Soil and Environment Testing Equipment

Project Report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

By

Ankit Gupta-061021

Nihit Baluni-061080

under the Supervision of

Ms. Vanita Rana

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

May 2010

Jaypee University of Information Technology

Waknaghat, Solan - 173 215, Himachal Pradesh

Certificate

This is to certify that the project report entitled "Soil and Environment Testing Equiment", submitted by **Ankit Gupta**(061021) and **Nihit Baluni** (061080) in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

Signature

Date :

(Ms. Vanita Rana)

Lecturer ECE

Certified that this work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma

Signature

1. (Ankit Gupta)
Roll No. 061021

Signature

2. (Nihit Baluni)
Roll No. 061080

ACKNOWLEDGEMENT

First of all, with profound veneration and humble submission, I would like to express my sense of gratitude to **Shri S.V. Bhooshan,** HOD (ECE department), Jaypee University of Information Technology, Solan, for giving us the opportunity to work on this project and providing us valuable guidance.

We would also like to express our appreciation to our project guide **Ms. Vanita Rana**. We wholeheartedly thank her for all her support and help. We feel motivated and encouraged every time we discuss our tribulations to her. Without her encouragement and guidance this project would not have materialized.

A special thanks to all members of lab staff specially Mr. Pramod , Lab in charge, microprocessor lab who helped us and cooperated with us throughout the project.

Date:

Names of the students

Ankit Gupta (061021)

Nihit Baluni (061080)

CONTENTS

List of Figures

List of Tables

ABSTRACT

Agriculture is the main stay of our economy. It is the source of income for more than 70% of our population directly or indirectly. Therefore it is important to enhance the productivity and quality of crops grown. Our project "Soil Testing Equipment" is also based on the same idea. This novel work would help more than four billion people of our country.

Soil testing equipment is a low cost microcontroller based device which displays various parameters of the soil and its respective environment. These parameters include moisture & pH of the soil and temperature & altitude of that area. Using these parameters one can know the current conditions of soil and surrounding environment and hence decide the best crop, fruit etc...for that region. This will help farmers to know the state of their soil and hence they can act accordingly to bring their soil to optimum level. This can also be used to see the shortcomings of unused arid land and hence can be used for cultivation.

This project emphasizes on low cost and this has been achieved by us by using easily available low cost components like microcontroller, ADC etc. Hence, this equipment if made commercial, will be available at low price and would therefore be easily affordable for our farmers.

CHAPTER 1

INTRODUCTION

We live in a world where everything can be controlled and operated automatically, but there are still a few important sectors in our country where automation has not been adopted or not been put to a full-fledged use, perhaps because of several reasons one such reason is cost. One such field is that of agriculture. Agriculture has been one of the primary occupations of man since early civilizations and even today manual interventions in farming are inevitable.

Due to varying atmospheric and other environmental conditions, panorama of many flora species changes. Altering temperature conditions, change in pH of soil and many other factors, the crops which were earlier suitable to grow are not proper for the present circumstances.

Soil testing equipment is a device which has been made to provide low cost soil testing to farmers or growers. The main aim of this project is to know the properties of soil and its corresponding environment so that the farmers may know the best crop for them to grow on that field which gives higher yield than other crops. This testing can also be used to know the shortcomings of the soil and hence steps can be taken to improve the same.

This device consists of 4 sensors namely temperature sensor, moisture sensor, pH sensor and pressure sensor which takes the readings from soil. Each sensor is selected one by one by using select pins to take the readings. The sensors give analog output which is amplified and then digitised into 8 bit data by analog to digital converter. Once digitised, these readings are fed to the microcontroller for further processing. The input is calibrated accordingly via program burned in the microcontroller. Once the input is processed, the corresponding output is generated and displayed on the LCD.

The main aim of the project was to identify these conditions and hence it is very *important to correctly identify the parameters that are going to be measured by the* controller's data acquisition interface, and how they are to be measured. The set of variables typically used in greenhouse control is shown below:

Sl. No.	Variable to be monitored	Its Importance
1	Temperature	Affects all plant metabolic functions.
2	Pressure	Affects growth, photosynthesis as well as transpiration.
3	Soil moisture	Affects salinity, and pH of irrigation water
4	pH	Affects acidity or basicity of soil.

Table1.1 Importance of the various parameters

CHAPTER 2

SYSTEM MODEL

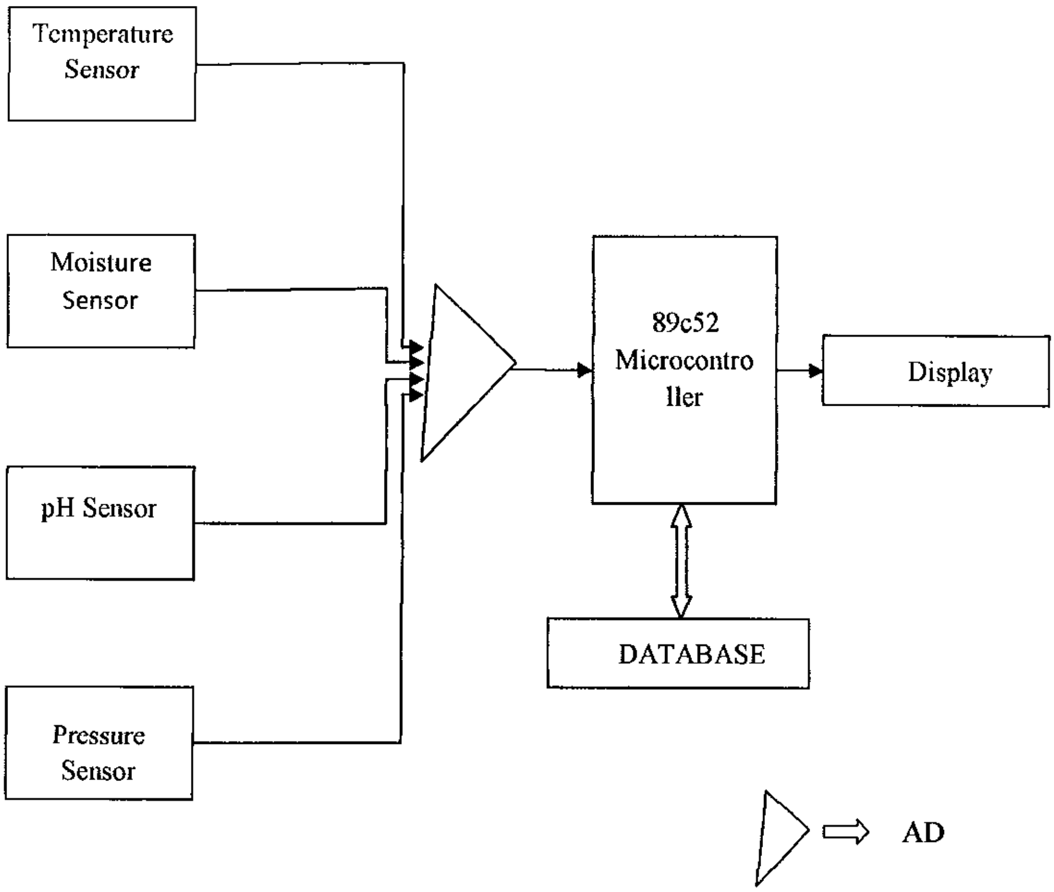


Fig 2.1 Block diagram of the system

2.1 PARTS OF THE SYSTEM:

- Sensors (Data acquisition system)

  Temperature sensor (LM35)

  Pressure sensor (CS 100)

  Moisture sensor (Cybersen)

  pH Sensor (PHE 45P)

- Analog to Digital Converter (ADC 0809)
- Microcontroller (AT89C52)
- Liquid Crystal Display (JHD162A)

2.1.1 TRANSDUCERS (Data acquisition system):

This part of the system consists of various sensors, namely soil moisture, humidity, temperature and pressure. These sensors sense various parameters- temperature, humidity, soil moisture and altitude and are then sent to the Analog to Digital Converter.

2.1.2 ANALOG TO DIGITAL CONVERDER (ADC):

The analog parameters measured by the sensors are then converted to corresponding digital values by the ADC.

2.1.3 MICROCONTROLLER:

The microcontroller is the heart of the proposed embedded system. It constantly monitors the digitized parameters of the various sensors and verifies them with the predefined threshold values. These values are then converted into ASCII and send to the display unit.

2.1.4 DISPLAY UNIT:

A Liquid crystal display is used to indicate the present status of parameters. It is a 2 line display in which the first line displays the sensor name and the second line displays the corresponding reading.

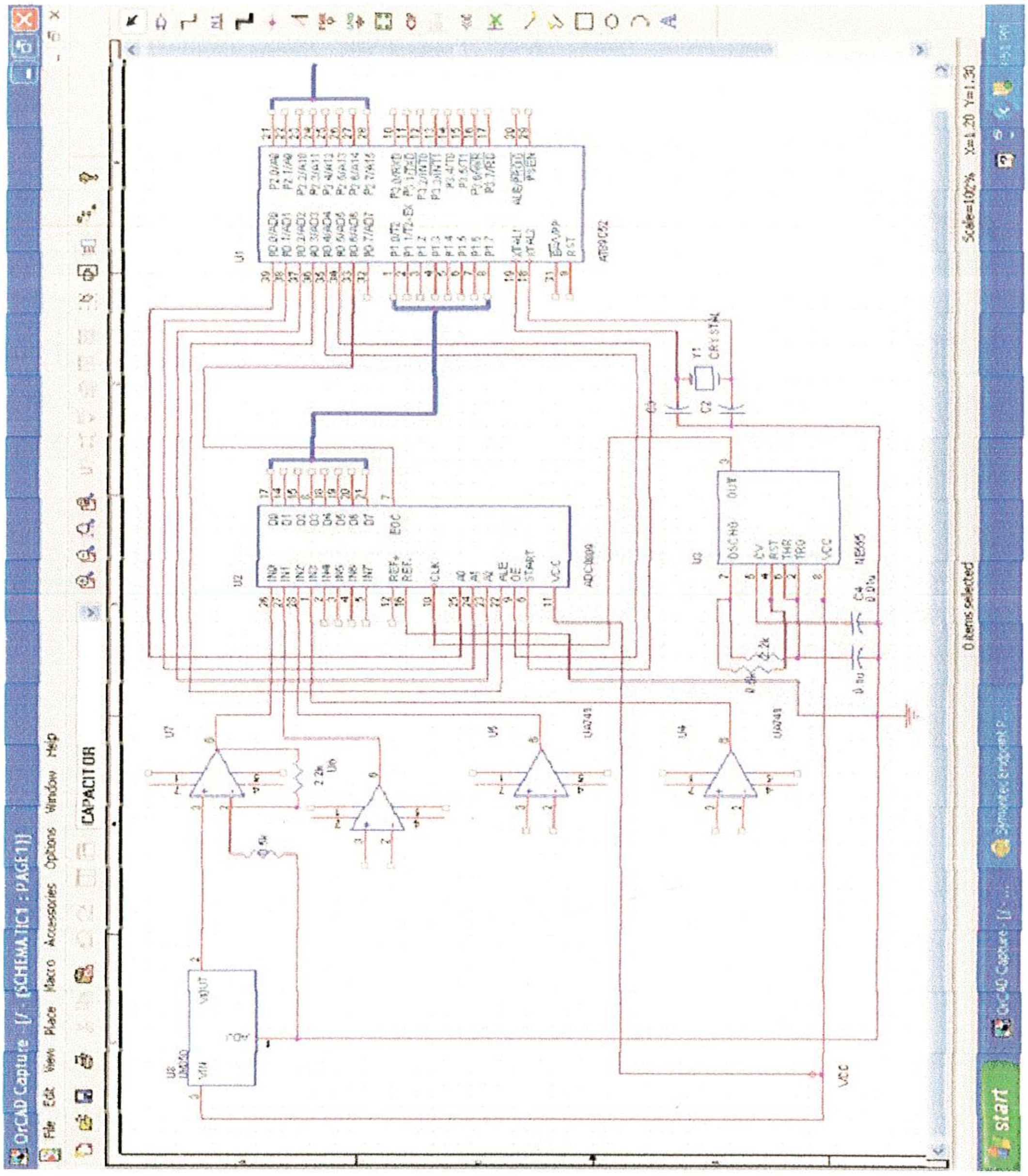
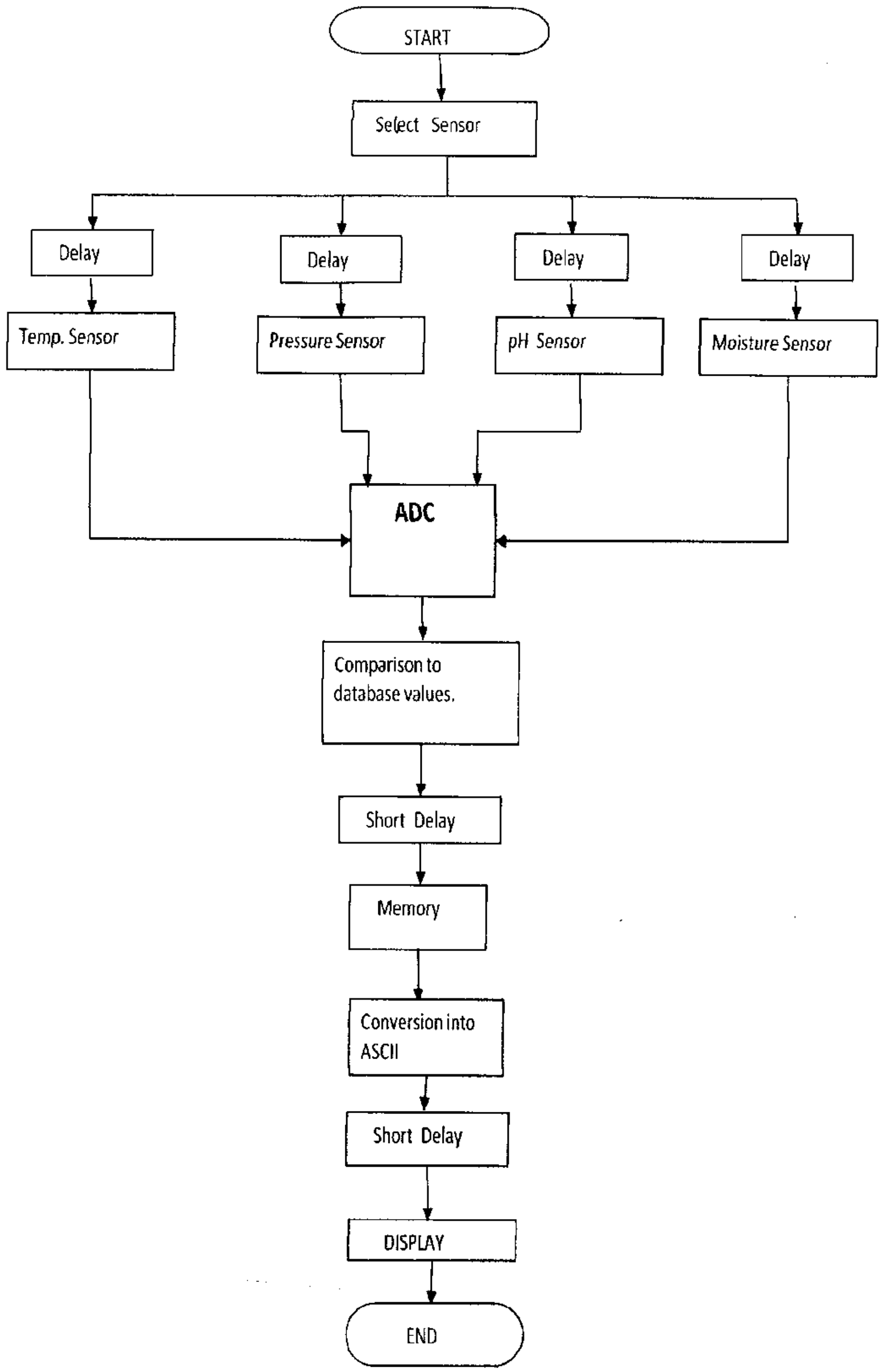


