ECEN-406

7-Segment Display implementation using the 8086 Microprocessor

Supervised By: Dr. Ahmed Soltan & Eng Habiba Mahmoud

Table of Contents

Table Of Figures

Introduction:

The Intel 8086 Microprocessor is a very important piece of hardware that can be used in implementing many ideas , it gives the engineer a good understanding of how to use many types of memory , how to use peripherals which also gives them a taste of how an actual computer might operate , in this report the address pins especially the ones that work in MINIMUM mode of the 8086 microprocessor are explained along with all the external hardware that was used in order to bring this project to life ,and show how a simple 60 second timer can be implemented and what type of memory and hardware latches along with peripherals are needed, a block diagram of the system along with screenshots of the actual system can be seen by the end of the report , and a simple memory map sketch was also included to explain how the memory system is going to work.

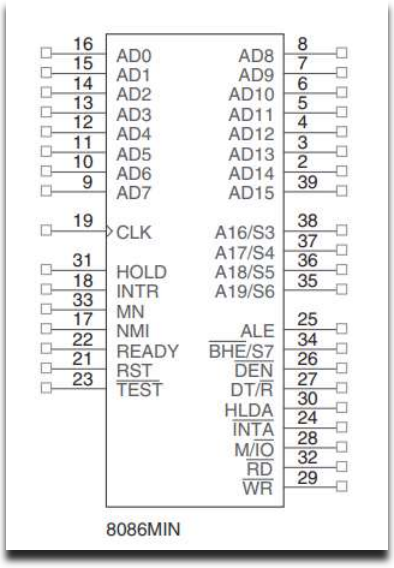
Methodology:

1) **Gathering Data**:

- Through Some research and using a qualitative data gathering method the team managed do unfold important data that was a big step in finishing this project.
- The team managed to find and understand how to implement several important latches, memory devices, address decoders, and peripherals, most importantly, the 8086 Microprocessor.
- Also understanding and knowing which specific instructions that need to be written for this project to work was a crucial point.

2) Understanding Gathered Data

- **The 8086 Microprocessor, Pinout Mode of operation**
  - **Pinout:**



  - **Pins Used (Minimum Mode):**

- Minimum mode operation is obtained by connecting the mode selection pin to +5.0 V
- Minimum mode operation is the least expensive way to operate the 8086/8088 microprocessors.
- It costs less because all the control signals for the memory and I/O are generated by the microprocessor

1. **the AD0 to AD15:** pins on the 8086 microprocessors consist of lower 16 bits of the 20-bit external address bus. They carry address information used by the CPU to access memory and I/O devices in real mode, enabling the processor to address up to 64 KB of memory directly.
2. $\overline{\mathbf{RD}}$: Whenever the read signal is a logic 0, the data bus is receptive to data from the memory or I/O devices connected to the system.
3. $\overline{\mathbf{WR}}$: The write line is a strobe that indicates that the 8086 is outputting data to a memory or I/O device. During the time that the is a logic 0, the data bus contains valid data for memory or I/O.
4. **READY**: The READY input is controlled to insert wait states into the timing of the microprocessor. If the

READY pin is placed at a logic 0 level, the microprocessor enters into wait states and remains idle. If the READY pin is placed at a logic 1 level, it has no effect on the operation of the microprocessor.

5. **RESET** : The reset input causes the microprocessor to reset itself if this pin is held high for a minimum of four clocking periods.

6. **M/$\overline{\text{IO}}$** : pin selects memory or I/O. This pin indicates that the microprocessor address bus contains either a memory address or an I/O port address.

7. **DT/$\overline{\text{R}}$** : The data transmit/receive signal shows that the microprocessor data bus is transmitting **(DT/$\overline{\text{R}}$ = 1)** or receiving **(DT/$\overline{\text{R}}$ = 0)** data. This signal is used to enable external data bus buffers.

8. **DEN**: Data bus enable activates external data bus buffers.

9. **ALE** : Address latch enable shows that the 8086/8088 address/data bus contains address information. This address can be a memory address or an I/O port number.

10. **MN/$\overline{\text{MX}}$** : The minimum/maximum mode pin selects either minimum mode or maximum mode operation for the microprocessor. If minimum mode is selected, the MN/ pin must be connected directly to +5.0 V.

3) Implementing gathered Data, & Explaining Hardware, Software used:

- Software Tools Used Narrowed Down to Proteus, because it has its own built in Compiler to write, test x86 assembly language and run directly on the 8086 microprocessors, along with its external Hardware.

- Speaking of Hardware:
- We used 2 (7 Segment Common Cathode Displays) to display our seconds on

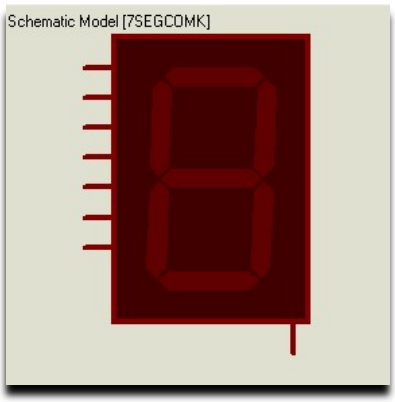


Figure 1(The 7 Segment Display)

- These Are the needed binary values to be able to display our desired numbers on the 7-segment display

