Intelligent Control and Automation, 2012, 3, 44-49

Scientific Research

http://dx.doi.org/10.4236/ica.2012.31006 Published Online February 2012 (http://www.SciRP.org/journal/ica)

PLC technique discussion

Thomas.J.Byers

Electronic Test Equipment-principles and Applications.Princeton University.
American.
Email:t_byers@hotmail.com

Received October 31, 2011; revised December 1, 2011; accepted December 9, 2011

ABSTRACT

The design and implementation of a classical control system laboratory based on PLC control system is introduced in this paper. To design and implement the system, two parts must be produced. The first is software for PLC and the second is hardware for experiments related to the programs. The PLC control system used in the design is LS industrial system company GM7-DR40A 24/16 Digital I/O and single Analog I/O module, two photoelectric sensors from Atonic company': the first with the model BR100-DDT-P, and the second BEN10M-TFR. An approximate sensor with 5-sides is detected, four of CMOS BCD-7-Segment driven by CD4511B, two relays: 2-poles and 3-poles, six voltages and an ammeter measurement, DC motor and 24 VDC power supply and many connections and pinions. Satisfactory results are obtained by executing twenty four experiments for classical control theory that fulfill the requirements of control theory in undergraduate stage and replace the old experiments executed by PID controller where the practice system is implemented by PLC control now.

Keywords: PLC Control System; Photoelectric Sensor

1 .About Programmable Logic Controllers (PLCs)

PLCs (programmable logic controllers) are the control hubs for a wide variety

of automated systems and processes. They contain multiple inputs and outputs that use transistors and other circuitry to simulate switches and relays to control equipment. They are programmable via software interfaced via standard computer interfaces and proprietary languages and network options.

Programmable logic controllers I/O channel specifications include total number of points, number of inputs and outputs, ability to expand, and maximum number of channels. Number of points is the sum of the inputs and the outputs. PLCs may be specified by any possible combination of these values. Expandable units may be stacked or linked together to increase total control capacity. Maximum number of channels refers to the maximum total number of input and output channels in an expanded system. PLC system specifications to consider include scan time, number of instructions, data memory, and program memory. Scan time is the time required by the PLC to check the states of its inputs and outputs. Instructions are standard operations (such as math functions) available to PLC software. Data memory is the capacity for data storage. Program memory is the capacity for control software.

Available inputs for programmable logic controllers include DC, AC, analog, thermocouple, RTD, frequency or pulse, transistor, and interrupt inputs. Outputs for PLCs include DC, AC, relay, analog, frequency or pulse, transistor, and triac. Programming options for PLCs include front panel, hand held, and computer.

Programmable logic controllers use a variety of software programming languages for control. These include IEC 61131-3, sequential function chart (SFC), function block diagram (FBD), ladder diagram (LD), structured text (ST), instruction list (IL), relay ladder logic (RLL), flow chart, C, and Basic. The IEC 61131-3 programming environment provides support for five languages specified by the global standard: Sequential Function Chart, Function Block Diagram, Ladder Diagram, Structured Text, and Instruction List. This allows for multi-vendor compatibility and multi-language programming. SFC is a graphical language that provides coordination of program sequences, supporting alternative

sequence selections and parallel sequences. FBD uses a broad function library to build complex procedures in a graphical format. Standard math and logic functions may be coordinated with customizable communication and interface functions. LD is a graphic language for discrete control and interlocking logic. It is completely compatible with FBD for discrete function control. ST is a text language used for complex mathematical procedures and calculations less well suited to graphical languages. IL is a low-level language similar to assembly code. It is used in relatively simple logic instructions. Relay Ladder Logic (RLL), or ladder diagrams, is the primary programming language for programmable logic controllers (PLCs). Ladder logic programming is a graphical representation of the program designed to look like relay logic. Flow Chart is a graphical language that describes sequential operations in a controller sequence or application. It is used to build modular, reusable function libraries. C is a high level programming language suited to handle the most complex computation, sequential, and data logging tasks. It is typically developed and debugged on a PC. BASIC is a high level language used to handle mathematical, sequential, data capturing and interface functions.

Programmable logic controllers can also be specified with a number of computer interface options, network specifications and features. PLC power options, mounting options and environmental operating conditions are all also important to consider.

2 . INTRODUCTION

For simple programming the relay model of the PLC is sufficient. As more complex functions are used the more complex VonNeuman model of the PLC must be used. A VonNeuman computer processes one instruction at a time. Most computers operate this way, although they appear to be doing many things at once. Consider the computer components shown in Figure 1.

