AUTOMATIC IRRIGATION AND GSM CONTROLLED FERTILIZATION AND PEST CONTROL

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

BACHELOR OF TECHNOLOGY

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Under the Supervision of

Mr. MUNISH SOOD

By

ANUBHAV JAIN – 081097

JASDEEP SINGH - 081140

to

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT, SOLAN – 173234, HIMACHAL PRADESH

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

WAKNAGHAT

SOLAN, HIMACHAL PRADESH

\

CERTIFICATE

This is to certify that project report entitled "**AUTOMATIC IRRIGATION AND GSM CONTROLLED FETILIZATION AND PEST CONTROL**", submitted by **ANUBHAV JAIN(081097)** and **JASDEEP SINGH(081140)** in partial fulfillment for the award of degree of Bachelor of Technology in Electronics and Communication Engineering to **JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,WAKNAGHAT**, has been carried out under my supervision.

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.

[illegible]

DATE: 31/5/2012 SUPERVISOR'S NAME : **MR. MUNISH SOOD**

DESIGNATION : LECTURER

ACKNOWLEDGEMENT

No venture can be completed without the blessing of almighty. We consider it our bounded duty to bow to almighty whose kind blessings always inspire us on the right part of life. This project is our combined effort and realizes the potential of team and gives us a chance to work in co-ordination.

Science has caused many frontiers so has human efforts towards human research. Our revered guide **Mr. MUNISH SOOD**, Lecturer, Department of ELECTRONICS AND COMMUNICATION, JUIT, has indeed acted as a light house showing us the need of sustained effort in the field of embedded systems and GSM technologies to learn more and more. So we take this opportunity to thank him, for lending us stimulating suggestions, innovative quality guidance and creative thinking. He provided us the kind of strategies required for the completion of task. We are grateful to him for the support, he provided us in doing things at our pace and for being patient for our mistakes.

DATE: Name of the students: ANUBHAV JAIN (081097)

JASDEEP SINGH (081140)

TALBLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

WSN: WIRELESS SENSOR NETWORK

DTMF: DUAL TONE MODULATION FREQUENCY

HSM: HUMIDITY SENSOR MODULE

GSM: GLOBAL SYSTEM FOR MOBILE

LCD: LIQUID CRYSTAL DISPLAY

IC: INTEGRATED CIRCUITS

RISC: REDUCED INSTRUCTION SET COMPUTER

CMOS: COMPLIMENTARY METAL OXIDE SEMICONDUCTOR

PCB: PRINTEDCIRCUIT BOARD

ABSTRACT

The project entitled as AUTOMATIC IRRIGATION AND GSM CONTROLLED FERTILIZATION AND PEST CONTROL is a system that is aimed to make various farming processes very easy and efficient for the farmers of our country at low prices. Automatic irrigation is widely used in the western countries but is not practiced in our part of the world. Our project basically emphasizes on increasing the yield for maximum profit to the farmer. India being an agricultural country has to bring in technology to improve the existing conditions in the field of agriculture. Improving irrigation efficiency can contribute greatly to reducing production costs of vegetables, making the industry more competitive and sustainable. Furthermore this system can be programmed according to the needs of farmer.

The basic thinking of our project is inspired by the thought of creating an environment friendly system which ensures maximum nourishment of soil with minimum wastage of water so that our country can gain food security. For this we are using the optimum use of electronics and semiconductor devices. The project uses microcontroller, soil moisture sensor, solenoid valves and various other integrated circuits

Besides performing irrigation processes automatically, the system can also be controlled by a GSM phone which can be very easily used by even an illiterate farmer.

Its benefits are:

- No inescapable delays in irrigation and fertilizing the fields.
- Low cost
- Even illiterate farmers can use the system without any difficulties.
- Reduces physical work of a farmer
- Saves Time
- Insures proper enrichment of the soil and hence better yields.
- Reduces labor cost of the farmer with one time investment.
- Negligible maintenance and operational cost.
- Saves Money
- Saves Water
- Improves Growth
- Weed Reduction

1: INTRODUCTION

Our project basically lies under the field of embedded systems. We have used a microcontroller and various other components and have programmed it on order to complete the task.

1.1: What is Embedded Systems?

An embedded system is a computer system designed for specific control functions within a larger system, often with real-time computing constraints. It is *embedded* as part of a complete device often including hardware and mechanical parts. By contrast, a general-purpose computer, such as a personal computer (PC), is designed to be flexible and to meet a wide range of end-user needs. Embedded systems control many devices in common use today.[3]

Embedded systems contain processing cores that are typically either microcontrollers or digital signal processors (DSP). The key characteristic, however, is being dedicated to handle a particular task. Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically, embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like lights, factory, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

1.2: Why irrigation based project?

Automatic irrigation is widely used in the western countries but is not practiced in our part of the world. Our project basically emphasizes on increasing the yield for maximum profit to the farmer. India being an agricultural country has to bring in technology to improve the existing conditions in the field of agriculture. Improving irrigation efficiency can contribute greatly to reducing production costs of vegetables, making the industry more competitive and sustainable. Through proper irrigation, average vegetable yields can be maintained (or increased) while minimizing environmental impacts caused by excess applied water and subsequent agrichemical leaching. Recent technological advances have made soil water sensors available for efficient and automatic operation of irrigation systems.

Here we have designed a module using a microcontroller and GSM. In this system the fields will get automatically with the help of soil moisture sensors. Farmer has to do nothing to irrigate the field, though he can control it manually if he wants. For fertilization and pest control activities, he can use his GSM to control the corresponding solenoid valves. He himself need not to be physically present at the field. So, all this system makes the work of the farmer very easy and reliable and hence improvises the yield to meet the food requirements of the country. If the farmer wants to switch on the solenoid valve he just needs to give a ring to the particular gsm phone number which is implemented on the project board which is auto picked and press the corresponding key of the function he desires to perform

1.3: WORKING

Here we have designed a module using a microcontroller and GSM.As the project name suggests it is about automatic irrigation and controlling fertilization and pesticides activities through the GSM phone.

- Suppose the water content of the soil has fallen down to an undesirable value, Soil moisture sensor sends the moisture value to the microcontroller which in turn directs solenoid valve driver to open the valve for irrigation. It is all done according to the microcontroller programming. Drip irrigation will be used as the method of irrigation as it is the most water efficient system present.

- Now, in the case when farmer wants to fertilize the field. He will dial the gsm number present on the project board. Suppose, key 4 has been assigned to fertilize the field for 2 minutes so the farmer presses the key 4 which produces dual frequency which is converted into a 4 digit binary number by DTMF decoder and the number is fed to the microcontroller which in turn directs solenoid valve driver to open up the fertilizer valve for 2 minutes and hence the field is fertilized for 2 minutes.

- In the case when farmer wants to apply pesticides to the field. He will dial the gsm number present on the project board. Suppose, key 5 has been assigned to apply pesticides to the field for a minute so the farmer presses the key 5 which produces dual frequency which is converted into a 4 digit binary number by DTMF decoder and the number is fed to the microcontroller which in turn directs solenoid valve driver to open up the valve for pesticides for a minutes and hence the field has been applied with pesticides for a minute.

2. HISTORY

One of the first recognizably modern embedded systems was the Apollo Guidance Computer, developed by Charles Stark Draper at the MIT Instrumentation Laboratory. At the project's inception, the Apollo guidance computer was considered the riskiest item in the Apollo project as it employed the then newly developed monolithic integrated circuits to reduce the size and weight. An early mass-produced embedded system was the Automatics D-17 guidance computer for the Minuteman missile, released in 1961. It was built from transistor logic and had a hard disk for main memory. When the Minuteman II went into production in 1966, the D-17 was replaced with a new computer that was the first high-volume use of integrated circuits. This program alone reduced prices on quad nand gate ICs from $1000/each to $3/each, permitting their use in commercial products.

Since these early applications in the 1960s, embedded systems have come down in price and there has been a dramatic rise in processing power and functionality. The first microprocessor for example, the Intel 4004, was designed for calculators and other small systems but still required many external memory and support chips. In 1978 National Engineering Manufacturers Association released a "standard" for programmable microcontrollers, including almost any computer-based controllers, such as single board computers, numerical, and event-based controllers.

As the cost of microprocessors and microcontrollers fell it became feasible to replace expensive knob-based analog components such as potentiometers and capacitors with up/down buttons or knobs read out by a microprocessor even in some consumer products. By the mid-1980s, most of the common previously external system components had been integrated into the same chip as the processor and this modern form of the microcontroller allowed an even more widespread use, which by the end of the decade were the norm rather than the exception for almost all electronics devices.