Decimal	Binary DCBA	7 Segment Code a b c d e f g
0	0000	1 1 1 1 1 1 0
1	0001	0 1 1 0 0 0 0
2	0010	1 1 0 1 1 0 1
3	0011	1 1 1 1 0 0 1
4	0100	0 1 1 0 0 1 1
5	0101	1 0 1 1 0 1 1
6	0110	0 0 1 1 1 1 1
7	0111	1 1 1 0 0 0 0
8	1000	1 1 1 1 1 1 1
9	1001	1 1 1 0 0 1 1

Figure 2 (7 segment display Binary Values)

- The 8255A Peripheral:

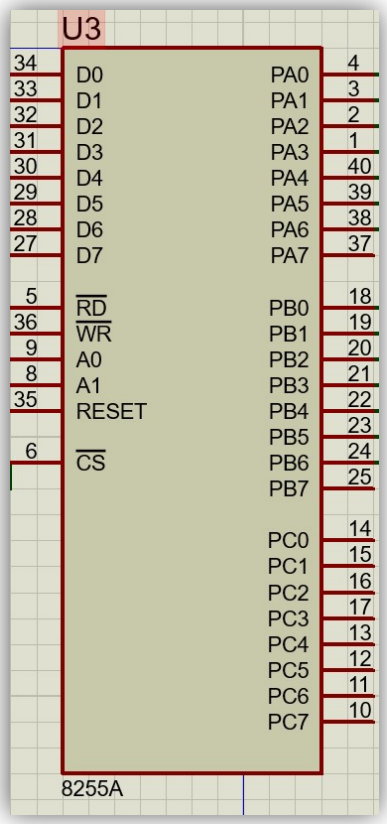


Figure 3 (8255A programmable peripheral interface (PPI))

- The 8255A is a general purpose programmable I/O device designed to be used with almost any microprocessor.
- It consists of three 8-bit bidirectional I/O ports (24I/O lines) which can be configured as per the requirement.
- 8255A has three ports, i.e., PORT A, PORT B, and PORT C.

- Port A contains one 8-bit output latch/buffer and one 8-bit input buffer.
- Port B is similar to PORT A.

- Port C can be split into two parts, i.e. PORT C lower (PC0-PC3) and PORT C upper (PC7-PC4) by the control word.

- **Operating Modes (This Project is using Mode 0):**
  - **Mode 0** − In this mode, Port A and B is used as two 8-bit ports and Port C as two 4-bit ports. Each port can be programmed in either input mode or output mode where outputs are latched and inputs are not latched. Ports do not have interrupt capability.
  - **Mode 1** − In this mode, Port A and B is used as 8-bit I/O ports. They can be configured as either input or output ports. Each port uses three lines from port C as handshake signals. Inputs and outputs are latched.
  - **Mode 2** − In this mode, Port A can be configured as the bidirectional port and Port B either in Mode 0 or Mode 1. Port A uses five signals from Port C as handshake signals for data transfer. The remaining three signals from Port C can be used either as simple I/O or as handshake for port B.

- The 2018 (2KX8) SRAM:

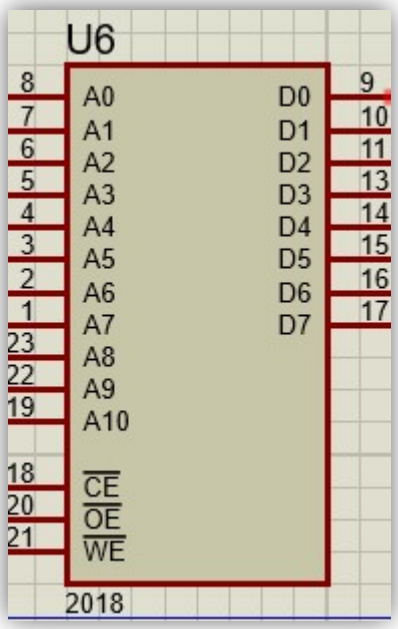


Figure 4(2k x 8 SRAM)

- This Sram can hold up to 2K bytes of memory
- The Pins on the left (A0-10) are used for addressing
- The pins on the Right (D0-7) are used for data movement writing and reading data.
- The CE pin is used to select and enable this Chip
- OE(Bar) enables reading
- WE(Bar) enable writing

- The 74LS373 address latches

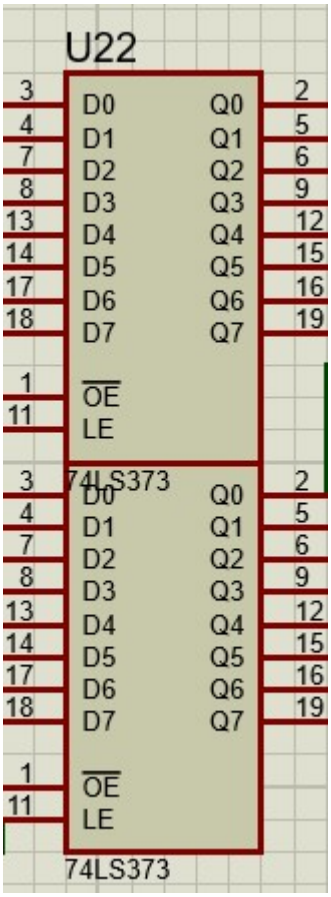


Figure 5(Two 74LS373 Latches)

- When a microprocessor needs to access memory or communicate with other devices, it generates memory or I/O addresses. Address latches temporarily store these addresses, stabilizing them for a brief period, ensuring that they remain available and unchanged while the system completes necessary operations.

