



PROGRAMMABLE **LOGIC** CONTROLLERS

HANDBOOK

Volume 1



Programmable Logic Controllers Handbook

Volume 1

Published by The Electricity Forum



The Electricity Forum
215 -1885 Clements Road
Pickering, Ontario L1W 3V4
Tel: (905) 686-1040 Fax: (905) 686 1078
E-mail: hq@electricityforum.com

The Electricity Forum Inc.
Suite 402 One Franklin Square,
Geneva, New York 14456
Tel: (315) 789-8323 Fax: (315) 789 8940
E-mail: forum@capital.net

Visit our website at
www.electricityforum.com

PROGRAMMABLE LOGIC CONTROLLERS HANDBOOK

Randolph W. Hurst
Publisher & Executive Editor

Don Horne
Editor

Cover Design
Alla Krutous

Handbook Sales
Lisa Kassmann

Advertising Sales
Carol Gardner
Beverly Hilton
Barbara John

The Electricity Forum
A Division of the Hurst Communications Group Inc.
All rights reserved. No part of this book may be reproduced without
the written permission of the publisher.

ISBN-0-9738854-3-2
The Electricity Forum
215 - 1885 Clements Road, Pickering, ON L1W 3V4

TABLE OF CONTENTS

What is a PLC?
 By Robert Holman, Owner and Chief Engineer of Automation & Controls Engineering Ltd.5

Advances in Microprocessor-based Distribution Relays
 By Edmund O. Schweitzer, and Mark W. Feltis, Schweitzer Engineering Laboratories, Inc.7

Compromises of Using a 10-Gbps Transceiver at Other Data Rates
 Courtesy of Altera11

Programmable Logic Controllers: Ladder Logic
 Courtesy of All About Circuits15

Ethernet/IP: Industrial Protocol
 By Paul Brooks, European Marketing Manager, Logix/NetLinx Technology Adoption, Rockwell Automation19

SCADA: Choosing a Polling Mode for DF1 Half-Duplex Master
 Courtesy of Allen-Bradley, Rockwell Automation29

Industrial Ethernet: A Control Engineer’s Guide
 Courtesy of Cisco Systems37

PACs for Industrial Control, the Future of Control
 Courtesy of National Instruments45

PLC Developments: The Near Future
 Courtesy of Reed Business Engineering staff49

Programmable Logic Controllers
 Courtesy of Grand Valley State University51

Tuning a PID Controller for a Digital Excitation Control System
 By Kiyong Kim and Richard C. Schaefer, Basler Electric55

Using PLDs for High-Performance DSP Applications
 Courtesy of Altera Corporation63

Web-Enabling your PLC
 By Mike Rothwell, Director of Advantech Automation Corp.’s eAutomation Engineering Center67

Input Signal Edge Rate Guidance
 Courtesy of Altera Corporation69

Removable Storage Media Add Flexibility to Modern day PLCS
 By Mark DeCramer, Advanced Electronics Product Manager, WAGO Corporation71

Assessment and Remediation of Vulnerabilities in the SCADA and Process Control Systems of Utilities
 Courtesy of Internet Security Systems, Inc.73

Basics of Programmable Logic Controllers
 Courtesy of the Motion Controllers Reference Center79

Field Programmable Controllers for Cost Sensitive Applications
 By Richard Griffin, XILINX81

TABLE OF CONTENTS

PLCs Pack It In By Gricha Raether, Product Manager, National Instruments Corp.	85
Logic Control for Robot Work Cells By Bennett Brumson	87
State Language for Machine Control By Kenneth C. Crater, Chairman, Control Technology Corporation	91

WHAT IS A PLC?

By Robert Holman, Owner and Chief Engineer of Automation & Controls Engineering Ltd.

Programmable Logic Controllers (PLC) continue to evolve as new technologies are added to their capabilities. The PLC started out as a replacement for banks of relays. Gradually, various math and logic manipulation functions were added. Today they are the brains of the vast majority of automation, processes and special machines. PLCs now incorporate smaller cases, faster CPUs, networking and various internet technologies.

You can think of PLC technology as a small industrialized computer that has been highly specialized for reliability in the factory environment. At its core, a PLC looks at digital and analog sensors and switches (inputs), reads its control program, makes mathematical calculations and, as a result, controls various hardware (outputs) such as valves, lights, relays, servo motors, etc. in a time frame of milliseconds.

While PLCs were very good at quickly controlling automation, they did not share data easily. At best, PLCs would exchange information with operator interfaces (HMI) and Supervisory Control and Data Acquisition (SCADA) software packages on the factory floor. Any data exchange with the Business Level of the company (information services, scheduling, accounting and analysis systems) had to be collected, converted and relayed through a SCADA package.

Typical of most PLCs, the communication networks were unique to the brand and limited in speed. With the acceptance of Ethernet, communication network speeds have increased but are still sometimes using proprietary protocols.

TRENDS: MORE POWER, WIDER DATA SHARING

Overall, PLCs are getting faster, smaller, cheaper and more powerful. As a result, they are gaining capabilities that used to be the exclusive domain of the Personal Computer (PC) and workstation arena. This translates into critical data quickly and cheaply being shared directly between the PLCs on the Factory Floor and the Business Level of the company. These are not your father's PLCs.

Some of the features that a PLC can bring to your automation projects are Web Servers, FTP Servers, sending E-mail and Internal Relational Databases. The following is a brief overview of these features and some of their uses.

WEB SERVER

PLCs can host a Web site on the internet or your company intranet. So what's that going to do for you? How about give you access to Real Time Data and Data Logging for starters. Do you need a backup Human Machine Interface (HMI) for a machine(s) or work cell? How about as a tool for your Maintenance group? Did you know that with some PLCs you can store documentation with the web server that lets you view machine drawings, schematics, maintenance and operator manuals and short video clips? They are all just a mouse click away

with your web browser.

Web servers in PLCs are probably the most varied and widely used of the newer technologies. PLC web server capabilities vary, depending on the manufacture and model from a single "canned" page to full blown sites using XML and JAVA based technology.

JAVA web servers can provide a high degree of versatility for interacting with a PLC. Three of the JAVA classes of small programs that enhance web server technology are; Applets, Modlets and Servlets. In general terms, the "lets" let you view, manipulate and transfer data faster.

Applets are small JAVA application programs that are sent from the web server to your web browser the first time you open a web page.

Modlets are JAVA modules that run independently of the PLC control program for servicing non-process event driven tasks of data handling or updating calculations separate of the PLC control program. Modlets are very useful for parallel processing functions that interact with the PLC database.

Servlets run in the web server after they are requested by your web browser. They are very useful for displaying live data and dynamically creating data log files (CSV).

SEND E-MAIL

A Send E-mail function automates and simplifies exporting critical and production data out of a PLC. Production data and material usage reports, status changes, alarms, internal PLC data and maintenance requests can be issued from within a PLC control program. With a little time and imagination you can send your alarm messages to the maintenance personnel who carry alphanumeric pagers or cell phones.

FTP SERVER

File Transfer Protocol (FTP) is your tool for easily and quickly moving or copying files in and out of a computer through an TCP/IP ethernet connection. Now it is available in some PLCs. While, on the surface, it does not sound like a big deal, this handy tool can be a major time saver. Why walk out to the PLC to copy files when you can access it though a network from your desk? How much time would you save by dialing into the Ethernet network the PLC is on (or a stand alone modem/router) if the PLC is in another city, state or country?

INTERNAL RELATIONAL DATABASE

One of the most exciting and useful new features that just showed up in the market from SoftPLC Corporation is the "Internal Relational Database" embedded in a PLC. As an internal database, it allows crucial data to be accessed in one program scan (milliseconds) rather than having to wait for it from an external source (another computer or PLC) sending the data

through a communications port.

This feature opens the door for a whole host of cost savings. For example in a sorting conveyor with a bar code reader, the bar code reader usually connects to a PC. The PC looks up the bar code in the database that it hosts and then sends the resulting information to the PLC. Only then can the PLC use the data to control a diverter, gate or bin. There are usually a minimum of 6 steps to get the information to the PLC.

1. Scan the bar code.
 2. Send the data to the PC.
 3. PC decodes the bar code.
 4. PC looks up the resulting information from the database.
 5. Move the data to the PC communications port.
 6. Send the information to the PLC across a serial communication connection.
- And optionally:
7. Move the data from the PLC communications buffer to the PLC program memory and use it (in some cases).

Using a PLC with an internal relational database reduces this to four steps:

1. Scan the bar code.
2. Send the data to the PLC.
3. PLC decodes the bar code.
4. PLC looks up the resulting information from the internal relational database and use it.

Using a PLC with an internal relational database eliminates the weak link of the communications from the PC to the PLC. A PC will usually perform the database lookup faster than a PLC. However, moving the data from the PC database across a serial network connection (usually limited to 19.2k baud) rate is much slower than a PLC that can retrieve usable data in one scan of the control program.



With this sorting conveyor example the first cost savings is reduced computer hardware by eliminating the PC with the database and the database software. A second savings is realized by eliminating the integration time required to get the PC and the PLC communicating. Another cost savings is from the lack of needing the Information Technology department to continuously maintain, backup and upgrade the PC.

There are many different applications that a PLC with an internal relational database can control in a more cost effective way for both the Integrator as well as the End User. Manufacturing machines and processes that assemble product based on recipes or build multiple products on the same machine are all good candidates for this technology. Also, projects that require setting “environment variables” for configuration of a machine or custom written instructions and drivers as well as fast keyed look ups for control values would find the power of a internal relational database especially useful.

SUMMARY

The technologies discussed are available in as many flavors and options as there are PLC manufacturers. This overview

only scratches the surface of the ongoing improvements in PLCs. It can't begin to cover all the features and uses available among these four technologies let alone the important related subjects of networking, security and the continuing shift to Open Architecture.

The significance of the technologies is that control system complexity is rapidly being simplified for greater company wide information exchange so there no longer needs to be “islands of automation”.

This article was written by Robert Holman, owner and chief engineer of Automation & Controls Engineering Ltd, located in the Minneapolis, MN area.

ADVANCES IN MICROPROCESSOR-BASED DISTRIBUTION RELAYS

By Edmund O. Schweitzer III, and Mark W. Feltis, Schweitzer Engineering Laboratories, Inc.

INTRODUCTION

Advanced microprocessor-based distribution relays have features that improve distribution protection and aid in event analysis and testing [References 1, 2, 3, and 4]. Significant relay advancements also handle system disturbances, relay failures, changes in protection philosophy, and changing system conditions.

SYSTEM DISTURBANCES AND EVENT REPORTING

Recreating the sequence of events for a distribution system disturbance with information available from traditional distribution protection equipment is difficult at best. In most instances, targets are the only source of information. Targets can show phase involvement and relative fault current magnitudes (did the instantaneous element operate?), but no time sequence or precise fault current magnitude information. If targets are not reset from a previous operation, the confusion is compounded when the next disturbance occurs.

Sequence-of-events recorders and oscillograms are generally considered too expensive for distribution applications. An experienced operator must interpret oscillogram output.

Technicians now quickly retrieve event reports from advanced distribution relays locally or by telephone modem and review every operation. Event reports simplify event analysis by combining currents, voltages, relay elements, and contact 110 in one report.

Event Report Advantages

- One report contains all information - several cycles worth.
- One report fits on one or two sheets of paper .
- Report retrieval is fast: only 30 seconds at 1200 baud.
- Event reporting is essentially free: it is part of the relay, and digital relays cost less than electromechanical or static analog relays.

System Knowledge Increased

- Sequence-of-events data are now available at all voltages and points in the system.
- We no longer need to rely on oscillographic data that may be available one or two stations away from the fault.
- The phasor voltage and current data, along with fault location, are being used to verify and improve system modeling.
- Scheme performance is easy to ascertain from the detailed data available after each operation.

This is an aid in testing, also.

System Problems Discovered

- Instrument transformer ratio and polarity errors
- Instrument transformer failures
- High-resistance fault analysis

- Repeated failures to synchronize a generator
 - Calculation of zero-sequence impedance of lines
- Reference 5 reviews some event reports and illustrates practical analytical tools. An example event report is given in Figure 1.

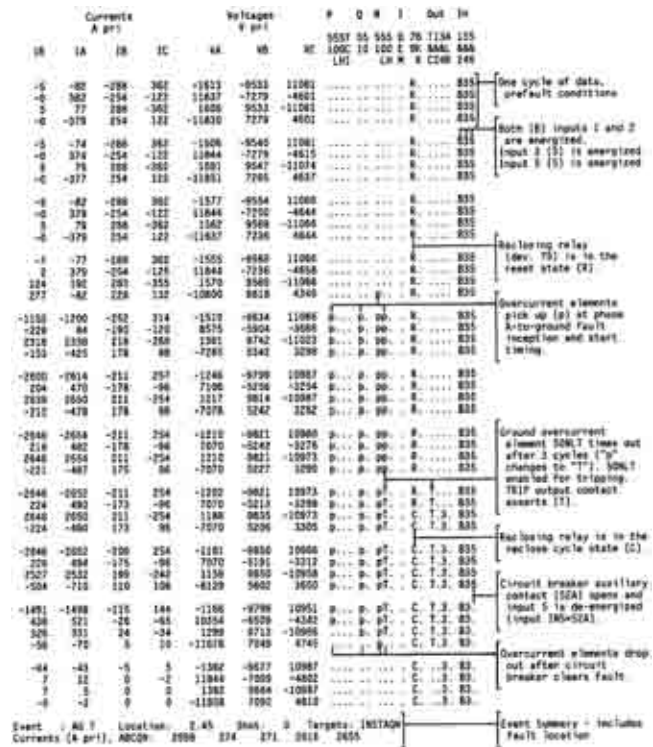


Fig. 1: Example Event Report

SHOULD SELF TESTING AND EVENT REPORTING CHANGE TESTING PHILOSOPHY?

Well-designed digital relays use fewer components than electromechanical or static analog designs, and thus provide more reliable protection. Failures can still occur, however. In the remote chance that a failure does occur, automatic self testing is almost certain to detect the problem and send operators an alarm via SCADA, etc. The result is much improved relay availability. Before digital relays, routine tests (or misoperations) were the only indicators of electromechanical and most solidstate relay problems.

Relays with automatic self testing and event reporting require less testing to ensure they are operating properly. Technicians can now better allocate their limited resources to relays without self testing and to analyzing event reports from relays having event reporting. When technicians review relay

event reports, they are reviewing data from a test of the system created by a real fault. Technicians add more value to the electric power system, because they are checking not only the relay, but the surrounding equipment. Did the relay get proper voltages and currents? Did the circuit breaker auxiliary contact respond correctly?

CHANGES IN PROTECTION PHILOSOPHY AND PROGRAMMABLE LOGIC

Protection schemes are constantly being modified and improved. Changing traditional protection schemes often requires purchasing and installing additional equipment.

Philosophy of distribution protection changes over time and differs from company to company; sometimes from location to location within the same company. If a distribution relay meets one company's requirements, it may not satisfy another's. Making factory modifications to satisfy different companies is costly, time consuming, and results in "one-of-a-kind" relays.

Relay design engineers devised user-programmable logic for new distribution relays. This logic handles future protection schemes as well as the multiplicity of protection schemes that protect different companies' distribution systems. With these new relays, there is less need for additional equipment as protection schemes are enhanced.

Internal relay elements (e.g., overcurrent, voltage) can be programmed to control output contacts.

This flexibility aids technicians in "isolating" and testing relay elements.

Programmable Logic Example

In the following example, relay opto-isolated input IN6 supervises the ground overcurrent elements (SINT and SONLT) for tripping. Input IN6 can be de-energized during circuit paralleling operations to prevent the ground overcurrent elements from initiating a trip on temporary current unbalance.

Figure 2 shows the example in relay logic form.

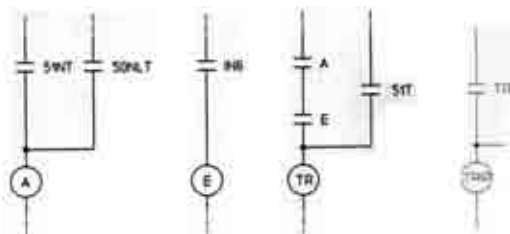


Fig. 2: Relay Logic Representation of Programmable Logic Example

$A = 51NT + 50NLT$
 $E = IN6$

$V = A * E$

$TR = 51T + V$

If 51 NT or 50NL T assert, "A" asserts.

If input IN6 is energized with nominal control voltage, variable E asserts.

If both variables A and E are asserted at the same time, variable V asserts.

Effectively, $V = (51NT + 50NLT) * IN6$.

TR is the TRIP output contact programmable trip variable. 51T is a phase overcurrent element. If 51T or V asserts, TR asserts.

Effectively, $TR = 51T + [(51NT + 50NLT) * IN6]$

Close TRIP output contact = TR + ...

CHANGING DISTRIBUTION SYSTEM AND MULTIPLE SETTING GROUPS

An electric power distribution system changes hourly to

seasonally:

- Scheduled switching for construction or maintenance projects
- Emergency switching for repairs
- Bus-tie breakers substituting for distribution feeder breakers

- Seasonal load transfers

The resulting system reconfigurations last from hours to months. Many reconfigurations are repeated.

The following problems can result:

- Major changes in load or unbalance
- Large variations in fault duties, due to source and feeder changes
- Coordination problems with different protective equipment

- Increased fault duty on conductor, cable, and equipment

Traditional protective equipment does not adapt readily to distribution system reconfigurations. If new settings are needed they have to be manually changed: there are no settings in reserve. The time to derive, enter, and test new settings slows down emergency responses and risks human error.

Sometimes relay settings are not changed for emergency or abnormal switching because it takes too long or is too difficult. System protection is compromised.

Advanced distribution relays handle system reconfigurations with multiple setting groups. A specific setting group can be enabled by:

- Command via communications port or
- Setting group selection inputs

Different setting groups can be programmed to cover many different contingencies. The optimal protection scheme and settings are enabled for highest service reliability.

Operational costs are reduced by not having to derive, enter, and test new settings. This is done conveniently in advance. For example, in a bustle circuit breaker application, the bus-tie relay stores the settings for the feeder relays it replaces (see Figure 3).

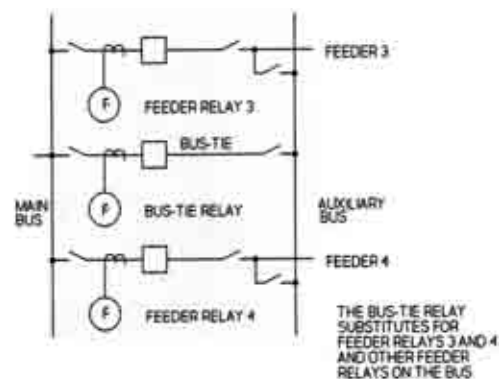


Fig. 3: Bus-tie Breaker Substitutes for Distribution Feeder Breakers

References 6 and 7 describe the first relay with multiple setting groups. In new distribution relays, the multiple setting group feature has been expanded to allow the programmable internal logic and output contact functions to change along with the relay element settings.

CONCLUSIONS

Event reports, self testing, programmable logic, and multiple setting group features in advanced microprocessor-based distribution relays improve distribution protection and aid in event analysis and testing.

REFERENCES

1. Data Sheet, SEL-151 Distribution Relay, Schweitzer Engineering Laboratories, Inc. (1991).
2. Data Sheet, SEL-151C Distribution Bus Relay, Schweitzer Engineering Laboratories, Inc. (1991).
3. Data Sheet, SEL-251 Distribution Relay, Schweitzer Engineering Laboratories, Inc. (1992).
4. "Improved Sensitivity and Security for Distribution Bus and Feeder Relays," Ahmed F. Elneweihi, British Columbia Hydro and Power Authority, Edmund O. Schweitzer, III and Mark W. Feltis, Schweitzer Engineering Laboratories, Inc. Proceedings of the 18th Annual Western Protective Relay Conference, Spokane, Washington, October 22-24, 1991. Washington State University, Conferences and Institutes, Pullman, Washington.
5. "Analysis of Event Reports," by Jeff Roberts and Edmund O. Schweitzer, III, Schweitzer Engineering Laboratories, Inc. Proceedings of the 16th Annual Western Protective Relay Conference, Spokane, Washington, October 24-26, 1989. Washington State University, Conferences and Institutes, Pullman, Washington.
6. Data Sheet, SEL-121B, Schweitzer Engineering Laboratories, Inc. (1989).
7. "Novel Applications of a Digital Relay with Multiple Setting Groups," Demetrios A. Tziouvaras, Pacific Gas and Electric Co., and William D. Hawbaker, Schweitzer Engineering Laboratories, Inc. Proceedings of the 17th Annual Western Protective Relay Conference, Spokane, Washington, October 23-25, 1990. Washington State University, Conferences and Institutes, Pullman, Washington.

BIOGRAPHIES

Edmund O. Schweitzer, III is President of Schweitzer Engineering Laboratories, Inc. (SEL), Pullman, Washington. SEL designs and manufactures microprocessor-based protective relays for electric power systems.

Mark W. Feltis is a development engineer at SEL. He previously worked for the Pacific Gas and Electric Company (California) in distribution engineering.

COMPROMISES OF USING A 10-GBPS TRANSCEIVER AT OTHER DATA RATES

Courtesy of Altera

INTRODUCTION

Many applications and designs are adopting clock data recovery-based (CDR) transceivers for interconnect data transfer. Designs and protocols are standardized on data rates between 1 and 3.2 Gbps for the current data node.

Protocols such as PCI Express, RapidIO, and Gigabit Ethernet all currently use these data ranges. As data requirements continue to rise, designers are turning their attention to next generation architecture, looking for solutions capable of supporting higher data rates, but over legacy systems and backplanes. This challenge is more difficult because of properties of FR4 printed circuit boards at high frequency, such as skin effect and dielectric absorption, which can cause severe degradation of signal quality.

A new generation of products is emerging that addresses the surge in the 1- to 3-Gbps market and enables the move to the next generation transceiver data node. Many standards bodies, including the Peripheral Component Interconnect Special Interest Group (PCISIG) and the Optical Internetworking Forum (OIF), believe the next logical node step for transceivers to be between 5 and 6 Gbps. However, some applications are starting to implement 10-Gbps interconnectivity (currently, more so for line applications than backplane). Considering the complexity of transceiver design, the challenge facing transceiver manufacturers is how to cater to all applications.

Established transceiver manufacturers favor dual transceiver product philosophy to support all applications. The manufacturers provide a generic transceiver to support applications between 500 Mbps and 6.375 Gbps and a more dedicated transceiver to support a data range with data rates between 10 to 11 Gbps.

This paper describes why a data rate transceiver that is too wide comes with system compromises and can introduce factors that may not be acceptable for some applications operating at lower data rates, where power and cost are generally most important. It also explains why good jitter performance at high data rates does not necessarily translate to good performance at lower data rates; specifically, data rate margin does not equal good signal integrity.

The point is that there are no easy compromises when selecting the relative transceiver data range. A transceiver optimized for 10 Gbps cannot deliver low power and low jitter at a low-cost at 3.125 Gbps. Rule-of-thumb assumptions that good performance at 10 Gbps provides extra margin at 2.5 Gbps are incorrect.

TRANSCEIVER AREA

FPGAs must provide solutions for wide ranging applications and their requirements. That is, devices must be feature-rich and provide for many eventualities or offer flexibility in the

architecture to cope with changes in industry.

As an extension of this, the transceiver must provide the same capabilities. A transceiver designed to operate across a wide bandwidth must therefore provide for the requirements for protocols across its data range, otherwise it becomes redundant for certain applications and is not viable as a product.

A transceiver designed to operate up to high data rates must support applications ranging from 155 Mbps to 11.1 Gbps.

This encompasses SDH/SONET with its exacting jitter requirements, standard protocols such as Gigabit Ethernet and PCI Express with 8B/10B encoding and subtle signaling requirements, and 10-Gbps Ethernet with its requirement for 64B/66B encoding techniques. The cost of providing this level of support is increased die area. Transceivers operating at the extreme data rates are particularly susceptible to this because the higher data rate protocol features become more complex, leading to the use of more transistors and a need for higher overall operational system clock.

The following are typical features required to support high data rate applications:

- Complex PLL implementations for robust jitter performance (see “PLL Considerations”)
 - New encoding schemes to support higher data rates (for example, 64B/66B)
 - New schemes for signal integrity (if the device is to stand any chance of operating on standard PCB fabric)
 - Faster bus interfacing into FPGA fabric
- Increased die area has some negative effects:
- Power dissipation
 - Device cost

POWER

An important consideration when selecting a transceiver is the power dissipation. FPGAs with embedded transceivers are generally used in environments where power and heat dissipation is imperative. The transceivers are often located close to backplane interconnects or chassis paneling, where forced-air cooling is difficult to manage. It is therefore prudent for transceiver manufacturers to generate products that are suitable for a wide range of applications and select a technology and feature set that does not increase power.

PLLs have a big impact on power. The transceivers within the Altera Stratix GX device family take up to 30% of the overall power budget of the transceiver. As the data rate widens, the PLL architecture becomes considerably more complex, leading to the use of more logic within the PLL because the PLL must be capable of providing good jitter quality across the entire data rate (see “PLL Considerations”). The additional circuitry is similar to having multiple PLLs within the transceiver architecture. This leads to a significant increase in die area and, therefore, power consumption.

Because the new protocols require greater layers of integration, the new generations of protocols based on higher data rates also have an impact on the digital blocks within the transceiver. Standards bodies are still scoping specifications for future backplane standards, which are likely to support 11.1-Gbps data interfaces. At these data rates, it might be necessary to change encoding schemes from the standard 8B/10B, used in many of today's standards, to schemes such as 64B/66B.

While 8B/10B encoding provides a good solution, it also adds a 25% overhead into the data stream, because every 8-bit character is encoded to 10 bits. At higher data rates, a number of protocols, for example 10-Gbps Ethernet, are moving to 64B/66B encoding schemes. Although they have the same properties as 8B/10B encoding schemes, they require significantly less data overhead. Unfortunately, 64B/66B encoding is relatively complex, so if it is implemented within the transceiver it can take up significant die area within the transceiver itself, thereby adding a further power requirement.

A further consideration with power is the data rate itself. Because:

$$\text{Power} = \text{Capacitance} \times \text{Voltage}^2 \times \text{Frequency}$$

any increasing system frequency has a direct effect on system power. Operating the system at a slower data- or edge-switching rate helps to reduce the total power budget. For example, with Stratix GX devices

1 transceiver operating at 1.25 Gbps = 120 mW

1 transceiver operating at 3.125 Gbps = 200 mW

Based on VOD = 400 mV Pre-emphasis = 0

The same principles follow as data rates increase further. In order to facilitate the higher data rates, designers must design transceivers with the requisite specifications at those higher data rates. From a transmitter PLL perspective, that translates to voltage controlled oscillator (VCO) design and sufficient buffer strength to drive the clock distribution network. In addition, the CDR must be programmable to be wide ranging, depending on the architecture. From the transmitter perspective, the deterministic jitter is partially determined by the intrinsic parasitic and the amount of drive power. However, at the lower data rates, this power consumption becomes a penalty. High-speed systems exhibit sufficient deterministic jitter that present signal integrity issues not only from inter-symbol interference but several other sources internal to the transceiver. Both on- and off-chip considerations to meet faster data rates mean increased power requirements and sometimes additional area consumption that does not offer the customer the most power and area-efficient solution. Finally, designers should remember that transceiver developments generally move to lower feature sizes (technology) as data rates increase. As a result, it is not obvious that power consumption penalties exist when migrating to higher data rates.

Most applications today utilize transceiver operation between 1 and 3 Gbps. Future road maps suggest many of the next-generation interconnectivity standards will require operation between 5 and 6 Gbps. It therefore follows that most next-generation applications will be captured by a transceiver operating between 622 Mbps and 6.375 Gbps.

Most applications will not require 10-Gbps transceivers until beyond 2008. This is partly because of the slow emergence and high cost of infrastructure, including connectors, development tools, and test equipment and a lack of clarity on backplane standards and transceiver design attributes. Applications using this data rate concentrate on line-side applications that typically use SHD/SONET or 10-Gbps Ethernet-based proto-

cols, which on average require only a single channel within a transceiver. Currently, this type of application is better addressed in a dedicated 10-Gbps transceiver external to the FPGA because of the following:

- It is easier to manage the exhaustive jitter requirements of the protocol specifications.
- The majority of customers that don't require 10 Gbps do not need to make the power and cost trade offs.
- The overall system performance of the FPGA does not become hampered by the bandwidth fMAX requirements of a single transceiver bus interface into the FPGA.

Embedded transceivers within the FPGA can provide a more complete solution at lower data rates, while removing the complexity required to support higher data rates and protocols. This is important, because most of the complexity will be redundant for most applications. Altera's next-generation FPGAs with embedded transceivers will follow this approach.

The transceiver design will be targeted at applications operating between 155 Mbps and 6.375 Gbps. This allows transceiver architects to concentrate on producing a device with exceptional jitter performance across the entire data range. It also ensures that power consumption is manageable. When comparing the Altera solution against a competitor's 10-Gbps solution using actual test chip results, there is a clear difference between the device architected for 6.375 Gbps and the device architected for 10 Gbps. Table 1 shows the power dissipation for Stratix GX, Stratix II GX, and a competitor's solution.

Table 1. Power Dissipation Comparison

Device	3.125 Gbps	6.375 Gbps
Stratix GX	200 mW -	
Stratix II GX (1)	125 mW	300 mW
V4 FX (2)	400 mW	670 mW

Notes to Table 1:

(1) Numbers anticipated from test chip results.

(2) Numbers taken from device user guide.

On a single channel, the difference of 250 mW between the Altera 6.375-Gbps solution and the competitor's solution is significantly large; multiplied by 20 channels, this is as much as 5W @ 3.125 Gbps. This is a considerable power penalty for a user who does not need the extra features of the transceiver.

PLL CONSIDERATIONS

Generally, transceivers are designed to transmit and receive data across an imperfect link. The transceiver architect must design a device capable of working at various operating speeds across various operating conditions. Techniques such as pre-emphasis and equalization can be added to the transceiver to help overcome transmission line losses, which are a major cause of inter-symbol-interference (ISI) or deterministic jitter. However, the PLL must also be specifically architected to manage transmit jitter and random jitter components. This management is increasingly difficult across wider data ranges, and although possible, can lead to an increase in complexity and the die area of the PLL.

TRANSMIT PLL

Achievable bit error rate (BER) largely depends on the quality of the transmitted data. Two major influences on data quality are the deterministic jitter components discussed earlier and the random jitter components. The random jitter compo-

nents seen at the near end of the link (at the transmitter) are primarily associated with the transmitter PLL. This random jitter component can be controlled for a specific data rate by designing the transmitter PLL jitter generation to be minimized for that specific data rate. This is managed by carefully designing the filter components of the PLL for the given range. The filter bandwidth is limited, so if the PLL is pushed to operate over a wide area, the upper and lower data rates in the range see more jitter.

Figure 1 shows the characteristics of a PLL when the VCO is optimized to operate at 6.375 Gbps. The normalized jitter generation becomes worse as the data rate is increased above the optimal data rate because a higher percentage of the signal's unit interval is made up of noise. This behavior is also apparent in the wide data range VCO design's phase noise, which also contributes to the overall jitter.

This data range can be widened in the PLL design by using multiple bands (or filters) within the PLL across a number of "specific" narrow frequencies or data rates. The narrow bands are configured inside the PLL to provide the full data bandwidth of the PLL. The bands are applied by changing divider ratios inside the PLL or by providing additional VCOs, both of which add to the die area of the transceiver.

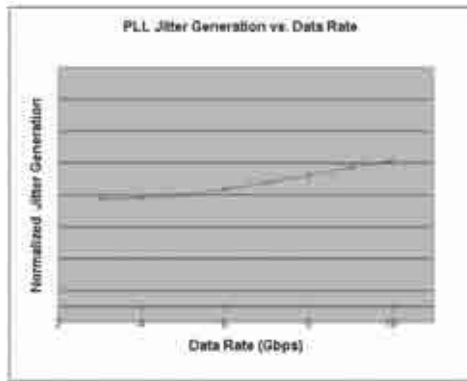


Figure 1. PLL Random Jitter Versus Data Rate

The banding method is beneficial to the transceiver architecture and allows for a widening of the data rate. However, as the level of banding increases to cover the transceiver range, it can be more efficient to use a multiple PLL architecture. With each PLL covering different data rates, this method reduces the complexity of a single PLL architecture and allows additional flexibility in the type of PLL architecture used.

A side effect of both methods is that the jitter performance of the transceiver can vary across its data range. A transceiver operating and characterized successfully at 10 Gbps uses different PLLs or bandings at, for example, 2 Gbps, so jitter performance will differ. Therefore, results seen at 10 Gbps will not necessarily correlate to a better margin at 2 Gbps because the components of the PLL will be different. Jitter performance can be better managed with a dual PLL structure because it allows different oscillators to be used at various frequencies.

In summary, the same PLL is used at 6 Gbps and 10 Gbps, which can be susceptible to higher jitter when out of optimal speed. Or, two different PLLs can be used, which means performance at one data rate does not correlate to another data rate. In either case, good performance at a higher data rate does not relate to better jitter margin at the lower data rate because the PLL characteristics may be different.

RECEIVER PLL

The bit error rate also depends on the receiver's ability to

recover the data in the CDR under jittered conditions. The CDR cannot track jitter beyond its bandwidth well, so the bandwidth must be sufficiently large enough to meet the most tasking or highest data-rate application's jitter tolerance mask. Unlike conventional PLLs, a property of CDR transceivers is that as data rate decreases, jitter tolerance dips. When noise is present at the input of a standard PLL or a CDR PLL, they both behave in the same way, that is, within the loop bandwidth they follow the noise and outside of the bandwidth they do not. For a standard PLL, the relation between the jitter transfer peaking and tolerance dip is correct as shown in Figure 2. For the CDR, the relation becomes much more complex because the single most important factor at high frequencies is the ability to follow high input jitter slope. However, PLL limitations apply and there is a limit to the slope or bandwidth that the CDR can follow. This translates to large phase differences between the recovered clock and the data, creating bit errors if the jitter tolerance mask is not met for a specific standard. Ideally, there is a margin above the mask to allow for any deterioration of the recovered signal caused across the transmission line.

The CDR for any given data rate experiences differences in the number of transitions in the data stream leading to this phenomenon and, incidentally, less stability in the loop. To make the CDR capable of accepting lower data rates it is again necessary to add more bands to the PLL.

Figure 2 shows the receiver jitter tolerance penalty for operating a high data rate CDR at lower frequencies. Frequency response has been normalized to allow both high and low data rate curves to be shown together.

TRANSCIVER COSTS

As this paper discussed, supporting wide high-speed data

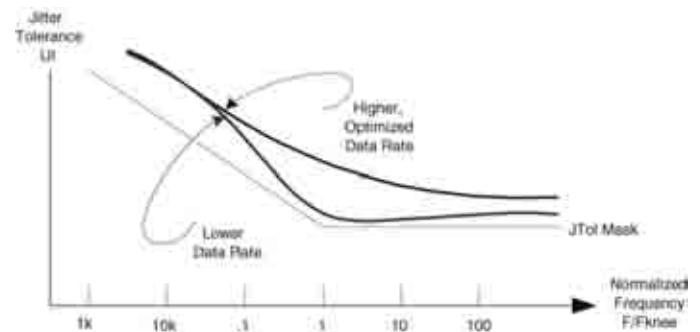


Figure 2. Receiver Jitter Tolerance Penalty for Operating the High Data Rate CDR at Lower Frequencies

rates requires extra flexibility and functionality within the transceiver so that it is as suitable for operation at 622 Mbps as it is at 10 Gbps. These requirements can significantly increase the area of the transceiver, causing it to take up more silicon die area. Even with shrinking process technology, second- and third-generation transceiver blocks are bigger in die area than their predecessors. The increase in die area directly impacts the cost of the device to the customer.

At 10 Gbps, many of these blocks are necessary, such as when the device is used in a 10 Gbps line-module application. However, a customer using a transceiver for a single-port PCI Express application requires a low-cost solution and may be unable to accept the additional cost of the additional features.

CONCLUSION

Many applications are now switching to transceiver technology. The mainstream requirement still remains at approximately the 3-Gbps data point. Although attention is turning to next-generation technology, currently most applications are addressed by a transceiver operating at up to 6.375 Gbps. Situations where higher speeds are required can be addressed in specialist devices designed to operate at a specific data rate. Providing a transceiver at a higher data rate does not guarantee delivery of better jitter performance. Jitter is a difficult problem to manage in transceiver design and it is not solved by higher transceiver speeds. The PLL must be architected to perform well across the entire data range, but stretching the range increases the PLL design challenges.

Transceivers within FPGAs can be architected to operate beyond 6.375 Gbps, but the impact on the transceiver complexity results in an increase in die area, which has a major impact on cost and, more importantly, power, even when the majority of applications do not need this extra performance. Altera's solution is to deliver a low-power solution suited to the broad base of applications while at the same time delivering a low jitter solution.

PROGRAMMABLE LOGIC CONTROLLERS: LADDER LOGIC

Courtesy of All About Circuits

PROGRAMMABLE LOGIC CONTROLLERS

Before the advent of solid-state logic circuits, logical control systems were designed and built exclusively around electromechanical relays. Relays are far from obsolete in modern design, but have been replaced in many of their former roles as logic-level control devices, relegated most often to those applications demanding high current and/or high voltage switching.

Systems and processes requiring “on/off” control abound in modern commerce and industry, but such control systems are rarely built from either electromechanical relays or discrete logic gates. Instead, digital computers fill the need, which may be programmed to do a variety of logical functions.

In the late 1960s an American company named Bedford Associates released a computing device they called the MODICON. As an acronym, it meant Modular Digital Controller, and later became the name of a company division devoted to the design, manufacture, and sale of these special-purpose control computers. Other engineering firms developed their own versions of this device, and it eventually came to be known in non-proprietary terms as a PLC, or Programmable Logic Controller. The purpose of a PLC was to directly replace electromechanical relays as logic elements, substituting instead a solid-state digital computer with a stored program, able to emulate the interconnection of many relays to perform certain logical tasks.

A PLC has many “input” terminals, through which it interprets “high” and “low” logical states from sensors and switches. It also has many output terminals, through which it outputs “high” and “low” signals to power lights, solenoids, contactors, small motors, and other devices lending themselves to on/off control. In an effort to make PLCs easy to program, their programming language was designed to resemble ladder logic diagrams. Thus, an industrial electrician or electrical engineer accustomed to reading ladder logic schematics would feel comfortable programming a PLC to perform the same control functions.

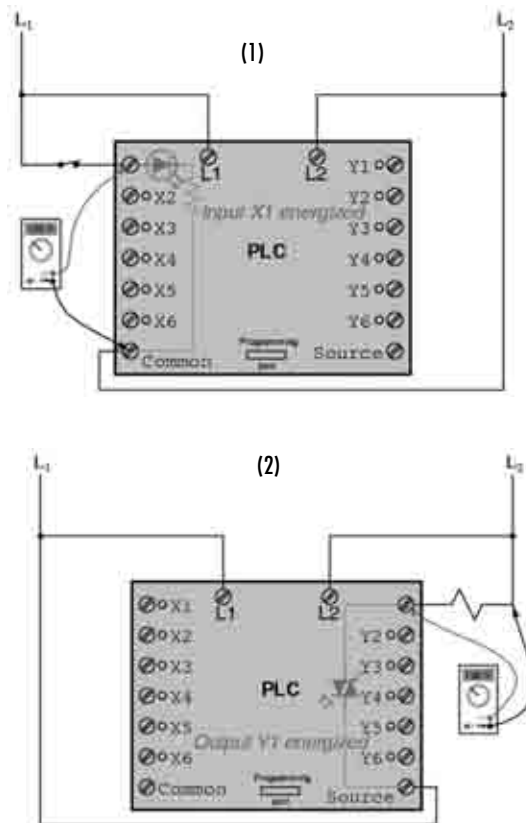


PLCs are industrial computers and, as such, their input and output signals are typically 120 volts AC, just like the electromechanical control relays they were designed to replace. Although some PLCs have the ability to input and output low-level DC voltage signals of the magnitude used in logic gate circuits, this is the exception and not the rule.

Signal connection and programming standards vary somewhat between different models of PLC, but they are similar enough to allow a “generic” introduction to PLC programming here. The following illustration shows a simple PLC, as it might appear from a front view. Two screw terminals provide connection to 120 volts AC for powering the PLC’s internal circuitry, labeled L1 and L2. Six screw terminals on the left-hand side provide connection to input devices, each terminal representing a different input “channel” with its own “X” label. The lower-left screw terminal is a “Common” connection, which is generally connected to L2 (neutral) of the 120 VAC power source.

Inside the PLC housing, connected between each input terminal and the Common terminal, is an opto-isolator device (Light-Emitting Diode) that provides an electrically isolated “high” logic signal to the computer’s circuitry (a photo-transistor interprets the LED’s light) when there is 120 VAC power applied between the respective input terminal and the Common terminal. An indicating LED on the front panel of the PLC gives visual indication of an “energized” input: (1)

Output signals are generated by the PLC’s computer circuitry activating a switching device (transistor, TRIAC, or even an electromechanical relay), connecting the “Source” terminal to any of the “Y-” labeled output terminals. The “Source” terminal, correspondingly, is usually connected to the L1 side of the 120 VAC power source. As with each input, an indicating LED on the



front panel of the PLC gives visual indication of an “energized” output: (2)

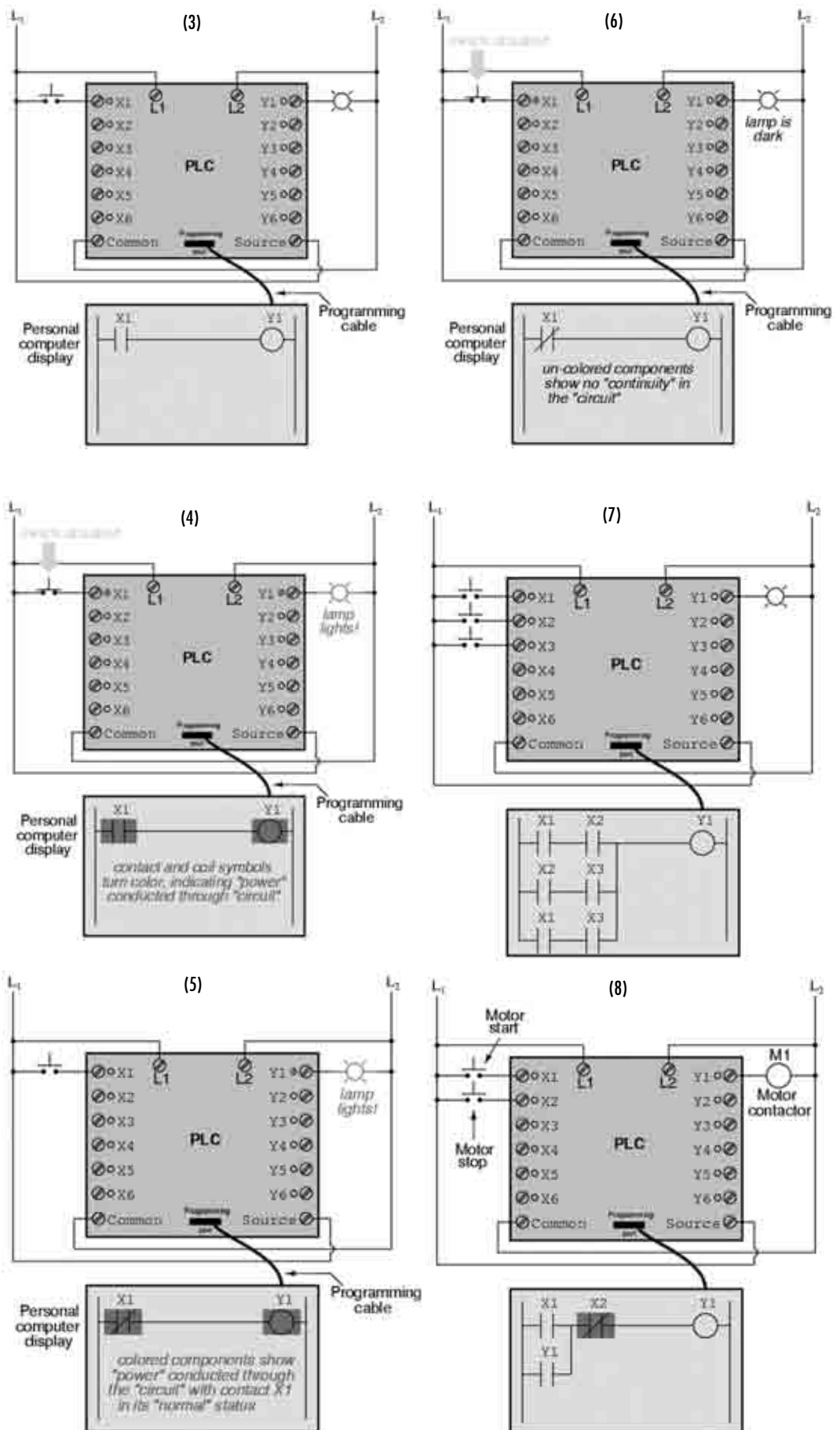
In this way, the PLC is able to interface with real-world devices such as switches and solenoids.

