INSTITUTE OF AERONAUTICAL ENGINEERING
(Autonomous)
Dundigal, Hyderabad - 500 043

Lab Manual:

EMBEDDED SYSTEMS LABORATORY (AECB11)

Prepared by

S Lakshmanachari (IARE10065)

Electronics and Communication Engineering
Institute of Aeronautical Engineering

July 31, 2021

Contents

Content **v**

1 Course Information **2**
1.1 Introduction . . . 2
1.1.1 Student Responsibilities . . . 2
1.1.2 Laboratory Teaching Assistant Responsibilities . . . 3
1.1.3 Faculty Coordinator Responsibilities . . . 3
1.1.4 Lab Policy and Grading . . . 3
1.1.5 Course Goals and Objectives . . . 4
1.2 Use of Laboratory Instruments . . . 5
1.2.1 Instrument Protection Rules . . . 5
1.3 Data Recording and Reports . . . 6
1.3.1 The Laboratory Worksheets . . . 6
1.3.2 The Lab Report . . . 6

2 LAB-1 ORIENTATION **8**
2.1 Introduction . . . 8
2.2 Objective . . . 8
2.3 Prelab Preparation: . . . 8
2.4 Equipment needed . . . 8
2.4.1 Hardware Requirements: . . . 8
2.4.2 Software Requirements: . . . 8
2.5 Procedure . . . 8

3 LAB-2 Develop Program using KEIL IDE Tool **10**
3.1 Introduction . . . 10
3.2 Objective . . . 10
3.3 Prelab Preparation: . . . 10
3.4 Equipment needed . . . 10
3.4.1 Hardware Requirements: . . . 10
3.4.2 Software Requirements: . . . 10
3.5 Background . . . 11
3.6 Procedure . . . 11
3.7 Precautions . . . 16
3.8 Result . . . 16
3.9 Further Probing Experiments . . . 17

4 LAB-3 INTERFACING LED WITH DIFFERENT PORT PINS **18**
4.1 Introduction . . . 18
4.2 Objective . . . 18
4.2.1 Educational . . . 18
4.2.2 Experimental . . . 18
4.3 Prelab Preparation: . . . 18

4.3.1 Reading . . . 18
4.3.2 Written . . . 18
4.4 Equipment needed . . . 18
4.4.1 Hardware Requirements: . . . 18
4.4.2 Software Requirements: . . . 19
4.5 Background . . . 19
4.6 Procedure . . . 20
4.7 Precautions . . . 21
4.8 Result . . . 21
4.9 Further Probing Experiments . . . 22

5 LAB-4 INTERFACING BUZZER AND SWITCH 23
5.1 Introduction . . . 23
5.2 Objective . . . 23
5.2.1 Educational . . . 23
5.2.2 Experimental . . . 23
5.3 Prelab Preparation: . . . 23
5.3.1 Reading . . . 23
5.3.2 Written . . . 23
5.4 Equipment needed . . . 23
5.4.1 Hardware Requirements: . . . 23
5.4.2 Software Requirements: . . . 24
5.5 Background . . . 24
5.6 Procedure . . . 25
5.7 Precautions . . . 26
5.8 Result . . . 26
5.9 Further Probing Experiments . . . 27

6 LAB-5 INTERFACING LCD DISPLAY 28
6.1 Introduction . . . 28
6.2 Objective . . . 28
6.2.1 Educational . . . 28
6.2.2 Experimental . . . 28
6.3 Prelab Preparation: . . . 28
6.3.1 Reading . . . 28
6.3.2 Written . . . 28
6.4 Equipment needed . . . 28
6.4.1 Hardware Requirements: . . . 28
6.4.2 Software Requirements: . . . 29
6.5 Background . . . 29
6.6 Procedure . . . 32
6.7 Precautions . . . 33
6.8 Result . . . 33
6.9 Further Probing Experiments . . . 34

7 LAB-6 INTERFACING HEXA KEYPAD 35
7.1 Introduction . . . 35
7.2 Objective . . . 35
7.2.1 Educational . . . 35
7.2.2 Experimental . . . 35
7.3 Prelab Preparation: . . . 35
7.3.1 Reading . . . 35

7.3.2 Written . . . 35
7.4 Equipment needed . . . 35
7.4.1 Hardware Requirements: . . . 35
7.4.2 Software Requirements: . . . 36
7.5 Background . . . 36
7.6 Procedure . . . 42
7.7 Precautions . . . 42
7.8 Result . . . 42
7.9 Further Probing Experiments . . . 43

8 LAB-7 INTERFACING SEVEN SEGMENT DISPLAY 44
8.1 Introduction . . . 44
8.2 Objective . . . 44
8.2.1 Educational . . . 44
8.2.2 Experimental . . . 44
8.3 Prelab Preparation: . . . 44
8.3.1 Reading . . . 44
8.3.2 Written . . . 44
8.4 Equipment needed . . . 44
8.4.1 Hardware Requirements: . . . 44
8.4.2 Software Requirements: . . . 45
8.5 Background . . . 45
8.6 Procedure . . . 48
8.7 Precautions . . . 49
8.8 Result . . . 49
8.9 Further Probing Experiments . . . 50

9 LAB-8 INTERFACING STEPPER MOTOR 51
9.1 Introduction . . . 51
9.2 Objective . . . 51
9.2.1 Educational . . . 51
9.2.2 Experimental . . . 51
9.3 Prelab Preparation: . . . 51
9.3.1 Reading . . . 51
9.3.2 Written . . . 51
9.4 Equipment needed . . . 51
9.4.1 Hardware Requirements: . . . 51
9.4.2 Software Requirements: . . . 52
9.5 Background . . . 52
9.6 Procedure . . . 55
9.7 Precautions . . . 56
9.8 Result . . . 56
9.9 Further Probing Experiments . . . 57

10 LAB-9 INTERFACING ANALOG TO DIGITAL CONVERTER (ADC) 58
10.1 Introduction . . . 58
10.2 Objective . . . 58
10.2.1 Educational . . . 58
10.2.2 Experimental . . . 58
10.3 Prelab Preparation: . . . 58
10.3.1 Reading . . . 58
10.3.2 Written . . . 58

10.4 Equipment needed . . . 59
10.4.1 Hardware Requirements: . . . 59
10.4.2 Software Requirements: . . . 59
10.5 Background . . . 59
10.6 Procedure . . . 61
10.7 Precautions . . . 62
10.8 Result . . . 62
10.9 Further Probing Experiments . . . 63

11 LAB-10 INTERFACING DIGITAL TO ANALOG CONVERTER (DAC) 64
11.1 Introduction . . . 64
11.2 Objective . . . 64
11.2.1 Educational . . . 64
11.2.2 Experimental . . . 64
11.3 Prelab Preparation: . . . 64
11.3.1 Reading . . . 64
11.3.2 Written . . . 64
11.4 Equipment needed . . . 65
11.4.1 Hardware Requirements: . . . 65
11.4.2 Software Requirements: . . . 65
11.5 Background . . . 65
11.6 Procedure . . . 68
11.7 Precautions . . . 68
11.8 Result . . . 68
11.9 Further Probing Experiments . . . 69

12 LAB-11 INTERFACE RELAY 70
12.1 Introduction . . . 70
12.2 Objective . . . 70
12.2.1 Educational . . . 70
12.2.2 Experimental . . . 70
12.3 Prelab Preparation: . . . 70
12.3.1 Reading . . . 70
12.3.2 Written . . . 70
12.4 Equipment needed . . . 71
12.4.1 Hardware Requirements: . . . 71
12.4.2 Software Requirements: . . . 71
12.5 Background . . . 71
12.6 Procedure . . . 74
12.7 Precautions . . . 74
12.8 Result . . . 75
12.9 Further Probing Experiments . . . 75

13 LAB-12 SERIAL COMMUNICATION INTERFACING FROM MICRO-CONTROLLER TO PC 76
13.1 Introduction . . . 76
13.2 Objective . . . 76
13.2.1 Educational . . . 76
13.2.2 Experimental . . . 76
13.3 Prelab Preparation: . . . 76
13.3.1 Reading . . . 76
13.3.2 Written . . . 76

13.4 Equipment needed . . . 77
13.4.1 Hardware Requirements: . . . 77
13.4.2 Software Requirements: . . . 77
13.5 Background . . . 77
13.6 Procedure . . . 79
13.7 Precautions . . . 80
13.8 Result . . . 80
13.9 Further Probing Experiments . . . 81

14 LAB-13 INTERFACING OF TEMPRATURE SENSOR, LM35 WITH 8051 MICROCONTROLLER 82
14.1 Introduction . . . 82
14.2 Objective . . . 82
14.2.1 Educational . . . 82
14.2.2 Experimental . . . 82
14.3 Prelab Preparation: . . . 82
14.3.1 Reading . . . 82
14.3.2 Written . . . 82
14.4 Equipment needed . . . 82
14.4.1 Hardware Requirements: . . . 82
14.4.2 Software Requirements: . . . 83
14.5 Background . . . 83
14.6 Procedure . . . 89
14.7 Precautions . . . 90
14.8 Result . . . 90
14.9 Further Probing Experiments . . . 91

A Appendix A - Safety, Do's and Don'ts 92

B Appendix B - Pin Description and Block diagram of 8051 Microcontroller 98

C Appendix C - Proteus 8.0 Professional Simulation 102

;

Course Information

1.1 Introduction

This course is intended to enhance the learning experience of the student in topics encountered in AEC111. In this lab, gives the general awareness of the 8051 Trainer board, writing programs using embedded C and Keil µvision tool for simulation of embedded systems. Students are expected to gain experience in using the basic input/output devices used in electronics and communication engineering and in interpreting the results of measurement operations in terms of the concepts introduced in the Embedded Systems course. How the student performs in the lab depends on his/her preparation, participation, and teamwork. Each team member must participate in all aspects of the lab to insure a thorough understanding of the equipment and concepts. The student, lab assistant, and faculty coordinator all have certain responsibilities toward successful completion of the lab's goals and objectives.

1.1.1 Student Responsibilities

The student is expected to be prepared for each lab. Lab preparation includes reading the lab experiment and related textbook material. If you have questions or problems with the preparation, contact your Laboratory teaching Faculty (LTF), but in a timely manner. Do not wait until an hour or two before the lab and then expect the LTF to be immediately available.

A large portion of the student's grade is determined in the comprehensive final exam, resulting in a requirement of understanding the concepts and procedure of each lab experiment for the successful completion of the lab class. The student should remain alert and use common sense while performing a lab experiment. They are also responsible for keeping a professional and accurate record of the lab experiments in the lab manual wherever tables are provided. Students should report any errors in the lab manual to the teaching Faculty.

The student responsibilities are:

1. Students are required to attend all labs.
2. Students should work individually while performing lab experiments.
3. Students have to bring the lab worksheet whenever they come for lab work.
4. Should take only the lab worksheet, calculator (if needed) and a pen or pencil to the work area.
5. Should learn the prelab questions. Read through the lab experiment to familiarize themselves with the basics of embedded C and 8051 Trainer board.
6. Should utilize 3 hour's time properly to perform the experiment and to verify the results. Do the experiments, draw the graphs if required and take the signature from the faculty.
7. If the experiment is not completed in the stipulated time, the pending work has to be carried out in the leisure hours or extended hours.

8. For practical subjects there shall be a continuous evaluation during the semester for 30 sessional marks and 70 end examination marks.

9. Out of 30 internal marks, 20 marks shall be awarded for day-to-day work and 10 marks to be awarded by conducting an internal laboratory test.

1.1.2 Laboratory Teaching Assistant Responsibilities

The LTF shall be completely familiar with each lab prior to class. The LTF shall provide the students with a syllabus and safety review during the first class. The LTF is responsible for ensuring that all the necessary equipment and/or preparations for the lab are available and in working condition. Lab experiments should be checked in advance to make sure everything is in working order. The LTF should fully answer any questions posed by the students and supervise the students performing the lab experiments. The LTF is expected to award the marks in the lab worksheets in a fair and timely manner. The marks awarded lab worksheets should uploaded in student portal at the same day.

1.1.3 Faculty Coordinator Responsibilities

The faculty coordinator should ensure that the laboratory is properly equipped, i.e., that the teaching assistants receive any equipment necessary to perform the experiments. The coordinator is responsible for supervising the teaching assistants and resolving any questions or problems that are identified by the teaching assistants or the students. The coordinator may supervise the format of the final exam for the lab. They are also responsible for making any necessary corrections to this manual and ensuring that it is continually updated and available.

1.1.4 Lab Policy and Grading

The student should understand the following policy:

ATTENDANCE: Attendance is mandatory for all lab hours as per the academic regulations.

LAB RECORD's: The student must:

1. Thoroughly read the prelab of corresponding experiment before entering to the lab.

2. Come with full preparation of corresponding lab experiment, after reaching the lab you should write the worksheet without seeing the lab manual.

3. Write the all the tabular columns and related calculations after completion of experiment.

4. Take the lab work sheet correction before leaving the lab.

GRADING POLICY:

1. The every week lab worksheet is evaluated for 20 Marks.

2. The 20 Marks are sub dived to 5 parts. i.e. preparation, algorithm/circuit design, program / procedure, execution and Viva. Each part is assigned for 4 marks.

3. Based the student performance the lab, the faculty grade the marks out of 20 marks in each lab.

4. Finally, the faculty grade the marks for day to day evaluation by making the average of conducted laboratory sessions.

5. 10 marks are evaluated from the lab internal exam.

PRE-REQUISTES AND CO-REQUISTIES: The lab course is to be taken during the same semester as AEC111, but receives a separate grade.

1.1.5 Course Goals and Objectives

The Embedded Systems Laboratory is designed to provide the student with the knowledge to use basic input/output devices and techniques with proficiency. These techniques are designed to complement the concepts introduced in AEC016. In addition, the student should learn how to record experimental results effectively and present these results in a lab worksheet.

More explicitly, the class objectives are:

- To gain proficiency in the usage of Keil IDE Tool.
- To enhance understanding the theory of embedded systems concepts including:
  - Operating systems basics,
  - Task scheduling
  - Embedded software development tools,
  - Multiprocessing and multitasking,
  - Task communication/synchronization issues,
  - Advance processors.
- To develop communication skills through:
  - Maintenance of succinct but complete laboratory notebooks as permanent, written descriptions of procedures, results, and analyses.
  - Verbal interchanges with the laboratory instructor and other students.
  - Preparation of succinct but complete laboratory reports.
- To compare theoretical predictions with experimental results and to determine the source of any apparent errors.

1.2 Use of Laboratory Instruments

One of the major goals of this lab is to familiarize the student with the interfacing of various input/output (I/O) devices with the 8051 microcontroller. Some understanding of the lab instruments is necessary to avoid personal or equipment damage. By understanding the device's purpose and following a few simple rules, costly mistakes can be avoided. Most of the instrumentation used in this laboratory is implemented through the interfacing modules, Personal computer, and Cathode ray oscilloscope.

In general, all devices have physical limits. These limits are specified by the device manufacturer and are referred to as the device rating. The ratings are usually expressed in terms of voltage limits, current limits, or power limits. It is up to the engineer to make sure that in device operation, these ratings (limit values) are not exceeded. The following rules provide a guideline for instrument protection.

1.2.1 Instrument Protection Rules

1. Setup Keil μVision 4.0 or above software in personal computer.
2. Be sure the Keil μVision 4.0 or above is installed properly.
3. To initiate the programming you must create a project using the Keil μVision IDE.
4. Selecting the type of device you are working with.
5. Adding C files to your project.
6. Compiling and building the C project using Keil μVision IDE.
7. Generating the hex file using Keil μVision IDE.
8. Burning the hex code into 8051 microcontroller using flash magic.

1.3 Data Recording and Reports

1.3.1 The Laboratory Worksheets

Students must record their experimental values in the provided tables in this laboratory manual and reproduce them in the lab worksheets. Worksheets are integral to recording the methodology and results of an experiment. In engineering practice, the laboratory worksheet serves as an invaluable reference to the technique used in the lab and is essential when trying to duplicate a result or write a report. Therefore, it is important to learn to keep accurate data. Make plots of data and sketches when these are appropriate in the recording and analysis of observations. Note that the data collected will be an accurate and permanent record of the data obtained during the experiment and the analysis of the results. You will need this record when you are ready to prepare a lab worksheet.

1.3.2 The Lab Report

Reports are the primary means of communicating your experience and conclusions to other professionals. In this course you will use the lab report to inform your LTA about what you did and what you have learned from the experience. Engineering results are meaningless unless they can be communicated to others. You will be directed by your LTA to prepare a lab report on a few selected lab experiments during the semester.

Your laboratory report should be clear and concise. The lab report shall be typed on a word processor. As a guide, use the format on the next page. Use tables, diagrams, sketches, and plots, as necessary to show what you did, what was observed, and what conclusions you can draw from this. Even though you will work with one or more lab partners, your report will be the result of your individual effort in order to provide you with practice in technical communication.

Formatting and Style

- The lab report shall be hand written in a lab worksheet.
- The first line of each paragraph should have a left indent.
- All the tables should have titles and should be numbered. Tables should be labeled numerically as Table 1, Table 2, etc. Table captions appear above the table.
- Graphs should be presented as figures. All the figures should have titles and should be numbered. Figure captions appear below the figure. Graphs should have labeled axes and clearly show the scales and units of the axes.
- All the figures and tables must be centered on the page.
- Do not place screenshots of your lab worksheet.

Order of Lab Report Components

COVER PAGE: Cover page must include lab name and number, your name, and the date the lab was performed.
OBJECTIVE: Clearly state the experiment objective in your own words.
Prelab: Indicate the required knowledge for performing the concerned experiment.
EQUIPMENT USED: Indicate which equipment was used in performing the experiment.
BACKGROUND: Give the brief information about corresponding lab experiment.
PROCEDURE: Describe the experimental set up and procedure to perform the experiment.
RESULT: After completion of experiment, provide a summary of what was learned from this

part of the laboratory experiment.
PROBING FURTHER EXPERIMENTS: Questions pertaining to this lab must be answered at the end of laboratory report.

LAB-1 ORIENTATION

2.1 Introduction

In the first lab period, the students should become familiar with the location of equipment and components in the lab, the course requirements, and the teaching instructor. Students should also make sure that they have all of the co-requisites and pre-requisites for the course at this time.

2.2 Objective

To familiarize the students with the lab facilities, equipment, standard operating procedures, lab safety, and the course requirements.

2.3 Prelab Preparation:

Read the Introduction and Appendix A, of this manual. Download and install the "Keil µvision IDE" and "Proteus 8.0 " software on your personal computer.

2.4 Equipment needed

2.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

2.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

2.5 Procedure

1. During the first laboratory period, the instructor will provide the students with a general idea of what is expected from them in this course. Each student will receive a copy of the syllabus, stating the instructor's contact information. In addition, the instructor will review the safety concepts of the course.

2. During this period, the instructor will briefly review the equipment which will be used throughout the semester. The location of instruments, equipment, and components (e.g. resistors, capacitors, connecting wiring) will be indicated. The guidelines for instrument use will be reviewed.

LAB-2 Develop Program using KEIL IDE Tool

3.1 Introduction

Keil µVision is free software which solves many of the pain points for an embedded program developer. This software is an integrated development environment (IDE), which integrated a text editor to write programs, a compiler and it will convert your source code to hex files too. Here is simple guide to start working with Keil Micro Vision which can be used for

- Writing programs in C/C++ or Assembly language
- Compiling and Assembling Programs
- Debugging program
- Creating Hex file
- Testing your program without Available real Hardware (Simulator Mode)

3.2 Objective

By the end of this lab, the student should learn how to work with the Keil µVision software. Keil C51 is the industry-standard tool chain for all 8051-compatible devices, it supports classic 8051, Dallas 390, NXP MX, extended 8051 variants, and C251 devices. The µVision IDE and debugger integrates complete device simulation, interfaces too many target debug adapters, and provides various monitor debug solutions.

3.3 Prelab Preparation:

Read the Introduction and Appendix C, of this manual. Download and install the "Keil µvision IDE" and "Proteus 8.0 " software on your personal computer.

3.4 Equipment needed

3.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

3.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.

3. Proteus 8 Professional.

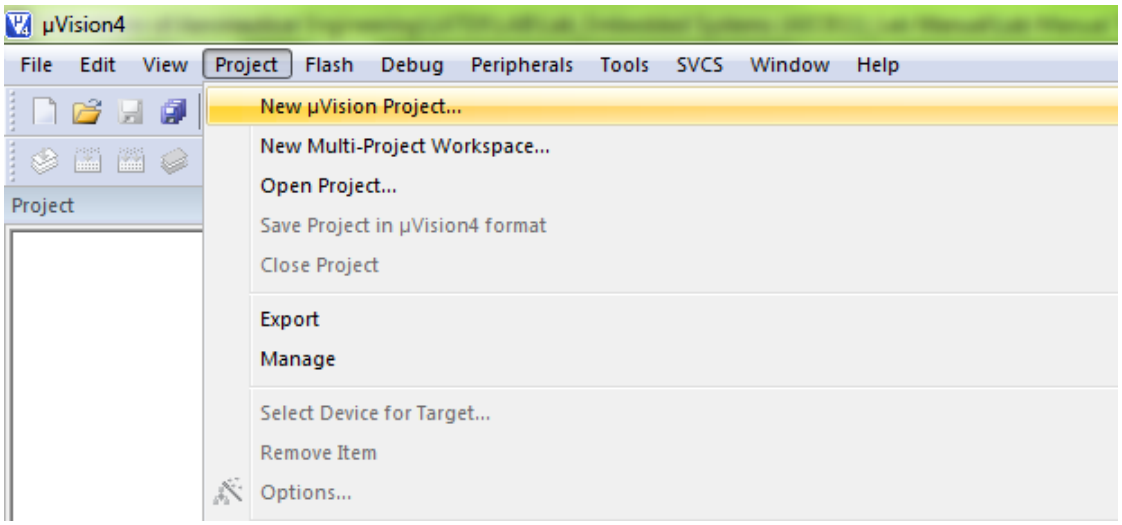
3.5 Background

The µVision IDE combines project management, run-time environment, build facilities, source code editing, and program debugging in a single powerful environment. µVision is easy-to-use and accelerates your embedded software development. µVision supports multiple screens and allows you to create individual window layouts anywhere on the visual surface.