- The 74LS245 Latch for Data Bus

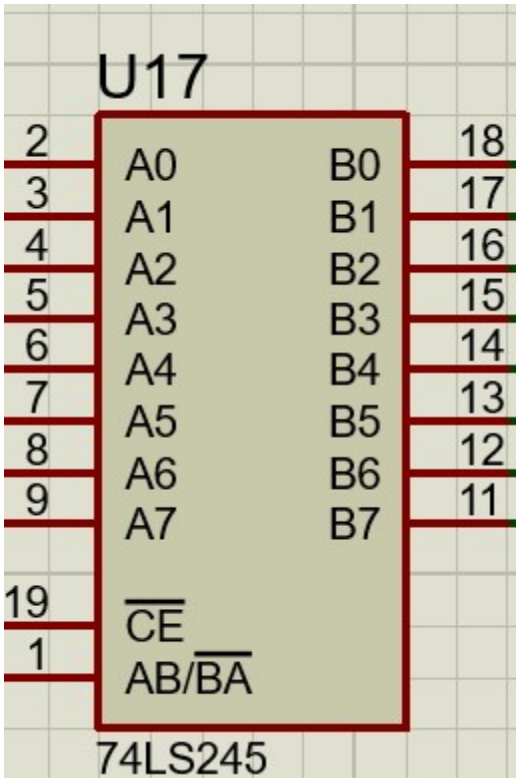


Figure 6 (The 74LS245 Latch)

- Data latches store binary information (0s and 1s) temporarily. When the microprocessor or another device needs to transmit or receive data, data latches can hold this information stable for a short period.

- The 74ALS139 Decoder

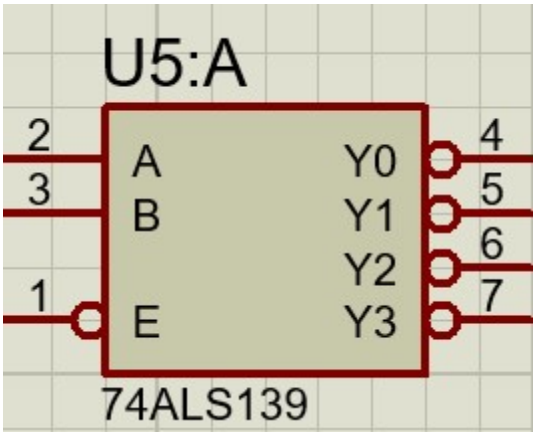


Figure 7(The 74ALS139(2 to 4) Address Decoder)

- In systems with multiple memory devices (such as RAM, ROM, or other memory units) that the microprocessor needs to access, an address decoder interprets the address bus signals to select the appropriate memory bank. It enables the microprocessor to access the desired memory location without disturbing other areas.

- The 74LS244

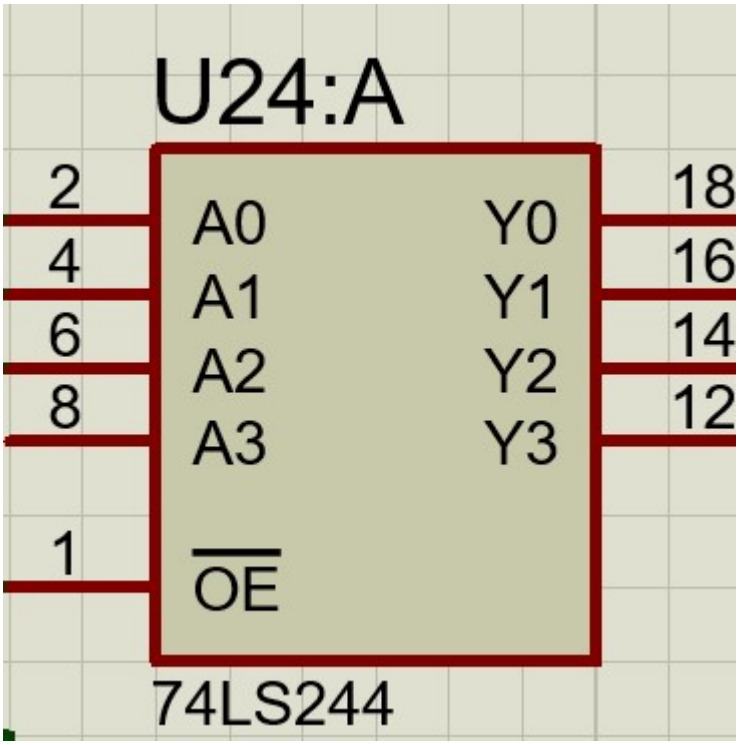


Figure 8(The 74LS244 Latch)

- Another latch that was used to store the values from:
- RD(Bar)
- WR(Bar)
- M/IO(Bar)

