**ABSTRACT**

Automation is the use of control systems and information technologies to reduce the need for human work in the production of goods and services. Automation plays an important role in the world economy and in daily life. Home automation is called a smart home or smart house. A home automation system will control lighting, climate, entertainment systems, and appliances. It may also include home security such as access control and alarm systems. When connected with the Internet, home devices are an important constituent of the Internet of Things. A home automation system typically connects controlled devices to a central hub or "gateway". The user interface for control of the system uses wall-mounted terminals, tablet or desktop computers, a mobile phone application, or a Web interface, that may also be accessible off-site through the Internet. There are lot more devices available in the market for home automation but that need to be configured for each appliance in the home, more over they are not cost effective. In the proposed system Raspberry pi is used as a main controller which will control all the appliances in the home and touch screen is used as wall-mounted terminals for user interface. The proposed system replaces the old style mechanical switch with touch based switch, surveillance system, and voice over control.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** | **iv** |
| **LIST OF FIGURES** | **viii** |
| **LIST OF ABBREVATIONS** | **x** |
| **1** | **INTRODUCTION** | **1** |
|  | 1.1HOME AUTOMATION | 1 |
|  | 1.2 OBJECTIVE | 1 |
|  | 1.3 EXISTING SYSTEM | 1 |
|  | 1.4 PROPOSED SYSTEM | 2 |
| **2** | **LITERATURE SURVEY** | **3** |
| **3** | **HARDWARE IMPLEMENTATION** | **6** |
|  | 3.1 INTRODUCTION | 6 |
|  | 3.1.1 BLOCK DIAGRAM AND CIRCUIT DIAGRAM | 6 |
|  | 3.2 RASPBERRY PI | 7 |
|  | 3.2.1 TECHNICAL SPECIFICATION | 8 |
|  | 3.3 RASPBERRY PI CAMERA | 12 |
|  | 3.3.1 RASPBERRY PI CAMERA BOARD FEATURES | 13 |
|  | 3.4 RASPBERRY PI HDMI LCD (5 INCH) | 14 |
|  | 3.5 HDMI | 17 |
|  | 3.5.1 OVERVIEW OF HDMI | 18 |
|  | 3.5.2 HOW HDMI WORKS | 19 |
|  | 3.5.3 BENEFITS | 21 |
|  | 3.6 RELAY | 24 |
|  | 3.6.2 SCHEMATIC DIAGRAM OF 4CHANNEL RELAY MODULE | 25 |
|  | 3.6.3 OPERATING PRINCIPLE | 26 |
|  | 3.7 TOUCH SCREEN CONTROLLER (XPT2046) | 28 |
|  | 3.7.1 THEORY OF OPERATION | 29 |
|  | 3.7.2 TOUCH SCREEN SETTLING | 30 |
|  | 3.7.3 DIGITAL INTERFACE | 31 |
|  | 3.8 SERIAL PERIPHERAL INTERFACE (SPI) | 33 |
|  | 3.8.1 SPI DETAIL | 34 |
|  | 3.8.2 DATA AND CONTROL LINES OF THE SPI | 36 |
|  | 3.8.3 SPI CONFIGURATION | 37 |
|  | 3.9 AUDIO AMPLIFIER (PAM8403) | 39 |
|  | 3.9.1 APPLICATIONS | 40 |
|  | 3.9.2 BLOCK DIAGRAM | 40 |
|  | 3.9.3 TYPICAL APPLICATIONS CIRCUIT | 40 |
|  | 3.10 AC POWER ADAPTER | 41 |
|  | 3.10.1 SPECIFICATIONS | 41 |
|  | 3.11 CONCLUSION | 42 |
| **4** | **SOFTWARE IMPLEMENTATION** | **43** |
|  | 4.1 INTRODUCTION | 43 |
|  | 4.1.1 FLOW CHART | 43 |
|  | 4.1.2 PROGRAM FOR GRAPHICAL USER INTERFACE | 44 |
|  | 4.2 PYTHON | 56 |
|  | 4.2.2 FEATURES AND PHILOSOPHY | 56 |
|  | 4.3 PYGAME | 58 |

|  |  |  |
| --- | --- | --- |
|  | 4.4 SNOWBOY, A CUSTOMIZABLE HOTWORD DETECTION ENGINE | 58 |
|  | 4.5 PROGRAM OUTPUT | 59 |
|  | 4.6 CONCLUSION | 60 |
| **5** | **CONCLUSION AND FUTURE SCOPE** | **61** |
|  | 5.1 FUTURE SCOPE | 61 |
|  | 5.2 CONC LUSION | 61 |
|  | **REFERENCES** | **62** |
|  | **APPENDIX 1** | **64** |
|  | **APPENDIX 2** | **66** |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **TITLE** | **PAGE NO** |
| 1.1 | Block Diagram of Existing System | 2 |
| 3.1 | Block Diagram of Proposed System | 6 |
| 3.2 | Circuit Diagram of Proposed System | 7 |
| 3.3 | Raspberry Pi Block Diagram | 8 |
| 3.4 | SoC (BCM2837) | 10 |
| 3.5 | GPIO Pin Diagram | 10 |
| 3.6 | USB Controller Diagram | 11 |
| 3.7 | Raspberry Pi Antenna | 12 |
| 3.8 | Raspberry Pi Board Layout | 12 |
| 3.9 | Camera Module | 13 |
| 3.10 | HDMI Display | 14 |
| 3.11 | TFT LCD Driver Block Diagram | 15 |
| 3.12 | HDMI Plug and Port | 19 |
| 3.13 | Block Diagram of HDMI | 20 |
| 3.14 | 4Channel Relay Module | 25 |
| 3.15 | Schematic 4Channel Relay Module | 26 |
| 3.16 | Relay Diagram | 27 |
| 3.17 | Block Diagram of XPT2046 | 28 |
| 3.18 | Pin Diagram of XPT2046 | 29 |
| 3.19 | Basic Operation of XPT2046 | 29 |
| 3.20 | Single master, single slave SPI implementation | 35 |
| 3.21 | Single master, multiple slave SPI implementation | 35 |
| 3.22 | Functional Block Diagram of PAM8403 | 40 |
| 3.23 | 3.24 Pin Diagram of PAM8403 | 40 |
| 3.24 | Power Adapter | 41 |
| 3.25 | Hardware Implemented Circuit | 42 |
| 4.1 | Flow Chart of Main GUI | 44 |
| 4.2 | Program Output Indicating “Light1” is Turned Off | 60 |
| 4.3 | Program Output Indicating “Light1” is Turned On | 60 |

**LIST OF ABBREVATIONS**

|  |  |
| --- | --- |
| **ABBREVATIONS** | **EXPANSION** |
| SPI | SERIAL PERIPHERAL INTERFACE |
| MOSI | MASTER OUT SLAVE IN |
| MISO | MASTER IN SLAVE OUT |
| SCK | SERIAL CLOCK |
| CS | CHIP SELECT |
| HDMI | HIGH DEFINITION MULTI MEDIA INTERFACE |
| Gbps | GIGABITS PER SECOND |
| GPIO | GENERAL PURPOSE INPUT OUTPUT |
| SOC | SYSTEM ON CHIP |
| DSI | DISPLAY SERIAL INTERFACE |
| CSI | CAMERA SERIAL INTERFACE |
| ASR | AUTOMATIC SPEECH RECOGNITION |
| ML | MACHINE LEARNING |

**CHAPTER 1**

**INTRODUCTION**

* 1. **HOME AUTOMATION**

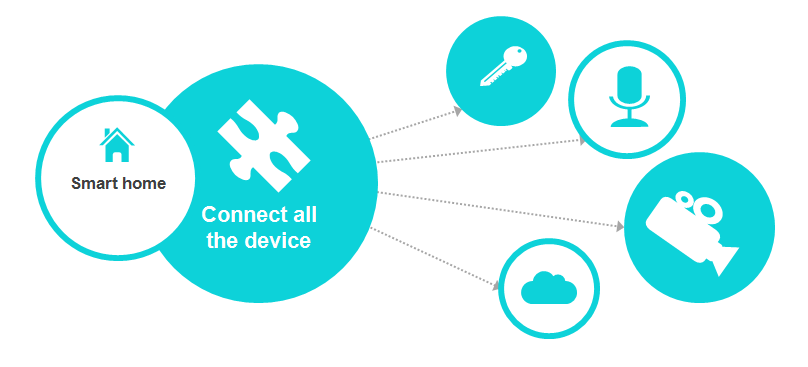
Automation is today’s fact, where more things are being controlled every day automatically, usually the basic tasks of turning on or off certain devices and beyond, either remotely or in close proximity. The control of the devices when completely taken over by the machines, the process of monitoring and reporting becomes more important. We are more and more relinquishing the power for simple but routine tasks while we need to maintain as much control as we can over the automated processes. Automation lowers the human judgment to the lowest degree possible but does not completely eliminate it. Depending on the location of its usage, automation differs in its name as industrial automation, home automation etc. With the development of low cost electronic components home automation migrated from being an industrial application to home automation. The home automation, our point of concern deals with the control of home appliances from a central location. Market researches claim that most of the homes will be equipped with home automation systems in the very near future.

**1.2 OBJECTIVE**

* To design easy to install home appliances controller
* To protect personal privacy from third party companies
* To provide low cost surveillance system

**1.3 EXISTING SYSTEM**

In existing system home appliances should be configured for each device as shown in Figure 1.1. This will make the home automation task very difficult for normal people.



**Figure 1.1 Block Diagram of Existing System**

**1.4 PROPOSED SYSTEM**

Raspberry pi is used as a main controller and home appliances are controlled through interactive graphical user interface with flexibility  and convenience in mind, it can be controlled by a range of touch screens, with different levels of functionality. A central master touch screen can be installed in a key site such as a lobby or sitting room giving complete control over every function around the property. Alternatively a number of discreet screens can be sited conveniently around the property; each is as clever as the next and so able to control the automated functions in each room. Graphical user interface programs are written in python language using pygame library. Pygame library is used to design graphical user interface. This is very useful and easy to use library for GUI. All the home appliances can be controlled through voice, though plenty of devices available in the market for voice control like Amazon’s Alexa and Google’s Google Home, which requires active internet connection. Due to internet connection all the existing devices are Streaming our data to third party companies, it can be the reason for personal privacy violation. So in this project all the voice recognition process are done only in on device through Python library called “Snow Boy”.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Bhavkanwal Kaur, Pushpendra Kumar Pateriya, Mritunjay Kumar Rai “An Illustration of Making a Home Automation System Using Raspberry Pi and PIR Sensor” in 2018 International Conference on Intelligent Circuits and Systems, pp.439-44, April-2018.**

Devices like Hue Lights and Hive Bulbs are playing a major role in converting our homes into smart homes. Smart homes do need smart protection against any sort of thefts or dangers to home. A simple smart home automation system can be developed using Raspberry Pi and some extra components. Raspberry Pi is a small microcomputer nearly size of a business card. In this paper we will present a small review on the Raspberry Pi and we will give an illustration explaining that how simple is to make a home automation system.

**2.2 Shafiq ur Rehman, Volker Gruhn “An Approach to Secure Smart Homes in Cyber- Physical Systems/Internet-of-Things” in Fifth International Conference on Software Defined Systems (SDS), pp.126-129,April-2018.**

Home automation plays an important role in our daily life to control home appliances from a single touch or click. In this case, users are able to check home appliances from laptops, iPhone, iPod or other smart phone devices which supports to swap household appliances. Sensors are one of the mode important parts for smart home technology. Smart homes also help elderly people; they can handle the home with simplicity from their remote or smart device which is linked up to central device that is connected to home automation. Security is the main concern while maintains the smart homes, nowadays hackers don’t need to travel to targeted homes, they can approach virtually to attack smart homes easily. In this paper, we propose a secure architecture for smart homes. We added a sicher firewall system between a central hub (LAN) that are linked to the cyberspace and other end that is connected to home appliances. In this way security threats become dejected and attackers are unable to reach home automation systems. A sicher firewall also protects system from internet threats. It generates a warning and protect system from external threats. Applying a sicher firewall, it enforces security system and increase security and concealment to smart homes. This experience will enhance user trustworthiness for using smart homes technology.

**2.3 Pasd Putthapipat, Chutitep Woralert, Phumiphat Sirinimnuankul “Speech Recognition Gateway for Home Automation on Open Platform” in International Conference on Electronics, Information, and Communication (ICEIC), pp.1-4, Jan-2018.**

The purpose of this paper is to implement the home automation voice gateway using PiFrame framework. With this system, user can control home appliances through voice command, by recognizing the keyword in the speech of the user. For example, turning on or off the light or any other electrical equipment at home automatically by talking to the gateway. With full-feature OS on the platform, the development process is shorten significantly and the limitation on embedded system is also reduced, allowing the complex functionality such as cloud computing, authentication and scalability, so that he developers are able to implement more functionality to the system without having to redesign the platform.

**2.4 Bhaumik Vaidya, Ankit Patel, Anand Panchal “Smart home automation with a unique door monitoring system for old age people using Python, OpenCV, Android and Raspberry pi” in International Conference on Intelligent Computing and Control Systems pp.82-86, June-2017.**

Smart home automation system particularly for old age people is proposed based on python, OpenCV, raspberry pi and android application. The appliances are controlled by the Raspberry pi server, which operates according to the user command (touch or voice) received from the mobile phone. A unique door monitoring system is designed based on face detection and recognition from a camera installed outside the main door, which can be accessed from the phone using android application. One interesting feature that has been added is that, all the appliances can also be controlled through the voice of user. For energy efficiency user can analyze the usage of each appliance from their phone. Moreover, user can also control the intensity of light as well as the speed of the fan. With all this features incorporated in a single system with good and simple user interface, this system is cost effective and perfect for old age people living alone in their houses.

**2.5 H.Bharathi, U.Srivani, MD.Azharudhin, M.Srikanth, M.Sukumarline “Home Automation by Using Raspberry Pi And Android Application” in International Conference on Electronics, Communication and Aerospace Technology, vol-2,pp.687-689, April-2017.**

Automation is the use of control systems and information technologies to reduce the need for human work in the production of goods and services. Automation plays an important role in the world economy and in daily life. In this we describe the design and development of a remote household appliance control system using raspberry pi and android mobile phone. We will be able to control any appliance in home from anywhere throughout the world, when you have done with this Raspberry Pi home automation. It is necessary to control the home from any location. Home automation is the control of any electrical and electronics device in our home and office whether we are there or anywhere.