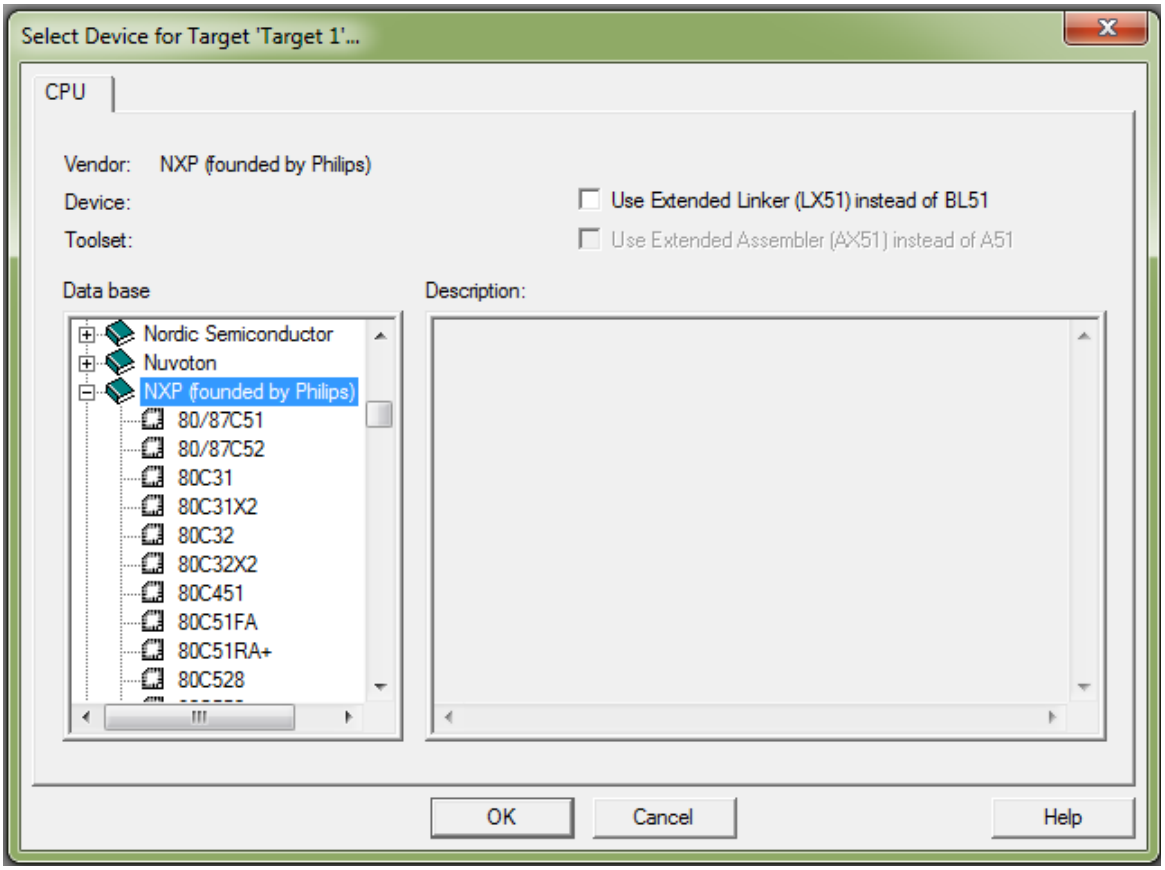
3.6 Procedure

These are the simple steps to get off the mark your inning!

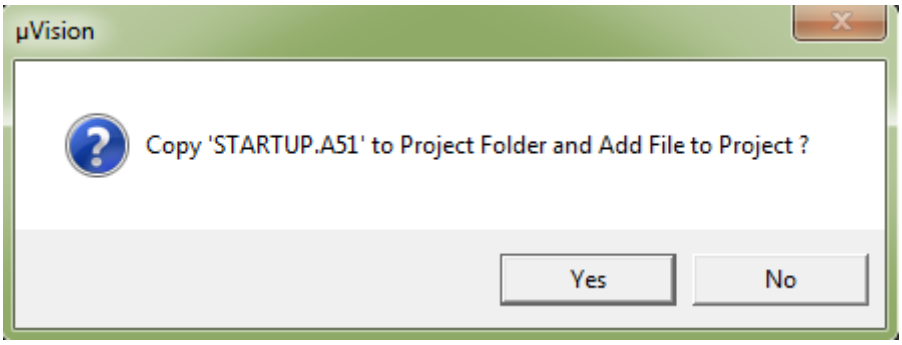
Step 1: After opening Keil µV4, Go to Project tab and Create new µVision project
Now Select new folder and give name to Project.



Step 2: After Creating project now Select your device model. Example.NXP-P89V51RD2 (You can change it later from project window.)



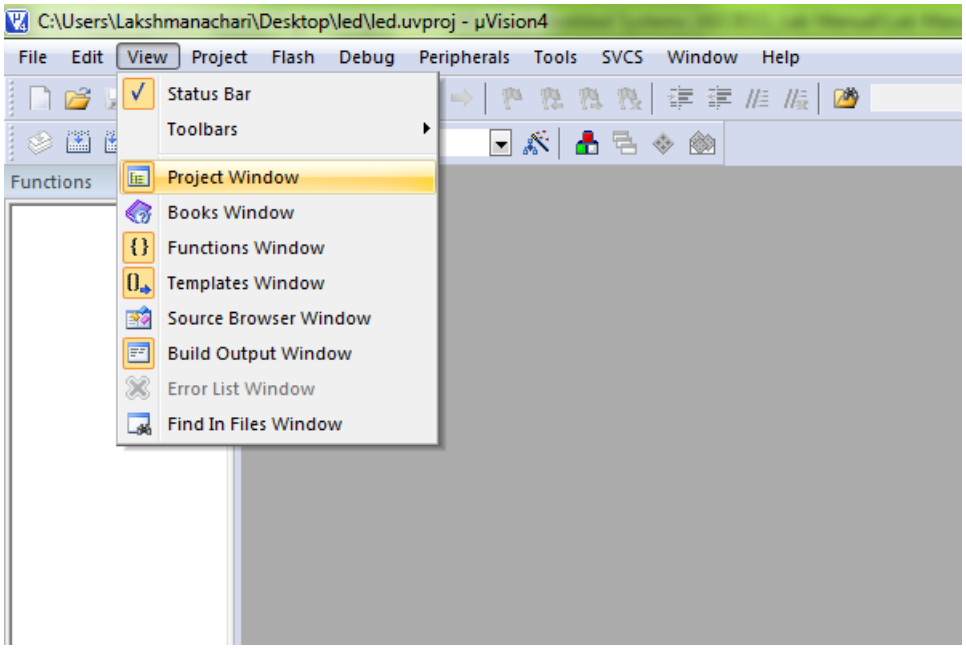
Step 3: So now your project is created and Message window will appear to add startup file of your Device click on Yes so it will be added to your project folder.



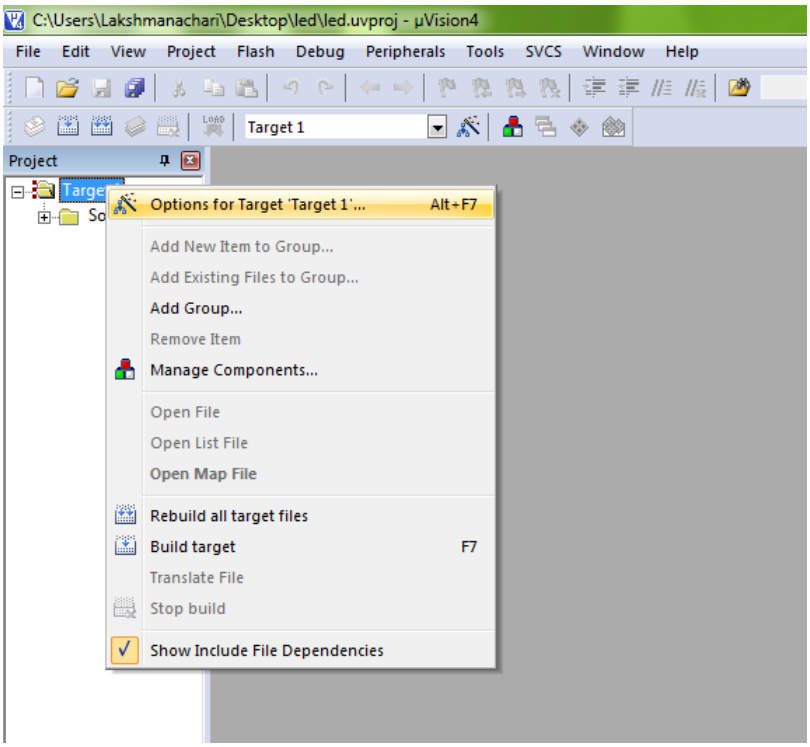
Step 4: Now go to File and create new file and save it with .C extension if you will write program in C language or save with .asm for assembly language.
i.e., Led.c

Step 5: Now write your program and save it again.

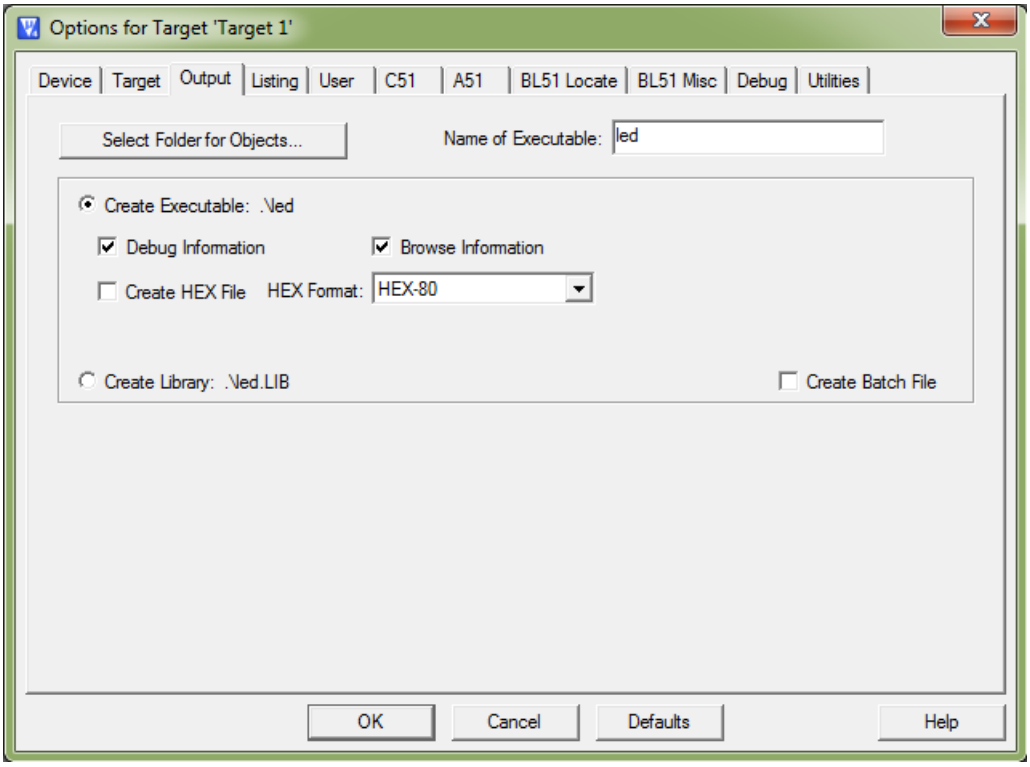
Step 6: After that on left you see project window [if it's not there. . . .go to View tab and click on project window] Now come on Project window.



Right click on target and click on options for target.

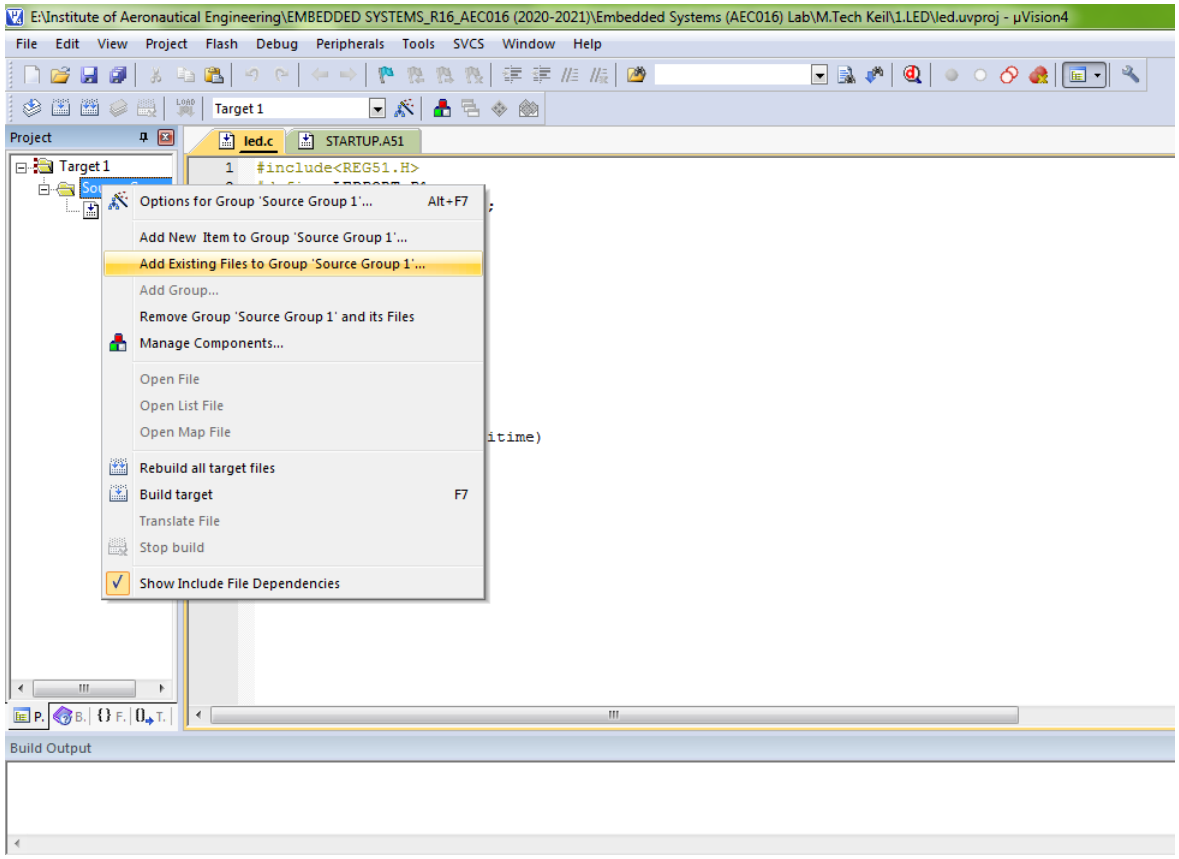


A new window will appear shown below and here you can change your device also.



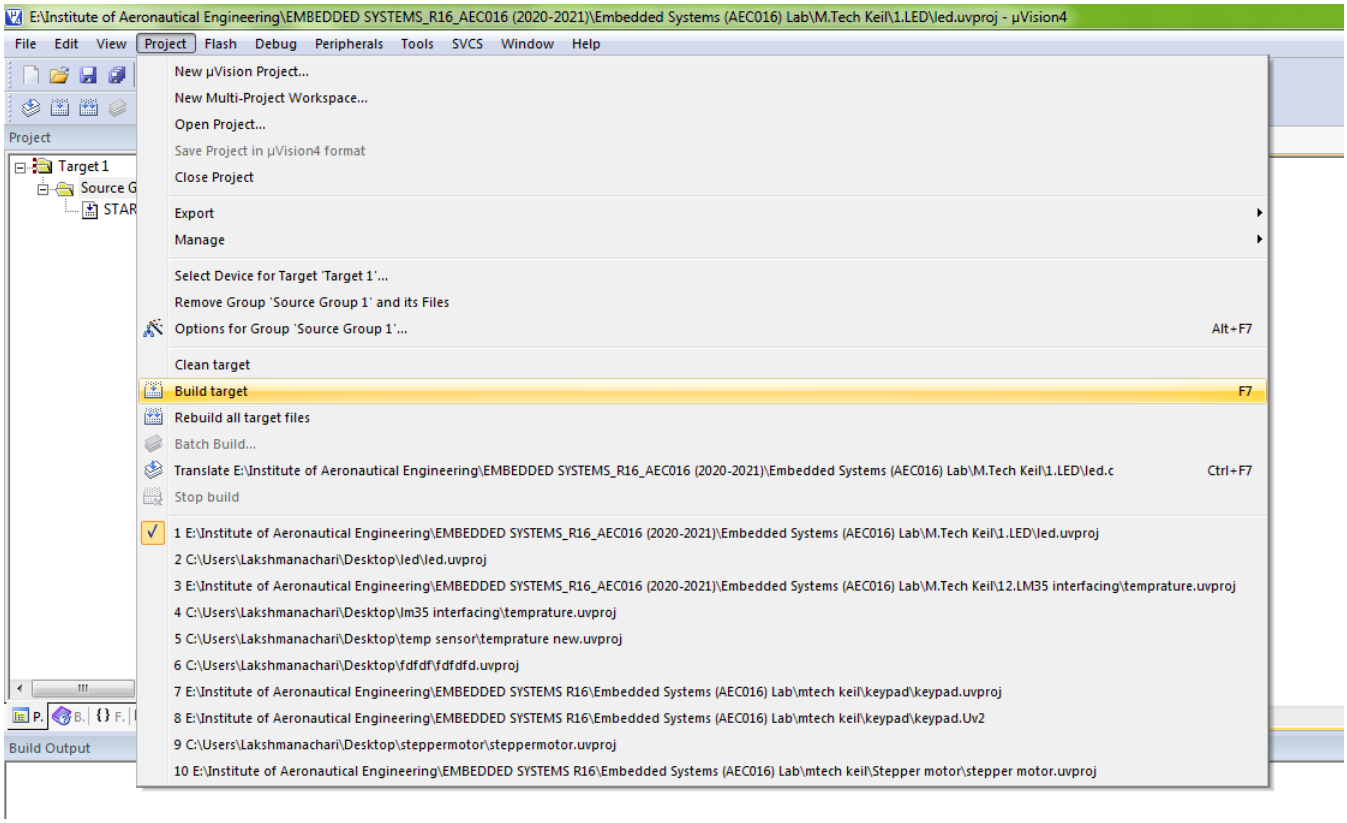
Click output tab here and check create Hex file if you want to generate hex file Now click on ok so it will save changes.

Step 7: Now Expand target and you will see source group
Right click on source group and click on Add files to source group

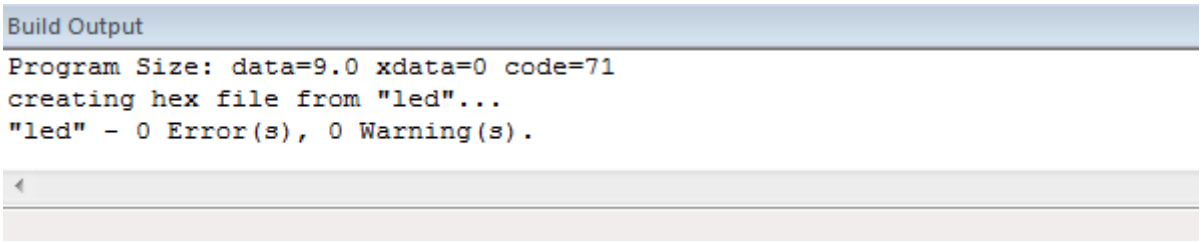


Now add your program file which you have written in C/assembly.
You can see program file added under source group.

Step 8: Now Click on Build target. You can find it under Project tab or in toolbar. It can also be done by pressing F7 key.



Step 9: you can see Status of your program in Build output window
(If it's not there go to view and click on Build output window)



Now you are done with your program. Next time we will look at Debugging and Simulation of Program.

3.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Observe the results carefully.

3.8 Result

3.9 Further Probing Experiments

Q1. What are embedded systems and its applications?

Q2. What is the use of Keil IDE software?

Q3. What is the use of flash magic?

LAB-3 INTERFACING LED WITH DIFFERENT PORT PINS

4.1 Introduction

The main principle of the circuit is to interface LEDs to the 8051 family micro controller. Commonly, used LEDs will have voltage drop of 1.7v and current of 10mA to glow at full intensity. This is applied through the output pin of the micro controller. The Light Emitting Diode is similar to the normal PN diode but it emits energy in the form of light. The color of light depends on the band gap of the semiconductor. This experiment aims to toggle all the bits of Port1 continuously using the software keil u vision 4 or 5. Firstly the header files reg51.H is declared for the intended 8051. Then the main function starts. In the main function an infinite for loop starts where each time 0x00 followed by 0xFF is sent to Port1 and the loop goes on.

4.2 Objective

4.2.1 Educational

- Learn the working principle of LED.
- Understand how LED emits light.
- Learn to simulate Keil IDE software.

4.2.2 Experimental

- Explain the interfacing of LED with 8051 microcontroller.
- Understand the Keil IDE simulation procedures.
- Understand the hardware connections of 8051 Microcontroller trainer board.

4.3 Prelab Preparation:

4.3.1 Reading

- Read and study the Background and procedure sections of this Laboratory.

4.3.2 Written

- After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

4.4 Equipment needed

4.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.