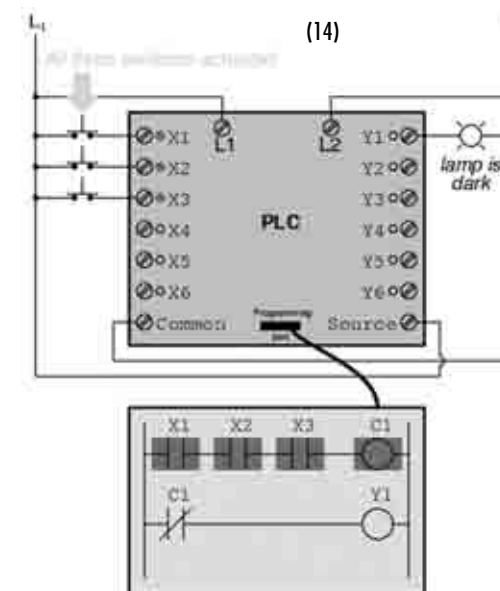
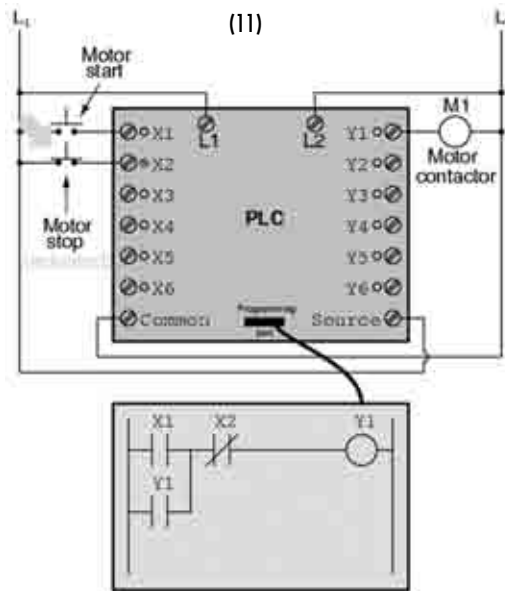
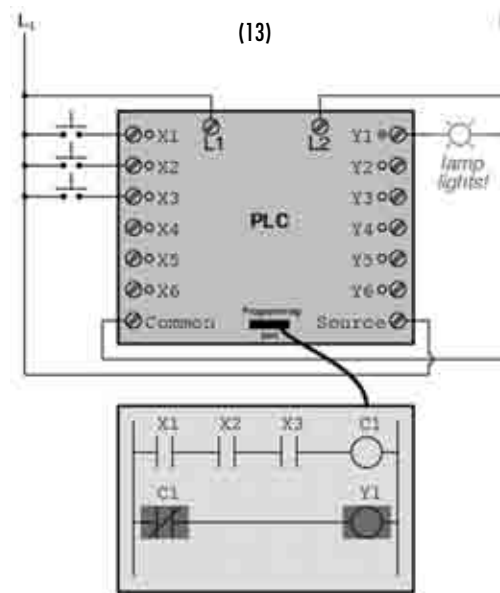
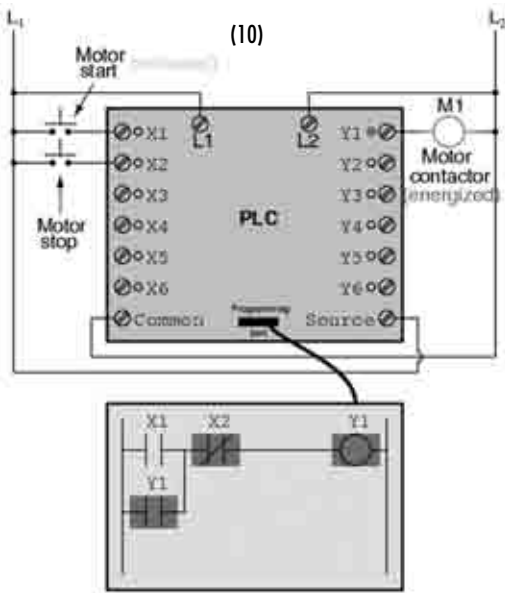
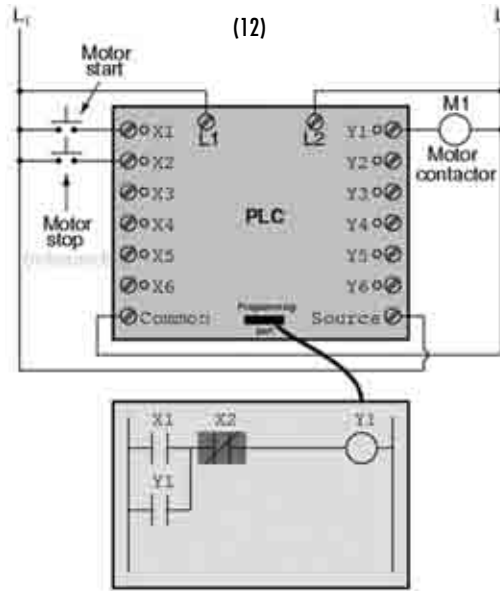
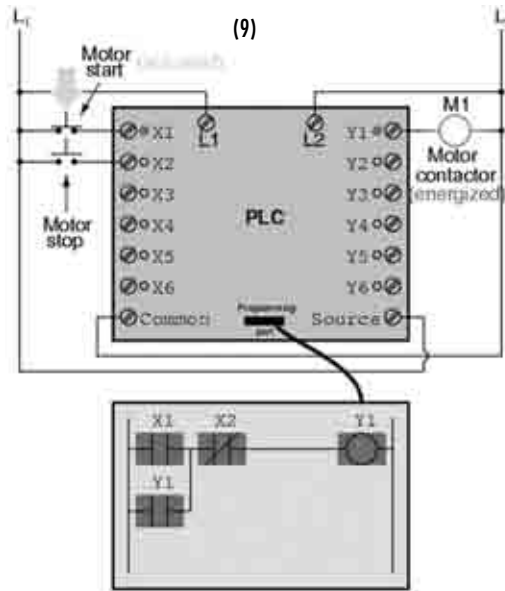
The actual logic of the control system is established inside the PLC by means of a computer program. This program dictates which output gets energized under which input conditions. Although the program itself appears to be a ladder logic diagram, with switch and relay symbols, there are no actual switch contacts or relay coils operating inside the PLC to create the logical relationships between input and output. These are imaginary contacts and coils, if you will. The program is entered and viewed via a personal computer connected to the PLC's programming port.

Consider the following circuit and PLC program: (3)

When the pushbutton switch is unactuated (unpressed), no power is sent to the X1 input of the PLC. Following the program, which shows a normally-open X1 contact in series with a Y1 coil, no "power" will be sent to the Y1 coil. Thus, the PLC's Y1 output remains de-energized, and the indicator lamp connected to it remains dark.

If the pushbutton switch is pressed, however, power will be sent to the PLC's X1 input. Any and all X1 contacts appearing in the program will assume the actuated (non-normal) state, as though they were relay contacts actuated by the energizing of a relay coil named "X1". In this case, energizing the X1 input will cause the normally-open X1 contact will "close", sending "power" to the Y1 coil. When the Y1 coil of the program "energizes", the real Y1 output will become energized, lighting up the lamp





connected to it: (4)
 It must be understood that the X1 contact, Y1 coil, connecting wires, and “power” appearing in the personal computer’s display are all virtual. They do not exist as real electrical components. They exist as commands in a computer program — a piece of software only — that just happens to resemble a real relay schematic diagram.

Equally important to understand is that the personal computer used to display and edit the PLC’s program is not necessary for the PLC’s continued operation. Once a program has been loaded to the PLC from the personal computer, the personal computer may be unplugged from the PLC, and the PLC will continue to follow the programmed commands. I include the personal computer display in these illustrations for your sake only, in aiding to understand the relationship between real-life conditions (switch closure and lamp status) and the program’s status (“power” through virtual contacts and virtual coils).

The true power and versatility of a PLC is revealed when we want to alter the behavior of a control system. Since the PLC is a programmable device, we can alter its behavior by changing the commands we give it, without having to reconfigure the electrical components connected to it. For example, suppose we wanted to make this switch-and-lamp circuit function in an inverted fashion: push the button to make the lamp turn off, and release it to make it

turn on. The “hardware” solution would require that a normally-closed pushbutton switch be substituted for the normally-open switch currently in place. The “software” solution is much easier: just alter the program so that contact X1 is normally-closed rather than normally-open.

In the following illustration, we have the altered system shown in the state where the pushbutton is unactuated (not being pressed): (5)

In this next illustration (at right), the switch is shown actuated (pressed): (6)

One of the advantages of implementing logical control in software rather than in hardware is that input signals can be re-used as many times in the program as is necessary. For example, take the following circuit and program, designed to energize the lamp if at least two of the three pushbutton switches are simultaneously actuated: (7)

To build an equivalent circuit using electromechanical relays, three relays with two normally-open contacts each would have to be used, to provide two contacts per input switch. Using a PLC, however, we can program as many contacts as we wish for each “X” input without adding additional hardware, since each input and each output is nothing more than a single bit in the PLC’s digital memory (either 0 or 1), and can be recalled as many times as necessary.

Furthermore, since each output in the PLC is nothing more than a bit in its memory as well, we can assign contacts in a PLC program “actuated” by an output (Y) status. Take for instance this next system, a motor start-stop control circuit: (8)

The pushbutton switch connected to input X1 serves as the “Start” switch, while the switch connected to input X2 serves as the “Stop”. Another contact in the program, named Y1, uses the output coil status as a seal-in contact, directly, so that the motor contactor will continue to be energized after the “Start” pushbutton switch is released. You can see the normally-closed contact X2 appear in a colored block, showing that it is in a closed (“electrically conducting”) state.

If we were to press the “Start” button, input X1 would energize, thus “closing” the X1 contact in the program, sending “power” to the Y1 “coil”, energizing the Y1 output and applying 120 volt AC power to the real motor contactor coil. The parallel Y1 contact will also “close”, thus latching the “circuit” in an energized state: (9)

Now, if we release the “Start” pushbutton, the normally-open X1 “contact” will return to its “open” state, but the motor



will continue to run because the Y1 seal-in “contact” continues to provide “continuity” to “power” coil Y1, thus keeping the Y1 output energized (above right): (10)

To stop the motor, we must momentarily press the “Stop” pushbutton, which will energize the X2 input and “open” the normally-closed “contact,” breaking continuity to the Y1 “coil:” (11)

When the “Stop” pushbutton is released, input X2 will de-energize, returning “contact” X2 to its normal, “closed” state. The motor, however, will not start again until the “Start” pushbutton is actuated, because the “seal-in” of Y1 has been lost: (12)

In addition to input (X) and output (Y) program elements, PLCs provide “internal” coils and contacts with no intrinsic connection to the outside world. These are used much the same as “control relays” (CR1, CR2, etc.) are used in standard relay circuits: to provide logic signal inversion when necessary.

To demonstrate how one of these “internal” relays might be used, consider the following example circuit and program, designed to emulate the function of a three-input NAND gate. Since PLC program elements are typically designed by single letters, I will call the internal control relay “C1” rather than “CR1” as would be customary in a relay control circuit: (13)

In this circuit, the lamp will remain lit so long as any of the pushbuttons remain unactuated (unpressed). To make the lamp turn off, we will have to actuate (press) all three switches, like this: (14)

This section on programmable logic controllers illustrates just a small sample of their capabilities. As computers, PLCs can perform timing functions (for the equivalent of time-delay relays), drum sequencing, and other advanced functions with far greater accuracy and reliability than what is possible using electromechanical

logic devices. Most PLCs have the capacity for far more than six inputs and six outputs. The following photograph shows several input and output modules of a single Allen-Bradley PLC.

With each module having sixteen “points” of either input or output, this PLC has the ability to monitor and control dozens of devices. Fit into a control cabinet, a PLC takes up little room, especially considering the equivalent space that would be needed by electromechanical relays to perform the same functions: (photos on this page)

One advantage of PLCs that simply cannot be duplicated by electromechanical relays is remote monitoring and control via digital computer networks. Because a PLC is nothing more than a special-purpose digital computer, it has the ability to communicate with other computers rather easily. The following photograph shows a personal computer displaying a graphic image of a real liquid-level process (a pumping, or “lift,” station for a municipal wastewater treatment system) controlled by a PLC.

The actual pumping station is located miles away from the personal computer display.

ETHERNET/IP: INDUSTRIAL PROTOCOL

By Paul Brooks

European Marketing Manager, Logix/NetLinx Technology Adoption, Rockwell Automation

Abstract – DeviceNet and ControlNet are two well-known industrial networks based on the CIP protocol (CIP = Control an Information Protocol). Both networks have been developed by Rockwell Automation, but are now owned and maintained by the two manufacturer’s organizations ODVA (Open DeviceNet Vendors Association) and ControlNet International. ODVA and ControlNet International have recently introduced the newest member of this family – EtherNet/IP (“IP” stands for “Industrial Protocol”). This article describes the techniques and mechanisms that are used to implement a fully consistent set of services and data objects on a TCP/UDP/IP based Ethernet network.

I. INTRODUCTION

Automation architectures must provide users with three primary services. The first, control, is the most important. Control services involve the exchange of time-critical data between controlling devices such as PLCs and I/O devices such as variable speed drives, sensors and actuators. Networks that are tasked with the transmission of this data must provide some level of priority setting and/or interrupt capabilities. Second, networks must provide users configuration capabilities to set up and maintain their automation systems. This functionality typically involves the use of a personal computer (PC) or equivalent tool for programming of various devices in the system. This can be performed at commission, and also during runtime, such as recipe management in batch operations. Lastly, an automation architecture must allow for collection of data for the purposes of display in MMI stations, analysis and trending, and/or troubleshooting and maintenance. Networks that can provide all three services – control, configuration, and collection of data – deliver the greatest amount of flexibility and efficiency for better overall system performance.

Networks that are based on the producer/consumer model – where data is identified, rather than tied to explicit source and destination addresses – can support control, configuration, and collection of data services. Application layers using distributed objects and producer/consumer communication services meet the requirements of automation architectures.

In providing these services (Figure 1 shows a typical automation architecture), it cannot be assumed that a single network level will meet all application requirements as each network’s physical and data link layers have their own attributes and benefits. Where a multi-level network structure is required,

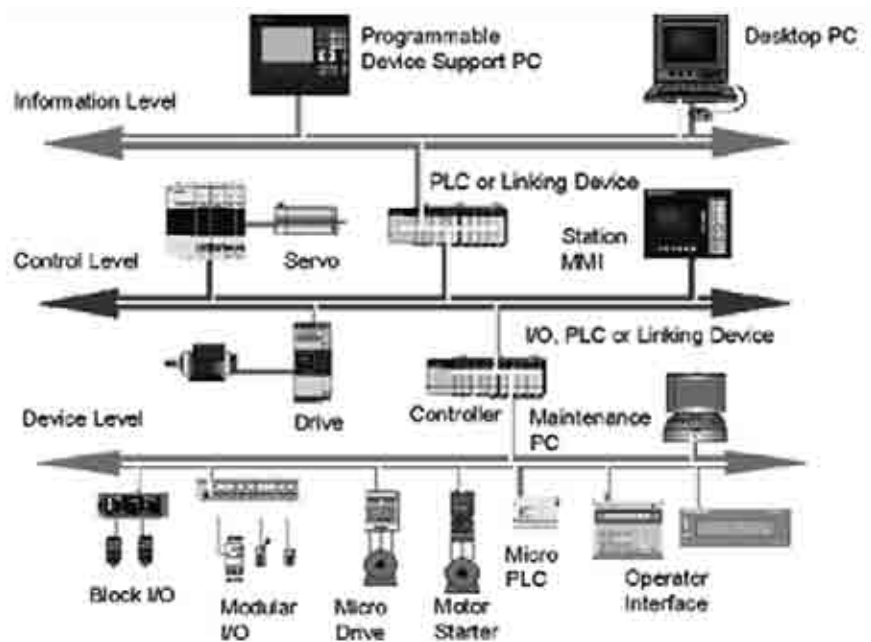


Figure 1: Typical Multi-Level Automation Architecture

then the network architecture must provide consistency of data between disparate network segments.

Similarly, where these services are provided on an Ethernet network, it cannot be assumed that other services will be not required on that network segment. The producer/consumer services must be fully coexistent with other services that may exist on the network segment (e.g. http for web pages).

The typical architecture shown in Figure 1 above includes an information level network can be provided for by an Ethernet network segment, which most controller vendors have been providing for many years. It has a control level, which traditionally has valued attributes such as hard determinism and media redundancy (such as ControlNet from ControlNet International) and a device level requiring low data volumes with power and data provided in a single robust network cable (such as DeviceNet from ODVA).

One member of the CIP family – EtherNet/IP (“IP” stands for “Industrial Protocol”) implements the full suite of control, configuration, and data collection data services on an Ethernet network, and can thus be used for both the information and control levels in the typical architecture shown in Figure 1.

II. CIP IMPLEMENTATION ON ETHERNET

The EtherNet/IP Specification is available for free download from the ODVA and ControlNet International web sites in

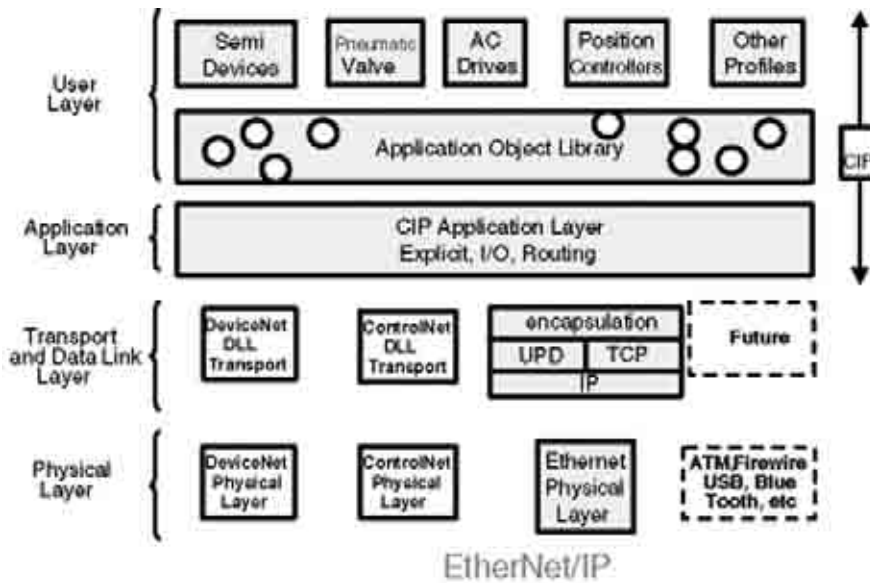


Figure 2: EtherNet/IP Scope of Specification

addition to the EtherNet/IP website. Apart from being subdivided into several chapters and appendices, the following features are described in the document:

- Object Modeling
- Explicit and Implicit (I/O) Messaging
- Communication Objects
- General Object Library
- Device Profiles
- Electronic Data Sheets (EDSs)
- Explicit Messaging Services
- Data Management

As can be seen from Figure 2, the application layer, application object libraries and device profiles are consistent between EtherNet/IP, DeviceNet and ControlNet. It is only the lowest 4 layers of the OSI 7-layer model (Figure 3) that are network dependent.

It is, however, the way that those layers are used that, in turn, determine the optimum implementation of the collection of data, configuration of devices, and control services on EtherNet/IP and that make it practical and safe to use EtherNet/IP at the control level.

III. COEXISTENCE WITH INTERNET (AND OTHER) PROTOCOLS

The primary benefit that most users of EtherNet/IP will be hoping to gain will be the leveraging of Ethernet training and knowledge within their enterprise and maximizing the return on their investments in Ethernet infrastructures.

Similarly, they will be looking to make best and most effective use of commercial off-the-Shelf (COTS) Ethernet components that are available today from a wide variety of competing manufacturers to drive down the cost of ownership of their network infrastructures.

These benefits could not be realized if EtherNet/IP required a separate, custom built (i.e. non-generic) network infrastructure with vendor specified physical media. Similarly, they could not be realized if EtherNet/IP required dedicated net-

work installations to operate with minimal or no connection to the remainder of the corporate network infrastructure. Compatibility with existing Internet and intranet protocols is a must. That means that TCP/IP will be everywhere, so we will start there.

COMMUNICATION PROTOCOLS USED WITH ETHERNET

Ethernet technology by itself provides a set of physical media definitions, a scheme for sharing that physical media (CSMA/CD), and a simple frame format and source/destination addressing scheme for moving packets of data between devices on a LAN. By itself, Ethernet lacks the more complex features required of a fully functional LAN. For that reason, all installed Ethernet networks support one or more communication protocols that run on top of Ethernet and provide sophisticated data transfer and network management functionality. It is the communication protocol that determines what level of functionality is supported by the network, what types of devices may be connected to the network, and how devices interoperate on the network.

Some of the protocols that have been implemented over Ethernet are DECnet, Novell IPX, MAP, TOP, the OSI stack, AppleTalk, and TCP/IP. Of these, TCP/IP is receiving the most attention due to the emergence of the global Internet (including the World Wide Web) as well as the corporate intranets that are transforming how corporations distribute information internally. TCP/IP is the protocol of the Internet. Although TCP/IP will run on physical media other than Ethernet, and Ethernet supports other communication protocols, the two have become increasingly linked due to the desire of organizations to seamlessly integrate their internal intranets with the global Internet. It is safe to say that TCP and IP are now, or will soon be, the dominant "middle layer" protocols (see Figure 3) running on Ethernet networks on the factory floor as well.

TCP/IP ORIGIN AND FEATURES

Over the years, TCP/IP has been ported to every major computer platform in the world. It is currently embedded in every copy of the Windows operating system distributed, making it a popular choice for networking PCs. Many organizations have a mix of workstations, networked printers, servers, and mid-range and mainframe computers, rarely provided by a single vendor. TCP/IP is available for all of these devices and is a logical choice for integrating these devices into a LAN.

TCP/IP is a layered protocol that can be mapped approximately to the OSI 7-layer network model (see Figure 3). On this diagram, Ethernet represents layers 1 (physical) and 2 (data link). The Internet protocol (IP) maps to layer 3 (network). The TCP and UDP transports map to layer 4 (transport). The user services commonly associated with TCP/IP networks map to layer 7 (application). The TCP/IP protocol suite has no specific mapping to layers 5 and 6 of the model.

Each layer of the OSI model uses the services provided by the layer immediately below it. For example, when a TCP connection needs to send a packet of data to another device over

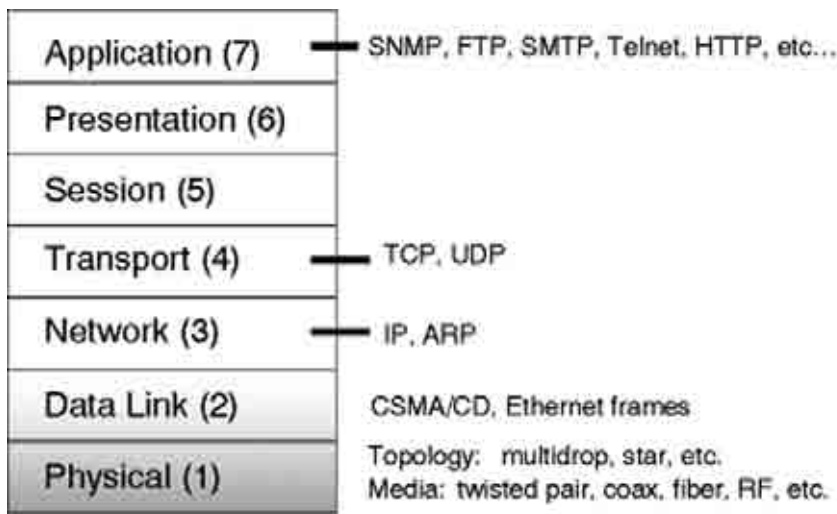


Figure 3: OSI Seven Layer Model

Ethernet, it passes the packet to IP for transmission. IP then handles the interface to Ethernet and ensures that the packet gets transmitted onto the Ethernet network to the destination device. On the receiving end, the IP layer receives the packet from the Ethernet interface and passes it to the appropriate TCP connection within the receiver.

The lowest layer of TCP/IP is the network layer, where IP resides. IP provides a connectionless and unacknowledged method of sending data packets (datagrams) between two devices on a network. IP does not guarantee delivery; it relies on a transport-layer or application-layer protocol to do that. IP can run over Ethernet or over a variety of other LAN or WAN (wide area network) technologies. That is one reason why IP can seamlessly move data from corporate intranets to the global Internet.

The network layer is also where the Address Resolution Protocol (ARP) resides. ARP is used to map Ethernet addresses to IP addresses and to maintain mapping tables in each device on the network. When a device wants to transmit an IP datagram to another device, it attempts to look up the Ethernet address corresponding to that IP address in its internal ARP table. If no such address exists, the ARP protocol is used to query the network via a local broadcast message to ask the device with the corresponding IP address to return its Ethernet address to the sender. The response is placed in an internal table, and used for subsequent communication. It is important to note that Ethernet broadcast messages pass through hubs and switches and bridges, but they do not pass through routers. Therefore broadcast messages are confined to Ethernet subnets, and do not propagate onto the worldwide Internet.

IP addresses are 32-bit quantities that are administered by InterNIC, an independent authority, and must be unique on any network. Any device that wishes to communicate outside of a closed corporate LAN must use an IP address from the assigned block. Unlike Ethernet addresses, which are fixed in the Ethernet hardware by the manufacturer and cannot be changed, the user, in accordance with the local policies of their corporate Information Systems department, configures IP addresses and subnets. IP addresses may be changed, but doing so must be done in a carefully planned manner to avoid breaking existing network applications that assume a particular

device is located at a particular IP address.

If a LAN is connected to the global Internet via a router, then its IP addresses must follow the rules regarding use of IP addresses from the assigned block. Corporate LANs that are not connected to the Internet at any point can follow their own rules regarding assignment of IP addresses. IP addresses are being depleted as the popularity of the global Internet grows. Steps are being taken to implement a new form of IP address (known as IPv6) that will extend the number of bits used (to 48 bits) and therefore the number of IP addresses available.

Like Ethernet addresses, IP addresses can be unicast (single destination), multicast (group destination) or broadcast addresses (received by everyone). IP addresses must be mapped to the proper supporting Ethernet address type by the IP software and Ethernet driver.

The transports supported by the TCP/IP protocol suite are TCP (Transmission Control Protocol) and UDP (User Datagram Protocol). They both map to the transport layer of the OSI model. TCP is a connection-oriented transport that provides reliable transmission of data from one device to another. Once a TCP connection is established between two devices, TCP handles fragmentation and re-assembly of message packets, detects failures, performs retries, and generally provides a high quality of service between the two devices. TCP guarantees the data will get from one device to the other if it is possible. If the transmission fails for any reason, TCP ensures that the applications on both ends of the TCP connection know it. TCP presents data to the application layer above it in the form of a continuous byte stream. The receiving application must be capable of recognizing any message delimiters that might be embedded in the byte stream by the transmitting application.

TCP works only in unicast (point-to-point) mode, and is used by applications such as Telnet (terminal emulation), FTP (File Transfer Protocol), and HTTP (Web Server). In an industrial automation application, TCP is typically used to download ladder programs between a workstation and a PLC, for MMI software that reads or writes PLC data tables, or for peer-to-peer messaging between two PLCs.

UDP is a much simpler transport protocol. It is connectionless and provides a very simple capability to send datagrams between two devices. It does not guarantee that the data will get from one device to another, does not perform retries, and does not even know if the target device has received the data successfully. Application layers that implement their own handshaking or connection management between two devices and, therefore, only need a minimal transport service, use UDP. For instance, UDP is used by applications such as SNMP and NFS. UDP is smaller, simpler and faster than TCP due to its minimal capabilities and use of resources. UDP can operate in unicast, multicast or broadcast mode. In an industrial automation application, UDP is typically used for network management functions, applications that do not require reliable data transmission or applications that are willing to implement their own reliability scheme, such as a flash memory programming of network devices.

The protocols and applications that comprise the TCP/IP protocol suite are all documented in Request for Comment (RFC) documents that are maintained by the Internet

Engineering Task Force (IETF). The IETF is an independent organization that acts as a standards organization for Internet protocols. The RFCs are open and are distributed freely, and can be obtained without charge from the IETF Website.

THE APPLICATION LAYER AND INTEROPERABILITY

The TCP/IP protocol suite provides a set of services that two devices may use to communicate with each other over an Ethernet LAN or over a wide area network that spans the globe. However, using TCP/IP alone does not guarantee that the two devices can communicate effectively, if at all; it only guarantees that application-level messages will be successfully transferred between the two devices.

Effective communication between two devices requires that the application software on both sides be compatible. The applications in both devices must understand the attributes and services provided by the other and that they use a common messaging scheme to communicate over TCP/IP (or UDP/IP). The RFCs for popular Internet applications such as FTP, HTTP, Telnet, SNMP, SMTP (e-mail) fully document how those applications should behave. Any vendor that follows those RFCs should produce applications that can communicate with their counterparts on another device, even if another vendor developed that device. The ability for devices from different vendors to communicate up through the application layer is called interoperability.

Although common services for file transfer (FTP), terminal emulation (Telnet), e-mail (SMTP) and others have been established under the guidance of the IETF, the situation is not so simple in the area of industrial automation. Each vendor of automation equipment that runs over Ethernet TCP/IP has implemented its own application layer protocol. As a result, equipment from different automation vendors connected to the same plant-floor intranet can physically coexist on the LAN but cannot interoperate. PLCs from one vendor cannot readily share information with PLCs from another vendor over a TCP/IP connection, nor can vendor A's workstation software download programming or configuration information into vendor B's device. This lack of interoperability makes it very difficult for customers to integrate Ethernet-based automation equipment from different vendors within the same application on the same Ethernet network.

Consequently, we can see from Figure 4 that the EtherNet/IP protocol can coexist with any other protocol that is running on top of the standard TCP/UDP Transport layers.

ETHERNET TCP/IP IN INDUSTRIAL AUTOMATION TODAY

Today, a typical installation of an Ethernet TCP/IP network may extend plant-wide and be connected to a corporation's worldwide network via the Internet. It is generally used to conduct program maintenance, send data to and from MIS and MES systems, serve up intranet web pages, perform supervisory control, provide connectivity for operator interfaces, and log

Control and Information Protocol

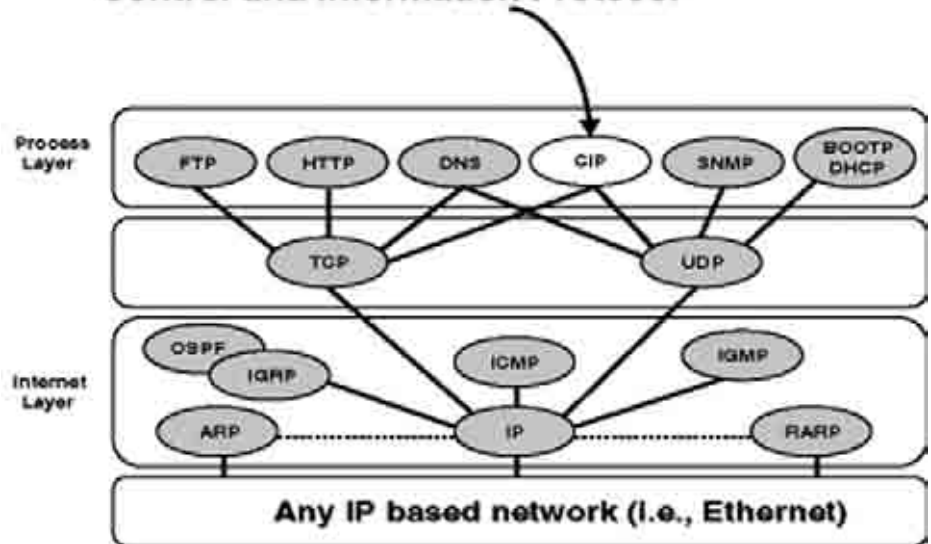


Figure 4: CIP Application Layer Coexistence

events and alarms. These functions require the high throughput and widespread accessibility that Ethernet offers. Response time is secondary to overall data capacity. Some customers currently use Ethernet for limited control purposes, such as processor data sharing, but the applications where this has been successful take advantage of Ethernet's high capacity, but do not require a high level of determinism or repeatability of message response time.

This is a clear example of the collection of data and configuration services offered by EtherNet/IP that are available across the Internet.

IV. DATA EXCHANGE ON AND BETWEEN NETWORKS

While EtherNet/IP provides the capability to collect data from devices directly on the Ethernet network and to configure those devices in real-time, it cannot be assumed that a single network will supply all needs. Individual vendors may not have an EtherNet/IP connection available for their device. It is unlikely that it will be cost-effective in the near future to embed a full EtherNet/IP connection into simple devices like photoelectric cells and inductive proximity switches.

This does not mean that the user should be prohibited from making use of EtherNet/IP as the primary point of contact into the target network. On the contrary, the user should be able to work on the remote network as if he were locally connected. Further, this should be independent of any application level programming or intermediate computer devices.

To achieve this, networks at all levels (see Figure 1) must implement a common set of services, and all devices must organize their data into a common object model. Once this consistency of data has been achieved, then a mechanism must be defined for routing data between networks.

OBJECT ORIENTED DATA STRUCTURES

The future paradigm for the Internet is one of distributed objects communicating in a peer-to-peer fashion within corporate intranets and across the Internet. Competing "middleware" standards like DCOM and CORBA may differ in implementation but agree on the distributed object approach to interoperability. The advantage of a distributed object architecture is that

it will enable both software developers and end users to enjoy a simple, object-oriented, network-wide interface to data in their network devices that appears to be independent of the physical location of the data. Details of network addressing and internal device data structures will be transparent to the users who will access data using an object naming and addressing scheme that will conceal such details from them.

Aging paradigms that rely on simple source/destination messaging will not thrive in the future Internet environment where plant-floor Ethernet devices will be required to interoperate with information applications, as well as support control, often on the same network. Customers will require that devices from different vendors interoperate on the same network. Achieving this goal will require the use of an application protocol that has several features that are critical needs if universal interoperability over Ethernet TCP/IP is to be realized:

- layered on TCP/IP and UDP/IP
- implements a distributed object model
- based on an open industry standard
- provides an efficient model for I/O messaging
- allows control and information to coexist on the same Ethernet network
- meets the diverse requirements of the industrial automation industry
- is accepted and implemented by multiple automation vendors.

GENERAL OBJECT LIBRARY

The CIP family of protocols contains a fairly large collection of commonly defined objects (currently 46 object classes). Only a few of these object classes (1 for DeviceNet, 3 for ControlNet, and 2 for EtherNet/IP) are specific to the individual link layer; all others are common objects that can and will be used in all three networks.

Further objects are added according to the functionality of the device type. This allows very good scalability of devices, e.g. a proximity sensor on DeviceNet is not burdened with unnecessary overhead. A developer typically uses publicly defined objects, but can also create his own objects in the vendor specific addressing range, e.g. class ID 100 – 199 in the 8-bit object class code space. However, it is strongly encouraged to work in the Special Interest Groups (SIGs) of ODVA and ControlNet International to create common definitions for further objects instead of inventing private ones.

As an example of a required public object, the instance attributes of the identity object (class code: 1) are described in the table below.

IDENTITY OBJECT	
Mandatory Attributes	Optional Attributes
A. Vendor ID	A. State
A. Device Type	A. Configuration Consistency Value
A. Product Code	A. Heartbeat Interval
A. Revision	
A. Status	
A. Serial Number	
A. Product Name	

Typically, devices do not change their identity, so all attributes (with the exception of the Heartbeat Interval attribute) are read-only.

ELECTRONIC DATA SHEETS

Having a consistent object model is not helpful if there is no mechanism of identifying which objects have been implemented in a device to external applications. CIP has made provisions for several options to configure devices:

- A printed data sheet
- Parameter Objects and Parameter Object Stubs
- An Electronic Data Sheet (EDS)
- A combination of an EDS and Parameter Object Stubs
- A Configuration Assembly and any of the above methods

When using configuration information collected on a printed data sheet, configuration tools can only provide prompts for service, class instance and attribute data and relay this information to a device. While this procedure can do the job, it is the least desirable solution since it does not determine the context, content, or format of the data.

Parameter objects, on the other hand, provide a full description of all configurable data of a device. This allows a configuration tool to gain access to all parameters and maintain a user-friendly interface since the device itself provides all the necessary information. Attributes include the data type, engineering units, minimum, maximum and default values, scaling factors, whether it is non-volatile, read and/or write.

However, this method burdens a device with full parameter information, and this may be too much for a small device, e.g. a simple DeviceNet slave. Therefore, an abbreviated version of the parameter object, called parameter object stub may be used. This still allows access to the parameter data, but it does not describe any meaning of this data. This is where an EDS is very handy. An EDS supplies all the information that a full parameter object contains on top of what the parameter object stub provides. The combination of EDS and parameter object stub thus provides the full functionality and ease of use of the parameter object without burdening the individual devices.

Finally, a configuration assembly allows the bulk upload and download of a full block of parameters.

MESSAGING PROTOCOL

As can be seen from the object model shown in figure 5, access to the internal object model of any device is controlled by one of two objects, the unconnected message manager and the connection manager.

This is because EtherNet/IP is a connection-based network. A CIP connection defines a packet that will be produced on the network. When a connection is established, the transmissions associated with those connections are assigned a Connection ID (CID). If the connection involves a bi-directional exchange, then two Connection ID values are assigned. (see Figure 6).

Since most messaging on a CIP network is done through connections, a process has been defined to establish such connections between devices that are not “connected” yet. This is done through the Unconnected Message Manager (UCMM), which is responsible for processing the connection requests. Once a connection has been established, then all communication resources needed in the devices including any intermediate CIP bridges/routers are reserved. And the overall network loading and bandwidth required for that data exchange is minimized.

All connections in a CIP network can be divided into explicit messaging connections and implicit (or I/O) messaging

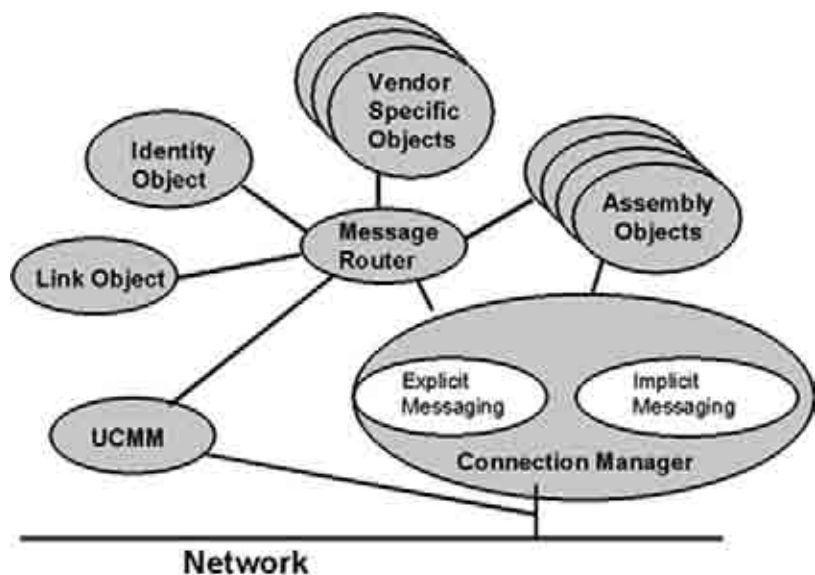


Figure 5: CIP Object Model

connections.

- Explicit messaging connections provide generic, multi-purpose communication paths between two devices. These connections are often referred to as just messaging connections. Explicit messages provide the typical request/response-oriented network communication and are always made to the message router (Object). Each request contains explicit information that the receiving node decodes, acts upon, and then generates an appropriate response.

- Implicit messaging connections provide dedicated, special purpose communication paths (ports) between a producing application object and one or more consuming application objects. This type of messaging is always used for application-specific I/O data which moves through these ports and so the messaging is often referred to as I/O connections. However, there are many more applications for implicit messaging. They are called implicit messages because the data that will to be exchanged is identified at the time that the connection is established and ConnectionIDs are assigned. Each transmission contains the current values for the application object(s) that was agreed upon when the connection was established. In other words, the meaning of the data is implicitly defined by the ConnectionID.

Both of these types of connections can be bridged between networks and will be discussed in more detail later.

EXPLICIT CONNECTIONS

As is stated above, all explicit connections are direct connections between two devices, which require a source address, a destination address, and a ConnectionID in each direction. Explicit messages are triggered by events external to the application layer of the CIP protocol.

This is true of DeviceNet and ControlNet in which the

source and destination addresses are both node numbers on the network and of EtherNet/IP where the source and destination address are both IP addresses.

However, the CIP frame exists within a TCP packet and may include additional information about the destination nodes – the communication path and equally important, which ‘hop’ in that path the current frame is taking.

Hence, taking the typical automation architecture in Figure 1, a message initiated at the programmable device support PC connected to the information level network and targeted at the motor starter on the device level network will have at least three ‘hops’; one onto the information level network, one onto the control level network, and a final hop onto the device level network. Throughout each of these hops, the CIP frame remains intact throughout this entire journey, while existing consecutively in a TCP packet, a ControlNet packet, and a CAN packet.

The only requirement on the motor starter is to ensure that the path remains intact in the CIP frame and addresses the return message in a CAN packet to the node number in the return path. Whether the originating node is physically on the same network, bridged to a local EtherNet/IP network, or even routed to a remote location across the Internet is transparent to the target device.

Given that the motor starter in this example has been designed in accordance with the DeviceNet specification, and the programming terminal designed in accordance with the EtherNet/IP specification, the 2 devices have a common understanding of the way that the data in the other is organized.

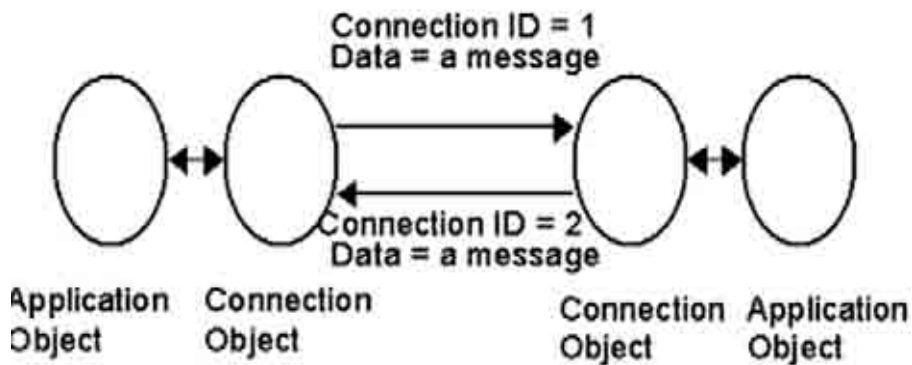


Figure 6: Connections and Connection IDs

As discussed earlier, one of the mandatory objects in all devices is the identity object, and mandatory attributes of that object include vendor ID, device type, product code and revision. This data can be queried from a target node without knowing physically what that device is before the message is issued. From this data, it is possible to uniquely identify the EDS file for the device and thus know which public objects, have been implemented and in many cases what vendor specific objects may have been implemented. For devices that contain full parameter objects it is possible to get this data directly from the

device without an EDS. These mechanisms are target network independent – as can be seen from Figures 2 and 5 the data objects are isolated from the network such that the same message can be issued to get to the same data object not only independently of the device but also independently of the network connection.

EtherNet/IP, being based on TCP/IP provides a further potential. In this example, there is no requirement on the source node being at the information level. A PLC sitting on a control level network, which is isolated from the information level by a PLC or bridge device (it is not important whether the control level network is EtherNet/IP or ControlNet) can originate a multi-hop message which uses the information level as one of the intermediate steps. This allows two PLCs connected on ControlNet, on opposite sides of the world, to communicate using explicit messages across the Internet.

V. I/O CONTROL OVER ETHERNET/IP ETHERNET AS A CONTROL NETWORK?

One of the most common arguments that traditionally have been used against the use of Ethernet for control is that Ethernet is non-deterministic. Determinism enables users to accurately predict the worst-case data transmission. But users also need high repeatability (or low jitter); that is a guarantee of its arrival at the same time every time (or to quickly recognize that it did not arrive and take appropriate action). The improvements in Ethernet technology detailed below have improved the determinism, repeatability, and performance of Ethernet to a great extent. Switches break up collision domains into single devices or small groups of devices, effectively reducing the number of collisions to almost zero. CSMA/CD provides the mechanism for detecting and recovering from contention for the network when it does occur. Furthermore, there are efforts in place to create a prioritization scheme for messages over Ethernet (IEEE 802.1p) that if implemented inside switches and TCP/IP stacks could potentially be used to prioritize control/alarm message packets over programming/data packets or routine network diagnostic traffic (SNMP).

However, all of these are untried technologies in high-speed control applications. In many applications with sensitive timing, a single message received later than anticipated can shut down the process, resulting in lost production or even damaged goods and equipment. Variable packet latency or dropped packets within Ethernet switches could potentially cause this to happen. Losing a hub or switch in an information only application may result in lost production data; losing one in a control application can result in lost production and possible damage to the production equipment itself. These and other issues must be carefully considered by users in order to rationally determine the types of control applications for which TCP/IP Ethernet technology is a good or even an acceptable solution.

B. THE EVOLUTION OF SWITCHING TECHNOLOGY

In recent years, both repeater hub technology and Ethernet bridge technology have been supplanted by a new technology that uses high-speed switching techniques to allow traffic between any two ports on the switch to pass through the switch with an extremely low latency on the order of microseconds. This technology has been enabled by specialized hardware that can support a very high bandwidth backplane within the device. The speed of the backplane is typically greater than the sum of the speeds of the Ethernet ports on the device and can

accommodate all of the ports running at full speed without collisions. Furthermore, these new devices are capable of buffering frames temporarily to handle short-term contention for the same output port.

These new devices are called switching hubs, layer 2 switches, or simply switches. In fact, a switch is a multiport bridge. Every port on a switch is its own collision domain, so collisions between devices attached to the switch do not occur. Furthermore, each port on a switch can usually be configured to run at half duplex (traditional Ethernet) or at full duplex operation. Full duplex provides an effective 10 Mbit/sec connection in each direction (20 Mbit/sec total) between an attached device and the switch. For fast Ethernet, the full duplex speed is 100 Mbit/sec in each direction (200 Mbit/sec total). Like traditional bridges, switches build and maintain internal tables that map Ethernet addresses to ports. A packet received on one port is rapidly “switched” to the appropriate output port, typically within microseconds.

Advanced switches support a virtual LAN (VLAN) feature that allows users to configure the switch so that ports are subdivided into groups such that all packets received on one port of a group will only be transmitted to another port within the group. The receiving port and the group of transmitting ports constitute a VLAN. VLANs may typically be overlapped within a switch, such that any one port may appear on multiple VLANs. This feature allows the user a great deal of flexibility over partitioning the ports on a switch into multiple overlapping collision domains.

Switches are capable of handling a greater throughput than repeater hubs without experiencing the collision-induced delays that can be experienced by devices on a repeater hub as network traffic increases. This makes them a good choice for replacing repeater hubs on loaded networks that are experiencing an unacceptable level of collision-induced delays. Although switches are currently more expensive than repeater hubs, their cost is dropping and will soon be low enough that switches will likely replace repeater hubs as the network concentrator of choice for all Ethernet networks, not just those used for control.

It is important to note that switches do have some performance limitations that may affect some applications. If a switch experiences internal congestion due to message packets on multiple input ports contending for transmission on the same output port, the switch may simply drop packets. Or it may force a collision back to the transmitting devices, so they back off long enough for the congestion to clear. The approach that is taken depends upon the implementation chosen by the switch vendor. In either case, a variable latency is inserted into the message stream, which is generally not a problem for office applications but may have profound impact on industrial automation applications.

Although switches isolate separate collision domains on each port, they do not create separate broadcast domains. However, each VLAN is a separate broadcast domain, if this feature is enabled on the switch. An Ethernet broadcast message that is received on any port will be re-transmitted on all switch ports to all attached devices. This means that switches do not eliminate the problem of excessive broadcast traffic that can cause severe performance degradation across an entire Ethernet network when a damaged or improperly configured device is attached to the network. Some switch vendors are working on proprietary methods for suppressing excessive broadcast messages in their switches, but this is not universal. Broadcast mes-

sages are common on Ethernet networks that carry the TCP/IP protocol because Ethernet broadcast messages are used by TCP/IP for address resolution. However, broadcast traffic represents a small percentage of network traffic on a network that is properly configured and operating normally.

Also, switches and repeater hubs are active devices, containing complex digital circuitry and requiring power (AC in most cases) to operate. The failure of a switch or hub will effectively cause a communication failure for all of the devices attached to that device's ports, including other hubs or switches that may be attached to one or more ports of the failed device. The devices attached to the failed hub or switch will be unable to communicate with the rest of the plant network until the switch is replaced or repaired. Furthermore, most Ethernet media components have been designed for use in an office or light industrial environment. They have not been designed and tested for compliance to the rigorous environmental standards typical of industrial control devices (i.e., extended temperature range, industrial CE mark, shock and vibration, etc.). This may become an issue as the mission for Ethernet on the plant floor is expanded into new areas.