- **The Memory Map:**

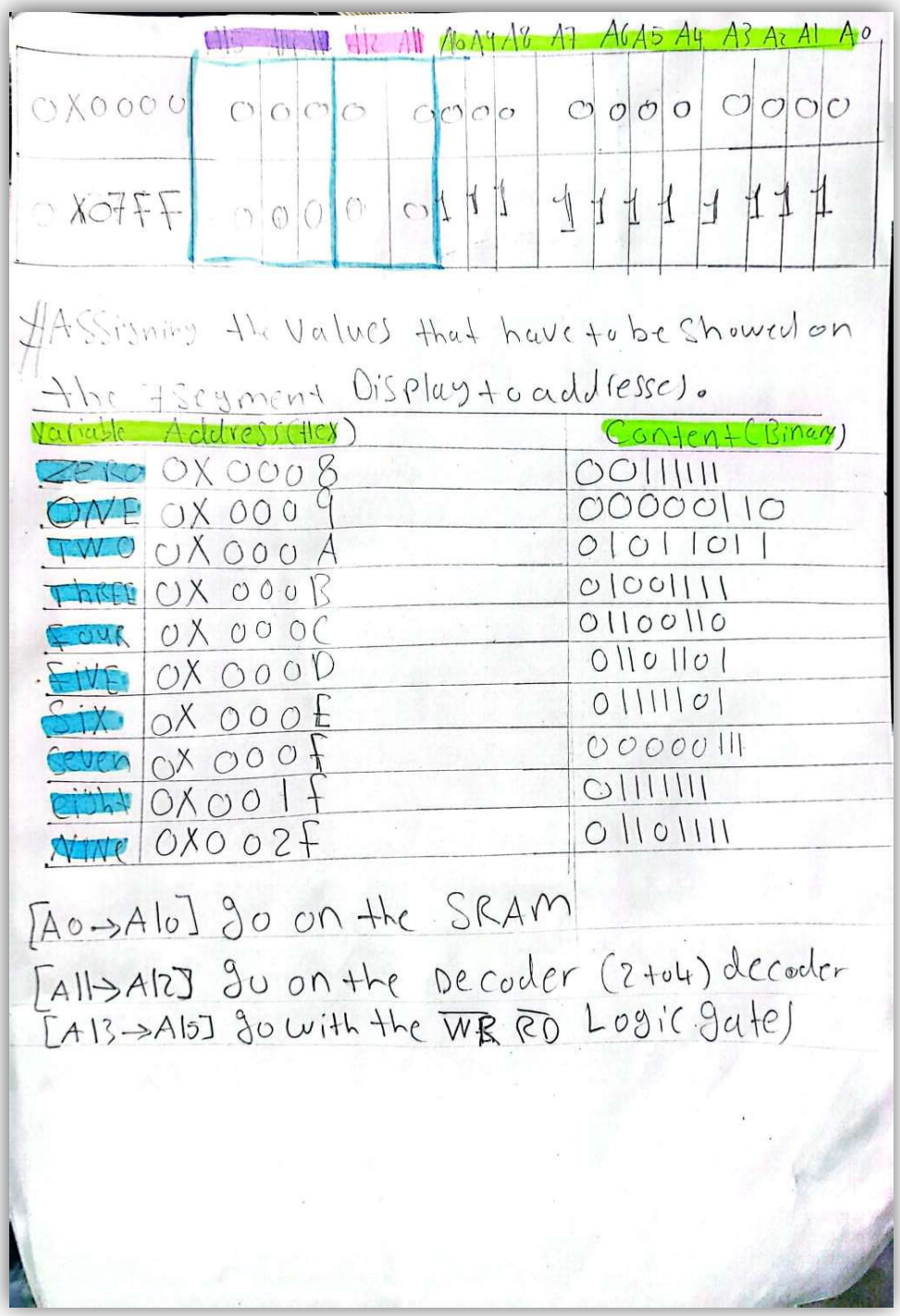


Figure 9(Sketch of the applied memory map concept)

- **Each Value that is to be showed on the 7 Segment display has a binary value for it that is saved on a specific address in our memory**

- **From 0x0008 to 0x002F (These are the used addresses not the entire memory)**
- **As seen from the figure the entire memory spans from 0x0000 to 0x07FF**
- **The address pins used to store and write in the memory are from A0 to A10**
- **The address pins used to enable and select the Chip using our 2 to 4 decoders are A11 to A12**
- **The address pins used to select the mode of operation WR(Bar) or RD(Bar)**
- **Are A13 to A15 as seen in the upcoming figure**

