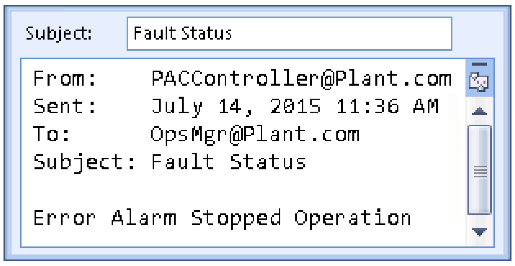


Web Server and Mobile Apps

With Web server functionality, you can access a PLC remotely using a standard Web browser and the configured IP address of your CPU. View system diagnostic data or process values offsite. Mobile apps are also very popular and you may want to see if the PLC you are interested in has mobile capabilities. This way you can monitor system operations on the go from anywhere using a Wi-Fi or cellular network connection.

Email

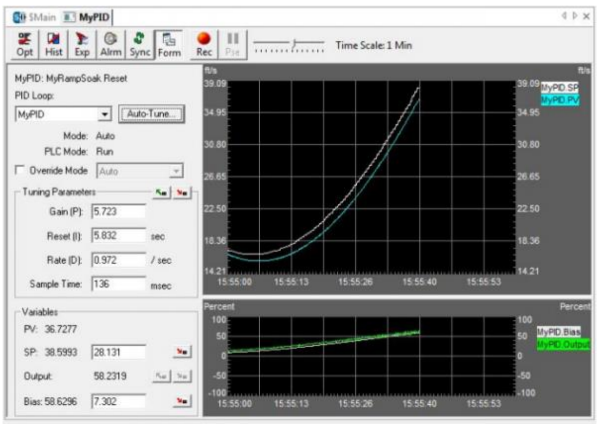
Integrated email capabilities allow your PLC to notify you and any other needed personnel when critical events occur at your facility. This way you can stay informed about any possible failures and can take immediate corrective action.



Subject: Fault Status

```
From:     PACController@Plant.com
Sent:     July 14, 2015 11:36 AM
To:       OpsMgr@Plant.com
Subject: Fault Status

Error Alarm Stopped Operation
```

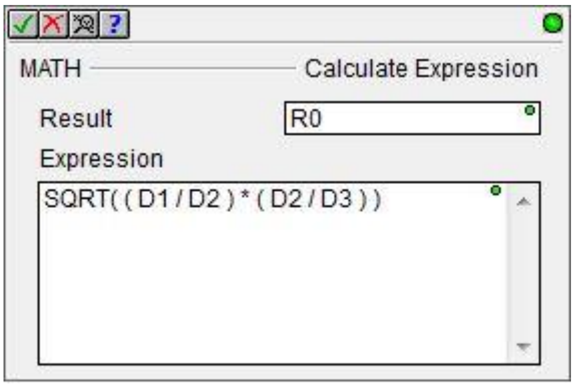


PID Options

For all the process control engineers out there, this is definitely one you'll want to watch for. Check to see how easy it is to implement PID loops in your program. Are there integrated PID instructions and how easy is it to monitor and tune the process? Some PLC software provides auto tuning, PID simulators and other helpful tools dedicated to PID loops.

Powerful Math Functions

Powerful math functions that are easy to use are a definite plus to any PLC software. The ability to enter complex mathematical expressions (Ex. SQRT((D1/D2)*(D2/D3))) with floating points and/or integers directly into an editor, without having to load accumulators or use multiple lines of code, is a big timesaver.


MATH — Calculate Expression

Result: R0

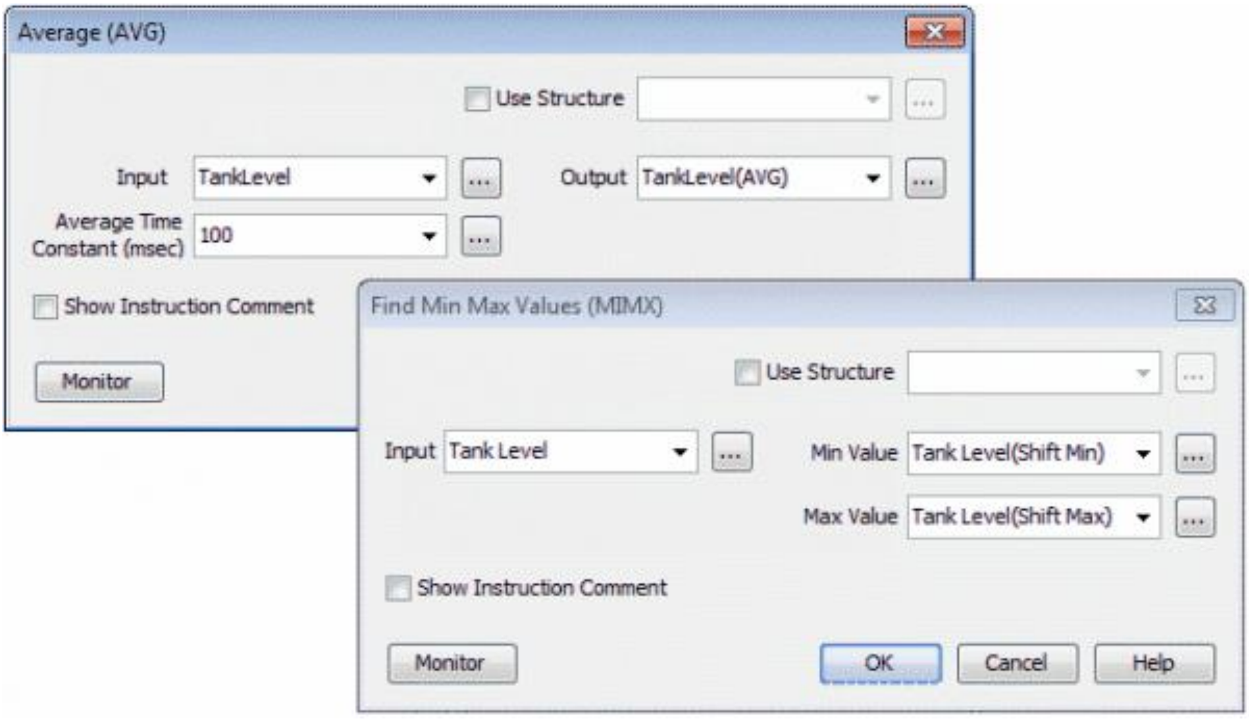
Expression: SQRT((D1 / D2) * (D2 / D3))

Task Manager

The task manager is a great way to organize and control the execution cycle of your program without any extra lines of code. Breaking your program into subroutines that execute only when needed (every scan, every second, first scan only, or when called) can be a lifesaver when scan time is a concern.

Integrated Function Blocks

Integrated function blocks make programming a lot faster. Instead of coding a large amount of ladder logic to handle a complex task, function blocks can do the work for you. PID loops, motion commands, communication functions, etc., can be handled quickly and easily with configurable function blocks.


Average (AVG)

Use Structure

Input: TankLevel

Output: TankLevel(AVG)

Average Time Constant (msec): 100

Show Instruction Comment

Monitor

Find Min Max Values (MIMX)

Use Structure

Input: Tank Level

Min Value: Tank Level(Shift Min)

Max Value: Tank Level(Shift Max)

Show Instruction Comment

Monitor | OK | Cancel | Help

Other Things to Consider…

Programming Languages

Although ladder logic is still by far the preferred programming language in our industry, there are other languages that you may find appealing. Structured text, instruction lists, sequential function charts and function block diagrams are all options that are available. If you come from a computer programming world you might prefer the "If…Then" loops that structured text offers, or if you are experienced with assembly language, instruction list programming might be for you. Each language offers something different, but choose the one that is easiest for you to program, troubleshoot and maintain.

Tag name vs. Fixed memory

There are many preferences and objections to both but you should definitely consider which is most useful to you and your particular application. Tag name databases can be easily integrated with HMIs and other database software but fixed memory controllers may offer better searchability. If you are unsure, feel free to download our tag name based Productivity Suite software or our fixed memory Do-more Designer software to get a feel for the differences. They are both free!

This list is in no way intended to encompass every aspect of "good" PLC software but it provides you with some features and tools that we think are important and that we think would greatly help with your PLC project development. If you would like to get more information on the PLC software we offer, take them for a test run, or watch informative videos on how to use our software, check out the links below:

- CLICK
- Do-more Designer
- Productivity Suite

Learn more about…

Productivity2000

http://N2ADC.com/a2-7c

Learn more about…

Do-more

Do-more PLC - Benefits and Features of Do-more Programmable Logic Controller
Do-more
How To Do-more
Overview
AUTOMATIONDIRECT
0:00 / 2:56

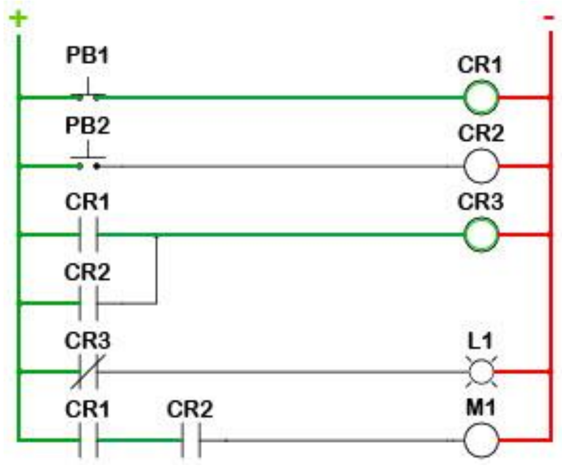
http://N2ADC.com/v580w

Chapter 5 - 1

Understanding Ladder Logic

The Structure

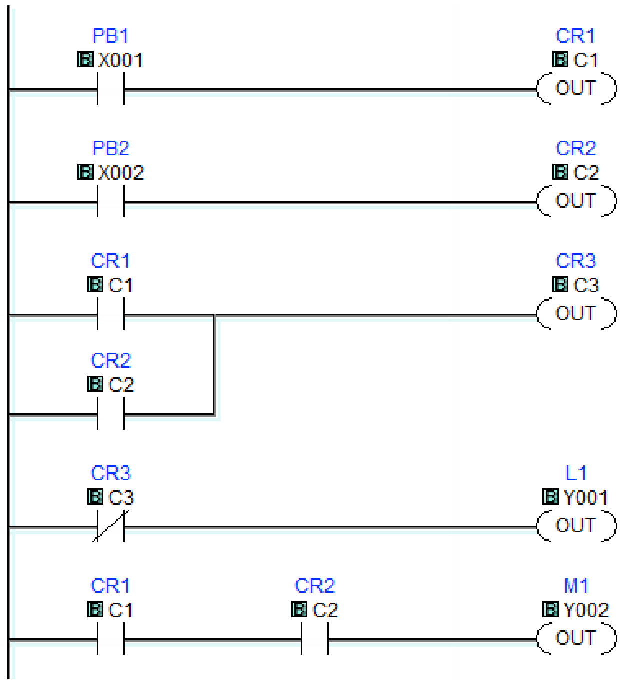
The structure behind ladder logic is based on the electrical ladder diagrams that were used with relay logic. These diagrams documented how connections between devices were made on relay panels. They are called “ladder” diagrams because they are constructed in a way that resembles a ladder with two vertical rails and rungs between them. The positive power rail (on the left) flows to the negative power rail (on the right) through the physical devices connected on the rung. In this example, “PB” stands for “pushbutton” and “CR” stands for “control relay”. “L1” is a light.



The Ins and Outs

Ladder logic was designed to have the same look and feel as ladder diagrams, but with ladder logic the physical contacts and coils are replaced with memory bits. Let’s take a look.

For this program, we took the relay logic’s ladder diagram and duplicated it with ladder logic. No more hard-wired logic, but memory locations instead. Some of these memory locations are used internally and others are used with external inputs and outputs. To monitor and control real world devices, they will need to be wired to I/O modules. For this particular PLC, these inputs and outputs are assigned to X and Y memory addresses like the X001 seen with PB1. This normally open contact’s state is read from the input on the I/O module where the physical pushbutton is connected. On the other hand, each Y bit will have an output device wired to it as seen with the light controlled by Y001. All of the other locations are assigned to internal bits that we can use as needed. One side note, today’s PLC CPUs offer many types of functions, not just simple contacts and coils. Math, Shift Registers, Drum Sequencers, etc., are available to aid in your programming but for now, we’ll keep it simple.



The Execution

The CPU will interpret the logic in a sequential order. Starting at the top left of the program, the CPU will work its way down the rail executing each rung or sub rung from left to right. So if PB1 is pressed, the CPU will turn ON CR1. Since CR1 has changed states, in rung 3 the CPU will activate CR3. CR3's normally-closed state is used in rung 4, so the CPU will then turn OFF L1. Even though we still refer to coils and contacts in ladder logic, remember that they are memory representations, not actual devices. Once the CPU reaches the last rung it will loop back to the start of the program and run it all over. This process will continue as long as the CPU is powered and in the RUN mode. The time it takes the CPU to loop back to the beginning is known as scan time. Scan time can be important to applications where timing is critical. Subroutines and special purpose I/O modules can be used to help reduce the scan time if needed.

The Logic Behind The Ladder

So what logic can ladder logic actually perform? With the increasing demand for functionality and ease of use, many of today's PLCs incorporate function blocks with ladder logic. The structure of the program is still ladder with the more complex instructions being function blocks. So to answer the question, let's look at a few examples:

1. **Boolean Logic:** The ON/OFF, TRUE/FALSE algebra of binary systems. The basics of which are AND, OR, and NOT operators. To put it simply, rung 5 in our code needs CR1(C1) AND CR2(C2) to turn ON motor M1 (Y002).

2. **Timing:** Timer instructions are available to allow for on-delayed or off-delayed events. Once triggered, the timer will turn its associated output ON (on-delay) or OFF (off-delay) after the set time has elapsed.

3. **Counting:** Count-up and count-down functions increase or decrease the counter value on every transition of the input.

4. **Comparisons:** Compare instructions are available to determine if values are less than, equal to, or greater than each other.

5. **Math:** These instructions not only allow for the simple addition and subtraction but also for more complex operations like tangents, square roots, etc.

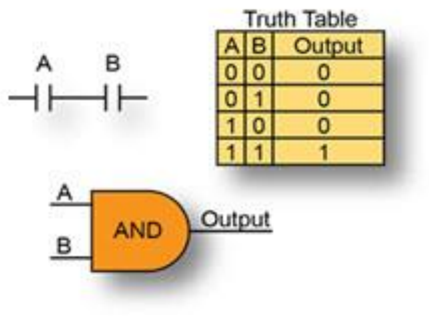
6. **Special functions:** PID loops, communication instructions, shift registers, drum sequencers, ramp generators, etc.

Chapter 5 – 2

Basic Instructions in Ladder Logic

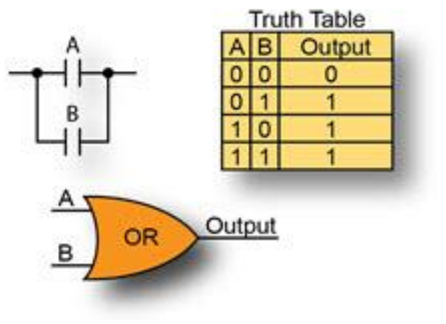
Now that we have a better understanding of what Ladder Logic is we can dig a little deeper into how it works. And to do that, we need to first understand Boolean math and logic gates. Now, this blog is not intended to be a Digital Systems class so put away your Karnaugh maps, you won't need them. Instead, we'll just look at a few logic gates and how they work.

The two logic gates that you will see represented most often in Ladder Logic are the AND and OR gates. As the truth tables below show, the AND gate needs all inputs (A and B) to be true before the output becomes true, whereas the OR gate needs just one input (A or B) to be true to make the output true. These truth tables can be duplicated with relay contact logic by connecting normally open contacts in series (AND gate) or parallel (OR gate) as seen below.



Truth Table

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1

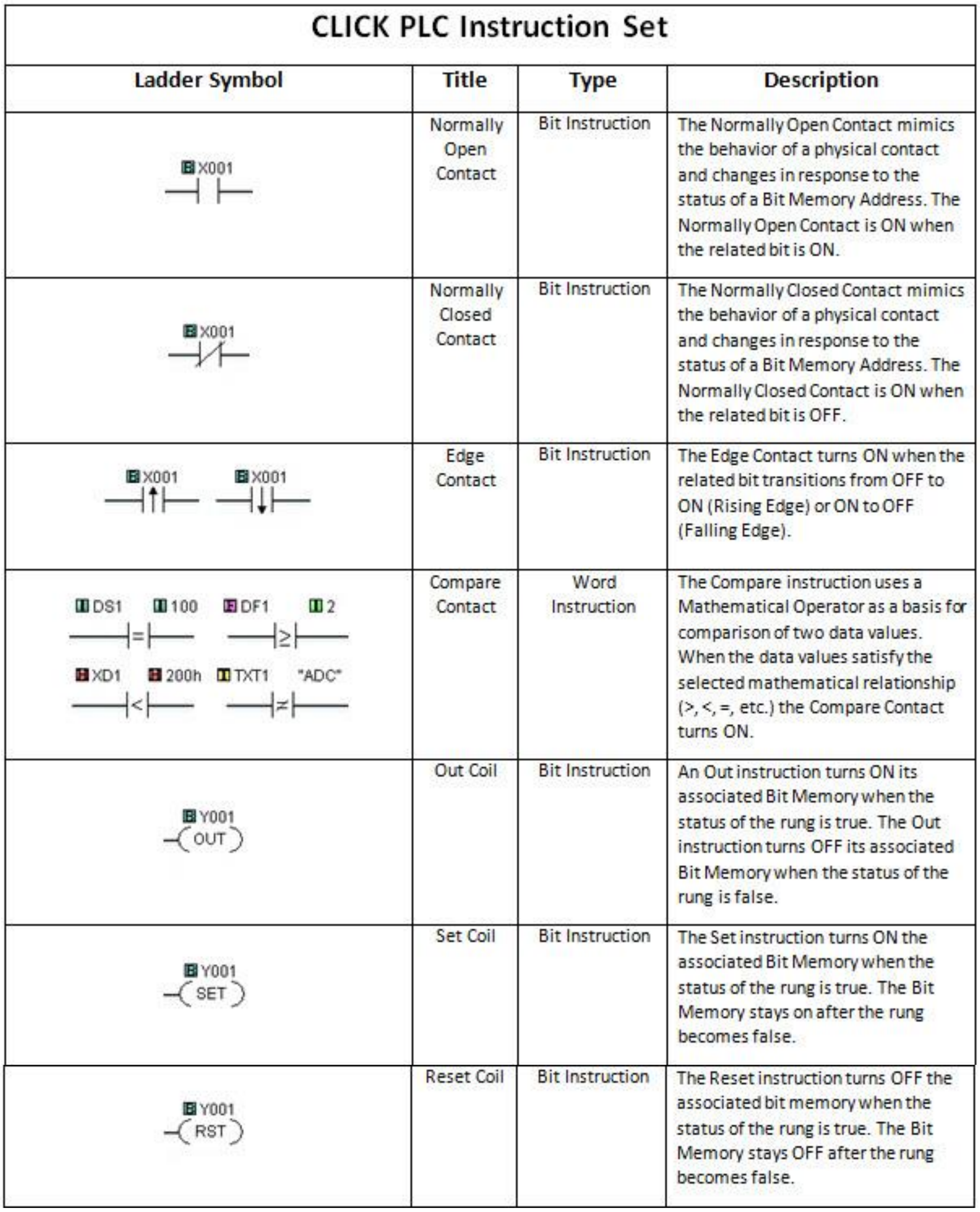


Truth Table

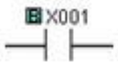
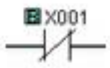
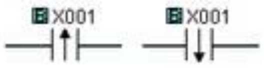
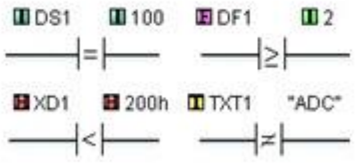
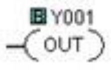
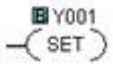
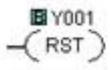
A	B	Output
0	0	0
0	1	1
1	0	1
1	1	1

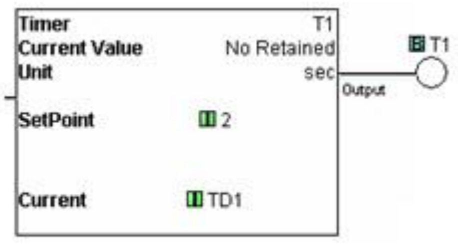
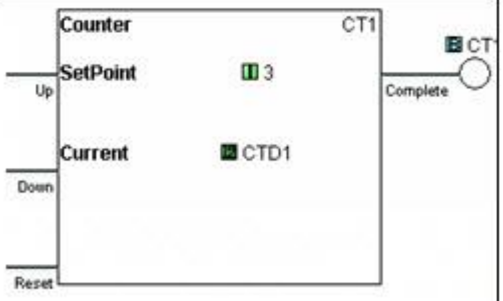
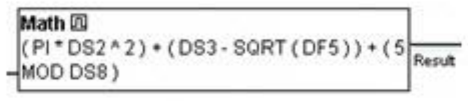
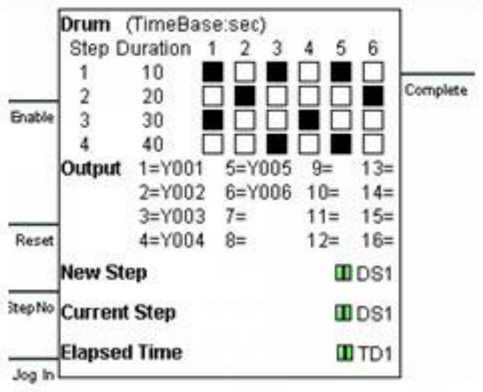
To equate this to a control circuit, let's say Input A and Input B (often referred to as permissive bits) are the **Door Open** and **Motion Detected** inputs of a home alarm system. When wired in series, as with the AND gate, both conditions will have to be met before the alarm is activated. The door will have to be opened and the motion detector tripped before the alarm is triggered. With the OR gate, the inputs are wired in parallel and only one of the conditions is needed. In this case, the door opening or the motion detection will trigger the alarm. The option you choose is dependent on your application and on how the system is expected to perform. Pretty simple, right? And in its very basic form, that's what Ladder Logic is. It's the arrangement of permissive bits or contacts into a Boolean expression that determines whether an output should be ON or OFF.

