1. INTRODUCTION

1.1 OVERVIEW

This project deals with protection of data in a system, security for homes based on embedded systems. digital code lock provides security to the home and also to other sensitive data. since security is major concern in our day to day life, and digital locks have became an important part of these security systems.

An embedded system is a system which is going to do a predefined specified task is the embedded system and is even defined as combination of both software and hardware. A general-purpose definition of embedded systems is that they are devices used to control, monitor or assist the operation of equipment, machinery or plant. The very simplest embedded systems are capable of performing only a single function or set of functions to meet a single predetermined purpose.

This arduino is a platform for both software and hardware. It consists of microcontroller, USB serial communication pins, external supply pins. Its programming is done in arduino software. Arduino Uno contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

1.2 AIM OF THE PROJECT

The overall objective of the project is to deliver security to the user at and achieve continuity and maximum coverage at affordable cost without any loss by developing an embedded based password based locking system.

1.3 METHODOLOGY

The project presented here is based on arduino and is more simple and reliable than simple micro controller based digital code lock. Here is a LCD display which is used to interface with the project to output lock status. In this project we have an additional advantage that the user can change the password. The user will be prompted to set a password at installation. This password inputted at installation will continue to serve the lock until it is change. The program will check for current password and allows the user to change Password only if the current password is input correctly. Applications: It can be used in places where we need more security. It can also be used in door, lockers, offices, main gate of house, ATM etc.

1.4 SIGNIFICANCE OF WORK

Security represents protection of our life and assets. Ensuring safety of peoples and their valuable things is very important for the prevention of illegal handling. Hence, mainly focusing on door lock security or gate security is very important to avoid the further problems in monitored area [2]. Even with the use of mechanical locks, the crime, robberies get happened due to the fact that such locks were easily broken. So, there is a need to invent other kind of locks which cannot be easily broken. So, many authors present different kinds of digital door locks, automatic password based door locks, software based door locks etc. which have been widely used in houses and offices.

The prevention of unauthorized entry into buildings through the main doors is done by using ordinary, electronically operated locks, digital codes and biometrics technique like the finger print technology or some are based on thumb printing only. Nowadays, advanced automatic door security systems are available with the use of palmtop recognition systems face recognition systems, face detection systems, wireless sensors, PIR sensors, RFID techniques, smart cameras and many more that helps people to make their home or organizations secure from long distance. Hence, people need not to be worry about the home security though they are away from home.

Doors are to keep people out. They are being made of metals not simply wood any longer. The security sectors are experiencing viciousness as it has never seen before. So, demand is to audit the authenticity of currently available systems and need is to research for the creation of more reliable and good systems which operate smartly with no more efforts. The important thing is to provide higher security.

1.5 ORGANISATION OF THESIS

The thesis explains the implementation of “Digital code lock using arduino”.
Chapter 1 presents introduction to the overall thesis and the overview of the project.
Chapter 2 literature survey and actual problem for this project to be implemented.

Chapter 3 presents the introduction to embedded systems.

Chapter 4 presents hardware description.

Chapter 5 presents software and code description.

Chapter 6 presents working of the project.

Chapter 7 presents the results of the project.

Chapter 8 presents the merits and demerits project.

Chapter 9 presents the applications project.

Chapter 10 presents the conclusion project.

Chapter 11 presents the future scope project.

2. LITERATURE REVIEW AND THEORY

2.1 ORDINARY INTERCOM

Conventionally, the ordinary intercom which installed in our home, have the drawbacks of limited working range, that the owner should be inside the home to control the door lock. Use of a mobile phone for control the door can overcome these limitations. It provides many advantages, working range as large as the coverage area of the service provider, you can control the door lock from anywhere in the world. When entering of numbers is finished, a compare will made by pic to make a decision for open the door or not. In addition, when you forget your keys anywhere and you want to enter the home you can't access the door without them!! In our project you can access it remotely without needing them just by enter the correct password.

Also you can know who enter your home all the day by received alarm message tell you that there is a person with specific ID enter your home, so if there is any crime happens, you can easily know who is it using messages you received in a specific time.

2.2 LITERATURE SURVEY

There are many automated advanced door locking system has been developed and it's popularly used in commercial buildings and organization. Some of these automated doors locking system are based on password. When the password is entered correctly, it identifies and thus verifies the key. However these systems are expensive. Various control systems have been designed over the years to prevent access to unauthorized user. The main aim for providing locks for our home, school, office, and building is for security of our lives and property. It is therefore important to have convenient way of achieving this goal. Automatic door locking system has become a standard feature on many different types of buildings and homes.

Lia Kamelia, Alfin Noorhassan S.R, Mada Sanjaya and W.S., Edi Mulyana has implemented a "Door – Automation System", implementation cost is less and affordable by a common use. Shilpi Banerjee has implemented an "Automatic Password Based Door Lock System". This system works on pre-decided password concept. It increases the security level to prevent an unauthorized unlocking done by attacker. In case the user forgets the passwords, system gives the flexibility to the user to change or reset the password. This automatic password based lock system gives user more secure way of locking-unlocking the system.

Arpita Mishra, Siddharth Sharma, Sachin Dubey, S. K. Dubey has implemented a “Password Based Security Lock System”, the system works using keypad to enter a password to the system. If entered password is correct then door is open by motor which is used to rotate the handle of the door lock. System also includes extra features like adding new users and changing old password etc. We surveyed many smart doors locking system. We found that these products are very expensive. Some of the implementation mentioned in the literature survey is very cost effective in implementation but do not provide multi user or multi level functionalities. We identified these requirements and thought to develop a system which is cost effective in implementation and having more advanced features like multi user and multilevel. These features are the need of time and such functionalities will make the system more useful.

3. INTRODUCTION TO EMBEDDED SYSTEMS

3.1 INTRODUCTION

An embedded system is a system which is going to do a predefined specified task is the embedded system and is even defined as combination of both software and hardware. A general-purpose definition of embedded systems is that they are devices used to control, monitor or assist the operation of equipment, machinery or plant. "Embedded" reflects the fact that they are an integral part of the system. At the other extreme a general-purpose computer may be used to control the operation of a large complex processing plant, and its presence will be obvious.

All embedded systems are including computers or microprocessors. Some of these computers are however very simple systems as compared with a personal computer. The very simplest embedded systems are capable of performing only a single function or set of functions to meet a single predetermined purpose. In more complex systems an application program that enables the embedded system to be used for a particular purpose in a specific application determines the functioning of the embedded system. The ability to have programs means that the same embedded system can be used for a variety of different purposes. In some cases, a microprocessor may be designed in such a way that application software for a particular purpose can be added to the basic software in a second process, after which it is not possible to make further changes. The applications software on such processors is sometimes referred to as firmware.

The simplest devices consist of a single microprocessor (often called a "chip”), which may itself be packaged with other chips in a hybrid system or Application Specific Integrated Circuit (ASIC). Its input comes from a detector or sensor and its output goes to a switch or activator which (for example) may start or stop the operation of a machine or, by operating a valve, may control the flow of fuel to an engine.

Software deals with the languages like ALP, C, and VB etc., and Hardware deals with Processors, Peripherals, and Memory.

Memory: It is used to store data or address.