The integration of microcontrollers has further increased the applications for which embedded systems are used into areas where traditionally a computer would not have been considered. A general purpose and comparatively low-cost microcontroller may often be programmed to fulfill the same role as a large number of separate components. Although in this context an embedded system is usually more complex than a traditional solution, most of the complexity is contained within the microcontroller itself. Very few additional components may be needed and most of the design effort is in the software. The intangible nature of software makes it much easier to prototype and test new revisions compared with the design and construction of a new circuit not using an embedded processor.

3. BLOCK DIAGRAM OF THE PROJECT:

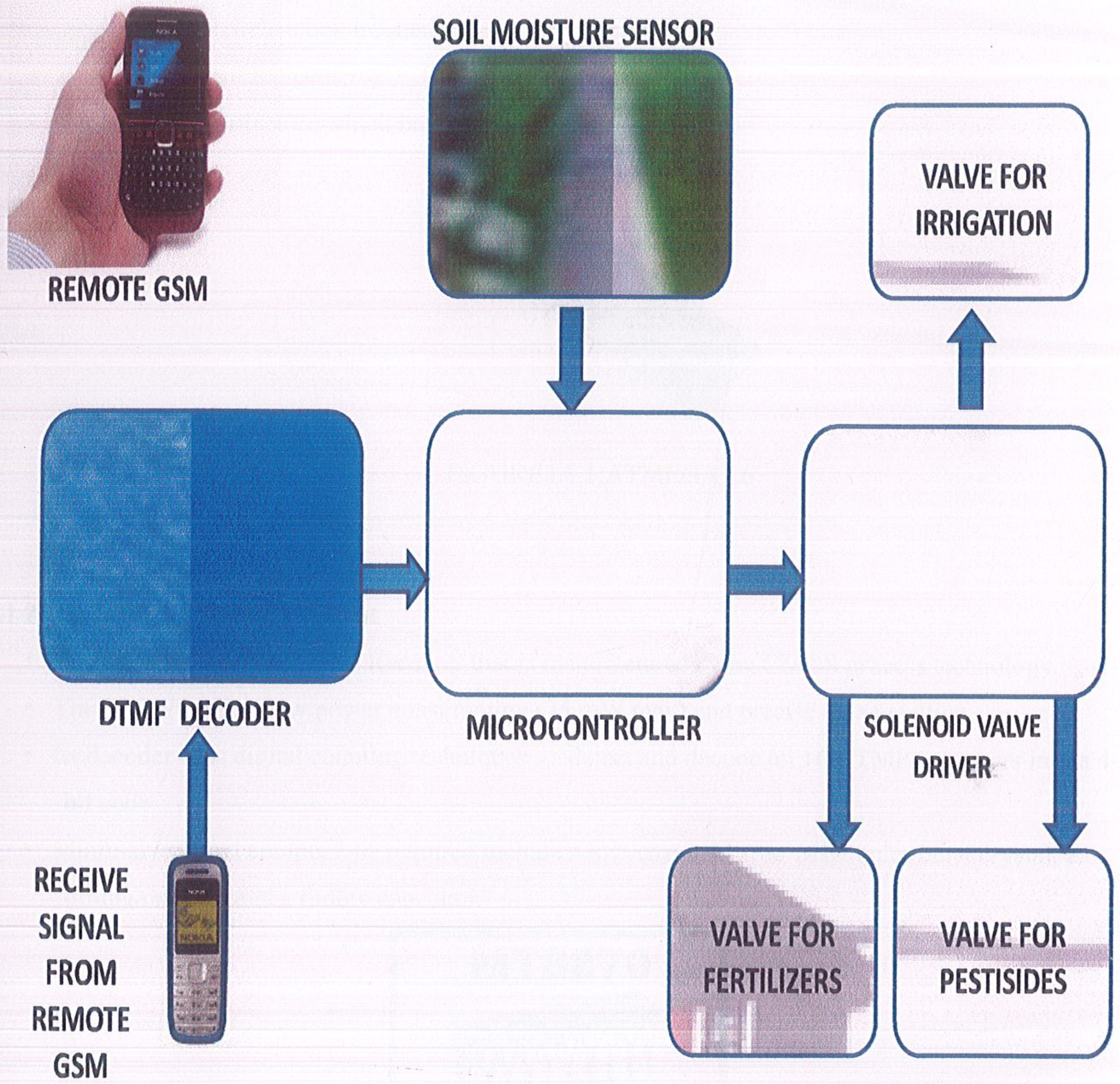


FIG 3.1: BLOCK DIAGRAM OF THE PROJECT

3.1 : COMPONENTS USED:

3.1.1 : ATMEGA16 MICROCONTROLLER

- ATMEGA16 is a MEGA series AVR microcontroller with 16kb of memory, 40 pins with 32 I/O pins and 0-16 MHz clock frequency.
- It also has inbuilt 8 channel ADC(analog to digital converter) .
- It has RISC architecture which has less instruction.

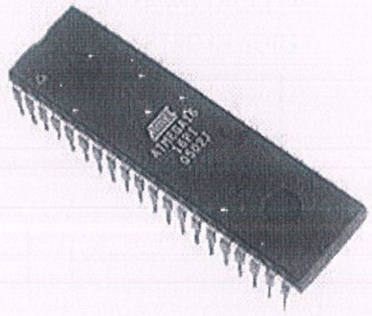


FIGURE3.1.1:ATMEGA 16

3.1.2 : M-8870 DTMF DECODER

- The M-8870 is a full DTMF Receiver that is manufactured using CMOS process technology.
- The M-8870 offers low power consumption (35 mW max) and precise data handling.
- Its decoder uses digital counting techniques to detect and decode all 16 DTMF tone pairs into a 4-bit code.
- Minimal external components required include a low-cost 3.579545 MHz color burst crystal, a timing resistor, and a timing capacitor.

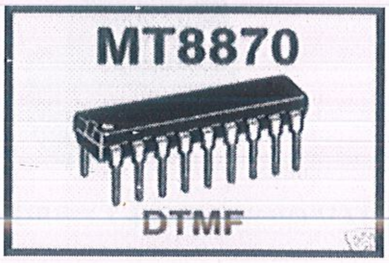


FIGURE3.1.2: DTMF DECODER

3.1.4 : SOIL MOISTURE SENSOR

- This self made soil moisture sensor is an analog sensor that converts the soil moisture into standard output voltages which can be measured through the ADC channels of Atmega8.
- With the use of the experiments and practical implementation these analog voltages can be converted back to the soil moisture values.
- Silver Metal clips are used in the sensor so that maximum conductivity is obtained.

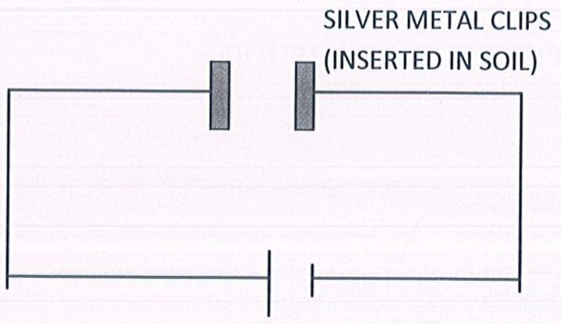


FIGURE 3.1.3:SOIL MOISTURE SENSOR

3.1.5 : SOLENOID VALVES

- This solenoid valve works on the principle of solenoid and will work as electronic taps.
- Will be used in project for the purpose of irrigation, fertilization and pest control.

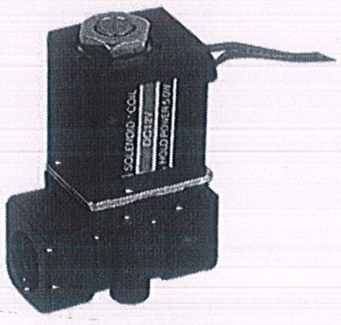


FIGURE 3.1.4: SOLENOID VALVE

3.1.6 : 16x2 LCD

- LCD display to show the running operation of the model and the analog sensor values

FIGURE 3.1.5: LCD DISPLAY

3.1.7 : IC 7805

- 7805 is used in electronic circuits requiring a regulated power supply of 5 volts.

