



Jaypee University of Information Technology  
Solan (H.P.)

LEARNING RESOURCE CENTER

Acc. Num. *SP07114* Call Num:

**General Guidelines:**

- ◆ Library books should be used with great care.
- ◆ Tearing, folding, cutting of library books or making any marks on them is not permitted and shall lead to disciplinary action.
- ◆ Any defect noticed at the time of borrowing books must be brought to the library staff immediately. Otherwise the borrower may be required to replace the book by a new copy.
- ◆ The loss of LRC book(s) must be immediately brought to the notice of the Librarian in writing.

Learning Resource Centre-JUIT



SP07114



# WIRELESS DIGITAL SCORE BOARD

**RISHABH SHARMA**

**071106**

**VIKRAM PRATAP SINGH**

**071158**

**AKSHAT PANDEY**

~~071120~~ 071103



**May – 2011**



**Submitted in partial fulfillment of the Degree of Bachelor of Technology**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION**

**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,**

**WAKNAGHAT**

## CERTIFICATE

This is to certify that the work titled "WIRELESS DIGITAL SCORE BOARD" submitted by "AKSHAT PANDEY, RISHABH SHARMA & VIKRAM PRATAP SINGH" in partial fulfillment for the award of degree of Electronics and Communication of Jaypee University of Information Technology, Wagnaghat have been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Vikas

VIKAS HASTIR

Department of Electronics and Communication Engineering,

Jaypee University of Information Technology

Wagnaghat, Solan -173215, India




May 17, 2011

## ACKNOWLEDGEMENT

Apart from the efforts by us, the success of this project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We would like to show our greatest appreciation to Mr Vikas Hastir, the project guide. We also owe our heartiest thanks to Prof. Dr. Sunil Bhooshan (Professor Head of Department –Electronics and Communication Engineering) who has always inspired confidence in us to take initiative. He has always been motivating and encouraging.

Finally, thanks to all our family members who have supported us in our every grim phase.

Akshat Pandey	Roll No. 071103	(Electronics and Communication Engineer)	
Rishabh Sharma	Roll No. 071106	(Electronics and Communication Engineer)	
Vikram Pratap Singh	Roll No. 071158	(Electronics and Communication Engineer)	

Date: 24/5/2011

## ACKNOWLEDGEMENT

Apart from the efforts by us, the success of this project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We would like to show our greatest appreciation to Mr Vikas Hastir, the project guide. We also owe our heartiest thanks to Prof. Dr. Sunil Bhooshan (Professor Head of Department –Electronics and Communication Engineering) who has always inspired confidence in us to take initiative. He has always been motivating and encouraging.

Finally, thanks to all our family members who have supported us in our every grim phase.

Akshat Pandey	Roll No. 071103	(Electronics and Communication Engineer)
Rishabh Sharma	Roll No. 071106	(Electronics and Communication Engineer)
Vikram Pratap Singh	Roll No. 071158	(Electronics and Communication Engineer)

Date:



## SUMMARY

Wireless Digital Score Board is the development of a prototype of an interface between keyboard and a microcontroller, LCD and a microcontroller, PC and a microcontroller and IR module and microcontroller. This interface allows a user to operate display of a score board using a remote control. LCD interfacing is used to display the current score. Coverage area and transfer rate are major factors for considering wireless communication. There are various modes of wireless communication e.g. via radio waves, via infrared waves. This development will make it possible to store hundreds of thousands of transistors into one chip.

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development will make it possible to store hundreds of thousands of transistors into one chip. In our project, we have tried to demonstrate the infrared remote control technology sending digital data through infrared spectrum.

Our main objective behind this project is to be more familiar with the technology used to transmit data using IR, microcontroller's interfaces with various peripherals. Another objective behind choosing this project is to understand the use of microcontroller as a system controlling the protocol for transmission and reception of the digital data. We tried to demonstrate this technology as a part of the project. For the future modifications, we can modify this project for certain application like accessing devices at greater distances.

Akshat Pandey

Rishabh Sharma

Vikram Pratap Singh

Engineering

Date:

Vikas  
Supervisor

Mr. Vikas Hastir

Department of Electronics  
and Communication

Date:

## **TALBLE OF CONTENTS**

<b>Chapter No.</b>	<b>Topics</b>	<b>Page No.</b>
	<b>CERTIFICATE</b>	<b>I</b>
	<b>ACKNOWLEDGEMENT</b>	<b>II</b>
	<b>SUMMARY</b>	<b>III</b>
	<b>LIST OF FIGURES</b>	<b>VI</b>
	<b>LIST OF TABLES</b>	<b>VIII</b>
<b>CHAPTER 1</b>	<b>INTRODUCTION</b>	
	1.1 OVERVIEW	
	1.2 LITERATURE SURVEY	
	1.3 OBJECTIVE	
	1.4 ORGANIZATION OF THE PROJECT REPORT	
<b>CHAPTER 2</b>	<b>CIRCUIT SIMULATION OF WIRELESS SCORE BOARD</b>	
	2.1 CIRCUIT DIAGRAM	
	2.2 FLOWCHART	
	2.3 ALGORITHM	
<b>CHAPTER 3</b>	<b>HARDWARE DESIGN &amp; DEVELOPMENT OF WIRELESS DIGITAL SCORE BOARD</b>	
	3.1 MICROCONTROLLER	
	3.2 LCD	
	3.3 IR MODULE	
	3.4 RS- 232	
	3.5 PUSH BUTTONS	

## CHAPTER 4 TESTING & RESULTS OF WIRELESS DIGITAL SCORE BOARD

CONCLUSION

FUTURE ASPECTS

REFERENCES

APPENDIX A

APPENDIX B

APPENDIX C

APPENDIX D



## LIST OF FIGURES

Fig 2.1	.....	4
Fig 2.2	.....	5
Fig 2.3	.....	9
Fig 2.4	.....	10
Fig 2.5	.....	10
Fig 2.6	.....	12
Fig 2.7	.....	12
Fig 2.8	.....	14
Fig 2.9	.....	15
Fig 2.10	.....	16
Fig 2.11	.....	17
Fig 2.12	.....	17
Fig 2.13	.....	18
Fig 2.14	.....	18
Fig 2.15	.....	18
Fig 3.1	.....	21
Fig 3.2	.....	25
Fig 3.3	.....	28
Fig 3.4	.....	31
Fig 3.5	.....	32

Fig 3.6	.....	33
Fig 3.7	.....	35
Fig 3.8	.....	37
Fig 3.9	.....	38
Fig 3.10	.....	39
Fig 4.1	.....	40

## LIST OF TABLES

Table 2.1 .....	7
Table 2.2 .....	11
Table 3.1 .....	26
Table 3.2.....	36



## CHAPTER 1

### INTRODUCTION

#### 1.1 Overview

Wireless Digital Score Board is the development of a prototype of an interface between keyboard and a microcontroller, LCD and a microcontroller, PC and a microcontroller and IR module and microcontroller. This interface allows a user to operate display of a score board using a remote control. LCD interfacing is used to display the current score. Serial communication has been used to communicate with the main unit (display unit). IR interfacing has been used to operate display unit using IR communication. The interfaces were implemented using the Atmel's AT89S52 microcontroller.

Circumstances that we find ourselves in today in the field of microcontrollers had their beginnings in the development of technology of integrated circuits. This development will make it possible to store hundreds of thousands of transistors into one chip. That is a prerequisite for production of microprocessors, and the first computers were made by adding external peripherals such as memory, input-output lines, timers and other. Further increasing of the volume of the package resulted in creation of integrated circuits. These integrated circuits contained both processor and peripherals. That is how the first chip containing a microcomputer, or what would later be known as a microcontroller came about.

Now a day, wireless communication is preferred over wired communication. A lot of research is being done on wireless communication. Coverage area and transfer rate are major factors for considering wireless communication. There are various modes of wireless communication e.g. via radio waves, via infrared waves. Radio waves are very commonly used for long range communication with high power transmitter. Infrared communication is commonly used for short range LOS communication. Almost all consumer electronics items are operated using an infrared remote control. Only this way, this technology came into the hands of common man. Almost every person knows how to operate devices using remote control and loves to use it.

But there are very rare person who know how this technology works. In this project, we will try to demonstrate this technology by sending digital data through infrared spectrum.

## **1.2 Literature survey**

On earlier days of wireless communication, radio waves were most popular. When inventions led to use of infrared spectrum as a media for digital data to be send, the major advantage was the higher bandwidth and very simple circuitry. Infrared (IR) LEDs are used as IR transmitters. Photo transistors were used as receivers. Inventions led to better system performance but increases system complexity.

For transmitting data using IR in this project, we have used carrier modulated digital data receiver which is very widely used in IR remote control receivers. They have carrier frequency varying from 25 KHz to 45 KHz. Normally the very popular carrier frequency is 38 KHz. Amplitude shift keying is the type of modulation for transmitting digital data.

For transmitting data in ASK fashion, it is necessary to transmit data in serial fashion. To send data to receiver successfully it is very much necessary to transmit data compatible to the receiver. This function is performed, in this project, using a microcontroller which makes the transmitting data compatible with the receiver. It makes different codes for different switches pressed and transmits it serially which makes it very much compatible with IR receivers.

In home appliances with remote control, there are various protocols which both transmitter and receiver obey to make a system consistent. RC5 protocol is widely used. It was first made by Philips for their remote controls. Almost all IR receivers are made keeping this protocol in mind. The main feature of this protocol is that the data is sent in small burst. New versions of protocol although different from RC5, but they all transmit data in small bursts. So every IR receivers are made to receive data in small burst. In this project we are also using this RC5 protocol.

## **1.3 Objective**

Our main objective behind this project is to be more familiar with the technology used to transmit data using IR, microcontroller's interfaces with various peripherals.

We tried to demonstrate this technology as a part of the project. For the future modifications, we can modify this project for certain application like accessing devices at greater distances.

Another objective behind choosing this project is to understand the use of microcontroller as a system controlling the protocol for transmission and reception of the digital data.



## **1.4 Organization of project report**

This report has been divided into various chapters. Brief description of these chapters is given below:

### **Chapter 1**

Chapter 1 gives the general introduction of the project. It contains the literature survey i.e. the literature that has been surveyed to get information regarding the project. It also contains objective behind this project.

### **Chapter 2**

In this chapter we have circuit diagram, algorithm and flowchart of the wireless digital score board and the description of the softwares that we have used for coding and simulation of the project.

### **Chapter 3**

Chapter 3 mainly talks about the components used in the project. It tells about the theory, working and design of components. It also tells about advantages and disadvantages of using these components in the project.

### **Chapter 4**

This is the last chapter of the project report which states the testing results of the project.

At the end of report we have Conclusion, Appendices and References, etc.