Fig 2.2 Schematic Diagram

Fig 2.3 FLOWCHART REPRESENTING THE WORKING OF THE SYSTEM



CHAPTER 3

HARDWARE DESCRIPTION

3.1 TRANSDUCERS (sensors)

A transducer is a device which measures a physical quantity and converts it into a signal which can be read by an observer or by an instrument. The parameters which are of importance are the temperature which affect the photosynthetic and transpiration processes, moisture content in the soil, atmospheric pressure, pH etc.

The sensors used in this system are:

- Pressure Sensor
- Moisture Sensor
- Temperature Sensor
- pH sensor

3.1.1 TEMPERATURE SENSOR

National Semiconductor's *LM35 IC* has been used for sensing the temperature. It is an integrated circuit sensor that can be used to measure temperature with an electrical output proportional to the temperature (in ^{o}C). The temperature can be measured more accurately with it than using a thermistor. The sensor circuitry is sealed and not subject to oxidation, etc.

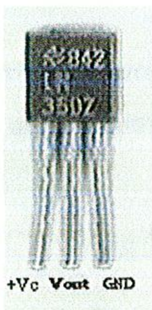


Fig. 3.1 LM35 temperature sensor

Features:

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteed (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 μA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±1⁄4°C typical

Functional description:

- The sensor has a sensitivity of 10mV / °C.
- The output of LM35 is amplified using a LM324 single power supply (+5V) op-amp.
- The op-amp is designed to have a gain of 5.
- *The circuitry measures temperatures with a resolution of up to 0.5 degree* Celsius.
- The output voltage is converted to temperature by a simple conversion factor. *The general equation used to convert output voltage to temperature* is:

  $$\text{Temperature } (^{\circ}C) = (V_{out} * 100) / 5\ ^{\circ}C$$

  So if V_{out} is 5V, then, Temperature = 100 °C

- The output voltage varies linearly with temperature

3.1.2 SOIL MOISTURE SENSOR

Features of the Soil moisture sensor:

1. The circuit designed uses a 5V supply, fixed resistance of 100Ω, variable resistance of 10KΩ, two copper leads as the sensor probes, 2N222N transistor.
2. It gives a voltage output corresponding to the conductivity of the soil.
3. The conductivity of soil depends upon the amount of moisture present in it. It increases with increase in the water content of the soil.
4. The voltage output is taken at the transmitter which is connected to variable resistance. This variable resistance is used to adjust the sensitivity of the sensor

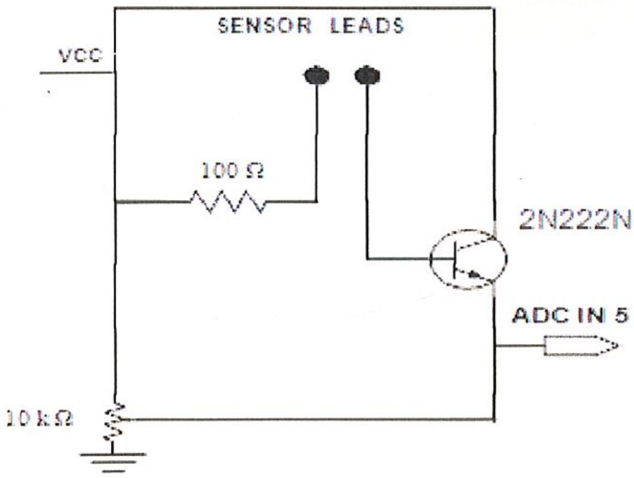


Fig. 3.2 Soil Moisture Sensor

3.2 ANALOG TO DIGITAL CONVERTER (ADC 0809)

In physical world parameters such as temperature, pressure, humidity, and velocity are analog signals. A physical quantity is converted into electrical signals. We need an analog to digital converter (ADC), which is an electronic circuit that converts continuous signals into discrete form so that the microcontroller can read the data. Analog to digital converters are the most widely used devices for data

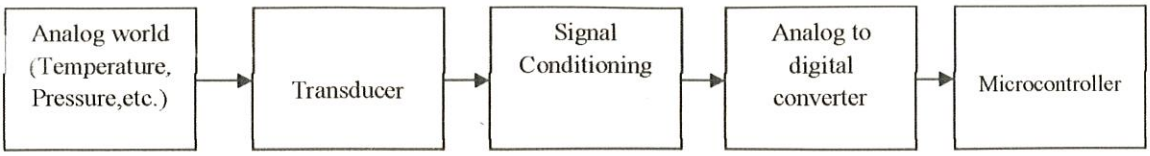


Fig3.3 Getting data from analog world

3.2.1 DESCRIPTION

The ADC0809 data acquisition component is a monolithic CMOS device with an 8- bit analog-to-digital converter, 8-channel multiplexer and microprocessor compatible control logic. The 8-bit A/D converter uses successive approximation as the conversion technique. The converter features a high impedance chopper stabilized comparator, a 256R voltage divider with analog switch tree and a successive approximation register. The 8-channel multiplexer can directly access any of 8-single-ended analog signals.

The design of the ADC0809 has been optimized by incorporating the most desirable aspects of several A/D conversion techniques. The device offers high speed, high accuracy, minimal temperature dependence, excellent long-term accuracy and repeatability, and consumes minimal power. These features make it ideally suited for applications from process and machine control to consumer and automotive applications

3.2.2 FEATURES

1. Easy interface to all microcontrollers.
2. Operates ratio metrically or with 5 VDC or analog span adjusted voltage reference.
3. No zero or full-scale adjust required.
4. 8-channel multiplexer with address logic.
5. 0V to 5V input range with single 5V power supply
6. Outputs meet TTL voltage level specifications.
7. 28-pin molded chip carrier package.