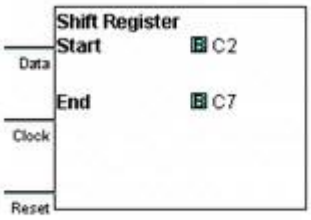
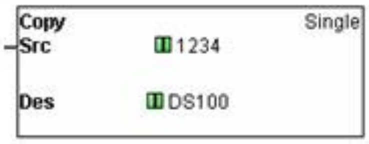
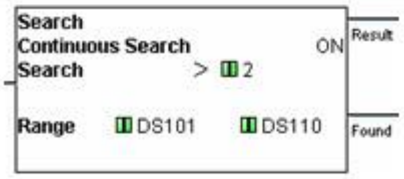
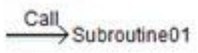
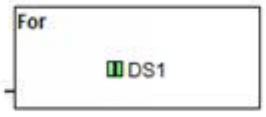
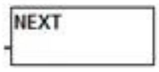
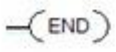
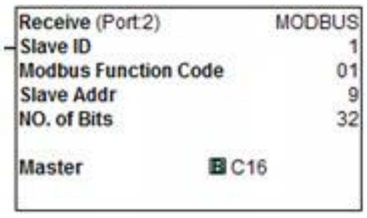
Today's Ladder Logic programming has advanced to include more than just simple contacts and coils but the same underlying principle holds true. The true or false state of each input element in a rung and how it is connected will determine the output's state. So let's take a look at some of the instructions available in current PLCs. And for that we will use the CLICK PLC since it has a simple, easy-to-use instruction set. The table below contains all of the available Ladder Logic elements in the CLICK programming software with descriptions of their functions.

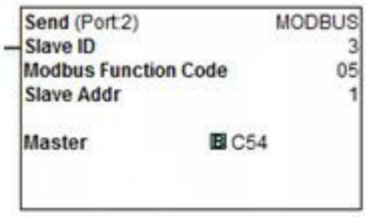
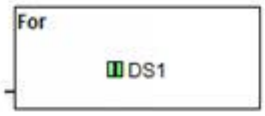
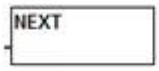
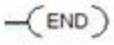
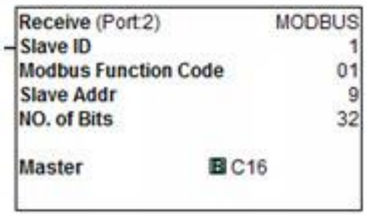
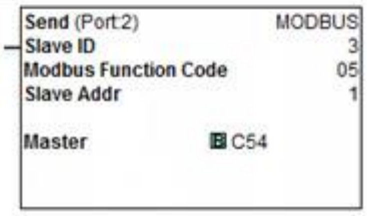


CLICK PLC Instruction Set

Ladder Symbol	Title	Type	Description
	Normally Open Contact	Bit Instruction	The Normally Open Contact mimics the behavior of a physical contact and changes in response to the status of a Bit Memory Address. The Normally Open Contact is ON when the related bit is ON.
	Normally Closed Contact	Bit Instruction	The Normally Closed Contact mimics the behavior of a physical contact and changes in response to the status of a Bit Memory Address. The Normally Closed Contact is ON when the related bit is OFF.
	Edge Contact	Bit Instruction	The Edge Contact turns ON when the related bit transitions from OFF to ON (Rising Edge) or ON to OFF (Falling Edge).
	Compare Contact	Word Instruction	The Compare instruction uses a Mathematical Operator as a basis for comparison of two data values. When the data values satisfy the selected mathematical relationship (>, <, =, etc.) the Compare Contact turns ON.
	Out Coil	Bit Instruction	An Out instruction turns ON its associated Bit Memory when the status of the rung is true. The Out instruction turns OFF its associated Bit Memory when the status of the rung is false.
	Set Coil	Bit Instruction	The Set instruction turns ON the associated Bit Memory when the status of the rung is true. The Bit Memory stays on after the rung becomes false.
	Reset Coil	Bit Instruction	The Reset instruction turns OFF the associated bit memory when the status of the rung is true. The Bit Memory stays OFF after the rung becomes false.

 Timer T1 Current Value No Retained Unit sec T1 Output SetPoint 2 Current TD1	Timer	Word Instruction	An ON Delay Timer measures the time duration that begins with a transition of the enable rung from OFF to ON. Beyond this transition point, the Timer increases the Current Value; when it reaches the SetPoint, the Timer Bit is turned ON. An OFF Delay Timer measures the time duration that begins with a transition of the enable rung from ON to OFF. Beyond this transition point, the Timer increases the Current Value; when it reaches the SetPoint, the Timer Bit is turned OFF.
 Counter CT1 CT SetPoint 3 Complete Up Current CTD1 Down Reset	Counter	Word Instruction	When enabled, a Counter instruction counts up or down (depending on user settings) until it reaches the SetPoint. The Counter counts in response to the transition from OFF to ON of the enabling rung (up or down rung).
 Math (PI * DS2 ^ 2) + (DS3 - SQRT (DF5)) + (5 MOD DS8) Result	Math	Word Instruction	The Math instruction solves a user-defined formula during the execution of the Ladder Program. Once the enable rung transitions from OFF to ON the formula will be solved and the result will be stored in the data format and location selected for the Result.
 Drum (TimeBase:sec) Step Duration 1 2 3 4 5 6 1 10 2 20 3 30 4 40 Complete Enable Output 1=Y001 5=Y005 9= 13= 2=Y002 6=Y006 10= 14= 3=Y003 7= 11= 15= 4=Y004 8= 12= 16= Reset New Step DS1 StepNo Current Step DS1 Elapsed Time TD1 Jog In	Drum Instruction	Special Instruction	The Drum instruction simulates an electromechanical drum sequencer, using either a Time Based or an Event Base sequencing strategy. Each Drum instruction is capable of sequencing through 1 to 16 steps and turning ON as many as 16 outputs in a user defined pattern. Outputs can be either physical outputs or internal control relays.

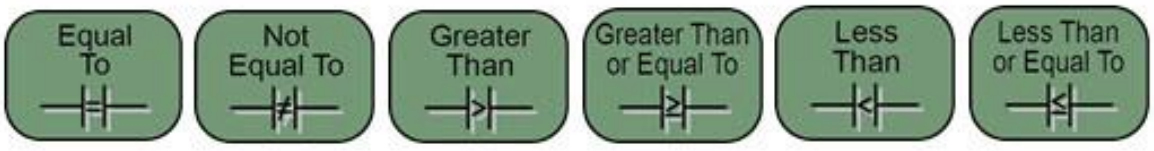
 Shift Register Start C2 End C7 Data Clock Reset	Shift Register	Bit Instruction	The Shift Register instruction shifts a range of control bits one memory location with each OFF-to-ON transition of the Clock pulse. If the Starting Address is lower than the Ending Address, the Shift register will Shift from the Starting Address to the Ending Address. If the Ending Address is lower than Starting Address then Shift Register will Shift from Ending Address to the Starting Address.
	Copy Instruction	File Instruction	The Copy instruction is used to copy a data or text value from its Source location to a specified Destination register.
 Search Continuous Search ON Result Search > 2 Range DS101 DS110 Found	Search Instruction	File Instruction	The Search instruction is used to search for a data value that meets the specified condition and that is located within a specified range of data registers.
	Call Instruction	Program Control	The Call instruction is required to call or execute a Subroutine Program from the Main Program.
	For Instruction	Program Control	The For instruction indicates the starting point of a For-Next loop, and based on its user setting, determines how many times the For-Next loop will be executed in one program scan. Between the For instruction rung and the Next instruction rung, place the rungs of logic that should be repeated multiple times.
	Next Instruction	Program Control	The Next instruction indicates the end of a For-Next loop.
	End Instruction	Program Control	The End instruction marks the termination point of the normal program scan. The End instruction tells the CPU that there are no more rungs to be processed.
 Receive (Port 2) MODBUS Slave ID 1 Modbus Function Code 01 Slave Addr 9 NO. of Bits 32 Master C16	Receive Instruction	Communication	The Receive instruction allows you to use the available communication ports on the CLICK CPU modules as a network master and read data from external devices.

 Send (Port 2) MODBUS Slave ID 3 Modbus Function Code 05 Slave Addr 1 Master C54	Send Instruction	Communication	The Send instruction allows you to use the available communication ports on the CLICK CPU modules as a network master and write data to external devices.
	For Instruction	Program Control	The For instruction indicates the starting point of a For-Next loop, and based on its user setting, determines how many times the For-Next loop will be executed in one program scan. Between the For instruction rung and the Next instruction rung, place the rungs of logic that should be repeated multiple times.
	Next Instruction	Program Control	The Next instruction indicates the end of a For-Next loop.
	End Instruction	Program Control	The End instruction marks the termination point of the normal program scan. The End instruction tells the CPU that there are no more rungs to be processed.
 Receive (Port 2) MODBUS Slave ID 1 Modbus Function Code 01 Slave Addr 9 NO. of Bits 32 Master C16	Receive Instruction	Communication	The Receive instruction allows you to use the available communication ports on the CLICK CPU modules as a network master and read data from external devices.
 Send (Port 2) MODBUS Slave ID 3 Modbus Function Code 05 Slave Addr 1 Master C54	Send Instruction	Communication	The Send instruction allows you to use the available communication ports on the CLICK CPU modules as a network master and write data to external devices.

As you can see, there are many more options available than just normally open contacts and many function blocks have been incorporated into Ladder Logic. For instance, the Drum and communication instructions are blocks of code that are stored and available for you to configure and use in your program. Now, we aren’t going to discuss every instruction available but one of the more frequently used instructions is the Compare contact.

The Compare contact can be programmed to look at two different numerical values; either or both values can be a variable or fixed value. The Compare instruction will evaluate if the two values are:

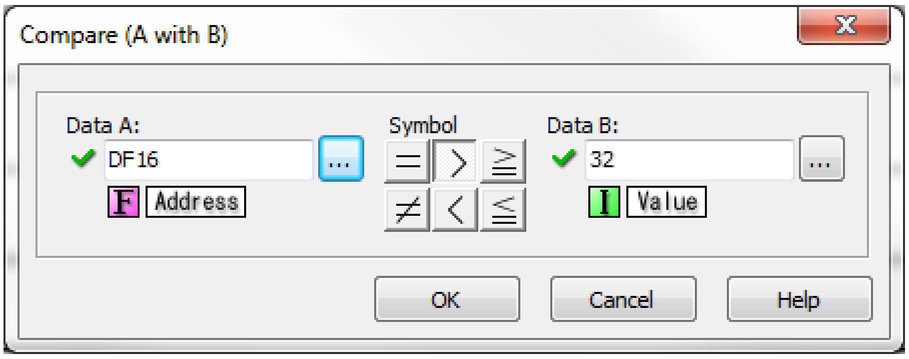
- Equal
- Not Equal
- One Greater Than the other
- One Greater Than or Equal To the other
- One Less Than the other
- One Less Than or Equal To the other



If the evaluation condition is met, then the result will be true, passing the logic path along to the next instruction or turning the rung output ON.

So, how can we use the Compare contact in an application? Imagine having a freezer full of frozen fish. The key word there is frozen! We don't want to risk a smelly disaster if the fish were to thaw, so we've installed a sensor inside the freezer to monitor the temperature. The temperature sensor's output produces an analog signal that is wired into a PLC. The PLC is programmed to take the analog signal and convert it into degrees Fahrenheit. If the temperature reads greater than 32 degrees, we can sound an alarm horn to alert us that the frozen fish will soon thaw and spoil.

Using the Compare contact instruction dialog box as seen below, we can compare our actual Freezer Temperature (stored in memory location DF16) to the constant 32, which represents 32 degrees Fahrenheit.


Compare (A with B)

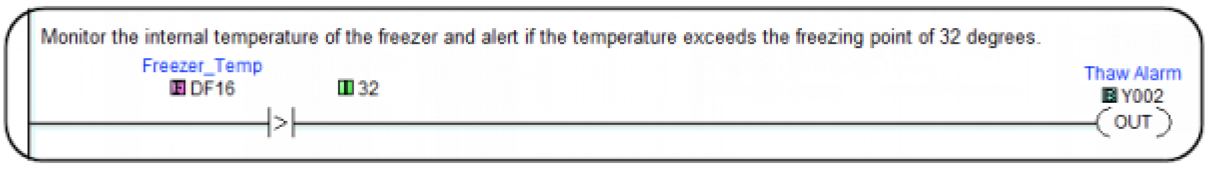
Data A: DF16 (F Address)

Symbol

Data B: 32 (I Value)

The Compare contact dialog box allows us to select any of the six different comparison methods. For this example, we have selected Greater Than so that if our Freezer Temperature ever becomes Greater Than the number 32, our Compare contact will become true.

An Out coil is then programmed from our Compare contact, as seen below, and will provide the signal that is wired to the external Alarm Horn, which alerts us to a problem with our freezer and hopefully giving us time to avoid disaster.


Monitor the internal temperature of the freezer and alert if the temperature exceeds the freezing point of 32 degrees.

Ladder Logic is the most widely used programming language in industrial automation today. Its ease of use, traceability, and visual representation of physical components make it the favored programming method of many engineers. If you are new to PLC programming and would like to try Ladder Logic for yourself, download any of our programming software packages mentioned below for free and see what you think. After all, experience is the best teacher.

- CLICK
- Do-more Designer
- Productivity Suite

Give it a Try…

If you would like to try an exercise in ladder logic, AutomationDirect has created a beginner's programming exercise. This exercise was actually created for the Boy Scouts of America to help teach ladder logic to future PLC programmers. It uses the simulator included with the Do-more Designer programming software for our Do-more PLC series. The software is free and you do not need any hardware, so try it. Download the free software and watch the video below.

BSA Programming Merit Badge - Pinewood Derby Controller via Factory Auto...
Programming
Merit Badge
0:00 / 11:42
YouTube

For Free Do-more Software Visit: http://n2adc.com/zge3x

Chapter 5 - 3

Ladder Logic in Action

Ladder logic programming for industrial controllers has evolved significantly over the past 30 years, and now supports advanced functionality such as process control, motion control, data manipulation, networking and data acquisition. However, when you start with a blank page to design a new control system, large or small, there are many basic functions this tried-and-true language is still asked to perform, and in fact, perform effortlessly.

Most every controls designer has implemented at least one of these functions in every system they have commissioned. Even when moving from one supplier's platform to another, the ladder code design can be copied and adjusted for the particular controller, becoming tested "building blocks" of logic to speed up programming and reduce troubleshooting time. Let's look at some of the most common control system functions and some practical ways to implement them.

Object Detection

One of the most pervasive functions of a control system is detecting the presence of an object. Whether you are detecting the presence of an object passing by on a conveyor, the closure of a gate, or the presence of a machine part as it goes through its motions, object detection is a staple of the automation industry. There are a myriad of object sensing technologies, such as mechanical; inductive, capacitive and ultrasonic devices that detect nearness (proximity); and photoelectric sensing using light beams.

Limit Switch

The most basic sensor is a limit switch, an electromechanical device used to detect the presence or absence of an object. The switch operates its set of contacts when its actuator comes into physical contact with the sensed object. Actuator styles offer application-specific means of contact – rollers, levers, springs, wands, plungers, etc. However, since they consist of moving parts, they are prone to wear and damage; making physical contact with the sensed object is not always desirable or possible.

Inductive proximity Switches

Inductive proximity switches are the most common and inexpensive non-contact sensing technology, used to detect the presence of metallic objects without actually touching it. Their high-speed switching and small size make them indispensable in automation applications. Inductive proximity switches consist of a coil driven by an oscillator. The oscillator creates an electromagnetic field that appears at the active face of the switch. If a metal target enters this area, the electromagnetic field is reduced and the switch turns on or off. Some typical inductive sensor applications are counting metallic objects, monitoring the position of elements in a machine, sensing the presence of metallic parts like screws, etc., and measuring rotational speed.

Proximity Sensors

Proximity sensors typically come in shielded and unshielded styles. With a shielded proximity sensor, the face of the sensor can be mounted flush with metal, whereas an unshielded sensor should NOT be mounted flush with metal (otherwise the sensor will always be ON). In many applications, flush mounting is a requirement. Note that unshielded proximity sensors allow for greater sensing distances. Typical sensing ranges for inductive proximity sensors vary from 1 to 35 mm.

Capacitive Sensors

Capacitive sensors detect objects with a dielectric constant different from air, which makes them ideal for a much wider range of material detection, such as wood, liquids and plastic. Their operation is similar to inductive sensors, but instead creating an electrostatic field (vs. electromagnetic) that is changed by the presence of the object. Capacitive sensing ranges can typically reach up to 40mm.

Ultrasonic Proximity Sensors

Ultrasonic proximity sensors are based on the emission of a sound impulse and the measurement of the time elapsing of the return echo signal reflected by the detected object. The ultrasonic beam is well reflected by almost all materials (metal, wood, plastic, glass, liquid, etc.) and is not affected by colored, transparent, or shiny objects. This allows the user to standardize on one sensor for many materials without any extra setup or sensing concerns. This sensing technology typically offers ranges up to 6 meters.

Photoelectric Sensors

Photoelectric sensors use a variety of sensing technologies that address diverse application configurations, all using light beams as the detecting medium. The three most popular are diffuse, reflective and through-beam styles. The light source used – visible, infrared, LED, laser – will affect the sensing distance.

Diffuse Sensors

In diffuse sensors, the emitter and receiver form part of the same unit. The optical beams are either parallel or slightly converging. The presence of an object in the optical field causes diffused reflection of the luminous beam. The receiver detects the reflection from the object itself. The reflective properties of the object are important. It is generally possible to reliably detect the presence of any object unless it is perfectly reflective or black. Diffuse sensors offer a typical sensing range of up to approximately 2 meters.

Reflective Sensors

A reflective sensor also houses the emitter and receiver in the same unit, with the optical beams being parallel. The emitter's luminous beam hits a reflector mounted on the opposite side of the object's travel path and is redirected toward the receiver. Detection occurs when the path of the beam is interrupted by the presence of an opaque object. Operating distance mainly depends on the quality of the reflector used and on the optical-beam angle; ranges are typically up to 15 meters.

Through-beam Sensors

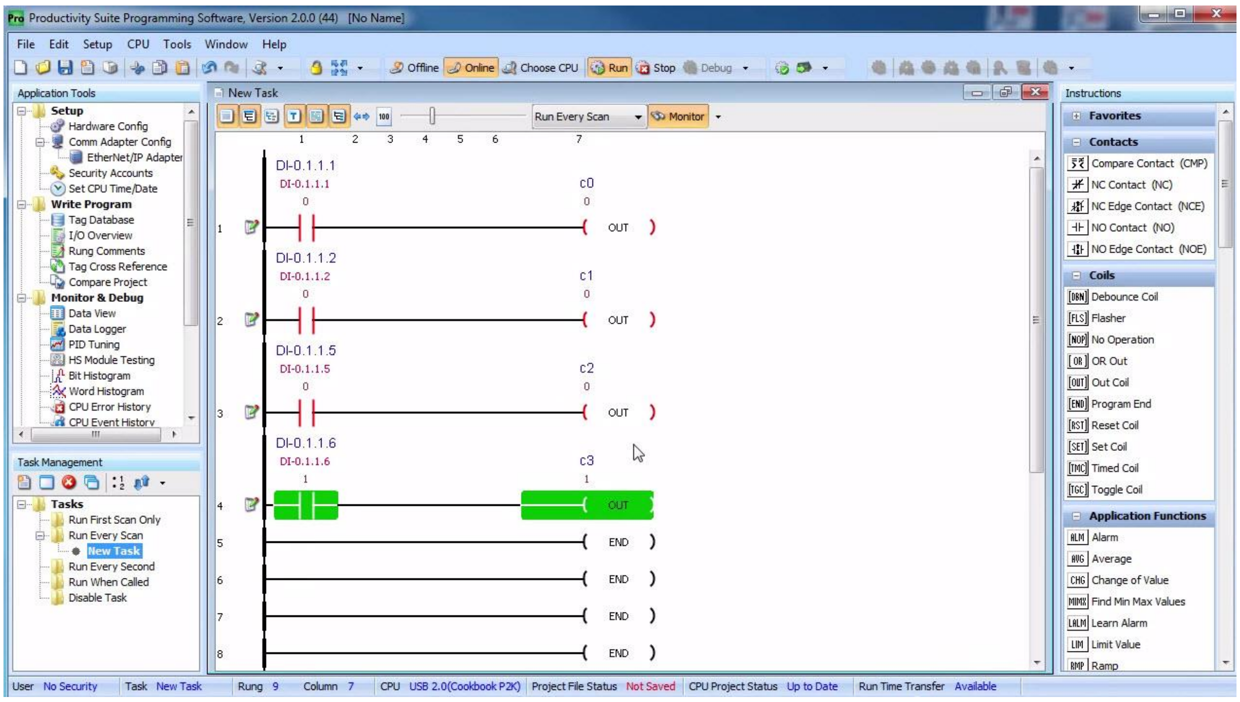
With through-beam sensors, the emitter and receiver are housed in two separate units and are installed in line with each other on either side of the object path. Detection occurs when the beam is interrupted by the presence of an opaque object. This style is the most accurate, with the farthest sensing ranges, up to 30 meters, and 50 meters or longer when using laser sensors.

Object Detection + Ladder Logic

So how does this look in Ladder Logic? Well, once the sensor has been wired to the input module and the module has been configured in the software's hardware configuration (done automatically with the auto-discover feature), the PLC CPU will assign a memory location to it. In the image on the next page, you can see that the Productivity2000, which is a tag name-based controller, has assigned the names DI-0.1.1.1, DI-0.1.1.2, DI-0.1.1.5 and DI-0.1.1.6 to the sensor inputs. These tag names identify the type of module (DI = discrete input), base group (DI-0 = local base, DI-1 = remote base), expansion rack number (DI-0.1 = 1st rack in local base group, DI-1.2 would be 2nd rack in the remote base group), slot of base the module is in (DI-0.1.1 = slot 1, DI-0.1.5 = slot 5, etc.) and the point on the module the sensor is wired to (for our example the sensors are wired to points 1, 2, 5, and 6). The programmer can easily change these to more meaningful names.

Four normally-open contacts are then added to four individual rungs in the main program of the ladder code. These contacts are assigned the tag names that were configured for our object detection sensors.

Normally-closed contacts and edge-driven contacts could have also been chosen. As you can see in rung 4, when the sensor detects an object, the contact will change state and the rung will activate the C3 output coil. This coil is an internal bit but if a discrete output module is installed, this coil can be programmed as a physical output used to activate an alarm, turn on a light, fire a diverter arm, etc.



Learn More..

To learn more, see this series of videos on the different types of sensing technologies and their suitable application spaces; see typical ladder logic programming for detecting objects with CLICK, Do-more and Productivity series PLCs.