THE EVOLUTION OF ETHERNET PERFORMANCE

More recent developments in Ethernet technology include Fast Ethernet and Gigabit Ethernet. Fast Ethernet is defined and documented in IEEE specification 802.3u. Fast Ethernet is basically Ethernet running at 100 Mbits/sec. Fast Ethernet and 10 Mbit Ethernet use the same frame structure, addressing scheme, and CSMA/CD access method. However, all network timing parameters must be scaled by a factor of 10 when configuring a Fast Ethernet network. This tends to reduce the distances between nodes in some configurations when compared to a 10 Mbit network.

Fast Ethernet provides a wire speed that is 10 times as fast as traditional Ethernet, which tends to benefit bandwidth hungry applications, such as video and audio transmission, as well as the transfer of large data files over the network. However, most applications will not enjoy a substantial performance increase due to increased wire speed alone. In particular, a plant-floor network of small microprocessor-based intelligent I/O blocks, sensors, actuators, drives and other device interfaces are likely to consume and produce small amounts of data encapsulated in 64 byte Ethernet frames (the smallest frame size supported by Ethernet). The performance of these devices is more likely to be limited by the speed of their microprocessor and embedded firmware than the wire speed. It is unlikely a network of such devices would fully utilize the full 10 Mbit/sec Ethernet bandwidth, unless an inefficient application layer protocol was utilized that repeatedly polled the devices in point-to-point fashion.

One area of performance wherein 100 Mbit Ethernet may show noticeable improvement over 10 Mbit Ethernet is in the area of collision recovery. As mentioned earlier, the backoff times for 100 Mbit Ethernet are one tenth of those for 10 Mbit Ethernet. On a loaded network where collisions are an issue, 100 Mbit Ethernet may show noticeably better performance than 10 Mbit Ethernet. Additionally, it would be expected that a 100 Mbit Ethernet network would be able to handle a larger offered load than a 10 Mbit Ethernet network before collisions became an issue. If the application requires the use of multiple switches, the links between the switches may benefit from the higher speed. However, if loading and collisions are not already

an issue on a 10 Mbit Ethernet network, simply upgrading to 100 Mbit Ethernet may not show sufficient improvement to justify the investment.

IMPLICIT (I/O) MESSAGING OVER ETHERNET/IP

In section 4.5, we discussed the communication path and the use of explicit messages and unconnected messages to exchange point-to-point messages between nodes.

The second type of messaging, implicit messaging is used where the exchange of data between nodes is transparent to the user and takes place within the application layer of the protocol, with both producing and consuming nodes aware of the content of the message before transmission. While commonly used for I/O messages, these make full use of the producer/consumer model and are commonly used for scheduled communication between controllers as well.

There are four principal types of implicit message available within the CIP specification:

- Polled
- Change of State
- Cyclic
- Strobed

Polled messages have largely the same attributes as any 'old fashioned' I/O network, in which the scanner (master) device sequentially queries all of the adapter (slave) devices by sending them their output data and receives a reply with their input data.

Strobed is a special case of polled in which the scanner sends out a single multicast request for data and the slaves sequentially reply with their data with no further messages required from the master.

Cyclic messages are produced by a device on a pre-determined scheduled basis, with a connection ID associated with the message. Any other device that requires the data from the producing device is made aware of the connection ID and accepts any packets that it sees on the network with this connection ID.

Change of State is similar to Cyclic, except that (as the name implies) data is produced in response to an event which caused the data to change, rather than a timed event. Change of State also maintains a background cyclic rate (heartbeat) so that consuming applications can know that the node is still online and functioning.

Of these 4, Cyclic is the preferred implicit message exchange format on an EtherNet/IP network, providing a balance between data integrity and network traffic optimization.

For the implementation of the CIP protocol on Ethernet, the critical aspect of an implicit message is that there can be many consumers of a single packet of data on the wire. This makes the use of TCP packets, which are for point-to-point applications impossible. Nor is it desirable to flood the network with broadcast packets that cannot be rejected at the IP layer and are likely to overload the terminal devices.

UDP/IP packets support multicast operations and have the added benefit of being the 'thinnest' application layer and thus requiring the smallest amount of processing time in the terminal device.

For typical applications, it is anticipated that connections will run as frequently as low single-digit millisecond periodicity.

UDP packets are not transmitted directly to the 'true' IP address of the receiving device, but rather are transmitted with a

specific device allocated IP multicast address. This address is used in parallel with (one-to-one correspondence) the CIP connection ID in the EtherNet/IP implementation, allowing packets that are not relevant to a specific node to be filtered prior to presentation at the application layer.

The consuming device must be made aware of this IP multicast address (which has been allocated by the producer) before it can use the produced data.

To achieve this, the UnConnected Message Manager must be used (see Figure 5).

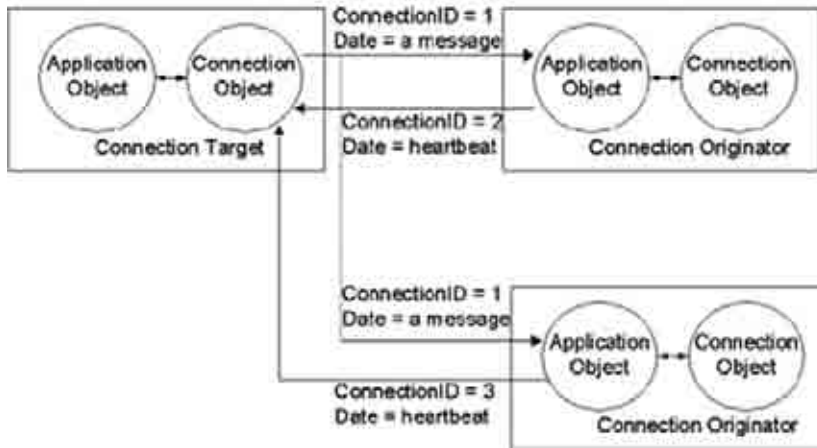


Figure 7: Multicast Implicit Connection

A point-to-point (TCP) packet is transmitted from the connection originator (the PLC in an I/O configuration; the consumer in a PLC-to-PLC or equivalent application), which indicates the data object that the connection originator wishes to receive and the rate at which it wishes to receive the data.

The connection manager object is now interrogated to identify if there is a match in its connection table to the data object and periodic rate. If there is a match, then the data object is already being produced (i.e. it will be multicast) and the Connection ID and related multicast IP address will be returned to the prospective consumer (see Figure 7 above). If there is no match, then a UDP related IP address and Connection ID will be allocated and loaded into the connection manager object. The data will start being produced and can be consumed by any device cognizant of its multicast IP address and Connection ID.

The final piece to this is that there must be a mechanism to shut the connection down and that mechanism must operate when the consumer is no longer connected to the network. As UDP and IP are unacknowledged transmission mechanisms, the producer has no way of knowing if the consumer is online and receiving the data. To achieve this, the data producer must reverse the process and establish a special cyclic connection to each of the consuming devices. There is no application data transmitted through this connection, which is called a 'heartbeat.' If the producing node times out all of the heartbeat con-

nections that are associated with a specific produced data object, then all connections associated with that data object are closed.

Consequently, by use of TCP packets to establish the connection between devices and then UDP connections to pipe the I/O data objects, network bandwidth utilization is minimized.

With the advent of 100MBps Ethernet, this is not as critical as it once was. More significantly, the number of packets that each terminal node has to process is minimized, and, thus, its ability to handle implicit connections is maximized.

VI. CONFORMANCE TESTING

As interoperability is one of the principal goals of both ODVA and ControlNet International in generating the EtherNet/IP specification, those organizations have put a number of independent conformance test facilities in place, with one based in Europe, one in North America, and one in Japan. A joint special interest group (JSIG) has been set up between ODVA and ControlNet International to ensure that consistent test procedures are run across these labs.

By providing these independent facilities, it will be practical for competitive vendors to offer products with no fear of loss of intellectual property but with the confidence that the products will integrate seamlessly.

This will, in turn, ensure that users are able to make decisions based on the merits of individual components and suppliers rather than feel tied

to a single vendor.

The first of these labs, at the University of Michigan, USA was opened August 10, 2001. This is to be followed by similar units at the University of Warwick, UK and the Advanced Software Technology and Mechatronics Research Institute (ASTEM RI) in Kyoto, Japan.

VII. BENEFITS OF ETHERNET/IP

(See Figure 8.) Because ControlNet, DeviceNet and EtherNet/IP use a common application layer protocol, they also

share an object library and device profiles. These objects and profiles allow for plug-and-play interoperability among complex devices from multiple vendors. The object definitions are rigorous and support real-time I/O messaging, configuration and diagnostics over the same network. This means that users can connect to complex devices like drives, robot controllers, bar code readers, and weigh scales without custom software. The result is faster start ups and superior diagnostics.

In addition, EtherNet/IP provides users with both explicit (information) and implicit (control) messaging services. EtherNet/IP, as a result, supplies every service that is essential in control and device-level networks – from polled, cyclic, and change-of-state trigger mechanisms



Figure 8: EtherNet/IP Benefits

to point-to-point and multicast data transfer.

Finally, given the rapid adoption of ControlNet and DeviceNet, nearly 400 vendors from across the globe have already developed more than 500 interoperable products for any of the three networks. This is important if only to illustrate that support for EtherNet/IP is unparalleled and will only continue to grow.

VIII. CONCLUSION

Three technological advances: the use of 100MBps Ethernet, the use of switches and full duplex operation of terminal devices, have reduced both the probability of collisions and the consequences of them to the point of making I/O control over Ethernet a manageably low risk option.

The global acceptance of Ethernet TCP/IP has made it a popular choice for many end users and for a wide variety of network applications. It offers an abundance of compatible products, high data throughput, and commercially available components at relatively low costs. The future paradigm for Ethernet is one of distributed objects communicating in a peer-to-peer fashion, within corporate intranets and across the Internet. In this environment, plant-floor Ethernet devices will be required to interoperate with corporate information applications, as well as support control, often on the same network. Customers will require that devices from different vendors interoperate on the same network. Achieving this goal will require the adoption of an Application Layer that:

- is layered on TCP/IP (and UDP/IP)
- implements a distributed object model
- allows control and information to coexist on the same Ethernet network
- provides producer/consumer network services
- meets the diverse requirements of the industrial automation industry
- is accepted and implemented by multiple automation vendors

This is a critical need in the industry if real-time control and universal interoperability over Ethernet TCP/UDP/IP is to be realized.

IX. REFERENCES

[1] DeviceNet Specification, Release 2.0, including Errata 4, April 1, 2001, © 1995-2001 by Open DeviceNet Vendor Association.

[2] ControlNet Specification, Release 2.0, including Errata 2, December 31, 1999, © 1998, 1999 by ControlNet International.

[3] EtherNet/IP Specification, Release 1.0, June 5, 2001, © 2000, 2001 by ControlNet International and Open DeviceNet Vendor association.

SCADA: CHOOSING A POLLING MODE FOR DF1 HALF-DUPLEX MASTER

Courtesy Allen-Bradley, Rockwell Automation

A master station can be configured to communicate with slave stations in either Message-based polling mode or Standard polling mode. The pros and cons of each polling mode are described below.

MESSAGE-BASED POLLING MODE

Message-based polling mode is best used in networks when communication with the slave stations is not time critical and where the user needs to be able to limit when and how often the master station communicates with each slave station. It is NOT recommended for systems that require time continuous communication between the master and all the slave stations have MSG instructions in their programs.

With Message-Based polling mode, the only time a master station communicates with a slave station is when a message (MSG) instruction in ladder logic is triggered to that particular slave station's address. This polling mode gives the user complete control (through ladder logic) over when and how often to communicate with each slave station.

If multiple MSG instructions are triggered simultaneously, they will be executed in order, one at a time, to completion (i.e., the first MSG queued up will be transmitted and completed to done or error before the next queued up MSG is transmitted. Refer to appendix E for sample application programs). Any time a message is triggered to a slave station that can't respond (for instance, if its modem fails), the message will go through retries and timeouts that will slow down the execution of all the other queued up messages. The minimum time to message to every responding slave station increases linearly with the number of slave stations that can't respond.

If the Message-based selection is 'don't allow slaves to initiate messages,' then even if a slave station triggers and queues up a MSG instruction in its ladder logic, the master station will not process it. This mode is similar to how a master/slave network based on Modbus protocol would work, since Modbus slave stations cannot ever initiate a message.

If the Message-based selection is 'allow slaves to initiate messages,' when a slave station initiates a message to the master station (polled report by exception messaging) or to another slave station (slave-to-slave messaging), the MSG command packet will remain in that slave station's transmit queue until the master station triggers its own MSG command packet to it (which could be seconds, minutes or hours later, depending on the master's ladder logic).

STANDARD POLLING MODE

Standard polling mode is strongly recommended for larger systems that require time critical communication between the master and all the slave stations, or for any system where slave station-initiated messages are going to be used (this includes slave programming over the network, since this uses the same

mechanism that slave-to-slave messaging uses). The Active Node Table automatically keeps track of which slaves are (and are not) communicating. Standard polling mode should NOT be used in cases where the user needs to be able to limit when and how often the master station communicates with each slave station.

Standard polling mode causes the master station to continuously send one or more 4-byte poll packets to each slave station address configured by the user in the poll list(s) in round robin fashion. As soon as the end of the polling list is reached, the master station immediately goes back and starts polling slave stations from the top of the polling list over again. This is independent and asynchronous to any MSG instructions that might be triggered in the master station ladder logic. In fact, this polling continues even while the master station is in program mode. Refer to chapter 3 of the DF1 Protocol and Command Set Reference Manual, publication 1770-RM516, for additional information.

When a MSG instruction is triggered while the master station is in run mode, the master station will transmit the message packet just after it finishes polling the current slave station in the poll list and before it starts polling the next slave station in the poll list (no matter where in the poll list it is currently at). If multiple MSG instructions have been triggered simultaneously, at least four message packets may be sent out between two slave station polls. Each of these messages will have an opportunity to complete when the master polls the slave station that was addressed in the message packet as it comes to it in the poll list.

If each of the transmitted message packets is addressed to a different slave station, the order of completion will be based upon which slave station address comes up next in the poll list, not the order that the MSG instructions were executed and transmitted in.

When a slave station receives a poll packet from the master station, if it has one or more message packets queued up to transmit (either replies to a command received earlier or MSG commands triggered locally in ladder logic), the slave station will transmit the first message packet in the transmit queue.

If the standard mode selection is 'single message per poll scan', then the master station will go to the next station in the poll list. If the standard mode selection is 'multiple messages per poll scan' the master station will continue to poll this slave station until its transmit queue is empty.

The master station knows the slave station has no message packets queued up to transmit when the slave station responds to the master poll packet with a 2-byte poll response.

Every time a slave station responds or doesn't respond to its poll packet, the master station automatically updates its active node list (again, even if it's in program mode). In this list, one bit is assigned to each possible slave station address (0 to

254). If a slave station doesn't respond when it is polled, its active node list bit is cleared. If it does respond when it is polled, its active node bit is set. Besides being an excellent online troubleshooting tool, two common uses of the active node list are to report good/bad communication status for all slave stations to an operator interface connected to the master station for monitoring, alarming and logging purposes, and to precondition MSG instructions to each particular slave.

This second use is based on the supposition that if a slave station didn't respond the last time it was polled (which was just a few seconds ago, if that long), then chances are it won't be able to receive and respond to a MSG instruction now, and so it would most likely just end up going through the maximum number of retries and timeouts before completing in error (which slows down both the poll scan and any other messaging going on). Using this technique, the minimum time to message to every responding slave station actually decreases as the number of slave stations that can't respond increases.

ABOUT POLLED REPORT-BY-EXCEPTION

Polled report-by-exception lets a slave station initiate data transfer to its master station, freeing the master station from having to constantly read blocks of data from each slave station to determine if any slave input or data changes have occurred. Instead, through user programming, the slave station monitors its own inputs for a change of state or data, which triggers a block of data to be written to the master station when the master station polls the slave.

If your SCADA application is time-critical and any two or more of the following apply, then you can benefit from polled report-by-exception messaging:

- communication channel is slow (2400 bps or less)
- average number of words of data to monitor in each slave station is greater than five

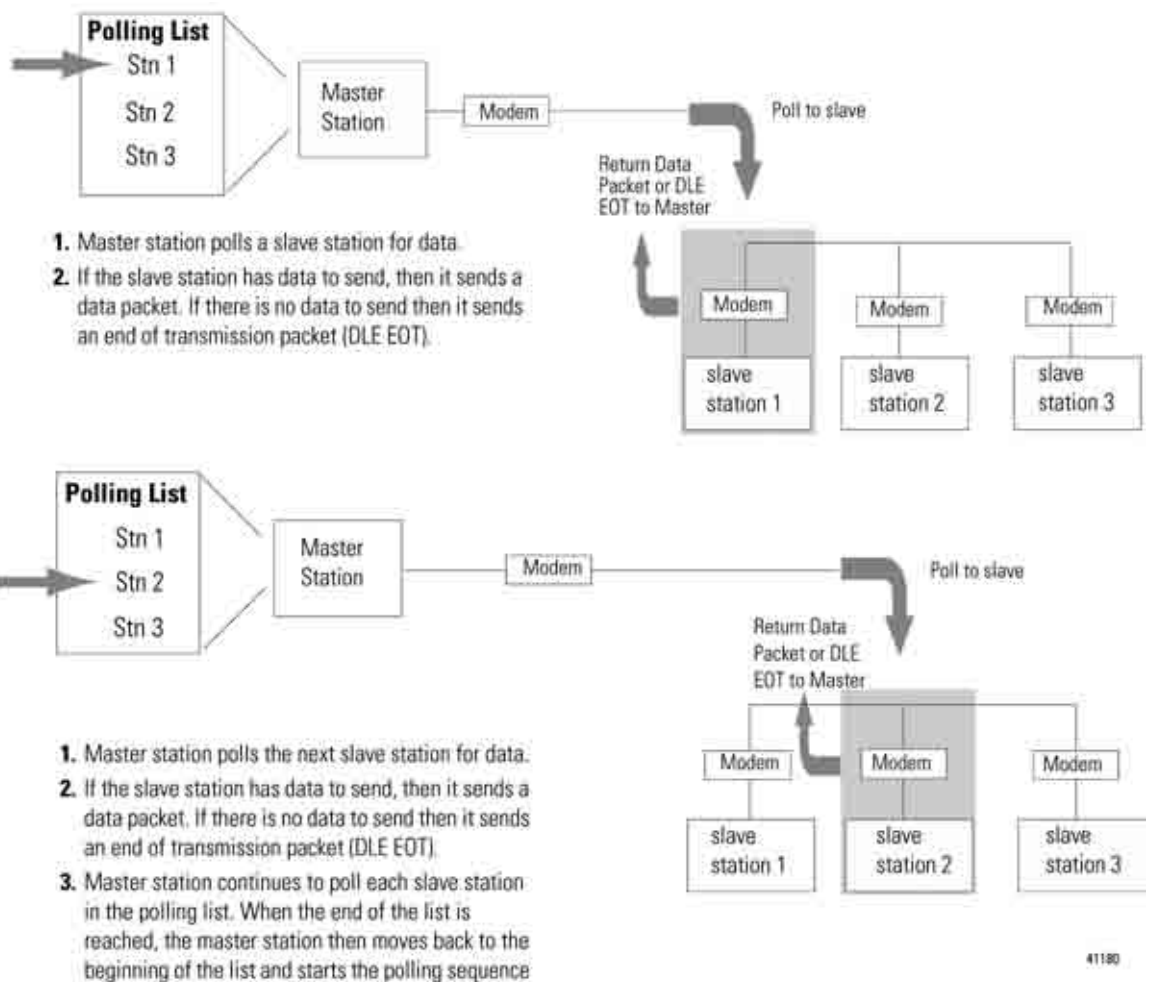


Figure 1.1 Slave Station Polling and Response

- number of slave stations is greater than ten

ABOUT SLAVE-TO-SLAVE MESSAGING

Most SCADA half-duplex protocols do not allow one slave station to talk to another slave station, except through special application-specific code, which requires processing overhead in the master station.

If one slave station has a message to send to another, it simply includes the destination slave station's address in the message instruction's destination field in place of the master station's address when responding to a poll. The master station checks the destination station address in every packet header it receives from any slave station. If the address does not match its own station address, the entire message is forwarded back onto the telemetry network to the appropriate slave station, without any further processing.

ADDRESSING TIPS

Each station on the network including the master station must have a unique address. The address range is 0 to 25410 (3768), so you can have a maximum of 254 stations on a single telemetry network. Station address 25510 (3778) is the broadcast address, which you cannot select as a station's individual address.

A remote programming terminal station address should be reserved, even if remote programming is not considered a requirement initially. This address will need to be periodically polled, even though it will remain on the inactive poll list unless a remote programming terminal is online.

SLC 500 AND MICROLOGIX 1000 PROCESSOR ADDRESSING CONSIDERATIONS

When an SLC 5/02 or MicroLogix 1000 slave station issues a PLC@-2-type message to a PLC-5 master station, the message's destination in the PLC-5 processor's data table is an integer file with the file number equal to the SLC 500 or MicroLogix 1000 processor station address.

An address lower than 9 may interfere with a PLC-5 processor master station since files 0-8 are usually left in their default configuration; file 9 is often used by programmers for the I/O list. Station address 25510 is the broadcast address. So, assign addresses between 1010-25410.

When using an SLC 5/03, 5/04, or 5/05 processor, or a MicroLogix 1100, 1200 or 1500 controller, as a master station, the poll list configuration consists of a contiguous block of addresses. Therefore, assign slave station addresses in a contiguous block in order to avoid polling for nonexistent slave stations.

SLC 500 PROCESSORS WITH A 1747-KE MODULE ADDRESSING CONSIDERATIONS

Since you can have up to 254 devices on a half-duplex network and 32 devices on a DH-485 network, to allow 255 DH-485 nodes requires using a group number. This parameter defines the address group of the SLC 500 half-duplex address. Each address group can consist of 32 addresses.

The slave address of the SLC 500 processor is determined with the following formula: $(32 * G) + A$, where G is the group number (0 to 7) and A is the DH-485 node address of the SLC 500 processor.

One station address within each group of size 32 must be reserved for any 1747-KE modules configured with that group number. A second address within each group should also be reserved for local DH-485 programming terminals. These 16 addresses (two per group) should never have to be polled by the master station.

COMMUNICATION SCHEME DESIGN USING STANDARD-MODE

Standard-communication mode for an Allen-Bradley master station uses centralized polling to gather data from slave stations. A master station using this communication technique

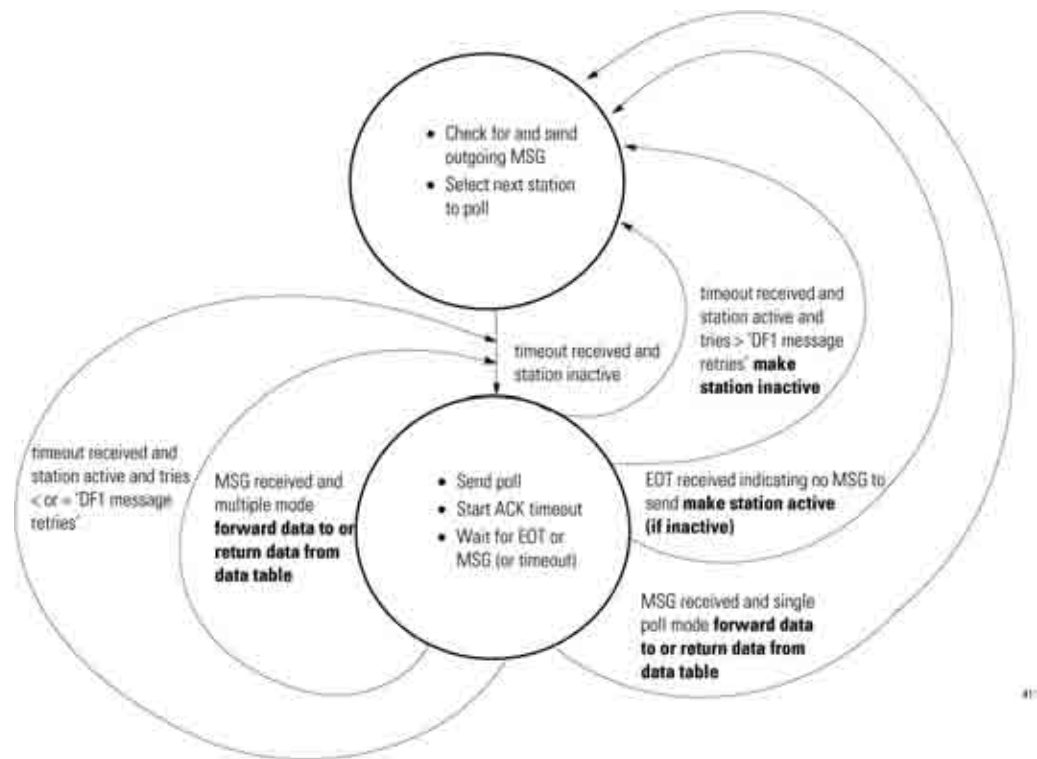


Figure 1.2 Standard Communication Mode

asks (polls) individual slave stations if they have any information to send. All stations on the link 'hear' the master station's requests, but only the slave station to which a request is addressed replies. PLC-5, Logix and RSLinx master stations poll slave stations based on an ordered list (polling list) configured by the system designer. SLC 500 and MicroLogix master stations poll slave stations sequentially in a range of addresses configured by the system designer. Figure 1.1 shows how a slave station gets polled and how it responds.

A master station polls the slave stations in the order the slave stations appear on the list. Slave stations send either a data packet or a packet indicating that the station has no data to send.

1. Master station polls a slave station for data.
2. If the slave station has data to send, then it sends a data packet. If there is no data to send then it sends an end of transmission packet (DLE EOT).

1. Master station polls the next slave station for data.
2. If the slave station has data to send, then it sends a data packet. If there is no data to send then it sends an end of transmission packet (DLE EOT).

3. Master station continues to poll each slave station in the polling list. When the end of the list is reached, the master station then moves back to the beginning of the list and starts the polling sequence over again.

When the master station is configured for standard-communication mode, you do not need to program any master-station message instructions to communicate with slave stations. Communication with slave stations occurs by the master station sending polling packets to slave stations. You only need message instructions when you want the master station to write data to or read data from a location within a slave station's data table.

CHOOSING NORMAL OR PRIORITY POLLING LISTS

Slave stations listed in a priority poll list are polled more frequently than those listed in the normal poll list. Place the slave stations that you need information from more frequently in a priority poll list.

Within each poll list, slave stations are assigned a status, which is either active or inactive. A slave station becomes inactive when it does not respond to a master station's poll packet after the configured number of retries.

If your master station is a Logix controller or PLC-5, you can use application logic to reorder the polling lists and priority while the application logic is executing.

Figure 1.4 and Figure 1.5 show how normal and priority lists relate to one another.

CHOOSING SINGLE OR MULTIPLE MESSAGE TRANSFER

Depending on your application's requirement, you can choose the number of messages you want to receive from a slave station during its turn.

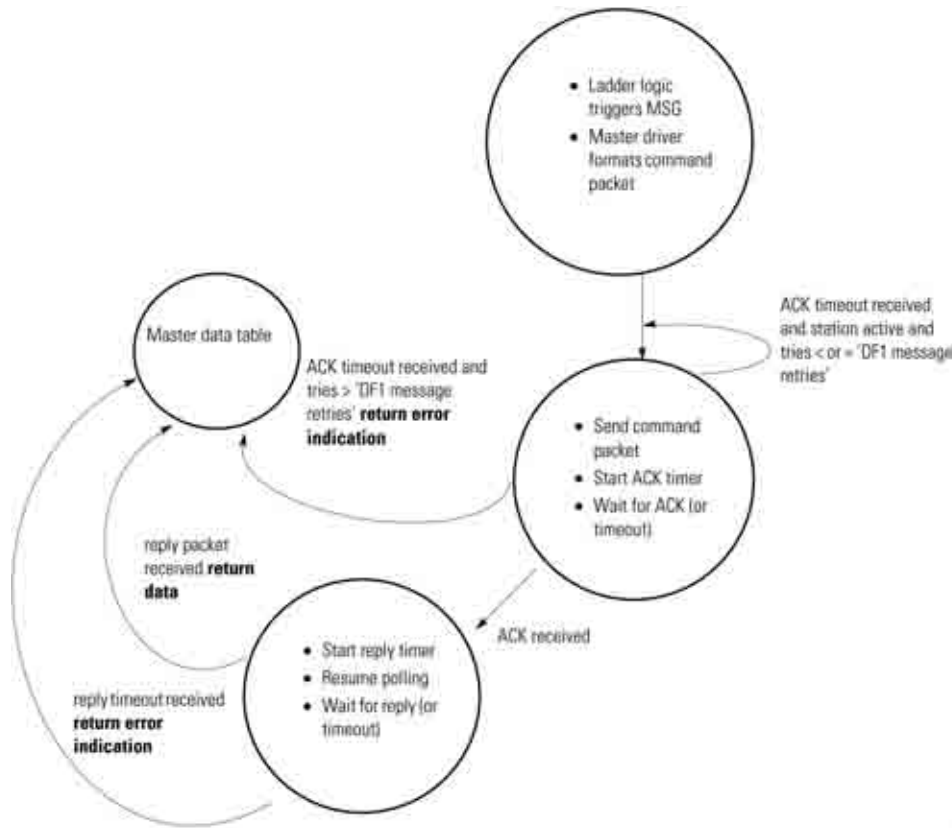


Figure 1.3 How a Master Station Requests Data

To help you understand standard-communication mode how a master station requests data

See Figure 1.2 Figure 1.3

To design a communication scheme using standard-communication mode, you must do the following:

- design a polling scheme
- plan for timing issues

DESIGNING A POLLING SCHEME

Each master station in a SCADA application must have a polling scheme configured. To design a polling scheme, do the following:

- choose the type of scheme best suited for your application
- optimize your polling scheme to obtain the best efficiency

The master station you are using determines the type of polling choices you have; however, Allen-Bradley master stations offer similar choices, such as:

- normal and priority polling lists
- ability to poll a slave station:
 - once per occurrence in the poll list (single)
 - until it has no more messages to send (multiple)

If you want to receive

only one message from a slave station per poll per a station's turn. Choose this method only if it is critical to keep the poll list scan time to a minimum.

Choose single transfer

as many messages from the slave station as it has in its queue.

multiple transfer

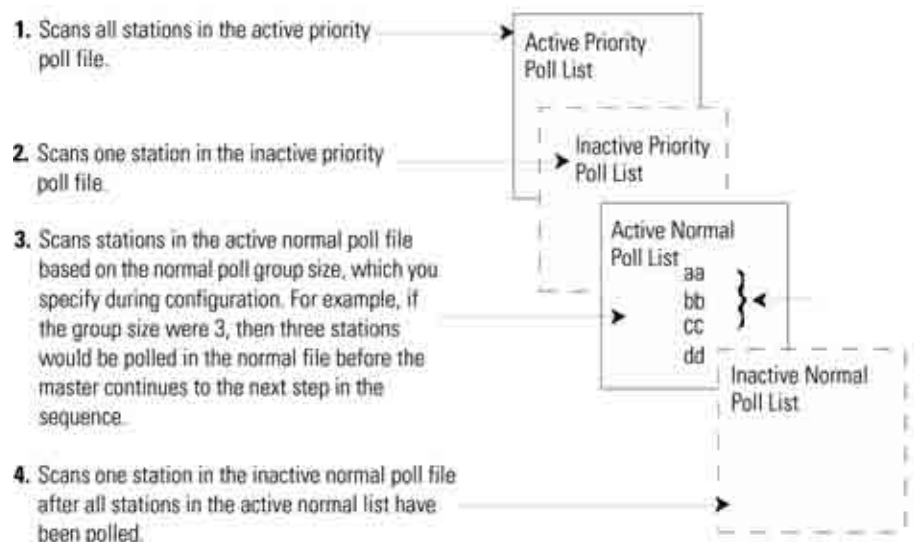


Figure 1.4 The master stations scans slave stations in a set sequence.

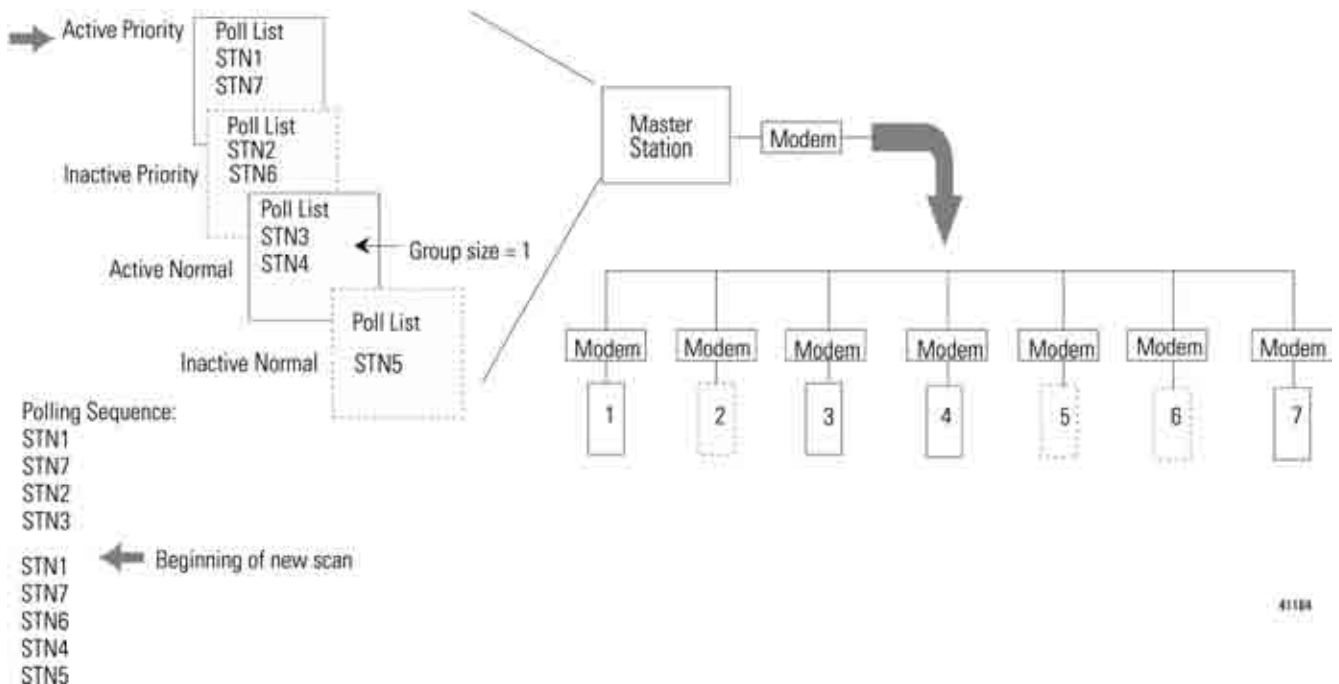


Figure 1.5 Here is how the polling sequence applies to an application.

PLANNING FOR TIMING ISSUES

Two types of timing categories exist.

- Protocol timers, which specify how long a master station will wait to ‘hear’ from a slave station.
- Request to send (RTS) timers, which you can use to make sure the modem is ready to accept data or has passed on the data (does not apply when control line is configured for No Handshaking).

Set and adjust these timing values as necessary for your application. Set your RTS times based on the communication media and modem you are using.

Design Considerations

- Define a polling list type to use (normal or priority).
- Define a station list.
- Use Figure 1.6 (see next page) to help understand how the MSGs are handled using standard communication.

COMMUNICATION SCHEME DESIGN USING MESSAGE-BASED MODE

In message-based communication mode, the master station sends solicited messages (messages programmed via ladder logic) to a specific slave station when the master requires information. In this mode, the communication link is inactive until the master station has a message to send to a slave station. Figure 1.7 explains the communication sequence that occurs.

DESIGNING COMMUNICATION FOR DF1 FULL-DUPLEX PROTOCOL

When designing communication using DF1 full-duplex protocol, you must configure time out values and retry counts that control the communication between a transmitting station and a receiving station. Consider the type of link media you are using to help you determine the best values for the timer and counters. For example, you can expect a message being sent

over a satellite link to take longer than one being sent over a telephone leased-line link. Figure 1.8 shows the communication sequence for DF1 full-duplex protocol.

DESIGNING COMMUNICATION FOR DF1 RADIO MODEM PROTOCOL

When designing communication using DF1 Radio Modem protocol, you must consider the capabilities of both the controllers and radio modems. The DF1 Radio Modem protocol can only be used with controllers that support and are configured for this protocol.

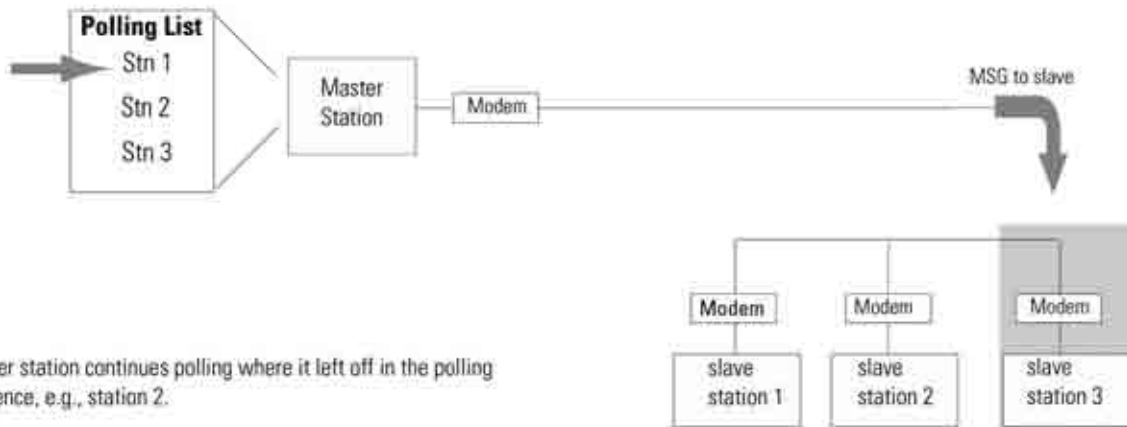
DETERMINING WHEN TO USE DF1 RADIO MODEM PROTOCOL

If your radio modem can handle full-duplex data port buffering and radio transmission collision avoidance, you can use peer-to-peer message initiation capability in every node (i.e., the ladder logic in any node can trigger a MSG instruction to any other node at any time). For messaging between nodes that are outside of radio transmission/reception range of each other, you may use either the Store and Forward capability of the protocol or the repeater capability of the radios.

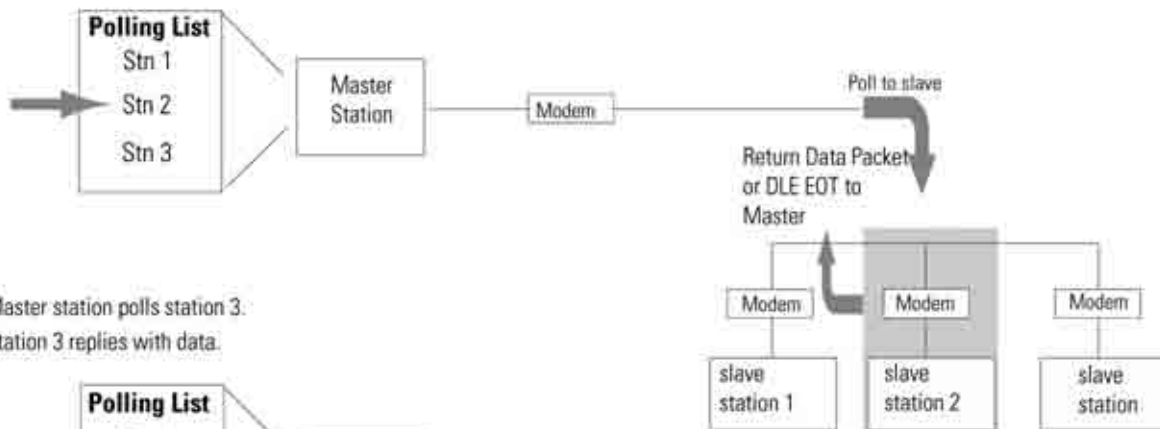
If your radio modem cannot handle full-duplex data port buffering and radio transmission collision avoidance, you can still use DF1 Radio Modem protocol in a Master/Slave configuration, with message initiation limited to a single master node. If you still require slave node message initiation, then you must use the DF1 Half-Duplex protocol.

The primary advantage of using DF1 Radio Modem protocol for radio modem networks is in the transmission efficiency. Each read/write transaction (command and reply) requires only one transmission by the initiator (to send the command) and one transmission by the responder (to return the reply) as illustrated in Figure 1.9. The number of transmissions is mini-

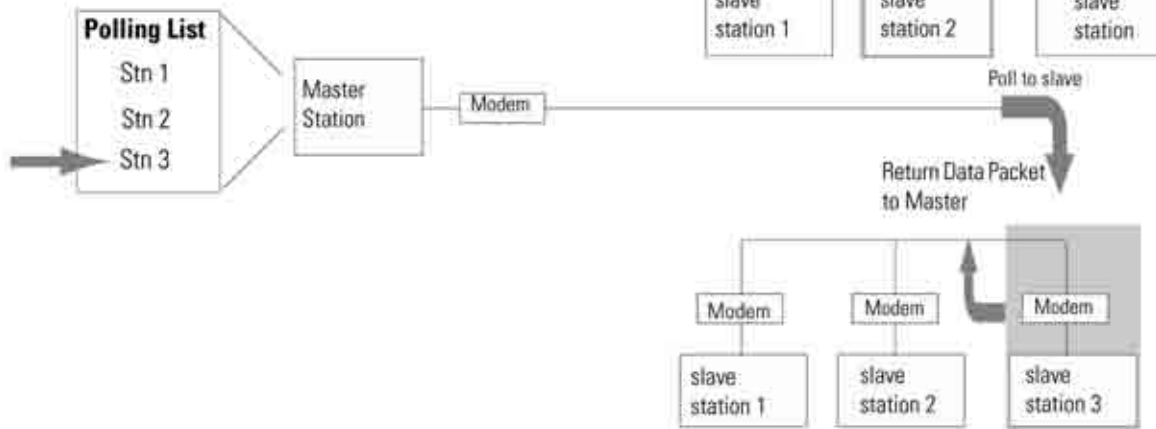
1. Polled station 1; ready to poll station 2.
2. MSG sent to station 3 (MSG was waiting in queue).



3. Master station continues polling where it left off in the polling sequence, e.g., station 2.



4. Master station polls station 3.
5. Station 3 replies with data.



6. Master station returns to beginning of the poll list.

Figure 1.6 Effect of MSGs on Logix, PLC-5, SLC 500 and MicroLogix Polling

mized, radio power is minimized, and throughput is maximized. In contrast, DF1 Half-Duplex protocol requires five transmissions for the DF1 Master to complete a read/write transaction with a DF1 Slave as illustrated in Figure 1.7. Figure 1.10 illustrates the DF1 Radio Modem protocol.

An efficiency trade-off exists in that the DF1 Radio Modem protocol does not provide immediate feedback (ACK) to the initiator to indicate that the responder successfully received the communications packet without error.

The Store and Forward capability of the DF1 Radio Modem protocol allows messages between nodes that are outside of radio transmission/reception range of each other to be routed through intermediary nodes that are within range. Each of the intermediary nodes needs a Store and Forward table. The configuration needs to indicate, based on the source and destination addresses in the message packet, which packets to receive (store) and then re-broadcast (forward). Figure 1.11 illustrates the Store and Forward capability.

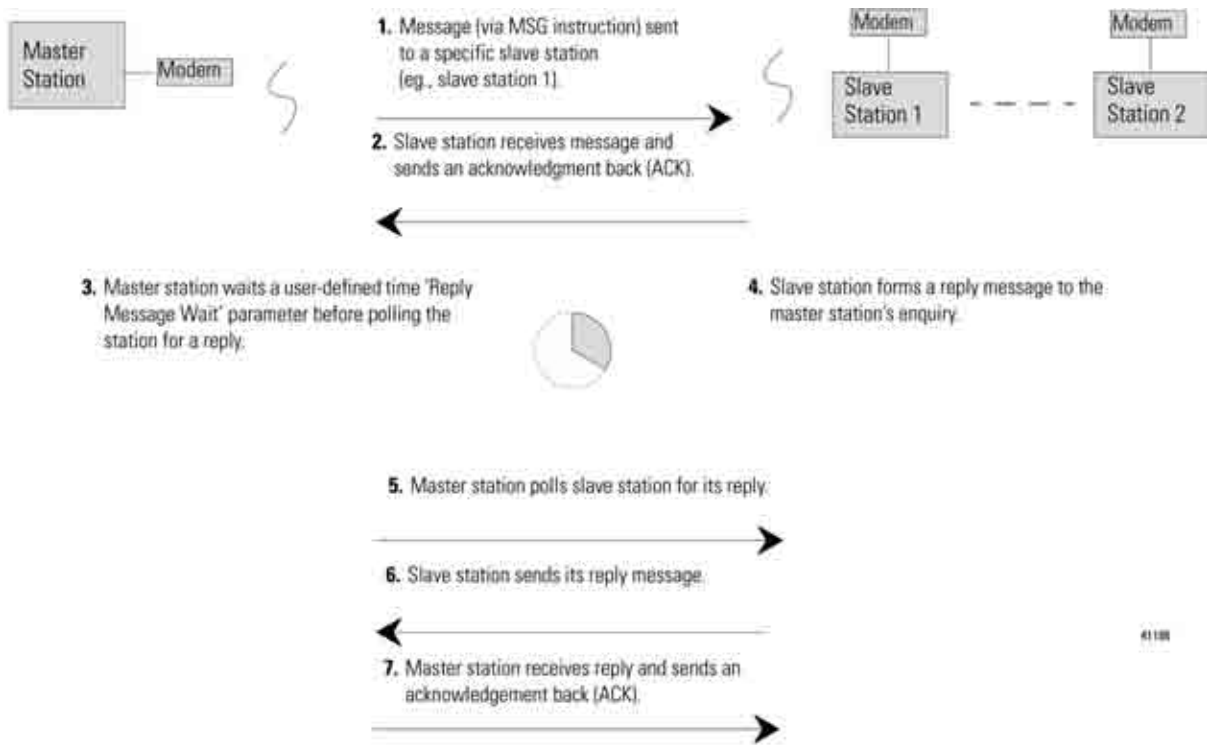


Figure 1.7 Message-Based Communication

WHAT TO DO NEXT?

Make sure you:

- choose the communication method best suited for your application.
- make initial configuration choices for the communication method you have chosen.
- use this chapter as a reference as you configure the devices in your SCADA system.