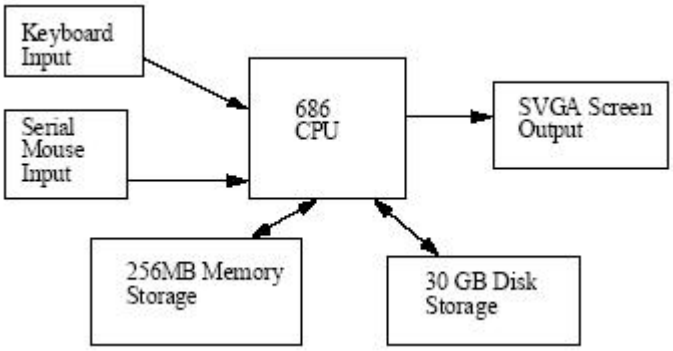


Figure 1 *1* Simplified Personal Computer Architecture

Input is obtained from the keyboard and mouse, output is sent to the screen, and the disk and memory are used for both input and output for storage. (Note: the directions of these arrows are very important to engineers, always pay attention to indicate where information is flowing.) This figure can be redrawn as in Figure 2 to clarify the role of inputs and outputs.

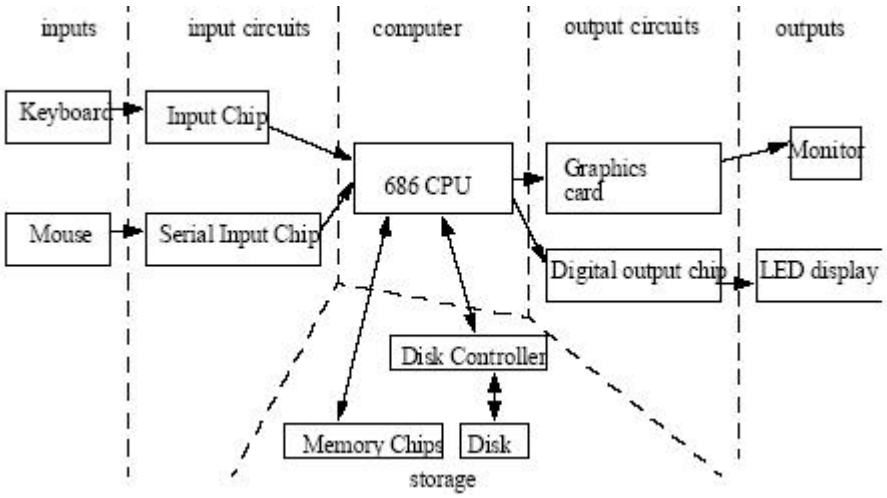


Figure 2 An Input-Output Oriented Architecture

In this figure the data enters the left side through the inputs. (Note: most

engineering diagrams have inputs on the left and outputs on the right.) It travels through buffering circuits before it enters the CPU. The CPU outputs data through other circuits. Memory and disks are used for storage of data that is not destined for output. If we look at a personal computer as a controller, it is controlling the user by outputting stimuli on the screen, and inputting responses from the mouse and the keyboard.

A PLC is also a computer controlling a process. When fully integrated into an application the analogies become;

inputs - the keyboard is analogous to a proximity switch
input -circuits - the serial input chip is like a 24Vdc input card
computer - the 686 CPU is like a PLC CPU unit
output - circuits - a graphics card is like a triac output card
outputs - a monitor is like a light
storage - memory in PLCs is similar to memories in personal computers

It is also possible to implement a PLC using a normal Personal Computer, although this is not advisable. In the case of a PLC the inputs and outputs are designed to be more reliable and rugged for harsh production environments.

3 . OPERATION SEQUENCE

All PLCs have four basic stages of operations that are repeated many times per second. Initially when turned on the first time it will check it's own hardware and software for faults. If there are no problems it will copy all the input and copy their values into memory, this is called the input scan. Using only the memory copy of the inputs the ladder logic program will be solved once, this is called the logic scan. While solving the ladder logic the output values are only changed in temporary memory. When the ladder scan is done the outputs will be updated using the temporary values in memory, this is called the output scan. The PLC now restarts the process by starting a self check for faults. This process typically repeats 10 to

100 times per second as is shown in Figure 3.

