

**E-PLC: THE DEVELOPMENT OF A PROGRAMMABLE LOGIC CONTROLLER
TRAINER THAT TRANSLATES MNEMONIC CODES TO HARDWARE
SIMULATION**

M. H. A. H. A. M. Faseh¹, F. N. Ismail¹, M. A. Majid¹, A. F. Z. Abidin^{3,*}, Z. M. Yusoff¹, R. Rifin¹, K. K. Hasan¹, N. M. Ali² and Z. I. Rizman⁴

¹Faculty of Electrical Engineering, Universiti Teknologi MARA, Pasir Gudang, Johor, Malaysia

²Faculty of Electrical Engineering, Universiti Teknikal Malaysia Melaka, Malaysia

³Faculty of Engineering Technology, Universiti Teknikal Malaysia Melaka, Malaysia

⁴Faculty of Electrical Engineering, Universiti Teknologi MARA, 23000 Dungun, Terengganu, Malaysia

Published online: 01 February 2018

ABSTRACT

Programmable Logic Controller subject has been a core subject to most of the electrical engineering courses. Although Programmable Logic Controller is relevant to industry, it is expensive. This project aim is to design a low cost Programmable Logic Controller trainer that simulates user inserted mnemonic codes to the given electronic components. Furthermore, the trainer kit is designed to not only cater the tertiary level but also suitable for secondary level. The architecture of this trainer is using Arduino Mega as the brain that translates the mnemonic codes to electronic components execution. This trainer consists of seven basic instructions and nine electronic components.

Author Correspondence, e-mail: amarfaiz.utm@gmail.com

doi: <http://dx.doi.org/10.4314/jfas.v10i2s.36>



To verify the functionality of the trainer kit, all possible scenarios had been tested and the result obtained shows the trainer works as expected.

Keywords: programmable logic controller; mnemonic codes; Arduino Mega; PLC; trainer.

1. INTRODUCTION

Programmable Logic Controller (PLC) has been widely opted by industries as the controller that controlled most of the machineries. Therefore, it is no surprise to see that PLC subject had been a core subject in most of the engineering courses. Most of teaching and learning sessions of PLC in the universities are divided into three categories-lectures, tutorials and laboratory sessions. While lecture sessions are the time where the knowledge is imparted by the lecturer to the students, the tutorial sessions required the students to do their homework in advance before having a much more active learning during the sessions. These two categories give more emphasize on cognitive learning, thus left most of the psychomotor-based learning to be done in the laboratory sessions. The laboratory sessions are limited and require expert matter (such as lecturer, teaching engineer, assistant engineer, tutor, etc.) to conduct the laboratory session. To make matter worse, most of this laboratory sessions is instructional-based where the students read the laboratory sheet, follow the instruction and observed the outcome, thus limited the creativity and curiosity of the students to try out something outside of the laboratory sheet. One point worth to mention is that most of the trainer used in this laboratory sessions are expensive. This project aims to create a low cost teaching kit or trainer that allows student to simulate the PLC's mnemonic codes on the hardware without any supervision of expert matter. The trainer also must be easy to use and learn so that the primary students can also use it. This is necessary as there is a push from the ministry of education to introduce programming and embedded programming to secondary students.

Due to the significant importance of PLC in the industries, there are many PLC trainers that available in the market. Most of the trainers are teaching kit that include all electrical components and PLC controller being built in a single board with all the "hardwiring" are done inside the board and the user uses the trainer to do the "soft wiring" and upload the

program into the PLC board in order to see the outcome. Examples of this type of trainers are United Electronics's PLC Trainer [1], LearnLab's PLC Training System [2], Instrument India's PLC Trainer [3], Compactlogix Training System [4], AD Instruments PLC Trainer [5], ABB PLC-Trainer 500 [6] and Amatrol Portable's PLC Learning System 990-P [7]. Another kind of teaching aid/kit is the simulators. There are many simulators related to PLC where all simulation can be done using the personal computer (PC). Examples of PLC simulators are PSIM PLC Training Simulator [8], SMC Simulator [9] and Simutech Multimedia's Troubleshooting PLC [10]. This project deviates that the mentioned examples above in which the trainer focus on basic mnemonic codes that cater not only tertiary level students but also down to primary school students.

2. METHODOLOGY

Fig. 1 shows proposed PLC trainer kit, e-PLC. This trainer kit consists of three main components which are the place where user inserts the desire mnemonic code command, the display that shows the selected mnemonic codes and the electronic components used for the simulation purposed.

