

# Spirit Level Reaction Time Tester

---

## Submitted by Group 25

Ameya Thete, 2018B5A70885G ([f20180885@goa.bits-pilani.ac.in](mailto:f20180885@goa.bits-pilani.ac.in))

Gourav Saha, 2018B1AA0639G ([f20180639@goa.bits-pilani.ac.in](mailto:f20180639@goa.bits-pilani.ac.in))

Ishika Parihar, 2018B3AA0738G ([f20180738@goa.bits-pilani.ac.in](mailto:f20180738@goa.bits-pilani.ac.in))

Ishita Jaiswal, 2018B5A80613G ([f20180613@goa.bits-pilani.ac.in](mailto:f20180613@goa.bits-pilani.ac.in))

Pranav Ballaney, 2018B1A70635G ([f20180635@goa.bits-pilani.ac.in](mailto:f20180635@goa.bits-pilani.ac.in))

Saransh Gokhale, 2018B3A30033G ([f20180033@goa.bits-pilani.ac.in](mailto:f20180033@goa.bits-pilani.ac.in))

Date: 18 April 2021

# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>List of Tables</b>	<b>3</b>
<b>List of Figures</b>	<b>3</b>
<b>User Requirements &amp; Technical Specifications</b>	<b>4</b>
User Requirements	4
Working of the device	4
User Interface	4
Detailed Working	4
<b>Assumptions and Justifications</b>	<b>4</b>
<b>Components</b>	<b>5</b>
<b>Address Map</b>	<b>6</b>
Memory Addresses	6
IO Addresses	6
<b>Design</b>	<b>7</b>
Block Diagram	7
LED to Sobriety Score Map	7
States within the Program	8
Timing Signal Generation	9
Debouncing Button Presses	10
<b>Flow Charts</b>	<b>11</b>
Main Program	11
Start Button Interrupt Routine	12
Stop Button Interrupt Routine	13
Counter Interrupt Routine	14
<b>Variations in Proteus Implementation with Justification</b>	<b>15</b>
<b>Firmware</b>	<b>15</b>
<b>List of Attachments</b>	<b>15</b>

## List of Tables

1. Memory Address Map	6
2. I/O Address Map	6
3. Mapping of the Sobriety Score to the LED count	7

## List of Figures

1. A high-level block diagram of the major components used in this Design	7
2. States within the Program	8
3. Generating Timing Signals using an 8254 Programmable Interval Timer	9
4. Flowchart outlining the Main Program	11
5. Flowchart for the ISR triggered by the Start Button	12
6. Flowchart for the ISR triggered by the Stop Button	13
7. Flowchart for the ISR triggered by the 8254 counter	14

# User Requirements & Technical Specifications

## User Requirements

To design a system that tests for the intoxication level of a person by measuring their reaction time.

## Working of the device

- When a START button is pressed, the device waits for a random time interval (between four and eight seconds) and then begins incrementing LEDs on a bar graph display such that they appear to 'rise' upwards.
- When the user sees the LEDs moving, they press a STOP button as soon as possible – the earlier the button is pressed, the fewer LEDs that are lit.
- The entire bar graph will sweep to the top in 0.4 seconds (with 50ms between LED steps).

## User Interface

- Features two push buttons – Start and Stop.
- Nine LEDs on a bar graph.
- A seven-segment display that displays the sobriety level of the user.

## Detailed Working

- When the START button is pressed, a random time delay is generated, after which the bottom-most LED lights up.
- After a 50ms time interval, the bottom-most and the next-highest LED both light up. After another 50ms delay, the three bottom LEDs light up. The process continues until either the STOP button is pressed (in which case the LEDs stop rising) or when all the LEDs are lit.
- The program must also check the STOP button just before lighting the bottom LED to ensure the user isn't cheating by holding the STOP button continually. If the user tries to cheat, blink all LEDs several times and simply go back to the start (waiting once again for the START button).
- The sobriety of a person on a scale of 1-5 (1 indicating maximum intoxication) has to be displayed on a seven-segment display.

To generate a random number, we run a timer continually from an internal clock. When the START button is pressed, simply latch the value of the timer (this process is essentially random). That number can be used to generate a random delay.

## Assumptions and Justifications

SPST switch bounce time is  $< 20\text{ms}$  (usually holds as most SPST bounce times are  $\leq 10\text{ms}$ ).