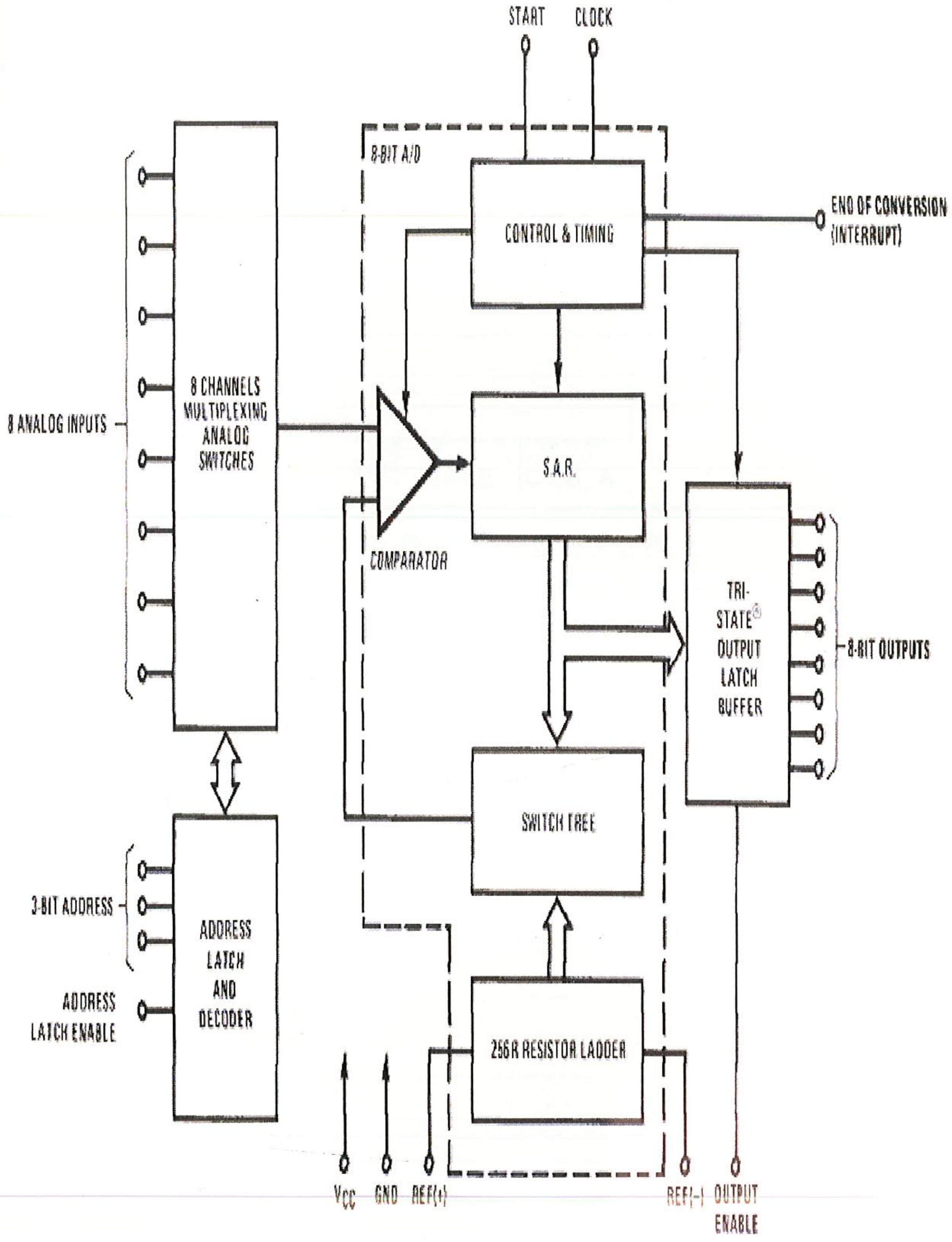


Fig 3.4 Block diagram of ADC 0809

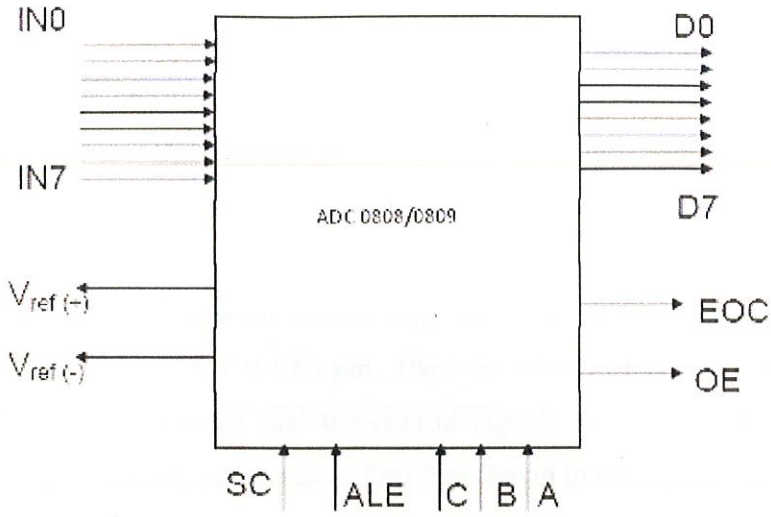


Fig 3.5 Pin diagram of ADC 0809

We use A, B, C addresses to select IN0-IN7 and activate Address latch enable (ALE) to latch in the address. SC is for Start Conversion. EOC is for End of Conversion and OE is for Output Enable. The output pins D0-D7 provides the digital output from the chip. Vref (-) and Vref (+) are the reference voltages.

3.2.3 SELECTING AN ANALOG CHANNEL

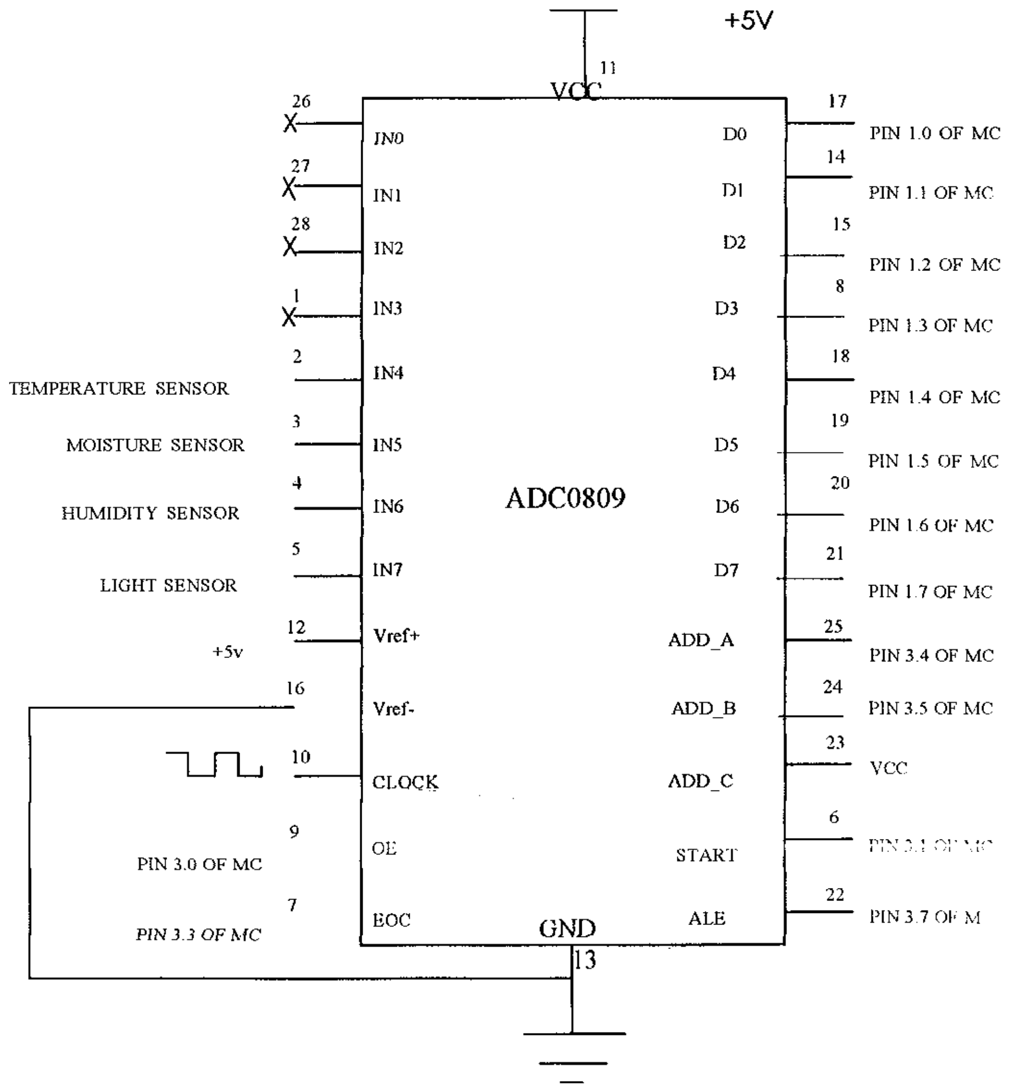
How *to select the channel using three address pins* A, B, C *is shown in* Table below:

Select Analog Channel	C	B	A
IN0	0	0	0
IN1	0	0	1
IN2	0	1	0
IN3	0	1	1
IN4	1	0	0
IN5	1	0	1
IN6	1	1	0
IN7	1	1	1

Table 3.1 Selection of the input channels

The ADC 0804 is most widely used chip, but since it has only one analog input, ADC 0809 is chosen as this chip allows the monitoring of up to 8 different transducers using only a single chip. The 8 analog input channels are multiplexed and selected according to the requirement. But for the proposed application only the last 4 channels i.e., IN4, IN5, IN6 and IN7 are used to monitor the four parameters-temperature, humidity, soil moisture and light intensity. Hence the address line ADD_C is given to Vcc (+ 5V) as it is always high in this case. V_{ref} (+) and V_{ref} (-) set the reference voltages. If V_{ref} (-) =Gnd and V_{ref} (+) =5V, the step size is 5V/256=19.53.

Since there is no self clocking in this chip, the clock must be provided from an external source to the Clock (CLK) pin. The 8-bit output from the ADC is given to Port 0 of the microcontroller and the control signals ADD_A, ADD_B, ADD_C, ALE, START, OE, EOC are given to Port 1 as shown in the figure below.



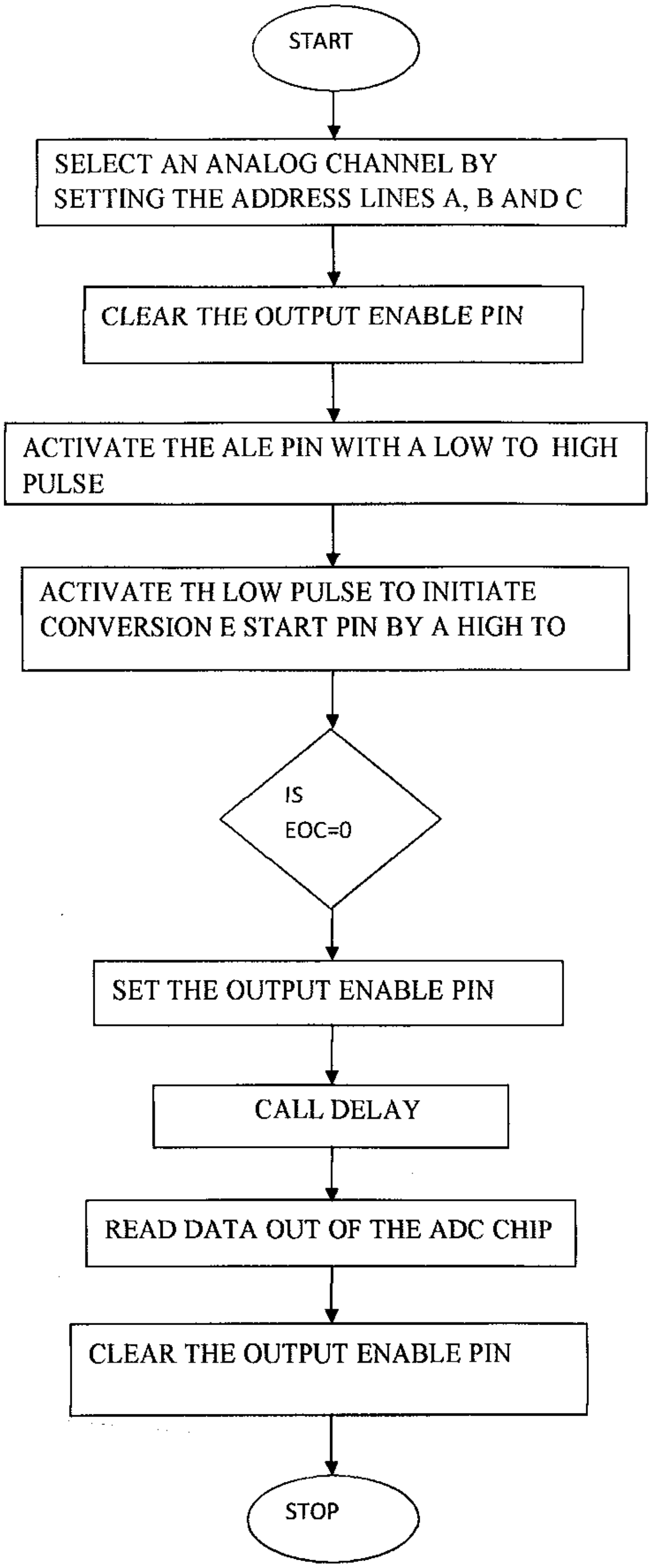


Fig3.7 FLOWCHART FOR ADC INITIALIZATION

At a certain point of time, even though there is no conversion in progress the ADC0809 is still internally cycling through 8 clock periods. A start pulse can occur any time during this cycle but the conversion will not actually begin until the converter internally cycles to the beginning of the next 8 clock period sequence. As long as the start pin is held high no conversion begins, but when the start pin is taken low the conversion will start within 8 clock periods. The EOC output is triggered on the rising edge of the start pulse. It, too, is controlled by the 8 clock period cycle, so it will go low within 8 clock periods of the rising edge of the start pulse. One can see that it is entirely possible for EOC to go low before the conversion starts internally, but this is not important, since the positive transition of EOC, which occurs at the end of a conversion, is what the control logic is looking for. Once EOC does go high this signals the interface logic that the data resulting from the conversion is ready to be read. The output enable (OE) is then raised high.

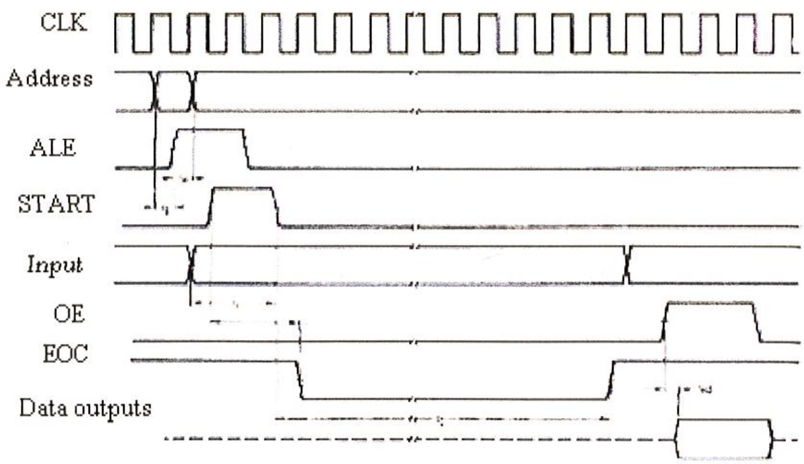


Fig 3.8 Timing diagram of ADC 0809

Depending on the manufacturer, the standard 555 package includes over 20 transistors, 2 diodes and 15 resistors on a silicon chip installed in an 8-pin mini dual-in-line package (DIP-8). Variants available include the 556 (a 14-pin DIP combining two 555s on one chip), and the 558 (a 16-pin DIP combining four slightly modified 555s with DIS & THR connected internally, and TR falling edge sensitive instead of level sensitive). Ultra-low power versions of the 555 are also available, such as the 7555 and

TLC555. The 7555 requires slightly different wiring using fewer external components and less power.

