International Journal of Research Publication and Reviews

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

Analysis of Microcontroller/LCD Interface for Embedded Systems

Isizoh A.N.[1], Ufoaroh S.U.[2] Alagbu E.E.[3], Ugwuanyi C.J.[4], Abonyi D.C.[5]

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

ABSTRACT

This paper analyzes the way Liquid Crystal Display (LCD) is interfaced to a microcontroller for information display in embedded systems. To achieve this purpose, a background study of LCD and microcontroller was x-rayed. The LM016L type of LCD (an alphanumeric LCD) was used as a case study of the LCD for the analysis, while AT89C51 microcontroller was used as a case study of the microcontroller analyzed. It was seen that by programming a microcontroller adequately to execute LCD Command and Data write codes, information (alphabets and numbers) could be displayed on the LCD. The interface program was developed in Assembly language, and the output of the interface was displayed in a Proteus environment.

KEYWORDS: Microcontroller, LCD, subroutine, display, embedded systems.

I. INTRODUCTION

1.1 Background Study of Liquid Crystal Display (LCD)

LCD means Liquid Crystal Display. In recent years, it is finding widespread use, thus, replacing Light Emitting Diodes (LEDs), Seven Segment Displays, and others. This is due to the following reasons:

1) The declining prices of LCDs.

2) Ability to display numbers, characters and graphics. This is in contrast to 7-Segment, which its display is limited to some numbers and few characters.

3) Ease of programming for characters and graphics display.

4) Incorporation of refreshing controller into the LCD, thereby hiring the CPU of the task of refreshing the LCD. In contrast, the 7-segment must be refreshed by the CPU to keep displaying data [1].

There are many places where LCDs are applied in these days' modern electronic gadgets. For example, modern television sets are made with LCD for display, replacing the old cathode ray tube system. Modern wrist watches use LCD instead of seven segment, plus other examples.

1.1.1 LCD Pins Arrangement and Description

The diagram showing the LCD pins arrangement is shown in figure 1, while the pins description is shown in Table 1.

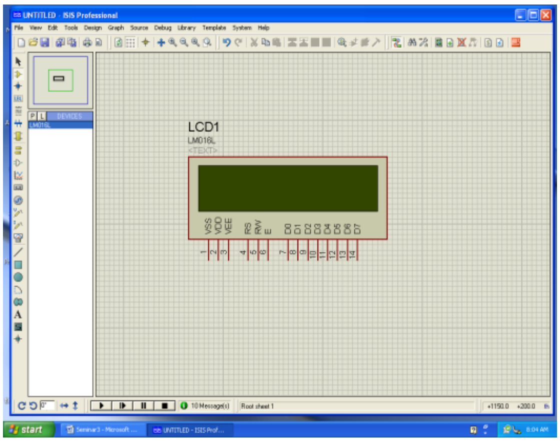

Figure 1: LCD Pins arrangement

Table 1: The description of the pins of an LCD.

Pin	Symbol	I/O	Description
1	VSS	-	Ground
2	VCC	-	+5V Power Supply
3	VEE	-	Power Supply to Control Contrast
4	RS	1	RS = 0, to select command register; RS = 1, to select data register
5	R/W	1	R/W = 0 for write R/W = 1 for read
6	E	I/O	Enable
7	DB0	I/O	The 8-bit data bus
8	DB1	I/O	The 8-bit data bus
9	DB2	I/O	The 8-bit data bus
10	DB3	I/O	The 8-bit data bus
11	DB4	I/O	The 8-bit data bus
12	DB5	I/O	The 8-bit data bus
13	DB6	I/O	The 8-bit data bus
14	DB7	I/O	The 8-bit data bus

The VCC and VSS provide +5V and ground respectively, while VEE is used for controlling the LCD contrast.

