International Journal of Advanced Multidisciplinary Research
ISSN: 2393-8870
www.ijarm.com
DOI: 10.22192/ijamr Volume 6, Issue 4-2019

Review Article

DOI: http://dx.doi.org/10.22192/ijamr.2019.06.04.004

‘Programmable logic controller’

Abu Sayem[1], Tajmin Nahar[2], Al-Shams[2], Abu Sayed[3], Masum Rana[3], Alamin Bin Golam[3], Md Khaza Main Uddin[4]

[1]Jiangsu Agri-Animal Husbandry Vocational College, Taizhou, Jiangsu; China.
[2] Hebei College Of Industry & Technology, China.
[3] Jiangsu Agri-Animal Husbandry Vocational College, Taizhou, Jiangsu; China.
[4] University of Electronic Science and Technology, China.
Corresponding Author: **Abu Sayem[1],** E-mail: *abusayem199738@hotmail.com*

Keywords

PLC programming , Control.

Abstract

A programmable logic controller (PLC) or programmable controller is an industrial digital computer which has been ruggedized and adapted for the control of manufacturing processes, such as assembly lines, or robotic devices, or any activity that requires high reliability control and ease of programming and process fault diagnosis.

Introduction

PLCs can range from small modular devices with tens of inputs and outputs (I/O), in a housing integral with the processor, to large rack-mounted modular devices with a count of thousands of I/O, and which are often networked to other PLC and SCADA systems.

They can be designed for multiple arrangements of digital and analog I/O, extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. Programs to control machine operation are typically stored in battery-backed-up or non-volatile memory.

It was from the automotive industry in the USA that the PLC was born. Before the PLC, control, sequencing, and safety interlock logic for manufacturing automobiles was mainly composed of relays, cam timers, drum sequencers, and dedicated closed-loop controllers. Since these could number in the hundreds or even thousands, the process for updating such facilities for the yearly model change-over was very time consuming and expensive, as electricians needed to individually rewire the relays to change their operational characteristics.

When digital computers became available, being general-purpose programmable devices, they were soon applied to control sequential and combinatorial logic in industrial processes. However these early computers required specialist programmers and stringent operating environmental control for temperature, cleanliness, and power quality. To meet these challenges the PLC was developed with several key attributes. It would tolerate the shop-floor environment, it would support discrete (bit-form) input and output in an easily extensible manner, it would not require years of training to use, and it would permit its operation to be monitored. Since many industrial processes have timescales easily addressed by millisecond response times, modern (fast, small, reliable) electronics greatly facilitate building reliable controllers, and performance could be traded off for reliability.]

Programming

PLC programs are typically written in a special application on a personal computer, then downloaded by a direct-connection cable or over a network to the PLC. The program is stored in the PLC either in battery-backed-up RAM or some other non-volatile flash memory. Often, a single PLC can be programmed to replace thousands of relays.

Early PLCs, up to the mid-1990s, were programmed using proprietary programming panels or special-purpose programming terminals, which often had dedicated function keys representing the various logical elements of PLC programs.[2] Some proprietary programming terminals displayed the elements of PLC programs as graphic symbols, but plain ASCIIcharacter representations of contacts, coils, and wires were common. Programs were stored on cassette tape cartridges. Facilities for printing and documentation were minimal due to lack of memory capacity. The oldest PLCs used non-volatile magnetic core memory.

Control example shown in ladder diagram

This is a programming example in ladder diagram which shows the control system. A ladder diagram is a method of drawing control circuits which pre-dates PLCs. The ladder diagram resembles the schematic diagram of a system built with electromechanical relays.

In ladder diagram, the contact symbols represent the state of bits in processor memory, which corresponds to the state of physical inputs to the system. If a discrete input is energized, the memory bit is a 1, and a "normally open" contact controlled by that bit will pass a logic "true" signal on to the next element of the ladder. Therefore, the contacts in the PLC program that "read" or look at the physical switch contacts in this case must be "opposite" or open in order to return a TRUE for the closed physical switches. Internal status bits, corresponding to the state of discrete outputs, are also available.

```
| (N.C. physical   (N.C. physical                    |

|  Switch)        Switch)                         |
|  Low Level      High Level             Fill Valve     |
|------[ ]------|------[ ]---------------------(OUT)---------|
|           |                            |
|           |                            |
|           |                            |
|  Fill Valve  |                                |
|------[ ]------|                               |
|                                        |
|                                        |
```

Functionality

The functionality of the PLC has evolved over the years to include sequential relay control, motion control, process control, distributed control systems, and networking. The data handling, storage, processing power, and communication capabilities of some modern PLCs are approximately equivalent to desktop computers. PLC-like programming combined with remote I/O hardware, allow a general-purpose desktop computer to overlap some PLCs in certain applications. Desktop computer controllers have not been generally accepted in heavy industry because the desktop computers run on less stable operating systems than do PLCs, and because the desktop computer hardware is typically not designed to the same levels of tolerance to temperature, humidity, vibration, and longevity as the processors used in PLCs. Operating systems such as Windows do not lend themselves to deterministic logic execution, with the result that the controller may not always respond to changes of input status with the consistency in timing expected from PLCs. Desktop logic applications find use in less critical situations, such as laboratory automation and use in small facilities where the application is less demanding and critical.