The 555 has three operating modes:

- **Monostable mode**: in this mode, the 555 functions as a "one-shot". Applications include timers, missing pulse detection, bouncefree switches, touch switches, frequency divider, capacitance measurement, pulse-width modulation (PWM) etc
- **Astable** - free running mode: the 555 can operate as an oscillator. Uses include LED and lamp flashers, pulse generation, logic clocks, tone generation, security alarms, pulse position modulation, etc.
- **Bistable mode** or Schmitt trigger: the 555 can operate as a flip-flop, if the DIS pin is not connected and no capacitor is used. Uses include bouncefree latched switches, etc.

3.3 MICROCONTROLLER (AT89C52)

3.3.1 CRITERIA FOR CHOOSING A MICROCONTROLLER

The basic criteria for choosing a microcontroller suitable for the application are:

1) The first and foremost criterion is that it must meet the task at hand efficiently and cost effectively. In analyzing the needs of a microcontroller-based project, it is seen whether an 8- bit, 16-bit or 32-bit microcontroller can best handle the computing needs of the task most effectively. Among the other considerations in this category are:

(a) **Speed:** The highest speed that the microcontroller supports.

(b) **Packaging:** It may be a 40-pin DIP (dual inline package) or a QFP (quad flat package), or some other packaging format. This is important in terms of space, assembling, and prototyping the end product.

(c) **Power consumption:** This is especially critical for battery-powered products.

(d) The number of I/O pins and the timer on the chip.

(f) How easy it is to upgrade to higher –performance or lower consumption versions.

(g) ***Cost per unit***: This is important in terms of the final cost of the product in which a microcontroller is used.

2) The second criterion in choosing a microcontroller is how easy it is to develop products around it. Key considerations include the availability of an assembler, debugger, compiler, technical support.

3) The third criterion in choosing a microcontroller is its ready availability in needed quantities both now and in the future. Currently of the leading 8-bit microcontrollers, the 8051 family has the largest number of diversified suppliers. By supplier is meant a producer besides the originator of the microcontroller. In the case of the 8051, this has originated by Intel several companies also currently producing the 8051. Thus the microcontroller AT89C52, satisfying the criterion necessary for the proposed application is chosen for the task.

3.3.2 DESCRIPTION

The 8051 family of microcontrollers is based on an architecture which is highly optimized for embedded control systems. It is used in a wide variety of applications from military equipment to automobiles to the keyboard. Second only to the Motorola 68HC11 in eight bit processors sales, the 8051 family of microcontrollers is available in a wide array of variations from manufacturers such as Intel, Philips, and Siemens. These manufacturers have added numerous features and peripherals to the 8051 such as I2C interfaces, analog to digital converters, watchdog timers, and pulse width modulated outputs. Variations of the 8051 with clock speeds up to 40MHz and voltage requirements down to 1.5 volts are available. This wide range of parts based on one core makes the 8051 family an excellent choice as the base architecture for a company's entire line of products since it can perform many functions and developers will only have to learn this one platform.

The AT89C52 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 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 in-system programmable Flash on a monolithic chip, the Atmel AT89C52 is a powerful microcontroller which provides a highly-flexible and cost- effective solution to many embedded control applications. 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 con-tents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

3.3.3 FEATURES:

The basic architecture of AT89C51 consists of the following features:

- Compatible with MCS-51 Products
- 8K Bytes of In-System Programmable (ISP) Flash Memory
- 4.0V to 5.5V Operating Range
- 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
- Full Duplex UART Serial Channel
- Low-power Idle and Power-down Modes

- Interrupt Recovery from Power-down Mode
- Watchdog Timer
- Fast Programming Time

PIN CONFIGURATION:

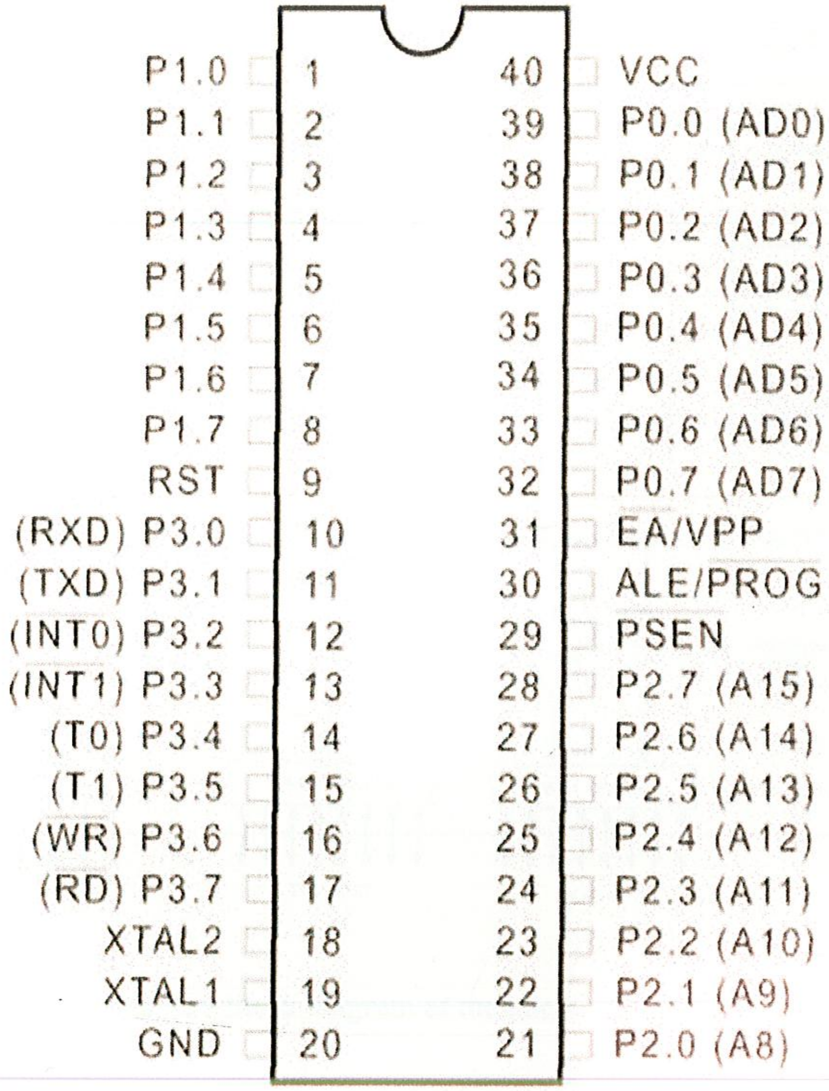


Fig 3.9 Pin diagram of AT89C52

BLOCK DIAGRAM

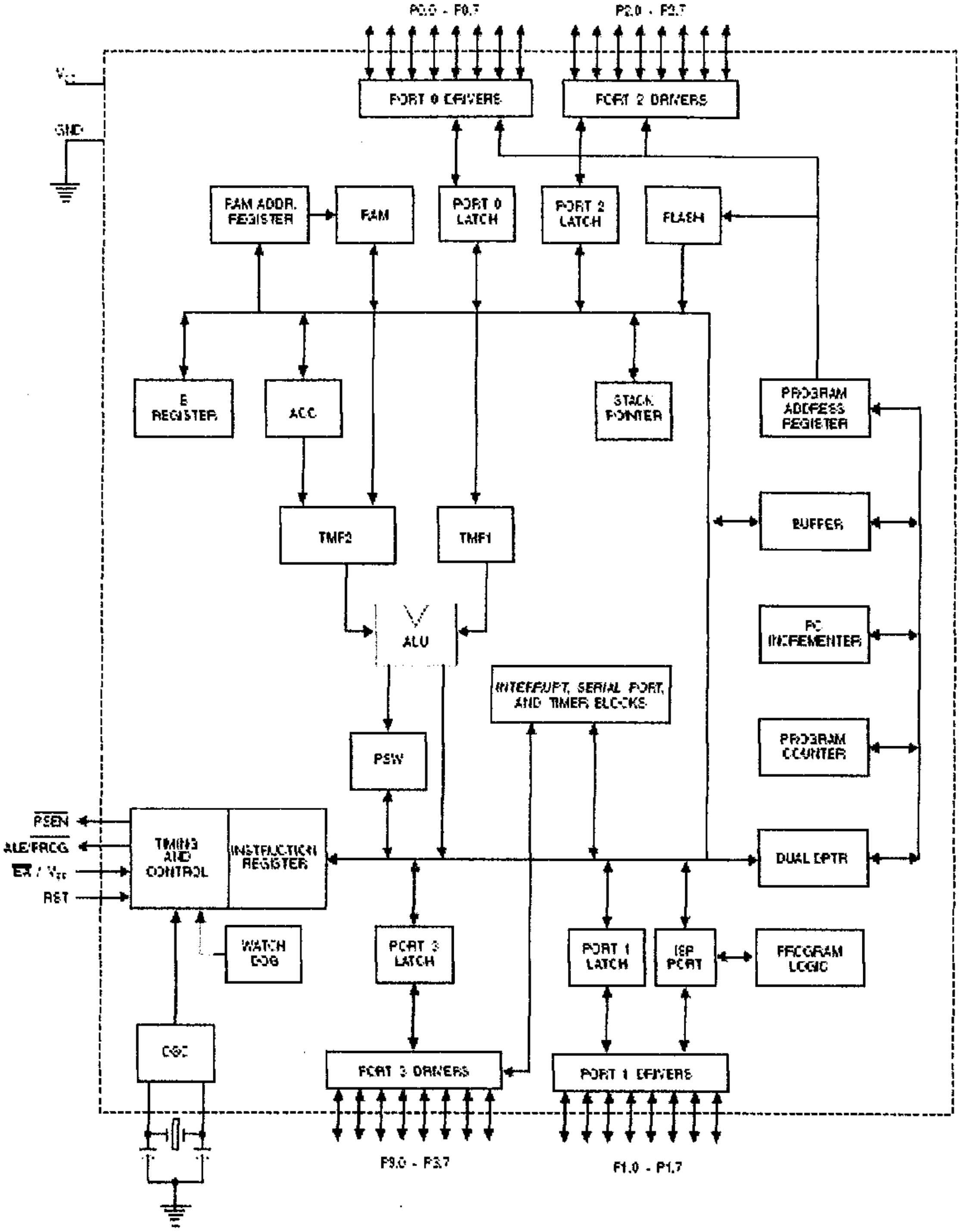


Fig 3.10 Block diagram of the microcontroller

3.3.4 PIN DESCRIPTION

- **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 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), respectively, as shown in the following table.

- **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 @ DPTR). In this application, Port 2 uses strong internal pull- ups when emitting 1s. During accesses to external data memory that uses 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function register.

- **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 receives some control signals for Flash programming an verification Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table.

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

Table3.2 Alternate functions of Port 3

- **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 98 oscillator periods after the watchdog times out.

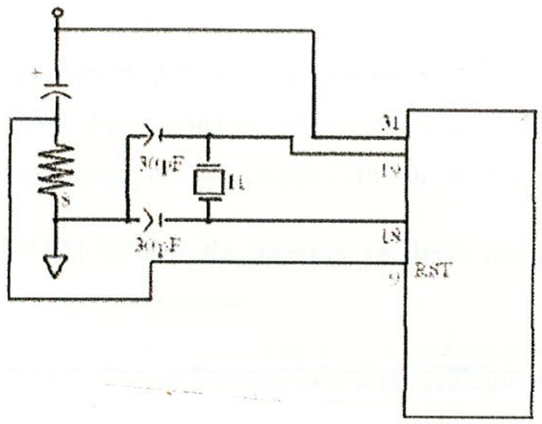


Fig3.11 Power-on reset circuit

In order for the RESET input to be effective, it must have a minimum duration of two *machine cycles.*

- **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:** 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.

The AT89S52 *oscillator clock circuit*

- It uses a quartz crystal oscillator.
- We can observe the frequency on the XTAL2 pin.