Peripherals: These are the external devices connected

Processor: It is an IC which is used to perform some task

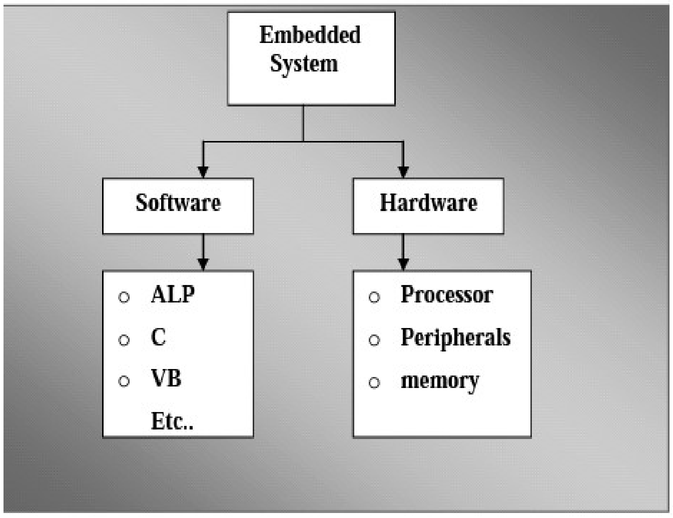

Figure 3.1 Block diagram of Embedded System

3.2 APPLICATIONS OF EMBEDDED SYSTEMS

1. Manufacturing and process control
2. Domestic service
3. Communications
4. Office systems and mobile equipment
5. Banking, finance and commercial
6. Medical diagnostics, monitoring and life support
7. Testing, monitoring and diagnostic systems

3.3 TECHNICAL SPECIFICATIONS

HARDWARE

1. keypad module	:	4x4 matrix keypad
2. buzzer	:	piezoelectric
3. lcd	:	HD44780
4. transistor	:	bc547
5. resistor		
6. breadboard		
7. power supply	:	0-5V
8. connecting wires		

SOFTWARE

1. arduino version : arduino 1.7.10

4. HARDWARE DESCRIPTION

4.1 BLOCK DIAGRAM

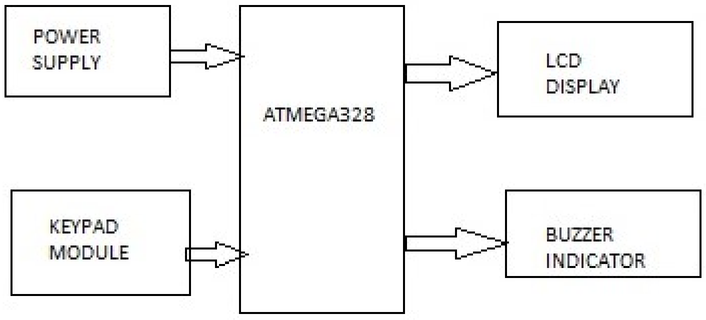

Figure 4.1 block diagram of digital code lock

The block diagram consists of microcontroller atmega328, keypad module, lcd display, buzzer indicator, and an external power supply or supply from laptop via usb cable.

4.2 ARDUINO UNO

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform.

Atmega328 is a low power CMOS 8-bit microcontroller based on the AVR architecture. By executing powerful instructions in a single clock cycle, the atmega328 achieves throughputs approaching 1MIPS per MHz allowing the system designed to optimise power consumption versus preceding speed.

Technical specifications

1.	Microcontroller	ATmega328
2.	Operating Voltage	5V
3.	Input Voltage (recommended)	7-12V
4.	Input Voltage (limits)	6-20V
5.	Digital I/O Pins	14 (of which 6 provide PWM output)
6.	Analog Input Pins	6
7.	DC Current per I/O Pin	40 mA
8.	DC Current for 3.3V Pin	50 mA
9.	Flash Memory	32 KB
10.	SRAM	2 KB
11.	EEPROM	1 KB
12.	Clock Speed	16 MHz

4.3 FEATURES OF ATMEGA328

1. High Performance, Low Power Atmel AVR 8-Bit Microcontroller Family
2. Advanced RISC Architecture
3. 131 Powerful Instructions
4. Most Single Clock Cycle Execution
5. 32 x 8 General Purpose Working Registers
6. Fully Static Operation – Up to 20 MIPS Throughput at 20MHz
7. On-chip 2-cycle Multiplier
8. High Endurance Non-volatile Memory Segments
9. 32KBytes of In-System Self-Programmable Flash program Memory
10. 1KBytes EEPROM – 2KBytes Internal SRAM – Write/Erase Cycles: 10,000 Flash/100,000 EEPROM – Data Retention: 20 years at 85°C/100 years at 25°C(1)
11. Up to 64 sense channels
12. Peripheral Features
13. Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
14. One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
15. Real Time Counter with Separate Oscillator
16. Six PWM Channels

17. 8-channel 10-bit ADC in TQFP and QFN/MLF package
18. Temperature Measurement
19. 6-channel 10-bit ADC in PDIP Package
20. Temperature Measurement
21. Two Master/Slave SPI Serial Interface
22. One Programmable Serial USART
23. Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
24. I/O and Packages
25. 23 Programmable I/O Lines
26. 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
27. Operating Voltage:
28. 1.8 - 5.5V
29. Temperature Range: – -40°C to 105°C
30. Speed Grade: – 0 - 4MHz @ 1.8 - 5.5V
31. 0 - 10MHz @ 2.7 - 5.5V
32. 0 - 20MHz @ 4.5 - 5.5V
33. Power Consumption at 1MHz, 1.8V, 25°C
34. Active Mode: 0.2mA – Power-down Mode: 0.1µA
35. Power-save Mode: 0.75µA (Including 32kHz RTC)

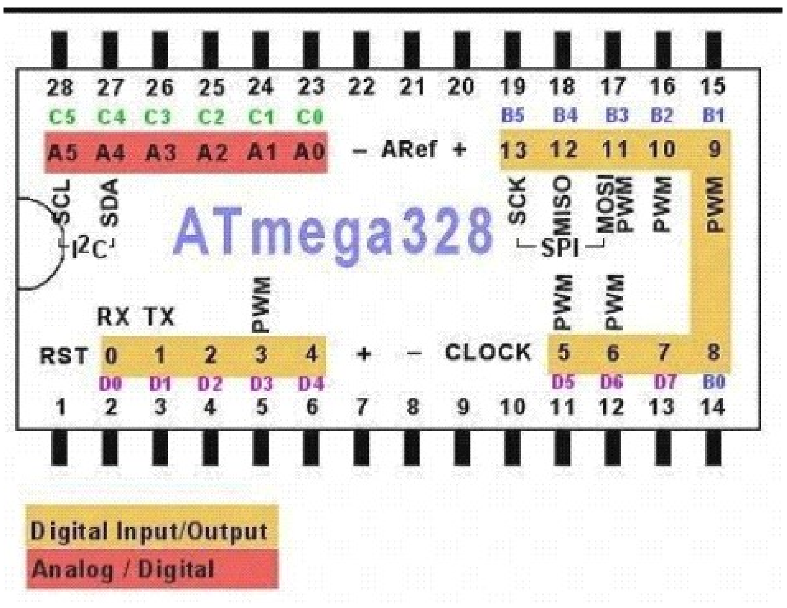
ATmega328
28 27 26 25 24 23 22 21 20 19 18 17 16 15
C5 C4 C3 C2 C1 C0 B5 B4 B3 B2 B1
A5 A4 A3 A2 A1 A0 – ARef + 13 12 11 10 9
SCL SDA I2C
SCK MISO MOSI PWM PWM PWM SPI
RX TX PWM PWM PWM
RST 0 1 2 3 4 + – CLOCK 5 6 7 8
D0 D1 D2 D3 D4 D5 D6 D7 B0
1 2 3 4 5 6 7 8 9 10 11 12 13 14
Digital Input/Output
Analog / Digital

Figure 4.2 Atmega 328

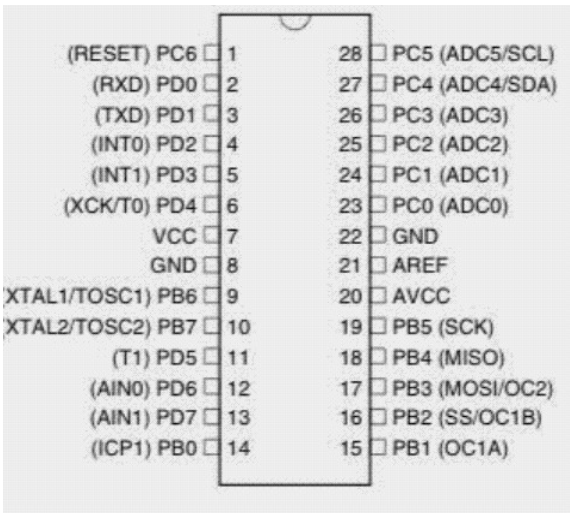

Figure 4.2 pin diagram of ATMEGA328

4.4 PIN DESCRIPTIONS

1. VCC

Digital supply voltage.

2. GND

Ground

3. Port B(PB[7,0]) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bidirectional 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 extremely 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.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output to the inverting oscillator amplifier.

If the internal calibrated RC oscillator is used as chip clock source, PB[7:6] is used as TOSC[2:1] input for the asynchronous timer/counter2 if the AS2 bit in ASSR is set.

4. Port C(PC[5:0])

Port C is a 7 bit bi-directional I/O port with internal pull up resistors (selected for each bit). The PC [5:0] output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, port C pins that are extremely 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.

5. PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of other pins of port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a reset input. a low level on this for longer than the minimum pulse length will generate a reset.

6. Port D(PD[7:0])

Port D is a 7 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 extremely 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.

7. AVCC

AVcc is the supply voltage pin for the A/D converter, PC [3:0], and PE [3:2]. 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. Note that PC [6:4] use digital supply voltage, Vcc.

8. AREF

AREF is the analog reference pin for the A/D converter/

9. ADC[7:6](TQFP and VFQFN Package Only)

in the TQFP and VFQFN package, ADC[7:6] serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

4.5 ARDUINO ARCHITECTURE

The AVR core combines a rich instruction set with 32 general purpose working registers. All these registers are directly connected to ALU (arithmetic logic unit), allowing two independent registers to be accessed in one single instruction executed in one cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access to memories, perform calculations, control peripherals, and handle interrupts.

The AVR uses Harvard architecture- with separate memories and buses for program and data. Instructions in the program memory are executed with single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. The program memory is in-system reprogrammable flash memory.

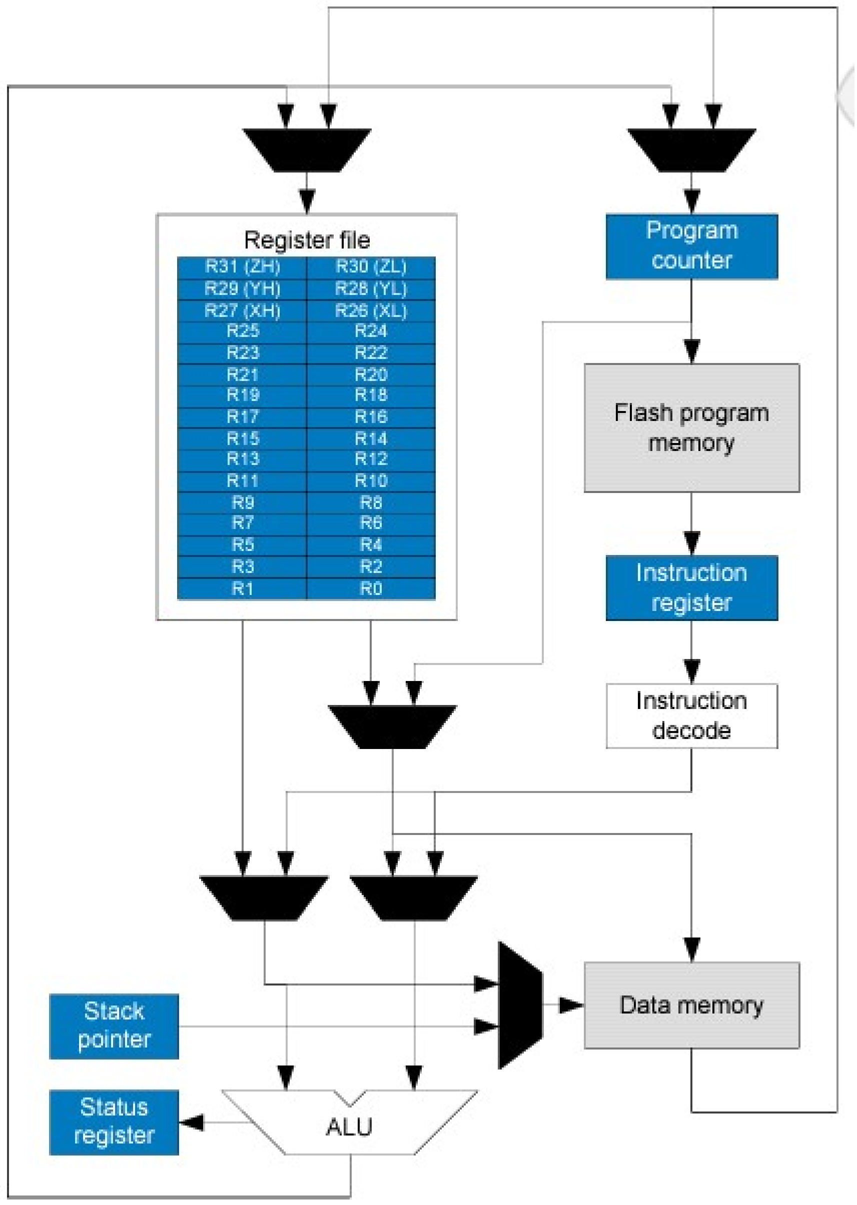

Figure 4.3 atmega328 architecture

The fast access register files contain 32x8 bit general purpose working registers with single clock cycle access time. This allows single cycle ALU operation. In atypical ALU operation, two operands are output from the register file, the operation is executed. Six of the 32 registers can be used as three 16 bit indirect addresses register pointers for data space addressing- enabling efficient address calculations. One of these address pointers are 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- register.

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. In addition, this device has Extended I/O space from 0x60 - 0xFF in SRAM where only the ST/STS/STD and LD/LDS/LDD instructions can be used.

4.5.1 ALU

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.

4.5.2 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. 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.

4.5.3 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:

1. One 8-bit output operand and one 8-bit result input
2. Two 8-bit output operands and one 8-bit result input
3. Two 8-bit output operands and one 16-bit result input
4. One 16-bit output operand and one 16-bit result input

Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions. As shown in the figure, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y-, and Z-pointer registers can be set to index any register in the file.

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 the figure.

In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement.

4.5.4 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 is implemented as growing from higher to lower memory locations. The Stack Pointer Register always points to the top of the Stack.

The Stack Pointer points to the data SRAM Stack area where the Subroutine and Interrupt Stacks are located. A Stack PUSH command will decrease the Stack Pointer. The Stack in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. Initial Stack Pointer value equals the last address of the internal SRAM and the Stack Pointer must be set to point above start of the SRAM.

Stack Pointer Register High byte

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 offset addresses.

Name: SPH

Offset: 0x5E

Reset: RAMEND

Property: When addressing I/O Registers as data space the offset address is 0x3E

Stack Pointer Register Low byte

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 offset addresses.

Name: SPL

Offset: 0x5D

Reset: 0x11111111

Property: When addressing I/O Registers as data space the offset address is 0x3D

4.5.5 AVR MEMORIES

This section describes the different memory types in the device. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the device features an EEPROM Memory for data storage. All memory spaces are linear and regular.

In-System Reprogrammable Flash Program Memory

The ATmega328/P contains 32Kbytes 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 16K x 16. For software security, the Flash Program memory space is divided into two sections - Boot Loader Section and Application Program Section in the device.

The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATmega328/P Program Counter (PC) is 14 bits wide, thus addressing the 16K program memory locations. The operation of Boot Program section and associated Boot Lock bits for software protection are described in detail in Boot Loader Support Read-While-Write Self-Programming. Refer to Memory Programming for the description on Flash data serial downloading using the SPI pins.

Constant tables can be allocated within the entire program memory address space, using the Load Program Memory (LPM) instruction.

SRAM Data Memory

The device is a complex microcontroller with more peripheral units than can be supported within the 64 locations reserved in the Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60 - 0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

The lower 2303 data memory locations address both the Register File, the I/O memory, Extended I/O memory, and the internal data SRAM. The first 32 locations address the Register File, the next 64 location the standard I/O memory, then 160 locations of Extended I/O memory, and the next 2K locations address the internal data SRAM.

EEPROM Data Memory

The ATmega328/P contains 1K 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.

I/O Memory

All device I/Os and peripherals are placed in the I/O space. All I/O locations may be accessed by the LD/LDS/LDD and ST/STS/STD 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.

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. The device is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from 0x60.0xFF in SRAM, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

4.5.6 POWER MANAGEMENT AND SLEEP MODES

Sleep modes enable the application to shut down unused modules in the MCU, thereby saving power. The device provides various sleep modes allowing the user to tailor the power consumption to the application requirements. When enabled, the Brown-out Detector (BOD) actively monitors the power supply voltage during the sleep periods. To further save power, it is possible to disable the BOD in some sleep modes.

Sleep Modes

To enter any of the six sleep modes, the Sleep Enable bit in the Sleep Mode Control Register (SMCR.SE) must be written to '1' and a SLEEP instruction must be executed. Sleep Mode Select bits (SMCR.SM [2:0]) select which sleep mode (Idle, ADC Noise Reduction, Power-down, Power-save, Standby, or Extended Standby) will be activated by the SLEEP instruction.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the Register File and SRAM are unaltered when the device wakes up from sleep. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset Vector.