## CHAPTER 2

### CIRCUIT SIMULATION OF WIRELESS DIGITAL SCORE BOARD

#### 2.1 Circuit Diagram

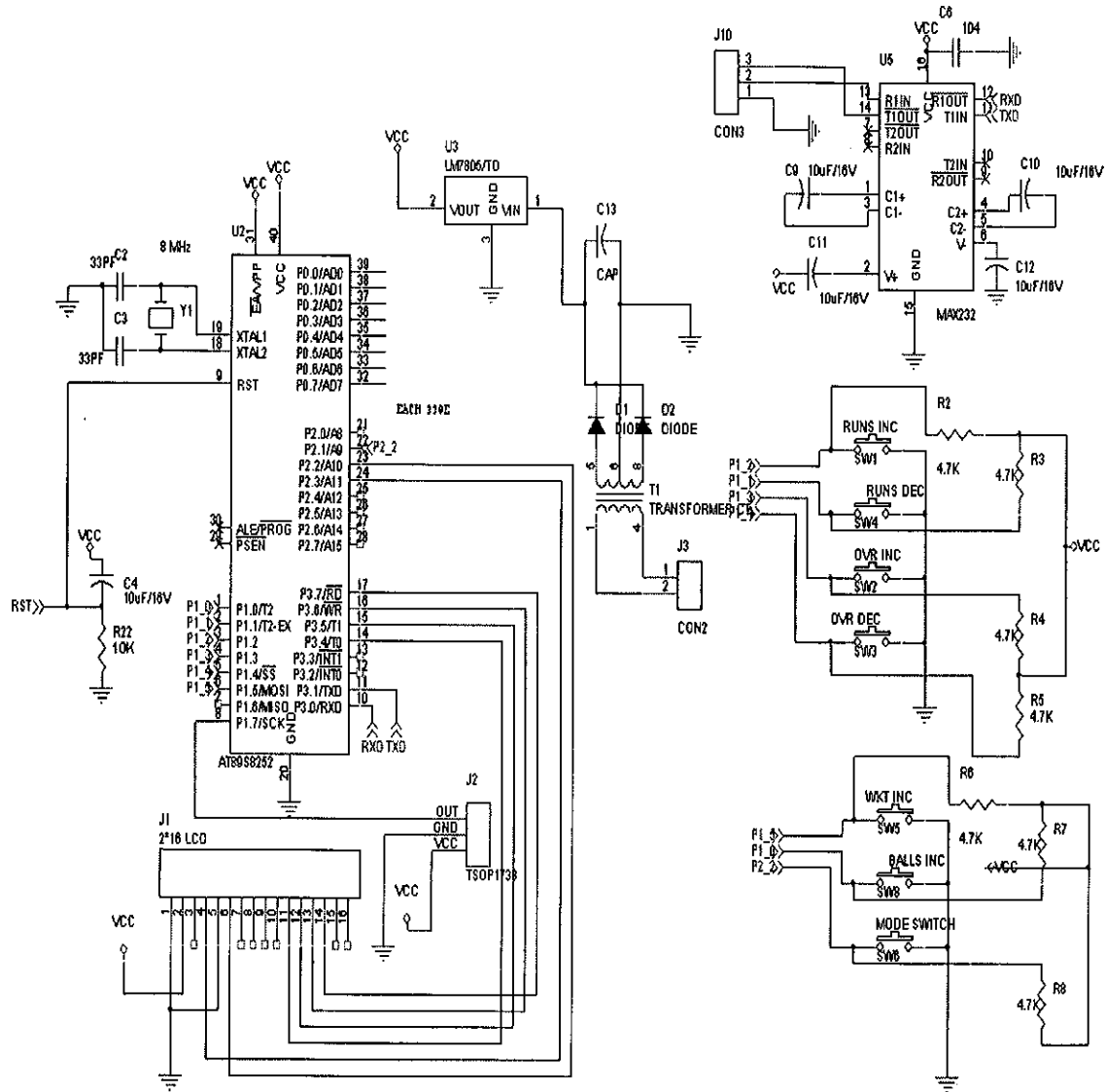


Fig 2.1: Circuit Diagram

## 2.2 Flowchart

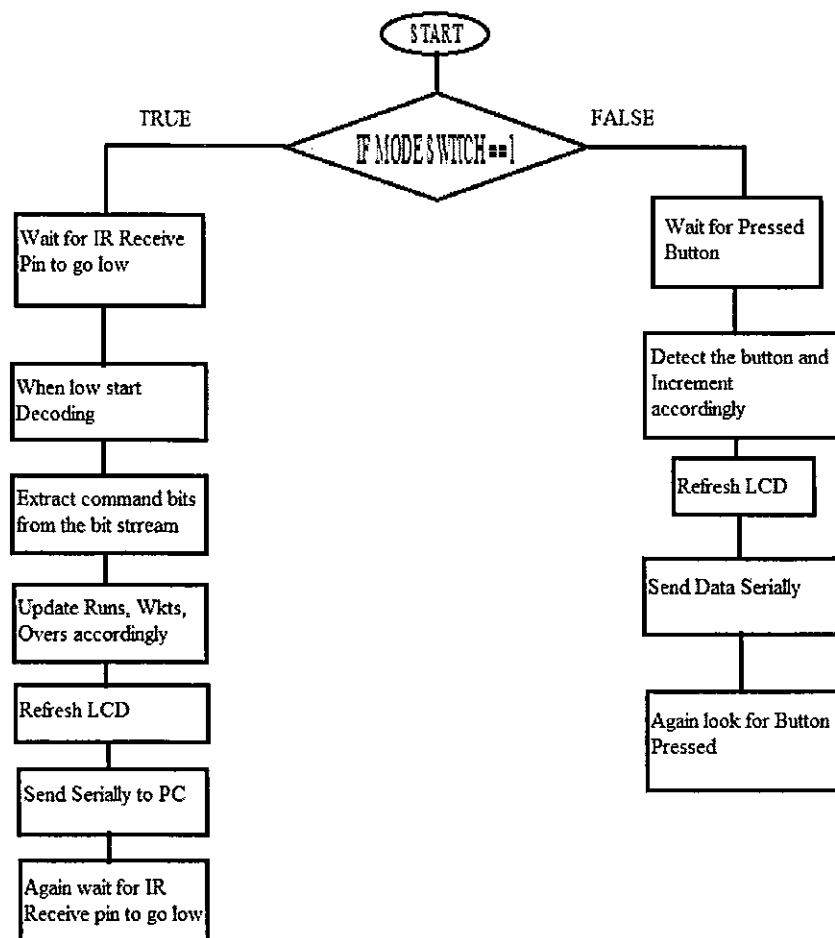


Fig 2.2: Flow Diagram of project

## 2.3 Algorithm

Check If Mode Switch==1

If true

1. Wait For Receive Pin To Go Low.
2. When Low Start Decoding.

3. Extract Command Bits from the BIT STREAM.
4. Update Runs, Overs, Wkts, Balls accordingly.
5. Refresh LCD
6. Send serially to PC

If False

1. Wait for Button Pressed.
2. Detect the button and Increment corresponding value
3. Refresh LCD
4. Send data serially
5. Again look for Button pressed.

For simulation of this project we have used two softwares. These are
















- Kiel uVision3
- Proteus 7 Professional

#### **(i) KEIL- $\mu$ VISION3:**

The  $\mu$ Vision3 IDE is a Windows-based software development platform that combines a robust editor, project manager, and makes facility.  $\mu$ Vision3 integrates all tools including the C compiler, macro assembler, linker/locator, and HEX file generator.



**TABLE 2.1: Debug menu and debug commands**

Debug Menu	Toolbar	Shortcut	Description
Start/Stop Debug Session		<b>Ctrl+F5</b>	Start or stop $\mu$ Vision3 Debug Mode
Go		<b>F5</b>	Run (execute) until the next active breakpoint
Step		<b>F11</b>	Execute a single-step into a function
Step Over		<b>F10</b>	Execute a single-step over a function
Step Out of current Function		<b>Ctrl+F11</b>	Maintain project components (targets, groups, files), configure tool environment, and manage books
Run to Cursor Line			Execute until the current cursor line is reached
Stop Running		<b>ESC</b>	Select a CPU from the Device Database
Breakpoints...			Open Breakpoint dialog
Insert/Remove Breakpoint			Toggle breakpoint on current line
Enable/Disable Breakpoint		<b>Alt+F7</b>	Enable/disable breakpoint on current line
Disable All Breakpoints			Disable all breakpoints in the program
Kill All Breakpoints		<b>F7</b>	Remove all breakpoints in the program
Show Next Statement			Show next executable statement/instruction
Enable/Disable Trace Recording		<b>Ctrl+F7</b>	Enable trace recording for instruction review
View Trace Records			Review previous executed instructions
<u>Execution Profiling</u>			Set the Execution Profiling to Off, Time or Calls
<u>Setup Logic Analyzer</u>			Open the setup dialog for the Logic Analyzer
Memory Map...			Open the Memory Map dialog
Performance Analyzer...			Open setup dialog for the Performance Analyzer
Inline Assembly...			Open the inline assembler dialog
Function Editor (Open Ini File)...			Edit debug functions and the debugger initialization file



**Create project file and select CPU:****Create project file folder and specify project name:**

To create a new project file select from the  $\mu$ Vision3 menu Project – New Project.... This opens a standard Windows dialog that asks you for the new project file name. You should use a separate folder for each project. You can simply use the icon Create New Folder in this dialog to get a new empty folder.

Select this folder and enter the file name for the new project, i.e. Project1.  $\mu$ Vision3 creates a new project file with the name PROJECT1.UV2 which contains a default target and file group name. You can see these names in the Project Workspace – Files.

Select microcontroller from device database:

When you create a new project  $\mu$ Vision3 asks you to select a CPU for your project. The Select Device dialog box shows the  $\mu$ Vision3 device database. Just select the microcontroller you use. For the example in this chapter we are using the Philips LPC2106 controller. This selection sets necessary tool options for the LPC2106 device and simplifies in this way the tool configuration.

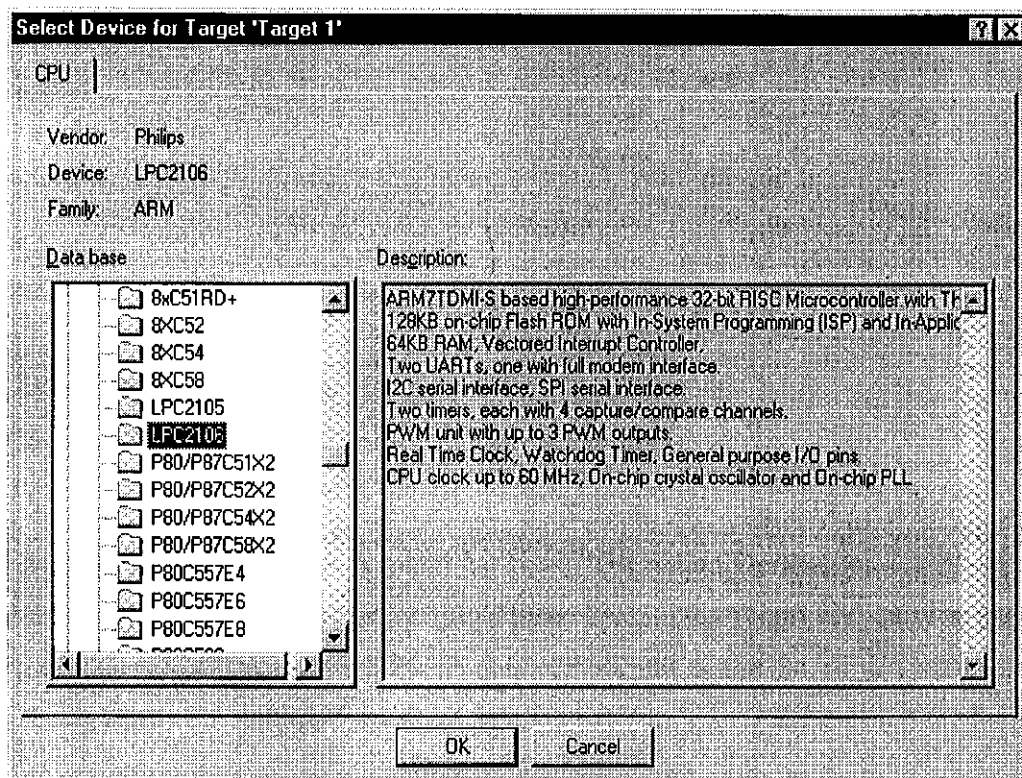


Fig 2.3: Window for selecting device

### Copy and add the CPU startup code:

A embedded program requires CPU initialization code that needs to match the configuration of your hardware design. This Startup Code depends also on the tool chain that you are using. Since you might need to modify that file to match your target hardware, the file should be copied to your project folder.