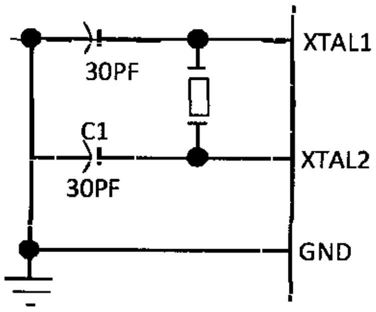


Fig 3.12 AT89S52 oscillator clock circuit

The AT89S52 oscillator clock circuit

- The crystal frequency is the basic internal frequency of the microcontroller.

- The internal counters must divide the basic clock rate to yield standard communication bit per second (baud) rates.
- An 11.0592 megahertz crystal, although seemingly an odd value, yields a crystal
  frequency of 921.6 kilohertz, which can be divided evenly by the standard communication baud rates of 19200, 9600, 4800, 2400, 1200, and 300 hertz.

3.3.5 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) contain memory locations that are used for special tasks. Each SFR occupies internal RAM from 0x80 to 0xFF.They are 8-bits wide.

- The A (accumulator) register or accumulator is used for most ALU operations and Boolean Bit manipulations.

- Register B is used for multiplication & division and can also be used for general purpose storage.

- PSW (Program Status Word) is a bit addressable register

- PC or program counter is a special 16-bit register. It is not part of SFR. Program instruction bytes are fetched from locations in memory that are addressed by the PC.

- Stack Pointer (SP) register is eight bits wide. It is incremented before data is stored during PUSH and CALL executions. While the stack may reside anywhere in on-chip RAM, the Stack Pointer is initialized to 07H after a reset. This causes the stack to begin at location 08H.

- DPTR or data pointer is a special 16-bit register that is accessible as two 8- bit registers: DPL and DPH, which are used to used to furnish memory addresses for internal and external code access and external data access.

- Control Registers: Special Function Registers IP, IE, TMOD, TCON, SCON, and PCONcontain control and statusbits for the interrupt system, the Timer/Counters, and the serial port.

- Timer Registers: Register pairs (TH0, TL0) and (TH1, TL1) are the 16-bit Counter registers for Timer/Counters 0 and 1, respectively.

3.3.6 MEMORY ORGANIZATION

MCS-51 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

- **Program Memory:** If the EA pin is connected to GND, all program fetches are directed to external memory. On the AT89S52, if EA is connected to VCC, program fetches to addresses 0000H through 1FFFH are directed to internal memory and fetches to addresses 2000H through FFFFH are to external memory.

- **Data Memory:** The AT89S52 implements 256 bytes of on-chip RAM.

The upper 128 bytes occupy a parallel address space to the Special Function Registers. This means that the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space. When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions which use direct addressing access the SFR space. The lower 128 bytes of RAM can be divided into three segments:

1. Register Banks 0-3: locations 00H through 1FH (32 bytes). The device after reset defaults to register bank 0. To use the other register banks, the user must select them in software. Each register bank contains eight 1-byte *registers R0-R7. Reset initializes the stack point to location 07H, and is* incremented once to start from 08H, which is the first register of the second register bank.

2. Bit Addressable Area: 16 bytes have been assigned for this segment 20H-2FH. Each one of the 128 bits of this segment can be directly addressed (0-7FH). Each of the 16 bytes in this segment can also be addressed as a byte.

3. Scratch Pad Area: 30H-7FH are available to the user as data RAM. *However, if the data pointer has been initialized to this area, enough bytes* should be left aside to prevent SP data destruction.

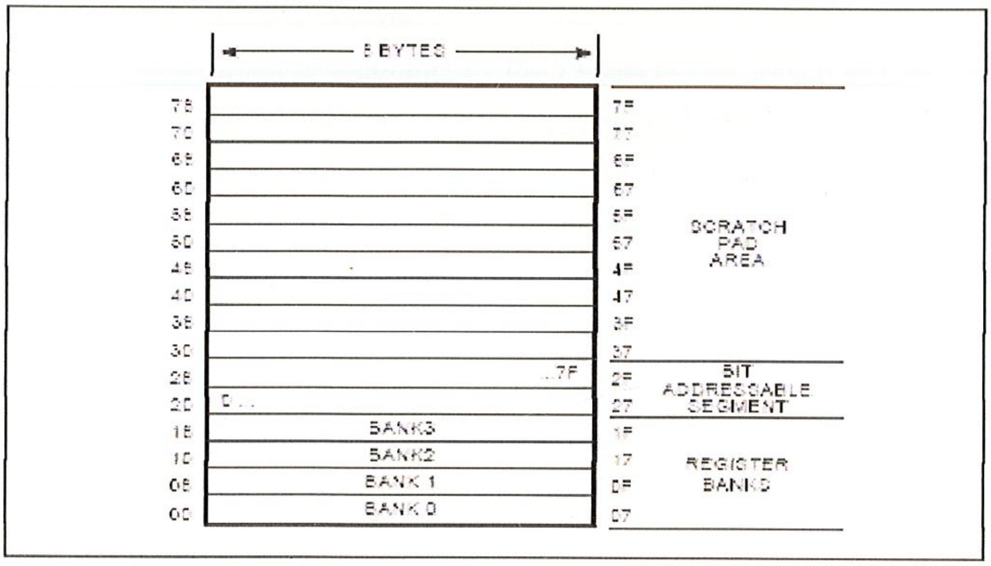


Fig 3.13 Internal memory block

3.3.7 WATCHDOG TIMER (One-time Enabled with Reset-out)

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets. The WDT consists of a 14-bit counter and the Watchdog Timer Reset (WDTRST) SFR. The WDT is defaulted to disable from exiting reset. To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, it will increment every machine cycle while the oscillator is running. The WDT timeout period is dependent on the external clock frequency. There is no way to disable the

WDT except through reset (either hardware reset or WDT overflow reset). When WDT over-flows, it will drive an output RESET HIGH pulse at the RST pin.

3.3.8 TIMERS AND COUNTERS

Many microcontroller applications require the counting of external events such as the frequency of a pulse train, or the generation of precise internal time delays between computer actions. Both of these tasks can be accomplished using software techniques, but software loops for counting or timing keep the processor occupied so that, other perhaps more important, functions are not done. Hence the better option is to use interrupts & the two 16- bit count- up timers. The microcontroller can programmed for either of the following:

1. Count internal - acting as timer
2. Count external - acting as counter

All counter action is controlled by the TMOD (Timer Mode) and the TCON (Timer/Counter Control) registers. TCON Timer control SFR contains timer 1& 2 overflow flags, external interrupt flags, timer control bits, falling edge/low level selector bit etc. TMOD timer mode SFR comprises two four-bit registers (timer #1, timer #0) used to specify the timer/counter mode and operation. The timer may operate in any one of four modes that are determined by modes bits M1 and M0 in the TMOD register:

TIMER MODE-0: Setting timer mode bits to 00b in the TMOD register results in using the TH register as an 8-bit counter and TL as a 5-bit counter. Therefore mode0 is a 13-bit counter.

TIMER MODE-1: Mode-1 is similar to mode-0 except TL is configured as a full 8-bit counter when the mode bits are set to 01b in TMOD.

TIMER MODE-2: Setting the mode bits to 10b in TMOD configures the timer to use only the TL counter as an 8-bit counter. TH is used to hold a value that is loaded into TL every time TL overflows from FFh to 00h. The timer flag is also set when TL overflows.

TIMER MODE-3: In mode-3, timer-1 simply hold its count, where as timer 0 registers TL0 and TH0 are used as two separate 8-bit counters. TL0 uses the Timer-0 control bits. TH0 counts machine cycles and takes over the use of TR1 and TF1 from Timer-1.

3.3.9 INTERRUPTS

A computer has only two ways to determine the conditions that exist in internal and external circuits. One method uses software instructions that jump to subroutines on the states of flags and port pins. The second method responds *to hardware signals, called interrupts that force the program to call a subroutine.*

The AT89S52 has a total of six interrupt vectors: two external interrupts (INT0 and INT1), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

Each interrupt forces the processor to jump at the interrupt location in the memory. The interrupted program must resume operation at the instruction where the interrupt took place. Program resumption is done by storing the interrupted PC address on to stack.

RETI instruction at the end of ISR will restore the PC address.

3.3.10 MICROCONTROLLER CONFIGURATION USED IN THE SET-UP

The microcontroller is interfaced with the ADC in polling mode. INT0 is used for the LCD mode selection switch in order to switch between two modes of display:

1) Sensor output display
2) Actuator status display

Port details:

- Port 0: Interfaced with the LCD data lines.
- Port 1: Interfaced with the ADC data lines
- Port 2: Interfaced with the LCD Control lines
- Port 3: Interfaced with the ADC control lines

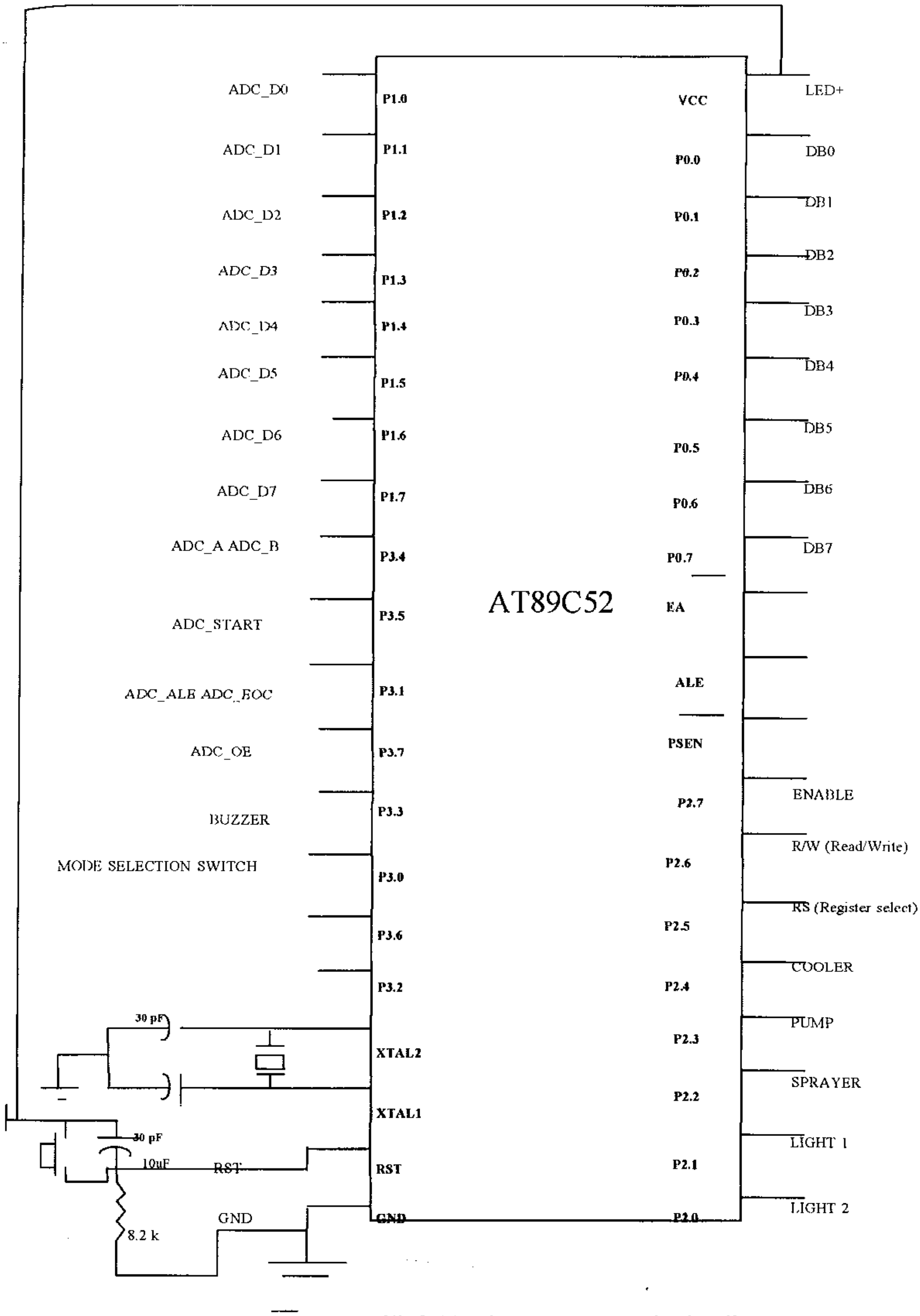


Fig3.14 Microcontroller pin details

3.4 LIQUID CRYSTAL DISPLAY

A liquid crystal display (LCD) is a thin, flat display device made up of any number of color or monochrome pixels arrayed in front of a light source or reflector. Each pixel consists of a column of liquid crystal molecules suspended between two transparent electrodes, and two polarizing filters, the axes of polarity of which are perpendicular to each other. Without the liquid crystals between them, light passing through one would be blocked by the other. The liquid crystal twists the polarization of light entering one filter to allow it to pass through the other.