# Components

1. **8086 Microprocessor:** Used to coordinate I/O devices, run the firmware, *et cetera*.
2. **8284 Clock Oscillator:** Generates CLK signals for 8086 and 8254.
3. **8254 Programmable Interval Timer:** Used to generate the random number, the initial random delay using the random number as a seed, and the 50ms delay.
4. **8255 Programmable Peripheral Interface:** Used to interface the LEDs, switches and the seven-segment display to 8086.
5. **8259 Programmable Interrupt Controller:** Used to generate the interrupts when the switches are pressed.
6. **Start and Stop buttons:** SPST Momentary Action.
7. **Common anode seven-segment display.**
8. **7447 BCD to seven-segment converter.**
9. **9 LEDs:** Form the bar graph display
10. **2716 ROM:** 4 x 2KB ROM Chips. Store the IVT, firmware and reset routines.
11. **6116 RAM:** 2 x 2KB RAM Chips. Used for temporary variables and stack storage.
12. **LS138:** Generating CS' signals for memory and I/O devices.
13. **LS373:** Octal D latches for demultiplexing address lines.
14. **LS244:** Octal buffer for control signals.
15. **LS245:** Octal Bus transceiver for data lines.
16. **Logic gates and resistors as needed.**

# Address Map

We have 2 ROMs and 1 RAM. The smallest available memory chips for both RAM and ROM are 2KB, and since banking has to be implemented to interface memory, we will require 2 chips for each memory segment. Hence in total, we require  $2 \times 2\text{KB} + 2 \times 2\text{KB} + 2 \times 2\text{KB} = 12 \text{ KB}$  of memory in total.

## Memory Addresses

Memory Segment	Starting Address	Ending Address	Size	Purpose
ROM 1	00000h	00FFFh	4KB	IVT and program
RAM 1	01000h	01FFFh	4KB	Scratch memory
ROM 2	FF000h	FFFFFh	4KB	Reset

**Table 1:** Memory Address Map.

## IO Addresses

Only even addresses are used.

Component	Starting Address	Ending Address
8255	00h	06h
8254	08h	0Eh
8259	10h	12h

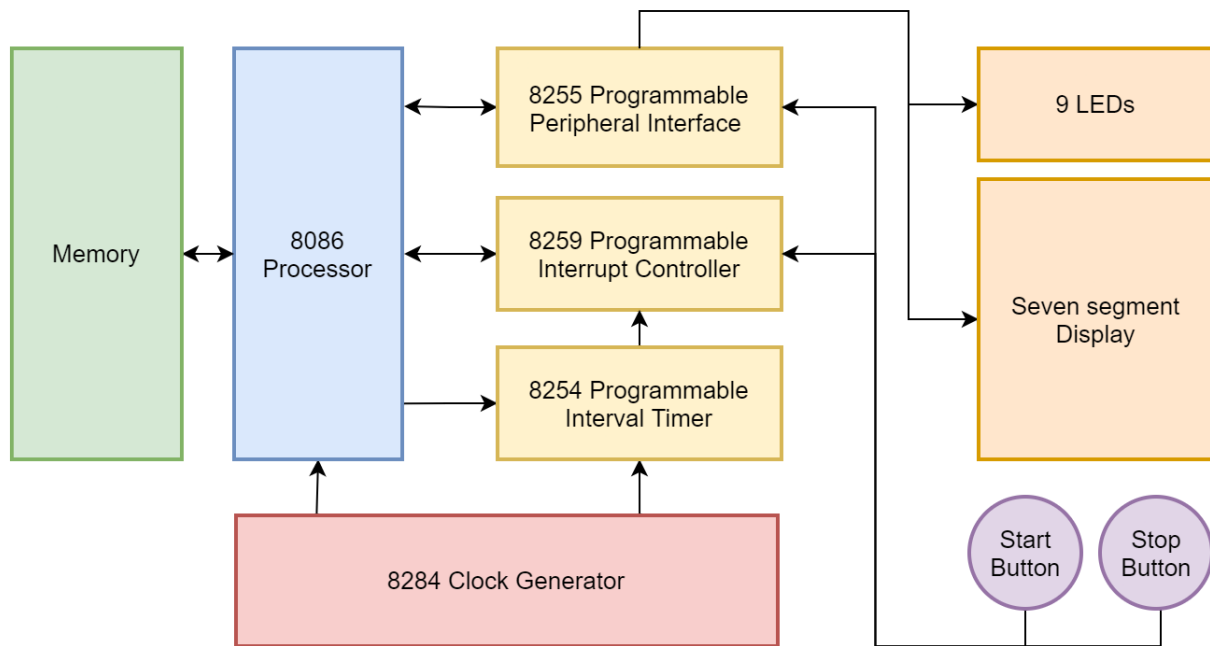
**Table 2:** I/O Address Map

This space has been intentionally left blank

# Design

A complete document indicating the connections for each component in our on-paper design has been provided with this report, however, we have also sought to explain some of the major components of our design here for better clarity and completeness.