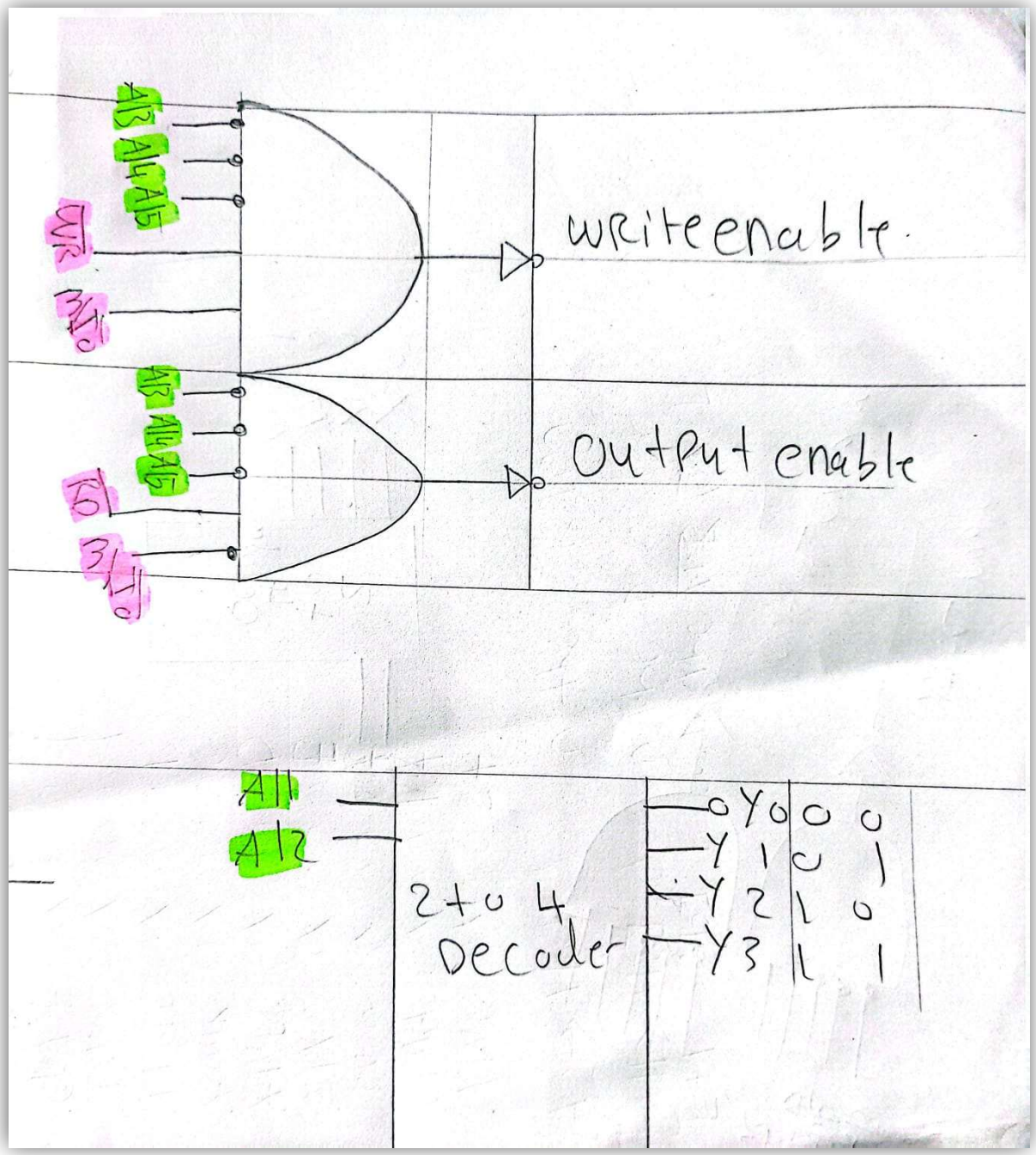


Figure 10(Sketch of the 2 to 4 decoder and the logic circuits used to choose the reading or writing mode for memory)

The Block Diagram and Flowchart Visualization:

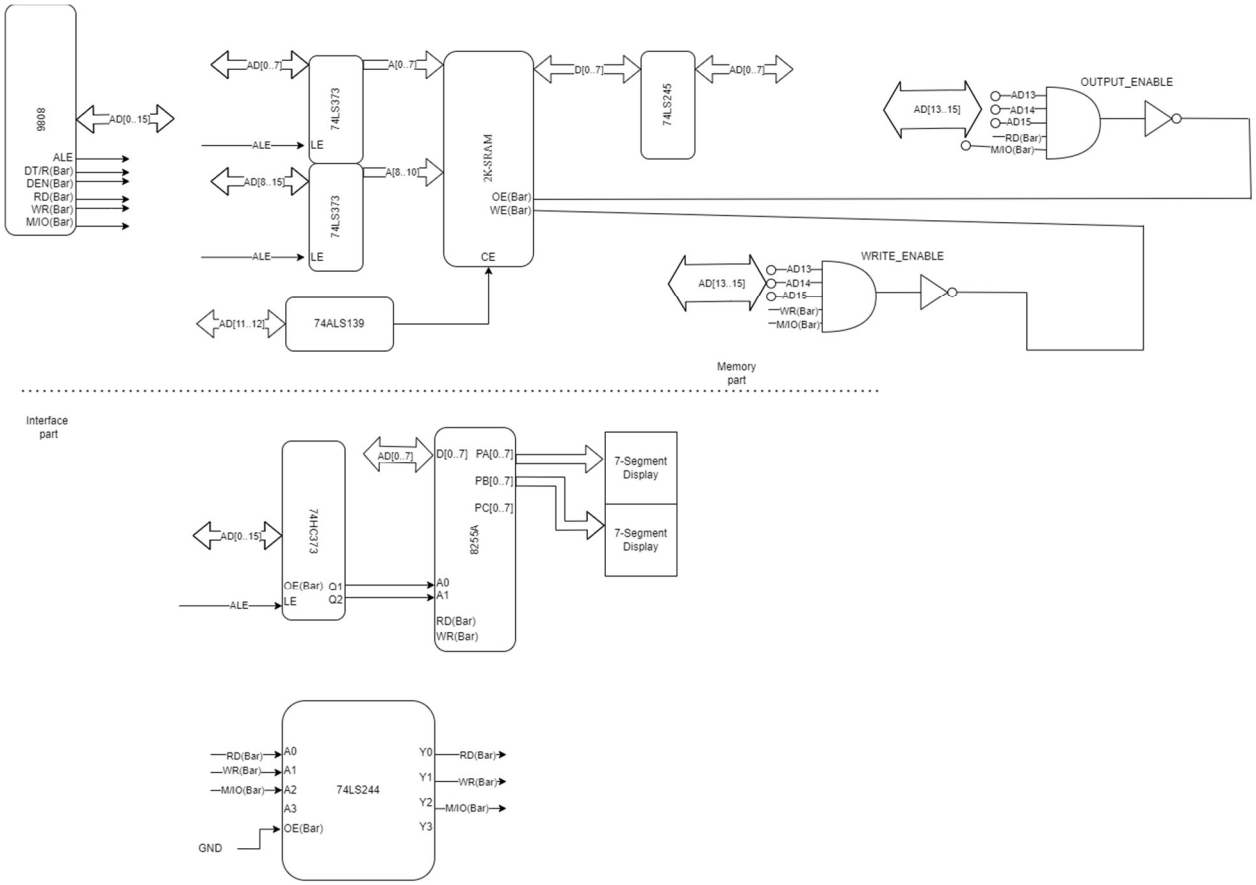


Figure 11(Block Diagram Depicting the Whole System)

