Programming Logic controllers

Programmable Logic Controller (PLC) is a microprocessor based system that uses programmable memory to store instructions and implement functions such as logic, sequencing, timing, counting and arithmetic in order to control machines and processes.

PLC Features

- PLC features can be concluded in the following:
- They are rugged, withstand industrial environment, such as heat, humidity, mechanical shocks and vibrations
- The interfacing for inputs and outputs is inside the controller
- They are easily programmed
- PLC is capable of both logic and PID control.

Typical PLC

Forms of PLCs

- First developed in 1968, they are now widely used:
- Two types:
- **1- Single – box type:** for small programmable controllers, is supplied as an integral compact package, complete with power supply, processor, memory and input/output units. Typically they may have 6, 8, 12, or 24 inputs and 4, 8, or 16, outputs and a memory store some 300-1000 instructions. Eg. MELSEC FX3U
- **2- Rack mounted type**: for all sizing of programming controllers. It consists of separate modules for power supply, processor, input/output, etc. They are mounted on rails with a metal cabinet. E.g. SIMATIC S7-300/400

Input/output processing

- PLC is continuously running through its program and updating it as a result of the input signals, each such loop is called a cycle.
- Two methods of processing:
- 1- Continuous updating 2- Mass updating
- **continuous updating:**
- The cpu scanning the input channels as they occur in the program instructions. Each input is examined individually (delay time 3 ms). The output is latched so that they retain their status until the next update

Input/output processing

Mass input/ output copying: it works in the following process:

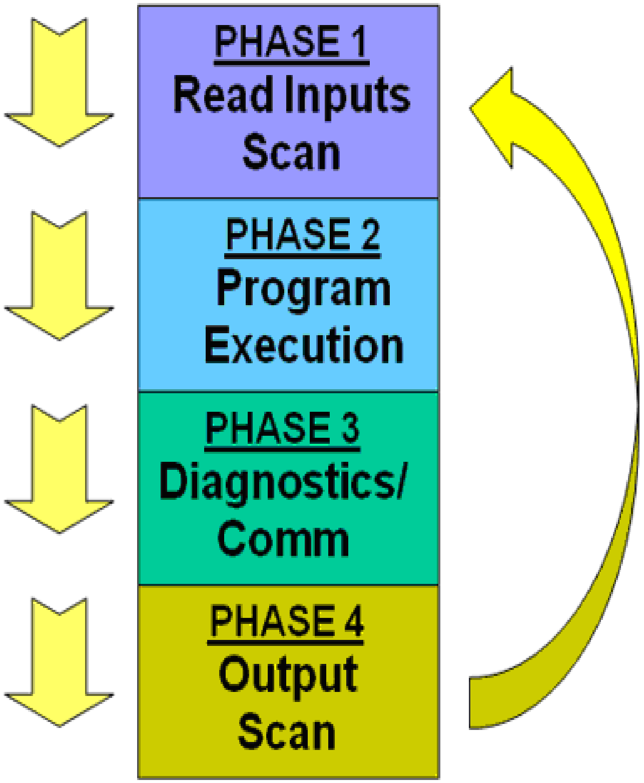
1. Scan all the inputs and copy into RAM

2. Fetch and decode and execute all program instruction in sequence, copying output instruction to RAM

3- Once the program is executed, the CPU performs diagnostics and communication tasks

4- update all outputs

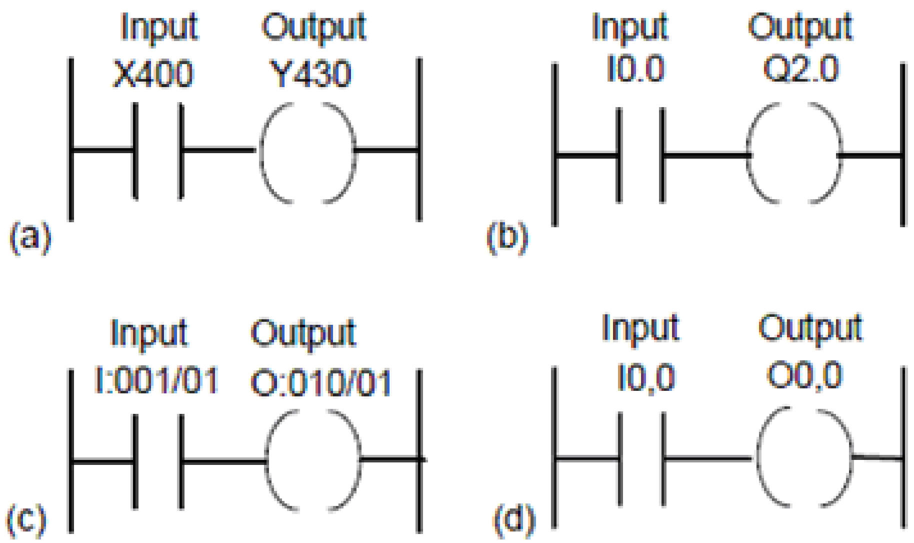
Repeat the sequence.



Input/Output address

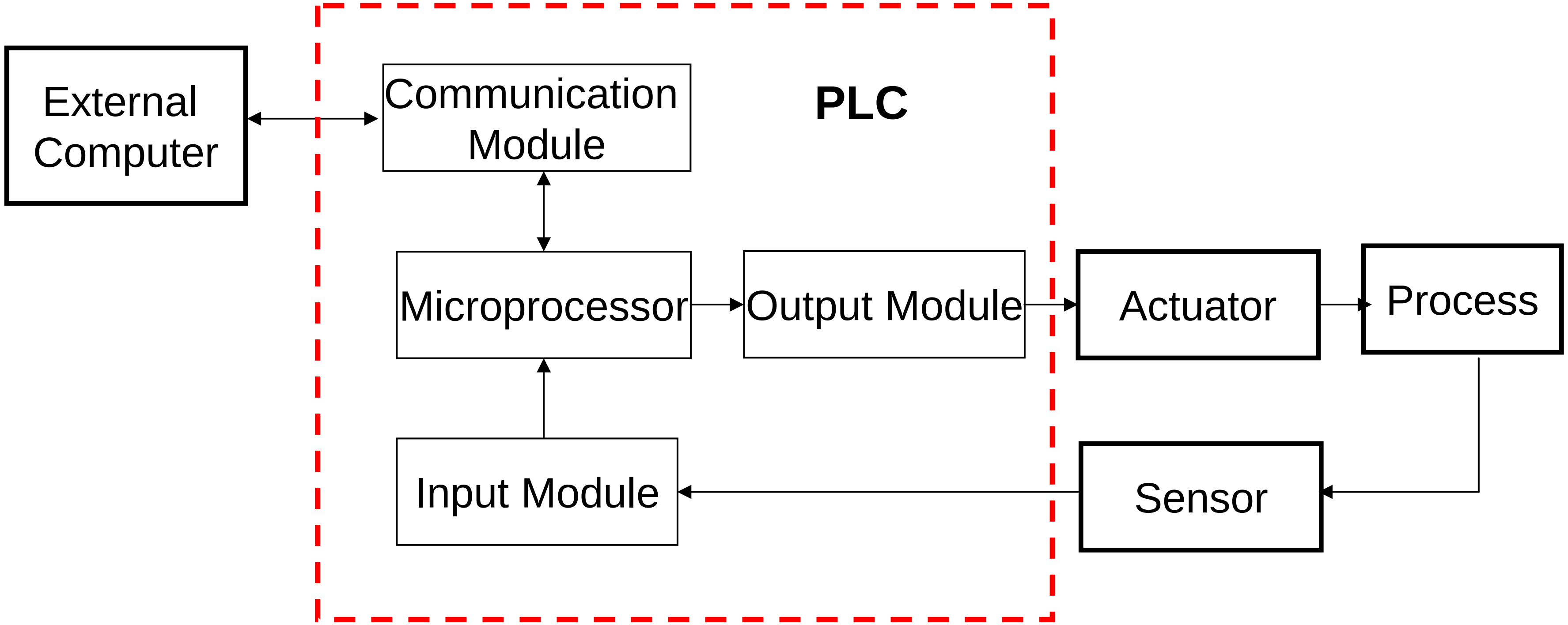
- The inputs and outputs are identified by their addresses, the notation used depending on the PLC manufacturer. This is the address of the input or output in the memory of the PLC. Its just a number preceded by a letter to indicate whether it is an input or an output
- With large PLCs having several racks of input and output and a number of modules in each rack, the rack and modules are numbered and so an input or output is identified by its rack number followed by the number of the module in that rack and the number to show its terminal in the module.:

The Allen-Bradley PLC-5 has I: 012/03 to indicate an input in rack 01 at module 2 and terminal 03

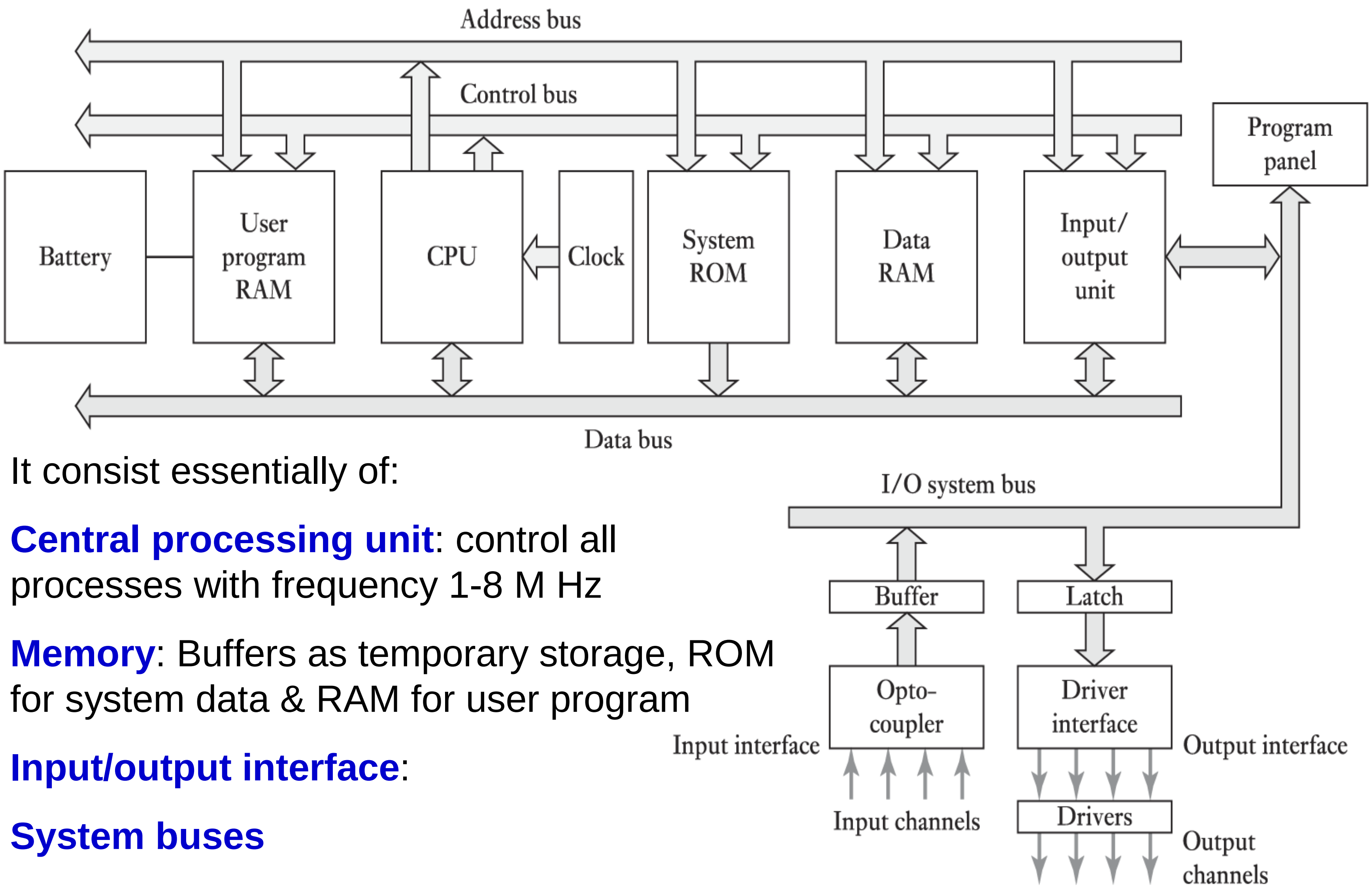


(a) Mitsubishi, (b) Siemens, (c) Allen-Bradley, (d) Telemecanique

PLC In Control Circuits Architecture



Architecture of a PLC



It consist essentially of:

Central processing unit: control all processes with frequency 1-8 M Hz

Memory: Buffers as temporary storage, ROM for system data & RAM for user program

Input/output interface:

System buses

Input/output unit

- **The input/output unit** provides the interface between the system and the outside world.
- The input/ output interface provides isolation and signal conditioning functions so that sensors and actuators can often be directly connected to them without the need for other circuitry.
- Out devices: motors, starting coils, solenoid valve, etc
- Input devices: temperature sensors, flow sensors, encoders, etc…

Input/output unit

Electrical isolation from the external world is usually provided by means of opt isolators. Figure shows the basic form of input channel

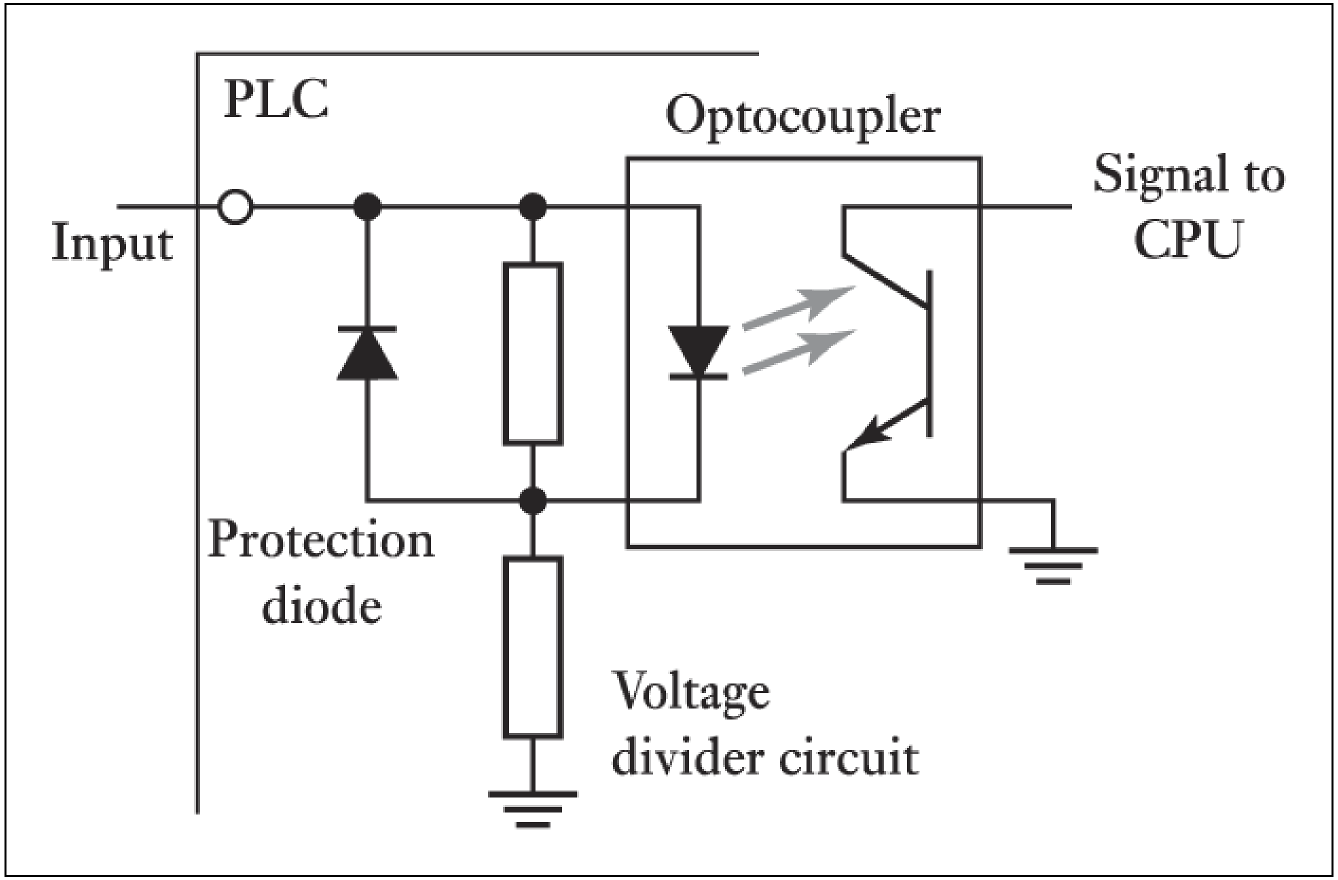


Figure 21.2 Input channel

Input/output unit

- The digital signal that is compatible with the microprocessor of the PLC is 5 volt dc however, signal conditioning in the input channel with isolation enables a wide range of input signals to be supplied. Possible input voltages 5V, 24V, 110V, and 240V.
- The output to the output unit is digital with a level of 5 V,
- Three types of outputs are available: Relay type, Transistor type & triac type

Input/output unit

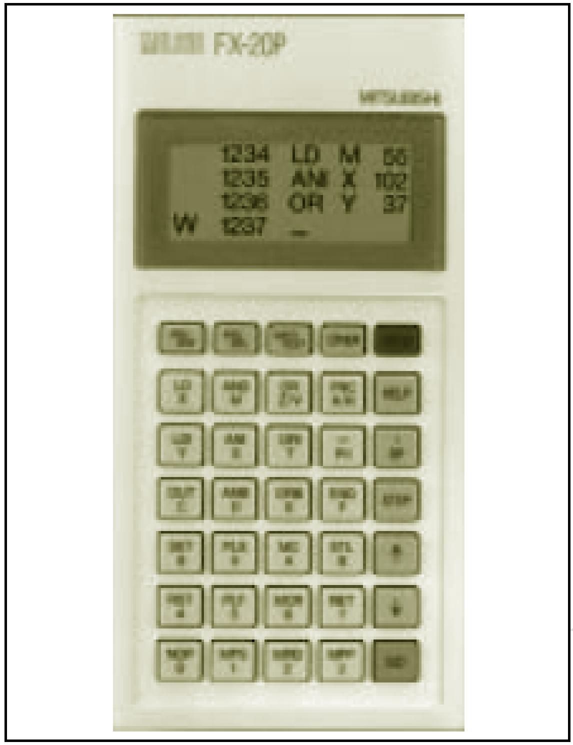
- Three types of outputs are available:
- Relay type: The signal from the PLC is used to operate a relay and so able to switch currents of a few amperes in an external circuit. The relay isolate the PLC from external world, can be used for AC and DC but they are slow
- Transistor type: It is used to switch current through external circuit, fast opto-isolators are used to provide isolation, used only for DC switching
- Triac type: used for both AC and DC
- Possible output from output channels: 24V,100 mA; 100V dc,1 A; 240V,1A ac; or 240V, 2A ac

Inputting Programs

Are entered into the input/output from:

Small hand- held programming devices, desktop consoles with a visual display or key board and screen

Courtesy of Mitsubishi Electronics, Mount Prospect, IL



Courtesy of Mitsubishi Electronics, Mount Prospect, IL

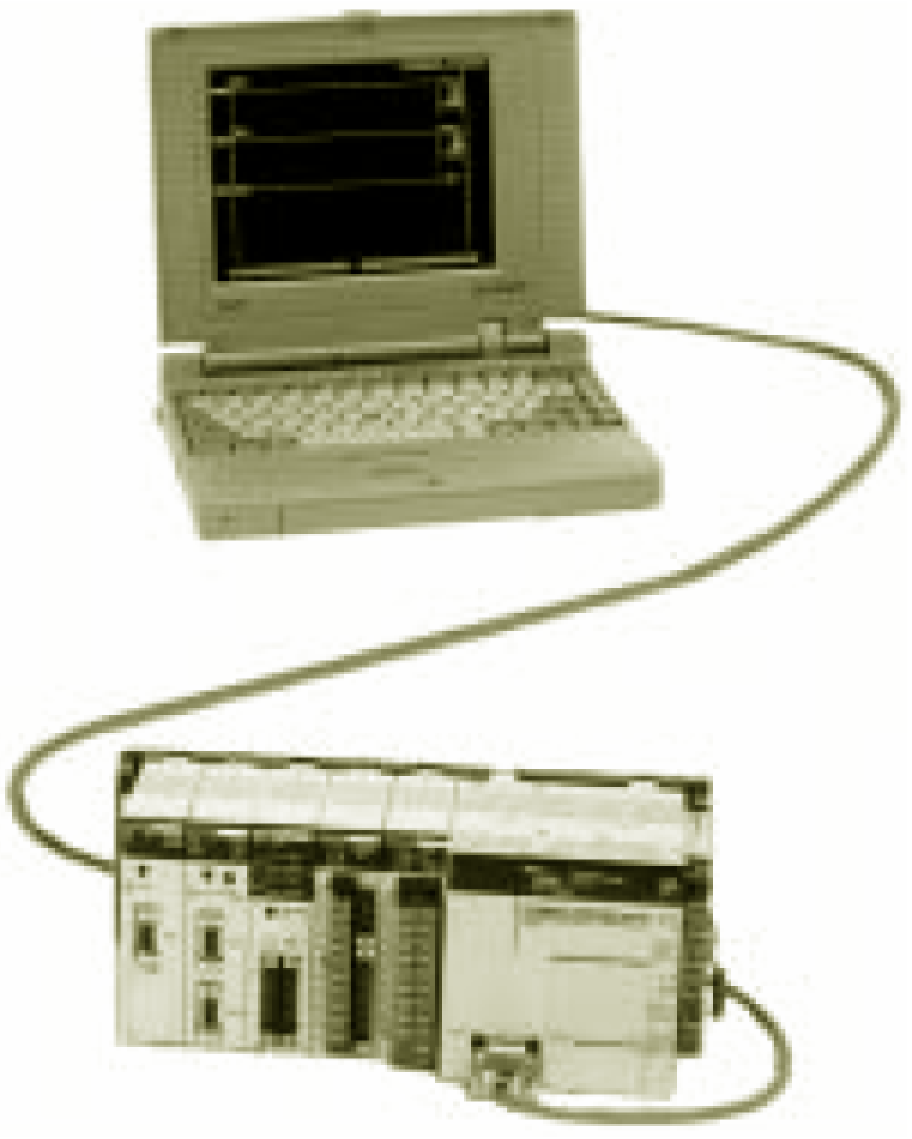
Inputting Programs

Only when the program has been designed and checked on the programming device is transferred to the memory of the PLC

Typical Program memory size is (specified in term of steps) 300 to 1000 step.

