RFID BASED RAILWAY PLATFORM TO IDENTIFY EXACT POSITION OF COACH

BY

D.RAMAYAMINI(Y11EC8I8)

D.V.RAMAVATHI(Y11EC819)

G.RAMYA SREE(L12EC929)

D.RAVIKRISHNA TEJESWI(Y11EC821)

Under the guidance of

Sri P.V.KRISHNA KANTH M.S

Department of Electronics & Communication Engineering

R.V.R. & J.C. College of Engineering, Chandramoulipuram

(Affiliated to Acharya Nagarjuna University)

GUNTUR – 522 019, Andhra Pradesh, INDIA[2014-15]

A Project Report on

RFID based railway platform to identify exact position of coach

Submitted in partial fulfillment for award of

Bachelor of Technology

Degree

Electronics & Communication Engineering

By

D.RAMAYAMINI (Y11E818)

D.V.RAMAVATHI(Y11EC819)

G.RAMYA SREE(L12EC929)

D.RAVI KRISHNA TEJASWI(Y11EC821)

Under the guidance of

Sri. P. V. KRISHNA KANTH M.S

Assistant Professor

R.V.R. & J.C. College of Engineering, Chandramoulipuram

(Affiliated to Acharya Nagarjuna University)

GUNTUR – 522 019, Andhra Pradesh, INDIA

[2014 - 15]

Department of

Electronics & Communication Engineering

CERTIFICATE

This is to certify that the project report entitled “RFID based railway platform to identify exact position of coach” that is being submitted by D. Ramayamini (Y11ec818), D.V.Ramavathi (Y11ec819), G.Ramya Sree (L12ec929), D. Ravikrishna Tejeswi(Y11ec821) in partial fulfillment for the award of the Degree of Bachelor of Technology in Electronics and Communication Engineering to the Acharya Nagarjuna University is a record of bonafide work carried out by him under my guidance and supervision. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Date:

Signature of Guide	**Signature of HOD**
Sri P.V.KRISHNA KANTH M.S	**Dr.T.RANGABABU**Ph.D
Assistant professor	**Professor&HOD**

ACKNOWLEDGEMENT

It gives us an immense pleasure to acknowledge all those who helped us throughout in making this project a great success.

With profound gratitude we thank our principal **Dr. A. SUDHAKAR** for his timely suggestions which helped us to complete this project work successfully.

We found great pleasure in expressing our gratitude to our H.O.D **Dr. T. RANGA BABU** for his valuable and inspiring guidance, comments, suggestions and encouragement throughout the course of the project.

We express a great pleasure to acknowledge my profound sense of gratitude to my project guide **P.V KRISHNA KANTH** for her valuable and inspiring guidance, comments, suggestions and encouragement throughout the course of the project.

We are thankful to both teaching and non teaching staff members of **ELECTRONICS AND COMMUNICATION ENGINEERING** for their kind co-operation and all sorts of help bringing out this project successfully.

.

D.RAMAYAMINI(Y11EC818)

D.V.RAMA VATHI(Y11EC819)

G.RAMYA SREE(L12EC929)

TABLE OF CONTENTS

ABSTRACT

Radio frequency identification (RFID) technology already plays a major role in many areas, current solutions ,however, are designed only identify objects in range, which is sufficient for most of the envisioned shop applications. for other uses, however not only the identification but also the exact position are orientation of objects would be interesting if not necessary. A good example is to display exact position of coach of a train on the platform. when coach comes in front of RFID reader, the reader read the information about the coach from the unique RFID and record into the data base..

Basically it has three components

1. Antenna
2. transceiver
3. transponder

So these are components that perform the whole task. when signal is sent by antenna it return back from the subject and the signal detected and by this the reorganiazation process performed.

LIST OF FIGURES PAGE NO

LIST OF TABLES PAGE NO

ABBRIVATIONS

Name	Function
A	Accumulator
B	Arithmetic
DPH	Addressing external memory
DPL	Addressing external memory
IE	Interrupt enable control
IP	Interrupt priority
P0	Input/Output port latch
P1	Input/Output port latch
P2	Input/Output port latch
P3	Input/Output port latch
PCON	Power control
PSW	Program status word
SCON	Serial port control
SBUF	Serial port data buffer
SP	Stack pointer
TMOD	Timer/counter mode control
TCON	Timer/counter control
TL0	Timer 0 low byte
TH0	Timer 0 high byte
TL1	Timer 1 low byte
TH1	Timer 1 high byte

Special Function Registers

CHAPTER 1

INTRODUCTION

1. INTRODUCTION

Radio Frequency Identification (RFID) technology already plays a major role in many areas. Radio-frequency identification (RFID) is a technology to record the presence of an object using radio signals A basic RFID system consists of three components:

a) An antenna or coil

b) A transceiver (with decoder)

c) A transponder (RF tag)

There are many different types of RFID systems out in the market. They are categorized according to their frequency ranges. Some of the most commonly used RFID kits are as follows:

1) Low-frequency (30 KHz to 500 KHz)

2) Mid-Frequency (900KHz to 1500MHz)

3) High Frequency (2.4GHz to 2.5GHz)

Current solutions, however, are designed to only identify objects in range, which is sufficient for most of the envisioned shop applications. For other uses, however, not only the identification, but also the exact position and orientation of objects would be interesting, if not necessary. A good example is to display the position of coach of a train on the platform

An embedded system is a special-purpose computer system designed to perform a dedicated function. Unlike a general-purpose computer, such as a personal computer an embedded system performs one or a few pre-defined tasks, usually with very specific requirements. Since the system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product. Embedded system comprises of both hardware and software. . Embedded system is fast growing technology in various fields like industrial automation, home appliances,

automobiles, aeronautics etc. However, the definition of “embedded system” is fluid and difficult to pin down, as it constantly evolves with advances in technology and dramatic decreases in the cost of implementing various hardware and software components.

CHAPTER 2

EXACT POSITIONING OF RAILWAY COACH USING RFID

2. EXACT POSITIONING OF RAILWAY COACH USING RFID

The present project exact positioning of railway coach using RFID automate the display system of coach position and also time saving .these are designed to identify the objects in the range, which is sufficient for most of the envisioned shop applications.

2.1 BOCK DIAGRAM

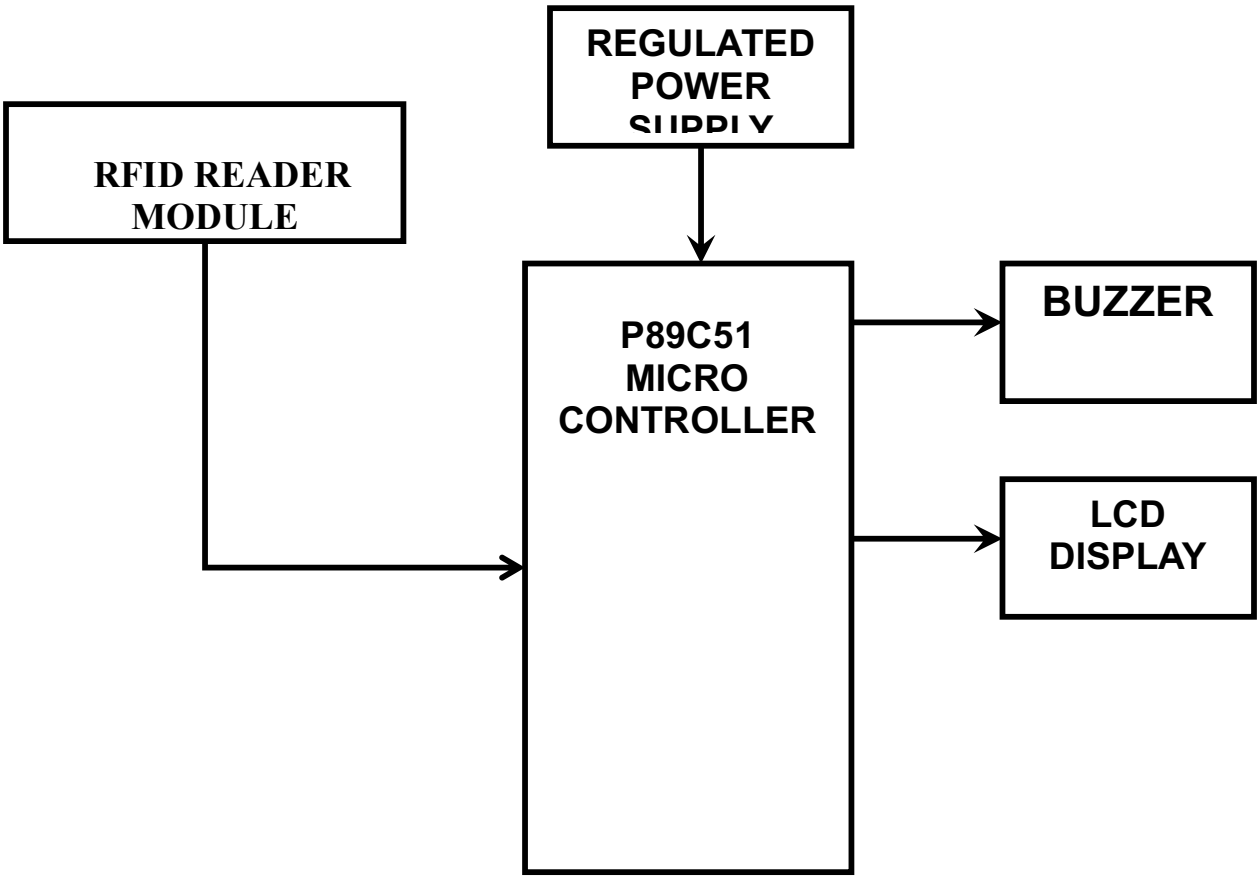


Fig 2.1 BLOCK DIAGRAM OF THE PROJECT

2.2 BLOCK DIAGRAM EXPLANATION

POWER SUPPLY

The Power Supply is a Primary requirement for the project work. The required DC power supply for the base unit as well as for the recharging unit is derived from the mains line. For this purpose center tapped secondary of 12V-012V transformer is used. From this transformer we getting 5V power supply. In this +5V output is a regulated output and it is designed using 7805 positive voltage regulator. This is a 3 Pin voltage regulator, can deliver current up to 800 milliamps.

Rectification is a process of rendering an alternating current or voltage into a uni directional one. The component used for rectification is called 'Rectifier'. A rectifier permits current to flow only during positive half cycles of the applied AC voltage. Thus, pulsating DC is obtained to obtain smooth DC power additional filter circuits required.

MICROCONTROLLER

In this project the 8052 family of microcontroller is used to perform the operation of railway coach identification. All the operations that must be embedded in to the microcontroller by writing the program as per the requirements. Here KEIL IDE is what the software that is used for this programming.

BUZZER

. It most commonly consists of a number of switches or sensors connected to a control unit that determines if and which button was pushed or a preset time has lapsed, and usually illuminates a light on the appropriate button or control panel, and sounds a warning in the form of a continuous or intermittent buzzing or beeping sound.

LCD DISPLAY

An image in an LCD is formed by applying an electric field to alter the chemical properties of each LCC (Liquid Crystal Cell) in the display in order to change a pixel's light absorption properties. These LCC's modify the image produced by the backlight into the screen output requested by the controller. Through the end output may be in color, the LCC's are monochrome, and the color is added later through a filtering process.

RFID READER MODULE

RFID reader module plays a major role to identify the railway coach. when coach comes in front of RFID reader, the reader read the information about the coach from the unique RFID and record into the data base. when signal is sent by antenna it return back from the subject and the signal detected and by this the reorganization process performed.

2.3 SCHEMATIC DIAGRAM

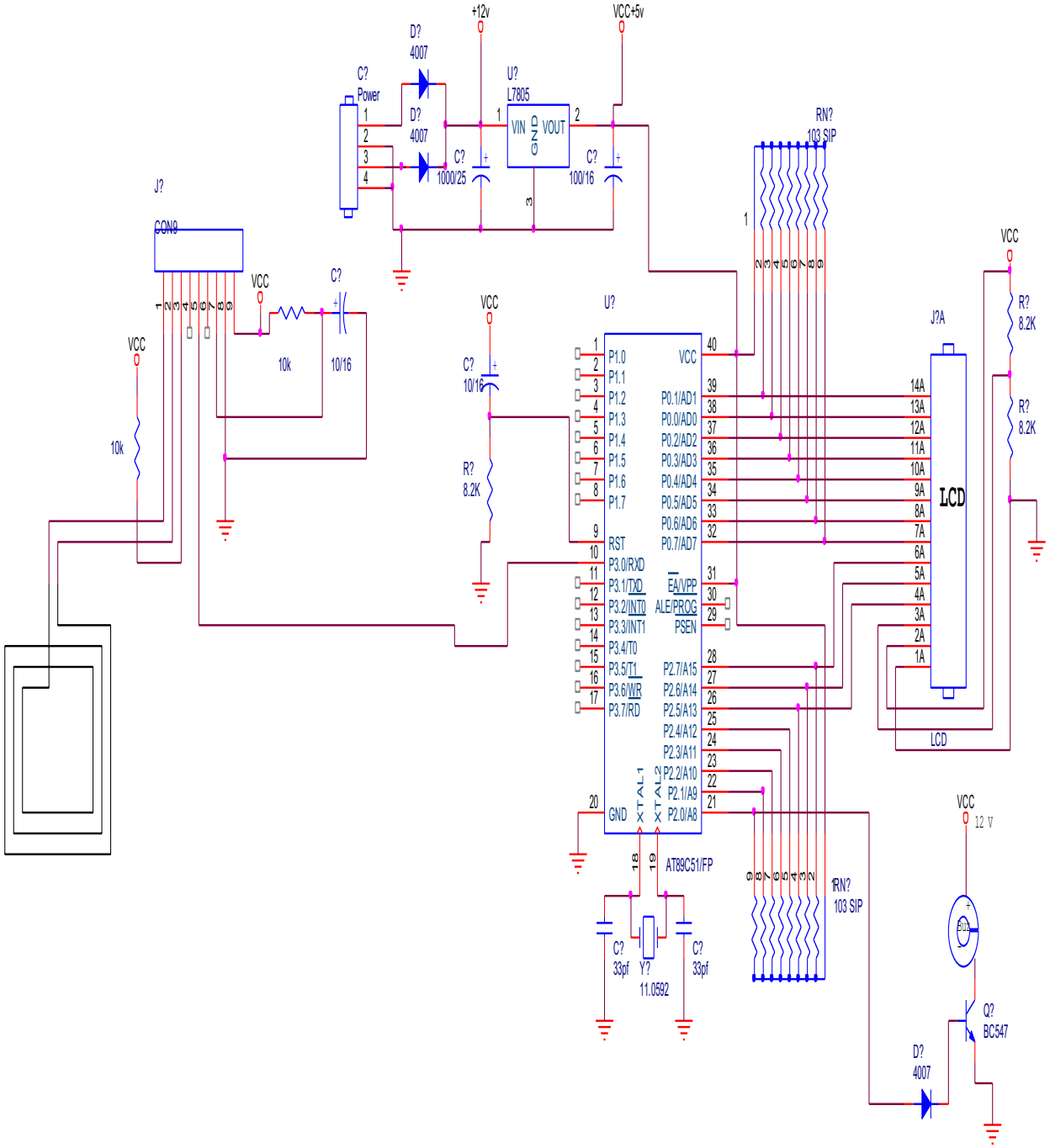


Fig 2.2 schematic diagram

2.4 SCHEMATIC DESCRIPTION

- A 5V supply is given to the 40^{th} pin of the controller. Which the voltage required by the controller.