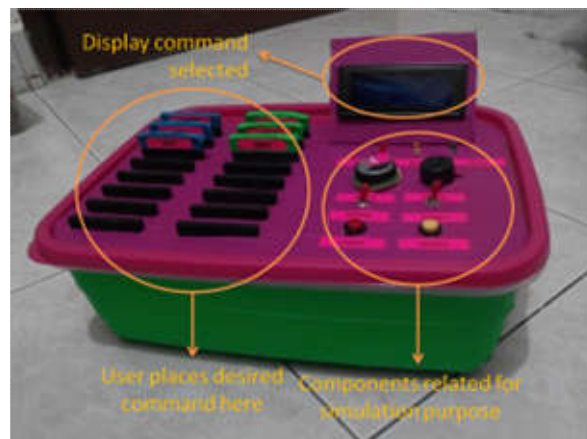


Fig.1. Prototype of the proposed trainer kit

The place to insert the commands consists of eight lines of commands, where each command consists of combination an instruction (left side) and component (right side). This trainer features seven basic instructions and nine electronic components as shown in Table 1. The seven basic instructions are LD, LD NOT, AND, AND NOT, OR, OR NOT and OUT. Meanwhile, the nine components are two toggle switches (with address of 0000 and 0001),

two push buttons (with address 0002 and 0003), three LEDs (red is 1000, yellow is 1001 and green is 1002), a DC motor (with address 1003) and a buzzer (with address 1004).

Table 1. List of instructions and electronic components included in EPLC and their corresponding resistor values

ID	Command	Type	Resistor Value
0	LD (Load)	Instruction	1000 Ω +/- 5%
1	LD NOT (Load Not)	Instruction	1500 Ω +/- 5%
3	AND	Instruction	2000 Ω +/- 5%
4	AND NOT	Instruction	2500 Ω +/- 5%
5	OR	Instruction	3000 Ω +/- 5%
6	OR NOT	Instruction	3500 Ω +/- 5%
7	OUT	Instruction	4000 Ω +/- 5%
8	1002 (Green LED)	Component	4500 Ω +/- 5%
9	1001 (Yellow LED)	Component	5000 Ω +/- 5%
10	1000 (Red LED)	Component	5500 Ω +/- 5%
11	1004 (Buzzer)	Component	6000 Ω +/- 5%
12	1003 (DC Motor)	Component	6500 Ω +/- 5%
13	0002 (Push Button 1)	Component	7000 Ω +/- 5%
14	0003 (Push Button 2)	Component	7500 Ω +/- 5%
15	0000 (Toggle Switch 1)	Component	8000 Ω +/- 5%
16	0001 (Toggle Switch 2)	Component	8500 Ω +/- 5%

Arduino Mega able to detect which command inserted by the user by detecting the resistor values that corresponds to the command which had been listed in Table 1. Note that the Arduino [11-13] cannot directly measure a resistor value, therefore a simple voltage divider circuit is used to determine the resistor value as shown in Fig. 2.

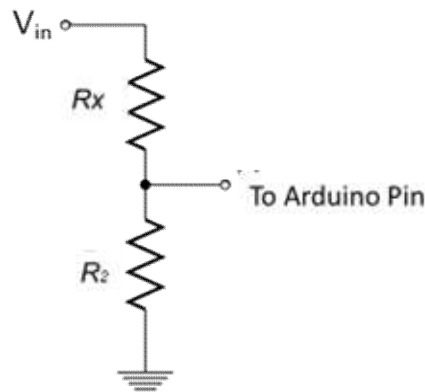


Fig.2. Voltage divider circuit to determine the corresponding command

The value of R_x is based on the command resistor value, for example R_x for command OUT is $3.5\text{k}\ \Omega$. R_2 is a fixed resistor value of $1\text{k}\ \Omega$. V_{in} is connected to corresponding Arduino digital pin that will energize to 5V when it wants to read the command inserted by the user. The output voltage of the voltage divider, V_o can be computed using Equation (1).

$$V_o = V_{in} \times \frac{R_x}{R_2} \quad (1)$$

Next, V_o is feed into the corresponding analogue pin. This voltage is translated into digital value, D by Arduino Mega's ADC that can vary from 0 to 1023 based on Equation (2). Note that Arduino Mega's ADC is a 10-bit ADC.

$$D = (2^{10} - 1) \times \frac{V_o}{5} \quad (2)$$

Then, the value of R_x can be calculated back using Equation (3).

$$R_x = 5 \times \frac{D}{(2^{10} - 1)} \times \frac{R_2}{V_{in}} \quad (3)$$

Once, the value of R_x calculated, the algorithm can identify the command selected. Note that in actual implementation, the resistor value can vary from its ideal value, ePLC used resistors $\pm 1\%$ while the algorithm accepts a margin of 5% . This means that for LD command is a resistor value of $1\text{k}\ \Omega$ which can have values of $990\ \Omega$ to $1010\ \Omega$ and the algorithm can accepts values of $950\ \Omega$ to $1050\ \Omega$. Fig. 3 shows the schematic of the circuit diagram of ePLC.