Tech Tip: Overview of Simple Object Detection and Sensing

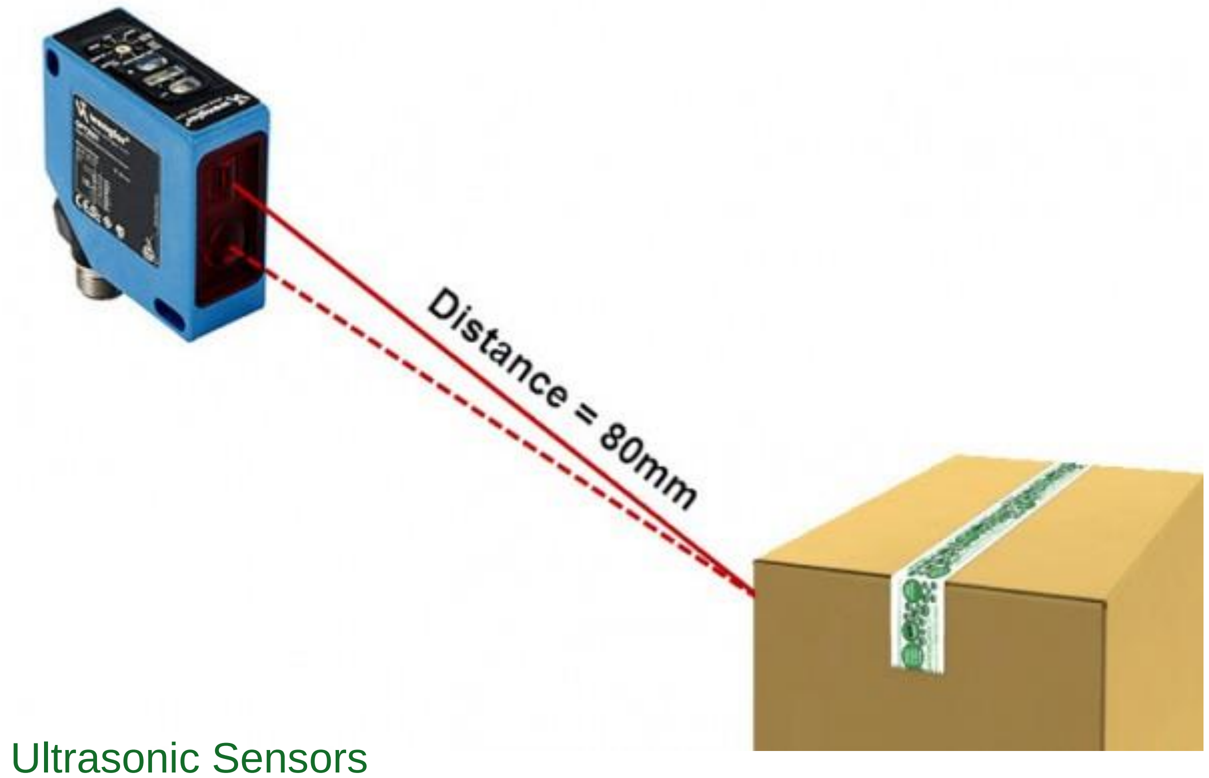
Tech Tip: Selecting The Right Limit Switch For Your Industrial Automation Project

Tech Tip: Proximity Sensors for Object Detection

Visit: www.AutomationDirect.com/Cookbook

Distance Measuring

A more advanced object detection function involves measuring the actual distance of an object from the sensing point. Ultrasonic sensors and laser distance sensors are commonly used to measure how far away an object is, out to 100 meters and beyond. Laser devices can be incredibly accurate, with resolutions down to 8um.



Ultrasonic Sensors

Ultrasonic sensors use sound waves to measure distance, sending out a burst of audio energy and waiting for it to return. Given the rate sound travels through air, the distance is just the time it takes for the energy to get to the object and back to the sensor, divided by two. Using sonic energy, these sensors do not have issues with things that plague optical sensors like smoke, dust, fog and steam. Conversely, they can be less effective in environments with high vibration.

Ultrasonic sensors fill the gap between laser distance sensors, which have long ranges but are more expensive, and proximity sensors, which have very short ranges and are typically inexpensive. Ultrasonic sensors can measure distances up to several meters, which is all that is needed in many applications such as bulk material distance measurements.

Laser Sensors

Laser sensors use highly-focused (coherent) light beams to perform distance measurement. Techniques include triangulation, time of flight, phase-shift and frequency modulation. Laser sensors for industrial use typically employ time of flight (transit time), similar to the ultrasonic method in that they measure the time it takes the laser pulse from the emitting device to reach the target and return.

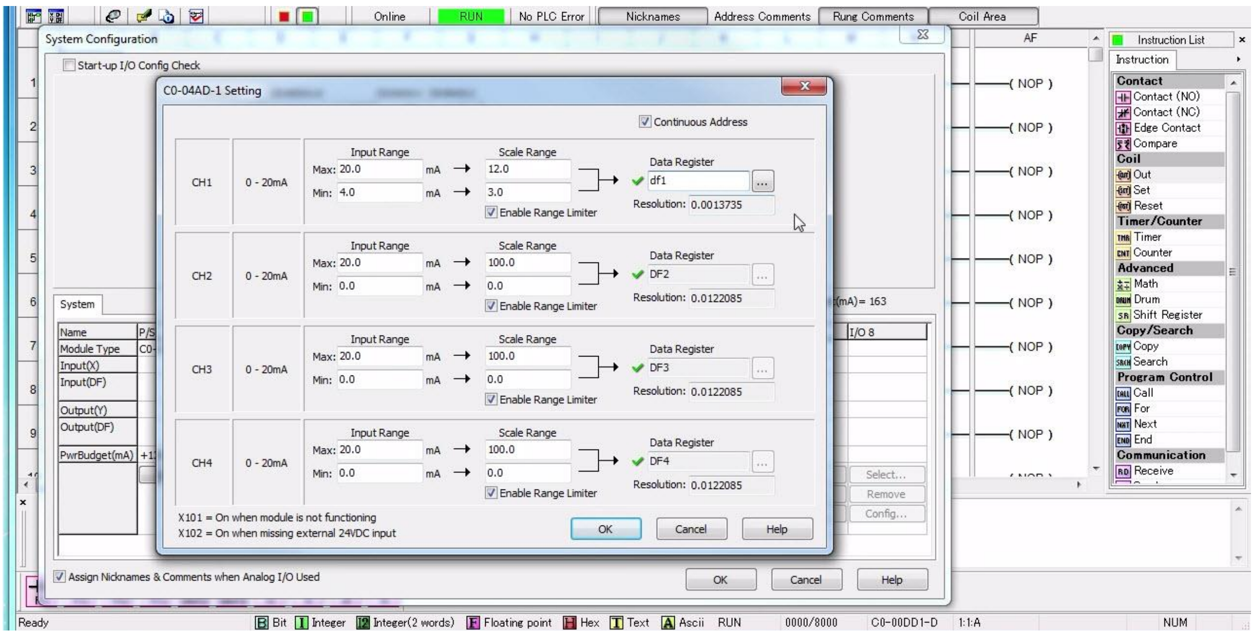
Typical laser distance sensors come in two power ranges referred to as Class-I and Class-II lasers. The main difference is Class-I lasers have a maximum output of 0.5 mW, where Class-II lasers can have up to 1mW of output. In an industrial environment with subdued lighting, the Class-I laser devices will work just fine. But, if in a brightly lit area the more powerful Class-II laser may warrant consideration. Keep in mind that the safety requirements are different for each. Class-I lasers are considered safe from creating any potential hazards. Class-II lasers are also considered safe because of the assumption that your eye's blink reflex will protect you. You probably don't want to be the one that tests that, so you will want to be sure the Class-II laser is not used where potential eye contact can be made either directly or via reflection.

The extreme resolution and fast response time of a laser sensor make it ideal for precision measurements, such as thin film thickness, or even detecting the diameter of rolls of film.

One issue with laser sensors is since they depend on visible light, they have a tough time detecting transparent objects like glass, plastic bottles, etc. Laser sensors will also suffer in environments with lots of dust and particulates floating in the air, since they have to 'see' the object, and they have different responses depending on the color of the object. Some objects reflect laser light better than others.

Distance Measurement + Ladder Logic

Now, let's look at some distance values in Ladder Logic. Laser and ultrasonic distance sensors provide an analog value to the PLC; therefore, they will need to be wired to an analog input module. For this example, the CLICK PLC will be used. The CLICK PLC does offer the auto-discover feature and once the modules have been installed, the PLC CPU will recognize all of the I/O modules available. Since these inputs are analog, we will need to scale them appropriately. With the CLICK PLC, analog scaling can be done in the system configuration as seen below. Our ultrasonic sensor is connected to channel one of the C0-04AD-1 analog input card. The raw analog (4-20mA) input value from this channel is scaled to a range of 3 to 12 inches and is stored in memory location DF1.

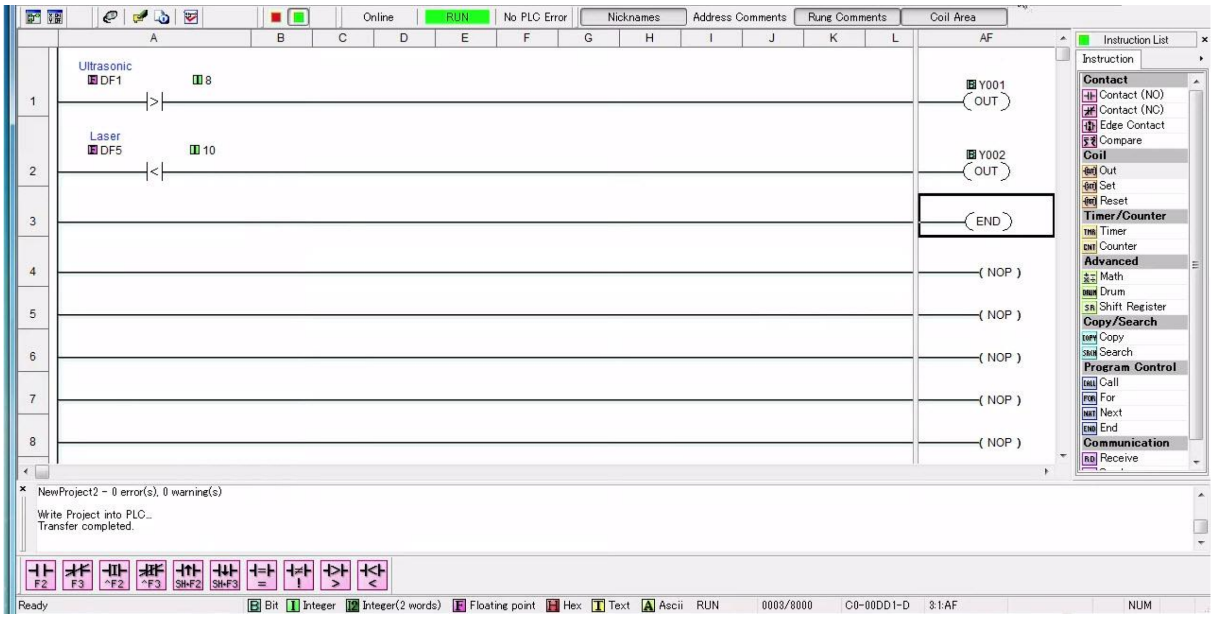

C0-04AD-1 Setting

☑ Continuous Address

		Input Range		Scale Range	Enable Range Limiter	Data Register	Resolution
CH1	0 - 20mA	Max: 20.0 mA	Min: 4.0 mA	12.0 / 3.0	☑	df1	0.0013735
CH2	0 - 20mA	Max: 20.0 mA	Min: 0.0 mA	100.0 / 0.0	☑	DF2	0.0122085
CH3	0 - 20mA	Max: 20.0 mA	Min: 0.0 mA	100.0 / 0.0	☑	DF3	0.0122085
CH4	0 - 20mA	Max: 20.0 mA	Min: 0.0 mA	100.0 / 0.0	☑	DF4	0.0122085

X101 = On when module is not functioning
X102 = On when missing external 24VDC input

Now that the input has been wired, configured, and scaled, it can be used in the ladder code. Below is the ladder program and you can see that the ultrasonic sensor input is being compared to a static value (8 inches) on rung one. The compare contact is provided by the software and the output of the comparison is connected to a physical output coil. As mentioned previously, this output can be used to turn on a light, activate an alarm, open a valve, etc.



NewProject2 - 0 error(s), 0 warning(s)

Write Project into PLC...
Transfer completed.

Ready | Bit | Integer | Integer(2 words) | Floating point | Hex | Text | Ascii | RUN | 0003/8000 | C0-00DD1-D | 3:1:AF | NUM

Learn More..

With this series of videos, you will learn more about distance measuring technologies including ultrasonic and laser sensors; selecting the correct measuring style depending on the environment and object types; how to set up and use distance sensors, see typical ladder logic programming for distance measuring with CLICK, Do-more and Productivity series PLCs.

Distance Measuring Videos
AUTOMATION VIDEO COOKBOOK
Measuring Distance with PLCs - U Can Do It
UCANDOIT
AUTOMATIONDIRECT
Ultrasonic Sensors Overview
Tech Tip: UHZ Series Ultrasonic Sensors
UK Series Ultrasonic Sensors
Tech Tip: UK Series Ultrasonic Sensors

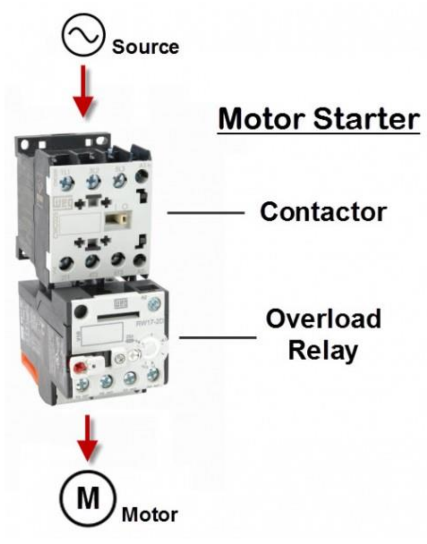
Visit: www.AutomationDirect.com/Cookbook

A/C Motors (ON/Off Control)

Most control systems have to make things move, and that usually involves motors. Lifting, pumping, robotics, conveyors, fans – pretty much everything uses a motor of some kind. General purpose three-phase AC motors are great for simple on/off systems, and Inverter-duty motors are specifically designed for operation with variable frequency drives, or VFDs.

General Purpose Motors

General purpose motors are typically connected to the main power circuit with a master circuit breaker or fuses, and use contactors to enable and disable the power to the motor; overloads protect your equipment from unexpected overcurrent/overheating that can be caused by jams or breakdowns.



Motor Starters

A motor starter is a combination of devices used to start, run, and stop an induction motor based on commands from an operator or a controller. The motor starter must have at least two components to operate: a contactor to open or close the flow of energy to the motor, and an overload relay to protect the motor against thermal overload.

Contactors

A contactor is a 3-pole electromechanical switch whose contacts are closed by applying voltage to a coil. When the coil is energized, the contacts are closed, and remain closed, until the coil is de-energized. The contactor is specifically designed for motor control, but can be used for other purposes such as resistive and lighting loads. Since a motor has inductance, breaking the current is more difficult, so the contactor has both a horsepower and current rating that needs to be adhered to.

Overload Relays

The overload relay is a device that has three current sensing elements and protects the motor from an overcurrent. Each phase going from the contactor to the motor passes through an overload relay current-sensing element. The overload relay has a selectable current setting based on the full load amp rating of the motor. If the overload current exceeds the setting of the relay for a sufficient length of time, a set of contacts opens to protect the motor from damage.

Surge Protection

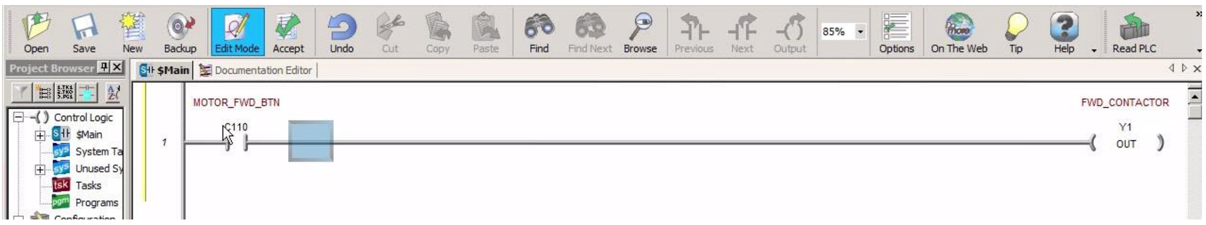
Since the signals between the contactor and PLC are inductive, it is incredibly important to protect your control system from power surges and inductive kickback. Surge protection comes in several different types – Diode, Transorb, RC networks. Which one you use depends on your application. Diodes are great if you have a DC circuit – they will offer the maximum protection BUT increase the time it takes to turn the motor off. Transorbs are great for AC or DC circuits, are inexpensive, and have a medium to short delay but only attenuate surges above their rated value, which can be above the threshold allowed by the PLC's I/O. RC circuits mostly filter noise and are not terribly effective for true surge protection.

Reversing Kits

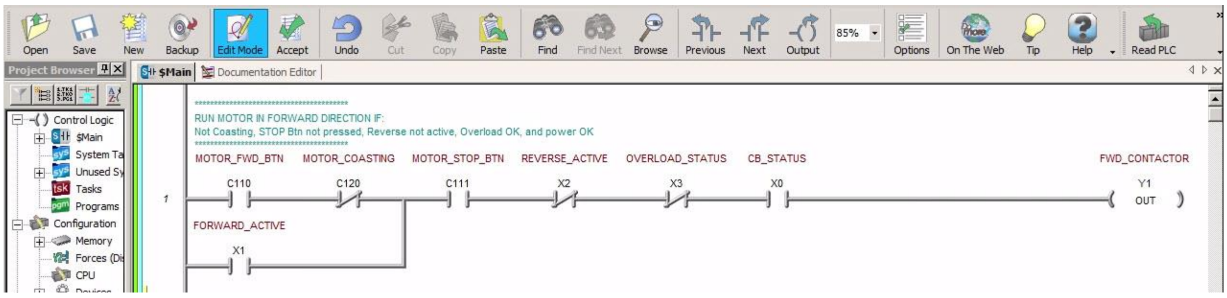
Some control systems require that the motor be reversible. On a three-phase motor that simply involves swapping any two of the three wires. Reversing kits are bus bars that are pre-wired to perform the wire swapping function and some even include extra wiring to help protect against trying to drive the motor in both directions at the same time.

AC Motor On/Off + Ladder Logic

In its simplest form, the ladder code to start an AC motor uses a start input to activate an output wired to a motor contactor. The image below is from a Do-more PLC and it shows how a start contact is used to activate the forward motor contactor. This start contact is controlled by a C-more HMI. The HMI will close the normally open start contact in rung one whenever the Start button on the HMI screen is pressed, and the motor will start in the forward direction.



In reality, most motor starter code is not that simple. There are a lot of other factors involved with starting a motor. The next image shows some of the possible permissive bits that can be used to start/stop a motor. You can see that a MOTOR_COASTING bit has been added. This bit will block the start command when the motor is still coasting to a stop. The MOTOR_STOP_BTN contact is added to stop the motor from starting when it is being requested to stop. A REVERSE_ACTIVE bit is added so the motor does not start in the forward direction when already started in reverse. The OVERLOAD_STATUS contact will stop the motor if an overload condition is detected. The CB_STATUS contact doesn’t allow the CPU to attempt a start when there is no power applied to the contactor. A “seal-in” or “latching” bit is also added below the rung to keep the motor running once the contactor is closed. The seal can be broken by any of the contacts to the right of the MOTOR_COASTING contact. Other possible additions include: an emergency stop pushbutton contact, a rung for the reversing contactor, a shunt trip contact, an alarm-on-start function, etc.



RUN MOTOR IN FORWARD DIRECTION IF:
Not Coasting, STOP Btn not pressed, Reverse not active, Overload OK, and power OK

Learn More..

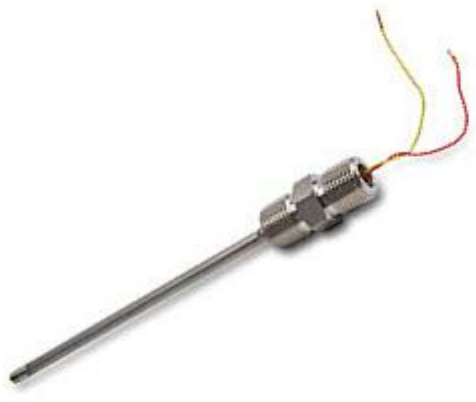
Learn more about AC motor control in this series of videos, from a simple switch to full reversing contactor configurations; selecting and using contactors and associated circuit breakers, overload protectors, and surge protectors; see typical hardware components and ladder logic programming for motor control with CLICK, Do-more and Productivity series PLCs.

A/C Motors (On/Off Control) Videos
AUTOMATION VIDEO COOKBOOK
A/C Motor On/Off Controls for Industrial Applications - U Can Do It
UCANDOIT
AUTOMATIONDIRECT
TECHTIPS
Tech Tip: A/C Motors for ON/OFF Control
Tech Tip: Selecting and Using Circuit Breakers for Industrial Automation Applications
Contactors - Overview
Tech Tip: How to Specify Contactors for Industrial Automation

Visit: www.AutomationDirect.com/Cookbook

Temperature Sensing

From curing ovens to milk pasteurization, temperature sensing and control has become an important part of automation. Two of the most commonly used temperature sensors in the industry are the Resistance Temperature Detector (RTD) and the thermocouple.



Resistance Temperature Detectors

RTDs are devices with an internal resistance that changes with temperature in a predictable, linear way. They are typically made from a very fine wire wrapped around a ceramic or glass core, or they can be created using thin film technology. Traditionally, platinum, copper, or nickel wire is used and is wound to achieve a specific resistance value. This resistance value will change as temperature does and by supplying a constant current, the measured voltage drop across the resistor can be used to determine the new resistance, and thus the temperature.

RTDs come in a variety of types, with the most common being a Pt100. It's made from platinum that has been calibrated to be 100 ohms at 0 degrees C. Platinum is an ideal metal for RTDs because of its stability, resistance to corrosion, and higher melting point. RTDs are great for applications up to around 600 degrees Fahrenheit. They provide accurate, repeatable results over the long term, but they are also fragile, have a long response time and are bigger than thermocouples.

Thermocouples

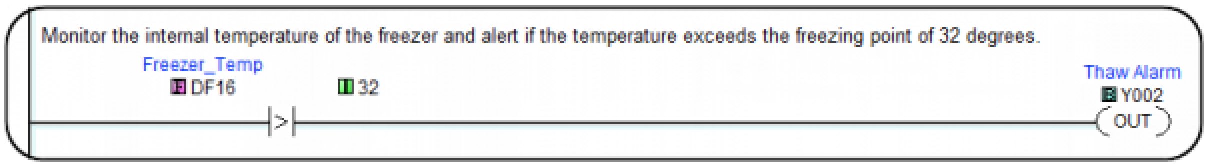
For wider temperature ranges up to a couple thousand degrees, a thermocouple is a better fit. Thermocouples work based on the Seebeck effect, a phenomenon where a small voltage is produced across a junction of two dissimilar metals when exposed to a temperature gradient. This is believed to be a result of heated electrons in the warmer metal traveling toward the colder area because of their thermal energy.

