| UNIT 22: | PROGRAMMABLE LOGIC CONTROLLERS |
|---|---|

**Unit code: A/601/1625 QCF level: 4 Credit value: 15**

# TUTORIAL – OUTCOME 2 Part 3

This work covers part of outcome 2 of the Edexcel standard module. The material is quite suitable for anyone wishing to study this interesting subject. This tutorial requires basic mathematical skills and a reasonable knowledge of digital electronic terminology. An industrial background will also be of great benefit to students. Obviously, access to suitable computer software such as Pneusim Pro™ or Bytronics™ simulation software will be a great help.

## SYLLABUS

**2 Understand PLC information and communication techniques**

*Forms of signal:* analogue (0-10 V dc. 4-20 digital

*Digital resolution and relationships:* 9-bit; 10-bit 12-bit

*Number systems:* decimal; binary; octal; hexadecimal; Binary-Coded Decimal (BCD)

*Evaluate communication standards:* comparison of typical protocols used in signal communication

*Evaluate networking methods and standards:* master to slave; peer to peer; ISO; IEE; MAP

*Logic functions:* writing programmes using logic functions based on relay ladder logic (AND; OR; EXCLUSIVE OR; NAND; NOR)

It appears that this section is not assessed for a pass grade.

| **Learning outcomes**<br>**On successful completion of this unit a learner will:** | **Assessment criteria for pass**<br>**The learner can:** |
|---|---|
| L02 Understand PLC information and communication techniques | 2.1 evaluate the different forms of signal used in programmable logic control<br>2.2 describe the resolution and relationship between analogue inputs and outputs and word length<br>2.3 express numbers using different number systems<br>2.4 describe typical protocols used in signal communication and evaluate networking methods and networking standards |

## CONTENTS

## 1. INTRODUCTION

Programming a PLC is helped by various aids such as LOGIC CIRCUITS, LADDER DIAGRAMS, TRUTH TABLES and BOOLEAN EXPRESSIONS. They are all linked and the following shows how they are related. We will consider the main logical functions and explain them in terms of each.

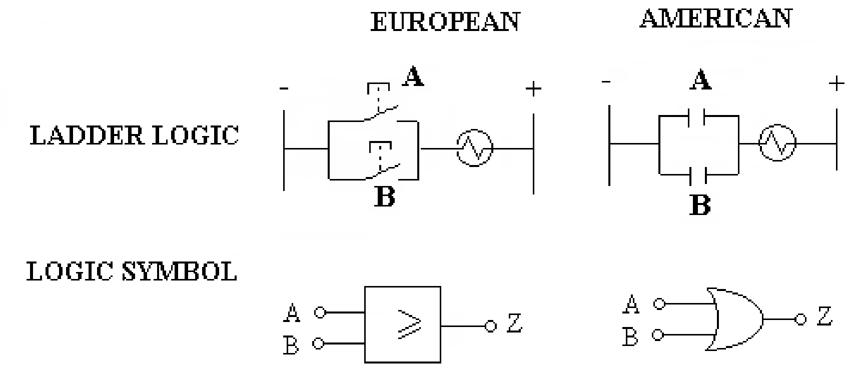
## 2. LOGIC FUNCTIONS

### OR FUNCTION

Suppose a light may be switched on by pressing switch A or B. This may be achieved in a hard wired circuit by placing two normally open switches in parallel so that when either A or B is closed, current flows in the circuit from plus to minus. This may be represented as **LADDER LOGIC**. The American Ladder Logic simply shows the switches as open contacts. European ladder logic symbols are more complex and indicate the type of switch as well (see symbols overleaf).

The same function may be made with a hardware item called a logic gate. Here we are considering electronic gates but pneumatic versions perform the same function with air instead of electricity. The European and American symbol is shown below. In general the output of the gate is labelled Z and this become live (high) when either A or B is made live (goes high).

The **TRUTH TABLE** is a way of showing the logic function. We indicate that a terminal is high (on or live) with a 1 and low (off or dead) with a 0. The table shows that the output is only low when both A and B are low.




| A | B | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

TRUTH TABLE

Figure 1

Yet another tool to help us understand these things is BOOLEAN ALGEBRA. Basically this is a method of turning words into symbols so when we say output Z is on when either line A or B is on or A and B is on together. We write:

$Z = A + B + A.B$ The plus sign means OR and the dot means AND

Boolean Algebra can be manipulated according to certain rules to reduce complex expressions to simpler expression. This is not covered here.