- Here we use 16*2 LCD display. port 0(8 bits), port 2.5,port 2.6,port 2.7of microcontroller AT89C52 are interfaced to this display.

- P2.0 of microcontroller is used for buzzer and driver circuit.

- P3.0 of microcontroller is used to receive the data.

CHAPTER 3

HARDWARE COMPONENTS

HARDWARE COMPONENTS

3.1 PCB MAKING

One of the most discouraging things about making a hardware project is building the printed circuit board-PCB.it is sometimes possible to use strip board or some other pre-fabricated board but more often than not the circuit complexity and performance requires a proper PCB to be made .The good news is that due to improvements in printing and processing technologies it is now relatively easy to make inexpensive high quality PCB's at home.

Making PCB's requires the use of Ferric Chloride(FeCi3) which is corrosive so avoid skin and eye contact .Remember safety-first so, use glasses, gloves and protective overalls .Ferric Chloride is also very good at distorting cloths weeks after you think you have washed it off. If you do get any on your skin then wash it off immediately with lots of water and soap.

3.1.1 THE ARTWORK

The first stage is to transfer the circuit layout from the PC to the special Press-n-Peel film. Put the film in the laser printer so that the print will appear on it. This will produce a contact print where the black image will end up as copper on the final PCB. Now to transfer the artwork to the Copper board by following the instructions with the Press-n-Peel film:

- Clean the copper board very well with the PCB cleaning rubber.
- Heat the cloths iron to 300 deg F.
- Hole the film with the print in contact to the copper and smoothly iron the film down until the print appears black through the film (about 1min).

- Allow 5min to cool down (or speed up this with water) then peel the film off.

This should produce a clean black print on to the copper. If you let the film move or overheat then you will find that the tracks and writing will be smeared and out of focus also the film may be wrinkled up. If you don't use enough heat or heat unevenly then the film may not stick or to be dark enough. In either case clean off the PCB and try again, you should get it right after a couple of goes.

3.1.2 THE ETCHING

- Etching the PCB is to remove the unwanted copper.
- Dilute the concentrated Ferric Chloride fluid with water (1:1) and pour into the one liter glass jar.
- Put the PCB copper side up on the top tray and pour all Ferric Chloride on top.
- Gently rock the top tray to keep the etch fluid moving avoiding spillage.
- After about 15min all of the unwanted copper disappears.
- Remove the board and drop it into a bucket of cold water to clean off.

3.1.3 DRILLING

Drilling with 0.8mm drill bits can be bit tricky as it is easy to break the drill bits. Always hold the drill straight and do not bend it when the hole has started .Using a 0.8mm PCB drill bit, drill out all of the component holes that are required. So, now the PCB is finished and it is ready to solder.

3.1.4 ADVANTAGES OF PCB

- Reducing wiring errors and Decreases assembly cost.

- Typically consume less space than traditionally build circuits.

3.2 MICROCONTROLLER

A microcontroller consist of a powerful CPU tightly coupled with memory(RAM,ROM or EPROM), various I/O features such as serial ports, parallel ports, timer/counter, interrupt controller, data acquisition interfaces-analog to digital converter(ADC), digital to analog converter(DAC), everything integrated on to a single silicon chip.

It does mean that any microcontroller should have all the above said features on chip, depending on the need and area of application for which it is designed, the on chip features present in it may or may not include all the individual section said above.

Any microcomputer system requires memory to store a sequence of instructions making up a program parallel port or serial port for communicating with an external system, timer/counter for control purpose like generating time delays, baud rate for the serial port, a part from the controlling unit called central processing unit.

ADVANTAGES OF MICROCONTROLLER

If a system is developed with microprocessor, the designer has to go for external memory such as RAM, ROM, EPROM and peripherals and hence the size of the PCB will be large enough to whole all required peripherals. But, the microcontroller has got all these peripheral facilities on a single chip so development of a similar system with a microcontroller reduces PCB size and cost of the design, one of the major differences between a microcontroller and a micro processer is that a controller often deals with bits, not bites as in the real world application. For example switch contacts can only be open or close , indicators should be lit or dark and motors can be either turned on or off.

3.2.1 INTRODUCTION TO 8051 MICROCONTROLLER

Intel corporation introduced an 8 bit microcontroller. This microcontroller had 128 bytes of RAM, 8K bytes of on chip ROM, Three timers, 8 interrupt sources, programmable serial channel, At that time it was also referred to as a "system on chip".

The 8052 is an eight bit microcontroller, meaning that the CPU can work on only eight bits of data at a time. Data larger than 8 bits has to be broken in to 8 pieces to be processed by the CPU.

FEATURES

- Three-level Program Memory Lock.
- 256 x 8-Bit Internal RAM.
- 32 Programmable I/O Lines.
- Three 16-bit Timer/Counters.
- Eight Interrupt Sources.
- Programmable Serial Channel.
- Low Power Idle and Power Down Modes
- Compatible with MCS-51 Products.
- 8K Bytes of In-System Reprogrammable Flash Memory.
- Endurance: 1,000 Write/Erase Cycles.
- Fully Static Operation: 0 Hz to 24 MHz

3.3.2 ARCHITECTURE OF ATMEL 89C52 MICROCONTROLLER

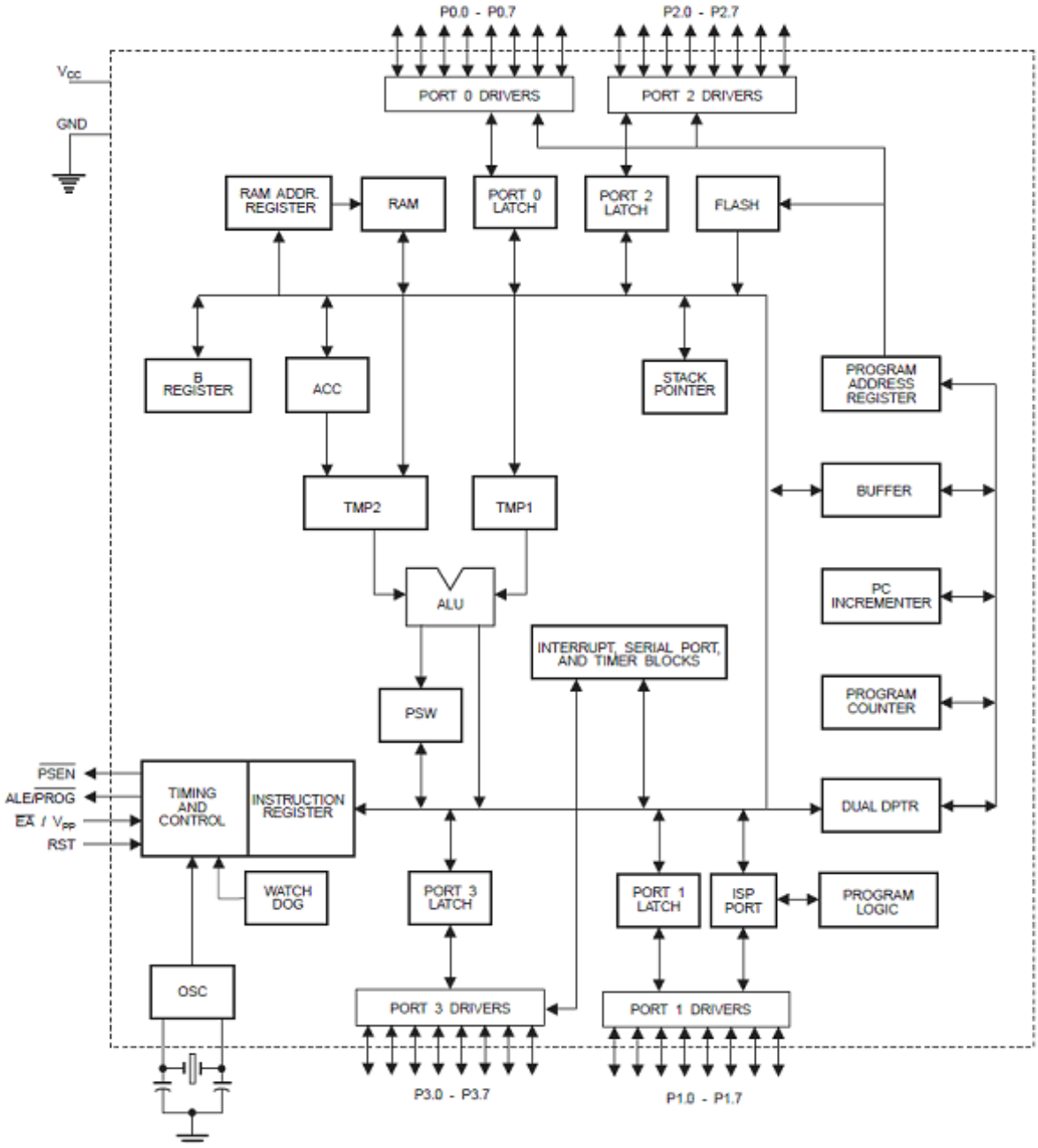


Fig 3.1 ARCHITECTURE OF MICROCONTROLLER AT89C52 IC

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 may 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. External pull-ups are required during program verification

PORT 1

Port 1 is an 8-bit bi-directional 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. Port 1 also receives the low-order address bytes during Flash programming and program verification.

Table 3.1 Alternate functions of PORT 1

Port 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 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 use 16-bit addresses (MOVX A,@DPTR). In this application it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that uses 8-bit addresses (MOVX A, @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 3pins 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 AT89C52, as shown in the following table. Port 3 also receives some control signals for Flash programming and verification.

Table 3.2 Alternate functions of PORT 3

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

RST

RST means RESET; 89C52 uses an active high reset pin. It must go high for two machine cycles. The simple RC circuit used here will supply voltage (VCC) to reset pin until capacitance begins to charge. At a threshold of about 2.5V, reset input reaches a low level and system begin to run.

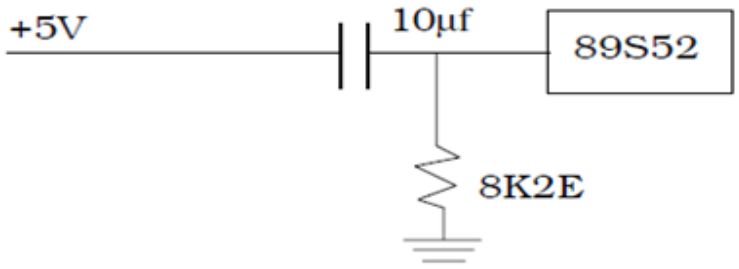


Fig: 3.2 Reset Connection

ALE/PROG

Address Latch Enable 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.

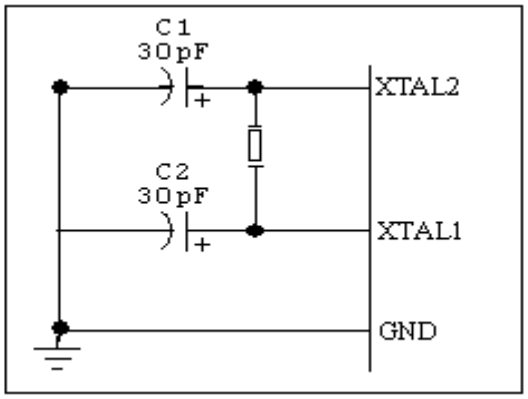
PSEN

Program Store Enable is the read strobe to external program memory. When the AT89C52 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 OOOOH 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, for parts that require 12-volt Vpp.

.



