Asian Journal of Information Technology, 5 (2) : 213-218, 2006

Design of a Low Cost 8051 Architecture Based Micro-controller Learning Kit

[1]Tareq Hasan Khan and [2]Nayeem Ahmed Ninad
[1]Department of Electrical and Computer Engineering, Concordia University, Montreal, Quebec, Canada
[2]Department of Electrical and Electronic Engineering, Islamic University of Technology, Dhaka, Bangladesh

Abstract: The demand and popularity of micro-controller based system has reached the zenith in the last few years. So it is the best time for the engineering students and professionals to learn how to use micro-controllers and how to program them. Micro-controller learning kits can be used for educational purpose. But they are expensive and sometimes it becomes difficult for schools or universities to manage them. In this study we will introduce a low cost 8051 architecture based micro-controller kit. It can be programmed using personal computer (PC) with the help of our machine code downloading software. Using this kit, we can glow LED's and 7-segment display, play sound, drive relay, generate square pulses of different frequencies according to the downloader program. To interact with human there is a keyboard. Besides, external analog signals can be inserted into the kit through an analog to digital interface.

Key words: 8051 Micro-controller, ADC, ISP Programming, LPT

INTRODUCTION

In this study we will discuss about the construction and programming method of 8051 architecture based micro-controller kit. We will program it using Personal Computer (PC) through the LPT port.

8051 MICROCONTROLLER

The 8051 is an 8 bit micro-controller originally developed by Intel in 1980[1]. It is one of the most popular micro-controllers in the world for its high performance, rich instruction set (MCS-51®) and low cost.
A typical 8051 contains:

- CPU
- Clock frequency: Its frequency range varies with its different family members. In most cases its maximum frequency is 24MHz-33MHz.
- Ports: It has four 8-bit ports, total 32 I/O lines. Each port can be used as Input or Output port. Each I/O lines are individually accessible.
- Memory: 8051 has Program Memory or ROM, RAM and EEPROM in some models as shown in Fig. 1. Its Program memory can be erased and reprogrammed using conventional memory programmer or by 'In System Programming (ISP)' method. In our kit we will use the 8051, which supports ISP Programming like AT89S51, AT89S52, AT89S53 etc. models.

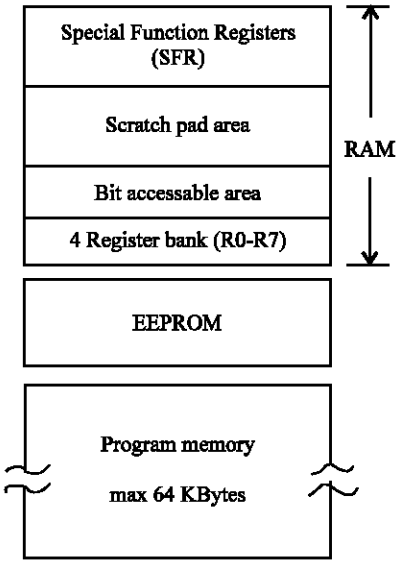

Fig. 1: 8051 Memory architecture

THE 8051 MICROCONTROLLER KIT

Modes of operation: The 8051 micro-controller[2] kit can operate in 2 modes. We can select these modes from our downloader software.

(a) Programming mode: In this mode the LPT_PGM goes high and LPT_RUN goes to low (Fig. 2.). The tri-state switches connects the 8051 to the PCs LPT Programming pins and separates the peripherals connected with those pins so that connected

Corresponding Author: Tareq Hasan Khan, Department of Electrical and Computer Engineering, Concordia University, Montreal, Quebec, Canada