FIGURE 3.1.6: VOLTAGE REGULATOR

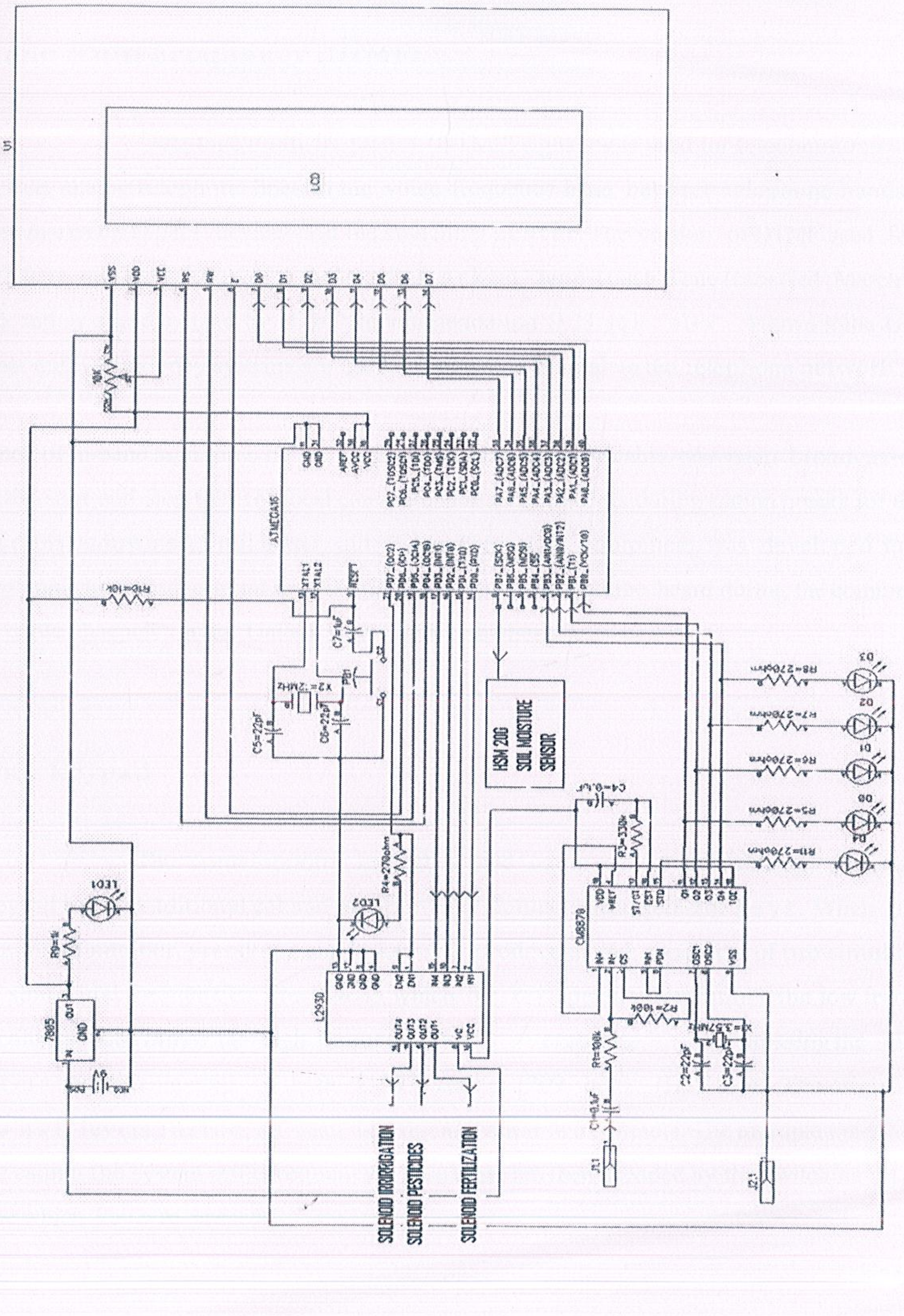
RESISTORS:

VALUE	QUANTITY
100K	2
10K	5
330K	1

CAPACITORS:

VALUE	QUANTITY
0.1 UF	2
22 pF	4

3.2 : CIRCUIT DIAGRAM



3.3 : DTMF DECODER

Dual-Tone Multi-Frequency (DTMF)

Dual-tone multi-frequency (DTMF) signaling is used for telecommunication signaling over analog telephone lines in the voice-frequency band between telephone handsets and other communications devices and the switching center. The version of DTMF used for telephone tone dialing is known by the trademarked term Touch-Tone (canceled March 13,1984), and is standardized by ITU-T Recommendation Q.23. It is also known inthe UK as MF4. Other multi-frequency systems are used for signalinginternal to the telephone network..

As a method of in-band signaling, DTMF tones were also used by cable television broadcasters to indicate the start and stop timcs of local commercial insertionpoints during station breaks for the benefit of cable companies. Until better out-of-band signaling equipment was developed in the 1990s, fast, unacknowledged, and loud DTMF tone sequences couldbe heard during the commercial breaks of cable channels in the United States and elsewhere.

Telephone Keypad

The contemporary keypad is laid out in a 3x4 grid, although the original DTMF keypad had an additional column for four now-defunct menu selector keys. When used to dial a telephone number, pressing a single key will produce a pitch consisting of two simultaneous pure tone sinusoidal frequencies. The row in which the key appears determines the low frequency, and the column determines the high frequency. For example, pressing the !1! key will result in a sound composed of both a 697 and a 1209 hertz (Hz) tone. The original keypads had levers inside, so each button activated two contacts. The multiple tones are the reason for calling the system multifrequency. These tones are then decoded by the switching center to determine which key was pressed.

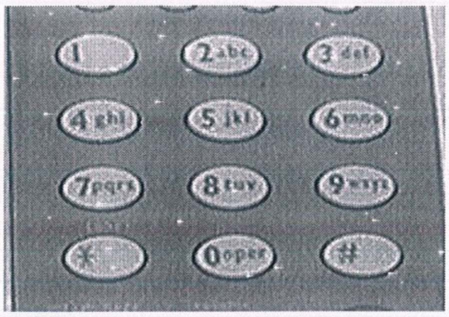


Figure 3.3.1:A DTMF Telephone Keypad

DTMF Keypad Frequencies (WithSound Clips)

	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

DTMF Event Frequencies

Event	Low Freq.	High Freq.
Busy Signal	480 Hz	620 Hz
Dial Tone	350 Hz	440 Hz
Ringback Tone(US)	440 Hz	480 Hz

Tones #, *, A, B, C, and D

The engineers had envisioned phones being used \to access computers, and surveyed anumberof companies to see what they would need for this role. This led to the addition ofthe number sign (#, sometimes called !octothorpe! in this context) and asterisk or 'star' (*) keys as well as a group of keys for menu selection: A, B, C and D. In the end, the lettered keys were dropped from most phones, and it was many years before these keys became widely used for vertical service codes such as *67 in the United States and Canada to suppress caller ID.

PROBLEM DESCRIPTION

This project aims at developing a DTMF BASED DEVICE CONTROL which involves the control of the devices connected to the microcontroller by a handset.

The phone will transmit a particular frequency corresponding to the key pressed to the decoder section, which will in turn provide a BCD output corresponding to each frequency. In accordance to the output, different devices will be switched on or off.

If key ‘1’ is pressed form the handset, the device1 will be switched on. In case, key ‘2’ is pressed form the handset, the device2 will be switched on. If these keys are pressed again the corresponding devices will be switched off. Same is the case for the key’3’ and key ‘4’.

Using this project we can control, both the switching on and switching off the devices connected to the microcontroller.