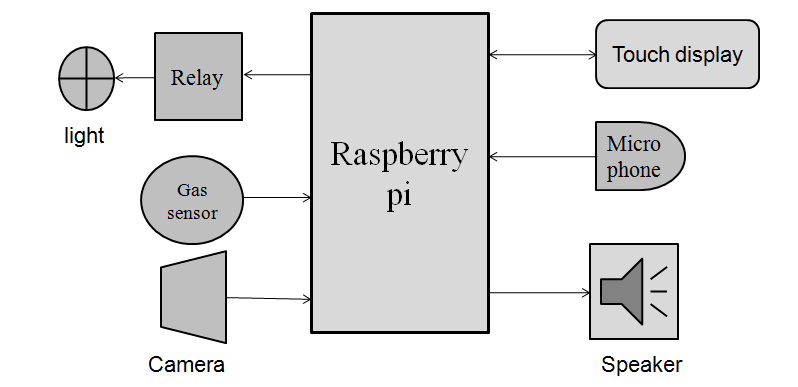
**CHAPTER 3**

**HARDWARE IMPLEMENTATION**

**3.1 INTRODUCTION**

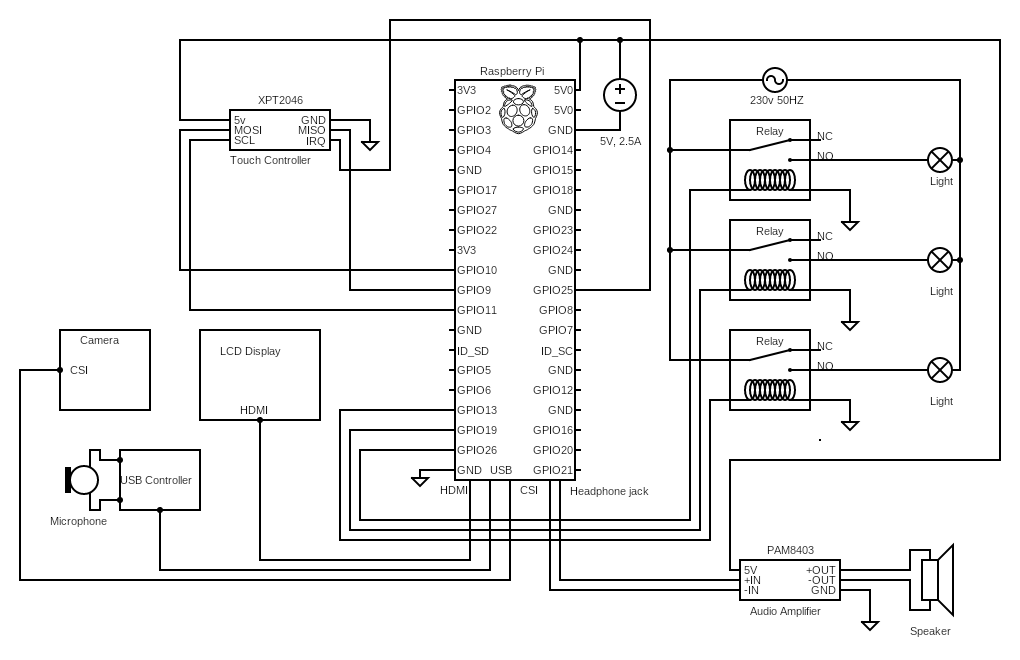
This chapter gives the detailed explanation of hardware components used and represents how the components are assembled together.

**3.1.1 BLOCK DIAGRAM AND CIRCUIT DIAGRAM**

****

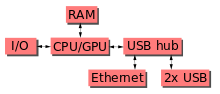
**Figure 3.1 Block Diagram of Proposed System**

The block diagram and circuit diagram of proposed system is shown in Figure 3.1 and Figure 3.2 as respectively. Raspberry pi is supplied from 5V, 2,5A AC adapter. 800\*480 resolution LCD is connected to the Raspberry pi through HDMI connector and touch controller XPT2046 is connected through SPI communication. For voice detection USB microphone is used. Audio amplifier PAM8403 is used to amplify signal from 3.5mm audio jack which gives feedback to user based on audio replay. 5MP camera provides motion detection based surveillance system and it is connected to Raspberry pi through CSI port.

**Figure 3.2 Circuit Diagram of Proposed System**

**3.2 RASPBERRY PI**

The Raspberry Pi is a low cost, credit-card sized computer that plugs into a computer mouse. The Raspberry Pi hardware has evolved through several versions that feature variations in memory capacity and peripheral-device support. This block diagram describes Model B and B+; Model A, A+, and the Pi Zero are similar, but lack the Ethernet and USB hub components. The Ethernet adapter is internally connected to an additional USB port. In Model A, A+, and the Pi Zero, the USB port is connected directly to the system on a chip (SoC). On the Pi 1 Model B+ and later models the USB/Ethernet chip contains a five-port USB hub, of which four ports are available, while the Pi 1 Model B only provides two. On the Pi Zero, the USB port is also connected directly to the SoC, but it uses a micro USB (OTG) port.



**Figure 3.3 Raspberry Pi Block Diagram**

**3.2.1 TECHNICAL SPECIFICATION**

**PROCESSOR**

* Broadcom BCM2387 chipset.
* 1.2GHz Quad-Core ARM Cortex-A53 (64Bit)

**B/G/N WIRELESS LAN AND BLUETOOTH 4.1**

* IEEE 802.11 b / g / n Wi-Fi. Protocol: WEP, WPA WPA2, algorithms AES-CCMP (maximum key length of 256 bits), the maximum range of 100 meters.
* IEEE 802.15 Bluetooth, symmetric encryption algorithm Advanced Encryption Standard (AES) with 128-bit key, the maximum range of 50 meters.

**GPU**

* Dual Core Video Core IV® Multimedia Co-Processor. Provides Open GL ES 2.0, hardware-accelerated Open VG, and 1080p30 H.264 high-profile decode.
* Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs with texture filtering and DMA infrastructure

**MEMORY**

* 1GB LPDDR2

**OPERATING SYSTEM**

* Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IoT

**POWER**

* Micro USB socket 5V1, 2.5A

**CONNECTORS**

**ETHERNET**

* 10/100 BaseT Ethernet socket

**VIDEO OUTPUT**

* HDMI (rev 1.3 & 1.4)
* Composite RCA (PAL and NTSC)

**AUDIO OUTPUT**

* Audio Output 3.5mm jack
* HDMI
* USB 4 x USB 2.0 Connector

**GPIO CONNECTOR**

* 40-pin 2.54 mm (100 mil) expansion header: 2x20 strip
* Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines

**CAMERA CONNECTOR**

* 15-pin MIPI Camera Serial Interface (CSI-2)

**DISPLAY CONNECTOR**

* Display Serial Interface (DSI) 15 way flat flex cable connector with two data lanes and a clock lane

**MEMORY CARD SLOT**

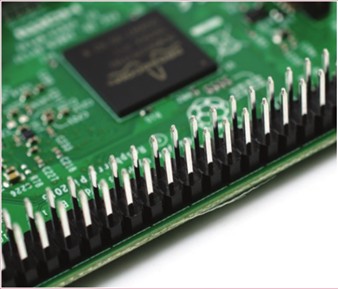
* Push/pull Micro SDIO

# SYSTEM ON CHIP (SOC)

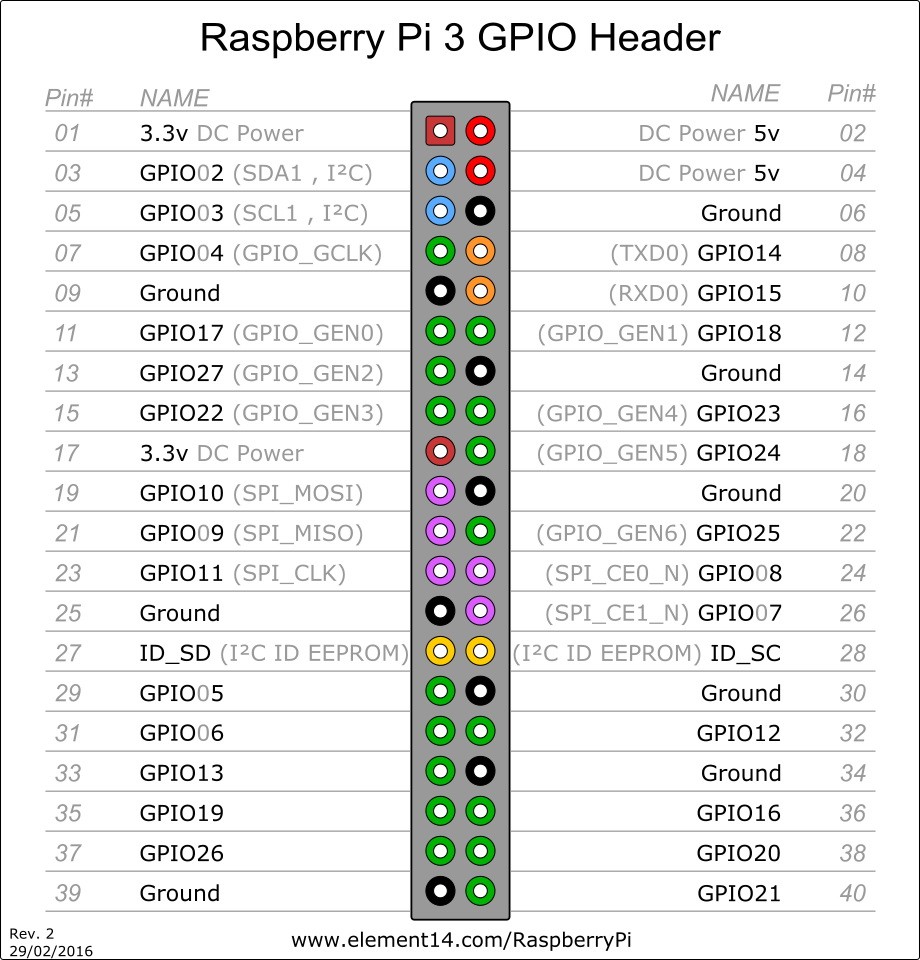
Built specifically for the new Pi 3, the Broadcom BCM2837 system-on-chip (SoC) includes four high-performance ARM Cortex-A53 processing cores running at 1.2GHz with 32kB Level 1 and 512kB Level 2 cache memory, a Video Core IV graphics processor, and is linked to a 1GB LPDDR2 memory module on the rear of the board

**Figure 3.4 SoC (BCM2837)**

# GPIO

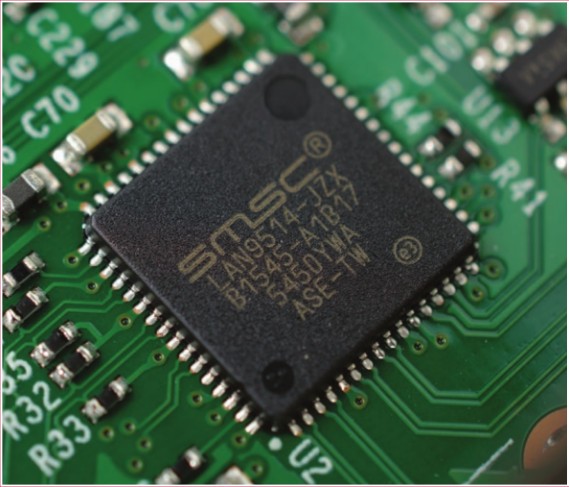
The Raspberry Pi 3 features the same 40-pin general-purpose input-output (GPIO) header as all the Pi’s going back to the Model B+ and Model A+. Any existing GPIO hardware will work without modification; the only change is a switch to which UART is exposed on the GPIO’s pins, but that’s handled internally by the operating system.

**Figure 3.5 GPIO Pin Diagram**



**Table 3.1 Raspberry pi Pin Configuration**

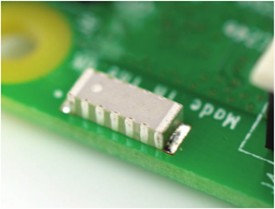
# USB CHIP

The Raspberry Pi 3 shares the same SMSC LAN9514 chip as its predecessor, the Raspberry Pi 2, adding 10/100 Ethernet connectivity and four USB channels to the board. As before, the SMSC chip connects to the SoC via a single USB channel, acting as a USB-to Ethernet adaptor and USB hub.

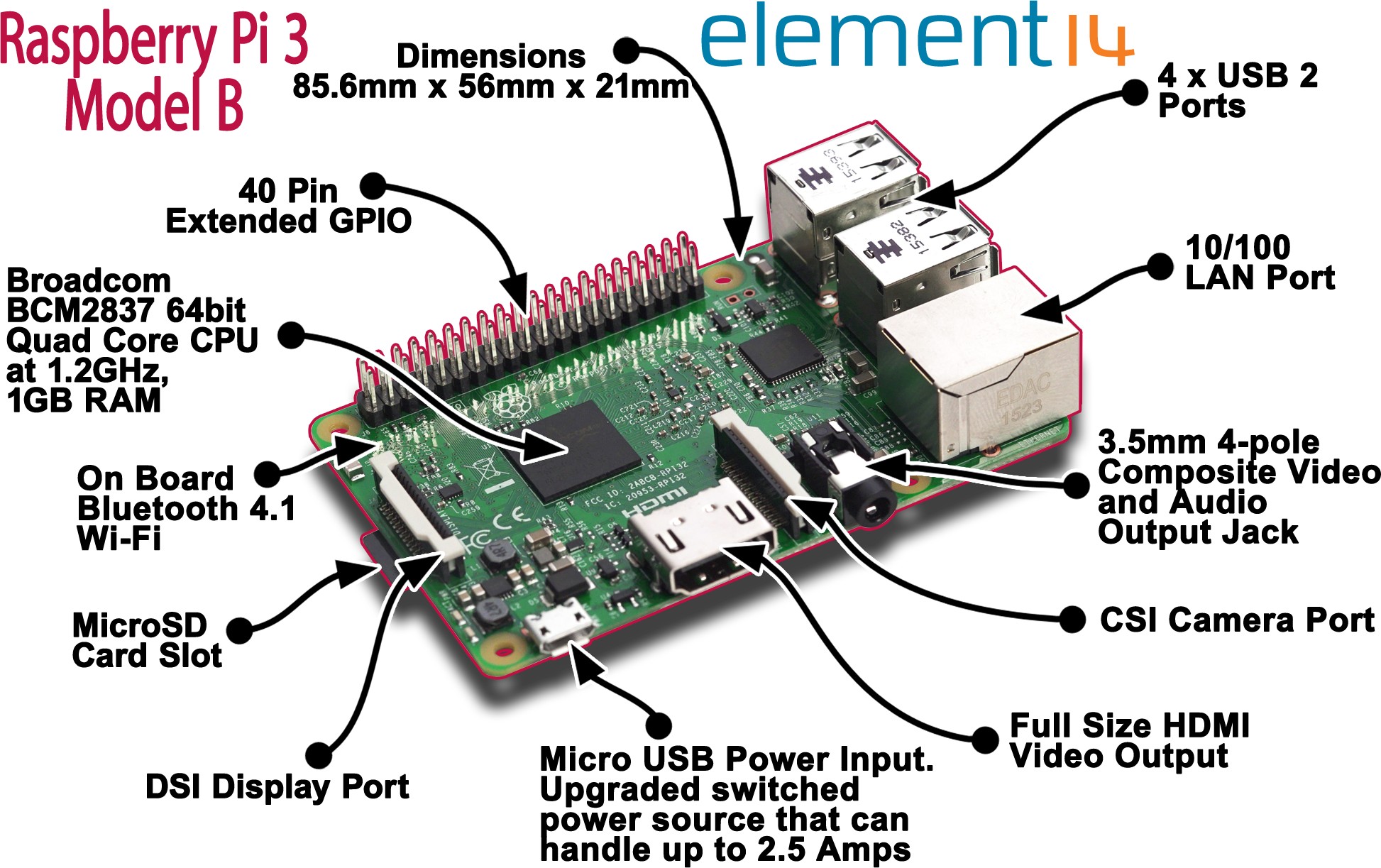
**Figure 3.6 USB Controller Diagram**

# ANTENNA

There’s no need to connect an external antenna to the Raspberry Pi 3. Its radios are connected to this chip antenna soldered directly to the board, in order to keep the size of the device to a minimum. Despite its diminutive stature, this antenna should be more than capable of picking up wireless LAN and Bluetooth signals – even through walls.