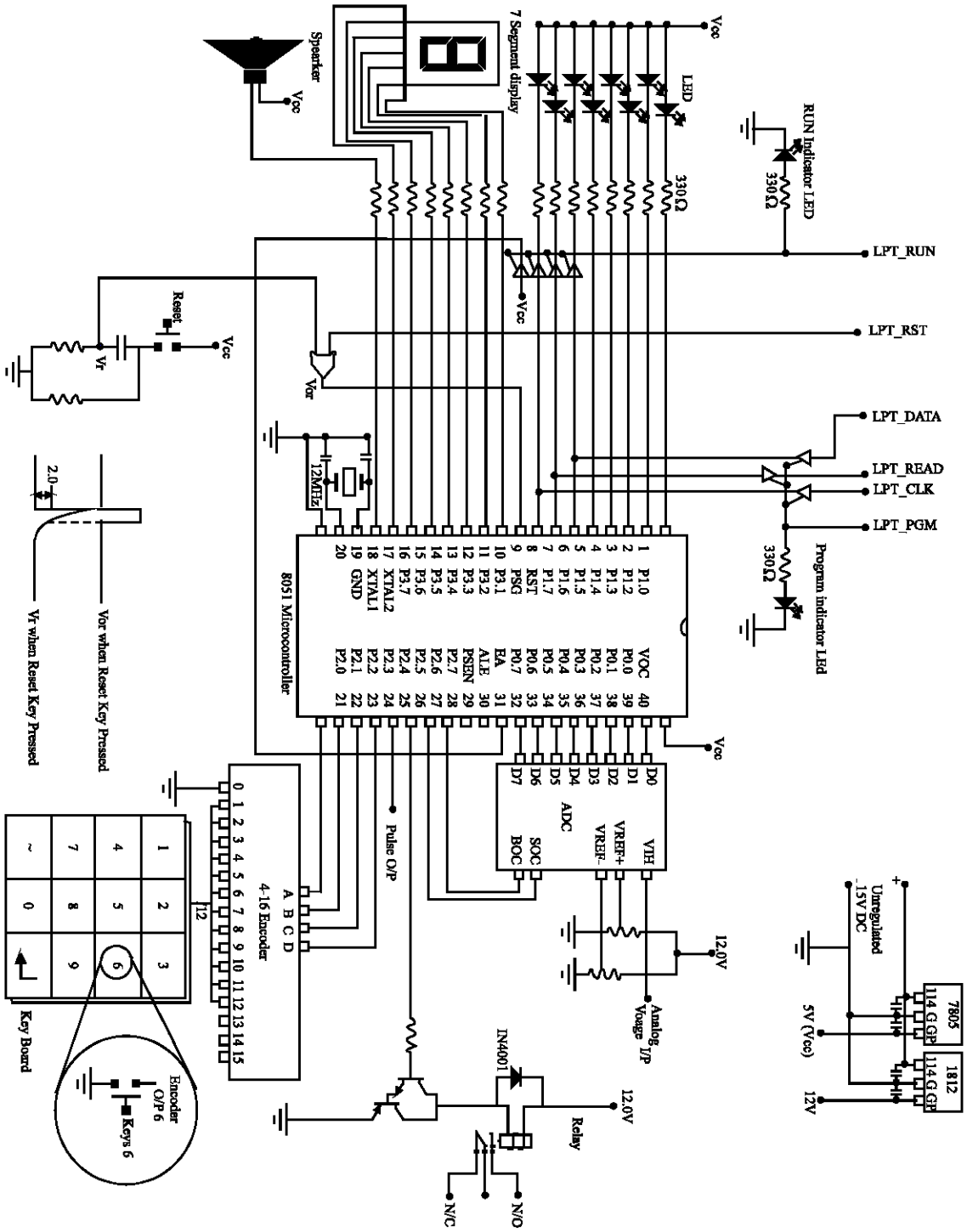

Fig. 2: 8051 Micro-controller kit

peripherals will not be disturbed at the time of programming.

(b) Run Mode: In this mode the LPT_RUN goes high and LPT_PGM goes to low. The tri-state switches connect the associated peripherals to 8051 and separates PCs LPT Programming pins from it so that PCs LPT port is not disturbed when we run our downloaded program in the kit. To start running the downloaded program, we can send a signal to LPT_RST using software, also we can restart the program by the external "Reset" switch from the hardware.

DESCRIPTION OF PERIPHERALS

LED: 8 LEDs are connected in port no 1. We can display binary numbers, or make other light flashing programs by glowing these LEDs. Here we connected the cathode pins of the LEDs with the 8051 ports and anode pins to Vcc (5v). When any pin in the port 1 gets low, the corresponding led glows sinking the current from the external source, not from the micro-controller. This will help the micro-controller to operate properly.

7 Segment display: A 7 segment display is connected to port 3 from pin no 3.0 to 3.6. We can display decimal numbers i.e., form 0-9 here. Here we used a Common Anode type 7-segment display so that it sinks current from the external source instead of from micro-controller.

Speaker: An 8Ω speaker is connected to port 3 at pin no 3.7. By sending square pulses of different frequencies we can generate different sounds with it.