3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

4.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

4.5 Background

LEDs Interfacing with 8051:

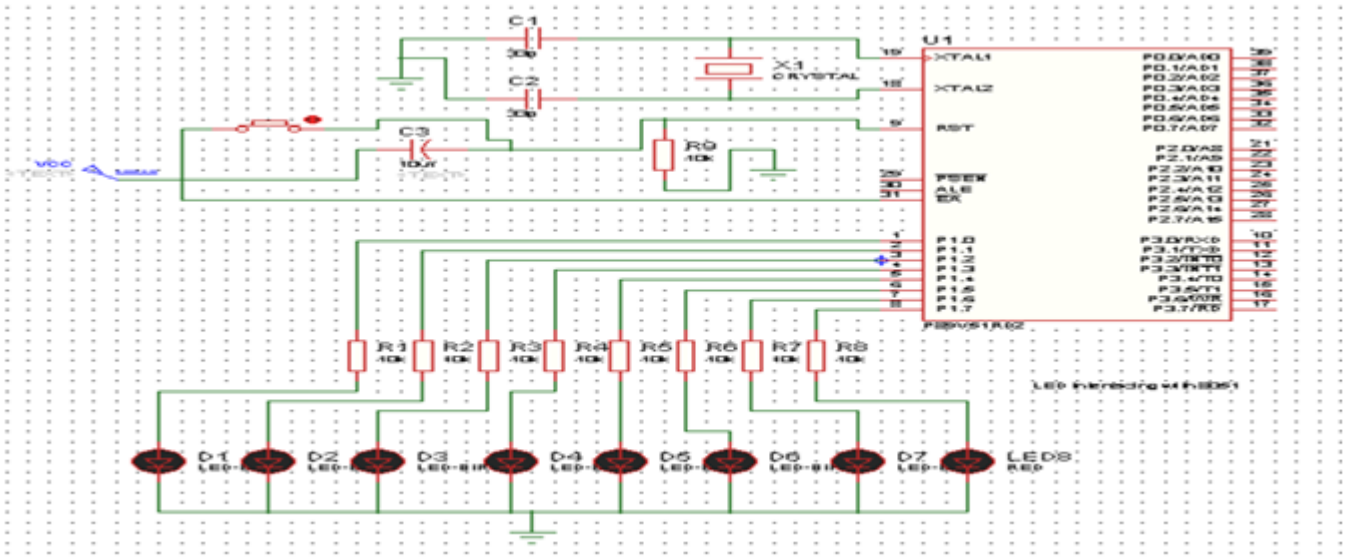


Figure 4.1: Interfacing LED's with different port pins of 8051 Microcontroller

Source Code:
/* Program to toggle all the bits of Port P1 continuously with 250 mS delay. */

```
#include<REG51.H>
#define LEDPORT P1
void delay(unsigned int);
void main(void)
{
        LEDPORT =0x00;
        while(1)
        {
                LEDPORT =       0x00;
                delay(250);
                LEDPORT = 0xff;
                delay(250);
```

```
        }
    }
    void delay(unsigned int itime)
    {
        unsigned int i,j;
        for(i=0;i<itime;i++);
        {
            for(j=0;j<1275;j++);
        }
    }
```

Flowchart:

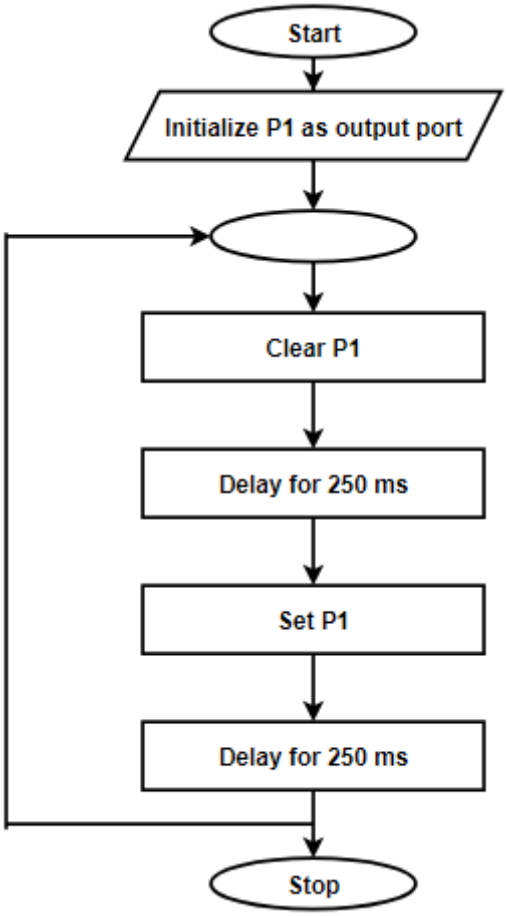


Figure 4.2: Flowchart

4.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

4.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Observe the results carefully.

4.8 Result

4.9 Further Probing Experiments

Q1. Program to toggle the odd number of bits of Port, P1 continuously with 250 ms delay.

Q2. Program to toggle the P0.5 bit with 500 ms delay.

Q3. Program to toggle the even number of bits of Port, P1 continuously with 100 ms delay.

LAB-4 INTERFACING BUZZER AND SWITCH

5.1 Introduction

The buzzer is interfaced to Port P3.6 of the microcontroller and switch is interfaced to Port P1.4. The simple ON/OFF program is written as an example using delay in between them. Whenever switch is pressed the buzzer will give sound and whenever it is released buzzer will OFF. One can also use the switch concept to control the ON/OFF operation of buzzer. Here, switch is connected to Port P3.6 pin and Debouncing concept is applied.

5.2 Objective

5.2.1 Educational

• Learn to interface buzzer and switch with 8051 microcontroller.
• Learn the pin configuration of 8051 microcontroller.
• Learn to use Keil μVision IDE tool to create a project using 8051 microcontroller.

5.2.2 Experimental

• Observe all the components/modules in the 8051 trainer board.
• Understand the hardware connections of interfacing of buzzer and switch with 8051 microcontroller.
• Observe the sound from buzzer when the switch is in pressed state.

5.3 Prelab Preparation:

5.3.1 Reading

• Read and study the Background section of this Laboratory.
• Learn to draw the switch and buzzer interfacing circuit with 8051 microcontroller.

5.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

5.4 Equipment needed

5.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.

4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

5.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

5.5 Background

Interfacing Switch and Buzzer with 8051:

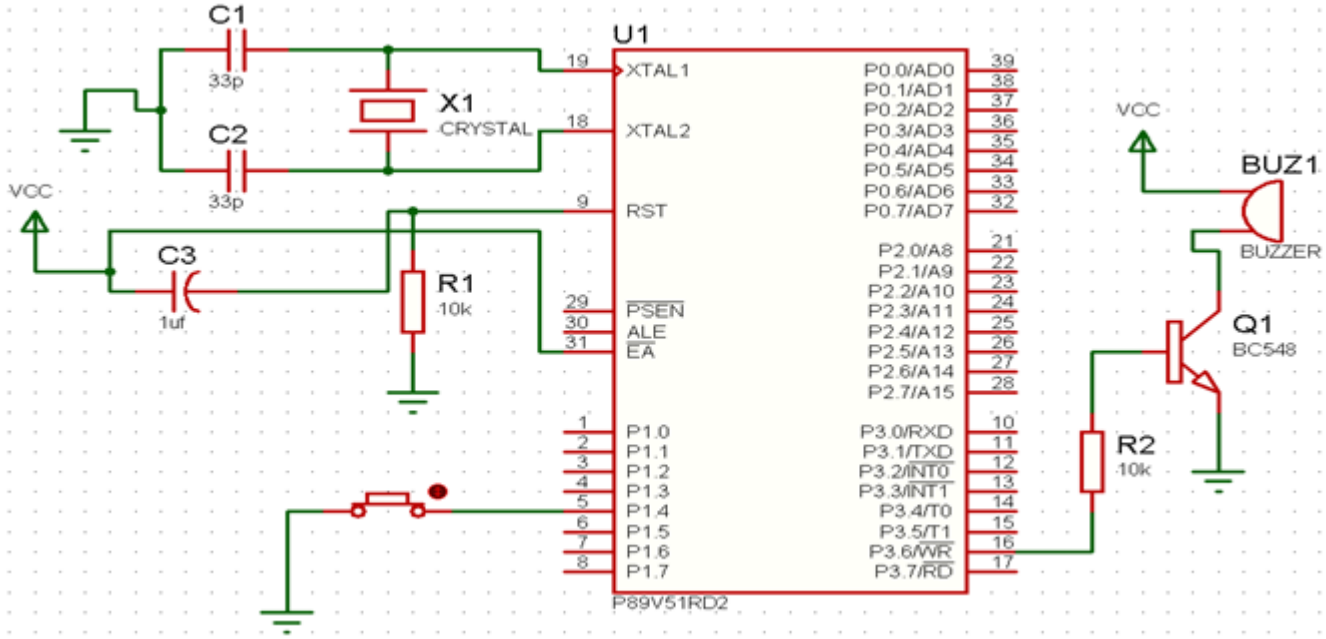


Figure 5.1: Interfacing Switch and Buzzer with 8051 Microcontroller

Source Code:
/* Program to toggle all the bits of Port P1 continuously with 250 mS delay. */

```
#include<REG51.H>
#define LEDPORT P1
void delay(unsigned int);
void main(void)
{
        LEDPORT =0x00;
        while(1)
        {
                LEDPORT = 0x00;
                delay(250);
```

```
                LEDPORT = 0xff;
                delay(250);
        }
}
void delay(unsigned int itime)
{
        unsigned  int i,j;
        for(i=0;i<itime;i++);
        {
                for(j=0;j<1275;j++);
        }
}
```

Flowchart:

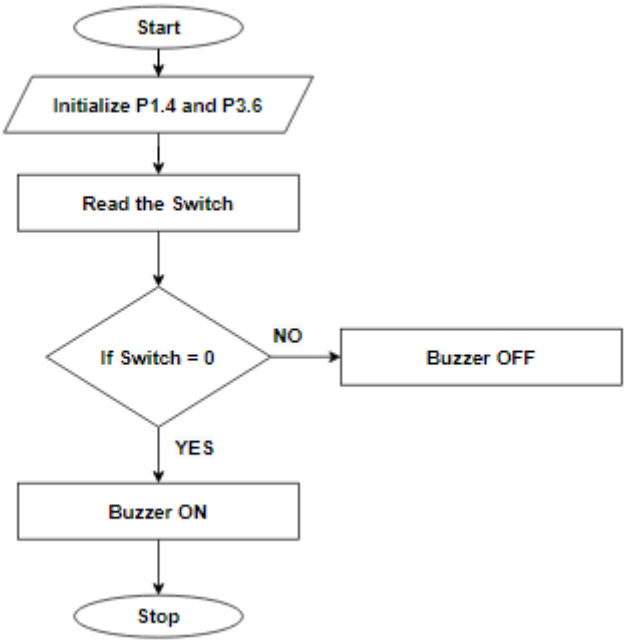


Figure 5.2: Flowchart

5.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

5.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Observe the results carefully.

5.8 Result

5.9 Further Probing Experiments

Q1. Program to toggle the odd number of bits of Port, P1 continuously with 250 ms delay.

Q2. Program to toggle the P0.5 bit with 500 ms delay.

Q3. Program to toggle the even number of bits of Port, P1 continuously with 100 ms delay.

LAB-5 INTERFACING LCD DISPLAY

6.1 Introduction

Liquid crystal display (LCD) displays use a combination of a backlight and tiny filters, controlled by an electrical current, to produce images. This technology lies behind displays that range from the personal tech gadgets such as watches to giant high-definition TVs. LCD screens stand on their own or work in tandem with the touch-screen technology found in smart phones and touch-capable computers. Interface the LCD display with Port 2 of 8051 microcontroller using embedded C language and display some message on LCD display.

6.2 Objective

6.2.1 Educational

• Learn the pin configuration of 8051 microcontroller.
• Learn the concept of interfacing.
• Learn the pin configuration of 16x2 LCD display.

6.2.2 Experimental

• Interface LCD display with 8051 in trainer board using jumpers.
• Understand the hardware connections of interfacing of buzzer and switch with 8051 microcontroller.
• Understand the usage of Keil IDE tool.

6.3 Prelab Preparation:

6.3.1 Reading

• Lean the Interfacing circuit given in the background section of this Laboratory.
• Learn the program to interface 16x2 LCD display with 8051 microcontroller.

6.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

6.4 Equipment needed

6.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.

3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

6.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

6.5 Background

Display units are the most important output devices in embedded projects and electronics products. 16x2 LCD is one of the most used display unit. 16x2 LCD means that there are two rows in which 16 characters can be displayed per line, and each character takes 5X7 matrix space on LCD. In this experiment we are going to connect 16X2 LCD module to the 8051 microcontroller (P89V51RD2). Interfacing LCD with 8051 microcontroller might look quite complex to newbie's, but after understanding the concept it would look very simple and easy. Although it may be time taking because you need to understand and connect 16 pins of LCD to the microcontroller. So first let's understand the 16 pins of LCD module.

We can divide it in five categories, Power Pins, contrast pin, Control Pins, Data pins and Backlight pins.

Category	**Pin NO.**	**Pin Name**	**Function**
Power Pins	1	VSS	Ground Pin, connected to Ground
	2	VDD or VCC	Voltage Pin +5V
Contrast Pin	3	V0 or VEE	Contrast Setting, connected to Vcc thorough a variable resistor.
Control Pins	4	RS	Register Select Pin, RS=0 Command mode, RS=1 Data mode
	5	RW	Read/ Write pin, RW=0 Write mode, RW=1 Read mode
	6	E	Enable, a high to low pulse need to enable the LCD
Data Pins	7-14	D0-D7	Data Pins, Stores the Data to be displayed on LCD or the command instructions
Backlight Pins	15	LED+ or A	To power the Backlight +5V
	16	LED- or K	Backlight Ground

Interfacing 16x2 LCD Display with 8051:

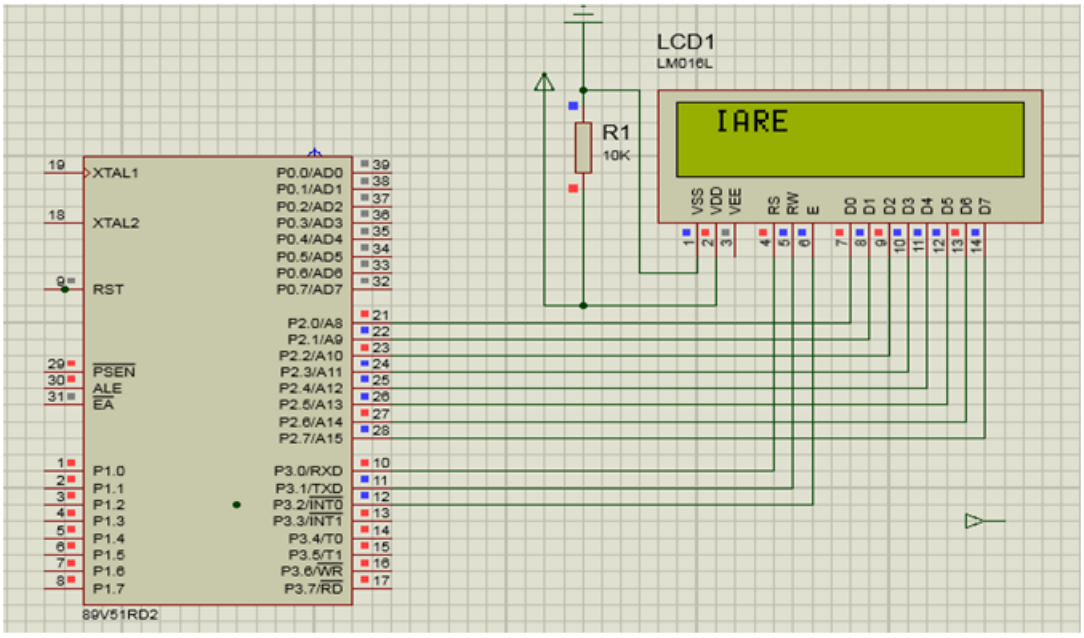


Figure 6.1: Interfacing 16x2 LCD Display with 8051 Microcontroller

Source Code:

/*Program to interface LCD data pins to port P2 and display a message on it.*/

```
#include<reg51.h>

sbit rs=P3^0;
sbit rw=P3^1;
sbit en=P3^2;

void lcdcmd(unsigned char);
void lcddat (unsigned char);
void delay();
void main()
{
        P2=0x00;
        while(1)
        {
                lcdcmd(0x38);
                delay();
                lcdcmd(0x01);
                delay();
                lcdcmd(0x10);
                delay();
                lcdcmd(0x0c);
                delay();
                lcdcmd(0x81);
```

```
                delay();
                lcddat('I');
                delay();
                lcddat('A');
                delay();
                lcddat('R');
                delay();
                lcddat('E');
                delay();
        }
}
void lcdcmd(unsigned char val)
{
        P2=val;
        rs=0;
        rw=0;
        en=1;
        delay();
        en=0;
}
void lcddat(unsigned char val)
{
        P2=val;
        rs=1;
        rw=0;
        en=1;
        delay();
        en=0;
}
void delay()
{
        unsigned int i;
        for(i=0;i<6000;i++);
}
```

Flowchart:

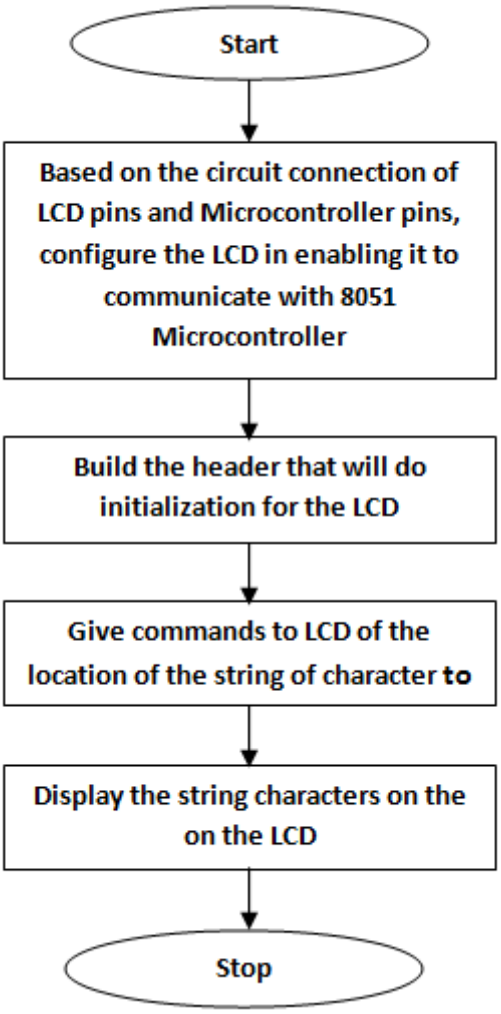


Figure 6.2: Flowchart

6.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

6.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between
4. LCD display and port pins of 8051 microcontroller. Observe the results carefully.

6.8 Result

6.9 Further Probing Experiments

Q1. Program to toggle the odd number of bits of Port, P1 continuously with 250 ms delay.

Q2. Program to toggle the P0.5 bit with 500 ms delay.

Q3. Program to toggle the even number of bits of Port, P1 continuously with 100 ms delay.

LAB-6 INTERFACING HEXA KEYPAD

7.1 Introduction

Write an embedded C program to interface 4x4 matrix keypad with 8051 microcontroller. Whenever a key is pressed, it should be displayed on LCD display using 8051 microcontroller.

7.2 Objective

7.2.1 Educational

• Learn the working of 4x4 matrix keypad.
Learn the switch debouncing concept.
• Learn the features of 8051 microcontroller.

7.2.2 Experimental

• Build the embedded C program to interface 4x4 matrix keypad with 8051 microcontroller.
• Understand the usage of Keil software for interfacing 4x4 matrix keypad with 8051 microcontroller.

7.3 Prelab Preparation:

7.3.1 Reading

• Learn the circuit diagram and program given in the background section of this Laboratory exercise.

7.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

7.4 Equipment needed

7.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

7.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

7.5 Background

Keypads are widely used input devices being used in various electronics and embedded projects. They are used to take inputs in the form of numbers and alphabets, and feed the same into system for further processing. In this tutorial we are going to interface a 4x4 matrix keypad with 8051 microcontroller.

4x4 Matrix Keypad
Before we interface the keypad with microcontroller, first we need to understand how it works. Matrix keypad consists of set of Push buttons, which are interconnected. Like in our case we are using 4X4 matrix keypad, in which there are 4 push buttons in each of four rows. And the terminals of the push buttons are connected according to diagram. In first row, one terminal of all the 4 push buttons are connected together and another terminal of 4 push buttons are representing each of 4 columns, same goes for each row. So we are getting 8 terminals to connect with a microcontroller.

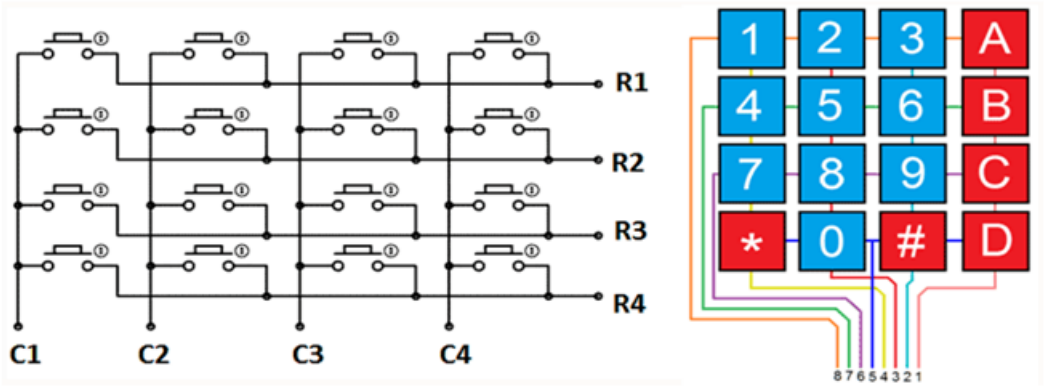


Figure 7.1: 4x4 Matrix Keypad

First we need to interface a LCD module to display the data which will be feed through KEYPAD, so please go through "LCD Interfacing with 8051 Microcontroller" article before interfacing KEYPAD.