The amount of voltage produced within the thermocouple is very small, usually millivolts, and is directly related to the difference in temperature across the junction. There are numerous types of thermocouples and each is represented by a letter. The most common ones are J, K and T, which designate a specific temperature range and the wires of each type are color coded to identify them.

Most PLCs today have I/O modules designed to accept RTD or thermocouple signals directly. If using a temperature transmitter, I/O modules that accept 4-20 mA or 0-10V are often used to bring the signals into the controller.

Object Detection + Ladder Logic

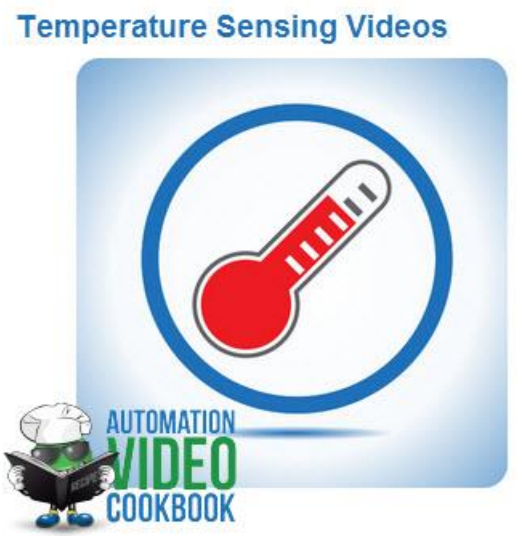
For temperature sensing using Ladder Logic, we can look back at an earlier example. Remember our freezer full of frozen fish from the Basic Instructions in Ladder Logic chapter? If not, no biggie, we were just using a temperature sensor to monitor the temperature of our fish freezer. To do so, we used a CLICK PLC and a compare contact. The temperature sensor's output produces an analog signal that is wired into the thermocouple module of the PLC. The CPU will convert this raw signal into degrees Fahrenheit and store it into DF16. Using the Compare contact instruction below, we compare our actual Freezer Temperature (DF16) to the constant 32, which represents 32 degrees Fahrenheit.

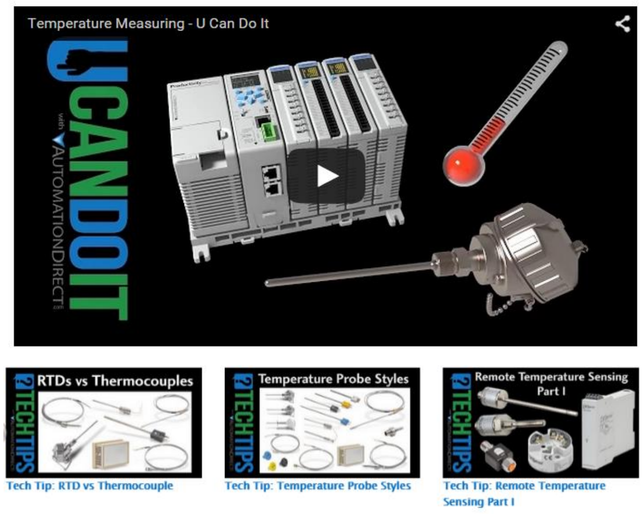

Monitor the internal temperature of the freezer and alert if the temperature exceeds the freezing point of 32 degrees.

An Out coil is then programmed from our Compare contact and will provide the signal that is wired to the external Alarm Horn. If the temperature of the freezer reads greater than 32 degrees, an alarm will sound to alert us that the frozen fish will soon thaw and spoil.

Learn More..

See this series of videos to learn about types of temperature sensors and how they work; hardware review of temperature probe styles and transmitters; how to select the right temperature sensor for your application; see typical ladder logic programming for temperature sensing with CLICK, Do-more and Productivity series PLCs.




Temperature Measuring - U Can Do It
UCANDOIT
AUTOMATIONDIRECT
RTDs vs Thermocouples
TECHTIPS
Tech Tip: RTD vs Thermocouple
Temperature Probe Styles
Tech Tip: Temperature Probe Styles
Remote Temperature Sensing Part I
Tech Tip: Remote Temperature Sensing Part I

Visit: www.AutomationDirect.com/Cookbook

Automation Cookbook

Automation Cookbook is a collection of short video chapters covering a wide variety of automation application topics. Each chapter starts with a “U can do it!” overview, which explains the capabilities and usefulness of a specific automation task. There is also a group of “Tech Tips” videos in each chapter with technical nuggets and the gotcha’s that can slow down experts and novices alike when specifying and implementing the topic in question. The practical “How To” videos show step-by-step demonstrations of exactly how to connect and troubleshoot, along with sample programming shown for the basics of the task. The Cookbook currently includes chapters on Temperature Sensing, Object Detection, A/C Motors (On/Off Control), and Distance Measuring.

What do you want to do?
AUTOMATION
VIDEO
COOKBOOK
RECIPE
Get up to speed quickly with these brief Automation Cookbook tutorial videos.
UCANDOIT
TECHTIPS
HOW TO...

Visit: www.AutomationDirect.com/Cookbook

Chapter 6

Practical PLC Topics

Now that we have covered the basics, let’s take a look at some of the more complex operations of a PLC system. In the next several chapters we will discuss some of the most widely-used, yet often misunderstood topics in industrial automation. First, we take a deeper look into PID control and the method behind the madness. PID control is essential in process control industries, such as oil and gas, and in this chapter we break down the mathematical method used in this type of control.

PLC communication is a vital component of any PLC system, no matter the industry, and it can also be one of the most confusing topics. In the PLC Communication chapter, we delve deeper into protocols and networks that PLCs use to communicate. Also included is a chapter on EtherNet/IP messaging, which is becoming one of the more popular protocols in industrial automation.

Lastly, this section will explore the motion control functionality of PLCs. Servo/Stepper systems are important when precise positioning or movement is required, and PLCs can supply these systems with the needed pulse and direction signals, discrete signals (to an indexing drive) or fieldbus communication.

This Chapter Includes:

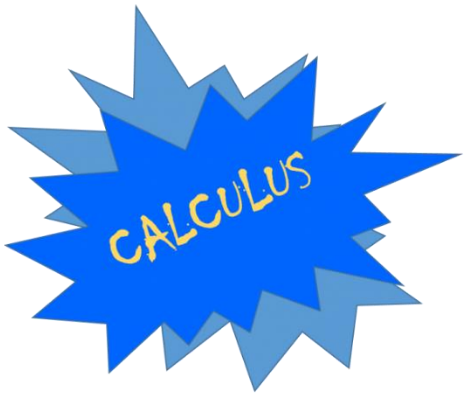
Chapter 6 - 1

Methods Behind PID Loop Control

Lock your doors, board up the windows, and prepare yourself for a deeper look into **PID loops**! Now, if you are still here, great! If you ran out of the room screaming like I did, you can catch up later. For some, PID loops can be scary. After all, they are pretty complex and involve CALCULUS. Hold on, I think I'm gonna turn on all the lights real quick… ok, that's better. PID loops shouldn't be something to fear, they can be confusing but they are not impossible.

First off, for the newbies, PID is an acronym for Proportional, Integral and Derivative. It is a control function that is frequently used in process control. You take a reference signal in from your process, compare it to your setpoint and then change the output to the control device accordingly to minimize the error. Think of the cruise control in your car. You set the speed you want to travel at, your car's computer will compare that to the current speed and either accelerate or decelerate to match that setpoint and remove the error. The bigger the error, the bigger the gain or response. For example, when your car on cruise control revs the engine when you start up a hill, the error between the setpoint and the current speed increased so much that the response was also greatly increased.

WARNING: Readers with sensitivity to math equations, including integral-induced headaches, differential skin rashes or anyone who has had prior calculus-related trauma of any kind, should avert their eyes and not proceed any further.



To help us better understand how a PID loop works, let's take a closer look at the PID equation:

$$M_n = K_c * e_n + Ki * \sum_{i=1}^{n} e_i + Kr * (e_n - e_{n-1}) + Mo$$

Stay with me, stay with me, it's not as bad as it looks. We'll take it in pieces:

M_n represents the control output at the moment of time "n". This is the amount of gain or response sent to the control device.

e_n is the calculated error at the moment of time "n". It is calculated by subtracting the current value from the set point.

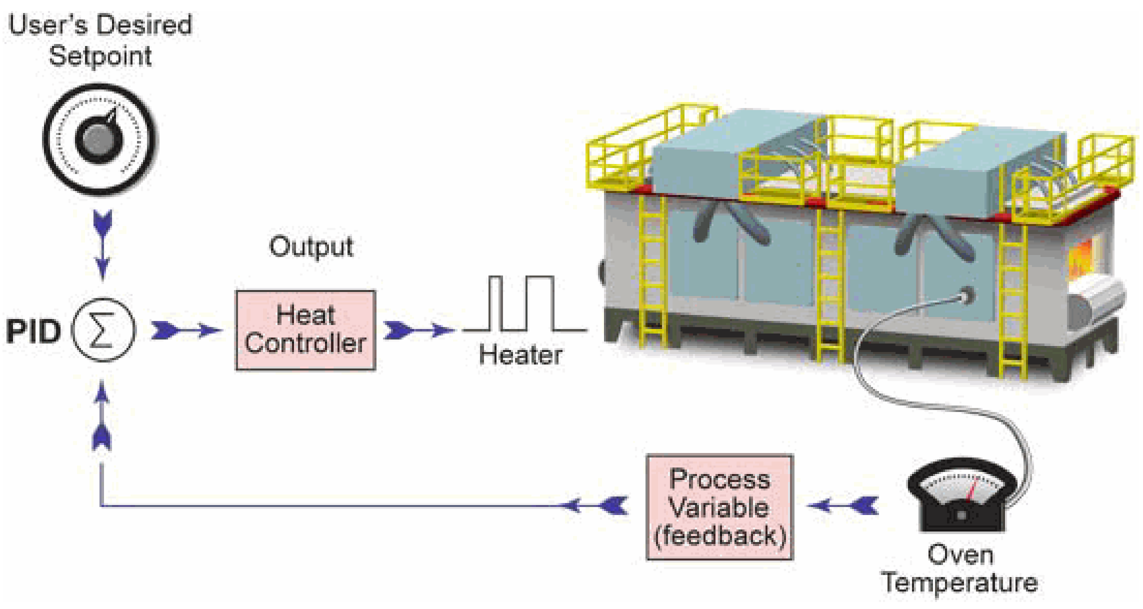
$K_c * e_n$ is the Proportional Term (P) of the PID equation. K_c is the proportional gain coefficient and is a fixed value.

$Ki * \sum_{i=1}^{n} e_i$ is the Integral Term (I) of the PID equation and the most frightening. It is simply the sum of the calculated errors from the first sample (i=1) to the current moment "n". This sum is then multiplied by the Integral coefficient K_i which is calculated using the formula: $K_i = K_c *$ (sample rate/integral time). Basically, the integral coefficient is the proportional coefficient times the ratio of sample rate to integral time. It will make sense once we start plugging in values….I hope.

$K_r * (e_n - e_{n-1})$ is the Derivative Term (D) of the PID equation. All you do here is subtract the error found at the previous sample (en-1) from the error now (en) and multiply it by the calculated derivative coefficient. The derivative coefficient is calculated using the formula: Kr = Kc * (derivative time/sample rate). Mathematically, the derivative term will look at the difference between the error now and the error before, determine how rapidly the process variable is deviating from or correcting to the setpoint and adjust the output accordingly. Simple, right?

M_o represents the initial control output when the loop is activated. If transitioning from a manual operation, this value is transferred when the loop transitions to auto mode.

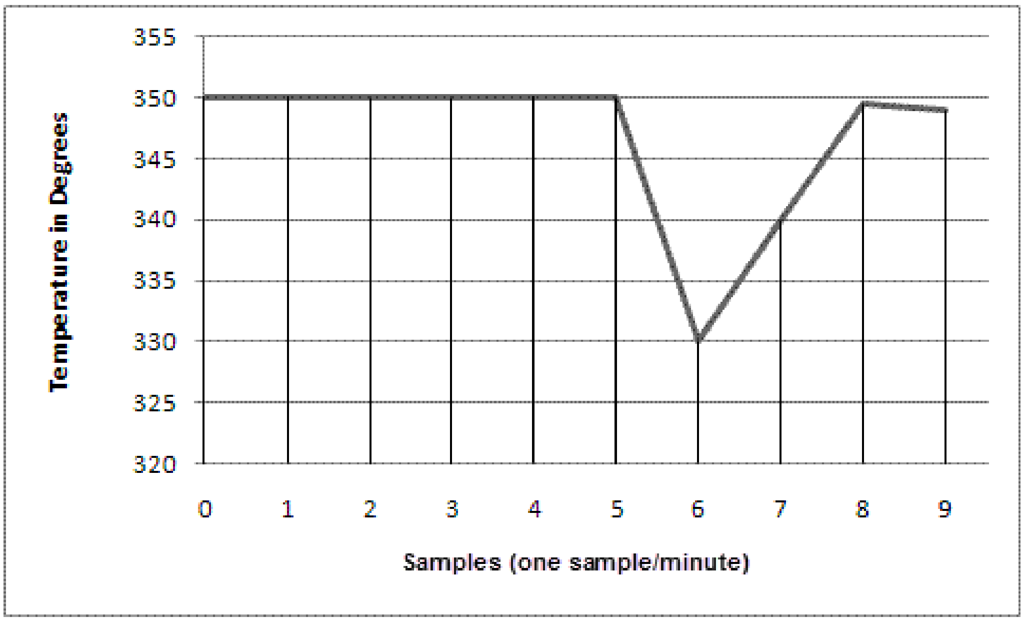
So, now that you know all of the pieces and what they do, you need to know how they fit together. The best way to do that is with an example. So, let's apply the equation to a temperature control loop. In our example, the loop is maintaining the temperature of an oven, as seen below, and the desired temperature of the oven is 350° F.



For this particular application, we will set the equation's parameters as follows:

- **Kr** = 0 (To achieve this, set the Derivative Time to zero. This will make the loop a PI controller; we will look at the derivative reaction later.)
- **Kc** = 3 (Proportional Gain)
- **Ts** = 60 seconds (Sample Rate)
- **Ki** = 1 (To achieve this, set the Integral Time to 180 seconds, Ki = Kc * (sample rate/integral time) or **Ki** = 3*60/180 = 1)
- **M(0)** = 30 (Initial Control Output)

The following graph shows the temperature fluctuation over the past 9 minutes. Let's see how the equation operates.



At sample zero, the loop has an initial control output (M(0)) of 30. Looking at the graph we can see that with this control output the loop has been stable at 350° F for the first five samples. After the fifth sample, something occurs to make the temperature drop 20° F. This is seen in our actual temperature reading at sample six. Therefore, the error in the system at sample six is calculated to be: e(6) = setpoint – current value = 350 – 330 = 20. Also, at this point we can easily calculate the sum of all the errors for the system:

$$\sum_{i=1}^{n} e_i = (e(1) + e(2) + e(3) + e(4) + e(5) + e(6)) = (0+0+0+0+0+20) = 20.$$

So using the parameters and calculated values above, the equation for the loop at sample six would become: (colors represent proportional, integral and derivative terms)

$$M_n = K_c * e_n + Ki * \sum_{i=1}^{n} e_i + Kr * (e_n - e_{n-1}) + Mo$$

M(6) = (3 * 20) + (1 * 20) + (0 * (20-0)) + 30
M(6) = 60 + 20 + 0 + 30
M(6) = 110

At this point, there is an increase in the control output of 80 from the initial control output and the proportional term is the main force driving this increase with a value of 60. This control output will be converted to an analog value (0-10V, 4-20mA, etc.) that is fed to an end device. In our example, that device is the heating element inside an industrial oven. As the control output drives the temperature toward the setpoint, we see, at sample seven, that the error has decreased to: e(7) = SP-CV = 350 – 340 = 10. Also at this point, the sum of all the sample errors will equal 30 ((e(1) + e(2) + e(3) + e(4) + e(5) + e(6) + e(7)) = (0+0+0+0+0+20+10) = 30) and the equation will become:

M(7) = (3 * 10) + (1 * 30) + (0 * (10-20)) + 30
M(7) = 30 + 30 + 0 + 30
M(7) = 90

Here we see that as the error is corrected, the control output will decrease due to the drop in the proportional term. However, notice that the integral term has risen slightly. At sample eight, the oven temperature has recovered to 349.5° F leaving a minimal error of: e(8) = SP-CV = 350 – 349.5 = 0.5. The sum of all sample errors is now 30.5 and the control output equals:

M(8) = (3 * 0.5) + (1 * 30.5) + (0*(0.5-10)) + 30
M(8) = 1.5 + 30.5 + 0 + 30
M(8) = 61.5

At this point, the proportional term has dropped off considerably and the integral bias has exceeded the proportional gain in controlling the output. You can see that across the three samples the proportional term has gone from 60 to 30 to 1.5, as the error decreases while the integral term has gone from 20 to 30 to 30.5. When tuning a loop, it is important to remember that with large errors the proportional term will drive the output and with small errors the integral term will take control.

Wow, you've made it this far and I am proud of you! For your reward we will now look at the derivative term and how it affects the equation. Wait, where you going?…The job of the derivative term is to keep stability in the loop. With our example, the rate of change is very rapid at sample six – a 20° F drop within one minute. With very rapid changes, there is a risk of instability and the derivative term attempts to alleviate this while aiding in the correction. As mentioned previously, the derivative term will look at the difference between the error now and the error before, determine how rapidly the process variable is deviating from or correcting to the setpoint and adjust the output accordingly.

If we add a derivative coefficient of Kr = 1 (set the Derivative Time (Td) to 20 seconds, Kr = Kc * (derivative time/sample rate) or Kr = 3*20/60 = 1) to the samples in our discussion they become:

M(6) = (3 * 20) + (1 * 20) + (1 * (20-0)) + 30
M(6) = 60 + 20 + 20 + 30
M(6) = 130

M(7) = (3 * 10) + (1 * 30) + (1 * (10-20)) + 30
M(7) = 30 + 30 – 10 + 30
M(7) = 80

M(8) = (3 * 0.5) + (1 * 30.5) + (1 * (0.5-10)) + 30
M(8) = 1.5 + 30.5 – 9.5 + 30
M(8) = 52.5

Looking at the samples above you can see that after our dramatic temperature drop at sample six, the derivative term starts to lessen the control output at sample seven, going from a +20 gain in sample M(6) to a -10 in sample M(7). At this point, the derivative term determined that the process variable was approaching the setpoint at a very rapid rate and dampened the control output to avoid an overshoot of the setpoint.

Well, there you have it, the basics of PID control. PID loops are very important in industrial automation and aren't something you should be intimidated by. Yes they are complex, and yes they use calculus, but they really aren't too bad. The built-in function blocks in some PLC ladder logic instruction sets can do the heavy calculation lifting for you; the most important tasks then are choosing the coefficient values and tuning the loop for maximum control and performance. Hopefully knowing what each term means and how they affect the equation will help you understand how to set them up. If you would like to see some informative videos on how to use PID loops, check out our video library.

Learn More… Watch The Video!

Productivity3000 PID Loop Programming 11Part Series

Visit: http://N2ADC.com/kel3w

Chapter 6 - 2

PLC Communications – Coming of Age

PLC communications has grown and changed over the many years since the controller's inception decades ago. Industrial needs and technological advances have kept PLC communications evolving, and this post will explore some past and present communication standards, and what the future may hold.

PLC Partners in Communications

PLCs need to communicate to a number of different devices. First, there are devices which are part and parcel of the PLC, such as remote I/O. Next, are other PLCs and control devices, such as motor drives and servo controllers. Communication with some type of operator interface device is also often required, as is Internet connectivity. Finally, PLCs often communicate to server level PCs running various manufacturing related applications.

All of these communications require two things:

- a physical connection or layer, which is the wiring and connection components, and
- a shared protocol, which is the common language allowing each device to understand what the bits and bytes in the communication messages mean.

Back in its infancy, PLC communications were typically proprietary, with each supplier having their own closed connections and protocol. This made communications relatively simple among a single supplier's products, but devilishly complex if you chose to venture outside the realm.

Proprietary Has Its Place

These old, proprietary communication methods still have their place, as they allow quick and easy connections among a single supplier's products. For simple systems with no plans to expand or connect to other components, proprietary communication is an option and will continue to be for quite some time.

Many PLC suppliers still support these older physical, wired connections—most often based on RS-232C, RS-422 and RS-485 connectivity—and the related proprietary communication protocols.