## Block Diagram



**Figure 1:** A high-level block diagram of the major components used in this Design.

## LED to Sobriety Score Map

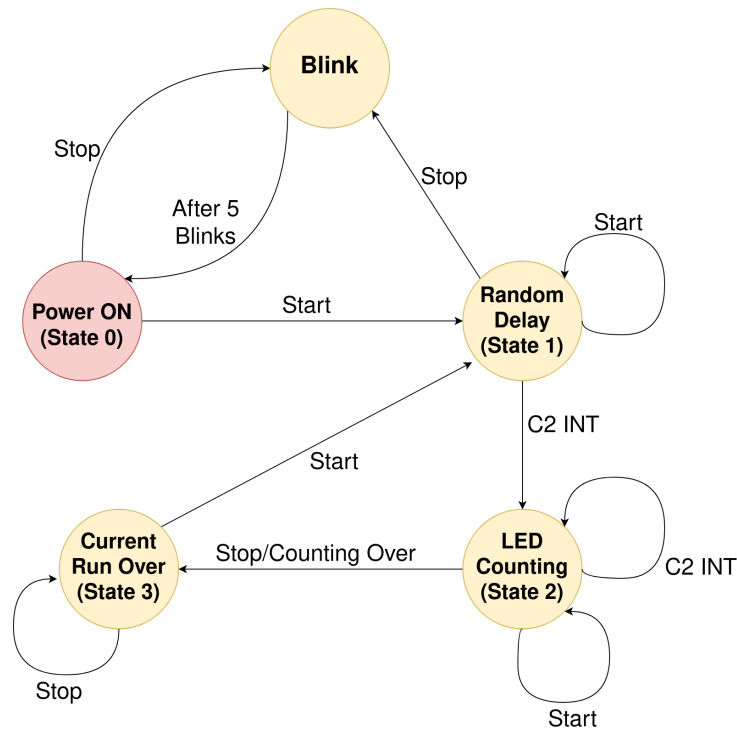
We have mapped the number of active LEDs to the sobriety score of the user as tabulated below. The sobriety of a person is measured on a scale that goes from 1 to 5, with 1 indicating maximum intoxication. The delay column indicates the time elapsed after the first LED glows following a random delay of four to eight seconds.

If the number of LEDs active when the user presses the STOP button is lesser than or equal to 3, then the score is 5; if the number of active LEDs is greater than 3 and lesser than or equal to 5, the score is 4, and so on.

Delay	Number of LEDs active	Score
100ms	3	5
200ms	5	4
250ms	6	3
300ms	7	2
400ms	9	1

**Table 3:** Mapping of the Sobriety Score to the LED count.

## States within the Program



**Figure 2:** States within the Program.

We keep track of the states shown here in the program and modify the behaviour of the Start and Stop buttons accordingly.