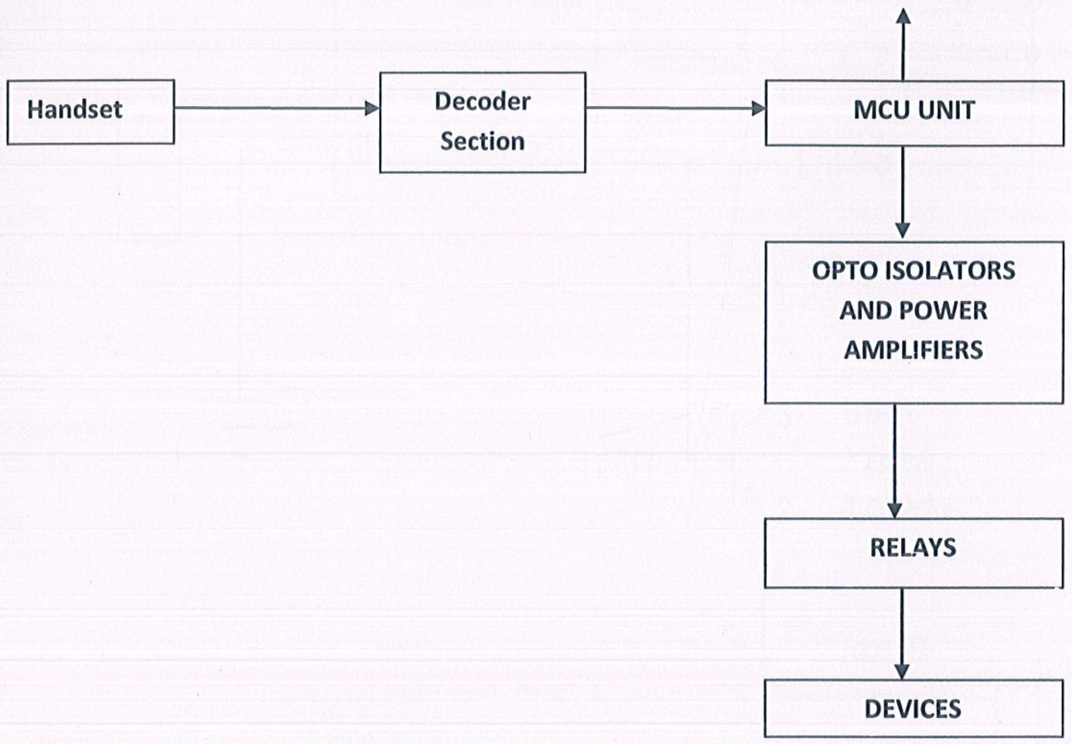


FIG 3.3.2 : BLOCK DIAGRAM OF DTMF DECODER

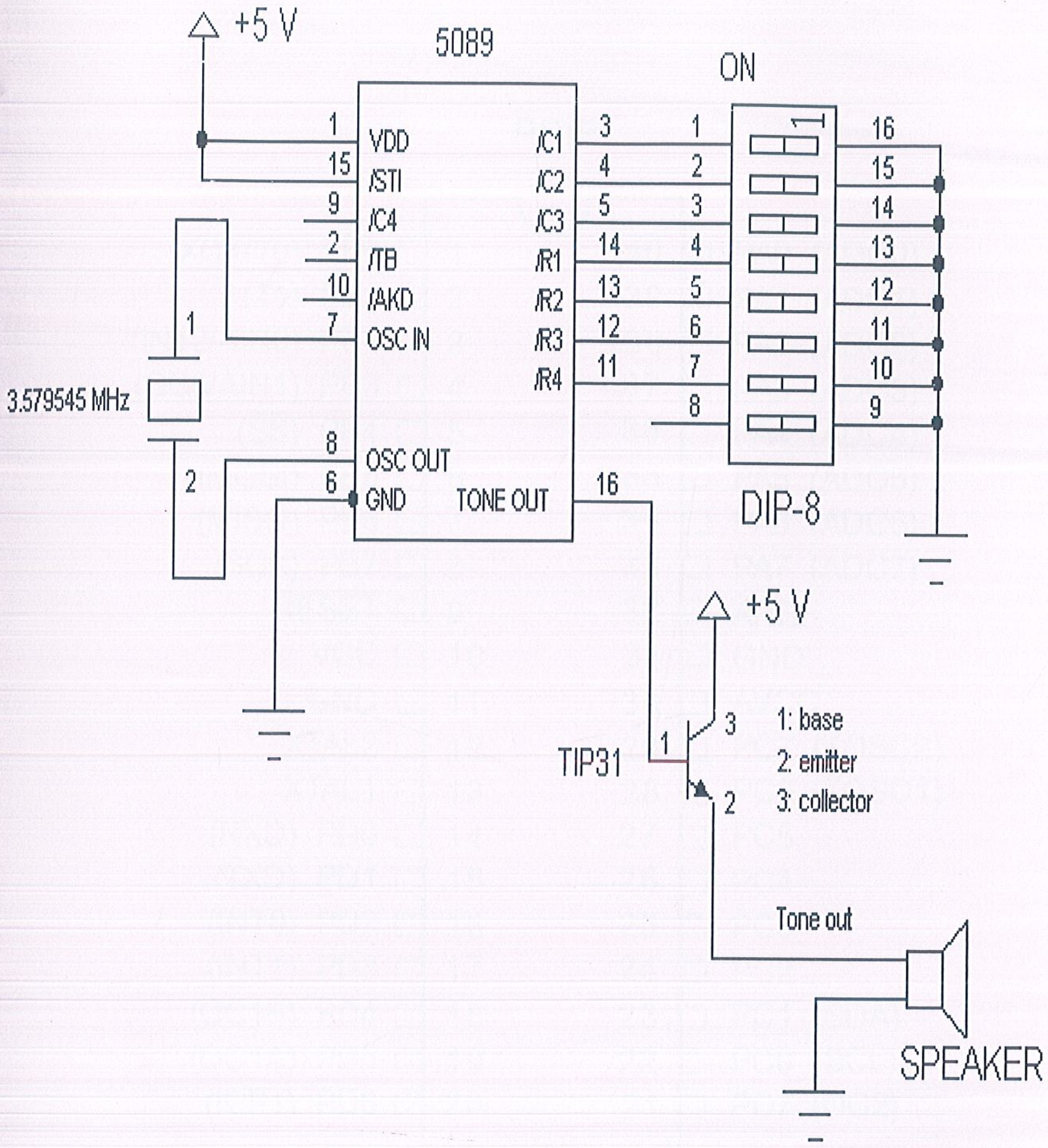


FIG:3.3.3 CIRCUIT OF DTMF DECODER

4 : DESCRIPTION OF ATMEGA 8(ATMEGA 8535)

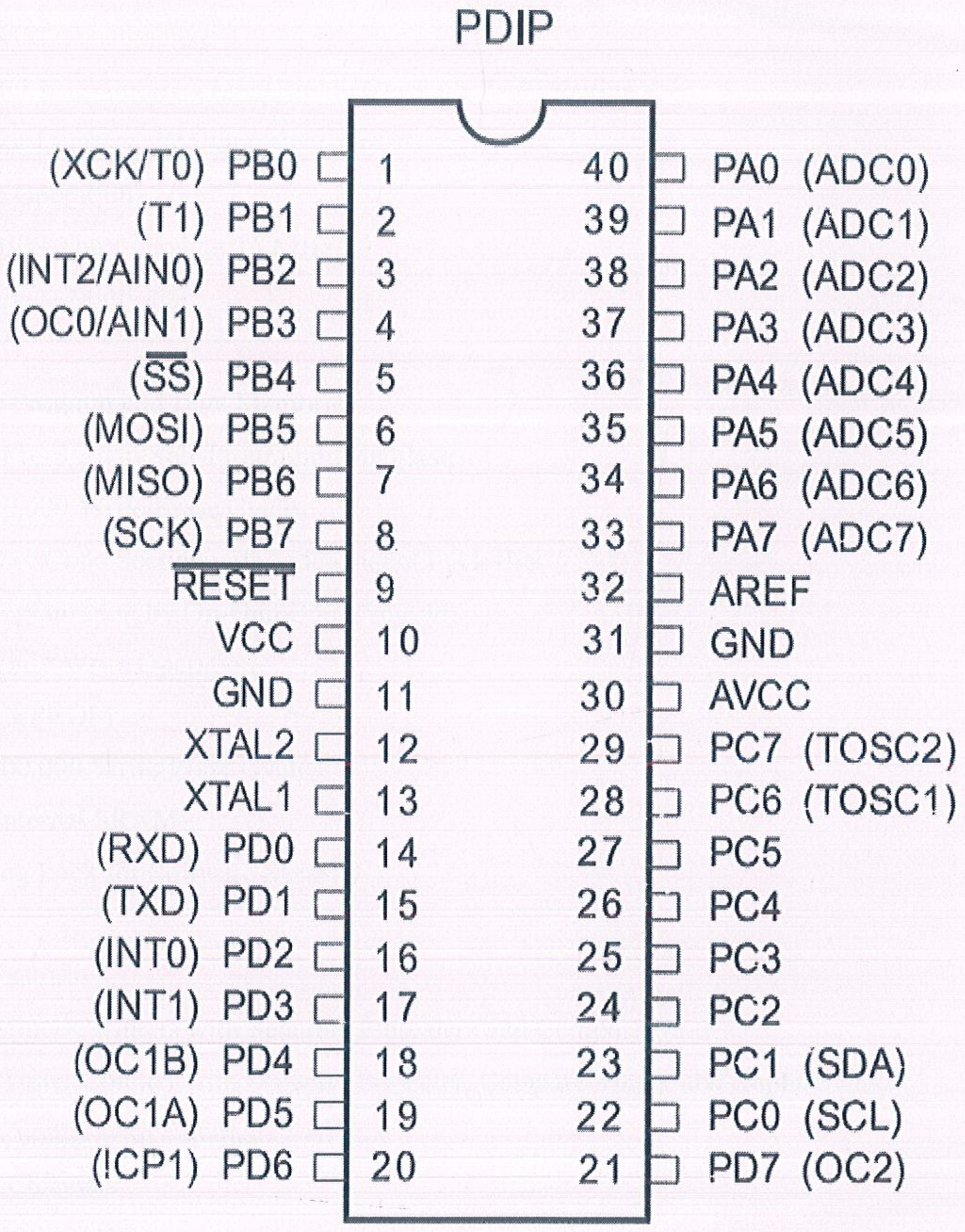


FIG 4.1 : PIN DIAGRAM OF ATMEGA

4.1 : FEATURES OF ATEMGA 8:

• High-performance, Low-power AVR 8-bit Microcontroller

• Advanced RISC Architecture

– 130 Powerful Instructions – Most Single Clock Cycle Execution

– 32 x 8 General Purpose Working Registers

– Fully Static Operation

– Up to 16 MIPS Throughput at 16 MHz

– On-chip 2-cycle Multiplier

• Nonvolatile Program and Data Memories

– 8K Bytes of In-System Self-Programmable Flash

Endurance: 10,000 Write/Erase Cycles

– Optional Boot Code Section with Independent Lock Bits

In-System Programming by On-chip Boot Program

True Read-While-Write Operation

– 512 Bytes EEPROM

Endurance: 100,000 Write/Erase Cycles

– 512 Bytes Internal SRAM

– Programming Lock for Software Security

• Peripheral Features

– Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes

– One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode

– Real Time Counter with Separate Oscillator

– Four PWM Channels

– 8-channel, 10-bit ADC

8 Single-ended Channels

7 Differential Channels for TQFP Package Only

2 Differential Channels with Programmable Gain at 1x, 10x, or 200x for TQFP Package Only

– Byte-oriented Two-wire Serial Interface

– Programmable Serial USART

– Master/Slave SPI Serial Interface

– Programmable Watchdog Timer with Separate On-chip Oscillator

– On-chip Analog Comparator

• Special Microcontroller Features

– Power-on Reset and Programmable Brown-out Detection

– Internal Calibrated RC Oscillator

– External and Internal Interrupt Sources

– Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby

• I/O and Packages

– 32 Programmable I/O Lines

– 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad QFN/MLF

• Operating Voltages

– 4.5 - 5.5V for ATmega8535

• Speed Grades

– 0 - 16 MHz for ATmega8535

4.2 : MICROCONTROLLER VERSUS MICROPROCESSOR

Microcontrollers are frequently found in automobiles, office machines, toys, and appliances.

The microcontroller is the integration of a number of useful functions into a single IC package. These functions are:

- The ability to execute a stored set of instructions to carry out user defined tasks.

- The ability to be able to access external memory chips to both read and write data from and to the memory.

Basically, a microcontroller is a device which integrates a number of the components of a microprocessor system onto a single microchip.

So a microcontroller combines onto the same microchip :

- The CPU core (microprocessor)
- Memory (both ROM and RAM)
- Some parallel digital I/O
- Also, a microcontroller is part of an embedded system, which is essentially the whole circuit board. Look up "embedded system" on Wikipedia.
- The difference is that microcontroller incorporates features of microprocessor(CPU,ALU,Registers)along with the presence of added features like presence of RAM,ROM,I\O ports,counter etc.Here microcontroller control the operation of machine using fixed programme stored in Rom that doesn't change with lifetime

4.3 : INTRODUCTION TO ATMEGA 8:

The AVR core combines a rich instruction set with 32 general purpose working registers. All 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega8535 provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes EEPROM, 512 bytes SRAM, 32general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain in TQFP package, a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. ThePower-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while

the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the asynchronous timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is

updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega8535 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.
The ATmega8535 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, In-Circuit Emulators, and evaluation kits.

4.4 : PIN DESCRIPTION:

- **VCC** Digital supply voltage.

- **GND** Ground.

- **Port A (PA7..PA0)** Port A serves as the analog inputs to the A/D Converter. Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

- **Port B (PB7..PB0)** Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

- **Port C (PC7..PC0)** Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

- **Port D (PD7..PD0)** Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

- **RESET** Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running.

- **XTAL1** Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

- **XTAL2** Output from the inverting Oscillator amplifier.

- **AVCC** AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.

- **AREF** AREF is the analog reference pin for the A/D Converter.

4.5 : INSIDE THE ATMEGA 8:

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

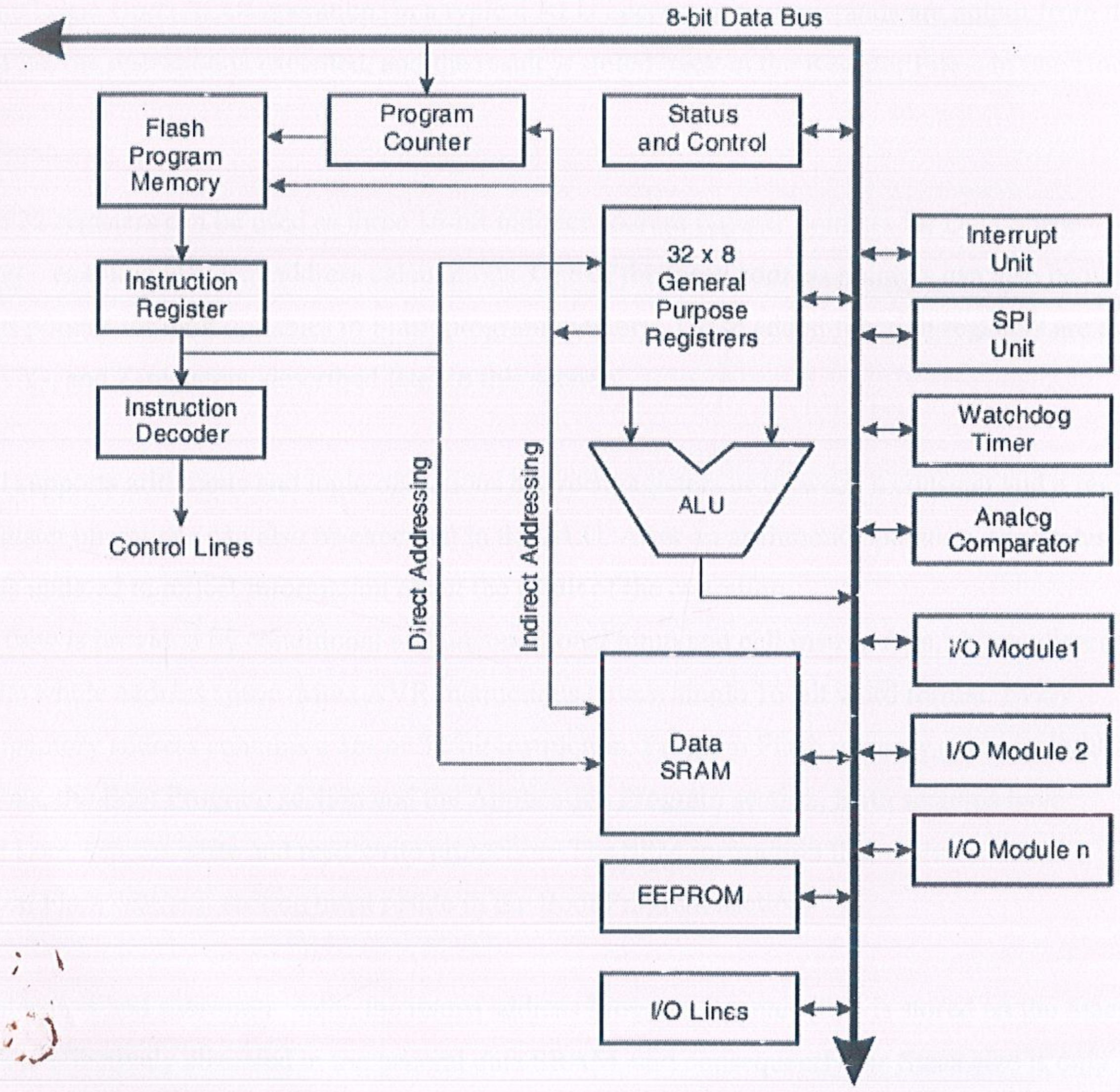


FIG 4.5.1: BLOCK DIAGRAM OF ATMEGA 8 ARCHITECHTURE

In order to maximize performance and parallelism, the AVR uses a Harvard architecture with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In System Re-Programmable Flash memory. The fast-access Register File contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register File, the operation is executed, and the result is stored back in the Register File – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash program memory. These added function registers are the 16-bit X-, Y-, and Z-registers, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.
Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction. Program Flash memory space is divided in two sections, the Boot Program section and the Application Program section. Both sections have dedicated Lock bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot Program section.

During interrupts and subroutine calls, the return address Program Counter (PC) is stored on the Stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The Stack Pointer SP is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.
The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority. The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register File, 0x20 - 0x5F.

ALU – Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the "Instruc-tion Set" section for a detailed description.

Status Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the Status Register is updated after all ALU operations, as specified in the Instruction Set Reference. This will, in many cases, remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The Status Register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

The AVR Status Register – SREG – is defined as:

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

• **Bit 7** – I: Global Interrupt Enable

The Global Interrupt Enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the Global Interrupt Enable Register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the instruction set reference.