For most devices,  $\mu$ Vision3 asks you to copy the CPU specific Startup Code to your project. This is required on almost all projects (exceptions are library projects and add-on projects). The Startup Code performs configuration of the microcontroller device and initialization of the compiler run-time system.

Therefore you should answer with YES to this question.

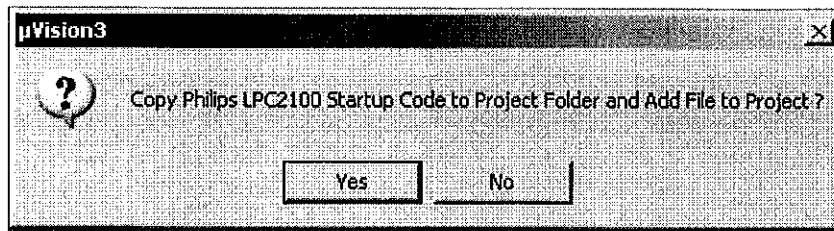


Fig 2.4: Dialog box to copy start up code

### Set Tool Options for Target:

µVision3 lets you set options for your target hardware. The dialog Options for Target opens via the toolbar icon or via the Project - Options for Target menu item. In the Target tab you specify all relevant parameters of your target hardware and the on-chip components of the device you have selected. The following dialog shows the settings for our example.

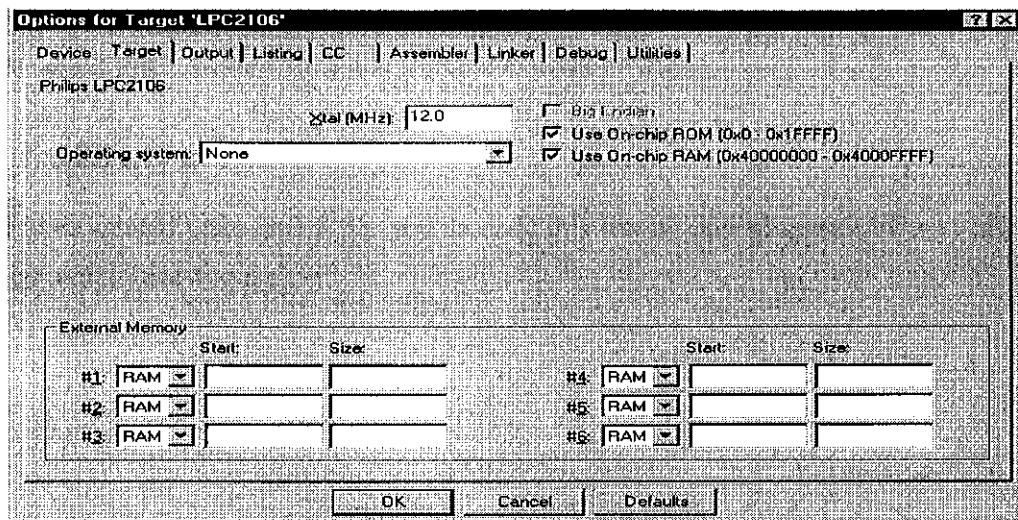


Fig 2.5: Window for selecting target

**Table 2.2: Options of the target dialog**

Dialog Item	Description
<b>Xtal</b>	specifies the external clock frequency of your device. Several microcontrollers use an on-chip PLL to generate the CPU clock. In this cases the value is not identical with the XTAL frequency. Check your hardware design carefully to determine the correct value.
<b>Operating System</b>	allows you to select a Real-Time Operating System for your project.
<b>Use On-chip ROM / RAM</b>	defines the address spaces for the on-chip memory components for the linker/locater. Note that on some devices you need to reflect this configuration in the Startup Code.

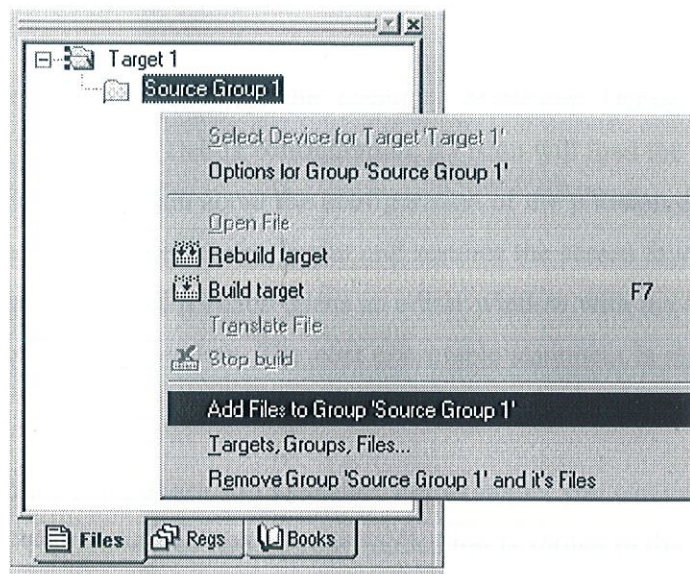


### **Create New Source Files:**

You may create a new source file with the menu option File – New. This opens an empty editor window where you can enter your source code.  $\mu$ Vision3 enables when you save your file with the dialog File – Save As... under a filename with the extension \*.ASM.

### **Add source files to project:**

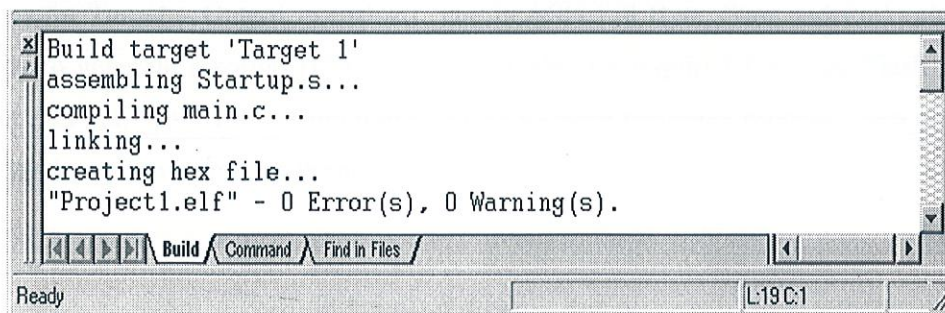
Once you have created your source file you can add this file to your project.  $\mu$ Vision3 offers several ways to add source files to a project.



**Fig 2.6: window for adding source files to project**

### **Build project:**

Typical, the tool settings under Options – Target are all you need to start a new application. You may translate all source files and link the application with a click on the Build Target toolbar icon. When you build an application with syntax errors, µVision3 will display errors and warning messages in the Output Window – Build page. A double click on a message line opens the source file on the correct location in a µVision3 editor window.



**Fig 2.7: window on clicking the build project**



### **@ Start debug mode:**

You start the debug mode of  $\mu$ Vision with the Debug – Start/Stop Debug Session command. Depending on the Options for Target – Debug configuration,  $\mu$ Vision will load the application program and run the startup code. For information about the configuration of the  $\mu$ Vision debugger refer to Set Debug Options.  $\mu$ Vision saves the editor screen layout and restores the screen layout of the last debug session. If the program execution stops,  $\mu$ Vision opens an editor window with the source text or shows CPU instructions in the disassembly window. The next executable statement is marked with a yellow arrow.

During debugging, most editor features are still available. For example, you can use the find command or correct program errors. Program source text of your application is shown in the same windows. The  $\mu$ Vision debug mode differs from the edit mode in the following aspects:

- The Debug Menu and Debug Commands are available. The additional debug windows are discussed in the following.
- The project structure or tool parameters cannot be modified. All build commands are disabled.

### **Create Hex File:**

Once you have successfully generated your application you can start debugging. After you have tested your application, it is required to create an Intel HEX file to download the software into an EPROM programmer or simulator.  $\mu$ Vision3 creates HEX files with each build process when Create HEX file under Options for Target – Output is enabled. The FLASH Fill Byte, Start and End values direct the OH166 utility to generate a sorted HEX files; sorted files are required for some Flash programming utilities. You may start your PROM programming utility after the make process when you specify the program under the option Run User Program #1.

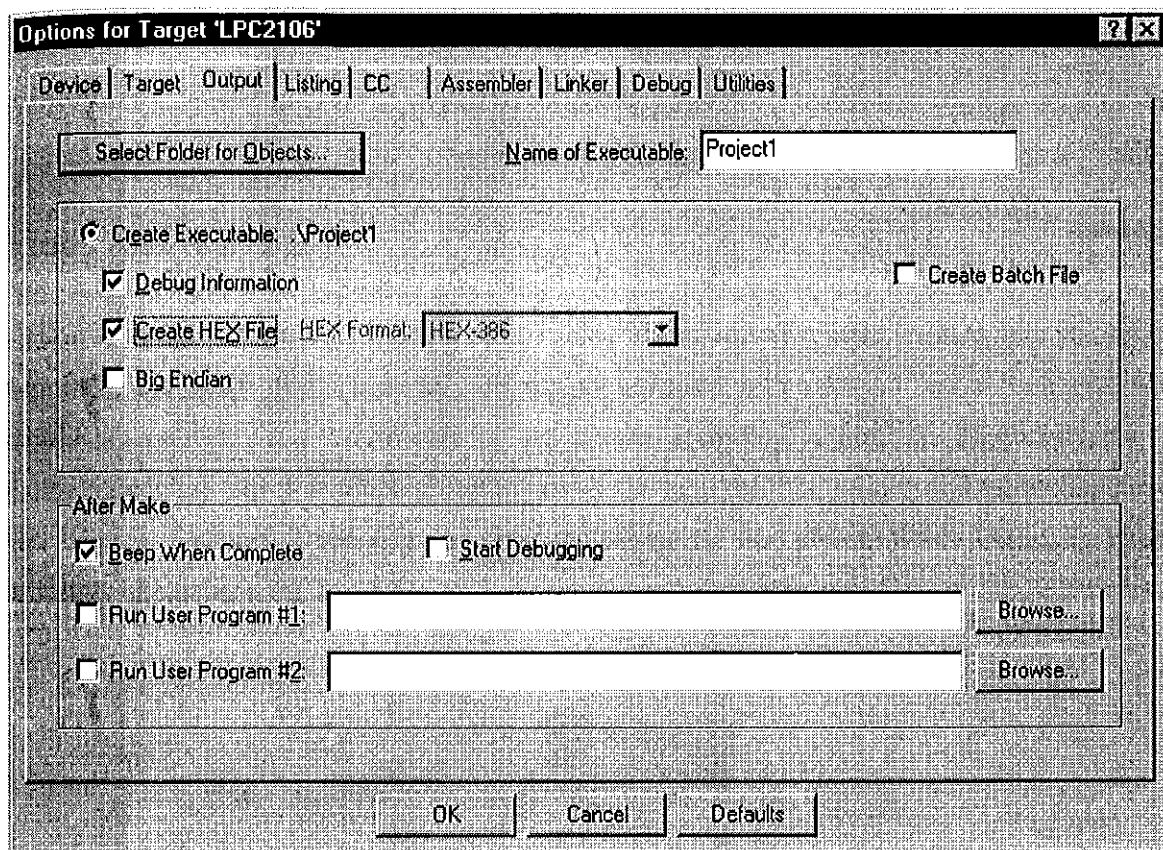


Fig 2.8: Creating hex file

## (ii) PROTEUS VERSION 7.4

### ISIS & ARES Core Applications:

Fully integrated Shape Based Auto-Router replaces the previous, grid based router. Users of PCB Design at Level 2 and above can drive the router interactively with the ability to route selected nets. All users can run a pre-configured routing schedule automatically. Per net 'strategies' are replaced by 'net classes' and the management of these is incorporated into the Design Rule Manager.