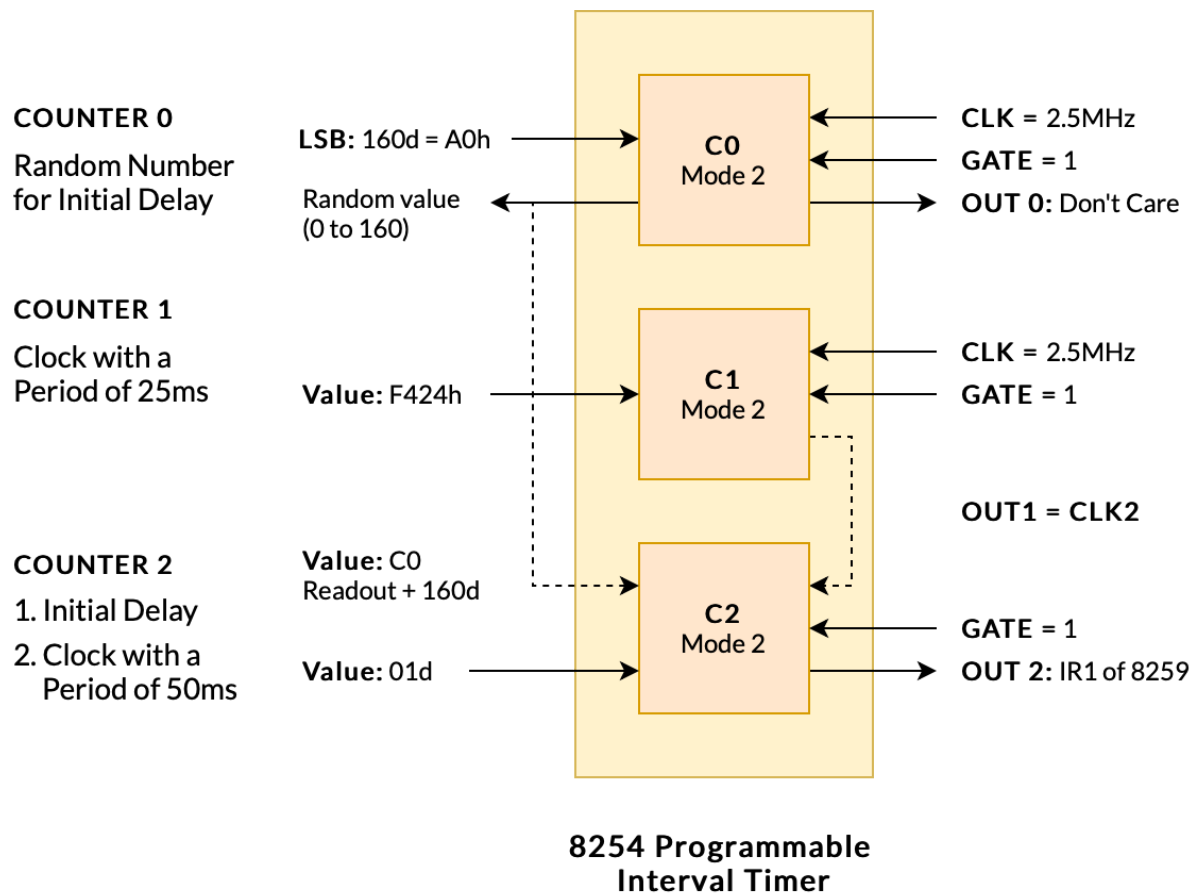
The microprocessor starts out in state 0, or the Power ON state, which consists of an infinite loop in the main program waiting for interrupts. Pressing the stop button when the system is in state 0 causes the cheating prevention mechanism to activate and all LEDs blink 5 times, after which the system is ready to receive another input. When the start button is pressed, the system moves to state 1 where a counter is latched to generate a random delay. In this state, pressing the start button again has no effect on the system; however, pressing the stop button causes the system to move into the blinking state described earlier.

After waiting for a period of time between four to eight seconds, the system lights the first LED and moves to state 2 (LED counting state). Here pressing the start button, again, has no effect. The system continues to receive an interrupt every 50ms to light an additional LED until either all LEDs are lit or the user has pressed the stop button. Pressing the stop button halts the LED progression and moves the system into state 3 (Current run over state), where the sobriety score of the user is calculated according to the map previously described and displayed on the seven-segment display.

If the stop button is not pressed, the system automatically moves into state 3 after all 9 LEDs are lit and displays '1' on the display (indicating maximum intoxication). To reuse the device, the user must press the Start button once again, which causes the system to move to State 1 and wait for a random delay.



## Timing Signal Generation



**Figure 3:** Generating timing signals using an 8254 Programmable Interval Timer.

To generate timing signals for our design, we use an 8254 Programmable Interval Timer as shown above.

Counter 0 cycles from 0d to 160d throughout the run and when a random number is required for the initial delay, we read the value using the data lines.

Counter 1 generates a signal with a time period of 25ms which is used to create the initial delay as well as a signal with a time period of 50ms pulse for turning ON successive LEDs.

Counter 2 takes the 25ms signal from Counter 1 as its CLK and is programmed twice to achieve the following functionality:

1. To generate the initial delay, C2 is loaded with the random value from C0 (0 to 160d) + 160d, and then connected with IR1 of the 8259. This allows C2 to generate a delay between 4 to 8 seconds.
2. To generate a 50ms pulse for turning ON successive LEDs, C2 is loaded with a value of 1. In this mode, C2 generates a regular interrupt on IR1 of the 8259.