Program step is an instruction for some event to occur, ex: check status of switch A

After developing the program in RAM it may be transferred permanently to the EPROM chip



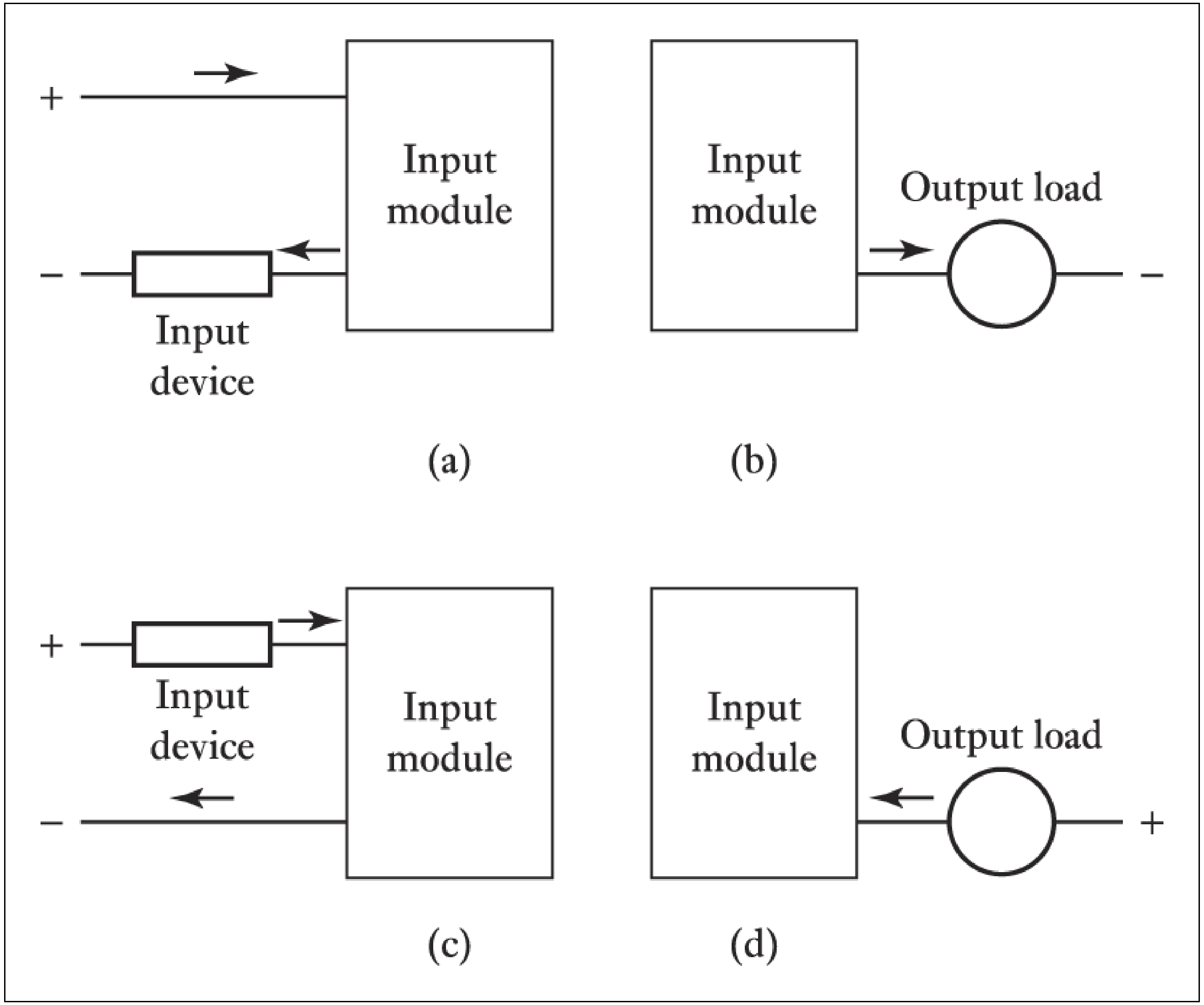


Figure 21.3 (a), (b) Sourcing, (c), (d) sinking

Ladder programming

The form of programming commonly use with PLC is ladder programming. Each program task is specified as though a rung of a ladder.

Thus a rung could specify that the state of switches A and B be examined and if both A and B are closed then a solenoid, the output is energized.

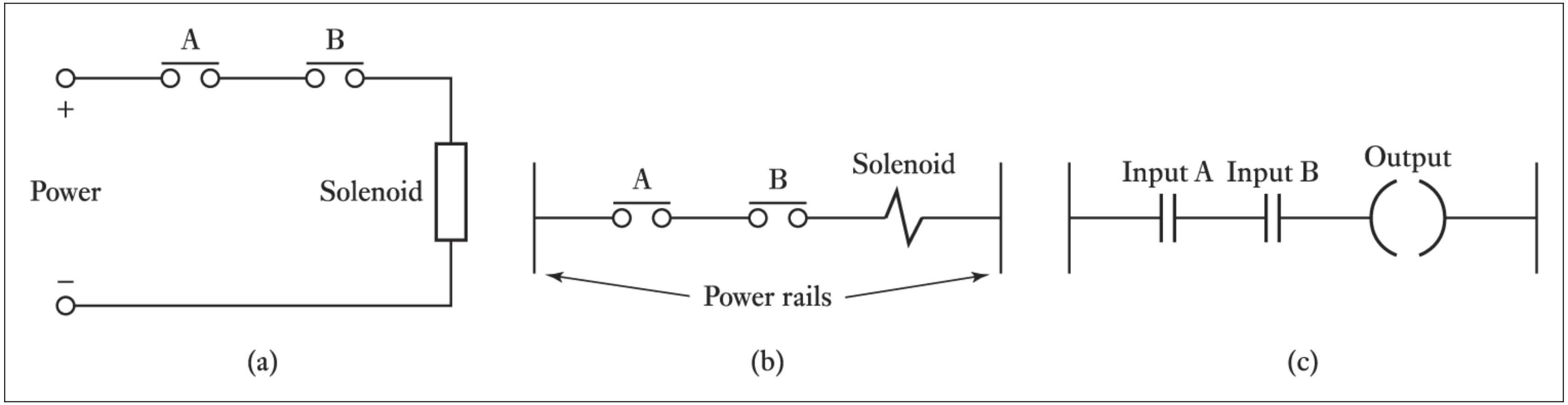


Figure 21.4 (a), (b) Alternative ways of drawing an electric circuit,

(c) comparable rung in a ladder program

Ladder programming

The sequence followed by a PLC when carrying out a program:

1- Scan the inputs associated with one rung of the ladder program

2- solve the logic operation involving those inputs

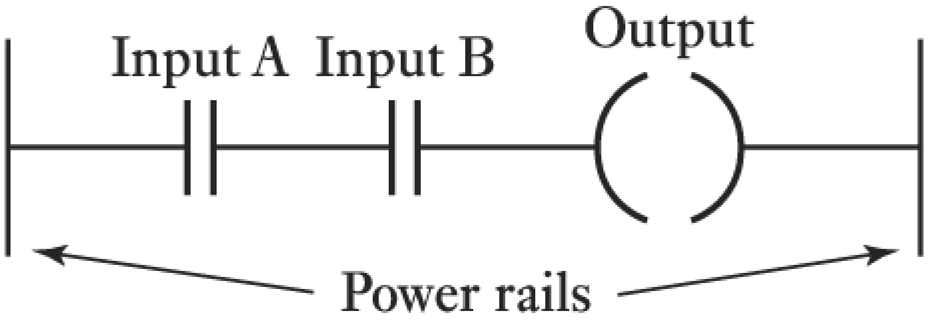
- Set/ reset the outputs for that rung

3- move on to the next rung and repeat operations 1, 2, 3

....and so on until the end of program with each rung of the ladder scanned in turn.

- The PLC then goes back to the begining of the program and starts again
- The ladder diagram consists of two vertical lines representing the power rails.

Circuits (rung) are connected as horizontal lines,

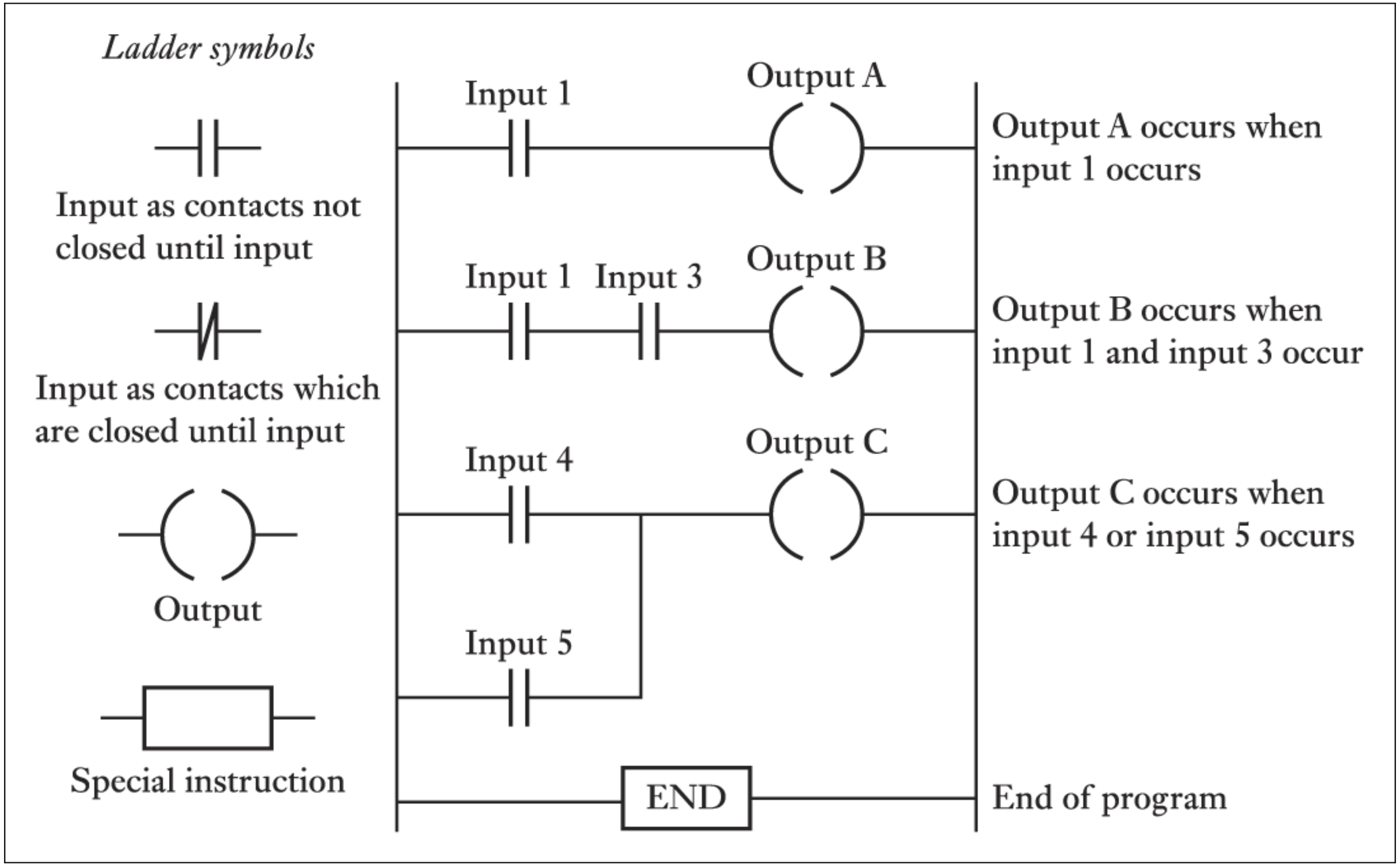


Ladder programming

Fig.21.5 shows a basic standared symbols that are used and rung,

Inputs must always preceede outputs and there must be at least one output on each line

