TOUCHSCREEN AND GLCD BASED DIGITAL DEVICE CONTROL SYSTEM

A Report submitted to
MSRIT
Bangalore
for partial requirement of award of degree of
Bachelor of Engineering in Electronics and Communication Engineering

by

SUMAN SHANKAR SHETTY (1MS08EC115)

VEOLA MAVIS NAZARETH (1MS08EC124)

MEGHA KAIWAR (1MS08EC139)

PAWAN SWAROOP (1MS07EC066)

under the guidance of
MRS. FLORY FRANCIS
Dept of E & C

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

M S RAMAIAH INSTITUTE OF TECHNOLOGY
(Autonomous Institute, Affiliated to VTU)

May 2011

Department of Electronics and Communication Engineering
M S Ramaiah Institute of Technology
Bangalore - 54

CERTIFICATE

This is to certify that the following students who were working under our guidance, have completed their work as per our satisfaction with the topic **TOUCHSCREEN AND GLCD BASED DIGITAL DEVICE CONTROL SYSTEM**. To the best of our understanding the work to be submitted in this report does not contain any work, which has been previously carried out by others and submitted by the candidates for themselves for the award of any degree anywhere.

(SUMAN SHANKAR SHETTY - 1MS08EC115) (MEGHA KAIWAR - 1MS08EC139)
(VEOLA MAVIS NAZARETH - 1MS08EC124) (PAWAN SWAROOP - 1MS07EC066)

MRS. FLORY FRANCIS	DR. S. SETHU SELVI
Asst.Professor	Professor and HOD
Department of E&C	Department of E&C

Department of Electronics and Communication Engineering
M S Ramaiah Institute of Technology
Bangalore - 54

DECLARATION

We hereby declare that the entire work embodied in this report has been carried out by us at M S Ramaiah Institute of Technology under the supervision of MRS. FLORY FRANCIS. This report has not been submitted in part or full for the award of any diploma or degree of this or any other University.

(SUMAN SHANKAR SHETTY - 1MS08EC115) (MEGHA KAIWAR - 1MS08EC139)
(VEOLA MAVIS NAZARETH - 1MS08EC124) (PAWAN SWAROOP - 1MS07EC066)

Abstract

A home automation system is one in which a controller in the user's home is coupled with a remotely located central control facility using a continuous connectivity access line providing a data channel. The controller programs an in-house device so that the user controls the operation of appliances connected to the system, like turning on a light at a specific time or speed control of fan including all of the HVAC (Heating, Ventilating and Air Conditioning) applications. The controller can also be designed to monitor sensors located throughout the house and can sound an alarm or phone, acting as a monitoring agency when the sensors are triggered. Some advanced innovations provide remote-access among other features allowing the user to access the in-house device through mobile phones.

Conventional home automation systems have a couple of disadvantages like a much restricting and simple user interface. Some systems are highly advanced incorporating, or providing a connection to, a device comparable to a personal computer but their high cost limits their wide-spread use. Another disadvantage is that changing built-in control programs typically requires a change to the in-house equipment itself. Present day techniques are provided with features to overcome a few of these drawbacks.

This project mainly aims in designing a simple and completely automated switch board with the help of touch screen sensor and graphical LCD to control the house hold appliances, providing a user friendly environment for the user to operate the devices effectively. It features a reliable system for illiterates and old people who find difficulty in operating few high end devices like AC, water heaters etc. On the other hand, the project is also designed in a way so as to illustrate the clear-cut simplicity and importance of automation in the present scenario. In the further sections, the system design of the developed module along with the necessary hardware and software details are explained.

Contents

1 INTRODUCTION **1**

1.1 Block Diagram . . . 3

1.2 Working in Brief . . . 3

1.3 Organization of the Report . . . 3

2 LITERATURE SURVEY AND MARKET STUDY **5**

2.1 Touch Screen . . . 5

2.1.1 Infrared Touch Screens . . . 5

2.1.2 Resistive Touch Screens . . . 6

2.1.3 Capacitive Touch Screens . . . 6

2.2 Liquid Crystal Display . . . 7

2.2.1 Passive Displays . . . 8

2.2.2 Active Matrix Displays (TFTs) . . . 9

2.2.3 LCD Screen Colors . . . 10

2.3 Microcontrollers . . . 10

2.4 AVR Programmer . . . 11

2.5 Embedded 'C' Programming Software . . . 12

2.6 Simulation Software . . . 12

2.7 L293 Motor Driver . . . 12

3 HARDWARE AND SOFTWARE SELECTION **13**

3.1 Touch Screen . . . 13

3.2 Touch Screen Controller . . . 15

3.3 Graphical LCD . . . 16

3.4 Microcontrollers . . . 17

3.5 Embedded Programming Software . . . 17

3.6 Simulation software . . . 18

3.7 Software to download firmware in the MCU . . . 19

4 HARDWARE IMPLEMENTATION 21
- 4.1 Touch Screen 21
- 4.2 Graphical LCD - JHD12864E 24
  - 4.2.1 Pin Assignments 25
  - 4.2.2 Setting up the LCD and turning it ON 25
  - 4.2.3 Data transfer 26
  - 4.2.4 Control bits 27
  - 4.2.5 How the Input Bits translate into what is on the Screen 28
  - 4.2.6 Functions implemented 30
- 4.3 Microcontroller 34
  - 4.3.1 Overview 34
  - 4.3.2 Features 34
  - 4.3.3 Interface 36
    - 4.3.3.1 I/O Ports 36
    - 4.3.3.2 Analog to Digital Converter (ADC) 37
    - 4.3.3.3 UART 39
- 4.4 Motor Driver 40
  - 4.4.1 Features 40
  - 4.4.2 Pin Description 41

5 INTEGRATED ASSEMBLY 42
- 5.1 Circuit and Software Description 42
  - 5.1.1 Schematic representing control of GLCD by Atmega32 microcontroller 42
  - 5.1.2 Schematic of Atmega8 controller 43
  - 5.1.3 L293D DC motor driver 44

6 CONCLUSION AND FUTURE SCOPE 48
- 6.1 Conclusion 48
- 6.2 Future Scope 49

A 50
- A.1 ATmega32 50
  - A.1.1 Pin Configuration 50
  - A.1.2 Pin Description and Overview 51
  - A.1.3 Electrical Characteristics 52
  - A.1.4 Typical Characteristics 54
  - A.1.5 Block Diagram 57
- A.2 **ATmega8** 58
  - A.2.1 Pin Description and Overview 58
  - A.2.2 Electrical Characteristics and Typical Characteristics 60

B **67**

B.1 JHD12864E 67

B.1.1 Features 67

B.1.2 Mechanical Data 67

B.1.3 Maximum Ratings 68

B.1.4 Electrical Characteristics 68

B.1.5 Electro-Optical Characteristics 69

B.1.6 Reliability Characteristics 69

B.1.7 Block Diagram 70

B.1.8 Power Supply 70

C REFERENCES **71**

List of Figures

1.1 Block Diagram . . . 2

3.1 Infrared Touch screen . . . 13
3.2 Resistive Touch screen . . . 14
3.3 Capacitive Touch screen . . . 14
3.4 BGA IC Package . . . 15
3.5 Graphical lCD(JHD12864E) . . . 16
3.6 Atmel AVR Studio 4 . . . 17
3.7 Arduino software . . . 18
3.8 Proteus VSM . . . 18
3.9 USBAsp programmer . . . 19

4.1 Touch screen side view . . . 21
4.2 Mechanical Drawing of a 4-Wire TouchScreen . . . 22
4.3 Touch Screen 3D electrical equivalent . . . 23
4.4 Touch Screen equivalent circuits . . . 23
4.5 Touch Screen interface with Microcontroller . . . 24
4.6 The JHD12864E Graphical LCD . . . 24
4.7 Pin connections . . . 26
4.8 Write operation timing diagram . . . 26
4.9 Read operation timing diagram . . . 27
4.10 GLCD left half (or right half) divided into X pages and Y columns . . . 28
4.11 Writing data to GLCD in pixels . . . 29
4.12 I/O Pin Equivalent Schematic . . . 36

5.1 Schematic of Interfacing of GLCD with Atmega32 . . . 45
5.2 Schematic of Atmega8 . . . 46
5.3 L293D DC motor driver . . . 47

A.1 Pinout ATmega32 - PDIP . . . 50
A.2 Block Diagram of Atmega32 . . . 57

A.3 Pinout ATmega8 - PDIP 58
A.4 Block Diagram of Atmega8 66

List of Tables

2.1 Difference between PIC and AVR families 11

4.1 Touch Screen Terminals' Connections 24
4.2 Pin Description 25
4.3 GLCD initial settings 25
4.4 Control bits Pin assignments 27
4.5 Memory features 34
4.6 Timers / Counters 34
4.7 MCU Secific features 35
4.8 Analog I/O 35
4.9 Serial I/O 35
4.10 Programming Modes 35
4.11 Port Pin Configurations 37

Chapter 1

INTRODUCTION

In the computer age of the 21st century, more and more tasks are becoming automated. Automation can make things easier, safer, and often more cost efficient [1]. The stuff, once of world's fairs and science fiction, is a reality today. Automation is the use of control systems and information technologies to reduce the need for human work in the production of goods and services. In the scope of industrialization, automation is a step beyond mechanization [2]. Clearly when involved with routine day to day manual analysis, it is not hard to understand the many possible benefits of an automated technique for a particular procedure, which has made the term 'automation' all the more popular [3].

The ever growing hunger for automation has brought many revolutions in the existing technologies. One of those revolutionary innovations is the touch screen sensor. The release of iPhone in 2007 opened a whole new world of applications involving touch screens [4]. These have greater importance than any other technologies due to their user-friendly nature.

Touch screen based devices can easily be reachable to the common man due to its simpler operation at the same time they challenge the designers of the devices for a complete interactive interface. These touch screen sensors form an indispensable part of home automation where they are used as a replacement to the existing switches. Conventional switch boards are easily prone to accidents, producing sparks and a couple of fire accidents. The main benefit of automation using touch screen sensors is controlling "unknown loss of power". Lights, water heaters, pumping systems, and air-conditioners that may remain switched on because of negligence can be programmed to function for a particular amount of time just by a single 'touch' [5]. The touch screen has two main attributes. First, it enables one to interact directly with what is displayed rather than indirectly with a pointer controlled by a mouse or touchpad. Secondly, it lets one do so without requiring any intermediate devices that would need to be held in the hand. Considering these advantages, most of the advanced automation systems today employ touch screens. The extremely intuitive, effortless and straightforward Graphical User Interface (GUI) using a Touch Screen for navigating through menus, provides an efficient interface which can solve the troublesome way of controlling all of the in-house appliances from a single point.

This provided the rationale for our project. We envisioned making a simple module to control a few in-house appliances like bulb and fan using a touch screen sensor. In addition a small password protection feature for the operation of the module is also implemented. The user can navigate through the menu for control of the appliances. Lighting control (on/off) and speed control of the fan are incorporated giving the user an ease to efficiently use the appliance by a single touch. A detailed study led to the block diagram as shown in fig 1.1.

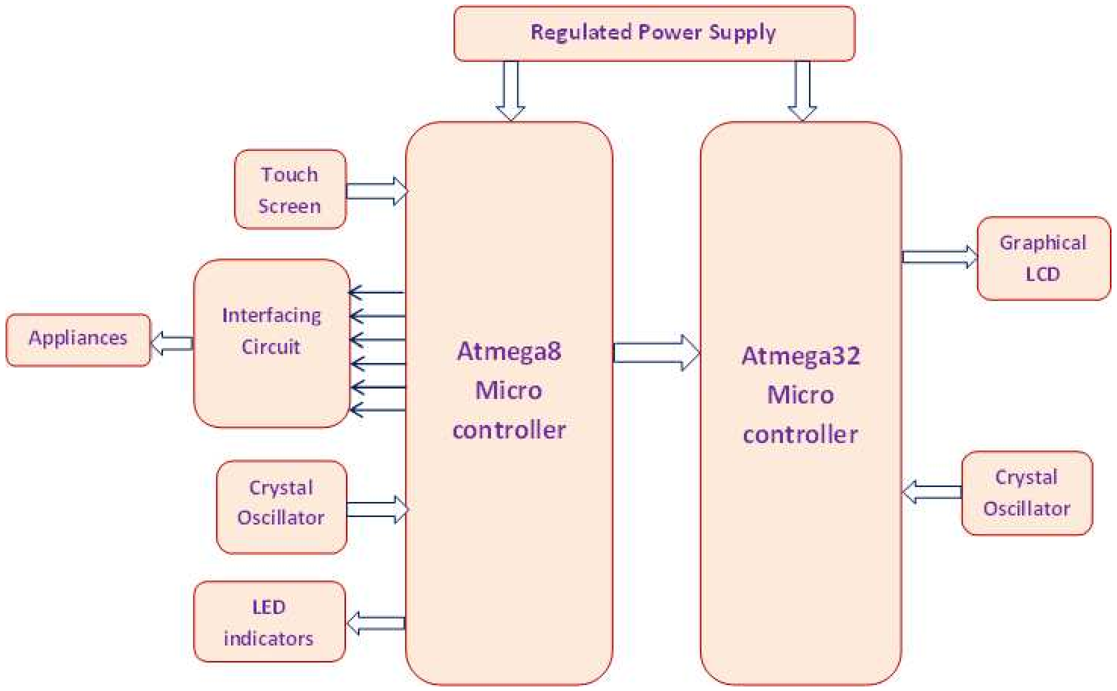


Figure 1.1: Block Diagram

1.1 Block Diagram

The device consists of two microcontrollers - Atmega32 and Atmega8 , which are interfaced with the input and output modules. Since the controllers provide an intermediate medium between the modules, they are collectively termed as the control unit. The input module is a touch screen sensor, which takes the input from the user and feeds it to Atmega8. The output module is a Graphical LCD (GLCD) controlled by Atmega32. The appliances are interfaced to Atmega8 through the required interfacing circuit. Depending on the user input, the necessary control and display is coupled by the controllers to the appliances and the output module, respectively. LeDs indicate the status of the system.

1.2 Working in Brief

The Touch Screen is a transparent glass like screen which senses the touch input by altering one of its characteristics like resistance, capacitance etc. The touch screen maps the input to give 'x' and 'y' co-ordinates of the touch point, in analog form. The co-ordinates of the point are then converted into digital format by the on-chip ADC of the microcontroller to which the touch screen is interfaced (Atmega8). The touch input is processed by the controller and accordingly the control circuit of the desired appliance is activated. The user is given the ability to choose the necessary control action from the menu provided on the GLCD. Depending upon the input from the user, lighting and speed control are initiated by the controller. Simultaneously, signals are sent out to the output controller (Atmega32) to update the display in accordance with the status of the appliance.

1.3 Organization of the Report

The report consists of seven chapters.
Chapter 1 introduces the history of and need of automation and the intention of the project. It also outlines the working of the project in brief with the help of block diagram.
Chapter 2 lists out what all components are required and explains their role in the project. It also gives the details of the literature survey and market research done for each component.

Chapter 3 gives the important points considered in the selection of the particular hardware and software component for the project.
Chapter 4 explains the interfacing of discrete hardware components with the micro-controller.
Chapter 5 deals with the integrated working of the entire project.
Chapter 6 provides a synopsis and some suggestions for future work in this area.

Chapter 2

LITERATURE SURVEY AND MARKET STUDY

As per the introduction, the requirements for the project needed to be researched from various sources. The literature survey involved the detailed study of the various hardware and software components required. This chapter also deals with the market survey done for each.

2.1 Touch Screen

The basic input unit in the project is the Touch Screen. Below is a brief of the various touch screen technologies available.

2.1.1 Infrared Touch Screens

There are two main infrared systems for touch screens[6]: a standard grid and an internal reflection system.

In the standard grid array, infrared LEDs (light emitting diodes) are arranged on opposite sides of the unit underneath the glass. The diodes project infrared light into sensors located directly across from them. The sensors read the strength of the beams, and "when a user makes contact with the screen, the system measures the drop in the sensoroutput signal; this measurement allows the system to compute the location of the touch [7]. This is to say that when the finger touches the screen, the infrared beams are obstructed by the users finger; however, some light continues to pass to the sensor. The sensors send the measurements of light to the operating system, which analyzes the data and recognizes where the user touched. This technology has multi-touch capability because the beams of light are never fully obstructed by the users touch.

The second type of infrared system requires more space than the first. This system is based upon internal reflection; a beam of light is emitted from within the unit, hits the glass, and part of the beam exits through the lens while the other part goes back into the unit. Cameras are placed inside the unit and are calibrated to the standard reflection to the beams so that, "when objects such as fingers touch the surface, the light diffuses at the contact point, causing the acrylics internal-reflection pathways to change. A camera below the surface captures the diffusion and sends the information to image-processing software, which can read multiple touches simultaneously and translate them into a command [8].

2.1.2 Resistive Touch Screens

Resistive touch screen systems are the most common type of touch screen technology in todays market[6].

The resistive touch screen technology operates in a very simple way. These screens are built using two layers of the conductive material Indium Tin Oxide (ITO), separated by a small gap of air. The bottom layer is generally on glass, and the top on a flexible material, often plastic [9]. When the user presses down on the top ITO layer, it physically bends to make contact with the bottom ITO layer, changing the resistance of the two layers [9]. A typical resistive touch screen uses 4 wires, 2 of them on each panel. Each panel corresponds to a different axis. These perpendicular axes allow the computer to take the measurements of the change in resistance from each panel, and calculate the position of the touch point from its X and Y components [7].

2.1.3 Capacitive Touch Screens

Capacitive touch screens are very important within the field of touch screen technology. In the early 1990s, this technology made its initial appearance into the touch screen market in laptop computers, as touch pads [7]. Recently, capacitive popularity has grown, as it has become one of the leading technologies used in touch screen devices. This increase in attention is likely due to the effectiveness of its design, its use of multitouch technology [10]. Capacitive touch screens come in two technologies : Projected Capacitive Touch Screens and Surface Capacitive Touch Screens.

The design of projected capacitive touch screens is somewhat similar to that of resistive touch screens, in that they both utilize 2 layers of ITO, with perpendicular conductive measuring strips on the ends of each layer [9], which are encased between two glass layers. This “grid, formed by the perpendicular conductive layers, projects the electric field through the top layer of glasshence the name projected capacitive touch screens. Because of this projection, when the user touches the top layer of glass it “changes the measured capacitance values of the electrodes closest to it [7]. This change in capacitance is due to the slight electromagnetic charge contained in the human body . These changes in capacitance are measured and calculated as touch points in a very similar way to resistive touch screens, by using the X and Y components.

Surface capacitive is another form of capacitive touch screen technology. The primary difference between surface capacitive and projected capacitive is that surface capacitive uses only one ITO surface [7]. This layer calculates touch points using principles that are very similar to projected capacitive touch screens, in that touch points are observed by changes in capacitance if the ITO layer in the touch screen. However, these touch points are measured in a very different way. The computer measures the change in capacitance from each corner of the ITO layer, and with these 4 separate measurements, the X and Y coordinates of the touch point are calculated [7].

Like the above mentioned, different types of Touch Screens and their details of working were studied from the internet and other sources. Finally, the decision to use the Resistive type was taken for the reasons explained in section 3.1. Touch Screens come in modules with Graphical LCD or as separate unit. The cost of Touch Screen with GLCD module is approximately Rs. 2500. So it was decided to go for a separate unit. The touch screens are easily available everywhere so we bought one for a price of Rs. 500/-.

2.2 Liquid Crystal Display

Since the Touch Screen is transparent, there must be a LCD below it to display whatever touch is registered on the Touch Screen.A liquid crystal display (LCD) is a flat panel display, electronic visual display, or video display that uses the light modulating properties of liquid crystals (LCs)[2]. LCDs are used in a wide range of applications, including computer monitors, television, instrument panels, aircraft cockpit displays, signage, etc. They are common in consumer devices such as video players, gaming devices, clocks, watches, calculators, and telephones.They are available in a wider range of screen sizes than CRT and plasma displays, and since they do not use phosphors, they cannot suffer image burn-in. LCDs are, however, susceptible to image persistence.

The LCD is more energy efficient and offers safer disposal than a CRT. Its low electrical power consumption enables it to be used in battery-powered electronic equipment. It is an electronically modulated optical device made up of any number of segments filled with liquid crystals and arrayed in front of a light source (backlight) or reflector to produce images in color or monochrome. The most flexible ones use an array of small pixels. LCD screens come in passive and active technologies, which are explained here [11].

2.2.1 Passive Displays

Passive displays are widely used with segmented digits and characters for small readouts in devices such as calculators, fax machines and remote controls, most of which are monochrome or have only a few colors. The passive technology is also used for graphics displays, comprising a matrix of rows and columns, typically not larger than 240 rows (320x240). Used in myriad applications today as well as in the first laptops years ago, these graphics-based “passive matrix” screens can be monochrome or full color.

Compared to active matrix, passive matrix is less costly because transistors are used only to activate rows and columns, not each subpixel, resulting in fewer manufacturing steps. However, passive matrix screens have a narrower viewing angle than active matrix and suffer from “submarining,” which is the disappearance of the cursor when moved quickly. A few types of passive displays are as given:

- *TN - Twisted Nematic (90 Twist)*: The first LCD type, TN is used in low-cost readouts for consumer products, and it is also the foundation for active matrix color.

- *STN - Supertwisted Nematic (240-270 Twist)*: Widely used in the past, STN LCDs use birefringence to absorb and pass selective light wavelengths.