Keyboard: A keyboard with 12 keys is also provided with the kit to get external signals from human. Here we used a 16-4 priority encoder[3] to interface the keyboard with the micro-controller[4]. The ABCD pins are connected with P2.0-P2.3 and the pins from 1-12 are connected with the keyboard. The construction of the keyboard is such that when a key is pressed say 6, the encoder pin 6 gets connected to ground and according to the truth table of the priority encoder, the binary value is written inversely in the ABCD pins. Here we connected key 1-9 to encoder pin 1-9, Key 0 to encoder pin 10 and key '←' and '~' to encoder pin 11 and 12 respectively. After reading the encoder value through our software we will invert it to get the actual key code. We will consider key code 0 as 'no key pressed'.

Pulse output: Here we kept an open pin P 2.4. We can send square pulses of different frequencies to this pin that can be used in other external circuits.

Relay: A 12 Volt relay is also interfaced with this kit to turn on and off external loads. It is connected with port no 2 at pin no 2.5. Here we used a Darlington pair to activate the relay coil. This gives more current than normal transistor. When we make pin 2.5 to high, the Darlington pair goes to saturation region and its collector and emitter conducts resulting the relay to trip.

Analog to Digital Converter (ADC): Using this ADC[5] interface we can insert external analog signals into the kit. It's VREF+ and VREF-pins are connected with two voltage divider circuits where variables resisters are used. This gives the flexibility to adjust the reference voltages manually. We can read the converted digital word from port 0 . Start of Conversion (SOC) and End of Conversion (EOC) pins are connected to pin 2.6 and 2.7 with the micro-controller to manage the A-D conversion timing.

PROGRAMMING THE KIT

Instruction set: 8051 micro-controller[2] accepts MCS-51® instruction set [6]. MCS-51® is a rich instruction set. It supports various instructions like:

- Data transfer instructions: MOV, MOVC etc.
- Mathematical operations: ADD, SUB, MUL, DIV, INC, DEC etc.
- Logical operations: ANL (AND), ORL (OR), XRL (XOR) etc.
- Program and control instructions: JMP, LJMP, SJMP, CALL, CJNE etc.

We can write the program in any text editor like notepad or edit and save our program as.asm file.

Compilation: We can compile our written program using asm51 cross assemblers. After compilation the following files are created:

- List file (.lst)
- Intel® Hex File (.hex)

The Intel® Hex file contains the absolute machine codes.

DOWNLOADING PROGRAM FROM PC TO KIT

After compilation we will download the machine codes from pc to the kit serially through the LPT port pins. We can write software in Microsoft™ Visual Basic® or in Microsoft™ Visual C++® to do these tasks. The key functions of the software are described below:

Opening hex file: User will be prompt to browse the created hex file to be opened. The software will then dissect the hex file and put the machine codes in the CodeBuffer.

Viewing buffer: User will be able to see the machine codes in the CodeBuffer in a CodeBuffer Window.

Writing to the Kit: User can download the machine codes by choosing this option. Here the software will place the CodeBuffer bytes in the format of ISP Programming Instruction set[1]. Then it will send the ISP instructions serially through the LPT_DATA pin. For Writing, the software will first erase the chip and then it will start sending data. We can use the output pins of LPT port (pin 2- pin 9) for LPT_DATA, LPT_CLK, LPT_PGM, LPT_RUN and LPT_RST.

Reading from Kit and Verification: User will also be able to read and verify their downloaded program. Here the software will send the read ISP instructions[1] and we can get the written data in the micro-controller from the LPT_READ pin. We can use LPT input pin (pin 10, Acknowledgment) as LPT_READ pin. The software will gather the bits form that pin and place them in a separate CodeBuffer. By comparing this CodeBuffer with the actual CodeBuffer we can verify whether our program was downloaded correctly or not.

Resetting: User will be able to reset and restart the microcontroller. This can be simply done by sending a high value in the LPT_RST pin for at least 2 machine cycles of the kit.

ALGORITHMS TO ACCESS PERIPHERALS

LED:

Input: $MEM\_LOCATION_{LED}$ ← Data
Function Name: GlowLED
{
Invert [$MEM\_LOCATION_{LED}$]
Send it to Port 1
}
Example:

```
ML_LED EQU 30H          ; set memory location
PORT1 EQU 90H           ; set port address
MOV     A, ML_LED       ; move led data to Accumula-
                          tor
XRL     A, 0FFH         ; get the inverted data
MOV     PORT1, A        ; write data to port1
END                     ; end of program
```

Segment display:

Input: $MEM\_LOCATION_{7\text{-}SEGMENT}$ ← Data
Function name: Glow7Segment
{
Make Inverted 7 Segment Look Up Table from Address LAdd
A_LAdd := LAdd + [$MEM\_LOCATION_{7\text{-}SEGMENT}$]
Send Content of A_LAdd to Port3
}