### Proteus VSM Additions:

Added syntax highlighting for EASYHDL scriptable generators. Interactive configuration of the trace diagnostics that is you can enable or disable particular diagnostics whilst the simulation is running.

### Visual Aids to Design:

ISIS is designed to be as user friendly as possible and provides two main ways to help you see what is happening during the design process – objects are encircled with a dashed line or ‘twitched’ when the mouse is over them and mouse cursors will change according to function. Essentially, the object-twitching scheme tells you which object the mouse is over (the ‘hot’ object) and the mouse cursor tells you what will happen when you left click the mouse on that object. While extremely intuitive, a summary of cursors used, together with their actions, is provided below

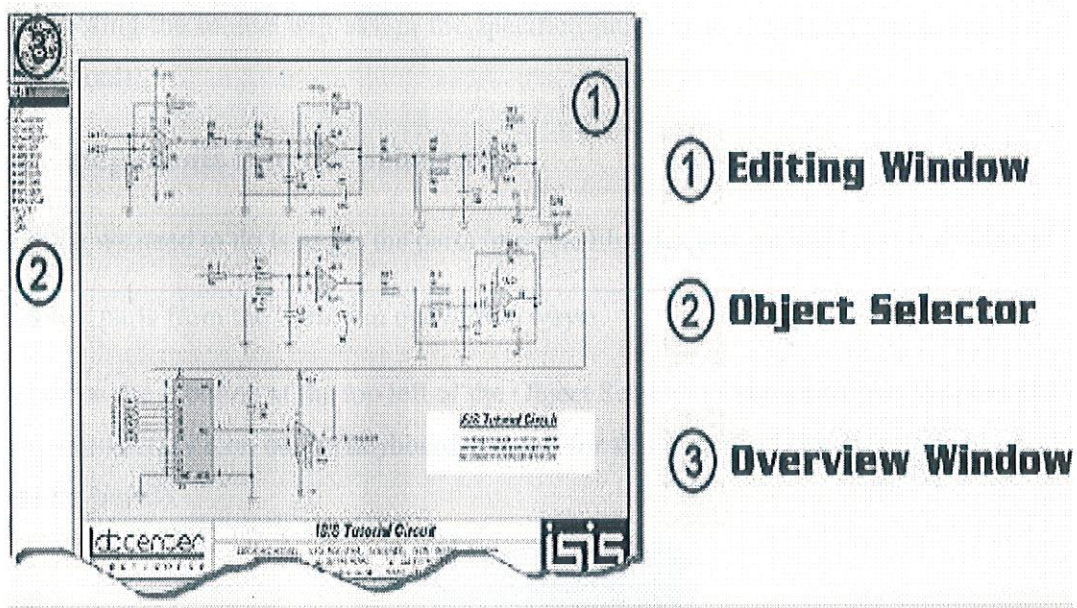





Fig: 2.9: A Proteus main window snapshot


Standard Cursor – used in selection mode when not over a ‘hot’ object.


Placement Cursor – placement of an object will commence on left click.


 Hot placement cursor for wires - a wire will either start (or stop if currently placing a wire) when the mouse is left clicked.

 Hot placement cursor for buses - a bus will either start (or stop if currently placing a wire) when the mouse is left clicked.

 Object under the mouse will be selected on left click.

 The object under the mouse can be moved by left depressing the mouse and dragging it into the desired position.

 The wire segment can be dragged by left depressing the mouse and dragging it to the desired position.

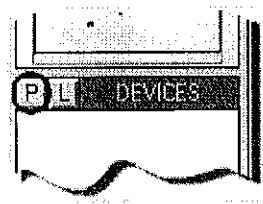
 Left clicking the mouse will assign the specified property to the object (used with the Property Assignment Tool).

### **Picking Components into the Schematic:**

The first thing we need to do is to get the parts from the libraries that we need in our schematic.

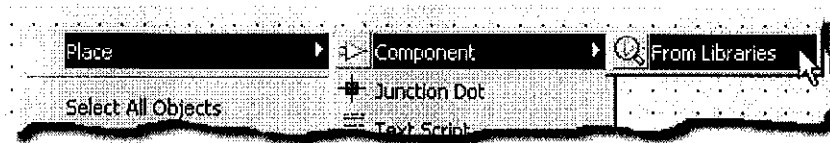
You can select parts from the library in one of two ways:

1. Click on the P button at the top left of the Object Selector as shown below. You can also use the Browse Library icon on the keyboard shortcut for this command (by default this is the P key on the keyboard).



**Fig 2.10: Picking components from the Object Selector**

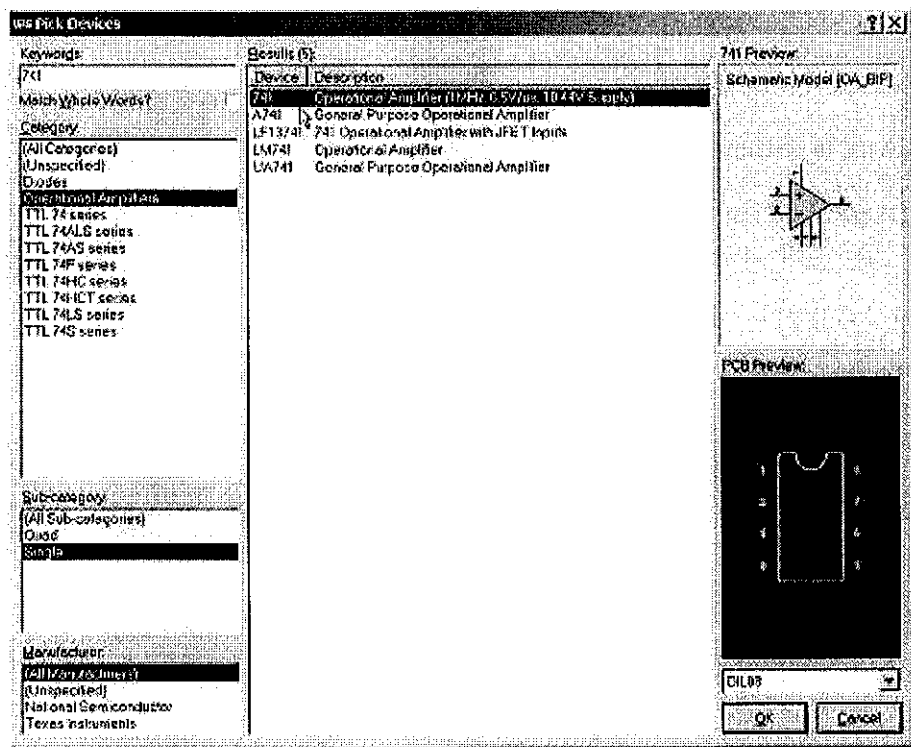
2. Right click the mouse on an empty area of the schematic and select Place – Component - From Libraries from the resulting context menu as shown below :



**Fig 2.11: Picking components from the Context Menu**

Either of these two methods will cause the Device Library Browser dialogue form to appear.

3. The next stage is to find the desired components in the libraries. You can search the library for components in a number of ways. In the case of parts where you know the part name its usually best to start the search with that. Try entering 741 into the Keywords field. There are a lot of parts with 741 in their name or description, but you can refine the results to a more reasonable set by selecting the Operational Amplifiers category, as shown below:



**Fig 2.12: The Browse Libraries dialogue form**

## Placing Components onto the Schematic:

Having selected the parts we need the next thing is to actually place them on the drawing area – the Editing Window. The easiest place to start is by doing the buffer circuit shown at the top left of the tutorial circuit. This is reproduced in more detail below:

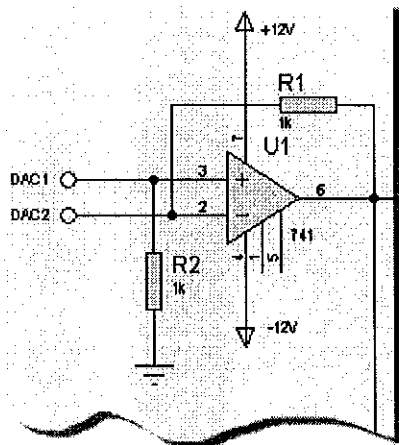


Fig 2.13: A close up view of the initial section of circuitry to be drawn

Make sure that you are in component mode (i.e. that the component icon is selected) and start by clicking the mouse on the 741 in the Object Selector. You should see that the Overview Window above the selector changes to display a preview of the selected device. The screenshots below show the state of the Object Selector and the Overview Window after selection of the 741 part.



Fig 2.14: Object Selector

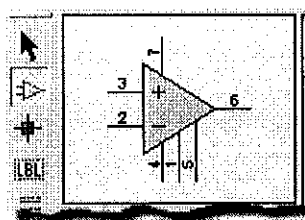


Fig 2.15: Overview Window

Not only does the Overview Window show a preview of the device but it does so at the current orientation of the device. As you rotate or mirror the part (via the Rotation and/or Mirror icons), the



device is redrawn to preview the new orientation. The preview remains until the device is placed or until another command or action is performed.

## CHAPTER 3

### **HARDWARE DESIGN & DEVELOPMENT OF WIRELESS DIGITAL SCORE BOARD**

The hardware set up consist of following major components

- Microcontroller
- LCD
- IR Module
- RS232
- Push buttons

#### **3.1 Microcontroller**

Microcontroller, as the name suggests, are small controllers. They are like single chip computers that are often embedded into other systems to function as processing/controlling unit. For example, the remote control you are using probably has microcontrollers inside that do decoding and other controlling functions. They are also used in automobiles, washing machines, microwave ovens, toys ... etc, where automation is needed.

The AT89S52 microcontroller is a low-power, high power CMOS 8-bit microcontroller with 8K bytes of in system programmable Flash Memory. The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry standard 80C51 instruction set and pinout. It is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications.

## Features:

The AT89S52 provides the following standard features:

- 8K Bytes of In-System Reprogrammable Flash Memory
- Fully Static Operation: 0 Hz to 33 MHz
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Eight Interrupt Sources
- Programmable Serial Channel
- 4.0V to 5.5V Operating Range

## Pin description:

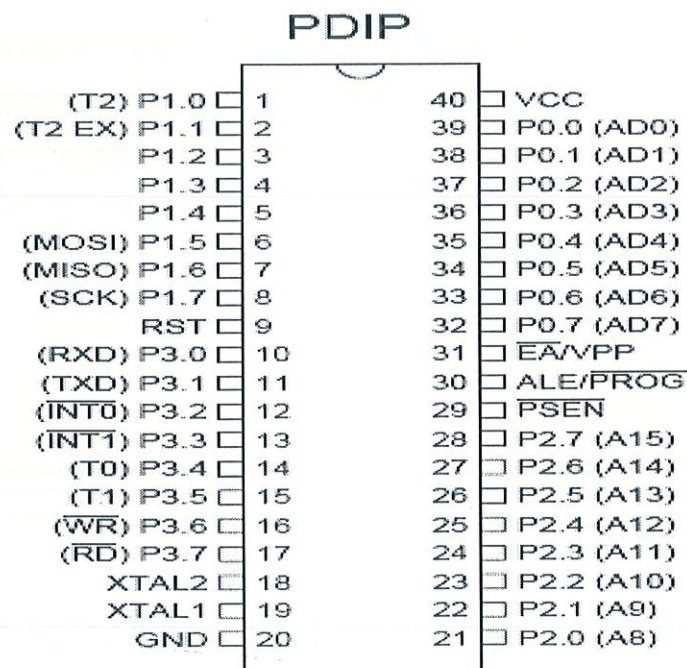


Fig 3.1: Pin Diagram of AT89S52 microcontroller

## **VCC**

Supply voltage.