Each rung must start with an input or series of inputs and end with an output



Ladder symbols
Input as contacts not closed until input
Input as contacts which are closed until input
Output
Special instruction
Input 1
Output A
Output A occurs when input 1 occurs
Input 1
Input 3
Output B
Output B occurs when input 1 and input 3 occur
Input 4
Output C
Output C occurs when input 4 or input 5 occurs
Input 5
END
End of program

Ladder programming

Example of a ladder diagram

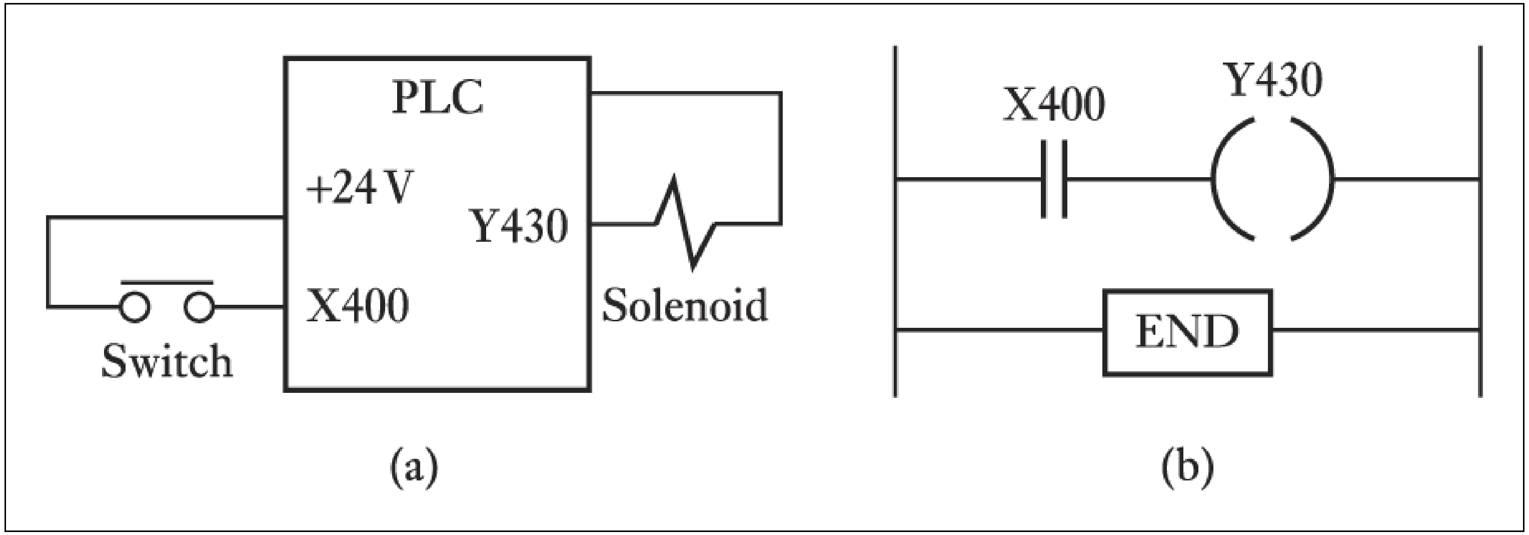


Figure 21.6 Switch controlling a solenoid

The output from the PLC is to energise a solenoid when a normally open start switch connnected to the input is activated by being closed This might be a solenoid valve which opens to allow water to enter a vessel.

Ladder programming

Example of a ladder diagram: An ON/OFF temperature control

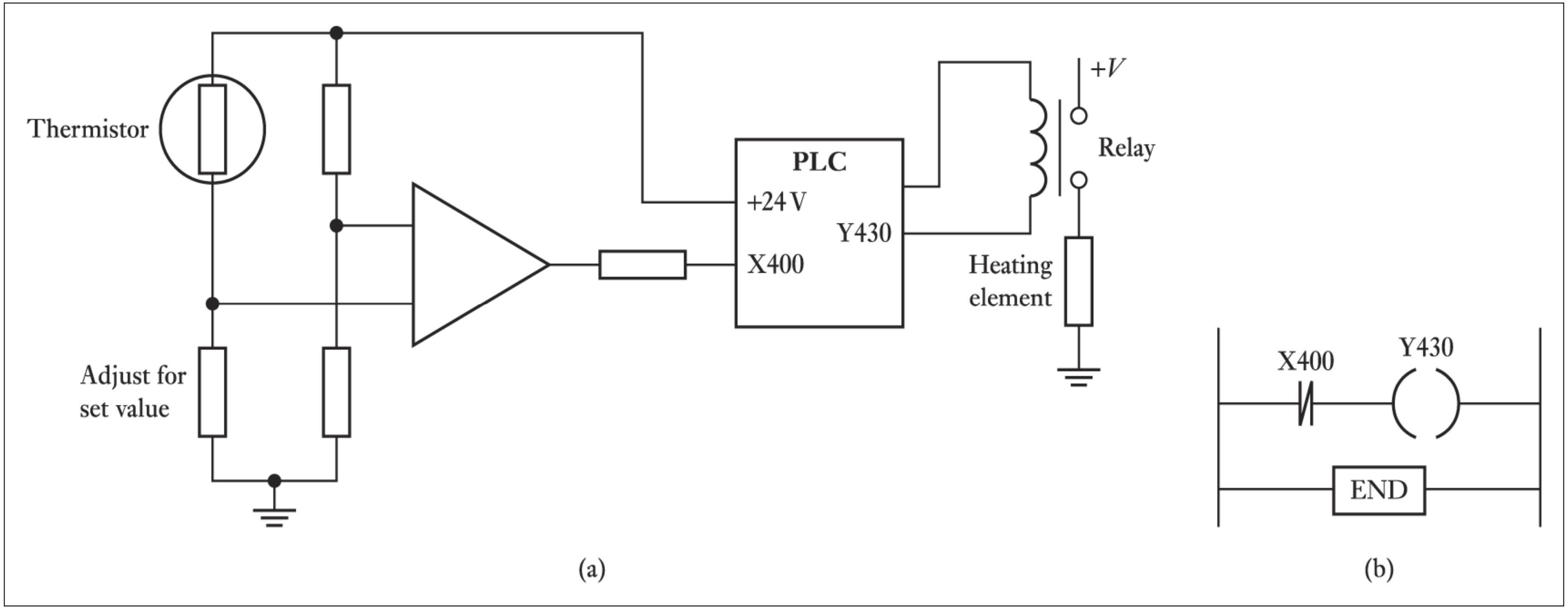


Figure 21.7 Temperature control system

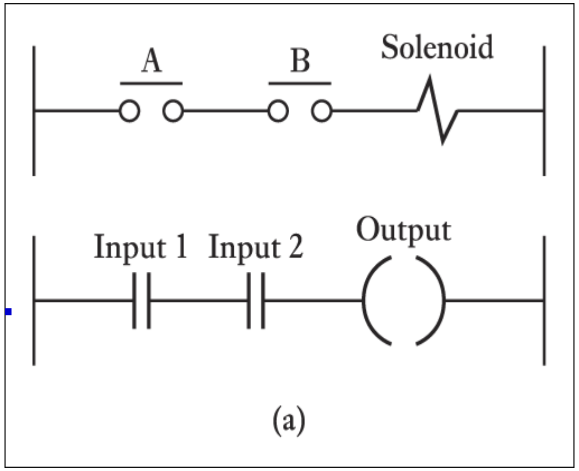
The input goes from low to high when the temparature sensor reaches the set temperature. The output is then to go from ON to OFF.

Ladder programming

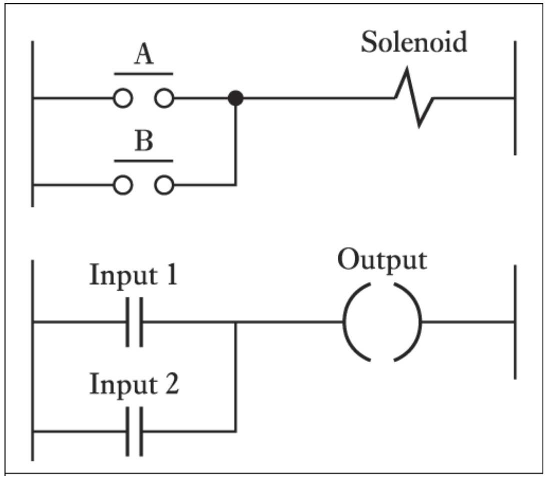
Logic Functions

The logic Functions can be obtained by combinations of switches.