BOD Disable

When the Brown-out Detector (BOD) is enabled by BODLEVEL fuses (see also section Fuse Bits), the BOD is actively monitoring the power supply voltage during a sleep period. To save power, it is possible to disable the BOD by software for some of the sleep modes. The sleep mode power consumption will then be at the same level as when BOD is globally disabled by fuses. If BOD is disabled in software, the BOD function is turned off immediately after entering the sleep mode. Upon wake-up from sleep, BOD is automatically enabled again. This ensures safe operation in case the VCC level has dropped during the sleep period.

When the BOD has been disabled, the wake-up time from sleep mode will be approximately 60µs to ensure that the BOD is working correctly before the MCU continues executing code.

Idle Mode

When the SM[2:0] bits are written to '000', the SLEEP instruction makes the MCU enter Idle mode, stopping the CPU but allowing the SPI, USART, Analog Comparator, 2-wire Serial Interface, Timer/ Counters, Watchdog, and the interrupt system to continue operating. This sleep mode basically halts clkCPU and clkFLASH, while allowing the other clocks to run.

Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and USART Transmit Complete interrupts. If wake-up from the Analog Comparator interrupt is not required, the Analog Comparator can be powered down by setting the ACD bit in the Analog Comparator Control and Status Register – ACSR. This will reduce power consumption in Idle mode.

ADC Noise Reduction Mode

When the SM[2:0] bits are written to '001', the SLEEP instruction makes the MCU enter ADC Noise Reduction mode, stopping the CPU but allowing the ADC, the external interrupts, the 2-wire Serial Interface address watch, Timer/Counter2(1), and the Watchdog to continue operating (if enabled). This sleep mode basically halts clkI/O, clkCPU, and clkFLASH, while allowing the other clocks to run.

This improves the noise environment for the ADC, enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically when this mode is entered. Apart from the ADC Conversion Complete interrupt, only these events can wake up the MCU from ADC Noise Reduction mode:

1. External Reset
2. Watchdog System Reset
3. Watchdog Interrupt
4. Brown-out Reset
5. 2-wire Serial Interface address match
6. Timer/Counter2 interrupt
7. SPM/EEPROM ready interrupt
8. External level interrupt on INT
9. Pin change interrupt

Power-Down Mode

When the SM[2:0] bits are written to '010', the SLEEP instruction makes the MCU enter Power-Down mode. In this mode, the external Oscillator is stopped, while the external interrupts, the 2-wire Serial Interface address watch, and the Watchdog continue operating (if enabled). Only one of these events can wake up the MCU:

1. External Reset
2. Watchdog System Reset
3. Watchdog Interrupt
4. Brown-out Reset
5. 2-wire Serial Interface address match
6. External level interrupt on INT
7. Pin change interrupt

This sleep mode basically halts all generated clocks, allowing operation of asynchronous modules only.

Power-save Mode

When the SM[2:0] bits are written to 011, the SLEEP instruction makes the MCU enter Power-save mode. This mode is identical to Power-down, with one exception.

If Timer/Counter2 is enabled, it will keep running during sleep. The device can wake up from either Timer Overflow or Output Compare event from Timer/Counter2 if the corresponding Timer/Counter2 interrupt enable bits are set in TIMSK2, and the Global Interrupt Enable bit in SREG is set.

The Timer/Counter2 can be clocked both synchronously and asynchronously in Power-save mode. If Timer/Counter2 is not using the asynchronous clock, the Timer/Counter Oscillator is stopped duringsleep. If Timer/Counter2 is not using the synchronous clock, the clock source is stopped during sleep. Even if the synchronous clock is running in Power-save, this clock is only available for Timer/Counter2.

Standby Mode

When the SM[2:0] bits are written to '110' and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Standby mode.

This mode is identical to Power-Down with the exception that the Oscillator is kept running. From Standby mode, the device wakes up in six clock cycles.

Extended Standby Mode

When the SM[2:0] bits are written to '111' and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Extended Standby mode. This mode is identical to Power-Save mode with the exception that the Oscillator is kept running. From Extended Standby mode, the device wakes up in six clock cycles.

Power Reduction Register

The Power Reduction Register (PRR) provides a method to stop the clock to individual peripherals to reduce power consumption. The current state of the peripheral is frozen and the I/O registers can not be read or written. Resources used by the peripheral when stopping the clock will remain occupied, hence the peripheral should in most cases be disabled before stopping the clock. Waking up a module, which is done by clearing the corresponding bit in the PRR, puts the module in the same state as before shutdown. Module shutdown can be used in Idle mode and Active mode to significantly reduce the overall power consumption. In all other sleep modes, the clock is already stopped.

4.6 POWER SUPPLY

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm centre-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

1. Vin. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
2. 5V. the regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
3. 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
4. GND. Ground pins.

4.7 MATRIX MEMBRANE KEYPAD

This 16-button keypad provides a useful human interface component for microcontroller projects. Convenient adhesive backing provides a simple way to mount the keypad in a variety of applications.

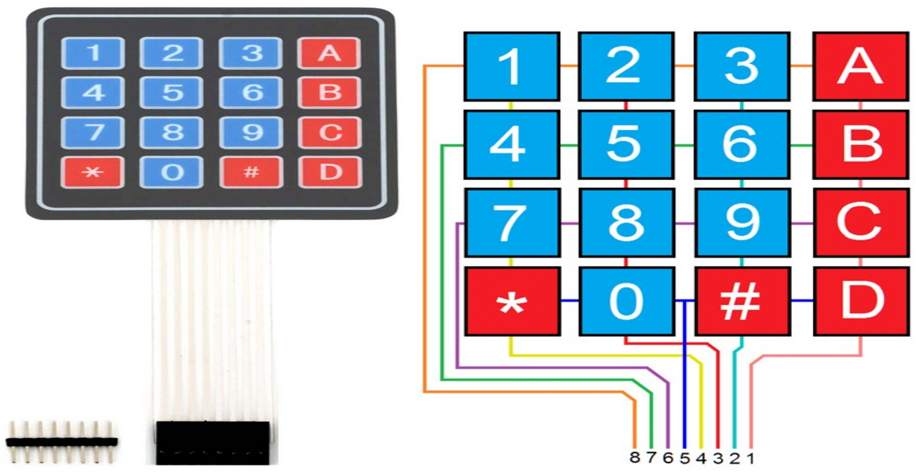

Figure 4.4 keypad

Features

1. Ultra-thin design.
2. Adhesive backing.
3. 4*4 matrix circuitry to reduce connecting pins.
4. Sixteen push-on DPST Tactile switches.
5. Life expectancy for standard force models (1.67 N- 160 gF) is 500000 operations minimum and that for high force models (2.26 N- 230 gF) is 300000 operations

minimum.

6. Excellent price/performance ratio.
7. Easy interface to any microcontroller.

Key Specifications

1. Maximum Rating: 24 VDC, 30 mA.
2. Interface: 8-pin access to 4x4 matrix.
3. Operating temperature: 32 to 122 °F (0 to 50°C).
4. Dimensions: Keypad, 2.7 x 3.0 in (6.9 x 7.6 cm) Cable: 0.78 x 3.5 in (2.0 x 8.8 cm)

Applications:

1. Security systems.
2. Menu selection.
3. Data entry for embedded systems.

How it Works

Matrix keypads use a combination of four rows and four columns to provide button states to the host device, typically a microcontroller. Underneath each key is a pushbutton, with one end connected to one row, and the other end connected to one column. These connections are shown in Figure 1.

In order for the microcontroller to determine which button is pressed, it first needs to pull each of the four columns (pins 1-4) either low or high one at a time, and then poll the states of the four rows (pins 5-8). Depending on the states of the columns, the microcontroller can tell which button is pressed.

For example, say your program pulls all four columns low and then pulls the first row high. It then reads the input states of each column, and reads pin 1 high. This means that a contact has been made between column 4 and row 1, so button 'A' has been pressed.

4.8 LCD

It is display that uses the light modulating properties of liquid crystals. A liquid crystal display is a flat panel display or other electronically modulated optical device that uses the light modulating properties of liquid crystals. Liquid crystals do not emit light directly, instead using a backlight or reflector to produce images in colour or monochrome. LCDs are available to display arbitrary images or fixed images with low information content. Arbitrary images are made up of a large number of small pixels, while other displays have larger elements.