## **GND**

Ground.

## **Port 0**

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high impedance inputs. Port 0 can also be configured to be the multiplexed low order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups. Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification.

## **Port 1**

Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups. In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX). Port 1 also receives the low-order address bytes during flash programming and verification.

## **Port 2**

Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL) because of the internal pull-ups. Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that uses 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data Memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register. Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.



### **Port 3**

Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pull-ups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL) because of the pull-ups. Port 3 also serves the functions of various special features of the AT89S52. Port 3 also receives some control signals for Flash programming and verification.

### **RST**

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 96 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

### **ALE/PROG**

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming. In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory. If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

### **PSEN**

Program Store Enable (PSEN) is the read strobe to external program memory. When the AT89S52 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

## **EA/VPP**

External Access Enable, EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset. EA should be strapped to VCC for internal program executions. This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming.

## **XTAL1**

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

## **XTAL2**

Output from the inverting oscillator amplifier.

## **Port 1 Pin Alternate Functions**

P1.0 T2 (external count input to Timer/Counter 2), clock-out.

P1.1 T2EX (Timer/Counter 2 capture/reload trigger and direction control)

P1.5 MOSI (used for In-System Programming)

P1.6 MISO (used for In-System Programming)

P1.7 SCK (used for In-System Programming)

## **Port 3 Pin Alternate Functions**

P3.0 RXD (serial input port)

P3.1 TXD (serial output port)

P3.2 INT0 (external interrupt 0)

P3.3 INT1 (external interrupt 1)



P3.4 T0 (timer 0 external input)

P3.5 T1 (timer 1 external input)

P3.6 WR (external data memory write strobe)

P3.7 RD (external data memory read strobe)

### 3.2 LCD

LCD stands for Liquid Crystal Display. The most commonly used LCDs found in the market today are 1 Line, 2 Line or 4 Line LCDs which have only 1 controller and support at most of 80 characters.

#### Pin Description:

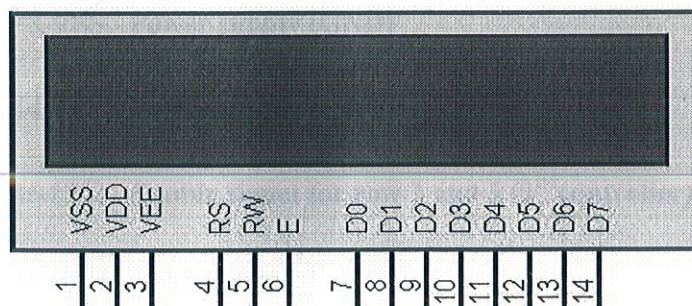


Fig 3.2: LCD pin configuration

**Table 3.1: LCD pin description**

Pin No.	Name	Description
<b>Pin no. 1</b>	<b>D7</b>	<b>Data bus line 7 (MSB)</b>
<b>Pin no. 2</b>	<b>D6</b>	<b>Data bus line 6</b>
<b>Pin no. 3</b>	<b>D5</b>	<b>Data bus line 5</b>
<b>Pin no. 4</b>	<b>D4</b>	<b>Data bus line 4</b>
<b>Pin no. 5</b>	<b>D3</b>	<b>Data bus line 3</b>
<b>Pin no. 6</b>	<b>D2</b>	<b>Data bus line 2</b>
<b>Pin no. 7</b>	<b>D1</b>	<b>Data bus line 1</b>
<b>Pin no. 8</b>	<b>D0</b>	<b>Data bus line 0 (LSB)</b>
<b>Pin no. 9</b>	<b>EN1</b>	<b>Enable signal for row 0 and 1 (1<sup>st</sup> controller)</b>
<b>Pin no. 10</b>	<b>R/W</b>	<b>0 = Write to LCD module</b> <b>1 = Read from LCD module</b>
<b>Pin no. 11</b>	<b>RS</b>	<b>0 = Instruction input</b> <b>1 = Data input</b>
<b>Pin no. 12</b>	<b>VEE</b>	<b>Contrast adjust</b>
<b>Pin no. 13</b>	<b>VSS</b>	<b>Power supply (GND)</b>
<b>Pin no. 14</b>	<b>VCC</b>	<b>Power supply (+5V)</b>
<b>Pin no. 15</b>	<b>EN2</b>	<b>Enable signal for row 2 and 3 (2<sup>nd</sup> controller)</b>
<b>Pin no. 16</b>	<b>NC</b>	<b>Not Connected</b>

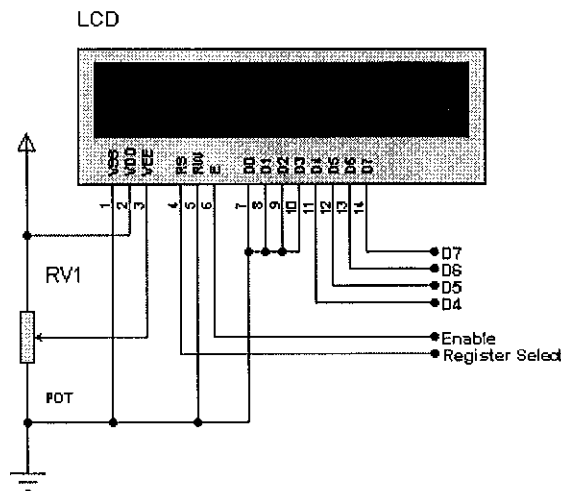
In wireless digital scoreboard we have done LCD interfacing in 4-bit mode. There are many reasons why sometime we prefer to use LCD in 4-bit mode instead of 8-bit. One basic reason is lesser number of pins are needed to interface LCD.

In 4-bit mode the data is sent in nibbles, first we send the higher nibble and then the lower nibble. To enable the 4-bit mode of LCD, we need to follow special sequence of initialization that tells the LCD controller that user has selected 4-bit mode of operation. We call this special sequence as resetting the LCD. Following is the reset sequence of LCD.

- Wait for about 20mS
- Send the first init value (0x30)
- Wait for about 10mS
- Send second init value (0x30)
- Wait for about 1mS
- Send third init value (0x30)
- Wait for 1mS
- Select bus width (0x30 - for 8-bit and 0x20 for 4-bit)
- Wait for 1mS

The busy flag will only be valid after the above reset sequence. Usually we do not use busy flag in 4-bit mode as we have to write code for reading two nibbles from the LCD. Instead we simply put a certain amount of delay usually 300 to 600uS. This delay might vary depending on the LCD we are using, as we might have a different crystal frequency on which LCD controller is running. So it actually depends on the LCD module we are using. So if we feel any problem running the LCD, simply try to increase the delay. This usually works.

### LCD connections in 4-bit Mode:



**Fig 3.3: LCD connections in 4-bit mode**

In 4-bit mode data is sent nibble by nibble, first we send higher nibble and then lower nibble. This means in both command and data sending function we need to separate the higher 4-bits and lower 4-bits.

The common steps are:

- Mask lower 4-bits
- Send to the LCD port
- Send enable signal
- Mask higher 4-bits
- Send to LCD port
- Send enable signal

### BF-Busy Flag:

Busy Flag is a status indicator flag for LCD. When we send a command or data to the LCD for processing, this flag is set (i.e. BF = 1) and as soon as the instruction is executed successfully this flag is cleared (BF = 0). This is helpful in producing an exact amount of delay for the LCD processing. To

read Busy Flag, the condition  $RS = 0$  and  $R/W = 1$  must be met and The MSB of the LCD data bus (D7) act as busy flag. When  $BF = 1$  means LCD is busy and will not accept next command or data and  $BF = 0$  means LCD is ready for the next command or data to process.

### **Instruction Register (IR) and Data Register (DR):**

There are two 8-bit registers controller Instruction and Data register. Instruction register corresponds to the register where you send commands to LCD e.g. LCD shift command, LCD clear, LCD address etc. and Data register is used for storing data which is to be displayed on LCD. When send the enable signal of the LCD is asserted, the data on the pins is latched in to the data register and data is then moved automatically to the DDRAM and hence is displayed on the LCD.

### **Commands and Instruction set:**

Only the instruction register (IR) and the data register (DR) of the LCD can be controlled by the MCU. Before starting the internal operation of the LCD, control information is temporarily stored into these registers to allow interfacing with various MCUs, which operate at different speeds, or various peripheral control devices. The internal operation of the LCD is determined by signals sent from the MCU.

### **Sending Commands to LCD:**

To send commands we simply need to select the command register. Everything is same as we have done in the initialization routine. But we will summarize the common steps and put them in a single subroutine.

Following are the steps:

- Move data to LCD port
- Select command register
- Select write operation
- Send enable signal



- Wait for LCD to process the command

### **Interfacing LCD to Microcontroller:**

The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit mode then 8 data lines plus 3 control lines i.e. total 11 lines are required. And hence 11 pins of the microcontroller are used up when we use LCD in 4-bit mode.

In case of 4-bit mode only 4 data lines are required & the 8-bit data is sent by first sending the higher nibble followed by the lower nibble. Since we are operating the LCD in 4 bit mode only 7 port pins are required.

The three control lines are **RS**, **RW** & **Enable**.

**RS:** Register Select. When RS = 0 the command register is selected. When RS=1 the data register is selected.

**RW:** Read or Write. When RW=0 write to LCD. When RW=1 read from LCD.

**Enable:** Used to Latch data into the LCD. A HIGH TO LOW edge latches the data into the LCD.

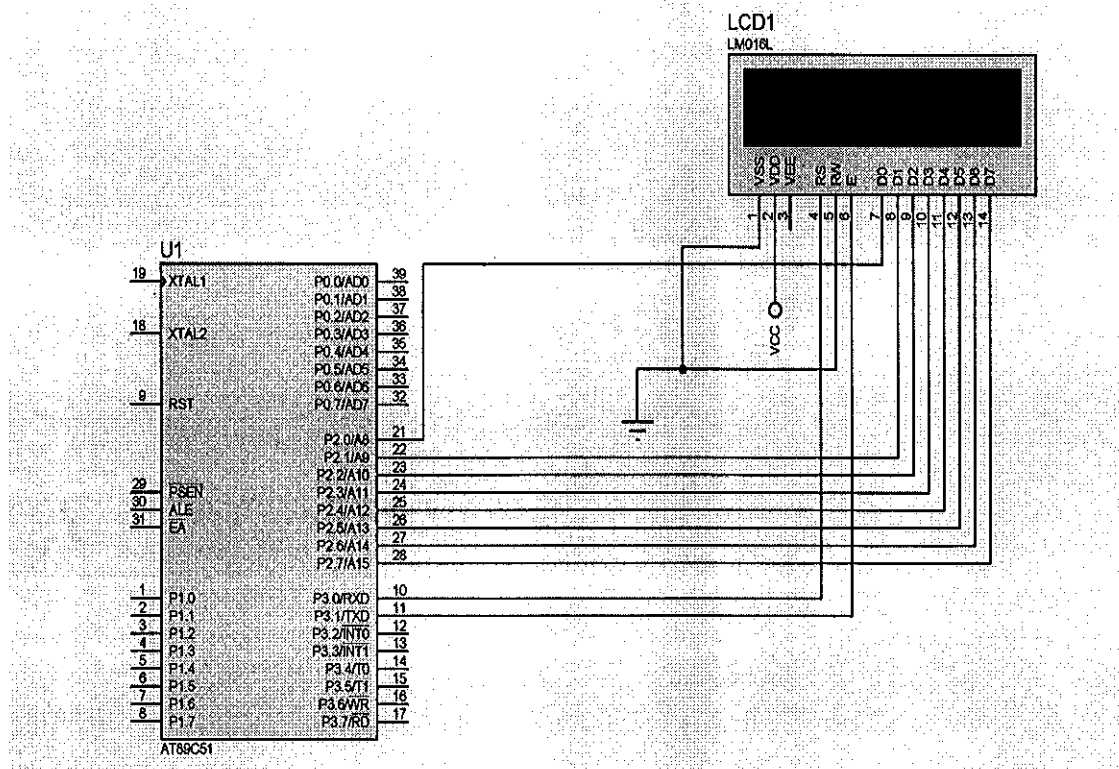


Fig 3.4: LCD interfacing to microcontroller

### 3.3 IR Module

IR module consists of two parts:-

- Transmitter
- Receiver

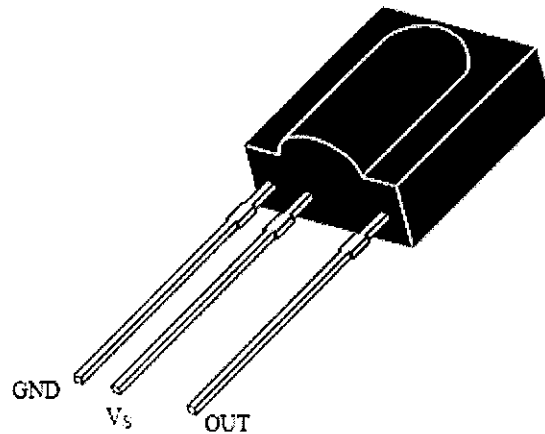
IR transmitter consists of an IR LED which is made up of AlGaAs material emitting the wavelength of 850nm. It is derived by the LED driver circuit.