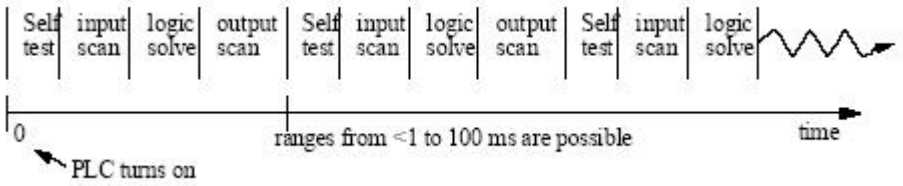


Figure 3 PLC Scan Cycle

SELF TEST - Checks to see if all cards error free, reset watch-dog timer, etc. (A watchdog timer will cause an error, and shut down the PLC if not reset within a short period of time - this would indicate that the ladder logic is not being scanned normally).

INPUT SCAN - Reads input values from the chips in the input cards, and copies their values to memory. This makes the PLC operation faster, and avoids cases where an input changes from the start to the end of the program (e.g., an emergency stop). There are special PLC functions that read the inputs directly, and avoid the input tables.

LOGIC SOLVE/SCAN - Based on the input table in memory, the program is executed 1 step at a time, and outputs are updated. This is the focus of the later sections.

OUTPUT SCAN - The output table is copied from memory to the output chips. These chips then drive the output devices.

The input and output scans often confuse the beginner, but they are important. The input scan takes a *snapshot* of the inputs, and solves the logic. This prevents potential problems that might occur if an input that is used in multiple places in the ladder logic program changed while half way through a ladder scan. Thus changing the behaviors of half of the ladder logic program. This problem could have severe effects on complex programs that are developed later in the book. One side effect of the input scan is that if a change in input is too short in duration, it might fall between input scans and be missed.

When the PLC is initially turned on the normal outputs will be turned off. This does not affect the values of the inputs.

3 .1 The Input and Output Scans

When the inputs to the PLC are scanned the physical input values are copied into memory. When the outputs to a PLC are scanned they are copied from memory to the physical outputs. When the ladder logic is scanned it uses the values in memory, not the actual input or output values. The primary reason for doing this is so that if a program uses an input value in multiple places, a change in the input value will not invalidate the logic. Also, if output bits were changed as each bit was changed, instead of all at once at the end of the scan the PLC would operate much slower.

3 .2 The Logic Scan

Ladder logic programs are modelled after relay logic. In relay logic each element in the ladder will switch as quickly as possible. But in a program elements can only be examines one at a time in a fixed sequence. Consider the ladder logic in Figure 4, the ladder logic will be interpreted left-to-right, top-to-bottom. In the figure the ladder logic scan begins at the top rung. At the end of the rung it interprets the top output first, then the output branched below it. On the second rung it solves branches, before moving along the ladder logic rung.

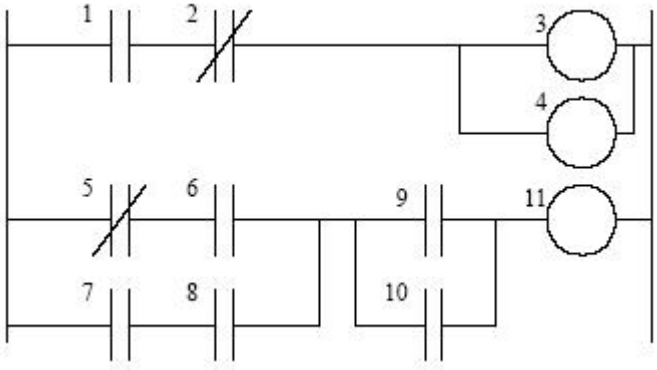


Figure 4 Ladder Logic Execution Sequence

The logic scan sequence become important when solving ladder logic programs which use outputs as inputs. It also becomes important when considering output usage. Consider Figure 5, the first line of ladder logic will examine input *A* and set output *X* to have the same value. The second line will examine input *B* and set the output *X* to have the opposite value. So the value of *X* was only equal to *A* until the second line of ladder logic was scanned. Recall that during the logic scan the outputs are only changed in memory, the actual outputs are only updated when the ladder logic scan is complete. Therefore the output scan would update the real outputs based upon the second line of ladder logic, and the first line of ladder logic would be ineffective.