The LCD we are using in our project is a 16*2 Alphanumeric LCD. **Characteristics:**

Intelligent, with built-in Hitachi HD44780 compatible LCD controller and RAM providing simple interfacing

- 61 x 15.8 mm viewing area
- 5 x 7 dot matrix format for 2.96 x 5.56 mm characters, plus cursor line
- Can display 224 different symbols
- Low power consumption (1 mA typical)
- Powerful command set and user-produced characters
- TTL and CMOS compatible
- Connector for standard 0.1-pitch pin headers.

Many microcontroller devices use 'smart LCD' displays to output visual information. LCD displays designed around Hitachi's LCD JHD162A module, are inexpensive, easy to use, and it is even possible to produce a readout using the 8x80 pixels of the display. They have a standard ASCII set of characters and mathematical symbols.

For an 8-bit data bus, the display requires a +5V supply plus 11 I/O lines. For a 4-bit data bus it only requires the supply lines plus seven extra lines. When the LCD display is not enabled, data lines are tri-state and they do not interfere with the operation of the microcontroller.

Pin	Name	Pin function	Connection
1	VSS	Ground	GND
2	VCC	Positive supply for LCD	5V
3	VEE	Brightness adjust	Connected to a preset to adjust brightness
4	RS	Select register, select instruction or data register	RB7
5	R/W	Select read or write	GND
6	E	Start data read or write	RB6
7	DB0	Data bus pin	RD0
8	DB1	Data bus pin	RD1
9	DB2	Data bus pin	RD2
10	DB3	Data bus pin	RD3
11	DB4	Data bus pin	RD4
12	DB5	Data bus pin	RD5
13	DB6	Data bus pin	RD6
14	DB7	Data bus pin	RD7
15	LED-	Backlight positive input	5V

Fig 3.15 LCD pin description

Data can be placed at any location on the LCD. For 16×2 LCD, the address locations are:

First line	**80**	**81**	**82**	**83**	**84**	**85**	**86**	**through**	**8F**
Second line	**C0**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**through**	**CF**

Fig3.16 Address locations for a 2x16 line LCD

SIGNALS TO THE LCD

The LCD also requires 3 control lines from the microcontroller:

1) Enable (E)

This line allows access to the display through R/W and RS lines. When this line is low, the LCD is disabled and ignores signals from R/W and RS. When (E) line is high, the LCD checks the state of the two control lines and responds accordingly.

2) Read/Write (R/W)

This line determines the direction of data between the LCD and microcontroller When it is low, data is written to the LCD. When it is high, data is read from the LCD.

3) **Register select (RS)**

With the help of this line, the LCD interprets the type of data on data lines. When it is low, an instruction is being written to the LCD. When it is high, a character is being written to the LCD.

Logic status on control lines:

- E - 0 Access to LCD disabled
  - 1 Access to LCD enabled
- R/W - 0 Writing data to LCD
  - 1 Reading data from LCD
- RS - 0 Instructions
  - 1 Character

Writing and reading the data from the LCD:

Writing data to the LCD is done in several steps:

1) Set R/W bit to low

2) Set RS bit to logic 0 or 1 (instruction or character)

3) Set data to data lines (if it is writing)

4) Set E line to high

5) Set E line to low

Read data from data lines (if it is reading):

1) Set R/W bit to high
2) Set RS bit to logic 0 or 1 (instruction or character)
3) Set data to data lines (if it is writing)
4) Set E line to high
5) Set E line to low

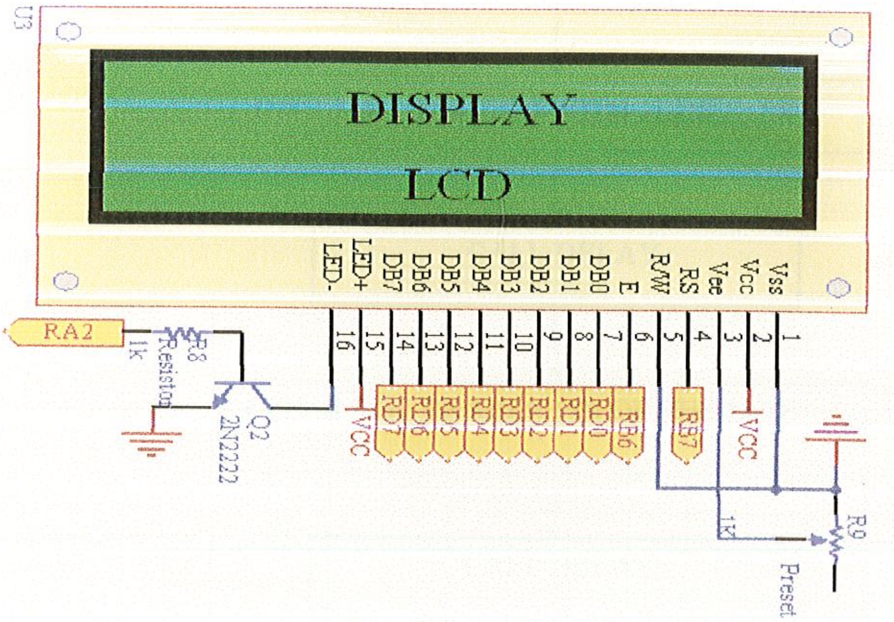
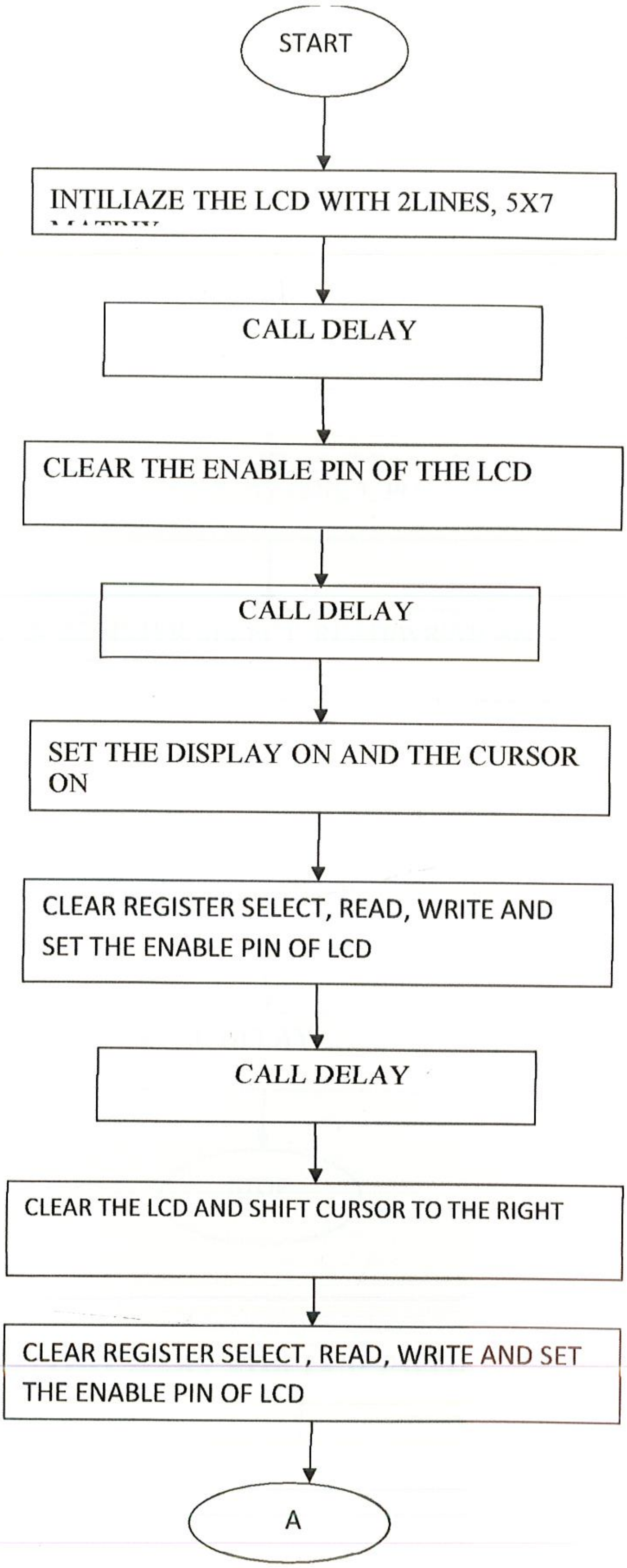


Fig 3.17 LCD Interference

LCD Initialization



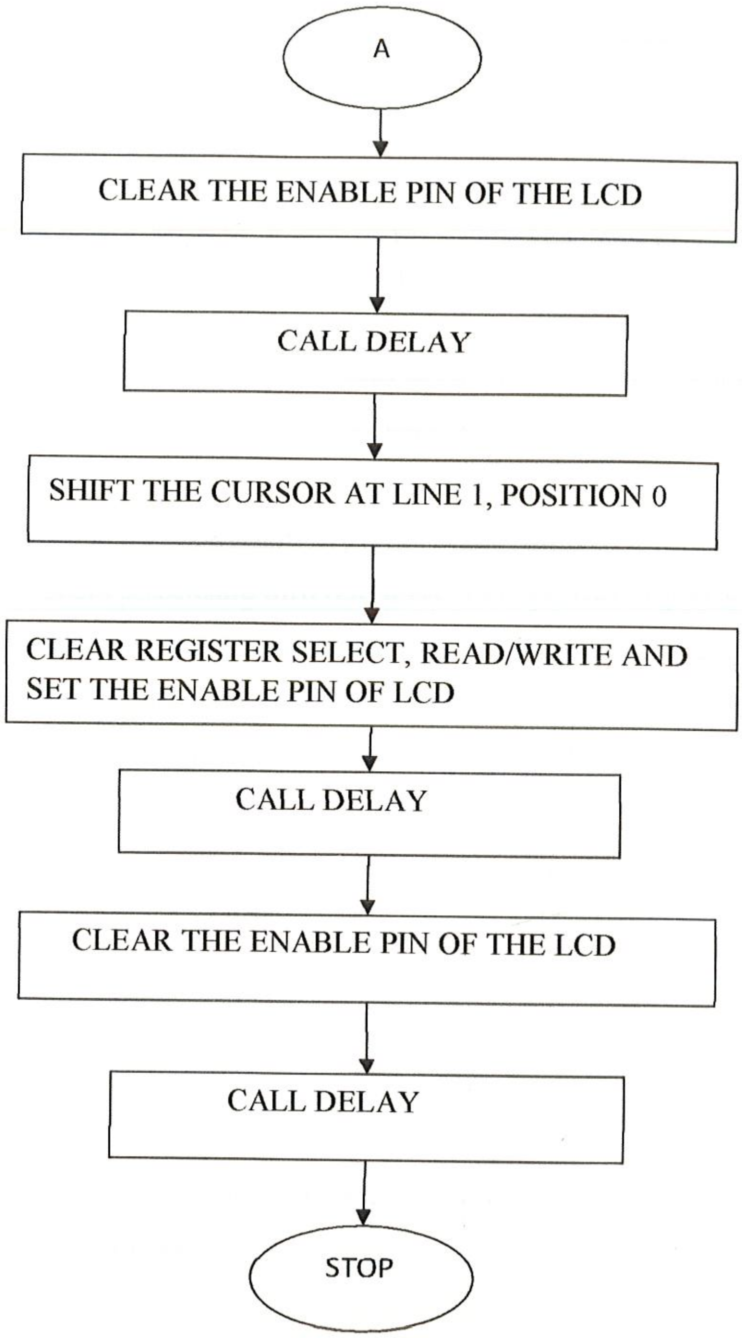


Fig3.18 FLOWCHART FOR LCD INITIALIZATION

CHAPTER 4

SOFTWARE DESCRIPTION

4.1 INTRODUCTION TO TOPVIEW SOFTWARE

Top view Simulator gives an excellent simulation environment for the Industry's most popular 8 bit microcontroller family, MCS 51. It gives all the required facilities to enable the system designers to start projects right from the scratch and finish them with ease and confidence.