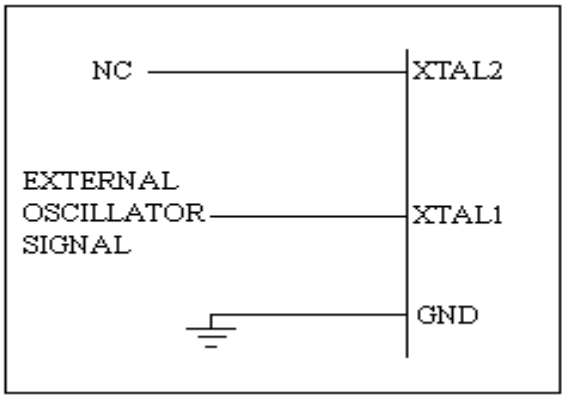


Fig 3.3 XTAL Connection to 8052

Fig 3.4 XTAL Connection to an External Clock

XTAL1 and XTAL2

The 8051 has an on-chip oscillator but requires an external clock to run it. Most often a quartz crystal oscillator connected to inputs XTAL1(pin 19) and XTAL2 (pin 18). The quartz crystal oscillator connected to XTAL1 and XTAL2 also needs two capacitors of 30 pF value. One side of each capacitor is connected to the ground as shown in Figure a.

It must be noted that there are various speeds of the 8051 family. Speeds refers to the maximum oscillator frequency connected to XTAL. For example, a 12-MHz chip must be connected to a crystal with 12 MHz frequency or less. Likewise, a 20-MHz microcontroller requires a crystal frequency of no more than 20 MHZ. When the 8051 is connected to a crystal oscillator and is powered up, we can observe the frequency on the XTAL2 pin using the oscilloscope.

If you decide to use a frequency source other than a crystal oscillator, such as a TTL oscillator, it will be connected to XTAL1; XTAL2 is left unconnected, as shown in figure(b)

T2: External count input to Timer/Counter 2, Clock out.

T2EX: Counter 2 capture/reload trigger & direction control.

3.2.3 THE ON-CHIP OSCILLATORS

Pins XTAL1 and XTAL2 are provided for connecting a resonant network to form an oscillator. The crystal frequency is basic internal clock frequency. The maximum and minimum frequencies are specified from 1to 24MHZ.

Program instructions may require one, two or four machine cycles to be executed depending on type of instructions. To calculate the time any particular instructions will take to be executed, the number of cycles 'C',

T = C*12d / Crystal frequency

Here, we chose frequency as 11.0592MHZ. This is because,

baud= 2*clock frequency/(32d. 12d[256d-TH1]).The oscillator is chosen to help generate both standard and nonstandard baud rates. If standard baud rates are desired, an 11.0592MHZ crystal should be selected. From our desired standard rate, TH1 can be calculated. The internally implemented value of capacitance is 33 pf.

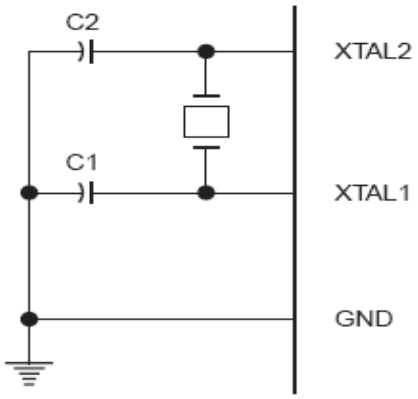


Fig 3.5 On-Chip Oscillators

POWER DOWN MODE

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and special function registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change VCC is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

POWER ON RESET

When power is turned on, the circuit holds the RST pin high for an amount of time that depends on the capacitor value and the rate at which it changes.

To ensure a valid reset, the RST pin must be held high long enough to allow the oscillator to start up plus two machine cycles. On power up, VCC should rise within approximately 10ms. The oscillator frequency. For a 10MHZ crystal, the start-up time is typically 1ms. With the given

circuit, reducing VCC quickly to 0 causes the RST pin voltage to momentarily fall below 0v. however, this voltage is internally limited and will not harm the device.

PROGRAM MEMORY LOCK BITS

On the chip there are three lock bits which can be left un programmed (U) or can be programmed (P) to obtain the additional features .When lock bit 1 is programmed, the logic level at the EA pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value, and holds that value until reset is activated. It is necessary that

the latched value of EA be in agreement with the current logic level at that pin in order for the device to function properly.

3.2.4 MEMORY ORGANISATION

control bus. 8051 is based on Von Neumann architecture. Hence 8051 consist of two separate Memory organization is depends on type of architecture used. There are two type of architectures are used in controller or processor generally

1) Von Neumann architecture

2) Harvard architecture

1) VON NEUMANN ARCHITECTURE

The term Von Neumann architecture, also known as the Von Neumann model or the Princeton architecture. This architecture consist of address memory and data memory on a single unit. 8085 is based on Von Neumann architecture.

2) HARWARE ARCHITECTURE

Harvard architecture consists of program memory and data memory as separate unit. Thus for accessing Harvard architecture we need separate address bus, data bus anmemory units, program memory as well as data memory

INTERNAL MEMORY

The 89C52 has internal RAM and ROM memory for the functions. Additional memory can be added externally using suitable circuits. This has a Hardware architecture, which uses the same address, in different memories, for code and data.

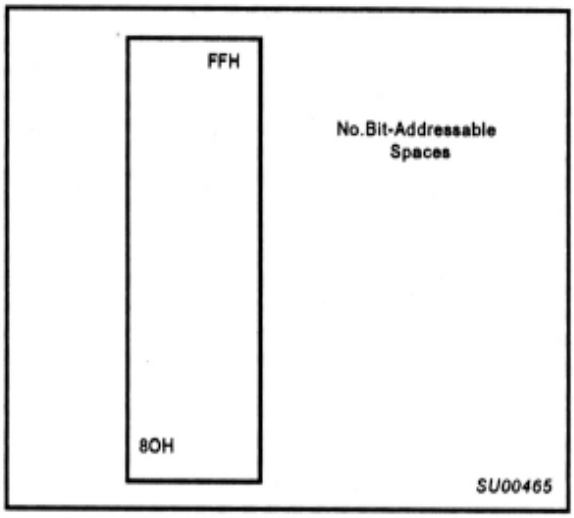
INTERNAL RAM

The 256-byte internal RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. Instructions that use indirect addressing access the upper 128 bytes of RAM. Stack operations are examples of indirect addressing.

Internal Data Memory addresses are always one byte wide, which implies an address space of only 256 bytes. However, the addressing modes for internal RAM can in fact accommodate 384 bytes, using a simple trick. Direct addresses higher than 7FH access one memory space, and indirect addresses higher than 7FH access a different memory space. Thus Figure shows the Upper 128 and SFR space occupying the same block of addresses, 80H through FFH, although they are physically separate entities.

The Lower 128 bytes of RAM are present in all 89C52 devices as mapped in Figure. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use.

This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing



Upper 128 Bytes of Internal RAM

.. The next 16 bytes above the register banks form a block of bit addressable memory space. The 89C52 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions The bit addresses in this area are 00H through 7FH. All of the bytes in the Lower 128 can be accessed by either direct or indirect adressing.The Upper 128 can only be accessed by indirect addressing. SFRs include the Port latches, timers, peripheral controls, etc. These registers can only be accessed by direct addressing. Sixteen addresses in SFR space are both byte- and bit-addressable. The bit-addressable SFRs are those whose address ends in OH or 80H

3.2.5 REGISTERS

The 89C52 contains 34 general-purpose, working registers. Two of these, registers A and B, hold results of many instructions, particularly math and logical operations, of the 89C52 CPU. The other 32 are arranged as part of internal RAM in four banks, B0-B3, of eight

registers. The A register is also used for all data transfers between the 89C52 and any external memory. The B register is used for with the A register for multiplication and division operations

In the CPU registers are used to store information temporally. That information could be a byte of data processed, or an address pointing to the data to be fetched. The vast majority of 8052 registers are 8-bit registers. In the 8052 there is only one data types:8bits. With an 8 bit data type , any data larger than 8 bits must be broken into 8 bit chunks before it is processed.

The most widely used registers of the 8052 are A(accumulator), B, R0, R1, R2, R3, R4, R5, R6, R7, DPTR(data pointer), PC(program counter). All the above registers are 8-bits, except DPTR and the program counter. The accumulator register A is used for all arithmetic and logic instructions.

SFR (SPECIAL FUNCTION REGISTERS)

The 128 bytes of on-chip additional RAM locations from 80H to FFH are reserved for the special functions and therefore these are called as special function registers SFRs. These SFRs are used for control or to show the status of various functions done by the 89C52 microcontroller. All SFRs are directly addressable and can be read or written to as well. Note

that SFRs space is only reserved for the special functions and cannot be used for any other purposes. Some SFRs are bit addressable and allow their individual bits to be set or cleared by instructions.

A REGISTER (ACCUMULATOR)

The most important of all special function registers, that's the first comment about **Accumulator** which is also known as **ACC** or **A.** The Accumulator (sometimes referred to as Register A also) holds the result of most of arithmetic and logic operations. ACC is usually accessed by direct addressing and its physical address is **E0H.** Accumulator is both byte and bit addressable. You can understand this from the figure shown below. To access the first bit (i.e bit 0) or to access accumulator as a single byte (all 8 bits at once), you may use the same physical address E0H. Now if you want to access the second bit (i.e bit 1), you may use E1H and for third bit E2H and soon.

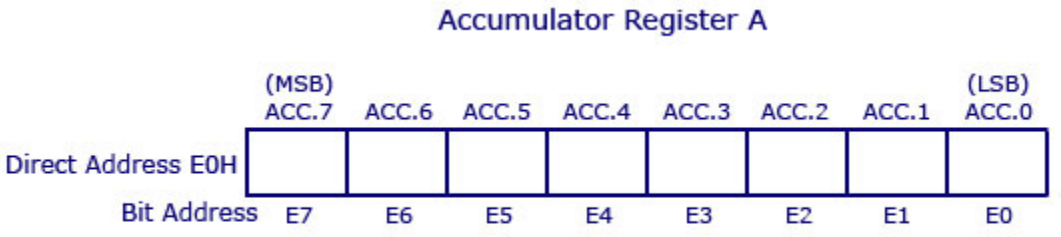


Fig 3.6 Accumulator

REGISTER B

The major purpose of this register is in executing multiplication and division. The 8051 micro controller has a single instruction for multiplication **(MUL)** and division **(DIV)**. If you are familiar with 8085, you may now know that multiplication is repeated addition, where as division is repeated subtraction. While programming 8085, you may have written a loop to execute repeated addition/subtraction to perform multiplication and division. Now here in 8051 you can do this with a single instruction.

Register B is also byte addressable and bit addressable. To access bit o or to access all 8 bits (as a single byte), physical address F0 is used. To access bit 1 you may use F1 and so on. Please take a look at the picture below.

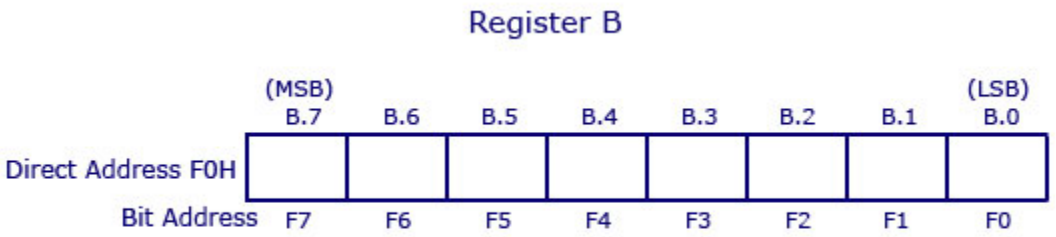


Fig 3.7 B Register

DATA POINTER (DPTR)

DPL and DPH work together to represent a 16-bit value called the *Data Pointer*. The Data Pointer (DPTR) is the only user-accessible 16-bit (2-byte) register. The Accumulator, "R" registers, and "B" register are all 1-byte values.

DPTR, as the name suggests, is used to point to data. It is used by a number of commands which allow the 8052 to access external memory. When the 8052 accesses external memory it will access external memory at the address indicated by DPTR.

While DPTR is most often used to point to data in external memory, many programmers often take advantage of the fact that its the only true 16-bit register available. It is often used to store 2-byte values which have nothing to do with memory locations.

PROGRAM COUNTER(PC)

The Program Counter (PC) is a 2-byte address which tells the 8052 where the next instruction to execute is found in memory. When the 8051 is initialized PC always starts at 0000h

and is incremented each time an instruction is executed. It is important to note that PC is not always incremented by one. Since some instructions require 2 or 3 bytes the PC will be incremented by 2 or 3 in these cases.

The Program Counter is special in that there is no way to directly modify its value. That is to say, you cannot do something like PC=2430h. On the other hand, if you execute LJMP 2430h you have effectively accomplished the same thing.

PROGRAM STATUS WORD (PSW)

The Program Status Word is used to store a number of important bits that are set and cleared by 8051 instructions. The PSW SFR contains the carry flag, the auxiliary carry flag, the overflow flag, and the parity flag. Additionally, the PSW register contains the register bank select flags which are used to select which of the "R" register banks are currently selected.

P-parity bit

if a number in accumulator is even then this bit will be automatically set(1), otherwise it will be cleared(0). It is mainly used during data transmission and receiving via serial communication.

Bit 1

This bit is intended for the future versions of the microcontroller, so it ts not supposed to be here.

OV-Over flow

Overflow occurs when the result of arithmetical operation is greater than 255(decimal), so that it cannot be stored in one register. In that case, this bit will be set(1). If there is no overflow, this bit will be cleared(0).

FO-Flag 0

This is a general-purpose bit available to the user

AC-Auxiliary carry flag

Auxiliary carry flag is used for BCD operations only.

CY-Carry flag

Carry flag is the(ninth) auxiliary bit used for all arithmetical operations and shift instructions.

STACK POINTER(SP)

This SFR indicates where the next value to be taken from the stack will be read from in Internal RAM. If you push a value onto the stack, the value will be written to the address of SP +

1. That is to say, if SP holds the value 07h, a PUSH instruction will push the value onto the stack at address 08h. This SFR is modified by all instructions which modify the stack, such as PUSH, POP, and LCALL, RET, RETI, and whenever interrupts are provoked by the microcontroller.