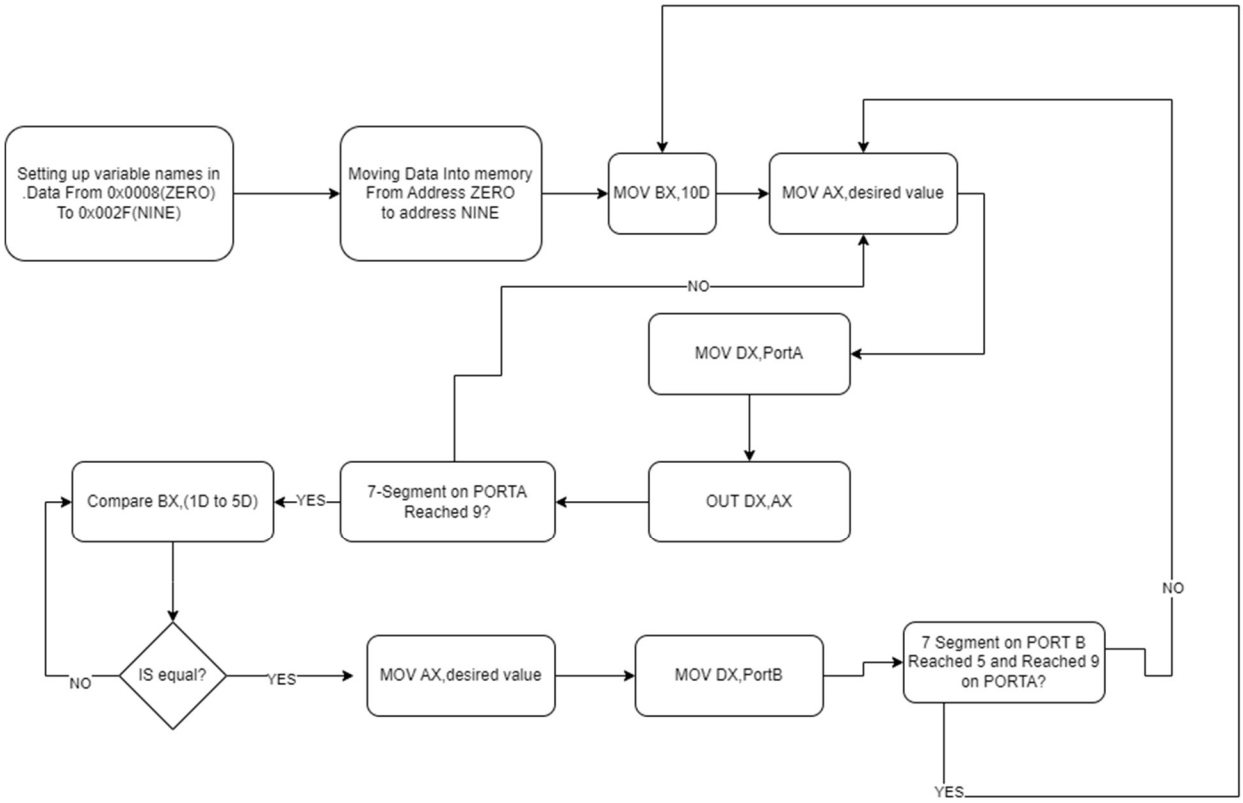

Setting up variable names in .Data From 0x0008(ZERO) To 0x002F(NINE)
Moving Data Into memory From Address ZERO to address NINE
MOV BX,10D
MOV AX,desired value
NO
MOV DX,PortA
OUT DX,AX
7-Segment on PORTA Reached 9?
YES
Compare BX,(1D to 5D)
NO
IS equal?
YES
MOV AX,desired value
MOV DX,PortB
7 Segment on PORT B Reached 5 and Reached 9 on PORTA?
NO
YES

Figure 12

Results:

After Connecting the Whole system, Writing the code, and debugging

These were the outcomes:

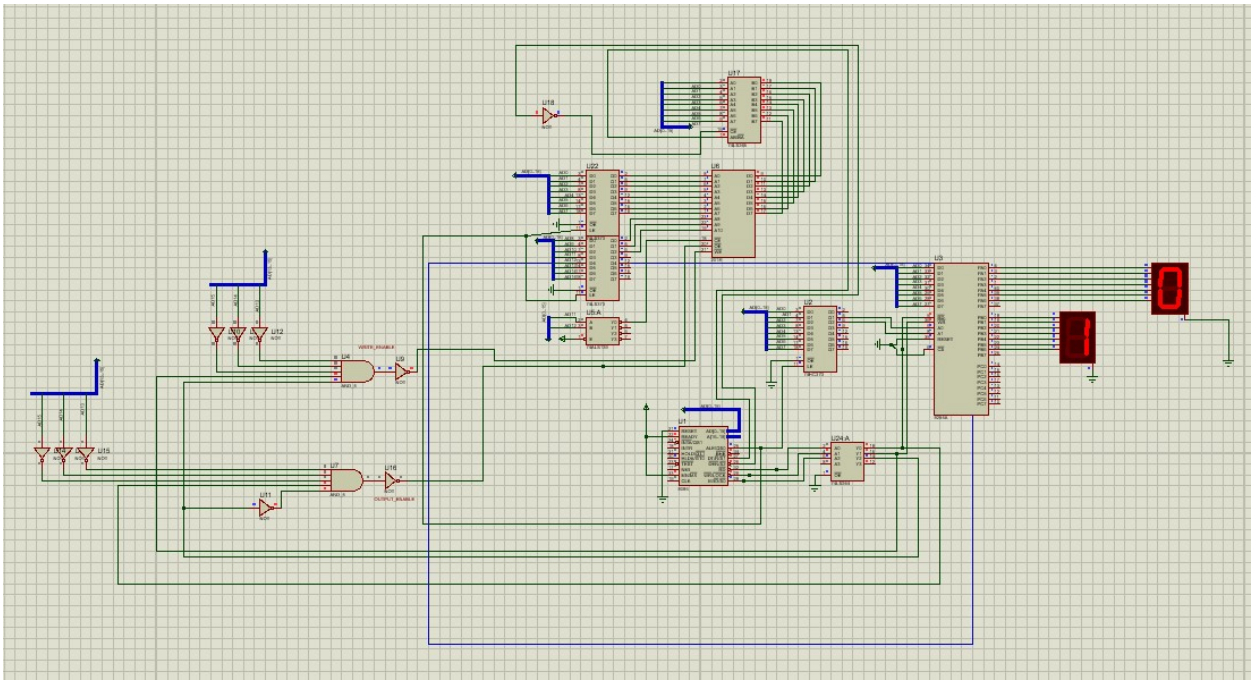


Figure 13(Screenshot of the Actual System at 10 seconds time)

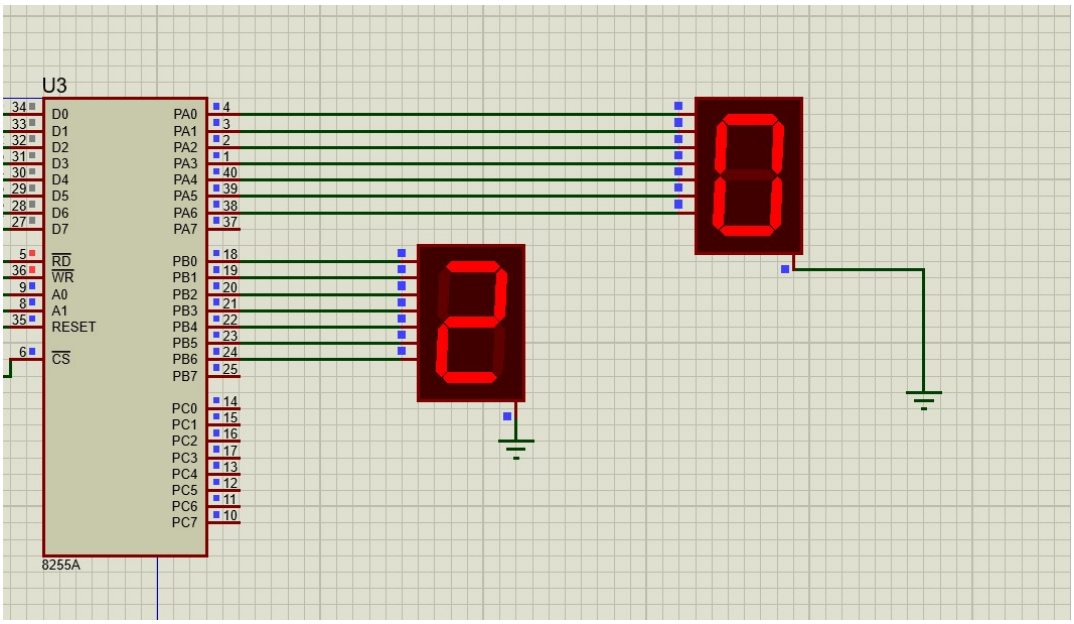


Figure 14(The System at 20 Seconds)

