Unizik Journal of Technology, Production and Mechanical Systems (UJTPMS)

journal homepage: https://journals.unizik.edu.ng/index.php/ujtpms/about

A cost-effective design for read-only memory based digital system emulation

Ifeyinwa C. Obiora-Dimson[1*], Charles I. Nduka[1], Lois O. Nwobodo[2]

[1]Department of Electronic and Computer Engineering,
Nnamdi Azikiwe University, Awka, NIGERIA
[2] Department of Computer Engineering,
Enugu State University of Science & Technology, (ESUT), Enugu, NIGERIA

ARTICLE INFO

Article history:
Received Nov. 20, 2022
Revised Dec. 14, 2022
Accepted Jan. 10, 2023

Available online Feb. 22, 2023

Keywords:
Emulation, Logic trainers, ROM, Truth table.

ABSTRACT

A digital systems design approach using emulation technique and Read Only Memory (ROM) as a major component has been presented. Emulation as implied here, simply means using a logic device to mimic the behavior of one or more logic devices or a control system. To effectively mimic these logic devices, their characteristics drawn from their individual truth tables (or logic table) where programmed into the ROM using their input patterns as the address to the memory location where their output patterns are stored. With the connection of other peripheral devices such as input and output devices, display, select buttons etc., a ROM based digital systems design is developed. A specific design problem involving the development of a logic trainer was demonstrated. The trainer developed is portable, has simple logic section and is cost effective. The emulation technique used in this design is systematic and can be applied to develop other digital system designs.

1. Introduction

Digital systems is a compulsory course for university undergraduates of Departments of Electronic, Computer engineering and Electrical engineering. Digital systems design as a course is practical intensive. Suffice it to say also that visual learning impacts better to any individual than only oral teaching. To achieve this visual impact required in this course, Digital laboratory trainer kits and virtual labs [1] are of essence in carrying out these practical. The formation and impact of technical skills as a result of the use of digital trainers has been rated 'Much needed' [1], and 60% effective [2] in universities. But these practical has not been fully harnessed in some institutions developing nations because of non-availability of Digital logic trainers. [3]

Unfortunately these kits are usually not available as a result of high cost of importation, lack of operational manual, non-availability of essential part(s) of the kit and inability of lab personnel to understand the usage of the kit. Therefore some of these kits end up unused and often rust away in some laboratories. Most available digital trainers have been confirmed to have complex logic section, very expensive, non-portable and costly to maintain [4]. Again, because of non-affordability of some of these trainer kits, departments usually resort to buying a maximum of two. Thus, making it only for use by the lab instructor for demonstration to the students. Thus it ends up also in lecture for a supposed hands-on practical class for the students.

Emulation simply means making one device behave like and perform the function of another or several other devices. In logic design, universal gates i.e. NAND and NOR could be used to carry out the function of other basic logic gates like AND, OR etc. In the same manner a ROM or a microcontroller [5] can be used to emulate several other logic gates.

There is need for indigenous effort geared towards design of logic trainers and other digital systems. This paper therefore aims at showcasing practical design steps towards development of digital systems using ROM. A ROM based digital logic trainer kit for use in digital systems laboratories in studying the properties of various digital logic gates and components in order to verify and

*Corresponding author: ifeyinwaodimson@gmail.com

interpret their Boolean expression would be used as a case study. This would be achieved by employing emulation technology to mimic the characteristics of the basic logic gates in a single kit.

2. Methods and Materials

The method that would be used is emulation. Every control system whose characteristics can be tailored to a logic table (i.e. state transition table or truth table) such that there is an input and output pattern can be programmed into a ROM. The input pattern serves as address to the ROM location where the output pattern is stored. Where the digital system requires processing, the details of the control system could be programmed into the ROM of the processor and upon receiving the input pattern, the control system pattern can be executed and output released.
The following steps are suggested to guide a developer to successfully emulate logic gates using a ROM.

