LabPLC

User's manual for LabPLC software

Projected and designed by Mirko Bjelica (mikanb@Eunet.yu)
May 2000.

I would like to express very special gratitude to Mr. Mark C. Alberts who sponsored me with the ***BASCOM-8051*** *software and helped me to create LabPLC, my final work at Polytechnic Engineering College.*

Short introduction to PLCs

(Originally taken from Panayiotis Stassinopoulos)

A typical industrial automation system, consist of three discrete parts:

Inputs, which represent the senses of the system and has to do with the facts that are taking place at the environment of the application. Those inputs could be pressure sensors, temperature sensors, velocity sensors, approach sensors, buttons, switches etc.

Outputs, with which the system affects its environment, such as relays, actuators etc.

Control unit, which is the brain of the entire automation system.

The operation of the system is realized with a combination of the above three parts. The data flow from the inputs through the control unit, in which they are processed, to the outputs.

Once, a control unit was consisted of analog circuits or digital circuits who made an excessive usage of logic components or relays. Last years, with the great development of computer science many microcomputer-based systems have started to come on the foreground. The most popular device that was designed is the PLC (Programmable Logic Controller). A PLC is a complete computer system, specialized for use with automation systems. It contains a microprocessor, ROM, RAM for storing its program and a battery to protect its data due to a low voltage condition.

The inputs and outputs of the application are connected on the PLC. First implemented PLCs, were supporting control only on digital signals, so inputs and outputs should be in digital format. Later, there were designed PLCs which could control analog signals through analog inputs and outputs.

Programming PLCs

In order to realize the required automation, a PLC should first be programmed. There have been developed three PLC programming languages.

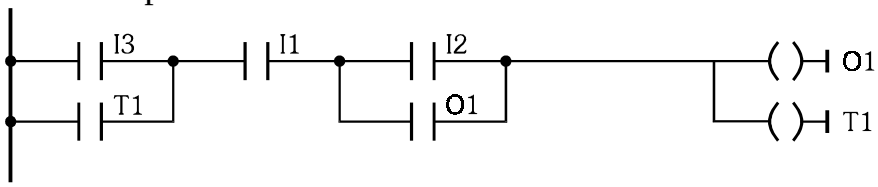
STL: It is the most popular and powerful language. It consists of instructions' list.

For example:

```
L     I3
O     T1
=     M1
L     I2
...
```

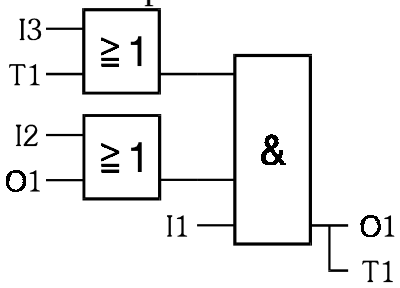
LAD: It's a graphical programming language which looks like the auxiliary circuit of a relay-based system.

For example:



CSF: It is also a graphical programming language which is the same with the circuit of the automation made of logic gates.

For example:



Although the basic programming instructions are the same, their usage may differ from one PLC to another, depending on its model and manufacturer. Modern PLC's offer advanced programming techniques, including program blocks, subroutines and functions which can be accessed from different points of the main program.

In order to import a program in a PLC we must use a special device called Programmer, or a PC that "runs" the appropriate software.

Once a program is stored in the PLC's memory, it starts executing cyclically. By saying this we mean that after the execution of the last instruction of the program, PLC's controller goes back to the beginning of the program and executes it again. Acting this way there is a continuous scanning and processing of the inputs (polling method).