## Debouncing Button Presses

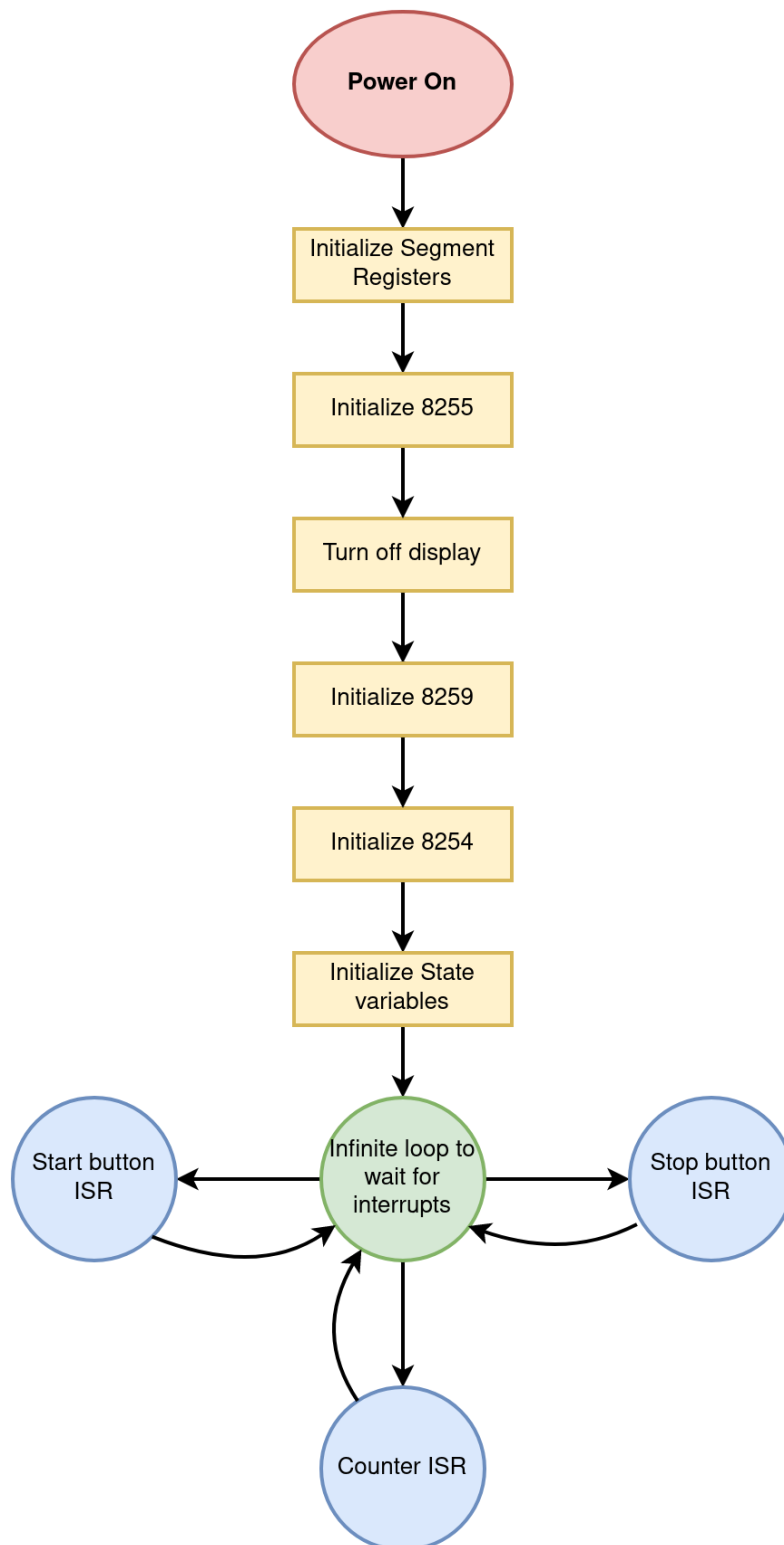
Start and Stop buttons are connected to IR ports of the 8259 as well as the lower port C of the 8255. When an interrupt is triggered by a button press, we perform the following debouncing routine.

Initially, the IR port corresponding to that button is disabled to prevent subsequent interrupts. Now, the processor waits for 20ms and then reads the Start button from port C. If the button press is detected on port C as well, the IR routine proceeds, but otherwise, the IR pin is re-enabled and the ISR ends.