Since LCD panels produce no light of their own, they require external light to produce a visible image. in a "transmissive " type of LCD, this light is provided at the back of the glass “stack" and is called the backlight. The common implementations of LCD backlight technology are: CCFL,EL-WLED, WLED array, RGB- LED

LCD modules can be split into two groups: those that have built-in controller and driver chips, and those that have only driver chips. LCD displays that do not have controllers are typically used with powerful hardware, such as a laptop computer, where a video controller is available to generate the complex drive signals necessary to run the display. Most color and large (greater than 320x240) monochrome displays are of this type.

The category of display modules that have built-in controllers can be split again into character LCD modules and graphic LCD modules. Character modules can display only text and perhaps some special symbols, while graphic modules can display lines, circles, squares, and patterns in addition to text. Some examples of graphic LCD controller chips are the Toshiba T6963, Seiko-Epson SED1330, and Hitachi HD61202. Here, we will be primarily concerned with character LCD modules that have the Hitachi HD44780 controller built-in.

Nearly every pixel-based alphanumeric LCD module made today uses the Hitachi HD44780 LCD controller chip. This apparent standardization in character LCDs has become extremely beneficial to design engineers and hobbyists. The smallest of these displays is only one line of 8 characters; the largest is four lines of 40 characters each. Other common sizes are 16x1, 20x1, 20x2, 20x4, 40x1, and 40x2 (characters x lines).

Fortunately, all HD44780-based displays (of any size) use the same standard 14-wire interface. Therefore, code and hardware made for one size/type display can be painlessly adapted to work for any HD44780 compatible.

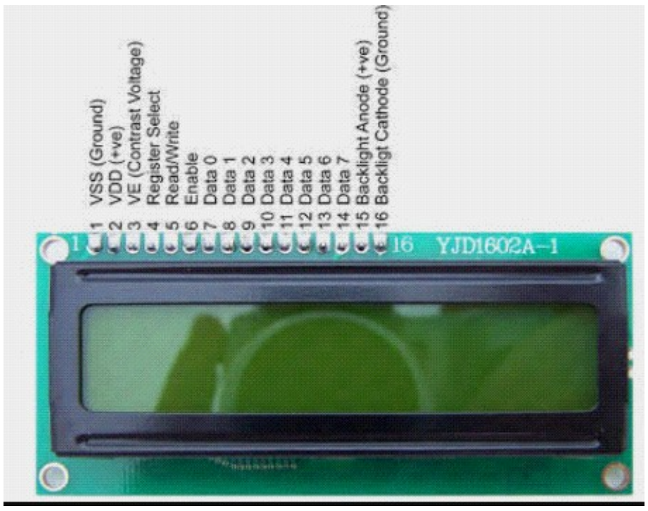

Figure 4.5 lcd pin diagram

Interfacing your LCD module

The microcontroller/microprocessor interface to HD44780 LCD modules (hereafter generically referred to as character LCD modules) is almost always 14 pins. Table 1 shows the basic pin out. You may find that some displays have additional pins for backlighting or other purposes, but the first 14 pins still serve as the interface.

The first three pins provide power to the LCD module. Pin 1 is GND and should be grounded to the power supply. Pin 2 is VCC and should be connected to +5V power. Pin 3 is the LCD Display Bias. By adjusting the voltage or duty cycle of pin 3, the contrast of the display can be adjusted. Note that greater contrast comes with lower voltage and you should never apply a VLCD higher than VCC.

Table 4.1 Character LCD Pin out

ITEM	SYMBOL	LEVEL	FUNCTIONS
1	LEDA	+5V	Power supply For LED Backlight
2	LEDK	0V	Power supply For LED Backlight
3	V_{SS}	0V	Power Ground
4	V_{DD}	+5V	Power Supply For Logic
5	V_0	-	Contrast adjust
6	RS	H/L	H:data L:command
7	R/W	H/L	H:read L:write
8	E	H. H->L	Enable signal
9-16	DB0-DB7	H/L	Power Ground

Pins 4, 5, and 6 are the control lines for the LCD. These lines indicate what kind of transaction you are proposing to do over the data lines DB0-7. The state of RS indicates whether you wish to transfer commands or display data. The R/W line indicates whether you intend to read or write. Finally, the E line tells the display when you are actually ready to perform the transaction. The control lines RS, R/W, and E, along with the data lines DB0-7 are standard digital logic inputs or outputs.

Accessing your LCD module

Character LCD modules are accessed through only two “registers”, the Command Register, and the Data Register. When you perform a read or write with RS low, you are accessing the Command Register and giving the module instructions. When you read or write with RS high, you are accessing the Data Register and reading or writing characters/data from or to the display. Table 2 contains pseudo-code examples of reads and writes to the two registers. The pseudo-code can be implemented on any microcontroller and assumes that DBPORT is the port to which DB0-7 are connected.

Reading from a register writing to a register

1. set RS line high or low to designate the register you wish to access
2. set R/W line high to indicate a read
3. set DBPORT to input
4. set E line high to begin read cycle
5. pause to allow LCD to fetch the data
6. read data from DBPORT
7. set E line low to finish read cycle
8. set RS line high or low to designate the register you wish to access
9. set R/W line low to indicate a write
10. set DBPORT to output
11. write data to DBPORT
12. set E line high to begin write cycle
13. pause to allow LCD to accept the data
14. set E line low to finish write cycle

Initializing and programming your LCD Character LCDs must be initialized after power-on and before writing data to the display. The initialization must consist of at least a Function Set command, preferably followed by an Entry Mode Set, Display Control, and a Clear Display. Issuing each of these commands after the Function Set ensures that you know the state of your display.

LCD Applications

1. Computer monitors, televisions
2. Instrument panels, aircraft cockpit displays
3. Small LCD screens are common in portable consumer devices such as digital cameras, watches, calculators, and mobile telephones including smart phones
4. Consumer electronics products such as DVD players, video game devices and clocks

4.9 BUZZER

The buzzer used here to as indicator is piezoelectric buzzer. Buzzer is an acoustic signal generator. A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanically, and piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

The electric buzzer was invented in 1831 by Joseph Henry. They were mainly used in early doorbells until they were phased out in the early 1930s in favour of musical chimes, which had a softer tone. Piezoelectric buzzers, piezobuzzers, as they are sometimes called, were invented by Japanese manufacturers and fitted into a wide array of products during the 1970s to 1980s. This advancement mainly came about because of cooperative efforts by Japanese manufacturing companies. In 1951, they established the Barium Titanate Application Research Committee, which allowed the companies to be "completely cooperative" and bring about several piezo electric innovations and inventions.

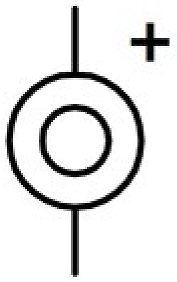

Figure 4.6 buzzer

Applications of buzzers

While technological advancements have caused buzzers to be impractical and undesirable, there are still instances in which buzzers and similar circuits may be used. Present day applications include:

1. Novelty uses.
2. Judging panels.
3. Educational purposes.
4. Annunciate panels.
5. Electronic metronomes.
6. Game show lock-out device.
7. Microwave ovens.
8. Household appliances.
9. Sporting events such as basketball games.
10. Electrical alarms.
11. Joy buzzer(mechanical buzzer used for pranks)

4.10 BC547 TRANSISTOR

A transistor is a semiconductor device used to amplify and switch electronic signals and electrical power. Transistors work wonderfully for computer production. These help computers power through huge numbers of calculations in a short time. The simple switch operation of transistors is what enables our computer to complete massively. One computer chip can have millions of transistors continually switching, helping complete complex calculations.

BC547 is an NPN Bi-polar junction transistor. A transistor, stands for transfer of resistance, is commonly used to amplify current. A small current at its base controls a larger current at collector and emitter terminals.

It is used as the active component for switches and amplifiers. It has an emitter terminal, a base or control terminal, and a collector terminal. In atypical configuration, the current flowing from the base to the emitter controls the collector current. A short vertical line, which is the base can indicate the transistor schematic for an NPN transistor, and the emitter, which is a diagonal line connecting the base, is an arrowhead pointing away from the base.

It has a maximum current gain of 800. The transistor terminals require a fixed DC voltage to operate in the desired region of its characteristic curves. This is known as the biasing. This transistor is used in common emitter configuration for amplifiers. The voltage divider is the commonly used biasing mode. For switching applications, it remains fully on if there is a signal at its base. in the absence of the base signal, it gets completely off.