**Figure 3.7 Raspberry Pi Antenna**



**Figure 3.8 Raspberry Pi Board Layout**

**3.3 RASPBERRY PI CAMERA**

Raspberry Pi Camera Board plugs directly into the CSI connector on the Raspberry Pi. It's able to deliver a crystal clear 5MP resolution image or 1080p HD video recording at 30fps! Latest Version 1.3! Custom designed and manufactured by the Raspberry Pi Foundation in the UK, the Raspberry Pi Camera Board features a 5MP (2592\*1944 pixels) Omnivision 5647 sensor in a fixed focus module. The module attaches to Raspberry Pi, by way of a 15 Pin Ribbon Cable, to the dedicated 15-pin MIPI Camera Serial Interface (CSI), which was designed especially for interfacing to cameras. The CSI bus is capable of extremely high data rates, and it exclusively carries pixel data to the BCM2835 processor. The board itself is tiny, at around 25mm x 20mm x 9mm, and weighs just over 3g, making it perfect for mobile or other applications where size and weight are important. The sensor itself has a native resolution of 5 megapixel, and has a fixed focus lens onboard. In terms of still images, the camera is capable of 2592 x 1944 pixel static images, and also supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 video recording. The camera is supported in the latest version of Raspbian, the Raspberry Pi's preferred operating system.

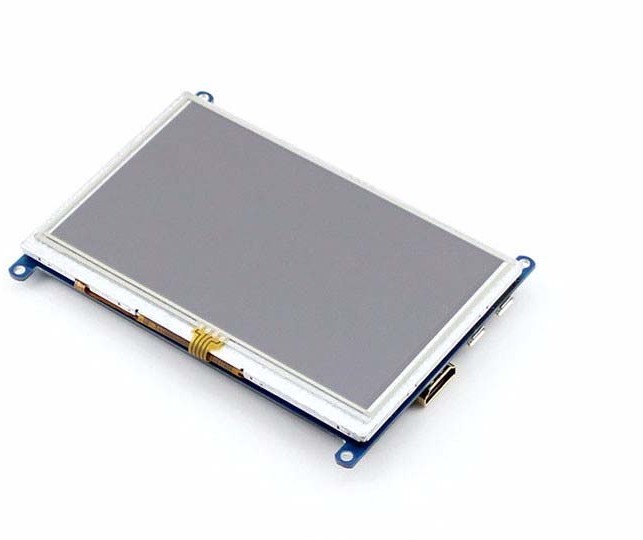


**Figure 3.9 Camera Module**

**3.3.1 RASPBERRY PI CAMERA BOARD FEATURES**

* Fully Compatible with Both the Model A and Model B Raspberry Pi
* 5MP Omnivision 5647 Camera Module
* Still Picture Resolution: 2592 x 1944
* Video: Supports 1080p @ 30fps, 720p @ 60fps and 640x480p 60/90 Recording
* 15-pin MIPI Camera Serial Interface - Plugs Directly into the Raspberry Pi Board
* Size: 20 x 25 x 9mm

**3.4 RASPBERRY PI HDMI LCD (5 INCH)**

Raspberry Pi HDMI LCD (5 inch) is a low-cost 5 inch Resistive Touch Screen LCD, HDMI interface, designed for Raspberry Pi for Raspberry Pi.

**Figure 3.10 HDMI Display**

**Key Features**

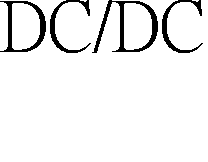
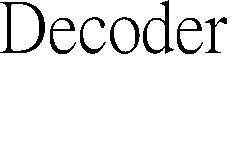
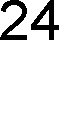
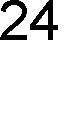
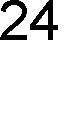
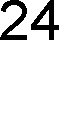
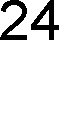
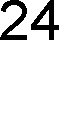
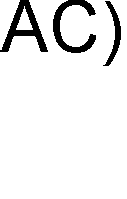
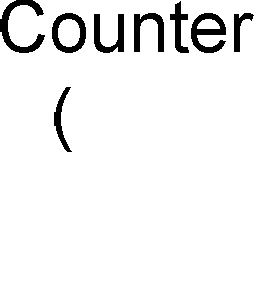
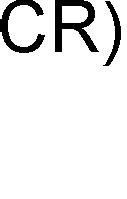
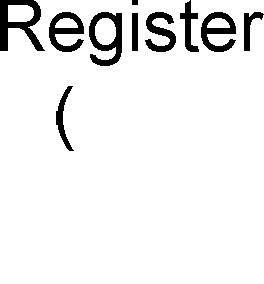
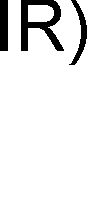
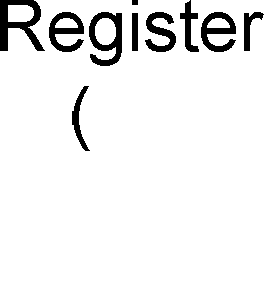
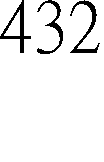
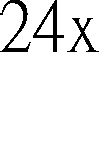
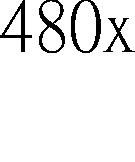
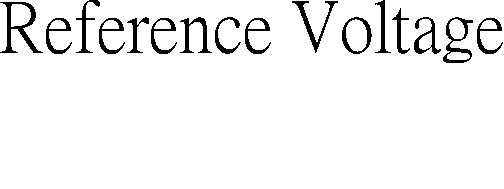
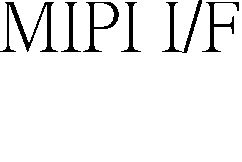
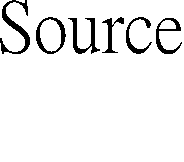
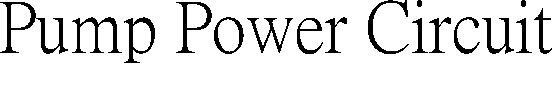
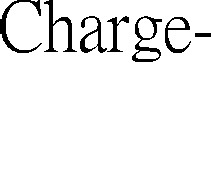
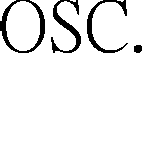
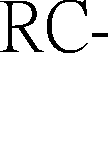
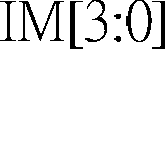
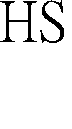
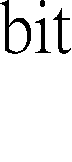
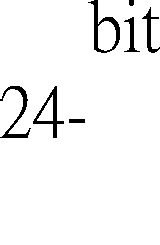
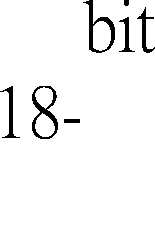
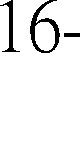
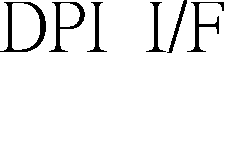
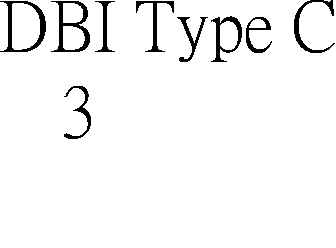
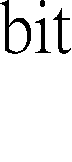
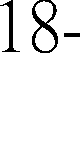
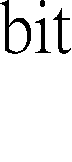
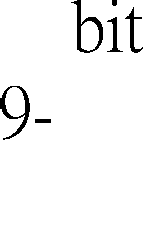
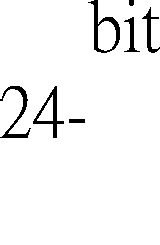
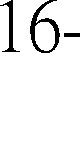
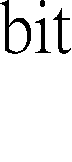
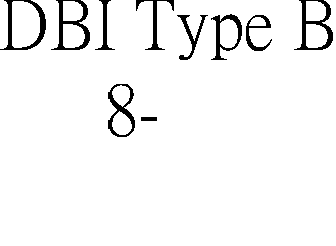
* 800×480 high resolution, touch control
* Compatible and Direct-connect with any revision of Raspberry Pi (except the Pi 1 model B or Pi Zero, which requires an HDMI cable)
* Drivers provided (works with your own Raspbian/Ubuntu directly)
* HDMI interface for displaying, no I/Os required (however, the touch panel still needs I/Os)
* Back light control to lower power consumption
* High quality immersion gold surface plating

**3.4.2 TFT LCD SINGLE CHIP DRIVER (ILI9806H)**

The ILI9806H is a 16.7M single-chip SOC driver for a-Si TFT liquid crystal display panels with a resolution of up to 480 (RGB) x 864 with memory. The ILI9806H is comprised of a 1440-channel source driver, a gate-IC-less level shifter, 622,080-byte GRAM for graphic data of 480 (RGB) x 864 dots, and a power supply circuit.

The ILI9806H supports parallel MPU 8-/9-/16-/18-/24-bit data bus interfaces and a 3-line serial peripheral interface to input commands. The ILI9806H supports a RGB (16-/18-/24-bit) data bus for video image display. For high-speed serial interface, the MIPI DSI interface mode, the ILI9806H supports two data lanes and one clock lane for high-speed and low power transmission in both directions with low EMI noise.

The ILI9806H operates with a wide range of an analog power supply. The ILI9806H supports 8-color display, sleep mode and deep standby power



**Figure 3.11 TFT LCD Driver Block Diagram**

management functions, ideal for medium or small size portable products where battery power conservation is desirable, such as digital cellular phones, smart phones, MP3 players, personal media players, and similar devices with color graphics display. Additionally, it has an internal DC/DC converter that generates the LCD driving voltage and the voltage follower circuit for LCD driver.

**Features**

**Display Resolution Options**

* 480 (RGB) (H) x 864 (V) with GRAM
* 480 (RGB) (H) x 854 (V) with GRAM
* 480 (RGB) (H) x 800 (V) with GRAM
* 480 (RGB) (H) x 720 (V) with GRAM
* 480 (RGB) (H) x 640 (V) with GRAM

**Display Color modes**

**Full Color Mode**

* 16.7M colors (24-bit data, R: 8-bit, G: 8-bit, B: 8-bit)

**Reduced color modes**

* 262K colors (18-bit data, R: 6-bit, G: 6-bit, B: 6-bit)
* 65K colors (16-bit data, R: 5-bit, G: 6-bit, B: 5-bit)
* 8 colors (Idle Mode On): 8 colors (3-bit data, R: 1-bit, G: 1-bit, B: 1-bit)

**Display Module**

* On-chip Frame Memory size 622,080 byte , 480 (RGB) (H) x 864 (V) x 24 bits
* Supports 1440 source channel outputs
* Supports gate control signals to gate driver in the panel
* Supports column/1-dot/2-dot/3-dot inversion
* Gamma correction (1 preset Gamma curve)
* On module VCOM control (0V to -4V common electrode output voltage range)

**Display Interface Types**

**Mpu Mode**

* MIPI-DBI Type B (Display Bus Interface, 80 System) interface, 8/9/16/18/24-bit bus
* MIPI-DBI Type C (Serial data transfer interface, 3-line SPI) interface

**MIPI-DSI (Display Serial Interface) Interface**

* Supports one data lane/maximum speed 850Mbps or
* Supports two data lanes/maximum speed 500Mbps
* Supports DSI version 1.01.00
* Supports D-PHY version 1.00.00
* Supports DCS version 1.02.00

**MIPI-DPI (Display Pixel Interface) Interface**

* 16 bit/pixel (R: 5-bit, G: 6-bit, B: 5-bit)
* 18 bit/pixel (R: 6-bit, G: 6-bit, B: 6-bit)
* 24 bit/pixel (R: 8-bit, G: 8-bit, B: 8-bit)

**Power saving modes**

* Deep-standby mode

**3.5 HDMI**

HDMI (High-Definition Multimedia Interface) is an interface standard used for audio visual equipment such as high-definition television and home theatre systems. With 19 wires wrapped in a single cable that resembles a USB wire, HDMI is able to carry a bandwidth of 5 Gbps(gigabits per second). This is more than twice the bandwidth needed to transmit multi- channel audio and video. This and several other factors make HDMI much more desirable than its predecessors, component video, S-Video and composite video.

HDMI is the first & only industry supported, uncompressed, all-digital audio/video interface. HDMI provides an interface between any A/V source, such as a set-top box, DVD player, or A/V receiver and an audio and/or video monitor, such as a digital television (DTV), over a single cable. HDMI supports standard, enhanced, or high-definition video, plus multi-channel digital audio on a single cable. Transmits all ATSC HDTV standards and supports 8-channel, 192kHz, uncompressed digital audio, all currently-available compressed formats & lossless digital audio formats with bandwidth to spare to accommodate future enhancements and requirement

**3.5.1 OVERVIEW OF HDMI**

HDMI can deliver high quality sound or vision without the risk of quality loss due to the conversion or compression of a video or audio signal.HDMI pictures are smoother and sharp. Sound is also crisp and taut, without any distortion. And of course, using the single cable HDMI can get rid of a lot of messy cables snaking around your home theatre kit.

Because of its digital nature, HDMI also works well with fixed-pixel displays such as LCD, plasma or DLP screens and projectors. A HDMI cable allows you to exactly match pixel-by pixel the native resolution of the screen with whatever source device you've got connected. HDMI systems will also automatically convert a picture into its most appropriate format, such as 16:9 or 4:3. HDMI signals are digital in nature while conventional TVs and radios operate on analog signals, on the contrary HDTVs works on digital signals. HDMI has some built-in smarts that allow you to control any device connected via HDMI through the one remote. HDMI connectors shown in Figure 3.12

****

**Figure 3.12 HDMI Plug and Port**

**3.5.2 HOW HDMI WORKS**

HDMI uses transition minimized differential signaling (TMDS) to move information from one place to another. TMDS is a way of encoding the signal to protect it from degrading as it travels down the length of the cable.

