International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Enhanced Technique of Interfacing AT89C55 Microcontroller with Seven Segment Display for Real-Time Display Applications

Isizoh A.N.[1], Alagbu E.E.[2], Ebih U.J.[3], Ejimofor I.A.[4], Obianyo O.R.[5]

[1,2,3]: Dept. of Electronic and Computer Engineering, Nnamdi Azikiwe University, Awka, Nigeria,
[4,5]: Dept. of Computer Engineering, Madonna University of Nigeria, Akpugo Campus.

ABSTRACT

This paper discusses how AT89C55 microcontroller can be interfaced with seven segment to display an information for real-time application. A microcontroller is a single chip computer which is used in several embedded and real-time systems, while a seven segment is usually needed for information display. In this paper, an AT89C55 microcontroller was particularly used and a four-in-one Common Anode Seven Segment was also used to show how the two devices could be interfaced for display of information. In order to achieve the purpose of this paper, a background study of seven segment display and microcontroller was x-rayed. The interface program was developed in Assembly language, and the system was able to display the word ECUP in a Proteus environment. The result of the research shows that by proper interfacing and programming of AT89C55 microcontroller with seven segment, a nice display of information could be obtained.

KEYWORDS: Microcontroller, proteus software, seven segment, interrupt, processors.

I. INTRODUCTION

1.1 Background Study of Microcontroller

A microcontroller is a single chip computer. It is a computer because it has all the component parts of a typical digital computer [1]. Those component parts are: the Central Processing Unit (CPU), Memory (ROM and RAM) and Input/output units. The difference between a microprocessor and microcontroller is that a microprocessor is only a processor without an internal memory, but can access external memory; while a microcontroller contains a processor as well as an inbuilt (internal) memory.

There are different types of microcontrollers from different manufacturers. The common ones are PIC microcontroller, Motorola (68HC05) Microcontroller, ATMEL Microcontroller, Intel Microcontroller, etc. In this dissertation, ATMEL microcontroller was used. Different types of ATMEL microcontrollers exist in markets. The type used for any design is determined by program size and its availability in the market. Some examples of ATMEL microcontrollers are: AT87C51, AT89C51, AT89C52, AT89C55, AT89C2051 etc. The difference between these microcontrollers is their memory size and sometimes their port configuration, for instance, AT89C55 or simply AT8955 has 20kb of ROM. The AT89C2051 has 2 ports with 4kb of ROM, while others listed above have 4 ports each. Therefore, it can be said that AT89C2051 is mainly used for small projects because of its number of ports [2].

Memory size may not be a hindrance in choosing any microcontroller for projects. This is true because one can simply use AT89C55 and if the project needs modification that requires large ROM memory space, one can just buy a program memory i.e. Read Only Memory (ROM) and store those programs. This additional ROM must be connected to the microcontroller through transparent latch (74373) or D-flip flop (74374) [3].

1.2 The AT89C55 Microcontroller

This microcontroller is what was used for this research work. The standard features of this microcontroller as contained in its architecture shown in Figure 1 are:

- 20 KB of Internal ROM (Flash)
- 256 bytes of RAM (Static RAM)
- 32 I/O pins
- 8 special pins
- Two 8/16 bit timers
- Four programmable I/O Ports (P0, P1, P2, and P3)
- Multiple internal and external interrupt sources
- Programmable serial port
- Interface for up to 128 kB of external memory