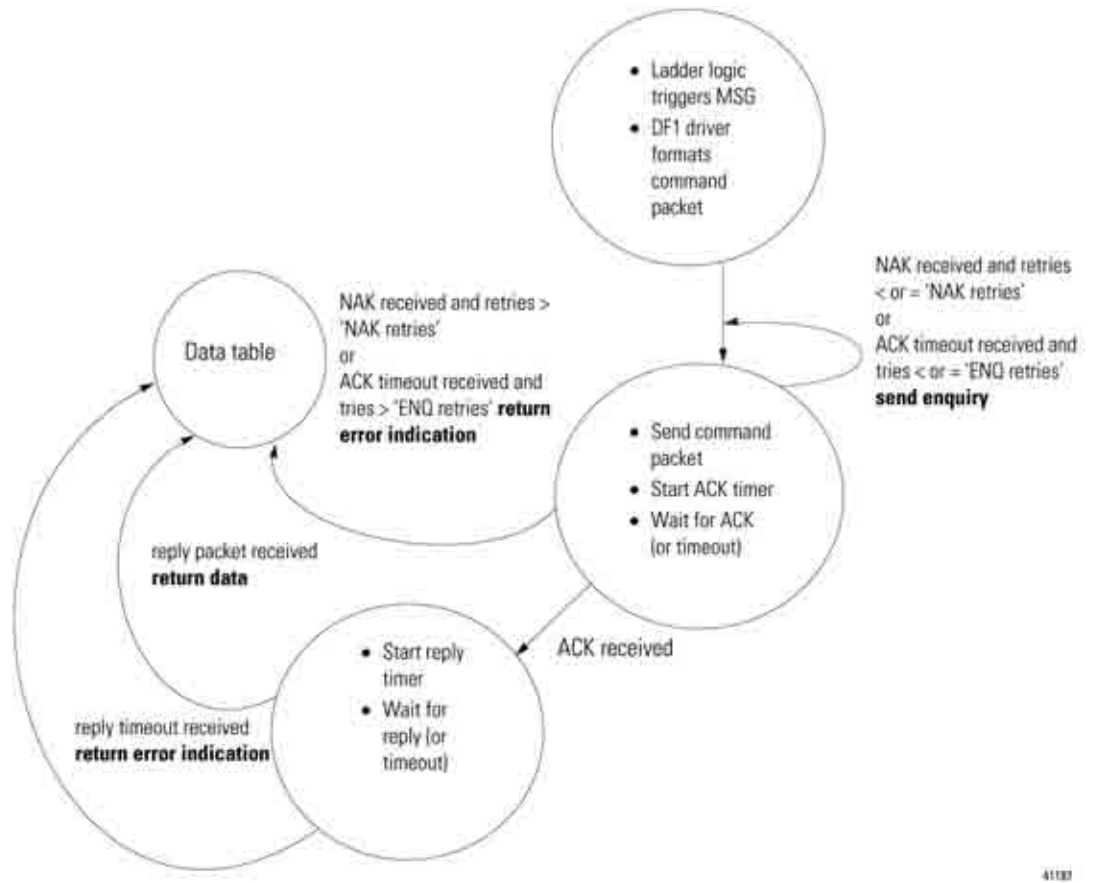


Figure 1.8 Read or Write Requests via DF1 Full-Duplex

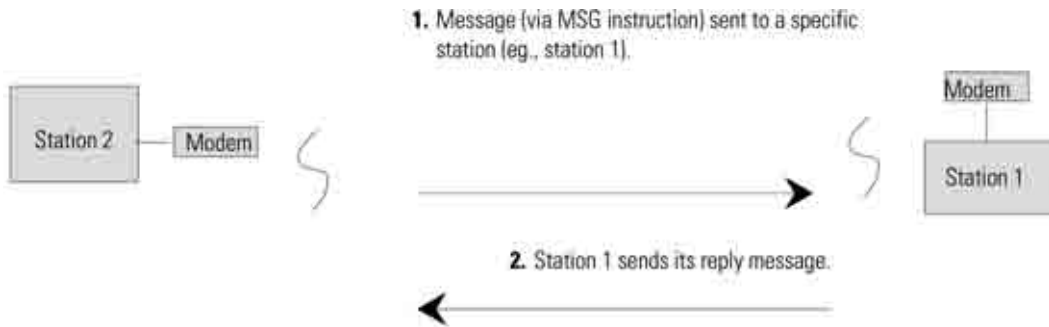


Figure 1.9 DF1 Radio Communication

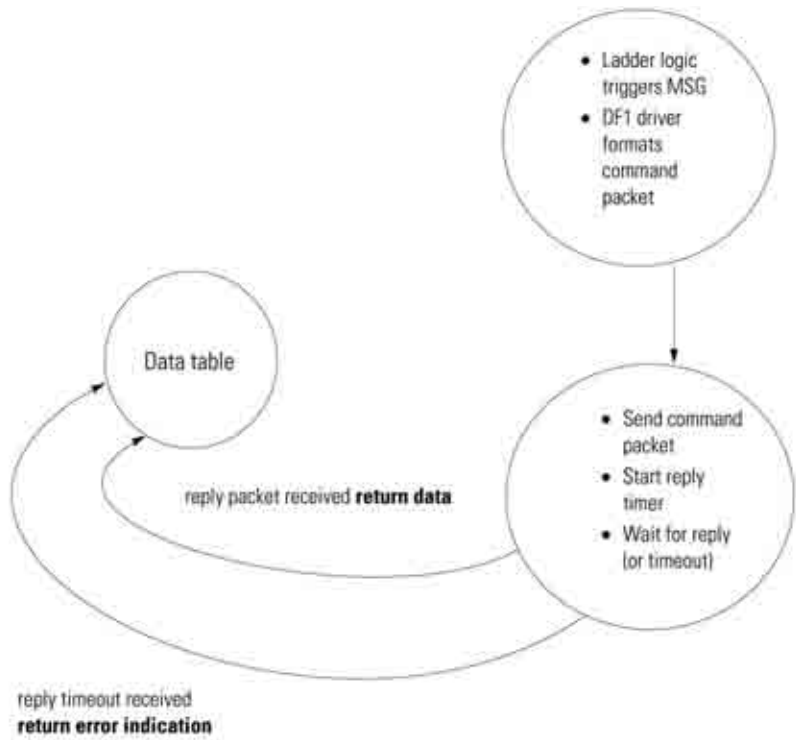


Figure 1.10 Read or Write Requests via DF1 Radio Modem

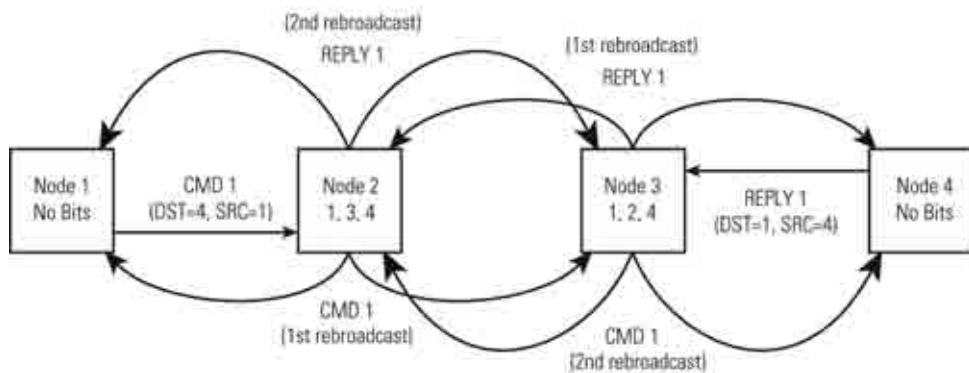


Figure 1.11 Applying Store and Forward in DF1 Radio Modem Protocol

INDUSTRIAL ETHERNET: A CONTROL ENGINEER'S GUIDE

Courtesy of Cisco Systems

ARTICLE SUMMARY

As part of a continuing effort to make their organizations more efficient and productive, manufacturers are rapidly migrating to Industrial Ethernet technology.

This standards-based technology enables organizations to control costs by moving from costly proprietary systems to a proven technology that is more secure, reliable, and deterministic.

This article provides an overview of Ethernet technology and its benefits in the data networking environment. It discusses the benefits of using a switched Ethernet architecture in industrial networking environments, including:

- Determinism
- Latency
- Minimal packet loss
- Broadcasts and multicast support
- Network analyzer monitoring
- Standardized infrastructure

Industrial Ethernet applies the Ethernet standards developed for data communication to manufacturing control networks. By implementing an intelligent Industrial Ethernet solution, organizations can build a manufacturing infrastructure that delivers the resiliency and network security of traditional fieldbus solutions, as well as improved bandwidth, open connectivity, and standardization that Ethernet provides. Industrial Ethernet provides organizations substantially greater control over their networked manufacturing equipment.

INTRODUCTION

As manufacturers seek to improve processes, reduce expenses, and improve productivity, many are turning to Ethernet technology on the factory floor. This migration is rapidly gaining momentum. According to a recent ARC Advisory Group study, the worldwide market for Industrial Ethernet devices is expected to grow at a rate of more than 84 percent over the next five years. Once considered a solution that was limited to corporate network environments, Ethernet technology has proven to be a robust alternative that can meet the unique needs of the manufacturing arena.

Industrial Ethernet networks that use intelligent switching technology can offer a variety of advantages compared to traditional industrial network installations.

Industrial Ethernet applies the Ethernet standards developed for data communication to manufacturing control networks. The technology can be deployed using a switched Ethernet architecture that has proven successful in multiple critical applications in different markets. Because the technology is based on industry standards, Industrial Ethernet enables organizations to save money by moving away from expensive, proprietary systems.

At the same time, it delivers the network security, performance, and availability required to support critical manufacturing applications.

To deploy this new technology, engineers on the manufacturing floor should be familiar with some of the important concepts behind Industrial Ethernet. This paper will provide a general overview of the most important traditional Ethernet technologies in use today. It will also discuss how Industrial Ethernet upgrades traditional, proprietary factory-floor networks to a low-cost, high-performance, scalable architecture. Finally, this paper will review some of the intelligent features that make Industrial Ethernet an attractive choice for manufacturing organizations.

WHAT IS ETHERNET?

Ethernet is the major local-area network (LAN) technology in use today, and is used for approximately 85 percent of the world's LAN-connected PCs and workstations. Ethernet refers to the family of LAN products covered by the IEEE 802.3 standard, and the technology can run over both optical fiber and twisted-pair cables. Over the years, Ethernet has steadily evolved to provide additional performance and network intelligence. This continual improvement has made Ethernet an excellent solution for industrial applications. Today, the technology can provide four data rates.

- 10BASE-T Ethernet delivers performance of up to 10 Mbps over twisted-pair copper cable.

- Fast Ethernet delivers a speed increase of ten times the 10BASE-T Ethernet specification (100 Mbps) while retaining many of Ethernet's technical specifications. These similarities enable organizations to use 10BASE-T applications and network management tools on Fast Ethernet networks.

- Gigabit Ethernet extends the Ethernet protocol even further, increasing speed tenfold over Fast Ethernet to 1000 Mbps, or 1 Gbps. Because it is based upon the current Ethernet standard and compatible with the installed base of Ethernet and Fast Ethernet switches and routers, network managers can support Gigabit Ethernet without needing to retrain or learn a new technology.

- 10 Gigabit Ethernet, ratified as a standard in June 2002, is an even faster version of Ethernet. It uses the IEEE 802.3 Ethernet media access control (MAC) protocol, the IEEE 802.3 Ethernet frame format, and the IEEE 802.3 frame size. Because 10 Gigabit Ethernet is a type of Ethernet, it can support intelligent Ethernet-based network services, interoperate with existing architectures, and minimize users' learning curves. Its high data rate of 10 Gbps makes it a good solution to deliver high bandwidth in wide-area networks (WANs) and metropolitan-area networks (MANs).

More than 300 million switched Ethernet ports have been installed worldwide. Ethernet technology enjoys such wide

acceptance because it is easy to understand, deploy, manage, and maintain. Ethernet is low-cost and flexible, and supports a variety of network topologies. Although traditional, non-Ethernet-based industrial solutions have a data rate of between 500 Kbps to 12 Mbps, Ethernet technology can deliver substantially higher performance. And, because it is based on industry standards, it can run and be connected over any Ethernet-compliant device from any vendor.

THE OPEN SYSTEMS INTERCONNECTION REFERENCE MODEL

At the heart of data networking is the Open Systems Interconnection (OSI) reference model. This conceptual model describes how information from a software application in one computer moves through a network medium to a software application in another computer. The model was developed by the International Organization for Standardization (ISO) in 1984, and it is now considered the primary architectural model for intercomputer communications.

The OSI reference model divides the tasks involved in moving information between networked computers into seven smaller, more manageable task groups. These tasks are then assigned to seven layers in the OSI model. Each layer is self-contained so that the tasks assigned to it can be implemented independently. Figure 1 shows the seven OSI layers.

Figure 1 Layers of the OSI Reference Model

Layer 1—Physical	Layer 5—Session
Layer 2—Data link	Layer 6—Presentation
Layer 3—Network	Layer 7—Application
Layer 4—Transport	

FUNCTIONS OF THE OSI LAYERS

The seven layers of the OSI reference model can be divided into lower layers (1–4) and upper layers (5–7). The lower layers of the OSI model focus on data-transport issues while the upper layers focus on the applications. The physical layer and the data link layer are implemented in hardware and software. The lowest layer, the physical layer, is closest to the physical network medium, such as network cabling. Ethernet resides in Layer 2, as do some implementations of traditional fieldbuses such as DeviceNet, which uses the Controller Area Network (CAN) protocol. Layer 3 takes care of the logical addressing and routing (which way to send data). Its most common implementation uses the Internet Protocol (IP), which is the core of World Wide Web addressing and routing. Layer 4, the last of the lower layers, is the transport layer. It ensures that data is delivered error-free and in the correct sequence. Industrial Ethernet is broader than traditional Ethernet technology. While Ethernet technology refers only to Layer 2, most Industrial Ethernet solutions also encompass Layer 3 and 4, using IP addressing in Layer 3, and Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) in Layer 4.

The upper layers of the OSI reference model are responsible for application tasks and are usually implemented only in software. The highest layer, the application layer, is closest to the end user. Both users and application-layer processes interact with software applications that involve network communications.

For more information about the OSI reference model, visit:

http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_d oc/introint.htm#xtocid5

BENEFITS OF A SWITCHED ETHERNET ARCHITECTURE

Organizations can choose from a variety of devices and architectures when building an Ethernet LAN. For industrial networking environments, a switched Ethernet architecture is the most appropriate choice. Switches make it possible for several users to send information over a network at the same time without slowing each other down.

In a fully switched network, there are no hubs so each Ethernet network has a dedicated segment for every node.

Because the only devices on each segment are the switch and the node, the switch picks up every transmission before it reaches another node. The switch then forwards the data over to the appropriate segment. In a fully switched network, nodes only communicate with the switch and never directly with each other.

Fully switched networks employ either twisted pair or fiber-optic cabling, both of which use separate conductors for sending and receiving data. This allows nodes to transmit to the switch at the same time the switch transmits to them, for a collision-free environment. Transmitting in both directions also can effectively double the apparent speed of the network when two nodes are exchanging information. For example, if the speed of the network is 10 Mbps, each node can transmit at 10 Mbps at the same time.

Switches usually work at Layer 2 (data link) of the OSI reference model using MAC addresses, and deliver a number of important advantages compared to hubs and other LAN devices. Some of these advantages include the following:

- **Determinism** — Determinism, the ability to ensure that a packet is sent and received in a specific period of time, is an important design goal for industrial networks. For the network to be deterministic, the design must be as simple and highly structured as possible.

- **Latency** — Switches normally have very low latencies, which refers to the time it takes for a network packet to transit between a source and a target. Most control operations in industrial applications can tolerate latencies of 10 to 50 milliseconds (ms). Because control traffic frames in industrial applications are usually below 500 bytes, the latency introduced by a switch at 100 Mbps is only about 30 microseconds with a worst-case scenario of close to 100 microseconds — well below the limit and 100 times faster than most applications require.

- **Packet loss under congestion** — Today's intelligent switches offer quality-of-service (QoS) features that make it possible to prioritize critical traffic so that it will not be dropped due to congestion. By implementing simple QoS parameters in an intelligent switch, organizations can prioritize critical traffic over noncritical traffic at wire speed, helping to ensure packet integrity for the control network. Even under heavy congestion, QoS features help ensure that important traffic will reach its destination.

- **Broadcasts and multicast** — Industrial applications often rely on broadcast or multicast communication. Intelligent switching platforms can dynamically configure the interfaces so that traffic is forwarded only to ports associated with requested data. This feature reduces the load of traffic crossing the network and relieves the client devices from processing unneeded

frames.

- Network analyzers — Intelligent switches allow traffic analyzers to remotely monitor any port in a network, which saves organizations time and money and reduces the amount of hardware that must be deployed to monitor and optimize network usage.

- Standardization — One of the main motives for Industrial Ethernet is the need to standardize around a common infrastructure. Unlike proprietary technologies that often tie companies to a particular vendor, standardized solutions free users to choose the best application for a given solution. And a standard Ethernet network brings to the factory floor the economies of scale enjoyed by today's large base of Ethernet users, lowering costs and increasing the number of potential equipment vendors and products.

WHAT IS INDUSTRIAL ETHERNET?

Recognizing that Ethernet is the leading networking solution, many industry organizations are porting the traditional fieldbus architectures to Industrial Ethernet. Industrial Ethernet applies the Ethernet standards developed for data communication to manufacturing control networks (Figure 2). Using IEEE standards-based equipment, organizations can migrate all or part of their factory operations to an Ethernet environment at the pace they wish.

For example, DeviceNet, a device-level network based on the Common Industrial Protocol, can be ported to the Ethernet environment (Figure 3). The fieldbus data structure is applied to Layers 5, 6, and 7 of the OSI reference model over Ethernet, IP, and TCP/UDP in the transport layer (Layer 4).

The advantage of Industrial Ethernet is that organizations and devices can continue using their traditional tools and applications running over a much more efficient networking infrastructure.

Industrial Ethernet not only gives manufacturing devices a much faster way to communicate, but also gives the users better connectivity and transparency, enabling users to connect to the devices they want without requiring separate gateways.

TRADITIONALLY SEPARATE NETWORKS

Today, many manufacturing companies maintain separate networks to support their factory floor operations and business operations. Over the years, these networks were developed to respond to the different information flows and control

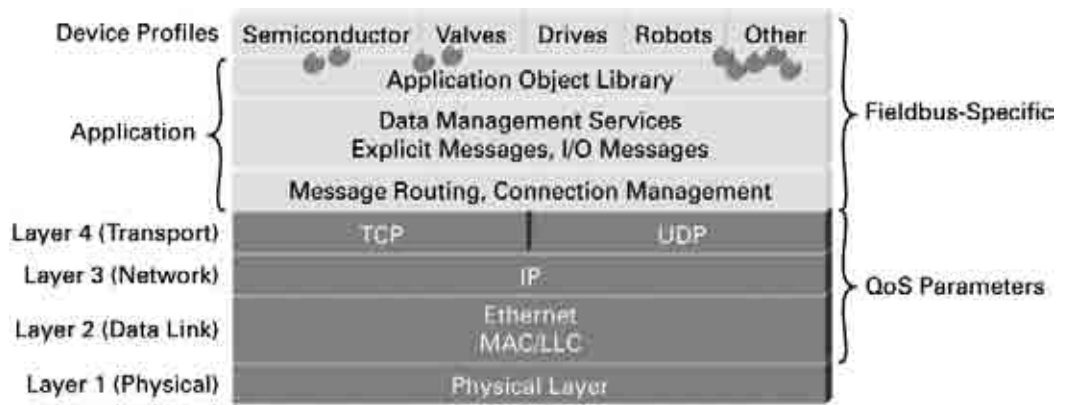


Figure 2 Using Intelligent Ethernet for Automation Control

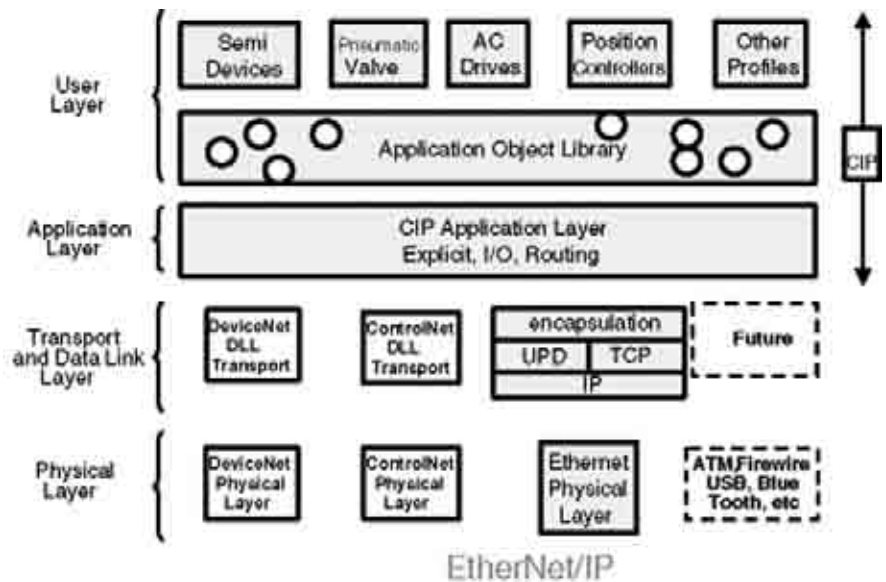


Figure 3 Proprietary Fieldbus Architecture

requirements involved with manufacturing processes.

The corporate IT network supports traditional administrative functions and corporate applications, such as human resources, accounting, and procurement. This network is usually based on the Ethernet standard.

The control-level network connects control and monitoring devices, including programmable logic controllers, PC-based controllers, I/O racks, and human-machine interfaces (HMIs). This network, which has not been Ethernet in the past, requires a router or, in most cases, a gateway to translate application-specific protocols to Ethernet-based protocols. This translation lets information pass between the control network on the factory floor and the corporate network infrastructure.

The device-level network links the plant floor's I/O devices, including sensors such as transducers, photoeyes, and flowmeters, and other automation and motion equipment, such as robotics, variable frequency drives, and actuators.

Interconnectivity between these devices was traditionally achieved with a variety of fieldbuses such as DeviceNet,

Profibus, and Modbus. Each fieldbus has specific power, cable, and communication requirements, depending on the factory application it supports. This has led to a replication of multiple networks in the same space and the need to have multiple sets of spares, skills, and support programs within the same organization.

Instead of using architectures composed of multiple separate networks, Industrial Ethernet can unite a company's administrative, control-level, and device-level networks to run over a single network infrastructure. In an Industrial Ethernet network, fieldbus-specific information that is used to control I/O devices and other manufacturing components are embedded into Ethernet frames. Because the technology is based on industry standards rather than on custom or proprietary standards, it is more interoperable with other network equipment and networks.

TECHNOLOGY TAILORED FOR MANUFACTURING

Although Industrial Ethernet is based on the same industry standards as traditional Ethernet technology, the implementation of the two solutions is not always identical. Industrial Ethernet usually requires more robust equipment and a very high level of traffic prioritization when compared with traditional Ethernet networks in a corporate data network.

The primary difference between Industrial Ethernet and traditional Ethernet is the type of hardware used. Industrial Ethernet equipment is designed to operate in harsh environments. It includes industrial-grade components, convection cooling, and relay output signaling. And it is designed to operate at extreme temperatures and under extreme vibration and shock. Power requirements for industrial environments differ from data networks, so the equipment runs using 24 volts of DC power. To maximize network availability, it also includes fault-tolerant features such as redundant power supplies.

Industrial Ethernet environments also differ from traditional Ethernet networks in their use of multicasting by hosts for certain applications. Industrial applications often use producer-consumer communication, where information "produced" by one device can be "consumed" by a group of other devices (see box). In a producer-consumer environment, the most important priority for a multicast application is to guarantee that all hosts receive data at the same time. A traditional Ethernet network, on the other hand, focuses more on the efficient utilization of bandwidth in general, and less on synchronous data access. To help optimize synchronous data access, Industrial Ethernet equipment must include the intelligence and QoS features needed to enable organizations to appropriately prioritize multicast transmissions.

INCREASING INDUSTRY SUPPORT

Industrial Ethernet technology is rapidly being embraced by multiple organizations and vendors, including the Industrial Ethernet Association (IEA), the Open DeviceNet Vendor Association (ODVA), Modbus.org, Fieldbus Foundation, and the Industrial Automation Open Networking Alliance (IAONA).

NETWORK REQUIREMENTS: THE NEED FOR INTELLIGENCE

When implementing an Industrial Ethernet solution, companies should be careful to select Ethernet products that offer the intelligent features required to support manufacturing applications. Network intelligence enables organizations to build a manufacturing infrastructure that matches the resiliency and network security of traditional fieldbus solutions, while at

the same time providing the benefits of higher bandwidth, open connectivity, and standardization offered by Ethernet-based platforms. The important qualities behind an intelligent Industrial Ethernet solution include network security, reliability, and determinism.

NETWORK SECURITY

Ethernet technology can provide not only excellent performance for manufacturing applications, but a wide range of network security measures to provide both confidentiality and data integrity. Confidentiality helps ensure that data cannot be accessed by unauthorized users. Data integrity protects data from intentional or accidental alteration.

These network security advantages protect manufacturing devices like programmable logic controllers (PLCs) as well as PCs, and apply to both equipment and data security.

Manufacturers can use many methods to help ensure network confidentiality and integrity. These network security measures can be grouped into several categories, including access control and authentication, and secure connectivity and management.

ACCESS CONTROL AND AUTHENTICATION

Access control is commonly implemented using firewalls or network-based controls protecting access to critical applications, devices, and data so that only legitimate users and information can pass through the network. However, access-control technology is not limited to dedicated firewall devices. Any device that can make decisions to permit or deny network traffic, such as an intelligent switch, is part of an integrated access-control solution.

When designing an access-control solution, network administrators can set up filtering decisions based on a variety of criteria, such as an IP address or TCP/UDP port number. Intelligent switches can provide support for this advanced filtering to limit network access to authorized users. At the same time, they can enable organizations to enforce policy decisions based on the IP or MAC address of a laptop or PLC.

Virtual LANs (VLANs) are another access-control solution, providing the ability to create multiple IP subnets within an Ethernet switch. VLANs provide network security and isolation by virtually segmenting factory-floor data from other data and users. VLANs can also be used to enhance network performance, separating low-priority end devices from high-priority devices.

Access controls can also include a variety of device or user-authentication services. Authentication services determine who may access a network and what services they are authorized to use. For example, the 802.1x authentication protocol provides port-based authentication so that only legitimate devices can connect to switch ports. Authentication services are an effective complement to other network security measures in a manufacturing environment.

SECURE CONNECTIVITY AND MANAGEMENT

To provide additional protection for manufacturing networks, organizations can take several approaches to authenticate and encrypt network traffic. Using virtual private network (VPN) technology, Secure Sockets Layer (SSL) encryption can be applied to application-layer data in an IP network. Organizations can also use IP Security (IPSec) technology to encrypt and authenticate network packets to thwart network

attacks such as sniffing and spoofing.

VPN client software, together with dedicated VPN network hardware, can be used to encrypt device monitoring and programming sessions, and to support strong authentication. Manufacturers can also use Secure Shell (SSH) Protocol encryption for remote terminal logins to network devices. Version 3 of Simple Network Management Protocol (SNMP) also offers support for encryption and authentication of management commands and data.

RELIABILITY

Because factory-floor applications run in real time, the network must be available to users on a continuous basis, with little or no downtime. Manufacturers can help ensure network reliability using effective network design principles, as well as intelligent networking services.

NETWORK TOPOLOGY

Manufacturers deploying an Ethernet solution should design networks with redundant paths to ensure that a single device outage does not take down the entire network. Two network topologies most often used are ring and hub-and-spoke. In hub-and-spoke designs (Figure 4), three layers of switches are usually installed. The first layer is often referred to as the access layer. These switches provide connections for end-point devices like PLCs, robots, and HMIs. A second layer called the distribution layer provides connectivity between the access-layer switches. And a third layer called the core layer provides connectivity to other networks or to the Internet service provider (ISP) via routers. The distribution layer may include switches with routing functions to provide inter-VLAN routing.

Access-layer switches, on the other hand, generally provide only Layer 2 (data link) forwarding services. For optimum performance, network equipment at each of these layers must be aware of the information contained within the Layer 2 through Layer 4 packet headers.

In ring topologies (Figure 5), all devices are connected in a ring. Each device has a neighbor to its left and right. If a connection on one side of the device is broken, network connectivity can still be maintained over the ring via the opposite side of the device. In some situations, manufacturers install dual counter-rotating rings to maximize availability. In a ring topology, each switch functions as both an access-layer and as a distribution-layer switch.

SPANNING TREE PROTOCOLS

To prevent loops from being formed in the network when devices are interconnected via multiple paths, some organizations use the Spanning Tree Protocol. If a problem occurs on a network node, this protocol enables a redundant alternative link to automatically come back online.

The traditional Spanning Tree Protocol has been considered too slow for industrial environments. To address these performance concerns, the IEEE standards committee has ratified a new Rapid Spanning Tree Protocol (802.1w).

This protocol provides subsecond convergence times that vary between 200 and 800 ms, depending on network topology. Using 802.1w, organizations can enjoy the benefits of Ethernet networks, with the performance and reliability that manufacturing applications demand.

Another spanning-tree option is Multiple Spanning Tree Protocol (802.1s). This enables VLANs to be grouped into span-

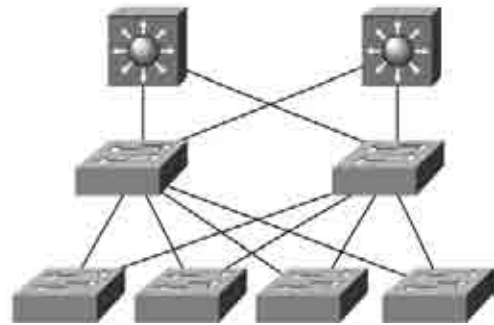


Figure 4 Hub-and-Spoke Network Topology

ning-tree instances. Each instance has a spanning-tree topology that is independent from other spanning-tree instances. This architecture provides multiple forwarding paths for data traffic, enables load balancing, and reduces the number of spanning-tree instances needed to support a large number of VLANs.

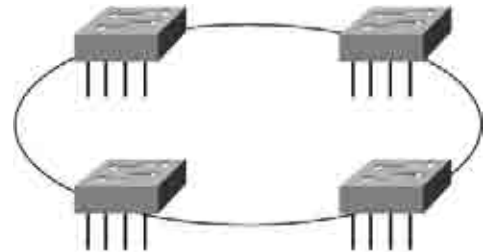


Figure 5 Ring Topology

USING OPTION 82

Ethernet switches provide excellent connectivity and performance; however, each switch is another device that must be managed on the factory floor. To make switched Ethernet networks easy to support and maintain, intelligent switches include built-in management capabilities. These intelligent features make it easy to connect manufacturing devices to the network, without creating additional configuration tasks. And they help minimize network downtime if part of the network should fail. One of the most useful intelligent features in a switched Ethernet network is Option 82.

In an Ethernet network, Dynamic Host Configuration Protocol (DHCP) lets devices dynamically acquire their IP addresses from a central server. The DHCP server can be configured to give out the same address each time or generate a dynamic one from a pool of available addresses.

Because the interaction of the factory-floor devices requires specific addresses, Industrial Ethernet networks usually don't use dynamic address pools. However, static addresses can have drawbacks. Because they are linked to the MAC address of the client, and because the MAC address is often hard-coded in the network interface of the client device, the association is lost when a client device fails and needs to be replaced.

Extended fields in the DHCP packet can be filled in by

the switch, indicating the location of the device requesting an IP address. The 82nd optional field, called Option 82, carries the specific port number and the MAC address of the switch that received the DHCP request. This modified request is sent on to the DHCP server. If an access server is Option 82-aware, it can use this information to formulate an IP address based on the Option 82 information.

Effective use of Option 82 enables manufacturers to minimize administrative demands and maintain maximum network uptime even in the event of the failure of individual devices.

DETERMINISM

Because manufacturing processes depend on the precise synchronization of processes, network determinism must be optimized to deliver the best possible performance. Data must be prioritized using QoS to ensure that critical information is received first. And the multicast applications that are prevalent in manufacturing environments must be well-managed using Internet Group Management Protocol (IGMP) snooping.

THE PRODUCER-CONSUMER MODEL IN INDUSTRIAL ETHERNET

Many Industrial Ethernet applications depend on IP multicast technology. IP multicast allows a host, or source, to send packets to another group of hosts called receivers anywhere within the IP network using a special form of IP address called the IP multicast group address.

While traditional multicast services, such as video or multimedia, tend to scale with the number of streams, Industrial Ethernet multicast applications do not. Industrial Ethernet environments use a producer-consumer model, where devices generate data called “tags” for consumption by other devices. The devices that generate the data are producers and the devices receiving the information are consumers. Multicast is more efficient than unicast, because consumers will often want the same information from a particular producer. Each device on the network can be both a producer and a consumer of traffic.

While most devices generate very little data, networks with a large number of nodes can generate a large amount of multicast traffic, which can overrun end devices in the network. Using mechanisms like QoS and IGMP snooping, organizations can control and manage multicast traffic in manufacturing environments.

QUALITY OF SERVICE

An Industrial Ethernet network may transmit many different types of traffic, from routine data to critical control information, or even bandwidth-intensive video or voice. The network must be able to distinguish among and give priority to different types of traffic.

To address these issues, organizations can implement

QoS using several techniques. QoS involves three important steps. First, different traffic types in the network need to be identified through classification techniques. Second, advanced buffer-management techniques need to be implemented to prevent high-priority traffic from being dropped during congestion. Finally, scheduling techniques need to be incorporated to transmit high-priority traffic from queues as quickly as possible.

In Layer 2 switches on an Ethernet network, QoS usually prioritizes native, encapsulated Ethernet frames, or frames tagged with 802.1p class of service (CoS) specifications. More advanced QoS mechanisms take this definition a step further. For example, advanced Ethernet switches can study and interpret the flow of QoS traffic as it is processed through the switch.

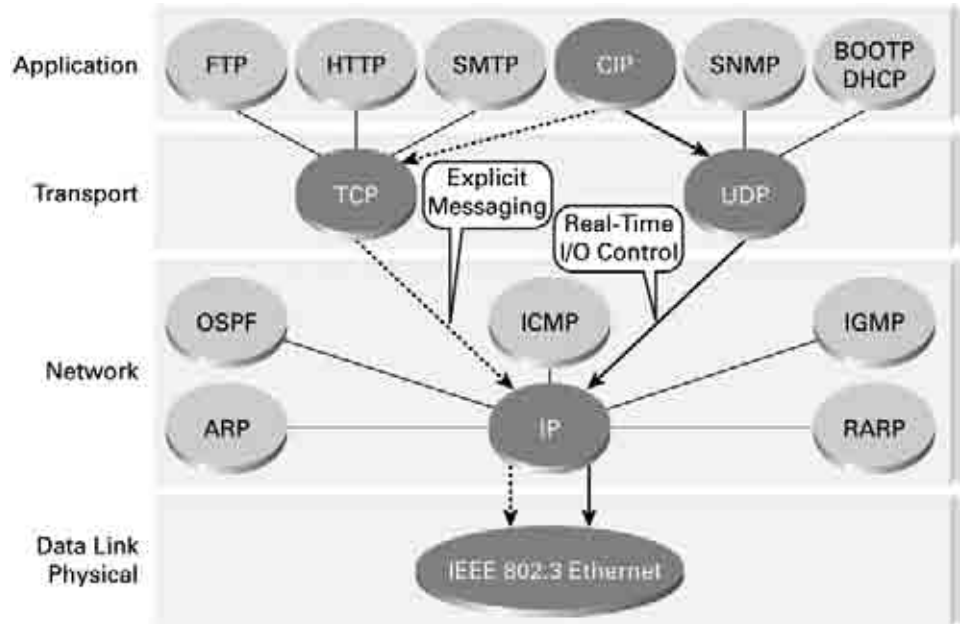


Figure 6 Applying QoS to Industrial Applications

A switch can be configured to prioritize frames based on given criteria at different layers of the OSI reference model (Figure 6). For example, traffic could be prioritized according to the source MAC address (in Layer 2) or the destination TCP port (in Layer 4). Any traffic traveling through the interface to which this QoS is applied is classified, and tagged with the appropriate priority. Once a packet has been classified, it is then placed in a holding queue in the switch, and scheduled based on the scheduling algorithm desired.

In an Industrial Ethernet application, real-time I/O control traffic would share network resources with configuration (FTP) and data-collection flows, as well as other traffic, in the upper layers of the OSI reference model. By using QoS to give high priority to real-time UDP control traffic, organizations can prevent delay or jitter from affecting any control functions.

IGMP SNOOPING

Many manufacturing applications depend on multicast traffic, which can introduce performance problems in the network. To address these challenges in an Industrial Ethernet environment, organizations can deploy IGMP snooping. IGMP snooping limits the flooding of multicast traffic by dynamically configuring the interfaces so that multicast traffic is forwarded only to interfaces associated with IP multicast devices. In other

words, when a multicast message is sent to the switch, the switch forwards the message only to the interfaces that are interested in the traffic.

This is very important because it reduces the load of traffic traversing through the network. It also relieves the hosts from processing frames that are not needed.

In a producer-consumer model used by Industrial Ethernet, IGMP snooping can limit unnecessary traffic from the I/O device that is producing, so that it only reaches the device consuming that data. Messages delivered to a given device that were intended for other devices consume resources and slow performance, so networks with many multicasting devices will suffer performance issues if IGMP snooping or other multicast limiting schemes are not implemented.

The IGMP snooping feature allows Ethernet switches to “listen” to the IGMP conversation between hosts. With IGMP snooping, the Ethernet switch examines the IGMP traffic coming to the switch and keeps track of multicast groups and member ports. When the switch receives an “IGMP join” report from a host for a particular multicast group, the switch adds the host port number to the associated multicast forwarding table entry. When it receives an IGMP “leave group” message from a host, it removes the host port from the table entry. After the switch relays the IGMP queries, it deletes entries periodically if it does not receive any IGMP membership reports from the multicast clients. A Layer 3 router normally performs the querying function.

When IGMP snooping is enabled in a network with Layer 3 devices, the multicast router sends out periodic IGMP general queries to all VLANs. The switch responds to the router queries with only one “join” request per MAC multicast group. The switch then creates one entry per VLAN in the Layer 2 forwarding table for each MAC group from which it receives an IGMP join request. All hosts interested in this multicast traffic send “join” requests and are added to the forwarding table entry.

Layer 2 multicast groups learned through IGMP snooping are dynamic. However, in a managed switch, organizations can statically configure MAC multicast groups. This static setting supersedes any automatic manipulation by IGMP snooping. Multicast group membership lists can consist of both user-defined settings and settings learned via IGMP snooping.

CONCLUSION

The migration to Ethernet in manufacturing environments has been growing steadily as companies recognize the many benefits that Industrial Ethernet can deliver. According to an ARC Advisory Group study, the market for Industrial Ethernet devices grew at more than a 50 percent annual rate from 2001 to 2003.

The reasons behind the success of Industrial Ethernet are clear. The technology lets manufacturers standardize and consolidate their different manufacturing network architectures, using products offered by a variety of equipment vendors. Because Industrial Ethernet is a standards-based technology, it enables companies to take advantage of economies of scale, while still providing the flexibility needed to support their specific factory-floor requirements.

Because Industrial Ethernet uses the intelligent networking features found in corporate data Ethernet environments, organizations can enjoy substantially greater control over their networked manufacturing equipment.

A well-implemented Industrial Ethernet network can do

much more than simply emulate the functions of a traditional manufacturing network. It enables companies to more closely link their internal data networks with the factory floor to make the entire company’s operations more efficient. And by enabling manufacturers to tap the innovation underway that supports the millions of existing Ethernet networks, it can make possible a wide range of new applications to support business needs well into the future.

PACS FOR INDUSTRIAL CONTROL, THE FUTURE OF CONTROL

Courtesy of National Instruments

A passionate debate has raged about the advantages and disadvantages of PLCs (programmable logic controllers) compared to PC-based control. As the technological differences between PC and PLC wane, with PLCs using commercial off the shelf (COTS) hardware and PC systems incorporating real-time operating systems, a new class of controllers, the PAC is emerging. PAC, a new acronym created by Automation Research Corporation (ARC), stands for Programmable Automation Controller and is used to describe a new generation of industrial controllers that combine the functionality of a PLC and a PC. The PAC acronym is being used both by traditional PLC vendors to describe their high end systems and by PC control companies to describe their industrial control platforms.

THE "80-20" RULE

During the three decades following their introduction, PLCs have evolved to incorporate analog I/O, communication over networks, and new programming standards such as IEC 61131-3. However, engineers create 80 percent of industrial applications with digital I/O, a few analog I/O points, and simple programming techniques. Experts from ARC, Venture Development Corporation (VDC), and the online PLC training source PLCS.net estimate that:

77% of PLCs are used in small applications (less than 128 I/O)

72% of PLC I/O is digital

80% of PLC application challenges are solved with a set of 20 ladder-logic instructions

Because 80 percent of industrial applications are solved with traditional tools, there is strong demand for simple low-cost PLCs. This has spurred the growth of low-cost micro PLCs with digital I/O that use ladder logic. It has also created a discontinuity in controller technology, where 80 percent of applications require simple, low cost controllers and 20 percent relentlessly push the capabilities of traditional control systems. The applications that fall within the 20 percent are built by engineers who require higher loop rates, advanced control algorithms, more analog capabilities, and better integration with the enterprise network.

In the 80s and 90s, these "20 percenters" evaluated PCs for industrial control. The PC provided the software capabilities to perform advanced tasks, offered a graphical rich programming and user environment, and utilized COTS components allowing control engineers to take advantage of technologies developed for other applications. These technologies include floating point processors; high speed I/O busses, such as PCI and Ethernet; non-volatile data storage; and graphical development software tools. The PC also provided unparalleled flexibility, highly productive software, and advanced low-cost hardware.

However, PCs were still not ideal for control applica-

tions. Although many engineers used the PC when incorporating advanced functionality such as analog control and simulation, database connectivity, web based functionality, and communication with third party devices, the PLC still ruled the control realm. The main problem with PC-based control was that standard PCs were not designed for rugged environments.

THE PC PRESENTED THREE MAIN CHALLENGES:

Stability: Often, the PC's general-purpose operating system was not stable enough for control. PC-controlled installations were forced to handle system crashes and unplanned rebooting.

Reliability: With rotating magnetic hard drives and non-industrially hardened components, such as power supplies, PCs were more prone to failure.

Unfamiliar Programming Environment: Plant operators need the ability to override a system for maintenance or troubleshooting. Using ladder logic, they can manually force a coil to a desired state, and quickly patch the affected code to quickly override a system. However, PC systems require operators learn new, more advanced tools.

Although some engineers use special industrial computers with rugged hardware and special operating systems, most engineers avoided PCs for control because of problems with PC reliability. In addition, the devices used within a PC for different automation tasks, such as I/O, communications, or motion, may have different development environments.

So the "twenty percenters" either lived without functionality not easily accomplished with a PLC or cobbled together a system that included a PLC for the control portion of the code and a PC for the more advanced functionality. This is the reason many factory floors today have PLCs used in conjunction with PCs for data logging, connecting to bar code scanners, inserting information into databases, and publishing data to the Web. The big problem with this type of setup is that these systems are often difficult to construct, troubleshoot and maintain. The system engineer often is left with the unenviable task of incorporating hardware and software from multiple vendors, which poses a challenge because the equipment is not designed to work together.

BUILDING A BETTER CONTROLLER

With no clear PC or PLC solution, engineers with complex applications worked closely with control vendors to develop new products. They requested the ability to combine the advanced software capabilities of the PC with the reliability of the PLC. These lead users helped guide product development for PLC and PC-based control companies.

The software capabilities required not only advanced software, but also an increase in the hardware capabilities of the controllers. With the decline in world-wide demand for PC com-

ponents, many semiconductor vendors began to redesign their products for industrial applications. Control vendors today are incorporating industrial versions of floating point processors, DRAM, solid-state storage devices such as CompactFlash, and fast Ethernet chipsets into industrial control products. This enables vendors to develop more powerful software with the flexibility and usability of PC-based control systems that can run on real-time operating systems for reliability.

The resulting new controllers, designed to address the “20 percent” applications, combine the best PLC features with the best PC features. Industry analysts at ARC named these devices programmable automation controllers, or PACs. In their “Programmable Logic Controllers Worldwide Outlook” study, ARC identified five main PAC characteristics. These criteria characterize the functionality of the controller by defining the software capabilities:

“Multi-domain functionality, At least 2 of logic, motion, PID control, drives, and process on a single platform.” Except for some variations in I/O to address specific protocols like SERCOS; logic, motion, process, and PID are simply a function of the software. For instance, motion control is a software control loop which reads digital inputs from a quadrature encoder, performs analog control loops, and outputs an analog signal to control a drive.

“Single multi-discipline development platform incorporating common tagging and a single database for access to all parameters and functions.” Because PACs are designed for more advanced applications such as multi-domain designs, they require more advanced software. In order to make system design efficient, the software must be a single integrated software package instead of disparate software tools which are not engineered to seamlessly work together.

“Software tools that allow the design by process flow across several machines or process units, together with IEC61131-3, user guidance, and data management.” Another component that simplifies system design is high level graphical development tools that make it easy to translate an engineer’s concept of the process into code that actually controls the machine.

“Open, modular architectures that mirror industry applications from machine layouts in factories to unit operations in process plants.” Because all industrial applications require significant customization, the hardware must offer modularity so the engineer can pick and choose the appropriate components. The software must enable the engineer to add and remove modules to design the required system.

“Employ de-facto standards for network interfaces, languages, etc., such as TCP/IP, OPC & XML, and SQL queries.” Communication with enterprise networks is critical for modern control systems. Although PACs include an Ethernet port, the software for communication is the key to trouble-free integration with the rest of the plant.

Traditional PLC software vendors start with a reliable and easy-to-use scanning architecture and work to add new functionality. PLC software follows a general model of scanning inputs, running control code, updating outputs, and performing housekeeping functions.

TWO APPROACHES TO SOFTWARE

While software is the key difference between PACs and PLCs, vendors vary in their approach to providing the advanced software. They typically start with their existing control software and work to add the functionality, reliability, and ease-of-use required to program PACs. Generally, this creates two camps of PAC software providers: those with a background in PLC control and those with a background in PC control.

SOFTWARE BASED ON PLC PHILOSOPHY

Traditional PLC software vendors start with a reliable and easy-to-use scanning architecture and work to add new functionality. PLC software follows a general model of scanning inputs, running control code, updating outputs, and performing housekeeping functions. A control engineer is concerned only with the design of the control code because the input cycles, output cycles, and housekeeping cycles are all hidden. With much of the work done by the vendor, this strict control architecture makes it easier and faster to create control systems. The rigidity of these systems also eliminates the need for the control engineer to completely understand the low-level operation of the PLC to create reliable programs. However, the rigid scanning architecture which is the main strength of the PLC, can also make it inflexible. Most PLC vendors create PAC software by adding into the existing scanner architecture new functionality such as Ethernet communication, motion control, and advanced algorithms. However, they typically maintain the familiar look and feel of PLC programming and the inherent strengths in logic and control. The result is PAC software generally designed to fit specific types of applications such as logic, motion, and PID, but is less flexible for custom applications such as communication, data logging, or custom control algorithms.