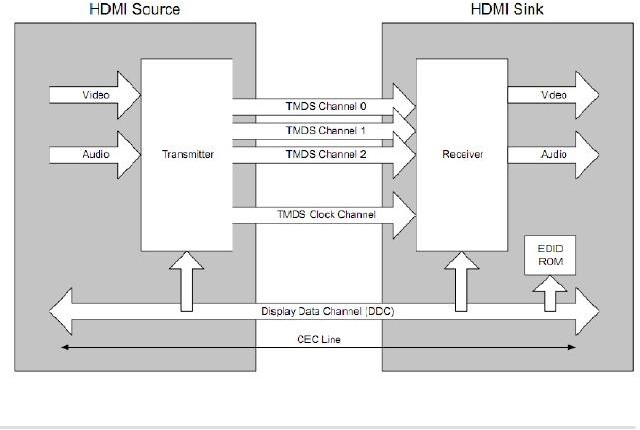
Transition Minimized Differential Signaling(TMDS) is a technology for transmitting high- speed serial data and is used by the DVI and HDMI video interfaces, as well as other digital communication interfaces. This standard codes an eight-bit data into a 10-bit signal and transfers them using differential transmission.

The sending device, such as an HD-DVD player, encodes the signal to reduce the number of transitions between one (on) and zero (off). Think of each transition as a sharp drop-off -- as the signal travels, this drop-off can begin to wear away, degrading the signal. The encoding step helps protect signal quality by reducing the number of chances for the signal to degrade.

One of the cables in the twisted pair carries the signal itself. The other carries an inverse copy of the signal.

The receiving device, such as an HDTV, decodes the signal. It measures the differential, or the difference between the signal and its inverse. It uses this information to compensate for any loss of signal along the way.

* follow all of the rules for an HDMI Sink and HDMI Source.
* four differential pairs that make up the TMDS data and clock channels.
* DDC is used for configuration and status exchange between a single Source and a single Sink.
* CEC protocol provides high-level control functions.



**Figure 3.13 Block Diagram of HDMI**

**FEATURES OF HDMI**

* HDMI technology eliminates unnecessary signal conversions.
* HDMI technology supports standard, enhanced, or high-definition video at 24 bits/pixel, 165MHz max clock frequency.
* HDMI technology supports up to 8 channel digital audio on a single cable eliminating costly A/D signal conversions.
* HDMI offers Bi-directional control signal transfer.
* HDMI offers 5 Gbps bandwidth, 55% spared for future expansion.
* HDMI offers 1 simple, user-friendly connector.
* HDMI technology is backward compatible to DVI hot plug enabled assemblies up to 5 meters in length.
* Long lengths available, ATC tested up to 12M and up to 40 M when using EQ Technology

**3.5.3 BENEFITS**

* **High-Definition Audio and Video:** HDMI is the only interface in consumer electronics that can carry both uncompressed high-definition video and uncompressed multi-channel audio in all HD formats including 720p, 1080i, and 1080p. An uncompressed, all-digital signal translates into the highest quality video and audio, direct from the source.
* **Simple Connectivity:** A single, all-digital cable connection means no more confusing wires and no more mess. With an HDMI connection, you can connect any HD audio or video device to your system with a single input, for a streamlined system and no confusing
* **Integrated Control:** With an HDMI connection, it’s easy to integrate all your audio and visual devices into a single remote control. HDMI has built-in sensors that automatically recognize and configure your AV devices on demand. With the click of a button, your remote control knows which components to turn on to watch a DVD, listen to music, or watch TV programmes.
* **Automatic Format Adjustment:** An HDMI connection automatically configures the visual format of your display to match the content. Whether you’re watching TV programmes, sports, films, or video games, you’ll see everything as it’s meant to be seen.
* **PC Compatibility:** HDMI PC-compatibility makes it easy to connect your PC to your HDTV, so you can display PC gaming or PC entertainment contents.

**ADVANTAGES OF HDMI**

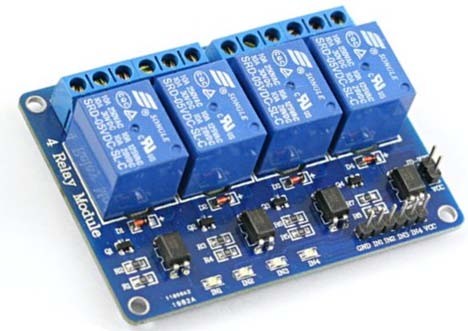
* **Higher Quality:** HDMI enables loss-less transmission and better quality video at low brightness scenes at higher resolutions. So, the video quality is much better than their analog counterparts as there is no conversion involved either. High contrast details like text etc. are displayed more sharper.
* **Intelligence:** Two way communications between video sources are enabled by HDMI Interface, which enables automatic configuration (between 480p or 720p,16:9 or 4:3 for example). So, external intervention to identify the best resolutions and audio formats is minimized. This is accomplished by using a standard known as EDID – Extended Display Identification Data.
* **Authentication and Encryption:** HDMI standard supports authentication to ensure that the devices are authorized to receive the content sent by the HDMI enabled sources. They also enable authentication to make sure that people cannot tap in to the cables to copy or pirate content sent through them.
* **Signal Integrity:** They enable digital signals to be stored, transmitted and viewed without changes from the original (unlike other media which require analog to digital conversion) and hence the signal degradation is not prominent. So, it is better to transmit HD content.
* **Single Cable:** This is truly a single cable solution as there is only a single cable that carries audio, video and control information. So, the complexity of implementing an audio video control system is lesser.
* **Deep Colors:** HDMI supports 10 bit, 12 bit and 16 bit color depths which can render over one billion colors in good detail.
* **No Compression:** Since HD signals are not compressed while transmission, there is no de-gradation in signal quality.
* **Compatibility with DVI:** Since it is backward compatible with DVI interface, DVI enabled PC’s can send HD content to display devices.
* **Supports multiple audio and video formats:** Multiple audio and video formats like standard stereo, multi channel surround sound, 720p, 1080i, 1080p, NTSC,PAL etc. are supported by HDMI standard.
* **Hot Plug Detect:** The sink device (display unit) can indicate its presence to a source with a hot plug detect signal to identify when a cable has been connected and to start authentication.
* **CEC:** Consumer Electronics Control is a communication link that enables devices connected via HDMI to talk to each other. For example, multiple DVD players from a single manufacturer can communicate with each other so that only one plays at a time. Some video sources can also send a power-off signal to turn off certain displays connected to it.
* **Display port compatibility:** Display port is a parallel technology for audio/video interface like HDMI and HDMI devices are compatible with Display Port interfaces.
* **Dolby/DTS:** HDMI supports Dolby/DTS-8 channel audio streams for highest quality.

**DISADVANTAGES OF HDMI**

* **Distance Limitations:** Maximum distance for HDMI Cat1 cables is up to 35meters (approx) for full capacity and maximum distance for HDMI Cat2 cables is up to 10 meters (approx) for full capacity. Beyond this limit, they need extenders. There are extenders like UTP cable extenders for HDMI, HDMI cable extenders, Fiber extenders for HDMI and Coaxial extenders for HDMI. Even Switches, Distribution amplifiers, audio/video processors act as repeaters. But extending HDMI cables this way has its limitations.
* **Switching Delays:** Sometimes, blank screens can be caused due to authentication delays. There can also be screen flashing errors.
* **Field Termination:** There are limitations to field terminations of HDMI cables. They cannot be easily terminated in the field like their analog counterparts.
* **Costly:** DMI cables are more expensive (per meter) than their analog counterparts.
* **Multiple locations:** They are difficult to run in multiple applications spanning various locations. There are more complexities in such situations than just the distance limitations.
* **EDID:** One such complexity is with EDID(Extended Display Identification Data) which is used in automatic identification of resolutions and audio formats. Both HDMI and EDID specifications offer limited guidance in multiple location deployment scenarios.

**3.6 RELAY**

This is a LOW Level 5V 4-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V/10A or DC30V/10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller.

****

**Figure 3.14 4Channel Relay Module**

**Features**

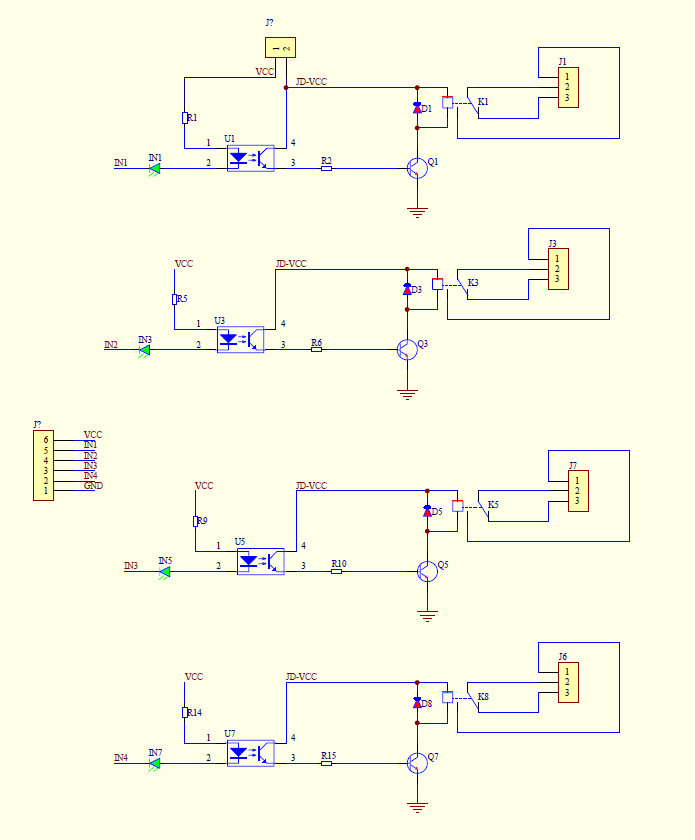
* Relay Maximum output: DC 30V/10A, AC 250V/10A.
* 4 Channel Relay Module with Opto-coupler.
* Standard interface that can be controlled directly by microcontroller ( 8051, AVR, PIC, DSP, ARM, ARM, MSP430, TTL logic).
* Relay of high quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.
* Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller

**3.6.2 SCHEMATIC DIAGRAM OF 4CHANNEL RELAY MODULE**

Vcc and RY-Vcc are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-Vcc to supply the relay; connect Vcc to 5V of the MCU board to supply input signals.

**NOTE**: If you want complete optical isolation, connect "Vcc" to Raspberry Pi +5 volts but do NOT connect Raspberry Pi Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils.

If relay isolation is enough for your application, connect Raspberry Pi +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.

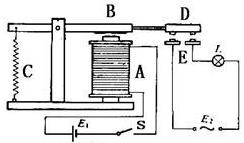


**Figure 3.15 Schematic 4Channel Relay Module**

It is sometimes possible to use this relay boards with 3.3V signals, if the JD-Vcc (Relay Power) is provided from a +5V supply and the Vcc to JD-Vcc jumper is removed. That 5V relay supply could be totally isolated from the 3.3V device, or have a common ground if opto-isolation is not needed. If used with isolated 3.3V signals, Vcc (To the input o f the opto-isolator, next to the IN pins) should be connected to the 3.3V device's +3.3V supply.

**3.6.3 OPERATING PRINCIPLE**

A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is on as shown in Figure 3.16.



**Figure 3.16 Relay Diagram**

Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts results in power on and off of the circuit.