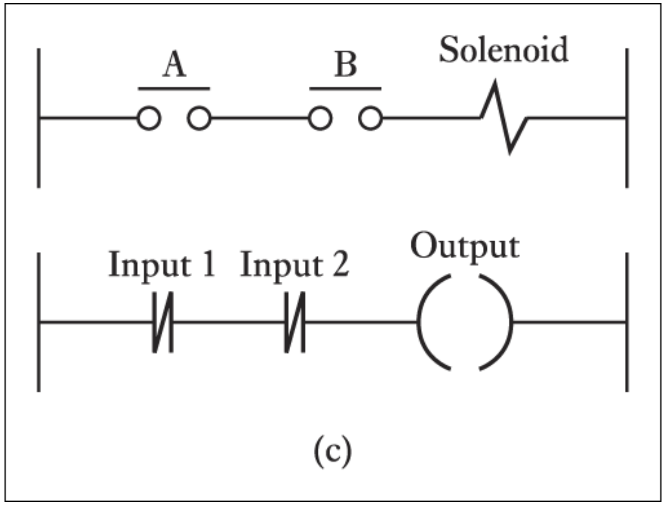
The Figures shows how ladder programs can be written for such combination



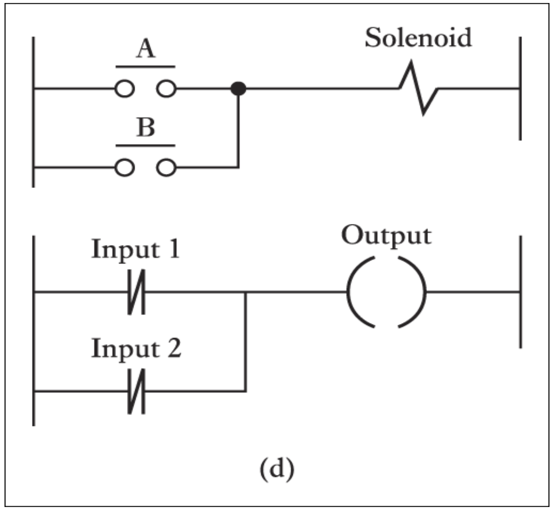
AND circuit



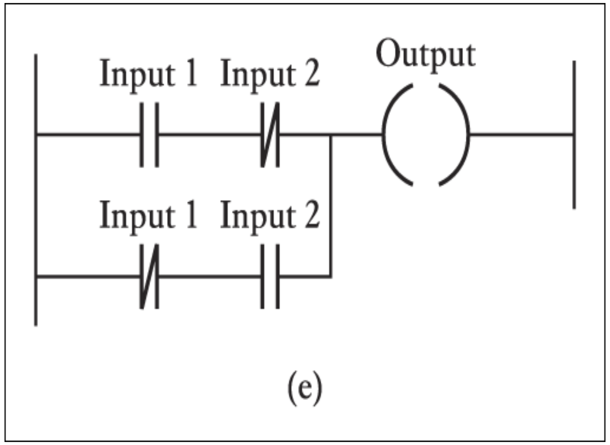
OR circuit



NOR circuit



NAND circuit



XOR circuit

Ladder programming

Logic Functions

The basic logic functions can be used to obtained more complicated combinations of switches.

Consider a situation where a normally open switch A must be activated and either of two other, normally open switches B and C must be activated for a coil to be energised.

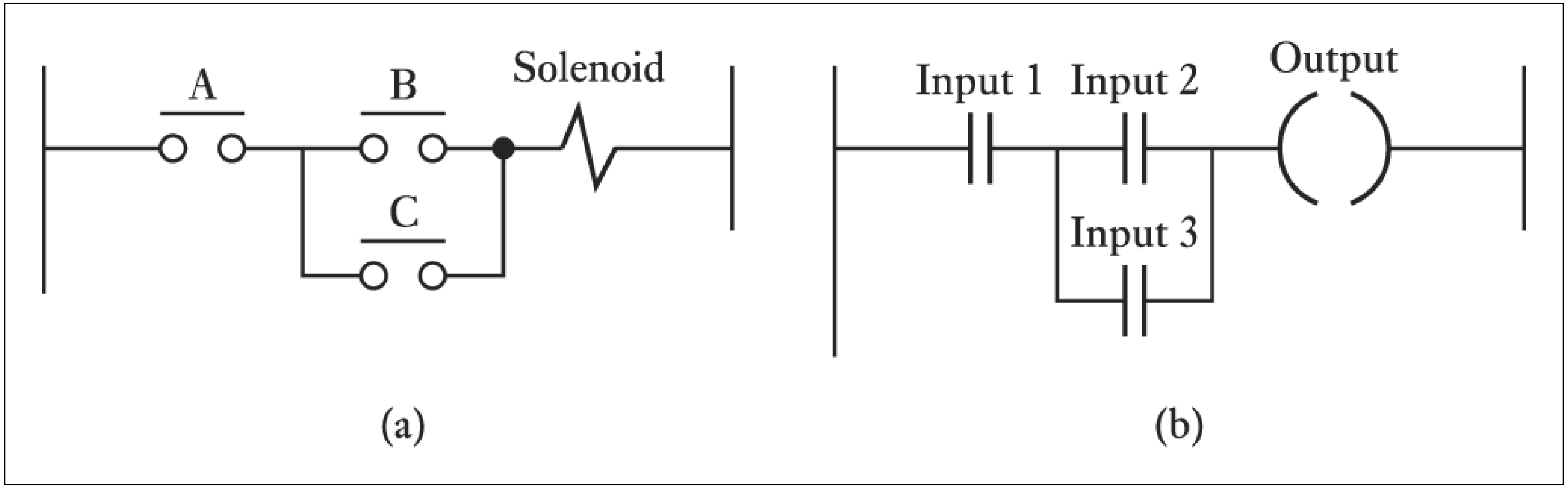


Figure 21.9 Switches controlling a solenoid

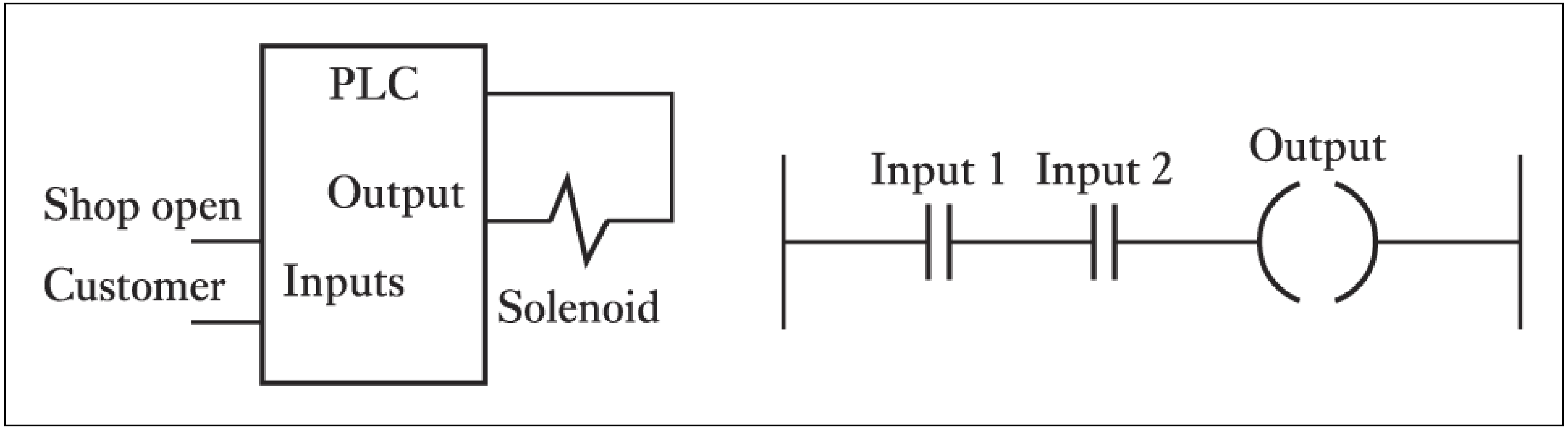


Figure 21.10 Shop door system

PLC programming: Instruction List

- Each horizontal rung on the ladder represents a line in the program and the entire ladder gives the complete program in the ladder language.
- Using a graphic interface, a programmer can build his program, then translate these symbols into machine language that can be stored in the PLC memory.
- Alternatively, the ladder program can be translated into an instruction list and entered into the programming panel or computer.

PLC programming: Instruction List

- Instruction lists consist of a series of instruction with each instruction being on a separate line.
- An instruction consists of an

operator followed by one or more operand

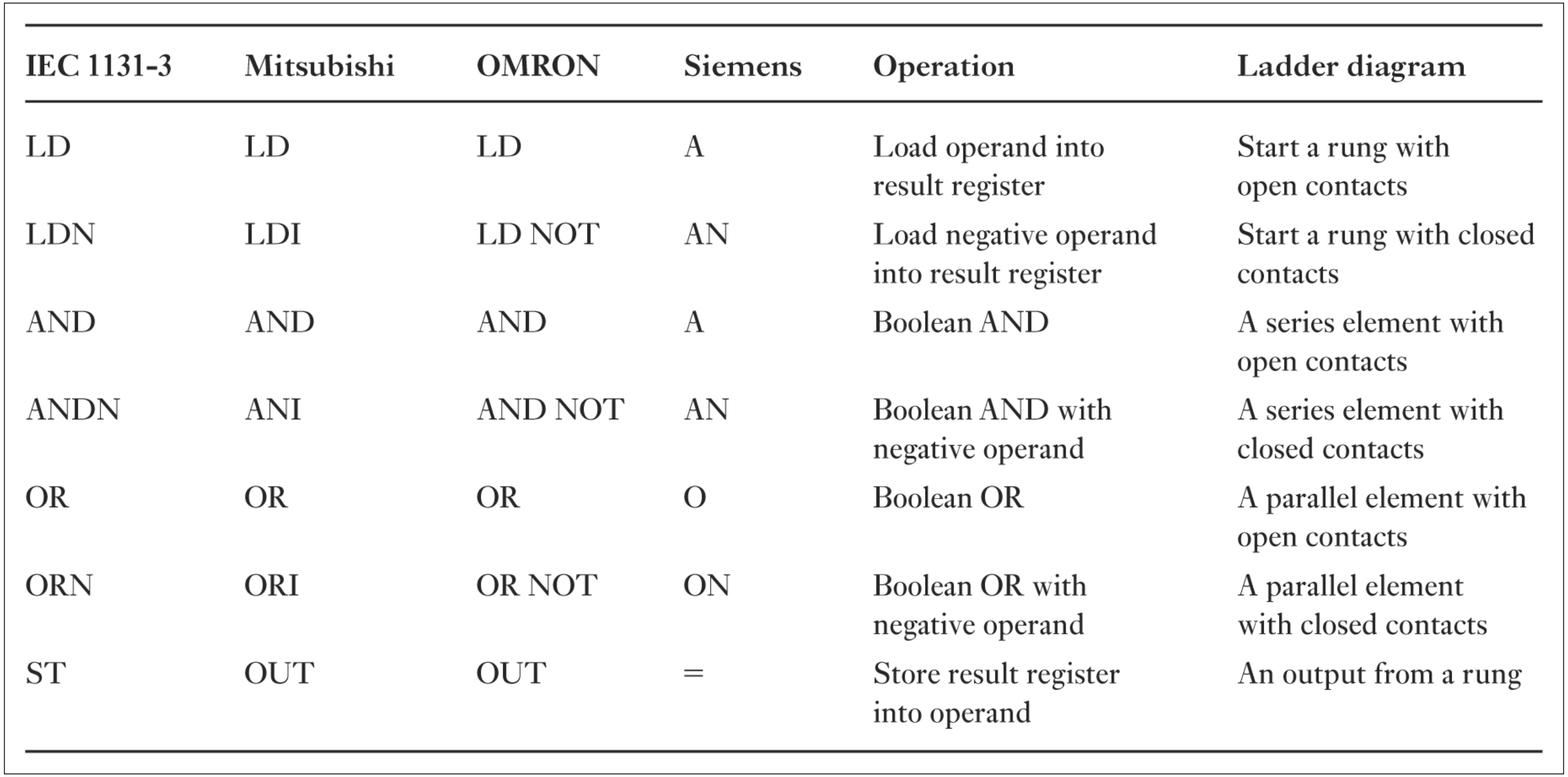
- ..e.g. LD A (*load input A*)