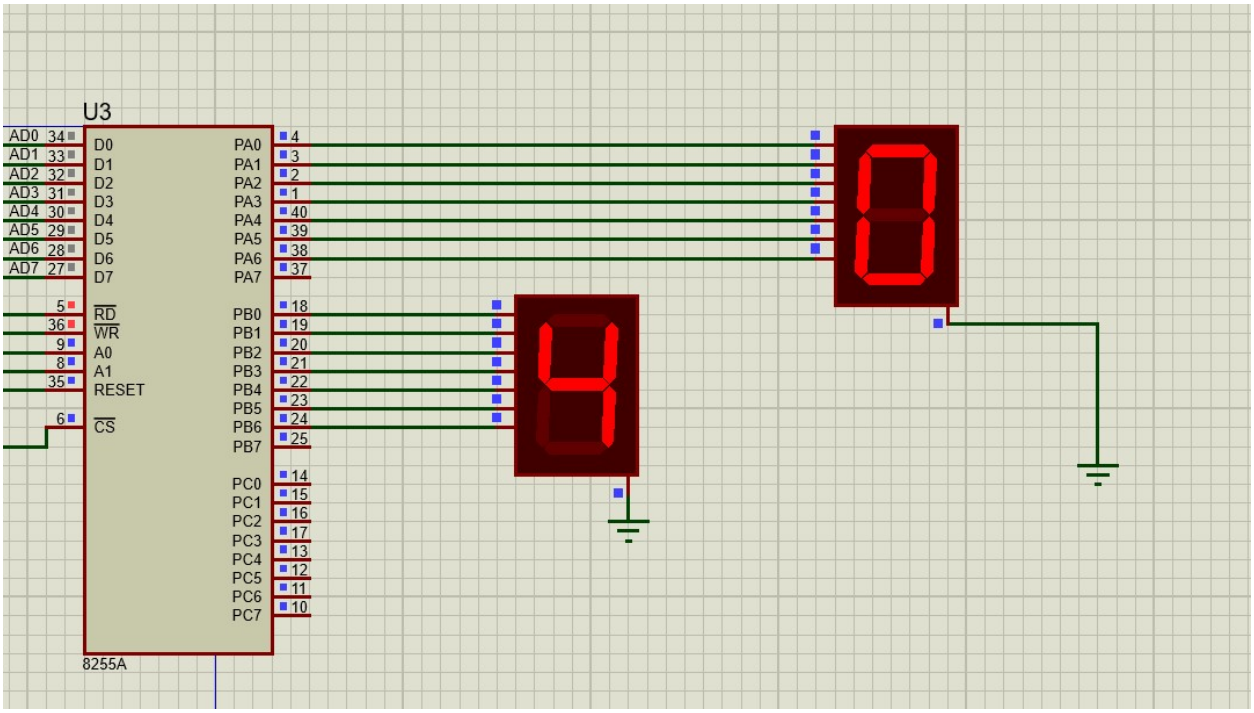


Figure 15(The System at 40 Seconds)

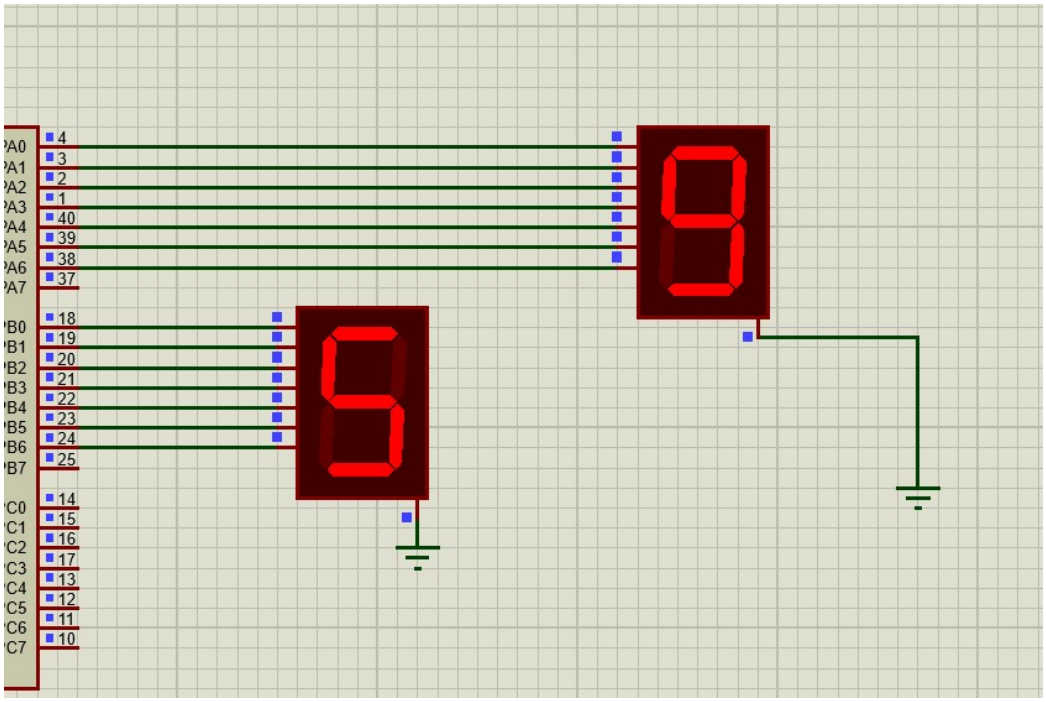


Figure 16(The system reaching 59 seconds and then resets to 0 after that)

Conclusion:

So, in conclusion This system is not very complex but it requires some basic understanding of assembly language, and how to correctly address memory, basically any system with I/O peripherals and memory needs to have latches and decoders in order to control the data flowing in and out of that hardware, the 7 segment displays can be utilized for many tasks just like this 60 second counter, they can be used in many applications in our daily lives

References:

1- The Intel microprocessors - University of Maryland, Baltimore County. (n.d.). https://userpages.umbc.edu/~squire/intel_book.pdf

2- GeeksforGeeks. (2023, May 14). *Programmable peripheral interface 8255*. GeeksforGeeks. https://www.geeksforgeeks.org/programmable-peripheral-interface-8255/

3- *Analog: Embedded processing: Semiconductor company*. Analog | Embedded processing | Semiconductor company | TI.com. (n.d.). https://www.ti.com/