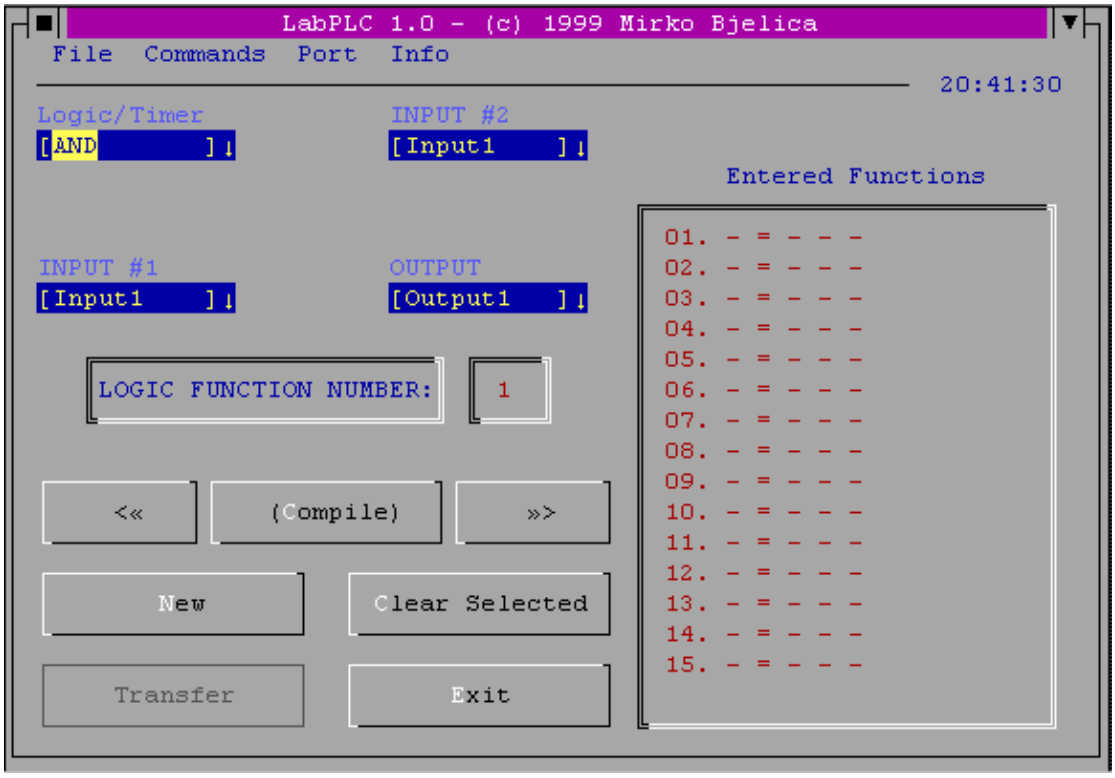
LabPLC *Software*

For the proper work of PLC device it's necessary to put in his memory a program on which it will work. It can be accomplished in several ways, and one, that I used, is to use personal computer for downloading program into PLC. This is of course possible if PC has installed proper software and in further reading I will explain how to use software that I made. Application is created using programming language *Visual Basic 1.0 for DOS.* It's been realized to maximally simplify functions entering by only selecting them and not entering logic functions, inputs, outputs or memory markers by writing names from keyboard. Next table shows codes of logic functions, inputs, outputs and memory markers. Of course, this codes are completely useless for user, but they could be useful for future upgrades or if someone would wish to make other versions of this software (Windows version for example).

NAME	CODE	NAME	CODE
AND	**10**	**OUTPUT 3**	**22**
OR	**11**	**OUTPUT 4**	**23**
NOT	**12**	**MARKER 0**	**30**
NAND	**13**	**MARKER 1**	**31**
NOR	**14**	**MARKER 2**	**32**
EXOR	**15**	**MARKER 3**	**33**
TIMER	**24**	**MARKER 4**	**34**
INPUT 1	**16**	**MARKER 5**	**35**
INPUT 2	**17**	**MARKER 6**	**36**
INPUT 3	**18**	**MARKER 7**	**37**
INPUT 4	**19**	**MARKER 8**	**38**
OUTPUT 1	**20**	**MARKER 9**	**39**
OUTPUT 2	**21**	**END CODE**	**99**

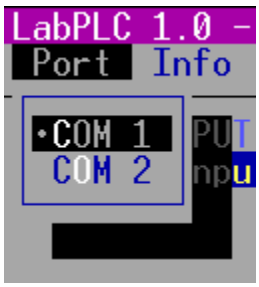
Table 1. Codes

Software starts when user changes path into directory where LabPLC files are copied and types into command line "labplc.exe". The path could be C:\LABPLC\labplc.exe. After this action, start screen will show.