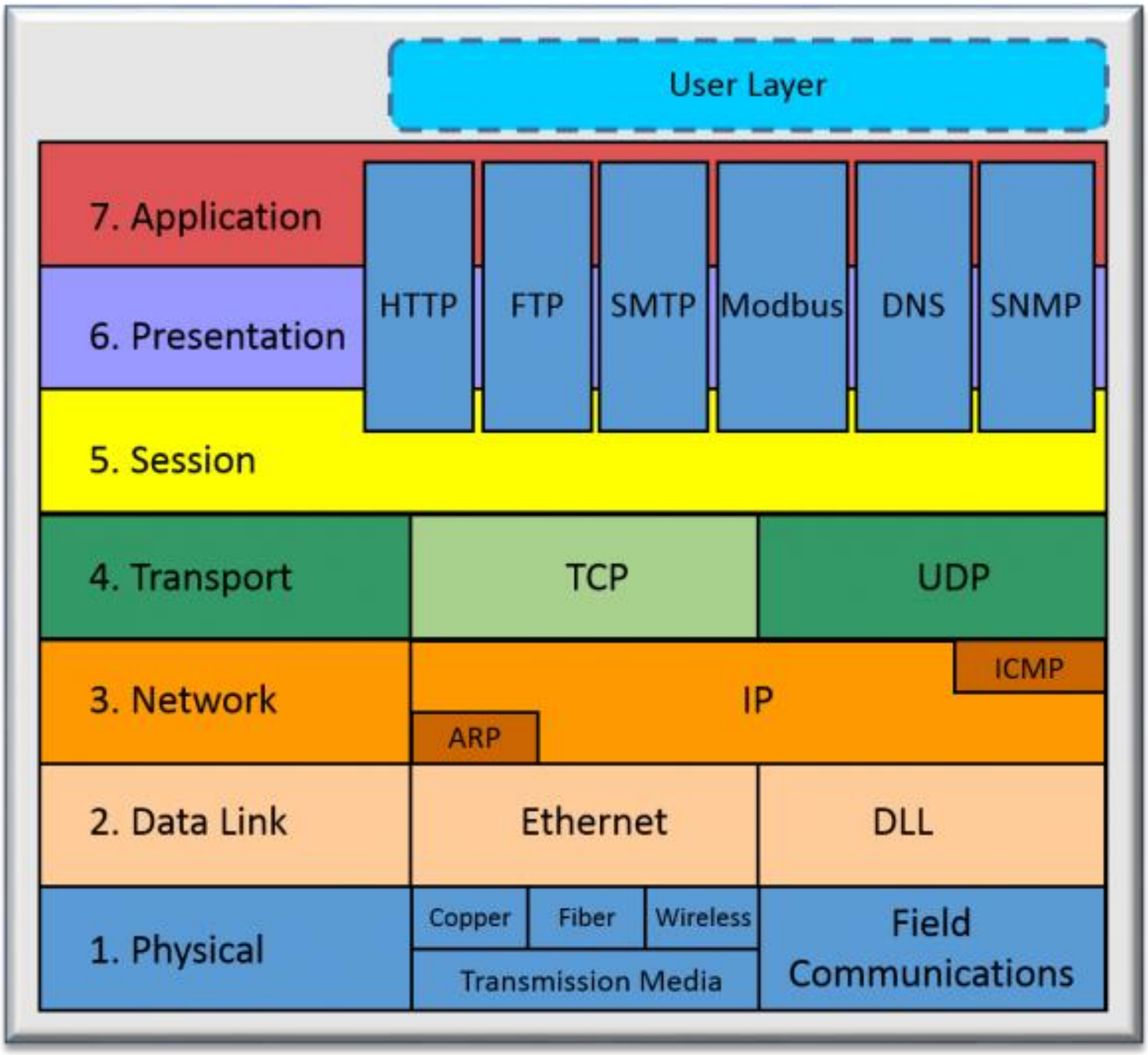
For example, some AutomationDirect PLCs use a serial RS-232C or RS-422 physical layer network with the DirectNET protocol for communications. This protocol can be used for PLC-to-PLC communication over a point-to-point or multipoint network using standard PLC instructions, or to talk to AutomationDirect's C-more HMIs.

When the control system requires venturing outside a single supplier, communication standards come to the forefront, starting at the physical layer.

Physical Layer is both Mechanical and Electrical

Many times the physical layer is confused with the physical medium such as the cable, connectors, network interface cards and wireless transmission hardware. It's not physical in that sense, although the physical layer does define the required mechanical and electrical interface to the physical medium.

The physical layer defines how to connect the upper data link layer in the Open Systems Interconnection (OSI) communications model within a computer to physical devices. It is basically the hardware requirements with schematics and specifications for successful bit-level communication to different devices. The physical layer not only defines items such as bit rates, transmission electrical, light or radio signals, and flow control in asynchronous serial communication, it also defines types of cables and the shape of connectors.



OSI model builds a protocol in layers

Once the physical layer's hardware requirements are defined, some examples of protocols that can reside within different physical layers include:

- Proprietary protocol for a barcode reader using an RS-232 point-to-point connection
- Motor drives protocol connecting multiple devices on an RS-422/485 multi-drop connection
- Printer protocol over a parallel interface
- Ethernet for connection to a plant or control network
- Universal Serial Bus (USB) for connection to keyboard
- Bluetooth for connection to a wireless microphone

RS-232, RS-422/485 and parallel interfaces have been around for decades and are still in widespread use. But Ethernet, USB and Bluetooth are the physical layer protocols most commonly specified in new products and applications, and represent the future direction of PLC communications.

The physical media and connections are affected by the type of physical layer chosen as it defines the cable type, topology, number of devices, maximum transmission speed, and maximum cable distance. For additional information, be sure to check out this application note on Ethernet Fundamentals and this informative article on Demystifying Network Communications.

Protocols Ride on the Physical Layer

A protocol is a set of rules for communication among networked devices. Some common protocols used in the industrial arena include:

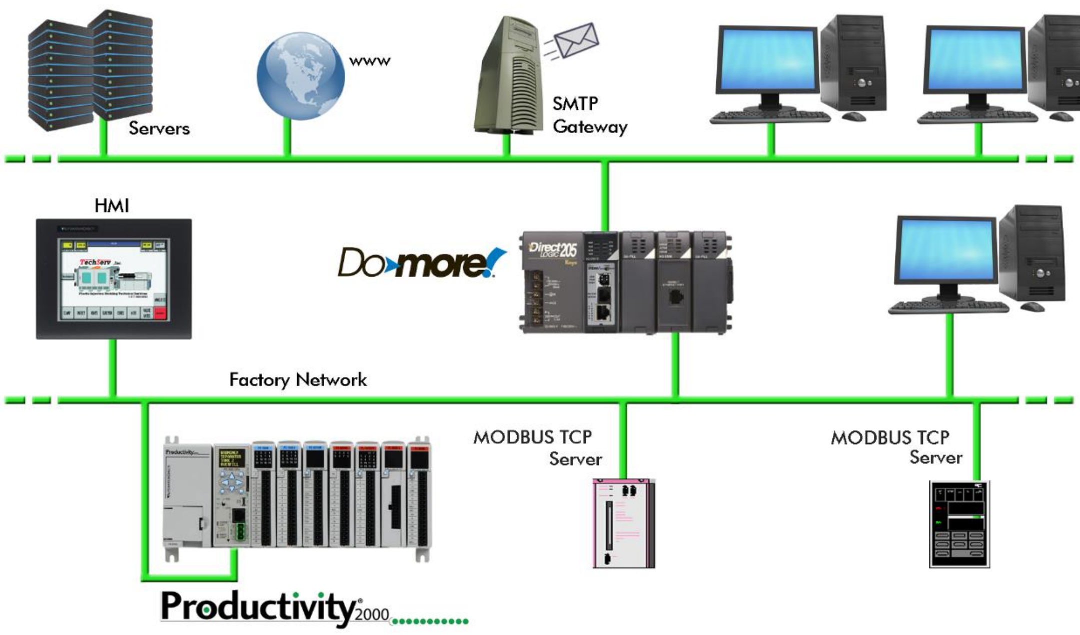
- Modbus RTU
- EtherNet/IP
- Ethernet TCP/IP
- Modbus TCP/IP
- Profinet

Perhaps the most common industrial serial communication protocol is Modbus RTU, developed by Modicon, which usually runs on an RS-485 network. This and other popular serial protocols are supported by a wide variety of suppliers, and are very familiar to a wide group of automation professionals. But performance is limited, making serial protocols a poor choice for high-speed and other demanding applications.

Because of performance and other advantages, Ethernet has emerged as the dominant standard for the physical layer of many industrial protocols such as EtherNet/IP, Ethernet TCP/IP, Modbus TCP/IP and Profinet. And unlike serial protocols, multiple Ethernet protocols can run on the same Ethernet physical layer.

Using Ethernet, it's not too difficult to interconnect several devices such as PLCs, HMIs, field I/O and valve banks. Plus, the communication remains fast while talking to several dissimilar devices on the same cable, due to the very high speed of Ethernet as compared to older serial networks.

Here's an example of a typical Ethernet network connecting a wide variety of control systems and other components:



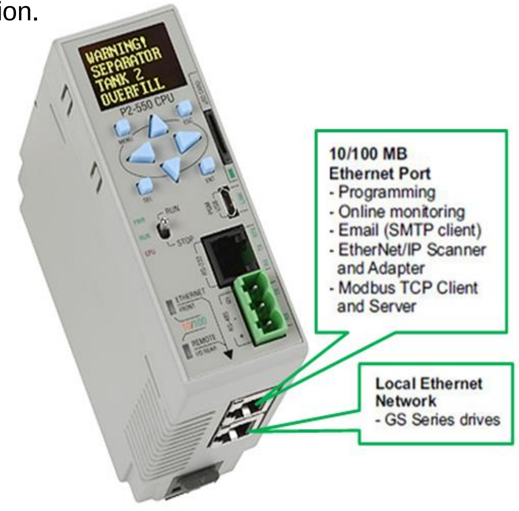
Ethernet Supports Multiple Protocols

Today's physical, wired layer is moving to Ethernet for most control system communications. Wiring, connectors and managed or unmanaged switches are now standard—with a wide variety of devices capable of connecting to the Ethernet hardware and industry-standard networking technologies.

Most users just need to interact with the physical Ethernet layer, plugging in the cables and letting the protocols worry about error-free communication.

Popular industrial Ethernet protocols like EtherNet/IP and a common network infrastructure allow consistent data and communication from the HMI and desktop PCs at the information level—through the PLCs, HMIs, servo controllers and drives at the control level—to the device level distributed I/O and field components.

And, of course, Ethernet is the portal to the Internet which allows a PLC to communicate worldwide. Some PLCs have built-in web server capability, allowing access to it from any web browser.



The prevalence of Ethernet communication in industrial automation could result in even more standardized communication protocols in the near future. But for now, if your system is simple, a proprietary protocol may be the best solution. When requirements grow in terms of speed, connectivity or expansion, an Ethernet protocol may be the way to go.

For more information on what to consider when considering Ethernet, please see this article Industrial Ethernet or Fieldbus Network on our website.

Chapter 6 - 3

EtherNet/IP: Implicit vs. Explicit Messaging

EtherNet/IP is younger than you and me, but still approaching maturity as it just reached its 15th birthday. Over the years, it has grown into an impressive industrial Ethernet communication network for connecting automation equipment, processes and machines. Keys to its success are the network and transport layers where explicit and implicit messaging takes place; here's how to use these features to optimize your EtherNet/IP networks.

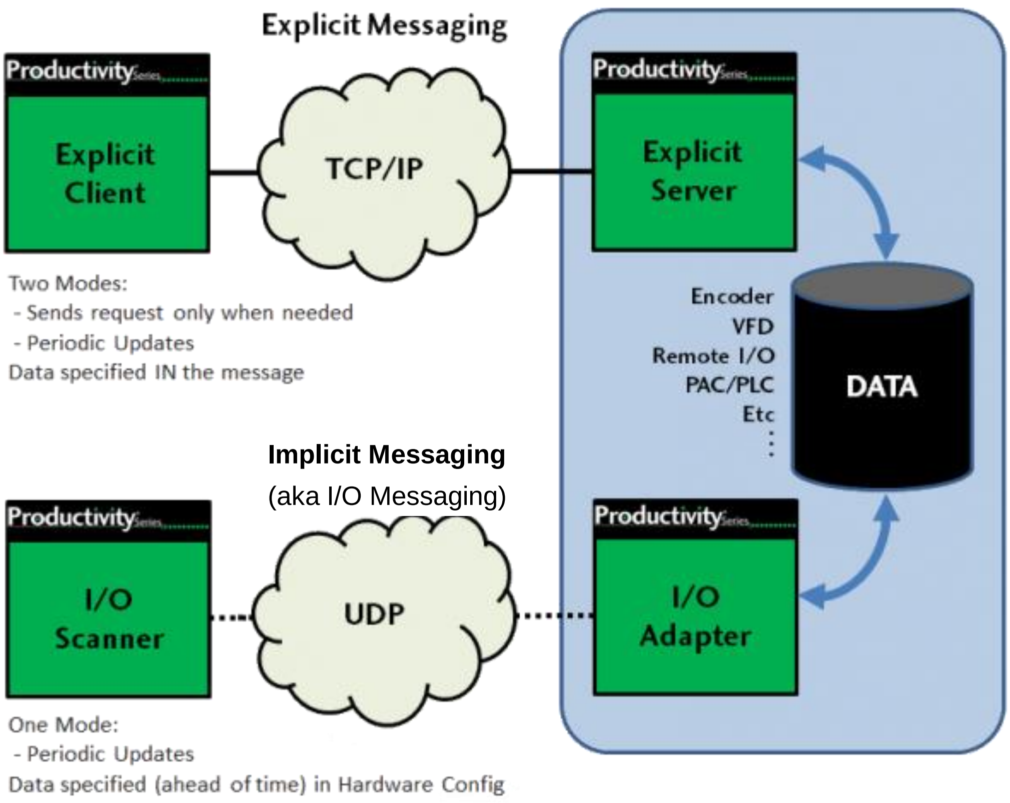
Please refer to the Table and Diagram below for clarification as we discuss explicit and implicit messaging.

Table: Explicit versus Implicit Messaging Cheat Sheet

EtherNet/IP Connection or Feature	Explicit Messaging – Requests and defines information	Implicit Messaging – Data only, no protocol info is included (aka I/O Messaging)
Originator (master)	Client (controller)	I/O Scanner (controller)
Target (slave)	Server (field device)	I/O Adapter (field device)
Form of messaging	Unconnected but can be connected	Connected
Typical use	Diagnostic/event/configuration data	Real-time control data

With explicit messaging, the controller is called the client and the field devices are called servers. With implicit messaging, the controller is called the I/O Scanner and the field devices are called I/O Adapters.

It is important to select a controller that will support explicit messaging as a client or server, and implicit (real I/O) messaging as a scanner or adapter.



Which Messaging Mode is Best For You?

The selection of explicit or implicit messaging often depends on the choice of field devices, as each may only support only one messaging mode. A controller, on the other hand, generally supports both modes as a client, server, scanner or adapter.

If your application requires large amounts of data, explicit messaging is the preferred choice because bandwidth is saved, as data is only requested when necessary.

For high speed, real-time applications, implicit messaging is the way to go. Think of the "i" in implicit as the "I" in I/O messaging, which of course requires high-speed as I/O is generally used for real-time control.

Explicit messaging requires programming in the controller for setup. You need to request the data, add handshaking, acknowledge the data, and move the data where it's needed in the controller.

Real-time implicit messaging can be quickly configured. Just set the controller up as a scanner to receive data, and configure the connection to a remote EtherNet/IP device by filling in the blanks in the controller programming software. This configuration is basically just defining what the data is and where it will be in the controller data table. Once configured, the data just shows up in the controller memory with no handshaking required.

For outgoing commands from the controller to field devices, just write to local memory and the data is automatically sent to the device. The data is transferred at the rate you specify, typically in the 5ms to 20ms range.

Explicit Messaging Details

Within EtherNet/IP, the explicit message connection can be thought of as a client/server relationship. The client, such as the PLC controller, asks or requests the information from a server, such as a VFD field device, and the server sends the requested info back to the controller.

Because the client requests the information from the server via TCP/IP services, the request has all the information needed to respond explicitly to the message. The Client basically says, "Server, I need this information, formatted exactly as specified in this message, please send it." The server responds with a message containing the formatted information, perhaps confirmation that the VFD speed set point has changed, to let the controller know everything is OK.

This configuring and monitoring ability, common to explicit messaging, works well for non-real time messaging as the client (controller) can send a message request anytime, and the server (field device) can respond when it's available. Explicit messaging is typically used for client/server communication which isn't time critical.

Implicit Messaging Details

EtherNet/IP uses implicit messaging, sometimes called I/O messaging, for time-critical applications such as real-time control. Implicit messaging is often referred to as I/O messaging because it's frequently used for communication between a controller and remote I/O. It's a much more efficient communication connection than explicit messaging as both the client and server ends are pre-configured to know implicitly or exactly what to expect in terms of communication.

Real-time, implicit messaging basically copies data with minimal additional information built into the message. Each end of the communication link doesn't need to be told much about what the information is for, as the data has been predefined. The meaning of the data is "implied" or "implicit" so there is no extra baggage because both ends already know exactly what each bit and byte mean.

Making the Connection

There are two forms of messaging connections used with EtherNet/IP: unconnected and connected. Unconnected messaging is typically used for explicit messages. Connected messaging uses features built into each device and configured in advance for real-time I/O messaging, and it's commonly used with implicit messaging.

Not all controllers with EtherNet/IP support both explicit and implicit messaging. Be sure to verify that messaging capabilities of your controller match those required by your application.

Learn More… Watch The Video!

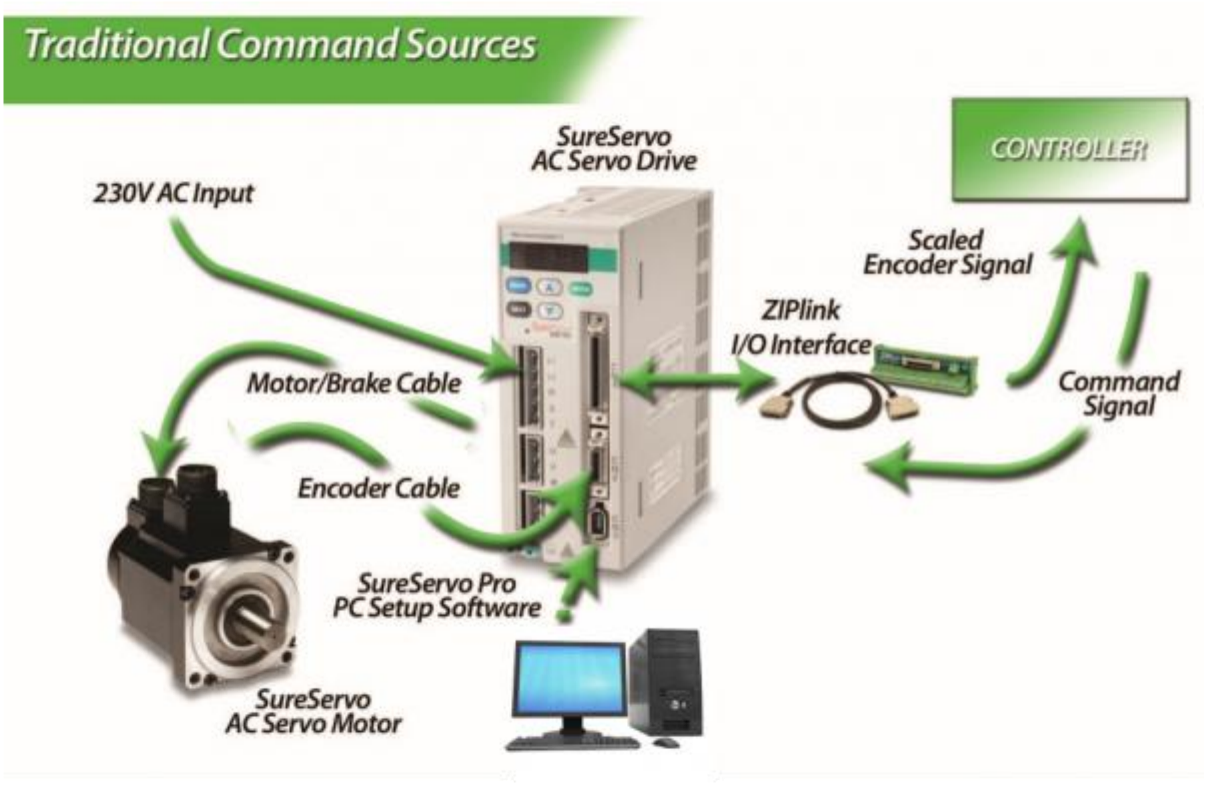
Productivity Series – EtherNet/IP - Overview

Visit: http://n2adc.com/aedkg

Chapter 6 - 4

Motion Control Explained

Motion control is generally understood to mean the use of servo and/or stepper systems as the "muscle" to move a given load. These motion control systems are capable of extremely precise speed, position, and torque control. Applications which require positioning of product, synchronization of separate elements, or rapid start/stop motion are all perfect candidates for the use of motion control. PLCs are very capable of providing the signals required to command these servo and stepper systems in a cost-effective and digital (noise-free) manner.



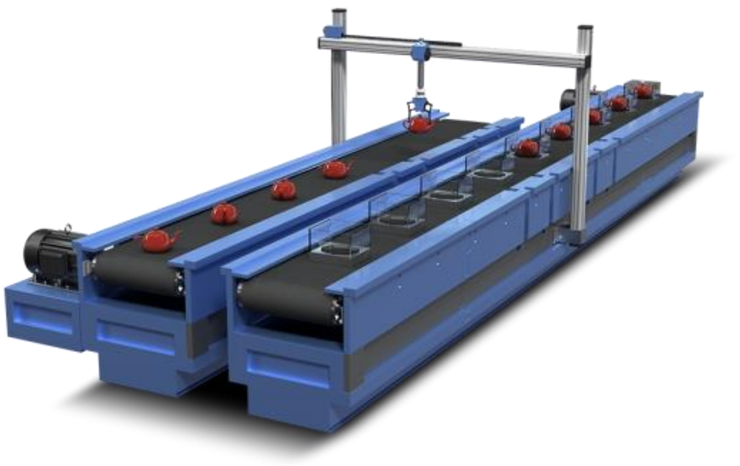
In a typical motion control system there are three basic components: the controller, the drive (sometimes referred to as an amplifier), and the motor. The path planning or trajectory calculations are performed in the controller, which sends low-voltage command signals to the drive, which in turn applies the necessary voltage and current to the motor, resulting in the desired motion. Sometimes feedback devices on the motor or the load are used to notify the drive or the controller with specific details about the actual movement of the motor shaft or the load. This feedback data is used to increase the accuracy of the motion, and can be used to compensate for dynamic changes that may occur at the load, such as changes in mass, friction or other disturbances. Servo systems operate in a closed-loop fashion and vary output torque to move into/stay at the commanded position, while most stepper systems typically provide open-loop position control (a stepper will drive at full force to get to the commanded position or fail trying).

The choice of open-loop versus closed-loop control depends on many factors and both are useful methods for controlling motion. PLC-based controllers can be used for either type of system. Applications that can be accomplished with a low-cost PLC and servo/stepper components include cut-to-length, indexing tables or conveyors, and x/y tables (plotter/cutter/router/placer).

The classic ‘pulse and direction’ signals that are widely used with PLCs provide an inexpensive, noise-free (digital) method for precision motion control. Extensions or function blocks within the PLC ladder logic are typically used for programming and are easy for factory personnel to understand and maintain. While typically limited to a few axes of control and where coordination between axes is limited, PLC controllers with pulse and direction capability are an excellent fit for many motion applications. Often, low-cost PLCs are already being used for logic control on the machinery and can also handle the motion tasks with the addition of a pulse output card and some additional programming. This can eliminate the need to integrate the logic controller with a separate motion controller. Machine builders can also save considerable time when implementing PLC-based systems, especially if they are already familiar with the PLC and its programming software.