- *FSTN - "Film Compensated" STN*: Widely used for passive color matrix screens, an optical film layer turns the STN color into a neutral density light source. Red, green and blue filters are added for full color.
- *DSTN - "Double Layer" STN*: Used in high-temperature environments, a second, but inactive, LCD layer functions like the film in FSTN, except that the layer's optical properties change at the same rate as the working layer. DSTN used to mean "dual scan" STN, which enabled higher laptop resolution by addressing two modules simultaneously; for example, two 240-line passive matrix subsystems created 480 lines.
- *ESTN and ISTN*: Proprietary STN displays from Varitronix.

2.2.2 Active Matrix Displays (TFTs)

Used in all LCD TVs and desktop computer monitors and 99.9% of all laptops, active displays are essentially "active matrix" displays and almost always color. Whether active or passive, a pixel matrix is addressed by rows and columns, one row (line) at a time for each electronic frame and then repeated for the next frame. Unlike passive matrix LCDs, which have no internal transistors, active matrix displays have a transistor at each red, green and blue subpixel that keeps the subpixel at the desired intensity until that row is addressed in the next frame.

By driving the subpixels independently, active matrix screens are sharper and have more contrast than passive matrix, and their faster response times eliminate submarining. In addition, active matrix screens are very bright indoors because they use a backlight. However, when active matrix cellphones and laptops are taken into bright sunlight, they can be overwhelmed with reflected ambient light and difficult to read.

In the early days of laptops, active matrix screens cost a lot more than passive matrix, and both types were offered. Today, active matrix is the only type of LCD on laptops. Also called a "thin film transistor LCD" (TFT LCD) because a thin layer of transistors is deposited on the back of the screen, active matrix displays use TN liquid crystals with a 90 twist.

2.2.3 LCD Screen Colors

Passive displays are monochrome TN, STN and FSTN, and passive color displays are TN and FSTN. Active matrix displays are color TN.

As already said active LCDs have higher cost, since our project incorporates a very basic module we decided to go for a passive LCD. Among the passive displays there are mainly two types:

1. Alphanumeric LCD
2. Graphical LCD

Alphanumeric LCDs display characters in pre-designated blocks (matrix of 5x7 pixels). This limits their use to simple number and character displays and crude images drawn from numbers or characters. While this is suitable for many applications, our project needed to display various images. So it was found better to use a graphical LCD.

The cost of the original Samsung 128x64 GLCD is Rs.3000 and that of the Chinese version varied from Rs.550 to Rs.750. The Chinese version was bought for Rs.550/-.

2.3 Microcontrollers

The main component controlling all the components listed above is the microcontroller. Today there are innumerable microcontrollers from different companies, with different features, in different versions and with different advantages over others. We conducted a detailed study with a few of the recent popular microcontrollers and found that PIC and AVR series were most easy and feasible for interfacing with GLCDs. Table 2.1 [12] gives the differences between the two sets and gives a clear cut definition as to why the AVR series were chosen in our project.

Other than those mentioned in Table 2.1, AVRs have non-banked access to data memory unlike PICs [13], which seem to require setting bank registers to access beyond 256 bytes of memory. Also, some AVRs support hooking up external SRAM in a way that allows the MCU to use it natively (rather than going through a series of port accesses). AVRs have 32 general purpose registers, the PIC only has one.

The ATmega and PIC18F have hardware multipliers. AVRs are a GCC targets and also appear to support a more generalized interrupt system, as opposed to the PIC high/low priority interrupt vectors.

Table 2.1: Difference between PIC and AVR families

PIC	AVR
One instruction per 4 clock cycles (4Mhz = 1MIPS)	One instruction per clock cycle (4Mhz = 4MIPS)
Wide voltage range and reliable operation in conditions that exceed specifications	Works only under condition that conform to the specifications
Most parts require an external clock source (e.g. crystal)	Most parts are able to work by using AVR's internal clock sources.
More years in the market	About two years less in the market
Easy to learn instruction set (only 33 asm instructions)	More instructions (110 asm instructions)
More than one operation to achieve simple tasks	Everything can be completed with one ore two instructions
Difficult indirect memory access and computed gotos	Very easy indirect memory access and computed gotos with pointer registers and special instructions
Difficult to use Memory and Program memory paging	Easy to use memory model
Development enviroment: MPLAB IDE	Development enviroment: AVR Studio
Device specific assembly difficult to compile C code to	Assembly optimized for use with C compilers and similar to other mCs
Many support devices, Simulators - Programmers integrated with IDE	STK 500 all in one. Complete integrated into the IDE
Easy to learn, introductory, perfect for educational use	High performance, more powerful assembly, less expensive

On the basis of this background, the AVR series are selected. Due to their favorable features, (discussed in 3.4) Atmega32 and Atmega8 were selected the former costing Rs.200/- and latter Rs.100/-

2.4 AVR Programmer

This is used to download files into the Atmega32 microcontroller. The programmer that we are using is AVR DUDE which downloads the program into the AVR and is capable of programming the fuse bits. For this particular project we are using the USBasp which is a USB in-circuit programmer for. It simply consists of an Atmega8 and a couple of passive components. This was bought for Rs 450/-.

2.5 Embedded 'C' Programming Software

Writing such a huge code in assembly is tedious and hence a C compiler for the microcontroller was needed. This would ensure easy programming and handling of various device libraries. Atmel AVR studio and Arduino software were chosen for this purpose. While AVR Studio has a Integrated Development Environment (IDE) for developing and debugging embedded Atmel AVR applications, Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Both are freely available online.

2.6 Simulation Software

A simulation software was needed considering the fact that the hardware need not be changed in accordance with the changes in the software. This flexibility was needed since it would require less time to design the hardware. The benefit is that the hardware design may be changed just as easily as the software design. Proteus VSM suite was chosen for the purpose. It offers the ability to co-simulate both high and low-level micro-controller code in the context of a mixed-mode SPICE circuit simulation. This was available on the Internet for free.

2.7 L293 Motor Driver

As the MCUs PORT are not powerful enough to drive DC motors directly so we need some kind of drivers. A very easy and safe is to use popular L293D chips. The L293 is an integrated circuit motor driver that can be used for simultaneous, bidirectional control of two small motors. The L293 is limited to 600 mA, but in reality can only handle much small currents unless you have done some serious heat sinking to keep the case temperature down. The driver IC costs Rs 40/-.

Chapter 3

HARDWARE AND SOFTWARE SELECTION

After noting down the hardware and software components required, the next thing to go for is the selection of a particular component depending on its features, cost and advantages over others from the same category. This chapter lists out the specific components selected and discusses the points considered for their selection.

3.1 Touch Screen

Three types of touch screens depending were discussed in the previous chapter. A few factors helped us choose resistive type over the other touch screen technologies.

The two types of infrared systems are internal reflection and infrared grid. Infrared grids systems are reliable and can be manufactured inexpensively into appropriate sizes.

Due to the grid of lasers and sensors, users do not need to press fully down on the screen putting less wear on screen increasing the life expectancy of the unit. The grid also increases the precision of the users touch. Internal reflection systems are large systems because of the space required for cameras to accurately measure the shadow produced by the infrared LEDs.

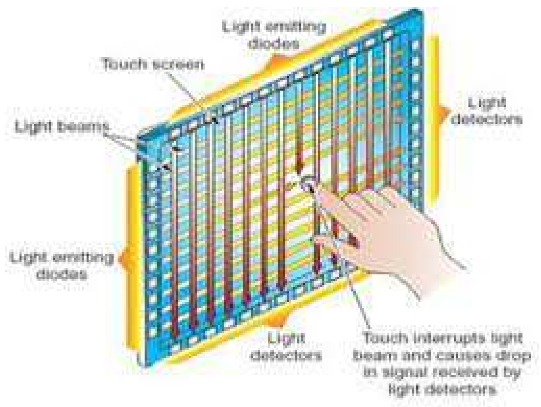


Figure 3.1: Infrared Touch screen

The large space required for larger instruments does make internal reflection devices the most accurate touch screen technology. IR (infrared) screens are among the most durable surfaces and can handle hostile environments, making them well suited for military applications. Infrared touch screens are best suited to devices like the Microsoft Surface, which require a very large touch screen.

Resistive touch screen technology is the cheapest of the different types of touch screens. These devices are “less than ideal for harsh environments, due to the fact that they are vulnerable to temperature and humidity changes, which would affect the accuracy of the touch screen [7]. This is not to say that they do not perform consistently under standard conditions. One benefit of this technology is that the user is able to use his or her finger or a stylus as input devices. Overall, resistive touch screens perform very well, considering that they are the cheapest of the different touch screens. Resistive touch screens are best suited for mobile applications, in which conditions will be consistent.

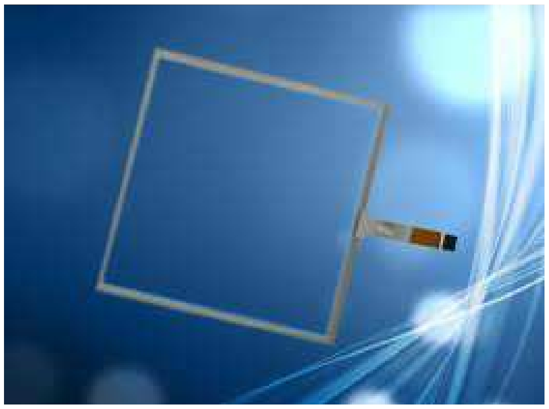


Figure 3.2: Resistive Touch screen

Capacitive touch screens are very common in many consumer devices. Although there are two different types of capacitive touch screens, their performance is very similar, with the exception that projected capacitive touch screens are a little more accurate than surface capacitive touch screens, but this difference is relatively negligible. A drawback of this technology is that you can only touch the screen with your finger. This means that stylus and gloves, depending on their thickness, will not work with this technology. Another drawback is the cost of the screen. These screens are more expensive than resistive touch screens. Overall, capacitive touch screens are very effective in their current uses. Capacitive

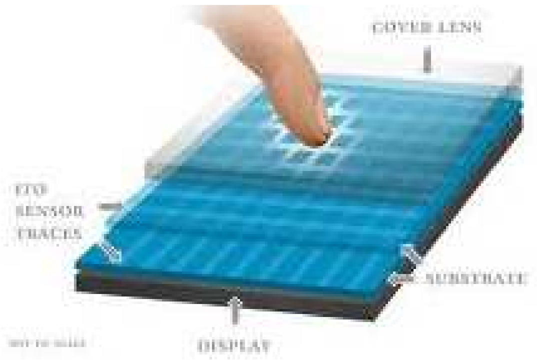


Figure 3.3: Capacitive Touch screen

touch screens are best suited for high end, portable electronic devices, and devices that need to perform consistently in many conditions.

The main difference between touch screen technologies is size and cost. Infrared touch screens are by far the biggest of the touch screen technologies. They are also more expensive than resistive and capacitive touch screens. The size of resistive and capacitive touch screens is relatively similar, due to the similar nature of their technologies. However, there is quite a difference in price between these touch screens. Capacitive touch screens are more expensive than resistive touch screens because the systems of circuitry and measurement are more complex.

From the above discussed issues, resistive type was found apt for our project. Resistive touch systems are available in 4-, 5-, and 8-wire variants. 5-wire touch screens are more reliable than 4-wire but are not quite expensive. Eight-wire technology, in comparison with four-wire resistive technology, uses four additional sensing points to stabilize the system against drift, but the additional points do not improve the durability or life expectancy of the screen.

It was decided to use the 4 wire touch screen due to low cost and easy usage, the details of its operation are explained in chapter 4.

3.2 Touch Screen Controller

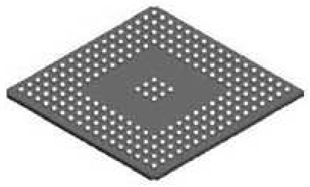


Figure 3.4: BGA IC Package

The location of the point touched on the Touchscreen can be found by microcontroller. But there are ICs present in the market specifically made for this purpose. TSC2005 is one such touch screen controller from Texas Instruments which provides the X-Y co-ordinate of the touched point through SPI. But the BGA (Ball Grid Array) IC package of these controllers makes it unfeasible for students to solder it because of its extremely small size (4mm x 4mm with 28 pins). An example of BGA

IC package is shown in Figure 3.4.

So it was finally decided to make the microcontroller get the touched point's co-ordinates from the Touch Screen, using ADC.

3.3 Graphical LCD

Graphical LCDs are categorized by the type of controllers built into them. There are three types of on-chip controllers for Graphical LCDs:

1. T6963
2. KS0108
3. SED

The Graphical LCD with T6963 and SED controllers were not available in the local market. So we went for KS0108 with a resolution of 128 pixel x 64 pixel, model no. is JHD12864E (Yellow-Green version) as shown in Figure 3.5.

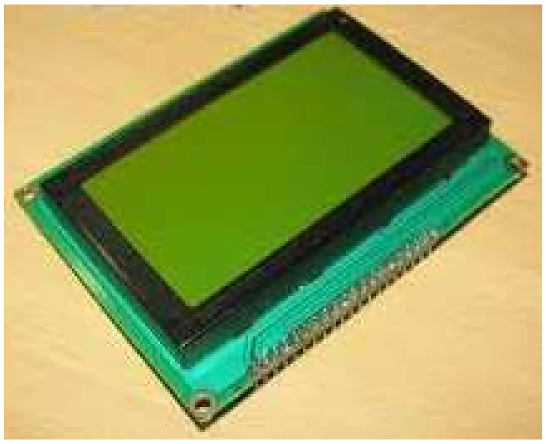


Figure 3.5: Graphical lCD(JHD12864E)

Once this GLCD was bought, it needed a software driver (library of functions that can be used directly in the main 'C' program). But there are no such drivers available for KS0108 GLCD. Also there is very little information on writing graphical LCD drivers for all the three types. Even the little information that was available was very crude. Finally, after a lot of research we were able to find a library which provided all the necessary functions of GLCD in the favor of the project. This library and its functions will be discussed in the next chapter. The details of working of the GLCD are explained in section 4......

3.4 Microcontrollers

Atmel AVR microcontrollers like ATmega 8 / 32 were selected since they provide many features like:

- Bidirectional I/O ports (for the Touch Screen).
- On chip ADC (for the Touch Screen), Timer/Counter (for RTC and mouse interface) and UART (for serial communication).
- On chip I2C (for RTC) and SPI controller (for SD Card interface).
- They can work at a maximum frequency of 16MHz and hence are quite fast.
- 16kB/32kB Flash and 512Bytes EEPROM memories for lower power consumption and more write/erase cycles (for storing large images that needed to be displayed on the GLCD).
- In System Programmable i.e. the microcontrollers can be programmed without removing them from the circuit.
- Very easy to program with a simple home made programmer (STK200 or USBASP).

Other parameter that influenced the choice of microcontroller in favour of Atmel AVR included the programming software “Atmel AVR Studio”.

3.5 Embedded Programming Software

We used two softwares to program the two microcontrollers. Atmega32 was programmed with “Atmel AVR Studio” while Atmega8 was programmed with “Arduino software”. The programming with two different softwares was just done for the sake of simplicity.

The Atmel AVR Studio 4 is an Integrated Development Environment for debugging AVR software. The AVR Studio allows chip simulation and in-circuit emulation for the AVR family of microcontrollers.

Figure 3.6: Atmel AVR Studio 4

The user interface is specially designed to be easy to use and to give complete information overview. The AVR uses the same user interface for both simulation and emulation providing a fast learning curve.

The Arduino environment is based on Atmel Atmega microcontrollers. The AVR language is a "C" environment for programming Atmel chips. Much of the Arduino language is written with AVR constants and functions and there are many things that are still not easy to accomplish with the Arduino language without using some AVR code. Because the Arduino environment uses GCC to compile code, all of the port and register variables found in the Atmega datasheets and tutorials on AVR code assembly language are supported when using the Arduino IDE.

Figure 3.7: Arduino software

3.6 Simulation software

Proteus is the simulation software that was selected for simulation of our code. It has a huge library of components and a good online support. Thus, everything from a capacitor to microcontroller, temperature sensors and RTC can be simulated in Proteus.

The only area in which Proteus' performance degrades is while executing memory operations for example, interfacing with SD Card. But since the project didn't involve such critical memory operations, Proteus helped to simulate almost the entire code except touchscreen (since that needed to be checked using the actual hardware).

PROTEUS
DESIGN SUITE
ISIS SCHEMATIC CAPTURE
Electronics Design
From Concept
To Completion
ISIS Professional v7.1 SP2 © Labcenter Electronics 1989-2007
Loading library "REALTIME.LIB"
Figure 3.8: Proteus VSM

3.7 Software to download firmware in the MCU

AVRDUDE is an open source utility to download/upload/manipulate the Flash and EEPROM contents of AVR microcontrollers using the in-system programming technique (ISP). The USBAsp programmer as shown in figure 3.9 is used in collaboration with AVRDUDE to form the programmer interface to the controllers.

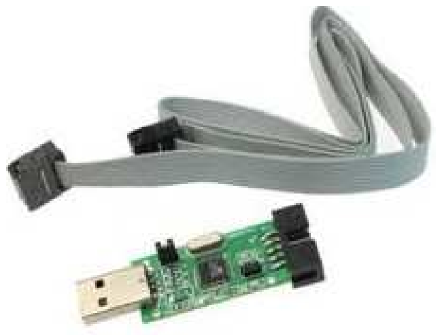


Figure 3.9: USBAsp programmer

The major features of AVRDUDE include:

- Command-line driven user interface for all downloading and uploading features (including handling fuse bytes), for easy automation.
- Interactive examination and modification of various memory regions in the so-called terminal mode.
- Known to run on FreeBSD, MacOS X, Linux, and Win32 platforms. On Win32 platforms, parallel port access requires the previous installation of a driver (giveio.sys) that grants a user process direct access to the IO registers.
- Supports a wide range of programming hardware, from cheap ISP plugs that connect the AVR's ISP interface directly to a computer's parallel port (no additional circuitry) or serial port (some additional circuitry needed), more advanced ISP adapters using a buffer/driver chip (like a 74HC373), up to (more complex) serially connected programmers like AVR910-style ISP devices, the Atmel STK500 board, and the Atmel JTAG ICE mkII. Most popular adapters come pre-defined, adding a new parallel-port adapter is as simple as editing a configuration file (no recompilation needed).

- Supports Intel Hex, Motorola S-Record, and raw binary files for input and output, as well as direct memory contents specification on the command-line (useful e.g. for fuse bytes). On input, the file format can be auto-detected.
- In “terminal mode”, the device’s memory areas can be examined, and possibly modified. This allows to set fuses interactively, or to modify a few EEPROM cells.

After the selection of all the hardware and software components it is required to implement these units. The next chapter deals with the working and implementation of each hardware component. It gives all details that are required to use the particular unit in the project.

Chapter 4

HARDWARE IMPLEMENTATION

This chapter gives all the details required to implement the various hardware units selected in chapter 3 along with the construction, to interface the units with MCU.

4.1 Touch Screen

Analog touch screens, as already stated, are available in three types, 4-wire, 5-wire and 8-wire. All analog touch screens consist of a ridged layer and a flexible layer with a separation layer between them. The flexible layer is exposed to the outside, towards the user. The inside surfaces of the ridged and flexible layers are coated with a resistive coating, usually ITO (indium tin oxide). When the flexible layer is pushed its conductive surface will make contact with the conductive surface of the ridged layer making an electrical connection between the two layers at the point of contact.

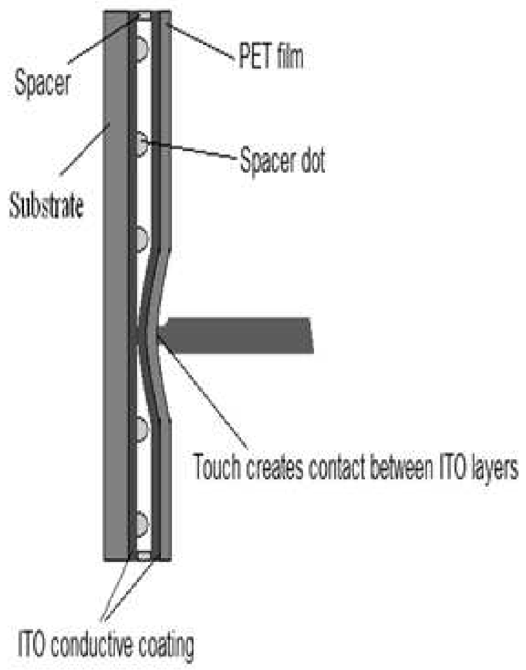


Figure 4.1: Touch screen side view

Measurements are then made to determine the point of contact. A 4-wire touch screen was chosen due to the advantage of having a very simple interface with the microcontroller; the construction and working being explained below. It was required to operate the Touchscreen within a specific area called as the active area which is as shown in Figure 4.2.

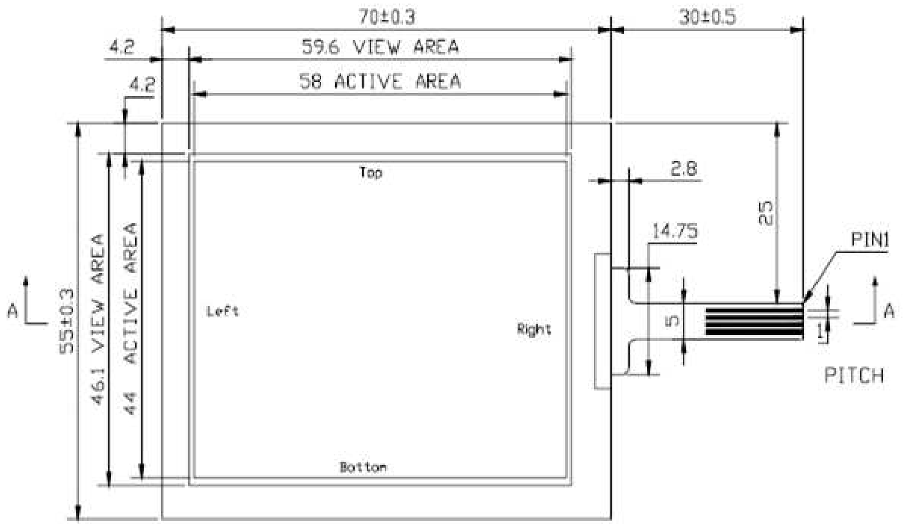


Figure 4.2: Mechanical Drawing of a 4-Wire TouchScreen

Usually a resistive touch screen consists of at least three layers as shown in Figure 4.1: A flexible membrane made from PET film is suspended over a rigid substrate made from glass or acryl. Both surfaces are coated with a transparent conductive film like ITO (Indium tin oxide). The conductive ITO layers are kept apart by an insulting spacer along the edges, and by spacer dots on the inner surface of the two ITO layers. In this way there will be no electrical connection unless pressure is applied to the top sheet (PET film).