### INVERSION

An O added to any input terminal inverts the signal (i.e. it is a not gate). The following gate produces a Boolean statement of $Z = \overline{A}.B$

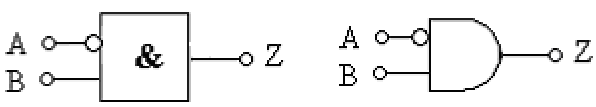
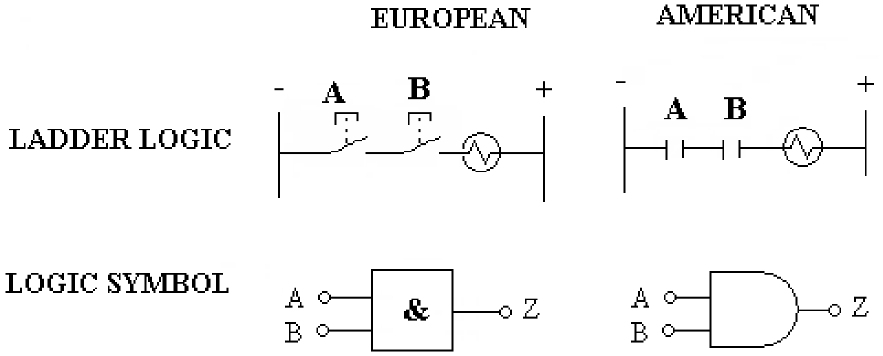



Figure 2

## AND FUNCTION

The output Z is turned on when input A and B is turned on. The hard wire circuit would be two switches in series and this is shown as ladder logic. The Boolean statement is Z = A.B meaning Z is high only when A AND B are high.



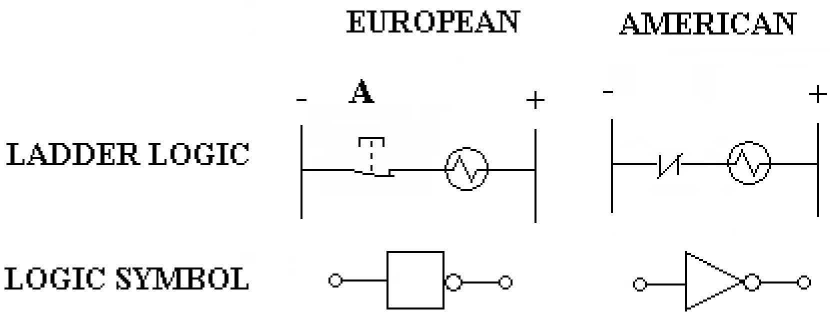

| A | B | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

TRUTH TABLE

Figure 3

## NOT FUNCTION

This is a function equivalent to a normally closed switch. The light is normally on but when the switch is pressed the light goes out. In logic terms the output is on when the input is off and off when the input is on. The Boolean statement is either Z = $\overline{A}$ or $\overline{Z}$ = A The over score indicates low or off.




| A | Z |
|---|---|
| 0 | 1 |
| 1 | 0 |

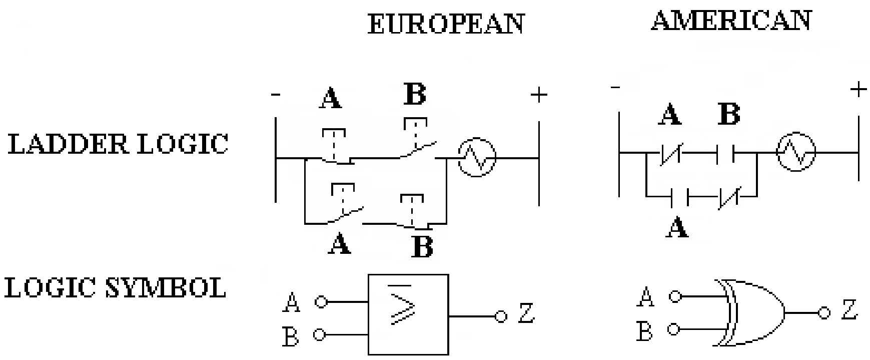
TRUTH TABLE

Figure 4

## EXCLUSIVE OR (XOR)

The output is on when A or B is on but not when both are on. The hard wire circuit requires each switch to have a normally open and a normally closed contact. The Boolean statement is:

Z = A + B which means Z is on when A is on and B is off or when A is off and B is on.




| A | B | Z |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

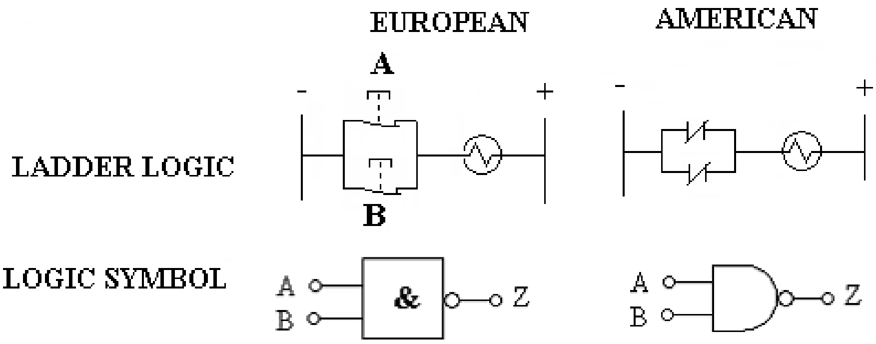
TRUTH TABLE

Figure 4

## NAND FUNCTION

This is the opposite of an and. The equivalent hard circuit uses two normally closed switches in parallel so the light is always on except when both are pressed. The Boolean statement is:

$\overline{Z} = A.B$ meaning Z is Off when A and B are both on.




| A | B | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

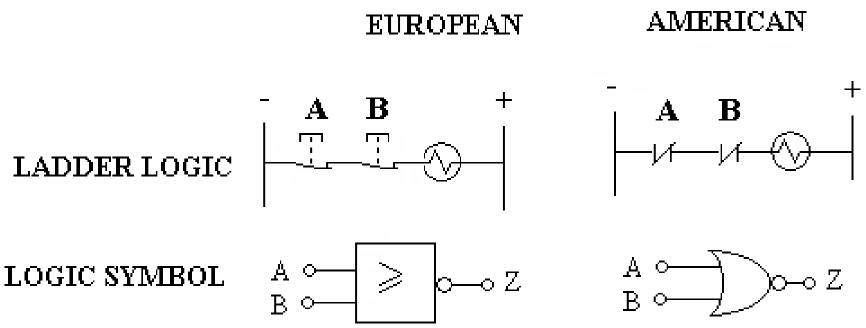
TRUTH TABLE

Figure 5

## NOR FUNCTION

This is the reverse of the OR gate and it can be created by two normally closed switches in series. The output is only on when both switches are not operated (off). The Boolean statement is:

$$Z = \overline{A} + \overline{B}$$




| A | B | Z |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

TRUTH TABLE

Figure 6

## MULTIPLE INPUTS

Logic gates may have as many inputs as you like. This one has three inputs so the output Z is turned on when A and B and C is turned on.

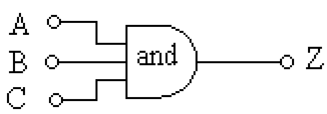
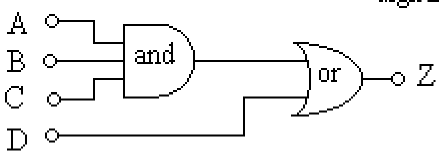



Figure 7

## BUILDING A CIRCUIT

A more complex logic circuit is shown below with four inputs. The truth table is shown with only the conditions for turning the output on. The pattern of the inputs can be interpreted as a digital number.




| digital value | 8 | 4 | 2 | 1 | |
|---|---|---|---|---|---|
| | A | B | C | D | Z |
| | 1 | 1 | 1 | 0 | on |
| | 0 | 0 | 0 | 1 | on |
| | 1 | 1 | 1 | 1 | on |

Figure 8

There are three possible combinations that can turn the output on. The three digital numbers that will turn the output on are hence 14, 1 and 15. (With A forming the most significant bit). If A, B, C and D were connected to a computer data bus then the computer could switch the output on when it recognises these numbers.