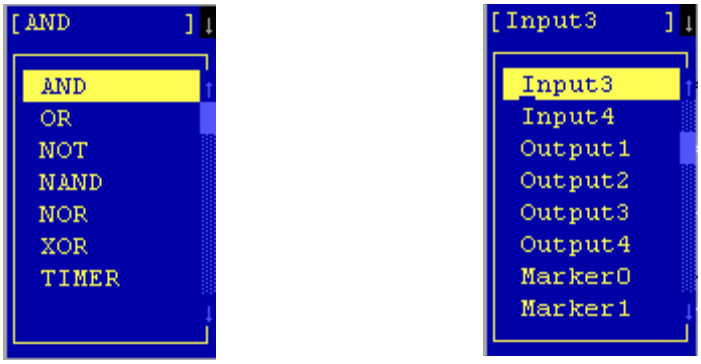
Picture 1. Start Screen

After starting the program, user should select proper communication port (COM Port) for correct connecting with LabPLC. Default setting is for COM1. Selection is made from “Port” menu.



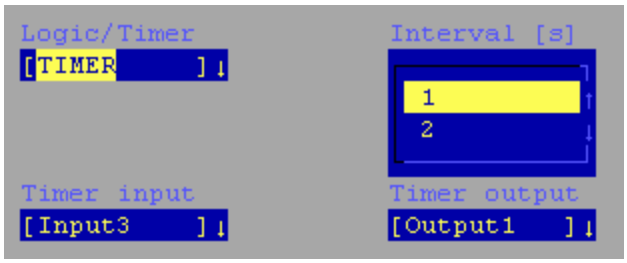
Picture 2. COM Port selection

In order to consider program working and explaining the function of some elements from start menu it can be split into three parts. Part for logic elements and their inputs and outputs selection, which consists of four *combo box* controls, keypad and window for showing entered functions.



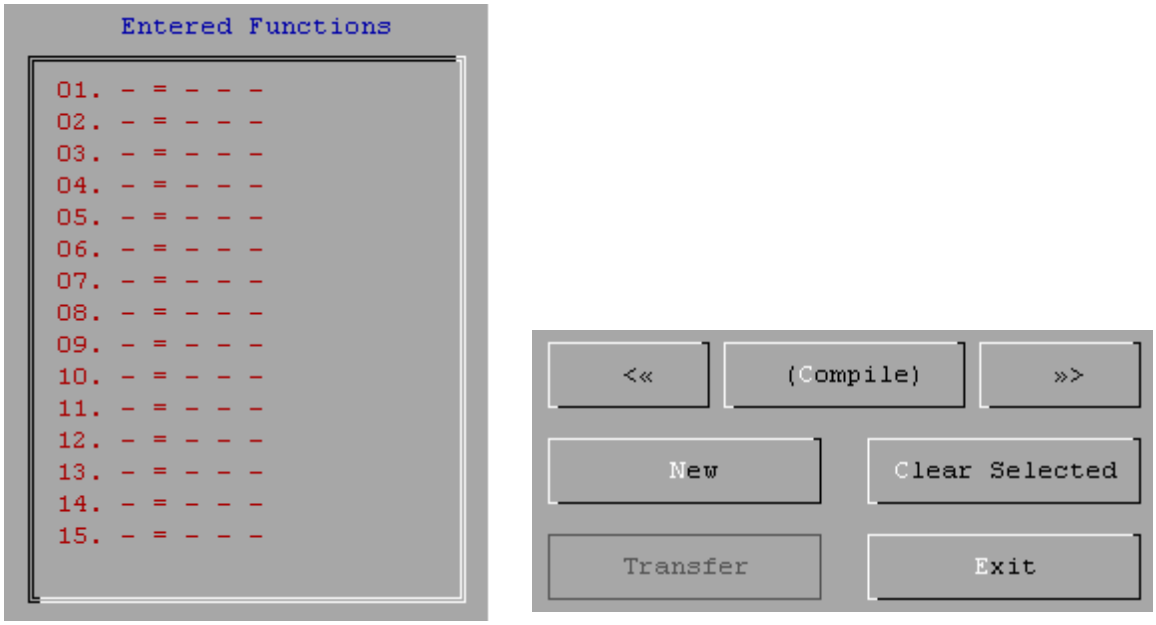
Picture 3. Combo Box controls for logic function selection and one of inputs

