Journal of
INDUSTRIAL
TECHNOLOGY

Volume 15, Number 3 - May 1999 to July 1999

A Systematic Procedure for Designing State Combination Circuits in PLCs

By Dr. Chien-Pen Chuang, Ms. Xing Lan, & Dr. Joseph C. Chen

KEYWORD SEARCH

CIM
Computer Programming

Reviewed Article

The Official Electronic Publication of the National Association of Industrial Technology • www.nait.org

A Systematic Procedure for Designing State Combination Circuits in PLCs

By Dr. Chien-Pen Chuang, Ms. Xing Lan, & Dr. Joseph C. Chen

Dr. Chien-Pen Chuang is an associate professor in the Department of Industrial Education at National Taiwan Normal University. His research interests are in digital signal processing, image processing, digital communication and computerized skill test. He has been an experienced researcher for National Science Council of Taiwan and a patent reviewer in the field of electronic technology after his graduation from Iowa State University.

Ms. Xing Lan is a Ph.D student in the Department of Industrial Education and Technology of Iowa State University. She received a B.S. at 1985 and M.E. at 1990 at Hefei University of Technology at China. She has ten years teaching experience and taught many courses on Computer Language, Database, Artificial Intelligence, Data Structure, and Computer Graphics.

Dr. Joseph C. Chen, PE is an associate professor in the Department of Industrial Education and Technology at Iowa State University. Recently, he received Early Achievement in Teaching Award from the Iowa State University. His research interests are in cellular manufacturing systems, machine tool dynamics, and intelligent control as it relates to neural networks and fuzzy control.

Introduction

A Programmable Logic Controller (PLC) is a user-friendly, microprocessor-based, specialized computer that can carry out control functions of many types at many different levels of complexity. They can operate any system with output devices that turn on and off, as well systems with variable outputs. On-off devices or variable analog sensors can serve as input, and PLC can be programmed, controlled, and operated by any individual with a basic computer background. Due to its easy operation and powerful functions, PLC has become widely used in industry.

The advantages of PLC make it the brain of modern industrial control systems. Its annual sales have reached the billion-dollar mark and continued steady growth is expected (Webb & Reis, 1995). For these reasons, learning PLC operations is integral for modern Industrial Technologists.

A variety of applications of the PLC in industrial control have been published. One example showed how to use the PLC in the paper mill load-shed system (Fievet, 1997). In the paper mill-shed system, the control process consists of three basic parts: sensing, deciding, and responding. *Sensing* focuses on determining if there is an event requiring load-shedding, *deciding* controls decisions about the appropriate loads to shed, and *responding* conducts the load-shedding task consequentially.

Other topics addressed include an analysis of the concepts and functions of PLCs in order to achieve greater teaching effectiveness (Reis, 1996), and the properties and advantages of PLCs when applied to automation problems (How the PLC, 1997). Many other issues have been discussed in the literature, including: a standard programming resource for PLCs (Babb, 1996), an explanation of PLC functions and where they can be employed (Martin, 1996), and a new friendly operator interface system (Morris, 1996). However, the literature is weak in describing any systematic way of designing ladder logic circuits in PLCs.

Purpose

Fievet stated that "Because the number of potential faults and combinations of faults is infinite, any system that relies on preprogrammed scenarios will eventually encounter a situation that it does not recognize" (Fievet, 1997, p. 195). However, if a logical relationship between inputs and outputs can be achieved and a ladder diagram can be drawn based on Boolean theory, many potential faults and combinations of faults can be avoided.

It is important to understand the basic procedure used to construct a ladder diagram. Webb (1995) proposed a method for building a large-process ladder diagram construction (Fievet, 1997):

1. Define the process to be controlled.
2. Make a sketch of the process operation.
3. Create a written sequence listing the process step by step.
4. On the sketch, add the sensors needed to carry out the control sequence.
5. Add manual controls needed for the process-setup or for operational checks.

6. Consider the safety of the operating personnel and make additions and adjustments as needed.
7. Add master stop switches required for safe shutdown.
8. Create a ladder logic diagram that will be used as a basis for the PLC program.
9. Consider the possible points where the process-sequence may go astray.