4-wire touch screens use a single pair of electrodes or Bus bars on each ITO layer. The bus bars in the top sheet and substrate are perpendicular to each other. The bus bars are connected to the touch screen controller through a 4-wire flex cable. The 4 wires are referred as X+ (left), X- (right), Y+ (top) and Y- (bottom).

An advantage of the 4-wire touch screens is that it is possible to determine the touch pressure by measuring the contact resistance i.e. (R_{touch}) between the two ITO layers. R_{touch} decreases as the touch pressure increases. This logic was used to find the ADC values before using the touch screen. The code written works using the ADC conversion without using interrupts.

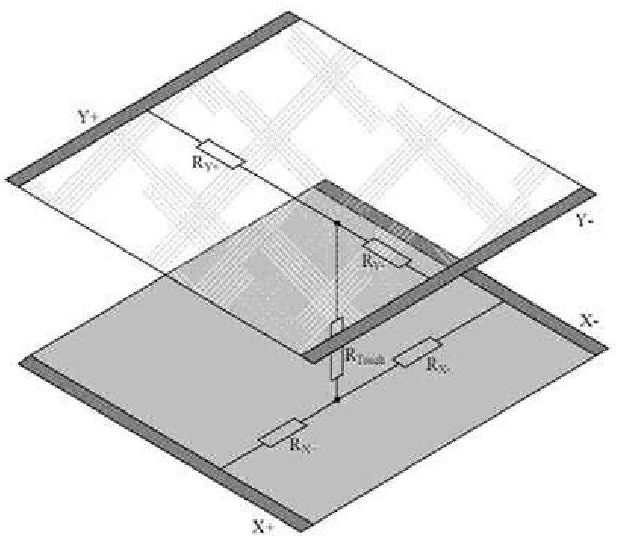


Figure 4.3: Touch Screen 3D electrical equivalent

The point of contact divides each layer in a series resistor network with two resistors and a connecting resistor (R_{touch}) between the two layers. By measuring the voltage at this point the user gets information about the position of the contact point orthogonal to the voltage gradient. To get a complete set of coordinates, the voltage gradient must be applied once in vertical and then in horizontal direction: first a supply voltage must be applied to one layer and a measurement of the voltage across the other layer is performed, next the supply is connected to the other layer and the opposite layer voltage is measured.

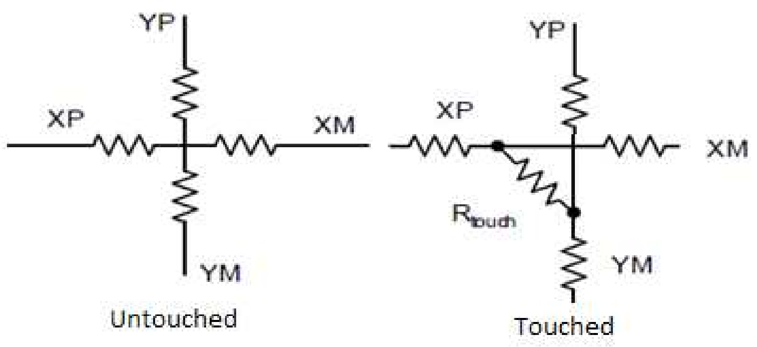


Figure 4.4: Touch Screen equivalent circuits

The X and Y positions on the screen were implemented in C with the help of the Figure 4.5 and Table 4.1.

Table 4.1: Touch Screen Terminals' Connections

Measurement	X+ Excite	X- Excite	Y+ Excite	Y- Excite
Standby	GND	Hi-Z	Hi-Z	Pull upInt
X-Coordinate	GND	VCC	Hi-Z	Hi-ZADC
Y-Coordinate	Hi-Z	Hi-ZADC	Gnd	VCC

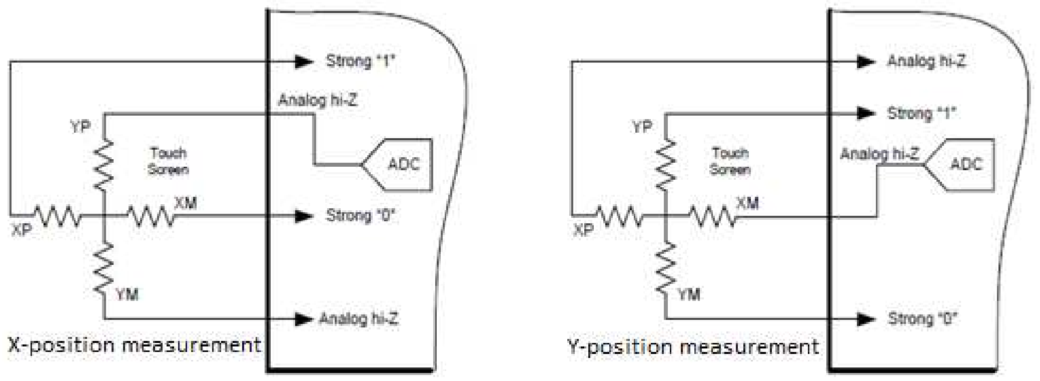


Figure 4.5: Touch Screen interface with Microcontroller

4.2 Graphical LCD - JHD12864E

The JHD12864E is a 128 x 64 pixel graphical LCD with backlight. It is driven by two 64 x 64 pixel Samsung KS0108 drivers. Figure 4.6 shows an image of what the graphical LCD(blue version) looks like with a sample output of a cartoon character. The actual view area of the LCD is 60 mm x 32.6 mm.

Figure 4.6: The JHD12864E Graphical LCD

Table 4.2: Pin Description

PIN NO	SYMBOL	DESCRIPTION	FUNCTION
1	VSS	GROUND	0V(GND)
2	VDD	POWER SUPPLY FOR LOGIC CIRCUIT	+5V
3	V0	LCD CONTRAST ADJUSTMENT	
4	RS	INSTRUCTION/DATA REGISTER SELECTION	RS=0: Instruction register
			RS=1: Data Register
5	R/W	Read/write selection	R/W=0: Register write
			R/W=1: Register read
6	E	Enable Signal	
7	DB0	DATA INPUT/OUTPUT LINES	
8	DB1		
9	DB2		
10	DB3		8BIT:DB0-DB7
11	DB4		
12	DB5		
13	DB6		
14	DB7		
15	CS1	CHIP SELECTION	Cs1=1, chip select signal
			for IC1
16	CS2	CHIP SELECTION	Cs2=1, chip select signal
			for IC2
17	RST	RESET SIGNAL	RSTB=0,DISPLAY OFF,
			DISPLAY FROM LINE0
18	VEE	NEGATIVE VOLTAGE FOR LCD DRIVER	-10V
19	LED+	SUPPLY VOLTAGE FOR LED+	+5V
20	LED-	SUPPLY VOLTAGE FOR LED-	0V

4.2.1 Pin Assignments

This display has 14 I/O pins (for control and data) and 6 power related pins. Table 4.2 gives brief details of the GLCD pins.

4.2.2 Setting up the LCD and turning it ON

In order for the LCD to power up, the connections as shown in Table 4.3 should be made.

Table 4.3: GLCD initial settings

Pin No	Symbol	Input/Output	Typical Value
1	VSS	Input	0V
2	VDD	Input	+5V
3	VO	Input	More negative than -3.5V
18	VEE	Output	-10V
19	LED+	Input	5V
20	LED-	Input	0V

As mentioned in Table 4.3, pin3 VO of GLCD is used for contrast adjustment. The voltage on this pins should be more negative than -3.5V for blue version of

GLCD and more negative than -7.5V for the green version. Hence the pins should be connected as illustrated in Figure 4.7.

Pin18 generates -10V as an output and it must be run through a trimpot (or voltage divider) and fed into Pin 3. This provides the differential voltage of VDD - VO which must be above the threshold mentioned above. Adjusting this value adjusts the contrast but the Pins must be connected in this way in order for the image to be seen on the screen (if it isn't there is essentially no contrast and nothing will be displayed).

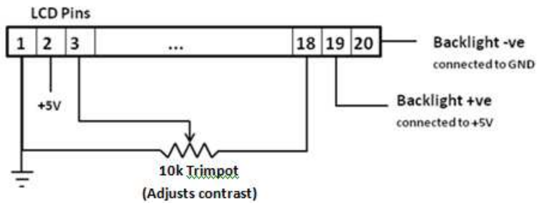


Figure 4.7: Pin connections

4.2.3 Data transfer

There are 8 data bits that provide a number of functions in the operation of the LCD. They are the main information carriers to the LCD. They are located on Pins 4 - 11 with Pin 4 assigned to Data Bit 0 and Pin 11 assigned to Data Bit 7. The pixel information to be displayed, commands that are given to the GLCD and the data that is read from the GLCD display RAM, all are transferred through these 8 pins. These 8 pins should be connected to an 8 bit port of the microcontroller.

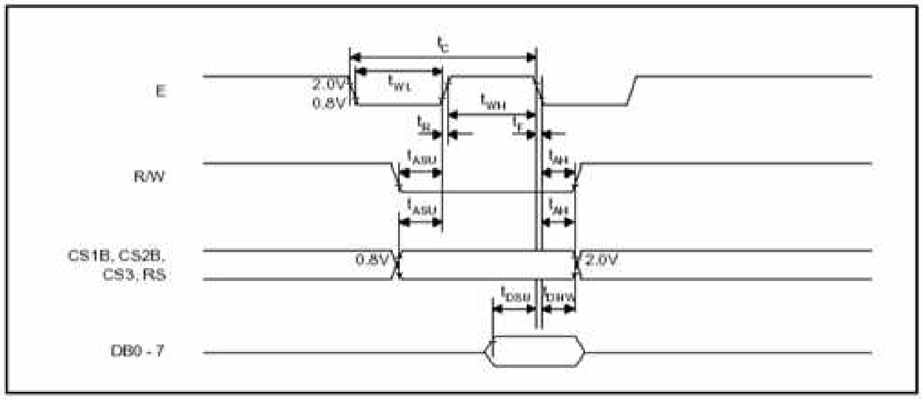


Figure 4.8: Write operation timing diagram

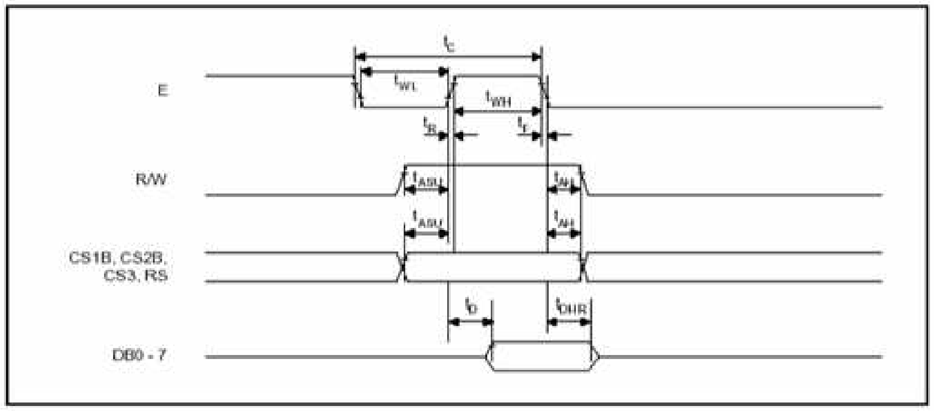


Figure 4.9: Read operation timing diagram

4.2.4 Control bits

The KS0108 Graphical LCD is divided into two sections- left and right. Each section is controlled by a separate on-chip controller. This particular configuration makes it difficult to write to the GLCD and creates a lot of problems which are explained later.

The JHD12864E has 6 control bits which are used to control the operation of the KS0108 hardware drivers and display data on the LCD. The function of each pin is listed in Table 4.4.

Table 4.4: Control bits Pin assignments

Pin no	Symbol	Description	Function
			It tells the LCD whether data or command is being written
4	D/I	Data/Instruction	to the GLCD by the MCU. Set high for data transfer,
			Set low to designate an instruction is being performed.
			Selects whether the MCU wants to read from or write
5	R/W	Read/Write	to the GLCD.Set high to read from the LCD to the MCU,
			Set low to write from the MCU to the LCD.
			The enable is used to clock operations to the LCD -
6	E	Enable	it ends up being the clock of the LCD and when it clocks,
			sequential instructions are performed
15	CS1	Chip Select 1	Selects the left KS0108 driver
			which is also left half of the screen
16	CS2	Chip Select 2	Selects the right KS0108 driver
			which is also right half of the screen
17	RST	Reset	Set low to reset the display, high otherwise

After knowing about the function of each control bit next thing to look at is the timing of these signals required. The timing diagrams (Figures 4.8 and 4.9) explain the order in which the signals need to be given and the details of the timing are given in detail in the JHD12864E datasheet in the Appendix.

The most important parameter in the GLCD timing is t_{WL}, t_{WH} and t_C. In the datasheet, t_{WL}= t_{WH} = 450ns. But the practical value was found to be 600ns by trial and error. Such strict timing should not (or cannot) be implemented in 'C'. It should be done in assembly language with NOP instruction which is available in all MCUs. If the t_{WL}, t_{WH} values are less than 600ns, the GLCD is obviously not able to respond and nothing or garbage values will be displayed. They can be more than 600ns but they should not very high (i.e. in few ms) otherwise the GLCD response gets sluggish which is never desired.

4.2.5 How the Input Bits translate into what is on the Screen

In order to place any information on the screen, it is important to understand how the bits control what is on the screen. There are 8192 pixels on a 128 X 64 pixel screen and each pixel is controlled by a series of instructions. Figure 4.10 and Figure 4.11 show how the screen is broken down into its X and Y axes. Figure 4.10 shows the left half of the screen (if CS1 is 1 and CS2 is 0). The Y address refers to which line the pixels should be written to and the X page sets the column to which they will be written to. Please see the Appendix for how to set the Y address and X page functions, but there are three basic steps that must be done in order to determine where the pixels will go. First, the Y address must be set. The Y address actually has a counter so it need only be set once and then every time when there is a data write or data read, it will be incremented to the next column. This allows the driver to scan through the lines and display the proper data. The next step that must be completed is setting the X page. This determines which column of the screen will now be written to. The third step is issuing a data write command. Whatever data bits are high will be darkened on the Y address line

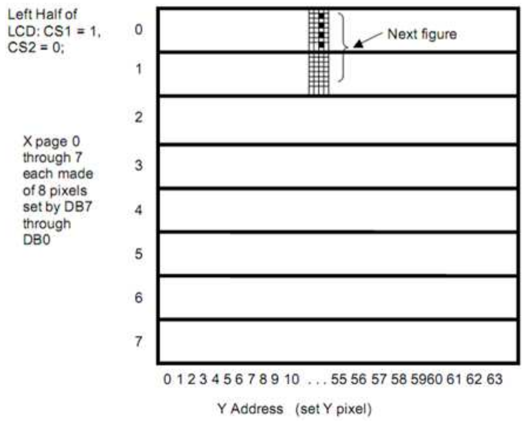

Left Half of LCD: CS1 = 1, CS2 = 0;
Next figure
X page 0 through 7 each made of 8 pixels set by DB7 through DB0
0
1
2
3
4
5
6
7
0 1 2 3 4 5 6 7 8 9 10 . . . 55 56 57 58 5960 61 62 63
Y Address (set Y pixel)

Figure 4.10: GLCD left half (or right half) divided into X pages and Y columns

in the X page as is illustrated in Figure 4.11. The driver scans through the pages, using the internal Y address counter to its advantage and resetting the X page (and eventually the CS) as it scans through the lines.

Figures 4.10 and 4.11 are added to this section because they do a much better job of explaining what is going on pixel by pixel on the screen than do any of the data sheets that you may see. One very important thing to note is that the X and Y axes are reversed from what they usually would be. It might have something to do with how the two KS0108 controllers are connected. Every 128 x 64 graphical LCD is arranged in this way even if it is not controlled by the KS0108. So it is the convention.

To reiterate, to select which bit on the screen will be effected:

- Set Y address
- Set the X page
- Write data to or read data from the 8 bits DB7-DB0.

The process of selecting where on the screen the bit is going to go is just a narrowing process. CS1 and CS2 narrow it down to half the screen. Y address narrows it down to a 1 bit wide stripe of height 64 in that half of the screen. X page picks an 8 bit chunk of this 64 bit stripe and data is written to or read from that chunk in 8 bits. This is the essence of putting bits on the screen.

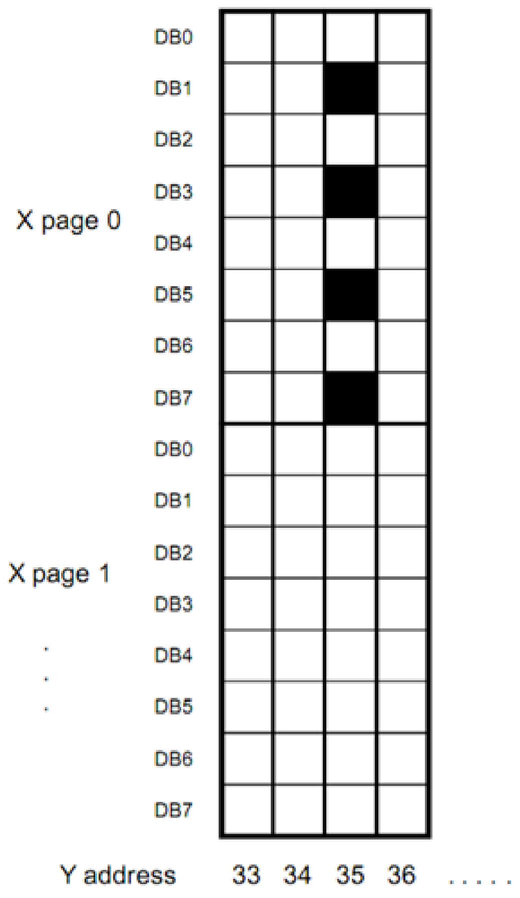


Figure 4.11: Writing data to GLCD in pixels

As shown in Figure 4.10, the Y - address determines what Y line the pixels will be placed on, the X page determines which of the eight vertical 8 pixel strips the pixels will be placed on and the Data bits of the write data instruction will be the places across the X page on the specified Y line.

Figure 4.11 shows a zoomed version. It also gives an example of how a pixel is written. If the Y address is set to 35, X page is set to 0, and a data write instruction is given with PORTx = $(10101010)_2$, then the pixels in those locations will be dark. Now if without changing X page and Y address another data write is done, then it will come on the next column (Y=36) due to the internal Y address counter which increments after every data read or write instruction.

4.2.6 Functions implemented

To help in using the GLCD for generating the user interface, various functions are made. All these functions together make the ProGFX. ProGFX is a free Graphics Engine For Embedded Systems. It is a full-featured driver for KS0108 based Graphic LCDs. The important header file is the 'gfx.h'. Here the basic working of each function is explained in brief with respect to its use in the project.

- *void GFXInit()*
  This function Initializes the ProGFX Graphics Engine. This function must be called before using any other graphic functions.

- *void GFXUpdate()*
  This function transfers the internal graphic buffer to the LCD Module. After the drawing is complete, this function must be called to make it visible on screen.

- *void GFXRawPutPixel(UINT8 x , UINT8 y , UINT8 color)*
  This function Draws a single pixel on screen. Here "x" represents the x co-ordinate, "y" the y co-ordinate and color can be either GFX _COLOR _BLACK or GFX _COLOR _WHITE.
  Eg: GFXRawPutPixel (void (0 , 0 , GFX _COLOR _BLACK) ;
  This command draws a single pixel at the origin, that is at the location (0,0) in Black color.

- *void GFXHLine(UINT8 x1 , UINT8 y1 , UINT8 len , UINT8 color)*
  This function Draws a Horizontal Line. Here "x1" represents the starting x co-ordinate, "y1" the starting y co-ordinate, "len" the length of line and colour can either be GFX _COLOR _BLACK or GFX _COLOR _WHITE.

Eg: GFXHLine(10, 20, 5, GFX _COLOR _WHITE);
This command draws a line starting at co-ordinates (10,20) having a length of 5 units in white color.

- *void GFXVLine (UINT8 x1 , UINT8 y1 , UINT8 len , UINT8 color)*
  This function Draws a Vertical Line. Here "x1" represents the starting x co-ordinate, "y1" the starting y co-ordinate, "len" the length of line and colour can either be GFX _COLOR _BLACK or GFX _COLOR _WHITE.
  Eg: GFXHLine(64, 20, 5, GFX _COLOR _WHITE);
  This command draws a line starting at co-ordinates (64,20) which comes in the second half of the screen having a length of 5 units in white color.