As shown in above circuit diagram, to interface Keypad, we need to connect 8 terminals of the keypad to any port (8 pins) of the microcontroller. Like we have connected keypad terminals to Port 1 of 8051. Whenever any button is pressed we need to get the location of the button, means the corresponding ROW and COLUMN no. Once we get the location of the button, we can print the character accordingly.

Now the question is how to get the location of the pressed button? I am going to explain this in below steps and also want you to look at the code:

1. First we have made all the Rows to Logic level 0 and all the columns to Logic level 1.

2. Whenever we press a button, column and row corresponding to that button gets shorted and makes the corresponding column to logic level 0. Because that column becomes connected (shorted) to the row, which is at Logic level 0. So we get the column no. See main() function.

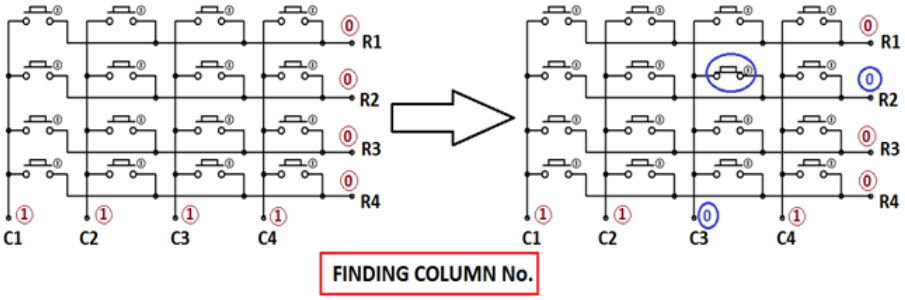


Figure 7.2: Finding Column No

3. Now we need to find the Row no., so we have created four functions corresponding to each column. Like if any button of column one is pressed, we call function row_finder1 (), to find the row no.

4. In row_finder1() function, we reversed the logic levels, means now all the Rows are 1 and columns are 0. Now Row of the pressed button should be 0 because it has become connected (shorted) to the column whose button is pressed, and all the columns are at 0 logic. So we have scanned all rows for 0.

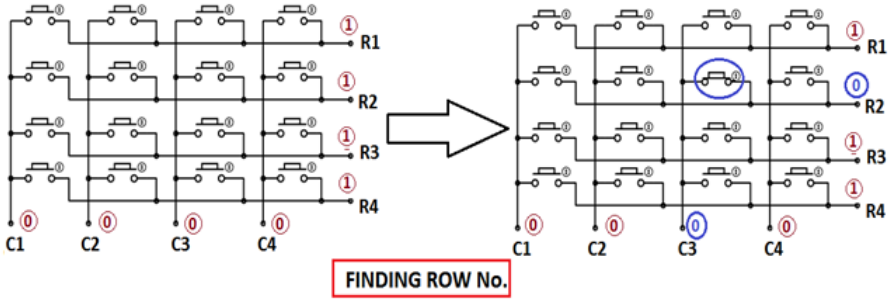


Figure 7.3: Finding Row No

5. So whenever we find the Row at logic 0, means that is the row of pressed button. So now we have column no (got in step 2) and row no., and we can print no. of that button using lcd_data function.
   Same procedure follows for every button press, and we are using while(1), to continuously check, whether button is pressed or not.

4*4 Matrix keyboard interfacing with 8051:

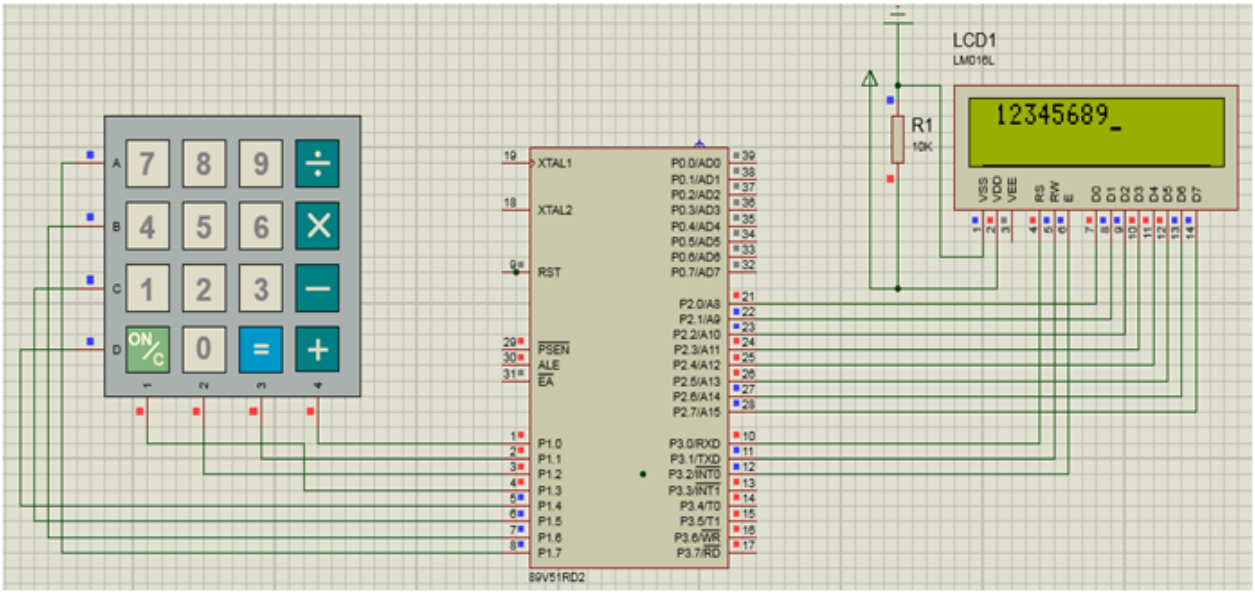


Figure 7.4: 4x4 Matrix keyboard interfacing with 8051 Microcontroller

Source Code:
/*Program to interface keypad. Whenever a key is pressed, it should be displayed on LCD.*/

```
#include<reg51.h>
#define display_port P2 //Data pins connected to port 2
on microcontroller

sbit rs = P3^0;  //RS pin connected to pin 2 of port 3
sbit rw = P3^1;  // RW pin connected to pin 3 of port 3
sbit e =  P3^2;  //E pin connected to pin 4 of port 3

sbit C4 = P1^0;      // Connecting keypad to Port 1
sbit C3 = P1^1;
sbit C2 = P1^2;
sbit C1 = P1^3;
sbit R4 = P1^4;
sbit R3 = P1^5;
sbit R2 = P1^6;
sbit R1 = P1^7;

void msdelay(unsigned int time)  // Function for creating delay
in milliseconds.
{
        unsigned i,j ;
        for(i=0;i<time;i++)
        for(j=0;j<1275;j++);
}
void lcd_cmd(unsigned char command)  //Function to send command
instruction to LCD
```

```
{
        display_port = command;
        rs= 0;
        rw=0;
        e=1;
        msdelay(1);
        e=0;
}

void lcd_data(unsigned char disp_data)  //Function to send
display data to LCD
{
        display_port = disp_data;
        rs= 1;
        rw=0;
        e=1;
        msdelay(1);
        e=0;
}

void lcd_init()    //Function to prepare the LCD  and get it ready
{
        lcd_cmd(0x38);  // for using 2 lines and 5X7 matrix of LCD
        msdelay(1);
        lcd_cmd(0x0F);  // turn display ON, cursor blinking
        msdelay(1);
        lcd_cmd(0x01);  //clear screen
        msdelay(1);
        lcd_cmd(0x81);  // bring cursor to position 1 of line 1
        msdelay(1);
}

void row_finder1() //Function for finding the row for column 1
{
        R1=R2=R3=R4=1;
        C1=C2=C3=C4=0;

        if(R1==0)
        lcd_data('7');
        if(R2==0)
        lcd_data('4');
        if(R3==0)
        lcd_data('1');
        if(R4==0)
        lcd_data('N');
}

void row_finder2() //Function for finding the row for column 2
{
        R1=R2=R3=R4=1;
        C1=C2=C3=C4=0;
```

```
        if(R1==0)
        lcd_data('8');
        if(R2==0)
        lcd_data('5');
        if(R3==0)
        lcd_data('2');
        if(R4==0)
        lcd_data('0');
}

void row_finder3() //Function for finding the row for column 3
{
        R1=R2=R3=R4=1;
        C1=C2=C3=C4=0;

        if(R1==0)
        lcd_data('9');
        if(R2==0)
        lcd_data('6');
        if(R3==0)
        lcd_data('3');
        if(R4==0)
        lcd_data('=');
}

void row_finder4() //Function for finding the row for column 4
{
        R1=R2=R3=R4=1;
        C1=C2=C3=C4=0;

        if(R1==0)
        lcd_data('%');
        if(R2==0)
        lcd_data('*');
        if(R3==0)
        lcd_data('-');
        if(R4==0)
        lcd_data('+');
}

void main()
{
        lcd_init();
        while(1)
        {
                msdelay(30);
                C1=C2=C3=C4=1;
                R1=R2=R3=R4=0;
                if(C1==0)
                row_finder1();
```

```
                else if(C2==0)
                row_finder2();
                else if(C3==0)
                row_finder3();
                else if(C4==0)
                row_finder4();
            }

        }
```

Flowchart:

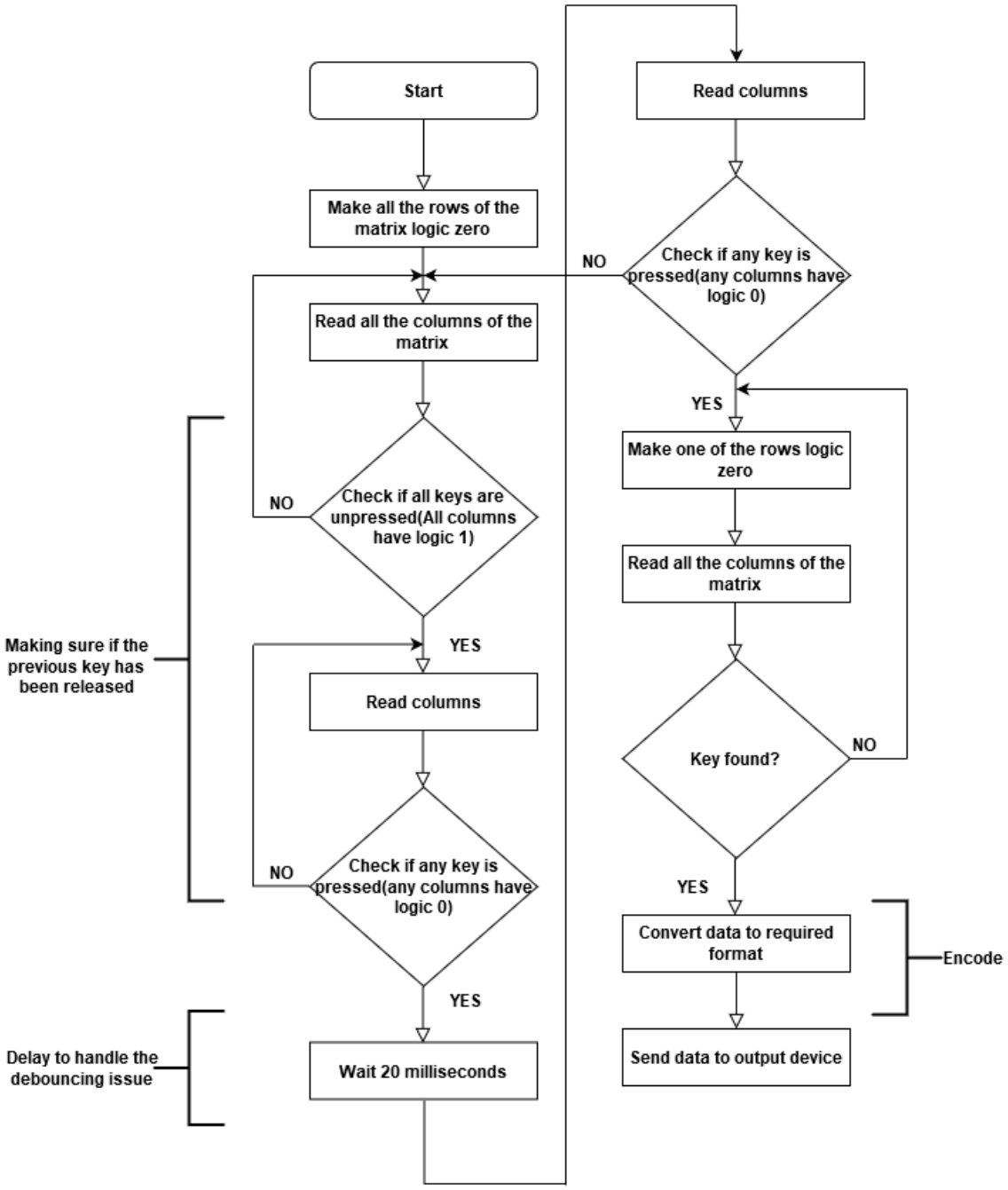


Figure 7.5: Flowchart

7.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

7.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the results carefully.

7.8 Result

7.9 Further Probing Experiments

Q1. Program to toggle the odd number of bits of Port, P1 continuously with 250 ms delay.

Q2. Program to toggle the P0.5 bit with 500 ms delay.

Q3. Program to toggle the even number of bits of Port, P1 continuously with 100 ms delay.

LAB-7 INTERFACING SEVEN SEGMENT DISPLAY

8.1 Introduction

This laboratory studies the use of seven segment displays for various applications. Seven segment displays are important display units in Electronics and widely used to display numbers from 0 to 9. It can also display some character alphabets like A,B,C,H,F,E etc. In this experiment, we are going to learn how to interface a 7 segment display with 8051 microcontroller. We are using P89V51RD2 microcontroller from 8051 series.

8.2 Objective

8.2.1 Educational

• Learn the different types of seven segment display types.
• Learn how the seven segment display, displays numbers as well as character alphabets.
• Learn the hexadecimal code for displaying numbers as well as character alphabets on seven segment display

8.2.2 Experimental

• Understand how to interface seven segment displays with 8051 in trainer board using jumpers.
• Understand the hardware connections of interfacing of seven segment displays with 8051 microcontroller.
• Understand the usage of Keil IDE tool.

8.3 Prelab Preparation:

8.3.1 Reading

• Understand the circuit diagram given in the background section of this Laboratory exercise.
• Learn the program given in the background section of this Laboratory exercise.

8.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

8.4 Equipment needed

8.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.

3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

8.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

8.5 Background

Before interfacing, we should learn about 7 segment display. It's the simplest unit to display numbers and characters. It just consists 8 LEDs, each LED used to illuminate one segment of unit and the 8th LED used to illuminate DOT in 7 segment display. We can refer each segment as a LINE, as we can see there are 7 lines in the unit, which are used to display a number/character. We can refer each line/segment "a,b,c,d,e,f,g" and for dot character we will use "h". There are 10 pins, in which 8 pins are used to refer a,b,c,d,e,f,g and h/dp, the two middle pins are common anode/cathode of all he LEDs. These common anode/cathode are internally shorted so we need to connect only one COM pin.

There are two types of 7 segment displays: Common Anode and Common Cathode.

Common Anode: In this all the Negative terminals (cathode) of all the 8 LEDs are connected together (see diagram below), named as COM. And all the positive terminals are left alone.

Common Cathode: In this all the positive terminals (Anodes) of all the 8 LEDs are connected together, named as COM. And all the negative thermals are left alone. As shown below the circuit diagram for interfacing 7 segment display with 8051 microcontroller, we have connected a,b,c,d,e,f,g,h to pins 0.0 to 0.7 means we are connecting 7 segment to port 0 of microcontroller. Now suppose we want to display 0, then we need to glow all the LEDs except LED which belongs to line "g", so pins 0.0 to 0.6 should be at 1 (should be 1 to TURN ON the LED as per positive logic) and pin 0.7 and 0.8 should be at 0 (should be 0 to TURN OFF the LED as per positive logic). So the LEDs connected to pins 0.0 to 0.6 (a,b,c,d,e,f) will be ON and LEDs connected to 0.7 and 0.8 (g and h) will be OFF, that will create a "0" in 7 segment. So we need bit pattern 00111111 (Pin 8 is the highest bit so starting from P0.7 to P0.0), and the HEX code for binary 00111111 is "3F". Similarly we can calculate for all the digits. Here we should note that we are keeping "dot/h" always OFF, so we need to give LOGIC "0" to it every time. A table has been given below for all the numbers while using Common cathode 7 segment display.

Interfacing Seven Segment Display with 8051:

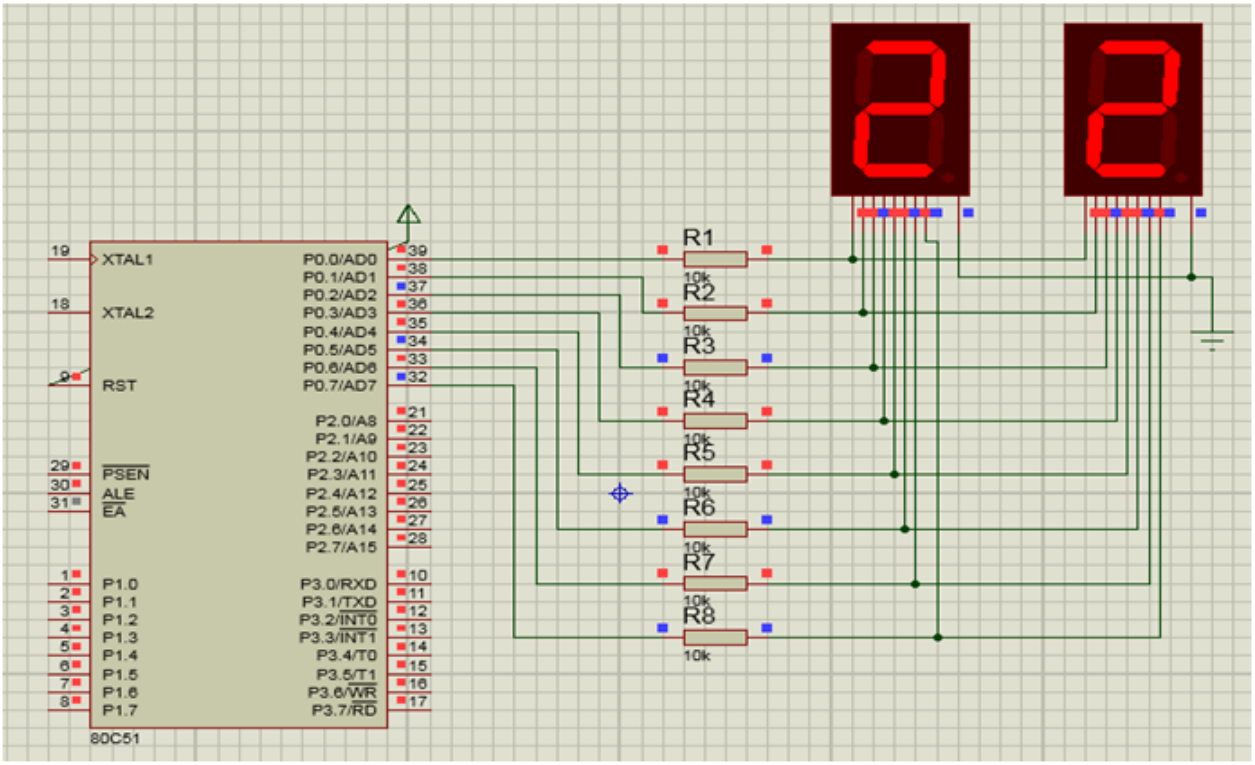


Figure 8.1: Interfacing Seven Segment Display with 8051 Microcontroller

Source Code:
/*Program to interface seven segment display unit.*/

```
#include <REG51.H>

#define LEDPORT P0
#define ZERO 0x3f
#define ONE 0x06
#define TWO 0x5b
#define THREE 0x4f
#define FOUR 0x66
#define FIVE 0x6d
#define SIX 0x7d
#define SEVEN 0x07
#define EIGHT 0x7f
#define NINE 0x6f
#define TEN 0x77
#define ELEVEN 0x7c
#define TWELVE 0x39
#define THIRTEEN 0x5e
#define FOURTEEN 0x79
#define FIFTEEN 0x71
void Delay(void);
void main (void)
{
```

```
        while(1)
        {
                LEDPORT = ZERO;
                Delay();
                LEDPORT = ONE;
                Delay();
                LEDPORT = TWO;
                Delay();
                LEDPORT = THREE;
                Delay();
                LEDPORT = FOUR;
                Delay();
                LEDPORT = FIVE;
                Delay();
                LEDPORT = SIX;
                Delay();
                LEDPORT = SEVEN;
                Delay();
                LEDPORT = EIGHT;
                Delay();
                LEDPORT = NINE;
                Delay();
                LEDPORT = TEN;
                Delay();
                LEDPORT = ELEVEN;
                Delay();
                LEDPORT = TWELVE;
                Delay();
                LEDPORT = THIRTEEN;
                Delay();
                LEDPORT = FOURTEEN;
                Delay();
                LEDPORT = FIFTEEN;
                Delay();
        }
}
void Delay(void)
{
        int j; int i;
        for(i=0;i<30;i++)
        {
                for(j=0;j<10000;j++)
                {
                }
        }
}
```

Flowchart:

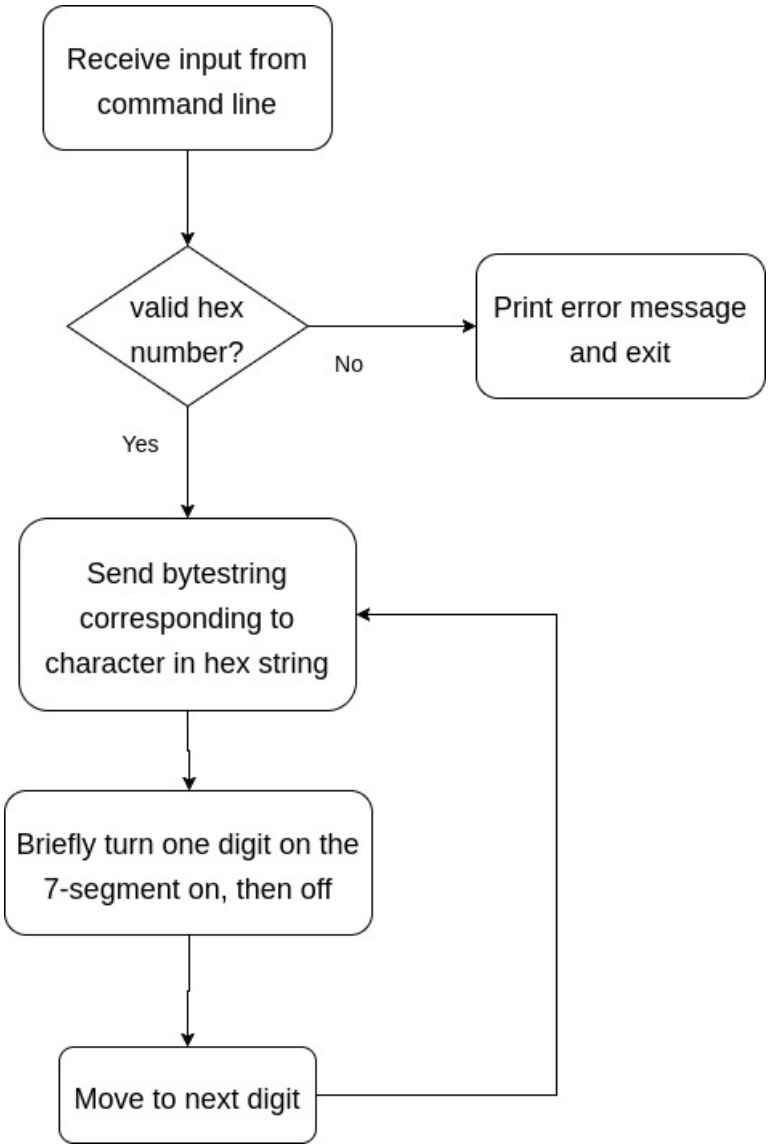


Figure 8.2: Flowchart

8.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

8.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the results carefully.