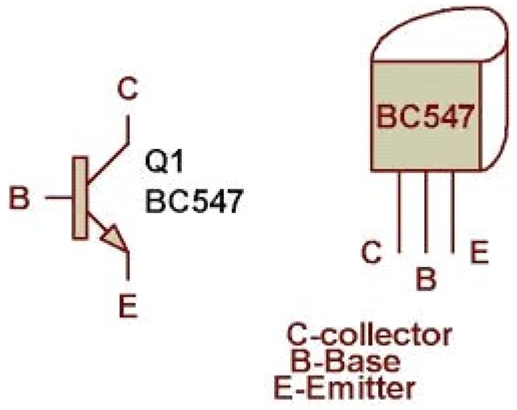

Figure 4.7 NPN BC547 transistor

NPN transistor working principle

The voltage between the base and emitter (VBE), is positive at the base and negative at the emitter because for an NPN transistor, the base terminal is always positive with respect to the emitter. Also the collector supply voltage is positive with respect to the emitter (VCE). So for a bipolar NPN transistor to conduct the collector is always more positive with respect both the base and the emitter.

In forward biased condition, the collector is connected to high positive voltage with respect to base i.e. VCB is very high. so C-B junction is reverse biased. VCB>>VBE. The base is connected to low positive voltage with respect to the emitter i.e. VBE is low. When we increase VBE>=0.7V the transistor is forward biased. Now large number of electrons in emitter layer is repelled by negative terminal of VBE and they flow towards B-E junction. They cross the junction and enter into small base layer. Here some electrons combine with holes. Also some of them are attracted by positive terminal of VBE and remaining maximum number of electrons flow into collector layer, crossing the second junction i.e. C-B junction.

In reverse biased condition, both the junctions are reverse biased as the batteries are connected in opposite direction, Due to VCB battery, the base emitter junction is also reverse biased. So charges cannot flow and current in the transistor is practically zero. This method is not useful in "cut-off" state since current is zero.

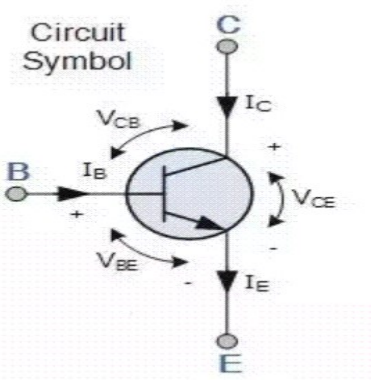

Figure 4.8 npn transistor symbol

4.11 BREADBOARD

A breadboard is a construction base for prototyping of electronics. It is solder less breadboard and does not require soldering, it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. a strip board and similar prototyping printed circuit boards, which are used to build semi permanent soldered prototypes, cannot easily be reused.

The breadboard most commonly used today is usually made of white plastic and is a pluggable(solder less) breadboard. It was designed by Ronald J.Portugal of El instruments Inc. in 1971. Breadboards are one of the most fundamental pieces when learning how to build circuits. A breadboard also known as protoboard is a type of solder less electronic circuit building.

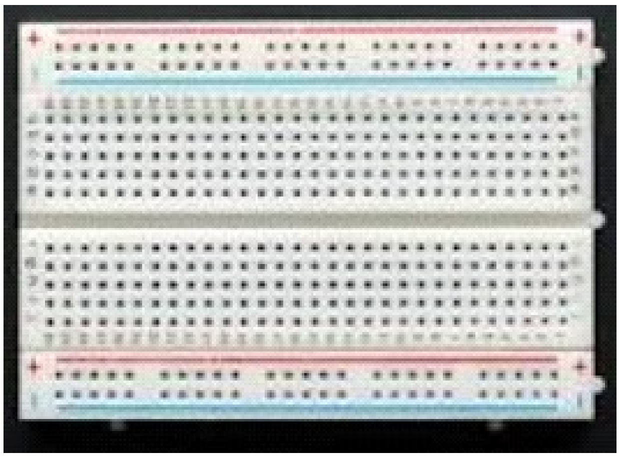

Figure 4.9 breadboard

Typical specifications of breadboard

A modern solder less breadboard consists of a perforated block of plastic with numerous tin pasted phosphor bronze or nickel silver alloy spring clips under the perforations. The clips are often called tie points or contact points. The number of tie points is often given in the specification of the breadboard.

The spacing between the clips is typically 0.1 in (2.54 mm). Integrated circuits in dual in-line packages can be inserted to straddle the centreline of the block. Interconnecting wires and the leads of discrete components (such as capacitors, resistors, and inductors) can be inserted into the remaining free holes to complete the circuit. Typically the spring clips are rated for 1 ampere at 5 volts and 0.333 amperes at 15 volts (5 watts). The layout of a typical solder less breadboard is made up from two types of areas, called strips. Strips consist of interconnected electrical terminals.

Terminal strips

The main areas, to hold most of the electronic components. in the middle of a terminal strip of a breadboard, one typically finds a notch running in parallel to the long side. The notch is to mark the centreline of the terminal strip and provides limited airflow to DIP ICs straddling the centreline.

The five rows on the left of the notch are often marked as A, B, C, D, and E, while the ones on the right are marked F, G, H, I, J.

Bus strips

To provide power to the electronic components. A bus strip usually contains two rows: one for ground and one for supply voltage. Bus strips typically run down one or both sides of a terminal strip or between terminal strips. On large breadboards additional bus strips can often be found on the top bottom of terminal strips.

A "full size" terminal breadboard strip typically consists of around 56 to 65 rows of connectors, each row containing the above mentioned two sets of connected clips.

5. SOFTWARE DESCRIPTION

5.1 ARDUINO SOFTWARE

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required.

The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to- serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A Software Serial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega328 datasheet.

5.1.1 PROGRAMMING

The Arduino Uno can be programmed with the Arduino software. Select "Arduino Uno w/ ATmega328" from the Tools > Board menu (according to the microcontroller on your board). For details, see the reference and tutorials. The ATmega328 on the Arduino Uno comes preburned with a boot loader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the boot loader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header. The ATmega8U2 firmware source code is available. The ATmega8U2 is loaded with a DFU boot loader, which can be activated by connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2. You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU boot loader).

5.1.2 HOW TO USE ARDUINO

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).
Arduino is a cross-platform program. You can plug the Arduino to your PC via USB cable.

Now you're actually ready to "burn" your first program on the arduino board. To select "blink led", the physical translation of the well known programming "hello world", select File>Sketchbook> Arduino-0017>Examples> Digital>Blink Once you have your sketch you'll see something very close to the screenshot on the right. In Tools>Board select Now you have to go to Tools>Serial Port and select the right serial port, the one arduino is attached to.