This space has been intentionally left blank

# Flow Charts

## Main Program



**Figure 4:** Flowchart outlining the Main Program.

## Start Button Interrupt Routine

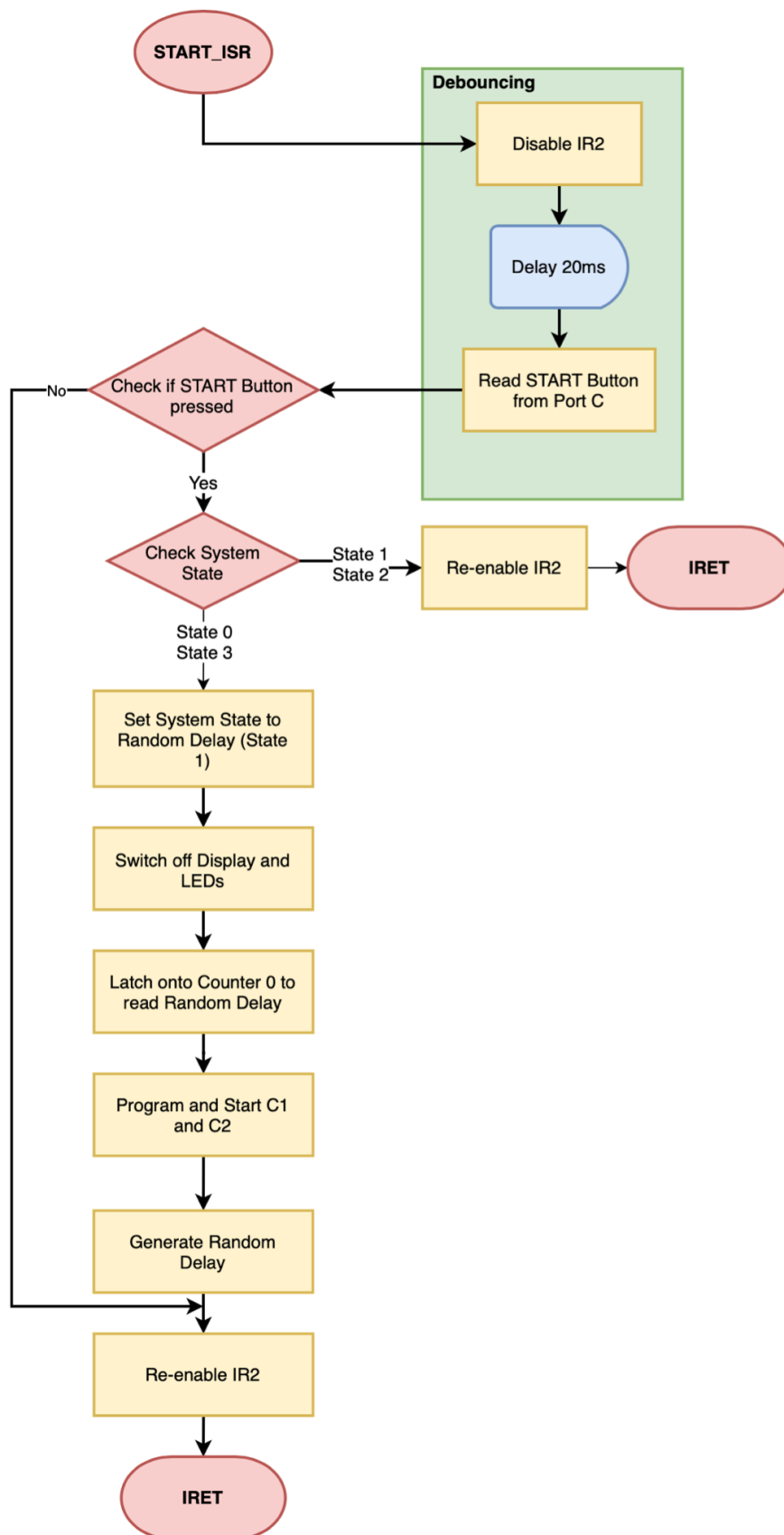


Figure 5: Flowchart for the ISR triggered by the Start Button.