• **Bit 6** – T: Bit Copy Storage

The Bit Copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T-bit as source or destination for the operated bit. A bit from a register in the Register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register File by the BLD instruction.

• **Bit 5** – H: Half Carry Flag

The Half Carry Flag H indicates a Half Carry in some arithmetic operations. Half carry is useful in BCD arithmetic. See the "Instruction Set Description" for detailed information.

• **Bit 4** – S: Sign Bit, $S = N \oplus V$

The S-bit is always an exclusive or between the Negative Flag N and the Two's Complement Overflow Flag V. See the "Instruction Set Description" for detailed information.

• **Bit 3** – V: Two's Complement Overflow Flag

The Two's Complement Overflow Flag V supports two's complement arithmetics. See the "Instruction Set Description" for detailed information.

• **Bit 2** – N: Negative Flag

The Negative Flag N indicates a negative result in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

• **Bit 1** – Z: Zero Flag

The Zero Flag Z indicates a zero result in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

• **Bit 0** – C: Carry Flag

The Carry Flag C indicates a carry in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

General Purpose

Register File

The Register File is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register File:

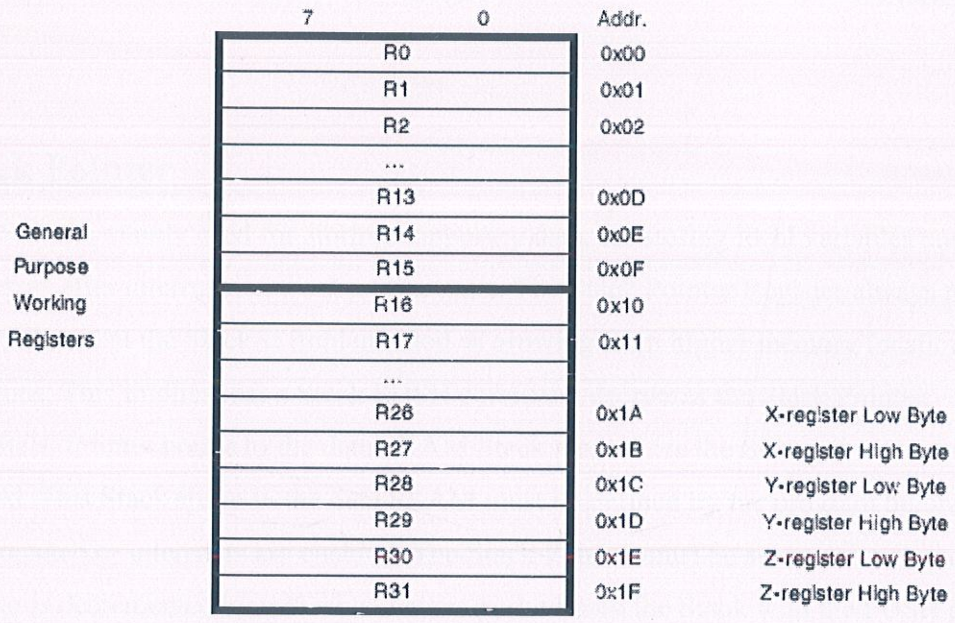
• One 8-bit output operand and one 8-bit result input

• Two 8-bit output operands and one 8-bit result input

• Two 8-bit output operands and one 16-bit result input

• One 16-bit output operand and one 16-bit result input

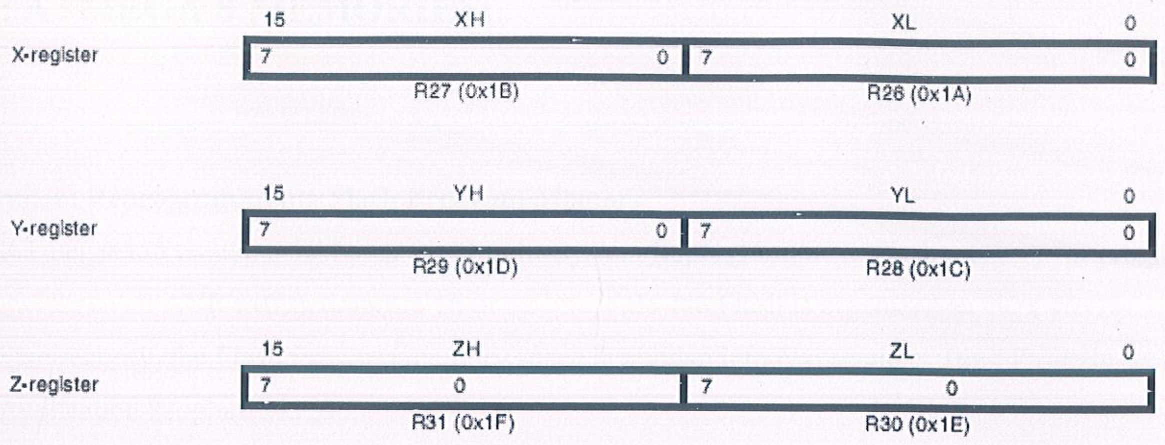
Figure shows the structure of the 32 general purpose working registers in the CPU.



The X-register, Y-register, and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the Data Space.

The three indirect address registers X, Y, and Z are defined as described in Figure .



Stack Pointer

The Stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. The Stack Pointer Register always points to the top of the Stack. Note that the Stack is implemented as growing from higher memory locations to lower memory locations. This implies that a Stack PUSH command decreases the Stack Pointer.

The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. This Stack space in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. The Stack Pointer must be set to point above 0x60. The Stack Pointer is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when the return address is pushed onto the Stack with subroutine call or interrupt. The Stack Pointer is incremented by one when data is popped from the Stack with the POP instruction, and it is incremented by two when data is popped from the Stack with return from subroutine RET or return from interrupt RETI.

The AVR Stack Pointer is implemented as two 8-bit registers in the I/O space. The num ber of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH Register will not be present.

4.6: ATMEGA 8 MEMORIES:

In-System Reprogrammable Flash Program Memory

The ATmega8535 contains 8K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 4K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Program section and Application Program section.

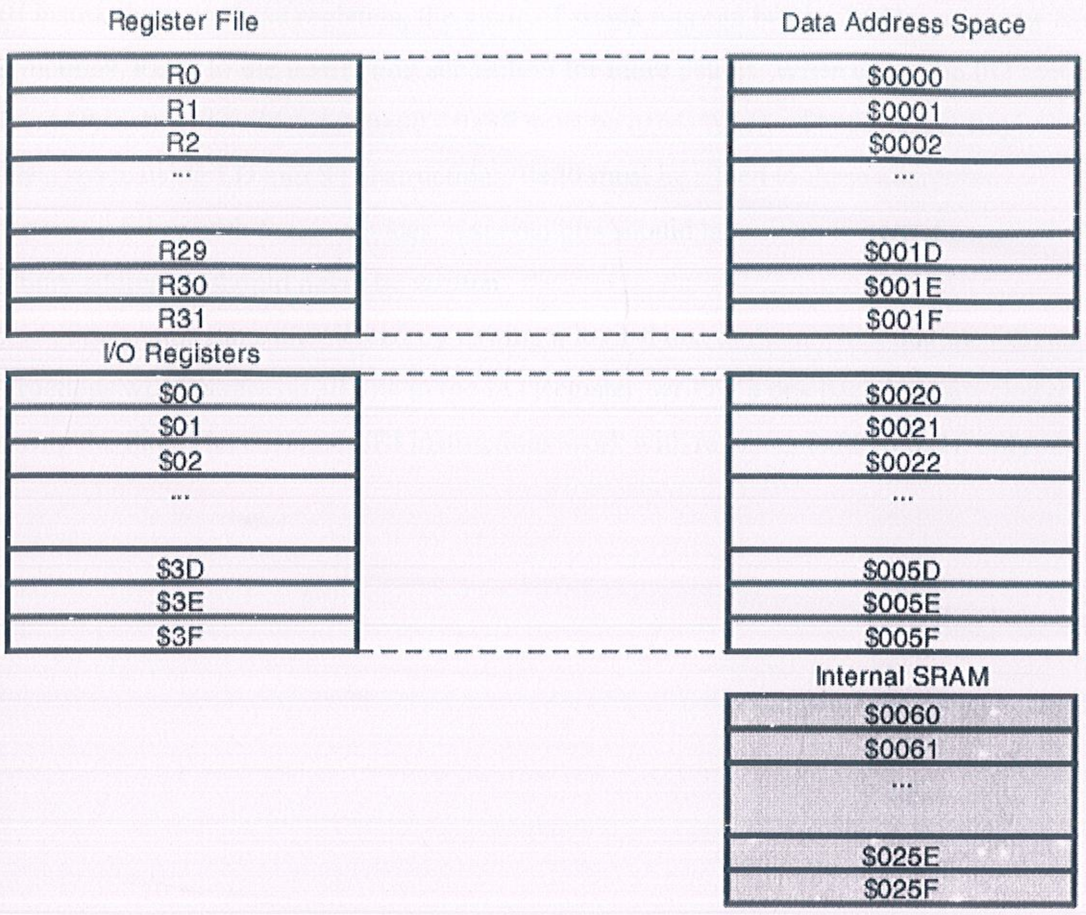
The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATmega8535 Program Counter (PC) is 12 bits wide, thus addressing the 4K program memory locations.