## 4. INTRODUCTION TO LADDER DIAGRAMS

In the previous section you have seen how a logic circuit can be made with hardware. A PLC programme simulates a hardware circuit and produces the same result. The input and output elements are real items but the way they are connected is purely simulation. In addition, imaginary hardware items such as counters and timers may be added to produce the desired results. Programming is covered in more detail in outcome 3. The following is an introduction to the subject.

Basically, to switch something on, a circuit has to be made continuous between the plus and minus power buses of the system. When a circuit is laid out in this way it resembles the rung of a ladder, hence the name ladder diagram. Consider the simple case of turning on a motor by pressing push button switch.

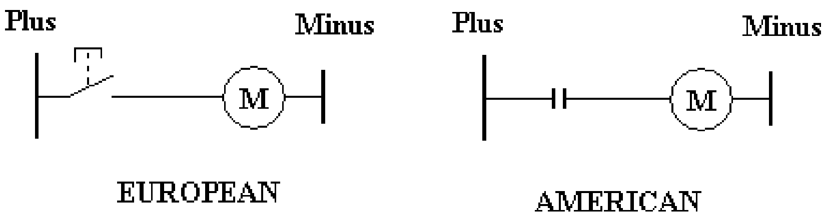



Figure 9

Things can be switched on by sensor switches such as a position sensor as well as manual switches. These are input elements. Things that can be switched on (motors, relays, lamps etc) are called output elements. The European symbols show more information about the type of element and is more widely used for real hardware circuits. As this is of no importance in programming a PLC the American system is widely adopted for PLC work. The next diagram shows a solenoid operated directly by a sensor with no relay.

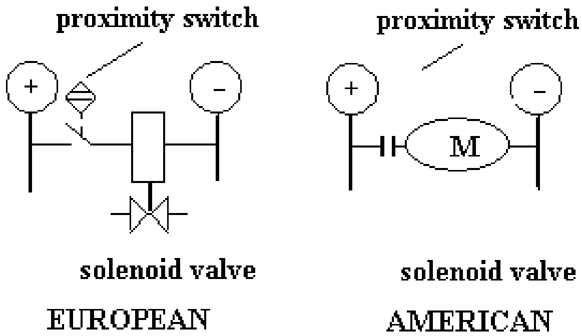



Figure 10

This next diagram shows a solenoid valve operated by a relay. The relay is operated by the position sensor. The sensor contacts may be normally open or normally closed. Note the need to use identifiers such as X for input components and Y for outputs.

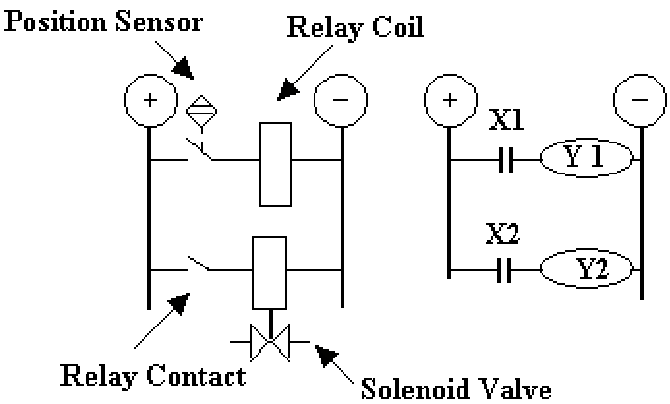



Figure 11

A ladder diagram may have many circuits connected between the power busses and hence the diagram resembles a ladder with many rungs. The rungs must be numbered. The same elements may appear in several rungs. Logic functions such as AND, OR, NAND and so on are created by using parallel and serial branches. Latches may be created and outputs can be put in parallel. The words shown are usually changed to simple code called MNEMONICS. This is covered in outcome 3. Here is a typical example.

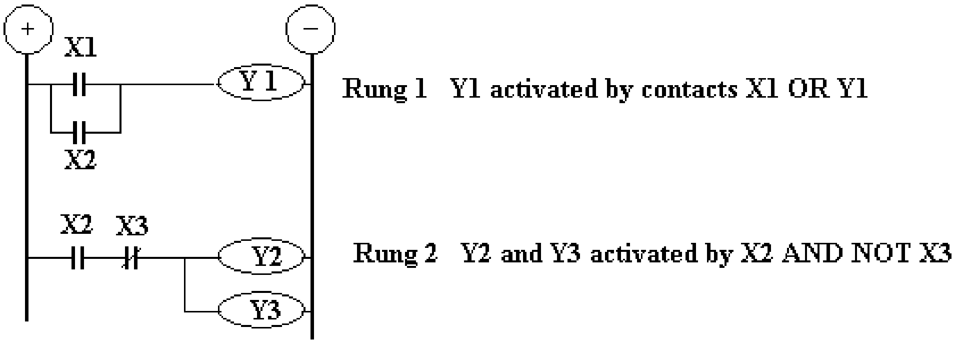



Figure 12

The following symbols show various switches and sensors.