8.8 Result

8.9 Further Probing Experiments

Q1. What are the different types of seven segment displays?

Q2. Compute the codes to display odd numbers from 0 to 9 numbers on common cathode seven segment display.

Q3. Compute the codes to display even numbers from 0 to 9 numbers on common cathode seven segment display.

Q4. List out the features of 8051 microcontroller.

LAB-8 INTERFACING STEPPER MOTOR

9.1 Introduction

Stepper motors are used to translate electrical pulses into mechanical movements. In some disk drives, dot matrix printers, and some other different places the stepper motors are used. The main advantage of using the stepper motor is the position control. Stepper motors generally have a permanent magnet shaft (rotor), and it is surrounded by a stator.

9.2 Objective

9.2.1 Educational

• To learn the working principle of stepper motor.
• To learn the different operating modes of unipolar stepper motor.

9.2.2 Experimental

• To learn the rotation of stepper motor in clockwise direction and anti clockwise direction.
• Understand how to interface unipolar stepper motor with P89V51RD2 using ULN2003A drivers.

9.3 Prelab Preparation:

9.3.1 Reading

• Learn the working of stepper motor and there operating modes.
• Study the Background section given below.

9.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

9.4 Equipment needed

9.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

9.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

9.5 Background

Stepper motors are used to translate electrical pulses into mechanical movements. In some disk drives, dot matrix printers, and some other different places the stepper motors are used. The main advantage of using the stepper motor is the position control. Stepper motors generally have a permanent magnet shaft (rotor), and it is surrounded by a stator.

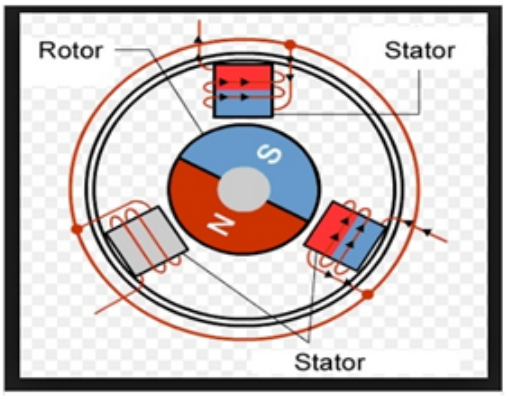


Figure 9.1: Stepper Motor

Normal motor shafts can move freely but the stepper motor shafts move in fixed repeatable increments.
Some parameters of stepper motors -

Step Angle: The step angle is the angle in which the rotor moves when one pulse is applied as an input of the stator. This parameter is used to determine the positioning of a stepper motor.

Steps per Revolution: This is the number of step angles required for a complete revolution. So the formula is 360° /Step Angle.
Steps per Second: This parameter is used to measure a number of steps covered in each second.

RPM: The RPM is the Revolution Per Minute. It measures the frequency of rotation. By this parameter, we can measure the number of rotations in one minute.
The relation between RPM, steps per revolution, and steps per second is like below:

Steps per Second = rpm x steps per revolution / 60

We are using Port P2 of 8051 for connecting the stepper motor. Here ULN2003A is used. This is basically a high voltage, high current Darlington transistor array. Each ULN2003A has seven NPN Darlington pairs. It can provide high voltage output with common cathode clamp diodes for switching inductive loads. The Unipolar stepper motor works in three modes.

Wave Drive Mode: In this mode, one coil is energized at a time. So all four coils are energized one after another. This mode produces less torque than full step drive mode.
The following table is showing the sequence of input states in different windings.

Steps	Winding A	Winding B	Winding C	Winding D
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Full Drive Mode: In this mode, two coils are energized at the same time. This mode produces more torque. Here the power consumption is also high.
The following table is showing the sequence of input states in different windings.

Steps	Winding A	Winding B	Winding C	Winding D
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

Half Drive Mode: In this mode, one and two coils are energized alternately. At first, one coil is energized then two coils are energized. This is basically a combination of wave and full drive mode. It increases the angular rotation of the motor.
The following table is showing the sequence of input states in different windings.

Steps	Winding A	Winding B	Winding C	Winding D
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

Interfacing Stepper Motor with 8051 Microcontroller:

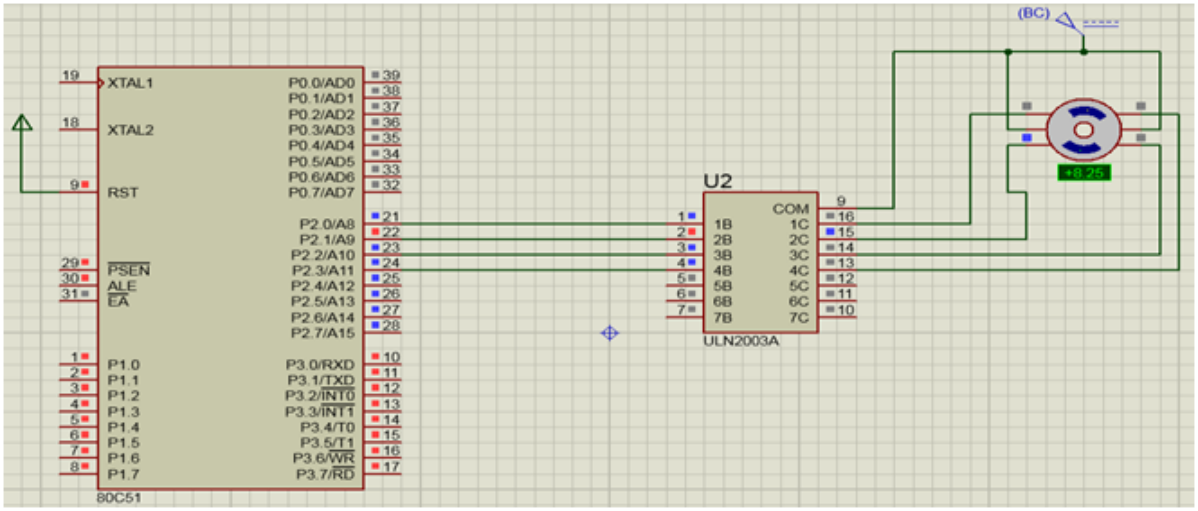


Figure 9.2: Interfacing Stepper Motor with 8051 Microcontroller

Source Code:
/*Program to interface Stepper Motor with 8051 and to rotate the motor in clockwise directions*/

```
#include<reg51.h>

void msdelay(unsigned int time)
{
        unsigned i,j ;
        for(i=0;i<time;i++)
        for(j=0;j<1275;j++);
}

void main()
{
        while(1)
        {
                P2 = 0x08;              //1000
                msdelay(1);
                P2 = 0x04;              //0100
                msdelay(1);
                P2 = 0x02;              //0010
                msdelay(1);
                P2 = 0x01;              //0001
                msdelay(1);
        }
}
```

/*Program to interface Stepper Motor with 8051 and to rotate the motor in anticlockwise directions*/

```
#include<reg51.h>

void msdelay(unsigned int time)
{
        unsigned i,j ;
        for(i=0;i<time;i++)
        for(j=0;j<1275;j++);
}

void main()
{
        while(1)
        {
                P2 = 0x01;          //0001
                msdelay(1);
                P2 = 0x02;          //0010
                msdelay(1);
                P2 = 0x04;          //0100
                msdelay(1);
                P2 = 0x08;          //1000
                msdelay(1);
        }
}
```

9.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose “ADD s to Group” option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select “Options for Target”.

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

9.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the rotation of stepper motor in both clockwise and anti clockwise directions.

9.8 Result

9.9 Further Probing Experiments

Q1. What is stepper motor?

Q2. Give the reason behind the motor name as stepper motor or step motor.

Q3. On which factors the Step angle of the stepper motor depends?

Q4. Explain the term: Full step operation, Half step operation.

Q5. State the different types of stepper motor.

LAB-9 INTERFACING ANALOG TO DIGITAL CONVERTER (ADC)

10.1 Introduction

The data we process in a microcontroller normally deals with digital signals. But there may a situation where we have to deal with external inputs such as analog signals. All most all the input signals from physical sensors are of analog signals. In such cases, we can interface the microcontroller with an external device such as an ADC0808 to convert the analog signal to a digital signal. Because our microcontrollers can only understand 0 and 1. In this experiment, we look into the details of ADC interfacing with 8051. In the present time, there are lots of microcontrollers in the market which has inbuilt ADC with one or more channels, E.g.: PIC18F4550, LPC1768, etc. And by using their ADC registers we can interface. Unfortunately, 8051 doesn't have an internal module so we will go for an external ADC which is ADC0808.

10.2 Objective

10.2.1 Educational

• The objective of this laboratory exercise is to convert the given analog signal to the digital signal using the ADC0808 converter.
• Learn the pin configuration of ADC0808 converter.

10.2.2 Experimental

• Determine the digital signal value of corresponding analog signal given as input.

10.3 Prelab Preparation:

10.3.1 Reading

• Study the Background section below.
• Read the chapter in your textbook on data converters. Pay particular attention to the way in which how the given analog signal to digital signal.

10.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

10.4 Equipment needed

10.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

10.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

10.5 Background

ADC0808 is a commonly used External 8 bit ADC and it has 28 pins. It can measure up to eight ADC values from 0 to 5 volt since it has eight channels. when voltage reference is +5V, its Step size will be 19.53mV. That is, for every increase of 19.53mV on the input side there will be an increase of 1 bit at the output side. ADC0808 needs an external clock to operate. The ADC needs some specific control signals for its operations like start conversion and bring data to output pins. When the conversion is complete the EOC pins go low to indicate the end of a conversion and that the data is ready to be picked up.

Features:

- Easy interface to all microprocessors.
- Operates ratio metrically or with 5 V DC or analog span adjusted voltage reference.
- No zero or full-scale adjust required.
- 8-channel multiplexer with address logic.
- 0V to 5V input range with single 5V power supply.
- Outputs meet TTL voltage level specifications.
- Standard hermetic or molded 28-pin DIP package.
- 28-pin molded chip carrier package.

Analog to Digital Converter (ADC) Interfacing with 8051 Microcontroller:

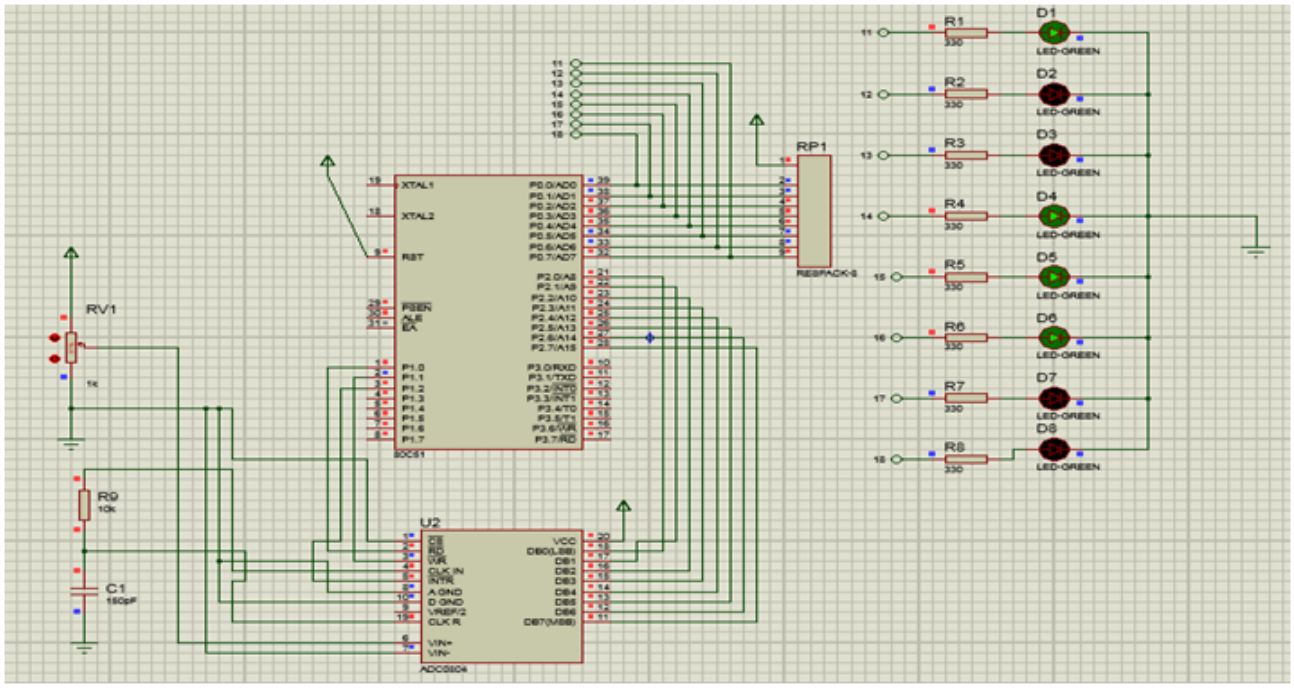


Figure 10.1: Analog to Digital Converter (ADC) Interfacing with 8051 Microcontroller

Source Code:

```
#include <reg51.h>
#define input P2  // Input port to read the values of ADC
#define output P0  // Output port, connected to LED's.

sbit wr= P1^1;  // Write pin. It is used to start the conversion.
sbit rd= P1^0;  // Read pin. It is used to extract the data
from internal register to the output pins of ADC.
sbit intr= P1^2;        // Interrupt pin. This is used to indicate
 the end of conversion. It goes low when conversion is complete.

void delay(unsigned int msec ) // The delay function provides
delay in msec.
{
        int i,j ;
        for(i=0;i<msec;i++)
        for(j=0;j<1275; j++);
}

void adc()  // Function to read the values from ADC and
display on the LED's.
{
        rd=1;
        wr=0;
        delay(1);
        wr=1;
```

```
        while(intr==1);
        rd=0;
        output=input;
        delay(1);
        intr=1;
}
void main()
{
        input=0xff;  // Declare port 0 as input port.
        while(1)
        {
                adc();
        }
}
```

10.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose “ADD s to Group” option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select “Options for Target”.

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

10.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the digital output of given analog input using LED's.

10.8 Result

10.9 Further Probing Experiments

Q1. What is analog to digital converter (ADC)?

Q2. List out the features of analog to digital converter, ADC0808?

Q3. List out the features of 8051 microcontroller?

Q4. What is the necessity of analog to digital converter (ADC) in real life?

Q5. Explain the necessity of interfacing?

LAB-10 INTERFACING DIGITAL TO ANALOG CONVERTER (DAC)

11.1 Introduction

To interface digital to analog converter (DAC0808) with P89V51RD2 microcontroller and display the analog output on cathode ray oscilloscope (CRO). In this experiment we will see how DAC (Digital to Analog Converter) interfacing with Intel 8051 family microcontroller. We will also see the square wave generation using DAC.

The Digital to Analog converter (DAC) is a device that is widely used for converting digital pulses to analog signals. There are two methods of converting digital signals to analog signals. These two methods are binary weighted method and R/2R ladder method. In this experiment we will use the MC1408 (DAC0808) Digital to Analog Converter. This chip uses R/2R ladder method. This method can achieve a much higher degree of precision. DACs are judged by its resolution. The resolution is a function of the number of binary inputs. The most common input counts are 8, 10, 12 etc. Number of data inputs decides the resolution of DAC. So if there are n digital input pin, there are 2n analog levels. So 8 input DAC has 256 discrete voltage levels.

11.2 Objective

11.2.1 Educational

• The objective of this laboratory exercise is to convert the given digital pulses to the analog signal using the DAC0808 converter.

11.2.2 Experimental

• To determine the analog signal value of corresponding digital signals.

11.3 Prelab Preparation:

11.3.1 Reading

• Study the Background section below.
• Read the chapter in your textbook on data converters. Pay particular attention to the way in which how the given digital signal to analog signal.

11.3.2 Written

• After coming to lab, student should write the experiment on the worksheets provided by the teaching faculty on his/her own.

11.4 Equipment needed

11.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

11.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

11.5 Background

Generate a square wave using 8051 in Proteus. In order to generate a square wave using 8051 or in other words the P89V51RD2, we will be using proteus. It's a software used to simulate the project. We will also need a DAC, a Digital to Analog Converter in short, and the P89V51RD2 microcontroller itself.

The DAC we will be using is a simple DAC0808. It's a 16 pin DIP package and easy to use. The 8051 has two timers which can be configured in 3 modes. We will not be using the timers for the delay but instead the software delays to produce an approximate delay.

Steps to generate a square wave using 8051:

1. Set the output of any port on the 8051 to logic high.
2. Wait for some time.
3. Set the output of the same port to logic low.
4. Again wait for the same amount of time as you did earlier.
5. Loop around the same.

Subsequently, for obtaining the desired frequency on the square wave. We have to manipulate with the delay. We know that the machine cycle frequency is 1/12 of the crystal oscillator frequency. So, with the crystal oscillator's frequency as 11.0592 MHz the machine cycle frequency is 921.6 KHz. To sum up, that's equivalent to 1.085 second.

To generate a square wave of 1 KHz, in other words, a wave of time period 1 millisecond, we have to use the delay in such a way that it causes a delay of 1 millisecond. We will have to loop around doing nothing for about 461 machine cycles to generate a square wave of 50% duty cycle. That is to say, an on time and an off time of 0.5 millisecond.

Features:

- 8 bit parallel digital data input
- Fast settling time (typical value): 150 ns
- Relative accuracy at ±0.19% maximum error
- Full scale current match: ±1 LSB
- Non-inverting digital inputs are TTL and CMOS compatible
- High speed multiplying input slew rate: 8 mA/μs
- Power supply voltage range: ±4.5V to ±18V
- Low power consumption: 33 mW@ ±5V
- Maximum Power dissipation: 1000 mW
- Operating temperature range: 0^{o}C to $+75^{o}$C

DAC0808 is a D/A converter IC and is used for converting 8 bit digital data input to analog signal output. It is a monolithic IC featuring a full scale output current settling time of 150 ns while dissipating only 33 mW with ±5V supplies. The chip accuracy of conversion is good and power consumption is also low to make it popular. The power supply currents of the DAC0808 are independent of bit codes, and exhibits essentially constant device characteristics over the entire supply voltage range.

Pin configuration

DAC0808 is a sixteen pin device and description for each pin is given below.