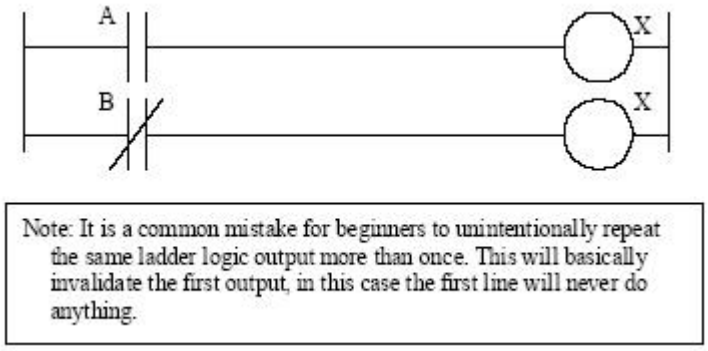


Figure 5 A Duplicated Output Error

4 . PLC STATUS

The lack of keyboard, and other input-output devices is very noticeable on a PLC. On the front of the PLC there are normally limited status lights. Common lights indicate;

power on - this will be on whenever the PLC has power

program running - this will often indicate if a program is running, or if no program is running

fault - this will indicate when the PLC has experienced a major hardware or

software problem

These lights are normally used for debugging. Limited buttons will also be provided for PLC hardware. The most common will be a run/program switch that will be switched to program when maintenance is being conducted, and back to run when in production. This switch normally requires a key to keep unauthorized personnel from altering the PLC program or stopping execution. A PLC will almost never have an on-off switch or reset button on the front. This needs to be designed into the remainder of the system.

The status of the PLC can be detected by ladder logic also. It is common for programs to check to see if they are being executed for the first time, as shown in Figure 6. The ’first scan’ input will be true on the very first time the ladder logic is scanned, but false on every other scan. In this case the address for ’first scan’ in a PLC-5 is ’S2:1/14’. With the logic in the example the first scan will seal on ’light’, until ’clear’ is turned on. So the light will turn on after the PLC has been turned on, but it will turn off and stay off after ’clear’ is turned on. The ’first scan’ bit is also referred to at the ’first pass’ bit.

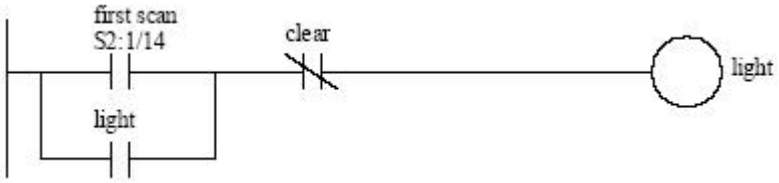


Figure 6 An program that checks for the first scan of the PLC

5 . MEMORY TYPES

There are a few basic types of computer memory that are in use today.

RAM (Random Access Memory) - this memory is fast, but it will lose its contents when power is lost, this is known as volatile memory. Every PLC uses this memory for the central CPU when running the PLC.

ROM (Read Only Memory) - this memory is permanent and cannot be erased. It is often used for storing the operating system for the PLC.

EPROM (Erasable Programmable Read Only Memory) - this is memory that can be programmed to behave like ROM, but it can be erased with ultraviolet light and reprogrammed.

EEPROM (Electronically Erasable Programmable Read Only Memory) – This memory can store programs like ROM. It can be programmed and erased using a voltage, so it is becoming more popular than EPROMs.

All PLCs use RAM for the CPU and ROM to store the basic operating system for the PLC. When the power is on the contents of the RAM will be kept, but the issue is what happens when power to the memory is lost. Originally PLC vendors used RAM with a battery so that the memory contents would not be lost if the power was lost. This method is still in use, but is losing favor. EPROMs have also been a popular choice for programming PLCs. The EPROM is programmed out of the PLC, and then placed in the PLC. When the PLC is turned on the ladder logic program on the EPROM is loaded into the PLC and run. This method can be very reliable, but the erasing and programming technique can be time consuming. EEPROM memories are a permanent part of the PLC, and programs can be stored in them like EPROM. Memory costs continue to drop, and newer types (such as flash memory) are becoming available, and these changes will continue to impact PLCs.

6 . SOFTWARE BASED PLCS

The dropping cost of personal computers is increasing their use in control, including the replacement of PLCs. Software is installed that allows the personal computer to solve ladder logic, read inputs from sensors and update outputs to actuators. These are important to mention here because they don’t obey the previous timing model. For example, if the computer is running a game it may slow or halt the computer. This issue and others are currently being investigated and good solutions should be expected soon.

7 . SUMMARY

• A PLC and computer are similar with inputs, outputs, memory, etc.