1.1.2 The Control Pins of LCD

There are three main control pins in LCD. They are: the Register Select (RS), Read/ Write (R/W), and Enable (E).
Register Select (RS)
The register select is a very important register inside the LCD. It controls two important registers: the Data register and the Command code register. The RS pin is used for selection as follows:
If RS = 0, the instruction command code register is selected, thus allowing the user to send a command such as "clear", " display cursor at home", etc.
If RS = 1, the data register is selected, allowing the user to send data to be displayed on the LCD.
Read/ Write (R/W)
This is another very important register inside the LCD. The R/W input allows the user to write information to the LCD or read information from it.
If R/W = 0, it writes data onto the LCD;
If R/W = 1, it reads data from the LCD.
Enable (E)
The Enable pin is used by the LCD to latch information that is present on its data pins. When data is supplied to the data pins, a high – to – low pulse must be applied to these pins in order for the LCD to latch in the data present at the pins [2].

1.2 Fundamentals of Microcontroller

A microcontroller is a single chip computer. It is a single chip because it is in a form of an integrated circuit (IC), and also a computer because it has all the component parts of a computer system. It is a low cost programmable electronic device that is characterized by a very high speed of about 12 MHz that is reliable and flexible. The micro suggests that the device is small, while controller suggests that the device can be used to control objects and processes. Thus, a microcontroller is a single chip computer [3].
The type of microcontroller used as a case study for this paper is AT89C51. It is an advanced single chip electronic device whose immense advantage is the vast range of functions to which it can be applied.
The AT89C51 microcontroller is a member of the 8051 microcontroller family. It is a single chip IC which contains all the logic circuitry that performs the following functions:

a) It provides timing and control signals.
b) It fetches instruction and data from the memory
c) It transfers data to and from memory and other input/output devices.
d) It interprets instructions.
e) It performs arithmetic and logic operations based on the instructions it receives [4].

1.2.1 AT89C51 Microcontroller Architecture

The AT89C51 microcontroller has the same architecture with Intel 8051. In fact, all '51 family members of microcontrollers have 8051 architecture [5]. All these microcontrollers are usually called 8051 microcontrollers because they can all be programmed using 8051 assembly language, and they all share certain features (although every model has its own special features). Figure 2 shows the architecture of Intel 8051 microcontroller.

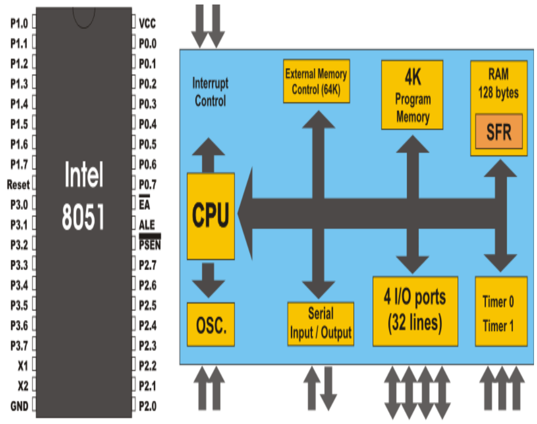

Figure 2: The Intel Microcontroller architecture

The Intel 8051 is an 8-bit microcontroller which means that most of its available operations are limited to 8-bit. There are three basic sizes of 8051; short, standard and extended. The short and standard chips are often available in DIP form, but the extended 8051 models often have a different form factor, and are not drop-in compactable.

1.2.2 Central Processing Unit (CPU): The Central Processing Unit (CPU) contains the Arithmetic and Logic Unit (ALU), the Registers, and the Control Unit (CU). The unit generally performs all the processing task of the microcontroller [6].

The ALU: This is responsible for solving arithmetic jobs (e.g. addition) and logical problems (e.g. NOR).
The Registers: The registers are very important temporary memories used internally by the microcontroller itself. They play important roles during microcontroller operations. Therefore, a very good understanding of each of the registers is very essential.
Accumulator - An accumulator is used as a general register to hold the result of large number of instructions. It is one of the registers that are mostly used. It can hold an 8-byte word. The result of a microcontroller operation can be stored in the accumulator for further processing, or it can be transferred to another register.
The 'B' Register - The 'B' register is very similar to the accumulator because it can hold a byte word. The 'B' register is only used by two AT89C51 instructions: MUL and DIV. They are auxiliary registers for multiplication and division.
The 'R' Register - The 'R' register is a set of eight registers named R_o, R_l, R_2, R_3, R_4, R_5, R_6 and R_7. This is an R-bit register. They are called auxiliary register or helper register.
Program Counter (PC) - The program counter is a 2-byte address that tells the microcontroller where the next instruction to be executed is found in the memory.
When the microcontroller is initialized, the PC starts from ooh and increases each time an instruction is executed. If the instruction is 2 or 3 bytes, the program counter increases by 2 or 3 respectively.