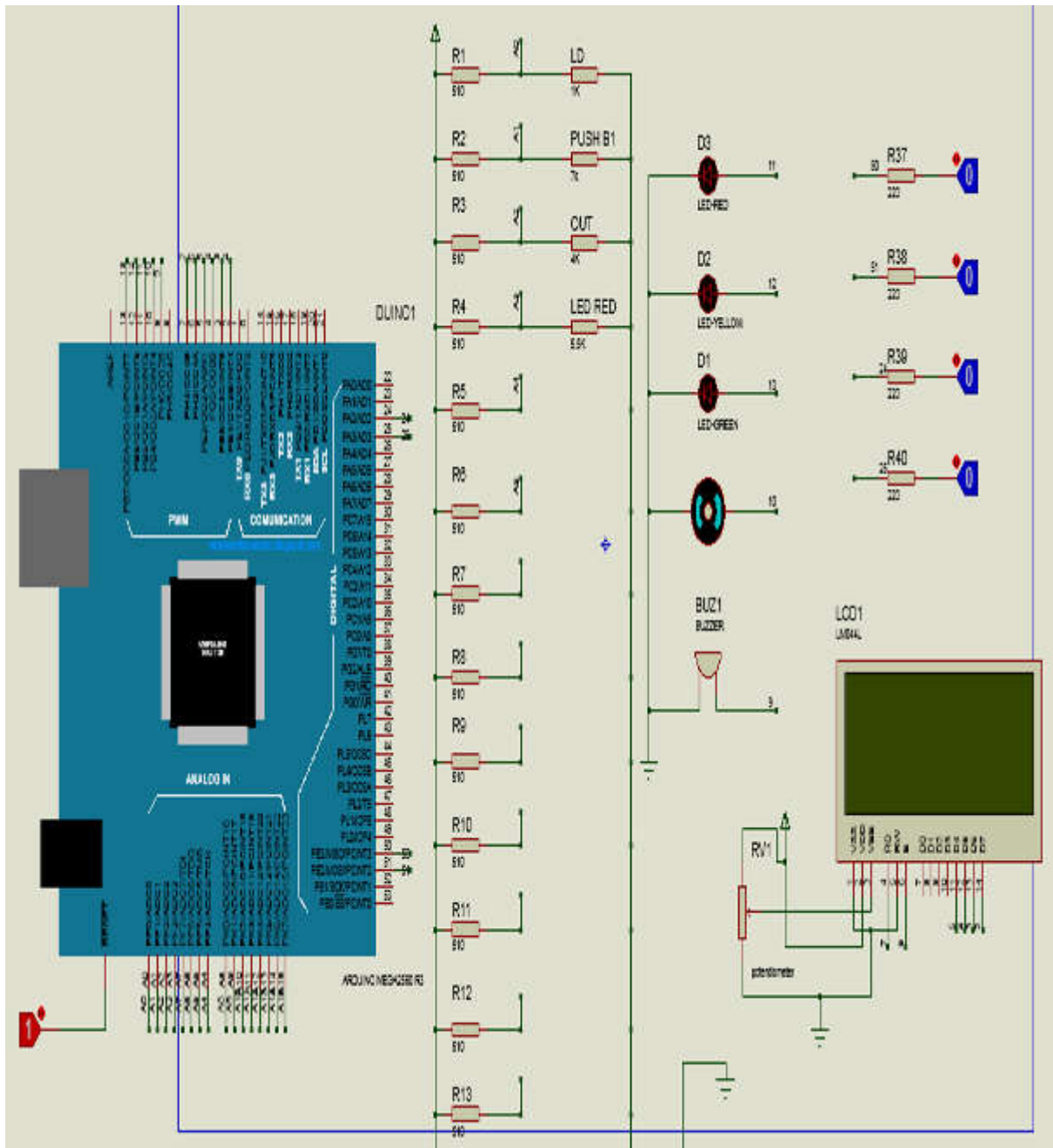













Fig.3. Schematic diagram of e-PLC







3. RESULTS AND DISCUSSION






Table 2 shows the list of scenarios tested and the result obtained. The aim of this scenarios testing is to ensure all the possible scenario that might be tested by the user has been validated the output, thus conclude e-PLC is fit for use. Scenario 1 until 5 test different outputs. Scenario 6 until 34 test different possible logic combinatorial of two inputs. Section 35 to 38 test different inputs. Result obtained for all scenarios are as expected.