**Input**

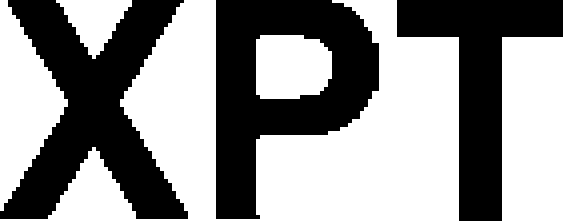
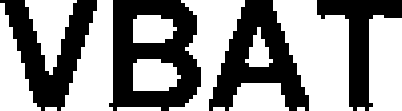
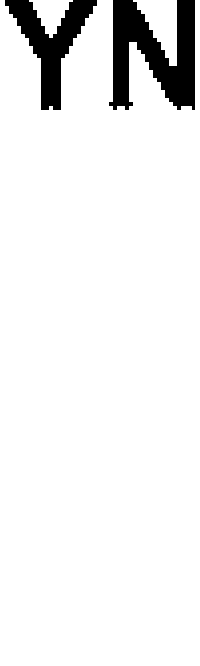
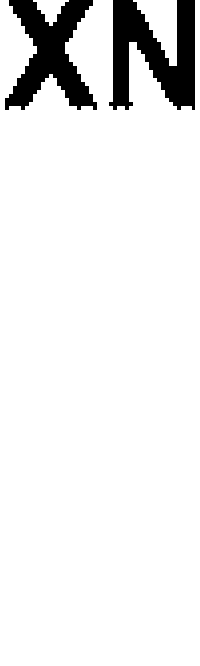
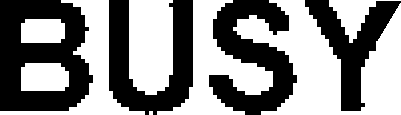
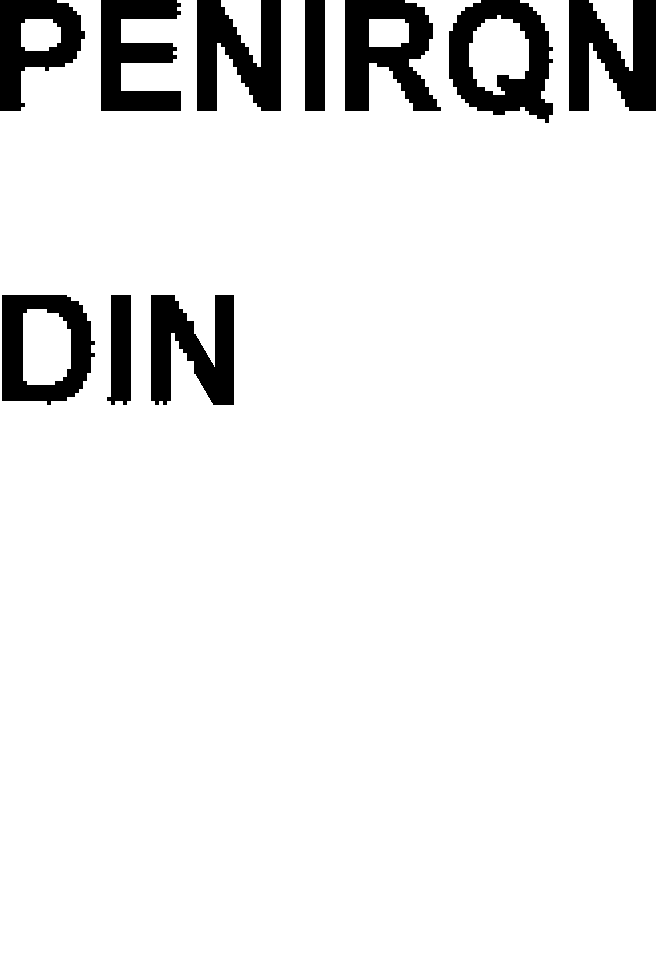
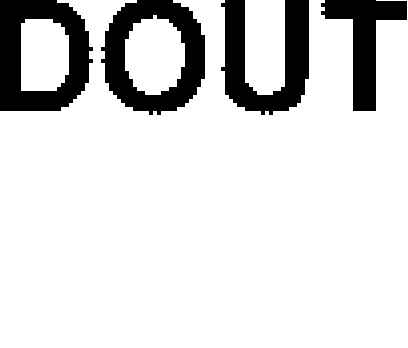
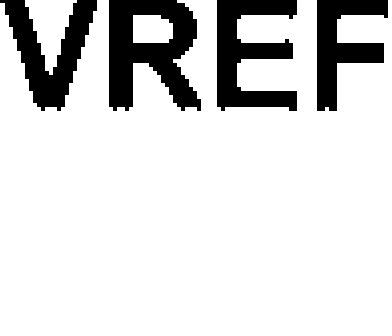
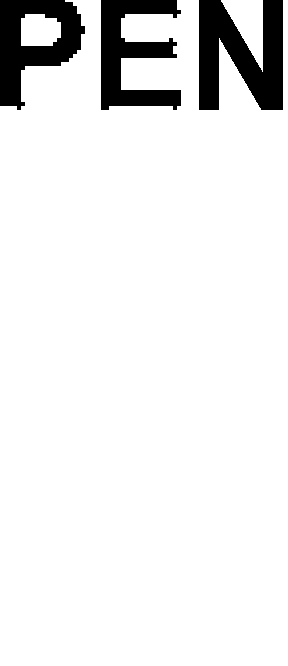
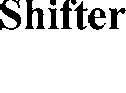
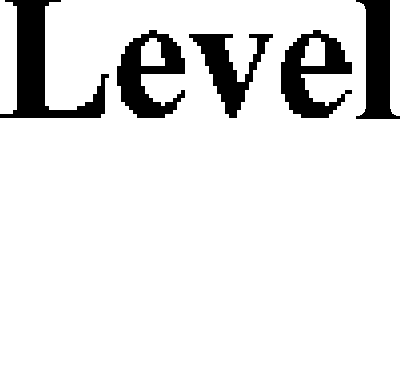
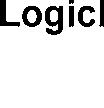
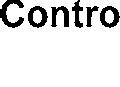
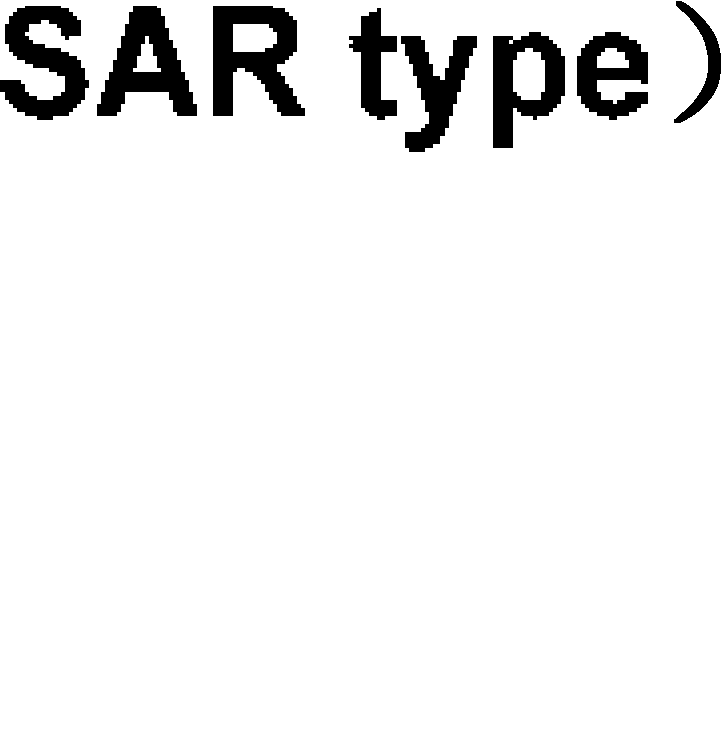
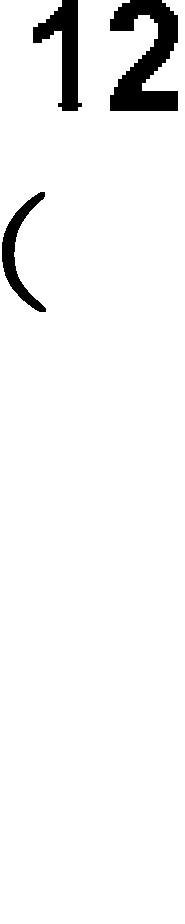
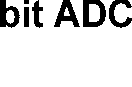
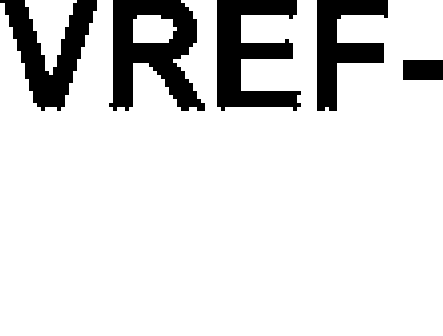
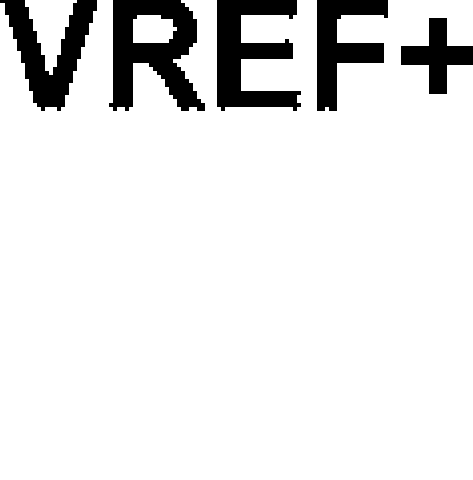
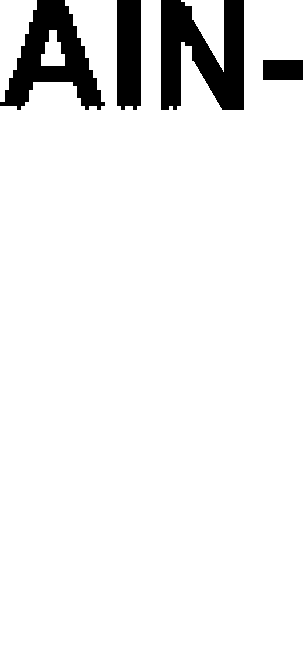
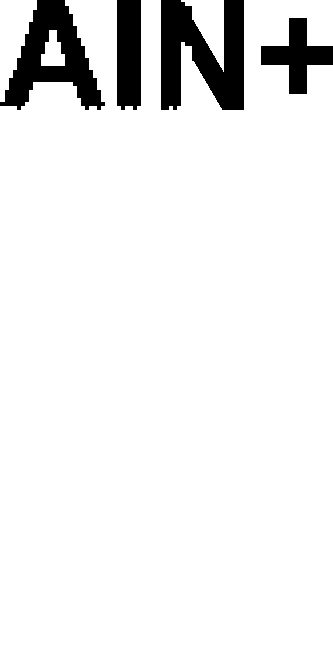
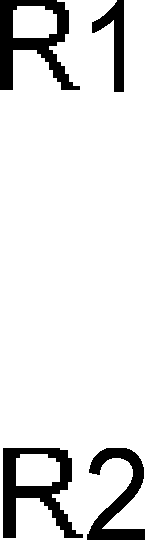
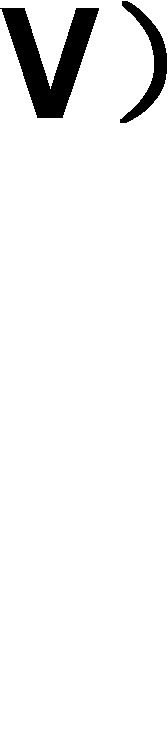
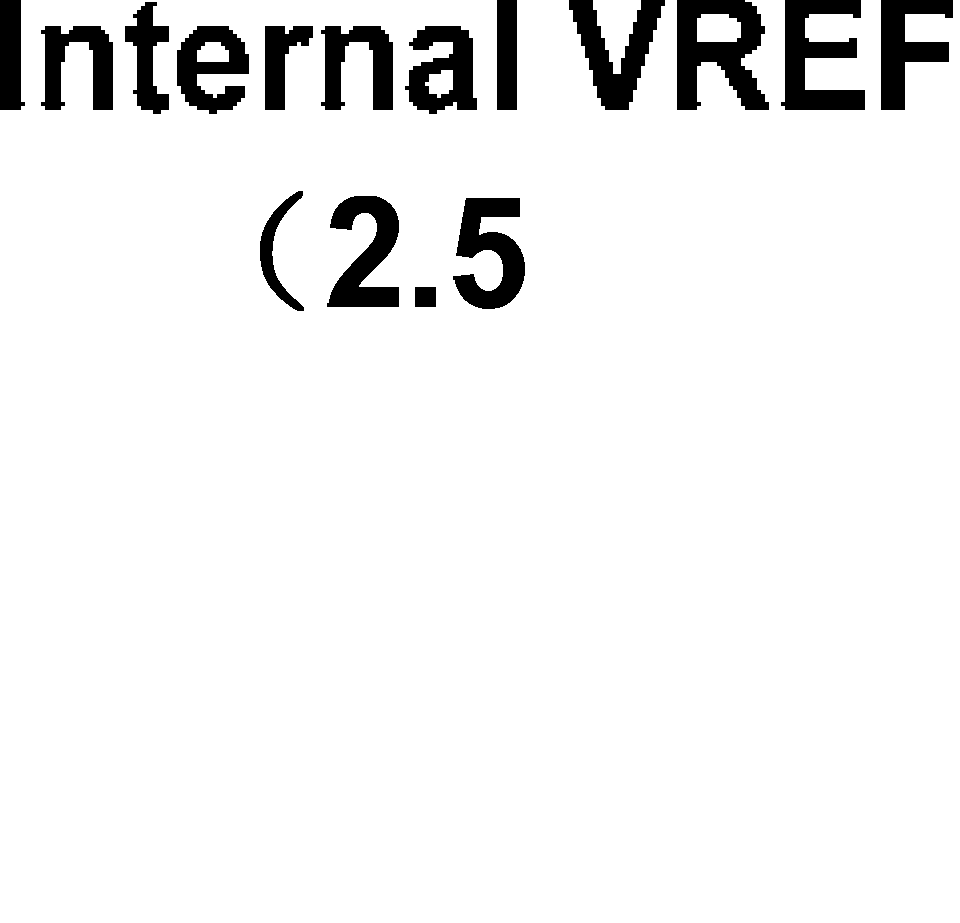
* Vcc : Connected to positive supply voltage (supply power according to relay voltage) GND : Connected to supply ground.
* IN1: Signal triggering terminal 1 of relay module
* IN2: Signal triggering terminal 2 of relay module
* IN3: Signal triggering terminal 3 of relay module
* IN4: Signal triggering terminal 4 of relay module

**Output**

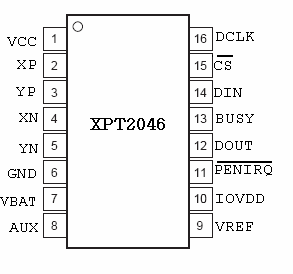
Each module of the relay has one NC (normally close), one NO (normally open) and one COM (Common) terminal. So there are 4 NC, 4 NO and 4 COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

**3.7 TOUCH SCREEN CONTROLLER (XPT2046)**

The XPT2046 is a 4-wire resistive touch screen controller that incorporates 12-bit 125 kHz sampling SAR type A/D converter. It operates down to 2.2V supply voltage and supports digital I/O interface voltage from 1.5V to VCC in order to connect low voltage up. The controller can detect the pressed screen location by performing two A/D conversions. In addition to location, the XPT2046 also measures touch screen pressure. On-chip Vref can be utilized for analog auxiliary input, temperature measurement and battery monitoring with the ability to measure voltage from 0V to 5V.The XPT2046 is available in 16pin QFN thin package (0.75mm in height) and has the operating temperature range of -40°C to 85°C.



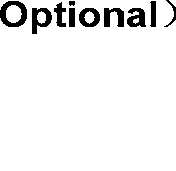
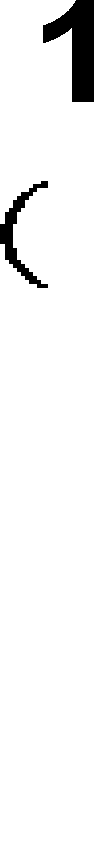
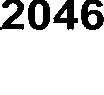
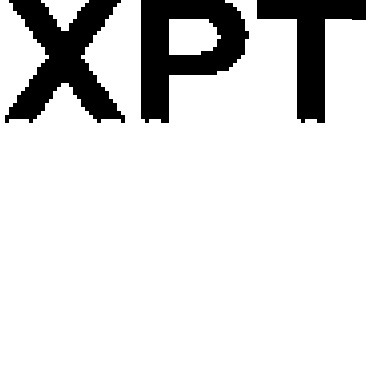
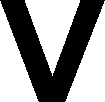
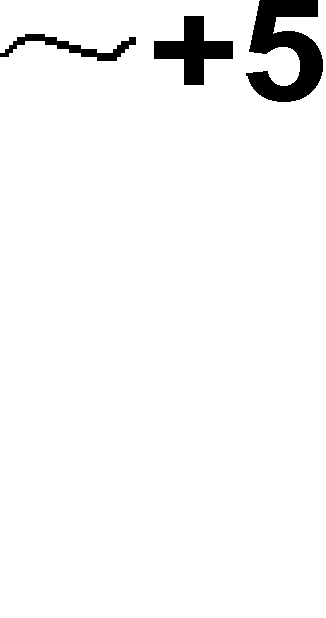
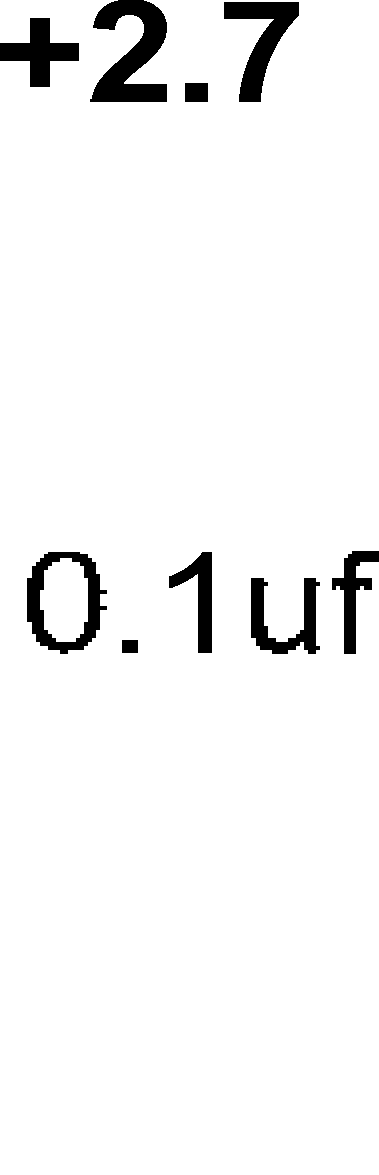
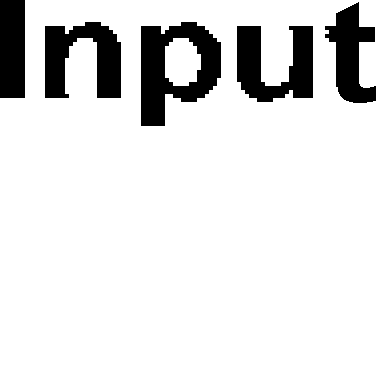
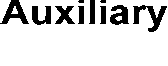
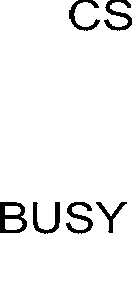
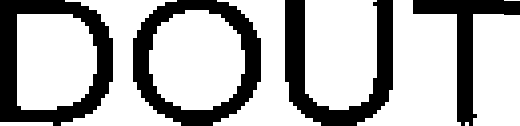
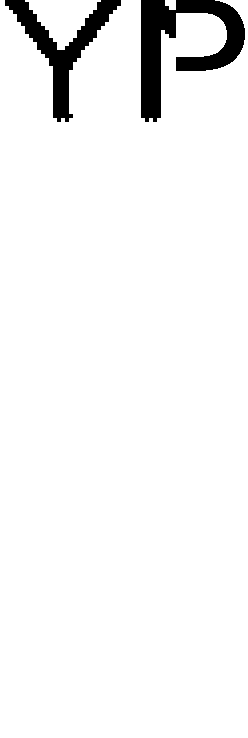
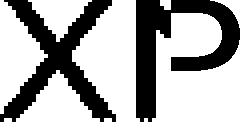
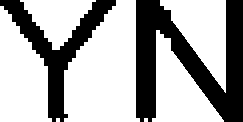
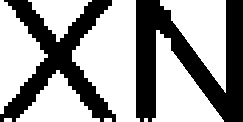
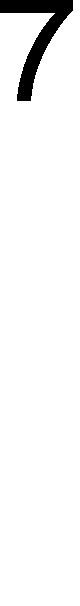
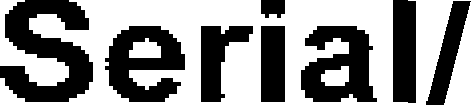
**Figure 3.17 Block Diagram of XPT2046**