The control task can be defined completely by following steps 1-7. An important issue for PLC users to consider is how to obtain the ladder diagram from a specified control task. Thus, a detailed procedure of the method used to construct ladder diagrams is a useful complement to the procedure outlined before. This paper expands upon step 8 above to provide a new procedure for ladder diagram construction.

The ladder logic program in a computer takes the place of much of the external wiring required for control of a process; thus it can be seen as a control circuit within a PLC. Basically, there are two types of logic control circuits: state combination and sequential. The feature distinguishing state combination from sequential control circuits is whether the control depends solely on a combination of the inputs or includes a sequence of events.

State combination circuits are solely determined by input states, whereas sequential control circuits are determined by both state combination and a sequence of occurring events. In essence, two types of ladder programs exist: one is developed from a state combination of logic control circuits, and the other is developed from a sequential logic control circuit. The focus of this paper is to provide a procedure that demonstrates how to develop state combination circuits in the PLC.

Methodology

In order to explain the procedure clearly, a detailed procedure accompanied by an example showing how to use the procedure to build a ladder logic diagram is outlined. This example has three inputs and two outputs. The task is to design a control circuit where any combination of two input signals causes the output switch L1 to turn on, and one or no input signals are identified by L2.

Step 1: Identify inputs and outputs. The task is to give each input and output device a specified name, so it can be conveniently represented later.

Three inputs:
- PB1: The first push button
- PB2: The second push button
- PB3: The third push button

Two outputs:
- L1: The first light
- L2: The second light

Step 2: Build an input State Table. The key to designing a state combination circuit is to construct a State Table. The Input State Table contains all possible combinations of the input state. A two-input system, for example, has a total of 4 (or 2^2) combinations of input states. Likewise, a four-input system would have a total of 16 (or 2^4) possible combinations of input states. In this example, a three-input system will have total of 8 (or 2^3) possible combinations of input states (Figure 1).

Step 3: Determine the output states. The Output State of each possible input combination is determined by the control task. In this case, L1 signal when only two of the three inputs have an input of "1". Thus, for output X/L1, row 4, row 6 and row 7 have "1" values and the other rows have "0". L2 will be "on" when just one of the three inputs is "1" or all of the three inputs are "0". Next, for output Y/L2, row 1, row 2, row 3 and row 5 have "1" values and the other rows have "0". Figure 2 depicts a completed State Table that includes all logic control task relationships, which are then used to derive a Boolean equation.

Step 4: Derive the Boolean equations. The rows of the state table which have "1" output state are used to develop the Boolean equations. Each of these rows represents a subset of the Boolean equations. The complete equation is the sum of these subsets. In this example, there are two outputs, X and Y, and the Boolean Equation for the X output is computed first. There are three rows with "1" output and the three subsets are:

ROW4: $L4=\overline{A}\cdot B\cdot C$

ROW6: $L6=A\cdot\overline{B}\cdot C$

ROW7: $L7=A\cdot B\cdot\overline{C}$.

Therefore, the completed Boolean equation for the X output becomes

$$X=L4+L6+L7=\overline{A}\cdot B\cdot C+A\cdot\overline{B}\cdot C+A\cdot B\cdot\overline{C}.$$

PB1/A	PB2/B	PB3/C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Figure 1. Input State Table

INPUT			OUPUT	
PB1/A	PB2/B	PB3/C	X/L1	Y/L2
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	0	0

Figure 2. Completed State Table

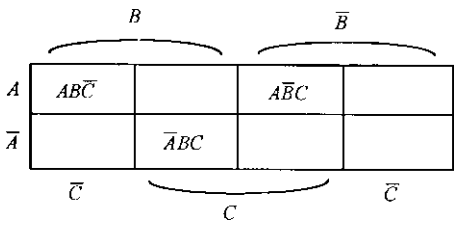

Figure 3. A Karnaugh Map of the Output X

Using the same method as above, the Y Boolean equation can be derived:

$$Y=\overline{A}\cdot\overline{B}\cdot\overline{C}+\overline{A}\cdot\overline{B}\cdot C+\overline{A}\cdot B\cdot\overline{C}+A\cdot\overline{B}\cdot\overline{C}.$$

Step 5: Simplify the Boolean equation. In most cases, the Boolean equations derived from the State Table can be simplified with Boolean laws or the Karnaugh map method. In the X output equation, no further simplification can be achieved because there are no adjacent pairs in the Karnaugh map. The Karnaugh map for the output X is illustrated in Figure 3.