3.2.6 HOW STACKS ARE ACCESSED IN 8052

If the stack is a section of RAM, there must be registers inside the CPU to a point to it. The register used to access the stack is called SP(stack pointer) register. The stack pointer in the 8052 is only 8 bits wide; which means that it can take values of 00 to FFH. When the 8052 is powered up, the SP register contain value 07. This Means that RAM location 08 is the first location used for the stack by the 8052.the storing in the CPU register in the stack is called a PUSH, and pulling the contents off the stack back into a CPU register is called a POP. In other words, a register is pushed on to the stack to save it and popped off the stack to retrieve it. The job of the SP is very critical when push and pop actions are performed.

PUSHING ON TO THE STACK

In the 8052 the stack pointer(SP) points to the last used location of the stack. As it pushes data onto the stack, the stack pointer is incremented by one. Note that this difference from many microprocessors, notably *86 processor in which the SP is decremented when the data is pushed

onto the stack. As each PUSH is executed, the contents of the register are saved on the stack and SP is incremented by 1. Note that for every byte of data saved on the stack, SP is incremented only

once. Note also that to push the registers onto the stack it must use it's RAM addresses. For example, the instruction "PUSH" pushes register R1 on to the stack.

POPPING FROM THE STACK

Popping the contents of the stack back into a given register is the opposite process of pushing. With every pop, the top byte of the stack is copied to the register specified by the instruction and the stack pointer is decremented once.

THE UPPER LIMIT OF STACK

The location 08 to 1FH in the 8052 RAM can be used for the stack. This is because location 20-2FH of RAM are reserved for bit-addressable memory. In a program when need more than 24 bytes(08H to 1FH=24bytes) of stack, it is preferred to change the SP to point to RAM locations 30 -7FH.

PCON (power control, 87h)

The Power Control SFR is used to control the 8052's power control modes. Certain operation modes of the 8052 allow the 8052 to go into a type of "sleep" mode which requires much less power. These modes of operation are controlled through PCON. Additionally, one of the bits in PCON is used to double the effective baud rate of the 8052's serial port

PORT REGISTERS

There are 4 ports for 8051, So 4 Input or Output ports named P0, P1, P2 and P3 has got four corresponding port registers with same name P0, P1, P2 and P3. Data must be written into

port registers first to send it out to any other external device through ports. Similarly any data received through ports must be read from port registers for performing any operation. All 4 port registers are bit as well as byte addressable.

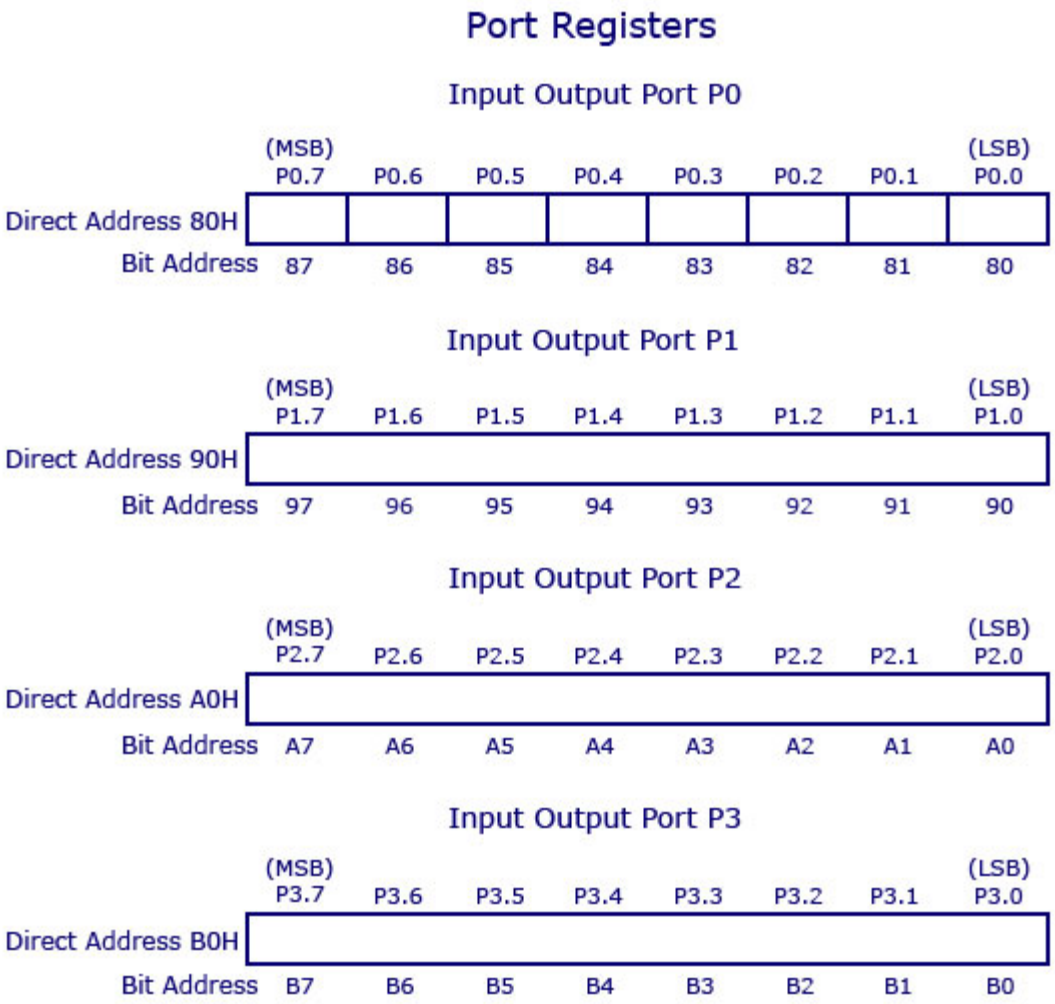


Fig 3.8 port registers

Timer SFRs

One timer is TIMER0 and the other is TIMER1. The two timers share two SFRs (TMOD and TCON) which control the timers, and each timer also has two SFRs dedicated solely to itself (TH0/TL0 and TH1/TL1).

Table 3.3 TIMER Register

SFR Name	**Description**	**SFR Address**
TH0	Timer 0 High Byte	8Ch
TL0	Timer 0 Low Byte	8Ah
TH1	Timer 1 High Byte	8Dh
TL1	Timer 1 Low Byte	8Bh
TCON	Timer Control	88h
TMOD	Timer Mode	89h

We have given SFRs names to make it easier to refer to them, but in reality an SFR has a numeric address. It is often useful to know the numeric address that corresponds to an SFR name. The SFRs relating to timers are:

TCON (Timer Control, Addresses 88h, Bit-Addressable)

The Timer Control SFR is used to configure and modify the way in which the two timers operate. This SFR controls whether each of the two timers is running or stopped and contains a flag to indicate that each timer has overflowed. Additionally, some non-timer related bits are located in the TCON SFR. These bits are used to configure the way in which the external interrupts are activated and also contain the external interrupt flags which are set when an external interrupt has occurred.

The TCON SFR has the following structure:

Table 3.4 TCON (88h) SFR

Bit	Name	Bit Address	Explanation of Function	Timer
7	TF1	8Fh	**Timer 1 Overflow**. This bit is set by the microcontroller when Timer 1 overflows.	1
6	TR1	8Eh	**Timer 1 Run**. When this bit is set Timer 1 is turned on. When this bit is clear Timer 1 is off.	1
5	TF0	8Dh	**Timer 0 Overflow**. This bit is set by the microcontroller when Timer 0 overflows.	0
4	TR0	8Ch	**Timer 0 Run**. When this bit is set Timer 0 is turned on. When this bit is clear Timer 0 is off.	0

As you may notice, we have only defined 4 of the 8 bits. That is because the other 4 bits of the SFR do not have anything to do with timers--they have to do with Interrupts

TMOD (Timer Mode, Addresses 89h)

The Timer Mode SFR is used to configure the mode of operation of each of the two timers. Using this SFR your program may configure each timer to be a 16-bit timer, an 8-bit auto reload timer, a 13-bit timer, or two separate timers. Additionally, you may configure the timers to only count when an external pin is activated or to count "events" that are indicated on an external pin.

1. TL0/TH0 (Timer 0 Low/High, Addresses 8Ah/8Ch): These two SFRs, taken together, represent timer 0. Their exact behavior depends on how the timer is configured in the TMOD SFR; however, these timers always count up. What is configurable is how and when they increment in value.

2. TL1/TH1 (Timer 1 Low/High, Addresses 8Bh/8Dh): These two SFRs, taken together, represent timer 1. Their exact behavior depends on how the timer is configured in the TMOD SFR; however, these timers always count up. What is configurable is how and when they increment in value.

Each bit of the SFR gives the microcontroller specific information concerning how to run a timer. The high four bits (bits 4 through 7) relate to Timer 1 whereas the low four bits (bits 0 through 3) perform the exact same functions, but for timer 0.

The individual bits of TMOD have the following functions:

Table 3.5 TMOD (89h) SFR

Bit	Name	Explanation of Function	Timer
7	GATE1	When this bit is set the timer will only run when INT1 (P3.3) is high. When this bit is clear the timer will run regardless of the state of INT1.	1
6	C/T1	When this bit is set the timer will count events on T1 (P3.5). When this bit is clear the timer will be incremented every machine cycle.	1
5	T1M1	Timer mode bit (see below)	1
4	T1M0	Timer mode bit (see below)	1
3	GATE0	When this bit is set the timer will only run when INT0 (P3.2) is high. When this bit is clear the timer will run regardless of the state of INT0.	0
2	C/T0	When this bit is set the timer will count events on T0 (P3.4). When this bit is clear the timer will be incremented every machine cycle.	0
1	T0M1	Timer mode bit (see below)	0
0	T0M0	Timer mode bit (see below)	0

As you can see in the above chart, four bits (two for each timer) are used to specify a mode of operation.

Table 3.6 The modes of operation are:

TxM1	TxM0	Timer Mode	Description of Mode

0	0	0	13-bit Timer.
0	1	1	16-bit Timer
1	0	2	8-bit auto-reload
1	1	3	Split timer mode

13-bit Time Mode (mode 0)

Timer mode "0" is a 13-bit timer. Generally the 13-bit timer mode is not used in new development. When the timer is in 13-bit mode, TLX will count from 0 to 31. When TLX is incremented from 31, it will "reset" to 0 and increment THX. Thus, effectively, only 13 bits of the two timer bytes are being used: bits 0-4 of TLX and bits 0-7 of THX. This also means, in essence, the timer can only contain 8192 values. If you set a 13-bit timer to 0, it will overflow back to zero 8192 machine cycles later.

16-bit Time Mode (mode 1)

Timer mode "1" is a 16-bit timer. This is a very commonly used mode. It functions just like 13-bit mode except that all 16 bits are used.

TLX is incremented from 0 to 255. When TLX is incremented from 255, it resets to 0 and causes THX to be incremented by 1. Since this is a full 16-bit timer, the timer may contain up to 65536 distinct values. If you set a 16-bit timer to 0, it will overflow back to 0 after 65,536 machine cycles.

8-bit Time Mode (mode 2)

Table 3.7 8-bit time mode

Machine Cycle	TH0 Value	TL0 Value
1	FDH	FEH
2	FDH	FFH
3	FDH	FDH
4	FDH	FEH
5	FDH	FFH
6	FDH	FDH
7	FDH	FEH

Timer mode "2" is an 8-bit auto-reload mode. When a timer is in mode 2, THX holds the "reload value" and TLX is the timer itself. Thus, TLX starts counting up. When TLX reaches 255 and is subsequently incremented, instead of resetting to 0 (as in the case of modes 0 and 1), it will be reset to the value stored in THX.

For example, lets say TH0 holds the value FDH and TL0 holds the value FEH. If we were to watch the values of TH0 and TL0 for a few machine cycles this is what we see:

Split Timer Mode (mode 3)

Timer mode "3" is a split-timer mode. When Timer 0 is placed in mode 3, it essentially becomes two separate 8-bit timers. That is to say, Timer 0 is TL0 and Timer 1 is TH0. Both timers count from 0 to 255 and overflow back to 0. All the bits that are related to Timer 1 will now be tied to TH0.

While Timer 0 is in split mode, the real Timer 1 (i.e. TH1 and TL1) can be put into modes 0, 1 or 2 normally--however, you may not start or stop the real timer 1 since the bits that do that are now linked to TH0. The real timer 1, in this case, will be incremented every machine cycle no matter what.

The only real use I can see of using split timer mode is if you need to have two separate timers and, additionally, a baud rate generator.

SBUF (Serial Control, Addresses 99H)

The Serial Buffer SFR is used to send and receive data via the on-board serial port. Any value written to SBUF will be sent out the serial port's TXD pin. Likewise, any value which the 8051 receives via the serial port's RXD pin will be delivered to the user program via SBUF. In other words, SBUF serves as the output port when written to and as an input port when read from.

SCON(serial control)

Table 3.8 SCON Register

Bit	Name	Bit Address	Explanation of Function
7	SM0	9FH	Serial port mode bit 0
6	SM1	9EH	Serial port mode bit 1.
5	SM2	9DH	Multiprocessor Communications Enable (explained later)
4	REN	9CH	Receiver Enable. This bit must be set in order to receive characters.
3	TB8	9BH	Transmit bit 8. The 9th bit to transmit in mode 2 and 3.
2	RB8	9AH	Receive bit 8. The 9th bit received in mode 2 and 3.
1	TI	99H	Transmit Flag. Set when a byte has been completely transmitted.
0	RI	98H	Receive Flag. Set when a byte has been completely received.