Comments

Instruction List

The mnemonics codes used by different PLC manufactures differ but an international standard (IEC 1131-3) has been proposed and is widely used

Table below shows core mnemonics. For the rest of the following instructions, Mitsubishi mnemonics will be used

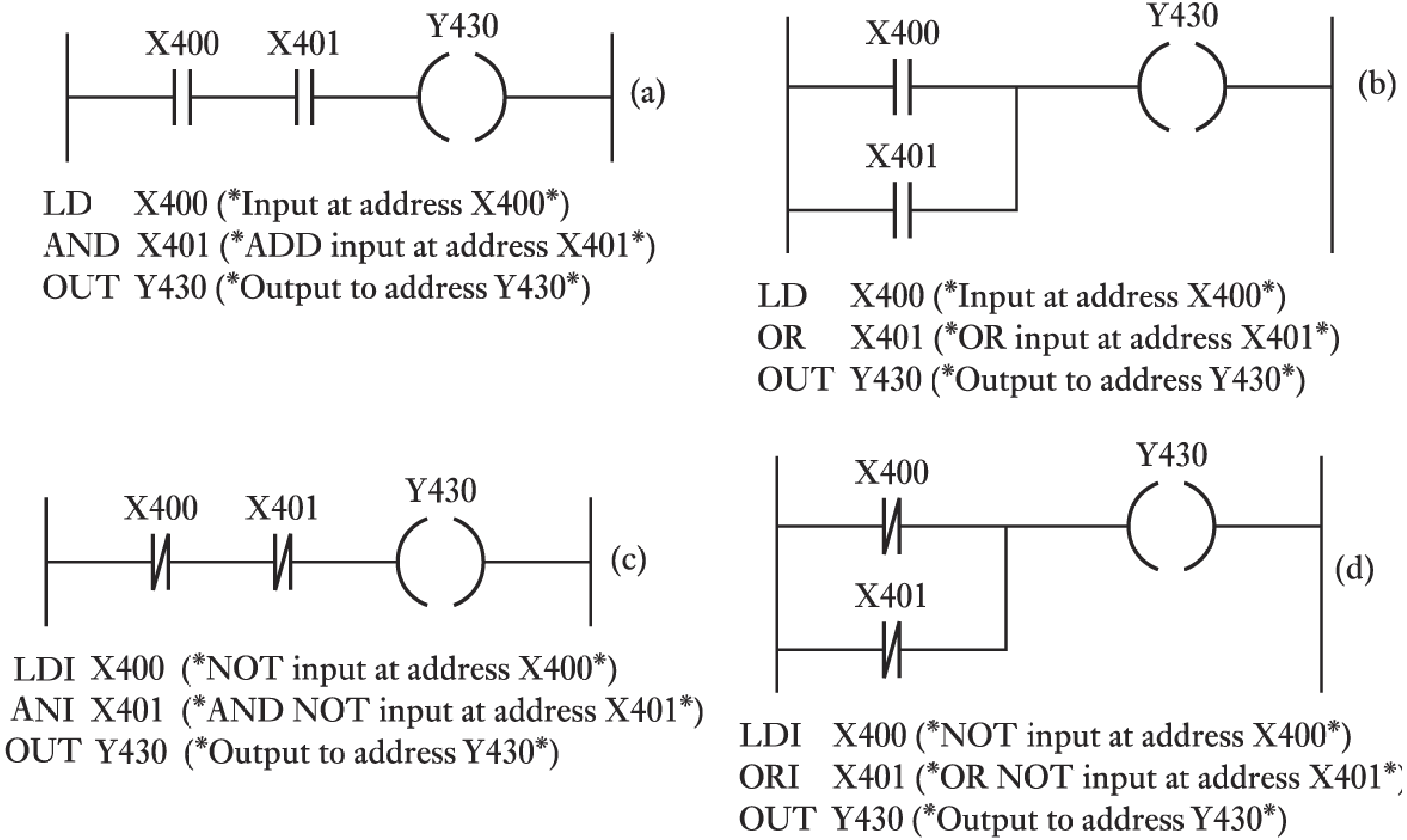


IEC 1131-3	Mitsubishi	OMRON	Siemens	Operation	Ladder diagram
LD	LD	LD	A	Load operand into result register	Start a rung with open contacts
LDN	LDI	LD NOT	AN	Load negative operand into result register	Start a rung with closed contacts
AND	AND	AND	A	Boolean AND	A series element with open contacts
ANDN	ANI	AND NOT	AN	Boolean AND with negative operand	A series element with closed contacts
OR	OR	OR	O	Boolean OR	A parallel element with open contacts
ORN	ORI	OR NOT	ON	Boolean OR with negative operand	A parallel element with closed contacts
ST	OUT	OUT	=	Store result register into operand	An output from a rung

Table 21.1 Some Instruction code mnemonics

Instruction List and Logic Function

Figures show how individual rungs on a ladder are entered using the Mitsubishi mnemonics where logic functions are involved



(a)

```
LD     X400 (*Input at address X400*)
AND  X401 (*ADD input at address X401*)
OUT  Y430 (*Output to address Y430*)
```

(b)

```
LD      X400 (*Input at address X400*)
OR      X401 (*OR input at address X401*)
OUT  Y430 (*Output to address Y430*)
```

(c)

```
LDI  X400  (*NOT input at address X400*)
ANI  X401  (*AND NOT input at address X401*)
OUT  Y430  (*Output to address Y430*)
```

(d)

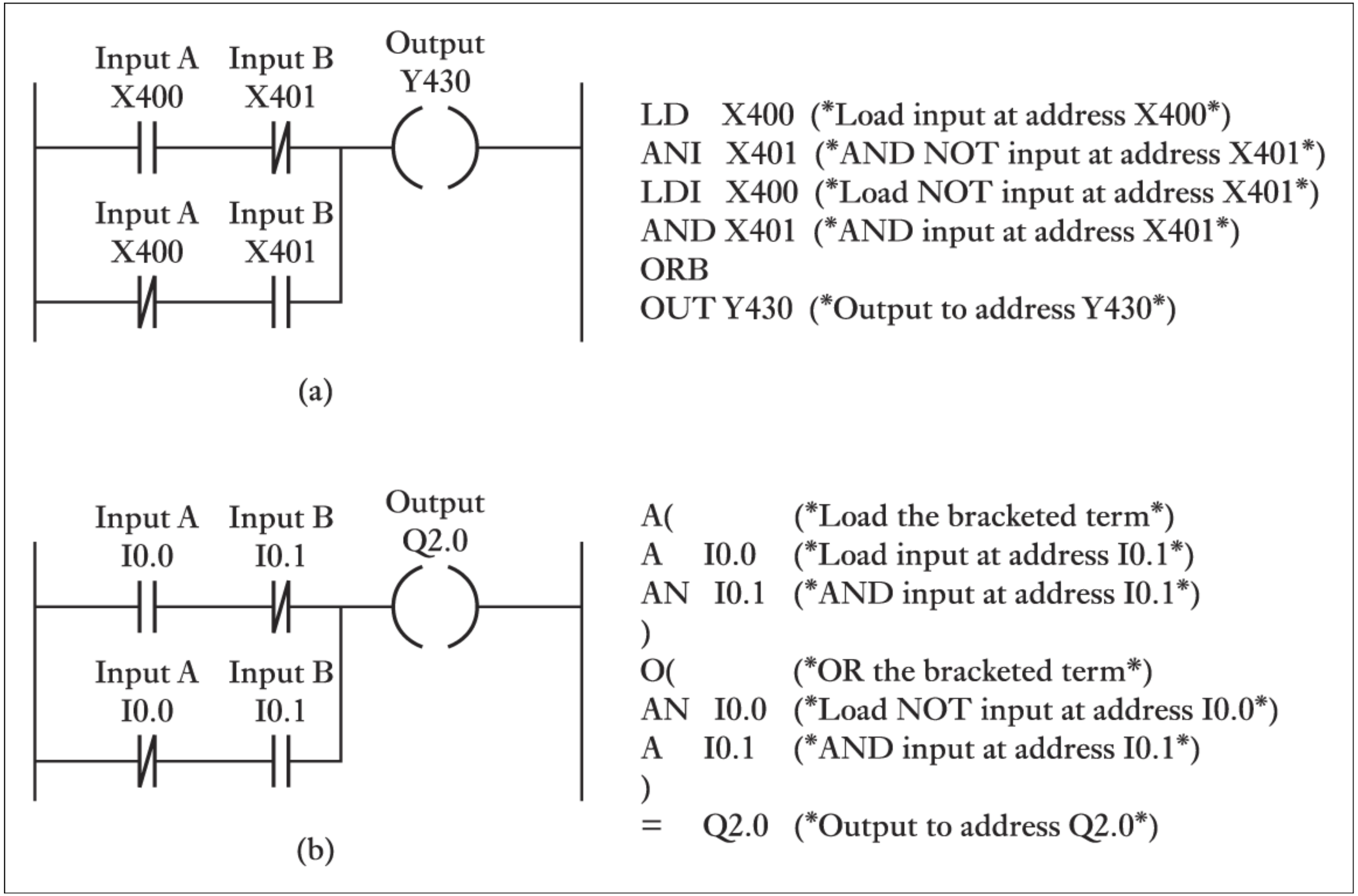
```
LDI    X400 (*NOT input at address X400*)
ORI    X401 (*OR NOT input at address X401*)
OUT  Y430 (*Output to address Y430*)
```

Figure 21.11 (a) AND, (b) OR, (c) NOR, (d) NAND

Instruction List and Branching

When two parallel arms are involved, Mitsubishi treats the situation by using an ORB instruction to indicate OR together parallel branches as shown in Fig.21.12a. Line 3 describe a new line since it starts with LD/LDI instruction

while Siemens use brackets as shown in Fig.21.12b,



```
LD   X400  (*Load input at address X400*)
ANI  X401  (*AND NOT input at address X401*)
LDI  X400  (*Load NOT input at address X401*)
AND X401   (*AND input at address X401*)
ORB
OUT Y430   (*Output to address Y430*)
```

```
A(          (*Load the bracketed term*)
A   I0.0    (*Load input at address I0.1*)
AN  I0.1    (*AND input at address I0.1*)
)
O(          (*OR the bracketed term*)
AN  I0.0    (*Load NOT input at address I0.0*)
A   I0.1    (*AND input at address I0.1*)
)
=   Q2.0    (*Output to address Q2.0*)
```

Figure 21.12

XOR

Latching and internal relays

The term latching is used for the circuit that able to hold the output energized even though the input which energizing it ceases. So the output remember its last state.