Table 2. List of scenarios tested and the result obtained

No.	Scenario	Result Obtained	Result
1	The user constructing the LD block with input address: 0000 and OUT block with output address: 1000	The result is as expected. The Red LED (1000) light up when the Toggle Button 1 (0000) was switch on.	
2	The user constructing the LD block with input address: 0000 and OUT block with output address: 1001	The result is as expected. The Yellow LED (1001) light up when the Toggle Button 1 (0000) was switch on.	
3	The user constructing the LD block with input address: 0000 and OUT block with output address: 1002.	The result is as expected. The Green LED (1002) light up when the Toggle Button 1 (0000) was switch on.	
4	The user constructing the LD block with input address: 0000 and OUT block with output address: 1003.	The result is as expected. The DC Motor (1003) was energized when the Toggle Button 1 (0000) was switch on.	
5	The user constructing the LD block with input address: 0000 and OUT block with output address: 1004.	The result is as expected. The Buzzer (1004) producing "Beep" sound when Toggle Button 1 (0000) was switch on.	

6	<p>The user constructing the LD block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was LOW because of AND logic $0 \times 0 = 0$ (LOW)</p>	
7	<p>The user constructing the LD block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was LOW because of AND logic $0 \times 1 = 0$ (LOW)</p>	
8	<p>The user constructing the LD block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was LOW because of AND logic $1 \times 0 = 0$ (LOW)</p>	
9	<p>The user constructing the LD block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was HIGH because of AND logic $1 \times 1 = 1$ (HIGH)</p>	
10	<p>The user constructing the LD block and OR block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was LOW because of OR logic $0 + 0 = 0$ (LOW)</p>	
11	<p>The user constructing the LD block and OR block with input addresses (0000 and 0001) and OUT block with</p>	<p>The result is as expected. The output of Red LED was HIGH because of OR logic $1 + 0 = 1$ (HIGH)</p>	

	output address: 1000		
	The user constructing the LD block and OR block with	The result is predictable. The	
12	input addresses (0000 and 0001) and OUT block with	output of Red LED was HIGH	
	output address: 1000	because of OR logic	
		$0 + 1 = 1$ (HIGH)	
	The user constructing the LD block and OR block with	The result is as expected. The	
13	input addresses (0000 and 0001) and OUT block with	output of Red LED was HIGH	
	output address: 1000	because of OR logic	
		$1 + 1 = 1$ (HIGH)	
	The user constructing the LD block and AND NOT block	The result is as expected. The	
14	with input addresses (0000 and 0001) and OUT block	output of Red LED was LOW	
	with output address: 1000	because of AND logic with	
		NOT logic on second input	
		$0 \times 1 = 0$ (LOW)	
	The user constructing the LD block and AND NOT block	The result is as expected. The	
15	with input addresses (0000 and 0001) and OUT block	output of Red LED was HIGH	
	with output address: 1000	because of AND logic with	
		NOT logic on second input	
		$1 \times 1 = 1$ (HIGH)	
	The user constructing the LD block and AND NOT block	The result is as expected. The	
16	with input addresses (0000 and 0001) and OUT block	output of Red LED was LOW	
	with output address: 1000	because of AND logic with	
		NOT logic on second input	
		$0 \times 0 = 0$ (LOW)	
	The user constructing the LD block and AND NOT block	The result is as expected. The	
17	with input addresses (0000 and 0001) and OUT block	output of Red LED was LOW	
	with input addresses (0000 and 0001) and OUT block	because of AND logic with	