Speaker:

Input: $MEM\_LOCATION_{SPK\_FREQ}$ ← Frequency of the Sound wave
$MEM\_LOCATION_{SPK\_DURATION}$ ← Duration of the Sound wave
Function Name: Sound
{
TotalCycle = [$MEM\_LOCATION_{SPK\_DURATION}$] * [$MEM\_LOCATION_{SPK\_FREQ}$]
Cycle := 0
Do {
Send High to P3.7
Wait for half of 1 Cycle Time
Send Low to P3.7
Wait for half of 1 Cycle Time
Cycle := Cycle + 1
} While cycle <=Total cycle
}

Keyboard:

Function name: GetKey
{

Read data from Port2
XOR with FFH to invert data
AND with 0FH to keep only the 1st nibble
Place the data in $MEM\_LOCATION_{KEY}$
}
Output: $MEM\_LOCATION_{KEY}$ ← Key Code of currently pressed key

Pulse Output:

Input: $MEM\_LOCATION_{SPK\_FREQ}$ ← Frequency of the Sound wave
$MEM\_LOCATION_{SPK\_DURATION}$ ← Duration of the Sound wave
Function name: sound
{
TotalCycle = [$MEM\_LOCATION_{SPK\_DURATION}$] * [$MEM\_LOCATION_{SPK\_FREQ}$]
Cycle := 0
Do {
Send High to P3.7
Wait for half of 1 cycle time
Send Low to P3.7
Wait for half of 1 cycle time
Cycle := Cycle + 1
} While cycle <=total cycle
}

Relay:

Input: $MEM\_LOCATION_{RELAY}$← 1-On, 0-Off
Function Name: SetRelay
{
IF [$MEM\_LOCATION_{RELAY}$] = 1
THEN: Send high to P2.5
ELSE: Send low to P2.5
}

Analog to digital converter:

Function name: GetAtoD
{
Send start conversion signal to P2.6
Do
Read end conversion signal from P2.7
Until it gets the end conversion signal from P2.7
Read the data of port 0
Place the data to $MEM\_LOCATION_{A-D}$
}

Output: $MEM\_LOCATION_{A-D}$ ← Converted digital value

SOFTWARE UTILITIES

The User/Learner can write their own Operating System for the kit that will be responsible for execution of different application programs. Varity of application programs can be written i.e. Alarm Clock, Calculator, Phone Book, Music generator (i.e., Tini Piano, Small games etc.). Moreover by using its relay control unit, we can write programs, which can control High AC voltage peripherals. The cost taken for the entire kit is approximately USD 20 while the cost for commercial micro-controller kit such as MINIMAX 51, Kanda and Goal Semi Conductor are USD 69, 95, 99 respectively.

FUTURE PLANS

We have successfully developed the hardware and software for the kit. In future, we are considering constructing a micro-controller kit with LCD Display, Matrix Keyboard, EEPROM[7] and RTC (Real Time Clock). A 16×4 Character LCD can be interfaced with the kit using 4 data wire mode. A 4×4 Matrix keyboard[3] can be interfaced in an 8-bit port to get external signals from human. To store the contents of RAM even if the power is off or to store some permanent data, an EEPROM can be used. By implementing RTC in the micro-controller kit, it is possible to keep track of current date and time.

CONCLUSION

This study shows a way to construct a cost effective 8051 micro-controller based learning kit covering all the major features. It is cheap and provides some features that are required for the present micro-controller based applications. We believe that this paper will be a helpful document for any one who wants to work with micro-controller kit.

REFERENCES

1. Atmel 8051 Microcontroller Datasheet, Available at http://www.atmel.com
2. Muhammad, A., Mazidi and J. Gillispie Mazidi, 2002. The 8051 Micro-controller and Embedded Systems, Pearson Education Inc., First Edn.
3. Malvino and Brown, 1995. Digital computer electronics, Tata McGraw-Hill, ISBN 0-07-462235-8.
4. Doughlas, V.H., 1992. Microprocessors and Interfacing; Programming and Hardware, Mcgraw-Hill.

5. Ramesh, S.G., 1993. Microprocessor, Architecture, Programming and Applications with the 8085/8080A, Wiley Eastern Limited.
6. Barry, B.B., 2002. The Intel Microprocessors: 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro Processor, Pentium 2, Pentium 3 and Pentium 4-Architecture, Programming and Interfacing, Prentice-Hall,.
7. Ronald, J.T., 1996. Digital System, Principles and Applications, Prentice Hall of India, Sixth Edition.