- On chip oscillator and clock circuitry

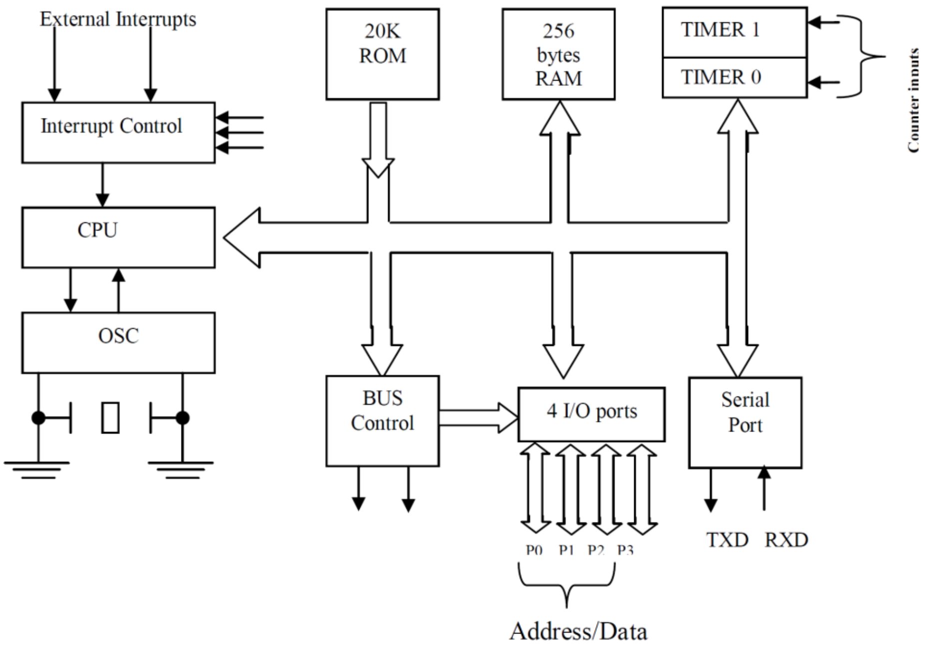

Figure 1: Internal Architecture of AT89C55 Microcontroller

1.3 Block Diagram of AT89C55 Microcontroller

The block diagram of an AT89C55 microcontroller consists of the Central Processing Unit (CPU), the Memory Unit, and the Input/Output Unit [4].

<u>The CPU</u> - The Central Processing Unit (CPU) of AT89C55 microcontroller comprises of the Arithmetic and Logic unit (ALU), the Registers, and the Control unit.

The function of the ALU is to perform all the arithmetic and logic expressions from or within the CPU according to the instruction it receives. Such arithmetic instructions are addition, subtraction, etc; and logic expressions like AND, OR, NOT, etc [5]. It receives two items of data either from registers or from a register and a memory cell. Its output, which is the result of the instructed operation, flows to a register.

The register part of the CPU is a temporary memory unit, which is able to contain only one number. This number can represent either an address or data, depending on the register's function. The register's size is determined accordingly (8 or 16 bits).

The most common operation that takes place inside a processor is the transfer of binary information from one register to another. The number and types of registers that a processor contains is a key part of its structures and it has a major effect on the programming effort required in a given application. The register structure of different processors varies considerably from manufacturer to manufacturer. However, the basic functions performed by the various registers are essentially the same in all processors. They are used to store data, addresses, instruction codes, and information on the status of various processor operations. Some are used as counters which can be controlled by software (program instructions) to keep track of such things as the number of times a particular sequence of instruction has been executed or sequential memory locations from which data are to be taken.

The most complex part of the CPU is the Control Unit. This unit receives the binary number signifying the instruction about to be carried out, decodes the instruction and carries it out according to the steps required. The number of instructions and their possibilities is enormous. The Control unit controls the internal operations of the microcontroller and sends control signals to other parts of the microcontroller during program execution [6].

An external crystal provides a timing reference for clocking the CPU. The AT89C55 microcontroller is an 8-bit single chip computer.

<u>The Memory</u> - Two on-chip memories exist in the chip. They are the Random Access Memory (RAM) and Read Only Memory (ROM).

The RAM - This is a volatile type of memory. The AT89C55 microcontroller uses the type of RAM called Static RAM (SRAM). Each SRAM cell contains at least four or five transistors to store a single bit of data. The transistors are arranged to form a flip-flop logic gate which can be flipped from one binary state to the other to store a binary unit.

So a Static RAM is essentially a matrix of flip flops. Therefore, we can write a new data word in a SRAM location at any time by applying the word to the flip-flop data inputs and clocking the flip flops. The stored data word will remain on the flip-flop output as long as the power is left on [7].

The eight data lines of AT89C55 SRAM tell you that RAM stores 8-bit words. When reading a word from the RAM, these lines function as outputs. When writing word to the RAM, the lines function as inputs. The Chip Enable (CE) input is used to enable the device for a read or for a write operation (low is for write and high is for read). The R/W input will be asserted high if we want to read from the RAM or asserted low if we want to write a word to the RAM [8].