In the Karnaugh map of the Y output (Figure 4), there are three adjacent pairs in the group able to simplify the logical equation:

1. $\overline{A}\cdot\overline{B}\cdot\overline{C}$ *and* $\overline{A}\cdot B\cdot\overline{C}$

2. $\overline{A}\cdot\overline{B}\cdot\overline{C}$ *and* $\overline{A}\cdot\overline{B}\cdot C$

3. $\overline{A}\cdot\overline{B}\cdot\overline{C}$ *and* $A\cdot\overline{B}\cdot\overline{C}$

The three pairs have $(\overline{A}\cdot\overline{B}\cdot\overline{C})$in common, so only one of the pairs needs to be selected in order to simplify the Boolean equation. Any pair can be selected for this purpose. The third pair is used to simplify the Boolean equation of Y in this example. The simplified output can be figured by using Boolean laws, however, using a Karnaugh map is a more direct method.

$$\begin{aligned}Y&=\overline{A}\cdot\overline{B}\cdot\overline{C}+A\cdot\overline{B}\cdot\overline{C}+\overline{A}\cdot\overline{B}\cdot C+\overline{A}\cdot B\cdot\overline{C}\\&=(\overline{A}+A)\cdot\overline{B}\cdot\overline{C}+\overline{A}\cdot(\overline{B}\cdot C+B\cdot\overline{C})\\&=\overline{B}\cdot\overline{C}+\overline{A}\cdot(\overline{B}\cdot C+B\cdot\overline{C}).\end{aligned}$$

Step 6: Construct the ladder diagram and the PLC circuit. In a Boolean equation, there are three logic operators: (1) the "×" is the "and" operator; (2) the "+" is the "or" operator; and (3) the "-" is the "not" operator. If the input has a "not" operator in the Boolean equation, a normal closed relay should be selected to represent it; otherwise, a normal open relay is selected to denote the input in the ladder diagram. If two input parts are connected by the "or" operator in the Boolean equation, these two parts should be connected so they are parallel in the ladder diagram. If two parts are connected by the "and" operator in the Boolean equation, these two parts should be connected serially in the ladder diagram.

A ladder diagram can be easily made according to the Boolean equation. Figure 5 is the ladder diagram for this example. In Figure 5, relays A, B, and C are used for direct control by PB1, PB2 and PB3.

Step 7: Verify the results. Review the state diagram. Trace the circuit through each of the possible input combinations and verify that the result is the same as indicated in the state table.

Step 8: Add the Memory Element to the circuit. In many applications "momentary" switches are used as the input device. In this case, if the actuation of an output must be maintained, even after the input element has been released, a memory element is normally added to provide the maintaining function. The memory is usually an open contact controlled by the output relay, as shown in Figure 6. When the output relay CR is energized by pressing PB, the normally open contact CR is closed to form a holding circuit to continuously energize the output relay.

The memory circuit shown in Figure 6 needs to be improved because there is no way to de-energize the relay coil CR. A normally closed PB needs to be added to the system to build a complete memory circuit (Figure 7). This circuit is often referred to as the power-holding circuit.

If the circuit needs to be transferred in order to maintain the power-holding function, the above method can be used to make the transfer. Suppose the X-output of the previously mentioned circuit needs to be transferred in order to hold power. A normal closed relay and a memory unit must be connected to the ladder diagram (Figure 8).

Conclusion

The logical relationships within the specified control task can be represented directly by the State Table. The State Table is straightforward, so that