**Figure 3.18 Pin Diagram of XPT2046**

**3.7.1 THEORY OF OPERATION**

****The XPT2046 is a classic successive approximation register (SAR) analog-to-digital converter (ADC). The architecture is based on capacitive redistribution, which inherently includes a sample-and-hold function. The converter is fabricated



**Figure 3.19 Basic Operation of XPT2046**

on a 0.6μm CMOS process. The basic operation of the XPT2046 is shown in Figure 3.19 The device features an internal 2.5V reference and uses an external clock. Operation is maintained from a single supply of 2.7V to 5.25V. The internal reference can be overdriven with an external, low-impedance source between 1V and +Vcc. The value of the reference voltage directly sets the input range of the converter. The analog input (X-, Y-, and Z-Position coordinates, auxiliary input, battery voltage, and chip temperature) to the converter is provided via a multiplexer. A unique configuration of low on-resistance touch panel driver switches allows an unselected ADC input channel to provide power and the accompanying pin to provide ground for an external device, such as a touch screen. By maintaining a differential input to the converter and differential reference architecture, it is possible to negate the error from each touch panel driver switch’s on-resistance (if this is a source of error for the particular measurement).

## 3.7.2 TOUCH SCREEN SETTLING

In some applications, external capacitors may be required across the touch screen for filtering noise picked up by the touch screen (e.g., noise generated by the LCD panel or backlight circuitry). These capacitors provide a low-pass filter to reduce the noise, but cause a settling time requirement when the panel is touched that typically shows up as a gain error. There are several methods for minimizing or eliminating this issue. The problem is that the input and/or reference has not settled to the final steady-state value prior to the ADC sampling the input(s) and providing the digital output. Additionally, the reference voltage may still be changing during the measurement cycle. Option 1 is to stop or slow down the XPT2046 DCLK for the required touch screen settling time. This allows the input and reference to have stable values for the Acquire period (3 clock cycles of the XPT2046; This works for both the single-ended and the differential modes. Option 2 is to operate the XPT2046 in the differential mode only for the touch screen measurements and command the XPT2046 to remain on (touch screen drivers ON) and not go into power-down (PD0 = 1). Several conversions are made depending on the settling time required and the XPT2046 data rate. Once the required number of conversions has been made, the processor commands the XPT2046 to go into its power-down state on th40e last measurement. This process is required for X-Position,Y-Position, and Z-Position measurements. Option 3 is to operate in the 15 Clock-per-Conversion mode, which overlaps the analog-to-digital conversions and maintains the touch screen drivers on until commanded to stop by the processor

**3.7.3 DIGITAL INTERFACE**

The typical operation of the XPT2046 digital interface. This diagram assumes that the source of the digital signals is a microcontroller or digital signal processor with a basic serial interface. Each communication between the processor and the converter, such as SPI (Serial Peripheral Interface), SSI, (Synchronous Serial Interface), consists of eight clock cycles. One complete conversion can be accomplished with three serial communications for a total of 24 clock cycles on the DCLK input.

The first eight clock cycles are used to provide the control byte via the DIN pin. When the converter has enough information about the following conversion to set the input multiplexer and reference inputs appropriately, the converter enters the acquisition (sample) mode and, if needed, the touch panel drivers are turned on. After three more clock cycles, the control byte is complete and the converter enters the conversion mode. At this point, the input sample-and-hold goes into the hold mode and the touch panel drivers turn off (in single-ended mode). The next 12 clock cycles accomplish the actual analog to digital conversion. If the conversion is ratio metric(SER/DFR= 0), the drivers are on during the conversion and a 13th clock cycle is needed for the last bit of the conversion result. Three more clock cycles are needed to complete the last byte (DOUT will be low), which are ignored by the converter.

##### **CONTROL BYTE**

The control byte (on DIN), as shown in Table 3.2, provides the start conversion, addressing, ADC resolution, configuration, and power-down of the XPT2046. Table 3.3 gives detailed information regarding the order and description of these control bits within the control byte.

**INITIATE START**—The first bit, the S bit, must always be high and initiates the start of the control byte. The XPT2046 ignores inputs on the DIN pin until the start bit is detected.

**ADDRESSING**—The next three bits (A2, A1, and A0) select the active input channel(s) of the input multiplexer (Table 3.2, Table 3.3), touch screen drivers, and the reference inputs.

**MODE**—The mode bit sets the resolution of the ADC. With this bit low, the next conversion has 12 bits of resolution, whereas with this bit high, the next conversion has eight bits of resolution.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **BIT7(MSB)** | **BIT 6** | **BIT 5** | **BIT 4** | **BIT 3** | **BIT2** | **BIT 1** | **BIT 0(LSB)** |
| **S** | **A2** | **A1** | **A0** | **MODE** | **SER/DFR** | **PD1** | **PD0** |

**SER/DFR**—The SER/DFR bit controls the reference mode, either single-ended (high) or differential (low). The differential mode is also referred to as the ratio metric conversion mode and is preferred for X-Position,Y-Position, and Pressure-Touch measurements for optimum performance. The reference is derived from the voltage at the switch drivers, which is almost the same as the voltage to the touch screen. In this case, a reference voltage is not needed as the reference voltage to the ADC is the voltage across the touch screen. In the single-ended mode, the converter reference voltage is always the difference between the VREF and GND pins.

**Table 3.2 Order of the Control Bits in the Control Byte**

|  |  |  |
| --- | --- | --- |
| **BIT** | **NAME** | **DESCRIPTION** |
| 7 | S | Start bit. Control byte starts with first high bit on DIN.A new control byte can start every 15th clock cycle in 12-bit conversion mode or every 11th clock cycle in 8-bit conversion mode. |
| 6-4 | A2-A0 | Channel Select bits. Along with the SER/DFR bit, these bits control the setting of the multiplexer input, touch driver switches, and reference inputs. |
| 3 | MODE | 12-Bit/8-Bit Conversion Select bit. This bit controls the number of bits for the next conversion: 12-bits(low) or 8-bits (high). |
| 2 | SER/DFR | Single-Ended/Differential Reference Select bit. Along with bits A2-A0, this bit controls the setting of the multiplexer input, touch driver switches, and reference inputs. |
| 1-0 | PD1-PD0 | Power-Down Mode Select bits. |

**Table 3.3 Descriptions of the Control Bits within the Control Byte**

**3.8 SERIAL PERIPHERAL INTERFACE (SPI)**

SPI is a general-purpose synchronous serial interface. During an SPI transfer, transmit and receive data is simultaneously shifted out and in serially. A serial clock line synchronizes the shifting and sampling of the information on two serial data lines.

Motorola created the SPI port in the mid 1980’s to use in their microcontroller product families. The SPI is mainly used to allow a microcontroller to communicate with peripheral devices such as EPROMs.

SPI devices communicate using a master-slave relationship. Due to its lack of built-in device addressing, SPI requires more effort and more hardware resources than I2C when more than one slave is involved. But SPI tends to be simpler and more efficient than I2C in point-to-point (single master, single slave) applications for the very same reason; the lack of device addressing means less overhead.

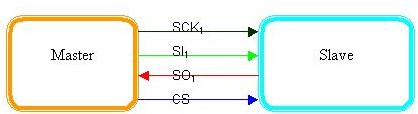
**3.8.1 SPI DETAIL**

SPI is a serial bus standard established by Motorola and supported in silicon products from various manufacturers. SPI interfaces are available on popular communication processors and microcontrollers. It is a synchronous serial data link that operates in full duplex (signals carrying data go in both directions simultaneously).

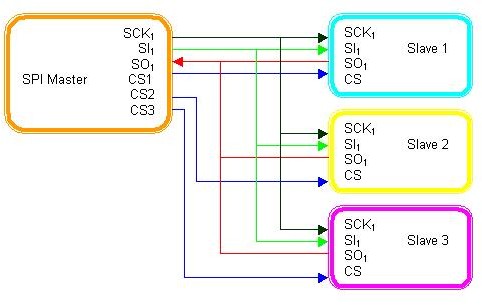
Devices communicate using a master/slave relationship, in which the master initiates the data frame. When the master generates a clock and selects a slave device, data may be transferred in either or both directions simultaneously. In fact, as far as SPI is concerned, data are always transferred in both directions. It is up to the master and slave devices to know whether a received byte is meaningful or not.

So a device must discard the received byte in a "transmit only" frame or generate a dummy byte for a "receive only" frame.

SPI specifies four signals: clock (SCK1); master data output, slave data input (SI1); master data input, slave data output (SO1); and chip select (CS). Figure 1 shows these signals in a single-slave configuration. SCK1 is generated by the master and input to all slaves. SI1 carries data from master to slave. SO1 carries data from slave back to master. A slave device is selected when the master asserts its CS signal.



**Figure 3.20 Single master, single slave SPI implementation**



**Figure 3.21 Single master, multiple slave SPI implementation**

If multiple slave devices exist, the master generates a separate slave select signal for each slave. These relationships are illustrated in Figure 3.21. The master generates slave select signals using general-purpose discrete input/output pins or other logic. This consists of old- fashioned bit banging and can be pretty sensitive. You have to time it relative to the other signals and ensure, for example, that you don't toggle a select line in the middle of a frame.

While SPI doesn't describe a specific way to implement multi-master systems, some SPI devices support additional signals that make such implementations possible. However, it's complicated and usually unnecessary, so it's not often done.

A pair of parameters called clock polarity (CPOL) and clock phase (CPHA) determines the edges of the clock signal on which the data are driven and sampled. Each of the two parameters has two possible states, which allows for four possible combinations, all of which are incompatible with one another. So a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are used that are fixed in different configurations, the master will have to reconfigure itself each time it needs to communicate with a different slave.

SPI does not have an acknowledgement mechanism to confirm receipt of data. In fact, without a communication protocol, the SPI master has no knowledge of whether a slave even exists. SPI also offers no flow control. If you need hardware flow control, you might need to do something outside of SPI.

Slaves can be thought of as input/output devices of the master. SPI does not specify a particular higher-level protocol for master-slave dialog. In some applications, a higher-level protocol is not needed and only raw data are exchanged. An example of this is an interface to a simple codec. In other applications, a higher-level protocol, such as a command-response protocol, may be necessary. Note that the master must initiate the frames for both its command and the slave's response.

**3.8.2 DATA AND CONTROL LINES OF THE SPI**

The SPI requires two control lines (CS and SCK) and two data lines (SI and SO).With CS (Chip-Select) the corresponding peripheral device is selected. This pin is mostly active-low. In the unselected state the SO lines are hi-Z and therefore inactive. The master decides with which peripheral device it wants to communicate. The clock line SCLK is brought to the device whether it is selected or not. The clock serves as synchronization of the data communication.

The majority of SPI devices provide these four lines. Sometimes it happens that SDI and SDO are multiplexed, for example in the temperature sensor LM74 from National Semiconductor, or that one of these lines is missing. A peripheral device which must or can not be configured, requires no input line, only a data output. As soon as it gets selected it starts sending data. In some ADCs therefore the SDI line is missing (e.g. MCCP3001 from Microchip).

There are also devices that have no data output. For example LCD controllers (e.g. COP472-3 from National Semiconductor), which can be configured, but cannot send data or status messages.

**3.8.3 SPI CONFIGURATION**

Because there is no official specification, what exactly SPI is and what not, it is necessary to consult the data sheets of the components one wants to use. Important are the permitted clock frequencies and the type of valid transitions.

There are no general rules for transitions where data should be latched. Although not specified by Motorola, in practice four modes are used. These four modes are the combinations of CPOL and CPHA. In table 3.4, the four modes are listed.

|  |  |  |
| --- | --- | --- |
| SPI-mode | **CPOL** | **CPHA** |
| **0** | **0** | **0** |
| **1** | **0** | **1** |
| **2** | **1** | **0** |
| **3** | **1** | **1** |

**Table3.4 SPI Modes**

If the phase of the clock is zero, i.e. CPHA = 0, data is latched at the rising edge of the clock with CPOL = 0, and at the falling edge of the clock with CPOL = 1. If CPHA = 1, the polarities are reversed. CPOL = 0 means falling edge, CPOL = 1 rising edge.

The micro controllers from Motorola allow the polarity and the phase of the clock to be adjusted. A positive polarity results in latching data at the rising edge of the clock. However data is put on the data line already at the falling edge in order to stabilize. Most peripherals which can only be slaves, work with this configuration. If it should become necessary to use the other polarity, transitions are reversed.