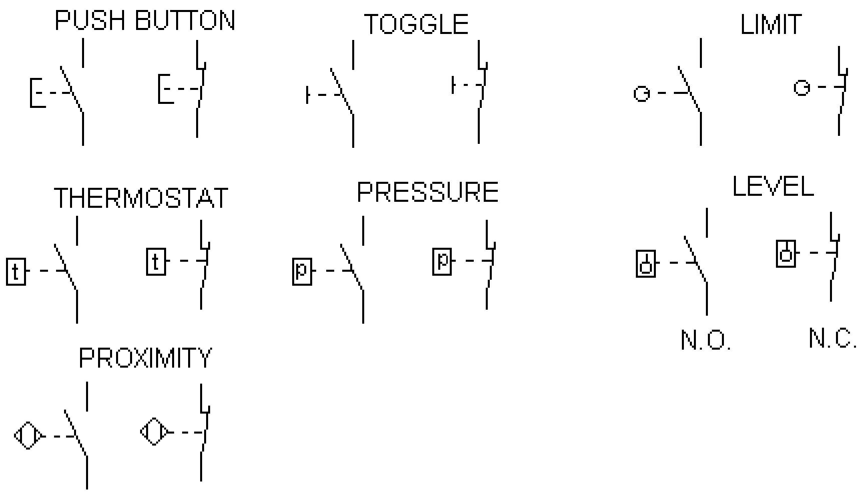
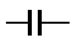
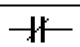
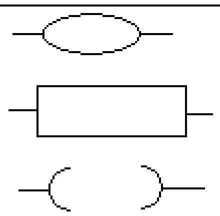



Figure 13

## AMERICAN LADDER LOGIC SYMBOLS

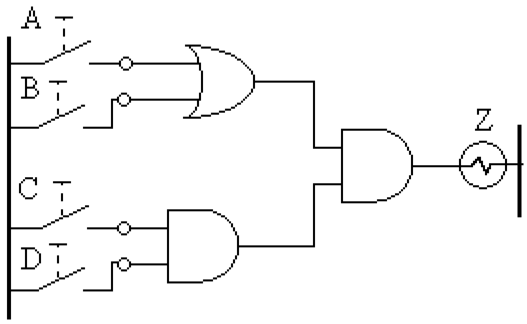
American symbols seem to be widely used and they do not indicate the form of input and output devices. They simply show the contacts as normally open or normally closed. Output devices are all the same.

| SYMBOL | DESCRIPTION |
|---|---|
|  | Normally Open Contact |
|  | Normally Closed Contact |
|  | Several ways of showing output elements |

## SELF ASSESSMENT EXERCISE No. 1

The diagram shows a lamp Z that is switched on by a combination of switches A, B, C and D. Complete the truth table and deduce the digital numbers that will turn the output on.

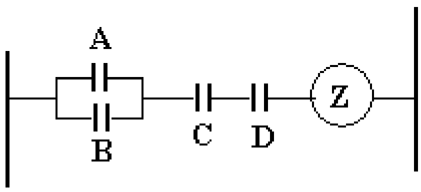
Construct a ladder logic diagram for the circuit.




| 8 | 4 | 2 | 1 | | digital value |
|---|---|---|---|---|---|
| A | B | C | D | Z | |
| | | | | on | |
| | | | | on | |
| | | | | on | |

Figure 14

## SOLUTION




| 8 | 4 | 2 | 1 | | digital value |
|---|---|---|---|---|---|
| A | B | C | D | Z | |
| **0** | **1** | **1** | **1** | on | **4+2+1=7** |
| **1** | **0** | **1** | **1** | on | **8+2+1=11** |
| **1** | **1** | **1** | **1** | on | **8+4+2+1=15** |

Figure 15