If Timer function is selected, *list box* will appear instead of *combo box* for second input selection and in this *list box* user can select preset time for timer in range from 1 to 99 seconds. When timer gets high logic level on his input, timer will set output to high level and this level will hold for preset time.



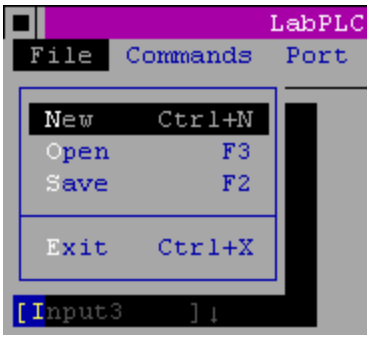
Picture 4. Timer properties

After first selection is made, it is necessary to interpret (Compile) logic function or timer into code that microcontroller would ”understand”. This action could be achieved by activating “(Compile)” key, which is located on the keypad or selecting this function from “Command” menu. After activation of this option, entered logic function is shown in “Entered functions” window and pointer is incremented.



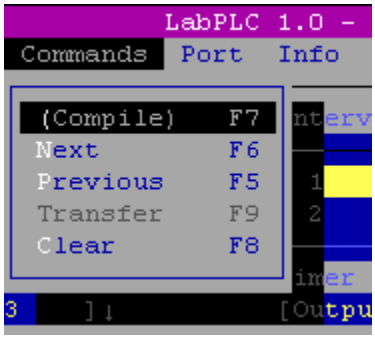
Picture 5. Part of the screen, which indicates entered functions and keypad

Besides "Port" falling menu, there is two more menus, "Commands" menu and "File" menu. In menu "File" there are four standard options: **New, Load, Save As** and **Exit**. Selecting first option, completely new input will be started and all previous functions will be erased. "Load" and "Save As" options are used for file manipulations. It is often in other programs that File menu has these two options. Option "Exit" is used to exit program.



Picture 6. "File" menu

Menu "Commands" consists of five options with theirs shortcuts and these options also could be found on keypad. It should be mentioned that "Compile" isn't real compile (There is no compiler software), it is more like interpreter but in this program word compile is more appropriate for use. Next picture shows this menu.