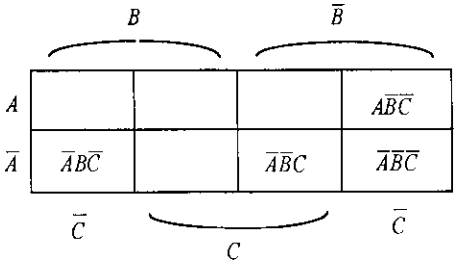

Figure 4. A Karnaugh Map of the Output Y

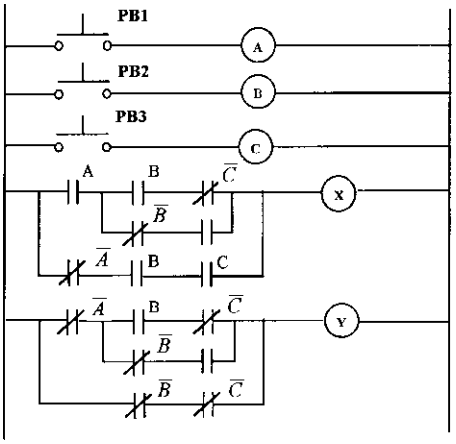

Figure 5. Ladder Diagram

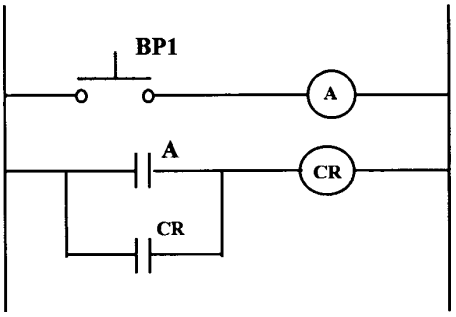

Figure 6. Power-holding Circuit

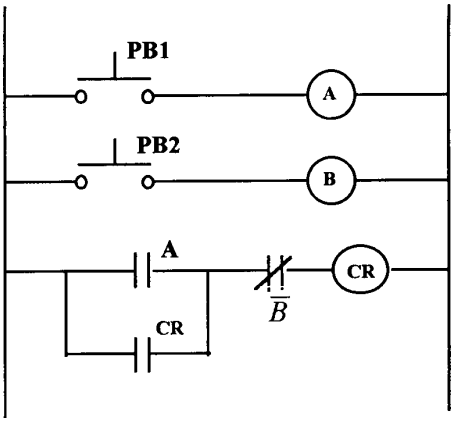

Figure 7. Completed Power-holding Circuit

any mistakes of the control logic can be easily avoided and any complications can also be easily represented. Based on the completed State Table, the Boolean equation can be obtained easily and correctly. By using a Karnaugh map and Boolean laws, the Boolean equation can be simplified directly from the State Table. Using these simplified Boolean equations to draw a ladder diagram is very easy, using the Boolean operation of "×", "+", and "-" to represent series connection, parallel connection, or closed contacts respectively. In essence, this procedure makes the construction of a ladder diagram easier and more reliable for Industrial Technologists.

References

Babb, M. (1996, February) IEC 1131-3: A standard programming resource for PLCs. *Control Engineering*, 67-8.

Fievet, J. P. (1997). The use of PLCs in paper mill load shed systems. *Tappi-Journal*, 80(3), 105-9.

How PLC opens new automation opportunities. (1997, February). *Electrical World*, 46-7.

Martin, V. D. (1996, April). Programmable Logic Controllers. *Electronics Now*, p. 37-8.

Morris, H. M. (1996, April). Operator interfaces let users and PLCs 'Talk' to each other. *Control Engineering*, 69-70.

Reis, R. A. (1996, September). A mini-course for introducing PLCs. *Tech Directions*, 42-45.

Webb, J. W., & Reis, R. A. (1995). *Programmable logic controllers: Principles and applications* (3rd Ed.). Englewood Cliffs, NJ: Prentice Hall, Inc.

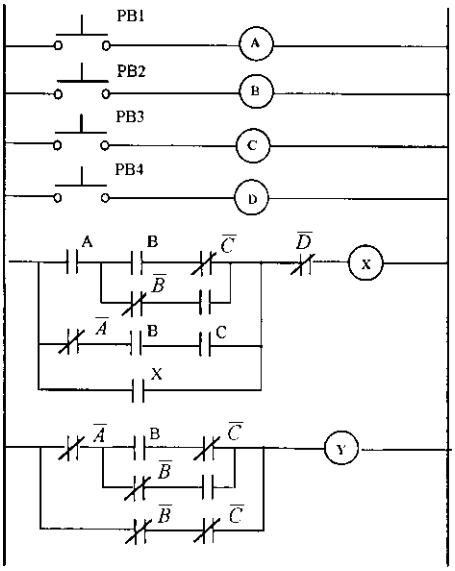

Figure 8.
Ladder Diagram with Memory Element