- *void GFXRect(UINT8 x1,UINT8 y1,UINT8 x2, UINT8 y2,UINT8 color)*
  This function Draws a Rectangle. Here "x1" represents the top left x co-ordinate, "y1" the top left y co-ordinate, "x2" represents the bottom right x co-ordinate, "y1" the bottot right y co-ordinate, "len" the length of line and color can either be GFX _COLOR _BLACK or GFX _COLOR _WHITE.
  Eg : GFXRect(10, 0 , 60 , 20 , GFX _COLOR _WHITE);
  This command draws a rectangle starting at top left co-ordinates (10 , 0) and ending at bottom right co-ordinates (60,20) in white color.

- *void GFXFillRect(UINT8 x1,UINT8 y1,UINT8 x2, UINT8 y2,UINT8 color)*
  This function Draws a Filled Rectangle. Here "x1" represents the top left x co-ordinate, "y1" the top left y co-ordinate, "x2" represents the bottom right x co-ordinate, "y1" the bottot right y co-ordinate, "len" the length of line and color can either be GFX _COLOR _BLACK or GFX _COLOR _WHITE.
  Eg: GFXFillRect(40, 20 , 50 , 40 , GFX _COLOR _BLACK);
  This command draws a rectangle starting at top left co-ordinates (40 , 20) and ending at bottom right co-ordinates (50,40) in black color.

- *void GFXClear()*
  This function Clears the screen. It must be used before any new screen is to be displayed in order to erase the previous screen.

- *INT8 GFXPutCharXY(UINT8 x, UINT8 y,char c,UINT8 color)*
  This function Draws the given char. The function is of type INT. Here "x" represents the x co-ordinate, "y" the y co-ordinate and color can be either GFX _COLOR _BLACK or GFX _COLOR _WHITE. Also "char" is the character

that the user wants to draw. Since the function is of type INT, it has a return value of 1 to indicate successful printing of character on the screen and -1 to indicate a failed attempt.
Eg: GFXPutCharXY(5 , 5 , x , GFX _COLOR _BLACK);
This command writes onto the screen, the character x at location (5,5) in Black color.

- *INT8 GFXGetCharWidth(char c)*
  Returns the width of given character c in the current selected font. The character can be anything like ‘a’. The function is of type INT and hence has a return value, that is the width of the character in pixel.
  Eg: GFXGetCharWidth(“a”);
  This command will give the width of character a in pixels.

- *INT8 GFXGetCharHeight(char c)*
  Returns the height of given character c in the current selected font. The character can be anything like ‘a’. The function is of type INT and hence has a return value, that is the height of the character in pixel.
  Eg: GFXGetCharWidth(‘h’);
  This command will give the height of character ‘h’ in pixels.

- *INT8 GFXGetStringWidth(char ∗string)*
  This function returns the width of given string in the current selected. The string could be any C string like “Hello World”. It has a return value that is the width of the string in pixels.
  Eg: GFXGetStringWidth(∗p);
  This command will get the width of the string pointed by the pointer p in pixels.

- *INT8 GFXWriteStringXY(UINT8 x,UINT8 y,const char ∗string,UINT8 color)*
  This function Draws a given string on screen. Here “x” represents the x co-ordinate, “y” the y co-ordinate and color can be either GFX _COLOR_BLACK or GFX _COLOR_WHITE. The string could be any C string like “Hello World”. Since the function is of type INT it has a return value that could be 1 to indicate successful writing or -1 to indicate failed attempt.
  Eg: GFXWriteStringXY(5 , 10 , ∗p , GFX _COLOR _WHITE);

This command will write the string pointed to be pointer p at the location (5,10) in white.

- *INT8 GFXCenterText(const char ∗string,UINT8 color)*
  This function Draws a string in exact center of screen. Here "x" represents the x co-ordinate, "y" the y co-ordinate and color can be either GFX _COLOR_BLACK or GFX _COLOR_WHITE. The string could be any C string like "Hello World". Since the function is of type INT it has a return value that could be 1 to indicate successful writing or -1 to indicate failed attempt.
  Eg: GFXCenterText(∗p, GFX _COLOR _BLACK);
  This command will write the string pointed to be pointer p at the exact center of the screen in white.

- *void GFXCircle(UINT8 cx,UINT8 cy,UINT8 rad,UINT8 color)*
  This function Draws a Circle with given center and radius. Here "cx" represents the x co-ordinate of the center, "cy" the y co-ordinate of the center and color can be either GFX _COLOR _BLACK or GFX _COLOR _WHITE.
  Eg: GFXCircle(10 , 20, 5 , GFX _COLOR _WHITE);
  This command draws a circle centered at (10,20) with a radius of 5 in white color.

- *void GFXDrawImage(UINT8 x,UINT8 y,prog _uint8 _t ∗img)*
  This function Draws an Image. Here "x" represents the x co-ordinate, "y" the y co-ordinate and color can be either GFX _COLOR _BLACK or GFX _COLOR _WHITE. Also "img" is the image to be drawn.
  Eg: GFXDrawImage(20,20,img_bmp);
  This command draws an image at the given location, the image being img_bmp.

- *void GFXWriteIntXY(UINT8 x,UINT8 y,INT16 val,INT8 field _length,UINT8 color)*
  This function Draws a number (integer) at given x,y location. "val" is the value of integer number to draw, color can either be GFX _COLOR _BLACK or GFX _COLOR _WHITE. Eg: GFXWriteIntXY (0 , 0 , 5 , GFX _COLOR _BLACK);
  This command writes integer 5 onto the screen in black color at the origin.

4.3 Microcontroller

AVR ATmega8 and ATmega32 are similar in all respects except for the flash memory size which is 8kB in ATmega8 and 32kB in ATmega32 (hence the names). So we will discuss ATmega32 in detail. Its features, interface and hardware connection details are as given below.

4.3.1 Overview

The ATmega32 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega32 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

4.3.2 Features

Tables 4.5-4.10 give all the features of the AVR controllers

Table 4.5: Memory features

Flash Memory	32 kB
EEPROM Data Memory	1024 Bytes
SRAM Data Memory	2048 Bytes
General Purpose Registers (Accumulators)	32

Table 4.6: Timers / Counters

Timer/Counters (8-bit)	2
Watchdog Timer with On-chip Oscillator	Yes
Real Time Counter	Yes
Timer/Counters (16-bit)	1
Pulse Width Modulator	4 channels

Table 4.7: MCU Secific features

Clock Frequency	0 - 16 MHz
Supply Voltage	4.5 - 5.5 V
Sleep Modes	6
Hardware Multiplier	Yes
I/O Pins	32
On Chip Oscillator	Yes
Interrupts	19
Interrupts, External pins	3
Brown-out Detection	Yes
Power-on Reset	Yes
Fully Static Operation	Yes
On-Chip Debug support via JTAG port	Yes
IEEE 1149.1 (JTAG) Boundary Scan	Yes

Table 4.8: Analog I/O

Analog Comparator	Yes
Analog-to-Digital Converter (10-bit)	8 channels
Analog Gain Stage	2 channels

Table 4.9: Serial I/O

Full Duplex Serial Peripheral Interface (SPI)	Yes
2-wire Serial Interface	Yes
Full Duplex USART	1

Table 4.10: Programming Modes

In-System Programming via SPI Port	Yes
High Voltage Parallel Programming (12V)	Yes
Self-Programming via on-chip Boot Program	Yes
In-System Programming via JTAG port	Yes

4.3.3 Interface

In this section we will see the specific information about each of the features listed above and how to use them with the programming software AVR Studio.

4.3.3.1 I/O Ports

ATmega32 has four 8-bit bidirectional I/O ports viz. PORT A to PORT D. All AVR ports are bit addressable. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both VCC and Ground as indicated in Figure 4.12.

All registers and bit references in this section are written in general form. A lower case 'x' represents the numbering letter for the port, and a lower case 'n' represents the bit number. However, when using the register or bit defines in a program, the precise form must be used i.e., PORTB.3 for bit no.3 in Port B, here documented generally as PORTx.n. Three I/O memory address locations are allocated for each port, one each for the Data Register - PORTx, Data Direction Register - DDRx, and the Port Input Register- PINx. The individual pins of these ports can be accessed as PORTx.n, DDRx.n and PINx.n respectively. The Port Input Register I/O location is read only, while the Data Register and the Data Direction Register are read/write.

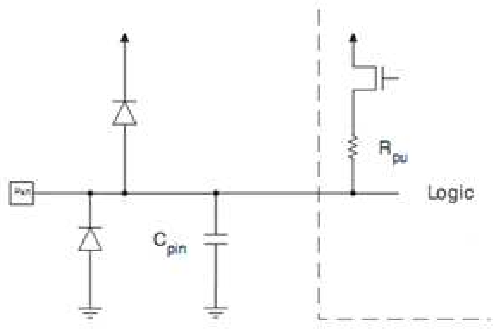


Figure 4.12: I/O Pin Equivalent Schematic

The DDRx.n bit in the DDRx Register selects the direction of this pin. If DDRx.n is written logic one, Px.n pin is configured as an output pin. If DDRx.n is written logic zero, Px.n is configured as an input pin. If the pin is configured as an input pin and PORTx.n is written logic one, the pull-up resistor is activated. To switch the pull-up resistor off, PORTx.n has to be written logic zero. The port pins

are tri-stated when a reset condition becomes active, even if no clocks are running. The logical value at the input pin is reflected in the PINx.n register bit from where it can be read.
C code example:
unsigned char i;
. . .
DDRA = 0x00; // Port A as input
i = PINA; // read input at port A into variable 'input'

If the pin is configured as an output pin, logic one or zero can be outputted from it by writing the corresponding value to PORTx.n register.
C code example:
unsigned char i=0xA0;
. . .
DDRA = 0xFF; // Port A as input
PORTA = i; // output value of 'i' to port A

Table 4.11 summarizes all the above information.

Table 4.11: Port Pin Configurations

DDRx.n	PORTx.n	I/O	Pull-up	Comment
0	0	Input	No	Tristate (Hi-Z)
0	1	Input	Yes	Px.n will source current if ext. pulled low
1	0	Output	No	Output Low (sink)
1	1	Output	No	Output High (source)

Alternate Port Functions
In addition to general digital input/output the port pins of AVR have alternate functions. The detailed information of how to select these functions and their uses is explained in the appendix of ATmega8/32.

4.3.3.2 Analog to Digital Converter (ADC)

Some of the many features of the on-chip ADC are:

- 10-bit Resolution

- 0.5 LSB Integral Non-linearity
- ± 2 LSB Absolute Accuracy
- 13 - 260 μs Conversion Time
- Up to 15 kSPS at Maximum Resolution
- 8 Multiplexed Single Ended Input Channels
- Optional Left adjustment for ADC Result Readout
- 0 - VCC ADC Input Voltage Range
- Selectable 2.56V ADC Reference Voltage
- Interrupt on ADC Conversion Complete

The ATmega8 features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows 8 single-ended voltage inputs constructed from the pins of Port A. The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. The ADC has a separate analog supply voltage pin, AVCC. AVCC must not differ more than ± 0.3 V from VCC. The reference voltage of ADC is also available as an external pin AREF. It is the maximum voltage that the ADC can convert to digital. So it is normally connected to VCC.

The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents GND and the maximum value represents the voltage on the AREF pin minus 1 LSB. The analog input channel and differential gain are selected by writing to the MUX bits in ADMUX. The ADC is enabled by setting the ADC Enable bit, ADEN in ADCSRA. The ADC generates a 10-bit result which is presented in the ADC Data Registers, ADCH and ADCL. By default, the result is presented right adjusted, but can optionally be presented left adjusted by setting the ADLAR bit in ADMUX. If the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. The ADC has its own interrupt which can be triggered when a conversion completes. A single conversion is started by writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be cleared by hardware when the conversion is completed. By default, the successive approximation circuitry requires an input clock frequency between 50 kHz and 200 kHz to get maximum resolution. The ADC module contains a prescaler, which generates an acceptable ADC clock frequency from any CPU frequency above 100 kHz. The prescaling is set by the ADPS bits in ADCSRA.

After the conversion is complete (ADIF is high), the conversion result can be found in the ADC Result Registers (ADCL, ADCH). The result is
ADC = VIN ∗ 1024 / VREF, where VIN is the voltage on the selected input pin and VREF the selected voltage reference.

AVR Studio helps a lot while configuring the above settings. The software allows one to set the ADC parameters. If ADC interrupt is not used, the ADC value of an analog signal connected on port pin A.n can be read by using a simple function call,
a = read _adc(n);
This will start the ADC, convert the analog value into digital and return the digital value. It will then be stored into variable 'a'. This simplifies the code a lot.

4.3.3.3 UART

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device. It supports synchronous data transfer also, but it was not used as the communication in our case was wireless. The main features of UART are:

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty, and RX Complete
- Double Speed Asynchronous Communication Mode

In the project, 8 bit UART with 9600 baud, no parity, 1 start and 1 stop bit was used. AVR Studio provides libraries for data transmission and reception. If one wants to transmit just use the printf() function as is done in normal 'C' language. If one wants to receive data use scanf() or getchar() functions. Some sample code from the final code using UART is as shown below:

```
printf("b");
for(i=0; i<100; i++)
    printf("s % 04d, % 03d",data1[i],data[i] * 3);
```

4.4 Motor Driver

L293D is a quadruple push-pull 4 channel driver capable of delivering 600mA per channel. The L293D is a high voltage, high current four channel driver designed to accept standard TTL logic levels and drive inductive loads (such as relays solenoids, DC and stepping motors) and switching power transistors. The L293D is suitable for use in switching applications at frequencies up to 5 KHz.

4.4.1 Features

- 600 mA Output Current Capability Per Driver
- Pulsed Current 1.2 A / Driver
- Wide Supply Voltage Range: 4.5 V to 36 V
- Separate Input-Logic Supply
- Has a 16 plastic package which has 4 center pins connected together and used for Heat Sinking
- Over temperature protection
- Internal clamp diodes for inductive Transient Suppression
- Peak output current 2 A per channel
- High-Noise-Immunity Inputs

4.4.2 Pin Description

In the pinout diagram as shown in the appendix, the left and right sides denote the two outputs. The OUTPUT1 / OUTPUT2 pair forms one output and OUTPUT3 / OUTPUT4 forms another pair. Current can flow through these pairs as dictated by the INPUT1 / INPUT2 and INPUT3 / INPUT4 pairs.

- **Vss** : is the logical voltage supply for a 1. For example, if you connect it to a 5V supply, 5 volts into any of the INPUTs would mean a logical 1. However, if you connect it to a 36V supply, the same 5 volts into any INPUT would mean a logical 0. Here, consider Vss/2 as the "threshold" for a logical 1. If a voltage is above Vss/2, then its a 1 otherwise its a 0. So for the 36V case, if any INPUT is given a voltage greater than 18V, only then will it be considered a logical 1.
- **Vs** : is the actual voltage that needs to be output. This has nothing to do with the logical 0s and 1s.
- **GND** : represents grounds. These are needed for the multiple solid state switches that are burned into the IC.
- **ENABLE pins** : enable/disable the corresponding sides. Putting a logical 1 into ENABLE1 would enable INPUT1/INPUT2 and OUTPUT1/OUTPUT2. Similarly, ENABLE2 would enable the other two input and output pins. A logical 0 disables the corresponding side.

Chapter 5

INTEGRATED ASSEMBLY

Using the hardware and software discussed in the previous chapters the final assembly of the Surveillance Vehicle was done. Initially keeping in mind all the basic electrical design rules and following the datasheets of the various components used in the project a basic circuit diagram was derived from the block diagram discussed in Chapter 1.

5.1 Circuit and Software Description

5.1.1 Schematic representing control of GLCD by Atmega32 microcontroller

Hardware: The basic diagram for the control of GLCD by Atmega32 microcontroller can be shown as in Figure5.1. Atmel's ATMega32 microcontroller working on an external crystal frequency of 11.0592MHz is an integral part of the control circuit that handles only data read and write functions. The data bus of the Graphical LCD is connected to the PORTC of the microcontroller as shown. The control lines are connected to the PORTD. Since the Graphical LCD is based on KS0108 controller, the screen is essentially divided into two halves that are controlled by two separate KS0108 controllers. Thus, two chip select pins associated with the screen selection have to be controlled by the microcontroller. The chip select pins, CS1 and CS2 are connected to PB0 and PB1 to enable and disable them as and when needed.

Contrast control is taken care of by the LED+ and LED- pins. This is achieved by the use of a 10K ohm potentiometer that can vary the contrast of the GLCD. The basic purpose of Atmega32 is to update the screen as and when the user makes a selection on the menu through a touch screen. It receives 3-bit data from Atmega8 on PORTA0, PORTA1 AND PORTA2. Based on the combinations received, it updates the screen to display the appropriate menu.

Software: Since this circuit essentially forms an embedded system it cannot sleep, hence the software keeps running when the power is ON. When the system is switched on the GLCD functions are invoked to present a graphical GUI to the user. The program waits for the Atmega8 controller to change the data on PORTA pins. Initially, since the device is password protected, it waits for the right combination to be sent by Atmega8. Until then the display holds the password screen and does not update the screen.

5.1.2 Schematic of Atmega8 controller

Hardware: The schematic representing Atmega8 controller connected to the touch screen and the interfacing circuit is as shown in Figure 5.2. The Atmega8 controller is the heart of the device since all the major functions are handled by it. The schematic shows the connections from the touch screen to the controller. The ADC port is used for this purpose and the reasons for it were elaborated in the previous chapters. PortC is the ADC port in Atmega8 and PortC0/ADC0, PortC1/ADC1, PortC2/ADC2 and PortC3/ADC3 are connected to the x+, x-, y+ and y- pins of the touch screen respectively. Touch screen senses the touch input by the user which is fed to the ADC pins of Atmega8.

The Atmega8 controller also provides to the Atmega32 contoller, 3-bit data to inform Atmega32 about the changes in touch input. It also assumes the responsibility of interacting with the interfacing circuit, that operates the devices to be handled. PortB3 of Atmega8 is dedicated for verifying whether the touch has been registered or if the user has to touch the preferred selection again. This is achieved by connecting an LED to the pin which glows when the touch is registered. Atmega8 consists of 3 pins, PB0, PB1 and PB2 which are dedicated to PWM. Here the PB2 is utilized for operating the fan. Also pin PB5 is used as enable to operate the relay for the bulb.

Software: This circuit, as mentioned, is the most integral part of the device and hence cannot sleep. Atmega8 is run by code written on Arduino platform and essentially operates to sense the touch. The software also manipulates the interfaced devices as and when required by the user. It continuously polls the ADC pins and waits for the user to touch the screen. It monitors user's selection and accordingly sends a combination of bits to Atmega32 in order for it to update the screen. The software is capable of enabling the Enable pins of L293D which forms the interfacing circuit, the working of which is described in the next section. The software provides values between 0 to 255 to the PWM pin in order to alter the fan speed as required.

5.1.3 L293D DC motor driver

Hardware: The pin diagram of L293D DC motor driver is as shown in Figure 5.3. One section of the driver is made use of to operate the fan while the other is coupled to a relay to operate the bulb. Since the fan is rotated in only one direction, the two inputs at the fan section are connected to 5V and ground always. The two outputs available at this section are connected to the fan. The enable pin is connected to Atmega8's pin PB2 which is a dedicated PWM pin. It receives values between 0 and 255 (in this case 90, 180 and 255) to obtain three different fan speeds. The other section's ouptut pins are connected to the relay and this section is enabled by the pin Enable 2 which is controlled by Atmega8. The inputs to this section are connected to 5V and Gnd to provide a steady input always. It generates 12V at the four outputs that is used to drive the fan and relay.