Data Pointer Register (DPTR) - The data pointer register is the AT89C51's only user accessible 16-bit (2 byte) register. The accumulator, 'R' and 'B' registers are all 8-bit registers. The DPTR as the name suggests is used to point to data. It is used by a number of commands, which allow the AT89C51 to access external memory. When the AT89C51 accesses external memory, it accesses external memory at the address indicated by the data point register. The DPTR is often used to store 2 byte values, which have nothing to do with memory locations.
Stack Pointer (SP) - The stack pointer is used to indicate where the next value to be removed from the stack should be taken from. When a value is pushed on to the stack, the AT89C51 increases the value of the SP and then stores the value at the resulting memory location. When a value is popped off the stack, the AT89C51 returns the value from the memory location indicated by the stack pointer and then decreases the value of the stack pointer.
The Control Unit: The main function of the control unit is to fetch and decode the instruction codes from memory locations. It also generates the control signal that is required by other units and sections of the microcontroller in order to enable the ALU carry out the execution of instructions.

1.2.3 Memories

Two on-chip memories exist in this microcontroller. They are the Program memory or Read Only Memory (ROM) and the Data memory or Random Access Memory.
Program memory or Read Only Memory (ROM): The ROM part of the AT89C51 microcontroller is called the Program memory, and has a 4Kb Electrically Erasable Programmable Read Only Memory (EEPROM). It is called the Program memory because that is the area where the written control program is stored. The ROM can only be read from by the control unit but data cannot be written to it by the CPU. The information contained in the ROM can only be written to it while programming the microcontroller. It is a non-volatile memory in the sense that the information it contains cannot be erased easily even if the power supply is off.

Data memory or Random Access Memory (RAM): The RAM is where data are stored for temporary use, and that is why it is called the data memory. The data memory is a read/write memory. The CPU can write to the RAM as well as reading from it. The control signal that reads the data memory is the RD control signal, while the control signal that enables write operation onto the data memory is the WR control signal. The RAM is a volatile memory in the sense that information could only be retained in the RAM pending the availability of power supply. Once the power supply is off, the data are lost.

A particular useful feature of the 8051 core is the inclusion of a Boolean processing engine which allows bit-level Boolean logic operations to be carried out directly and efficiently on internal registers and RAM. This feature helps to cement the 8051's popularity in industrial control applications. Another valued feature is that it has four separate register sets, which can be used to greatly reduce interrupt latency compared to the more common method of storing interrupt context on a stack.

1.2.4 Serial I/O ports

The AT89C51 UART makes it simple to use the chip as a serial communications interface. External pins can be configured to connect to internal shift registers in a variety of ways, and the internal timers can also be used, allowing serial communications in a number of modes, both synchronous and asynchronous. Some modes allow communications with no external components. A mode compatible with an RS-485 multi-point communications environment is achievable, but the AT89C51's real strength is fitting in with existing ad-hoc protocols (for example, when controlling serial controlled devices).

Once the UART, and the timer if necessary, have been configured, the programmer needs only to write a simple interrupt routine to refill the send shift register (copy the data somewhere else). The main program then performs serial reads and writes simply by reading and writing 8-bit data to stacks.

1.2.5 Crystal Oscillator (OSC)

The AT89C51 operates based on an external crystal. This is an electrical component which, when energy is applied, emits pulses at a fixed frequency. It is used for external (macro) operations. One can find crystals of virtually any frequency depending on the application requirements. When using AT89C51, the most common crystal frequencies are 12 megahertz and 11.059 megahertz (with 11.059 being much more common).

Microcontrollers (and many other electrical systems) use crystals to synchronize operations. The AT89C51 uses the crystal precisely for synchronizing its operation. Effectively, the microcontroller operates using what are called "machine cycles". A single machine cycle is the minimum amount of time in which a single AT89C51 instruction can be executed.