5.2 CODE

```
#include<keypad.h>
#include<LiquidCrystal.h>
#include<EEPROM.h>
#define buzzer 15
LiquidCrystal lcd(13,12,11,10,9,8);
char password[4];
char pass[4], pass1[4];
int i=0;
char customKey=0;
const byte ROWS=4;//four rows
const byte COLS=4;//four columns
char hexaKeys[ROWS][COLS]={
{'1','2','3','A'},
{'4','5','6','B'},
{'7','8','9','C'},
{'*','0','#','D'};
};
byte rowPins[ROWS]={3,2,1,0};// connect to the row pinouts of the keypad
byte colPins[COLS]= {4,5,6,7};// connect to the column pinouts of the keypad
//initialise an instance of class newkeypad
keypad customKeypad= Keypad(makeKeymap(hexaKeys), rowPins, colPins,
ROWS,COLS);
void setup()
{
lcd.begin(16,2);
pinMode(led, OUTPUT);
pinMode(buzzer, OUTPUT);
pinMode(m11, OUTPUT);
pinMode(m12, OUTPUT);
lcd.print("  Electronic ");
lcd.setCursor(0,1);
lcd.print("    Keypad lock  ");
delay(2000);
lcd.clear();
lcd.print("  Enter your passkey ");
lcd.setCursor(0,1);
for(int j=0;j<4;j++)
EEPROM.write(j, j+49);
for(int j=0;j<4;j++)
pass[j]=EEPROM.read(j);
}
void loop()
{
customKey= customKeypad.getKey();
if(customKey=='#')
change();
if(customKey)
```

```
{
password[i++]=customKey;
lcd.print(customKey);
beep();
}
if(i==4)
{
delay(200);
for(int j=0;j<4;j++)
pass[j]=EEPROM.read(j);
if(!(strncomp(password, pass, 4)))
{
digitalWrite(led, HIGH);
beep();
lcd.clear();
lcd.print("Passkey Accepted");
delay(2000);
lcd.setCursor(0,1);
lcd.print("#.change passkey");
delay(2000);
lcd.clear();
lcd.print("enter passkey:");
lcd.setCursor(0,1);
i=0;
digitalWrite(led,LOW);
}
else
{
digitalWrite(buzzer, HIGH);
lcd.clear();
lcd.print("Access Denied...");
lcd.setCursor(0,1);
lcd.print("#.change passkey");
delay(2000);
lcd.clear();
lcd.print("Enter passkey");
lcd.setCursor(0,1);
i=0;
digitalWrite(buzzer, LOW);
}
}
}
void change()
{
int j=0;
lcd.clear();
lcd.print("your current passkey");
lcd.setCursor(0,1);
while(j<4)
{
char key= customkeypad.getkey();
if(key)
{
pass1[j++]=key;
lcd.print(key);
beep();
}
key=0;
}
delay(500);
if((strncmp(pass1,pass,4)))
{
lcd.clear();
lcd.print("wrong passkey....");
lcd.setCursor(0,1);
lcd.print("better luck again");
delay(1000);
}
```

```
else
{
j=0;
lcd.clear();
lcd.print("enter new passkey");
lcd.setCursor(0,1);
while(j<4)
{
char key=customKeypad.getKey();
if(key)
{
pass[j]=key;
lcd.print(key);
EEPROM.write(j,key);
j++;
beep();
}
}
lcd.print("done..");
delay(1000);
}
lcd.clear();
lcd.print("enter your passkey");
lcd.setCursor(0,1);
customKey=0;
}
void beep()
{
digitalWrite(buzzer, HIGH);
delay(20);
digitalWrite(buzzer, LOW);
}
```

6. WORKING

in this circuit we have used multiplexing technique to interface keypad for input the password in the system. here we are using 4*4 keypad which contains 16 key. if we want to use 16 keys then we need 16 pin for connection to arduino but in multiplexing technique we need to use only 8 pins for interfacing 16 keys. so that it is a smart way to interface a keypad module.

Multiplexing technique: multiplexing technique is a very efficient way to reduce number of pins used with the microcontroller for providing input or password or numbers. Basically this technique is used in two ways- one is row scanning and the other is column scanning. But in this project we have used keypad library so we do not need to make any multiplexing code for this system. We only need to use keypad library for providing input.

Circuit description: circuit of this project is very simple which contains arduino, keypad module, buzzer and lcd. Arduino controls the complete processors like taking password from keypad module, comparing passwords, driving buzzer and sending status to lcd display. Keypad is used for taking password. Buzzer is used for indications and lcd is used for displaying status or messages on it. Buzzer is driven by using a npn transistor.

Keypad module's column pins are directly connected to pin 4, 5, 6, 7 and row pins are connected to 3,2,1,0 of arduino uno. a 16*2 lcd is connected with arduino in 4-bit mode. Control pin RS,RW and EN are directly connected to arduino pin 13, GND and12. And data pins D4-D7 is connected to pins 11,10,9 and 8 of arduino. And one buzzer is connected at pin 14(A1) of arduino through a BC547 NPN transistor.

Working:

We have used inbuilt arduino's EEPROM to save password, so when we run this circuit first time program read a garbage data from inbuilt arduino's EEPROM and compare it with the input password and give a message on lcd that is access denied because password does not match. For solving this problem we need to set a default password for the first time by programming which will set password "1234" to EEPROM of arduino.

After running it first time we need to remove this from program and again write the code in to the arduino and run. Now your system will run fine. And for your second time used password is now "1234".now you can change it by pressing # button and then enter your current password and then enter your new password.

When you will enter your password, system will compare your entered password with that password that is stored in EEPROM of arduino. if match is occurred then lcd will show "access granted" and if password is wrong then lcd show "access denied" and buzzer continuously beep for some time. And buzzer will also beep a single time whenever user will press any button from keypad.

7. RESULT

The project is being tested in proteus and is Working properly. It is also tested in bread board and is properly working. It has lots of application like we can use this project in door for home and office security, lockers, ATM, and anywhere where security is needed. Here we can easily change the password and it has features like if wrong password is entered more than three times it would be locked until and unless reset button is pressed. In this project we have used very less component so it is cost effective and it is less complicated than a simple micro controller based code lock system.