In a typical PLC-based motion control system, high-speed pulse output cards are used in the PLC to generate a ‘pulse train’ for each servo or stepper drive. The drive receives the pulses and indexes the motor shaft by a pre-set amount for each pulse. Typical stepper systems might index 1/200 of a revolution per pulse, while micro-stepping or servo systems might be configured for as little as 1/10,000 of a revolution for each incoming pulse. The amount of motion dictated by a single pulse can be adjusted in the drive to obtain a desired resolution, or to achieve a required top speed (as a resolution/top speed trade-off may be required to accommodate the maximum pulse output frequency from some PLCs). A separate signal is used to determine the direction of travel. This control method is referred to as ‘step and direction’. A similar but functionally equivalent method, ‘clockwise/counterclockwise’, uses a separate pulse train for each direction of travel. This method is somewhat less popular, but has advantages in some applications. Electronic gearing can usually be enabled in the drive to allow high-resolution moves at low speeds, as well as a high-speed mode for faster moves with lower resolution.

Encoder feedback, when used in these types of systems, is normally handled at the drive level. Two simple hardwired signals from the drive back to the PLC, drive fault and in-position, are often used to notify the PLC of exceptions and/or completion of each move.



Evolution of Electric Motion Control Technology

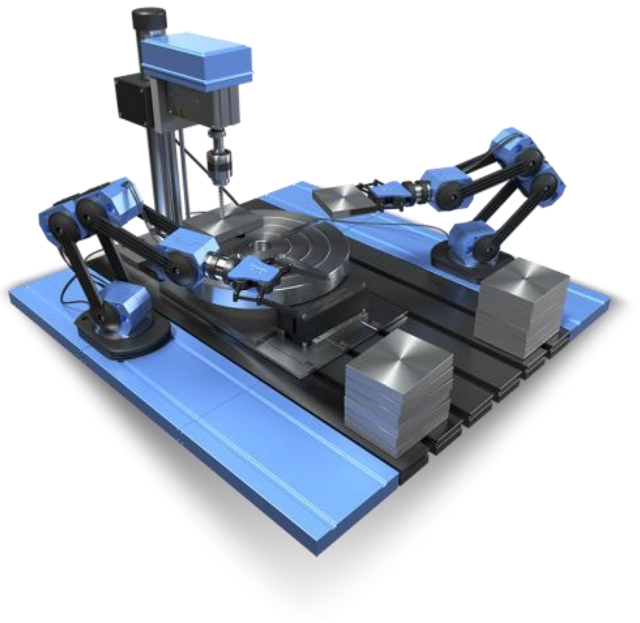
Electric motion control systems originated as alternatives to hydraulic motion systems. With most electric systems ranging in size from a few watts into the kilowatt range, electric motion control has become prevalent at these smaller sizes, while hydraulic systems continue to dominate the larger applications approximately 5kW and above, where electric power is impractical or unavailable and where the environment is harsh or extreme. The electric systems are more factory-friendly, less obtrusive, and are easier to install than the plumbing required by hydraulics. Early electric servo systems were usually operated in velocity or torque mode, accepted analog command signals, and were quite successful despite problems with electrical noise and drift. Early PLC-based controllers used analog output cards to provide the velocity or torque command signals.

While analog control signals are still used on some systems, most modern motion systems have migrated to some form of digital control. The advent of the digital servo drive, with the ability to close the position loop, was another major step forward. New types of signals between the controller and drive are now required to send position commands to these digital servo drives.

The three most common control signals used by today's PLC-based motion controllers are the pulse and direction signals detailed earlier, discrete signals to an intelligent or indexing drive, and fieldbus communications.

The pulse and direction interface that was originally developed for stepper systems has now become a standard feature on most servo products as well. A PLC with a high-speed output is one of the most cost-effective methods for controlling motion today. No intelligence is required in the drive and all programming is performed in the PLC. Almost all PLCs available from AutomationDirect offer some form of high-speed pulse output. For example, the smallest and least expensive DirectLOGIC® PLC, the DL05, includes a single 7kHz high-speed output which can be used for limited motion control applications. The DL05 will also accept an optional module which provides an additional high-speed output channel at up to 250kHz. The AutomationDirect SureStep stepping system can easily be controlled by this or newer PLCs such as the Productivity2000. In addition to the benefit of the low price for stepper components, all motion and logic instructions are programmed in the PLC's software for significant time savings.

Indexing drives offer another two options for PLC-based control. Indexing drives include a standalone controller which is built in to the servo or stepper drive. These single-axis devices have I/O capability, and can execute motion profiles based on a single PLC or real world input. This type of drive often includes a fieldbus connection and can perform moves based on commands and parameters received across such a connection.



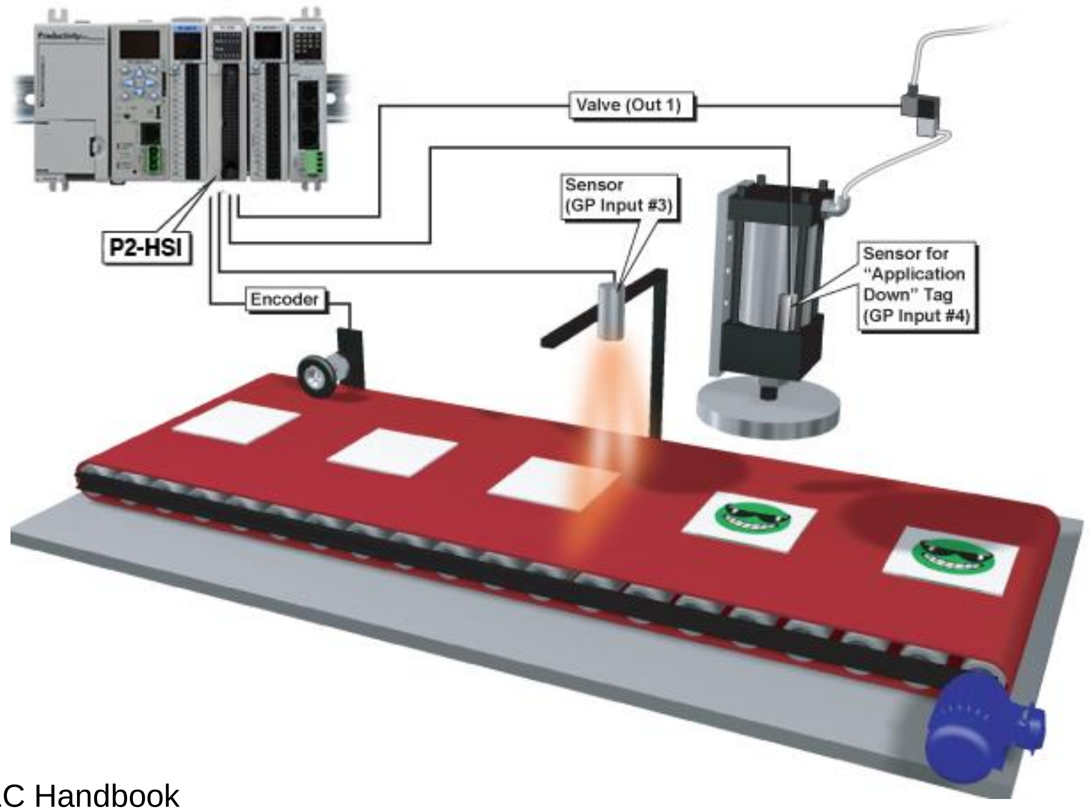
For example, the SureServo® line of servo products from AutomationDirect are indexing drives. The SureServo drives can be pre-programmed with parameters for up to 8 separate motion profiles, which can then be initiated via discrete inputs from a PLC or signals from other devices, even pushbuttons. These moves can consist of precise, user defined accelerations (ramps) to preset speeds, with accurate decelerations carefully timed to end at predefined positions. Dynamic velocities (with controlled ramps) and precise application of torque are also possible. Parameters for vibration suppression (notch filters), load inertia range, proportional and integral gain, and many others can also be customized for specific applications in the SureServo drive. The SureServo offers two adaptive auto-tuning modes, which continue to tune the system while it operates.

The AutomationDirect SureStep® line of stepper products also includes drive models with built-in indexers that offer similar capabilities.

The SureServo products also have a built-in Modbus interface. Modbus-enabled PLCs can initiate moves and download parameters to the SureServo drive across the Modbus link. The Modbus link can also supply information back to the controller about the performance and status of the servo motor and drive system. Multiple SureServo systems can be controlled via a single Modbus port on the PLC. The SureServo's ability to download custom motion profiles from a PLC on the fly, and execute these moves on command, allows the ultimate in flexibility and control with a PLC-based motion controller.

Trends in PLC-based Motion Control

Motion control modules for PLCs are increasing their capabilities year after year. High-speed output modules offered by AutomationDirect now have output frequencies up to 1 MHz. The Productivity2000™ high speed output module provides two 1MHz output channels that can allow extremely high resolution settings in the drives while also commanding very high (if not "top") motor speeds when required. These new high-speed output cards are designed to be low-cost "mini-coprocessors", capable of executing logic asynchronous to the PLC scan.

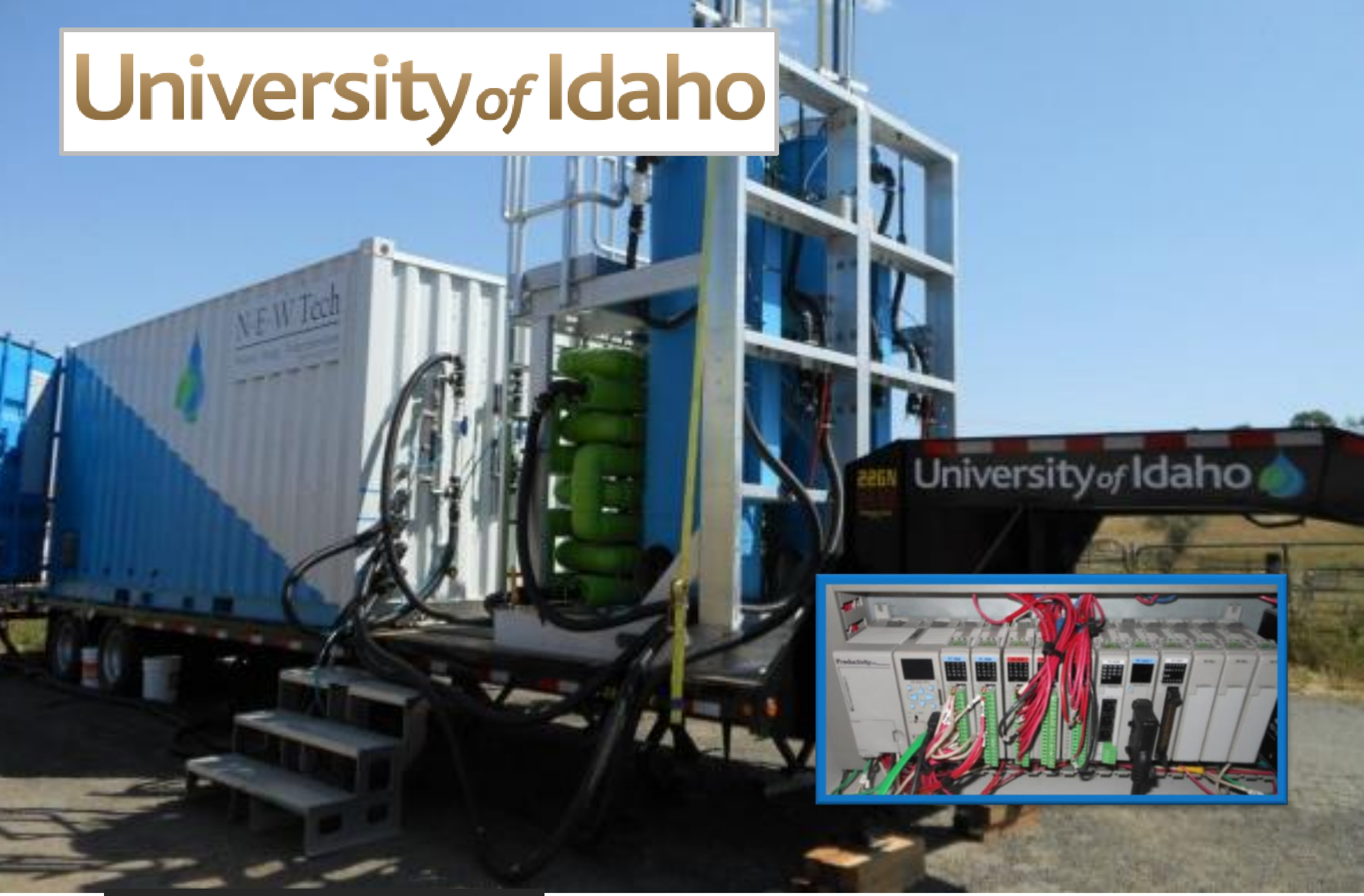


Software is also playing a bigger role in the evolution of PLC specialty modules for both discrete and hybrid control applications. Advancements in software that provide for intuitive graphical configuration are eliminating the need for large amounts of setup ladder logic previously needed in the beginning of the user's PLC program. Microsoft Windows-based point-and-click configuration utilities are becoming more common and provide substantial time savings for programmers and maintenance personnel. With these advancements in PLC technology, PLCs will continue to meet the needs of machine control applications for many years.

Chapter 7

Collection of PLC Application Stories

Want some examples of what people are doing with PLCs? You're in luck, we have a full collection of Application stories on our website at Library.AutomationDirect.com/applications. For your convenience we took the time to list some of our favorites on the next few pages. You can see a preview in the table of contents below and then pick the ones that look the most interesting to you. Just remember that there is a lot more where these came from. And you can check out all of our application stories on our website.



By Chip McDaniel
AutomationDirect

About The University of Idaho
Since 1889, the University of Idaho has been a place that expects more from itself, more from its students, more from knowledge and more from life. The University of Idaho is where students come to succeed and learn to lead. At our main campus in Moscow, Idaho – and our centers throughout the state – students find many opportunities to share in our community and culture.

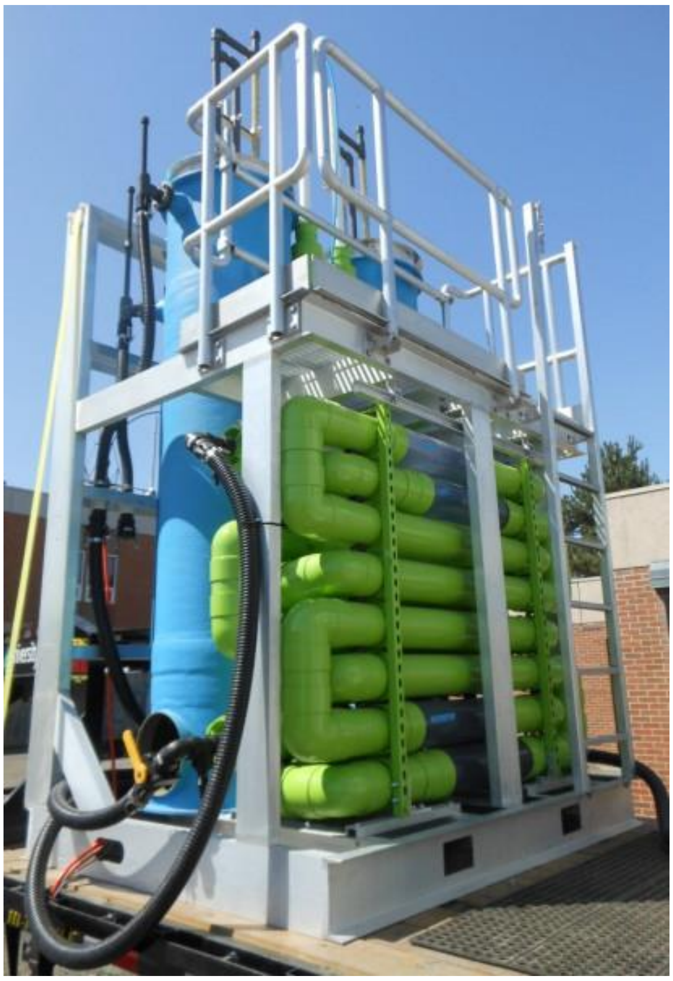
We offer a distinctive combination of outstanding majors and graduate programs, accomplished faculty, world-class facilities, renowned research and a residential campus in a spectacular natural setting. (Taken From www.uidaho.edu)

Portable Water Treatment Platform Removes Phosphorus Economically

A trailer-mounted water treatment process controlled by a Productivity2000 PLC has recently been developed at the University of Idaho. Professor Greg Möller, in the College of Agriculture and Life Sciences, along with engineers from the College of Engineering have teamed up to build the platform, and to further research the economical removal of phosphorus from various wastewater sources.

The engineering effort is part of a capstone project involving a number of engineering students and a French industrial water engineering intern from Suez, led by Martin Baker, a mechanical engineer on staff at UI. Gene Staggs is a recent graduate from the engineering school, and is the Automation and Control lead on the project. Gene contacted AutomationDirect in early June, asking about PLC options to control the project. In addition to the technical criteria, he also explained the accelerated timeline for completion of the project. AutomationDirect technical staff reviewed the capabilities of the Productivity2000, and assured him that his entire bill of materials could ship the day he placed the order.

While the primary function of the mobile platform is to go “on-site” for up to a week at various locations (paper mills, lagoons, and larger waste water treatment plants) to process effluent, remove phosphorus and collect samples, the platform is also a STEM Educational vehicle. The development team has already coordinated tours for elementary through high school teachers, and they have plans to take the platform to various professional conferences as well.

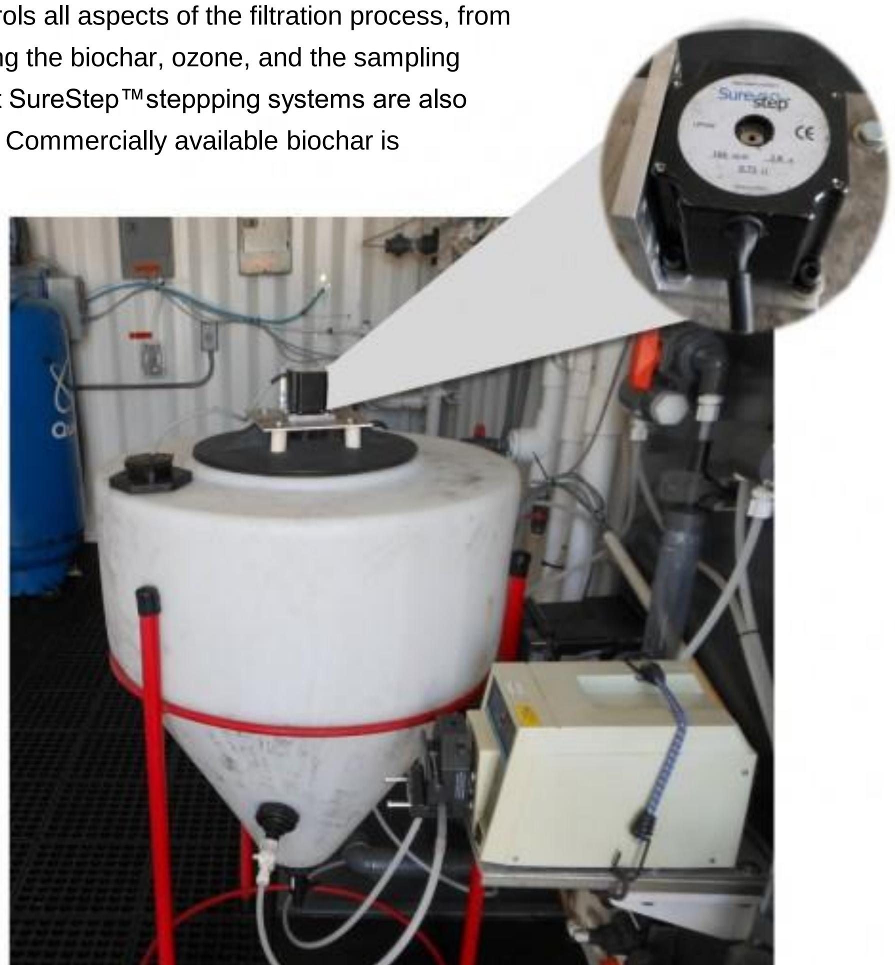


The treatment process is accomplished via a pair of “plug-flow” reactors in series, each followed by a corresponding self-cleaning sand filter. The reactors are 45 feet long, constructed from 6” diameter PVC pipe (the green tubing in the pictures). The first reactor introduces low doses of biochar, a less expensive, yet similar material to the activated carbon used for many filtering applications. After moving through the first reactor, samples are taken and the effluent moves into the first sand filter. Samples are taken again, and the effluent moves into the second reactor where precise quantities of ozone are injected. After a third sample is collected, the effluent moves through the second sand filter, a final sample is collected, and processing is complete.

The entire process is known as “catalytic oxidation”; it allows for destructive removal of most organic contaminants (they are mineralized to carbonates) and it sterilizes the water, killing all microbial life including prions, viruses, and bacterial spores. Mineralized nutrient phosphorus and nitrogen are bound to the biochar and recovered from the water. The nutrient-laden biochar can be recovered and pelletized for use as fertilizer and as a soil amendment in agriculture.

The mobile platform processes about 15 gallons per minute, but because it is based on a commercially proven process, it is scalable to tens of millions gallons per day.

The Productivity2000 PLC controls all aspects of the filtration process, from opening and closing valves, to metering the biochar, ozone, and the sampling operations. A pair of AutomationDirect SureStep™steppping systems are also used for preprocessing of the biochar. Commercially available biochar is available in three forms: powdered, slurry, and suspension. The engineers developed a creative approach to manipulate and meter the biochar, regardless of type. Three sets of custom 3D printed parts were designed to fit on the shafts of the stepper motors. A simple auger processes the powdered form, an auger and a linear actuator are used for the slurry form, and a pair of mixer paddles and peristaltic pump are used for the suspension form of the biochar. The Stepper systems are controlled using a high-speed output card (P2-HSO) in the Productivity2000.



Additionally, the Productivity2000 is running eight separate PID loops (with auto-tuning); it also communicates with a C-more touch panel HMI for operator control. "It is a great system to work with – some of the best features were the custom tag names, PID features, and QR codes" said Gene, just after completing the three-week build. Gene also reports that the import/export of the tag database was a huge time saver, and the team was even able to use the spreadsheet style tag data to help document other parts of the project. He pointed out other benefits as well, "Because this is also a research vessel, the data logging and network capabilities are invaluable."

The mobile platform gives the team a great vehicle to further their research and to show wastewater treatment operators the benefits of the new technology. While it is still in the fairly early stages of development, the results are very promising. Existing studies conclude that reactive filtration is one-third to one-half the capital, maintenance, and operation costs of other approaches, AND the process will produce useful fertilizer, clean reusable water, and even carbon trading credits.