Additionally, it is necessary to define the function of SM0 and SM1 by an additional table:

SM0	SM1	Serial Mode	Explanation	Baud Rate
0	0	0	8-bit Shift Register	Oscillator / 12
0	1	1	8-bit UART	Set by Timer 1 (*)
1	0	2	9-bit UART	Oscillator / 64 (*)
1	1	3	9-bit UART	Set by Timer 1 (*)

The SCON SFR allows us to configure the Serial Port. Thus, well go through each bit and review its function.

The first four bits (bits 4 through 7) are configuration bits.

Bits **SM0** and **SM1** let us set the *serial mode* to a value between 0 and 3, inclusive. The four modes are defined in the chart immediately above. As you can see, selecting the Serial Mode selects the mode of operation (8-bit/9-bit, UART or Shift Register) and also determines how the baud rate will be calculated. In modes 0 and 2 the baud rate is fixed based on the oscillators frequency. In modes 1 and 3 the baud rate is variable based on how often Timer 1 overflows.

The next bit, **SM2**, is a flag for "Multiprocessor communication." Generally, whenever a byte has been received the 8051 will set the "RI" (Receive Interrupt) flag. This lets the program know that a byte has been received and that it needs to be processed. However, when SM2 is set the "RI" flag will only be triggered if the 9th bit received was a "1". That is to say, if SM2 is set and a byte is received whose 9th bit is clear, the RI flag will never be set. This can be useful in certain advanced serial applications.

The next bit, **REN**, is "Receiver Enable." This bit is very straightforward: If you want to receive data via the serial port, set this bit. You will almost always want to set this bit.

The last four bits (bits 0 through 3) are operational bits. They are used when actually sending and receiving data--they are not used to configure the serial port.

The **TB8** bit is used in modes 2 and 3. In modes 2 and 3, a total of nine data bits are transmitted. The first 8 data bits are the 8 bits of the main value, and the ninth bit is taken from TB8. If TB8 is set and a value is written to the serial port, the data bits will be written to the serial line followed by a "set" ninth bit. If TB8 is clear the ninth bit will be "clear."

The **RB8** also operates in modes 2 and 3 and functions essentially the same way as TB8, but on the reception side. When a byte is received in modes 2 or 3, a total of nine bits are received.

In this case, the first eight bits received are the data of the serial byte received and the value of the ninth bit received will be placed in RB8.

TI means "Transmit Interrupt." When a program writes a value to the serial port, a certain amount of time will pass before the individual bits of the byte are "clocked out" the serial port. If the program were to write another byte to the serial port before the first byte was completely output, the data being sent would be garbled. Thus, the 8051 lets the program know that it has "clocked out" the last byte by setting the TI bit. When the TI bit is set, the program may assume that the serial port is "free" and ready to send the next byte.

Finally, the **RI** bit means "Receive Interrupt." It functions similarly to the "TI" bit, but it indicates that a byte has been received. That is to say, whenever the 8051 has received a complete byte it will trigger the RI bit to let the program know that it needs to read the value.

3.2.7 PIN DIAGRAM AND ITS DESCRIPTION

The microcontroller generic part number actually includes a whole family of microcontrollers that have numbers ranging from 8031to 8751 and are available in N-Channel Metal Oxide Silicon (NMOS) and Complementary Metal Oxide Silicon (CMOS) construction in a variety of package types.

PIN DIAGRAM

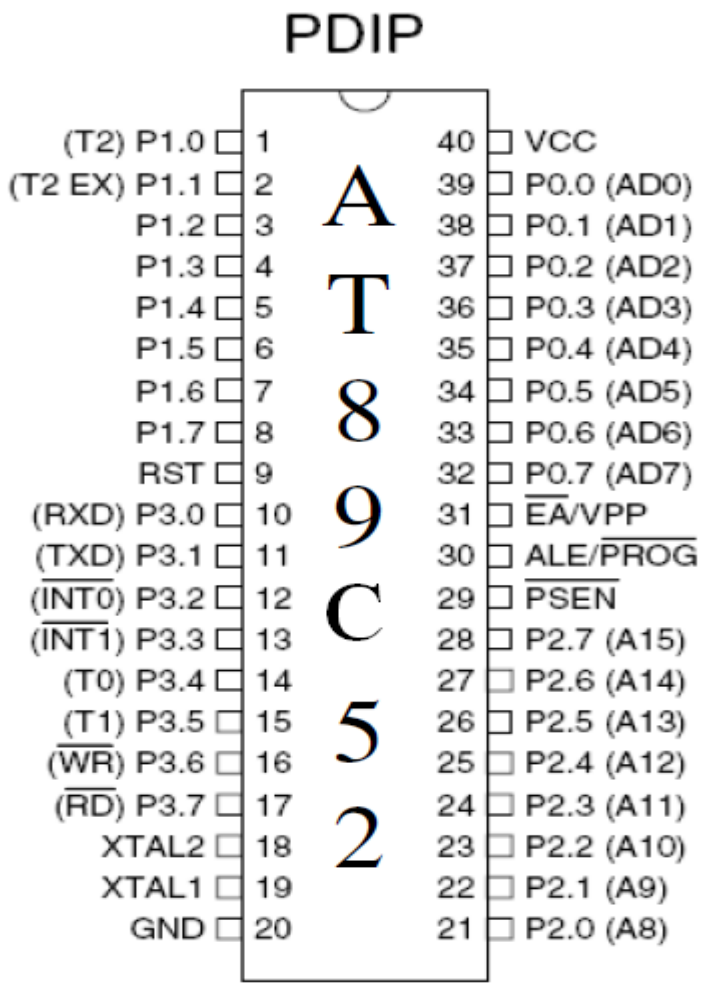


Fig 3.9 Pin Diagram

with 4Kbytes of Flash Programmable and Erasable Read Only Memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51 instruction set and pin out. 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 Flash on a monolithic chip, the Atmel AT89C52 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

The AT89C52 provides the following standard features: 4 Kbytes of Flash, 256 bytes of RAM, 32 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator and clock circuitry. In addition, the AT89C52 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 hardware reset.

3.2.8 INTERRUPTS

As the name implies, an **interrupt** is some event which interrupts normal program execution. As stated earlier, program flow is always sequential, being altered only by those instructions which expressly cause program flow to deviate in some way. However, interrupts give us a mechanism to "put on hold" the normal program flow, execute a subroutine, and then resume normal program flow as if we had never left it. This subroutine, called an interrupt handler, is only executed when a certain event (interrupt) occurs. The event may be one of the timers "overflowing," receiving a character via the serial port, transmitting a character via the serial port, or one of two "external events." The 8052 may be configured so that when any of these events occur the main program is temporarily suspended and control passed to a special section of code which presumably would execute some function related to the event that occurred. Once complete, control would be returned to the original program. The main program never even knows it was interrupted.

The ability to interrupt normal program execution when certain events occur makes it much easier and much more efficient to handle certain conditions. If it were not for interrupts we would

have to manually check in our main program whether the timers had overflow, whether we had received another character via the serial port, or if some external event had occurred. Besides making the main program ugly and hard to read, such a situation would make our program inefficient since wed be burning precious "instruction cycles" checking for events that usually do not happen. Any of the following events will cause an interrupt:

- Timer 0 Overflow.
- Timer 1 Overflow.
- Reception/Transmission of Serial Character.
- External Event 0.
- External Event 1.

This is accomplished by jumping to a fixed address when a given interrupt occurs When checking for interrupt conditions, it checks them in the following order:

- External 0 Interrupt
- Timer 0 Interrupt
- External 1 Interrupt
- Timer 1 Interrupt
- Serial Interrupt

Table 3.9 Interrupt handler

Interrupt	**Flag**	**Interrupt Handler Address**
External 0	IE0	0003h

Timer 0	TF0	000Bh
External 1	IE1	0013h
Timer 1	TF1	001Bh
Serial	RI/TI	0023h

IP (Interrupt Priority, Addresses B8h, Bit-Addressable)

Table 3.10 Interrupt priority

Bit	Name	Bit Address	Explanation of Function
7	-	-	Undefined
6	-	-	Undefined
5	-	-	Undefined
4	PS	BCH	Serial Interrupt Priority
3	PT1	BBH	Timer 1 Interrupt Priority
2	PX1	BAH	External 1 Interrupt Priority
1	PT0	B9H	Timer 0 Interrupt Priority
0	PX0	B8H	External 0 Interrupt Priority

The Interrupt Priority SFR is used to specify the relative priority of each interrupt. On the 8051, an interrupt may either be of low (0) priority or high (1) priority. An interrupt may only interrupt interrupts of lower priority. For example, if we configure the 8051 so that all interrupts are of low priority except the serial interrupt, the serial interrupt will always be able to interrupt the system, even if another interrupt is currently executing. However, if a serial interrupt is

executing no other interrupt will be able to interrupt the serial interrupt routine since the serial interrupt routine has the highest priority. interrupt priorities are controlled by the **IP** SFR (B8h). The IP SFR has the following format

When considering interrupt priorities, the following rules apply:

- Nothing can interrupt a high-priority interrupt--not even another high priority interrupt.
- A high-priority interrupt may interrupt a low-priority interrupt.
- A low-priority interrupt may only occur if no other interrupt is already executing.

If two interrupts occur at the same time, the interrupt with higher priority will execute first. If both interrupts are of the same priority the interrupt which is serviced first by polling sequence will be executed first

Serial Interrupts are slightly different than the rest of the interrupts. This is due to the fact that there are two interrupt flags: RI and TI. If either flag is set, a serial interrupt is triggered. As you will recall from the section on the serial port, the RI bit is set when a byte is received by the serial port and the TI bit is set when a byte has been sent.

This means that when your serial interrupt is executed, it may have been triggered because the RI flag was set or because the TI flag was set--or because both flags were set. Thus, your routine must check the status of these flags to determine what action is appropriate. Also, since the 8052 does not automatically clear the RI and TI flags you must clear these bits in your interrupt handler.

IE (Interrupt Enable, Addresses A8h)

The Interrupt Enable SFR is used to enable and disable specific interrupts. The low 7 bits of the SFR are used to enable/disable the specific interrupts, where as the highest bit is used to enable or disable ALL interrupts. Thus, if the high bit of IE is 0 all interrupts are disabled regardless of whether an individual interrupt is enabled by setting a lower bit.

3.2.9 SERIAL COMMUNICTION

Computers can transfer data in two ways-parallel and serial. In parallel data transfers, often 8 or more lines (wire conductors) are used to transfer data to a device that is only a few feet away. Examples of parallel transfers are printers and hard disks; each uses cables with many wire strips. Although in such cases a lot of data can be transferred in a short amount of time by using many wires in parallel, the distance cannot be great. To transfer to a device located many meters away, the serial method is used. serial communication, the data is sent one bit at a time, in contrast to parallel communication, in which the data is sent byte or more at a time. The 8052 has serial communication capability built into it, thereby making possible fast data transfer using only a few wires.

If data is to be transferred on the telephone line, it must be converted from 0s and 1s to audio tones, which are sinusoidal-shaped signals. This conversion is performed by a peripheral device called a modem, which stands for ” modulator/demodulator.”

Serial data communication uses two methods, asynchronous and synchronous. The synchronous method transfers a block of data at a time, while the asynchronous method transfers a single byte at a time.

In data transmission if the data can be transmitted or received, it is duplex transmission. This is in contrast to simplex transmission such as with printers, in which the computer only sends the data. Duplex transmissions can be half or full duplex, depending on whether or not the data transfer can be simultaneous. If data is transmitted

One way at a time, it is full duplex. Of course, full duplex requires two wire conductors for the data lines, one for transmission and one for reception, in order to transfer and receive data simultaneously.

ASYNCHRONOUS SERIAL COMMUNICATION AND DATA FRAMMING

The data coming in the receiving end of the data line in a serial data transfer is all 0s and 1s; it is difficult to make sense of the data unless the sender and receiver agree on a set of rules, a protocol, on how the data is packed, how many bits constitute a character, and when the data begins and ends.

START AND STOP BITS

Asynchronous serial data communication is widely used for character-oriented transmission, while block-oriented data transfers use the synchronous method. In the asynchronous method, each character is placed between start and stop bits. This is called framing. In the data framing for asynchronous communications, the data, such as ASCII Character, is packed between a start bit and a stop bit. The start bit is always one bit, but the stop bit is always one bit, but the stop bit can be one or two bits. The start bit is always a 0(low) and the stop bit(s) is 1(high)

DATA TRANSFER RATE

The rate of data transfer in serial data communication is stated in bps(bits per second). Another widely used terminology for bps is baud rate. However, the baud and bps rates are not necessarily equal. This is due to the fact that baud rate is the modem terminology and is defined as the number of signal change of signal, sometimes transfers the same, and for this reason bps and baud interchangeably used.

The data transfer rate of given computer system depends on communication ports incorporated into that system. for example, the early IBMPC/XT could transfer data at a the rate of100 to 9600 bps. In recent years, however, Pentium based PCS transfer data at rates as high as 56k bps. it must be noted that in asynchronous serial data communication, the baud rate is generally limited to 100,000bps.