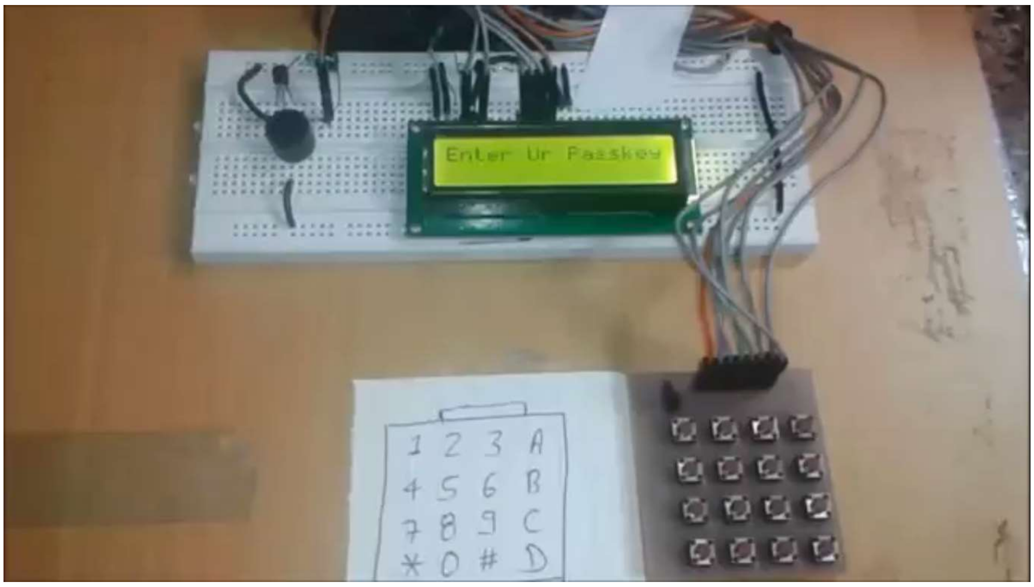

Figure 7.1 Enter passkey

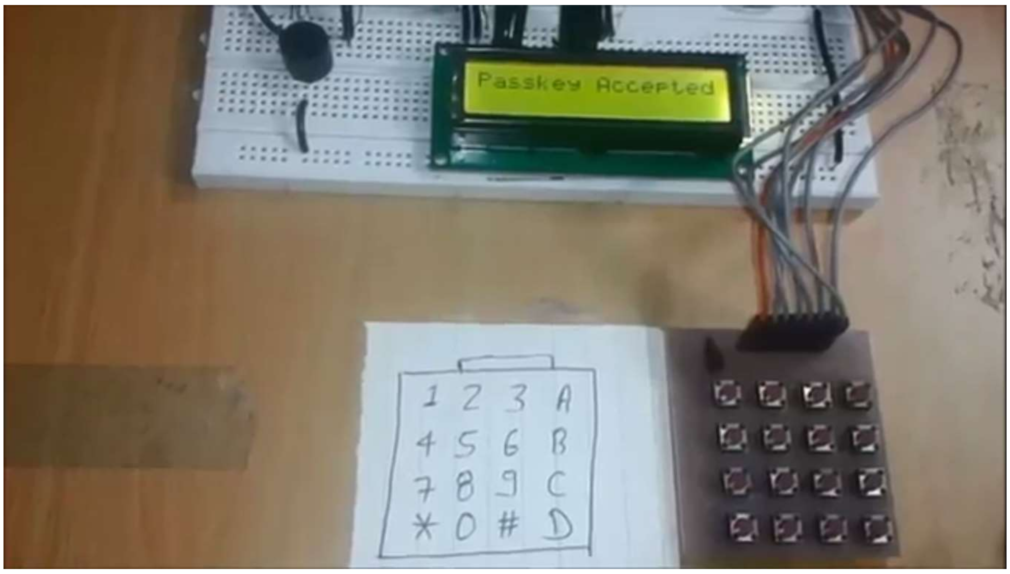

Figure 7.2 Passkey accepted

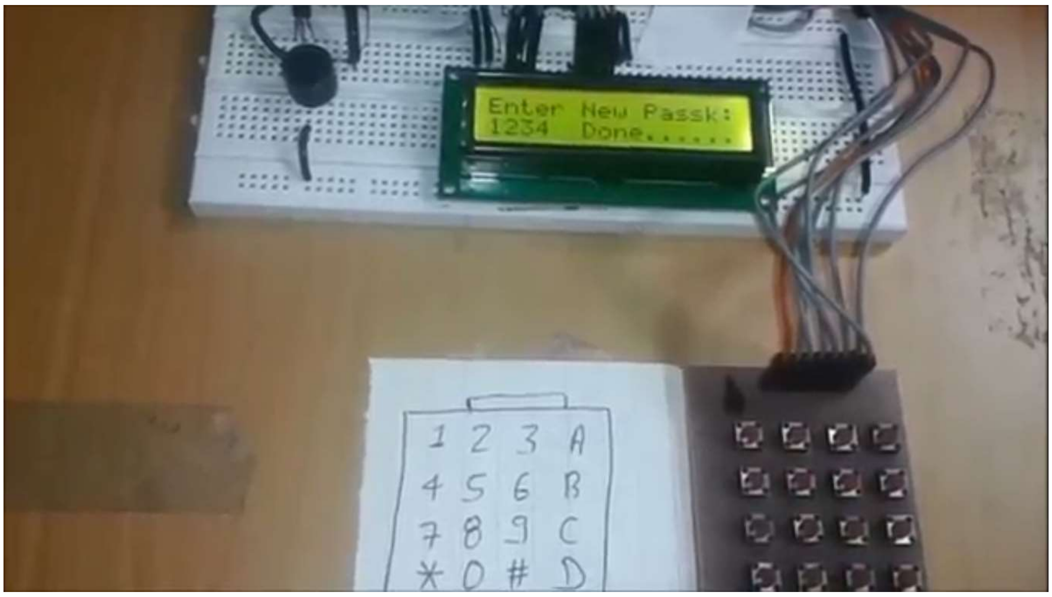

Figure 7.3 Enter new passkey

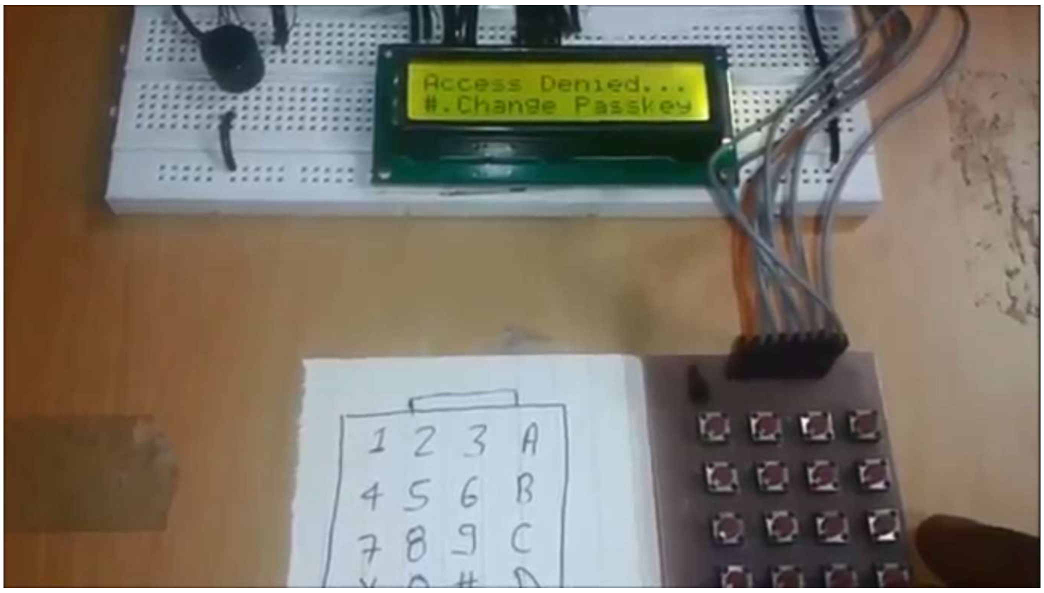

Figure 7.4 Access denied

8. MERITS AND DEMERITS

MERITS

1. **Pick-proof:** Because there is no place for a key with these locks, the prevent break-ins because burglars are unable to pick or bump the luck. criminals methods of breaking and entering are improving and the majority of criminals can pick an ordinary key lock.
2. **No more keys:** You won't have to carry around a large set of keys and they will be less likely to be lost or stolen.
3. **Control:** In a company building, you can control and restrict who goes into what part of building. it is incredibly easy to change the pin code whenever you like.
4. **Aesthetically pleasing:** Door locks can come in a range of stylish colours that look smart and professional.
5. **Perfect for the elderly and disabled:** It is advantageous to those who are unable to get to the door quickly or struggle with keys.
6. **Less pin code length:** It uses minimum code length up to four numbers and you can increase it by adding extra features.
7. **Less cost and easy to install:** Its cost is less and vary switch the different products and the installation process is easy.
8. **Reduced wear and tear:** Physical mechanisms are subject to a lot more wear and tear than electronic ones. Here there is no mechanism or keys to become worn-out, bent or broken.
9. Less administration, better security, increased revenue, more visually impressive, easier flexibility and monitoring.

DEMERITS

1. Don't be forgetful
2. Keep the pin code safe and the lock clean
3. Power failure
4. Limit the pin code length which is easy to remember and tough to guess and it is not clumsy.

9. APPLICATIONS

1. Used in bank lockers
2. Used in mobile phones for security purposes
3. In door lock systems
4. In lift, telephone locking, refrigerators etc
5. Used in jewellery lockers and almaraih lockers for security
6. Used in colleges and labs for locking the computer and electricity with the help of relay interfacing
7. Used in wheel car for locking bike and other automobiles
8. Used in care and nursing homes

Figure 9.1 different applications

10. CONCLUSION

Digital code lock is an electronic lock and is best suitable for more security and at low cost. It is simple to design and is more economical to all. It can be installed easily and anyone can understand how to operate this locks. The pin or password you are using for security is easy to remember.

It consumes less power and its implementation is easy. It cannot be damaged easily as like wear locks used before for door locks. By this we can provide security to our home and there is no chance left for theft. We can change password or pin with ease by following instructions.

It alerts the owner or the user when somebody misuses it or enters an incorrect password or pin. The whole system is controlled by the microcontroller which is a powerful device, and has different applications.

11. FUTURE SCOPE

In future, the improved version of microcontrollers will be more useful. In serial communication and robotics, security systems everywhere we need microcontrollers to store program in it. By interfacing and programming with embedded c language and other software languages, many devices can be connected and their functions can be utilised for other purposes.

All electronic devices play a very important role in present day applications and many electronic components like integrated circuits used in various industries, military equipment, medical equipment, space communications etc in various sizes and as a small or integrated part of the system, or as a main system to control the whole unit. By using different methods we can improve its present version to advanced technologies. Microcontrollers have many features for each pin and can be used for different applications. It is compatible and small in size, easy to understand its programming and execution. By using different simulation techniques such as proteus, masm, etc we can check the results of the designed circuits and can be verified if anything is wrong or imperfect.

Its specifications like power dissipation, input and output voltages, circuit connections, thermal dissipations, resolution, etc can be varied with the help of different devices, materials, using them at controlled temperatures and voltages.

REFERENCES

1. Microcontroller: architechture, progrmming and applications by Ayala Kenneth.J 2nd edition Penram.
2. microcontroller by Kenneth Ayala,publishers: Delmar cengage learning
3. http://www.societyofrobots.com/microcontroller_tutorial.shtml
4. piezosystems: history of piezoelectricity, www.piezo.com
5. buzzer: definition of buzzer by the free dictionary
6. Williams,R. (1963). "Domains in liquid crystals". J. Phys. Chem.
7. "Milestones: Liquid Crystal Display, 1968". IEEE global history network
8. https://www.wikipidea.com/matrix_keypad
9. https://learn.sparkfun.com/breadboard-tutorials
10. http://www.atmel.com/atmega328