SOFTWARE BASED ON PC PHILOSOPHY

Traditional PC software vendors start with a very flexible general-purpose programming language, which provides in-depth access to the inner workings of the hardware. This software also incorporates reliability, determinism, and default control architectures. Although engineers can create the scanner structure normally provided to the PLC programmer, they are not inherent to PC-based control software. This makes the PC software extremely flexible and well suited for complex applications that require advanced structures, programming techniques, or system level control but more difficult for simple applications.

The first step for these vendors is to provide reliability and determinism, which are often not available in a general-purpose operating system such as Windows. This is accomplished through real-time operating systems (RTOS) such as Phar Lap from Ardence (formerly Venturcom) or VxWorks from Wind River. These RTOSs provide the capability to control all aspects of the control system, from the I/O read and write rates to the priority of individual threads spawned on the controller. These vendors then add abstractions and I/O read/write structures to make it simpler for engineers to build reliable control applications. The result is flexible software suited for custom control,

data logging, and communication but lacking the familiar PLC programming architectures, making application development more demanding.

National Instruments manufactures a family of PAC deployment platforms that run LabVIEW software. LabVIEW is the de facto standard for test and measurement software. Its intuitive graphical programming style, similar to flow charts, provides the functionality of a full-featured programming language with an easy-to-use interface. With LabVIEW Real-Time and LabVIEW FPGA, we combined LabVIEW with a real-time operating system and the ability to directly target FPGAs (Field Programmable Gate Arrays) to provide reliability and determinism.

VISION AND MEASUREMENTS IN PACS

With a background in measurements, National Instruments is extending PAC beyond simple I/O by incorporating higher speed measurements and machine vision capabilities. Many industrial applications collect high speed measurements for vibration or power quality applications. The collected data is used to monitor the condition of rotating machinery, determine maintenance schedules, identify motor wear, and adjust control algorithms. The data is normally collected using specialized data acquisition systems or stand alone instrumentation and is incorporated into a control system using a communication bus. National Instruments PACs can directly take high accuracy measurements at millions of samples per second, which are then passed directly into their control systems for immediate processing.

Engineers also can incorporate vision into their control systems. Vision is an area of automation that has gained a lot of momentum in the last decade. In a manufacturing environment, there are many flaws or mistakes that can be identified through visual inspection that are difficult to detect using traditional measurement techniques. Common applications include part inspection for manufacturing or assembly verification, such as checking for correct component placement on a circuit board, optical character recognition (OCR) to examine date codes or to sort products, and optical measurements to find flaws in products or for sorting based on quality criteria. Many plants currently use stand-alone smart cameras that need to communicate to the manufacturing process controller. National Instruments PACs incorporate vision or high speed measurements with logic, and motion control eliminate the need for engineers to integrate dissimilar hardware and software platforms.

PACS ELIMINATE THE NEED FOR CUSTOM HARDWARE

Although PACs represent the latest in programmable controllers, the future for PACs hinges on the incorporation of embedded technology. One example is the ability to use software to define hardware. Field Programmable Gate Arrays (FPGAs) are electronic components commonly used by electronics manufacturers to create custom chips, allowing intelligence to be placed in new devices. These devices consist of configurable logic blocks that can perform a variety of functions, programmable interconnects that act as switches to connect the function blocks together, and I/O blocks that pass data in and out of the chip. By defining the functionality of the configurable logic blocks and the way they are connected to each other and to the I/O, electronics designers can create custom chips without the expense of producing a custom ASIC. FPGAs are comparable to having a computer that literally rewires its internal

circuitry to run your specific application.

FPGA technology has only been available to hardware designers who were proficient in low level programming languages like VHDL. However, controls engineers today can use LabVIEW FPGA to create custom control algorithms that are downloaded onto FPGA chips. This capability enables engineers to incorporate extremely time-critical functions to hardware such as limit and proximity sensor detection and sensor health monitoring. Because the control code runs directly in silicon, it is possible for engineers to quickly create applications that incorporate custom communication protocols or high speed control loops: up to 1 MHz digital control loops and 200 kHz analog control loops.

LABVIEW FOR CONTROL

Because of the capabilities of LabVIEW and the ease-of-use of graphical programming, LabVIEW based PACs are well suited for applications that require:

Graphics. Because a LabVIEW programmer natively builds a user interface, you can easily incorporate graphics and an HMI for control systems.

Measurements (high-speed data acquisition, vision, and motion). National Instruments has a strong history in high-speed I/O, including vision acquisition, so you can incorporate measurements such as vibration or machine vision into your standard control systems.

Processing Capabilities. In some applications, you need specialized control algorithms, advanced signal processing, or data logging. Using LabVIEW, you can incorporate custom control code built using NI or third-party tools, implement signal processing such as JTFA, or log data locally or remotely.

Platforms. With LabVIEW, you can create code that runs a variety of platforms including a PC, embedded controller, FPGA chip, or handheld PDA.

Communication. LabVIEW makes it easy for you to pass data up to the enterprise with tools like databases connectivity, OPC, and operator interfaces via a web browser

NATIONAL INSTRUMENTS PACS

National Instruments offers five LabVIEW based PAC platforms.

PXI is a multi-vendor industry standard PAC based on the CompactPCI architecture that offers a modular, compact, and rugged industrial system. A PXI system is controlled by an embedded controller with a high performance multi-GHz processor. You can choose modules from National Instruments or third-party PXI and CompactPCI vendors. PXI offers the widest range of I/O including 1000 V isolated analog input, high-density digital I/O, analog and digital frame grabbers for machine vision, and coordinated multi-axis motion. It provides easy access to cabling with connectors on the front of PXI modules. The PXI platform offers a broad range of measurement modules, and connectivity to field devices using CAN, DeviceNET, RS-232, RS-485, Modbus, and Foundation Fieldbus.

The Compact FieldPoint product line consists of hot-swappable analog and digital I/O modules and controllers with Ethernet and serial interfaces. I/O modules provide direct connectivity with thermocouples, RTDs, strain gauges, 4-20 mA sensors, and 5-30 VDC and 0-250 VAC signals. Compact FieldPoint network communication interfaces automatically publish measurements through an Ethernet network. You can

access I/O points nearby or miles away on the network using the same simple read/write software framework. With a simple software interface, Compact FieldPoint is quick to setup and program, but provides enough power to perform complex control, data logging, and communications.

The Compact Vision System combines a high performance Intel processor with an FPGA, digital I/O and three 1394 ports. This PAC is designed to incorporate vision into control applications through the use of FireWire (IEEE 1394) technology, compatible with more than 80 industrial cameras. With a reconfigurable FPGA and digital I/O lines on the CVS, you also get low-channel digital and stepper motor control. When programmed with LabVIEW, the system can be configured for both high performance vision and high speed digital control and stepper motor control.

CompactRIO is an FPGA based reconfigurable control and acquisition system designed for applications that require a high degree of customization and high speed control. The architecture combines a real-time embedded processor for complex algorithms and custom calculations with a reconfigurable I/O (RIO) FPGA core. The CompactRIO platform will accommodate up to eight analog or digital I/O modules manufactured by either National Instruments or other companies. The CompactRIO platform is ideal for complex and high speed applications such as machine control and with FPGA is a good option for applications that would normally require custom hardware development.

Standard Industrial PCs can also be used with the wide range of PCI modules manufactured by National Instruments. These plug-in boards include hardware designed for analog and digital I/O, motion control, and machine vision. For deterministic, real-time performance, combine PCI hardware with LabVIEW running on a PC-based real-time operating system. LabVIEW Real-Time can be loaded on most standard Industrial PCs to provide a low-cost platform for industrial measurement and control.

PLC DEVELOPMENTS: THE NEAR FUTURE

Courtesy of Reed Business Engineering staff

Over the past five years, the big story in programmable controllers (PLCs) has really been a “little” one: the vast expansion of the micro and nano PLC market. Tiny controllers as small as a credit card now carries functionality similar to mid-size controllers of the previous decade.

At the high end of the market, PLCs have been seriously challenged by industrial PCs whose data integration and communication capabilities have attracted the attention of users with large requirements in this area.

The PCs brought along with them the idea of Ethernet TCP/IP networking and integration with the Internet. But now the PLC manufacturers are saying, “If that’s what you want, we can do that too.”

Europe’s leading PLC manufacturers, Siemens A&D and Schneider Electric, both provide full-functioned, high end controller lines, in the form of PLCs and their PC equivalents. Improvements in both areas continues.

Siemens recently announced significant performance enhancements in its S7-400 PLC line, with CPUs now offering three times the performance and double the communications power.

And Siemens has added another first to its top-of-the-line S7-400: a communications processor with an integrated, four-port Ethernet switch. It makes external switches unnecessary and can be used for the economical formation of small Ethernet communication cells. But the real significance of this communications processor is that it is an intermediate step between now and the time when Siemens will offer all of its high-end PLCs and peripheral equipment with integrated Profinet V3 switches, which will provide real-time control.

Schneider Electric has greatly increased the capacity of its high-end CPUs, in part to take advantage of its new Unity programming software that uses XML as the backbone of its data. The new CPUs look more and more like PCs, with embedded web servers and multiple communication ports.

INTEGRATED PLATFORM DESIGNS

For Mitsubishi Electric, the developments expected in PLCs over the next two or three years are already apparent: multiprocessor and integrated platform designs. “From the users’ perspective, this means being able to combine several functions onto one single backplane,” says Hugh Tasker, PLC and Networking Specialist. “These could include standard PLC technology being blended with motion and process control and even full-function PC capabilities running a local SCADA or SoftHMI solution.”

Where would this lead? He thinks the greatest benefit for users is the simplification of the architecture of control systems. “Instead of having to integrate many PLCs, wiring them up, commissioning and maintaining them, many systems will be reduced to just a few — or maybe even just one — PLC. The

savings in installation and maintenance time will be massive, and the robustness of the overall system will increase exponentially,” he says.

He likes to speculate even further. “On a longer term basis, we may see wide spread use of Bluetooth and wireless Ethernet. This would probably start off in niche applications but as reliability, robustness and acceptability improve, manufacturers will probably seek to incorporate such wireless technologies... and who knows, maybe one day they’ll remove one of the last bastions of the control panel... the cabling.”

B&R Automation is a high-end controls provider who makes free and open use of the industrial PC. They have done this for longer than most other PLC makers. For them, it was never a discussion of PC vs. PLC; the two were developed cooperatively. But now B&R sees some changes in this direction; perhaps the future will bring more blending of the two technologies.

“In the last few years, the development of classic programmable logic controllers has been moving consistently in the direction of embedded PC technology,” says Anton Meindl, Business Manager Controls, B&R. “The advantages of this PC technology – tremendous computing power, the integration of Internet technology, Fast Ethernet, and USB interfaces – can be combined with the advantages of conventional PLC systems such as their high degree of availability, extreme robustness, and enormous focus on modularity.”

OMRON’S PLC+INVERTER

It wasn’t so long ago that Omron introduced its first AC inverter for its motion control product line, the 3G3MV. Designed for small motor control (0.1 to 7.5 kW), the little inverter is nevertheless a powerhouse. It allows users, for example, to select sensorless vector control for applications requiring high torque at low speeds, and can produce up to 150% torque at 1 Hz. It is also loaded with I/O and communications options.

But last November, it acquired another ‘option’, and this one is a bit more unusual than the rest. Omron added a plug-in PLC module — not just another attachment but a complete PLC equivalent to Omron’s standard CPM2C-S PLC. Using dual-port RAM, the PLC connects directly with the inverter’s I/O, key data, and parameters. The PLC has its own I/O and can provide these data directly to the inverter: encoder input, interrupt inputs, and digital, pulse, and PWM outputs.

The plug-in PLC (it is designated 3G3MV-P10CDT) has its own digital I/O. Of the six inputs, three can be configured as a high-speed up/down counters, which typically are used in conjunction with an encoder to allow the drive to track speed or position. Similarly, the first two outputs can work as pulse generators, making it easy to slave drives together in conveying and similar applications.

The combination, says Omron, makes an ideal control

solution for networked or production systems requiring precise control of positioning and sequencing functions. It is also a solution for manufacturers of multi-step packaging lines, winders, and intelligent conveyor lines.

Omron shows some interesting possible applications. In one, a PLC+inverter is used as a pump sequencer to control water pressure in a three pump system. The PLC+inverter provides continuous closed loop control to the first pump and on/off control to two other pumps in parallel water pipes. An MMI screen gives local supervisory options, and a wireless transmitter sends SMS signals to maintenance personnel.

In another application example, a series of networked 3G3MV-P10CDTs provide distributed control over a small production line; the PLC in this situation would not only control the drive to adjust the speed of the conveyor, but also control machinery at a local station that is picking and placing components into the production operation.

PACS IN AMERICA

In the U.S., where marketing strategy always shines more light than technical capability, the biggest excitement of 2003 was a new buzz word, "Programmable Automation Controller".

This latest addition to the vocabulary comes from Craig Resnick, of the marketing research firm ARC Advisory Group, who commented, "The PLC is very much alive and well, with a long life ahead. However, this industrial workhorse has morphed (changed) in numerous ways to expand its appeal. Automation suppliers continue to improve the PLC to serve market opportunities and specific user needs. The additional functionality has allowed a new class of system to emerge. Programmable Automation Controllers offer open industry standards, extended domain functionality, a common development platform, and advanced capabilities. ARC Advisory Group has coined this new term to help users define their application needs and manufacturers to more clearly communicate the capabilities of their products."

Whether or not manufacturers change the name of their products from PLC to PAC remains to be seen. But GE Fanuc didn't hesitate; it quickly embraced the PAC vision and announced, at Hannover Fair, a new flagship line of controllers it called, rather appropriately, PACSystems. For GE Fanuc the name is supposed to convey the idea that the product is a blend of traditional PLC and a PC hardware, with a universal programming environment, the Simplicity Machine Edition. There is a single software control engine, one development tool, but multiple hardware targets. The hardware targets would include not only the two new PACSystems processors, but GE Fanuc's other automation products such as the panel PC, embedded PC, MMI, and its new motion control system S2K.

"PACSystems represents a revolutionary change in the control industry, one that enables control convergence rather than integration of disparate parts and pieces," explained John Pritchard, President PLC Business, GE Fanuc Automation Europe. "With one engine, coupled with a single development tool, users can take advantage of a powerful engineering environment for multiple applications. Through this innovation, the GE Fanuc PACSystems family addresses major engineering and business issues—such as higher productivity and communications openness."

WE HAVE PACS, TOO

Not to be left out, Rockwell has issued a press release to

declare that it was a "leading provider of PAC technology" and that its ControlLogix and CompactLogix PLCs were, in fact, PACs.

"We've also seen an unprecedented thirst for control system data, as users gather information for safety, finance, quality assurance programs, regulatory control, and countless other initiatives," said Ken Deken, VP, Rockwell Automation Control Systems. "To quench this thirst, what users need is a direct connection between the controller and business systems, ultimately providing information whenever it is needed."

Moving forward, he says PACs will provide a wide array of ways to serve data, become even more scalable, while making major strides in multi-discipline control, integration with business systems, diagnostics and life-cycle cost management.

PROGRAMMABLE LOGIC CONTROLLERS

Courtesy of Grand Valley State University

INTRODUCTION

Control engineering has evolved over time. In the past humans were the main method for controlling a system. More recently electricity has been used for control and early electrical control was based on relays. These relays allow power to be switched on and off without a mechanical switch. It is common to use relays to make simple logical control decisions. The development of low cost computer has brought the most recent revolution, the Programmable Logic Controller (PLC). The advent of the PLC began in the 1970s, and has become the most common choice for manufacturing controls.

PLCs have been gaining popularity on the factory floor and will probably remain predominant for some time to come. Most of this is because of the advantages they offer.

- Cost effective for controlling complex systems.
- Flexible and can be reapplied to control other systems quickly and easily.
- Computational abilities allow more sophisticated control.
- Trouble shooting aids make programming easier and reduce downtime.
- Reliable components make these likely to operate for years before failure.

LADDER LOGIC

Ladder logic is the main programming method used for PLCs. As mentioned before, ladder logic has been developed to mimic relay logic. The decision to use the relay logic diagrams was a strategic one. By selecting ladder logic as the main programming method, the amount of retraining needed for engineers and tradespeople was greatly reduced.

Modern control systems still include relays, but these are rarely used for logic. A relay is a simple device that uses a magnetic field to control a switch, as pictured in Figure 1. When a voltage is applied to the input coil, the resulting current creates a magnetic field. The magnetic field pulls a metal switch (or reed) towards it and the contacts touch, closing the switch. The contact that closes when the coil is energized is called normally open. The normally closed contacts touch when the input coil is not energized. Relays are normally drawn in schematic form using a circle to represent the input coil. The output contacts are shown with two parallel lines. Normally open contacts are shown as two lines, and will be open (non-conducting) when the input is not energized. Normally closed contacts are shown with two lines with a diagonal line through them. When the input coil is not energized the normally closed contacts will be closed (conducting).

Relays are used to let one power source close a switch for another (often high current) power source, while keeping them isolated. An example of a relay in a simple control application

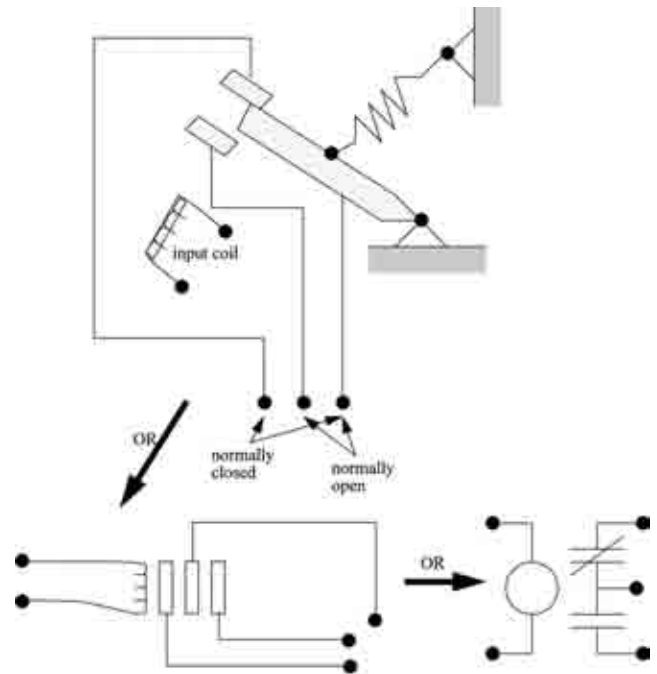


Figure 1 Simple Relay Layouts and Schematics

is shown in Figure 2. In this system, the first relay on the left is used as normally closed, and will allow current to flow until a voltage is applied to the input A. The second relay is normally open and will not allow current to flow until a voltage is applied to the input B. If current is flowing through the first two relays then current will flow through the coil in the third relay, and close the switch for output C. This circuit would normally be drawn in the ladder logic form. This can be read logically as C will be on if A is off and B is on.

The example in Figure 2 does not show the entire control system, but only the logic. When we consider a PLC, there are inputs, outputs, and the logic. Figure 3 shows a more complete representation of the PLC. Here there are two inputs from push buttons.

We can imagine the inputs as activating 24V DC relay coils in the PLC. This, in turn, drives an output relay that switches 115V AC that will turn on a light. Note, in actual PLCs inputs are never relays, but outputs are often relays. The ladder logic in the PLC is actually a computer program that the user can enter and change. Notice that both of the input push buttons are normally open, but the ladder logic inside the PLC has one normally open contact, and one normally closed contact. Do not think that the ladder logic in the PLC needs to match the inputs or outputs. Many beginners will get caught trying to make the ladder logic match the input types.

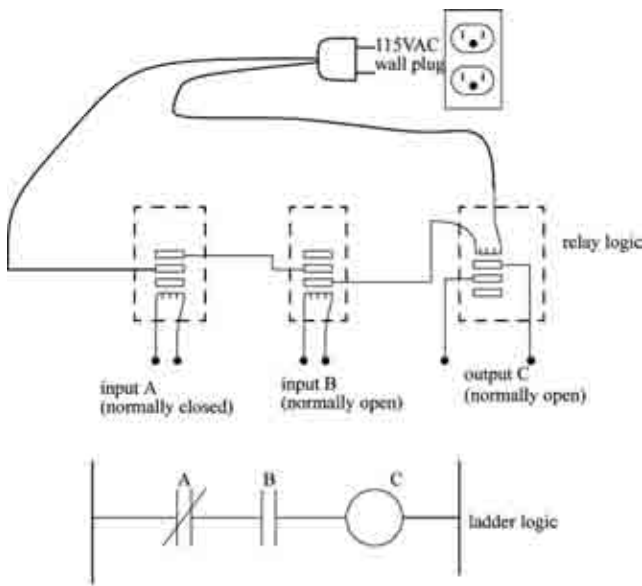


Figure 2 A Simple Relay Controller

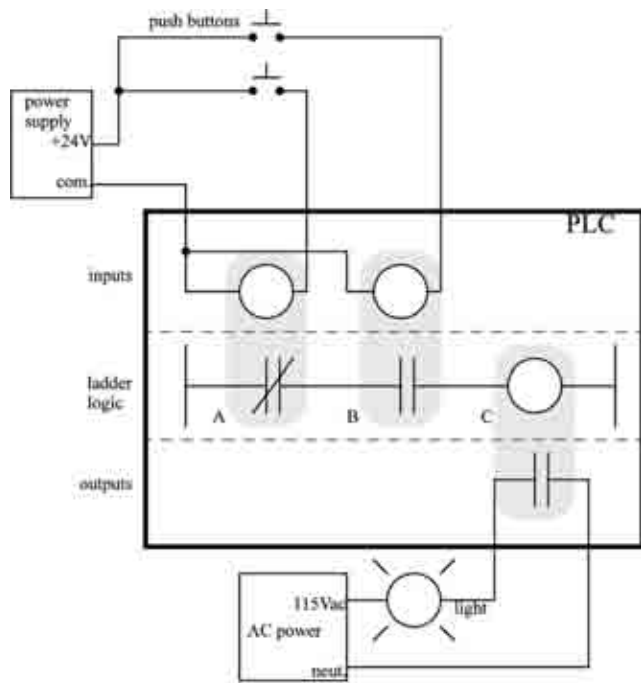


Figure 3 A PLC Illustrated With Relays

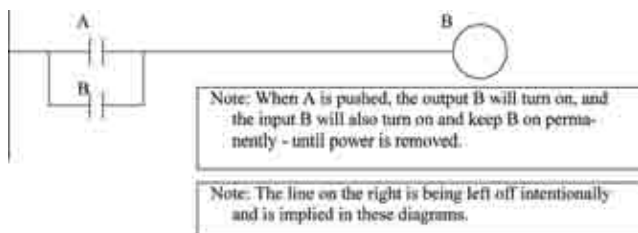
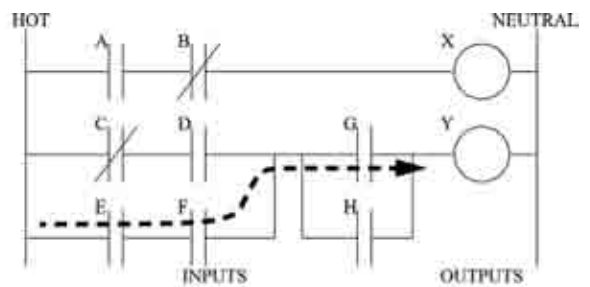


Figure 4 A Seal-in Circuit

Many relays also have multiple outputs (throws) and this allows an output relay to also be an input simultaneously. The circuit shown in Figure 4 is an example of this. It is called a seal in circuit. In this circuit the current can flow through either branch of the circuit, through the contacts labelled A or B. The input B will only be on when the output B is on. If B is off, and A is energized, then B will turn on. If B turns on then the input B will turn on, and keep output B on even if input A goes off. After B is turned on the output B will not turn off.

PROGRAMMING

The first PLCs were programmed with a technique that was based on relay logic wiring schematics. This eliminated the need to teach the electricians, technicians and engineers how to program a computer - but, this method has stuck and it is the most common technique for programming PLCs today. An example of ladder logic can be seen in Figure 5. To interpret this diagram imagine that the power is on the vertical line on the left hand side. We call this the hot rail. On the right hand side is the neutral rail. In the figure there are two rungs, and on each rung there are combinations of inputs (two vertical lines) and outputs (circles). If the inputs are opened or closed in the right combination, to power the outputs, and finally to the neutral rail. An input can come from a sensor, switch, or any other type of sensor. An output will be some device outside the PLC that is switched on or off, such as lights or motors. In the top rung the contacts are normally open and normally closed. Which means if input A is on and input B is off, then power will flow through the output and activate it. Any other combination of input values will result in the output X being off.



Note: Power needs to flow through some combination of the inputs (A,B,C,D,E,F,G,H) to turn on outputs (X,Y).

Figure 5 A Simple Ladder Logic Diagram

The second rung of Figure 5 is more complex, there are actually multiple combinations of inputs that will result in the output Y turning on. On the left most part of the rung, power could flow through the top if C is off and D is on. Power could also (and simultaneously) flow through the bottom if both E and F are true. This would get power half way across the rung, and then if G or H is true the power will be delivered to output Y.

There are other methods for programming PLCs. One of the earliest techniques involved mnemonic instructions. These instructions can be derived directly from the ladder logic diagrams and entered into the PLC through a simple programming terminal. An example of mnemonics is shown in Figure 2.6. In this example the instructions are read one line at a time from top

to bottom. The first line 00000 has the instruction LDN (input load and not) for input 00001. This will examine the input to the PLC and if it is off, it will remember a 1 (or true). If it is on it will remember a 0 (or false). The next line uses an LD (input load) statement to look at the input. If the input is off, it remembers a 0. If the input is on, it remembers a 1 (note: this is the reverse of the LD). The AND statement recalls the last two numbers remembered and if they are both true the result is a 1. Otherwise, the result is a 0. This result now replaces the two numbers that were recalled, and there is only one number remembered. The process is repeated for lines 00003 and 00004 but when these are done, there are now three numbers remembered. The oldest number is from the AND. The newer numbers are from the two LD instructions. The AND in line 00005 combines the results from the last LD instructions and now there are two numbers remembered. The OR instruction takes the two numbers now remaining and if either one is a 1 the result is a 1. Otherwise, the result is a 0. This result replaces the two numbers and there is now a single number there. The last instruction is the ST (store output) that will look at the last value stored and if it is 1, the output will be turned on. If it is 0, the output will be turned off.

The ladder logic program in Figure 6 is equivalent to the mnemonic program.

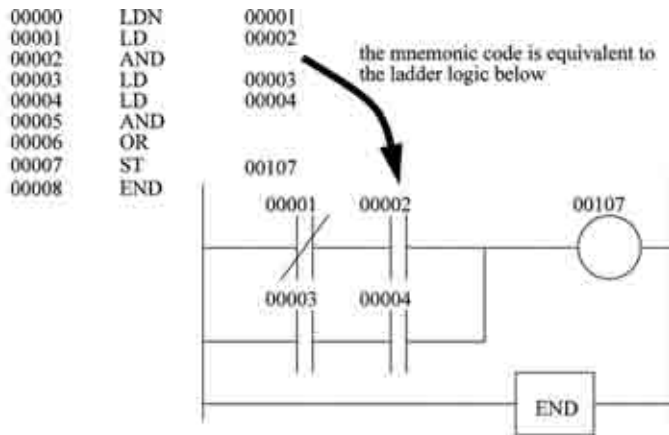


Figure 6 An Example of a Mnemonic Program and Equivalent Ladder Logic

Even if you have programmed a PLC with ladder logic, it will be converted to mnemonic form before being used by the PLC. In the past mnemonic programming was the most common, but now it is uncommon for users to even see mnemonic programs.

Sequential Function Charts (SFCs) have been developed to accommodate the programming of more advanced systems. These are similar to flowcharts, but much more powerful. The example seen in Figure 7 is doing two different things. To read the chart, start at the top where it says start. Below this there is the double horizontal line that says follow both paths. As a result the PLC will start to follow the branch on the left and right hand sides separately and simultaneously. On the left there are two functions the first one is the power up function. This function will run until it decides it is done, and the power down function will come after. On the right hand side is the flash function; this will run until it is done. These functions look unexplained, but each function, such as power up will be a small lad-

der logic program. This method is much different from flowcharts because it does not have to follow a single path through the flowchart.

Structured Text programming has been developed as a more modern programming language. It is quite similar to lan-

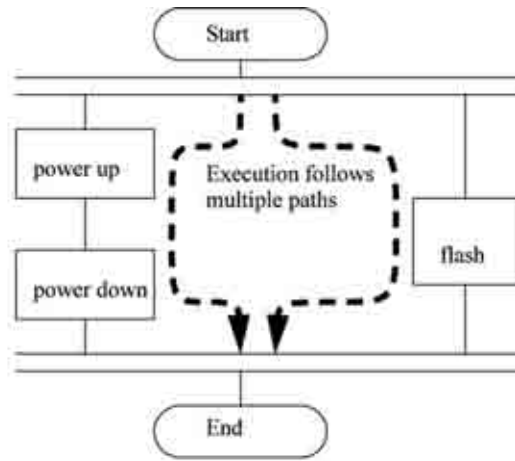


Figure 7 An Example of a Sequential Function Chart

guages such as BASIC. A simple example is shown in Figure 8. This example uses a PLC memory location N7:0. This memory location is for an integer. The first line of the program sets the value to 0. The next line begins a loop, and will be where the loop returns to. The next line recalls the value in location N7:0, adds 1 to it and returns it to the same location. The next line checks to see if the loop should quit. If N7:0 is greater than or equal to 10, then the loop will quit, otherwise the computer will go back up to the REPEAT statement continue from there. Each time the program goes through this loop N7:0 will increase by 1 until the value reaches 10.

```
N7:0 :=0;
REPEAT
N7:0 := N7:0 + 1;
UNTIL N7:0>= 10
END_REPEAT;
```

Figure 8 An Example of a Structured Text Program

PLC CONNECTIONS

When a process is controlled by a PLC, it uses inputs from sensors to make decisions and update outputs to drive actuators, as shown in Figure 9. The process is a real process that will change over time. Actuators will drive the system to new states (or modes of operation). This means that the controller is limited by the sensors available. If an input is not available, the controller will have no way to detect a condition.

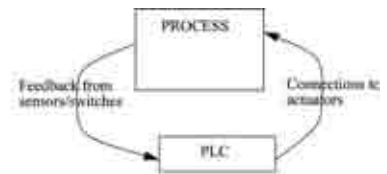


Figure 9 The Separation of Controller and Process

The control loop is a continuous cycle of the PLC reading inputs, solving the ladder logic, and then changing the out-

puts.

puts. Like any computer, this does not happen instantly. Figure 10 shows the basic operation cycle of a PLC. When power is turned on initially the PLC does a quick sanity check to ensure that the hardware is working properly.

If there is a problem, the PLC will halt and indicate there is an error. For example, if the PLC backup battery is low and power was lost, the memory will be corrupt and this will result in a fault. If the PLC passes the sanity check it will then scan (read) all the inputs. After the inputs values are stored in memory, the ladder logic will be scanned (solved) using the stored values - not the current values. This is done to prevent logic problems when inputs change during the ladder logic scan. When the ladder logic scan is complete, the outputs will be scanned (the output values will be changed). After this, the system goes back to do a sanity check, and the loop continues indefinitely. Unlike normal computers, the entire program will be run every scan. Typical times for each of the stages is in the order of milliseconds.

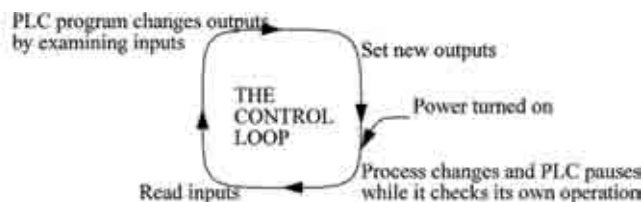


Figure 10 The Scan Cycle of a PLC

LADDER LOGIC INPUTS

PLC inputs are easily represented in ladder logic. In Figure 11 there are three types of inputs shown. The first two are normally open and normally closed inputs, discussed previously. The IIT (Immediate Input) function allows inputs to be read after the input scan, while the ladder logic is being scanned. This allows ladder logic to examine input values more often than once every cycle.

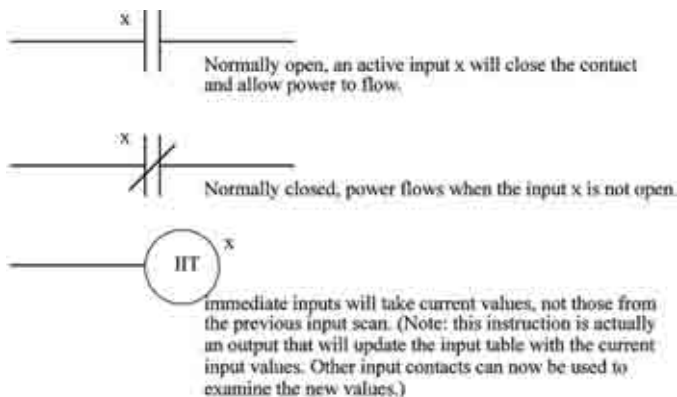


Figure 11 Ladder Logic Inputs

LADDER LOGIC OUTPUTS

In ladder logic, there are multiple types of outputs, but these are not consistently available on all PLCs. Some of the

outputs will be externally connected to devices outside the PLC, but it is also possible to use internal memory locations in the PLC. Six types of outputs are shown in Figure 12. The first is a normal output. When energized the output will turn on and energize an output. The circle with a diagonal line through it is a normally an output. When energized the output will turn off. This type of output is not available on all PLCs. When initially energized, the OSR (One Shot Relay) instruction will turn on for one scan but then be off for all scans after, until it is turned off. The L (latch) and U (unlatch) instructions can be used to lock outputs on. When an L output is energized, the output will turn on indefinitely, even when the output coil is deenergized. The output can only be turned off using a U output. The last instruction is the IOT (Immediate Output) that will allow outputs to be updated without having to wait for the ladder logic scan to be completed.

When power is applied (on) the output x is activated for the left output, but turned off for the output on the right.



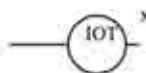
An input transition on will cause the output x to go on for one scan (this is also known as a one shot relay)



When the L coil is energized, x will be toggled on, it will stay on until the U coil is energized. This is like a flip-flop and stays set even when the PLC is turned off.



Some PLCs will allow immediate outputs that do not wait for the program scan to end before setting an output. (Note: This instruction will only update the outputs using the output table, other instruction must change the individual outputs.)



Note: Outputs are also commonly shown using parentheses -() - instead of the circle. This is because many of the programming systems are text based and circles cannot be drawn. Figure 12 Ladder Logic Outputs

TUNING A PID CONTROLLER FOR A DIGITAL EXCITATION CONTROL SYSTEM

By Kiyong Kim and Richard C. Schaefer, Basler Electric

Abstract — Some of the modern voltage regulator systems are utilizing the Proportional, Integral, and Derivative (PID) control for stabilization. Two PID tuning approaches, pole placement and pole-zero cancellation, are commonly utilized for commissioning a digital excitation system. Each approach is discussed including its performance with three excitation parameter variations [1]. The parameters considered include system loop gain, uncertain exciter time constants, and forcing limits. This article is intended for various engineers, and technicians to provide a better understanding of how the digital controller is tuned with pros and cons for each method.

restore the rotor back to its steady state position after a fault. A fast excitation system will also improve relay tripping coordination due to the excitation systems' ability to restore terminal voltage quickly, and providing more fault current to protective relays for optimum tripping time.

I. INTRODUCTION

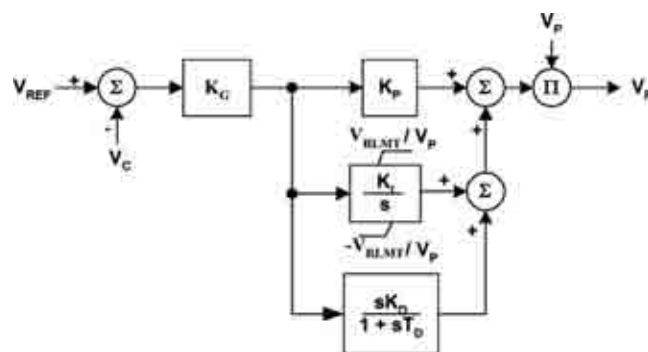
Today's digital excitation systems offer numerous benefits for performance improvements and tuning over their analog voltage regulator predecessors. The days of potentiometers, screwdriver, and voltmeters for tuning are replaced with a laptop computer. Unlike the past, when excitation systems were tuned by analog meters, today's excitation can be tuned very precisely to desired performance and recorded into a file for future performance comparison in the form of an oscillography record.

II. PID CONTROL

The present day digital regulator utilizes a PID controller in the forward path to adjust the response of the system. For main field excited systems, the derivative term is not utilized. The proportional action produces a control action proportional to the error signal. The proportional gain affects the rate of rise after a change has been initiated into the control loop.

The integral action produces an output that depends on the integral of the error. The integral response of a continuous control system is one that continuously changes in the direction to reduce the error until the error is restored to zero. The derivative action produces an output that depends on the rate of change of error. For rotating exciters, the derivative gain is used which measures the speed of the change in the measured parameter and causes an exponentially decaying output in the direction to reduce the error to zero. The derivative term is associated with the voltage overshoot experienced after a voltage step change or a disturbance. The basic block diagram of a PID block utilized in the automatic voltage regulator control loop is shown in Fig. 1. In addition to PID block, the system loop gain (K_G) provides an adjustable term to compensate for variations in system input voltage to the power converting bridge. When performance is measured, the voltage rise time is noted at the 10% and 90% level of the voltage change. The faster the rise time, the faster the voltage response [5].

The benefits of a fast excitation controller can improve the transient stability of the generator connected to the system, or stated another way; maximize the synchronizing torque to



V_{REF} is generator voltage Reference
 V_C is sensed voltage
 V_{RLMT} is max field forcing
 V_P is power input voltage
 V_R is voltage regulator output

Fig. 1: Simplified Block Diagrams of Automatic Voltage Regulators

III. CHARACTERISTIC OF EXCITATION CONTROL SYSTEMS

An optimally-tuned excitation system offers benefits in overall operating performance during transient conditions caused by system faults, disturbances, or motor starting [5]. During motor starting, a fast excitation system will minimize the generator voltage dip and reduce the I²R heating losses of the motor.

After a fault, a fast excitation system will improve the transient stability by holding up the system and providing positive damping to system oscillations.

A fast excitation system offers numerous advantages improved relay coordination and first swing transient stability. However an excitation system tuned too fast can potentially cause MW instability if the machine is connected to a voltage weak transmission system.

For these systems, a power system stabilizer may be required to supplement machine damping.

The evaluation of system performance begins by performing voltage step responses to examine the behavior of the excitation system with the generator. It is performed with the generator breaker open, since the open circuited generator represents the least stable condition, i.e. the highest gain and the

least saturation (See Fig. 2). Fig. 3 represents a bode plot of the generator and exciter field for a sweep frequency from 0-10 Hertz. The frequency plot represents the potential voltage oscillation frequency after a disturbance. The time constant of the generator field and the exciter field is plotted and illustrated to show that as the frequency increases, the phase angle becomes more lagging. The phase angle of the generator field adds directly with the phase angle of the exciter field. As the phase angles add to 150 degrees, the system will most likely become unstable because of the combined gain of the generator and the excitation system. Unless compensated properly through the PID controller, the synchronous machine may become oscillatory after a fault.

Besides the open circuit voltage step test, another test performed is the voltage step test with the generator breaker closed. When voltage step tests are performed with the generator breaker closed, very small percentage voltage steps are introduced to avoid large changes in generator vars. In this case, a 1-2% voltage step change is typical [7].

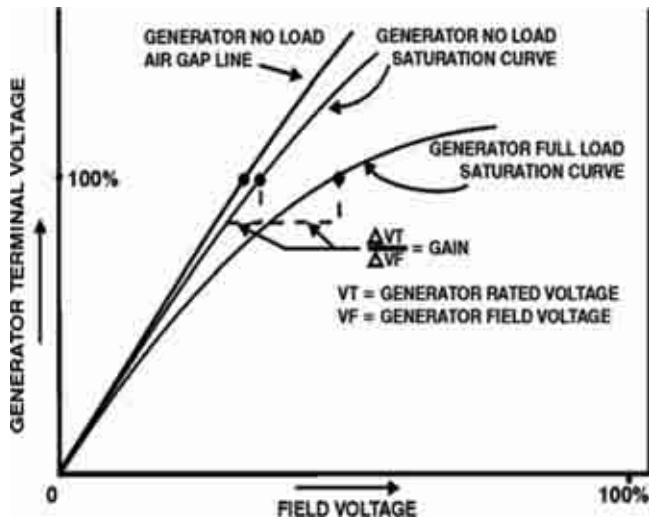


Fig. 2: Generator Saturation Curve Illustrating Generator Gain

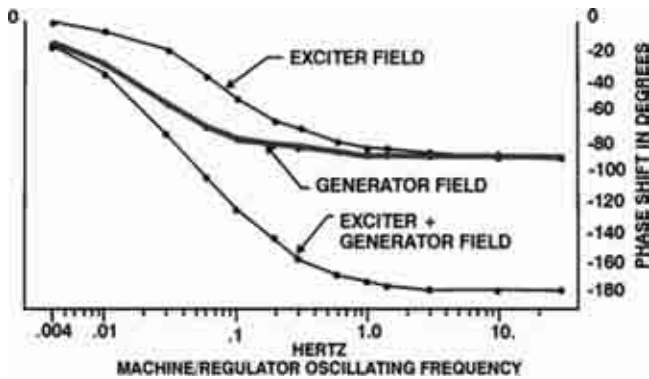


Fig. 3: Phase Shift of the Exciter Field, the Generator Field, and the Sum of the Two

IV. TUNING OF PID CONTROLLER

The controller parameters are determined with several

excitation system parameters, such as voltage loop gain and open circuit time constants [1].

These parameters vary with not only the system loading condition but also gains dependent on the system configuration, such as the input power voltage via PPT to the bridge rectifier as shown in Figure 4.

Commissioning a new AVR can be a challenging task of checking excitation system data in a short time, without any test data, and with no other link to the actual equipment, except for an incomplete manufacturer's data sheet, or some typical data set.

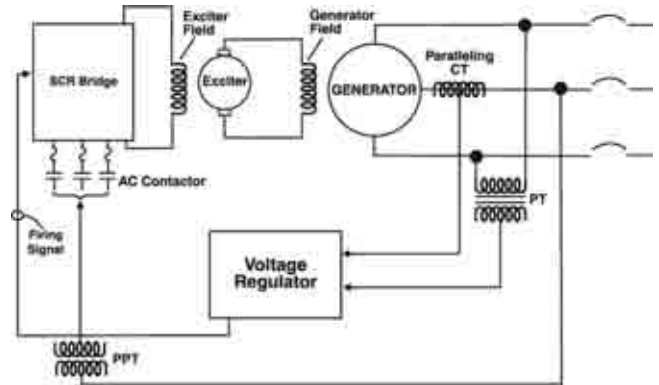


Fig. 4: One-line Diagram

To tune the digital controller, two methods are predominantly used, one being the pole placement method and the other being the pole zero cancellation approach [1, 2]. To simplify the design of the PID controller, we assume $T_D=0$ in Fig. 1.

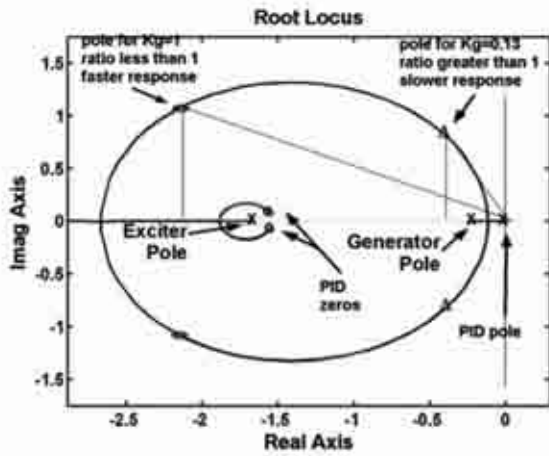
Every PID controller contains one pole and two zero terms with low-pass filter in the derivative block ignored. For generators containing rotating exciters, the machine contains two open loop poles, one derived from the main field and the other derived from the exciter field. A pole represents a phase lag in the system while the zero tends to provide a phase lead component. The location of poles and zeros with relation to the exciter and generator field poles determines the performance of the excitation control system.

Root locus is used to describe how the system responds based upon gain in the system. Using the pole placement method and referencing Fig. 5a, the poles of the generator main field and exciter field are located on the real axis. The generator main field pole is located close to the origin while the exciter field pole is typically tens times the distance, the distance depending upon the time constant of the exciter field versus the main field. The smaller the exciter field time constant, the greater the distance. The PID controller consists of one pole and two zeros.

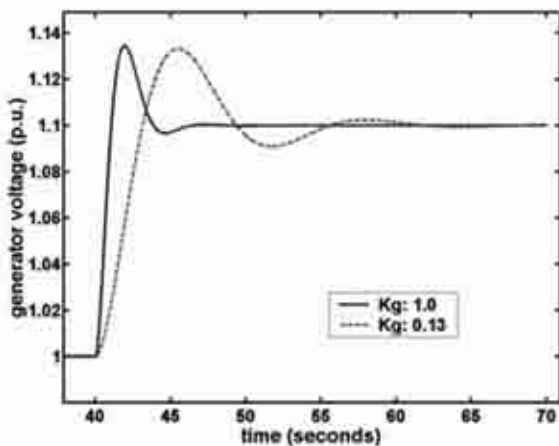
Fig. 5a shows how the closed loop poles move as the loop gain increases. The loop gain, K_G , represents the totalized gain that includes the generator, field forcing of the excitation system and PID controller. The open loop zeros becomes the closed loop zeros and do not depend on the loop gain. On the other hand, the poles of the closed loop system (exciter, generator, controller) are moving toward zeros in a certain path as the loop gain increases. The path of the closed loop poles depends on the relative location of poles and zeros based upon its time constants. With fixed PID gains, a certain system gain (K_G) deter-

mines the closed loop poles. Two cases of the closed poles are shown in Fig 5a, one for system gain 1.0 and the other for system gain 0.13.

In general, the poles nearest to the origin determine the system responses. When the poles become a conjugate pair, the system response will be oscillatory. The conjugate pair represents the ratio of the imaginary to real value of poles to determine the voltage overshoot. Absolute value of pole determines the frequency of voltage oscillation. The faster voltage response can be achieved by moving the poles from the origin and the less oscillatory response with the smaller ratio. See Fig. 5b.



(a) Root Locus of Different Loop Gains
Gains: $K_p=90$; $K_i=70$; $K_D=25$



(b) Step Responses Performance
Gains: $K_p=90$; $K_i=70$; $K_D=25$

Fig. 5: Root Locus and Voltage Response with Two Different System Gains (K_g)

Pole Placement: In the pole placement method, the desired closed-loop pole locations are decided on the basis of meeting a transient response specification.

The design forces the overall closed-loop system to be a dominantly second-order system. Specifically, we force the two dominant closed-loop poles (generator and controller) to be complex conjugate pair resulting in an underdamped response. The third pole (exciter) is chosen to be a real pole and is placed so that it does not affect the natural mode of the voltage

response. The effect of zeros to the transient response is reduced by a certain amount of trial and error and engineering judgment involved in the design.