IR receiver used here is TSOP 17xx series photo modules for PCM remote controls.

#### Description:

The TSOP17xx series are miniaturized receivers for infrared remote control systems. PIN diode and preamplifier are assembled on lead frame, the epoxy package is designed as IR filter. The demodulated

output signal can directly be decoded by a microprocessor. TSOP17xx is the standard IR remote control receiver series, supporting all major transmission codes.



**Fig 3.5: TSOP Pin Diagram**

**Features:**

- Photo detector and preamplifier in one package
- Internal filter for PCM frequency
- Improved shielding against electrical field disturbance
- TTL and CMOS compatibility
- Output active low
- Low power consumption
- High immunity against ambient light
- Continuous data transmission possible (up to 2400 bps)
- Suitable burst length .10 cycles/burst

### Internal Block Diagram:

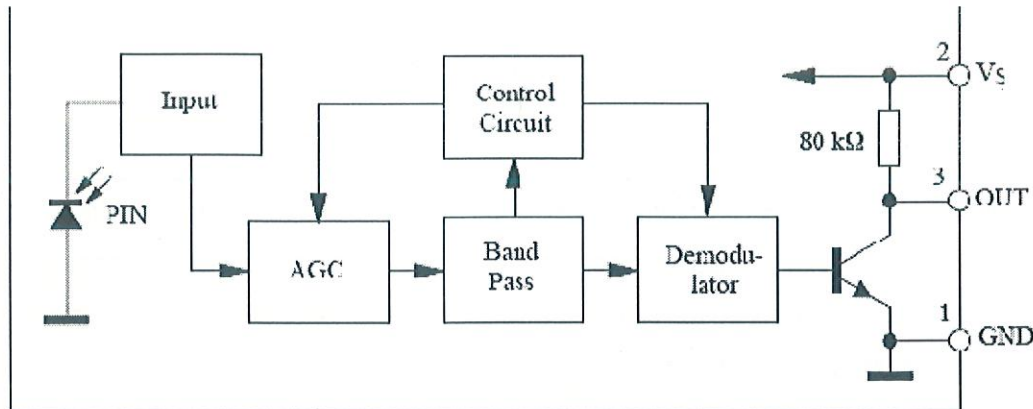


Fig 3.6: Internal Block Diagram IR module

### Suitable Data Format:

The circuit of the TSOP1738 is designed in that way that unexpected output pulses due to noise or disturbance signals are avoided. A band pass filter, an integrator stage and an automatic gain control are used to suppress such disturbances. The distinguishing mark between data signal and disturbance signal are carrier frequency, burst length and duty cycle. The data signal should fulfill the following condition:

- Carrier frequency should be close to center frequency of the band pass (e.g. 38 kHz).
- Burst length should be 10 cycles /burst or longer.
- After each burst which is between 10 cycles and 70 cycles a gap time of at least 14 cycles is necessary.
- For each burst which is longer than 1.8ms a corresponding gap time is necessary at some time in the data stream. This gap time should have at least same length as the burst.
- Up to 1400 short bursts per second can be received continuously.

### 3.4 RS-232

The serial interface described here, allows two digital systems to communicate using only a few wires. It is asynchronous, so it doesn't need a clock. The serial interface requires at least two wires:

- a transmitter wire (Tx)
- a receiver wire (Rx)

The interface is full-duplex, i.e. it can transmit and receive at the same time.

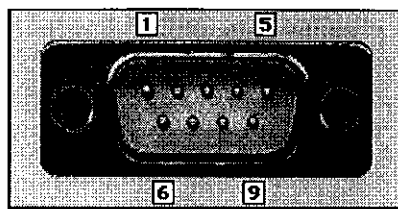
Independent channels are established for two-way (full-duplex) communications. The RS232 signals are represented by voltage levels with respect to a system common (power / logic ground). The "idle" state (MARK) has the signal level negative with respect to common, and the "active" state (SPACE) has the signal level positive with respect to common. RS232 has numerous handshaking lines (primarily used with modems), and also specifies a communications protocol.

The RS-232 is a standard developed by the Electronic Industries Association (EIA) and other interested parties, specifying the serial interface between Data Terminal Equipment (DTE) and Data Communications Equipment (DCE). The RS-232 standard includes electrical signal characteristics (voltage levels), interface mechanical characteristics (connectors), functional description of interchange circuits (the function of each electrical signal), and some recipes for common kinds of terminal-to-modem connections. The most frequently encountered revision of this standard is called RS-232C. Parts of this standard have been adopted (with various degrees of fidelity) for use in serial communications between computers and printers, modems, and other equipment. The serial ports on standard IBM-compatible personal computers follow RS-232.

#### RS-232 Cabling

Devices that use serial cables for their communication are split into two categories. These are DCE and DTE. DCE are devices such as a modem, TA adapter, plotter, and so on, while DTE is a computer or terminal. RS-232 serial ports come in two sizes, the D-Type 25-pin connector and the D-Type 9-pin connector. Both of these connectors are male on the back of the PC. Thus, you require a female connector on the device. Table 1 shows the pin connections for the 9-pin connector.





**Fig 3.7: DB9 connector**

The RS-232 interface presupposes a common ground between the DTE and DCE. This is a reasonable assumption when a short cable connects the DTE to the DCE, but with longer lines and connections between devices that may be on different electrical busses with different grounds, this may not be true.

RS232 data is bi-polar, +3 TO +12 volts indicates an "ON or 0-state (SPACE) condition" while -3 to -12 volts indicates an "OFF" 1-state (MARK) condition. Modern computer equipment ignores the negative level and accepts a zero voltage level as the "OFF" state. In fact, the "ON" state may be achieved with lesser positive potential. This means circuits powered by 5 VDC are capable of driving RS232 circuits directly; however, the overall range that the RS232 signal may be transmitted/received may be dramatically reduced. The output signal level usually swings between +12V and -12V.

**Table 3.2: DB9 pin configuration**

Function	Signal	PIN	DTE	DCE
Data	TxD	3	Output	Input
	RxD	2	Input	Output
Handshake	RTS	7	Output	Input
	CTS	8	Input	Output
	DSR	6	Input	Output
	DCD	1	Input	Output
	STR	4	Output	Input
Common	Com	5	—	—
Other	RI	9	Output	Input

An RS-232 port can supply only limited power to another device. The number of output lines, the type of interface driver IC, and the state of the output lines are important considerations.

Data is transmitted and received on pins 2 and 3 respectively. Data Set Ready (DSR) is an indication from the Data Set (i.e., the modem or DSU/CSU) that it is on. Similarly, DTR indicates to the Data Set that the DTE is on. Data Carrier Detect (DCD) indicates that a good carrier is being received from the remote modem.

Pins RTS (Request To Send - from the transmitting computer) and CTS (Clear To Send - from the Data set) are used to control. In most Asynchronous situations, RTS and CTS are constantly on throughout the communication session. However where the DTE is connected to a multipoint line, RTS is used to turn carrier on the modem on and off. On a multipoint line, it's imperative that only one station is transmitting at a time (because they share the return phone pair). When a station wants to transmit, it raises RTS. The modem turns on carrier, typically waits a few milliseconds for carrier to stabilize, and then raises CTS. The DTE transmits when it sees CTS up. When the station has finished its transmission, it drops RTS and the modem drops CTS and carrier together.

Transmit and receive leads (2 or 3) can be reversed depending on the use of the equipment - DCE Data Communications Equipment or a DTE Data Terminal Equipment.

## Max 232:

The MAX232 is an integrated circuit that converts signals from an RS-232 serial port to signals suitable for use in TTL compatible digital logic circuits. The MAX232 is a dual driver/receiver and typically converts the RX, TX, CTS and RTS signals.

The drivers provide RS-232 voltage level outputs (approx.  $\pm 7.5$  V) from a single + 5 V supply via on-chip charge pumps and external capacitors. This makes it useful for implementing RS-232 in devices that otherwise do not need any voltages outside the 0 V to + 5 V range, as power supply design does not need to be made more complicated just for driving the RS-232 in this case.

The receivers reduce RS-232 inputs (which may be as high as  $\pm 25$  V), to standard 5 V TTL levels. These receivers have a typical threshold of 1.3 V, and a typical hysteresis of 0.5 V.

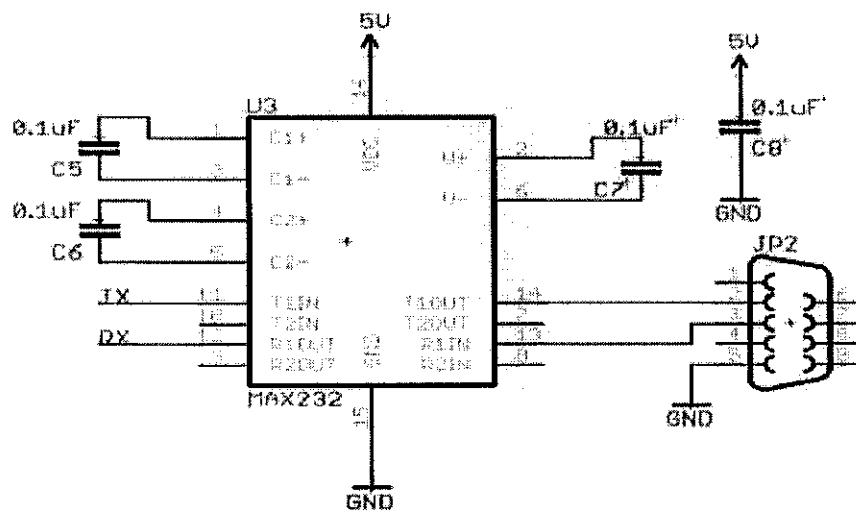


Fig 3.8: MAX 232 pin configuration

The GND of the serial cable and circuit should be same. The Tx/Rx (transmitter/receiver) pin (pin number 11 and 12) should be connected to your microcontroller. In microcontroller AT89S52 the receiver line from MAX232 should be connected to P3.0 and the Tx from MAX232 should be connected to P3.1. Also GND of all the circuits should be the same.

### 3.5 Push Buttons

Push buttons are used to enter the score into the microcontroller. We have used three push buttons to feed input data to the microcontroller. First button is used to change the cursor position and the second button is used to increment the value at that place. Third buttons is used decrement the value at that place.