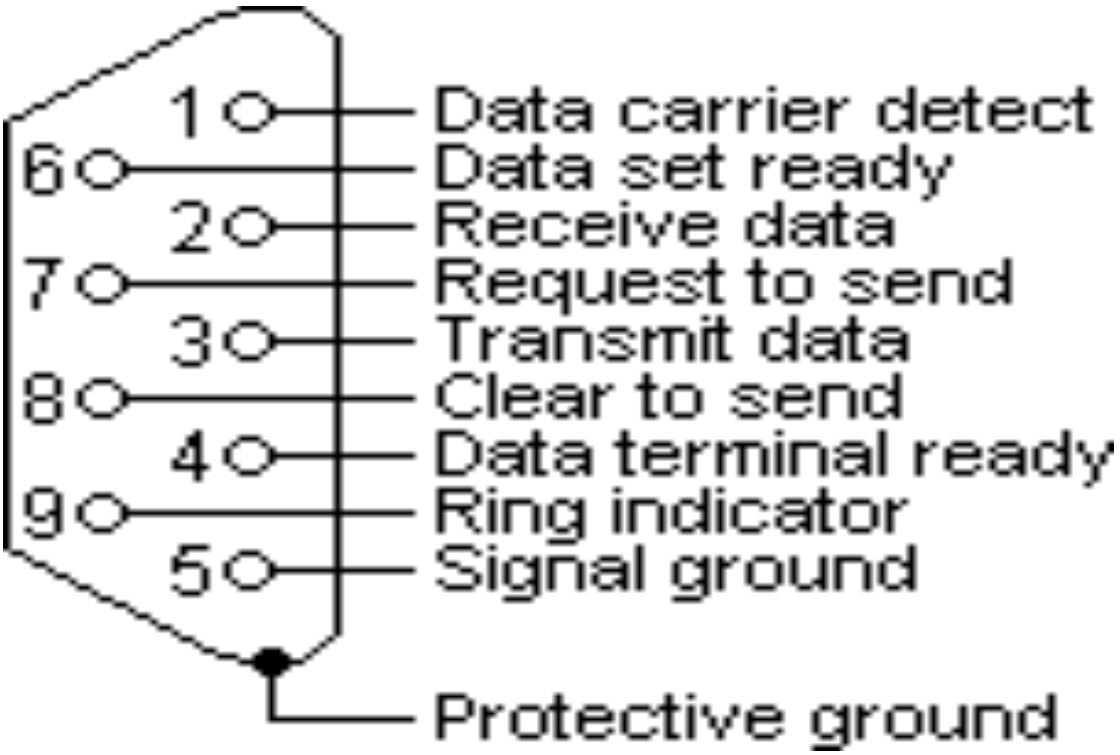
RS232 STANDARDS

To allow compatibility among data communication equipment made by various manufacturers, an interfacing standard called RS232 was set by the electronics industries association(EIA) in 1960. In 1963 it was modified and called RS232A. RS232B AND RS232C were issued in 1965 and 1969, respectively. Today, RS232 is the most widely used serial I/O interfacing standard. This standard is used in PCs and numerous types of equipment. However, since the standard is used in PCs and numerous types of equipment.

However, since the standard was set along before the advert of the TTL logic family, its input and output voltage levels are not TTL compatible. I

DB-9 PIN CONNECTOR

Fig 3.10 DB-pin connector



RS232, a 1 is represented by -3 to -25, while a 0 bit is +3 to +25v, making -3 to +3 undefined. For this reason, to connect any RS232 to a microcontroller system use voltage convertors such as MAX232 to convert the TTL logic levels, and vice versa. MAX232 IC chips are commonly referred to as line drivers.

RS232 PINS

RS232 cable, commonly referred to as the DB-25 connector. In labeling, DB-25P refers to the plug connector(male) and DB-25S is for the socket connector(female). Since not all the pins

are used in PC cables, IBM introduced the DB-9 version of the I/O standard, which uses 9 pins only, as shown in table.

Table 3.11 signal description

Signal	In/Out	Description
DCD	In	Data Carrier Detect
RXD	In	Receive Data
TXD	Out	Transmit Data
DTR	Out	Data Terminal Ready
GND	-	Ground
DSR	In	Data Set Ready
RTS	Out	Request To Send
CTS	In	Clear To Send
RI	In	Ring Indicator

COMMUNICATION SIGNALS

TXD: carriers data from DTE to the DCE.

RXD: carries data from DCE to the DTE.

SG: signal ground

Note: DCD, DSR, RTS and CTS are active low pins

The TXD (TRANSMIT DATA)

Wire is the one through which data from a DTE device is transmitted to a DCE device. This name can be deceiving, because this wire is used by a DCE device to receive its data. The TD line is kept in a mark condition by the DTE device when it is idle. The RD (receive data) wire is the one on which data is received by a DTE device, and the DCE device keeps this line in a mark condition when idle.

RTS (REQUEST TO SEND)

This line and the CTS line are used when "Hardware flow control" is enabled in both the DTE and DCE devices. The DTE device puts this line in a mark condition to tell the remote device that it is ready and able to receive data. If the DTE device is not able to receive data (typically because its receive buffer is almost full), it will put this line in the space condition as a signal to the DCE to stop sending data. When the DTE device is ready to receive more data (i.e. after data has been removed from its receive buffer), it will place this line back in the mark condition. The complement of the RTS wire is CTS, which stands for Clear To Send. The DCE device puts this line in a mark condition to tell the DTE device that it is ready to receive the data. Likewise, if the DCE device is unable to receive data, it will place this line in the space condition. Together, these two lines make up what is called RTS/CTS or "hardware" flow control. DTR stands for Data Terminal Ready. Its intended function is very similar to the RTS line. DSR (Data Set Ready) is the companion to DTR in the same way that CTS is to RTS. Some

serial devices use DTR and DSR as signals to simply confirm that a device is connected and is turned on. The Software Wedge sets DTR to the mark state when the serial port is opened and

leaves it in that state until the port is closed. The DTR and DSR lines were originally designed to provide an alternate method of hardware handshaking. It would be pointless to use both RTS/CTS and DTR/DSR for flow control signals at the same time. Because of this, DTR and DSR are rarely used for flow control.

CD(CARRIER DETECT)

Carrier Detect is used by a modem to signal that it has a made a connection with another modem, or has detected a carrier tone.

RI(RING INDICATOR)

A modem toggles the state of this line when an incoming call rings your phone. The Carrier Detect (CD) and the Ring Indicator (RI) lines are only available in connections to a modem. Because most modems transmit status information to a PC when either a carrier signal is detected (i.e. when a connection is made to another modem)

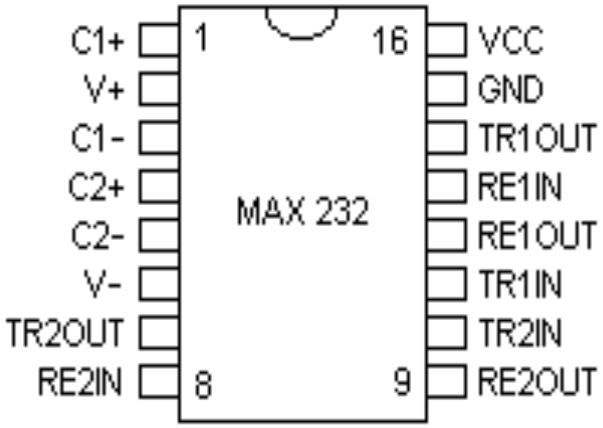
MAX-232 PIN DIAGRAM

Serial RS-232 communication works with voltages (between -15V… -3V are used to transmit a binary 1 and +3V … +15V to transmit a binary 0) which are not compatible with today's computer logic voltages. On the other hand,

classic TTL computer logic operates between 0V … +5V (0V are referred to as low for binary 0, +5V are referred to as high for binary 1).

Therefore, to receive serial data from an RS-232 interface the voltage has to reduced, and the 0 and 1 voltage levels inverted. In the other direction (sending data from some logic over RS-232) the low voltage has to "bumped up", and a negative voltage MAX 232C is used to match between the RS232 and TTL levels. The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply RS232 voltage levels from a single 5V supply. has to be generated, too. Today, RS232 is the most widely used serial I/O interfacing standard. This standard is used in PCs and numerous types of equipment

Fig 3.11 MAX 232



PIN DIAGRAM OF MAX-232

Table 3.12 MAX232(A) DIP Package Pin Layout

No.	Name	Purpose	Signal Voltage	Capacitor Value MAX232	Capacitor Value MAX232A
1	C1+	+ connector for capacitor C1	capacitor should stand at least 16V	1µF	100nF
2	V+	output of voltage pump	+10V, capacitor should stand at least 16V	1µF to VCC	100nF to VCC
3	C1-	- connector for capacitor C1	capacitor should stand at least 16V	1µF	100nF
4	C2+	+ connector for capacitor C2	capacitor should stand at least 16V	1µF	100nF
5	C2-	- connector for capacitor C2	capacitor should stand at least 16V	1µF	100Nf
6	V-	output of voltage pump / inverter	-10V, capacitor should stand at least 16V	1µF to GND	100nF to GND
7	T2out	Driver 2 output	RS-232		

8	R2in	Receiver 2 input	RS-232		
9	R2out	Receiver 2 output	TTL		
10	T2in	Driver 2 input	TTL		
11	T1in	Driver 1 input	TTL		
12	R1out	Receiver 1 output	TTL		
13	R1in	Receiver 1 input	RS-232		
14	T1out	Driver 1 output	RS-232		
15	GND	Ground	0V	1µF to VCC	100nF to VCC
16	VCC	Power supply	+5V		

8052 CONNECTION TO RS232

The RS232 standard is not TTL compatible; therefore, it requires a line driver such s the MAX232 chip to convert RS232 voltage levels to TTL levels, and vice versa. The 8052 has two pins that are used specifically for transferring and receiving data serially. These two pins are called

TXD and RXD and a part of the port 3 group (P3.0 and P3.1), pin 11 of the 8052 is assigned to TXD and pin 10 is designed as RXD. These pins are TTL compatible

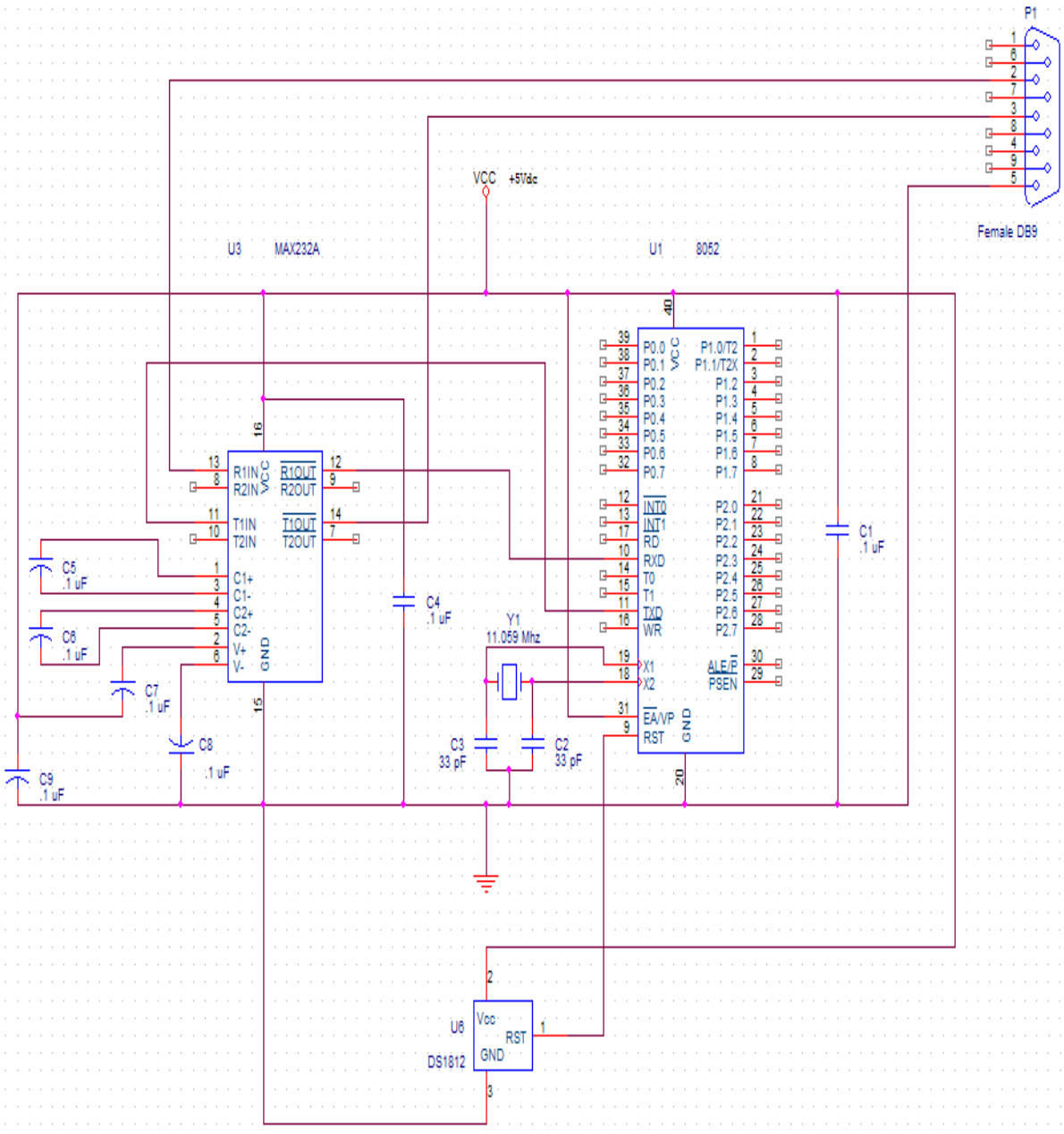


Fig 3.12 8052 connection to RS 232

.;

therefore, they require a line driver to make them RS232 compatible. One such line driver is the MAX232 chip.

MAX232 convert from RS232 voltage levels to TTL voltage levels, and vice versa. One advantage of the MAX232 chip is that it uses a +5v power source which, is the same as the source voltage for the 8052. In other words, with a single +5v power supply it is possible power both the 8052 and MAX232, with no need for the power supplies that are common in many older systems. The MAX232 has two sets of line drivers for transferring and receiving data. The line drivers used for TXD are called T1 and T2, while the line drivers for RXD are designed as R1 and R2. In many applications only of each is used.