SRAM Data Memory

Figure shows how the ATmega8535 SRAM Memory is organized. The 608 Data Memory locations address the Register File, the I/O Memory, and the internal data SRAM. The first 96 locations address the Register File and I/O Memory, and the next 512 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement, and Indirect with Post-increment. In the Register File, registers R26 to R31 feature the indirect addressing pointer registers. The direct addressing reaches the entire data space. The Indirect with Displacement mode reaches 63 address locations from the base address given by the Y- or Z-register. When using register indirect addressing modes with automatic pre-decrement and postincrement, the address registers X, Y, and Z are decremented or incremented.

The 32 general purpose working registers, 64 I/O Registers, and the 512 bytes of internal data SRAM in the ATmega8535 are all accessible through all these addressing modes.



EEPROM Data Memory

The ATmega8535 contains 512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register. "Memory Programming" on page 237 contains a detailed description on EEPROM Programming in SPI or Parallel Programming mode.

I/O Memory

All ATmega8535 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O Registers within the address range 0x00 -0x1F are directly bit-accessible using the SBI and

CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these addresses.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O Register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers 0x00 to 0x1F only

5.POWER SUPPLY

POWER SUPPLY DESCRIPTION:

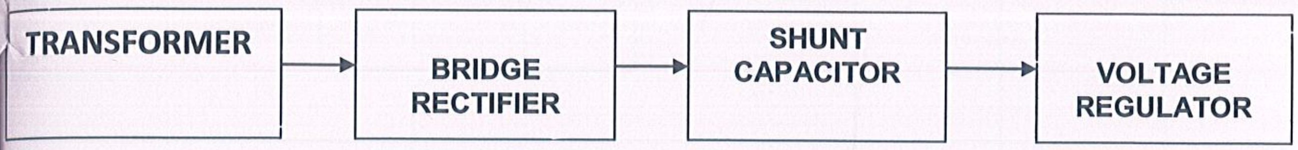


FIG. 5.1 : BLOCK DIAGRAM OF POWER SUPPLY

The power supply circuit comprises of four basic parts:

The transformer steps down the 220 V a/c. into 12 V a/c. The transformer work on the principle of magnetic induction, where two coils: primary and secondary are wound around an iron core. The two coils are physically insulated from each other in such a way that passing an a/c. current through the primary coil creates a changing voltage in the primary coil and a changing magnetic field in the core. This in turn induces a varying a/c. voltage in the secondary coil.

The a/c. voltage is then fed to the bridge rectifier. The rectifier circuit is used in most electronic power supplies is the single-phase bridge rectifier with capacitor filtering, usually followed by a linear voltage regulator. A rectifier circuit is necessary to convert a signal having zero average value into a non-zero average value. A rectifier transforms alternating current into direct current by limiting or regulating the direction of flow of current. The output resulting from a rectifier is a pulsating D.C. voltage. This voltage is not appropriate for the components that are going to work through it.

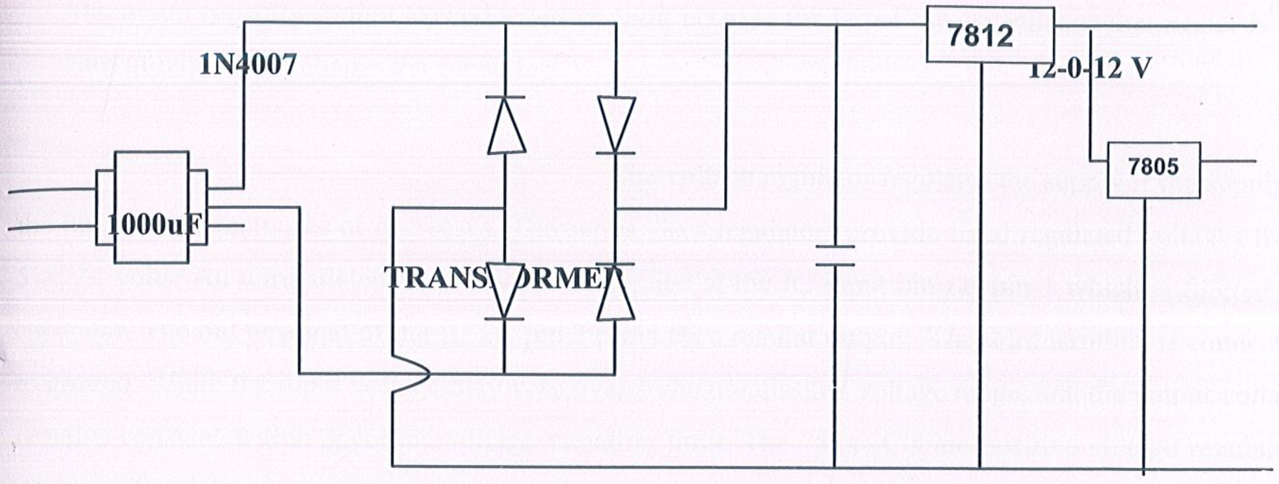


FIG. 5.2 : CIRCUIT DIAGRAM OF POWER SUPPLY

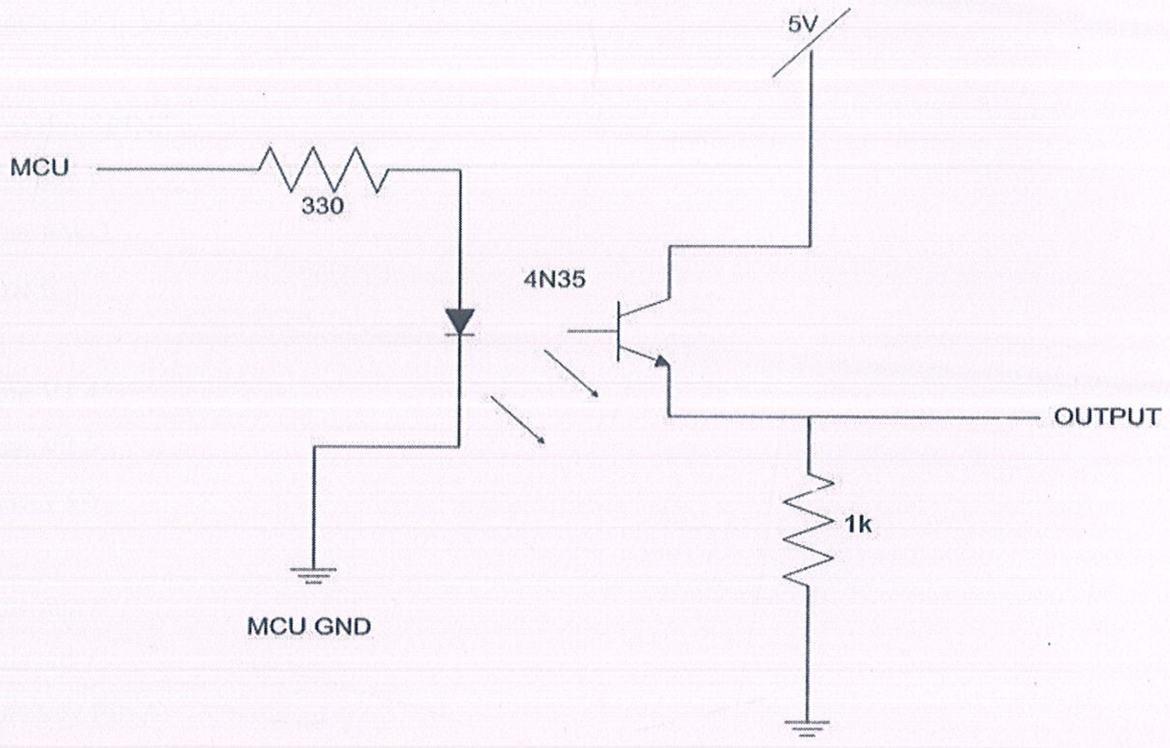
The ripple of the D.C. voltage is smoothened using a filter capacitor of 1000 microF 25V. The filter capacitor stores electrical charge. If it is large enough the capacitor will store charge as the voltage rises and give up the charge as the voltage falls. This has the effect of smoothing out the waveform and provides steadier voltage output. A filter capacitor is connected at the rectifier output and the d.c voltage is obtained across the capacitor. When this capacitor is used in this project, it should be twice the supply voltage. When the filter is used, the RC charge time of the filter capacitor must be short and the RC discharge time must be long to eliminate ripple action. In other words the capacitor must charge up fast, preferably with no discharge.