The ROM – A ROM is a non-volatile memory. There is an on-chip Read Only Memory (ROM) embedded in AT89C55 microcontroller. There are different types of ROM: Programmable Read Only Memory (PROM), Erasable Programmable Read Only Memory (EPROM), Electrically Erasable Programmable Read Only Memory (EEPROM), and Flash. The type of ROM being used by AT89C55 is a 20K EEPROM. Its content can be erased by applying a suitable electrical voltage to the device [9].

EEPROM adds the feature of electrical erasability through the addition of a thin oxide region above the drain of MOSFET memory cell. By applying a high voltage between the MOSFET's gate and drain, a charge can be induced onto the floating gate, where it will remain even when power is removed. Reversal of the same voltage causes a removal of the trapped charges from the floating gate and erases the cell [10].

Ports/Pins Configurations - The AT89C55 Microcontroller has four ports (P0, P1, P2 and P3) and a total of 40 pins. Each of these ports can be addressed to in a program for the purpose of fetching or sending instruction. Out of these 40 pins, 32 pins are for inputs/outputs; while 8 pins are special pins. These eight pins are not included in the ports. They are special in the sense that their connections in a circuit are specific. These eight pins are Pin 9, Pin 18, Pin 19, Pin 20, Pin 29, Pin 30, Pin 31 and Pin 40. The pins/ports configurations are shown in figure 1.4.

So there are 32 input and output lines, which make up the four ports of the microcontroller. The ports enable the microcontroller to read and write to external memory and other components. The AT89C55 has four 8-bit I/O ports (Port 0 to 3). As the name suggests, the ports can act as inputs (to be read) or outputs (to be written to). Many of the port bits have optional alternate functions relating to accessing external memory using the on-chip timer/counters, detecting external interrupts, and handling serial communications. Table 1 is the summary of the AT89C55 Pin functions.

Table 1: Summary of the AT89C55 Microcontroller Pin Functions

Pin	Symbol	Input/Output	AT89C55 Function
1	(T2)P1.0	I/O	Port 1, bit 0, timer 2 external input
2	T2(EX)P1.1	I/O	Port 1, bit 1 timer 2 external reload/capture
3	P1.2	I/O	Port 1, bit 2
4	P1.3	I/O	Port 1, bit 3
5	P1.4	I/O	Port 1, bit 4
6	P1.5	I/O	Port 1, bi 5
7	P1.6	I/O	Port 1, bit 6
8	P1.7	I/O	Port 1, bit 7
9	Reset	Input	Reset system
10	(RXD) P3.0	I/O	Port 3, bit 0 serial receive
11	(TXD)P3.1	I/O	Port 3, bit 1; serial transmit
12	(INTO)P3.2	I/O	Port 3, bit 2;external interrupt 0
13	(INTI) P3.3	I/O	Port 3,bit 3; external interrupt 1
14	(TO) P3.4	I/O	Port 3, bit 4; Timer 0 external input
15	(T1) P3.5	I/O	Port 3, bit 5, timer 1 external input
16	(WR) P3.6	I/O	Port 3, bit 6; writes strobe for external data memory
17	(RD)P3.7	I/O	Port 3, bit 7; reads strobe for external data memory
18	XTAL 2	I/O	Inverting oscillator amplifier (crystal)
19	XTAL 1	I/O	Inverting oscillator amplifier (crystal)
20	GND	I/O	Circuit ground
21	(A8)P2.0	I/O	Port 2, bit 0; address bit 8
22	(A9)P2.1	I/O	Port 2, bit 1; address bit 9
23	(A10)P2.2	I/O	Port 2, bit 2; address bit 10
24	(A11)P3.3	I/O	Port 2, bit 3; address bit 11
25	(A12)P3.4	I/O	Port 2, bit 4; address bit 12
26	(A13)P2.5	I/O	Port 2, bit 5; address bit 13
27	(A14)P2.6	I/O	Port 2, bit 6; address bit 14
28	(A15)P2.7	I/O	Port 2, bit 7; address bit 15
29	PSEN	Output	Program store enable, reads strobe for external program memory
30	ALE	Output	Address latch enable
31	EA	Input	External access enable for program
32	(AD7) P0.7	I/O	Port 0, bit 7; Address/data bit 7
33	(AD6) P0.6	I/O	Port 0, bit 6; Address/data bit 6
34	(AD5) P0.5	I/O	Port 0, bit 5; Address/data bit 5
35	(AD4) P0.4	I/O	Port 0, bit 4; Address/data bit 4
36	(AD3) P0.3	I/O	Port 0, bit 3; Address/data bit 3
37	(AD2) P0.2	I/O	Port 0, bit 2; Address/data bit 2
38	(AD1) P0.1	I/O	Port 0, bit 1; Address/data bit 1
39	(AD0) P0.0	Input	Port 0, bit 0; Address/data bit 0
40	Vcc	Input	Supply Voltage