1. Map out the logic gates to be emulated
2. Write out their truth tables
3. Collate the truth tables into a single table
4. Convert the input pattern per row into its hex equivalent
5. Convert the output patterns per row into hex equivalent
6. Program the hex patterns into ROM
7. Attached all peripheral devices needed for the emulator to work
8. Test the system

An area of application of emulation for logic gates is in the development of logic trainers. With the steps outlined above, several logic gates can be emulated and using a single ROM chip with the necessary interfacing devices, a logic trainer is produced. Among the gates that can be emulated includes:

i. One input gate
ii. Two input gates
iii. Three input gates
iv. Four input gates
v. One input decoder
vi. Two input decoder
vii. Three input decoder
viii. BCD to 7-segment decoder
ix. Mod N-counter
x. Up-down counter
xi. Down-up counter
xii. 1 input multiplexer
xiii. 2 input multiplexer
xiv. 3 input multiplexer etc.

Consider using a TMSJL27C128 ROM chip as an emulator to emulate the characteristics of several gates. This chip would take in 7-input and 8-outputs. Thus logic gates from numbers i to viii from the list shall be emulated. Each experiment would be represented or selected by binary numbers 000 to 111. These three bits already serves as part of the address thus 4-bits are left (of the 7 address bits). They would be used as input into (device select for) each device's truth table since a max of 4 bits is required for the 4-input gates (and also for the seven segment display).
From the steps outlined, one needs to generate the individual truth table of the gates. The tables are shown in tables 1 to 7.

Table 1 - Two input gates truth table

Input		**Output**					
A_1	A_0	AND	NAND	OR	NOR	EXOR	EXOR
0	0	0	1	0	1	1	0
0	1	0	0	1	0	0	1
1	0	0	0	1	0	0	1
1	1	1	0	1	0	1	0

Table 2 - Truth table for three input gates

Input			**Output**					
A_2	A_1	A_0	AND	NAND	OR	NOR	EXOR	EXNOR
0	0	0	0	1	0	1	0	1
0	0	1	0	0	1	0	1	0
0	1	0	0	0	1	0	1	0
0	1	1	0	0	1	0	0	1
1	0	0	0	0	1	0	1	0
1	0	1	0	0	1	0	0	1
1	1	0	0	0	1	0	0	1
1	1	1	1	0	1	0	1	0

Table 3 - Truth table for four input gates

Input				**Output**					
A_3	A_2	A_1	A_0	AND	NAND	OR	NOR	EXOR	EXNOR
0	0	0	0	0	1	0	1	0	1
0	0	0	1	0	1	1	0	1	0
0	0	1	0	0	1	1	0	1	0
0	0	1	1	0	1	1	0	0	1
0	1	0	0	0	1	1	0	1	0
0	1	0	1	0	1	1	0	0	1
0	1	1	0	0	1	1	0	0	1
0	1	1	1	0	1	1	0	1	0
1	0	0	0	0	1	1	0	1	0
1	0	0	1	0	1	1	0	0	1
1	0	1	0	0	1	1	0	0	1
1	0	1	1	0	1	1	0	1	0
1	1	0	0	0	1	1	0	0	1
1	1	0	1	0	1	1	0	1	0
1	1	1	0	0	1	1	0	1	0
1	1	1	1	1	0	1	0	0	1

Table 4 - Truth table for BCD to seven segment display

Input				**Output**						
A_3	A_2	A_1	A_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	0	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

Table 5 - Truth table for 1-to-2 line decoder

Input	**Output**	
A_0	D_1	D_0
0	0	1
1	1	0

Table 6-Truth table for 2-to-4 line decoder

Input		**Output**			
A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Table 7 -Truth table for 3-to-8 line decoder

Input			**Output**							
A_2	A_1	A_0	D_7	D_6	D_6	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

3. Results and Discussions

Next is to bring together all the tables (1 to 7) into a single table as shown in table 8. Recall that the table must be fully expanded, this implies that don't cares (unfilled locations) must be eliminated. Therefore the unfilled areas are populated with zeros. (see table 8). The hex codes are generated along the address and address content columns respectively. These Hex codes are then programmed into the ROM as the emulation software.