The pole placement method generally requires specific information of the exciter field and main generator field time constants to determine the gains needed for the digital controller for adequate response. Voltage overshoot of at least 10-15% is anticipated with the pole placement method with a 2 to 3 second total voltage recovery time, although its voltage rise time can be less than one second.

Fig. 6 illustrates the generator terminal voltage performance of a 100 MW steam turbine generator when a +5% open circuit voltage step change has been introduced. Generator voltage overshoot is 20% with a total voltage recovery time of 2.5 seconds. The PID gains are as follows: $K_P=90$, $K_I=70$, and $K_D=29$.

The excitation system bandwidth is used to characterize the response of the generator with the voltage regulator. The wider the voltage regulator bandwidth, the faster is the excitation system.

To derive the voltage regulator bandwidth, the gain and phase shift is plotted over a range of frequencies typically 0-10 Hertz by applying a signal oscillator into the voltage regulator summing point. The input signal is compared to the generator output signal by measuring the phase and gain of the frequency range of 0-10 Hertz, which represents the potential oscillating frequency of the generator interconnected to the system. For these bode plots, information of the generator and exciter time constant along with the excitation system gains are used to determine the bandwidth of the generator excitation system.

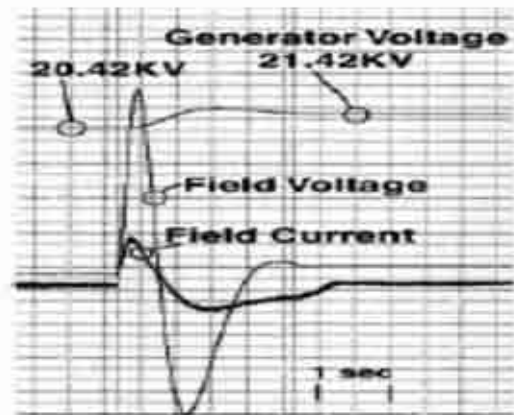


Fig. 6: 5% Voltage Step Response Using Pole Placement Method

A typical root locus with pole placement method is shown in Fig. 7. Notice how the close loop poles move along the circle with increasing gain. Fig. 8 shows the phase lag, -59.1 degrees at -3 db. The db rise prior to roll off confirms the voltage overshoot noted during the voltage step response in Fig. 6.

The ideal excitation system will maintain high gain with minimum phase lag. The point of interest is the degree of phase lag and gain at -3 db, the bandwidth of the excitation system. The less phase lag with higher gain gives the better performance of the excitation system. Fig. 9 highlights the open loop response of the excitation system. It shows the phase lag of -105 degrees at 0db, crossover frequency.

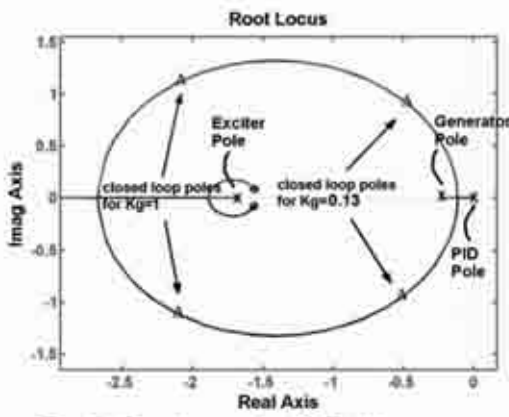


Fig. 7: Root Locus for 100MW Generator Using Pole Placement Method
Gains: $K = 90$; $K = 70$; $K = 29$; $K = 15$

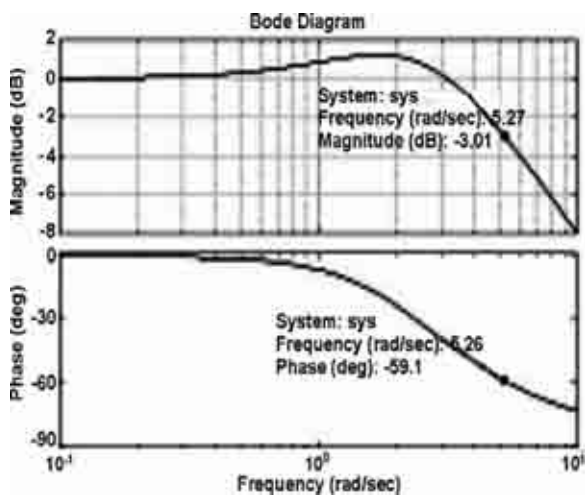


Fig. 8: Closed Loop Bode Diagram Of Pole Placement Method

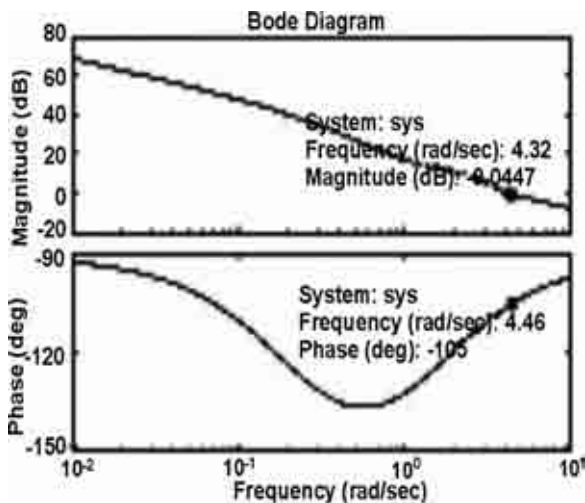


Fig. 9: Open Loop Bode Diagram Of Pole Placement Method

Pole Zero Cancellation: The pole zero cancellation method offers the benefit of performance with minimum voltage overshoot. This method uses the fact that the dynamic behavior of the pole is cancelled if zero is located close to the pole. The PID controller designed using pole-zero cancellation method

forces the two zeros resulting from the PID controller to cancel the two poles of the system. The placement of zeros is achieved via appropriate choice of the PID controller gains. Since exciter and generator poles are on the real axis, controller has zeros lying on the real axis. Unlike the pole placement method that uses high integral gain, a proportional gain is set to be at least four times greater than the integral term.

A typical root locus with inexact cancellation is shown in Fig. 10. The zeros are selected to cancel the poles corresponding to exciter and generator time constants. Thus the closed loop system will be dominantly first-order.

The exciter and generator time-constants vary with the

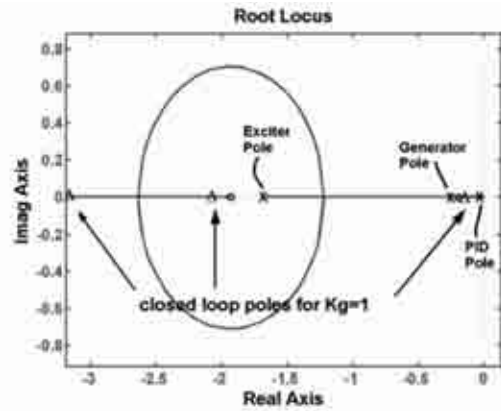


Fig. 10: Root Locus for 100MW Generator Using Pole-Zero Cancellation Method
Gains: $K_p=80$; $K_I=20$; $K_D=40$; $K_B=15$

system condition. The exact pole-zero cancellation is not practical. However, the exciter time constant is much smaller than generator time constant in rotary excitation control systems. As the loop gain increases, the poles move toward the corresponding zeros. Since the two time-constants are well separated, the effect of non-exact pole-zero cancellation is not detrimental.

The pole zero cancellation method provides for an extremely stable system with very minimum voltage overshoot and the field voltage remains very stable even with high K_g loop gains ($K_p=85$, $K_I=20$, $K_D=40$, $T_D=0.01$). Figure 11 shows the voltage response of a 100 MW steam turbine generator using pole-zero cancellation gains.

The closed and open loop bode plots are respectively illustrated in Fig. 12 and 13 when the pole-zero cancellation method is used. The gain remains high and the phase lag is com-

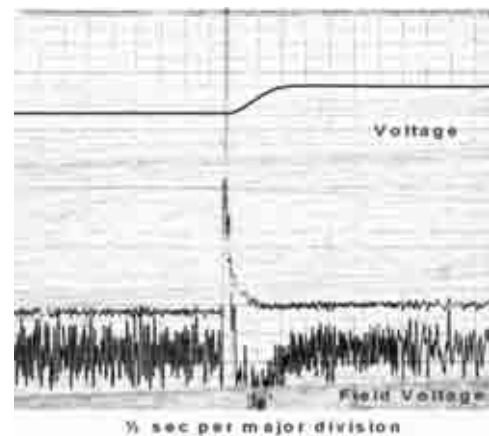


Fig. 11: Voltage Response Using Pole-Zero Method

compensated for a wide frequency. Again using the -3 db point that represents the bandwidth of the excitation system, the phase lag is -46.3 degrees - a very fast excitation system.

Notice the gain remains very flat before it begins to roll-off. The flat response illustrates a very stable system with little to no voltage overshoot. Fig. 13 shows the phase lag of -91 degrees at crossover frequency of zero db.

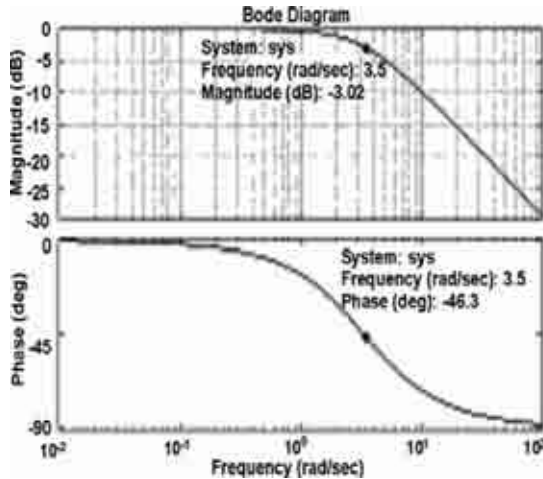


Fig. 12: Closed Loop Bode Diagram of Pole Zero Cancellation Method

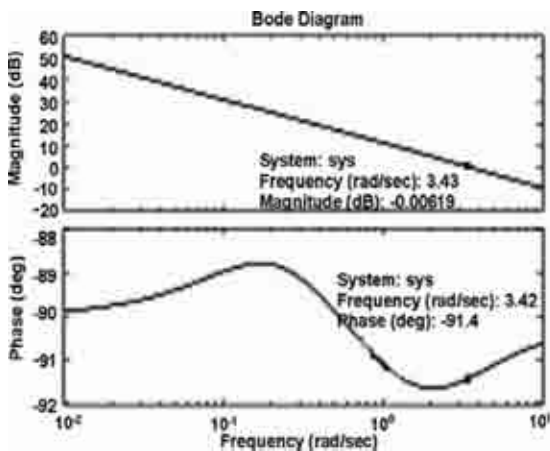


Fig. 13: Open Loop Bode Diagram of Pole Zero Cancellation

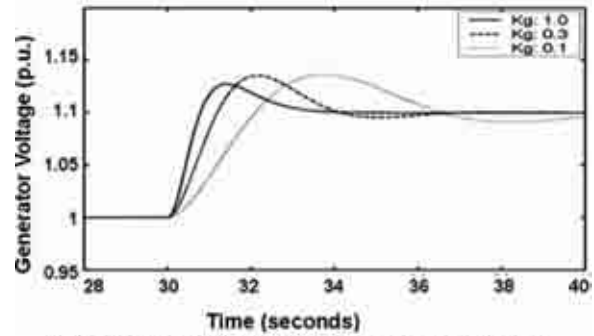
IV. EFFECT OF PARAMETER VARIATIONS

The performance of two methods is conducted using computer simulation in the presence of parameter variations. The parameters considered include system loop gain and uncertain exciter time constant.

Since in general, the calculation of loop gain requires several excitation system parameters that are generally not available during commissioning, specifically the machine time constant, this lack of information can make the use of the pole placement approach more time consuming for setup than pole zero cancellation approach.

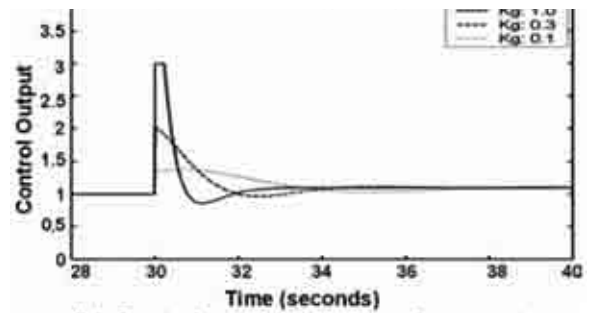
Variation due to uncertainty in the loop gain is considered by variations in KG from the value of 0.1, 0.3, and 1.0. All the other parameters remain unchanged. Fig. 14 illustrates the responses of the two methods described, pole-placement and

pole zero cancellation, while changing the loop gain (Kg) of the digital controller. Note that both Fig. 14-a and 14-c display similar rise time but voltage overshoot only occurs using the pole placement method in this example. The pole-zero cancellation method provides a means to quickly and accurately tune the generator excitation system. Faster voltage response can be achieved by simply increasing the loop gain.

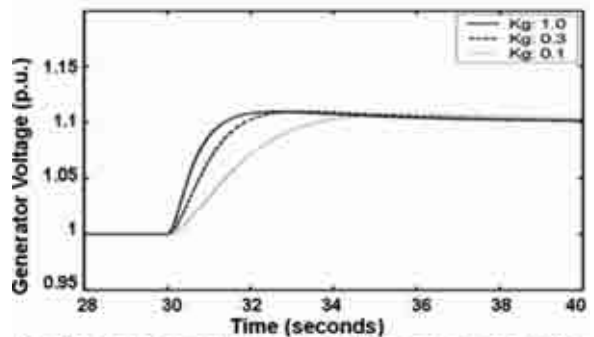


(a) Generator voltages for pole placement

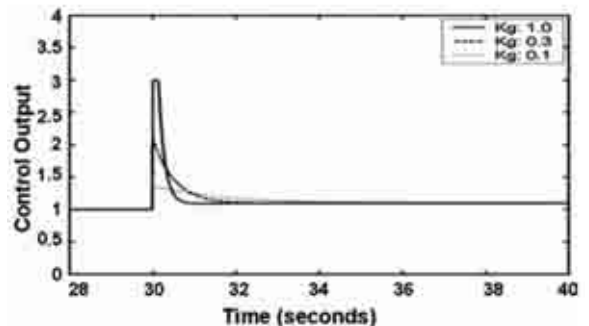
(a) Generator voltages for pole placement



(b) Control outputs for pole placement



(c) Generator voltages for pole zero cancellation

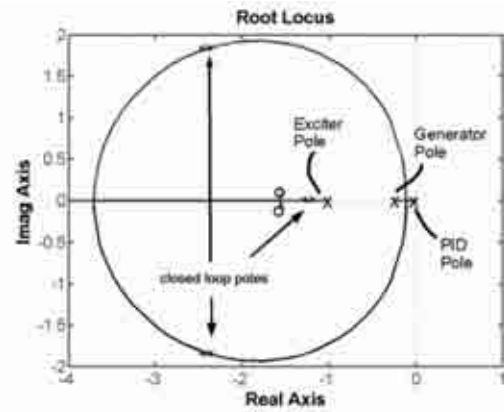


(d) Control outputs for pole zero cancellation

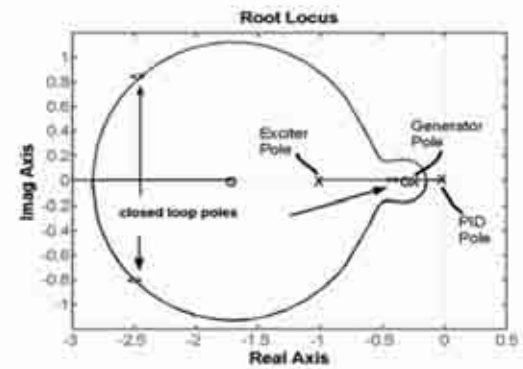
Fig. 14: Step Responses with Loop Gains

Uncertainty in the knowledge of the exciter time constant is considered from the value of 0.2 sec, 0.6 sec, and 1.0 sec (See Fig. 15). All other parameters remain unchanged. The comparison between Fig. 15-a and 15-c indicates the superiority of the performance resulting from pole-zero cancellation design over the pole placement design. The root locus of each method is respectively shown in Fig 16-a and 16-b.

For the pole placement design, the exciter pole at $s=1$ pushes the root locus to the right with small loop gain (KG). On the other hand, the root locus is moved to the left for the pole-zero cancellation design, providing a more stable system.



(a) Pole-placement – Less Stable System, More Voltage Overshoot



(b) Pole-zero cancellation – More Stable System

Fig. 16: Root Locus with Exciter Time Constant 1 sec.

V. SOME TIPS FOR TUNING THE PID CONTROLLER

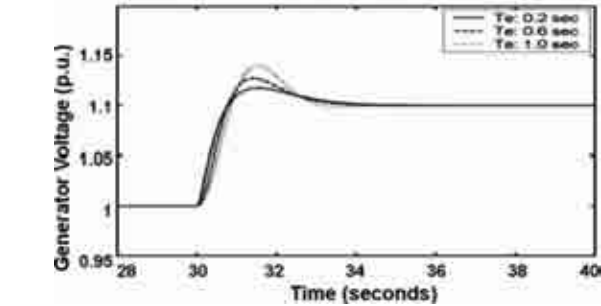
Main Field Excited Excitation Systems: For main field excited systems, since only one field exists, the derivative term is not used. The same gains ratio between the proportional term and the integral term is used with no derivative term applicable [3].

Applying Suitable Gains: The ratio between the integral term and the proportional term needs to be a minimum of four for good performance. Hence, with an Integral gain of 20, the Proportional gain will be 80.

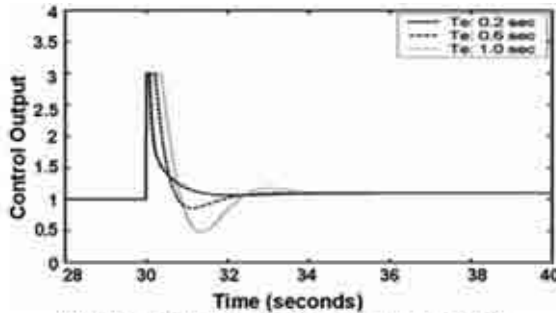
It has been found that it works best to have an Integral gain of not more than 20 in most cases to obtain satisfactory performance [8, 9]. The derivative term will affect the voltage overshoot, and here, a value of 20 will generally be adequate. For faster voltage rise time, the proportional term is increased.

On slow speed hydro machines the generator and exciter time constant tend to be quite large due the slow speed of the turbine. Often the KP term is increased to 150 to help overcome the large inductive lag of the machine's field which will normally slow the voltage reaction time of the machine's response. Fig. 17 illustrates the performance of a 70 MW hydro machine that has a 9 second main field time constant and a 2 second exciter field time constant. Notice the voltage overshoot and settling time [6]. Fig. 17 utilizes a $K_P=150$, $K_I=20$, and $K_D=20$.

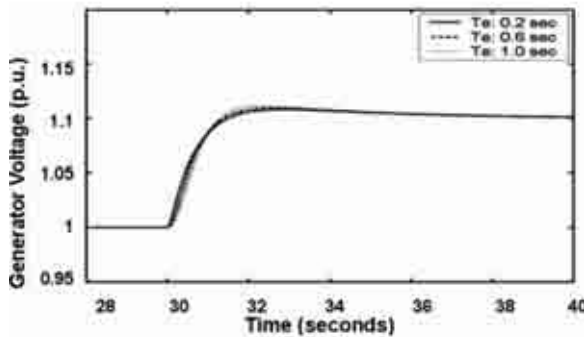
Where the generator such as a hydro, have large field time constant as mentioned above, and the exciter field ratio to main field becomes less, an additional damping factor may be



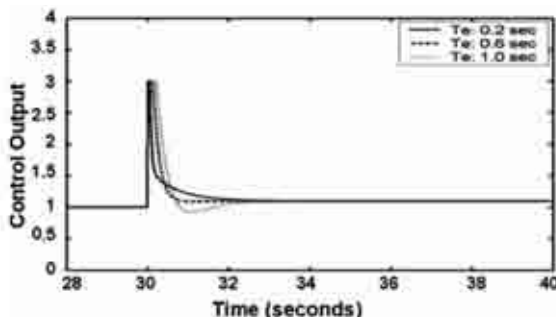
(a) Generator voltages for pole placement



(b) Control outputs for pole placement



(c) Generator voltages for pole zero cancellation



(d) Control outputs for pole zero cancellation

Fig. 15: Step Responses with Exciter Time Constants

required to make the field voltage more damped or stable, known as TD. It is an additional filter that affects the derivative term used with rotating exciter applications to reduce the effect of noise. For these systems where the field voltage may require additional damping, the TD gain value should be applied. Values of 0.01 to 0.08 can be used to reduce the noise content of the field voltage.

Gain for Non Negative Forcing Systems: For systems

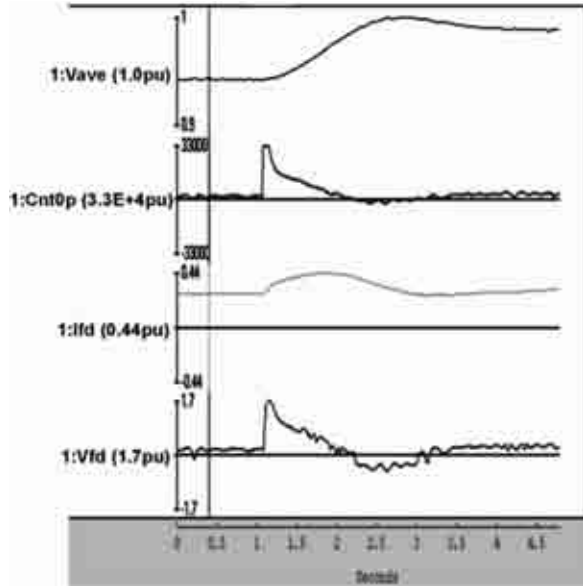
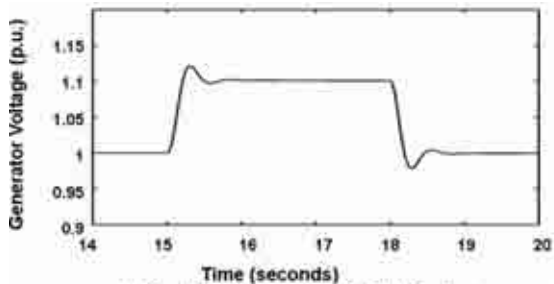
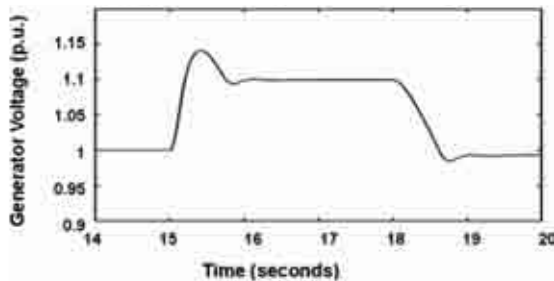


Fig. 17: 5% voltage Step Response, .976 Seconds Voltage Recovery, and .8% Voltage Overshoot

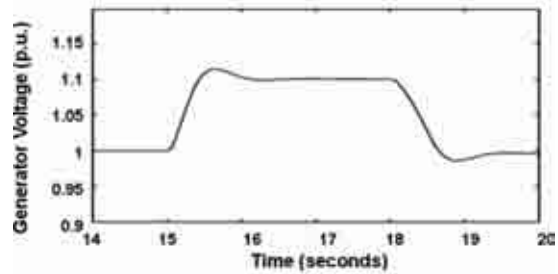
where negative field forcing is not used, the loop gain of the controller should be reduced to achieve best response as compared to systems having negative forcing. Notice in Fig. 18 how voltage overshoot is affected by the same gain terms with and without negative field forcing. To get a proper response for non negative forcing system, gain is reduced with KG from 5 to 1.



(a) KG=5 with negative field forcing



(b) KG=5 with no negative forcing



(c) KG=1 with no negative forcing

Fig. 18. Voltage Step Responses with and without negative field forcing.

VI. CONCLUSION

The pole placement and the pole-zero cancellation methods for designing PID controllers for excitation systems are presented. The sensitivity of the two designs to various uncertainties and parameter variations is presented. Using the pole zero cancellation method of tuning PID gains, commissioning can be accomplished very quickly with excellent performance results.

VII. REFERENCES

- [1] Design Experience with PID Controllers for voltage regulation of synchronous generator, Submitted for publication in the IEEE Transactions on Energy Conversion, 2004, K. Kim, M. J. Basler, and A. Godhwani.
- [2] Commissioning and Operational Experience with a Modern Digital Excitation System Accepted for publication in the IEEE Transactions on Energy Conversion, Vol. 13, No. 2, June 1998, K. Kim, A. Godhwani, M. J. Basler, and T.W. Eberly.
- [3] Application of Static Excitation Systems for Rotating Exciter Replacement, Presented at IEEE Pulp and Paper 1997, R.C. Schaefer.
- [4] Steam Turbine Generator Excitation System Modernization, Presented at IEEE Pulp and Paper 1995, R.C. Schaefer.
- [5] IEEE Std 421.2-1990, IEEE Guide for Identification, Testing, and Evaluation of the Dynamic Performance of Excitation Control Systems.
- [6] Voltage Versus Var/Power Factor Regulation on Hydro Generators, Presented at IEEE/PSRC, 1993, R.C. Schaefer.
- [7] IEEE Std 421.4 2004, IEEE Guide for Specification for Excitation Systems.

USING PLDS FOR HIGH-PERFORMANCE DSP APPLICATIONS

Courtesy of Altera Corporation

INTRODUCTION

Design engineers face the challenge of designing increasingly high performance communications systems in less time with fewer resources. Additionally, these designers must consider rapidly emerging/changing technologies. The wide range of performance needs for today's applications requires a solution that is flexible and easy to implement. One solution is programmable logic devices (PLDs), which provide flexibility, high performance, and fast time-to-market.

Designers are using PLDs more frequently for digital signal processing (DSP) applications. PLDs are found in a wide range of products, from high-performance wired applications such as digital subscriber line access multipliers (DSLAMs) to emerging 3G wireless applications. Designers use PLDs in DSP applications because of their flexibility and hardware-based performance.

To harness the true capabilities of PLDs, designers need a complete design environment with which they can go from system architecture to hardware implementation in a short time. A seamless design flow, like the one provided with the Altera DSP Builder and intellectual property (IP) MegaCore functions, extends the performance and cost benefits of programmable logic.

COMPARING DSP PROCESSORS, ASSPS, ASICS & PLDS

Engineers have various options when implementing DSP applications, including:

- DSP processors
- Application-specific integrated circuits (ASICs)
- Application-specific standard products (ASSPs)
- PLDs

Traditionally, designers chose DSP processors for digital signal processing applications (DSP). DSP processors have a general-purpose architecture, shown in Figure 1, that makes them flexible for a variety of applications. However, their flexibility ultimately limits their system performance.

Historically, DSP processors have had only one multiplier, but today, some have up to 8 multipliers. They use iterative looping for more than 2 to 8 multiplications and need additional clock cycles to calculate the result.

Therefore, DSP processors are most suited for back-end signal processing at low data rates. Many DSP processors have multipliers with special instructions to speed up the math calculations; however, they lack real-time performance. DSP processors are flexible and can be used for filtering or modulating applications by changing the processor's software code.

ASSPs and ASICs, which are designed to implement a specific function, have better performance than DSP processors for a low cost. These qualities make them attractive for designers. Because ASSPs are semi-custom integrated circuits that perform specific functions, such as finite impulse response (FIR) and infinite impulse response (IIR) filters, their performance is better than other hardware solutions on a similar process technology. However, ASSPs are inflexible and must be redesigned if the DSP application changes. ASICs provide a customizable, low-cost solution. However, they have long lead times—typical design cycles are 1 to 1.5 years—and require a minimum purchase quantity. Small design changes incur additional non-recurring engineering costs and result in a longer design cycle.

PLDs offer compelling advantages over DSP processors, ASSPs and ASICs. Designers can configure PLD logic to process complex routines in parallel or in serial like DSP processors (see Figure 2). In parallel, they offer greater performance than DSP processors by executing the equivalent of hundreds of instructions at once. Unlike ASSPs and ASICs, PLDs provide the flexibility to make design changes without sacrificing time-to-market.

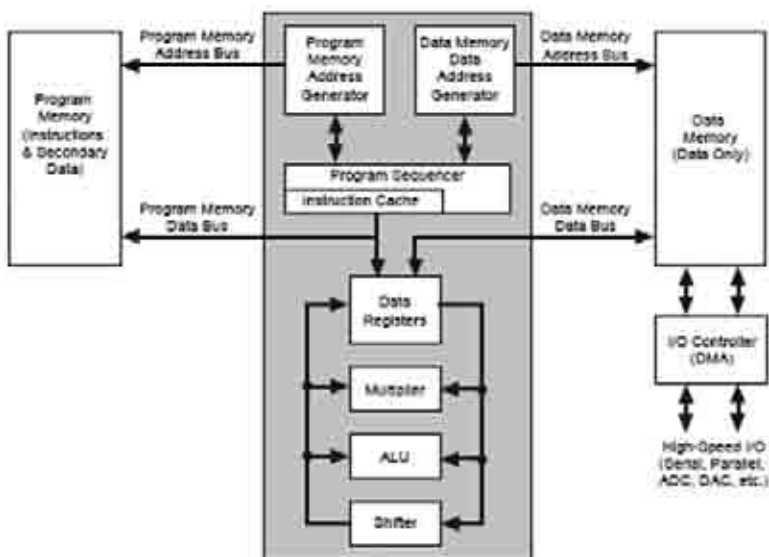


Figure 1. General DSP Processor Architecture
Source: The Scientist and Engineer's Guide to Digital Signal Processing

INNOVATIVE SUPPORT FOR DSP ALGORITHMS

DSP applications use algorithms such as fast Fourier transform (FFT), FIR, IIR, matrix multiplication, correlation, etc. Most of these algorithms are mathematical calculations that

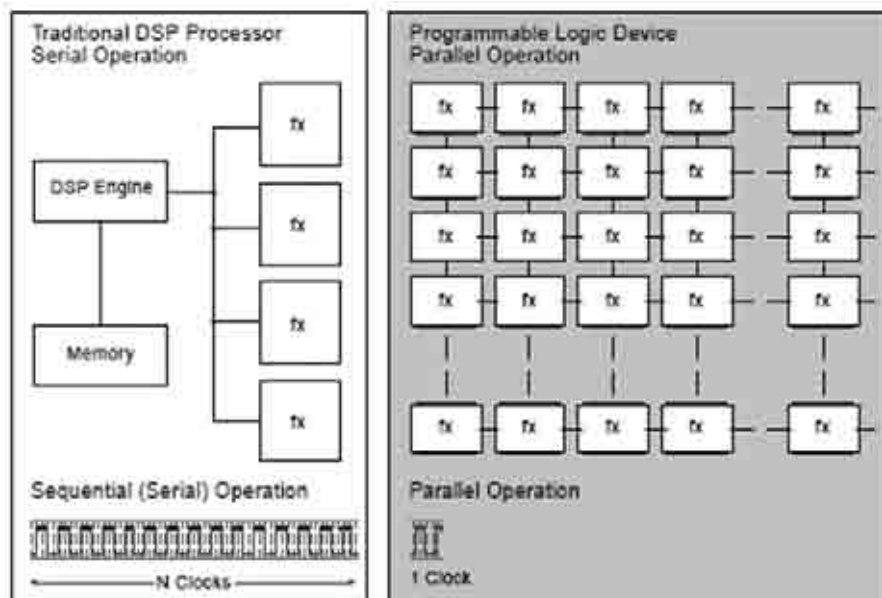


Figure 2. Architectural Differences between PLDs & DSP Processors

combine multiplication followed by an addition (e.g., $y = a \times b + c \times d$), which is called a multiply-accumulate (MAC) operation.

Altera Stratix PLDs combine the flexibility of programmable logic with the raw performance and capabilities of dedicated functional blocks, making the devices an ideal choice for next-generation wireless infrastructure systems. Stratix DSP blocks have hardware multipliers, adders/subtractors, accumulators, and pipeline registers. Flexible, efficient, and optimized for DSP applications requiring high data throughput, DSP blocks are ideal for the wireless communication, telecommunication, video, and image processing markets.

Stratix hardware multipliers offer high performance and

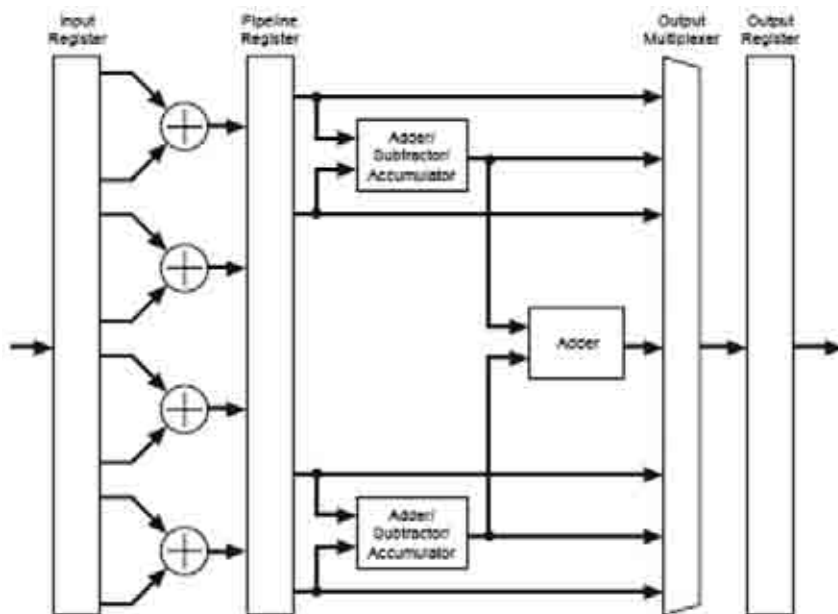


Figure 3. Stratix DSP Block Architecture

low resource utilization when implementing mathematical calculations. The DSP blocks contain dedicated multipliers of varying widths and the multiplier outputs can feed an adder/subtractor or an accumulator. Figure 3 shows the Stratix DSP block architecture. Additionally, Stratix embedded memory blocks can store data for DSP applications such as FIR filter coefficients.

DESIGN FLOW

DSP system design in Altera PLDs requires both high-level algorithms and hardware description language (HDL) development tools. The Altera DSP Builder integrates these tools by combining the algorithm development, simulation, and verification capabilities of The MathWorks MATLAB and Simulink system-level design tools with Altera development tools, HDL synthesis, and simulation.

To win more DSP designs, PLD vendors need to win over DSP engineers by providing a familiar design flow. Altera provides one such tool, the DSP Builder, which interfaces The MathWorks industry-leading, system-level DSP tool Simulink with Altera's Quartus II development software. DSP Builder provides a seamless design flow in which designers perform algorithmic design in the MATLAB software, system integration in the Simulink software, and port the design to hardware description language files for use in the Quartus II software.

Using the DSP Builder, designers can generate an RTL design and RTL testbench from Simulink automatically.

These pre-verified RTL output files are optimized for use in Altera's Quartus II design software so that the designer can perform quick timing and simulation comparisons. The design flow also enables floating-point to fixed-point analysis. With this simple and intuitive development flow, designers do not need experience using programmable logic design software.

Designers can use the signal processing IP blocks provided with the DSP Builder to create a hardware implementation of a Simulink system model. The DSP Builder contains bit- and cycle- accurate Simulink fixed-point blocks, which cover basic operations such as arithmetic or storage functions as well as complex functions such as forward error correction, filtering, and modulation.

ALTERA DSP IP PORTFOLIO

Algorithms that can benefit from PLD performance include filtering, forward error correction, modulation/demodulation, and encryption. Altera has a comprehensive portfolio of standard DSP functions, optimized for Altera PLDs, that designers can license to build system-on-a-programmable-chip (SOC) designs.

Alternatively, designers can use DSP IP cores individually to improve systems that require focused performance enhancements. For systems that require higher throughput, designers can instantiate Altera DSP IP cores using dedicated hardware in parallel.

With the performance advantages of parallel processing and the traditional flexibility of PLDs, DSP cores such as the FIR Compiler and Reed-Solomon Compiler are ideal for emerging applications such as multi-channel multipoint distribution services (MMDS) and orthogonal frequency division multiplexing (OFDM) systems. For example, the Altera Reed-Solomon MegaCore function decodes at rates up to 1 Gbps for 8-bit symbols. With nominal buffering and control overhead, a Reed-Solomon solution decodes at a rate of over 10 Gbps. In contrast, preliminary Texas Instruments benchmarks show that the C64xx DSP processor requires approximately 1,095 cycles to decode one Reed-Solomon codeword (source: Texas Instruments TMS320C6414 Fixed-Point Digital Signal Processor data sheet and TMS320C64x DSP Benchmarks page on the Texas Instruments web site). At 300 MHz, the C64xx processor decodes approximately at a rate of 450 Mbps using 100% of the device’s available processing power.

DESIGN EXAMPLE: FIR FILTER IMPLEMENTED IN STRATIX DEVICES

The following example — an 8-bit 224-tap FIR — illustrates the performance advantages of Stratix devices for DSP applications. Each register provides the unit sample delay. The

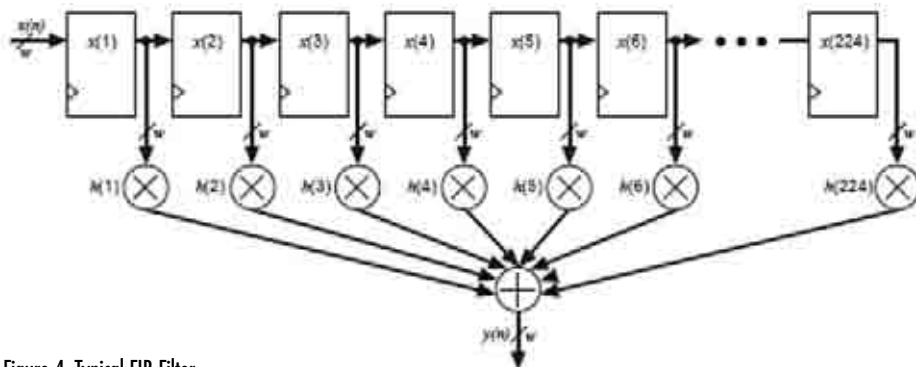


Figure 4. Typical FIR Filter

delayed inputs are multiplied with their filter coefficients and added together to produce the output. See Figure 4.

The FIR filter operation can be represented by the equation:

$$y(n) = \sum_{i=1}^n nx(n)h(n)$$

where x(n) and h(n) are the nth sample values of the input signal and the filter coefficient.

This 224-tap filter can be implemented in Stratix DSP blocks and the design fits into a single Stratix PLD. The design uses only a single clock cycle for the data output. In contrast, the

same filter design requires multiple DSP processors because a single DSP processor has only 2 to 8 multipliers. Table 1 shows the performance results of these two implementations. The Stratix device improves overall system performance almost twelve times.

Table 1. Stratix Device & DSP Processor Performance Comparison

Requirement	Stratix Device (EP1S120)	TI 320C64x Device (1)
Number of Taps	224	224
Number Of Multipliers	224 (28 DSP Blocks)	Maximum of 8
Internal Clock Speed	250 MHz	600 MHz
Clock Cycles Needed to Compute the Result	1	28
GMACs per second	56 GMACs	4.8 GMACs

Note:
 (1) Source: Texas Instruments TMS320C6414 Fixed-Point Digital Signal Processor data sheet.

CONCLUSION

Altera PLDs, such as Stratix devices, are very flexible and provide an efficient solution for hardware implementation of DSP application. Stratix devices have flexible, dedicated DSP blocks that support a wide variety of configurations, which makes it easy to implement DSP applications. Additionally, the DSP blocks have efficient routing that provides fast performance. The high-performance DSP blocks and on-chip memory allows designers to maximize system throughput. PLDs also support parallel processing, making them a powerful tool for increasing overall system performance.

To support a seamless design flow, Altera provides valuable tools, such as MegaCore functions and the DSP Builder, that designers can use to target a system design to PLDs. The Altera signal processing portfolio includes proven, high-performance, standard functions created to help engineers meet these design challenges and to implement a solution on a single Altera PLD. The DSP Builder provides a seamless design flow in which designers can perform algorithmic design in MATLAB, carry out the system integration in Simulink, and port the result to an HDL design for use with the Quartus II software.

REFERENCES

Smith, Steven W. The Scientist and Engineer’s Guide to Digital Signal Processing, 2nd ed. California Technical Publishing, San Diego, CA.

1999.

WEB-ENABLING YOUR PLC

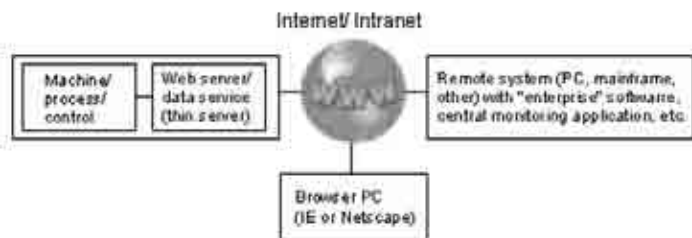
By Mike Rothwell, Director of Advantech Automation Corp.'s eAutomation Engineering Center

Enough has been written during the past couple years to convince most plant managers of the benefits of connecting plant floor equipment and processes to the Internet or Intranet. So, now that we're all convinced of the benefits of web-enabled automation, what pieces and parts are required to put this technology to work? The basic parts required for web-based data acquisition and control are:

- an interface to the machine or process to be monitored or controlled via the web (network) connection;
- a web server to make the desired display and/or control pages available to the remote browser or client; and
- a data service or interface to handle exchanging data between the local machine/process (server) and the remote system (client).

For remote viewing of the data and/or web pages, the only requirement is a standard browser interface. For applications requiring SPC, optimization, or enterprise level software to exchange real time data with the machine/process, a remote server PC and a compatible data exchange service are required.

In a simplified block diagram form, the parts might look like the figure below.



In this system, the web server/data service device ("thin server") provides the connection between the machine or process and the Internet. In a typical installation, a connection to an existing PLC or proprietary controller will be required to extract data or enable control over the equipment. Most PLCs, including products from ABB, Schneider/Modicon, and Siemens, support at least a serial connection using communication protocols available from the PLC vendor or third-party suppliers. In many cases, the communication driver will be available from the thin server vendor as part of the embedded software application.

If the equipment uses a standard PLC protocol or supports another open standard protocol such as Modbus RTU or TCP, then the job of connecting the thin server is usually greatly simplified. However, this is not always the case.

If the equipment in question uses a non-standard PLC or a proprietary controller, a connection to the thin server device is still possible as long as the controller supports some kind of serial protocol and the vendor can provide the protocol documentation. In this situation, either the thin server vendor or a third-party developer will be involved to implement the driver required to connect the thin server to the equipment controller.

In the extreme case, the equipment to be monitored may have a controller with no external communication port, or the protocol may be unavailable for some reason. In this situation, it may be necessary to externally instrument the equipment or process using additional sensors and I/O. These I/O devices can be installed to monitor strategic data or control points on the equipment. If this is required, the I/O devices should be chosen so that they support a standard communication protocol (e.g. Modbus or an open ASCII protocol) and can be connected to the thin server. Alternatively, an I/O device may be selected that includes an integral thin server function. If the I/O device includes an integrated server, then the separate thin server may not be required.

With the thin server installed in the equipment enclosure and the proper driver installed or selected, the next step is to configure the thin server to link the equipment communication data and control (inputs and outputs) to the network link. The details of this step vary depending on the features and configuration software provided with the thin server. For example, a gateway server may simply map PLC registers to network variables or a remote PC connection. More sophisticated thin servers may allow configuration of a web page to view equipment data, enable alarm monitoring and data trending, add standard connectivity interfaces such as OPC, and may even support paging on programmable equipment fault conditions. Make sure that the thin server selected for the project supports an adequate feature set to allow for future expansion.

Here is a close-up of the chiller control cabinet, showing the chiller control PLC and the embedded web server (gray box). The serial cable connecting the web server to the PLC for data exchange can be seen, as well as the blue Ethernet LAN connection for Intranet



Chiller Box (courtesy of Dumont Associates)

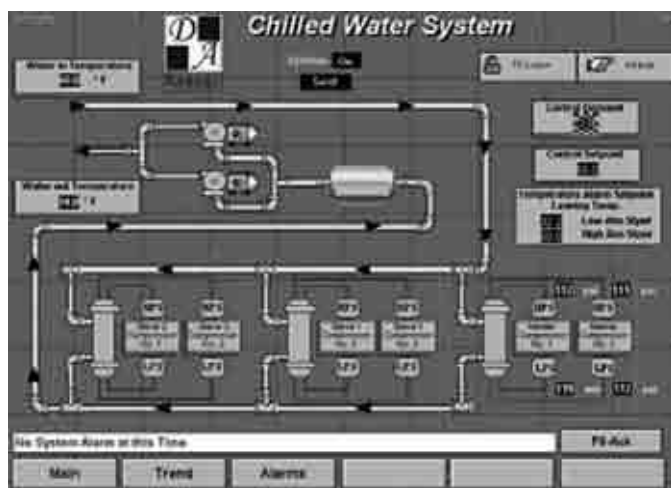
The Internet or Intranet (LAN) connection to the thin server may include a standard hardwired Ethernet line, a modem/phone line (dialup), or a wireless connection (802.11b Ethernet, for example). The type of connection is typically determined by the type of thin server and the existing plant network infrastructure – or lack of network infrastructure.

If a hardwired LAN already exists near the equipment to be connected, then a network connection is relatively simple with the addition of another cable drop to the equipment. If a hardwired connection is difficult or expensive due to complex cable runs or distances, a wireless connection may be the best alternative. However, when selecting wireless connections, be sure to carefully evaluate the proposed installation site and consider factors such as proper range and signal integrity. If using

a wireless network, look for a thin server that directly supports a wireless LAN interface to simplify the installation at the equipment site.

With a LAN connection established, the remote system with its application software can now access the machine equipment at the thin server's assigned IP address via the factory network. Applications such as an SPC package can now take advantage of two-way data exchange with the equipment, made possible by the thin server. If the thin server and remote application software support a common interface, such as OPC, setting up the data exchange can usually be accomplished in a matter of minutes.

Using this same Internet connection, a browser PC (desktop, laptop, PDA, or a "thin client") can access a web page resident in the thin server. This web page can be used to monitor data from the equipment as well as send data to the equipment via the thin server. And, since the web page is resident in the thin server, no software is required on the browser other than Internet Explorer or Netscape.



Chiller Screen (courtesy of Dumont Associates)

This is the main chiller system overview screen, as viewed from a remote PC via a browser. This web page with live data from the embedded web server gives a quick overview of the chiller status and allows adjustment of setpoints and other parameters remotely with proper login credentials.