Accessing External Memory: The largest alternate use of the ports has to do with accessing external memory. Although AT89C55 is a single-chip computer, a complete AT89C55 system requires additional components. It can have external RAM in addition to the microcontroller's internal RAM, and most systems also have EPROM, EEPROM, or battery-backed RAM for permanent storage of programs.

Accessing this external memory uses all of Ports 0 and 2, plus pins 6 and 7 of Port 3 to hold data, address, and control signals for reading and writing to external memory. Data here refers to a byte to be read or written and may be any type of information, including program code. The address defines the location in the memory to be read or written [11].

During a memory access, port 0's eight pins (ADO-AD7) first hold the lower byte of the address followed by the data to be read or written. This method of carrying both address and data on the same signal line is called a multiplexed address/data bus. It is a popular arrangement that many devices use, since it requires fewer pins on the chip, compared to giving each data and address line its own pin. Port 2's eight lines hold the higher byte of the address to be read or written to. These lines make up the higher address bus (A8-A15). Together, the 16 address lines can access 64 kilobytes (65,536 bytes) of memory, from 00000000 00000000 to 11111111 11111111 in binary, or 0000h to FFFFh in hexadecimal.

Besides the pins to hold the data and addresses, the AT89C55 also provides control signals to initiate the read and write operations.
Control signals include WR (write), RD (read), PSEN (Program Store Enable) for accessing external code memory, ALE (Address Latch Enable). Some of the address lines may function as control signals that help to select a chip during a memory access.

<u>Timers and Counters</u> - The AT89C55 has two 16-bit timers/counters, which make it easier to generate periodic signals or count signal transitions [12].

<u>Serial Port</u> - The microcontroller's serial ports automatically take care of many of the details of serial communications. On the transmit side, the serial port translates bytes to be sent into serial data, including adding start and stop bits and writing the data in a timed sequence to SER OUT. On the receive side, the serial port accepts serial data at SER In and sets a flag to indicate that a byte has been received.

<u>External Interrupts</u> - INTO and INTI are external interrupt inputs, which detect logic levels or transitions that interrupt the CPU and cause it to branch to a pre-defined program location.

<u>Additional Control Inputs</u> - Two additional control inputs need to be mentioned: the RESET pin and the EA pin inputs.
A logic high on the RESET resets the chip and causes it to begin executing the program that begins at 0000h in code memory.
EA is the External Memory Access. It determines whether the chip will access internal or external code memory in the area of 000h to 07FFh. When this pin is tied low, the controller executes codes from external code memory.

<u>Power Supply Connections</u> - The chip has two pins for connecting to a +5 Volts DC power supply (Vcc) and Ground (Vss).

II. LITERATURE SURVEY

Many researches about microcontroller interfaces have been carried out. According to [6], seven segment can be interfaced to a microcontroller to achieve a desired circuit for a display. The work further concluded that one limitation that this interface has is that it cannot display some alphabets like X, Z, V, W, etc; so the use of LCD takes care of this problem.

In their book published by Isizoh et al. [7] titled, "Assembly Language Programming for Embedded Systems and Real-Time Applications" the authors highlighted that several input devices or output devices can be interfaced with microprocessor or microcontroller to achieve any desired display. Such I/Os can be light emitting diode, seven segment, liquid crystal display (LCD), sensors/transducers, electric motors, bulbs, etc.

Shams [8] described several ways of interfacing microcontroller with other i/o devices in his published book, titled, "Microcontroller Engineering" in 2012. The work particularly discussed LCD interface with microcontroller, and analyzed different kinds of microcontroller interfaces.