Interested parties should contact the University of Idaho (ott@uidaho.edu) or professor Greg Möller (gmoller@uidaho.edu)bacterial

By Scott Martin,
President, KCC Software

About SMG Valves

SMG Valves is a wholely owned subsidiary of Southern Manufacturing Group, Inc. (SMG). Our line of valves include Key Port Valve® Series 400, Permaseal® Plug Valve and Chem-Plug® Valve lines. These valve lines were purchased from either DeZurik®/Copes-Vulcan® or Mueller Steam and represent the combined quality of trusted brand names and our commitment to quality. SMG Valves is ISO 9001:2008 Certified QMS.
SMG is a corporation established in 1993 under the laws of the State of Tennessee to pursue business opportunities in manufacturing. SMG operates from offices and manufacturing facilities in Morrison, Tennessee. (Taken From SMGvalves.com)

Automation System Retrofit Transforms 175-ton Hydraulic Forming Press

In less than one month, a rebuild automated press operations to increase production rates and improve quality. Southern Manufacturing Group (SMG) located in Morrison, Tennessee, was founded in 1993. The company makes automotive components and industrial valves. In the fall of 2012, the automation system for its 175-ton hydraulic forming press received electrical damage resulting from a lightning strike. Purchasing a new hydraulic forming press would have been prohibitively expensive, in the range of hundreds of thousands of dollars. Therefore, SMG needed a company that could repair the forming press in a minimal time-frame to maintain its production schedule. Required repairs mandated either replacement of components on the existing antiquated automation system, or a completely new automation system. The decision to repair or replace would be driven by multiple considerations including cost, schedule, and functionality of the old versus the new automation system.

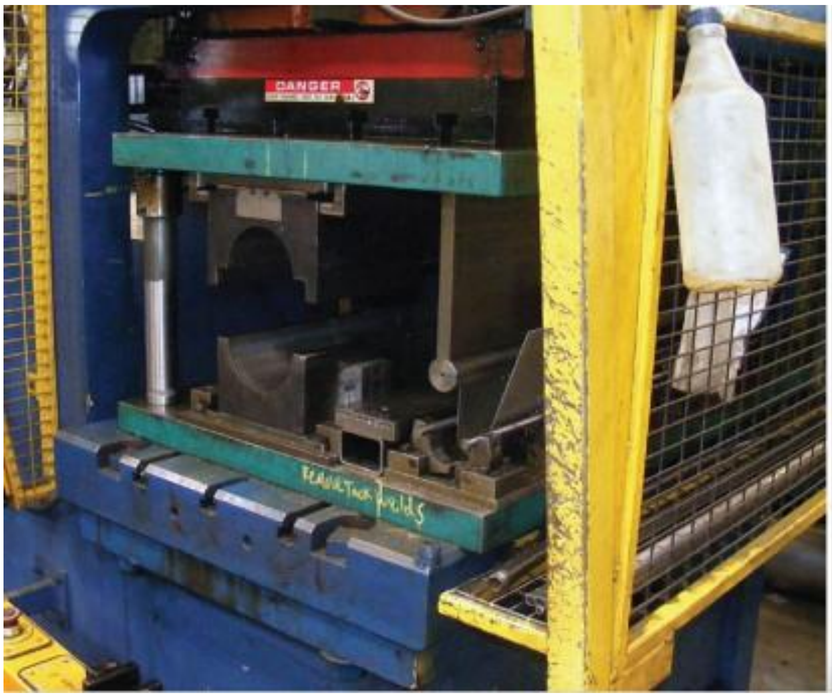


Figure 1: This 175-ton hydraulic press was severely damaged by lightning, necessitating repair or replacement of many key components, including the automation system.

Repairing and Improving the Press

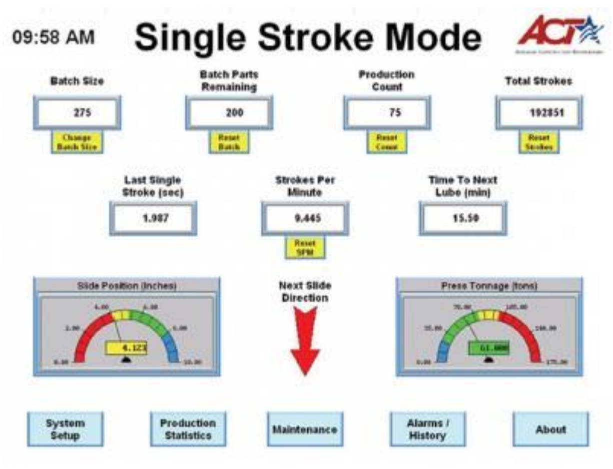
Although the press is 40 years old, it's still a very robust machine that was definitely worth repairing. A dual-action press, it shapes two separate parts in different ways at the same time. During the first step the hydraulic forming press bends the metal in half without creasing it; in the second step it rounds the metal piece. After the first step the metal piece moves to the second step, and a new piece is inserted in the machine for bending. This continuous process significantly increases production yield as compared to simpler machines that only shape one part at a time, but it also requires more than a simple automation system, as multiple operations need to be controlled simultaneously to close tolerances. In order to get the hydraulic forming press working again, SMG turned to fully licensed electrical and mechanical contractor, American Construction Technologies (ACT), in Nashville, Tennessee, to find an economical and timely solution. When ACT inspected the press, its engineers were faced with multiple challenges, including the repair or replacement of the automation system. At this point ACT called KCC Software in Huntsville, Alabama, for an automation solution that would not only work with the existing machine, but also improve it. Our longtime business partner ACT called us because we specialize in software programming and technology integration for manufacturing customers. While ACT would build the new automation system enclosure and perform all wiring, as well as improving and updating parts of the hydraulic system, we would take on the challenge of quickly creating a new, modern automation system.

Designing the New Automation System

Even presses designed 10 years ago often don't have the automated features and functionalities that help manufacturers decrease cycle times and improve production consistency. For example, the old automation system couldn't automate press runs, so operators needed to manually enter instructions with each job setup. Not only did this add to cycle times, it also significantly increased the possibility of error and inconsistent part runs. The old automation system also lacked modern features and functions, and was becoming difficult to support. For these and other reasons, SMG was willing to consider replacement rather than repair of the old automation system, provided the replacement system could be designed and installed quickly, and for a reasonable cost. Preliminary investigations were sufficient to allow us to satisfy SMG's requirements, so we began the next step of new automation system design. KCC Software began design of the automation system by considering the system requirements –pressure transducer feedback, encoder feedback, fast response times, accurate and timely valve controls -- indicating that a very flexible controller was needed. We've been a long-time satisfied user of AutomationDirect products, so we selected their Do-more series PLC CPU module along with a high-speed counter module to interface to the encoder on the press. The other I/O cards selected for the PLC were standard ***Direct***LOGIC 205-series cards for the discrete inputs and the relay outputs required by the press, along with an analog input card for pressure transducer feedback. There were many reasons why we selected the Do-more PLC for the automation system retrofit, besides previous positive experiences with AutomationDirect. The PLC is extremely fast and very easy to program, yet very affordable. This PLC offers programs and tasks, essential to a well-designed automation system controller program, particularly for this application. Not all PLCs provide these controllable program segments, especially not the less expensive and more compact models. These features enabled us to separate the code into manageable pieces; its programs and tasks can also be employed multiple times inside a program, as well as used to develop new programs. A key step was programming the force of the hydraulic press to match the preset movements of the machine. The correct pressure together with the corresponding ram travel plus other key parameters were stored in the new controller's memory by job number, and were configured to be automatically preset by the die setter. For deep drawing operations, the blank holder or hydraulic die cushion force was programmed to be varied through the press cycle, improving operations and reducing machine wear. The high-speed counter interface module helped us design, program and verify the encoder interface in just over an hour. Ironically, this was the part of the automation system design that concerned us the most, yet it turned out to be the easiest as the wiring and data connection were very simple, and the programming required very little math due to the built-in functionality of the module, plus its seamless integration to the PLC's CPU module. We chose the AutomationDirect 15-inch C-more touch screen for the system HMI. This panel offers large, easy-to-read screens with password-protected access to maintenance screens and documentation, along with other features not available on the older automation systems.

Better Than New

Adding a powerful PLC and HMI panel to the 175-ton hydraulic forming press gave the operators the ability to perform touchscreen commands to automatically control the press, eliminating the need for manual adjustments. In addition, the new automation system stores multiple programs or recipes, improving both cycle times and positional accuracies. Prior to the rebuild, operators had to manually enter the program parameters for each new job. Now, the press operator simply selects the desired recipe, which contains values that have been checked through qualifying runoffs.



09:58 AM Single Stroke Mode

Batch Size	Batch Parts Remaining	Production Count	Total Strokes
275	200	75	192851

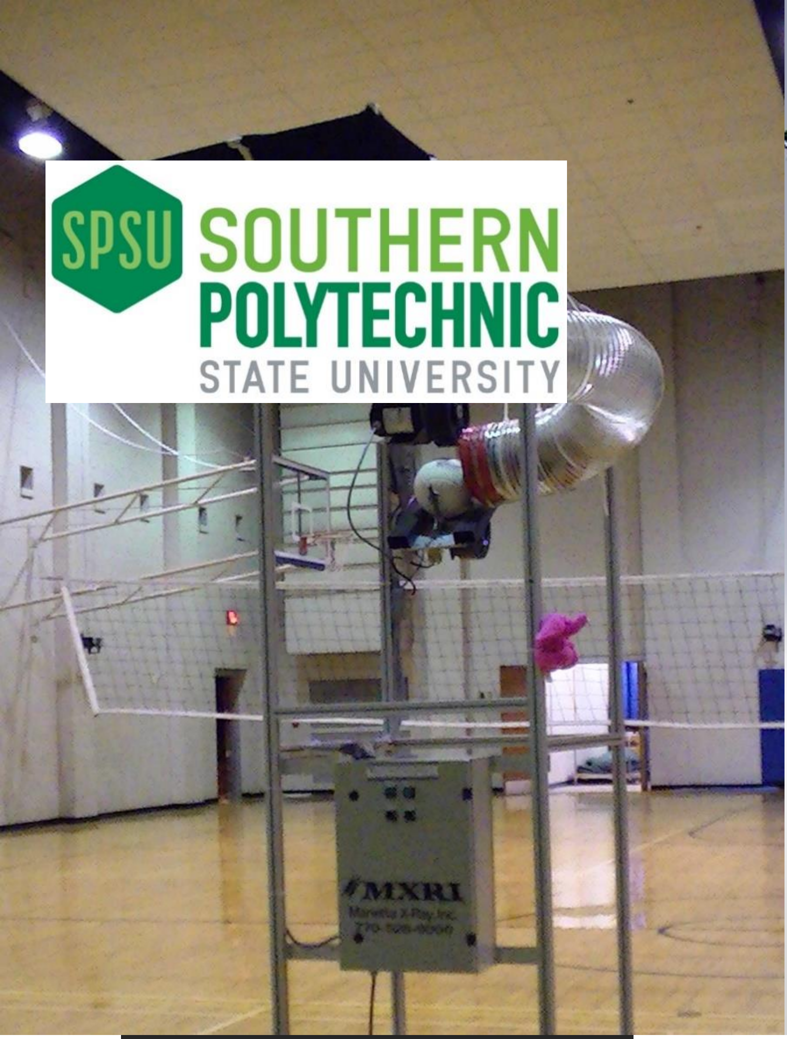
Last Single Stroke (sec)	Strokes Per Minute	Time To Next Lube (min)
1.987	9.445	15.50

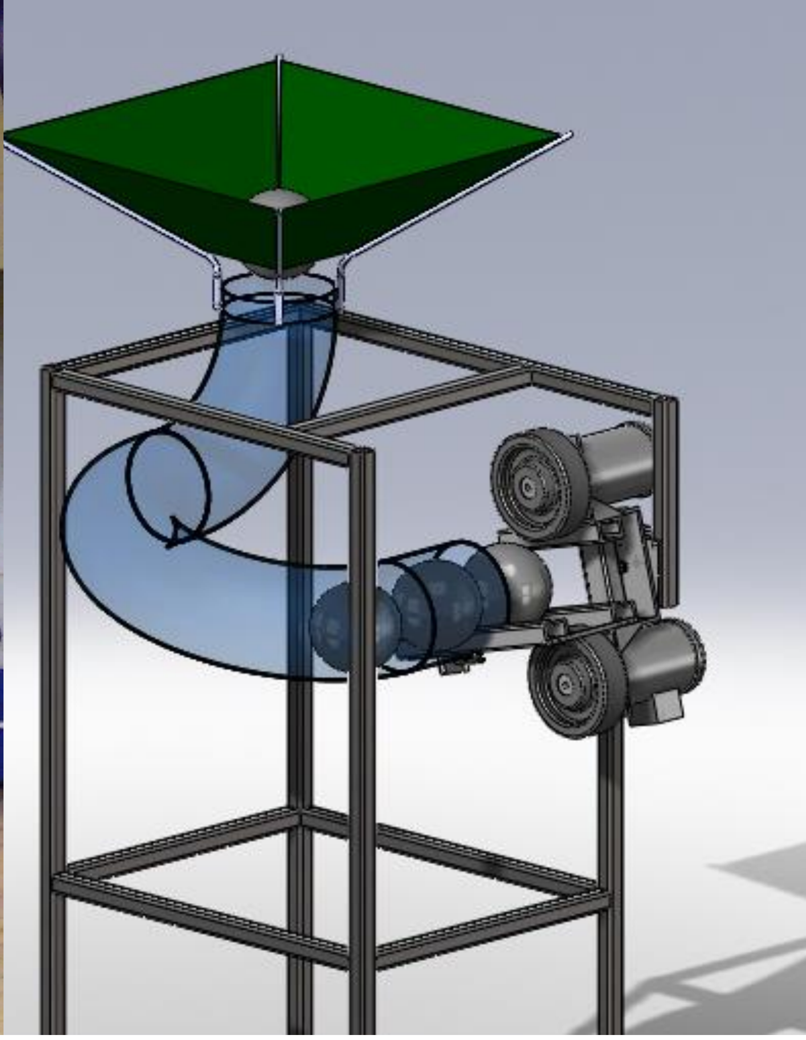
Slide Position (inches): 4.123

Another key feature of the system is statistical analysis of the press' performance. This enables SMG machine owners to detect potential hydraulic issues early on, and also ensures that the machine operates within established parameters. SMG also created additional quality checks to ensure the machine was running according to its specifications. The automation system now measures the depth of the press as it presses on a part. Operators now know the depth of a motion to 1/100th of an inch on a 6 to 7 inch stroke length. This helps them predict when the die needs sharpening, or when the hydraulics need repair. The reversal of the press is a critical production process as it prevents parts from falling out of tolerance. Prior to the rebuild, the old automation system required a manual proximity switch to signal positioning on reversal. Unfortunately, the manual switch accuracy was dependent on the steadiness of the operator's hand. The new PLC-based automation system has automated the reversal of direction step, which is now based on measured pressure. This has made the depth of the press motion extremely consistent, ensuring part uniformity and extending the life of the machine. As efficiency increases due to automation and improved operation, wear on the oil and hydraulic components is reduced. If the press depth starts to change, it's usually an indication of a hydraulic leak or a problem with the die that helps form the part. By being able to address these issues before the machine goes out of tolerance, SMG can keep operating without unscheduled downtime.

Installation One Month after Commissioning

Communication and cooperation among SMG, ACT, and KCC Software was critical to project speed and success, particularly as the schedule was very compressed to avoid downtime. The automation system retrofit began in late October 2012, and the replacement automation system was installed and operational before the end of November 2012. The PLC programming software includes a simulation feature, allowing users to program simulated versions of machine operation. Programming these simulated operations, and testing them against the PLC code, eliminated the need for I/O hardware during testing, which facilitated rapid prototyping and development. This capability also enabled the concurrent engineering with ACT that was essential to meeting the rapid schedule demanded by the project. By using the PLC base and modules, we were able to guide ACT in the integration and wiring of the control panel based on existing drawings and knowledge. Although the Do-More is a newer PLC, it leverages existing, familiar components and modules to make integration and documentation easier and quicker for those who have prior experience with AutomationDirect's legacy PLCs. Since installation, the press has run flawlessly. SMG is very pleased with the performance of the new automation system. Particularly appreciated are the recipe functionality, access to the performance data, and the ease of setup and maintenance when changing dies or performing routine preventative maintenance. The press now only requires semi-attended operation to achieve exact positioning, which is critical to both operator safety and product quality. The maximum press depth is extremely consistent, and the automation system's setup screens enable users to tailor desired performance to exacting standards. SMG was also very pleased that the final cost that was approximately 50 percent less than an earlier quote to simply replace the controller boards, confirming our original estimates. By using high-quality yet affordable automation system components, we were able to keep expenditures down, while still delivering a dramatically improved automation system in less than a month.





By Chip McDaniel
AutomationDirect

About SPSU

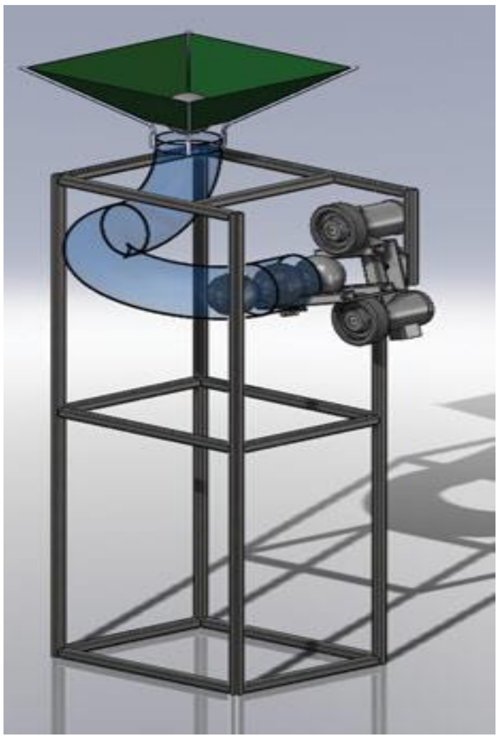
Today's industry must apply technology in every way imaginable. That's why at Southern Polytechnic State University, our students study the sciences and technologies in a unique, practical manner that provides an education that is career-based and balanced.

Southern Polytechnic is a residential, co-educational member of the very progressive University System of Georgia. Located on 203 acres of naturally wooded landscape in the historic and vibrant city of Marietta, we are just 20 minutes from downtown Atlanta. Approximately 5,500 students study here, including student representation from 36 states and 64 countries. (Taken from SPSU.com)

Automated Volleyball Server

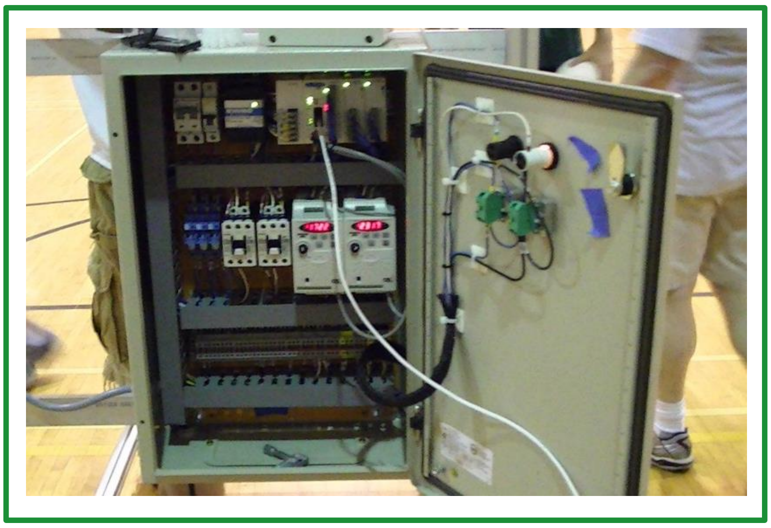
Larry May, a student at Southern Polytechnic State University (SPSU), contacted AutomationDirect, and outlined his goal of building a volleyball setting and serving machine for his senior design project. Larry is active as a recreational volleyball player and is the father and coach of a competitive high school player. Larry is also a long time AutomationDirect customer, and has worked in the Automation field for many years.

He led a team including three other students – Kelly Gray, Ryan Howell, and Trenton Phillips – to design and build a full scale working prototype of the “Ace Within Reach” volleyball setting and serving machine. The team goals were to improve on the existing “state of the art” volleyball serving machines. Several manufacturers offer similar machines today, but Larry and his teammates saw room for improvement in several areas. All the existing machines require the operator to feed the balls manually, one at a time, into the machine, forcing the operator (usually the coach) to be at the opposite end of the court from the player being coached. Larry and his team wanted to build a machine with a hopper to store and feed the balls into the launcher automatically, and to have a remote control interface so that the coach can stay near the player for maximum instructional advantage. They also envisioned a set of lights on the machine to simulate a 'virtual toss', giving the human players an intuitive warning that the launch of a (potentially 80 mph) volleyball was imminent. Other criteria included the ability to fit through a standard door, to roll on casters, and to hold up to 20 balls in the hopper awaiting launch.



The basic electrical design includes a CLICK PLC controlling two GS drives (VFDs) via Modbus. The GS drives power two small Ironhorse 3-phase motors. The 'remote control' consists of a C-more Micro HMI mounted in a handheld enclosure, and tethered to the machine via a long serial cable. The CLICK PLC also controls a series of lights on the front of the machine used to simulate the tossing of the ball into the air (prior to a serve) by a human server.

The mechanical set-up uses two wheel/tire assemblies driven by the Ironhorse motors – one above and one below the volleyball. While the two motors work in concert to determine the overall velocity of the launch, the machine can also impart varying degrees of top-spin to the ball by running the upper motor somewhat faster than the lower motor. Top spin and speed are the primary variables used by volleyball players to vary their serves, and the machine allows a coach to vary both speed and top-spin from the remote control. The mechanical setup also allows for manual adjustments to various angles of the 'launch head' delivery mechanism to allow a wide variety of 'serve' and 'set' simulations.