As a side benefit to this implementation, if the only requirement of the browser PC is to view web pages from the thin server, the PC can be solely dedicated to this task. In this case, the browser "PC" can be a very simple device. It requires only an LCD, CPU, and a small amount of memory; a Thin Client device. This can result in a lower installed cost and lower maintenance cost than a standard PC. Another advantage is that the device is not limited to viewing web pages from only the thin server. It can also view web pages from any source accessible via the network. So, a browser PC or Thin Client located on the shop floor could be used by an assembler to access HTML help files, with graphics and instructions, located on a remote server. The ruggedness and low cost of this type of thin client browser make it possible to provide data and control access in places previously not possible with standard or even industrial PCs.

For data exchange with the application software running on a remote system (as shown in the diagram) a common "language" is required. Several possibilities exist, including OPC,

mentioned previously, as well as XML. XML, or Extensible Markup Language, is a standard defined by the World Wide Web Consortium (W3C) that provides for a method of exchanging data between systems via the Internet and is an important element of Microsoft's .NET architecture. The data is exchanged along with a "tag" that defines the data in a manner that is independent of the sender's and receiver's hardware platform, OS, and application. This makes XML very powerful when implementing open systems within a company or between different companies, as in business-to-business applications.

And, since web enabled automation is all about enabling the use of data within and between companies to improve processes and reduce costs, XML is a very good fit in this distributed web architecture. Choosing a thin server that supports XML can be a real benefit to future plant floor connectivity.

Now that we've seen the power of web enabled automation and reviewed how it can be implemented, where do we go from here? Let's take the sample architecture from the previous section and see how it could be implemented using eAutomation products from Advantech Automation.

Thin server: Advantech's new WebLink was designed for this task. WebLink is a complete "intelligent embedded server" solution, including all hardware and runtime software required to web enable a system. It can connect to a device (machine/process controller, I/O, sensor, etc.) using a standard RS-232/485 serial port or an optional fieldbus adapter. A network connection is then made through WebLink's standard Ethernet 10/100BaseT port or via optional modem or wireless network/Internet connections. Development software enables web pages and data connections to remote application software to be easily created and maintained from anywhere via the network connection. Security is provided by WebLink through password protected user login and optional restricted access by user IP.

For applications where a local HMI is required to be web enabled, Advantech offers the WebOIT operator interface terminal. This product series combines the features of WebLink with an integrated LCD and HMI software functionality.

Browser/Thin Client: There are several choices here, depending on the need. Any standard laptop or desktop with browser software can access the web pages resident in WebLink, providing instant access to information about your equipment, plant, process, building, or business worldwide. For access from the plant floor or other locations requiring a ruggedized viewer, a standard industrial HMI from Advantech's Industrial Panel PC line can be used to provide browser capability along with full PC expandability and functionality. If full PC functionality is not required, Advantech's WebView can be a very cost-effective solution. WebView is a thin client terminal that supports viewing of remote web pages via its integrated Ethernet connection and standard Internet Explorer software. WebView also supports third-party or custom HMI applications based on Windows CE or Linux, including support for Java.

Web enabled automation does have real benefits and provides real competitive advantages for your enterprise. And, with solutions like WebLink, WebOIT, and WebView it can be easy and cost effective to make web enabled automation a reality.

This article was written and provided by Mike Rothwell, director of Advantech Automation Corp.'s eAutomation Engineering Center. He founded the R&D and product group in Cincinnati's Industrial Automation Group headquarters.

INPUT SIGNAL EDGE RATE GUIDANCE

Courtesy of Altera Corporation

INTRODUCTION

Large and fast PLD designs must have good signal integrity to function properly. Input signal edge rates that are not well constrained (either too fast or too slow) can degrade signal integrity and even cause functional failures in the design.

For clock input signals, very slow clock edges pick up large amounts of switching noise from the board and the device.

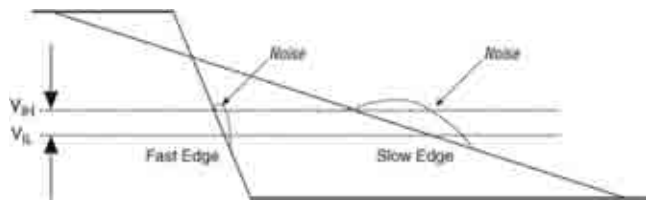


Figure 1. Noise Energy on Fast & Slow Clock Edges

This can cause signal integrity problems, such as false triggering on flip-flops. Figure 1 shows how it is possible for a fast edge and a slow edge in the same noise environment to pick up significantly different amounts of noise energy in the transition region. Large amounts of noise energy in the transition region of a clock can flip the logic state of a gate controlled by the clock and cause false triggering on flip-flops.

In addition to the noise problem, slower clock edges are more susceptible to jitter, which can reduce already tight timing margins in high-speed designs.

For data input signals, very fast edge rates cause simultaneously switching input (SSI) noise problems on wide data buses. Cross talk problems can also occur.

It is important to control input signal edge rates to avoid signal integrity issues. This white paper provides some guidance about input signal edge rates when designing with Altera devices.

GUIDANCE

The maximum rise and fall times for input signals are application dependent and vary based on the system and device noise and on the timing margins on the interface. Because of this dependency, Altera does not provide the maximum rise and fall time specifications for the following devices:

- Stratix II, Stratix, and Stratix GX devices
- Cyclone II and Cyclone devices
- HardCopy series devices
- APEX II and APEX 20K devices
- MAX II devices

EDGE RATE RECOMMENDATION FOR CLOCK & ASYNCHRONOUS CONTROL INPUT SIGNALS

Because clock and asynchronous control input signals such as reset and interrupt signals are critical in system designs, their integrity requires special attention. To avoid potential signal integrity problems and excessive clock jitter, the edge rates of these input signals should be as fast as possible.

In addition to fast edge rates for clock and asynchronous control inputs, it is important to minimize switching noise on boards and devices. If possible, shield clock and asynchronous control inputs with programmable ground pins for the best signal integrity performance.

Altera's MAX II devices have built-in Schmitt triggers for each input I/O pin. Schmitt triggers improve noise immunity on input signals. Altera recommends that designers turn on the Schmitt trigger for input pins with slow input signals.

EDGE RATE RECOMMENDATION FOR DATA INPUT SIGNALS

Potential cross talk and simultaneously switching input noise problems can occur in data input signals, especially wide bus data signals, that have fast signal edges. Altera recommends that the edge rates of the simultaneously switching inputs be fast, but not exceed the recommended maximum edge rates listed in Table 1.

For input signals with slower edge rates than recommended, Altera recommends that the designer do the following, if the feature is available on the device, to avoid noise sensitivity issues:

- Set programmable ground pins on the adjacent output pins.
- Turn on slow slew rate on the adjacent output pins.
- Turn on the Schmitt trigger on the input signal.

Device	Number of Simultaneously Switching Inputs per Bank	Recommended Maximum Data Input Signal Edge Rate
Stratix II	90	2.0 V/ns
	72	3.0 V/ns
Stratix and Stratix GX	90	1.0 V/ns
	72	1.5 V/ns
Cyclone II	From 64 to 72	0.8 V/ns
	From 32 to 36	1.0 V/ns
	From 16 to 18	1.5 V/ns
Cyclone	All I/O pins	1.0 V/ns
MAX II	All I/O pins	1.0 V/ns

The numbers shown in Table 1 are recommendations only. A faster edge rate may work fine, depending on your application. The actual performance of a device in a system is fully system dependent.

SUMMARY

Input signal edge rate requirements are application dependent. Altera recommends using as fast an edge rate as possible for clock and asynchronous control input signals to avoid potential signal integrity issues. For wide data bus input signals, Altera recommends that designers set a fast edge rate, but not exceed the maximum edge rates shown in Table 1.

REMOVABLE STORAGE MEDIA ADD FLEXIBILITY TO MODERN DAY PLCs

By Mark DeCramer, Advanced Electronics Product Manager, WAGO Corporation

The question “Why can’t my PLC interface easily to off-the-shelf devices?” is being answered by a new wave of PLCs with standard interfaces to common peripheral devices. Without sacrificing the deterministic operation, reliability, and ease of use of the traditional PLC, modern day PLCs are now incorporating more and more PC features than ever before. One such feature becoming commonplace in this new generation of PLCs is an interface to removable storage media. Interfaces to memory devices such as USB memory devices and CompactFlash are providing flexibility never before realized in PLC architectures.

Removable storage media bring flexibility to data collection and recipe storage applications in PLCs. For data collection, the PLC no longer needs to communicate directly to an external device, but can operate stand-alone, storing data to the removable memory device only to be retrieved at a later date. Data can still be collected by an external device, or can be retrieved by an onsite service technician. The technician can walk up to the PLC, remove one memory card replacing it with another, and walk away. Because these memory devices are capable of holding gigabytes of data, weeks, months, or even years of collected data can be stored to a single device. Similarly, an operator can change the recipe used in a process, or change the profile used in a motion application by swapping one storage device for another.

Data collection and recipe storage have always been challenging for PLCs because of their inefficiencies in sharing data with other devices. PLCs have long been used for data collection, and while the PLC may be good at collecting data, sharing that data with external devices is typically a cumbersome task. The PLC, being register-based, could only share information as blocks of raw unformatted data. Accurate management of these blocks of registers is critical, as the smallest addition or omission by the PLC significantly changes the meaning of the data to the external device. Similarly, downloading recipes that define process parameters to a PLC requires the same register management.

The block of registers in the PLC that is receiving the recipe data needs to be identical to the incoming data from the host device. The slightest variance gives a whole new meaning to the process being controlled.

A USB or CompactFlash memory device formatted as file memory offers the benefit of being able to hold data stored not as raw register data, as in the PLC, but as user-specified formatted strings of data. The formatting and syntax of the data is programmable and entirely up to the user. Data can be encrypted, or stored in commonly used formats, such as CSV (Comma Separated Value).

The CSV format is a commonly used way of transferring data between programs. It is typically used for spreadsheet and database applications because it is well-suited to data arranged in tables that are, made up of rows and columns. Files of the CSV format are often referred to as comma delimited. A delimiter is one or more characters used to separate data – in this case, a comma.

As an example, last week’s daily weather recorded in a CSV formatted file may look like this:

```
Monday,          75,Sunny
Tuesday,         72,Partly Sunny
Wednesday,       76,Sunny
Thursday,        68,Rain
Friday,          69,Cloudy
```

Most PLC programming software packages contain instruction sets for working with STRING variables, making it easy to manipulate data and/or text into files of various formats. PLCs with real-time clocks provide commands for embedding date and time values in data strings.

The example at the bottom shows how to create the first record of the daily weather data file:

Commands familiar to VB and C programmers are used to open files on memory devices, and read, write, and append data. Multiple files can usually be opened simultaneously, and commands are often supported for creating file folders on mem-

Example using Structured Text programming language in CoDeSys development software

```

0001 Record := 'Monday';      ( Begin the data string with the day of the week )
0002
0003 Record := CONCAT(Record, ','); ( Append a comma to separate the data )
0004
0005 Record := CONCAT(Record, WORD_TO_STRING(Temperature)); ( Append the string value of the temperature )
0006
0007 Record := CONCAT(Record, ','); ( Append a comma to separate the data )
0008
0009 Record := CONCAT(Record, 'Sunny'); ( Append the word 'Sunny' to describe the day's conditions )
0010
0011 Record := CONCAT(Record, '$N'); ( Append a new-line character to indicate the end of the record )

```

Example using Structured Text programming language in CoDeSys development software

Example using Structured Text programming language in CoDeSys development software

```

0014 (*Open File, Append Record to existing file contents, and Close File *)
0015 FileID:=SysFileOpen(filename,'a');
0016
0017 IF FileID <> 0 THEN
0018     SysFileWrite(FileID, ADR(Record), LEN(Record));
0019     SysFileClose(FileID);
0020 END_IF
0021

```

Example using Structured Text programming language in CoDeSys development software

ory devices to better organize data.

As you can see, the Structured Text programming language of the IEC-61131 standard is well suited for manipulating data strings. However, other supported languages such as Ladder Diagram or Function Block Diagram can also be used to perform the tasks shown above.

SUMMARY

The PLC is not dead, but has taken on new life incorporating several of the features that have allowed PCs to creep down to the plant floor. Interfaces to memory devices such as USB memory devices and CompactFlash give the PLC enhanced data sharing capabilities, including gigabyte volumes of data storage, portability, and control over data syntax. Instruction sets in today's IEC-61131 programming development systems support data formatting, read and write file operations, and other data organization tools for conveniently sharing data with analysis tools such as spreadsheet and database applications. Without sacrificing the deterministic operation, reliability, and ease of use of the traditional PLC, removable storage media are adding new flexibility to modern day PLCs.

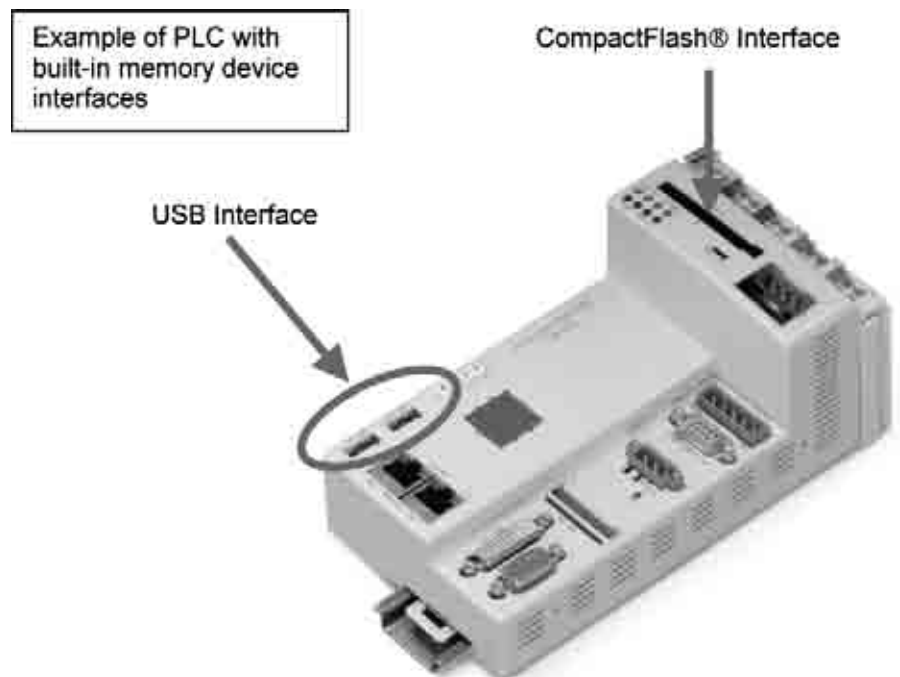
WHAT IS COMPACTFLASH?

CompactFlash is a very small removable mass storage device. First introduced in 1994 by SanDisk Corporation, CF cards weigh a half ounce and are the size of a matchbook. CompactFlash cards are designed with flash technology, a non-volatile storage solution that does not require a battery to retain data indefinitely. CompactFlash storage products are solid state, meaning they contain no moving parts, and provide users with much greater protection of their data than conventional magnetic disk drives. They are five to 10 times more rugged and reliable than disk drives, and consume only five percent of the power required by small disk drives. CompactFlash memory devices can hold several gigabytes of data.

WHAT IS USB?

Universal Serial Bus, or USB, is a standard designed to eliminate the guesswork in connecting peripherals devices. USB is a connectivity specifi-

cation developed by the USB Implementers Forum and was originally aimed at peripherals connecting outside the computer in order to eliminate the hassle of opening the computer case for installing cards needed for certain devices. USB interfaces have since found their way into PLC devices, providing this same hassle-free interconnection to peripheral devices. USB provides for ease of use, expandability, and speed for the end user.



Example of PLC with built-in memory device interfaces

ASSESSMENT AND REMEDIATION OF VULNERABILITIES IN THE SCADA AND PROCESS CONTROL SYSTEMS OF UTILITIES

Courtesy of Internet Security Systems, Inc.

OVERVIEW

Most SCADA and Process Control Systems being used by today's utility companies were developed years ago, long before public and private networks or desktop computing were a common part of business operations. As a result, the need for online security measures within these systems was not anticipated. At the time, good security for SCADA systems meant limiting and securing the physical access to the network and the consoles that controlled the systems. Planners rationalized that if they were suitably isolated from any physical entryways, and if access was limited to authorized personnel only, the systems were fully secure and unlikely to be compromised.

The increasingly networked and linked infrastructure of modern SCADA systems has rendered those early security plans obsolete.

As utility companies have added new applications, remote access points and links to other control systems, they have introduced serious online risks and vulnerabilities that cannot be addressed by their physical control policies. Often, these risks are underestimated due to the complexity of the network architecture, the lack of formal network security guidelines and assumptions about the privacy of the network. Organizations are now realizing the security of these systems means more than physically separating the system and the components they control and monitor.

In fact, online vulnerabilities may pose as much risk for potentially significant failures within a power generation system as a physical attack.

METHODOLOGY

Performing a comprehensive security assessment of power generation, distribution and transmission facilities requires special considerations beyond that of typical information technology (IT) security assessments.

The basic structure of power transmission and distribution systems covers a wide range of components and geographic area. Clearly, the utility company holds the most critical role in the assessment process as the generation and transmission facilities form the backbone of the entire system. But the operation of such a large and diverse infrastructure requires an extensive network of electronic devices, communications and control and monitoring systems, such as:

- Devices in the transmission, substation and generation networks to gather information and effect control:
 - Remote Terminal Units (RTU)

- Intelligent Electronic Devices (IED)
- Programmable Logic Controllers (PLC)
 - Management systems to monitor and control field equipment:
 - Generation plant Distributed Control Systems (DCS)
 - Distribution Energy Management Systems (EMS) and SCADA systems
 - Communications methods to connect management and field devices:
 - Ethernet, Wireless, Serial
 - Modbus, ICCP, DNP

GENERATION FACILITIES

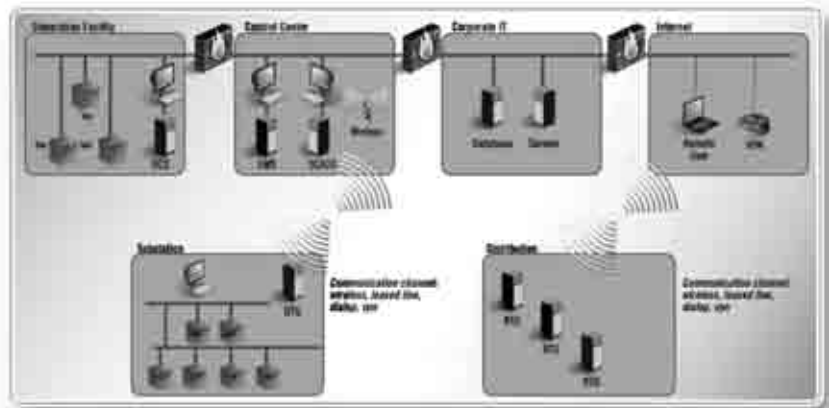


Diagram 1 represents these systems and the interconnections between them at a macro level. Performing a comprehensive security assessment involves evaluating security mechanisms at each component.

The electronic security environment in generation facilities centers on the management of various control devices throughout the facility through a Distribution Control System (DCS). The primary concern with generation facilities is the security of the DCS, as it serves as the nerve center for the entire facility. The security issues represented here are very traditional - matters of access control, system configuration, network access and policy.

However, these issues are compounded by the addition of specific control devices, such as Intelligent Electronic Devices, Programmable Logic Controllers or even custom control systems. The complex network of connections (serial, wireless, ethernet, etc.) must also be evaluated to ensure appropriate devices retain access and that communication security checks are in place to limit control to strictly authorized mechanisms.

CONTROL CENTERS (EMS, SCADA)

Centralized control systems such as Energy Management Systems (EMS) and SCADA systems provide similar management of the transmission and distribution network. In turn, the same fundamental system security issues (network connectivity, system configurations and access control present across DCS management systems and the IT infrastructure) apply to EMS/SCADA systems.

These devices must also be thoroughly evaluated using the same criteria as would be applied to plant control devices.

It is important to note that substation and distribution networks are at much higher risk of unauthorized remote access, since they require remote management. For example, a Remote Terminal Unit's (RTU) controlling devices commonly utilize modems with weak or no authentication. Some of these networks may even be Internet addressable and connected back to the control center through Virtual Private Networks (VPNs) that can be easily compromised. Wireless connectivity is also widely used for access to these devices, which increases the potential threat of data interception.

NETWORK ARCHITECTURE

Understanding the network architecture of utility systems is critical to an effective evaluation of their security posture. As mentioned earlier, all of these systems are now highly interconnected, with DCS and EMS/SCADA control systems feeding data back to corporate systems and remote access between sites and networks provided over VPN connections. ISS teams have found that even basic firewall protection isn't in place on many of these connections. Even more alarming is the prospect of undetected access to critical control networks via a remote access device in the field.

At the lower level, device and control component network architecture and communication can also pose security threats. The placement of devices on the network can allow access to a much broader audience than the organization intends. As more control devices support Ethernet, TCP/IP, and UDP/IP technologies, it is becoming much easier to access them without any physical presence. Control protocols are also standardizing with the introduction of protocols such as IEC61850, DNP, and modbus, making it relatively straightforward for an unauthorized user to decipher and manipulate device command sets.

This network data flow diagram provides a sample high-level illustration of a Power

Generation network infrastructure topology from an access point and data flow perspective.

COMMON VULNERABILITIES

The following provides a sample of common vulnerabilities that the Internet Security Systems SCADA analysis teams have tested and validated for utility clients. These vulnerabilities include, but are not limited to:

- Use of X-Windows in a clear text environment without being tunneled in an encrypted fashion, which allows for the data to be sniffed, keystrokes captured and data in the X-Display to be accessed.

- Use of clear text protocols, such as Telnet, FTP and NFS, without a secure encryption method.

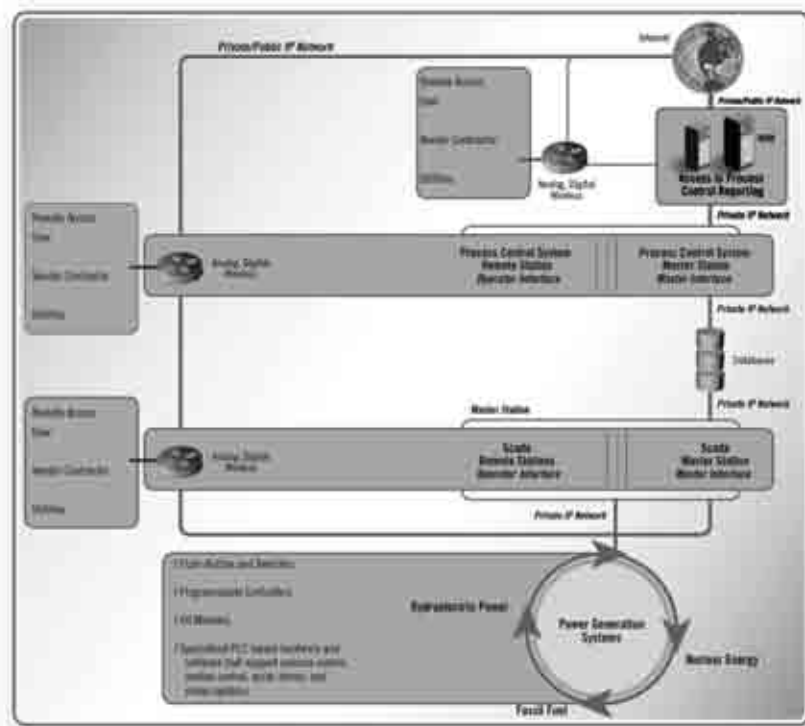
- Unnecessary, exploitable network services running on hosts, such as "SendMail" and "Finger".

- Remote Access Vulnerabilities - SCADA and Process Control System networks often function on a primary network of T-1 connections linking each related facility, with additional connections coming into the corporate network. Because remote access is often permitted to the corporate Internet connection, there is the potential to access the "separate" SCADA networks as well. Additionally, modem access is often provided for emergency maintenance. Because this remote access over the Internet is not always based on a Virtual Private Network (VPN), which uses strong two-factor authentication, authentication and access controls on these remote points are unacceptably weak. The connections are therefore prone to unauthorized exploitation and represent potentially dangerous access points to internal networks - namely, SCADA and Process Control System networks.

- Network Security Architecture - the network infrastructure environment within which SCADA and Process Control Systems reside has often been developed and modified based on

business and operational requirements, with little consideration for the potential security impacts of the changes. This means that over time, security gaps may have been inadvertently introduced within particular portions of the infrastructure and may not have received appropriate remediation. These gaps may represent a backdoor, or even a front door, into SCADA and Process Control Systems.

Robust security architecture in this area will be of particular concern to the ongoing success of the organization given that sensitivity of the SCADA and



Process Control Systems.

- Lack of formal and documented SCADA and Process Control System Policies, Processes and Procedures - Due to the highly proprietary and legacy nature of these systems, owners/administrators and vendors often do not follow strict configuration change management procedures. This may result in a lack of appropriate reviews, auditing and due diligence in day-to-day operational and maintenance procedures. This condition may lead to security oversights, which may again lead to serious exposures and risks.

This list represents just a few of the potential physical and logical risks associated with SCADA and Process Control Systems. The SCADA analysis team of ISS recommends that an organization should include these and other criteria in any assessment of SCADA vulnerabilities and/or an organization's broader security posture.

SCADA PROTECTION SOLUTIONS FROM ISS PROVENTIA INTRUSION PREVENTION SYSTEMS

SCADA systems are vulnerable to a wide array of internet threats attributable to the standard operating systems and network infrastructure upon which they rely, and to SCADA protocol specific weaknesses. Working "Ahead of the threat," Proventia intrusion prevention systems block intrusion attempts, denial of service (DoS) attacks, malicious code, backdoors, worms and a growing list of new hybrid threats affecting SCADA systems.

Backed by X-Force vulnerability-focused research, Proventia intrusion prevention systems feature Virtual Patch technology which provides out-of-the-box multi-layered protection for more than 830 high-risk and critical vulnerabilities or attacks by default. To minimize risk and maximize performance, suspicious traffic is analyzed using multiple traffic identification and analysis techniques, preventing the infection and unauthorized use of network resources. This preemptive protection is crucial for the security of mission critical SCADA systems and important in supporting regulation compliance and avoiding cleanup costs associated with security incidents.

Virtual Patch technology is a key feature of Proventia Intrusion Prevention Systems. A Virtual Patch provides an impervious shield for newly discovered vulnerabilities until a vendor-developed patch can be tested and installed. This "mean time between vulnerability discovery and patching" can be translated into real ROI savings for Proventia appliances by reducing the time spent in testing and patching affected systems (estimated savings of 260 work days per year for a large enterprise).

SCADA ASSESSMENT SERVICES

Internet Security Systems deploys a team of highly qualified security engineers to ensure a comprehensive approach to analyzing the vulnerabilities of SCADA systems. ISS will coordinate the SCADA and Process Control Systems Assessment closely with the client company to ensure an efficient use of resources and minimal impact on personnel.

Our proposed assessment of a client's SCADA system involves a comprehensive information gathering approach that includes, but is not limited to:

- System assessment and evaluation
- Facilities reviews and personnel interviews
- Vendor information
- Functional evaluations and analysis of vulnerabilities.

SCADA ASSESSMENT TASKS

PHASE 1: PLANNING AND DISCOVERY: SCADA AND PROCESS CONTROL SYSTEMS INFRASTRUCTURE

An essential ingredient to any security effort is an understanding of the underlying organizational, procedural and programmatic (application and network) framework. To accomplish this, ISS will begin with the following key tasks:

- Arrange and execute a "SCADA and Process Control Systems Assessment Planning Session" with key client management and operational stakeholders for the purpose of:
 - Obtaining sufficient preliminary information about the current security environment
 - Disclosure of our assessment plan and methodology
 - Obtaining concurrence on assessment procedures
 - Finalization of the Assessment Plan
- Perform an Assessment Process Walkthrough and obtain sign-offs from the Site IT Manager and/or Risk Manager at the commencement and completion of each SCADA and Process Control System review.
 - Conduct preliminary review of applicable policies and procedures that define and provide guidance for SCADA and Process Control System design, operations and security.
 - Determine organizational roles and responsibilities related to SCADA and Process Control System management, maintenance and security.
 - Review SCADA and Process Control System designs and the categorization of generated data and information.
 - Evaluate reporting relationships and requirements, including incident reporting.
 - Review Vendor access to the SCADA system via their system components.
 - Perform preliminary manual and passive discovery and reconnaissance data gathering procedures in order to better understand and plan specific assessment policies and procedures for client's SCADA and Process Control System environment.
 - Categorize and prioritize SCADA and Process Control System development and security initiatives.

PHASE 2: SCADA AND PROCESS CONTROL SYSTEM POLICIES, PROCESSES AND PROCEDURES

The ideal technical solution for any security environment includes protection layers using the latest and most comprehensive technology tools available. However, even the best technology can be circumvented by a simple personnel action. Phase two of the assessment consists of a thorough evaluation of both technical and personnel preparedness, based on the client's relevant policies, processes and procedures.

- Review all security training (general and SCADA/Process Control System-specific) provided by client. Evaluation will include a gap analysis of best practices versus current client practices.
- Review overall security awareness for the SCADA and Process Control Systems. A questionnaire will be prepared that will profile the current level of knowledge and sensitivity to SCADA and Process Control System security issues. Focus groups may also be required.
 - Perform a comprehensive review of all security policies (general and SCADA and Process Control System-specific) developed by client. This evaluation will include a gap analysis of best practices versus current client practices.

- Review business continuity planning process (including emergency operational configuration) to identify economic and operational consequences of short and long-term system disruption or security breach.

- Review emergency plans and planning process.
- Determine the type of testing, drills and response preparedness for security disruptions.
- Review who receives training for these types of disruptions.
- Review data backup systems and system redundancy.

PHASE 3: INTERNAL SCADA AND PROCESS CONTROL SYSTEM SECURITY ASSESSMENT

A crucial portion of ISS' assessment approach is a comprehensive evaluation of the system's design, functionality, architecture, data transmission, protective features, configuration management and current SCADA and Process Control System security features.

Analysis will be conducted for each of the following topical and functional areas, then on a comparative and overall basis to ensure that a full and accurate profile of the client's SCADA and Process Control System security is developed.

- Evaluate threat environment for the SCADA and Process Control System. Conduct a threat assessment as part of the scope of work. Customize results as applicable to SCADA systems.

- Review SCADA and Process Control Systems architecture and key design features. Evaluate all facilities and related systems and functions. A functional configuration of the SCADA and Process Control Systems will be developed to outline key characteristic security features and potential vulnerabilities.

- Conduct meeting with appropriate System Owners/Administrators to discuss detailed questions related to software features, functionality and security features.

- Discussions will be conducted with the technical and operational personnel responsible for the SCADA and Process Control Systems. To gain a comprehensive view, ISS will expect to meet with central staff and a representation of the staff at remote facilities.

- ISS will request that the client's assessment liaison contact component and subsystem vendors as necessary.

- Review SCADA and Process Control Systems system design and configuration management with specific consideration given to relevant related security issues.

- Verify the SCADA and Process Control Systems interface with other operational and monitoring systems.

Note: For the purposes of this evaluation, ISS will evaluate the most current configuration.

PHASE 4: EXTERNAL SCADA AND PROCESS CONTROL SYSTEM SECURITY DATA TRANSMISSION AND PROTOCOLS

For most corporations, the primary online vulnerability at their IT facility is their connection to the Internet. These connections are typically located within corporate buildings and are provided by commercial telecommunication providers that can be held responsible for the security of their media.

Utilities differ in this respect in that many own, lease and operate their own telecommunications systems in order to connect in distant plants. In addition, they use dial-in facilities to allow personnel and other "trusted" sources to access informa-

tion in the SCADA and Process Control Systems. Generally, utility telecom systems and communications facilities have not been designed with physical or information security in mind. However, with the high visibility of utilities as potential targets of attack, control over these telecom facilities has become extremely important.

In the past, the information traversing these systems was of no interest or use to anyone other than the recipient. Historically, the communication protocols (i.e. remote terminal units or RTU protocols) used between field locations and the utility control centers were not known beyond the small group of engineers who had developed and implemented them. However, with the availability of new technologies, these protocols can now be easily compromised or deciphered. This vulnerability has given rise to the possibility of both accidental and malicious acts that could compromise the service quality or security of the utility.

While new protocols are available that include password and encryption capabilities, these technologies affect the timely transmission of data. In addition, retrofitting RTUs with newer protocols can be costly.

Partial solutions, such as adding encryption modems at either end are also possible, but do not eradicate all security risks. The key must be an appropriate balance of the following security factors:

- The risk that a security threat will actually take place
 - The cost (in equipment, user training, on-going maintenance) of implementing security measures to minimize the risk
 - The amount that the risk can be reduced by the security measures implemented
 - The cost (in dollars, reputation, lawsuits) of recovering from the results of the security threat actually taking place
- ISS will perform the following tasks related to data transmission and protocols:
- Review client's information related to SCADA and Process Control System telecommunications and communication protocols
 - Discuss potential threats and risk factors associated with SCADA and Process Control Systems telecommunications
 - Describe alternatives and options for improving security
 - Recommend an approach for client to manage these risks

PHASE 5: SCADA AND PROCESS CONTROL SYSTEM ANALYSIS AND REPORTING

The SCADA and Process Control Systems review by the ISS' SCADA analysis teams will culminate in a report to management that presents a thorough assessment of the adequacy of the client's network and relevant systems connectivity environment. The review will also feature a section offering recommendations for improvement. ISS will submit printed and electronic copies as required.

REPORT CONTENTS & FORMAT

- An executive summary highlighting key findings and recommendations from the project in a management summary format
- An introduction stating the purpose of the report and a brief overview of the effort undertaken, including the scope, objectives and approach used in conducting the study
- Specific strategic recommendations to improve security

- Specific technical weaknesses uncovered by our review
- Analysis of threat data collected from the threat assessment
 - Detailed explanations of the security implications, impacts and risks for each of the identified exposures
 - The extent to which these exposures could impact other devices in the environment (for example, could the weakest link compromise the strongest?)
 - Results of Manual Audits performed on Firewalls, Proxy Servers, Databases and Web Servers
 - Specific tactical recommendations to address the identified exposures, prioritized from the most severe to the least severe exposure

PRESENTATION

ISS will prepare a PowerPoint presentation summarizing the results of the IT, SCADA and Process Control Systems security assessment performed by the analysis teams. This presentation will include the primary findings of the assessment, evaluation and analysis, including recommendations. This presentation will be delivered as requested to the client's management and key operational and technical personnel.

PREEMPTIVE PROTECTION

ISS' Intrusion Prevention Solutions include protection against vulnerabilities targeted specifically at SCADA systems and the underlying protocols supporting them.

- Stop attacks before they impact critical infrastructure and control system environments
- Preserve availability and prevent security breaches

SUMMARY

The complex architecture, interconnected nature and extreme sensitivity of SCADA and Process Control Systems mandates that utility organizations have a comprehensive plan for assessing and mitigating potential online vulnerabilities and threats. To do this successfully often requires the support of a security partner that not only has expertise in vulnerability assessment and planning, but that also has extensive experience working with SCADA and Process Control Systems.

BASICS OF PROGRAMMABLE LOGIC CONTROLLERS

Courtesy of the Motion Controllers Reference Center

A programmable logic controller, or PLC, is a software-based equivalent of a relay panel. A PLC is a general-purpose device. One PLC can be programmed to control a variety of machines, and programs can be changed easily for new jobs or changes in production routines.

Programmable logic controllers were once primitive devices capable of providing only minimal feedback about machine operation and status. The situation has changed drastically, however, with the advent of more powerful computer chips and new standards that give a PLC access to information throughout a manufacturing plant. Whereas the first PLCs generally provided only limited information about the status of relay contacts, new monitoring capabilities let the user know exactly what is happening on the floor.

Computers have expanded PLC power through greater speed and programming flexibility. Today's PLCs almost always have a port that permits a user to tie into a computer. Three developments have helped bring about this integration of PLCs and computers: "smart" PLCs with their own microprocessors and memory, multitasking software, and local-area networks (LANs).

New software has enhanced the capability of computers, particularly personal computers to operate with PLCs. Until recently, small computers were limited to performing one task at a time. If a computer was being used to check the status of a PLC, it could not perform data analysis or generate a report at the same time. However, with the development of concurrent DOS (disk operating system), the simultaneous juggling of two different tasks can be done.

More powerful microprocessors have resulted in PLCs able to perform multi-axis control and able to link with sophisticated vision systems. One reason new PLCs are taking on such a wide range of duties is that they can be installed in modules, thus simplifying any needed customization and future expansion. Because each PLC contains its own internal communication highway or bus, a PLC can be upgraded with additional memory or processing capability and can be added by snapping in additional modules.

Various PLC modules add RS-232 communication ports, multi-axis control and fault annunciation.

PLC software can be developed either on or off line; data-management and analysis programs are available. Many of the PLCs can be programmed from an IBM PC or compatible machine, and special industrially hardened programmers are also available when extreme temperature, dust, and vibration are problems. I/O stations can be located up to 2,000 ft from the CPU.

A PLC is no more powerful than the software available for it. Two relatively recent developments for PLCs are menu-

driven software and concurrent operating systems, which have simplified programming and made it more useful. Programs using menus allow an operator with only minimal training to monitor, analyze, and manage processes. Concurrent operating systems switch back and forth between two different programs so fast that both appear to be running at the same time.

Some PLCs are equipped to solve problems involving mathematical functions such as sine, cosine, tangent, xy , y root of x , e^x , natural logarithms, and common logarithms. Such calculations are often required for energy management, process control, process modeling, real-time error correction, and many other applications.

And while ladder logic is still the standard industry programming language for PLCs, the trend is toward state logic, sequential function charts, graphics, and versions that are programmable in Basic, C, or other high-level languages.

The ability to handle analog signals along with arithmetic and other complex calculations has made PLCs suitable for the control of processes as well as for the control of machines. Typical applications for PLCs are mineral and chemical processing, water and waste treatment, and petroleum collection and distribution. In many of these applications a PLC can complement conventional analog control systems by handling sequence problems as well as a portion of the analog calculation and control. In support of those functions, some PLCs now have the ability to store recipes for batch processing, reducing the need for manual inputs.

In further support of their process-control capabilities, some PLCs can be equipped to solve complex equations such as proportional-integral-derivative equations required for the control of many processes. A sophisticated PLC is capable of performing these calculations on many different portions of a process simultaneously.

PLCs are also capable of producing analog outputs and of providing position control functions. PLCs can even provide control functions normally performed by numerical controls.

A modern PLC can also pass information back to the operator. It can print out its own ladder diagram for record, review, or change, or it can provide status or progress reports in English on a CRT or printer routinely or on request. A PLC can also display messages in English to summarize data or guide the operator.

Data-analysis programs are becoming increasingly common. A spreadsheet format is often used. Usually each PLC is assigned a tag or number. Parameters such as data type, coil, input, and addresses are tracked. The PLC initiates changes to data within the computer database, which initiate other control tasks. For example, if the PLC closes a valve, a software routine

A programmable logic controller (PLC) is no more powerful than the software available for it.

can be started that measures resulting flow through the valve and sounds an alarm if the flow is not within desired limits.

The PLC also can track down external faults. This capability is useful because the machine and externally mounted control elements such as limit switches, solenoids, sensors, transducers, remote pushbuttons and selector switches are usually much less reliable, and more often a cause of machine downtime than the PLC.

Other maintenance aids are available to help solve malfunctions. One feature intensifies on a CRT that portion of a circuit that is carrying current. Another feature lets an operator command specific inputs or outputs to be unconditionally turned on or off, thus helping a technician determine whether a problem is being caused by an internal or external failure.

PLCS ROBOT CONTROL:

PLCs are often equipped with special firmware (software programmed by PLC manufacturers) through which the controllers can perform many complex procedures according to simple instructions in user programs. Sequencer firmware for PLCs provides the programming, storing, and accessing of data required for simulating electronic, or electromechanical sequencers and programmers. Sequencer data are stored in a data-table section of PLC memory, a section separate from that available for user programs. PLCs containing sequencer firmware are especially useful for controlling robots where the final position of each movement is determined by limit switches or other on-off, position-feedback devices.

Robot movements can be altered by using different masks for different robot tasks. One method of loading or programming sequencers is with a "teach" mode. This is a technique for loading or setting up the on-off contacts in a sequencer to correspond with the on-off status of I/O points. For this

method, a robot is jogged into a desired position. Pressing a "teach" pushbutton energizes a sequencer load instruction that causes the on-off status of all pertinent inputs to be copied into a sequencer step. By jogging the robot through steps sequentially, in each case pressing the "teach" pushbutton, a PLC is "taught" a sequence of movements.

Robots having closed-loop systems are controlled by PLCs through digital-to-analog (d/a) I/O modules. These modules convert digital PC signals to analog outputs having ± 10 -V range. The outputs serve as speed references for the hydraulic or electrical servosystems that typically power each axis. Each axis is mechanically coupled to a potentiometer or encoder that feeds position and velocity data back to a PLC, closing a feedback control loop. Digital commands from a PLC to a converter initiates motion, sets acceleration rates, determines speeds, and initiates deceleration. Motion is halted when the feedback indicates the robot has reached the proper position.

Suitably equipped PLCs can generally control point-to-point or vectored motion. Point-to-point movement along each axis is initiated and halted independently of other axes. This type of motion is easily programmed, requires little memory, and is suitable for many robot applications.

Vectored motion, however, requires that the movements along two or more axes be interdependent. The PLC adjusts acceleration rates and speed for each axis so that all movement terminates simultaneously. By this means a robot arm moves from point to point along the shortest path. Vectored motion control usually

reduces the time required for each move. To perform vectored control, PLCs must be equipped with arithmetic firmware that calculates speeds required for each axis. Although standardized algorithms are used for these calculations, vectored motion programs are somewhat more complex and require more memory than those for point-to-point motion.

Suitably equipped programmable logic controllers PLC can generally control point-to-point or vectored motion.

FIELD PROGRAMMABLE CONTROLLERS FOR COST SENSITIVE APPLICATIONS

By Richard Griffin, XILINX

INTRODUCTION

The introduction of embedded soft processors has offered substantial benefits to the world of digital electronics. The world's huge appetite for increasingly intelligent and sophisticated control systems has dictated the rapid advancement of processor technology. Perhaps most prevalent of all is the huge leap in demand of embedded microcontrollers. These processor-based devices can now be found in a colossal number of modern electronic devices. Microprocessors and microcontrollers are frequently confused; indeed the two terms are often incorrectly equated. For the purposes of this article, it is first necessary to clearly define the two terms and establish some important yet subtle differences.

MICROPROCESSORS

Microprocessors are single chip devices containing typically the core processor technology. This includes the execution units, the register file, program counter, memory interface, interrupt controller, and in some higher performance examples, the cache units and a larger peripheral set. In order to correctly function, the microprocessor requires a plethora of external components. Blocks of RAM and ROM are typical examples whether they exist to store executable code or provide a scratchpad memory area. Other examples include input / output (I/O) ports, timers, and serial communication ports. Figure 1 shows the typical layout of a microprocessor system.

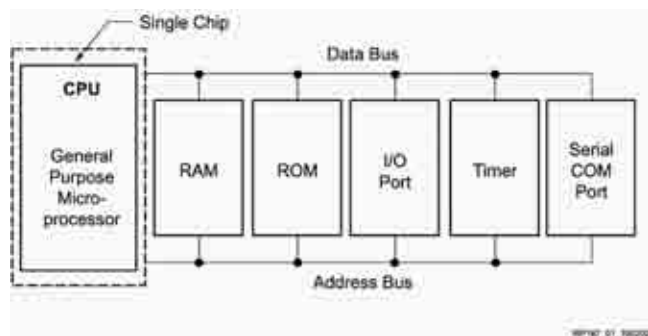


Figure 1: A Typical Microprocessor System

MICROCONTROLLERS

Microcontrollers, on the other hand, offer a considerably more integrated solution.

Rather than enforcing a building-block platform upon the user, a microcontroller system is comprised of blocks within the boundaries of the device package. The CPU, RAM, ROM, I/Os, and peripherals are all contained within the device, closely integrated using localized internal connections. The inputs and out-

puts of the microcontroller system are coupled to the other hardware blocks within the design via the pins on the microcontroller device. Microcontrollers present the designer with a solution that is far easier and faster to use. Figure 2 shows a microcontroller implementation. A single device prevents the user from making any mistakes connecting the CPU to the memory and other peripherals, while affording them rapid development time and ease of implementation. In essence, a microcontroller provides a solution to which only I/O signals need be connected and executable code be written.

The flexibility of the microcontroller is reduced when compared to the use of a full microprocessor, but this is quite often an acceptable loss for the application in question. The primary issue of importance is one of cost per unit, which often sways the decision in favor of the microcontroller solution.

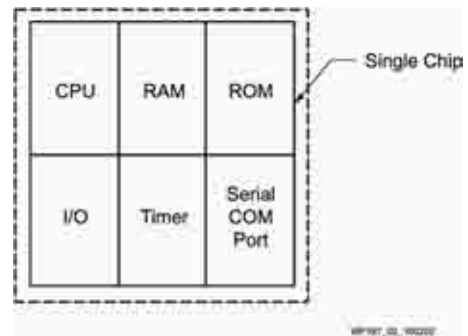


Figure 2: Microcontroller

End Markets Microcontrollers are used extensively in three major market areas: automotive, industrial, and consumer products. The modern automobile is replete with microcontroller-based systems providing automated control for just about every conceivable part of the car. Braking systems use microcontrollers to deliver advanced safety features like ABS and traction control to the driver. Windshield wipers are controlled to bring us timed interval wipes and even rain sensitive automatic wiper activation. Heating controls for the vehicle interior are frequently found monitoring multiple zones within the passenger compartment, adjusting the supply of air to the interior to maintain a user chosen ideal temperature. Seats even adjust electrically to remember the favored positions for different drivers of the car! All of this technological wizardry is achieved and implemented under the watchful eyes of a collection of microcontrollers.

The industrial world has been a changing place in recent years. Gone are the days of a large workforce all trained to monitor a specific area of a production plant. They have now been

replaced by a series of microcontroller based monitoring modules, often linked to a central station, which is monitored by a supervisor. The reduction in cost in microcontroller technology has permitted these modules to replace their human counterparts who were ultimately prone to lapses of concentration and the potential dangers of human error. Microcontrollers can work tirelessly around the clock without lapses in concentration, requiring only the most minimal maintenance and supervision.

The consumer market is also the home of a heavy concentration of microcontrollers.

Walk into any electronic store and you will be hard pushed to find a product that does not have some kind of microcontroller lurking beneath its covers: video recorders, televisions, dishwashers, video games, set-top boxes, refrigerators, washing machines, remotely controlled lighting dimmers, telephones, answering machines, ovens, toasters, printers, scanners, even children's toys are microcontroller equipped. So huge is the demand that microcontrollers are now outselling the more conventional microprocessor by around six units to one as shown in Figure 3. Microcontrollers are here to stay, it is therefore important that they are able to meet our ever increasing demands for the future. The only way to keep up with the demands on lower costs is to incorporate the microcontroller into existing hardware devices to form system-on-a-chip platforms.

For this to succeed, a flexible platform is needed into which the microcontroller can be embedded.