Jones [9] in his book on "Real-time Computing with Microcontrollers", he proposed several ways of interfacing microcontrollers with I/O devices for real-time systems. Analyses on display interfaces were also done. The book also stated that the first microcontroller which was Intel 8051 is a Harvard architecture; a single chip computer, developed by Intel in 1980 for use in embedded systems. Intel's original versions were popular in the 1980s and early 1990s, but have today largely been superseded by a vast range of faster and/or functionally enhanced 8051 compatible devices manufactured by more by more than 20 independent manufacturers including Atmel, Infineon Technologies, and Maxim Integrated products. Intel's official designation for the 8051 family of microcontrollers is MCS 51.

III. INTERFACING SEVEN SEGMENT WITH MICROCONTROLLER

Several input devices or output devices can be interfaced with microprocessor or microcontroller. Such I/Os can be light emitting diode, seven segment, liquid crystal display (LCD), sensors/transducers, electric motors, bulbs, etc. Let us examine how seven segment display is interfaced.

There are two types of Seven Segment Displays. One is Common Anode Seven Segment and the other is Common Cathode Seven Segment. For Common anode (CA), Low signal i.e. "0" will ON each LED, but for Common cathode (CC), High signal i.e. "1" will on each LED.

The symbol of a Seven Segment is shown in Figure 2, while types of Seven Segment Displays are shown in Figure 3.

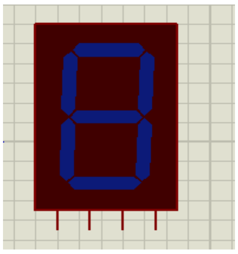

Figure 2: Symbol of a Seven Segment

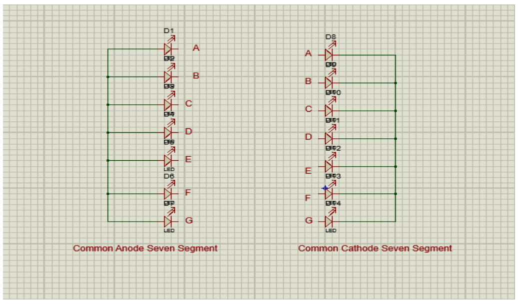

Figure 3: Types of Seven Segment Displays

Let us use a common anode for the interface. Please note that it is "0" that will ON each LED. The code table is shown in Table 2.

Table 2: The Code Table for Display of numbers on CA Seven Segment

	H	G	F	E	D	C	B	A	
0	1	1	0	0	0	0	0	0	
1	1	1	1	1	1	0	0	1	
2	1	0	1	0	0	1	0	0	
3	1	0	0	1	1	0	0	0	
4	1	0	0	1	1	0	0	1	
5	1	0	0	1	0	0	1	0	
6	1	0	0	0	0	0	1	0	
7	1	1	1	1	1	0	0	0	
8	1	0	0	0	0	0	0	0	
9	1	0	0	1	1	0	0	0	

H is "don't care". Any binary digit you give it does not matter.

Let us develop a circuit which has a microcontroller/seven segment interface for a stationary display of the word 'ECUP' using Common Anode Seven Segment Display. The circuit diagram is shown in Figure 4.