## Stop Button Interrupt Routine

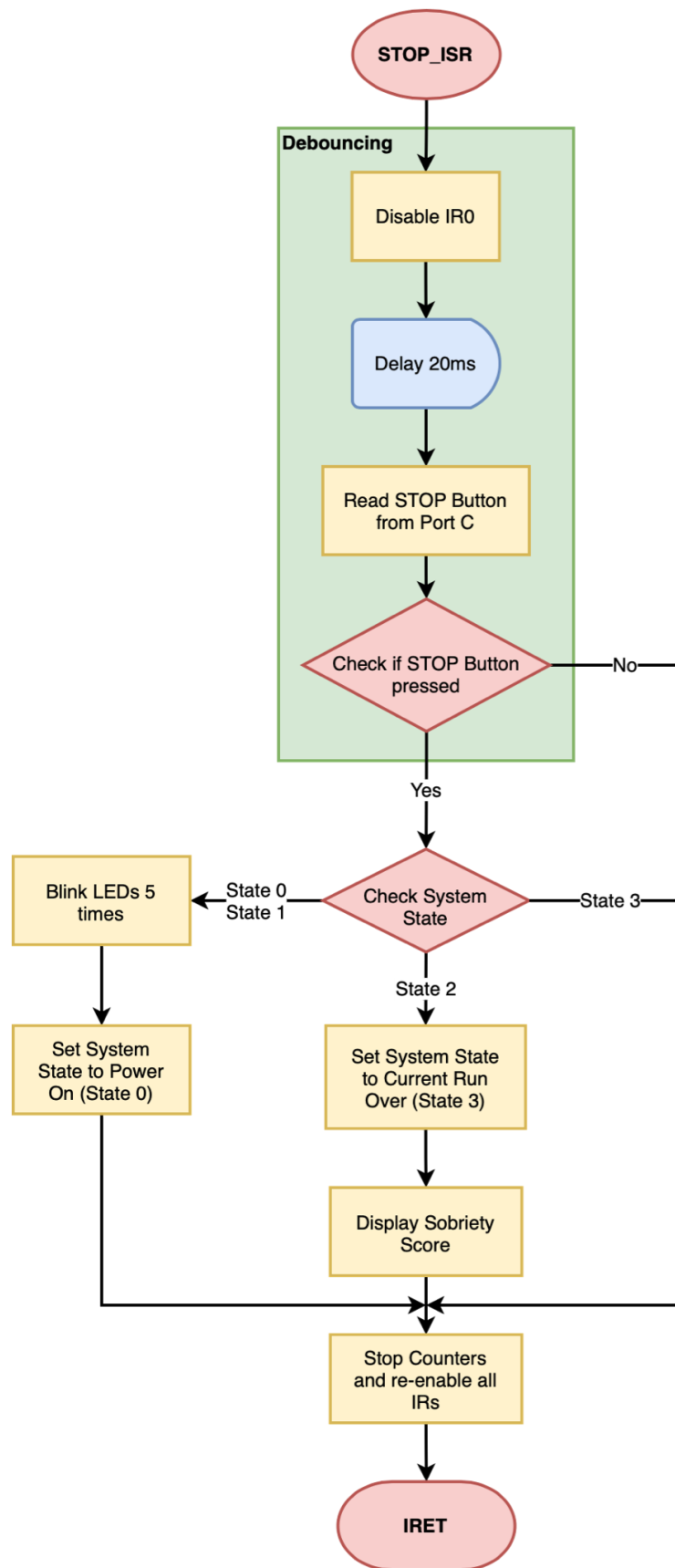
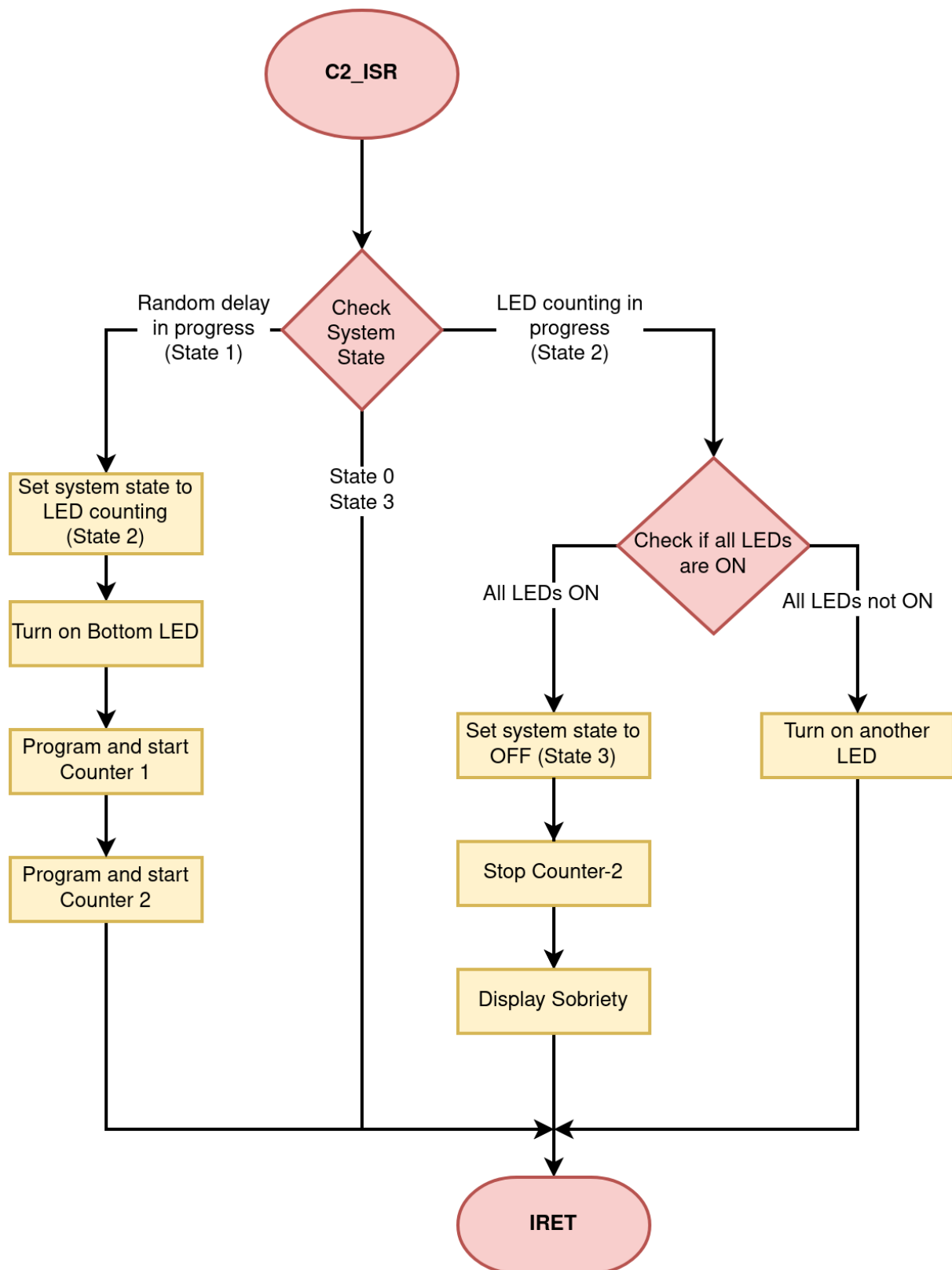