Table 11.1: Pin Configuration of DAC0808

Pin	Name	Description
1	NC	No connection
2	GND	Ground
3	VEE	Negative power supply
4	IO	Output signal pin
5	A1	Digital input bit 1 (Most Significant Bit)
6	A2	Digital input bit 2
7	A3	Digital input bit 3
8	A4	Digital input bit 4
9	A5	Digital input bit 5
10	A6	Digital input bit 6
11	A7	Digital input bit 7
12	A8	Digital input bit 8 (Least Significant Bit)
13	VCC	Positive power supply
14	VREF+	Positive reference voltage
15	VREF-	Negative reference voltage
16	COMPENSATION	Compensation capacitor pin

Digital to Analog Converter (DAC) Interfacing with 8051:

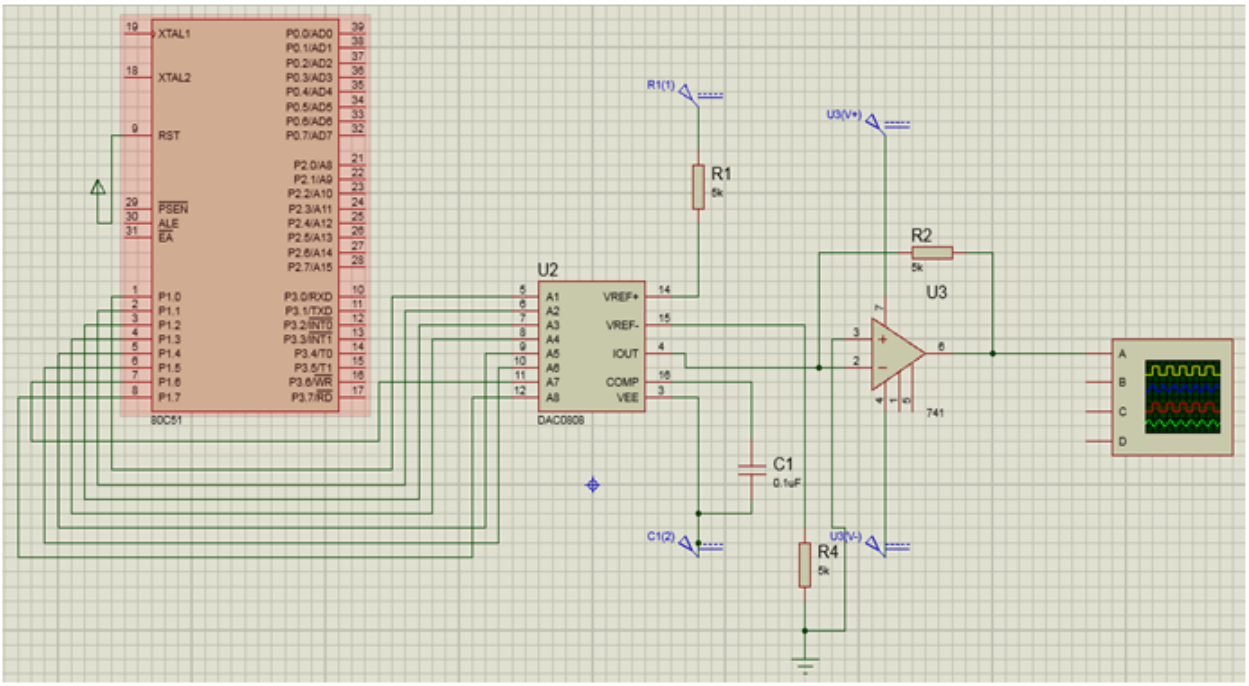


Figure 11.1: Digital to Analog Converter (DAC) Interfacing with 8051 Microcontroller

Source Code:

```
#include<reg51.h>

void delay( void );
void main()
{
        for(;;)
        {
                P1=0x00;
                delay();
                P1=0xFF;
                delay();
        }
}

void delay(void)
{
        unsigned int i,j;
        for(i=0;i<250;i++)
        {
                for(j=0;j<1275;j++);
        }
}
```

11.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

11.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the analog output of given digital input using logic analyzer/Cathode Ray Oscilloscope (CRO).

11.8 Result

11.9 Further Probing Experiments

Q1. What is digital to analog converter (DAC)?

Q2. List out the features of digital to analog converter, DAC0808?

Q3. List out the features of 8051 microcontroller?

Q4. What is the necessity of digital to analog converter (DAC) in real life?

Q5. Explain the necessity of interfacing digital to analog converter (DAC) with 8051 microcontroller?

LAB-11 INTERFACE RELAY

12.1 Introduction

In some electronic applications we need to switch or control high voltages or high currents. In these cases we may use electromagnetic or solid state relays. For example, it can be used to control home appliances using low power electronic circuits.

An electromagnetic relay is a switch which is used to switch High Voltage or Current using Low power circuits. It magnetically isolates low power circuits from high power circuits. It is activated by energizing a electromagnet, coil wounded on a soft iron core. For detailed working of relay please visit this page. A relay should not be directly connected to a microcontroller, it needs a driving circuit due to the following reasons. A microcontroller will not able to supply current required for the proper working of a relay. The maximum current that A89C51 microcontroller can sink is 15mA while a relay needs about 50 – 100mA current. A relay is activated by energizing its coil. Microcontroller may stop working by the negative voltages produced in the relay due to its back emf.

12.2 Objective

12.2.1 Educational

• To understand the working of relay in some electronic applications.

12.2.2 Experimental

• The objective of this laboratory exercise is to interface relay with P89V51RD2 microcontroller.

12.3 Prelab Preparation:

12.3.1 Reading

• Study the Background section below.
• Read the chapter in your textbook on relays.

12.3.2 Written

• Review the steps in the procedure below and plan how you will interface relay with P89V51RD2 microcontroller.

12.4 Equipment needed

12.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

12.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

12.5 Background

Relay is one of the most interesting and important components. Relays are switches that open and close circuits when actuated when an electrical signal. Because relays are the link between the low power digital electronics and high power devices. It allows digital circuits and digital microcontrollers to high power devices on and off. Simply, it is used to on/off power circuits using microcontrollers. Relays operate with AC or DC at common voltages like 12V, 24V, 48V, 120V and 230V. Relays can control currents ranging from 2A – 30A. When the switch is closed, the circuit is closed – or on. When the switch is opened, the circuit is open – or off. When a switch can be actuated with an electrical signal, the device is then typically referred to as a relay. The actuation of the relay will change the state of the contacts from open to closed or vice verses, depending on the contact configuration. We will look into the details of relay working, types and relay Interfacing with 8051 in this article.

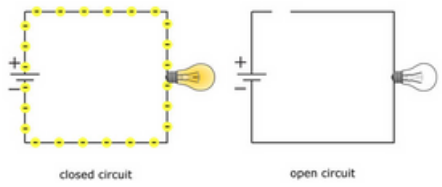


Figure 12.1: Closed circuit and Open circuit

Uses of Relays: The major use of relay started during the invention of telephones. Relay act as an important role in switching calls in a telephone exchange. They were used to switch the signal coming from one source to another destination. After the invention of computers, they were also used to perform Boolean and other logical operations. Relays are also used in our day to day devices like Refrigerators, Washing Machines, Heaters, and Air Conditioning.

Relays are mainly used for two basic operations. One is a low voltage application and the other is high voltage. For low voltage applications, more preference will be given to reduce the noise of the whole circuit. For high voltage applications, they are mainly designed to reduce a phe-

nomenon called arcing.

Working of Relays A simple electromagnetic relay consists of a coil of wire wrapped around a soft iron core, an iron yoke which provides a low reluctance path for magnetic flux, a movable iron armature, and one or more sets of contacts. The armature is hinged to the yoke and mechanically linked to one or more sets of moving contacts. It is held in place by a spring so that when the relay is de-energized there is an air gap in the magnetic circuit. In this condition, one of the two sets of contacts in the relay pictured is closed, and the other set is open. Other relays may have more or fewer sets of contacts depending on their function.

When an electric current is passed through the coil it generates a magnetic field that activates the armature, and the consequent movement of the movable contact either makes or breaks a connection with a fixed contact. If the set of contact was closed when the relay was de-energized, then the movement opens and breaks the connection, and vice versa if the contact were open. When the current to the coil is switched off, the armature is returned by a force, approximately half as strong as the magnetic force, to its relaxed position. Usually, this force is provided by a spring, but gravity is also used as commonly in industrial motor starters.

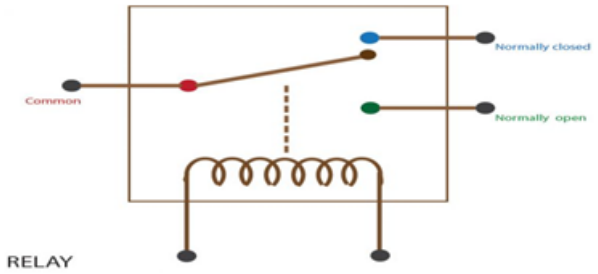


Figure 12.2: Relay

When the coil is energized with direct current, a diode is often placed across the coil to dissipate the energy from the collapsing magnetic field at deactivation, which would otherwise generate a voltage spike dangerous to semiconductor circuit components. An electromechanical relay consists of three terminals namely common (COM), normally closed (NC) and normally opened (NO) contacts. These can either get opened or closed when the relay is in operation.

Application of Relay:

- The major application of Relay is used for isolating a low voltage circuit from the high voltage circuit.
- Microprocessors use relays to control a heavy electrical load.
- Electromagnetic relays are employed for the protection of various ac and dc equipment.
- They are used for controlling multiple circuits.
- Used as auxiliary relays in the contact systems of protective relay schemes.
- They are also used as automatic change over.

Interfacing Relay with 8051 Microcontroller:

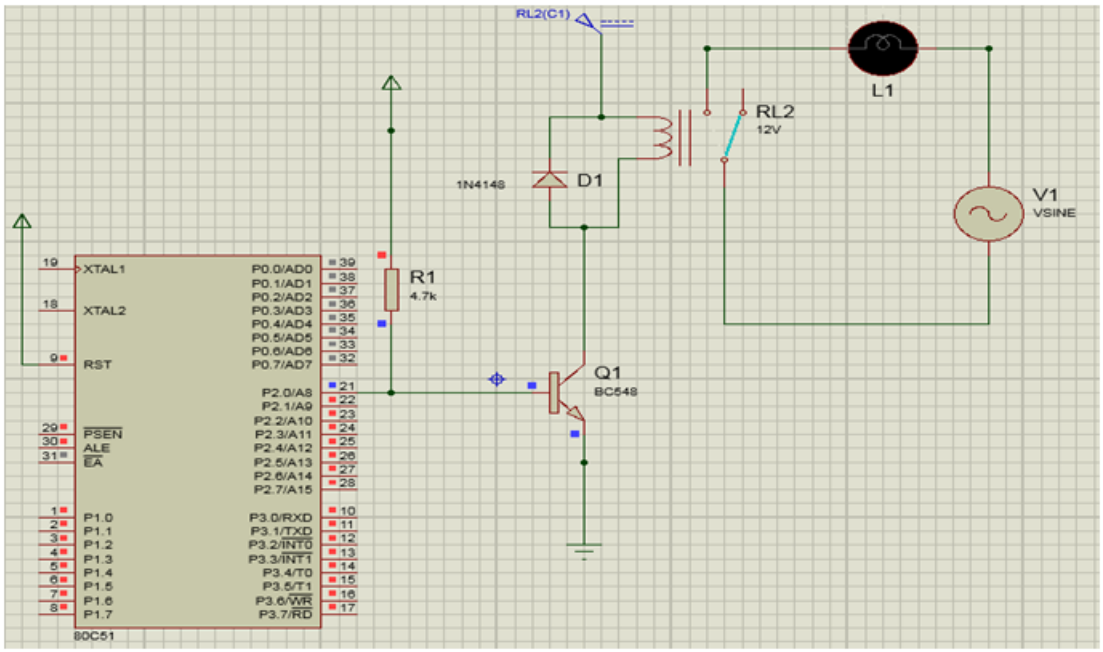


Figure 12.3: Interfacing Relay with 8051 Microcontroller

Source Code:

```
#include<reg51.h>
sbit relay_pin = P2^0;
void Delay_ms(int);
void main()
{
        do
        {
                relay_pin = 0; //Relay ON
                Delay_ms(1000);
                relay_pin = 1; //Relay OFF
                Delay_ms(1000);
        }
        while(1);
}

void Delay_ms(int k)
{
        int j;
        int i;
        for(i=0;i<k;i++)
        {
                for(j=0;j<100;j++)
                {
                }
        }
```

}

12.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose "ADD s to Group" option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select "Options for Target".

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

12.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the working of relay using bulb.

12.8 Result

12.9 Further Probing Experiments

Q1. What is Relay?

Q2. Explain the working of Relay?

Q3. List out the different types of relays?

Q4. List out the different applications of relays?

Q5. Explain the interfacing of relay with 8051 microcontroller?

LAB-12 SERIAL COMMUNICATION INTERFACING FROM MICROCONTROLLER TO PC

13.1 Introduction

Microcontrollers need to communicate with external devices such as sensors, computers and so on to collect data for processing. Data communication is generally done by means of two methods – Parallel and Serial mode. In parallel mode data bits are transferred faster using more data pins. But when comes to a Microcontroller, we cannot afford to dedicate many pins for data transfer. UART or Serial communication in 8051 microcontroller will allow the controller to send and receive data's just by using two pins Serial Communication uses only two data pins to establish communication between Microcontroller and external devices. In this mode of communication data is transferred one bit at a time. This article describes Interfacing of 8051 with PC to establish communication through its serial port RS232.

13.2 Objective

13.2.1 Educational

• To understand the serial communication and features of 8051 microcontroller..

13.2.2 Experimental

• The objective of this laboratory exercise is to understand how the data is transferred using serial communication from microcontroller to PC.

13.3 Prelab Preparation:

13.3.1 Reading

• Study the Background section below.
• Read the chapter in your textbook on serial communication concepts.

13.3.2 Written

• Review the steps in the procedure below and plan how you will transfer data using serial communication from microcontroller to PC.

13.4 Equipment needed

13.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

13.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

13.5 Background

To establish communication between a controller and PC, we must use serial I/O protocol RS-232 which was widely used in PC and several devices. PC works on RS-232 standards which operates at a logic level of -25V to +25V. But Microcontrollers use TTL logic which works on 0-5V is not compatible with the RS-232 voltage levels. MAX232 is a specialized IC which offers intermediate link between the Microcontroller and PC. The transmitter of this IC will convert the TTL input level to RS-232 Voltage standards. Meanwhile the receiver of this IC will convert RS-232 input to 5V TTL logic levels. Read the complete working of MAX232 IC.

SCON REGISTER:

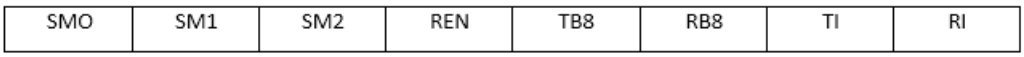


Figure 13.1:

It a bit addressable register used to set the mode in which serial communication takes place in the controller. The above figure shows the configuration of the SCON register. Here is the list of functions of each bit.

SM0, SM1: Serial Mode control Bits.

SMO	SM1	Mode	Description	Baud Rate
0	0	Mode 0	Shift register	(fosc/12)
0	1	Mode 1	9 bit UART	variable
1	0	Mode 2	9 bit UART	(fosc/64) or (fosc/32)
1	1	Mode 3	9 bit UART	variable

SM2: Multiprocessor mode control bit, logic 1 enables Multi processor mode and 0 for normal mode.

REN: Enables Serial reception. If set, it enables the reception otherwise the reception is disabled.

TB8: It is the 9th bit of the data that is to be transmitted.

RB8: It is used in modes 2 and 3, it is the 9th bit received by the microcontroller.
TI: It is known as Transmit Interrupt flag which is set by hardware to indicate the end of a transmission. It has to be cleared by the software.
RI: It is known as Receive Interrupt flag which is set by hardware to indicate the end of a reception. It has to be cleared by the software.

BAUD RATE: It is defined as number of bits transmitted or received per second and usually expressed in Bits per second bps. For mode 0 and mode 2 the baud rate is determined by means of 1/12, 1/32 or 1/64 of crystal frequency whereas for mode 1 and 3 it is determined by means of timer 1.

Baud Rate (bps)	TH1 (Hx value)
9600	FD
4800	FA
2400	F4
1200	E8

SBUF REGISTER: It is a 8 bit register that holds the data needed to be transmitted or the data that is received recently. The serial port of 8051 is full duplex so the microcontroller can transmit and receive data using the register simultaneously.

Serial Communication interfacing from microcontroller to PC:

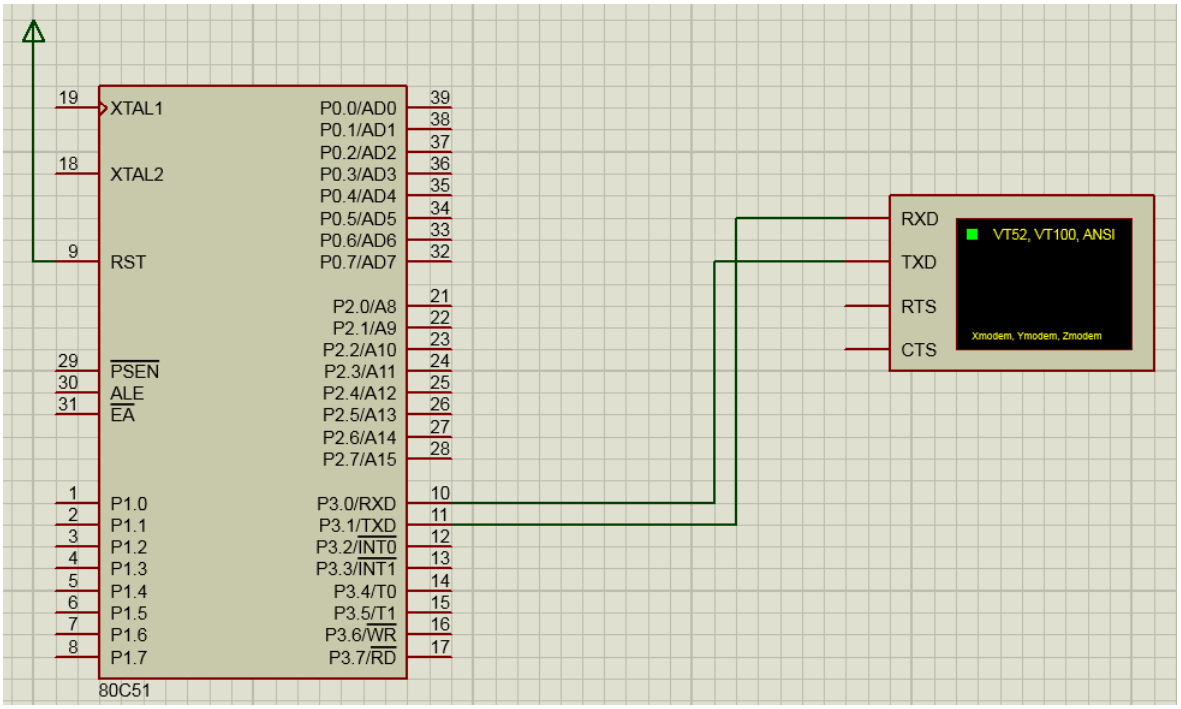


Figure 13.2: Serial Communication interfacing from 8051 Microcontroller to PC

Source Code:
/*Program to transmit a message from Microcontroller to PC serially using RS232.*/

```
#include<reg51.h>
```

```
void main()
{
        char A[]="Institute of Aeronautical Engineering";
        unsigned int i=0;
        P3=0x03;    //Initializing 8051 UART
        TMOD=0x20; // Timer1 Mode2 8 bit auto reload
        TH1=0xFD;  // 9600 bps
        SCON=0x50; // 8 Data bit, 1 start bit, bit 1 stop

        TR1=1;      // Timer1 ON

        while(1)
        {
                while(A[i]!='\0'){
                        SBUF=A[i];  // Placing character one
                         by one in the serial buffer register

                        while(TI==0); // It automatically poles
                        up to 1 when character is transmitted
                        TI=0;
                        i++;
                }
                i=0;
        }
}
```

13.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose “ADD s to Group” option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select “Options for Target”.

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

13.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the serial data received at the PC/terminal window.

13.8 Result

13.9 Further Probing Experiments

Q1. What is serial communication?

Q2. List out the different types of serial communications?

Q3. List out the features of 8051 microcontroller?

Q4. Define baud rate in serial communication?

Q5. Explain the different types of data transfer methods?

LAB-13 INTERFACING OF TEMPRATURE SENSOR, LM35 WITH 8051 MICROCONTROLLER

14.1 Introduction

Temperature sensors are widely used in electronic equipment to display the temperature. You can see the digital clock displaying the room temperature value. It is due to the temperature sensor embedded in it. The temperature value is analog. LM35 gives analog reading and microcontroller process digital data so we have to use a midway converter from Analog to Digital i.e. ADc0804 and display the result of a temperature on LCD.

14.2 Objective

14.2.1 Educational

- To understand the interfacing of temperature sensor with microcontroller.
- List out the features of P89V51RD2 microcontroller.

14.2.2 Experimental

- To learn how to measure the temperature using temperature sensor, LM35.

14.3 Prelab Preparation:

14.3.1 Reading

- Study the Background section below.

14.3.2 Written

- Review the steps in the procedure below to measure the temperature value.

14.4 Equipment needed

14.4.1 Hardware Requirements:

1. 8051 Development Board.
2. A serial 9 pin cable wired one to one from female connector to male connector.
3. PC with serial port.
4. 9V adaptor.
5. Connecting jumper and Connecting Wires.

14.4.2 Software Requirements:

1. Flash Magic tool.
2. Keil evaluation software.
3. Proteus 8 Professional.

14.5 Background

Temperature Sensor (LM35):

The LM35 Temperature Sensor provides the Analog Temperature Data to PCF8591, which it converts into Digital Values and sends to 8051. Upon receiving the digital values, the 8051 Microcontroller performs a small calculation and then displays the temperature on the LCD.

This simple experiment interface LM35 which is a very common Temperature Sensor giving high precision reading in terms of Analog Voltage with most popular 8051 Microcontroller P89V51RD2. LM35 gives analog reading and microcontroller process digital data so we have to use a midway converter from Analog to Digital i.e. PCF8591 and display the result of a temperature on LCD.

LM35 looks like a transistor it will give you temperature in Celsius in terms of milli volts. For example if the temperature is 25 C its output will give you 0.25V provided that you must supply at least 1V to it. Here how it looks:

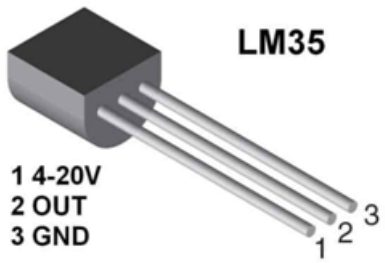


Figure 14.1: LM35 Pinout

PCF8591 8-bit A/D and D/A converter:

The PCF8591 is a single-chip, single-supply low-power 8-bit CMOS data acquisition device with four analog inputs, one analog output and a serial I2C-bus interface. Three address pins A0, A1 and A2 are used for programming the hardware address, allowing the use of up to eight devices connected to the I2C-bus without additional hardware. Address, control and data to and from the device are transferred serially via the two-line bidirectional I2C-bus.