THE FPC SOLUTION

In this example, the Spartan-IIE family of FPGAs has a regular, flexible, programmable architecture of Configurable

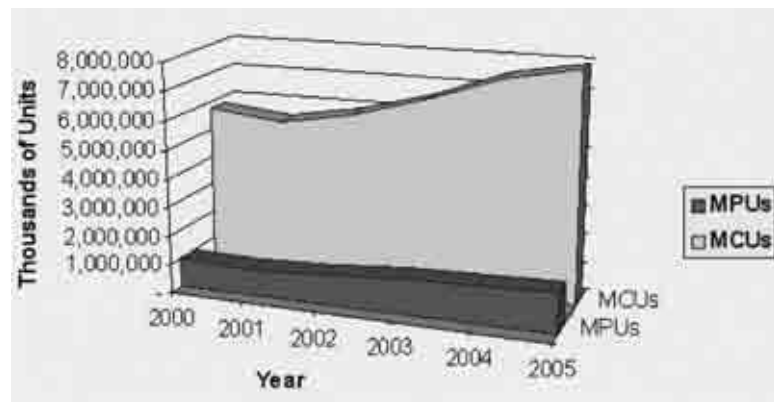


Figure 3: Worldwide Microcomponent Unit Shipment Forecast
Source: Gartner Dataquest (January 2002)

Logic Blocks (CLBs), surrounded by a perimeter of programmable Input/Output Blocks (IOBs). Two columns of block RAM lie on opposite sides of the die between the CLBs and the IOB columns. A hierarchy of versatile routing channels interconnects these functional elements. Spartan-IIE FPGAs are customized by loading configuration data into internal static memory cells while permitting unlimited reprogramming cycles to become a viable upgrade path for future enhancements. Figure 4 shows a block diagram of the Spartan-IIE FPGA device.

This flexible platform is a base upon which to implement a microcontroller system. We previously looked at the way in which microprocessor systems and their peripherals and memory were all integrated onto one device, providing huge cost savings to many modern electronics products. An embedded micro-

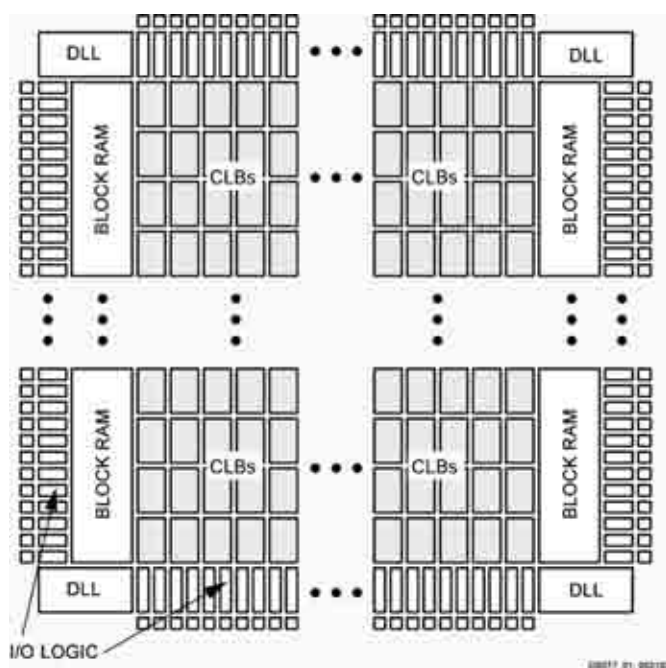


Figure 4: Spartan-IIE FPGA Block Diagram

controller takes the concept of integration one stage further by permitting the designer to embed the microcontroller system into a small section of a programmable device. No longer does the microcontroller have to exist in a stand-alone package; it can now be embedded deep within custom hardware.

The introduction of customized soft processor systems for Field Programmable Gate Arrays (FPGAs) has offered huge flexibility, but with this flexibility comes some new challenges for the designer. Traditionally the designer has approached the task of processor selection by comparing the needs of their system specification to the features listed on the processor datasheet. While this may sometimes be a trivial task, there are times when unusual processor configurations are desired by the system specification.

For example, the designer may desire a processor with 10 UARTs, an interrupt controller, and access to a block of external FLASH. Although many off-the-shelf processors offer multiple UARTs and the other desired peripherals, they would typically be of sufficient complexity to have numerous other peripherals that would be unused in this system. Not only is the designer paying for the additional peripherals, it is often necessary that unused peripherals in this type of processor have to be placed into a safe mode or otherwise disabled via software.

An additional burden now exists on the software design team. Not only do they have to make the used processor peripherals operate correctly; they also have to write code for the parts of the processor which are not being used. It is clear that purchasing an off the shelf solution for this scenario would be highly wasteful not only in terms of initial cost, but also in wasted engineering time during the design process.

Customized processors, from their very name, requires that someone performs the customization! This is where design automation tools and intellectual property play a key role (Figure 5). The processor core is placed at the heart of the sys-



Figure 5: Xilinx Platform Studio (XPS) Tool



Figure 6: System Settings Window

tem, and the required peripherals are then added to the system from a catalog of Intellectual Property (IP) cores. As each block of IP is added, the customization process deepens by allowing the user to select the behavior and functionality of each peripheral. UARTs can be configured to operate at the correct baud rate, communicate using the desired number of data and stop bits, and employ the required parity checking.

External memory controllers can be customized to insert sufficient wait states for correct and efficient memory device access; multiple (independently configurable) banks of memory are supported from a single controller giving the designer access to SRAM, FLASH, and EPROM memory. Interrupt controllers can be configured to respond to rising or falling edge inputs, or indeed to adopt a level triggered response.

Bus structures are added to connect the entire system, which are again configurable to meet the needs of system clock speed or silicon area.

ADDING SYSTEM LEVEL PARAMETERS

Once the basic structure of the processor system is in place, the designer can enter phase 2 of the design process and begin to allocate system level parameters to the processor design (Figure 6). These include the desired address map for the processor system, the selection of interrupt priorities, the allocation of the standard input and standard output devices from the included peripheral set, and so on.

The tools can now be used to configure the memory in the system to contain the executable software code taken from the supplied C compiler. Memory values are allocated to the internal block RAM memory, located within the core of the FPGA. External memory can also be configured using the various utilities supplied with the Embedded Development Kit (EDK).

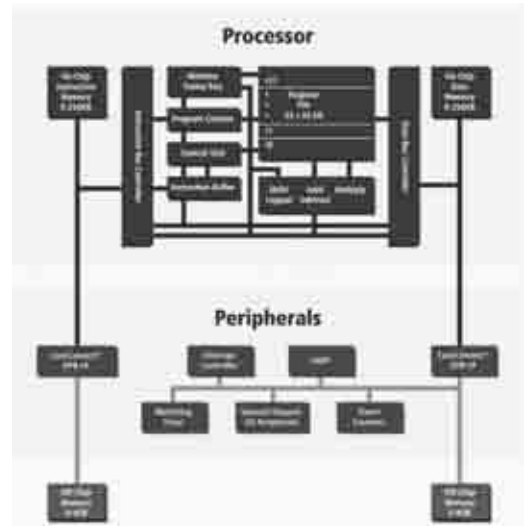


Figure 7: MicroBlaze Block Diagram

The finished processor system, although complete, remains totally flexible. Any change in the system specification can be quickly reflected by the designer using the automated toolset, allowing maximum flexibility with zero wastage.

The MicroBlaze soft processor used in this example is based upon the successful RISC/Harvard architecture combination (Figure 7).

PLCS PACK IT IN

By Gricha Raether, Product Manager, National Instruments Corp.

Programmable logic controllers, or PLCs, are ideal for discrete logic control. However, as PLC process control and automation complexity expands, shoehorning new requirements into these ubiquitous devices can prove difficult. There is a significant and still-growing need for devices with expanded processing power, advanced I/O interfaces, and more control capabilities. Craig Resnick of the ARC Advisory Group coined an appropriate name for these devices. He calls them programmable automation controllers, or PACs. PACs represent the next evolution in automation controllers made possible by exponential advances in computer technology. Thus, companies like General Electric, Rockwell Automation, Siemens, and National Instruments can pack more processing capability into small and highly rugged devices that require less power.

Differences between traditional PLCs and the newer PACs become apparent when comparing software-execution models. PLCs generally follow a linear execution of their programming code. The PLC reads values from input modules and updates its internal input registers. These input values are then used to create or modify other register values according to specific rules of logic dictated by the PLC program. Then the PLC updates its output modules with the newly calculated values and the process starts again.

All PLCs follow this input-process-output model in what is referred to as a scan cycle — an ideal model for discrete logic. In fact, IEC 61131-3 relates how programmable-control systems use ladder logic to reflect this model. Complex control challenges, such as performing several actions in parallel, require more-complex data-acquisition and processing architectures with greater processing power. PACs use advanced hardware architectures and high-level programming tools to concurrently process multiple simultaneous tasks.

Today's control processes rely on a myriad of signals and data, ranging from simple analog and digital I/O to high-resolution image recognition and multi-axis motion-control commands. Applications involving high-speed and high-precision production, real-time machine-condition monitoring, and complex process control require deterministic execution of advanced analysis and processing algorithms alongside high-speed data-acquisition systems. High-end PLCs do exist with enough processing power to satisfy some of these requirements. However, many lack resources like floating-point processors and sufficient memory capacity necessary to handle these signals efficiently. Yet this technology is readily available in commercial off-the-shelf hardware developed for the PC industry.

PACs integrate commercial hardware with real-time operating systems to provide a cost-efficient platform for high-performance automation.

PACs are not meant to replace PLCs but, rather, to complement them in existing applications. For example, PACs can easily optimize a local or specific part of a production line or process. PAC controllers integrate easily with existing systems thanks to their open architecture. For instance, an engineer may add a PAC to a production line for realtime thermal analysis. It updates the existing control system with the results of that analysis through standard Ethernet or shared registers in commercial gateways. PACs can perform any analysis, such as vibration monitoring, on equipment sending update signals to the overall system. If the PAC detects excessive or improper vibrations, it shuts the machine down before major damage occurs — possibly avoiding severe impact on plant capacity.

IEC 61131-3 presents a predetermined framework to create discrete logic control applications with ease. The standard removes worry about programming execution details making the only concern what goes in and what comes out. However, optimizing a process locally or performing more advanced control functions can be beyond the scope of IEC 61131-3. Programming such applications requires a more-thorough understanding of the subsystem needing optimization, expertise in programming languages that allows parallel task processing, and knowledge in the manipulation of advanced I/O and processing algorithms.

The rigidly defined structure of standard ladder-logic programming is not suited for the advanced nature of these applications. Fortunately, languages such as C/C++ and National Instruments LabView do provide the flexibility required because their execution code defines every detail — an advantage for more-advanced tasks but a challenge for the traditional PLC programmer. Not only must programmers configure and program the I/O, analysis, and control algorithms, they must also define the entire program structure.

This added complexity may overwhelm the simple programmer. Some advanced languages, such as NI LabView, help programmers by using identical development structures no matter what the task. With LabView, programmers use the same graphical interface for machine vision, advanced I/O, motion control, HMI/Scada, and even leading-edge technologies such as fieldprogrammable gate arrays.

Most PLCs today feature fixed-point processors because they are inexpensive and provide the computational power that discrete logic demands. The processors can even handle simple processing tasks like PID control. However, analyzing hundreds

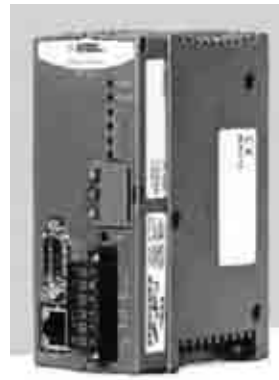


Programmable automation controllers, like these from National Instruments Corp., complement and enhance PLC operations by performing special tasks above and beyond an average PLC's capabilities, such as machine vision and highspeed process data acquisition and analysis.

of video frames per second, calculating fast-Fourier transforms, performing order analysis routines, or solving advanced control algorithms such as model-free adaptive control, require significantly higher processing levels. PACs routinely include Pentium-class floating-point processors for these intensive calculations.

PACs come in several models, depending on the application. Some dedicate all their processing power to only one type of application, such as machine vision. Others have high-speed processors powerful enough to combine multiple tasks like motion, vision, and high-speed measurements. The combined power updates hundreds of PID loops in microseconds or creates millimeter-precision trajectories for multi-axis motion systems. Intensive computing power like this also requires high-speed inputs and outputs that keep up with program execution.

PACs evolved from the need for easy-to-use devices with processing power comparable to PCs and the reliability and ruggedness of PLCs. Most PACs sport industrial-grade specifications such as expanded temperature ranges from 40 to 70°C, 50-g shock, and 5-g vibration. Real-time operating systems give deterministic performance and operate for extended periods without crashing — a recurring problem with general-purpose operating systems like Windows.



This cFP-2120 Compact Fieldpoint processor from NI is an example of the new PACs developed for embedded control or distributed I/O applications. Typical uses for these controllers include measurement taking with data logging, pid loop monitoring and control, realtime situation analysis, and simple I/O operations such as valve actuation and motor control. Ethernet connectivity with a built-in Web server transmits information gathered by the system to any OPC client or HMI/Scada display.

Vendors deal with long-term device availability and the effects of commercial off-the-shelf part obsolescence issues by designing PACs using interchangeable parts readily available from different sources. Backwards-compatible design of new versions preserve existing functionality with drop-in replacements, and typically expand processing power and enhance features. Vendors usually know about part obsolescence with sufficient lead time to design replacements with no impact on product availability.

LOGIC CONTROL FOR ROBOT WORK CELLS

By Bennett Brumson

Work cell control is an area of robotics that is changing rapidly due to the fast pace of change in technology. This is especially true of the evolution of controllers and their ability to monitor and manage elements within the work cell.

There are three major types of control units available for robotic work cells. These include programmable logic controllers (PLCs), industrial personal computers (IPCs) and the straightforward robotic controller. Each has its strengths and weaknesses although advances in technology are blurring the distinctions among them.

THE OLD STANDBY

PLCs are a good choice for low cost applications which require digital IO. More expensive PLCs can also be used for analog IO, RS 232, reading encoders, and other more complex IO schemes. One advantage of PLCs is programming. Due to standards such as IEC 1131-3, it is easy to change from one PLC product to another as the languages are very similar.

"PLCs are more cost-effective, less expensive and simpler than IPCs, but are less versatile," professes Michael Aulik. Aulik is flexible automation manager at Q Comp Technologies, Inc. of Greenville, WI.

Around since the 1970s, PLCs consist of a CPU, memory, as well as circuits to receive input and output data. PLCs analyze inputs and determine the phase of outputs and inputs. Using this information, the PLC will turn on or off appropriate outputs. The task that an operator needs performed is entered by use of a software interface. The late '70s and early '80s is when there was an evolution in PLC communication, giving them the ability to talk to other PLCs and to robots. PLCs contain relays, counters, timers and data storage areas.

"Festo makes PLCs that don't need a coordinated motion control, just discrete I/O. Our PLCs are optimized and are designed to operate continuously," said Frank Latino, a product manager at Festo Corporation of Hauppauge, NY. "There is less 'overhead' in the operating system of PLCs, so all its elements are used. IPCs are designed to run off Windows, which has a lot of overhead. PLCs have less superfluous parts to perform its I/O control functions. Right now, there is still a price advantage to PLCs. PLCs generally afford a better cost-effective solution than do other types of controllers," Latino continued. Festo is a maker of electronic interfaces with robotic controllers that connect to communication devices such as DeviceNet, TCP/IP, and Ethernet.

Latino went on to explain that a small PLC having eight inputs and four outputs would cost as little as \$150. "It depends on what you need. Unless there is a need for more complicated functionality, I think PLCs are a better solution. This is especially true if there is a low count of I/O," he said.

"A PLC circulates and looks for signals. A signal activates another signal. A robot will not activate or deactivate a signal until it gets to that point in the program," said Aulik.

NEW KID ON THE BLOCK

Industrial PCs are a good choice for more complex applications.

"The shortcoming of IPCs is they cannot circulate inputs and outputs constantly as do PLCs. IPCs have software timing issues and they don't circulate looking for signals through all inputs and outputs continuously like a PLC does. PLCs tend to work well with timing between it and the robot," said Aulik. "Both PLCs and IPCs can handle most applications, but the latter generally has a cost advantage. PLCs handle the timing issue a little bit better than IPCs but can not handle everything. On the downside, PLCs tend to have a monochrome graphic interface."

While it is true that many stand alone industrial computers run Windows-based operating systems, many industrial PCs are SBCs, or single board computers. These are typically embedded in equipment and usually run an industrial real-time OS. These are much better at handling IO, and the operating systems are much more compact and stable. They do cost more than PLCs and are not as easy to program.

"IPCs are better than PLCs if there is a very high amount of I/O. Also, IPCs have a better operator interface. This is where IPCs have the advantage over PLCs," asserted Festo's Frank Latino. "There is certainly enough horsepower in an IPC to perform its functions in a typical I/O control. If complicated features or high I/O counts are needed then an IPC is justified, if price is less of a

factor," Latino added.

Michael Aulik concurred with Latino on IPCs interface.

"Because of their flexibility, IPCs can be programmed to be more user-friendly and have a more sophisticated interface," Aulik said. He added that in the near future, there will be more use of IPCs due to their increasing power but they will not supplant PLCs.



A PLC from Festo



Servo Robot's Flexcell controller with screen showing green status

"Robot users will tend to drift towards IPCs because of their flexibility and because they are more user-friendly. Robotic users can add a little bit of flair to what they are doing with IPCs. To distinguish themselves, controller makers might start to produce more IPCs," Aulik said. "For the time being, PLCs are too easy to use and too cost-effective to be wholly replaced by IPCs. Unless there is a major technological breakthrough that changes the cost dramatically, or new software that makes it very easy to program IPCs with little effort, I don't see a major change in the near future," Aulik predicted.

Jeff Noruk, president of Servo-Robot Corp, amplifies on the pros and cons of IPCs and PLCs. "The increased speed of IPCs helps them manage a work cell, especially for part inspection applications. IPCs will get faster, collect more information, do more analysis, and provide more complete summaries of data and other useful information," he said. Servo-Robot Corp. is a U.S. subsidiary of Servo-Robot, Inc. of Saint Bruno, Quebec, Canada.

"In the short term, there will be more compatibility with other companies' equipment. Information will become easier to read, by use of simple green, yellow, and red status lights. There can be an IPC communicating with a master IPC, keeping track of data throughout a whole factory," Noruk foresees this in the next three to five years.

The idea of a more completely integrated control system has champions in many sectors.

"If there are specific information needs, such as process monitoring, data-base applications, or a lot of information flow going through other computers, IPCs are best at networking them," said Gary Zywiol. Zywiol is vice president of product development at FANUC Robotics North America, Inc., of Rochester Hills, MI.

IPCs are a major player in electronics assembly applications because of their precision.

"Cimetrix software is for assembly applications that have complex motion requiring vision and accuracy," said Steven Sorensen, chief technology officer at Cimetrix. "We haven't done much in heavy industry, where six-axis articulated robots are used for welding. More traditional heavy industries tend to use PLCs rather than IPCs as controllers. The electronics industry is very 'PC-centric', reflected Sorensen. Cimetrix of Salt Lake City, UT, is a manufacturer of software for PC-based controllers and communication software for the electronics industry.

CONTROL MASTER

Robotic controllers are the master component; used by robot users to successfully exercise direction over a work cell, and offer more than just robot control functionality.

Zywiol explained that controllers can do a lot of things besides running the robot itself, including capabilities beyond motion control. For instance, controllers can do all of the tooling I/O, manage fixtures and act as a communication device. He went on to explain that a controller can have links to networks in a manufacturing plant that gives it a great ability to carry

information when linked to PCs. The robotic controller itself generally does everything done in a cell, he maintains.

Robot controllers are starting to use industrial PCs. Many of the issues such as safety and IO scan times are not issues with the PC hardware, which is very reliable and problem free, but with the software. By switching traditional operating systems such as Windows to faster, more robust real time operating systems, these problems are solved. The next step will be for robot programming languages to adopt standards such as IEC 1131-3, just as PLCs have.

Zywiol remarked that in some instances, "There is PLC functionality within a robotic controller. Or, the robot controller is taking on the tasks done by PLCs in the past. A company that does things well will take advantage of the strengths of the PLC, IPC, and robot controller platforms. For example, in the PC world, the strength is information flow, its data base applications, the ability to communicate between computers, and other intelligent devices. We frequently use PCs when there is a need to move large amounts of information."

Zywiol also mentioned the downside of IPCs.

"The disadvantages of IPCs are their reliability and predictability. IPCs are still not as reliable as a dedicated motion control system in those areas. There is guaranteed performance with a robotic controller or a PLC," said Zywiol. "When you get into motion control, there are issues like safety. You must guarantee the safety of the customer. The robot's load can be up to 400 kg and moving at high speeds. By using robotic controllers for motion control, you get very high performance and guaranteed safety," he said.

As controllers and other robotic electronics get more sophisticated, there is a potential difficulty in servicing these components. Industrial Control Repair (ICR), a Warren, MI-based firm specializing in repair and replacement parts of robotic cell equipment, including controllers, shares its perspective.

"Controllers are similar to cars.

As they get more complex, there is less ability to do self-service, so there is a need for specialized technicians. Controllers are more difficult to troubleshoot and to dismantle to gain access to internal parts where the problem may be," observed Glenn Dantes. Dantes is vice president of ICR.

"Adding options on a robot intimidates some users. Those new to robotics want a controller that is older, simpler, easier to operate and maintain," added Dennis Edney. Edney is

Dantes' colleague at ICR and is the firm's robotic specialist. "Controllers mirror computers in that they are getting faster, have more memory, and are getting smaller. They take up one-third less space than they did in the 1980s. Newer controllers have circuit boards that make them simpler to use," he said.

It is somewhat ironic that the move toward smaller size in controllers, an attribute coveted by customers, has presented problems for repair and maintenance procedures. But some



Cimetrix's Core Motion pendant interface



Fanuc's RJ3i B Controller with i-Pendant

experts foresee an Internet-based solution.

“Controllers are more compact so they are more difficult to access. Many controllers have more features and can do more complex tasks. They are more software-intensive than ever before. There will be an increase in the trend of robot makers looking at the manufacturing process of their robotic customers over the Internet to do remote diagnostics,” said Edney.

Dantes agreed with his associate’s analysis. “New systems are more costly and complex. There could be a trend towards older, simpler systems. New isn’t necessarily better, especially for first time users of robotics.”

CONTROLLING THE FUTURE

As technology advances making robotic control systems more complicated, there will be less of a do-it-yourself aspect to their maintenance and repair. Although its day is far from over, there will be an inevitable replacement of the ubiquitous PLC with the ever-increasing power of IPCs and robotic controllers.

RECOMMENDED READING & RESOURCES

Robotics User Discovery Day... features insights from leading robotics users. They’ll share case studies on the strategic and financial benefits of robotics, as well as lessons learned along the way to successful installations. If you’re a robotics user, this will be an invaluable event. If you’re a supplier, you’ll gain critical information on what your prospective customers need from you in order to succeed.

STATE LANGUAGE FOR MACHINE CONTROL

By Kenneth C. Crater, Chairman, Control Technology Corporation

INTRODUCTION

State languages have recently come to the forefront of discussions on superior approaches to automation programming. The temptation is to view state language as a “new” automation tool, since it has only recently received attention in our industry’s trade journals. In fact, the use of state languages in control dates back at least to the late 1970s, when the early implementations of Quickstep[1][2] were developed to mimic the function of electromechanical cam programmers. The theoretical basis of state language control is older still, having its foundation in Petri Net theory as developed and described by Carl Adam Petri[3] his PhD thesis in the early 1960s.

There is still much confusion as to what constitutes a state language, how best to use one for automation applications, and what economies exist in their adoption. I hope to clarify some of these questions in this paper. Further, I will show that the increased use of state language technology in automation is a direct response to some very specific new demands being placed on manufacturing organizations. Far from being “just the next new language”, state languages came into existence to address problems which could no longer be adequately addressed using pre-existing programming techniques.

WHY STATE LANGUAGE? WHY NOW?

The emergence of state language as a preferred programming paradigm is not occurring in isolation. Rather, it reflects sweeping changes taking place in the practice of industrial automation. Referring to Figure 1, each stage in the development of automation technology was both a reflection and a determinant of the underlying technologies and techniques used by practitioners.

In the early days of hard automation, stand-alone fixed-function systems were the predominant practice. Information was transported via clipboard (the physical kind), actuation was largely mechanical (transitioning to pneumatic/hydraulic), and

programming and user controls were hardwired. These practices were self-consistent, and were also consistent with the primary goal of automation at that time: to reduce labor input and decrease unit cost of production.

The quickening pace of global markets, commencing in the 1970s and continuing through the 1980s, drove the need for flexible automation. Among the new technologies employed to meet this challenge were PLCs, increasingly integrated with minicomputers and later with personal computers, organized in workcells within which configuration and production information was shared. It was in this environment that the innovation of relay ladder logic thrived, allowing a modest degree of complexity in automation strategies and providing the decision-making tools for flexible machines. This language also allowed a further degree of flexibility through reprogramming, reducing the cost of responding to changing market needs.

Again, this was an internally-consistent toolset, adopted in response to environmental conditions. Flexible tooling allowed the increasing cost of automation, and floorspace, to be shared among several product variants, and improved the ability of production management to respond to quixotic markets.

Now, automation techniques are being driven by information requirements, coupled with an unprecedented need for agility. The factors in this environment which directly affect the selection of automation technologies include:

(1) Dramatic market and technological change - Agility has been called that characteristic which allows an organization to thrive in an environment of constant and unpredictable change[4]. In the face of such change, companies are adopting strategies which provide the greatest ease of retooling and the greatest extent of flexibility without retooling. Among these strategies are the use of more highly integrated systems and corresponding languages that reduce the costs of interfacing motion control, data acquisition, communications and other requisite technologies. This paper will show how state language can accommodate this higher level of integration and thus

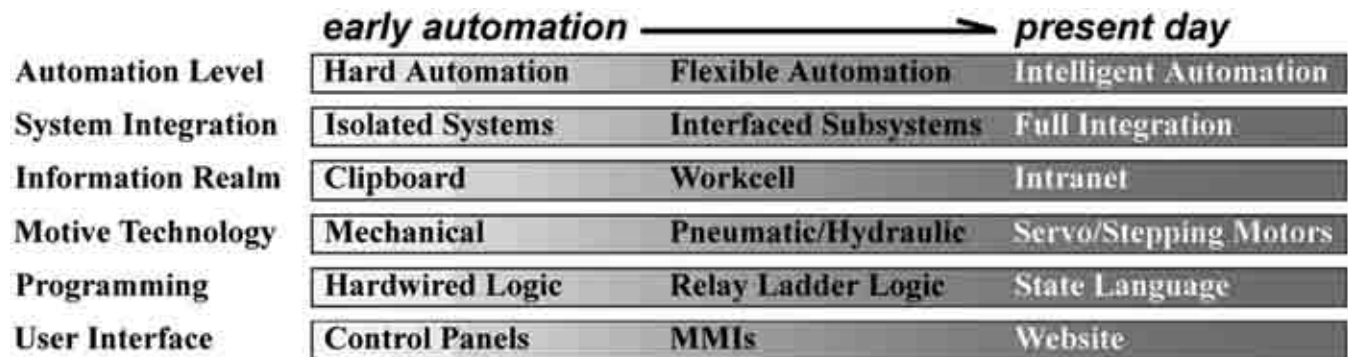


Figure 1

decrease development time substantially.

(2) Broadened expectations of quality - The definition of quality has changed, and it has become a non-negotiable expectation in many markets. Two critical technologies for implementing quality programs are analog data acquisition, for the purpose of making qualitative measurements, and communications for presenting data to remote monitoring, storage and retrieval systems. Previous control techniques were ill-equipped to accommodate these requirements, and often required supplementing with secondary systems. State language, however, provides a mechanism for integrating these requirements with the control program in a concise and meaningful manner.

(3) The advent of the information age - Every major manufacturing company has in place an extensive information network, and either has or will extend that network to the plant floor. The same expectations of access, visibility, and control which have come about in the office environment are now making their way to the plant floor. Older, boolean-based programming methods for automation do not present an information-rich resource, however, and fail to meet these expectations.

(4) The shifting role of the manufacturing worker - Initial automation efforts were focused on extending the physical capabilities of human workers exerting more power, extending their precision, allowing work in intolerable environments. Then, the focus moved to cost savings, increasing output per labor hour expended by replacing human effort with machine effort. In the process, however, the intellectual contribution of the worker was lost - the inflexible machine was uninterested in the human operator's opinion of the quality of output, therefore the worker lost the ability to make a difference. The inevitable result was a demotivation and de-skilling of the workforce.

Now, however, many companies are recognizing the necessity of recapturing the value added by the human worker's intellect. Ironically, this sometimes points to a conscious limiting of the extent of automation, providing some measure of additional qualitative control which may be exerted by the operator[5]. Maintaining reasonably high levels of automation, while simultaneously allowing such qualitative control requires automation systems with additional capabilities. Mathematical functions, extensive parameter storage and communication, enhanced operator interface facilities, sensing/reporting capabilities and integrated motion control for making mechanical adjustments are all enabling technologies to meet this requirement. The simple control languages of the last decade fail in many respects to fulfill these needs, although it will be shown that the state language framework can do so.

THE FRAMEWORK OF STATE LANGUAGES

The appeal of the state language framework resides in its simplicity and its fit with the problems being addressed. In the discrete manufacturing world and, to some extent, in the batch and continuous process worlds, these problems often consist of a series of steps or states which a machine must go through to perform a series of operations on a workpiece or product batch. Ignoring, for a moment, the more complex case of asynchronous operations, a state language can exactly mimic the structure of this type of problem.

The fundamental tool of a state language is the state, or "step". A step defines the complete status of the machine or process (or a portion thereof) for a finite period of time. Typically, this status consists of two components:

- a) One or more commands to create a motion or change,



Figure 2

thus causing a new physical state to be adopted by the machine or process.

- b) One or more instructions to limit the duration of the step and specify the next step to proceed to upon completion of the current step.

Note that either of these components may be arbitrarily complex. The motion command may be as simple as a single digital output change, perhaps to actuate a pneumatic solenoid valve. However, it may instead be as sophisticated as multiple commands to initialize and turn a series of servo motors, specifying motion and tuning parameters as well as terminal positions for each.

The instructions for terminating a step may be as simple as a time delay instruction, or an instruction to monitor a single digital input - perhaps wired to a limit switch. However, they may get as complex as instructions to read multiple analog input values, representing temperatures, positions, and other variables, combine them mathematically, and decide upon a new direction for the program to take depending on the outcome.



Figure 3

The flexibility provided by this framework is quite powerful. Because the framework allows very high-level commands to be implemented, a state language can "keep current" with evolving actuation and sensing technologies. But the real advantage may best be seen by examining a state language program in overview.

The structure of the resulting program bears a striking resemblance to a flowchart of the desired operation of the machine. This one-to-one correspondence is responsible for much of the savings in development, debug, and maintenance time attributable to state language use. It provides a further benefit by demystifying the operation of a control program, allowing other members of a design team to understand, review, and comment upon the program's structure. The program listing may therefore become a common point of communication among the controls engineer, machine designer, process engineer or industrial engineer, information systems engineering, maintenance personnel, and even

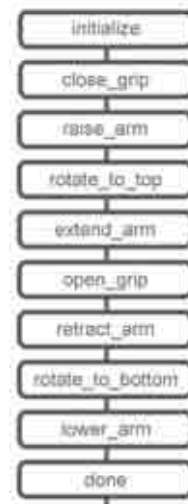


Figure 4

the machine’s operator. This creates an environment where each team member is able to maximize his/her contribution to the automation effort.

A COMPARISON TO RELAY LOGIC

By far the most prevalent language in the control of automated machinery today is relay ladder logic (RLL). The purpose of RLL was to provide a language to emulate the functionality of the electromechanical relays which had previously been used to build logic systems for control. Before engaging in a comparison, it must be noted that the adoption of RLL was an essential step in achieving the acceptance of electronic controls in the factory environment. The use of this language allowed the plant electricians who maintained the previous electromechanical controls to become familiar with a new generation of electronic controls. At the same time, RLL facilitated the migration of old control schemes from electromechanical methods to electronic methods.

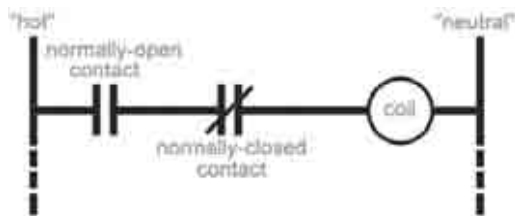


Figure 5

The essential elements of RLL are the coil, analogous to the electromagnetic coil of a mechanical relay, and the contact, typically associated via labeling with one of the coils which “actuates” the contact. These elements may be seen in Figure 5. Contacts may be of one of two types: normally-open or normally-closed. Normally-open contacts are said not to pass any “current” when the associated coil is inactive. Conversely, normally-closed contacts pass current only when the associated coil is inactive. In either case, the current, of course, is imaginary and is used only to represent the behavior of the program.

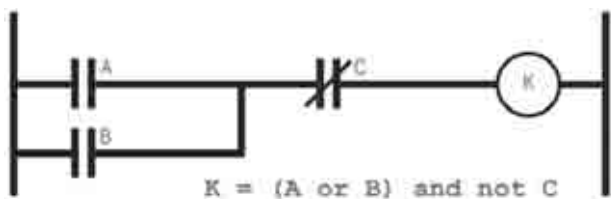


Figure 6

In actual practice, the contacts of an RLL program are used in various combinations in such a way that they model a Boolean equation. An example of such a combination, along with the equivalent equation, is shown in Figure 6. When contacts are paralleled, their operation is the equivalent of a Boolean OR function. When contacts are in series, their equivalent function is the Boolean AND.

The runtime execution of such a diagram consists of a cyclical scanning of the entire diagram, examining the current state of all contacts and then determining which coils should be active at that moment. The contacts associated with the active coils will, on the next scan, assume their active state which may, in turn, affect the status of other coils, etc.

In practical machine control applications, the use of RLL brings with it substantial additional overhead. Most automated machines have a natural sequence of events by which they convert a raw material or unfinished workpiece into a finished product. This sequence typically consists of a series of mechanical states the machine must assume, driven by the control system. To program such a series of states using RLL, it is necessary to use a number of latches, with each latch being built from a number of RLL elements.

Figure 7 illustrates this technique. An external event, sensed by LIMIT_SWITCH, provides energization to the coil M1. The contacts from M1 then close, bypassing LIMIT_SWITCH and providing continuous “current” to the coil for M1. This coil then remains latched, even if LIMIT_SWITCH de-energizes. Additional contacts from M1 may then be used elsewhere in the program to enable those events which should only occur subsequent to this portion of the machine’s sequence. This is illustrated in the second rung of Figure 7, where a contact from M1 in conjunction with a second external input, LIMIT_SW2, may then engage another coil, labeled M2. In this case, a normally-closed contact from M2 then opens and de-energizes the coil M1, signaling a progression from one state to another.

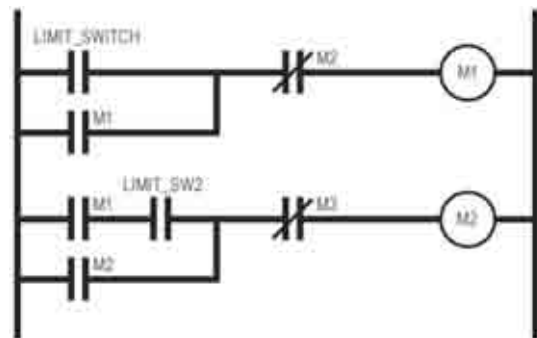


Figure 7

The necessity to build “statefulness” into a program through the programming of latches may not be a significant inconvenience in a small program, but as programs grow in complexity, the increased burden added by this requirement becomes substantial and cumbersome.

The proliferation of many non-Boolean technologies in automation has also greatly diminished the efficiency of RLL as a programming language. Servo and stepping motor control, analog data acquisition and control, and intensive data gathering and communications were not predicted by RLL, nor are they adequately accommodated. Such control requirements are usually addressed in RLL by injecting the new element of a function block which occupies the place of a coil in the relay diagram. The function block is triggered by one or more contacts, and then executes whatever instructions are contained within it. Often, these are expressed as a series of codes to be sent to a dissimilar motion controller, communications port, etc.

The necessity of supplementing RLL with instruction elements of a completely different paradigm, coupled with the lack of a pre-existing structure for the most common requirement of machine control, has now become a substantial obstacle to contemporary automation efforts. In large part, the increasing acceptance of state language technology is a response to that fact.

STATE LANGUAGE CONTRASTED WITH SFC

The field of programming languages for automation has seen much activity in recent years, again as a reflection of changing needs. One of the more visible efforts has been the move to standardize a group of older languages, mostly Boolean or procedural languages, under the nomenclature IEC-1131-3[6]. Unfortunately, there has been much controversy[7][8] and confusion generated by this effort, particularly with respect to the inclusion of a function chart framework (Sequential Function Chart, or "SFC") as part of the standard.

SFC is not a language, nor was it intended to be. Rather, it is a means of encapsulating Boolean or procedural code in the form of a flow chart. Yet, characterizations in the trade press have misled many into the belief that a structure and syntax exists in IEC-1131-3 for a state language.

As shown in Figure 8, taken from the draft version of the standard, SFC programs are actually composed of language elements drawn from the standard's underlying languages. These languages, two combinatorial (Ladder Diagram and Function



Figure 8

INTEGRATING NON-DIGITAL FUNCTIONS

Many of the operations which have now become necessary additions to manufacturing operations involve such non-digital functions as servo control, analog data acquisition, and data manipulation and storage. These functions are often complex, and require embedded descriptive and parametric data. The textual representation of state language instructions allows high-level instructions to be included to accommodate these functions.

Figure 9 illustrates the use of high-level commands to accomplish servo control. In this implementation, the motion

```
[125] EXTEND_ARM
... This step illustrates the use of high-level instructions
... in a state language framework. In this case, a 'profile'
... instruction establishes a new speed for a subsequent servo
... motion. This speed is automatically derived from a manual
... thumbwheel switch setting.
...
... The servo is then commanded to begin its motion to a new
... position. The final instruction takes the program flow
... to the next step, but only after the servo has completed
... its motion.
-----
<NO CHANGE IN DIGITAL OUTPUTS>
-----
profile arm_servo maxspeed=speed_thumbwheel
turn arm_servo to extended_position
monitor arm_servo:stopped goto Next
```

Figure 9

commands are in the same portion of the step as the monitor instruction for terminating the step. Although initially this may seem like a departure from the structure described earlier, which noted a clear separation between motion commands and step-transition instructions, there are important functional reasons for this intermingling.

The most significant reason is that intermingling allows complex decision making to be performed within a single step. For example, the series of instructions in Figure 10 call for successive values to be stored in a numeric register. After each new value is stored, a test is made of a gauging device on an analog input, testing for successively higher values. Once a test succeeds, an immediate jump is made to the next step, with the resulting classification value stored in the numeric register. Using this approach, the kind of decision making that is typical in a procedural programming language (and often necessary in machine control) may be accomplished without the awkwardness of using large numbers of steps.

HANDLING ASYNCHRONOUS OPERATIONS

The earliest attempts to develop a truly general automation language within a state framework found only limited application due to a single fundamental flaw. Although the vast majority of machine control applications can most suitably be cast as a state-based sequence, there are frequently multiple such sequences which must be controlled simultaneously and asynchronously on a single machine. Attempts to merge these sequences into a single linear flow of events not only result in a lack of clarity, but also may carry significant performance penalties.

This was resolved in the early 1980s with the implementation of multitasking in conjunction with a state language. The state language framework allows multitasking to be incorporated in an elegant and consistent manner, simply by the addition of a new instruction. In the example shown in Figure 12, the instruction will cause the program flow to split into two separate paths, each of which operates independent of the other. The targets of this instruction, two steps named load_workpiece and check_status, each commence a sequence of steps which will run asynchronously to completion.

The resultant program flow, shown in Figure 11, accommodates instances where a number of simultaneous, but independent, operations must be performed. It is possible to nest such tasks further, allowing highly complex machines to be

```
[115] PART_CLASSIFICATION
::: Here we perform a series of measurements on our finished
::: workpiece using a height gauge connected to an analog
::: input (named "height_gauge"). Before each measurement
::: we store a new value to register "category".
:::
::: As soon as a test succeeds, the program instantly jumps
::: to the next step, with the correct classification value
::: in "category". If all of the tests fail, the program
::: jumps to the step "REJECT_PART", where the part is sent
::: to the reject bin
-----
<NO CHANGE IN DIGITAL OUTPUTS>
-----
store 0 to category
if height_gauge <=bin_A goto Next
store 1 to category
if height_gauge <=bin_B goto Next
store 2 to category
if height_gauge <=bin_C goto Next
store 3 to category
goto REJECT_PART
```

Figure 10



Figure 11

divided down and controlled in terms of their native sequences.

This mechanism for multitasking has a utility beyond the simple accommodation of parallel asynchronous tasks. It encourages the modularization of programs by creating a new organizational unit beyond the step level: the task.

ENCAPSULATION: THE FUTURE OF CONTROL LANGUAGE EVOLUTION

The task construct points the way to a use of state language that makes even further progress toward the reduction of complexity. Today, state language programs may be organized such that most of a machine’s functionality resides in tasks, each task relating to one mechanism or mechanical module on the machine. Using this approach, the program can become almost trivially easy to maintain.

When this programming practice is followed, the “main” program for a machine becomes a short sequence of task invocations, clearly representing the “overview” functionality of the machine. To examine the details of a given operation, you move to the appropriate task and the sequence of events for that portion of the program is stated in a clear and concise manner.

This technique, referred to as encapsulation, allows the complexity of the detailed control of a machine’s actuators to be hidden from view when examining the “top level” program flow for a machine. This makes the program easier to understand, and acts as a navigation aid for finding the portion of a potentially lengthy and complex program you wish to examine or modify.

State languages provide a natural means to encapsulate complexity. Work is now being performed to develop higher-level, and more powerful, encapsulation tools to meet the projected demands of an increasingly complex manufacturing environment.

SUMMARY

Changes in automation requirements and practices have resulted in increasing dissatisfaction with traditional programming methods. This has caused a reexamination of language options, and a trend toward the adoption of state language as the programming paradigm of choice. State languages can substantially remove barriers to the incorporation of new sensing and actuation technologies, better accommodate complex applications, and facilitate understanding of machine functionality among an increasingly diverse base of stakeholders.

Encapsulation techniques, used in conjunction with underlying state language programs, promise further reductions in design time and apparent complexity.

REFERENCES

- [1] Crater, Kenneth C. A New Control Paradigm. Proceedings, 1991 Industrial Computing Society Conference, pp. 545-556. <http://www.ctc-control.com/>
- [2] Control Technology Corporation. Quickstep Language and Programming Guide. Control Technology Corporation, 1996.
- [3] You can visit Dr. Petri’s website at http://www.informatik.uni-hamburg.de/TGI/mitarbeiter/profs/petri_eng.html, although a more illuminating website for the investigation of Petri nets is at <http://www.daimi.aau.dk/PetriNets/>.
- [4] Dove, Rick. The 21st Century Manufacturing Enterprise Strategy, or What is All This Talk About Agility? Paradigm Shift International, 1992.
- [5] Martin, T. Human Software Requirements Engineering for Computer-controlled Manufacturing Systems. Automatica, Vol. 19, No. 6, pp. 755-758, 1983.
- [6] EC1131 Part 3-93. Programmable controllers - Part 3: Programming languages. 1st edition. Geneva, Switzerland: International Electrotechnical Commission, 1993.
- [7] Grenard, Jack, et al. IEC-1131-3 Roundtable

Discussion. <http://www.ctc-control.com/carefree/roundtables/1131.html>

[8] Crater, Kenneth C. When Technology Standards Become Counterproductive. <http://www.ctc-control.com/tutorials/language/counter.htm>, July 8, 1992.

Ken Crater is Chairman and cofounder of Control Technology Corporation, a producer of programmable automation controllers and software which make use of state language technology. Mr. Crater also cofounded the Industrial Computing Society, in which he served as first President.

Upgrading electrical switchgear? Magna has the solution.

Improve safety and ensure the reliability of your medium voltage switchgear with arc-resistant retrofits by Magna.

Arc-resistant retrofits are available for most manufacturers and vintages of metal-clad and metal-enclosed switchgear from 5-35 kV.

Designed and tested to EEMAC G14 -1 and exceeds IEEE standard C37-20-7

NFPA 70E requires this level of personal protective equipment when operating non-arc-resistant switchgear.

Magna offers:

- Power system arc flash studies
- Circuit breaker retrofits
- Relay protection upgrades
- NETA certified technicians
- 24-hour emergency service nation-wide

Arc-resistant retrofits provide through the door racking of circuit breakers, reducing the risk of worker injury in the event of a failure and eliminating the need for flash suits.

Heavy duty doors with multiple hinges and extra strength locking devices help ensure the door will not open under extreme force.

Reduced downtime. Arc-resistant retrofits can generally be done in less time than switchgear replacements.



Magna Electric Corporation

851 - 58th Street East,
Saskatoon SK
S7K 6X5

T: 1-877-955-8131
F: (306) 955-9181

www.magnaelectric.com

Regina, SK Winnipeg, MB North Bay, ON Mobile, AL
Edmonton, AB Calgary, AB Vancouver, BC Burlington, ON

#1 in Industrial Ethernet Control

750-841 Programmable Controller

*EtherNet/IP™ Conformance Tested
32 Bit/100 Mbps/512K Program
Memory*

750-842 Programmable Controller

*16 Bit/10 Mbps/128K Program
Memory*

750-341 I/O Buscoupler

*EtherNet/IP Conformance
Tested, 100 Mbps Fast Ethernet*

750-342 I/O Buscoupler

10 Mbps Ethernet

750-500 Industrial Switch

5-port 10/100 Mbps Ethernet



Looking to implement Ethernet in your next process or machine control application? Then look to WAGO. Our complete line of Ethernet enabled programmable controllers, buscouplers, industrial PCs and switches, combined with the following features, make WAGO the #1 choice for industrial Ethernet automation and control solutions.

- **#1 in Ethernet.** Protocol support includes Modbus/TCP, EtherNet/IP™, Modbus/UDP, PROFINet, HTTP, Multicast, FTP, SNMP, SMTP, SNTP, DNS, Boot P/DHCP, and more.
- **#1 in IT Integration.** Enterprise-wide connectivity, e-mail alarms, and other internet applications.
- **#1 in Granularity.** Modular 1, 2, 4, or 8 channel I/O modules - buy only the I/O you need.
- **#1 in Density.** Beginning at 2" wide for a controller and less than 1/2" wide for I/O modules.
- **#1 in Signal Mix.** Over 100 digital, analog, and special function I/O modules.
- **#1 in Reliability.** Due to the use of CAGE CLAMP® maintenance-free wiring technology and 100% testing of all WAGO-I/O-SYSTEM components.
- **#1 in Value.** Ethernet enabled Programmable Controller and fully compliant IEC 61131-3 programming tool for less than \$1,300.00.

For a free Ethernet brochure and tool-kit CD,
contact WAGO today at 1-800 DIN Rail (346-7245)
or info.us@wago.com
www.wago.us/ethernet.htm

WAGO®
INNOVATIVE CONNECTIONS