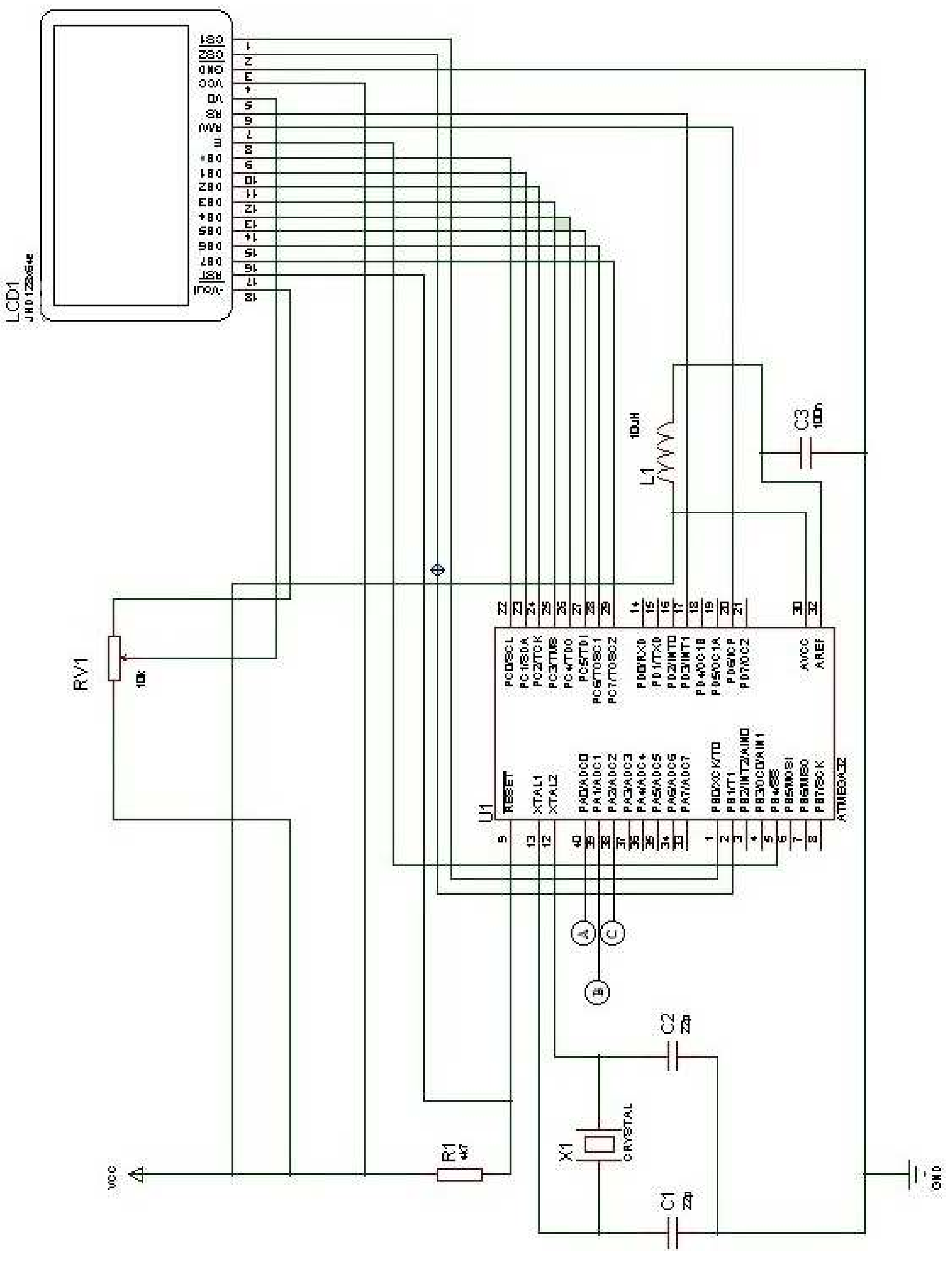


Figure 5.1: Schematic of Interfacing of GLCD with Atmega32

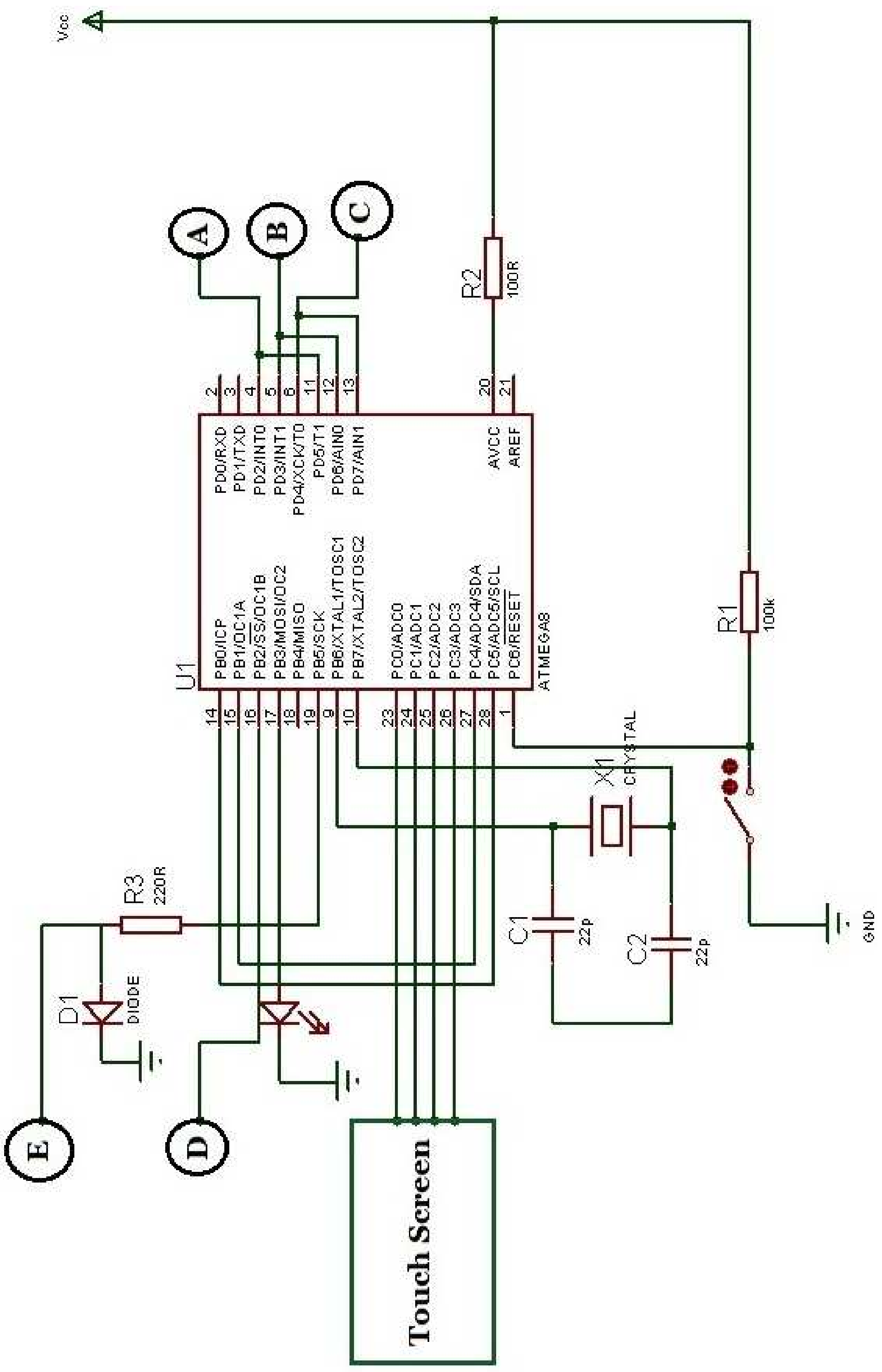


Figure 5.2: Schematic of Atmega8

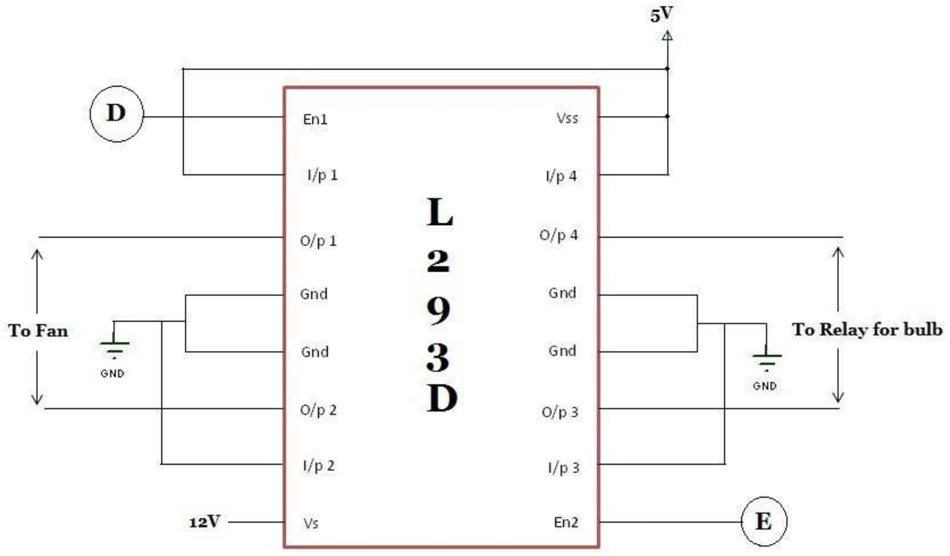


Figure 5.3: L293D DC motor driver

Chapter 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The primary aim of this project was to provide to the market, a less expensive and efficient controller that can replace the conventional use of switch boards. This is sought to be accomplished by interfacing a touch screen sensor and a glcd module to a microcontroller. Touch systems represent a rapidly growing subset of the display market, because of its flexible,user-friendly and mobile nature.

Moving towards the future, consumers will continue to see the growth of the touch screen industry, due to extensive engineering advancements in user interfaces. The ability to physically touch a screen is easier than searching for a specific key in a sea of buttons. Society, for these reasons, has found touch screens to be the future of many devices. The social norm of today includes walking down the street surfing the web on an iPhone or sifting through music on an iPod Touch. No additional buttons are necessary, just the small, portable device in ones pocket until needed. Society will continue to see the development of touch screen technology as human-device interaction is perfected. Not only did the project involve learning the commercial touch technology, but it also helped us get a clear understanding on so many vital concepts which otherwise remain hidden.

6.2 Future Scope

Since automation has encapsulated our lives tremendously, this project has a marked scope for development in the market. In this project, certain relatively smaller appliances such as a fan and a light bulb will be interfaced to exemplify the working of the device due to several practical limitations. However, it can be broadened to include various other devices that are monotonously operated. This could comprise of the geyser, closing and opening of the doors, etc. Thus, this project could be modelled as a subset of home automation which consists of security options as well.

Touch screen sensors, being an ever-evolving technology, can be upgraded to better touch screen sensors with increased durability and reliability. Also various graphical LCDs with higher resolutions can be used to achieve better graphics. This would mean an increase in the ease of use and operability of the device.

An additional modification can be added to this device that has revolutionized the use of touch screen controller. By adding the feature of remote operation or in other words, by making it a remote touch screen controller, the application of this device can penetrate into various other fields as well. For eg, a remote touch screen controller can be used as a universal remote that can not only operate household appliances such as geyser, lights, but can also be used to specify various other operations such as operation of a TV, radio, etc. Simple transmitters and receivers can be interfaced to make the setup a wireless type communication or the scope can be widened to include GSM techniques.

Appendix A

A.1 ATmega32

A.1.1 Pin Configuration

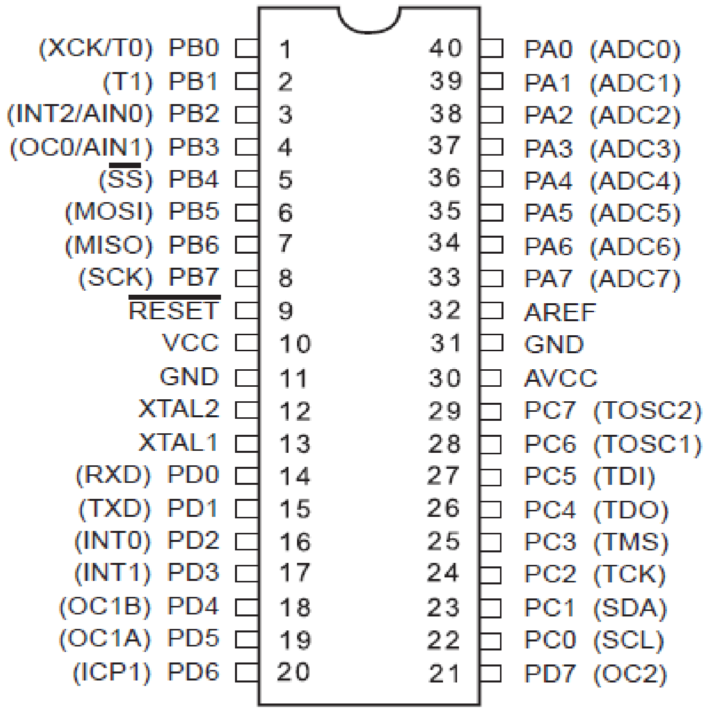


Figure A.1: Pinout ATmega32 - PDIP

A.1.2 Pin Description and Overview

- ***VCC*** Digital supply voltage
- ***GND*** Ground
- ***Port A (PA7...PA0)***
  Port A serves as the analog inputs to the A/D Converter. It 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. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. The TD0 pin is tri-stated unless TAP states that shift out data are entered.
- ***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 char-

acteristics 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. The minimum pulse length is given in the following graphs. Shorter pulses are not guaranteed to generate a reset.

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

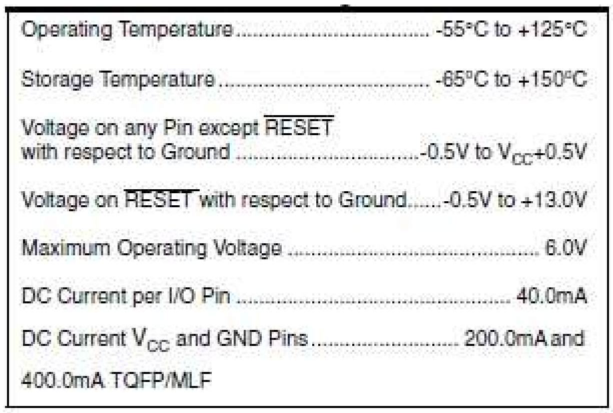
- ***XTAL2*** Output from the inverting Oscillator amplifier.

- ***AVCC*** It 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*** Analog reference pin for the A/D Converter.

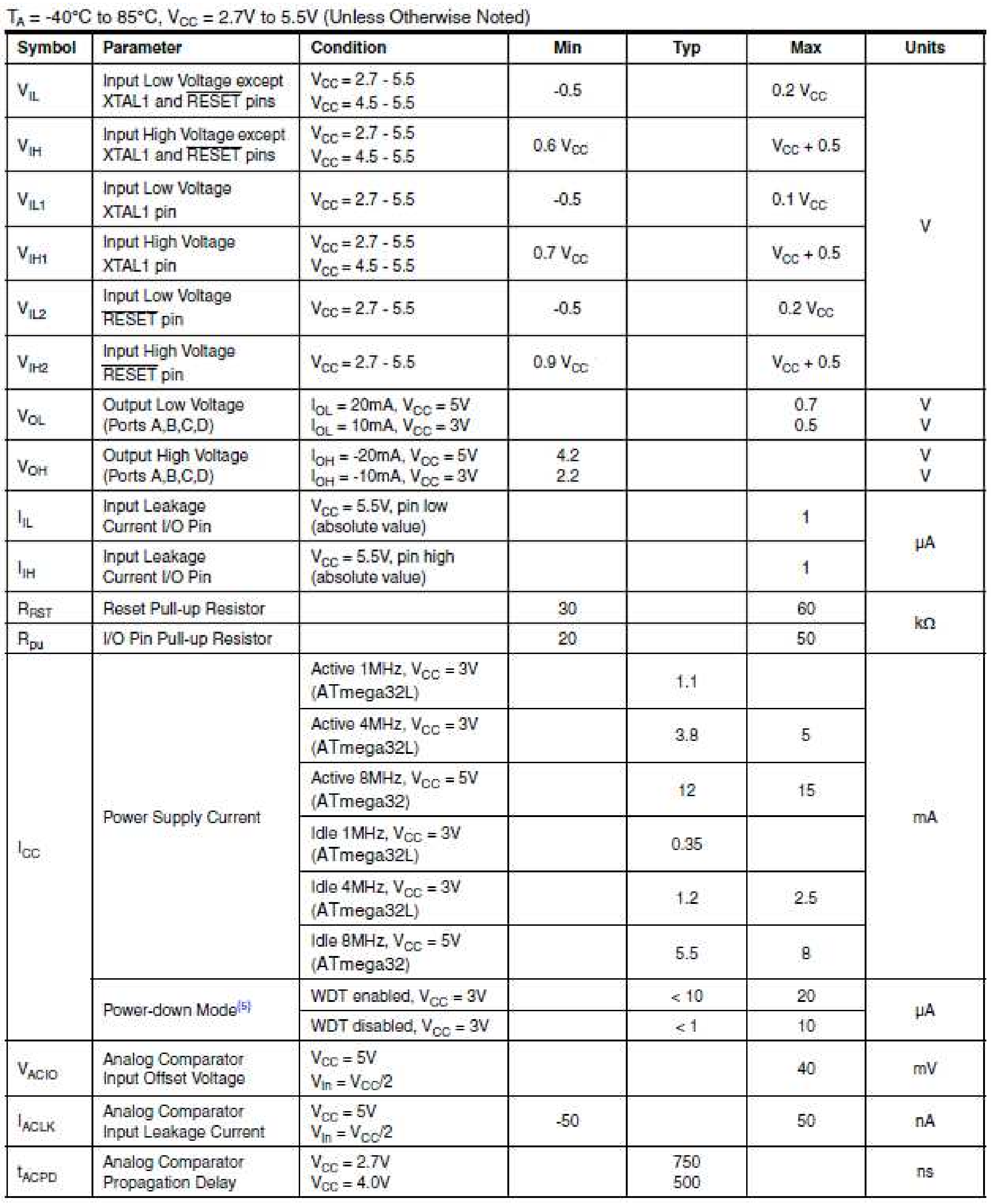
A.1.3 Electrical Characteristics

Absolute Maximum Ratings



Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0mA
DC Current V_{CC} and GND Pins	200.0mA and 400.0mA TQFP/MLF

DC Characteristics The following charts show typical behavior. These figures are

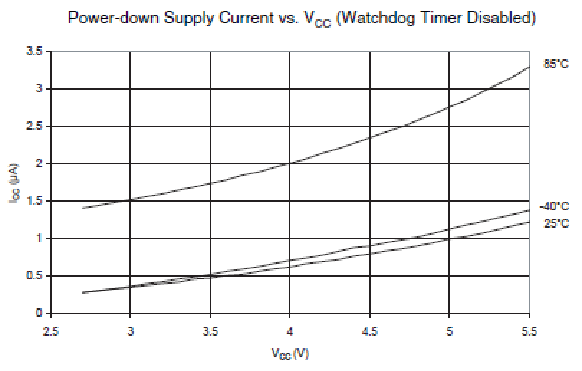


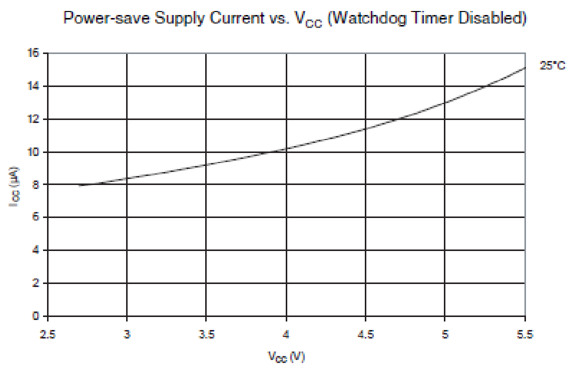
T_A = -40°C to 85°C, V_{CC} = 2.7V to 5.5V (Unless Otherwise Noted)

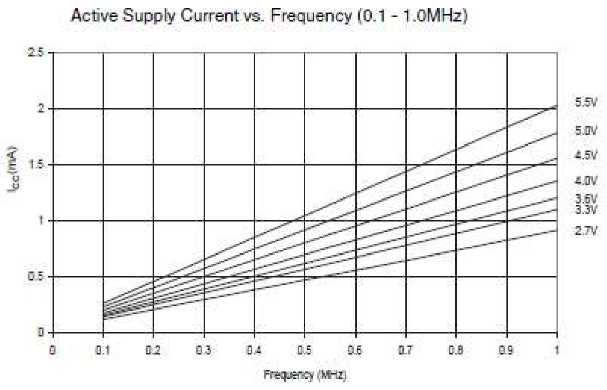
Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 2.7 - 5.5 V_{CC} = 4.5 - 5.5	-0.5		0.2 V_{CC}	V
V_{IH}	Input High Voltage except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 2.7 - 5.5 V_{CC} = 4.5 - 5.5	0.6 V_{CC}		V_{CC} + 0.5	
V_{IL1}	Input Low Voltage XTAL1 pin	V_{CC} = 2.7 - 5.5	-0.5		0.1 V_{CC}	
V_{IH1}	Input High Voltage XTAL1 pin	V_{CC} = 2.7 - 5.5 V_{CC} = 4.5 - 5.5	0.7 V_{CC}		V_{CC} + 0.5	
V_{IL2}	Input Low Voltage $\overline{\text{RESET}}$ pin	V_{CC} = 2.7 - 5.5	-0.5		0.2 V_{CC}	
V_{IH2}	Input High Voltage $\overline{\text{RESET}}$ pin	V_{CC} = 2.7 - 5.5	0.9 V_{CC}		V_{CC} + 0.5	
V_{OL}	Output Low Voltage (Ports A,B,C,D)	I_{OL} = 20mA, V_{CC} = 5V I_{OL} = 10mA, V_{CC} = 3V			0.7 0.5	V V
V_{OH}	Output High Voltage (Ports A,B,C,D)	I_{OH} = -20mA, V_{CC} = 5V I_{OH} = -10mA, V_{CC} = 3V	4.2 2.2			V V
I_{IL}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin low (absolute value)			1	µA
I_{IH}	Input Leakage Current I/O Pin	V_{CC} = 5.5V, pin high (absolute value)			1	
R_{RST}	Reset Pull-up Resistor		30		60	kΩ
R_{pu}	I/O Pin Pull-up Resistor		20		50	
I_{CC}	Power Supply Current	Active 1MHz, V_{CC} = 3V (ATmega32L)		1.1		mA
		Active 4MHz, V_{CC} = 3V (ATmega32L)		3.8	5	
		Active 8MHz, V_{CC} = 5V (ATmega32)		12	15	
		Idle 1MHz, V_{CC} = 3V (ATmega32L)		0.35		
		Idle 4MHz, V_{CC} = 3V (ATmega32L)		1.2	2.5	
		Idle 8MHz, V_{CC} = 5V (ATmega32)		5.5	8	
	Power-down Mode[5]	WDT enabled, V_{CC} = 3V		< 10	20	µA
		WDT disabled, V_{CC} = 3V		< 1	10	
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V V_{in} = V_{CC}/2			40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACPD}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 4.0V		750 500		ns

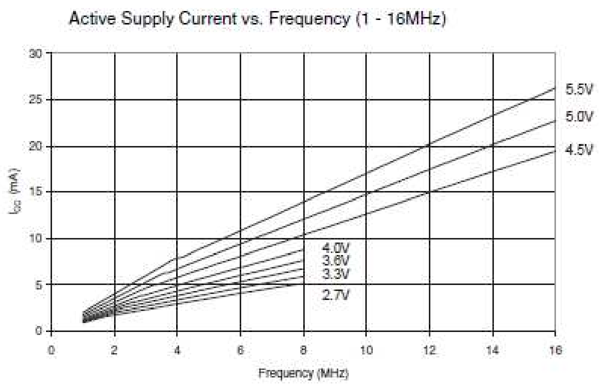
not tested during manufacturing. All current consumption measurements are performed with all I/O pins configured as inputs and with internal pull-ups enabled. A square wave generator with rail-to-rail output is used as clock source. The power consumption in Power-down mode is independent of clock selection. The current consumption is a function of several factors such as: operating voltage, operating frequency, loading of I/O pins, switching rate of I/O pins, code executed and ambient temperature. The dominating factors are operating voltage and frequency. The current drawn from capacitive loaded pins may be estimated (for one pin) as CL*VCC*f where CL = load capacitance, VCC = operating voltage and f = average switching frequency of I/O pin. The parts are characterized at frequencies higher than test limits. Parts are not guaranteed to function properly at frequencies higher than the ordering code indicates. The difference between current consumption in Power-down mode with Watchdog Timer enabled and Power-down mode with Watchdog Timer disabled represents the differential current drawn by the Watchdog Timer.