3.3 POWER SUPPLY

A diode can be used as rectifier. There are various types of diodes. However, semiconductor diodes are very popularly used as rectifiers. A semiconductor diode is a solid-state device consisting of two elements is being an electron emitter or cathode, the other an electron collector or anode. Since electrons in a semiconductor diode can flow in one direction only-form emitter to collector-the diode provides the unilateral conduction necessary for rectification. The rectified Output is filtered for smoothening the DC, for this purpose capacitor is used in the filter circuit.

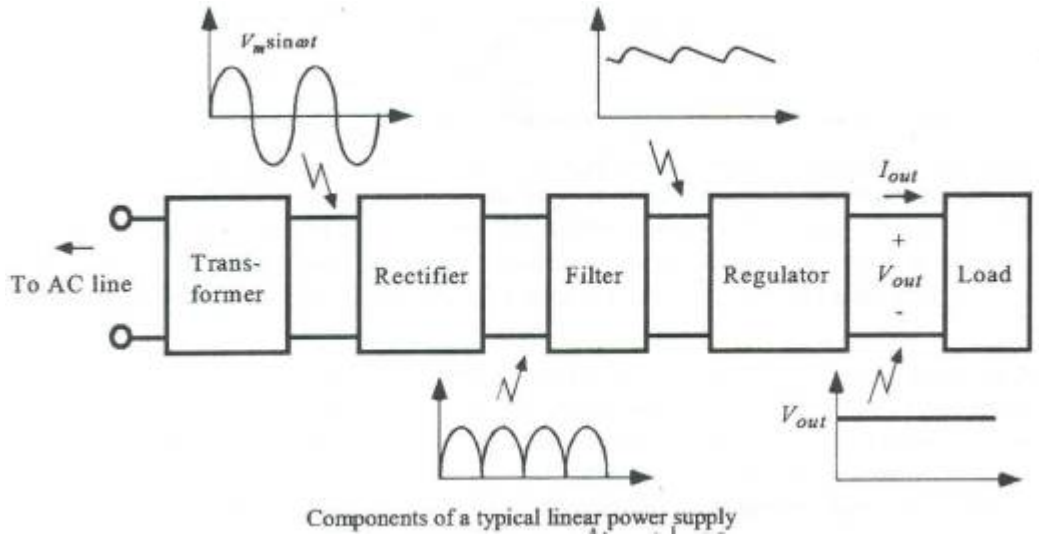


Fig 3.13 Power supply

The filter capacitors are usually connected in parallel with the rectifier output and the load. The AC can pass through a capacitor but DC cannot, the ripples are thus limited and the output becomes smoothed. When the voltage across the capacitor plates tends to rise, it stores up energy back into voltage and current. Thus, the fluctuation in the output voltage is reduced considerable.

3.3.1 VOLTAGE REGULATOR

LM 78XX SERIES VOLTAGE REGULATOR

The LM 78XXX series of the three terminal regulations is available with several fixed output voltages making them useful in a wide range of applications. One of these is local on card regulation. The voltages available allow these regulators to be used in logic systems, instrumentation and other solid state electronic equipment. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents. The LM78XX series is available in aluminum to 3 packages which will allow over 1.5A load current if adequate heat sinking is provided. Current limiting is included to limit the peak output current to a safe value. The LM 78XX is available in the metal 3 leads to 5 and the plastic to 92. For this type, with adequate heat sinking. The regulator can deliver 100mA output current.

The advantage of this type of regulator is, it is easy to use and minimize the number of external components.

The following are the features voltage regulators:

a) Output current in excess of 1.5A for 78 and 78L series

b) Internal thermal overload protection

c) No external components required

d) Output transistor sage area protection

e) Internal short circuit current limit.

f) Available in aluminum 3 package.

POSITIVE VOLTAGE REGULATOR

The positive voltage regulator has different features like

- Output current up to 1.5A
- No external components
- Internal thermal overload protection
- High power dissipation capability
- Internal short-circuit current limiting
- Output transistor safe area compensation
- Direct replacements for Fairchild microA7800 series

3.4 LIQUID CRYSTAL DISPLAY

In 1968, RCA Laboratories developed the first liquid crystal display (LCD). Since then, LCD's have been implemented on almost all types of digital devices, from watches to computer to projection TVs .LCD's operate as a light "valve", blocking light or allowing it to pass through. An image in an LCD is formed by applying an electric field to alter the chemical properties of each LCC (Liquid Crystal Cell) in the display in order to change a pixel's light absorption properties. These LCC's modify the image produced by the backlight into the screen output requested by the controller. Through the end output may be in color, the LCC's are monochrome, and the color is added later through a filtering process. Modern laptop computer displays can produce 65,536 simultaneous colors at resolution of 800 X 600.

To understand the operation of an LCD, it is easiest to trace the path of a light ray from the backlight to the user. The light source is usually located directly behind the LCD, and can use either LED or conventional fluorescent technology. From this source, the light ray will pass through a light polarizer to uniformly polarize the light so it can be acted upon by the liquid crystal (LC) matrix. The light beam will then pass through the LC matrix, which will determine whether this pixel should be "on" or "off". If the pixel is "on", the liquid crystal cell is electrically activated, and the molecules in the liquid will align in a single direction. This will allow the light to pass through unchanged. If the pixel is "off", the electric field is removed from the liquid, and the molecules with in scatter. This dramatically reduces the light that will pass through the display at that pixel.

In a color display, after the light passes through the liquid crystal matrix, it passes through a color filter (usually glass). This filter blocks all wavelengths of light except those within the range of that pixel. In a typical RGB display, the color filter is integrated into the upper glass colored microscopically to render each individual pixel red, green or blue. The areas in between

the colored pixel filter areas are printed black to increase contrast. After a beam of light passes through the color filter, it passes through yet another polarizer to sharpen the image and eliminate glare. The image is then available for viewing.

In an AMLCD, each LCC is stimulated individually by a dedicated transistor or diode. The two existing AMLCD technologies are Thin Film Transistor (TFT) and metal-insulator-metal (MIM). In an MIM display, dedicated diodes are fabricated at each pixel. MIM displays, currently being manufactured by Toshiba and Seiko-Epson, are not advantageous that TFT displays.

3.4.1 INTERFACING LCD TO THE MICROCONTROLLER

This is the first interfacing example for the parallel port. We will star with something simple. This example does not use the Bi-directional feature found on newer ports, thus it should work with most, if no all Parallel Ports. It however does not show the use of the status port as an input. So what are we interfacing? A 16 Character X 2 Line LCD Module to the Parallel Port. These LCD Modules are very common these days, and are quite simple to work with, as all the logic required running them is on board.

Features

- Interface with either 4-bit or 8-bit microprocessor.
- Display data RAM
- 80 × 8 bits (80 characters).
- Character generator ROM
- 160 different 5 × 7 dot-matrix character patterns.
- Character generator RAM

- 8 different user programmed 5 ⁄ 7 dot-matrix patterns.
- Display data RAM and character generator RAM may be
- accessed by the microprocessor.
- Numerous instructions
- Clear Display, Cursor Home, Display ON/OFF, Cursor
- ON/OFF, Blink Character, Cursor Shift, Display Shift.
- Built-in reset circuit is triggered at power On.

Pin diagram

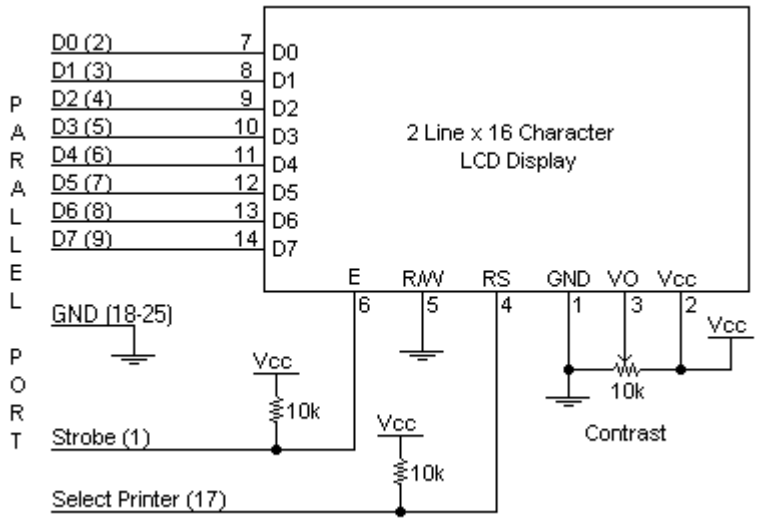


Fig 3.14 Interfacing LCD with 8952

Pins Definition

PIN	SYMBOL	FUNCTION
1	Vss	Power Supply(GND)
2	Vdd	Power Supply(+5V)
3	Vo	Contrast Adjust
4	RS	Instruction/Data Register Select
5	R/W	Data Bus Line
6	E	Enable Signal
7-14	DB0-DB7	Data Bus Line
15	A	Power Supply for LED B/L(+)
16	K	Power Supply for LED B/L(-)

In the above table VCC and VSS are supply pins and VEE (Pin no.3) is used for controlling LCD contrast. Pin No.4 is Rs pin for selecting the register, there are two very important registers are there in side the LCD. The RS pin is used for their selection as follows. If RS=0, the instruction command code register is selected, allowing the user to send data to be displayed on the LCD. R/W is a read or writes Pin, which allows the user to write information to the LCD or read information from it. R/W=1 when reading R/W=0 when writing. The LCD to latch information presented to its data pins uses the enable (E) pin. The 8-bit data pins, D0-D7, are used to send information to the LCD or read the contents of the LCD's internal registers. To display letters and numbers, we must send ASCII codes for the letters A-Z, and number 0 -9 to these pins while making RS=1.

ABSOLUTE MAXIMUM RATINGS:

1. ELECTRICAL ABSOLUTE MAXIMUM RATINGS

Table 3.13 Electrical ratings

ITEM	SYMBOL	CONDITION	MIN	MAX	UNIT
Supply Voltage (Logic)	Vdd – Vss	-	0	7.0	V
Supply Voltage (LCD Drive)	Vdd – V0	-	0	13.0	V
Input Voltage	Vi	-	-0.3	Vdd +0.3	V

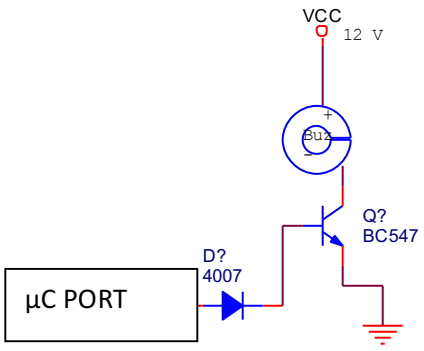
Quality control

Some LCD panels have defective transistors, causing permanently lit or unlit pixels which are commonly referred to as stuck pixels or dead pixels respectively. Unlike integrated circuits (ICs), LCD panels with a few defective pixels are usually still usable. It is also economically prohibitive to discard a panel with just a few defective pixels because LCD panels are much larger than ICs.

3.5 BUZZER

A buzzer or beeper is a signaling device, usually electronic, typically used in automobiles, household appliances such as a microwave oven, or game shows. It most commonly consists of a number of switches or sensors connected to a control unit that determines if and which button was pushed or a preset time has lapsed, and usually illuminates a light on the appropriate button or control panel, and sounds a warning in the form of a continuous or intermittent buzzing or beeping sound. Initially this device was based on an electromechanical system which was identical to an electric bell without the metal gong (which makes the ringing noise). Often these units were anchored to a wall or ceiling and used the ceiling or wall as a sounding board. Another implementation with some AC-connected devices was to implement a circuit to make the AC current into a noise loud enough to drive a loudspeaker and hook this circuit up to a cheap 8-ohm speaker. Now-a-days, it is more popular to use a ceramic-based piezo-electric sounder like a Son alert which makes a high-pitched tone. Usually these were hooked up to driver" circuits which varied the pitch of the sound or pulsed the sound on and off.

Fig 3.15 Buzzer Driver



The circuit is designed to control the buzzer. The buzzer ON and OFF is controlled by the pair of switching transistors (BC 547). The buzzer is connected in the Q2 transistor collector terminal. When high pulse signal is given to base of the Q1 transistors, the transistor is conducting and close the collector and emitter terminal so zero signals is given to base of the Q2 transistor. Hence Q2 transistor and buzzer is turned OFF state.

When low pulse is given to base of transistor Q1, the transistor is turned OFF. Now 12V is given to base of Q2 transistor so the transistor is conducting and buzzer is energized and produces the sound signal.

MICRO CONTROLLER PROGRAMMING

INTRODUCTION TO PROGRAMS

While the CPU can work only in binary, it can do so at a very high speed. For humans however it is quite tedious and slow to deal with 0s and 1s in order to program the computer. A program that consists of 0s and 1s is called machine language. Although the hexadecimal system was used as a more efficient way to represent binary numbers, the process of working in machine code was still cumbersome for humans.

Today, one can use many different programming languages, such as BASIC, PASCAL, C, C++, JAVA and numerous others. These languages are called high level languages because the programmer does not have to be concerned with the internal details of the CPU. Whereas an assembler is used to translate an Assembly Language program into machine code (sometimes also called as object code or op code for operation code), high level languages are translated into machine code by a program called as compiler. For instance, to write a program in c, one must use a C complier to translate the program into machine language.

OVERVIEW OF KEIL CROSS C COMPILER

It is possible to create the source files in a text editor such as Notepad, run the compiler on each C source file, specifying a list of controls, run the Assembler on each assembler source file, specifying another list of controls, run either the library manger or linker (again specifying list of controls) and finally running the object hex converter to convert the linker output file to an Intel hex file. Once that has been completed the hex file can be downloaded to the target hardware and debugged. Alternatively KEIL can be used to create source files: automatically compile, link and convert using option set with an easy to use at user interface and finally simulate or perform

debugging on the hardware with an access to C variables and memory. Unless having to use the tolls on the command line, the choice is clear. KEIL greatly simplifies the process of creating and testing an embedded application.