**ADVANTAGES**

* Full duplex communication in the default version of this protocol
* Push-pull drivers (as opposed to open drain) provide good signal integrity and high speed
* Higher throughput than I²C or SMBus. Not limited to any maximum clock speed, enabling potentially high speed
* Complete protocol flexibility for the bits transferred
* Not limited to 8-bit words
* Arbitrary choice of message size, content, and purpose
* Extremely simple hardware interfacing
* Typically lower power requirements than I²C or SMBus due to less circuitry (including pull up resistors)
* No arbitration or associated failure modes - unlike CAN-bus
* Slaves use the master's clock and do not need precision oscillators
* Slaves do not need a unique address – unlike I²C or GPIB or SCSI
* Transceivers are not needed - unlike CAN-bus
* Uses only four pins on IC packages, and wires in board layouts or connectors, much fewer than parallel interfaces
* At most one unique bus signal per device (chip select); all others are shared
* Signals are unidirectional allowing for easy galvanic isolation
* Simple software implementation

**DISADVANTAGES**

* Requires more pins on IC packages than I²C, even in the three-wire variant
* No in-band addressing; out-of-band chip select signals are required on shared buses
* No hardware flow control by the slave (but the master can delay the next clock edge to slow the transfer rate)
* No hardware slave acknowledgment (the master could be transmitting to nowhere and not know it)
* Typically supports only one master device (depends on device's hardware implementation)
* No error-checking protocol is defined
* Without a formal standard, validating conformance is not possible
* Only handles short distances compared to RS-232, RS-485, or CAN-bus. (Its distance can be extended with the use of transceivers like RS-422.)
* Many existing variations, making it difficult to find development tools like host adapters that support those variations
* SPI does not support hot swapping (dynamically adding nodes).
* Interrupts must either be implemented with out-of-band signals or be faked by using periodic polling similarly to USB 1.1 and 2.0.
* Some variants like dual SPI, quad SPI, and three-wire serial buses defined below are half-duplex.

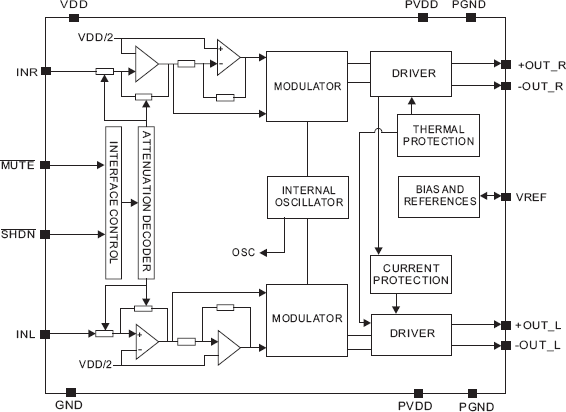
**3.9 AUDIO AMPLIFIER (PAM8403)**

The PAM8403 is a 3W, class-D audio amplifier. It offers low THD+N, allowing it to achieve high-quality sound reproduction. The new filter less architecture allows the device to drive the speaker directly, requiring no low-pass output filters, thus saving system cost and PCB area.

With the same numbers of external components, the efficiency of the PAM8403 is much better than that of Class-AB cousins. It can extend the battery life, which makes it well-suited for portable applications. The PAM8403 is available in SOP-16 package.

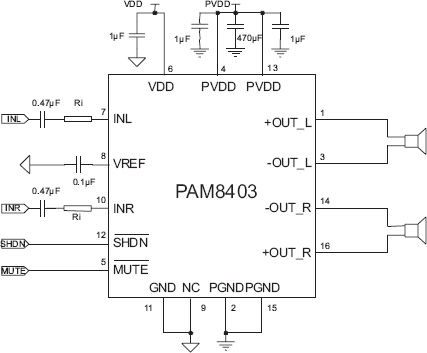
**3.9.1 APPLICATIONS**

* LCD Monitors / TV Projectors
* Notebook Computers

**3.9.2 BLOCK DIAGRAM**

**Figure 3.22 Functional Block Diagram of PAM8403**

**3.9.3 TYPICAL APPLICATIONS CIRCUIT**



**Figure 3.23 Pin Diagram**

**3.10 AC POWER ADAPTER**

This compact USB AC charger is UL and CSA listed and incorporates a highly regulated output switching power design to assure a constant, accurate DC power delivery to your sensitive portable 5V devices.

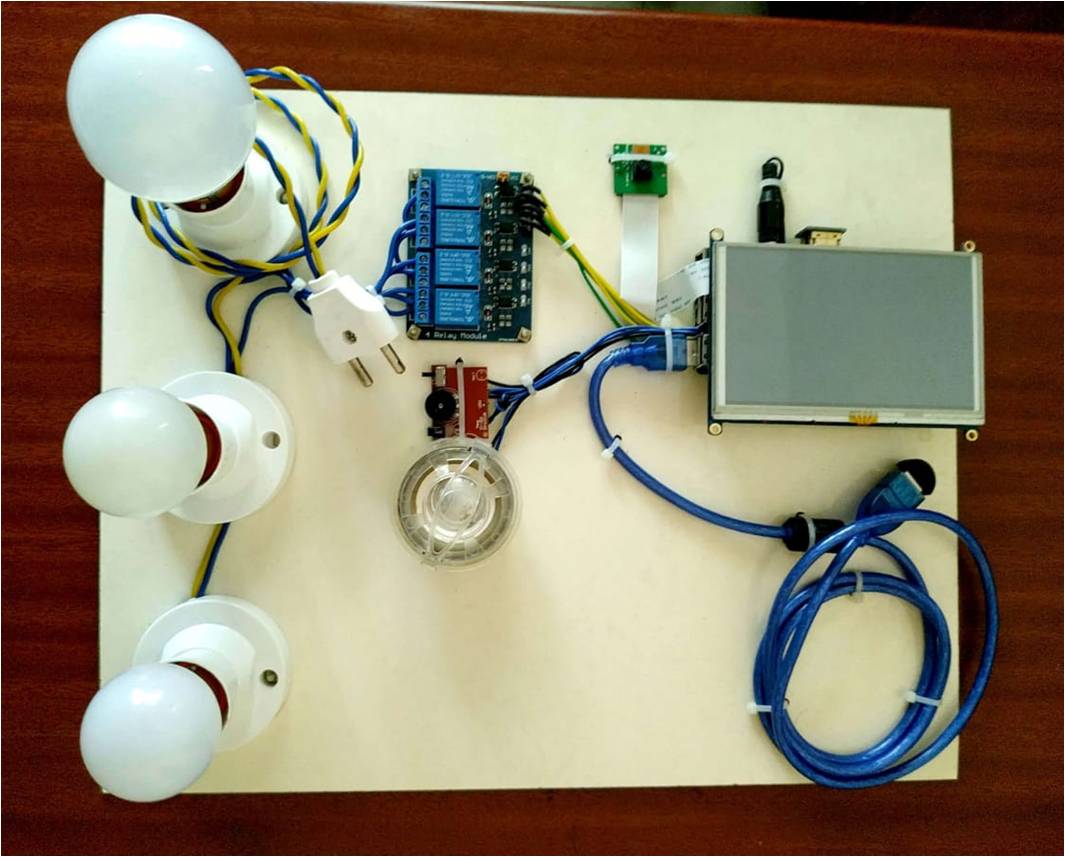
**Figure 3.24 Power Adapter**

**Features**

* Extremely high efficiency (up to 90%)
* Close output regulation ensures precise 5V DC output
* No-load voltage regulation within 3% of rated output
* Extremely compact and lightweight housing

**3.10.1 SPECIFICATIONS**

* Input voltage : 100V AC to 240V AC
* Load regulation : 3%
* Output voltage/capacity : 5.1V; 2.5A
* Output cord length : 4 Feet
* Output cord connector : Micro USB B type



**Figure 3.25 Hardware Implemented Circuit**

**3.11 CONCLUSION**

Thus suitable hardware was selected and designed to implement the prototype.

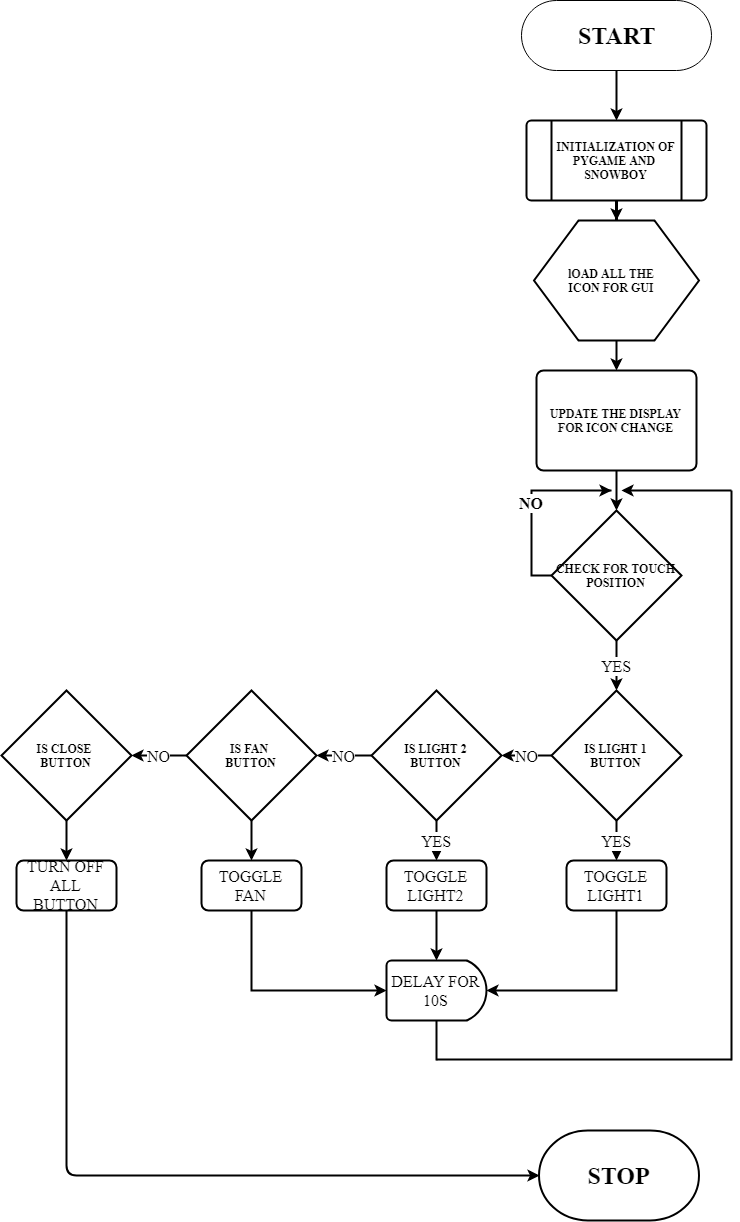
**CHAPTER 4**

**SOFTWARE IMPLEMENTATION**

**4.1 INTRODUCTION**

This chapter gives the detailed explanation of programming language used and represents how the programs are implemented. Python is the main programming language used in this project**.**

**4.1.1 FLOW CHART**



**FIGURE 4.1 Flow Chart of Main GUI**

Figure 4.1 explains about the main program flow of Graphical User Interface.

**4.1.2** **PROGRAM FOR GRAPHICAL USER INTERFACE**

import pygame

import time

import datetime

import os

import timer

import wallpaper

import camerapreview

x = -2

y = 0

os.environ['SDL\_VIDEO\_WINDOW\_POS'] = "%d,%d" % (x,y)

######## variable decleration and image loading #############

pygame.init()

display\_width = 800

display\_height = 480

black = (64,64,64)

white = (255,255,255)

line=(255,204,255)

red = (200,0,0)

bright\_red=(255,0,0)

light\_blue=(0,128,255)

bright\_green=(0,255,0)

violet=(147,112,219)

values=[5,5,5,10]

Display = pygame.display.set\_mode((display\_width,display\_height))

pygame.display.set\_caption('control panel')

icon = pygame.image.load("icon.png")

pygame.display.set\_icon(icon)

pygame.display.update()

clock = pygame.time.Clock()

pygame.mouse.set\_visible(False)

button1\_img= pygame.image.load("light.png").convert\_alpha()

button2\_img= pygame.image.load("fan.png").convert\_alpha()

button3\_img= pygame.image.load("plug.png").convert\_alpha()

button4\_img= pygame.image.load("microphone.png").convert\_alpha()

button5\_img= pygame.image.load("camera.png").convert\_alpha()

button6\_img= pygame.image.load("timer.png").convert\_alpha()

button7\_img= pygame.image.load("lamp.png").convert\_alpha()

button8\_img= pygame.image.load("Ac.png").convert\_alpha()

button9\_img= pygame.image.load("frame.png").convert\_alpha()

button10\_img= pygame.image.load("consumption.png").convert\_alpha()

rocket=pygame.image.load("rocket.png").convert\_alpha()

main=True

light=True

light1\_toggle=False

fan\_toggle=False

plug\_toggle=False

voice\_toggle=False

camera\_toggle=False

light2\_toggle=False

Ac\_toggle=False

rocket\_y=250

events=None

position=None

############## button creation and main gui function ###############

def text\_objects(text, font):

textSurface = font.render(text, True, white)

return textSurface, textSurface.get\_rect()

def get\_events():

global events

events=pygame.event.get()

def touch\_pos():

for event in events:

if event.type == pygame.MOUSEBUTTONDOWN:

return list(event.pos)

def quit\_window():

for event in events:

if event.type == pygame.QUIT:

pygame.quit()

quit()

def button(msg,x,y,w,h,ic,ac,action=None):

if type(position)==list and x+w > position[0] > x and y+h > position[1] > y and action!=None:

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

action(msg)

elif light1\_toggle==True and msg=="light1":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif fan\_toggle==True and msg=="fan":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif plug\_toggle==True and msg=="plug":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif voice\_toggle==True and msg=="voice":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif camera\_toggle==True and msg=="camera":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif timer.timer\_toggle==True and msg=="timer":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif light2\_toggle==True and msg=="light2":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

elif Ac\_toggle==True and msg=="Ac":

pygame.draw.rect(Display, bright\_green,(x,y,w,h))

else:

pygame.draw.rect(Display, ic,(x,y,w,h))

smallText = pygame.font.Font("COMIC.TTF",25)

textSurf, textRect = text\_objects(msg, smallText)

textRect.center = ( (x+(w/2)), (y+(h-19)) )

Display.blit(textSurf, textRect)

def sidewall():

global rocket\_y

h\_y=rocket\_y+53

a\_y=rocket\_y+73

v\_y=rocket\_y+97

e\_y=rocket\_y+120

a1\_y=rocket\_y+150

n\_y=rocket\_y+180

i\_y=rocket\_y+204

c\_y=rocket\_y+228

e1\_y=rocket\_y+250

d\_y=rocket\_y+ 280

a2\_y=rocket\_y+305

y\_y=rocket\_y+330

font = pygame.font.Font("design.graffiti.comicsansmsgras.ttf",25)

h=font.render("H", True, white)

a=font.render("A", True, white)

v=font.render("V", True, white)

e=font.render("E", True, white)

n=font.render("N", True, white)

i=font.render("I", True, white)

c=font.render("C", True, white)

d=font.render("D", True, white)

y=font.render("Y", True, white)

pygame.draw.rect(Display,violet,(722,10,68,305))