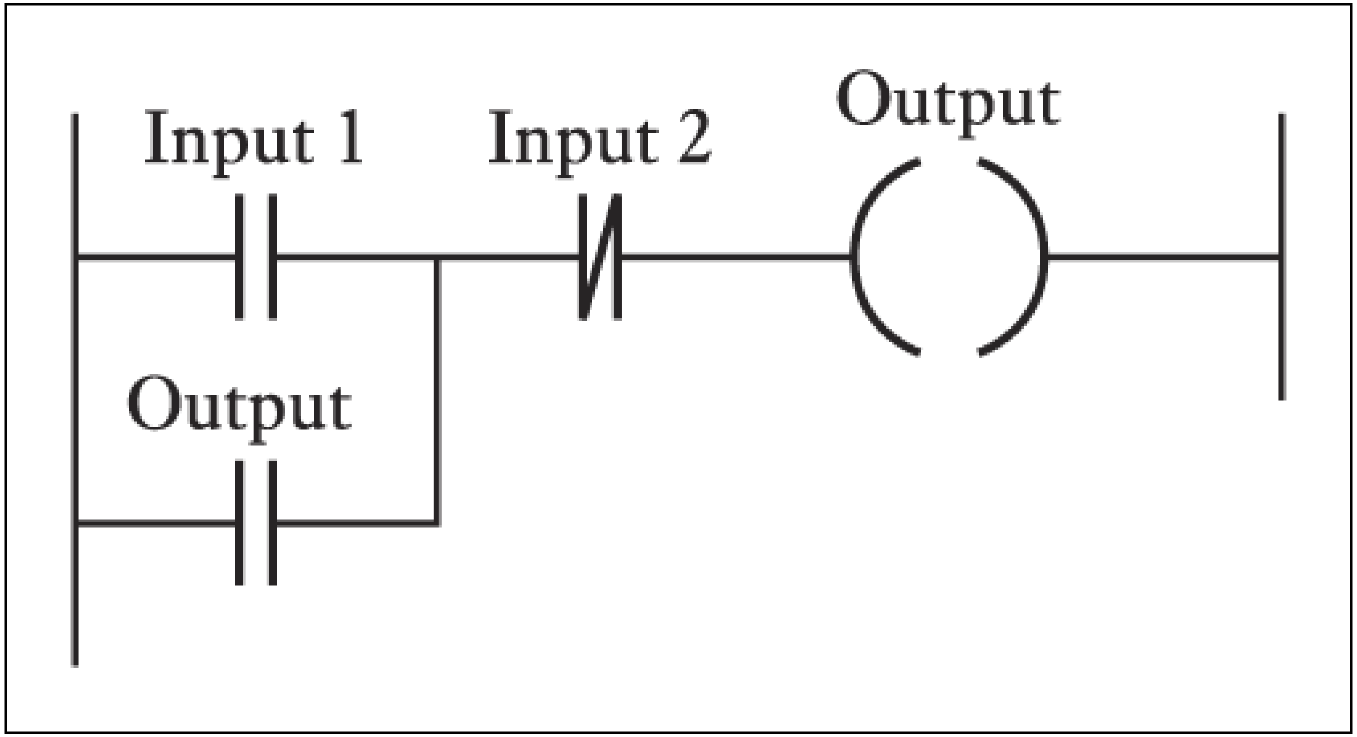


Figure 21.13 A latch circuit

Latching: Examples

It is required for the PLC to control a motor so that when the start signal button is momentarily pressed the motor starts and when the stop button is pressed, the motor switches OFF

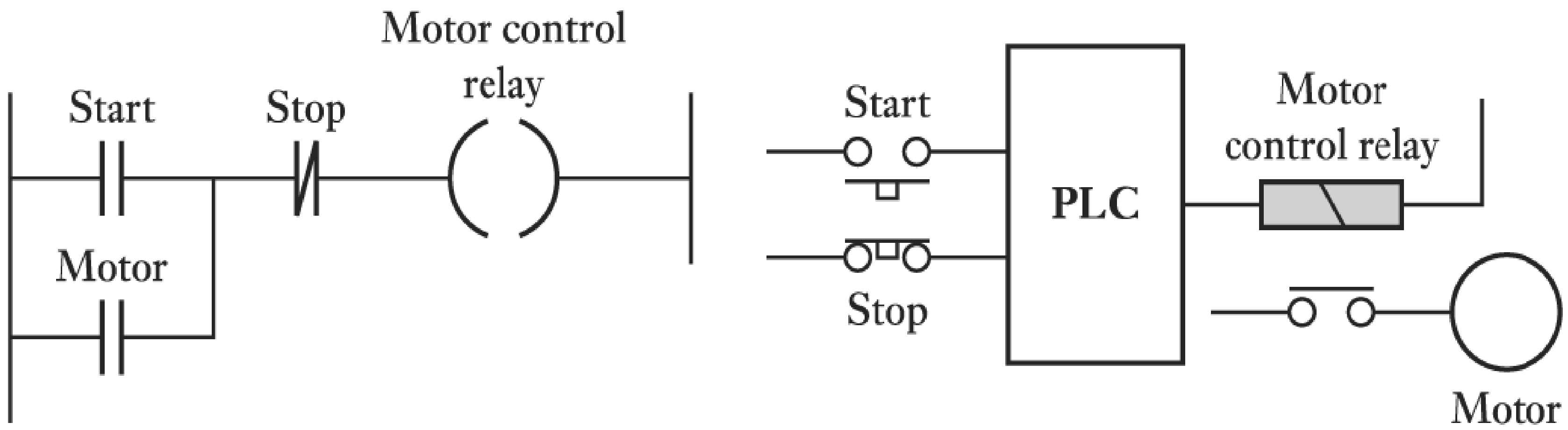


Figure 21.14 Stop system

Internal relays

- The term internal, auxiliary relay or marker is used for what can be considered as internal relay in PLC. It behaves like relays with their associated contacts, but in reality are not actual relays but simulation by the software of the PLC.
- Internal can be very useful aids in the implementation of switching sequences.
- They are often used when there are programs with multiple input conditions.

Internal relays: examples

They are used when there are programs with multiple input conditions. In Fig.21.15-a different input arrangement have been implemented by internal Relay

Multiple out can also started with internal relays as shown in Fig.21.15-b

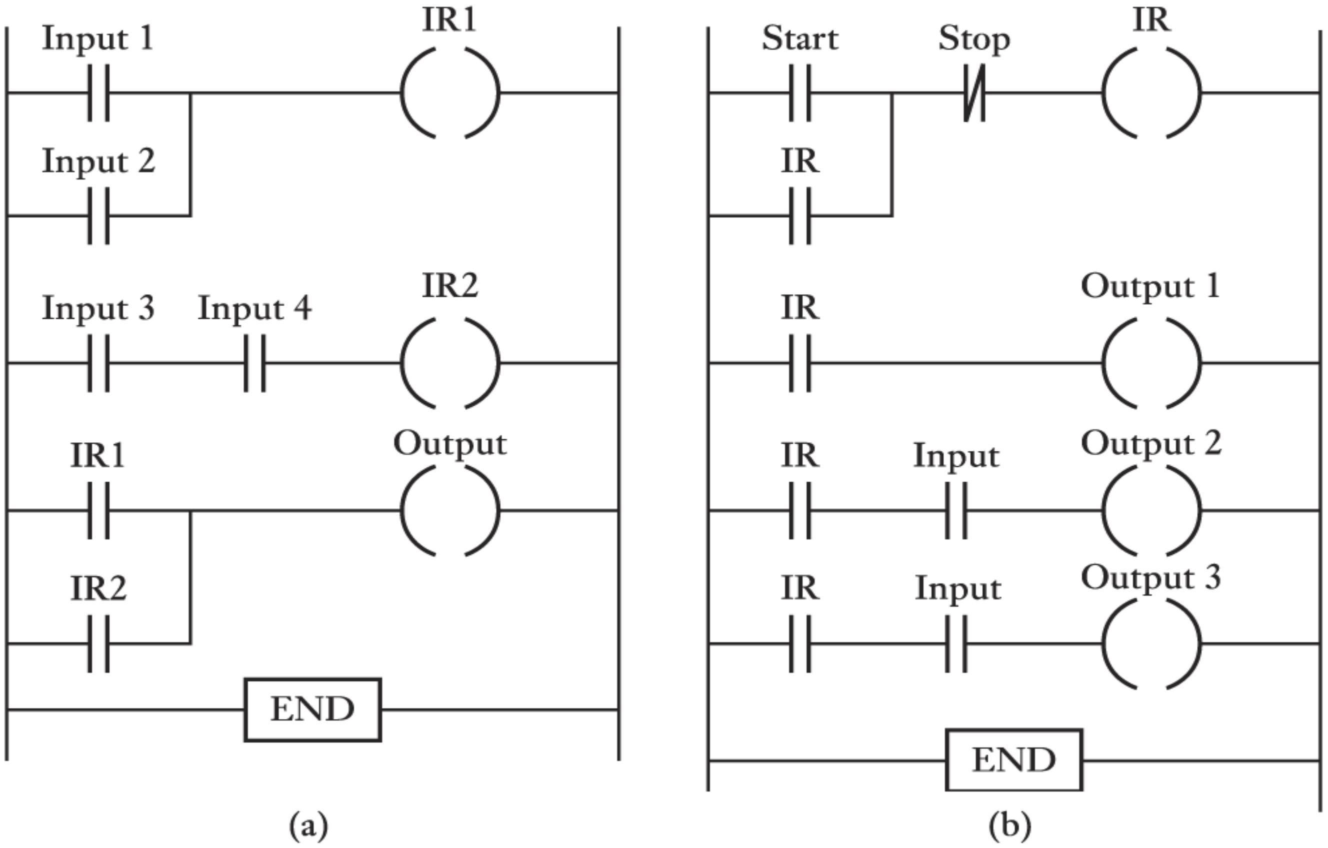


Figure 21.15 (a) An output controlled by two input arrangements, (b) starting of multiple outputs