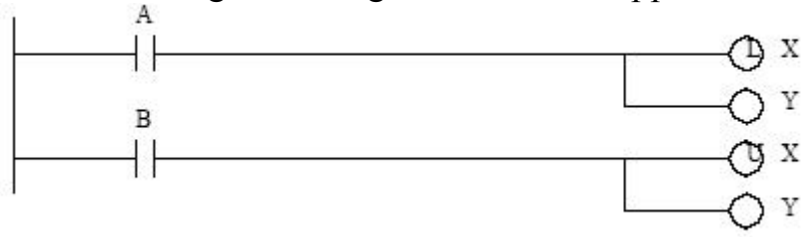
• The PLC continuously goes through a cycle including a sanity check, input scan, logic scan, and output scan.

• While the logic is being scanned, changes in the inputs are not detected, and the outputs are not updated.

• PLCs use RAM, and sometime EPROMs are used for permanent programs.

8 . PRACTICE PROBLEMS

1. Does a PLC normally contain RAM, ROM, EPROM and/or batteries?
2. What are the indicator lights on a PLC used for?
3. A PLC can only go through the ladder logic a few times per second. Why?
4. What will happen if the scan time for a PLC is greater than the time for an input pulse? Why?
5. What is the difference between a PLC and a desktop computer?
6. Why do PLCs do a self check every scan?
7. Will the test time for a PLC be long compared to the time required for a simple program?
8. What is wrong with the following ladder logic? What will happen if it is used?



9. What is the address for a memory location that indicates when a PLC has just been turned on?

9 . PRACTICE PROBLEM SOLUTIONS

1. Every PLC contains RAM and ROM, but they may also contain EPROM or

batteries.

2. Diagnostic and maintenance
3. Even if the program was empty the PLC would still need to scan inputs and outputs, and do a self check.
4. The pulse may be missed if it occurs between the input scans
5. Some key differences include inputs, outputs, and uses. A PLC has been designed for the factory floor, so it does not have inputs such as keyboards and mice (although some newer types can). They also do not have outputs such as a screen or sound. Instead they have inputs and outputs for voltages and current. The PLC runs user designed programs for specialized tasks, whereas on a personal computer it is uncommon for a user to program their system.
6. This helps detect faulty hardware or software. If an error were to occur, and the PLC continued operating, the controller might behave in an unpredictable way and become dangerous to people and equipment. The self check helps detect these types of faults, and shut the system down safely.
7. Yes, the self check is equivalent to about 1ms in many PLCs, but a single program instruction is about 1 micro second.
8. The normal output *Y* is repeated twice. In this example the value of *Y* would always match *B*, and the earlier rung with *A* would have no effect on *Y*.
9. S2:1/14 for micro logy, S2:1/15 for PLC-5

10. Conclusions

The practice results that are obtained in real implementation of all experiments in the laboratory and the simulation results by LD program, it can be seen that the implementation of PLC controller with classical control systems is necessary. This implementation has high performance, high accuracy and more speed response com-pared to the classical controller.

The trainer presents practice simulation for many real systems; therefore it is very suitable for classical laboratory unit with undergraduate students.

REFERENCES

P. Chevtsov, S. Higgins, S. Schaffner and D. Seidman, “PLC Support Software at Jefferson Lab,” Jefferson Lab, Newport News, 2002.

J. R. Hackworth and F. D. Hackworth, “Programmable Logic Controllers: Programming Methods and Applications,” Prentice Hall, Upper Saddle River, p. 13.

L. A. Bryan and E. A. Bryan, “Programmable Controllers Theory and Implementation,” 2nd Edition, Industrial Text Company Publication, Atlanta, 1997.

H. Jack, “Automating Manufacturing Systems with PLCs, Version 4.7,” Copyright (c) 1993-2005 Hugh Jack, 2005.

S. Yurkovich and K. M. Passino, “A Laboratory Course on Fuzzy Control,” *IEEE Transactions on Education*, Vol. 42, No. 1, 1999, pp. 15-21.

A. Sirinterlikci, “Development of a Comprehensive Industrial Controls Course in a Manufacturing Engineering Program,” American Society for Engineering Education, Washington DC, 2006.

X. Yu, X. Feng, C. Xiong and S. Jiaotong, “The Design and Implementation of Elevator Group Control System Research Platform,” *Proceedings of the* 2009 *International Workshop on Information Security and Application*, Qingdao, 21-22 November 2009.