Next, peripheral devices are added and the system coupled. Peripheral devices such as push buttons can be used to select the device type. This would require 3 buttons A6, A5, and A4. Also four gate type select buttons can be used to select the gate types i.e., A3, A2, A1, A0. LEDs can be used for the seven segment display. A single LED is also connected to indicate logic 1 or logic 0 whenever the trainer is being used. A probe can be used to connect to the output terminal (ports D_7, D_6 D_5, D_4, D_3, D_2, D_1, and D_0) one is reading its output.

Table 8 - Combined Truth table with HEX codes

Address	**Input**							**Output**								**Address content**	**Description**
	Device type			Gate input				D7									
	A_6	A_5	A_4	A_3	A_2	A_1	A_0	D_6F_6EXNOR	D_5F_5EXOR	D_4F_4NOR	D_3F_3EOR	D_2f_2NAND	D_1F_1AND	D_0F_0INV			
00	0	0	0	0	0	0	0	0	0	0	0	0	0	1		01	Inverter
01	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	11	
10	0	0	1	0	0	0	0	0	1	0	1	0	1	0	0	54	Two input
11	0	0	1	0	0	0	1	0	0	1	0	1	1	0	0	2C	gates
12	0	0	1	0	0	1	0	0	0	1	0	1	1	0	0	2C	
13	0	0	1	0	0	1	1	0	1	0	0	1	0	1	0	4A	
20	0	1	0	0	0	0	0	0	1	0	1	0	1	0	0	54	Three input
21	0	1	0	0	0	0	1	0	0	1	0	1	1	0	0	2C	gates
22	0	1	0	0	0	1	0	0	0	1	0	1	1	0	0	2C	
23	0	1	0	0	0	1	1	0	1	0	0	1	1	0	0	4C	
24	0	1	0	0	1	0	0	0	0	1	0	1	1	0	0	2C	
25	0	1	0	0	1	0	1	0	1	0	0	1	1	0	0	4C	
26	0	1	0	0	1	1	0	0	1	0	0	1	1	0	0	4C	
27	0	1	0	0	1	1	1	0	0	1	0	1	0	1	0	2A	
30	0	1	1	0	0	0	0	0	1	0	1	0	1	0	0	54	Four input
31	0	1	1	0	0	0	1	0	0	1	0	1	1	0	0	2C	gates
32	0	1	1	0	0	1	0	0	0	1	0	1	1	0	0	2C	
33	0	1	1	0	0	1	1	0	1	0	0	1	1	0	0	4C	
34	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	2C	
35	0	1	1	0	1	0	1	0	1	0	0	1	1	0	0	4C	
36	0	1	1	0	1	1	0	0	1	0	0	1	1	0	0	4C	
37	0	1	1	0	1	1	1	0	0	1	0	1	1	0	0	2C	
38	0	1	1	1	0	0	0	0	0	1	0	1	1	0	0	2C	
39	0	1	1	1	0	0	1	0	1	0	0	1	1	0	0	4C	
3A	0	1	1	1	0	1	0	0	1	0	0	1	1	0	0	4C	
3B	0	1	1	1	0	1	1	0	0	1	0	1	1	0	0	2C	
3C	0	1	1	1	1	0	0	0	1	0	0	1	1	0	0	4C	
3D	0	1	1	1	1	0	1	0	0	1	0	1	1	0	0	2C	
3E	0	1	1	1	1	1	0	0	0	1	0	1	1	0	0	2C	
3F	0	1	1	1	1	1	1	0	1	0	0	1	0	1	0	4A	
40	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	02	One
41	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	02	Input Decode
50	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	01	Two
51	1	0	1	0	0	0	1	0	0	0	0	0	0	1	0	02	Input
52	1	0	1	0	0	1	0	0	0	0	0	0	1	0	0	04	Decode
53	1	0	1	0	0	1	1	0	0	0	0	1	0	0	0	08	