Internal relays: examples

Can be used to reset a latch contact as shown below

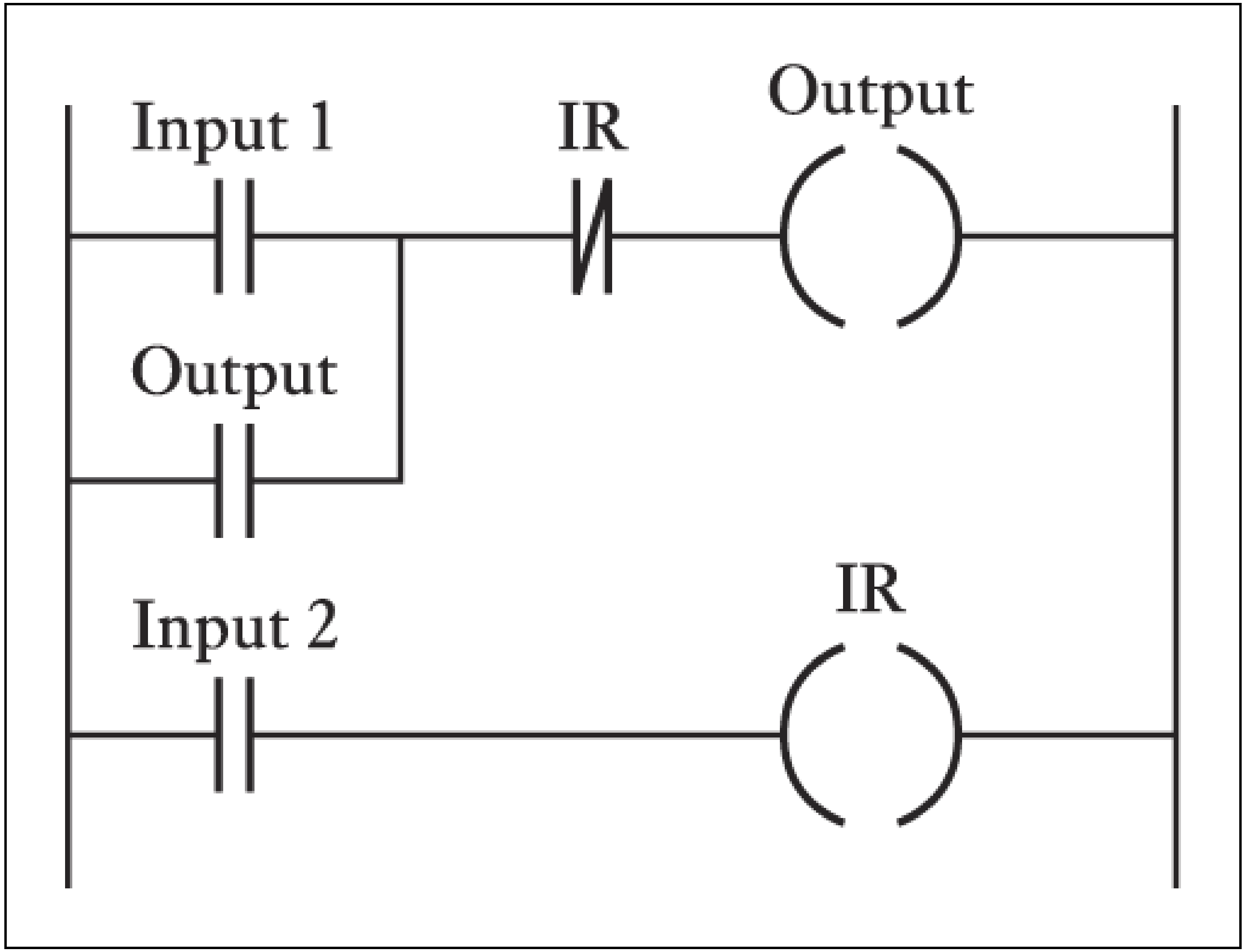


Figure 21.16 Resetting a latch

Internal relays

When the contact of input is closed, the coil battery is energized, this closes the internal relay contacts and so even if contact of the input open as result of power failure, the internal relay contact remain closed. This means that the output controlled by the internal relay remains energized

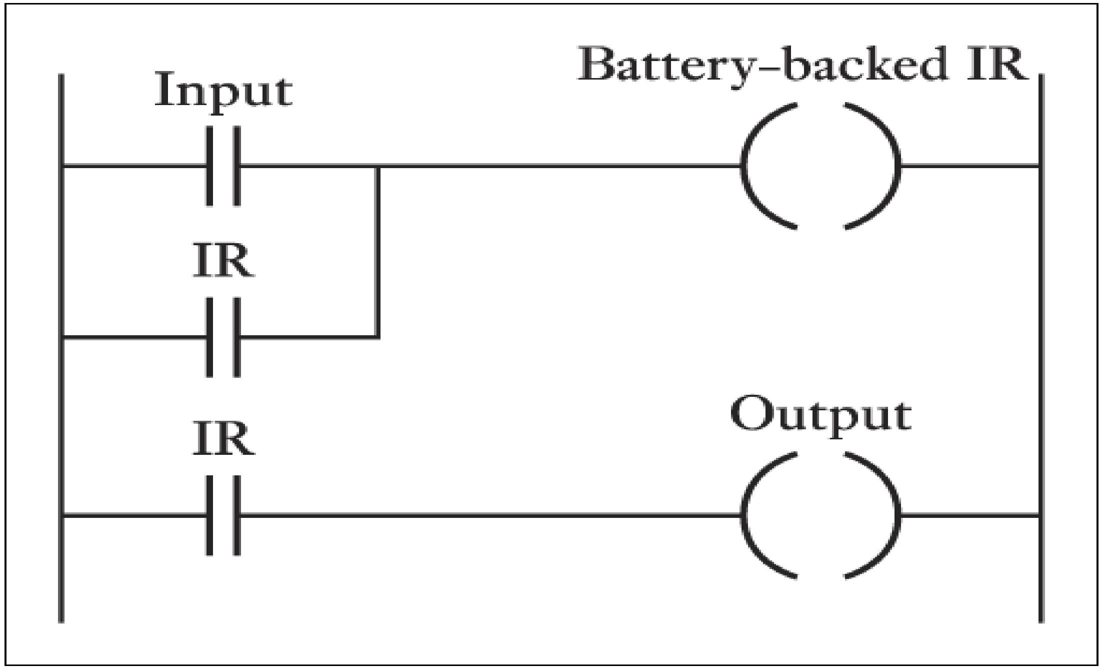


Figure 21.17 Use of a battery-backed internal relay

Data Handling

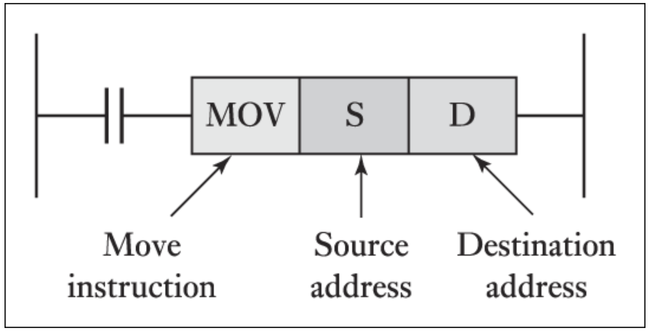
- The operations that may be carried out with a PLC on data words include:
- 1- Moving data
- 2- Comparison of magnitude of data
- 3- Arithmetic operations
- 4- Conversion between number system

Data instructions require memory addresses, so data registers are used to stored binary words (8 or 16 bits) and is given an address such as D0, D1, D2…

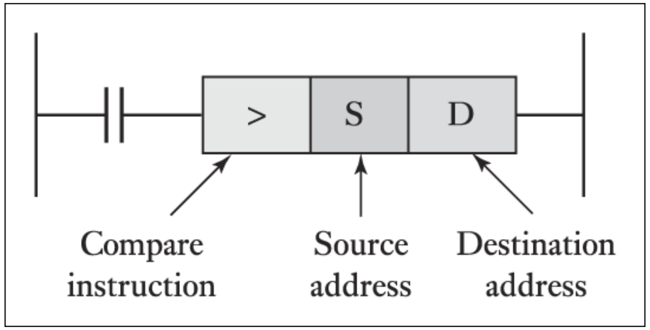
Each instruction has to specify the form of the operation, the source of the data used in terms of its data register and the destination data register of the data

Data Handling: Examples

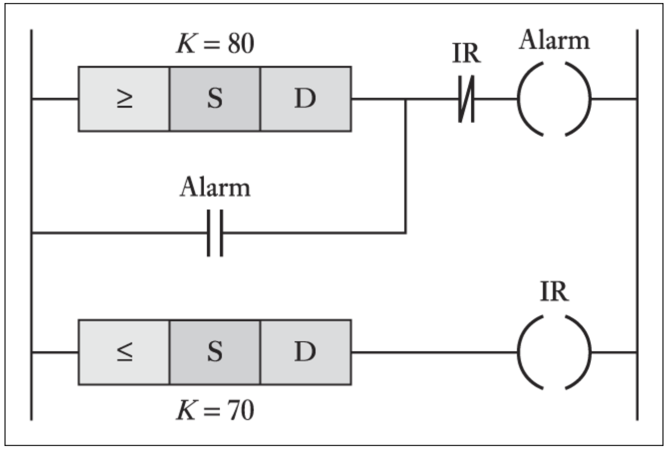
Data Movement



Data Comparison



Temperature alarm example:



Data Handling: Examples

Arithmetic operation

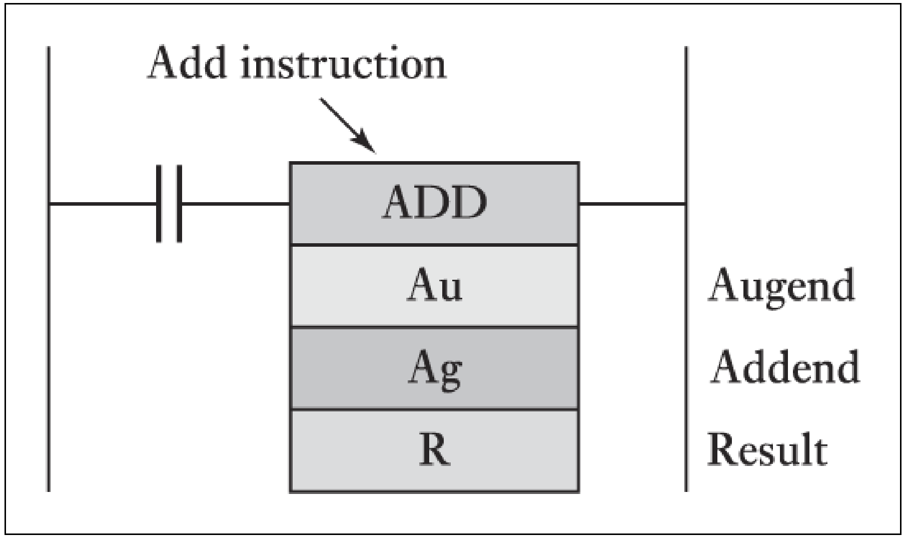


Figure 21.33 Add data

Data Handling: Examples

Code Conversion

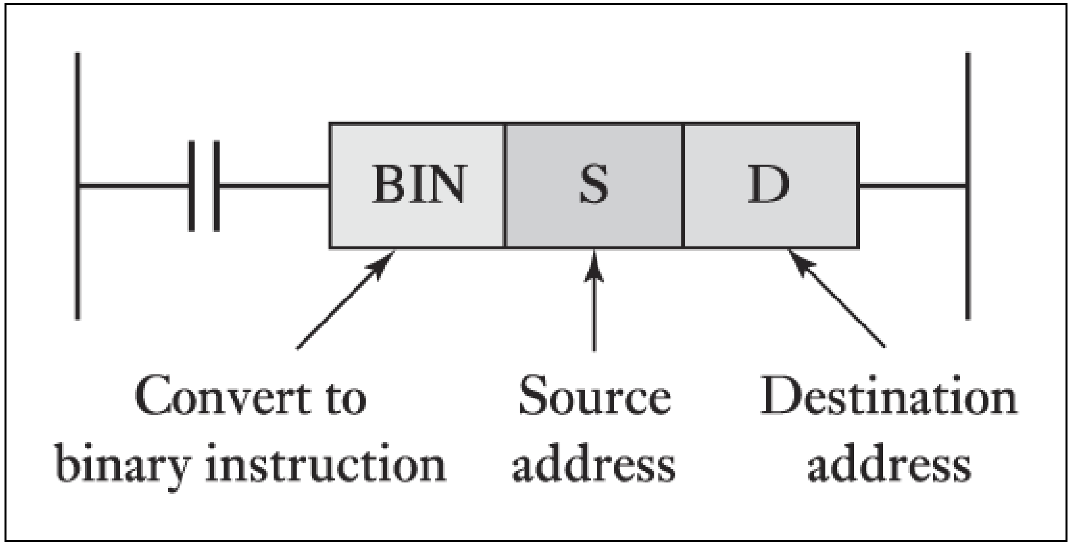


Figure 21.34 BCD to binary