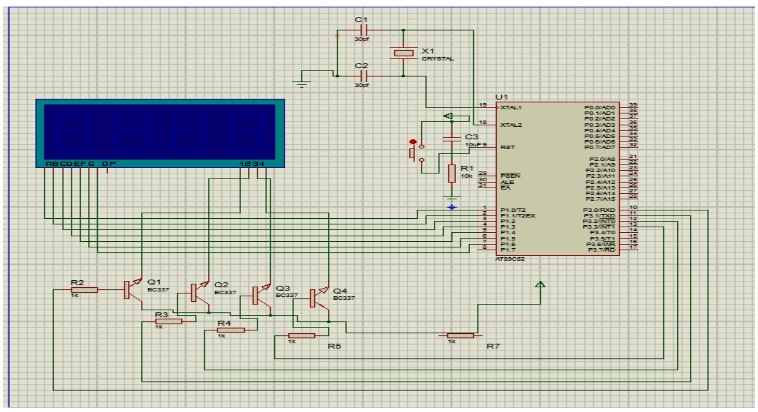

Figure 4: Circuit Diagram of Microcontroller/Seven Segment Interface

The Control Program for the system of operation is:

```
org 00h
start:
mov p1,#10000110b  ;sending data for letter E
mov p3,#00000000b  ;switching OFF p3 ie all transistors
setb p3.3          ;switching ON the transistor for the first 7-segment
call delay         ;delay time
mov p1,#11000110b  ;sending data for letter C
mov p3,#00000000b  ;switching OFF p3 ie all transistors
setb p3.2          ;switching ON the transistor for the second 7-segment
call delay         ;delay time
mov p1,#11000001b  ;sending data for letter U
mov p3,#00000000b  ;switching OFF p3 ie all transistors
setb p3.1          ;switching ON the transistor for the third 7-segment
call delay         ;delay time
mov p1,#10001100b  ;sending data for letter P
mov p3,#00000000b  ;switching OFF p3 ie all transistors
setb p3.0          ;switching ON the transistor for the fourth 7-segment
call delay         ;delay time
jmp start          ;go back to start
;delay subroutine
DELAY:
PET:Mov R1,#240
   Mov R2,#2
MEG:Djnz R1,MEG
   Djnz R2,MEG
Ret
end
```

V. CONCLUSION AND RECOMMENDATIONS

4.1 Conclusion

The analysis of how AT89C55 microcontroller can be interfaced with seven segment device to display any given information has been x-rayed in this work. In this seminar, the type of seven segment used was Four-in-one Common Anode Seven Segment Display. Assembly Language was used for the programming of the microcontroller for the display.

In order to check the workability of the program in a practical situation, a real-time simulation was performed in a Proteus environment using Proteus 7.5 software. It was observed that the interfaced circuit worked very well by displaying the word **ECUP** on the seven segment.

4.2 Recommendations

The following recommendations will improve the work for future use:

i) Any other type of microcontroller could be used as a controller. It depends on the requirements needed from the microcontroller.

ii) The voltage requirement for the seven segment and the AT89C55 microcontroller should not exceed 5V.

iii) Care should be taken to maintain safety in the laboratory.

iv) Any real-time simulation software, other than Proteus 7.5 can be used.

REFERENCES

[1] Schuster A., "Microcontroller Principles and Applications", Maxon Press Ltd, Rochester, 2008.

[2] Predko Myke, "Handbook of Microcontrollers", McGraw Hill, New York, USA, 2007, Pp 1-17.

[3] Douglas V.H., "Microcontrollers and Interfacing: Programming Hardware", McGraw Hill Inc, New York, 2008, Pp 270-344.

[4] Holliday D. and Resmick Robert, "Fundamentals of Microcontrollers", Don Peters Press Ltd, Fulmar, 2008, P 88.

[5] Katz P., "Digital Controls Using Microcontrollers", Prentice Hall International Inc, New Delhi, 2009, Pp 15-43.

[6] Wakerly John F., "Digital Designs: Principles and Practices", 4th Edition, PHI Learning Private Limited, New Delhi, 2008, Pp 5-6.

[7] Isizoh A.N., NwobodoNzeribe H.N., Aniedu A.N., "Asembly Language Programming: For Embedded Systems and Real-Time Applications", Scoa Heritage Publishers Ltd, Awka, Nigeria, 2018.

[8] Shams H. C., "Microcontroller Engineering", McGraw Hill, New York, USA, 2012, Pp 2-33.

[9] Jones F. D., "Real-time Computing with Microcontrollers", Prentice Hall of India Private Limited, New Delhi, 2014, Pp 21-25.

[10] Wakerly John F., "Programming Microcontrollers for Digital Designs: Principles and Practices", 4th Edition, PHI Learning Private Limited, New Delhi, 2013, Pp 5-6.

[11] Holliday D. and Resmick R., "Fundamentals of Microcontrollers", Don Peters Press Ltd, Fulmar, 2012, P 88.

[12] Sampath K., Venkatesh H., "8051 Microcontroller and Embedded Systems", S. K. Kataria & Sons Publishers, New Delhi, 2006, reprinted in 2013.