Research

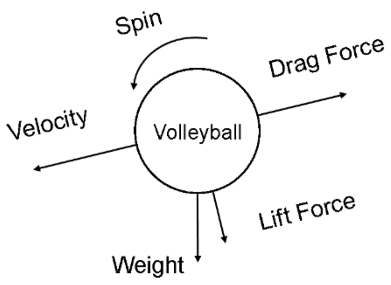
The team conducted a study of forces that apply to a volleyball in flight using Computational Fluid Dynamics (CFD). The purpose of this study was to predict the path the ball would follow as it flew through the air. CFD calculations were performed using the SolidWorks 2009 CAD software with Flow Simulation package. The coefficient of drag (Cd) and the coefficient of lift (Cl) are the two dimensionless parameters that need to be determined to find the trajectory of the ball in flight. The following formulas show how the coefficients are determined:

$$C_d = \frac{2 \times F_d}{\rho \times V^2 \times \pi \times r^2}$$

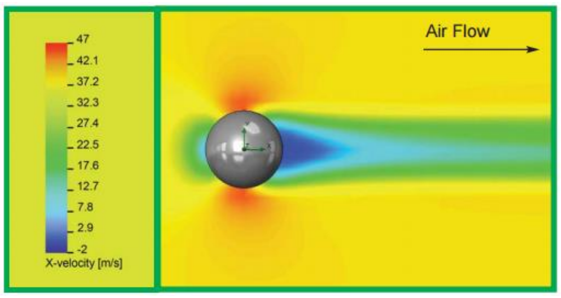
$$C_l = \frac{2 \times F_l}{\rho \times V^2 \times \pi \times r^2}$$

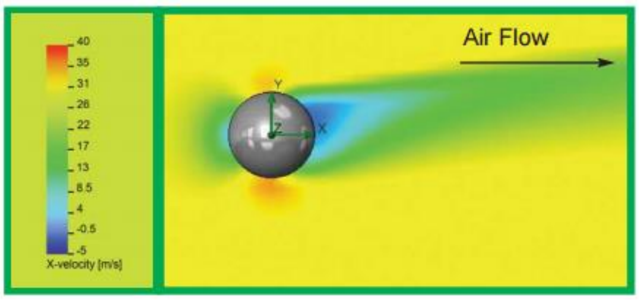
Where
Fd = drag force (N)
Fl = lift force (N)
ρ = density of air @ 25˚ = 1.184kg/m3
V = velocity (m/s)
r = volleyball radius (0.105 m)

Many CFD studies were performed to calculate the drag force (Fd) and lift force (Fl) acting on the volleyball at various combinations of velocity, spin, and surface roughness. The CFD 'solver' was used to determine the forces acting on the ball. These forces were then plugged into Equations 1 and 2 and the coefficients of drag (Cd) and lift (Cl) were calculated. The figure to the right shows a free body diagram of forces acting on the volleyball.



The drag force acts directly opposite to the direction of the velocity vector, and the lift force, due to spin, acts perpendicular to it. Some assumptions were made in order to allow for somewhat faster calculations and a simpler CAD model. The surface variations due to the seams were ignored, and the leather surface of the volleyball cover was modeled with a surface roughness of 6μm and 20μm.

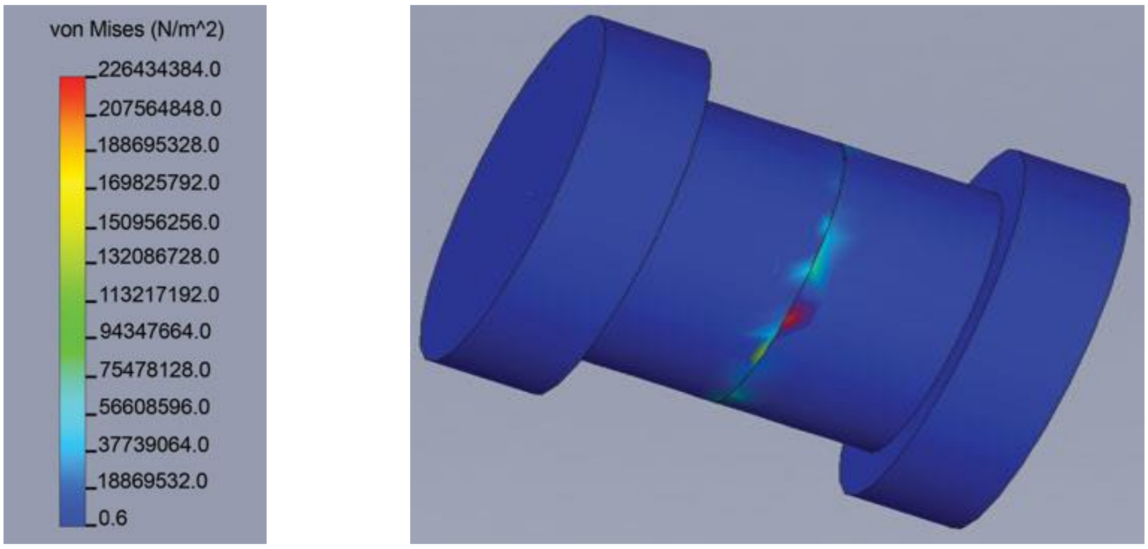




The first flow diagram shows a stagnation point at the front of the ball where the air velocity is very low. On the backside of the ball (dark blue area), some of the air actually moves in the direction of the ball's motion. The red areas, both above and below the ball, show that the airflow reaches a maximum velocity just above and just below the ball. This is due to the ball having the largest frontal area at that section perpendicular to the air flow.

The second flow diagram shows the flow around a spinning volleyball. When the ball has top spin, it resists the air flowing over the top and speeds up the flow under the ball. The red area under the ball is noticeably larger than the red area over the ball. Pressure and velocity are inversely related, so the higher velocity below the ball creates an area of low pressure underneath it. This is the inverse of the familiar airplane wing dynamic which creates the low pressure area on the top of the wing. This means that the spinning ball will encounter a downward 'lift' force, which will make it drop faster than a ball that is not spinning. This will come as no surprise to a volleyball player who has ever faced a powerful, overhand (top spin) serve.

The team also modeled several pressure contours, shear graphs, and flow trajectories. The value for the coefficient of drag (Cd) for all the calculations averaged 0.242. The average value for the coefficient of lift (Cl) for all calculations was 0.466. The team performed a Finite Element Analysis (FEA) on a critical shoulder bolt that holds (and helps to position) the 140 pound launching head assembly. Due to the complex detail of the actual bolt used, a simplier version was created for FEA testing. The static stresses on the bolt were determined to be 65 MPa with a displacement of 0.3 millimeters. The material of the bolt is medium carbon steel with a yield strength of 350 MPa. Since the maximum stress on the bolt is far below the yield strength, this off-the-shelf bolt was deemed acceptable for the application.



Cost Analysis

The costs for the electrical and mechanical components of the machine totaled less than $3000 for the completion of the prototype, right in line with the prices charged for several commercially available serving machines which do not include the multi-ball hopper or remote control capabilities. Of course the team spent many hours designing and building the prototype, but they believe that with certain refinements their design could be manufactured at a reasonably comparable cost – and sold at a competitive price – to the currently available machines on the market.

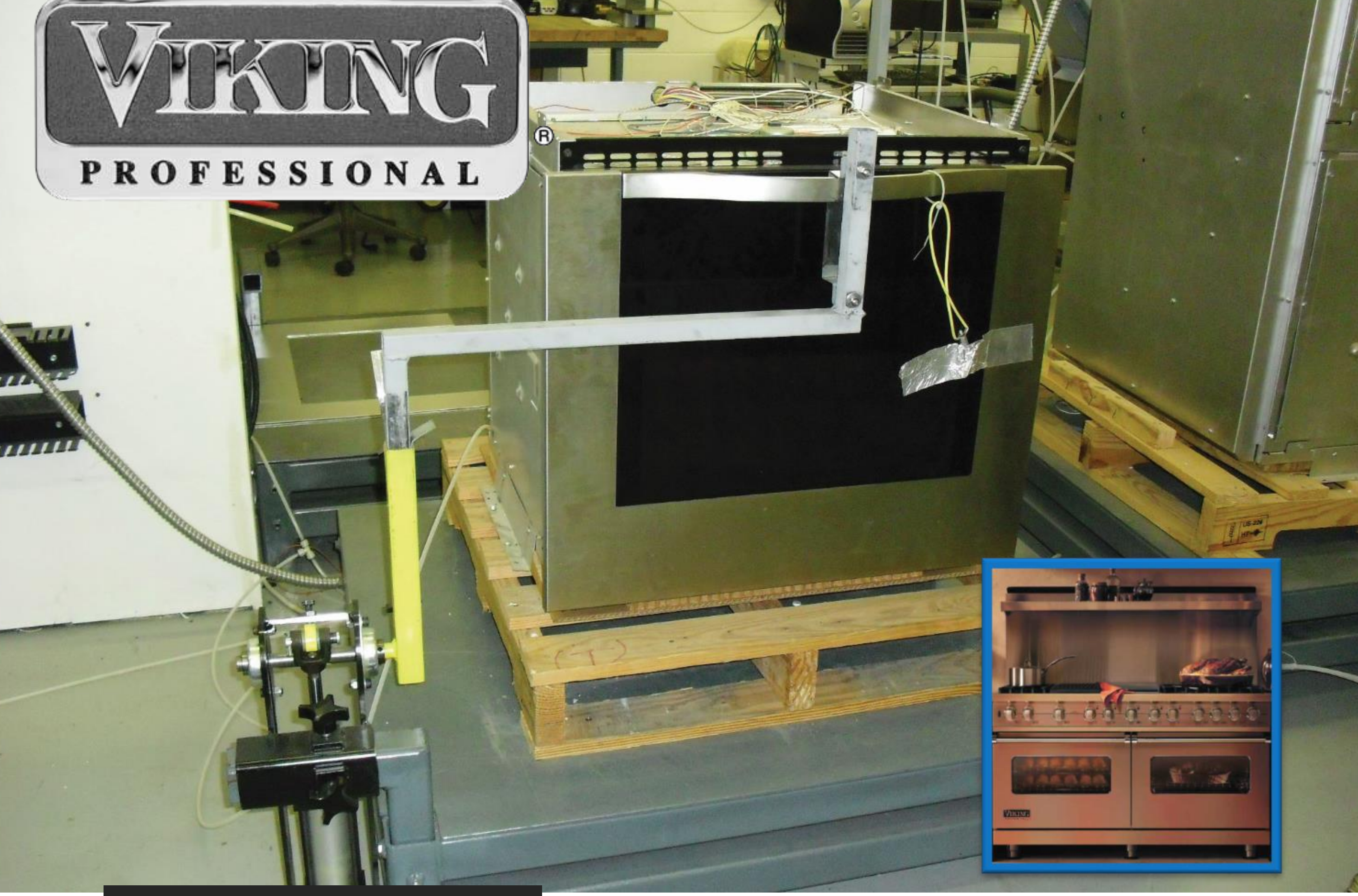
Testing

During the initial testing phase, the volleyballs were not launching with as much velocity as the team had anticipated. They determined that the volleyballs were slipping against the wheels in the launching head. The upper wheel was lowered to add more compression to the volleyball, thus increasing the velocity at which the volleyballs were being launched. While this helped, the machine was still not launching balls at realistic competition serving speeds. The team then wrapped the tire surfaces with a much stickier rubber layer to add extra friction, providing the desired launch speeds.

Conclusions

The team was pleased with the performance of the ‘Ace Within Reach’ Volleyball Serving and Setting Machine. The machine was tested by many bystanders during the testing phase and at the team’s final presentation. The repeatability of the machine allowed testers to improve their game by concentrating on a particular placement, speed, and spin of serve. The immediate feedback was very positive. The team has a few ideas for improvements, and they even have a local volleyball club who has eagerly volunteered to spend several weeks testing the machine and to provide detailed feedback. After they receive feedback from the volleyball club, the Ace Within Reach Team wishes to explore the possibilities of patenting and marketing the automatic volleyball serving and setting machine.

As for Larry, this senior design project was the last step required for graduation, and a key step towards his new Engineer position with Hansgrohe. Established in 1901, Hansgrohe designs, manufactures and sells luxury shower, bath and kitchen products. This internationally recognized company is based in Germany and has a U.S. manufacturing headquarters located in Atlanta, Georgia.



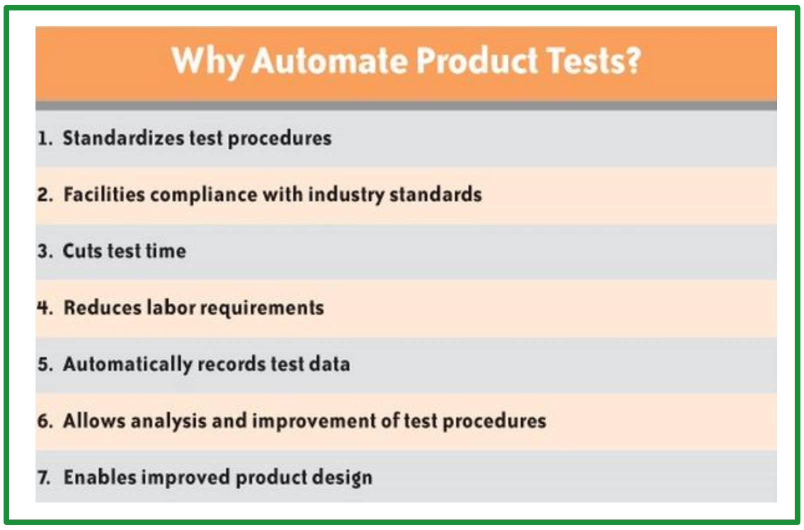
By Barry Hupp,
Reliability Manager,
Viking Range, Greenwood, Miss.

Viking Range, LLC originated ultra-premium commercial-type appliances for residential use, creating a whole new category of home appliances. Committed to innovative product design, unrivaled performance and peerless quality, Viking is headquartered in Greenwood, Mississippi, and is a subsidiary of The Middleby Corporation, a long-time leader in commercial kitchen technology. Viking appliances are recognized globally as the foremost brand in the high-end appliance industry and are sold through a network of premium appliance distributors and dealers worldwide.

Viking Range uses PLCs to Automate its Testing Labs

AutomationDirect PLCs enable continuous life-cycle testing, provide data collection, and shorten product design and verification cycles at Viking Range.

When Fred Carl, Jr. was designing his house in the early 1980s, he wasn't satisfied with the quality of available ranges. He came up with his own range design, and soon thereafter launched Viking Range (www.vikingrange.com). Viking Range was established on Carl's requirements for high reliability and quality, a mandate that has been carried forward ever since. Although contract manufacturers in California and, later Tennessee, made early Viking Range products, all production moved in-house to our Greenwood, MS, facility in 1990. In-house manufacturing enabled our company to provide higher quality and greater reliability, but we wanted to improve further, and saw automated testing as a means to that end.



Why Automate Product Tests?
1. Standardizes test procedures
2. Facilities compliance with industry standards
3. Cuts test time
4. Reduces labor requirements
5. Automatically records test data
6. Allows analysis and improvement of test procedures
7. Enables improved product design

Why Automate Product Testing?

The dishwasher lab and the cooking and refrigeration lab are our two main reliability test labs. We started the dishwasher lab in 2003 and the cooking and refrigeration reliability lab in late 2005.

Before we created the labs, product engineering did their own testing. At that time, product engineers weren't using PLCs, and testing products manually was burdensome, so we needed to build several automated test stands for testing components to be used in dishwasher designs.

By automating the test labs, we aimed to reach several goals as outlined in Table 1. We needed to standardize test procedures to facilitate compliance with industry standards, and to improve product testing in general. We wanted to cut test time, and to reduce labor requirements. Automatic recording of test data was another goal, as this would allow analysis and improvement of test procedures. Finally, better test data would enable us to improve our product designs.

To reach these goals, we began using PLCs for reliability test stands. An electronics engineer working on dishwasher products suggested using DirectLOGIC DL06 PLCs from AutomationDirect.

We found that the AutomationDirect PLCs were simple to use, economical to purchase, and that they could measure everything we needed to test the dishwasher components. When the time came to build more test stands as well as start the cooking and refrigeration lab, we turned to AutomationDirect.

We continue to use AutomationDirect DL05 and DL06 PLCs in our test labs. We use both AC input and DC input versions of both PLC models. Other modules we use include thermocouple input and 0-10 VDC analog input. We typically don't use analog or solid state outputs, but use relay outputs instead.

Other AutomationDirect products we use include C-more 3-inch Micro-Graphic touch panel HMIs, DirectSOFT5 PLC programming software, and KEPDirect OPC software.

Washing our Hands of Manual Test

In 2003, the design of a new dishwasher product started Viking Range on an evolutionary path toward creating reliability test labs and using PLCs to automate test-stand operation. The chief engineer on the dishwasher project wanted the major components tested to ensure a statistically high level of reliability.

We created, designed, and built six test stands to ensure dishwasher components exceeded Association of Home Appliance Manufacturers (AHAM) and Viking Range standards for long-term reliability. AHAM is an ANSI-accredited standards development organization that develops and maintains technical standards for various appliances to provide uniform, repeatable procedures for measuring specific product characteristics and performance features.

Figure 1: Automated testing of products like this range enables Viking Range to provide high quality products with superior reliability to their demanding customer base.

The components to be tested were dishwasher motors, drain pumps, water valves, soap dispensers, water heaters, and heaters/blowers for drying the dishes. We built a separate test stand or fixture for each of these part/component types, and each of the six test fixtures had the capacity to test 22 components. A DL06 PLC controlled each test fixture except for the water valve test fixture which used a DL05 PLC.

We built the test fixtures to simulate how these components would be used in a dishwasher. For example, a water valve may come on for 70 seconds to put water into the dishwasher five times during a wash cycle. We used a PLC to control the timing cycle. When temperature or current measurements were required, we simply added an analog input module.

Over the years, we've changed the test fixtures and PLC configurations to accommodate our reliability testing requirements. Today, we still use part of the original dishwasher test lab, but instead of the DL06 PLCs we started with, we use three DL05 PLCs. These PLCs are mounted on custom stands that test individual dishwasher parts such as soap dispensers and pump and drain motors.

Other dishwasher life testing tasks we've added include a door cycle fixture for opening and closing a dishwasher door from 30,000 to 50,000 times (Figure 2); an upper slide rack fixture that tests a dishwasher's upper rack by sliding it in and out; and a fixture that tests dishwasher water valves by cycling them on and off repeatedly.

The primary dishwasher test lab has four dishwasher life test racks with 16 test stations for each rack. A test station is where testing of an individual product such as a dishwasher takes place, whereas a rack consists of multiple test stations grouped together.

Dishwasher design engineering uses two of the four racks in the dishwasher test lab, and reliability engineering uses the other two. Each of the four racks has a DL06 PLC, and the two reliability engineering racks each have an additional DL06 PLC.

The tasks performed by each of the four DL06s include keeping track of the number of completed cycles for each dishwasher; monitoring the test rack drain pump safety circuit; staggering dishwasher start delays when lost utility/facility power is restored; and energizing soap injection pumps at the start of each main wash cycle.

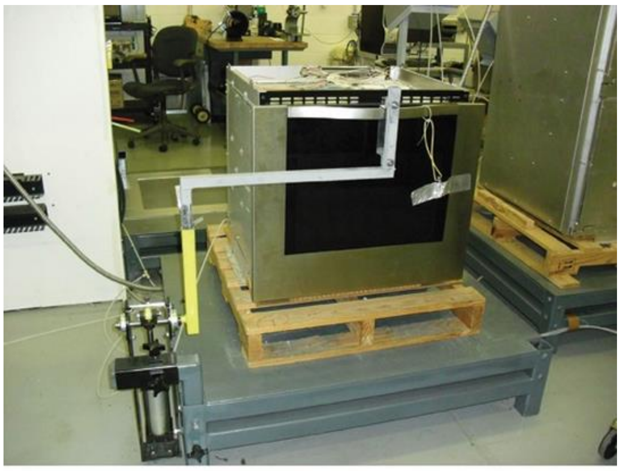


Figure 2: This custom door cycle fixture performs life testing by opening and closing a wall oven door up to 50,000 times, depending on test requirements.

The PLC for each rack receives a signal from each dishwasher at the beginning of the main wash cycle. The PLC subsequently energizes a small pump, which injects the appropriate amount of soap into the dishwasher. There are 16 DC pumps for each rack, one for each dishwasher. These pumps are external to the dishwashers as they are components of the test racks.

The two additional DL06 PLCs control the two test racks that belong to reliability engineering, and control air cylinders that push dishwasher cycle and start buttons. The relay outputs of the PLCs operate air control valves, which, in turn, operate the air cylinders.

To operate a Viking Range dishwasher, users must push control panel buttons while the door is open. We designed the dishwasher test stands with relays that simulate opening the door. We use small 4-inch air cylinders that physically push the wash cycle and start buttons. The air cylinders are positioned at the top of the dishwashers at each test position. Tests start every 2.5 hours, and continue to loop 24/7. Dishwasher current draw and power consumption are monitored using AC voltage and current transducers connected to an industrial PC.

Cooking and Refrigeration Test Runs Hot and Cold

The main reliability lab—also known as the cooking and refrigeration test lab—is our largest testing facility. Here, we test ranges, wall ovens, cook tops, refrigerators, ice makers, and compactors. Our cooking and refrigeration test lab uses more PLCs than the other labs, with most of them monitoring range temperatures.

The DL06 PLCs operate test stands and control cycle times. For example, cycle times for ranges can be 30 minutes on and 10 minutes off for 6,000 cycles. Range and wall oven doors receive rigorous workouts as well. In the cooking and refrigeration test lab, two test stands, each with a DL05 and C-more HMI, operate and monitor the range and wall oven door open-and-close cycles. The HMIs display the cycle counts, while the touch screen keyboard allows test engineers to reset the count.

Some of the DL06 PLCs are configured to measure current draw of electric ranges and cook tops. AC current transducers measure the current usage monitored by the PLCs and collect data for graphing this current.

DL06 PLCs measure temperatures of ranges, ovens, and cook tops under test. Thermocouples are attached to analog inputs cards. The PLCs are programmed to run repetitive cycles that turn units on for an hour then off for 15 minutes, for example. We log the monitored temperature data in Excel, with communications between each PLC and our three PCs via KEPDirect OPC software.

OPC communications provides us with an easy way to transfer data using standard protocols. We quickly learned how to use the OPC/DDE server to collect data, and to write or modify macros in Excel to graph the collected data. Daily graphs show dips in the oven temperatures if there are issues or failures. This functionality allows us to run our tests continually, reducing total project testing time.

We use RS-232 or Ethernet for these PLC-PC connections. We use four industrial PCs and three laptop PCs in the cooking and refrigeration test lab to log test data and to write test programs and profiles that run on the PLCs. We also use these PCs as HMIs.

We use DirectSOFT5 Windows-based PLC programming software to write or modify the test programs used to test appliances or component parts. For example, we recently created a program that tests three gas ranges in bake mode. The program repeatedly turns the ranges on for 30 minutes and off for 10 minutes.

Results Verify Expected Benefits

We've found the AutomationDirect PLCs to be simple to use, affordable, flexible, and highly reliable. Without much in the way of prior experience, we were able to learn how to program the PLCs in ladder logic programming using the product manuals and the AutomationDirect website. We found both the manuals and the website content to be very well-written and easy to follow.

The optional analog input cards and the OPC communications to our PCs running Excel allow us to measure temperature and current without having to buy separate and expensive data loggers. The availability of both serial and Ethernet communication ports on the PLCs facilitated the required PLC-to-PC connectivity.

The PLCs have enabled us to set up continuous component and product testing. We've been running these types of tests for more than seven years without any problems from AutomationDirect products. Due to our success with AutomationDirect products, Viking Range now uses DL05 and DL06 PLCs in two other manufacturing plants for monitoring and testing products.