Picture 7. "Commands" menu

FUNCTION	SHORTCUT
NEW	Ctrl+N
LOAD	F3
SAVE	F2
EXIT	Ctrl+X
COMPILE	F7
NEXT	F6
PREVIOUS	F5
TRANSFER	F9
DELETE	F8

Table2. Shortcut codes

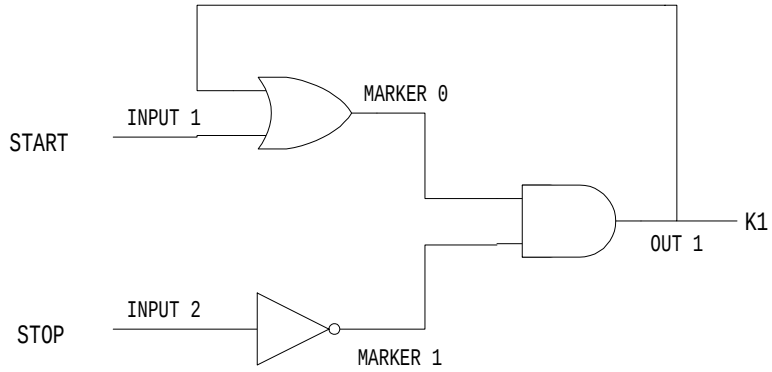
HOW TO HANDLE LabPLC

It is very easy to use LabPLC. When you turn on switch PR1 you are connecting 24V power supply to PLC and LED L1 (POWER ON) goes on. I used 24V because it is a standard for industrial voltage levels for supply and for control signals. At the same time LED L2 starts to blink and that signifies that PLC is READY for data transmission. Transmission between PC and PLC goes with a speed of 4800 bits per second with 8 data bits, none parity and 1 stop bit. When transfer ends, L2 goes off and L11 (READY) starts to shine. From that moment PLC starts cyclical program execution. Changing logic states on inputs would directly affect logic states on outputs and it's controlled by the logic, implemented true program.

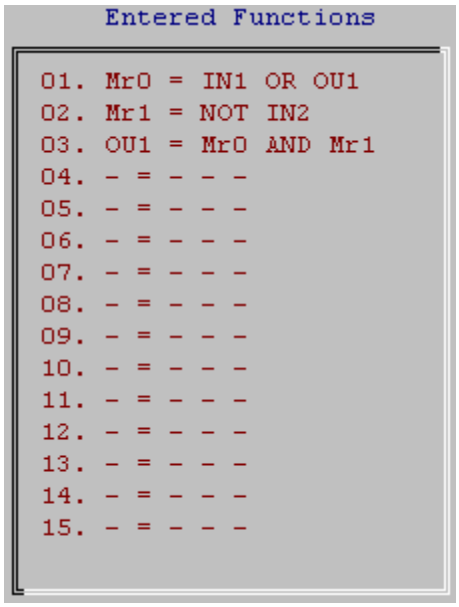
EXAMPLES

When I finished construction of PLC I decided to make few examples of electro-motor drive in order to show its use. First example illustrates how to start electro-motor using the PLC. Next example shows changing direction of shaft rotation and finally, the last example demonstrates starting of three phase electro-motor in star-delta mode.

Example1:

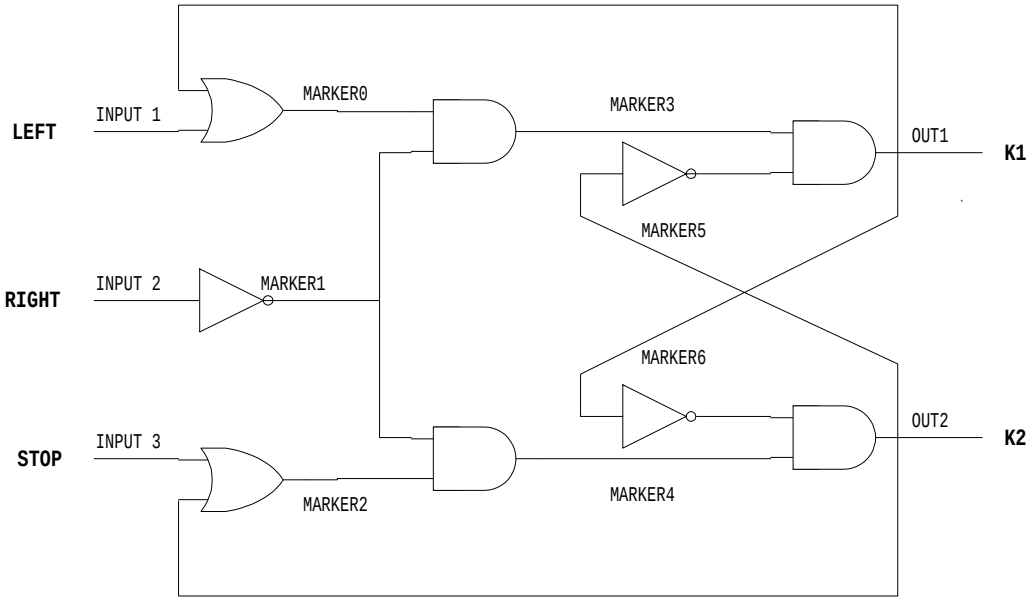


Example1. Electro-motor starting

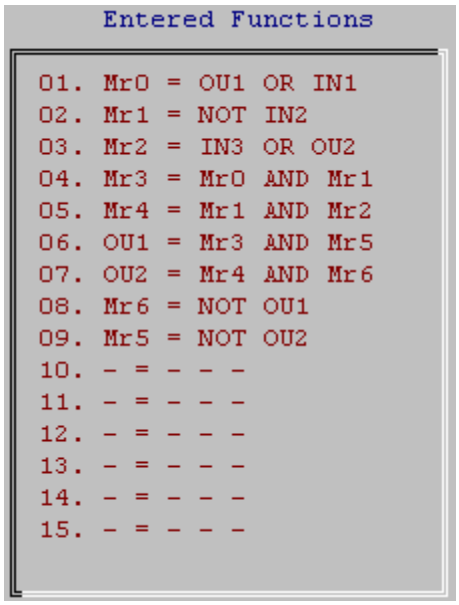


Example1-1. Program window of entered functions

Example2:

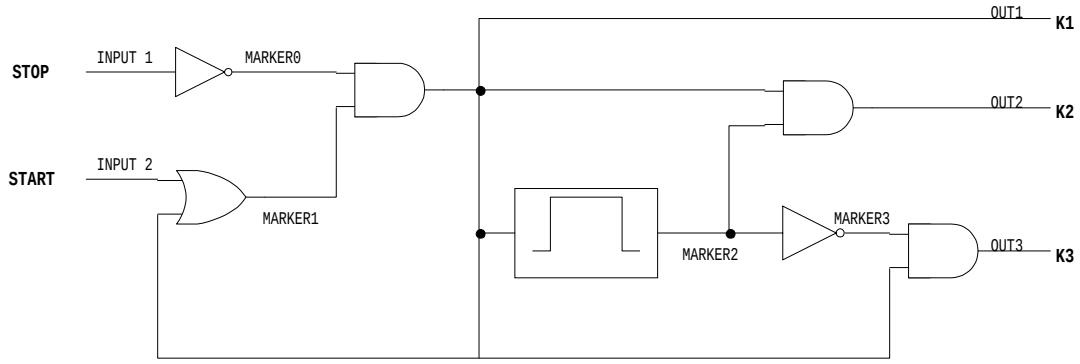


Example2. Changing direction of shaft rotation

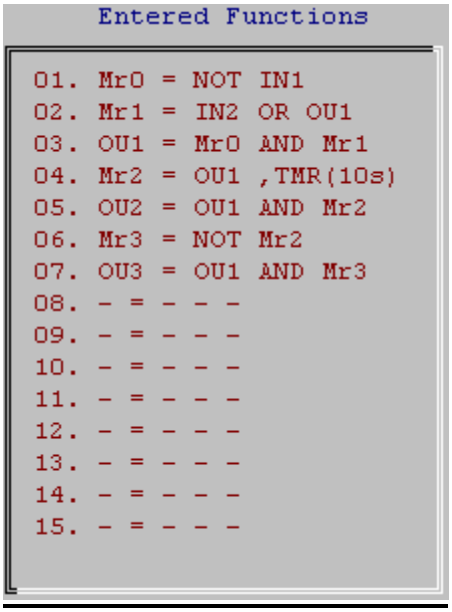


Example2-2. Program window of entered functions

Example3:



Example3. Starting 3 phase electro-motor in star-delta mode



Example3-3. Program window of entered functions