The following figure indicates the facilities available in the simulation environment that give you required development power to handle your next real time embedded system design applications.

Top view Simulator is the total solution giving many states of art features meeting the needs of the designers possessing different levels of expertise. If you are a beginner, then you can easily learn about 8031 based embedded solutions without any hardware. If you are an experienced designer, you may find most of the required facilities built in the simulator that enable you to complete your next project without waiting for the target hardware.

The simulator is designed by the active feedback from the demanding designers and when you use this in your next 8031 project, you are assured of definite savings in time and increase in productivity.

The features of the simulator are briefly tabulated here.

4.1.1 DEVICE SELECTION

A wide range of device selection, including generic 8031 devices and Atmel's AT89CXX series 8031 microcontrollers.

4.1.2 PROGRAM EDITING

Powerful editing feature for generating your programs and the facility to call an external assembler to process input programs.

4.1.3 CLEAR VIEW GUI ENVIROMENT

ClearView GUI facility gives all the internal architectural details in the strategically placed windows. Information about the Program, Data Memory, Registers, Peripherals, SFR Bits, Memory Bits are clearly presented in many windows to make you understand the program flow very easily.

4.1.4 PROGRAM EXECUTION

A variety of program execution options include Single Stroke full speed execution, Single Step, Step over and Breakpoint execution modes give you total control over the target program.

Clear View updates all the windows with the correct and latest data and it is a convenient help during your debugging operations.

You may find how this Topview Simulator simplifies the most difficult operation of the program development, debugging, into a most simple task.

4.1.5 SIMULATION FACILITIES

Powerful simulation facilities are incorporated for I/O lines, interrupt lines and the clock sources meant for Timers/Counters.

Many external embedded building blocks can be simulated:

- Range of Plain Point LED's and Seven Segment LED Display options.
- LCD modules in many configurations.
- Momentary ON keys, Toggle Switches.
- A variety of keypads upto 4 X 8 key matrix.
- All modes of on-chip serial port communication facility.
- IIC components including RTC, EEPROMs.
- SPI Bus based EEPROM devices.

4.1.6 CODE GENERATION FACILITIES

Powerful and versatile code generating facility enables you to generate the exact and compact assembly code for many possible application oriented interfacing options.

You can simply define your exact needs and get the target assembly code at a press of button at anywhere in your program flow. The code gets embedded into your application program automatically. You are assured trouble free working of final code in the real time.

- All modes of the serial port.
- Interfacing IIC, SPI Bus devices.
- Range of keypads.
- Many LED/LCD interfacing possibilities

4.2 PROGRAMMER

The programmer used is a powerful programmer for the Atmel 89 series of microcontrollers that includes 89C51/52/55, 89S51/52/55 and many more.

It is simple to use & low cost, yet powerful flash microcontroller programmer for the Atmel 89 series. It will Program, Read and Verify Code Data, Write Lock Bits, Erase and Blank Check. All fuse and lock bits are programmable. This programmer has intelligent onboard firmware and connects to the serial port. It can be used with any type of computer and requires no special hardware. All that is needed is a serial communication port which all computers have.

All devices also have a number of lock bits to provide various levels of software and programming protection. These lock bits are fully programmable using this programmer. Lock bits are useful to protect the program to be read back from microcontroller only allowing erase to reprogram the microcontroller.

Major parts of this programmer are Serial Port, Power Supply and Firmware microcontroller. Serial data is sent and received from 9 pin connector and converted to/from TTL logic/RS232 signal levels by MAX232 chip. A Male to Female serial port cable, connects to the 9 pin connector of hardware and another side connects to back of computer.

All the programming 'intelligence' is built into the programmer so you do not need any special hardware to run it. Programmer comes with window based software for easy programming of the devices.

CHAPTER 5

RESULT ANALYSIS

Readings taken at room temperature of 27^0C

5.1 SOIL MOISTURE SENSOR

Tolerance= ± 0.2 V

Soil Condition	Transducer Optimum Range
Soil is dry	0V
Optimum level of soil moisture	1.9- 3.5V
Slurry soil	>3.5V

Table 5.1 Soil moisture sensor readings

5.2 TEMPERATURE SENSOR

FORMULA:

Temperature (^{0}C) = (Vout/5) *100(^{0}C/V)

Temperature range in degree Celsius	**Temperature sensor output(V_{out})**
10^{0}C	0.5V
15^{0} to 20^{0}C	0.75-1.0V
20 ^{0}to 25^{0} C	1.0-1.25V
25^{0} to 30 ^{0}C	1.25-1.5V
30 ^{0}to 35 ^{0}C	1.5-1.75V
35^{0} to 40^{0} C	1.75-2.0V
40^{0} to 45 ^{0}C	2.0-2.25V
45^{0} to 50^{0} C	2.25-2.5V
50^{0} to 55 ^{0}C	2.5-2.75V
55^{0} to 60^{0}C	2.75-3.0V
60^{0} to 65^{0} C	3.0-3.25V
65^{0} to 70 ^{0}C	3.25-3.5V
70 ^{0}to 75^{0} C	3.5-3.75V
75 ^{0}to 80 ^{0}C	3.75-4.0V
80 ^{0}to 85^{0} C	4.0-4.25V

85 ^{0}to 90^0 C	4.25-4.5V
90^0 to 95 ^{0}C	4.5-4.75V
95^0 to 100^0 C	4.75-5V

Table 5.2 Temperature sensor reading

5.3 ADC Readings:

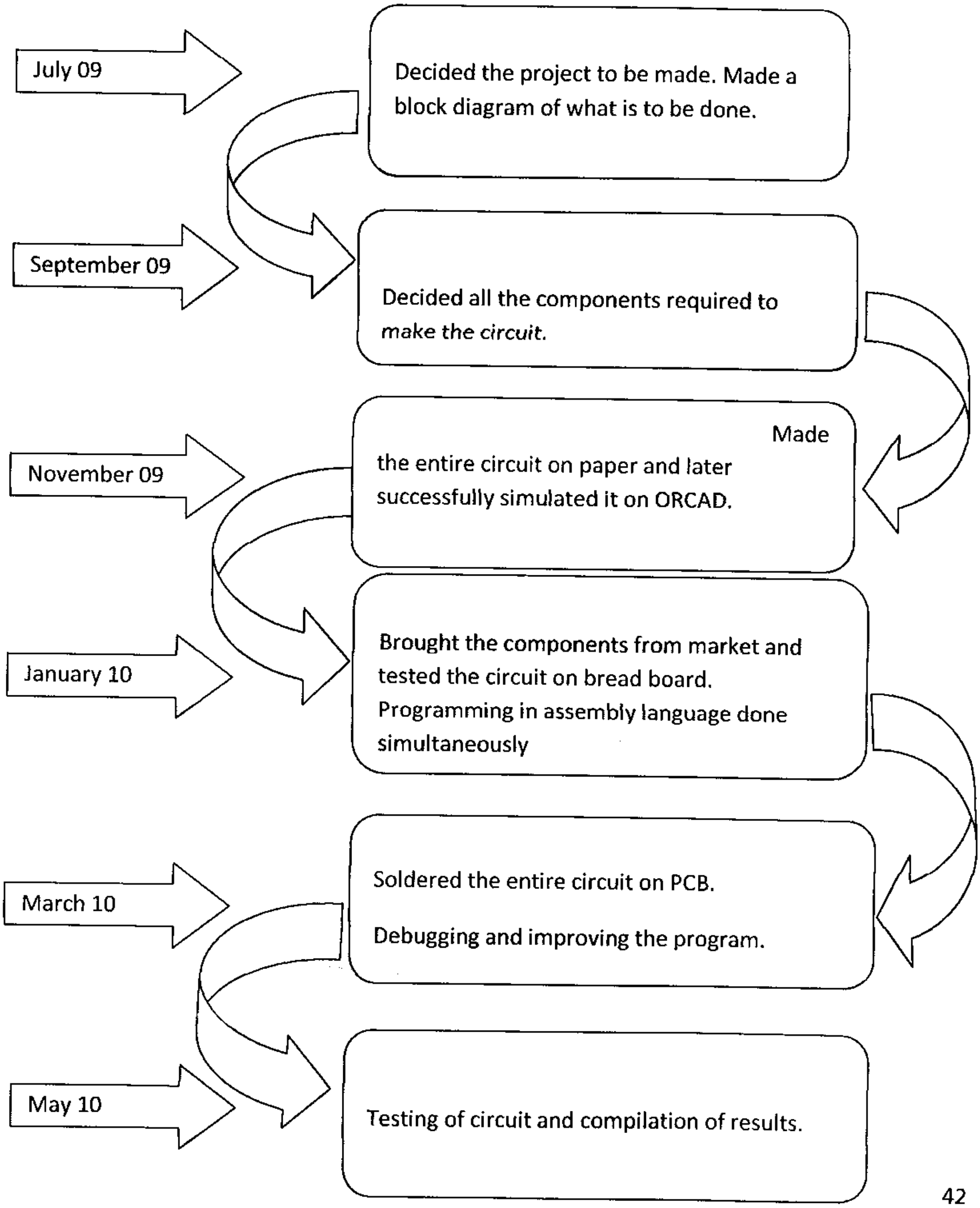
Bit Position	ADC Input Voltage										
	0V	0.5V	1V	1.5V	2V	2.5V	3V	3.5V	4V	4.5V	5V
D1	0	0	1	1	1	1	1	1	0	1	1
D2	0	1	0	0	0	0	0	1	0	0	1
D3	0	0	1	1	1	1	1	0	0	0	1
D4	0	1	0	0	0	0	0	0	1	0	1
D5	0	1	1	1	0	1	1	0	0	0	1
D6	0	0	1	0	1	1	0	0	0	1	1
D7	0	0	0	1	1	1	0	1	1	1	1
D8	0	0	0	0	0	0	1	1	1	1	1

Table 5.3 ADC output corresponding to amplified temperature sensor output

CHAPTER 6

Project Progress and Future Works

The project is evolved from the following process:



Future Works

Presently we have devised an equipment which takes all the readings of soil and its corresponding environment and displays it on LCD. These readings have to be noted down manually and then we have to look for the best crop correspondingly.

In future, to fully automate this entire procedure we can connect the microcontroller to a database containing all the information regarding different crops corresponding to different soils and environment and hence by just switching on the device on a particular terrain, suitable crop/crops would be displayed on screen.

To do this, firstly we'll have to create a database in any preferred language say SQL. The database will contain all crops corresponding to all the possible values of pH, temperature, moisture and altitude. Then all our readings will be matched with the database. This can be done by connecting any one of the ports of microcontroller to computer via serial port. All the readings taken can be stored in the microcontroller's internal memory or externally connected flash memory. Once the data is stored, it can then be send to computer. Once the readings reach the computer, they are matched with the database and the corresponding result is displayed on the screen.

Another addition to this project could be including **Bio-sensors** to detect the organic contents of the soil and hence predicting the corresponding crop/crops more accurately.

This can be used for soil and organic testing on other **celestial bodies**.

CONCLUSION

A step-by-step approach in designing the microcontroller based system for measurement and control of the four essential parameters for plant growth, i.e. temperature, soil moisture, pH and atmospheric pressure, has been followed. The results obtained from selected measurement have shown that the system performance is quite reliable and accurate.

We were unable to overcome a few shortcomings of the existing systems but we achieved reduced power consumption, easy maintenance and low complexity, at the same time providing a flexible and precise form of maintaining the environment.

The continuously decreasing costs of hardware and software, the wider acceptance of electronic systems in agriculture, and an emerging agricultural control system industry in several areas of agricultural production, will result in reliable control systems that will address several aspects of quality and quantity of production. Further improvements will be made as less expensive and more reliable sensors are developed for use in agricultural production.

Although the enhancements mentioned in the previous chapter may seem far in the future, the required technology and components are available, many such systems have been independently developed, or are at least tested at a prototype level. Also, integration of all

these technologies is not a daunting task and can be successfully carried out.