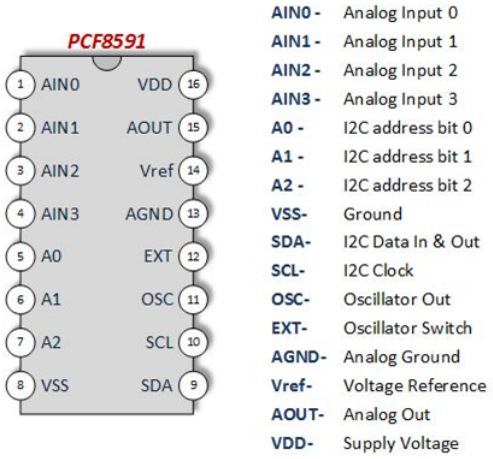


Figure 14.2: PCF8591 Pin diagram

Features and benefits

- Single power supply
- Operating supply voltage 2.5 V to 6.0 V
- Low standby current
- Serial input and output via I2C-bus
- I2C address selection by 3 hardware address pins
- Max sampling rate given by I2C-bus speed
- analog inputs configurable as single ended or differential inputs
- Auto-incremented channel selection
- Analog voltage range from VSS to VDD
- On-chip track and hold circuit
- 8-bit successive approximation A/D conversion
- Multiplying DAC with one analog output.

16×2 LCD Display

The term LCD stands for liquid crystal display. It is one kind of electronic display module used in an extensive range of applications like various circuits & devices like mobile phones, calculators, computers, TV sets, etc. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simply programmable, animations, and there are no limitations for displaying custom characters, special and even animations, etc.

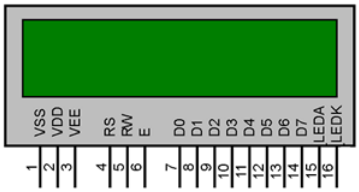


Figure 14.3: 16x2 LCD Module

- **Pin1 (Ground/Source Pin):** This is a GND pin of display, used to connect the GND ter-

minal of the microcontroller unit or power source.
• **Pin2 (VCC/Source Pin):** This is the voltage supply pin of the display, used to connect the supply pin of the power source.
• **Pin3 (V0/VEE/Control Pin):** This pin regulates the difference of the display, used to connect a changeable POT that can supply 0 to 5V.
• **Pin4 (Register Select/Control Pin):** This pin toggles among command or data register, used to connect a microcontroller unit pin and obtains either 0 or 1(0 = data mode, and 1 = command mode).
• **Pin5 (Read/Write/Control Pin):** This pin toggles the display among the read or writes operation, and it is connected to a microcontroller unit pin to get either 0 or 1 (0 = Write Operation, and 1 = Read Operation).
• **Pin 6 (Enable/Control Pin):** This pin should be held high to execute Read/Write process, and it is connected to the microcontroller unit & constantly held high.
• **Pins 7-14 (Data Pins):** These pins are used to send data to the display. These pins are connected in two-wire modes like 4-wire mode and 8-wire mode. In 4-wire mode, only four pins are connected to the microcontroller unit like 0 to 3, whereas in 8-wire mode, 8-pins are connected to microcontroller unit like 0 to 7.
• **Pin15 (+ve pin of the LED):** This pin is connected to +5V.
• **Pin 16 (-ve pin of the LED):** This pin is connected to GND.

Interfacing temperature sensor, LM35 with 8051 Microcontroller:

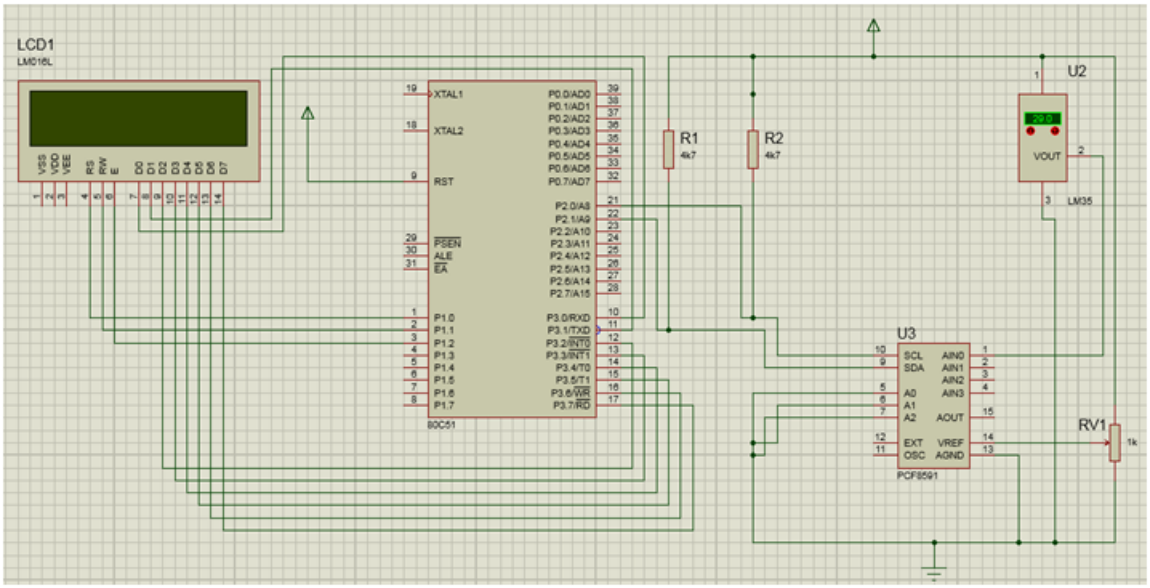


Figure 14.4: Interfacing temperature sensor with 8051 microcontroller:

Source Code:
/* Start — Device address/write — ack Check — control byte—ackcheck — start—device address/read—ack Check read dat — give ack— —stop*/

```
#include<reg51.h>
#include<intrins.h>

sbit sda=P2^1;   //i2c data line
sbit scl=P2^0;   //i2c clock line
```

```
sbit rs=P1^0;
sbit rw=P1^1;
sbit en=P1^2;

sbit a0=B^0;     //B sfr register individual bits
sbit a1=B^1;
sbit a2=B^2;
sbit a3=B^3;
sbit a4=B^4;
sbit a5=B^5;
sbit a6=B^6;
sbit a7=B^7;

void start();
void stop();
void check();
void delay();
void conversion(unsigned char);
void write(unsigned char);
unsigned char read();
void ack();
void disp(unsigned char *);
void enable();
unsigned char v;
void main()
{
        unsigned char com[4]={0x38,0x10,0x0e,0x80},sl;
        for(sl=0;sl<4;sl++)
        {
                P3=com[sl];
                rs=0;
                rw=0;
                enable();
        }
        disp("TEMPRATURE:");
        start();         //start i2c communication
        write(0x90);     //slave address in write mode
        check();         //acknowledgement check
        write(0x00);     //control byte-channel0 adc
        check();
        start();         //restart
        write(0x91);     //read mode
        check();
        while(1)         //infinite loop
        {
                v=read();
                ack();
                conversion(v);  //lcd screen displays the value
        }
        stop();
```

```
}
void start()
{
        scl=1;  //scl is high
        sda=1;  //sda is high
        _nop_();_nop_();        //no operation
        sda=0;
        scl=0;
}
void write(unsigned char val)
{
        unsigned char v1,v2,v3=0x80;
        v2=val;
        for(v1=0;v1<8;v1++)
        {
                sda=v2&v3;      //logical bitwise and
                scl=1;  //one clock cycle is req
                _nop_();_nop_();
                scl=0;
                v2=v2<<1;       //shift operation to send
                next bit of data
        }
}
void check()
{
        bit c;  //temp bit variable
        scl=1;
        c=sda;
        for(v=0;v<12;v++);
        scl=0;
        if(c==1)
        {
                stop();
        }
}
void stop()
{
        scl=1;  //scl is high
        sda=0;  //sda is low
        _nop_();_nop_();
        sda=1;
        scl=0;
}
void enable()
{
        unsigned int s2;
        en=1;
        for(s2=0;s2<2000;s2++); //delay
        en=0;
}
void disp(unsigned char *s)
```

```
{
        unsigned char s3;
        for(s3=0;s[s3]!='\0';s3++)
        {
                P3=s[s3];
                rs=1;
                enable();
        }
}
unsigned char read()
{
        sda=1;
        a7=sda; //msb bit of b register
        scl=1;
        _nop_();_nop_();
        scl=0;
        a6=sda;
        scl=1;
        _nop_();_nop_();
        scl=0;
        a5=sda;
        scl=1;
        _nop_();_nop_();
        scl=0;
        a4=sda;
        scl=1;
        _nop_();_nop_();
        scl=0;
        a3=sda;
        scl=1;
        _nop_();_nop_();
        scl=0;
        a2=sda;
        scl=1;
        _nop_();_nop_();
        scl=0;
        a1=sda;
        scl=1;
        _nop_();_nop_();
        scl=0;
        a0=sda; //lsb of b register
        scl=1;
        _nop_();_nop_();
        scl=0;
        v=B;
        return v;
}
void ack()
{
        sda=0;  //master acknowlidgement
        scl=1;
```

```
        _nop_();_nop_();
        scl=0;
}
void conversion(unsigned char val)
{
        unsigned char s4,s5,s6;
        s4=val;
        s5=s4/10;
        s6=s4%10;
        s5=s5|0x30;
        s6=s6|0x30;
        P3=0x8b;          //cursor pposition of lcd
        rs=0;
        enable();
        P3=s5;
        rs=1;
        enable();
        P3=s6;
        enable();
}
```

14.6 Procedure

Step 1: Give a double click on µvision 4 icon on the desk top, it will generate a window.

Step 2: To create new project go to project select new micro vision project.

Step 3: Select a drive where you would like to create your project.

Step 4: Create a new folder and name it with your project name.

Step 5: Open that project folder and give a name of your project executable and save it.

Step 6: After saving it will show some window there you select your microcontroller company i.e NXP from Phillips

Step 7: Select your chip as 8051 DEVELOPMENT KIT.

Step 8: After selecting chip click on OK then it will display some window asking to add STARTUP . Select YES.

Step 9: A target is created and startup is added to your project target.

Step 10: To write your project code select a new from menu bar.

Step 11: It will display some text editor, to save that select SAVE option from menu bar.

Step 12: By giving a name with extension .c and save it

Step 13: To add our c to target give a right click on Source Group, choose “ADD s to Group” option.

Step 14: It will display some window there select you have to add and click on ADD option.

Step 15: It will be added to our target and it shows in the project window.

Step 16: Now give a right click on target in the project window and select “Options for Target”.

Step 17: It will show some window, in that go to output option and choose Create Hex option by Selecting that box.

Step 18: : In the same window go to Linker option and choose Use Memory Layout from Target Dialog by Selecting the box, and click OK.

Step 19: Now to compile your project go to Project select Build Target option or press F7.

Step 20: Check the concern block of output and observe the results.

14.7 Precautions

1. Use the updated version of the Keil IDE tool.
2. Make sure correct power supply is given to the kit/Equipment.
3. Turn off the 8051 trainer board while giving hardware connections between Hexa Keypad and port pins of 8051 microcontroller.
4. Observe the temperature sensor value on LCD Display.

14.8 Result

14.9 Further Probing Experiments

Q1. What is LM35 temperature sensor?

Q2. Explain the uses of LM35 temperature sensor?

Q3. What will happen if we supply a voltage of 250V to the VCC of the LM35 temperature sensor?

Q4. List out the features of 8051 microcontroller?

Appendix A - Safety, Do's and Don'ts

Electricity, when improperly used, is very dangerous to people and to equipment. This is especially true in an industrial environment where large amounts of power is used, and where high voltages are present [1]; in environments where people are especially susceptible to electric shock such as maintenance of a high voltage system (while in operation) or in hospitals where electrical equipment is used to test or control physiological functions [2, 3]; and in an experimental or teaching laboratory where inexperienced personnel may use electrical equipment in experimental or nonstandard configuration.

Engineers play a vital role in eliminating or alleviating the danger in all three types of environments mentioned above. For conditions where standard equipment is used in standard configurations, govern- mental agencies and insurance underwriters impose strict laws and regulations on the operation and use of electrical equipment including switchgear, power lines, safety devices, etc. As a result, corporations and other organizations in turn impose strict rules and methods of operation on their employees and contractors. Engineers who are involved in using electrical equipment, in supervising others who use it, and in designing such systems, have a great responsibility to learn safety rules and practices, to observe them, and to see that a safe environment is maintained for those they supervise. In any working environment there is always pressure to "get the job done" and take short cuts. The engineer, as one who is capable of recognizing hazardous conditions, is in a responsible position both as an engineer and as a supervisor or manager and must maintain conditions to protect personnel and avoid damage to equipment.

Because of their non-standard activities, experimental laboratories are exempt from many of these rules and regulations. This puts more responsibility on the engineer in this environment to know and enforce the safest working procedures.

The knowledge and habit-forming experience to work safely around electrical equipment and the ability to design safe electrical equipment begins with the first student laboratory experience and con- tinues through life. This includes learning the types of electrical injuries and damage, how they can be prevented, the physiology of electrical injuries, and steps to take when accidents.

Physiology of Electrical Injuries

There are three main types of electrical injuries: electrical shock, electrical burns, and falls caused by electrical shock. A fourth type, 'sunburned' eyes from looking at electric arcs, such as arc-welding, is very painful and may cause loss of work time but is usually of a temporary nature. Other injuries may be indirectly caused by electrical accidents, e.g., burns from exploding oil-immersed switch gear or transformers.

Although electric shock is normally associated with high-voltage AC contact, under some circumstances death can occur from voltages from substantially less than the nominal 120 Volts AC found in residential systems. Electric shock is caused by an electric current passing through a part of the human body. The human body normally has a high resistance to electric currents so that a high voltage is usually required to cause lethal currents. This resistance is almost all in the skin, but when the skin is wet its resistance is much lower. When a person is hot and sweaty or is standing in water, contact with 120 Volts or less is likely to cause a fatal shock.

Electric shock is not a single phenomenon but is a disturbance of the nerves that is caused by electric current. A current through a part of the body such as the arm or leg will cause pain and muscle contraction. If a victim receives an electric shock from grasping a live conductor, a current of greater than 15 to 30 mA through the arm will cause muscle contractions so severe that the victim cannot let go. Similar currents through leg muscles may cause sudden contractions causing the victim to jump or fall, resulting in possible injuries or death. It is also possible for a prolonged period of contact of more than a minute or so to cause chest muscles to be contracted, preventing breathing and resulting in suffocation or brain damage from lack of oxygen.

The predominant cause of death by electric shock is generally attributed to ventricular fibrillation, which is an uncontrolled twitching or beating of the heart that produces no pumping action and therefore no blood circulation. Unless corrective action is taken, death follows quickly from lack of oxygen to the brain. While the amount of current that will cause fibrillation depends on several variables, 0.5 to 5A through the body will normally cause the very small current through the heart that causes fibrillation in most people. Larger currents than this through the heart causes contraction or clamping of the heart muscle and resulting death unless corrective action is taken. Prolonged contact of more than a minute or so may cause chest muscles to contract, preventing breathing and resulting in suffocation or brain damage from lack of oxygen.

Death by electric shock is most often attributed to ventricular fibrillation, which is an uncontrolled twitching or beating of the heart that produces no pumping action and therefore no blood circulation. Unless corrective action is taken, death follows quickly from lack of oxygen to the brain. While the amount of current that will cause fibrillation depends on several variables, 0.5 to 5 amperes through the body will normally cause the very small current (approximately 1 mA) through the heart that is sufficient to cause fibrillation in most people. Larger currents than this through the heart cause contraction or clamping of the heart muscle, resulting in death unless corrective action is taken.

Electric burns may be caused by electric currents flowing in or near parts of the body. Such burns are similar to burns from ordinary heat sources, except that those caused by high-frequency currents are generally deeper and take longer to heal the other burns. Electrocution will often leave severe burns at the points where the current entered and left the body.

Source of Electric Shock

Since electric shock is caused by an electric current through a part of the body, it is prevented by not allowing the body to become part of any electric circuit. From this viewpoint, electric circuits may be classified as either grounded or ungrounded.

Electric circuits may be classified as either grounded or ungrounded. Grounded circuits are safer for most conditions, since they result in known voltages at other points in the circuit and provide easier and better protection against faulty conditions in the circuit. The disadvantage is that a person standing on a non-insulated floor can receive a shock by touching only one conductor.

Almost all electric power generation, transmission, and distribution systems are grounded to protect people and equipment against fall conditions caused by windstorms, lightning, etc. Residential, commercial, and industrial systems such as lighting and heating are always grounded for greater safety. Communication, computer, and similar systems are grounded for safety reasons and to prevent or reduce noise, crosstalk, static, etc. Many electronic equipment or instruments are grounded for safety and noise prevention, also. Common examples are DC power supplies, oscilloscopes, oscillators, and analog and digital multimeters.

Ungrounded circuits are used in systems where isolation from other systems is necessary, where low voltages and low power are used, and in other instances where obtaining a ground connection is difficult or impractical. In the ungrounded circuit, contact with two points in the circuit that are at different potentials is required to produce an electrical shock. The hazard is that with no known ground, a hidden fault can occur, causing some unknown point to be grounded, in which case, touching a supposedly safe conductor while standing on the ground could result in an electric shock.

Protecting People and Equipment in the Laboratory

Prevention of electric shock to individuals and damage to equipment in the laboratory can be done by strict adherence to several common-sense rules summarized below:

Protecting People

1. When hooking up a circuit, connect to the power source last, while power is off.
2. Before making changes in a circuit, turn off or disconnect the power first, if possible.
3. Never work alone where the potential of electric shock exists.
4. When changing an energized connection, use only one hand. Never touch two points in the circuit that are at different potentials.
5. Know that the circuit and connections are correct before applying power to the circuit.
6. Avoid touching capacitors that may have a residual charge. The stored energy can cause a severe shock even after a long period of time.
7. Insulate yourself from ground by standing on an insulating mat where available.

The above rules and the additional rules given below also serve to protect instruments and other circuits from damage.

Protecting Equipment

1. Set the scales of measurement instrument to the highest range before applying power.
2. Before making changes in a circuit, turn off or disconnect the power first, if possible.
3. When using an oscilloscope, do not leave a bright spot or trace on the screen for long periods of time. Doing so can burn the image into the screen.
4. Be sure instrument grounds are connected properly. Avoid ground loops and accidental grounding of "hot" leads.
5. Check polarity markings and connections of instruments carefully before connecting power.
6. Never connect an ammeter across a voltage source, but only in series with a load.
7. Do not exceed the voltage or current ratings of circuit elements or instruments. This particularly applies to wattmeters, since the current or voltage rating may be exceeded with the needle still reading on the scale.
8. Be sure any fuses and circuit breakers are of suitable value.

When connecting electrical elements to make up a network in the laboratory, it easy to lose track of various points in the network and accidentally connect a wire to the wrong place. One procedure to help avoid this problem is to connect first the main series loop of the circuit, then go back and add the elements in parallel.

Types of Equipment Damage Excessive currents and voltages can damage instruments and other circuit elements. A large over- current for a short time or a smaller over-current for a longer time will cause overheating, resulting in insulation scorching and equipment failure.

Blown fuses are the most common equipment failure mode in this laboratory. The principal causes for these failures include:
• incorrectly wired circuits;
• accidental shorts;
• switching resistance settings while power is applied to the circuit;
• changing the circuit while power is applied;
• using the wrong scale on ammeter;
• connecting an ammeter across a voltage source;
• using a low-power resistor box (limit 1/2 amp) when high power is required;
• turning on an auto-transformer at too high a setting.

All of these causes are the result of carelessness by the experimenter.

Some type of insulating material, such as paper, cloth, plastic, or ceramic, separates conductors that are at different potentials in electrical devices. The voltage difference that this material can withstand is determined by design (type, thickness, moisture content, temperature, etc.). Exceeding the voltage rating of a device by an appreciable amount can cause arcing or corona, resulting insulation breakdown, and failure.

Some electrical devices can also be damaged mechanically by excessive currents. An example is the D'Arsonval meter, the indicator in most analog metering instruments. A large pulse of over current will provide mechanical torque that can cause the needle to wrap around the pin at the top of the scale, thereby causing permanent damage even though the current may not have been on long enough to cause failure due to overheating.

After Accident Action

Since accidents do happen despite all efforts to prevent them, plans for appropriate reaction to an accident can save time and lives. Such a plan should include immediate availability of first aid material suitable for minor injuries or for injuries that are likely because of the nature of the work. Knowledge of how to obtain trained assistance such as Emergency Medical Services (EMS) should be readily available for everyone.

Treating victims for electrical shock includes four basic steps that should be taken immediately. Step two requires qualification in CPR and step three requires knowledge of mouth-to-mouth resuscitation. Everyone who works around voltages that can cause dangerous electrical shock should take advantage of the many opportunities available to become qualified in CPR and artificial respiration.

Immediate Steps After Electric Shock

1. Shut off all power and remove victim from the electric circuit. If the power cannot be shut off immediately,use an insulator of some sort, such as a wooden pole, to remove victim from the circuit. Attempts to pull the victim from the circuit with your hands will almost always result in your joining the victim in the electric shock.

2. If you are qualified in CPR, check for ventricular fibrillation or cardiac arrest. If either is detected, external cardiac massage should be started at once. Whether you are qualified in CPR or not, notify EMS and the ECE Department at once, using the telephone numbers listed below.

3. Check for respiratory failure and take appropriate action. This may have resulted from physical paralysis of respiratory muscles or from a head injury. Sometimes many hours pass before normal respiration returns. Artificial respiration should be continued until trained EMS assistance arrives.

4. Check for and treat other injuries such as fractures from a fall or burns from current entry and exit sites. Investigations are always after accidents. As an engineer you will be involved as a part of the investigating team or in providing information to an investigator. Information obtained and notes written immediately after the emergency will aid this investigation and assist in preventing future accidents of a similar nature.

Investigations are always made after accidents. As an engineer, you will be involved as a part of the investigating team or in providing information to an investigator. Information obtained and notes written immediately after the emergency will aid the investigation and assist in preventing future accidents of a similar nature.