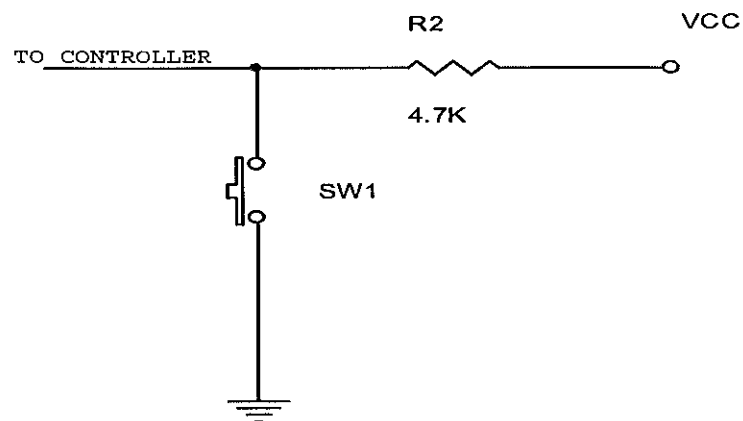
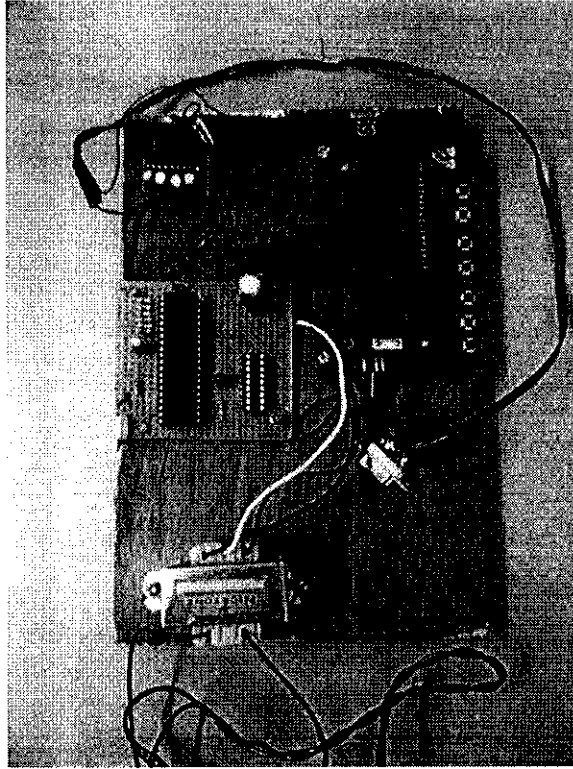


Fig 3.9: Connection of a push button





**Fig 3.10: Project snapshot**

## CHAPTER 4

### TESTING & RESULTS OF WIRELESS DIGITAL SCORE BOARD

The Wireless Digital Score Board project has been tested using software Proteus.

A screen shot shows it as follows:

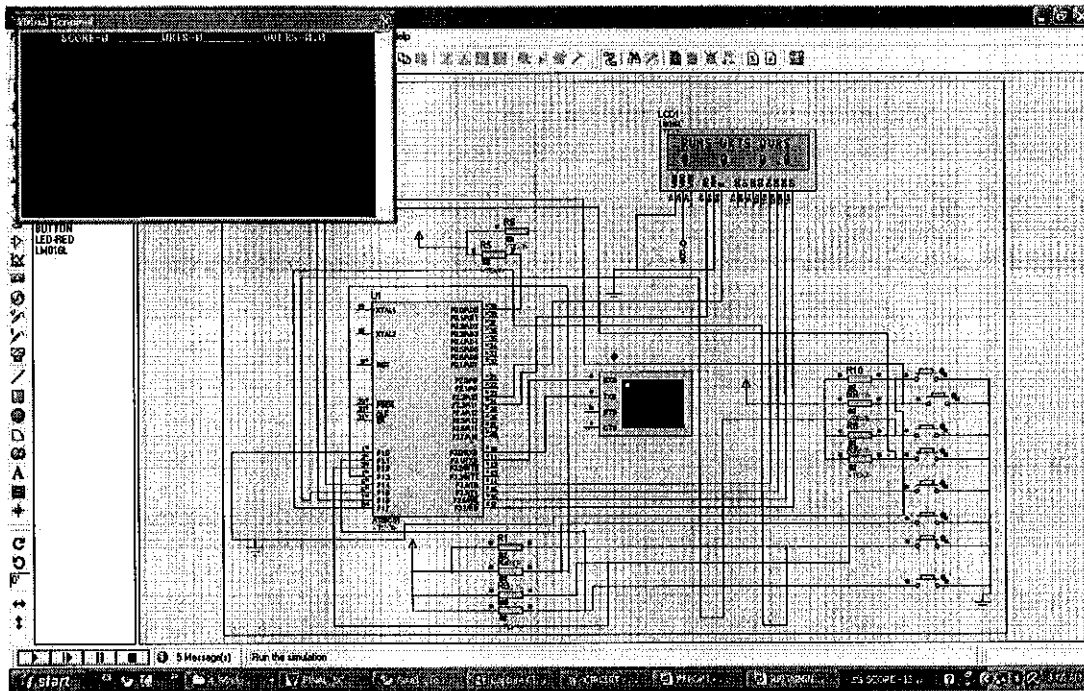


Fig 4.1: Screen shot of testing the project using proteus

During testing of the project, we faced some difficulty in serial communication with PC. First the data was transmitted at 4800 baud rate but the data received was not appropriate, then we used baud rate of 9600, this worked correctly.

## **CONCLUSION**

This project has made us familiar with actual situations being faced in the projects. We know that it was difficult task to establish at various points but we managed each and every thing collectively. Now we are familiar with IR transmitter and receiver, microcontroller programming, and we have worked on many components that have been used in this project which otherwise are difficult to be studied from the text books.

Troubleshooting which is an inherent part of any engineering project has widened our thinking capacity. Main problem that we found was in software development. Although we also faced problem in hardware, when some of components like IC of MAX 232 were not working, but guidance of many faculty members made this whole experience extremely wonderful.

## **FUTURE ASPECTS**

Our project 'Wireless Digital Scoreboard' is just a kind of practice work and although we gave our hundred percent but still many more advancements can be realized in future. Some of which are:

1. This project can be implemented using RF technology to increase its range.
2. Project can be used with GSM technology to display score.



## REFERENCES

- [1] Wireless Infrared Communications Joseph M. Kahn, Member, IEEE, and John R. Barry
- [2] T. S. Chu and M. J. Gans, "High speed infrared local wireless ommunication," *IEEE Commun. Mag.*, vol. 25, no. 8, pp. 4–10, Aug. 1987.
- [3] H. Hashemi, G. Yung, M. Kavehrad, R. Behbahani, and P. A. Galko, "Indoor propagation measurements at infrared frequencies for wireless local area networks applications," *IEEE Trans. Vehic. Technol.*, vol. 43, pp. 562–576.
- [4] Muhammad Ali Mazidi, Janice Gillispie Mazidi, Rolin D. Mc Kinlay , *The 8051 Microcontroller & Embedded Systems*, Pearson Education Inc. 2nd Edition, 2008.
- [5] Kenneth J Ayala, *The 8051 Microcontroller Architecture, Programming & Applications*, Penram International, 2nd Edition, 1996.
- [6] [www.8051projects.net/microcontroller\\_tutorials](http://www.8051projects.net/microcontroller_tutorials)
- [7] [www.8051projects.net/lcd-interfacing](http://www.8051projects.net/lcd-interfacing)
- [8] [http://www.atmel.com/dyn/resources/prod\\_documents/doc1919.pdf](http://www.atmel.com/dyn/resources/prod_documents/doc1919.pdf)
- [9] [http://www.ikalogic.com/tut\\_8051\\_2.php](http://www.ikalogic.com/tut_8051_2.php)
- [10] <http://people.csail.mit.edu/rivest/Rivest-rc5rev.pdf>

**APPENDIX A****List of Components**

<b>S No</b>	<b>Components</b>	<b>Specification</b>	<b>Quantity</b>
1	Microcontroller	AT89s52	1
2	Voltage Regulator	KA7805A	1
3	Transformer	12-0-12	1
4	Diodes	IN4007	2
5	Capacitors	1000uF	1
		33pF	2
		10uF	5
6	Resistors	10k	1
		4.7k	7
7	LED	Red	1
8	IR Receiver	TSOP 1738	1
9	Switches	SPST Push	7
10	Crystal	11.0592MHz	1
11	LCD	2*16 display	1
12	RS-232	DB-9	1
13	Driver/Receivers	Max-232	1

**APPENDIX A****List of Components**

<b>S No</b>	<b>Components</b>	<b>Specification</b>	<b>Quantity</b>
1	Microcontroller	AT89s52	1
2	Voltage Regulator	KA7805A	1
3	Transformer	12-0-12	1
4	Diodes	IN4007	2
5	Capacitors	1000uF	1
		33pF	2
		10uF	5
6	Resistors	10k	1
		4.7k	7
7	LED	Red	1
8	IR Receiver	TSOP 1738	1
9	Switches	SPST Push	7
10	Crystal	11.0592MHz	1
11	LCD	2*16 display	1
12	RS-232	DB-9	1
13	Driver/Receivers	Max-232	1

## Features

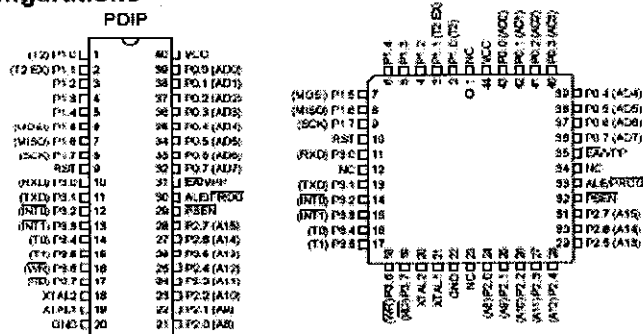
- Compatible with MCS-51® Products
- 8K Bytes of In-System Programmable (ISP) Flash Memory
  - Endurance: 1000 Write/Erase Cycles
- 4.0V to 5.5V Operating Range
- Fully Static Operation: 0 Hz to 33 MHz
- Three-Level Program Memory Lock
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Eight Interrupt Sources
- Full Duplex UART Serial Channel
- Low-power Idle and Power-down Modes
- Interrupt Recovery from Power-down Mode
- Watchdog Timer
- Dual Data Pointer
- Power-off Flag

## Description

The AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K bytes of in-system programmable Flash memory. The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with in-system programmable Flash on a monolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89S52 provides the following standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S52 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

