EPROM Programmer

Submitted by : Group 35

Manas Lohani
2019B4A70109G

Yashraj Patil
2019B5A31038G

Varada R
2019B4A80795G

Anurag Bajpai
2019B3AA0530G

Joel Jacob
2019B4A80762G

Eshaan Jain
2019B4AA0735G

17 April, 2022

Contents

User requirements and technical specifications

Design a microprocessor-based EPROM Programmer to program 2732.

The technical specifications are as follows:

1. The EPROM can be programmed by applying 25V at VPP and 5V at OE pin.
2. Before the EPROM location is programmed it must be checked for whether it is empty (data in all locations must be FFH if the location is empty). We must check all locations.
3. If not empty an LED labeled not empty should be glowed. Even if one location is not empty, the EPROM is not programmable.
4. If the EPROM is empty, it can be programmed location by location.

Assumptions and Justifications

Justifications

- A set of 4- 7 segment displays is used to show the address currently being written into at all times. Four are used to show the address currently being written into, 2 are used to show the data entered by the user
- Programming a 2732 requires a 25V signal at the Vpp, all the output from the 8255 being 5V, a relay switch is used to obtain a 25V signal at the Vpp pin.

Assumptions

- The EPROM is to be programmed only sequentially.
- Incremental addressing is used for programming the EPROM.
- BACKSPACE removes the complete byte of data.
- Pressing ENTER programs the value on the 7-segment display into the EPROM.
- The user cannot press more than one key at once.
- The user enters the LSB of the data first and then enters the MSB of the data.
- If no data is entered, pressing backspace removes nothing.
- Entering a hex input after entering two nibbles does not overwrite any nibble.
- By default, the data displays FF.
- The complete 4K of data is written one after the other.
- The user uses the device after a short instance, the delay being the time required to read all the memory addresses

Components used with justification wherever required

- **Intel 8086 Microprocessor (1nos)**
- **8284 Clock Generator (1nos)**
- **SP-SPDT-MOM (1 nos.)**- can be used to reset the microprocessor
- **6116 (2 nos.)** - Smallest available RAM chip is 2K and we require odd and even banks.
- **2732 (4 nos.)**- these are used as ROM chips for the microprocessor
- **74LS373 (5 nos.)** - used to demux the address lines from the microprocessor, and to demux, the outputs from the 8255A.
- **74LS245 (2 nos.)** - used to demux the data lines from the microprocessor
- **74LS138 (2 nos.)** -1 is used for memory selection, 1 nos. Is used for I/O chip selection
- **OR gates (10 nos.)** -4 nos. used to generate the strobe signals, the other 6 are used for memory chip selection
- **8255A (5 nos.)**- used for hex keypad interfacing, the enter, and backspace key interfacing, as an input to all the six 7-segment displays
- **SPST-Push Button (18 nos.)**- here, 16 nos. Buttons are used in the hex-keypad and 1 each are used for the “enter” and “backspace” button
- **RELAY SWITCH (1 nos.)** - used to get a 25V signal to be given at Vpp to make the target 2732 as programmable
- **LED-RED (1 nos.)-** used to show PROGRAMMING FAILED
- **LED-YELLOW (1 nos.)** - used to show a NON-EMPTY ROM
- **LED-GREEN (1 nos.)** - used to show PROGRAMMING FAILED
- **7 SEG-COM-AN-BLUE (6 nos.)** - Four are used to show the address currently being written into, 2 are used to show the data entered by the user.