Emergency Numbers

Fire / EMS: 911 or (864) 656-2222
Student Health Center: (864) 656-2233
ECE Department Office: (864) 656-5650

Appendix A References

1. W.F. Cooper, Electrical Safety Engineering, Newnes-Butterworth, London-Boston, 1978.

2. W.H. Buschsbaum and B. Goldsmith, Electrical Safety in the Hospital, Medical Economics Company, Oradel, NJ, 1975.

3. J.G. Wester, editor, Medical Instrumentation Application and Design, Houghton Mifflin Company, Boston, 1978.

Do's and Don'ts

Do's	
1	Program to blink all the alternative LEDs of Port1 with 100 ms in 8051 Microcontroller.
2	Design and write the common cathode display codes to display even numbers from 0 to 9 on seven segment display.
3	Embedded C program for 8051 microcontroller to rotate the stepper motor 360° clockwise by half step sequence and 360° anticlockwise by full step sequence.
4	Embedded C program to display the sine wave at the output of digital to analog converter (DAC).

Don'ts	
1	Program to blink all the alternative LEDs of Port1 with 100 ms in 8051 Microcontroller.
2	Design and write the common cathode display codes to display even numbers from 0 to 9 on seven segment display.
3	Embedded C program for 8051 microcontroller to rotate the stepper motor 360° clockwise by half step sequence and 360° anticlockwise by full step sequence.
4	Embedded C program to display the sine wave at the output of digital to analog converter (DAC).
SAFETY NORMS	
1	Program to blink all the alternative LEDs of Port1 with 100 ms in 8051 Microcontroller.
2	Design and write the common cathode display codes to display even numbers from 0 to 9 on seven segment display.
3	Embedded C program for 8051 microcontroller to rotate the stepper motor 360° clockwise by half step sequence and 360° anticlockwise by full step sequence.

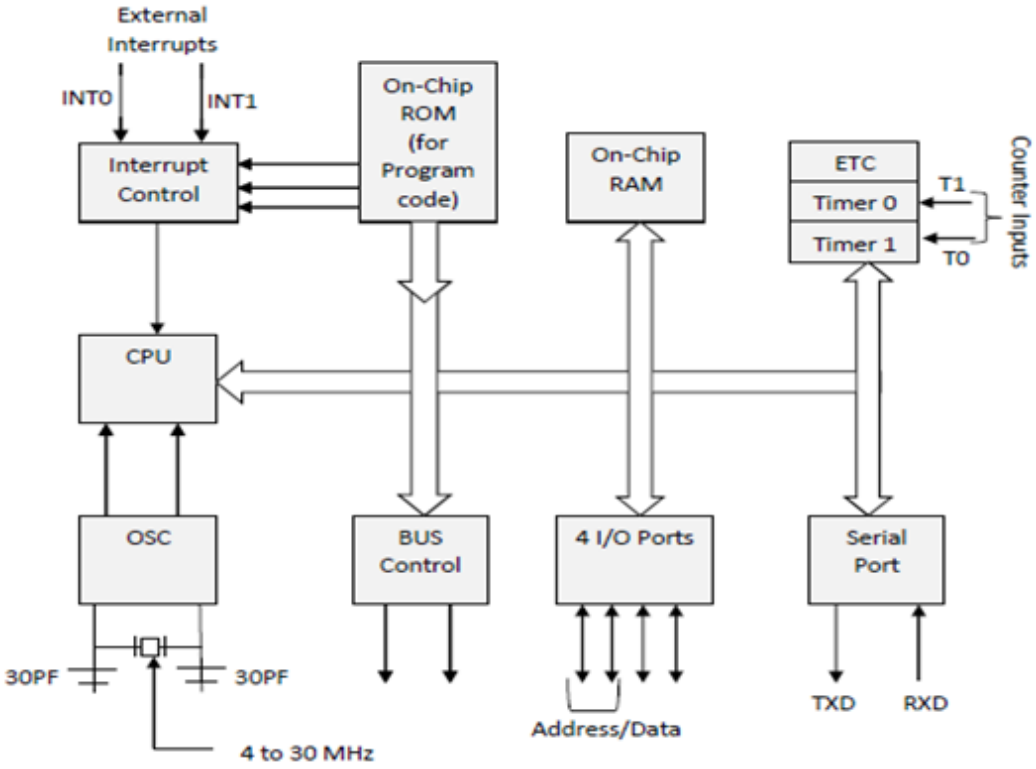
Appendix B - Pin Description and Block diagram of 8051 Microcontroller

8051 microcontroller is designed by Intel in 1981. It is an 8-bit microcontroller. It is built with 40 pins DIP (dual inline package), 4kb of ROM storage and 128 bytes of RAM storage, 2 16-bit timers. It consists of are four parallel 8-bit ports, which are programmable as well as addressable as per the requirement. An on-chip crystal oscillator is integrated in the microcontroller having crystal frequency of 12 MHz.

Let us now discuss the architecture of 8051 Microcontroller. In the following diagram, the system bus connects all the support devices to the CPU. The system bus consists of an 8-bit data bus, a 16-bit address bus and bus control signals. All other devices like program memory, ports, data memory, serial interface, interrupt control, timers, and the CPU are all interfaced together through the system bus.

8051 Microcontroller Architecture

Following diagram is 8051 Microcontroller architecture. Let us have a look at each part or block of this Architecture of microcontroller.



Central Processor Unit (CPU)
As we know that the CPU is the brain of any processing device of the microcontroller. It monitors and controls all operations that are performed on the Microcontroller units. The User has no control over the work of the CPU directly . It reads program written in ROM memory and executes them and do the expected task of that application.

Interrupts
As its name suggests, Interrupt is a subroutine call that interrupts of the microcontrollers main operations or work and causes it to execute any other program, which is more important at the time of operation. The feature of Interrupt is very useful as it helps in case of emergency operations. An Interrupts gives us a mechanism to put on hold the ongoing operations, execute a subroutine and then again resumes to another type of operations.

The Microcontroller 8051 can be configured in such a way that it temporarily terminates or pause the main program at the occurrence of interrupts. When a subroutine is completed, Then the execution of main program starts. Generally five interrupt sources are there in 8051 Microcontroller. There are 5 vectored interrupts are shown in below
• INTO
• TFO
• INT1
• TF1
• R1/T1

Out of these, (INT0) and (INT1) are external interrupts that could be negative edge triggered or low level triggered. When All these interrupts are activated, set the corresponding flogs except for serial interrupt,.The interrupt flags are cleared when the processor branches to the interrupt service routine (ISR). The external interrupt flags are cleared when the processor branches to the interrupt service routine, provides the interrupt is a negative edge triggered whereas the timers and serial port interrupts two of them are external interrupts, two of them are timer interrupts and one serial port interrupt terminal in general.

Memory
Microcontroller requires a program which is a collection of instructions. This program tells microcontroller to do specific tasks. These programs require a memory on which these can be saved and read by Microcontroller to perform specific operations of a particular task. The memory which is used to store the program of the microcontroller is known as code memory or Program memory of applications. It is known as ROM memory of microcontroller also requires a memory to store data or operands temporarily of the micro controller. The data memory of the 8051 is used to store data temporarily for operation is known RAM memory. 8051 microcontroller has 4K of code memory or program memory, that has 4KB ROM and also 128 bytes of data memory of RAM.

BUS Basically Bus is a collection of wires which work as a communication channel or medium for transfer of Data. These buses consists of 8, 16 or more wires of the microcontroller. Thus, these can carry 8 bits,16 bits simultaneously. Hire two types of buses that are shown in below
• Address Bus
• Data Bus

Address Bus: Microcontroller 8051 has a 16 bit address bus for transferring the data. It is used to address memory locations and to transfer the address from CPU to Memory of the microcontroller. It has four addressing modes that are

- Immediate addressing modes.
- Bank address (or) Register addressing mode.
- Direct Addressing mode.
- Register indirect addressing mode.

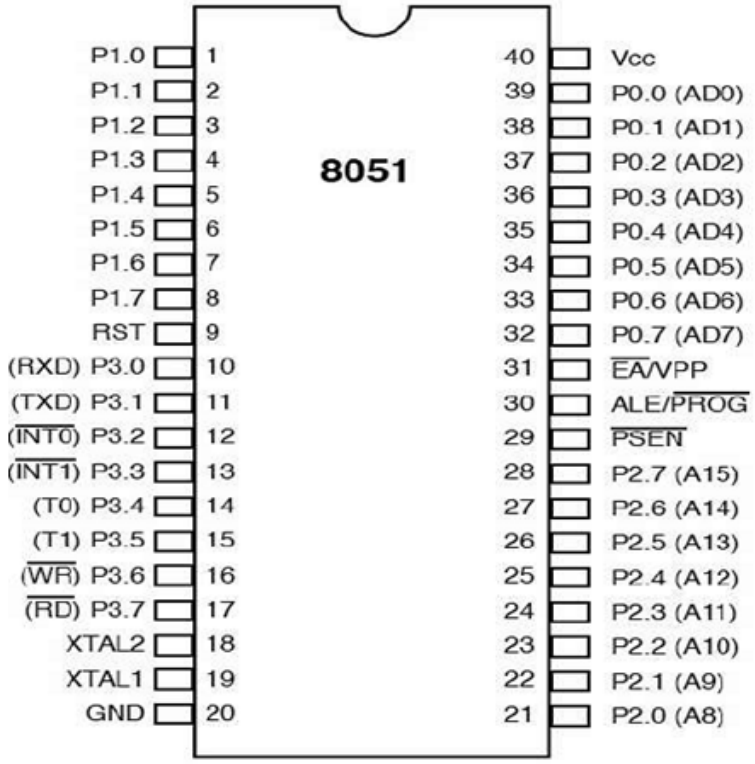
Data Bus: Microcontroller 8051 has 8 bits of the data bus, which is used to carry data of particular applications.

Oscillator
Generally, we know that the microcontroller is a device, therefore it requires clock pulses for its operation of microcontroller applications. For this purpose, microcontroller 8051 has an on-chip oscillator which works as a clock source for Central Processing Unit of the microcontroller. The output pulses of oscillator are stable. Therefore, it enables synchronized work of all parts of the 8051 Microcontroller.

Input/Output Port
Normally microcontroller is used in embedded systems to control the operation of machines in the microcontroller. Therefore, to connect it to other machines, devices or peripherals we require I/O interfacing ports in the microcontroller interface. For this purpose microcontroller 8051 has 4 input, output ports to connect it to the other peripherals.

Timers/Counters
8051 microcontroller has two 16 bit timers and counters. These counters are again divided into a 8 bit register. The timers are used for measurement of intervals to determine the pulse width of pulses.

8051 Microcontroller Pin Description



Pins 1 to 8: These pins are known as Port 1. This port doesn't serve any other functions. It is internally pulled up, bi-directional I/O port.
Pin 9: It is a RESET pin, which is used to reset the microcontroller to its initial values.
Pins 10 to 17: These pins are known as Port 3. This port serves some functions like interrupts, timer input, control signals, serial communication signals RxD and TxD, etc.
Pins 18 & 19: These pins are used for interfacing an external crystal to get the system clock.
Pin 20: This pin provides the power supply to the circuit.
Pins 21 to 28: These pins are known as Port 2. It serves as I/O port. Higher order address bus signals are also multiplexed using this port.
Pin 29: This is PSEN pin which stands for Program Store Enable. It is used to read a signal from the external program memory.
Pin 30: This is EA pin which stands for External Access input. It is used to enable/disable the external memory interfacing.
Pin 31: This is ALE pin which stands for Address Latch Enable. It is used to demultiplex the address-data signal of port.
Pins 32 to 39: These pins are known as Port 0. It serves as I/O port. Lower order address and data bus signals are multiplexed using this port.
Pin 40: This pin is used to provide power supply to the circuit.

Appendix C - Proteus 8.0 Professional Simulation

Download Free Proteus 8 Professional Full Version-Proteus 8 Professional is a software used to simulate microprocessors, schematics and PCB designs. Proteus 8 also includes components such as FLOW PCB layout, ISIS schematic capture and VSM (Visual System Modeling). This software, usually used by Electronics Engineering Students and Electronics work, because this software has a more complete component database than other Electronics software. If you are interested in trying it, please you can download the Free Proteus 8 Professional Full Version on the link provided.

Proteus Pro Crack Download is a powerful computer-designed (CAD) software designed specifically for designers to easily build and validate circuit boards. With this software, you can create complex electrical circuit designs. In addition to solid knowledge, you need a set of tools to get your job done. With Proteus PCB Design, you can design PCBs with very detailed electrical components such as microcontrollers, microprocessors, and other components commonly used in electrical circuits.

Proteus Professional Cracked For Free provides interactive simulation and customization tools that allow designers to view and edit the properties of all elements in the table. Useful for innovative PCB design, testing, and layout. Drawing, simulation, verification, and export of drawings. Proteus Design Suite provides a comprehensive software package for up-and-coming VSM simulation engineers and a new streaming engine that delivers a truly integrated and intuitive development environment.

The main components of the Proteus Design Suite
This software contains two key elements that rotate the entire functionality of the program.

ISIS: Acronym for intelligent schematic input system. A program that can perform the electrical design of a circuit, including all kinds of components such as resistors, coils, capacitors, power supplies, and even microprocessors.
ARES: Acronym for advanced routing and editing software. It is a PCB or PCB design tool with electronic routing, mounting, and processing functions.
Where can I download ISIS and ARES? Well, you can't get them as a standalone application so you can make the most of all the features you have to pay for, but you should have the full version of Proteus Labcenter Electronics. typically before deciding whether to buy.

In addition to these two programs, Proteus Pro Full Version Free includes a variety of VSMs that can be integrated into ISIS to simulate various integrated circuit functions in real-time, and Electra, an automatic routing module that can automatically detect routes. Find the best route to improve the circuit speed between the accessories that come with the unit.

Design, test and locate the full integrated system in the Proteus Keygen schematic system before ordering the original. Proteus VSM brings flexible growth to integrated workflows.

How to Install Proteus 8 Professional:

1. Turn off the Internet & Antivirus Connection.
2. Extract the file that you downloaded using WinRAR, then open the folder that you extracted.
3. Next, run the Installer file named "proteus8.0.SP1.exe".
4. Click "Next", then give a check mark on "I accept the terms . . . ", then click "Next".
5. Click "Next" 3x, then select "Typical" and wait until the Installation process is complete.
6. When it's finished, don't open the software first.
7. Open the "crack" folder, then copy all the files and paste them into the "Proteus" Installation Folder Directory.
   Ex: `C:\Program` Files (x86)`\Labcenter` Electronics`\Proteus` 8 Professional
8. Replace / Overwrite.
9. Done.

New features of Proteus 8.0 Professional Simulation:

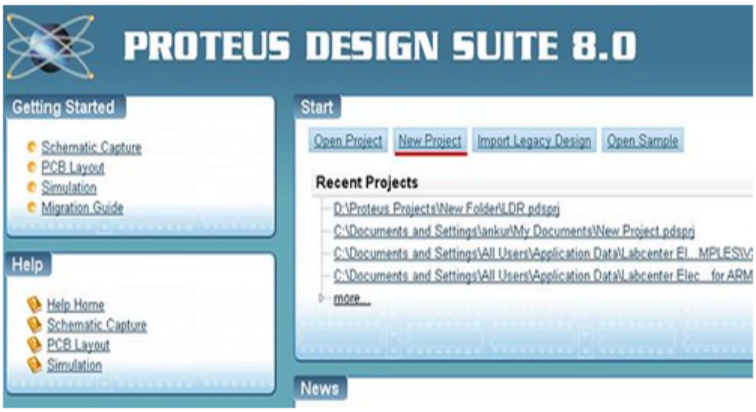
- Ease of use with powerful features.
- Also, easy PCB layout, test, and layout.
- More than 800 variants of microcontrollers.
- Moreover, easy to use the PCB layout kit.
- An integrated suite of tools for PCB design.
- Furthermore, a very smart development environment.
- Co-simulation microcontroller simulation.
- And much more.

System Requirements Proteus 8.0 Professional Simulation:

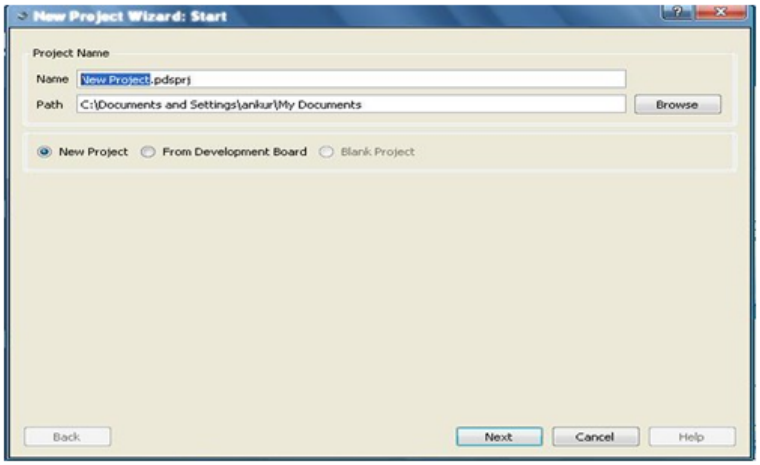
- Operating System: Windows XP, Windows Vista, Windows 7, Windows 8 / 8.1, Windows 10.
- Processor: Intel Pentium processor (2 GHz).
- Memory: 3 GB RAM.
- Storage: 500MB of free storage space.

Getting started with Proteus 8 Professional Simulation:

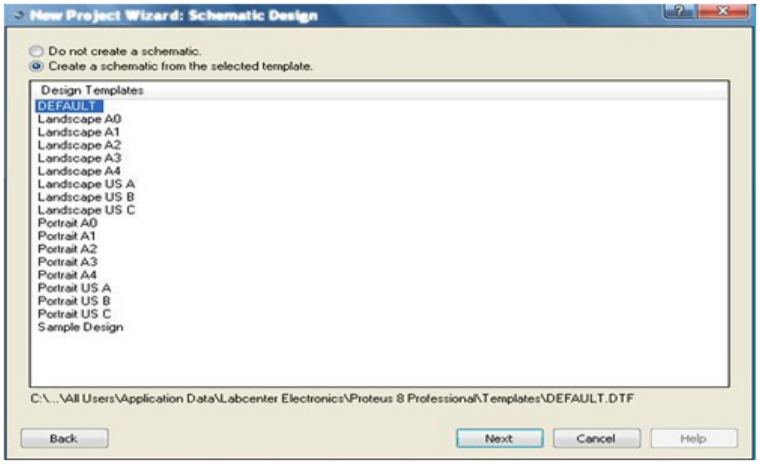
1. Click on the desktop icon of Proteus 8 and Open the Home page of Proteus 8. It will be different from its previous version which used to open up directly the schematic layout.



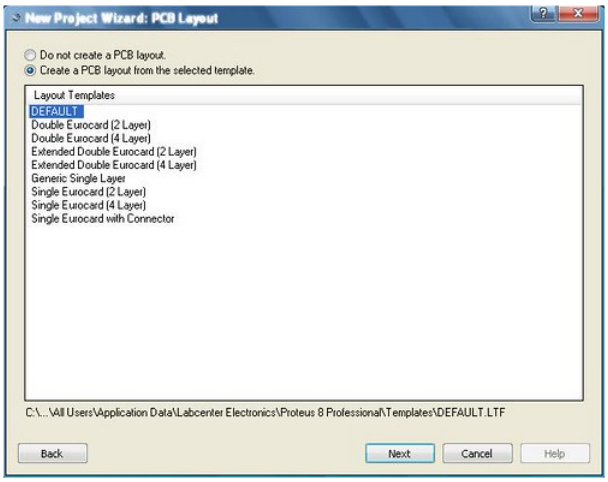
2. Now, click on "New Project" option and then, a pop-up window will be opened up in which you have to name the project and choose the directory to save the project. It will be saved automatically with ".pdsprj" extension.



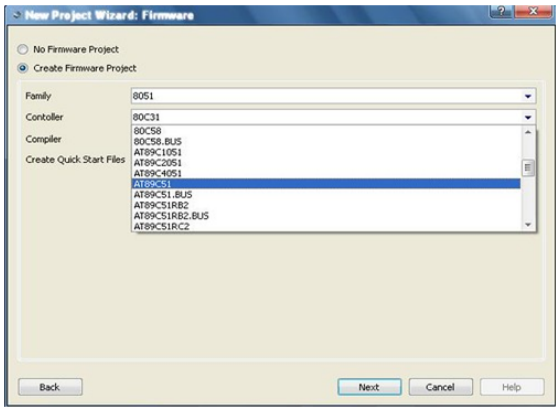
3. The Next step is to select the Schematic layout from the categories and 'Default' will be A4 size. You can select the other layout from the list provided.

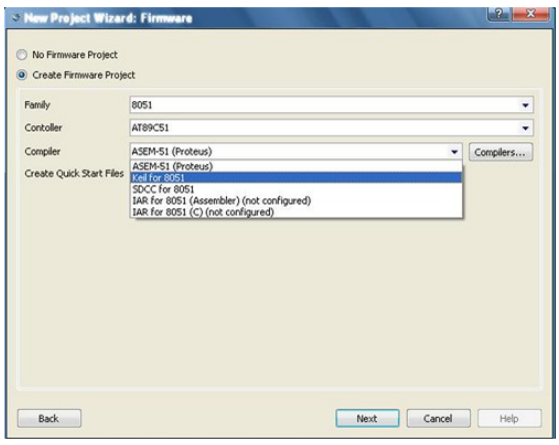


4. In next step, you can also create PCB layout in the next step by selecting it and choosing the required layout. You can also select 'Do not create PCB layout' option, if you don't want any PCB design of your schematic circuit.



5. Now, you have to choose the installed compiler and the microcontroller you are going to use. The compiler will allow you to write the code and directly execute it without uploading the hex file as we used to do in previous version.





Now, you can see the three tabs:- Schematic capture, PCB layout, and Source code. You will also find the pre-selection of microcontroller in Schematic capture window and header file selected in Source code window. All the work can be done in Proteus 8 application software.