A.1.4 Typical Characteristics

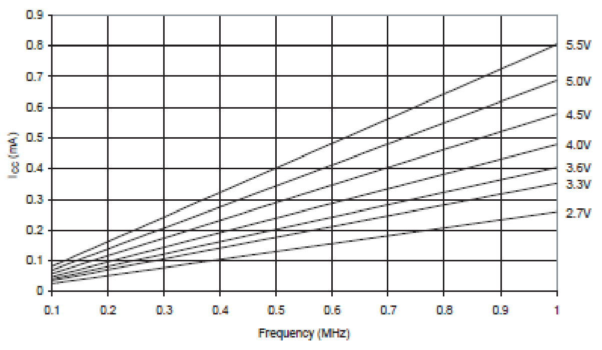




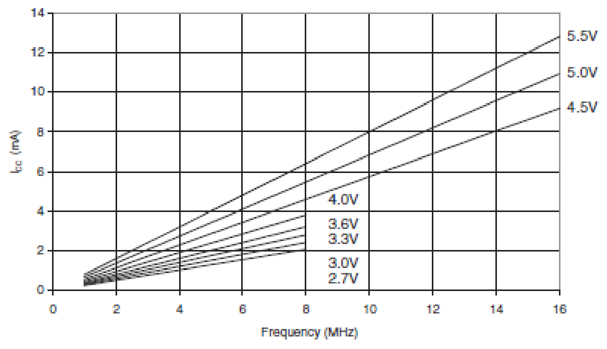




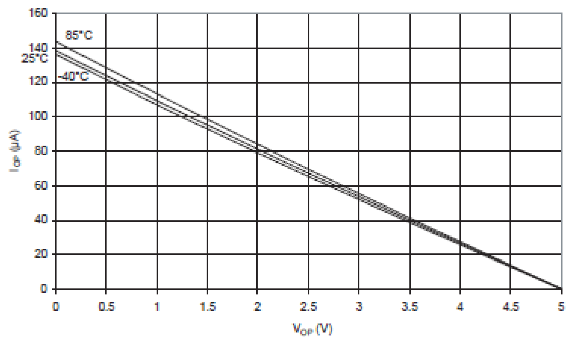
Idle Supply Current vs. Frequency (0.1 - 1.0MHz)



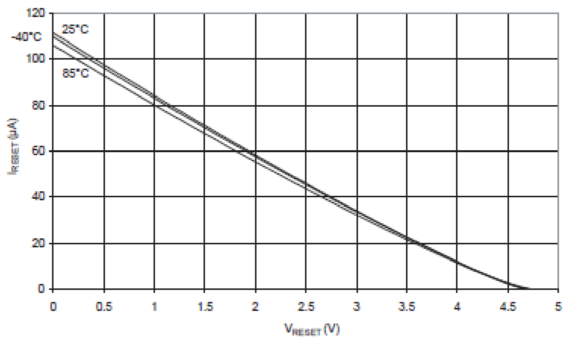
Idle Supply Current vs. Frequency (1 - 16MHz)



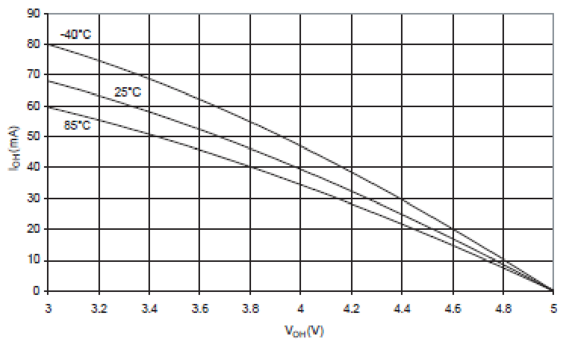
I/O Pin Pull-up Resistor Current vs. Input Voltage (V_{CC} = 5V)



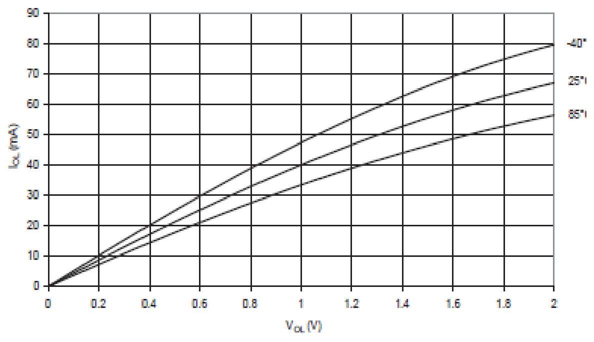
Reset Pull-up Resistor Current vs. Reset Pin Voltage (V_{CC} = 5V)



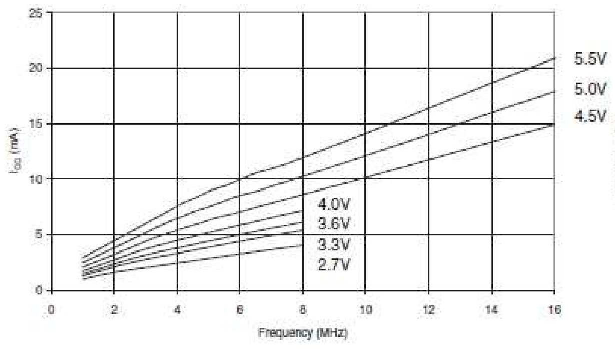
I/O Pin Source Current vs. Output Voltage (V_{CC} = 5V)



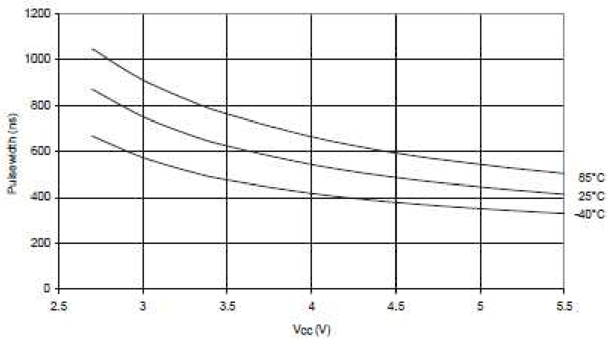
I/O Pin Sink Current vs. Output Voltage (V_{CC} = 5V)



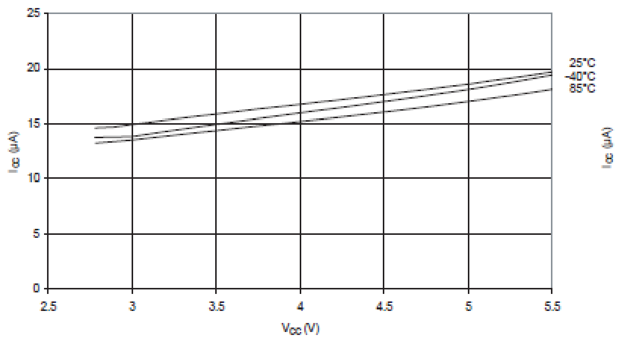
Reset Supply Current vs. V_{CC} (1 - 16MHz)



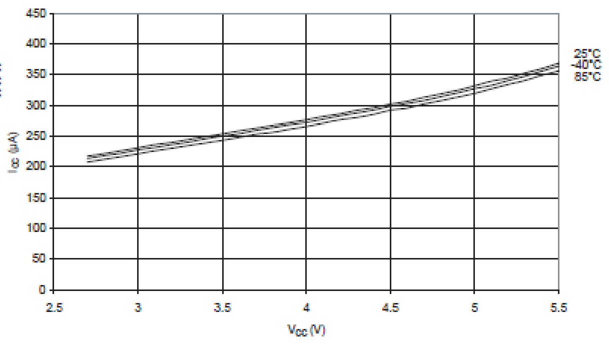
Minimum Reset Pulse Width vs. V_{CC}



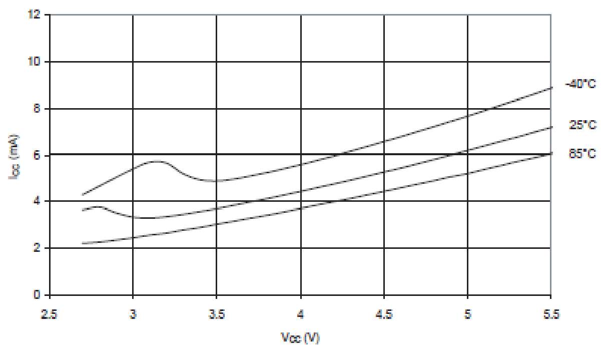
Brownout Detector Current vs. V_{CC}



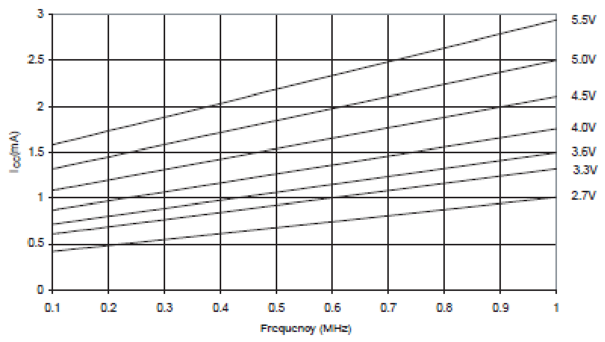
ADC Current vs. V_{CC} (AREF = AVCC)



Programming Current vs. V_{CC}

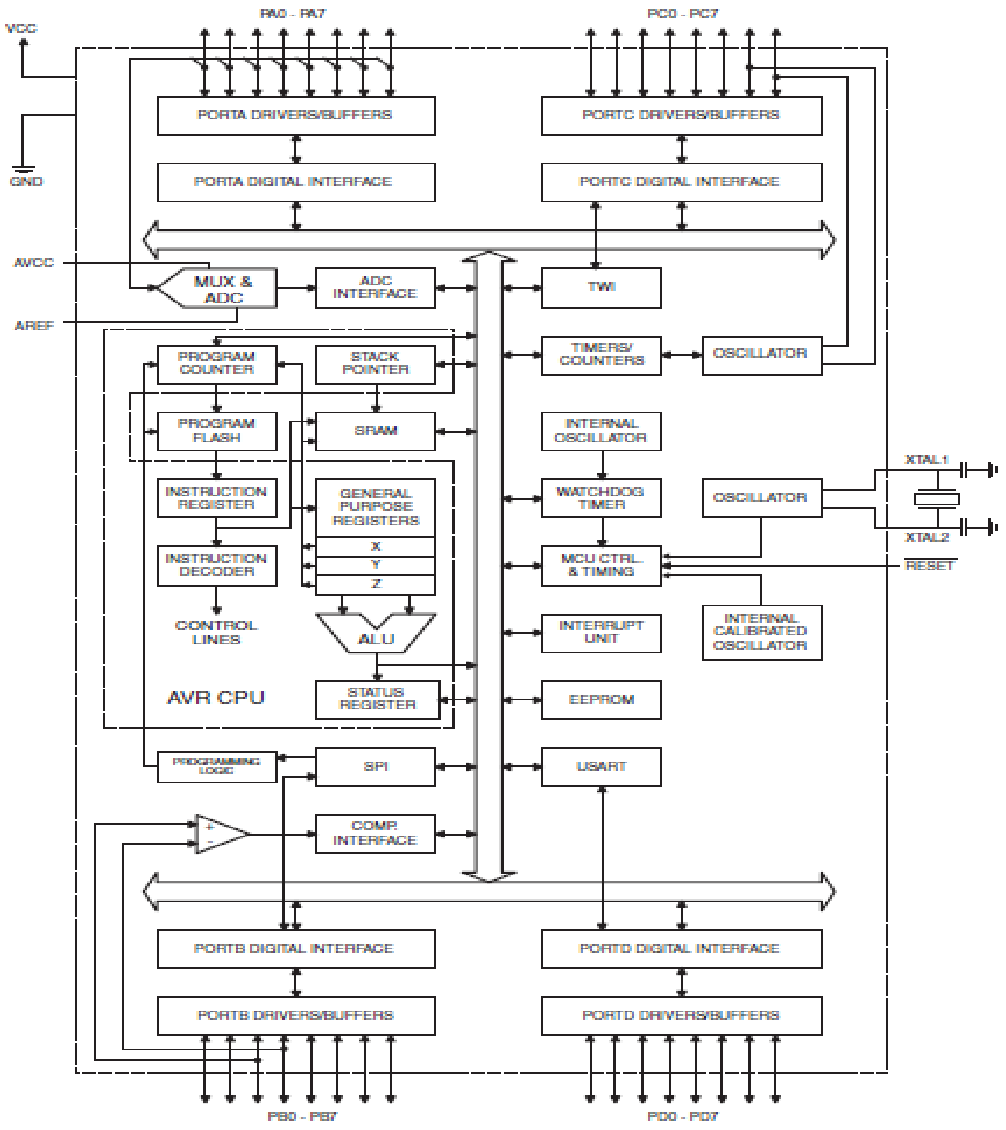


re 206. Reset Supply Current vs. V_{CC} (0.1 - 1.0MH)



A.1.5 Block Diagram

Figure A.2: Block Diagram of Atmega32



A.2 ATmega8

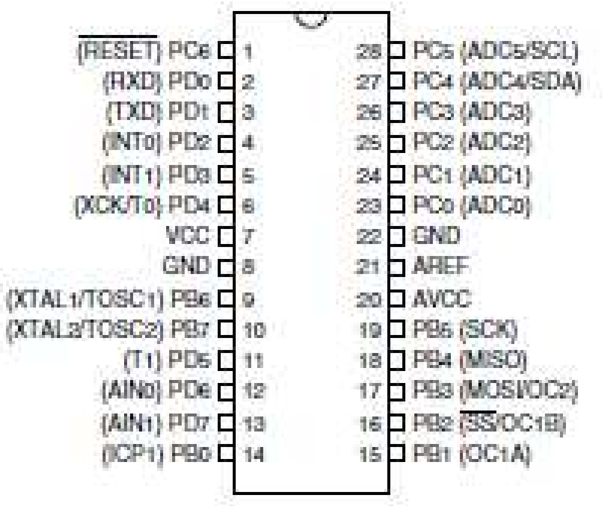


Figure A.3: Pinout ATmega8 - PDIP

A.2.1 Pin Description and Overview

- ***VCC*** Digital supply voltage.
- ***GND*** Ground.
- ***Port B (PB7...PB0)XTAL1/XTAL2/TOSC1/TOSC2***

  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.

  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 from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7. . . 6 is used as TOSC2. . . 1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

- ***Port C (PC5. . . PC0)***
  Port C is an 7-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.

  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 the other pins of Port C. If the RSTDISBL Fuse is unprogrammed, PC6 is used as a 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. The minimum pulse length is given in Table ....on page.... Shorter pulses are not guaranteed to generate a Reset.

- ***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.

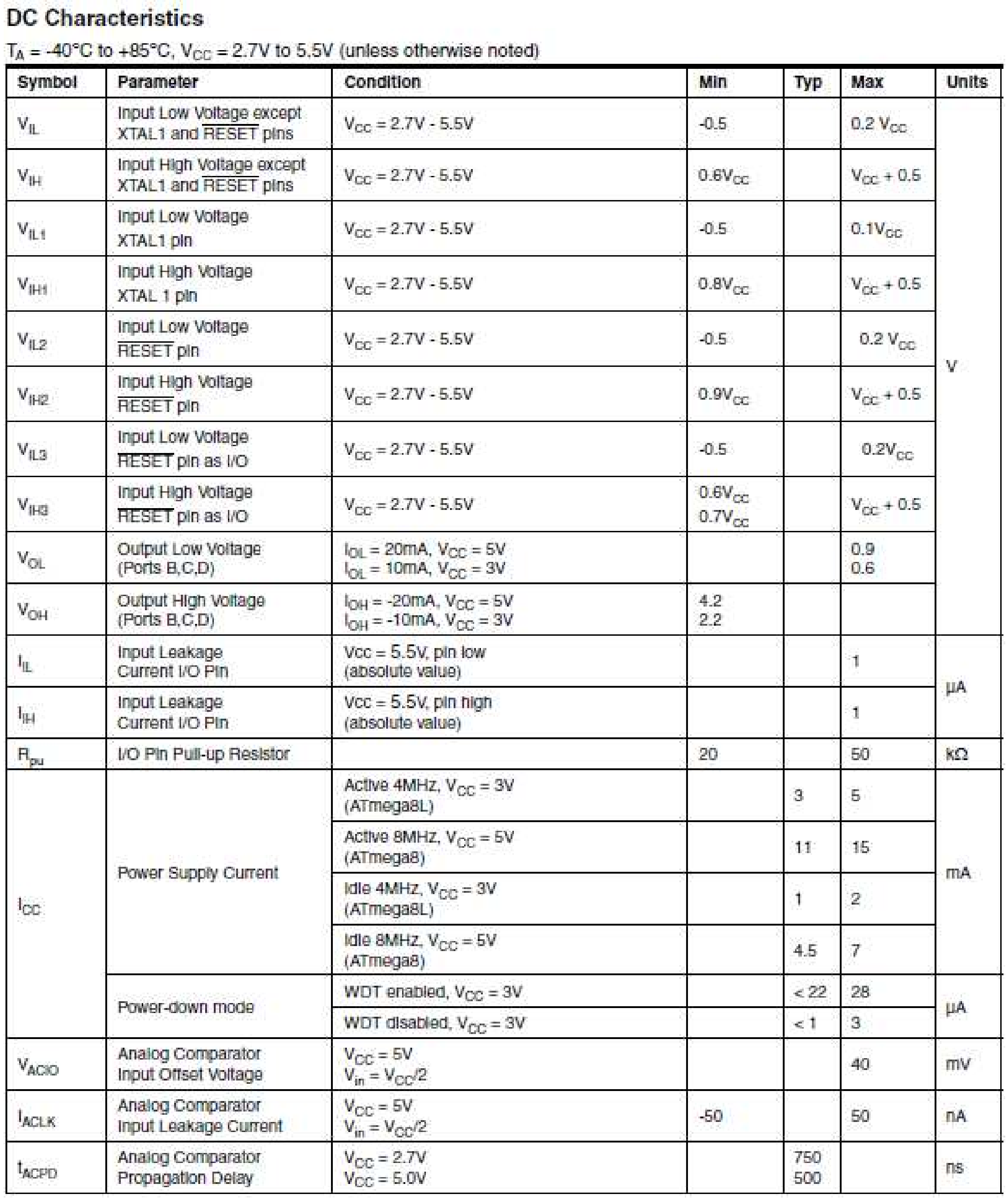
- ***RESET***
  Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in following sections. Shorter pulses are not guaranteed to generate a reset.

- ***AVCC***
  It is the supply voltage pin for the A/D Converter, Port C (3. . . 0), and ADC (7. . . 6). 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 Port C (5..4) use digital supply voltage, VCC.