Address Map

Memory Map

CHIP	ADDRESS
ROM 1	00000H - 01FFFH
ROM 2	FE000H - FFFFFH
RAM 1	02000H - 02FFFH

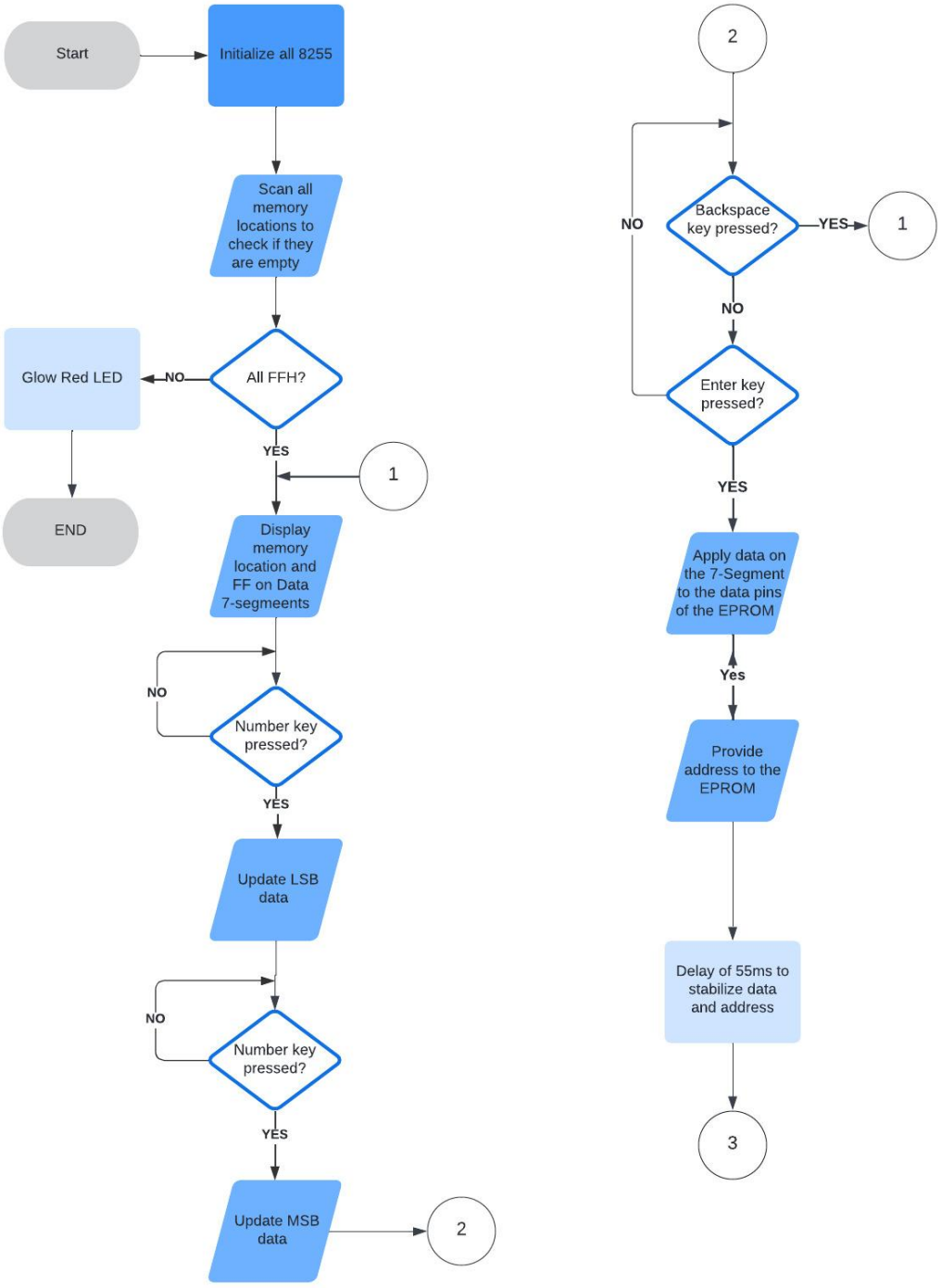
I/O Map

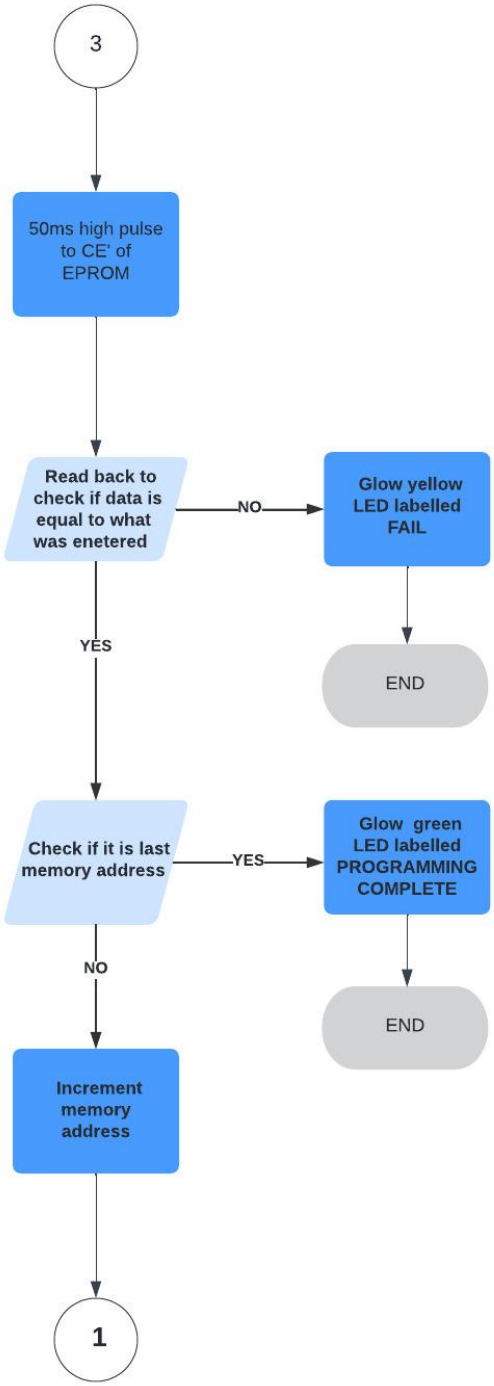
CHIP	ADDRESS
8255A -1	10H
8255A -2	20H
8255A -3	30H
8255A -4	40H

Design

Complete design is shown with proper labelling in attached file.

Algorithm


Start
Initialize all 8255
Scan all memory locations to check if they are empty
All FFH?
NO
Glow Red LED
END
YES
1
Display memory location and FF on Data 7-segmeents
NO
Number key pressed?
YES
Update LSB data
NO
Number key pressed?
YES
Update MSB data
2
2
Backspace key pressed?
NO
YES
1
NO
Enter key pressed?
YES
Apply data on the 7-Segment to the data pins of the EPROM
Yes
Provide address to the EPROM
Delay of 55ms to stabilize data and address
3


3
50ms high pulse to CE' of EPROM
Read back to check if data is equal to what was enetered
NO
Glow yellow LED labelled FAIL
END
YES
Check if it is last memory address
YES
Glow green LED labelled PROGRAMMING COMPLETE
END
NO
Increment memory address
1

List of Attachments

Reference Data Sheets

- **6116 :- https://ece-classes.usc.edu/ee459/library/datasheets/6116SA.pdf**
- **2732 :- https://www.futurlec.com/Memory/2732_Datasheet.shtml**
- **74LS138 :- https://www.futurlec.com/74LS/74LS138.shtml**
- **74LS373 :- https://www.futurlec.com/74LS/74LS373.shtml**
- **74LS245 :- https://www.futurlec.com/74LS/74LS245.shtml**
- **8255A :- https://www.datasheet-pdf.info/attach/1/5514326703.pdf**
- **7 SEG-COM-AN-BLUE :- https://www.display-leds.com/2-3-56-8-mm-numeric-LED-Display-with-common-pin-3-and-8-pd199204.html**