- When the rectifier output voltage is increasing, the capacitor charges to the peak voltage Vm. Just past the positive peak, the rectifier output voltage starts to fall but at this point the capacitor has +Vm voltage across it. Since the source voltage becomes slightly less than Vm, the capacitor will try to send current back through the diode of rectifier. This reverse biases the diode. The diode disconnects or separates the source the source form load. The capacitor starts to discharge through load. This prevents the load voltage from falling to zero. The capacitor continues to discharge until source voltage becomes more than capacitor voltage. The diode again starts conducting and the capacitor is again charged to peak value Vm. When capacitor is charging the rectifier supplies the charging through capacitor branch as well as load current, the capacitor sends currents through the load. The rate at which capacitor discharge depends upon time constant RC. The longer the time constant, the steadier is the output voltage. An increase in load current i.e. decrease in resistance makes time constant of discharge path smaller. The ripple increase and d.c output voltage V dc decreases. Maximum capacity cannot exceed a certain limit because the larger the capacitance the greater is the current required to charge the capacitor.

The voltage regulator regulates the supply if the supply if the line voltage increases or decreases. The series 78xx regulators provide fixed regulated voltages from 5 to 24 volts. An unregulated input voltage is applied at the IC Input pin i.e. pin 1 which is filtered by capacitor. The out terminal of the IC i.e. pin 3 provides a regular output. The third terminal is connected to ground. While the input voltage may vary over some permissible voltage range, and the output voltage remains constant within specified voltage variation limit. The 78xx IC's are positive voltage regulators whereas 79xx IC's are negative voltage regulators.

These voltage regulators are integrated circuits designed as fixed voltage regulators for a wide variety of applications. These regulators employ current limiting, thermal shutdown and safe area compensation. With adequate heat sinking they can deliver output currents in excess of 1 A. These regulators have internal thermal overload protection. It uses output transistor safe area compensation and the output voltage offered is in 2% and 4% tolerance.

6. OPTO ISOLATOR CIRCUIT :



In electronics, an **opto-isolator**, also called an **optocoupler**, **photocoupler**, or **optical isolator**, is "an electronic device designed to transfer electrical signals by utilizing light waves to provide coupling with electrical isolation between its input and output". The main purpose of an opto-isolator is "to prevent high voltages or rapidly changing voltages on one side of the circuit from damaging components or distorting transmissions on the other side."Commercially available opto-isolators withstand input-to-output voltages up to 10 kVand voltage transients with speeds up to 10 kV/μs

7. CODE

```
#define DATA PORTB
#define CONT_LCD PORTD

#define dtmf PINC
#define RS 1
#define RW 2
#define E 3

#define V1 4
#define V2 5
#define V3 6

#define msr 0
#define std 7
#define key PINA
#include<lcdrout.h>
#include<ADC.h>

void main()
{
        unsigned char adc_data;
        unsigned char v1,v2,v3;
        adc_init();
        lcd_init();
        v1=v2=v3=0;
        DDRC = 0x00;
        DDRB = 0xFF;
        DDRD = DDRD | ((1<<V1)|(1<<V2)|(1<<V3));
        DDRA  = DDRA & ~((1<<std));
```

```
PORTD=PORTD | ((1<<V1)|(1<<V2)|(1<<V3));

lcd_init();

lcd_cmd(0x80);
lcd_puts("  GREEN  HOUSE  ");
lcd_cmd(0xc0);
lcd_puts(" CONTROL SYSTEM ");
secdelay(3);
lcd_cmd(0x01);

        lcd_cmd(0x80);
        lcd_puts("V1     V2     V3");
while(1)
        {
                PORTC=0xff;

        whiie( ((1<<std) & key) !=0)
                {
                        adc_data=get_adc();
                        if( adc_data<150)
                                {
                                        PORTD=PORTD | (1<<V1);
                                        v1=0;
                                        lcd_cmd(0xc0);
                                        lcd_puts("OFF");
                                }
                        else if(adc_data>150)
                                {
                                        v1=1;
                                        PORTD=PORTD & ~(1<<V1);
                                        lcd_cmd(0xc0);
                        lcd_puts("ON ");
                }
        }


                if(PINC==1)
                {
                        v2=1;
                        PORTD=PORTD & ~(1<<V2);


                }

                else if(PINC==2)
                {
                        v2=0;
                        PORTD=PORTD | (1<<V2);


                }

                else if(PINC==3)
                {
                        v3=1;
                        PORTD=PORTD & ~(1<<V3);


                }
                else if(PINC==4)
                {
                        v3=0;
                        PORTD=PORTD | (1<<V3);
                }

        if (v1==0 && v2==0 && v3==0)
                {
                        lcd_cmd(0xc0);
                        lcd_puts("OFF   OFF   OFF");


            }
    if (v1==0 && v2==0 && v3==1)
            {
                        lcd_cmd(0xc0);
                        lcd_puts("OFF   OFF    ON");


            }
    if (v1==0 && v2==1 && v3==0)
            {
                        lcd_cmd(0xc0);
                        lcd_puts("OFF    ON   OFF");


            }
    if (v1==0 && v2==1 && v3==1)
            {
                        lcd_cmd(0xc0);
                        lcd_puts("OFF    ON    ON");


            }
    if (v1==1 && v2==0 && v3==0)
            {
                        lcd_cmd(0xc0);
                        lcd_puts("ON    OFF   OFF");


            }
    if (v1==1 && v2==0 && v3==1)
            {
                        lcd_cmd(0xc0);
                        lcd_puts("ON   OFF    ON");
```

```
                }
        if (v1==1 && v2==1 && v3==0)
                {
                                lcd_cmd(0xc0);
                                lcd_puts("ON    ON   OFF");

                }
        if (v1==1 && v2==1 && v3==1)
                {
                                lcd_cmd(0xc0);
                                lcd_puts("ON    ON    ON");


                }



        }
}
```

BENEFITS OF THE PROJECT

- **No inescapable delays in irrigation and fertilizing the fields.**
- **Even illiterate farmers can use the system without any difficulties.**
- **Reduces physical work of a farmer**
- **Saves Time**

  An automatic irrigation system will save you plenty of time that you in the past would have spent watering your lawns, gardens and flowers. You can now have your timers set, so that watering will take place at the times that best suits your landscape and the climate where you live. You can go on that holiday knowing that your lawns and flowers will be maintained and flourishing when you return.

- **Insures proper enrichment of the soil and hence better yields.**
- **Reduces labour cost of the farmer with one time investment.**
- **Negligible maintenance and operational cost.**
- **Saves Money**

  With an automatic irrigation system there is no money or water wasted, for everything is timed, programmed and these systems all have rain sensors, so every drop of water is used only when it is needed.

- **Saves Water**

Whatever type of irrigation system you install, there will definitely be a greater saving on water. You can help conserve water with automatic systems, for there is no wasting of water, every drop is used not wasted away. You can save between 30 and 50 percent of the water that you would normally use with other more conventional watering methods.

- **Improves Growth**

When plants, crops, lawns or flowers are watered with smaller amounts of water over a longer period of time, they grow faster, for it is the ideal condition for growth. You will enjoy greener and more luscious gardens and lawns

- **Weed Reduction**

You will notice a reduction in the amount of weeds appearing, this is due to the fact that those areas that need water are the only areas receiving water, with the implementation of a specifically designed irrigation system.

PROBLEMS FACED

- Components not available in nearby areas.
- Failure of the HSM 20G soil moisture sensor.
- Rare availability of soil moisture sensor.
- We could not fabricate the whole circuit on 1 PCB because the circuit was to complex to be auto routed on the DEMO version of EAGLE CAD software.

CONCLUSIONS

The system made, will be the boon for the farmers of our country if commercially implemented as unlike other present systems, it is a low cost system.

Further in present world scenario where renewable resources are also depleting and things like conservation of water plays a major role, this system will serve a great purpose.

The major benefit of this system is that it can be operated even by our illiterate farmers.

In future this system can be made crop specific by researching and assigning the values of sensor according to the requirement of crop.

Also, for play fields like golf course, football field, cricket field this system can be very beneficial.

REFERENCES

- Microcontroller and embedded system,pearson education 2nd edition,2008
- www.8051projects.net/microcontroller_tutorials
- www.8051projects.net/lcd interfacing
- www.youtube.com for EAGLECAD tutorials.
- www.google.com
- Excellent guidance by our project guide Mr. Munish Sood
- Mr. Pramod(advanced communication lab) for helping us with the circuit.
- Research paper by L Feuer on soil moisture sensor.