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

A.2.2 Electrical Characteristics and Typical Characteristics



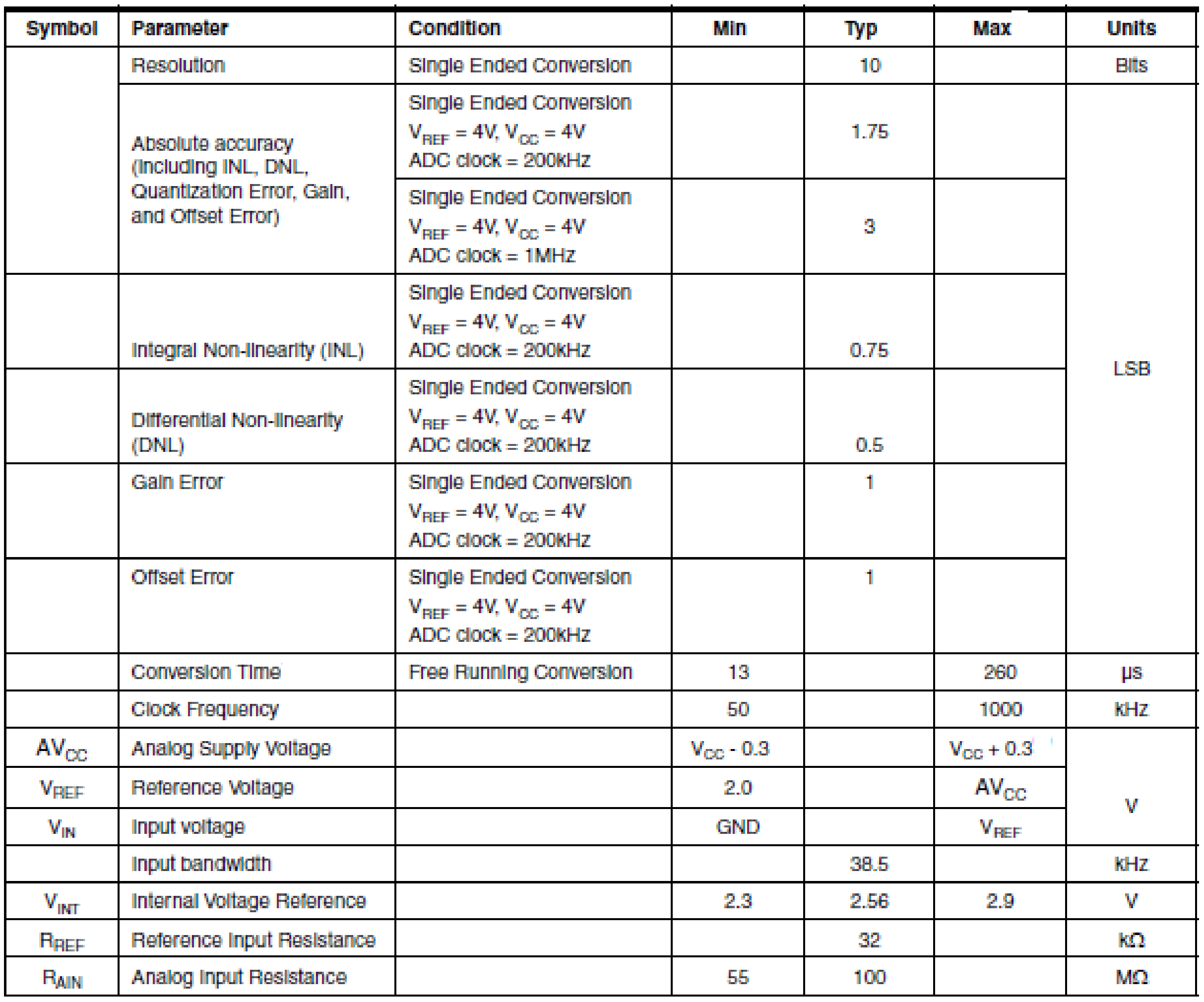
DC Characteristics

T_A = -40°C to +85°C, V_{CC} = 2.7V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{IL}	Input Low Voltage except XTAL1 and $\overline{RESET}$ pins	V_{CC} = 2.7V - 5.5V	-0.5		0.2 V_{CC}	V
V_{IH}	Input High Voltage except XTAL1 and $\overline{RESET}$ pins	V_{CC} = 2.7V - 5.5V	0.6V_{CC}		V_{CC} + 0.5	
V_{IL1}	Input Low Voltage XTAL1 pin	V_{CC} = 2.7V - 5.5V	-0.5		0.1V_{CC}	
V_{IH1}	Input High Voltage XTAL 1 pin	V_{CC} = 2.7V - 5.5V	0.8V_{CC}		V_{CC} + 0.5	
V_{IL2}	Input Low Voltage $\overline{RESET}$ pin	V_{CC} = 2.7V - 5.5V	-0.5		0.2 V_{CC}	
V_{IH2}	Input High Voltage $\overline{RESET}$ pin	V_{CC} = 2.7V - 5.5V	0.9V_{CC}		V_{CC} + 0.5	
V_{IL3}	Input Low Voltage $\overline{RESET}$ pin as I/O	V_{CC} = 2.7V - 5.5V	-0.5		0.2V_{CC}	
V_{IH3}	Input High Voltage $\overline{RESET}$ pin as I/O	V_{CC} = 2.7V - 5.5V	0.6V_{CC} 0.7V_{CC}		V_{CC} + 0.5	
V_{OL}	Output Low Voltage (Ports B,C,D)	I_{OL} = 20mA, V_{CC} = 5V I_{OL} = 10mA, V_{CC} = 3V			0.9 0.6	
V_{OH}	Output High Voltage (Ports B,C,D)	I_{OH} = -20mA, V_{CC} = 5V I_{OH} = -10mA, V_{CC} = 3V	4.2 2.2			
I_{IL}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin low (absolute value)			1	µA
I_{IH}	Input Leakage Current I/O Pin	Vcc = 5.5V, pin high (absolute value)			1	
R_{pu}	I/O Pin Pull-up Resistor		20		50	kΩ
I_{CC}	Power Supply Current	Active 4MHz, V_{CC} = 3V (ATmega8L)		3	5	mA
		Active 8MHz, V_{CC} = 5V (ATmega8)		11	15	
		Idle 4MHz, V_{CC} = 3V (ATmega8L)		1	2	
		Idle 8MHz, V_{CC} = 5V (ATmega8)		4.5	7	
	Power-down mode	WDT enabled, V_{CC} = 3V		< 22	28	µA
		WDT disabled, V_{CC} = 3V		< 1	3	
V_{ACIO}	Analog Comparator Input Offset Voltage	V_{CC} = 5V V_{in} = V_{CC}/2			40	mV
I_{ACLK}	Analog Comparator Input Leakage Current	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACPD}	Analog Comparator Propagation Delay	V_{CC} = 2.7V V_{CC} = 5.0V		750 500		ns

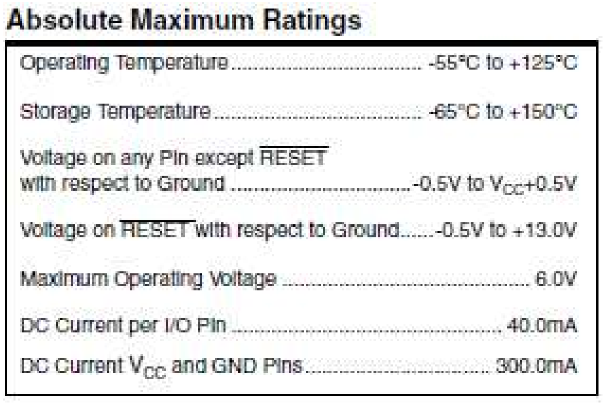
Notes

ADC Characteristics



Symbol	Parameter	Condition	Min	Typ	Max	Units
	Resolution	Single Ended Conversion		10		Bits
	Absolute accuracy (Including INL, DNL, Quantization Error, Gain, and Offset Error)	Single Ended Conversion V_{REF} = 4V, V_{CC} = 4V ADC clock = 200kHz		1.75		LSB
		Single Ended Conversion V_{REF} = 4V, V_{CC} = 4V ADC clock = 1MHz		3		
	Integral Non-linearity (INL)	Single Ended Conversion V_{REF} = 4V, V_{CC} = 4V ADC clock = 200kHz		0.75		
	Differential Non-linearity (DNL)	Single Ended Conversion V_{REF} = 4V, V_{CC} = 4V ADC clock = 200kHz		0.5		
	Gain Error	Single Ended Conversion V_{REF} = 4V, V_{CC} = 4V ADC clock = 200kHz		1		
	Offset Error	Single Ended Conversion V_{REF} = 4V, V_{CC} = 4V ADC clock = 200kHz		1		
	Conversion Time	Free Running Conversion	13		260	µs
	Clock Frequency		50		1000	kHz
AV_{CC}	Analog Supply Voltage		V_{CC} - 0.3		V_{CC} + 0.3	V
V_{REF}	Reference Voltage		2.0		AV_{CC}	
V_{IN}	Input voltage		GND		V_{REF}	
	Input bandwidth			38.5		kHz
V_{INT}	Internal Voltage Reference		2.3	2.56	2.9	V
R_{REF}	Reference Input Resistance			32		kΩ
R_{AIN}	Analog Input Resistance		55	100		MΩ

- *"Max" means the highest value where the pin is guaranteed to be read as low*
- *"Min" means the lowest value where the pin is guaranteed to be read as high*
- *Minimum VCC for Power-down is 2.5V*
- *Although each I/O port can sink more than the test conditions (20mA at Vcc = 5V, 10mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:*
  - The sum of all IOL, for all ports, should not exceed 300mA.
  - The sum of all IOL, for ports C0 - C5 should not exceed 100mA.
  - The sum of all IOL, for ports B0 - B7, C6, D0 - D7 and XTAL2, should not exceed 200mA.



Absolute Maximum Ratings

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0mA
DC Current V_{CC} and GND Pins	300.0mA

If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.

- *Although each I/O port can source more than the test conditions (20mA at Vcc = 5V, 10mA at Vcc = 3V) under steady state conditions (non-transient), the following must be observed:*
  - The sum of all IOH, for all ports, should not exceed 300mA.
  - The sum of all IOH, for port C0 - C5, should not exceed 100mA.
  - The sum of all IOH, for ports B0 - B7, C6, D0 - D7 and XTAL2, should not exceed 200mA.

If IOH exceeds the test condition, VOH may exceed the related specification. Pins are not guaranteed to source current greater than the listed test condition.

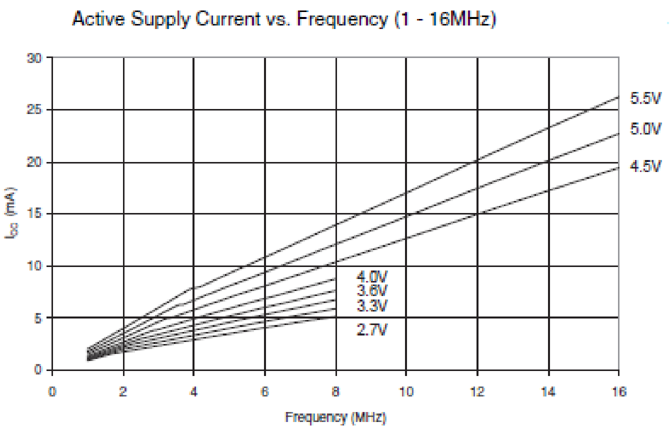
Sleep modes enable the application to shut down unused modules in the MCU, thereby saving power. The AVR provides various sleep modes allowing the user to tailor the power consumption to the applications requirements. To enter any of the five sleep modes, the SE bit in MCUCR must be written to logic one and a SLEEP instruction must be executed. The SM2, SM1, and SM0 bits in the MCUCR Register select which sleep mode (Idle, ADC Noise Reduction, Power-down, Power-save, or Standby) will be activated by the SLEEP instruction.

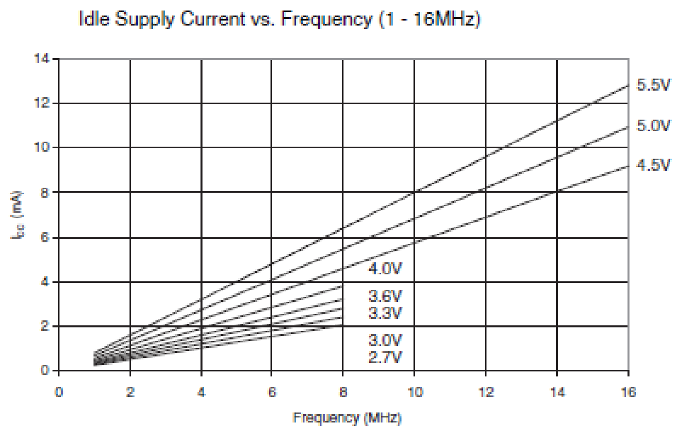
If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, it executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the Register File and SRAM are unaltered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

There are several issues to consider when trying to minimize the power consumption in an AVR controlled system. In general, sleep modes should be used as much as possible, and the sleep mode should be selected so that as few as possible of the devices functions are operating. All functions not needed should be disabled. In particular, the following modules may need special consideration when trying to achieve the lowest possible power consumption.

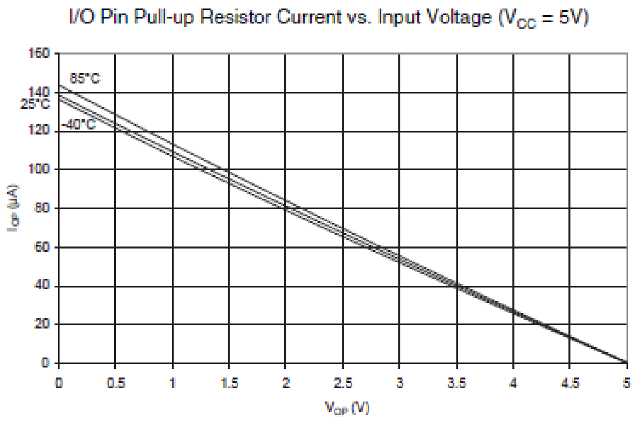
The following charts show typical behavior. These figures are not tested during manufacturing. All current consumption measurements are performed with all I/O pins configured as inputs and with internal pull-ups enabled. A sine wave generator with Rail-to-Rail output is used as clock source. The power consumption in Power-down mode is independent of clock selection. The current consumption is a function of several factors such as: operating voltage, operating frequency, loading of I/O pins, switching rate of I/O pins, code executed and ambient temperature. The dominating factors are operating voltage and frequency.

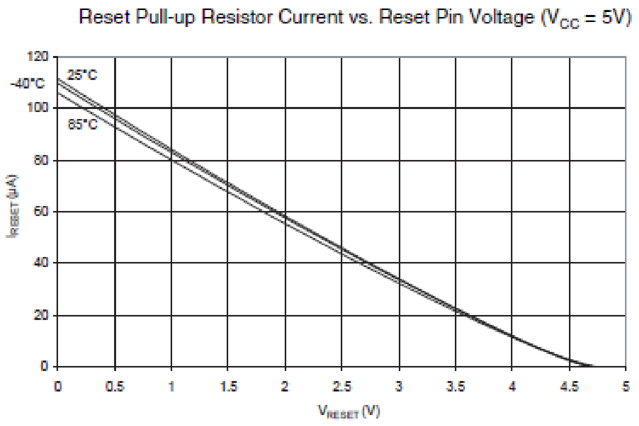
The current drawn from capacitive loaded pins may be estimated (for one pin) as: CL ∗ VCC ∗ f where CL = load capacitance, VCC = operating voltage and f = average switching frequency of I/O pin. The parts are characterized at frequencies higher than test limits. Parts are not guaranteed to function properly at frequencies higher than the ordering code indicates. The difference between current consumption in Power-down mode with Watchdog Timer enabled and Power-down mode with Watchdog Timer disabled represents the differential current drawn by the Watchdog Timer.

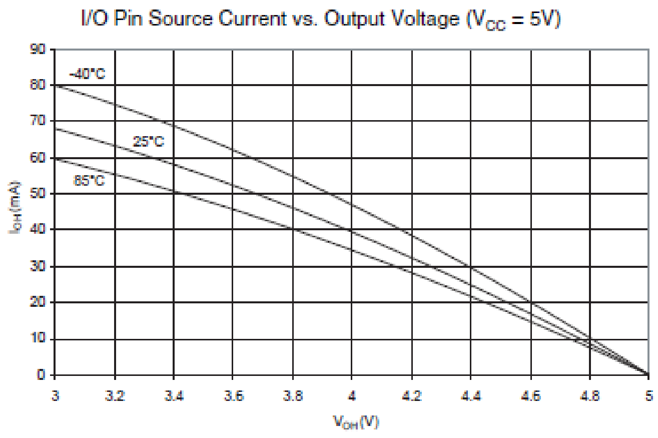


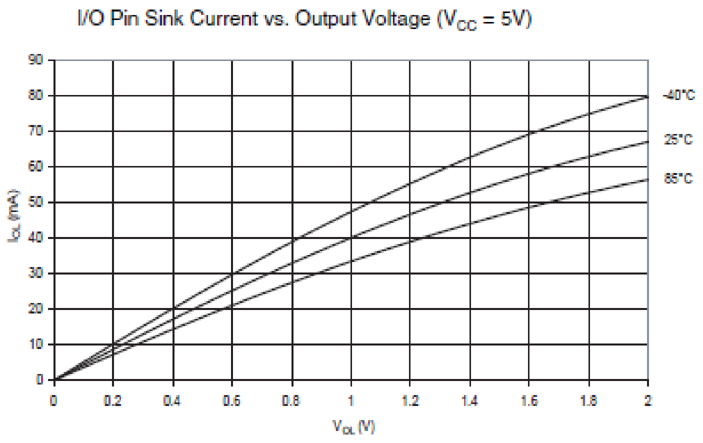


If enabled, the ADC will be enabled in all sleep modes. To save power, the ADC should be disabled before entering any sleep mode. When the ADC is turned off and on again, the next conversion will be an extended conversion. When entering

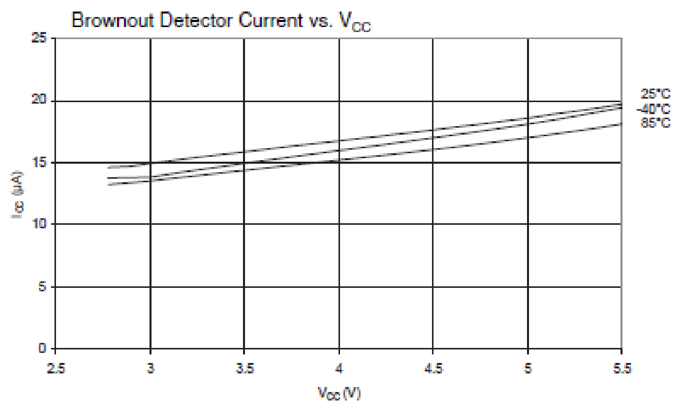


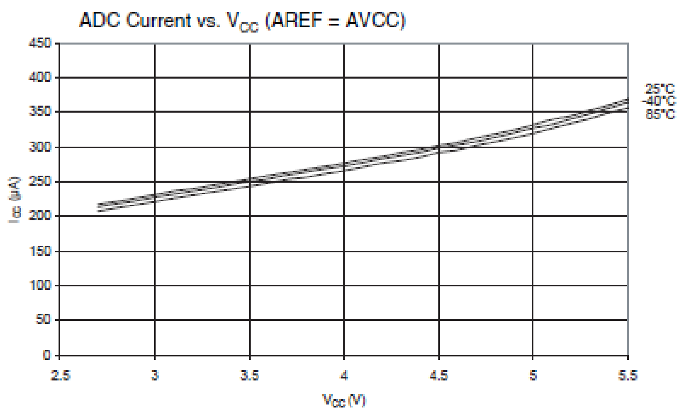


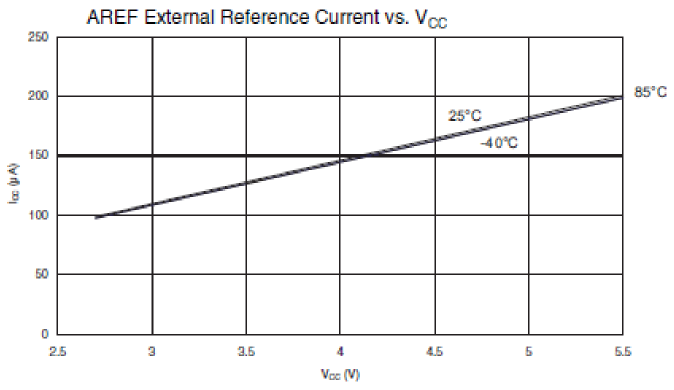


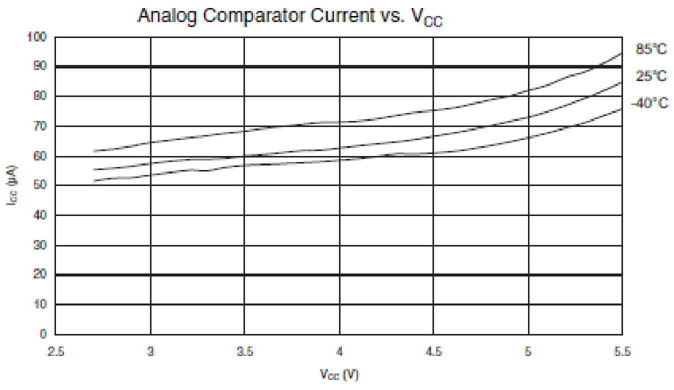


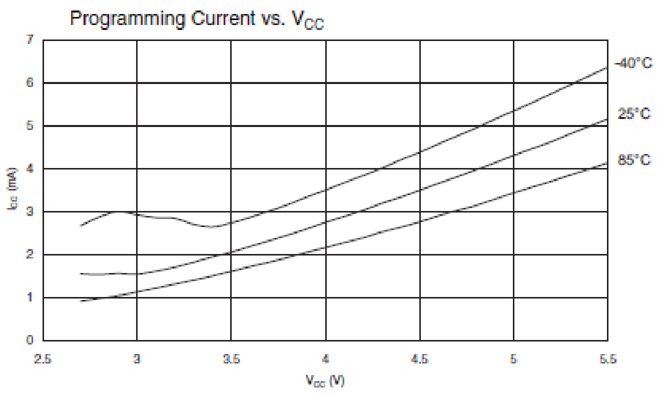
Idle mode, the Analog Comparator should be disabled if not used. When entering ADC Noise Reduction mode, the Analog Comparator should be disabled. In the other sleep modes, the Analog Comparator is automatically disabled. However, if the Analog Comparator is set up to use the Internal Voltage Reference as input, the Analog Comparator should be disabled in all sleep modes. Otherwise, the Internal Voltage Reference will be enabled, independent of sleep mode.