	and 0001) and OUT block	NOT logic on second input	
	with output address: 1000	$1 \times 0 = 0$ (LOW)	
	The user constructing the LD	The result is as expected. The	
	block and OR NOT block	output of Red LED was HIGH	
18	with input addresses (0000	because of OR logic with NOT	
	and 0001) and OUT block	logic on second input	
	with output address: 1000	$0 + 1 = 1$ (HIGH)	
	The user constructing the LD	The result is predictable. The	
	block and OR NOT block	Output of Red Led was HIGH	
19	with input addresses (0000	because of OR logic with NOT	
	and 0001) and OUT block	logic on second input	
	with output address: 1000	$1 + 1 = 1$ (HIGH)	
	The user constructing the LD	The result is as expected. The	
	block and OR NOT block	output of Red LED was LOW	
20	with input addresses (0000	because of OR logic with NOT	
	and 0001) and OUT block	logic on second input $0 + 0 = 0$	
	with output address: 1000	(LOW)	
	The user constructing the LD	The result is as expected. The	
	block and OR NOT block	output of Red LED was HIGH	
21	with input addresses (0000	because of OR logic with NOT	
	and 0001) and OUT block	logic on second input $1 + 0 = 1$	
	with output address: 1000	(HIGH)	
	The user constructing the LD	The result is as expected. The	
	NOT block and AND block	output of Red LED was LOW	
22	with input addresses (0000	because of AND logic also NOT	
	and 0001) and OUT block	logic on first input $1 \times 0 = 0$	
	with output address: 1000	(LOW)	

23	<p>The user constructing the LD NOT block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was HIGH because of AND logic also NOT logic on first input $1 \times 1 = 1$ (HIGH)</p>	
24	<p>The user constructing the LD NOT block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The Output of Red Led was LOW because of AND logic also NOT logic on first input $0 \times 0 = 0$(LOW)</p>	
25	<p>The user constructing the LOAD NOT block and AND block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was LOW because of AND logic also NOT logic on first input $0 \times 1 = 0$ (LOW)</p>	
26	<p>The user constructing the LD NOT block and OR block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was HIGH because of OR logic also NOT logic on first input $1 + 0 = 1$ (HIGH)</p>	
27	<p>The user constructing the LD NOT block and OR block with input addresses (0000 and 0001) and OUT block with output address: 1000</p>	<p>The result is as expected. The output of Red LED was HIGH because of OR logic also NOT logic on first input $1 + 0 = 1$ (HIGH)</p>	
28	<p>The user constructing the LD NOT block and OR block with input addresses (0000</p>	<p>The result is as expected. The output of Red LED was LOW because of OR logic also NOT</p>	

and 0001) and OUT block logic on first input
 with output address: 1000 $0 + 0 = 0$ (LOW)

The user constructing the LD The result is as expected. The
 NOT block and OR block output of Red LED was HIGH
 29 with input addresses (0000 because of OR logic also NOT
 and 0001) and OUT block logic on first input
 with output address: 1000 $0 + 1 = 1$ (HIGH)



The user constructing the LD The result is as expected. The
 NOT block and AND NOT output of Red LED was HIGH
 30 block with input addresses because of AND logic with
 (0000 and 0001) and OUT NOT logic on first and second
 block with output address: input
 1000 $1 \times 1 = 1$ (HIGH)



The user constructing the LD The result is as expected. The
 NOT block and AND NOT output of Red LED was LOW
 31 block with input addresses because of AND logic with
 (0000 and 0001) and OUT NOT logic on first and second
 block with output address: input
 1000 $1 \times 0 = 0$ (LOW)








The user constructing the The result is as expected. The
 LOAD NOT block and AND output of Red LED was LOW
 32 NOT block with input because of AND logic with
 addresses (0000 and 0001) NOT logic on first and second
 and OUT block with output input
 address: 1000 $0 \times 1 = 0$ (LOW)



The user constructing the LD The result is as expected. The
 NOT block and AND NOT output of Red LED was LOW
 33 block with input addresses because of AND logic with
 (0000 and 0001) and OUT NOT logic on first and second