Although many instructions take multiple cycles; a cycle is in reality 12 pulses of the crystal. That is to say that if an instruction takes one machine cycle to execute, it will take 12 pulses of the crystal to execute. Since we know that the crystal is pulsing 11,059,000 times per second and that one medium cycle is 12 pulses, we can then calculate how many instruction cycles the AT89C51 can execute per second.

11,059,000/12 = 921,583

This means that the microcontroller can execute 921,583 single-cycle instructions per second. Since a large number of AT89C51 instructions are single-cycle instructions, it is often considered that the microcontroller can execute roughly 1 million instructions per second, although in reality it is less (and depending on the instructions being used, an estimate of about 600,000 instructions per second is more realistic). For example, if you are using exclusively 2 cycle instructions, you will find that the AT89C51 will execute 460,791 instructions per second. The AT89C51 also has two really slow instructions that require a full 4 cycles to execute. If you were to execute nothing but those instructions you will find the performance to be about 230,395 instructions per second.

It is again important to emphasize that not all instructions are executed with the same amount of time. The fastest instructions require one machine cycle (12 crystal pulses), many others require two machine cycles (24 crystal pulses).

1.2.6 Timer

Since all the instructions require different amounts of time to execute, a very obvious question comes to mind. How can one keep track of time in a time-critical application if we have no reference to time in the outside world? Luckily, the AT89C51 microcontroller includes timers which allow us to time events with high precision.

1.2.7 Some features that Made AT89C51 Popular:

- 32 general purpose registers each of 8-bit
- 16-bit timers
- 3 internal and 2 external interrupts
- Bits as well as byte addressable RAM area of 16-byte
- 16 bit program counter and data pointer
- 8-bit ALU, Accumulator and registers, hence it is an 8-bit microcontroller
- 8-bit data bus- it can access 8 bits of data in one operation
- 16-bit addresses bus- it can access 2^{16} memory locations-64 kb (65536 locations) each of RAM and ROM
- On- chip RAM- 128bytes (data memory)
- On-chip ROM – 4kb (program memory)
- Four byte bi-directional input/output port
- UART (serial port)
- Two 16-bit counter/timers
- Two-level interrupt priority
- Power saving mode

II. LITERATURE SURVEY

The first microcontroller which is 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 [7].

Intel's original 8051 family was developed using NMOS technology, but later versions, identified by a letter C in their name (e.g 80C51) used CMOS technology and were less power hungry than their NMOS predecessors. This made more suitable for battery powered devices.
The 8051's predecessors, like the 8048 were used in the keyboard of the first IBM PC, where it converted key presses into a serial data stream which is sent to the main unit of the computer. The 8048 and derivatives are still used today for basic model keyboards.
The 8031 was a cut down version of the original Intel 8051 that did not contain any internal Program memory (ROM). To use this chip, external ROM had to be added containing the program that the 8031 would fetch and execute [8].

The 8052 was an enhanced version of the original 8051 that featured 256 bytes of internal RAM instead of 128 bytes, 8 Kb of ROM instead of 4Kb, and a third 16 bit timer. The 8032 had these same features except for the internal Program memory (ROM). The 8052 and 8032 are largely considered to be obsolete because these features and more are included in nearly all modern 8051 based microcontrollers [9].
Intel discontinued its MCS 51 product line in March 2007, however there are plenty of enhanced 8051 products or silicon IP.

LCD is very popular in modern electronics display because of the graphic nature of its display. In microcontroller applications, use of LCD for information display gives the output data a clear vision. In addition, it does not select characters to display, unlike seven segment display which cannot write or display letters like 'z', 's', 'm', 'k', 'x', etc.
Many LCDs exist in the market, and your choice is dependent on the mode of display to carry out. Below are some of the LCDs available:
LM016L which is 16x2 Alphanumeric LCD
LM017L which is 32x2 Alphanumeric LCD
LM018L which is 40x2 Alphanumeric LCD
LM020L which is 16x1 Alphanumeric LCD
LM032L which is 20x2 Alphanumeric LCD
LM041L which is 16x4 Alphanumeric LCD
LM044L which is 20x4 Alphanumeric LCD
MDLS40466L which is 40x4 Alphanumeric LCD, with dual controllers

LCDs have codes for positioning and other manipulations. These codes are what programmers do use when interfacing LCD with microcontroller to get a desired output. Cottle [10] used this to develop a new MDLS40466L which is 40x4 Alphanumeric LCD, with dual controllers.