ANNEXTURES

8.1 Source Code:

```
$mod51
RS        BIT    P2.5            ; P2.5 is connected to RS pin of LCD
RW        BIT    P2.6            ; P2.6 is connected to R/W pin of LCD
E         BIT    P2.7            ; P2.5 is connected to E pin of LCD
LCDDATA   EQU    P0              ; P0.0-P0.7 are connected to LCD data pins D0-D7
                                 ; Microcontroller connections to ADC0808/9 lines.
OE        EQU    P3.0            ; Pin 9 Output Enable
START     EQU    P3.1            ; Pin 6 Start
EOC       EQU    P3.3            ; Pin 7 EOC
ADD_A     EQU    P3.4            ; Pin 25 ADD A
ADD_B     EQU    P3.5            ; Pin 24 ADD B
ALE       EQU    P3.7            ; Pin 22 ALE
ADCDATA   EQU    P1              ; Data Lines

LCD_INTR  BIT    00h             ;LCD Interrupt flag bit
T_BUFFER  EQU    30h             ;Temperature buffer (ADC o/p)

org 00h
LJMP main
main:
        mov p1,#20h
        ACALL lcd_init          ;LCD initialisation
        ACALL line_1            ;Display welcome message
        lcd_init:                       ;LCD initialisation
        MOV A, #38H                     ;init. LCD 2 lines,5x7 matrix (CHANGE THIS!!!
        ACALL comnwrt           ;call command subroutine
        ACALL lcddelay          ;give LCD some time
        MOV A, #0EH             ;display on, cursor on
```

```
        ACALL comnwrt       ;call command subroutine
        ACALL lcddelay      ;give LCD some time
        MOV A, #01          ;clear LCD
        ACALL comnwrt       ;call command subroutine
        ACALL lcddelay      ;give LCD some time
        MOV A, #06H         ;shift cursor right
        ACALL comnwrt       ;call command subroutine
        ACALL lcddelay      ;give LCD some time
        RET

line_1:                             ;Displaying initialisation message
        MOV A, #080H                ;cursor at line 1, position 0
        ACALL comnwrt       ;call command subroutine
        ACALL lcddelay      ;give LCD some time
        MOV DPTR, #init1    ;address of line 1 of initialisation message
repeat1:
        CLR A
        MOVC A, @A+DPTR                 ;moving the data to accumulator
        JZ line_2
        ACALL datawrt               ;call data display routine
        ACALL lcddelay              ;give LCD some time
        INC DPTR            ;increment address
        SJMP repeat1

line_2:
        MOV A, #0C0h                ;cursor at line 2, position 0
        ACALL comnwrt       ;call command subroutine
        ACALL lcddelay      ;give LCD some time
        MOV DPTR, #init2                ;address of line 2 of initialisation message
repeat2:
        CLR A
```

```
            MOVC A, @A+DPTR             ;moving the data to accumulator
            JZ line_3
            LCALL datawrt               ;call data display routine
            LCALL lcddelay              ;give LCD some time
            INC DPTR                    ;increment address
            SJMP repeat2                ;jump to label repeat2

    line_3:                             ;Displaying initialisation message
            MOV A, #080H                ;cursor at line 1, position 0
            ACALL comnwrt               ;call command subroutine
            ACALL lcddelay              ;give LCD some time
            MOV DPTR, #init3            ;address of line 1 of initialisation message
    repeat3:
            CLR A
            MOVC A, @A+DPTR             ;moving the data to accumulator
            JZ line_4
            ACALL datawrt               ;call data display routine
            ACALL lcddelay              ;give LCD some time
            INC DPTR                    ;increment address
            SJMP repeat3

    line_4:
            MOV A, #0C0h                ;cursor at line 2, position 0
      ACALL comnwrt                     ;call command subroutine
            ACALL lcddelay              ;give LCD some time
            MOV DPTR, #init4            ;address of line 2 of initialisation message
    repeat4:
            CLR A
            MOVC A, @A+DPTR             ;moving the data to accumulator
            JZ line_5
            LCALL datawrt               ;call data display routine
```

```
        LCALL lcddelay              ;give LCD some time
        INC DPTR                    ;increment address
        SJMP repeat4                ;jump to label repeat4

line_5:                             ;Displaying initialisation message
        MOV A, #080H                ;cursor at line 1, position 0
        ACALL comnwrt               ;call command subroutine
        ACALL lcddelay              ;give LCD some time
        MOV DPTR, #init5            ;address of line 1 of initialisation message
repeat5:
        CLR A
        MOVC A, @A+DPTR             ;moving the data to accumulator
        JZ line_6
        ACALL datawrt               ;call data display routine
        ACALL lcddelay              ;give LCD some time
        INC DPTR                    ;increment address
        SJMP repeat5

line_6:
        MOV A, #080H                ;cursor at line 1, position 0
    ACALL comnwrt                   ;call command subroutine
        ACALL lcddelay              ;give LCD some time
        MOV DPTR, #init6            ;address of line 2 of initialisation message
repeat6:
        CLR A
        MOVC A, @A+DPTR             ;moving the data to accumulator
        JZ line_7
        LCALL datawrt               ;call data display routine
        LCALL lcddelay              ;give LCD some time
        INC DPTR                    ;increment address
        SJMP repeat6                ;jump to label repeat4
```

```
line_7:
        MOV A, #0C0h              ;cursor at line 2, position 0
        ACALL comnwrt             ;call command subroutine

        ACALL lcddelay            ;give LCD some time
        MOV DPTR, #init7          ;address of line 2 of initialisation message
repeat7:
        CLR A
        MOVC A, @A+DPTR           ;moving the data to accumulator
        JZ exit
        LCALL datawrt             ;call data display routine
        LCALL lcddelay            ;give LCD some time
        INC DPTR                  ;increment address
        SJMP repeat7              ;jump to label repeat7
exit:
        mov a,p1
        ;MOV  A, #01h             ;clear LCD
        lcall datawrt
        ;LCALL comnwrt
        LCALL   lcddelay          ;give LCD some time
        ACALL adcread             ;Read sensor data from ADC

comnwrt:                          ;send command to LCD
        MOV LCDDATA, A            ;copy reg A to port1
        CLR RS                    ;RS=0 for command
        CLR RW                    ;R/W=0 for write
        SETB E                    ;E=1 for high pulse
        MOV R3,#50
REG_3:
```

```
        MOV R4,#255
        DJNZ R4,$
        DJNZ R3,REG_3
        CLR E                         ;E=0 for H-to-L pulse
        RET

datawrt:                              ;write data to LCD
        MOV LCDDATA, A                ;copy reg A to port1
        SETB RS                       ;RS=1 for data
        CLR RW                        ;R/W=0 for write
        SETB E                        ;E=1 for high pulse
        MOV R3,#50
REG_4:
        MOV R4,#255
        DJNZ R4,$
        DJNZ R3,REG_4
        CLR E                         ;E=0 for H-to-L pulse
        RET
lcddelay:                             ;Delay routine for LCD
        MOV R3, #30
here2:
        MOV R4, #255
here3:
        DJNZ R4, here3                ;stay until R4 becomes 0
        DJNZ R3, here2
        RET
adcread:                              ;Initializing ADC
                                      ;SENSOR 1
        MOV ADCDATA, #01FH            ; Data lines for input
        mov r7,adcdata
        SETB EOC                      ; Make EOC i/p
```

```
        ;ACALL delay
        CLR ALE                         ; clearing ALE
        CLR START                       ; Make start high
        CLR OE                          ; Disable o/p
        CLR ADD_A                       ;A=0
        CLR ADD_B                       ;B=0    ;Select IN4
        ACALL delay
        SETB ALE                        ;latching the address
        ;ACALL delay
        SETB START                      ;start conversion pulse
        ACALL delay
        CLR ALE                         ;ALE H-L transition
        CLR START                       ;START H-L transition
    acall delay
        JB EOC,$
        JNB EOC,$
        ;ACALL delay
        SETB OE                         ;enable o/p
        ACALL delay
        MOV T_BUFFER, ADCDATA           ;store adc data to buffer
        mov r7,adcdata
        CLR OE                          ;disable o/p
        ACALL DATA_DISPLAY


data_display:                           ;LCD routine to display sensor and actuator
data

        JNB LCD_INTR, sensor_display    ;if ACT=0, jump to display sensor data label
```

```
        MOV  A, #01h                ;clear LCD
        ACALL comnwrt
        ACALL lcddelay              ;give LCD some time
        MOV  A,#80H                 ;cursor at line 1, position 0
        ACALL comnwrt               ;call command subroutine
        ACALL lcddelay              ;give LCD some time
        MOV R0, #60h                ;address of display message line 1
line1:
        CLR A
        MOV A, @R0                  ;moving the data to accumulator
        JZ nxt_line                 ;exit loop on completion of display and start
                                            ;display of LCD next line
        LCALL datawrt               ;call data display routine
        LCALL lcddelay              ;give LCD some time
        INC R0                      ;increment address
        SJMP line1                  ;repeat display till end of message

nxt_line:                           ;Displaying on line 2 of LCD
        MOV     A, #0C0h            ;cursor at line 2, position 0
        LCALL       comnwrt         ;call command subroutine
        LCALL       lcddelay        ;give LCD some time
        MOV R0, #70h                ;address of display message line 2 line2:
        CLR A
        MOV A, @R0                  ;moving the data to accumulator
        JZ delay                    ;exit loop on completion of display & call
routine
                                            ;to wait 5 sec
        LCALL datawrt               ;call data display routine
        LCALL lcddelay              ;give LCD some time
        INC R0                      ;increment address
        SJMP line1                  ;repeat display till end of message
```

```
acall delay

MOV   A, #01h                          ;clear LCD LCALL comnwrt
LCALL         lcddelay                 ;give LCD some time

sensor_display:                        ;displaying sensor data
MOV   A,#80H                           ;cursor at line 1, position 0
LCALL         comnwrt                  ;call command subroutine
LCALL         lcddelay                 ;give LCD some time

MOV   A, #'T' ;display letter T
LCALL         datawrt                  ;call display subroutine
LCALL         lcddelay                 ;give LCD some time
MOV   A, #':'                          ;display : symbol
LCALL         datawrt                  ;call display subroutine
LCALL         lcddelay                 ;give LCD some time

MOV A, t_buffer                        ;move sensor data from buffer to
accumulator
ACALL DISPLAYTENS
;MOV DPTR, #temp_tens                  ;address of ASCII code look up table (tens
place) MOVC          A, @A+DPTR        ;store corresponding ASCII code to
Accumulator
LCALL         datawrt                  ;call display subroutine
ACALL DATAWRT
LCALL         lcddelay                 ;give LCD some time
MOV A ,t_buffer
ACALL DISPLAYONES
MOVC A, @A+DPTR                        ;store corresponding ASCII code to
Accumulator
LCALL         datawrt                  ;call display subroutine
```

```
    ACALL DATAWRT
    LCALL       lcddelay          ;give LCD some time

    MOV   A, #'.'                 ;display . symbol
    LCALL       datawrt           ;call display subroutine
    LCALL       lcddelay          ;give LCD some time

    MOV A, T_BUFFER               ;move sensor data from buffer to
accumulator
    MOVC A, @A+DPTR               ;store corresponding ASCII code to
Accumulator
    LCALL       datawrt           ;call display subroutine
    LCALL       lcddelay          ;give LCD some time
    MOV   A, #0DFh                ;display degree symbol
    LCALL       datawrt           ;call display subroutine
    LCALL       lcddelay          ;give LCD some time
    MOV   A, #'C'                 ;display letter C
    LCALL       datawrt           ;call display subroutine
    LCALL       lcddelay          ;give LCD some time

    MOV   A, #' '                 ;display space
    LCALL       datawrt           ;call display subroutine
    LCALL       lcddelay          ;give LCD some time
    ACALL FINISH

displayTENS:                      ;conversion
    anl a,#0fh
    orl a,#30h
    mov r2,a
    RET
```

```
DISPLAYONES:

        mov a,r6
        anl a,#0f0h
        rr a
        rr a
        rr a
        rr a
        orl a,#30h
        mov r4,a
        RET
delay:                                  ;Delay subroutine
        MOV R3, #50
here1:
        MOV R4, #255
here:
        DJNZ R4, here
        DJNZ R3, here1
        RET

org 200h

        init1: DB '      Welcome to      ', 0
        init2: DB '      Project ', 0
        init3: DB '  soil testing      ', 0
        init4: DB '   BY          ', 0
        init5: DB '  ANKIT $ NIHIT       ', 0
        init6: DB '      Initializing the', 0
        init7: DB '      Sensors.........', 0     ; System initialization message
FINISH:
END
```

Bibliography

(All the references should be arranged alphabetically in following way)

1. M.A. Mazidi and J.G. Mazidi, "8051 microcontroller and embedded systems", 2006 edition, Pearson Education publications.
2. Ramakant A. Gayakwad," OP-AMP and Linear IC's, 2005 Edition", Prentice Hall publications.
3. Kenneth J. **Ayala,** The 8051 **Microcontroller,2005 edition,** Penram International Publishing