Programmable logic relay (PLR)

In more recent years, small products called PLRs (programmable logic relays), and also by similar names, have become more common and accepted. These are much like PLCs, and are used in light industry where only a few points of I/O (i.e. a few signals coming in from the real world and a few going out) are needed, and low cost is desired. These small devices are typically made in a common physical size and shape by several manufacturers, and branded by the makers of larger PLCs to fill out their low end product range. Popular names include PICO Controller, NANO PLC, and other names implying very small controllers. Most of these have 8 to 12 discrete inputs, 4 to 8 discrete outputs, and up to 2 analog inputs. Size is usually about 4" wide, 3" high, and 3" deep. Most such devices include a tiny postage-stamp-sized LCD screen for viewing simplified ladder logic (only a very small portion of the program being visible at a given time) and status of I/O points, and typically these screens are accompanied by a 4-way rocker push-button plus four more separate push-buttons, similar to the key buttons on a VCR remote control, and used to navigate and edit the logic. Most have a small plug for connecting via RS-232 or RS-485 to a personal computer so that programmers can use simple Windows applications for programming instead of being forced to use the tiny LCD and push-button set for this purpose.

PLC Topic

Process of a scan cycle

There are 5 main steps in a scan cycle:

1. Reading inputs
2. Executing the program
3. Processing communication requests
4. Executing CPU diagnostics
5. Writing outputs

System scale

A small PLC will have a fixed number of connections built in for inputs and outputs. Typically, expansions are available if the base model has insufficient I/O.

Modular PLCs have a chassis (also called a rack) into which are placed modules with different functions. The processor and selection of I/O modules are customized for the particular application. Several racks can be administered by a single processor, and may have thousands of inputs and outputs. Either a special high speed serial I/O link or comparable communication method is used so that racks can be distributed away from the processor, reducing the wiring costs for large plants. Options are also available to mount I/O points directly to the machine and utilize quick disconnecting cables to sensors and valves, saving time for wiring and replacing component.

Security

Prior to the discovery of the Stuxnet computer worm in June 2010, security of PLCs received little attention. Modern PLCs generally contain a real-time operating system such as OS-9 or VxWorks, and exploits for these systems exist much as they do for desktop computer operating systems such as Microsoft Windows. PLCs can also be attacked by gaining control of a computer they communicate wit

Discrete (digital) and analog signals

Discrete (digital) signals behave as binary switches, yielding simply an On or Off signal (1 or 0, True or False, respectively). Push buttons, limit switches, and photoelectric sensorsare examples of devices providing a discrete signal. Discrete signals are sent using either voltage or current, where a specific range is designated as *On* and another as *Off*. For example, a PLC might use 24 V DC I/O, with values above 22 V DC representing *On*, values below 2VDC representing *Off*, and intermediate values undefined. Initially, PLCs had only digital I/O.

Analog signals are like volume controls, with a range of values between zero and full-scale. These are typically interpreted as integer values (counts) by the PLC, with various ranges of accuracy depending on the device and the number of bits available to store the data. As PLCs typically use 16-bit signed binary processors, the integer values are limited between -32,768 and +32,767. Pressure, temperature, flow, and weight are often represented by analog signals. Analog signals can use voltage or current with a magnitude proportional to the value of the process signal. For example, an analog 0 to 10 V or 4-20 mA input would be converted into an integer value of 0 to 32767.

Current inputs are less sensitive to electrical noise (e.g. from welders or electric motor starts) than voltage inputs.

References

1. HRASNICA,HAIDINE,LEHNERT. Broadband Powerline Communications Network design. 2004. ISBN:0-470-85741-2.
2. PRASAD, K.V. Principles of Digital Communication Systems and Computer Networks. Charles RIver Media, 2003. ISBN 1584503297.
3. ModemTec. www.modemtec.cz [online]. 2008/04/10.
4. Keller, William L Jr. *Grafcet, A Functional Chart for Sequential Processes*, 14th Annual International Programmable Controllers Conference Proceedings, 1984, p. 71-96.
5. Gregory K. McMillan, Douglas M. Considine (ed), *Process/Industrial Instruments and Controls Handbook Fifth Edition*, McGraw-Hill, 1999 ISBN 0-07-012582-1 Section 3 *Controllers*

6.Petruzella,, Frank D. (2005). "Programmable logic controllers". Tata McGraw-Hill Educatio

7.Harms, Toni M. & Kinner, Russell H. P.E., *Enhancing PLC Performance with Vision Systems*. 18th Annual ESD/HMI International Programmable Controllers Conference Proceedings, 1989, p. 387-399.

Access this Article in Online	
	Website: www.ijarm.com
Quick Response Code	Subject: Computer Science
DOI:10.22192/ijamr.2019.06.04.004	

How to cite this article:
Abu Sayem, Tajmin Nahar, Al-Shams, Abu Sayed, Masum Rana, Alamin Bin Golam, Md Khaza Main Uddin. (2019). ‘Programmable logic controller’ . Int. J. Adv. Multidiscip. Res. 6(4): 21-24.
DOI: http://dx.doi.org/10.22192/ijamr.2019.06.04.004