60	1	1	0	0	0	0	0	0	0	0	0	0	0	1	01	Three		
61	1	1	0	0	0	0	1	0	0	0	0	0	0	1	0	02	Input	
62	1	1	0	0	0	1	0	0	0	0	0	0	1	0	0	04	decode	
63	1	1	0	0	0	1	1	0	0	0	0	1	0	0	0	08		
64	1	1	0	0	1	0	0	0	0	0	1	0	0	0	0	10		
65	1	1	0	0	1	0	1	0	0	1	0	0	0	0	0	20		
66	1	1	0	0	1	1	0	0	1	0	0	0	0	0	0	40		
67	1	1	0	0	1	1	1	1	0	0	0	0	0	0	0	80		

								a	b	c	d	E	f	g		
70	1	1	1	0	0	0	0	0	1	1	1	1	1	1	0	7E	BCD to 7	
71	1	1	1	0	0	0	1	0	0	1	1	0	0	0	0	30	segment	
72	1	1	1	0	0	1	0	0	1	1	0	1	1	0	1	6D	decoder	
73	1	1	1	0	0	1	1	0	1	1	1	1	0	0	1	79		
74	1	1	1	0	1	0	0	0	0	1	1	0	0	1	1	33		
75	1	1	1	0	1	0	1	0	1	0	1	1	0	1	1	5B		
76	1	1	1	0	1	1	0	0	1	0	1	1	1	1	1	5F		
77	1	1	1	0	1	1	1	0	1	1	1	0	0	0	0	70		
78	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	7F		
79	1	1	1	1	0	0	1	0	1	1	1	1	0	1	1	7B		

The trainer output for each device emulated reflected their true properties indicated in tables 1 to 7. This emulation technique developed here can be used for developing similar trainer for other logic devices not covered in this example in addition to other control systems design. It can also be used to program the memory of a microcontroller.

4. Conclusion

Development of cost effective ROM based digital systems design with focus on developing a logic trainer using emulation technology has being demonstrated in the foregoing. Several logic gates and their characteristics as indicated by their truth tables where logically harnessed and programmed into ROM by making the input patterns serve as address into the memory location where their output patterns are stored. Other peripheral devices needed for the ROM to work effectively such as input/output port, LED display, select buttons are then interconnected. The technique is not complex and the device developed is cost effective, has ease of maintenance due to uncomplicated logic sections and is portable.

References

[1] Dumaguit, James M. (2020). Technology requirements assessment for the development of digital logic trainer. *International Journal of Research Studies in Education* 2020, 9:5, pp53-58.

[2] Zulkarnain, O., M., Amar, Z.A., Syahrul M. H., Nur Dalila. K.A, & Ismail, N. (2019). E-Logic Trainer Kit: Development of an Electronic Educational Simulator and Quiz Kit for Logic Gate Combinational Circuit by using Arduino as Application. *International Journal of Online and Biomedical Engineering (iJOE)*, *15*:14 pp 67–77. https://doi.org/10.3991/ijoe.v15i14.11410.

[3] Vincent Mulwa, Mutwiri Joseph, Joshua M. Mwema, Antony Gitong (2020). Remote-Controlled Digital Electronics Trainer Board (RCDET). *International Journal of Innovative Science and Research Technology*. IJISRT20APR110 www.ijisrt.com, 5:4.

[4] Intan Shafinaz Mohammad, Syafiq Abdul Halim, Ahmad Luqman Jamil, Muhammad Rashidi, and Nur Farahiyah Mohammad. (2018). Development of Digital Electronics Book Trainer. *Journal of Engineering Research and Education*, 10, pp49-56

[5] G. O. Uzedhe, H. C Inyiama, Udeze Chidiebele, and Mbonu Ekene (2013). Microcontroller Based Real-Time Emulator for Logic Gate and Structured Logic Devices. *International Journal of Science and Technology*, 2:8.