III. INTERFACING ANALYSIS AND RESULT

3.1 Analysis

When an LCD is interfaced to a microcontroller, a program is used to make the microcontroller write on the LCD. In LCD programming, a minimum of 450µS is at least needed for a latch to take place. What this means is that when you are displaying characters, this time is to enable the register handle the display of a character before you send another character. For example, if you want to display a word 'BE'; when you send 'B', there must be a time of more than 450µS before you can send 'E'. In reality, if you are using TIME 8051 Delay software, it is better to use 1 second delay for latching.

3.2 Use of D0 – D7

The 8-bit data pins, D0 – D7 are used to send information to the LCD or read the contents of the LCD's internal registers. All the pins are connected to the 8 pins of the microcontroller. In order to display letters and numbers, you send ASCII codes for the letters A – Z, a – z, and numbers 0 – 9 to the pins, while making RS = 1.

3.3 Command Codes used in LCD Programming

There are instruction command codes that must be sent to the LCD to clear the display or force the cursor to the Home position or blink the cursor. These command codes are used for LCD/microcontroller interface programming. Below is the list of those Command codes to LCD:

Codes (Hex)	Commands to LCD
1h	Clear display screen
2h	Return home
4h	Shift cursor to the left (cursor not moving with character)
5h	Shift display to the right
6h	Shift cursor to the right
7h	Shift display to the left
8h	Display off, cursor off
Ah	Display off, cursor on
Ch	Display on, cursor off
Eh	Display on, cursor blinking
10h	Shift cursor to the left (cursor moving with character)
14h	Shift cursor to the right (cursor moving with character)
18h	Shift entire display to the left
1Ch	Shift entire display to the right
80h	Force cursor to begin on the first line
0C0h	Force cursor to begin on the second line
38h	Select two lines and 5x7 matrix

3.4 Subroutine Programs in LCD Programming

There are two important subroutines in LCD programming interface to a microcontroller: the Command Write and Data Write subroutines.

<u>Command write</u>: This is a subroutine for writing commands to the LCD. The general format is written below:

```
Mov p1, A    ; move data in the accumulator
               to port 1 e.g. #38h to p1
Clr RS       ; clear RS to make it zero, so as
               to select the command register
Clr RW       ; clear RW to make it zero, so as
               to write data to the LCD
Setb E       ; make E to be high
Call delay   ; delay e.g. for 1sec
Clr E        ; make E low i.e. zero, so as to
               be able to hold data
Ret
```

<u>Data Write</u>: This subroutine is used for writing data or character to the LCD for display. The general format is written as:

```
Mov p1, A
Setb RS      ; to select Data register
Clr RW
Setb E
Call delay
Clr E
Ret
```

3.5 Result of Practical Example of Microcontroller/ LCD Interface Using Proteus software

A practical example of an LCD interface to a microcontroller was carried out in a Proteus environment as shown in figure 3. This is to show a practical approach on how the interface is done.

The type of LCD used is LM016L, which is the most common LCD in the Nigerian markets. It is 16x2 LCD; meaning that it can accommodate 16 characters in one line, but has 2 lines. It is also an Alphanumeric LCD; meaning that it can display alphabets as well as numbers.

The result shows that with proper interface between a microcontroller and LCD, a good control program will drive a display to be made on LCD.