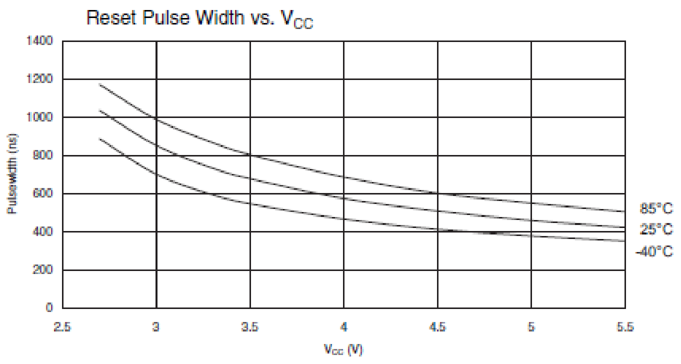


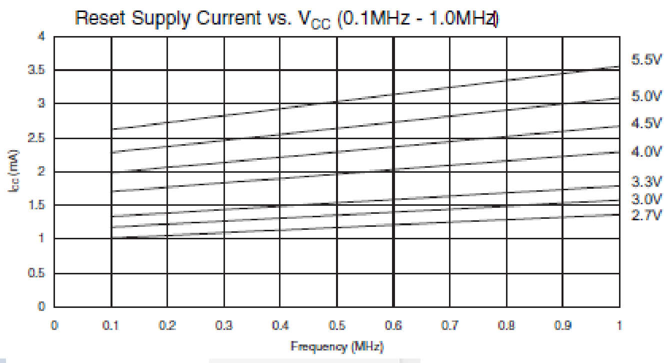


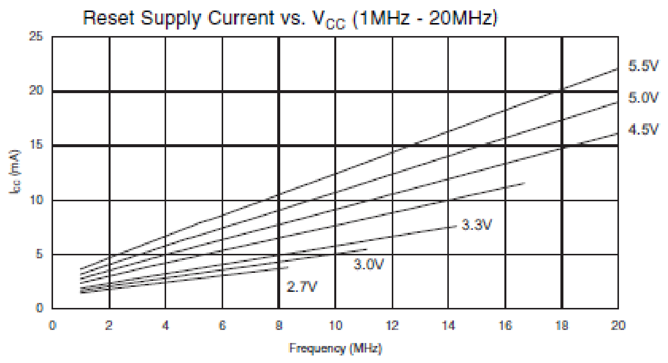






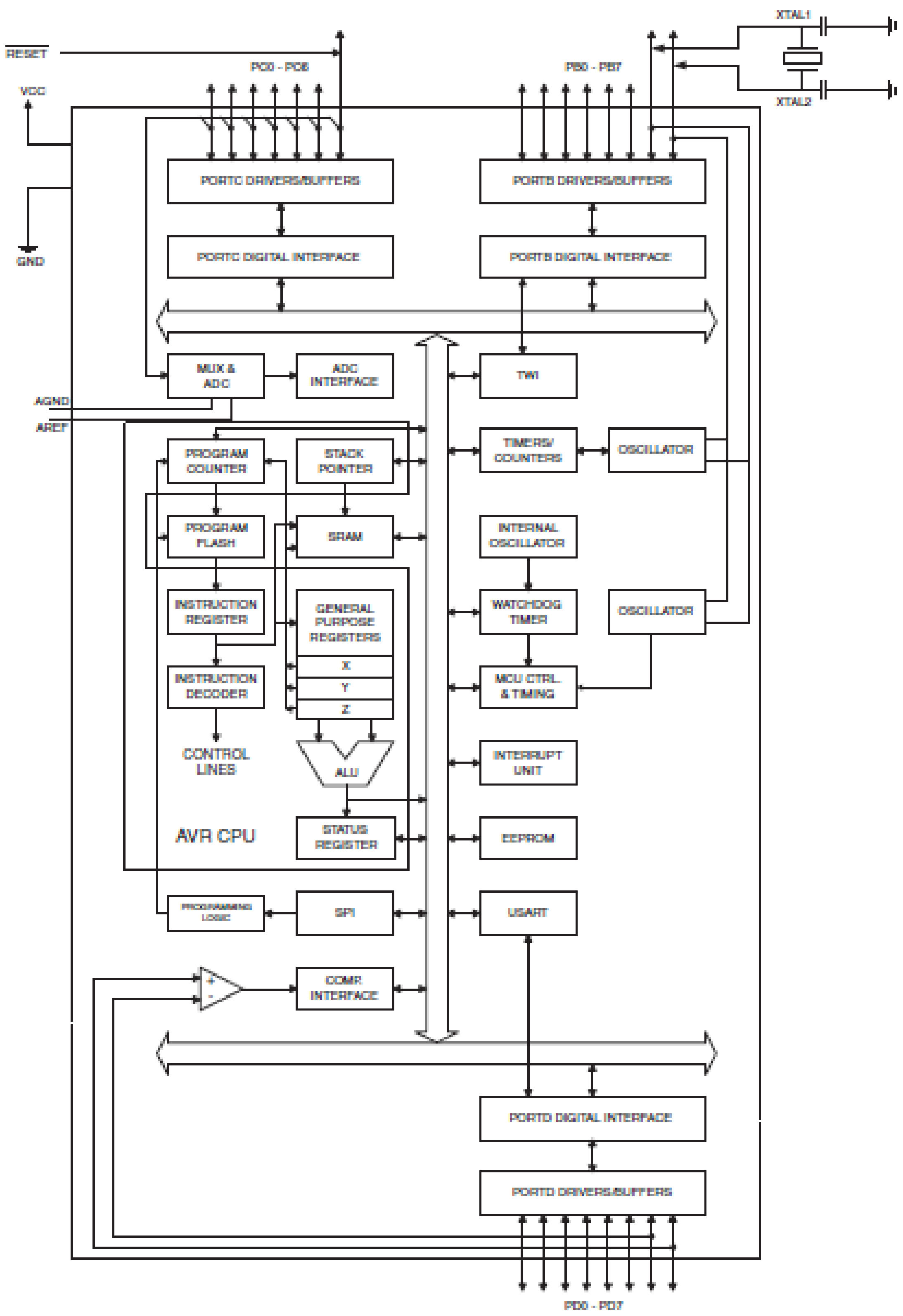






During Reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the boot section or vice versa. The I/O ports of the AVR are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running. After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset.

Figure A.4: Block Diagram of Atmega8



Appendix B

B.1 JHD12864E

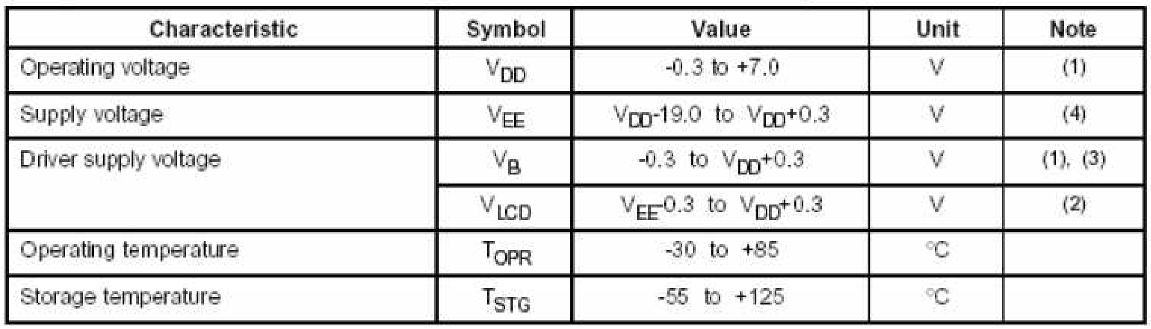
B.1.1 Features

•Display construction	128*64 DOTS
•Display mode	STN / Yellow Green
•Display type	Positive Tranflective
•Backlight	LED(Y/G)/5.0V
•Viewing direction	6 o' clock
•Operating temperature	Indoor
•Driving voltage	Single power
•Driving method	1/64 duty, 1/9 bias
•Type	COB (Chip On Board)
•Number of data line	8-bit parallel
•Connector	Pin

B.1.2 Mechanical Data

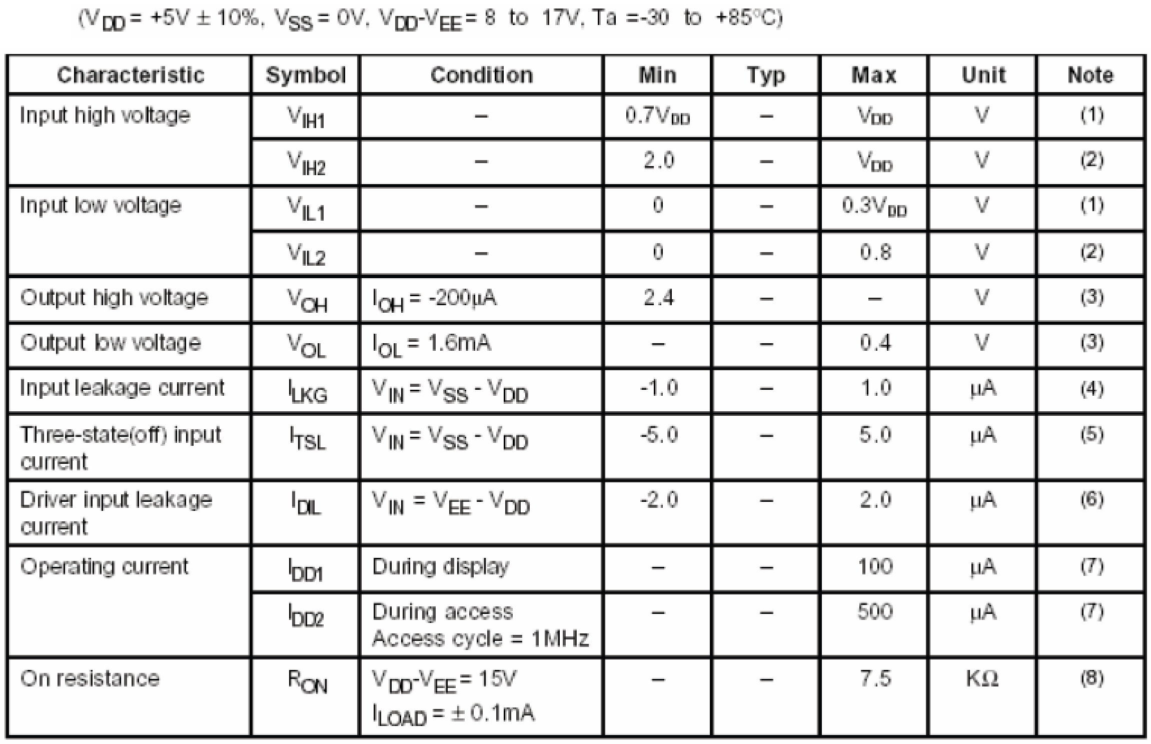
ITEM		WIDTH	HEIGHT	THICKNESS	UNIT
Module size		93.0	70.0	14.0(MAX)	mm
Viewing area		70.7	38.8	-	mm
Dot	Size	0.48	0.48	-	mm
	Pitch	0.52	0.52	-	mm
Diameter of mounting hole		2.7			mm
Weight		About 50			g

B.1.3 Maximum Ratings



Characteristic	Symbol	Value	Unit	Note
Operating voltage	V_{DD}	-0.3 to +7.0	V	(1)
Supply voltage	V_{EE}	V_{DD}-19.0 to V_{DD}+0.3	V	(4)
Driver supply voltage	V_B	-0.3 to V_{DD}+0.3	V	(1), (3)
	V_{LCD}	V_{EE}-0.3 to V_{DD}+0.3	V	(2)
Operating temperature	T_{OPR}	-30 to +85	°C	
Storage temperature	T_{STG}	-55 to +125	°C	

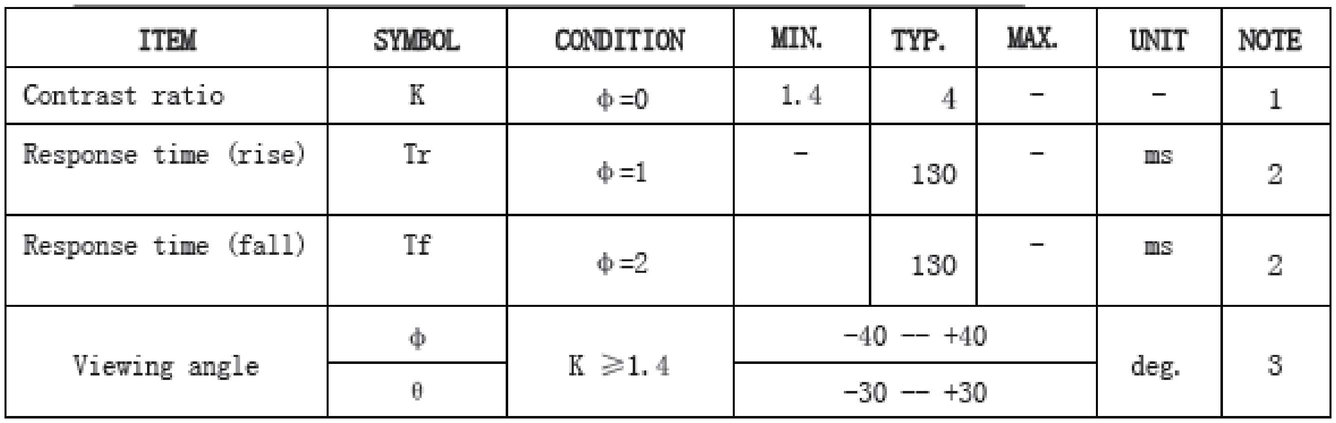
B.1.4 Electrical Characteristics



(V_{DD} = +5V ± 10%, V_{SS} = 0V, V_{DD}-V_{EE} = 8 to 17V, Ta =-30 to +85°C)

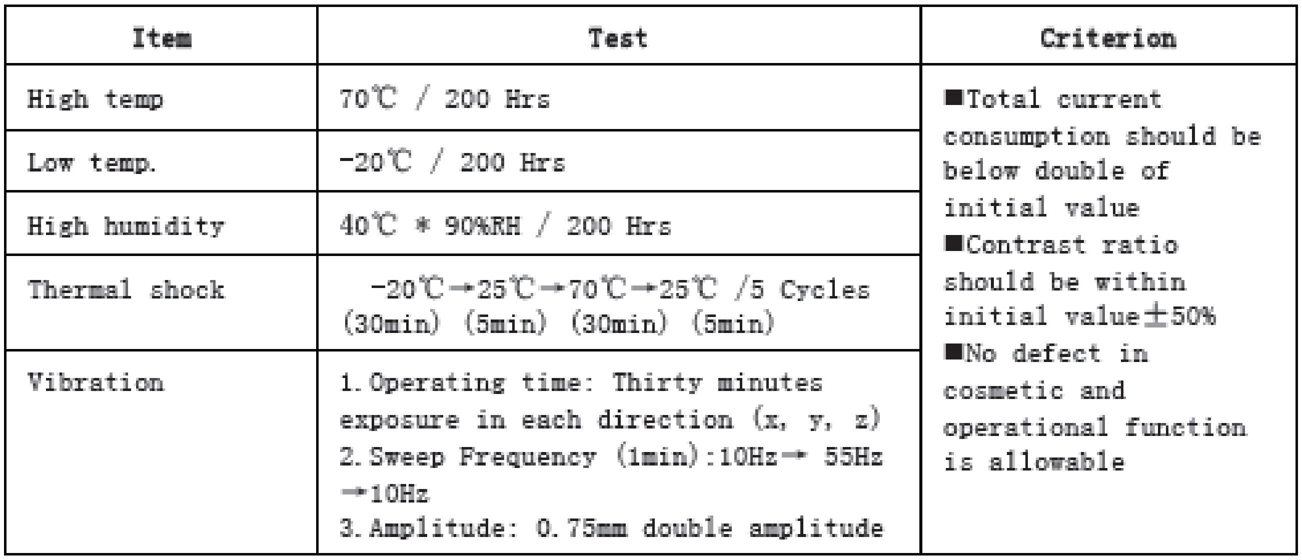
Characteristic	Symbol	Condition	Min	Typ	Max	Unit	Note
Input high voltage	V_{IH1}	–	0.7V_{DD}	–	V_{DD}	V	(1)
	V_{IH2}	–	2.0	–	V_{DD}	V	(2)
Input low voltage	V_{IL1}	–	0	–	0.3V_{DD}	V	(1)
	V_{IL2}	–	0	–	0.8	V	(2)
Output high voltage	V_{OH}	I_{OH} = -200μA	2.4	–	–	V	(3)
Output low voltage	V_{OL}	I_{OL} = 1.6mA	–	–	0.4	V	(3)
Input leakage current	I_{LKG}	V_{IN} = V_{SS} - V_{DD}	-1.0	–	1.0	μA	(4)
Three-state(off) input current	I_{TSL}	V_{IN} = V_{SS} - V_{DD}	-5.0	–	5.0	μA	(5)
Driver input leakage current	I_{DIL}	V_{IN} = V_{EE} - V_{DD}	-2.0	–	2.0	μA	(6)
Operating current	I_{DD1}	During display	–	–	100	μA	(7)
	I_{DD2}	During access Access cycle = 1MHz	–	–	500	μA	(7)
On resistance	R_{ON}	V_{DD}-V_{EE} = 15V I_{LOAD} = ± 0.1mA	–	–	7.5	KΩ	(8)

B.1.5 Electro-Optical Characteristics



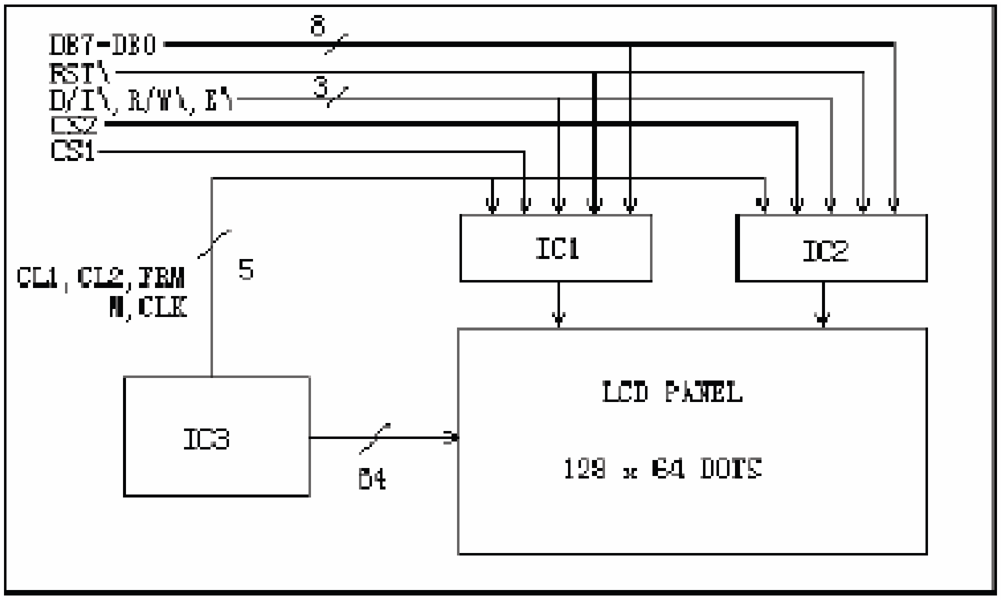
ITEM	SYMBOL	CONDITION	MIN.	TYP.	MAX.	UNIT	NOTE
Contrast ratio	K	ϕ=0	1.4	4	-	-	1
Response time (rise)	Tr	ϕ=1	-	130	-	ms	2
Response time (fall)	Tf	ϕ=2		130	-	ms	2
Viewing angle	ϕ	K ≥1.4	-40 -- +40			deg.	3
	θ		-30 -- +30				

B.1.6 Reliability Characteristics

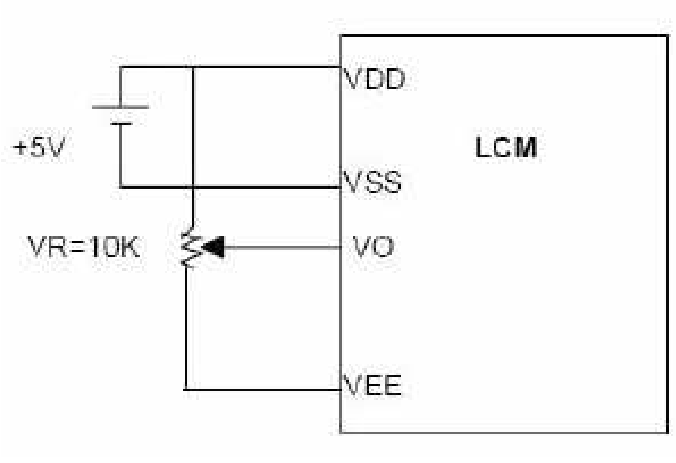


Item	Test	Criterion
High temp	70℃ / 200 Hrs	■Total current consumption should be below double of initial value ■Contrast ratio should be within initial value±50% ■No defect in cosmetic and operational function is allowable
Low temp.	-20℃ / 200 Hrs	
High humidity	40℃ * 90%RH / 200 Hrs	
Thermal shock	-20℃→25℃→70℃→25℃ /5 Cycles (30min) (5min) (30min) (5min)	
Vibration	1.Operating time: Thirty minutes exposure in each direction (x, y, z) 2.Sweep Frequency (1min):10Hz→ 55Hz →10Hz 3.Amplitude: 0.75mm double amplitude	

B.1.7 Block Diagram



B.1.8 Power Supply



Appendix C

REFERENCES

1. *www.electriciansnetworks.com*
2. *www.en.wikipedia.org*
3. *www.burkardscientific.co.uk analytical automation.htm*
4. *www.inventors.about.com od istartinventions a iPhone.htm*
5. *www.thehindu.com*
6. Timothy Hoye, Joseph Kozak *“Touch screens: A Pressing Technology”*, New Millennium Conference April, 2010.
7. Hsu, Andrew. *“Choosing a touch technology for handheld-system applications”* EDN, January 8, 2009: 40-44.
8. Nichols, Steven J. Vaughan *“New Interfaces at the Touch of a Fingertip”* IEEE Society August. 2007: 12-15.
9. Gray, Tony. *“Projected Capacitive Touch Screen Technology”* Ocular, Inc. Accessed 3 March 2010.
10. *www.apple.com*
11. *www.computer.yourdictionary.comlcd−types*
12. *www.lookfwd.doit4me.gr ge99149 electronics pic_avr.htm*
13. *www.arstechnica.com*