Figure 6: Flowchart for the ISR triggered by the Stop Button.

## Counter Interrupt Routine



**Figure 7:** Flowchart for the ISR triggered by the 8254 Counter.

# Variations in Proteus Implementation with Justification

1. Since 8254 is not available in Proteus, 8253 has been used instead.
2. Proteus does not have 8284, hence the 8086 clock has been set using the configuration settings of the 8086. The same approach has been adopted for the 8253 clock inputs.
3. Control signals have been generated using gates, and not explicitly as shown in the hardware design document.
4. A 2732 ROM has been used as 2716 is not available. Since Proteus allows changing the RESET address, ROM2 has been done away with. Hence, in the simulation, the memory map changes slightly to
  - a. ROM: 00000h - 01FFFh
  - b. RAM: 02000h - 02FFFh

No ROM at the end would be required as the RESET location in the simulation is at 00000h.

5. SPDT momentary switches are used for the start and stop buttons as SPST momentary switches in Proteus were malfunctioning.
6. Since the RESET address was set to 00000h, ISR0 was unavailable as that memory location contained the jump statement to the initialization segment of the code, hence in the Proteus simulation, IRO was left unused and devices were connected IR1 onwards.

## Firmware

Implemented using emu8086 and provided with this report.

## List of Attachments

1. A complete hardware document describing connections to all components (G25\_Hardware\_Design.pdf).
2. Hardware Manuals and Datasheets
  - a. IC 7447 BCD-to-seven-segment converter
  - b. SPST momentary switches
  - c. 7-segment display
3. Proteus 8 file: G25\_spirit\_level.pdsprj
4. emu8086 ASM file: G25\_spirit\_level.asm
5. Binary file after assembly: G25\_spirit\_level.bin