## Pin Configurations



Rev. 19/9A-07/01

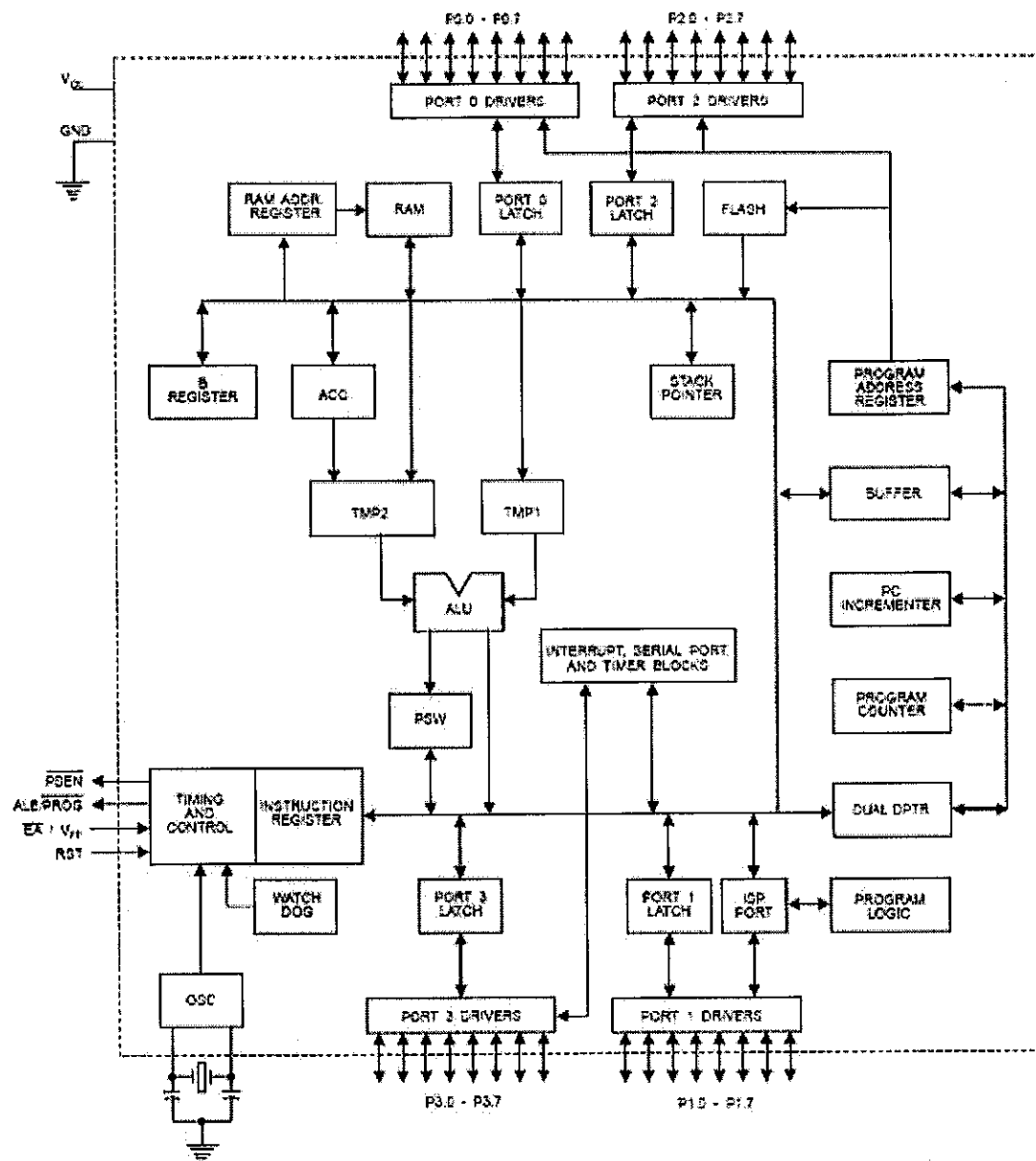


8-bit  
Microcontroller  
with 8K Bytes  
In-System  
Programmable  
Flash

AT89S52







**TSOP17..**

Vishay Telefunken

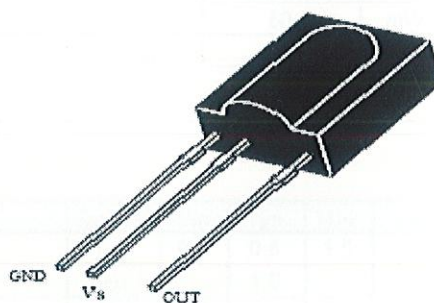
**Photo Modules for PCM Remote Control Systems****Available types for different carrier frequencies**

Type	f <sub>0</sub>	Type	f <sub>0</sub>
TSOP1730	30 kHz	TSOP1733	33 kHz
TSOP1736	36 kHz	TSOP1737	36.7 kHz
TSOP1738	38 kHz	TSOP1740	40 kHz
TSOP1758	58 kHz		

**Description**

The TSOP17.. — series are miniaturized receivers for infrared remote control systems. PIN diode and preamplifier are assembled on lead frame, the epoxy package is designed as IR filter.

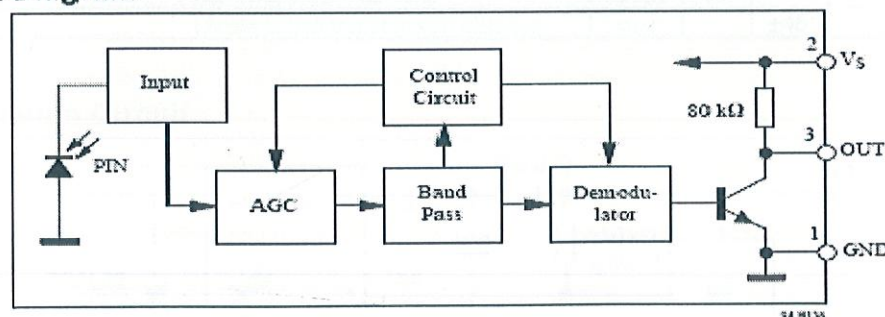
The demodulated output signal can directly be decoded by a microprocessor. TSOP17.. is the standard IR remote control receiver series, supporting all major transmission codes.



94 9291

**Features**

- Photo detector and preamplifier in one package
- Internal filter for PCM frequency
- Improved shielding against electrical field disturbance
- TTL and CMOS compatibility
- Output active low
- Low power consumption
- High immunity against ambient light
- Continuous data transmission possible (up to 2400 bps)
- Suitable burst length ≥10 cycles/burst

**Block Diagram**

94 8155

# TSOP17..

Vishay Telefunken



## Absolute Maximum Ratings

$T_{amb} = 25^{\circ}\text{C}$

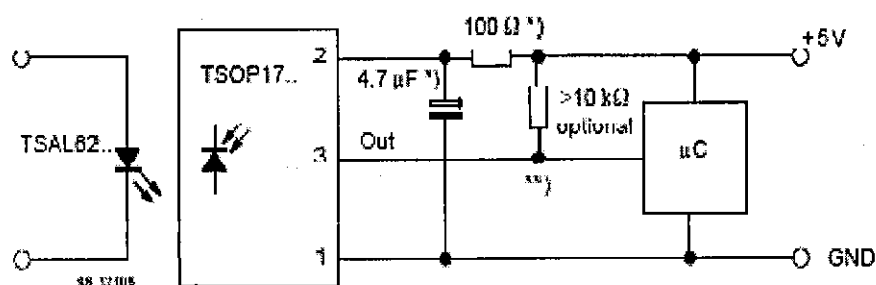
Parameter	Test Conditions	Symbol	Value	Unit
Supply Voltage	(Pin 2)	$V_S$	-0.3...6.0	V
Supply Current	(Pin 2)	$I_S$	5	mA
Output Voltage	(Pin 3)	$V_O$	-0.3...6.0	V
Output Current	(Pin 3)	$I_O$	5	mA
Junction Temperature		$T_J$	100	$^{\circ}\text{C}$
Storage Temperature Range		$T_{stg}$	-25...+85	$^{\circ}\text{C}$
Operating Temperature Range		$T_{amb}$	-25...+85	$^{\circ}\text{C}$
Power Consumption	( $T_{amb} \leq 85^{\circ}\text{C}$ )	$P_{tot}$	60	mW
Soldering Temperature	$t \leq 10\text{ s}$ , 1 mm from case	$T_{sd}$	260	$^{\circ}\text{C}$

## Basic Characteristics

$T_{amb} = 25^{\circ}\text{C}$

Parameter	Test Conditions	Symbol	Min	Typ	Max	Unit
Supply Current (Pin 2)	$V_S = 5\text{ V}$ , $E_v = 0$	$I_{SD}$	0.4	0.6	1.5	mA
	$V_S = 5\text{ V}$ , $E_v = 40\text{ klx}$ , sunlight	$I_{SH}$		1.0		mA
Supply Voltage (Pin 2)		$V_S$	4.5		5.5	V
Transmission Distance	$E_v = 0$ , test signal see fig. 7, IR diode TSAL620D, $I_F = 400\text{ mA}$	$d$		3.5		m
Output Voltage Low (Pin 3)	$I_{OHL} = 0.6\text{ mA}$ , $E_p = 0.7\text{ mW/m}^2$ , $f = f_0$ , $t_p/T = 0.4$	$V_{OHL}$			250	mV
Irradiance (30 – 40 kHz)	Pulse width tolerance: $t_{pl} - 5/f_0 < t_{po} < t_{pl} + 8/f_0$ , test signal (see fig. 7)	$E_{e\text{ min}}$		0.35	0.5	$\text{mW/m}^2$
Irradiance (56 kHz)	Pulse width tolerance: $t_{pl} - 5/f_0 < t_{po} < t_{pl} + 8/f_0$ , test signal (see fig. 7)	$E_{e\text{ min}}$		0.4	0.6	$\text{mW/m}^2$
Irradiance	$t_{pl} - 5/f_0 < t_{po} < t_{pl} + 8/f_0$	$E_{e\text{ max}}$	30			$\text{W/m}^2$
Directivity	Angle of half transmission distance	$\phi_{1/2}$		$\pm 45$		deg

## Application Circuit



\*) recommended to suppress power supply disturbances

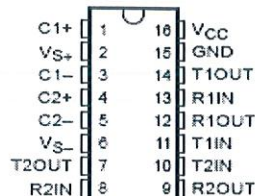
\*\*) The output voltage should not be hold continuously at a voltage below 3.3V by the external circuit.

# MAX232, MAX232I DUAL EIA-232 DRIVERS/RECEIVERS

SLLS047I – FEBRUARY 1999 – REVISED OCTOBER 2002

- Meet or Exceed TIA/EIA-232-F and ITU Recommendation V.28
- Operate With Single 5-V Power Supply
- Operate Up to 120 kbit/s
- Two Drivers and Two Receivers
- $\pm 30$ -V Input Levels
- Low Supply Current . . . 8 mA Typical
- Designed to be Interchangeable With Maxim MAX232
- ESD Protection Exceeds JESD 22 – 2000-V Human-Body Model (A114-A)
- Applications
  - TIA/EIA-232-F
  - Battery-Powered Systems
  - Terminals
  - Modems
  - Computers

MAX232 . . . D, DW, N, OR NS PACKAGE  
MAX232I . . . D, DW, OR N PACKAGE  
(TOP VIEW)



## description/ordering information

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply EIA-232 voltage levels from a single 5-V supply. Each receiver converts EIA-232 inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V and a typical hysteresis of 0.5 V, and can accept  $\pm 30$ -V inputs. Each driver converts TTL/CMOS input levels into EIA-232 levels. The driver, receiver, and voltage-generator functions are available as cells in the Texas Instruments LinASIC™ library.

## ORDERING INFORMATION

TA	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP (N)	Tube	MAX232N	MAX232N
		Tube	MAX232D	MAX232
	SOIC (D)	Tape and reel	MAX232DR	MAX232
		Tube	MAX232DW	MAX232
	SOIC (DW)	Tape and reel	MAX232DWR	MAX232
		Tube	MAX232NSR	MAX232
-40°C to 85°C	PDIP (N)	Tube	MAX232IN	MAX232IN
		Tube	MAX232ID	MAX232I
	SOIC (D)	Tape and reel	MAX232IDR	MAX232I
		Tube	MAX232IDW	MAX232I
	SOIC (DW)	Tape and reel	MAX232IDWR	MAX232I
		Tube	MAX232IDWR	MAX232I

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at [www.ti.com/sc/package](http://www.ti.com/sc/package).



Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

LinASIC is a trademark of Texas Instruments.

PRODUCTION DATA Information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.



POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

Copyright © 2002, Texas Instruments Incorporated