PROJECTS

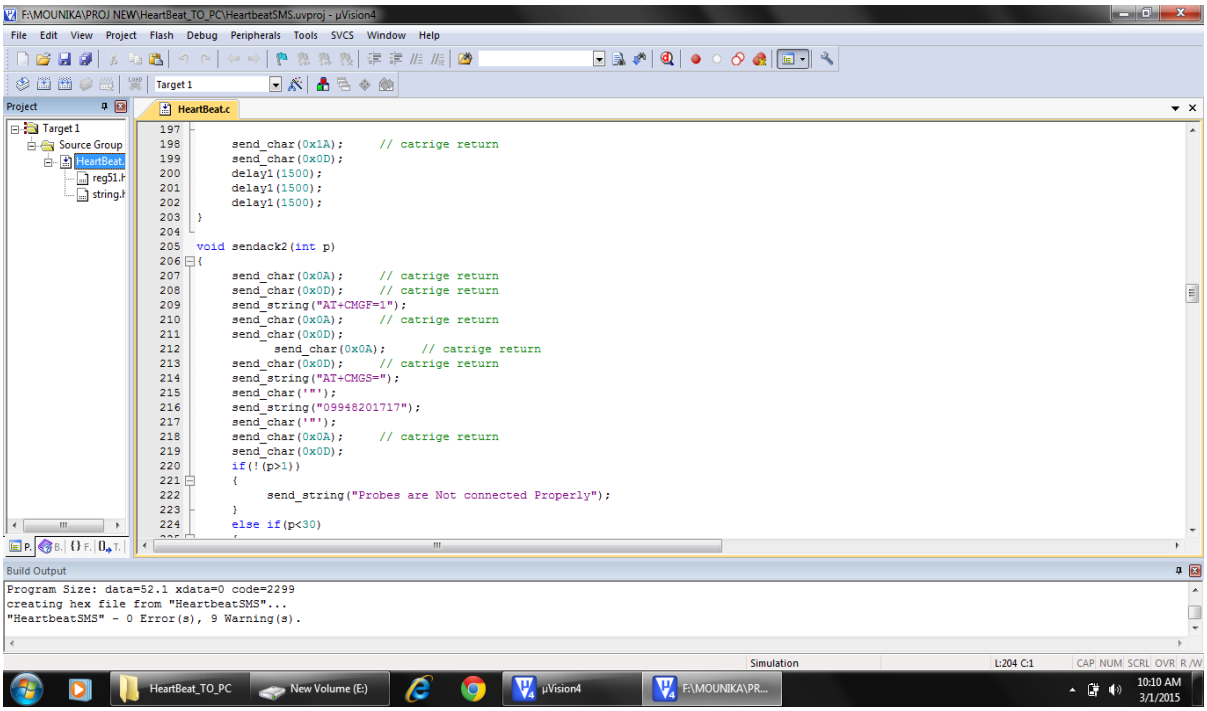
The user of KEIL centers on “projects”. A project is a list of all the source files required to build a single application, all the tool options which specify exactly how to build the application, and if required how the application will be simulated. A project contains enough information to take a set of source files and generate exactly the binary code for the application. Because of the high degree of flexibility required from the tools, there are many options that can be set to configure the tools to operate in a specific manner. It would be tedious to have to set these options up every time the application is being built; therefore they are stored in a project file. Loading the project file into KEIL informs KEIL which source files are required where they are, and how to configure the tools in correct way. KEIL can then execute each tool with correct options. It is also possible to create new projects in KEIL. Source files are added to project and the tool options are set as required. The project can then be save d to preserve settings. The project also stores such things as which windows were left open in the debugger/simulator, so when a project is reloaded and the simulator/debugger started, all the desired windows are opened. KEIL project files have the extension.

SIMULATOR/DEBUGGER

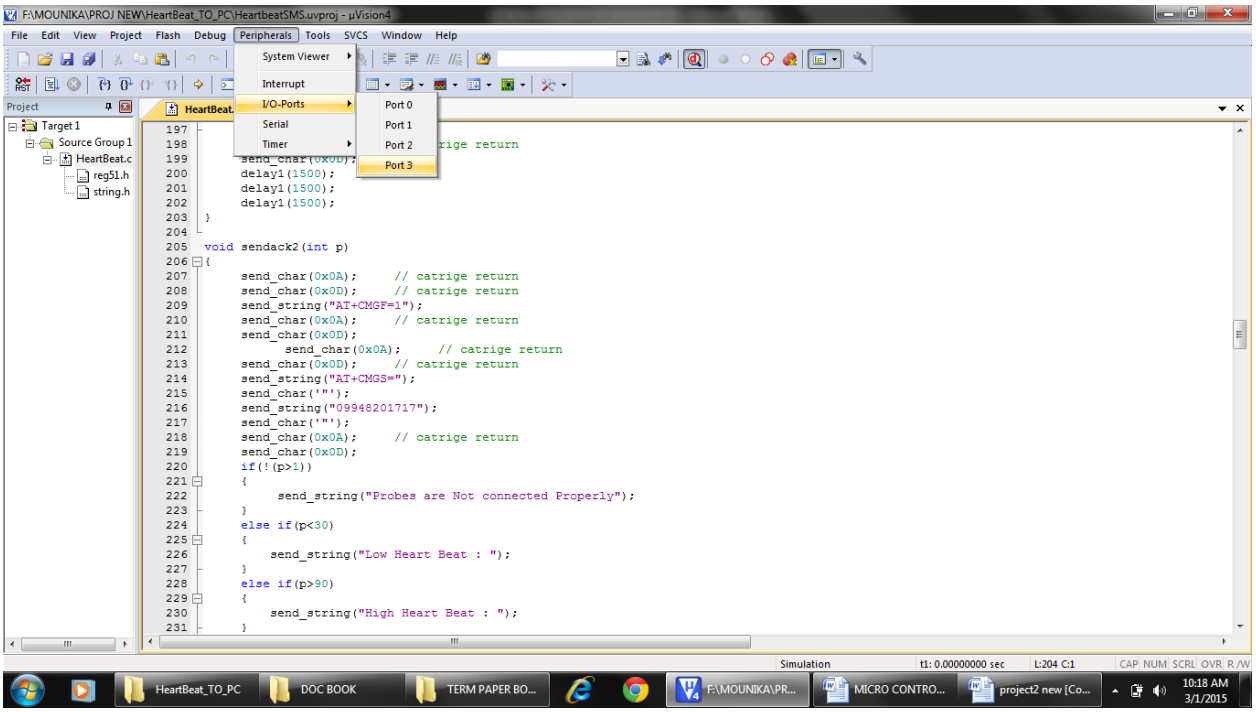
The simulator or debugger in KEIL can perform a very detail simulation of a microcontroller along with external signals. It is possible to view the precise execution time of a single assembly instruction, or a single line of c code, all the way up to the entire application, simply by entering the crystal frequency. A window can be opened for each peripheral on the

device, showing the state of the peripheral. This enables quick trouble shooting of mis configured peripherals. Break points may be set on either assembly instructions or lines of c code, and execution may be stepped through one instruction or c line at a time. The contents of all the memory areas may be viewed along with ability to find specific variables. In addition the registers may be viewed along the detail view of what the microcontroller is doing at any point of time. Entered the programming .C format and save it. Now press function key F7 to compile. Any error will appear so happen.

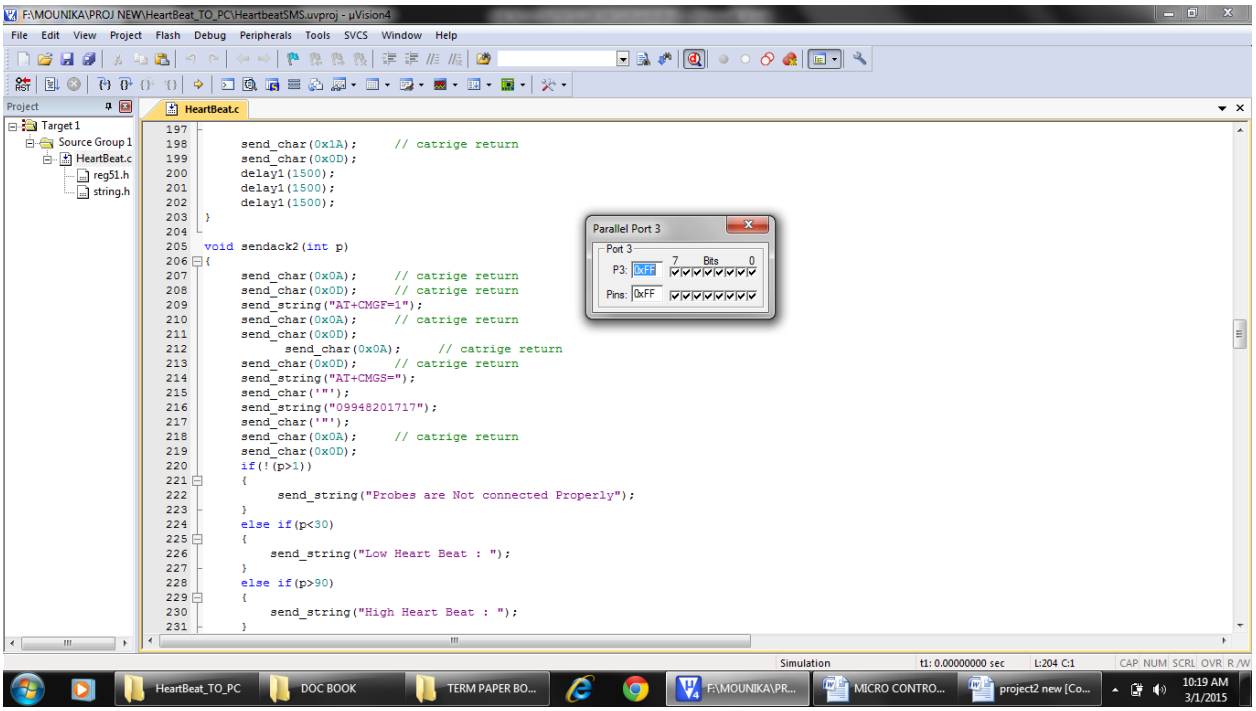
Now select as per your file extension given while saving the file. Click only one time on option ADD. Now press function key F7 to compile. Any error will appear so happen.



If the file contains no error, then press ctrl + F5 simultaneously. Now click on the peripherals from menu bar and check your required board as in the window below.



Drag the port aside and click in the program file.



Now keep pressing function key F11 and observe. Then the program is running successfully.

DATA TYPES IN EMBEDDED C

Some of the widely used data types for embedded c programming are shown in table .1

DATA TYPES	SIZE IN BITS	DATA RANGE/USAGE
Unsigned char	8 bit	0 to255
Signed char	8 bit	-128 to +127
Unsigned int	16 bit	0 to 65535
Signed int	16 bit	-32768 to +32767
S bit	1 bit	SFR bit addressable only
Bit	1 bit	RAM bit addressable memory
SFR	8 bit	RAM addresses 80 - FF H only

Table 4.1 data types

UNSIGNED CHAR

The unsigned char is an 8 bit data type the takes the value in the range of 0 to 255(0 to FFH). It is used in many situations such as setting a counter value where there is no need for signed data so used the unsigned char instead of the signed char. Remember that c compilers use the signed char as default.

SIGNED CHAR

The signed char is in 8 bit data type that uses most significant bit (D7 of D7 to D0) to represent the – or + values. As a result there are only 7 bits for the magnitude of the signed number giving us values from -128 to +127 in situations where + or – are needed to represent a given quantity such as a temperature, the use of the signed char data type is a must.

UNSIGNED INT

The unsigned int is a 16 bit data type that takes a value in the range of 0 to 65535(0000-FFFFH). It is also used to set counter values of more than 256. Use the int data type unless it has to. Since registers and memory are in 8 bit chunks, the misuse of int variables will result in a large hex file. To overcome this, use the unsigned char in place of unsigned int.

SIGNED INT

Signed int is a 16 bit data type that uses the most significant bit (D15 of D15-D0) to represent the – or + value. As a result there are only 15 bits for the magnitude of the number or values from -32768 to +32767.

S BIT (SINGLE BIT)

The s bit data type is widely used and designed specifically to access single bit addressable registers. It allows access to the single bits of the SFR registers.

CHAPTER 4

CONCLUSION

5. CONCLUSION

The project “RFID BASED RAILEAY PLATFORM TO IDENTIFY THE EXACT POSITION OF COACH” has been successfully designed and tested. It has been developed by integrating features of all the hardware components used. Presence of every module has been reasoned out and placed carefully thus contributing to the best working of the unit.

Secondly, using highly advanced IC’s and with the help of growing technology the project has been successfully implemented.

The field of RFID has created a large class with physical and navigational competencies. At the same time, society has begun to move towards incorporating RFID into everyday life. Many of the applications are being pursued by the RFID researches at present. Regardless of applications,

This will require more rather than less intelligence, and will there by have a significant impact on our society in the future as a technology expands new horizons.

Finally conclude that “RFID BASED RAILEAY PLATFORM TO IDENTIFY THE EXACT POSITION OF COACH” is an emerging field and there is a huge scope for research and development

CHAPTER 5

BIBLIOGRAPHY

5. BIBLIOGRAPHY

1. The 8051 Microcontroller Architecture, Programming and Applications by Kenneth J Ayala.

2. Fundamentals Of Digital Signal Processing By Robert. J. Schilling and Sandra. L .Harris, Thomson

3. MCS51 series authorized manual.

4. B.Ram “fundamentals of microprocessor and microcomputer”

4. Ramesh S.Gaonakar “Microprocessor architecture, programming& applications”