	<p>block with output address: input</p>		
	<p>1000</p>	<p>$0 \times 0 = 0$ (LOW)</p>	
	<p>The user constructing the LD</p>	<p>The result is as expected. The</p>	
	<p>NOT block and OR NOT</p>	<p>output of Red LED was HIGH</p>	
34	<p>block with input addresses</p>	<p>because of OR logic with NOT</p>	
	<p>(0000 and 0001) and OUT</p>	<p>logic on first and second input</p>	
	<p>block with output address:</p>	<p>$1 + 1 = 1$ (HIGH)</p>	
	<p>1000</p>		
	<p>The user constructing the LD</p>	<p>The result is as expected. The</p>	
	<p>block with different input</p>	<p>output of Green LED was HIGH</p>	
35	<p>address: 0000 and OUT block</p>	<p>because of the Toggle Switch 1</p>	
	<p>with output address: 1002</p>	<p>(0000) was set to HIGH</p>	
	<p>The user constructing the LD</p>	<p>The result is as expected. The</p>	
	<p>block with different input</p>	<p>output of Green LED was HIGH</p>	
36	<p>address: 0001 and OUT block</p>	<p>because of the Toggle Switch 2</p>	
	<p>with output address: 1002</p>	<p>(0001) was set to HIGH</p>	
	<p>The user constructing the LD</p>	<p>The result is as expected. The</p>	
	<p>block with different input</p>	<p>output of Green LED was HIGH</p>	
37	<p>address: 0003 and OUT block</p>	<p>because of the Push Button 1</p>	
	<p>with output address: 1002</p>	<p>(0002) was set to HIGH</p>	
	<p>The user constructing the LD</p>	<p>The result is as expected. The</p>	
	<p>block with different input</p>	<p>Output of Green Led was HIGH</p>	
38	<p>address: 1003 and OUT block</p>	<p>because of the Push Button 2</p>	
	<p>with output address: 1002</p>	<p>(0003) was set to HIGH</p>	

4. CONCLUSION

This paper presented the development of e-PLC, a low-cost, easy-to-use PLC trainer that allows students to insert mnemonic codes and simulate the outcome using the built-in hardware. Result obtained shows e-PLC is fit for use.

5. ACKNOWLEDGEMENTS

The authors would like to thank Ministry of Higher Education (MOHE) and Research Management Institute of Universiti Teknologi MARA (UiTM) for providing financial support under ARAS fund grant 600-IRMI/DANA 5/3/ARAS (0120/2016).

6. REFERENCES

- [1] Indiamart. PLC trainer. 2015, <https://www.indiamart.com/proddetail/plc-trainer-9657051862.html>
- [2] ITU Corporation. PLC training systems. 2017, <http://trainingpanels.com/product/plc-training-system/>
- [3] Indiamart. PLC trainer system. 2014, <https://www.indiamart.com/proddetail/plc-trainer-system-9816963933.html>
- [4] Motor Control International. PLC trainers. 2010, http://www.motorcontrolinternational.com/html/plc_trainers.html
- [5] AD Instruments. PLC Trainer (GLOFA GM-4, 3 Basic Modules) - ED4260. 2018, <http://www.adinstruments.es/PLC-Trainer-GLOFA-GM-4-3-Basic-Modules-ED4260/en>
- [6] IKH Didactic Systems. PLC-Trainer AC500 - with CPU and programming software. 2010, <http://www.ikhds.com/index.php/plc-trainer-ac500-290.html>
- [7] Amatrol Inc. Portable PLC Training System Siemens S7-1200. 2018, <http://www.amatrol.com/coursepage/990-ps712/>
- [8] The Learning Pit. PSIM-PLC Training Simulator. 2018, <http://www.thelearningpit.com/plc/psim/psim.html>
- [9] Consolidating of Nothings. PLC Simulator. 2018, <http://www.consolidationofthings.com/PLCsim.html>
- [10] Business Industrial Network. Troubleshooting PLC Controls, Circuits with PLC Simulator. 2017, <https://bin95.com/Troubleshooting-PLC-Controls-Simulator.htm>
- [11] Rizman Z I, Hashim F R, Yassin I M, Zabidi A, Zaman F K, Yeap K H. Smart multi-application energy harvester using Arduino. *Journal of Fundamental and Applied Sciences*, 2018, 10(1S):689-704

[12] Abdullah R, Rizman Z I, Dzulkefli N N, Ismail S I, Shafie R, Jusoh M H. Design an automatic temperature control system for smart tudungsaji using Arduino microcontroller. ARPN Journal of Engineering and Applied Sciences, 2016, 11(16):9578-9581

[13] Nawi B, Sulaini B, Mohd Z A, Shamsul A Z, Zairi I R. PID voltage control for DC motor using MATLAB Simulink and Arduino microcontroller. Journal of Applied Environmental and Biological Sciences, 2015, 5(9):166-173

How to cite this article:

Faseh MHAHAM, Ismail FN, Majid MA, Abidin AFZ, Yusoff ZM, Rifin R, Hasan KK, Ali N M, Rizman ZI. E-plc: the development of a programmable logic controller trainer that translates mnemonic codes to hardware simulation. J. Fundam. Appl. Sci., 2018, 10(2S), 499-513.