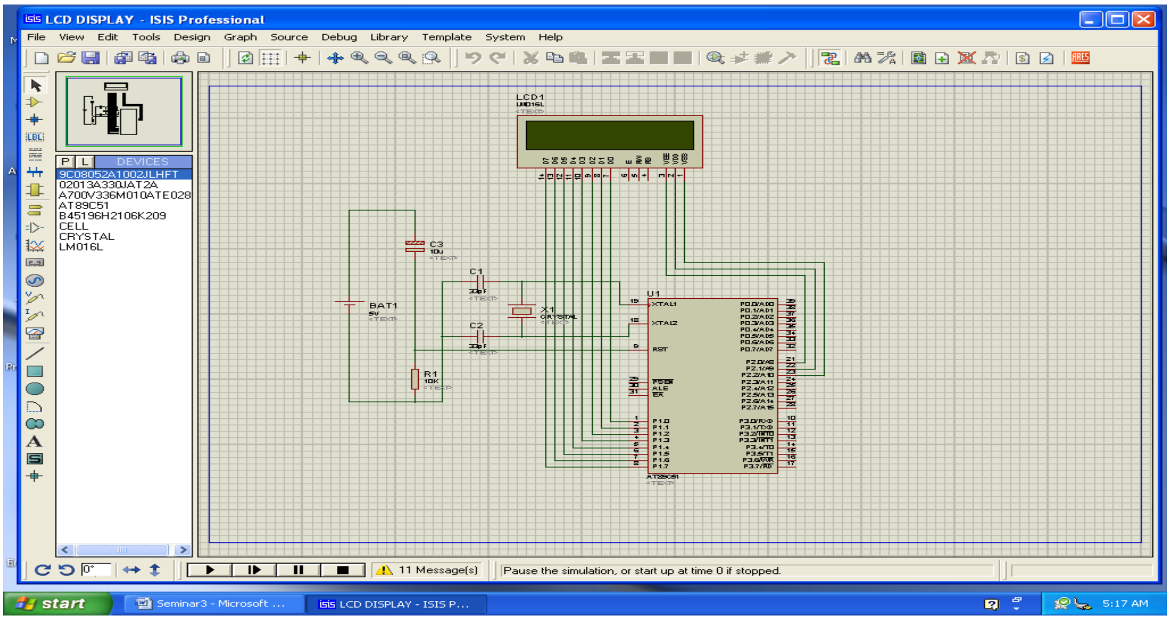

Figure 3: Diagram Showing Microcontroller/ LCD Interface

IV. CONCLUSION AND RECOMMENDATION

4.1 Conclusion

Analysis and practical approach were used to demonstrate how LCD can be interfaced to a microcontroller for information display. It was seen that for such an interface to be done, there is a need for proper connection of the control pins (RS, RW, and E) of the LCD to the appropriate pins of the microcontroller. Two main subroutines (the command write and the data write) must also be written without any error.

Then the actual main program must be written, bearing in mind the type of programming language to be used. The words to be displayed must be taken into consideration in the main program since the characters will still appear in the program as data to be transferred.

4.2 Recommendation

Interfacing an LCD to a microcontroller for information display is highly recommended because it is easier to be done than using other display-interface applications such as seven segment, dot-matrix, etc. Also, it occupies smaller space than others. Its use in electronic circuit designs should be encouraged as it can display any character unlike other display devices like the seven segment display.

REFERENCES

[1] Hall D. V., "LCD Principles and Applications", MCGraw Hill Inc., New York, 2009, Pp 3-25.
[2] Cottle J. G., "LCD Interfacing and Programming", Scientific Educational Systems, Lagos, 2007.
[3] Schuster A., "Microcontroller Principles and Applications", Maxon Press Ltd, Rochester, 2008.
[4] Predko Myke, "Handbook of Microcontrollers", McGraw Hill, New York, USA, 2007, Pp 1-17.
[5] Douglas V.H., "Microcontrollers and Interfacing: Programming Hardware", McGraw Hill Inc, New York, 2008, Pp 270-344.
[6] Holliday D. and Resmick Robert, "Fundamentals of Microcontrollers", Don Peters Press Ltd, Fulmar, 2008, P 88.
[7] Katz P., "Digital Controls Using Microcontrollers", Prentice Hall International Inc, New Delhi, 2009, Pp 15-43.
[8] Wakerly John F., "Digital Designs: Principles and Practices", 4th Edition, PHI Learning Private Limited, New Delhi, 2008, Pp 5-6.
[9] Floyd T.F., "Digital Fundamentals", 6th Edition, Prentice-Hall International Inc, New Delhi, 2008, Pp 4-7.
[10] Allen L., Wyatt, "Using Assembly Language", 3rd Edition, Que Corporation, 11711N, College Avenue, Carmel, USA, 2009.