Display.blit(rocket,[725,rocket\_y])

if h\_y<290:

Display.blit(h,[746,h\_y])

if a\_y<290:

Display.blit(a,[746,a\_y])

if v\_y<290:

Display.blit(v,[746,v\_y])

if e\_y<290:

Display.blit(e,[746,e\_y])

if e\_y<290:

Display.blit(e,[746,e\_y])

if a1\_y<290:

Display.blit(a,[746,a1\_y])

if n\_y<290:

Display.blit(n,[746,n\_y])

if i\_y<290:

Display.blit(i,[746,i\_y])

if c\_y<290:

Display.blit(c,[746,c\_y])

if e1\_y<290:

Display.blit(e,[746,e1\_y])

if d\_y<290:

Display.blit(d,[746,d\_y])

if a2\_y<290:

Display.blit(a,[746,a2\_y])

if y\_y<300:

Display.blit(y,[746,y\_y])

if rocket\_y<=-300:

rocket\_y=250

rocket\_y-=2

def home():

pygame.draw.rect(Display,violet,(366,318,424,151))

currentime = datetime.datetime.time(datetime.datetime.now())

Date=datetime.date.today().strftime("%A")[:3]+" /"+datetime.date.today().strftime("%B")[:3]+" /"+str(datetime.date.today().strftime("%d"))

font = pygame.font.Font(None, 60)

time1= font.render(currentime.strftime("%I:%M %p"), 1, (255,255,255))

date=font.render(Date, True, (255,255,255))

font = pygame.font.Font(None, 90)

text= font.render("LIVING ROOM", True, white)

Display.blit(time1,[505,320])

Display.blit(date,[480,365])

Display.blit(text,[370,410])

##################### function for buttton ############

def photoframe(msg):

wallpaper.wallpaper\_window()

def on\_off\_time():

timer.timer\_window()

def cameraframe():

print(timer.time\_lst)

#camerapreview.cam\_window()

def game\_loop(msg):

global light1\_toggle

global fan\_toggle

global plug\_toggle

global voice\_toggle

global camera\_toggle

global light2\_toggle

global Ac\_toggle

if msg=="light1":

light1\_toggle=not light1\_toggle

if msg=="fan":

fan\_toggle=not fan\_toggle

if msg=="plug":

plug\_toggle=not plug\_toggle

if msg=="voice":

voice\_toggle=not voice\_toggle

if msg=="camera":

camera\_toggle=not camera\_toggle

cameraframe()

if msg=="timer":

on\_off\_time()

if msg=="light2":

light2\_toggle=not light2\_toggle

if msg=="Ac":

Ac\_toggle=not Ac\_toggle

######### main loop ##########

def main\_window():

while main==True:

global position

get\_events()

quit\_window()

position=touch\_pos()

Display.fill((255,127,80))

button("light1",10,10,175,151,light\_blue,bright\_green,game\_loop)

button("fan",188,10,175,151,light\_blue,bright\_red,game\_loop)

button("plug",366,10,175,151,light\_blue,bright\_green,game\_loop)

button("voice",544,10,175,151,light\_blue,bright\_green,game\_loop)

button("timer",10,164,175,151,light\_blue,bright\_green,game\_loop)

button("light2",188,164,175,151,light\_blue,bright\_green,game\_loop)

button("Ac",366,164,175,151,light\_blue,bright\_green,game\_loop)

button("camera",544,164,175,151,light\_blue,bright\_green,game\_loop)

button("frame",10,318,175,151,light\_blue,bright\_green,photoframe) button("consumption",188,318,175,151,bright\_green,bright\_green,game\_loop)

Display.blit(button1\_img,(30,15))

Display.blit(button2\_img,(220,14))

Display.blit(button3\_img,(390,10))

Display.blit(button4\_img,(570,13))

Display.blit(button5\_img,(574,168))

Display.blit(button6\_img,(20,170))

Display.blit(button7\_img,(215,165))

Display.blit(button8\_img,(390,165))

Display.blit(button9\_img,(40,320))

Display.blit(button10\_img,(218,322))

home()

sidewall()

pygame.display.update()

clock.tick(15)

pygame.quit()

quit()

if \_\_name\_\_ == "\_\_main\_\_":

main\_window()

**4.2 PYTHON**

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural.

**4.2.1 FEATURES AND PHILOSOPHY**

Python is a multi-paradigm programming language which is also a Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by meta programming and meta objects (magic methods)).Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution(late binding), which binds method and variable names during program execution.

The language's core philosophy is summarized in the document the Zen of Python (PEP 20), which includes aphorisms such as:

* Beautiful is better than ugly
* Explicit is better than implicit
* Simple is better than complex
* Complex is better than complicated
* Readability counts

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

While offering choice in coding methodology, the Python philosophy rejects exuberant syntax (such as that of Perl) in favor of a simpler, less-cluttered grammar. As Alex Martelli put it: "To describe something as 'clever' is not considered a compliment in the Python culture." Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favor of "there should be one—and preferably only one—obvious way to do it".

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name—a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic.

Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonists, Pythonistas, and Pythoneers.

**4.3 PYGAME**

Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used with the Python programming language. Pygame uses the Simple DirectMedia Layer (SDL) library, with the intention of allowing real-time computer game development without the low-level mechanics of the C programming language and its derivatives. This is based on the assumption that the most expensive functions inside games, can be abstracted from the game logic, making it possible to use a high-level programming language, such as Python, to structure the game.

Other features that SDL doesn't have include vector math, collision detection, 2d sprite scene graph management, MIDI support, camera, pixel array manipulation, transformations, filtering, advanced free type font support, and drawing.

**4.4 SNOWBOY, A CUSTOMIZABLE HOTWORD DETECTION ENGINE**

Snowboy is an highly customizable hotword detection engine that is embedded real-time and is always listening (even when off-line) compatible with Raspberry Pi, (Ubuntu) Linux, and Mac OS X.

A hotword (also known as wake word or trigger word) is a keyword or phrase that the computer constantly listens for as a signal to trigger other actions.

Some examples of hotword include “Alexa” on Amazon Echo, “OK Google” on some Android devices and “Hey Siri”on iPhones. These hotwords are used to initiate a full-fledged speech interaction interface. However, hotwords can be used in other ways too like performing simple command & control actions.

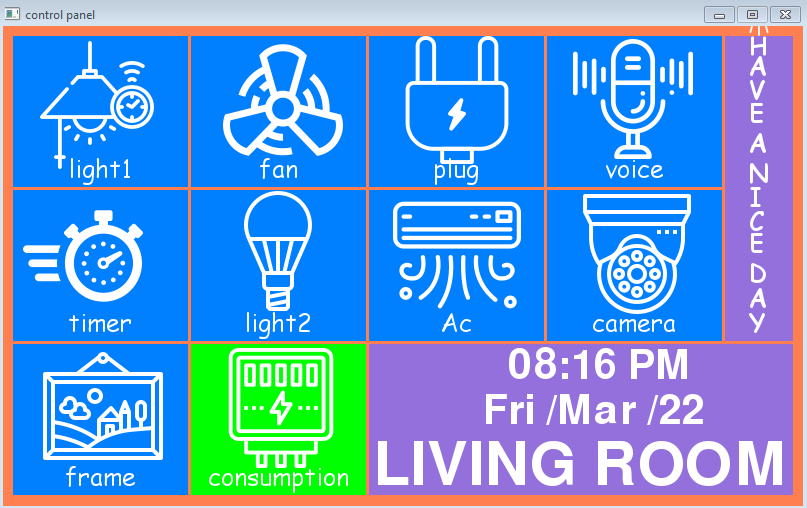
In one hacky solution, one can run a full ASR (Automatic Speech Recognition) to perform hotword detection. In this scenario, the device would watch for specific trigger words in the ASR transcriptions. However, ASR consumes a lot of device and bandwidth resources. In addition, it does not protect your privacy when one uses a cloud-based solution. Luckily, Snowboy was created to solve these problems!

**4.5 PROGRAM OUTPUT**

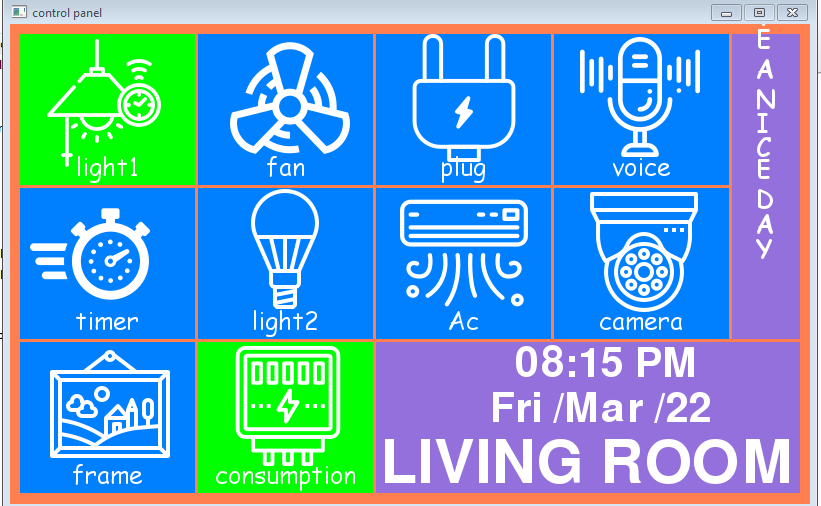
The main user interface is shown in Figure 4.2. Consider an example light1 button is said to be in off state, it is indicated through “blue colour “ background for a light1 button. When user clicks the light1 button so that it turns on light by calling this (button("light1",10,10,175,151,light\_blue,bright\_green,game\_loop)) function in python program .Now light1 button changes its background colour to “green “ as show in Figure 4.3. which indicates light1 is turned on.

**“**button("light1",10,10,175,151,light\_blue,bright\_green,game\_loop)” in this program line button is user defined function for creating button functionality. The parameter within parenthesis is a parameter passed to button function.

* 10,10,175,151 is a coordination and size of light1 button in display
* light\_blue,bright\_green is the colour parameter passed to buuton
* game\_loop is another subroutine which is executed when light1 is clicked

****

**Figure 4.2 Program Output Indicating “Light1” is Turned Off**

****

**Figure 4.3 Program Output Indicating “Light1” is Turned On**

**4.6 CONCLUSION**

Thus Python programming language was used to implement the prototype due to its rich pygame library.

**CHAPTER 5**

**CONCLUSION AND FUTURE SCOPE**

**5.1 FUTURE SCOPE**

In the world of automation, our project which is cost effective and have various add on features that will a boon to middle class people in the near future. Voice recognition of a person can be extended by training snowboy voice recognition model upto 2500 voices.

**5.2 CONC** **LUSION**

The system as the name indicates, Raspberry pi based home automation makes the system more flexible and provides attractive user interface compared to other home automation systems. The system consists of mainly four components is a resistive touch screen module, raspberry pi board, USB microphone and relay circuits. Touch screen is used as the user interface for the raspberry pi board. We hide the complexity of notions involved in the home automation system by including them into a simple, but comprehensive set of related concepts. This simplification is needed to fit as much of the functionality on the limited space offered by a resistive touch screen display.

**REFERENCES**

[1] Bhavkanwal Kaur, Pushpendra Kumar Pateriya, Mritunjay Kumar Rai “An Illustration of Making a Home Automation System Using Raspberry Pi and PIR Sensor” in 2018 International Conference on Intelligent Circuits and Systems, pp.439-44, April-2018.

[2] Shafiq ur Rehman, Volker Gruhn “An Approach to Secure Smart Homes in Cyber- Physical Systems/Internet-of-Things” in Fifth International Conference on Software Defined Systems (SDS), pp.126-129,April-2018.

[3] Pasd Putthapipat, Chutitep Woralert, Phumiphat Sirinimnuankul “Speech Recognition Gateway for Home Automation on Open Platform” in International Conference on Electronics, Information, and Communication (ICEIC), pp.1-4, Jan-2018.

[4] Bhaumik Vaidya, Ankit Patel, Anand Panchal “Smart home automation with a unique door monitoring system for old age people using Python, OpenCV, Android and Raspberry pi” in International Conference on Intelligent Computing and Control Systems pp.82-86, June-2017.

[5] H.Bharathi, U.Srivani, MD.Azharudhin, M.Srikanth, M.Sukumarline “Home Automation by Using Raspberry Pi And Android Application” in International Conference on Electronics, Communication and Aerospace Technology, vol-2,pp.687-689, April-2017.

[6] R A Ramlee, M. H. Leong, R. S. S. Singh, M. M. Ismail, M. A. Othman,H. A. Sulaiman, et al., "Bluetooth remote Home Automation System Using Android Application," The International Journal of Engineering And Science, vol. 2, pp. 149-153, 11, January 2013.

[7] Dhawan S. Thakur and Aditi Sharma, “Voice Recognition Wireless Home Automation System Based On Zigbee”, IOSR Journal, Volume 6, Issue 1, pp. 65-75, 2013.

[8] A.ElShafee and K. A. Hamed, "Design and Implementation of a Wi-Fi Based Home Automation System," World Academy of Science, Engineering and Technology, vol. 68, pp. 2177-2180, 2012.

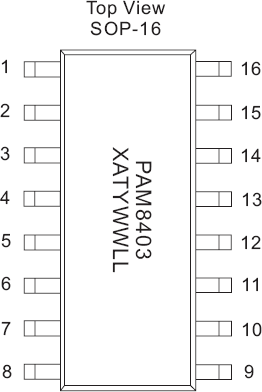
[9] Sajidullah S.Khan, Anuja Khoduskar, Dr. N.A,Koli,“Home automation system”, IJAET/Vol.II/April- June,2011/129-132.

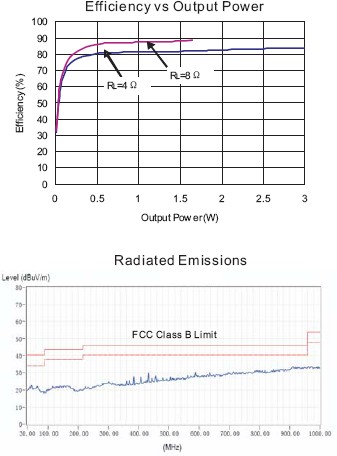
[10] R. Piyare and M. Tazil, "Bluetooth Based Home Automation System Using Cell phone," in IEEE 15th International Symposium on Consumer Electronics, Singapore 2011, pp. 192 - 195.

**APPENDIX 1**

**FEATURES OF PAM8403**

* 3W Output at 10% THD with a 4Ω Load and 5V Power Supply
* Filter less, Low Quiescent Current and Low EMI
* Low THD+N
* Superior Low Noise
* Efficiency up to 90%
* Short Circuit Protection
* Thermal Shutdown
* Few External Components to Save the Space and Cost
* Pb-Free Package





**ELECTRICAL CHARACTERISTICS (@TA = +25°C, VDD = 5V, GAIN = 24DB, RL = 8Ω, UNLESS OTHERWISE SPECIFIED.)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Symbol** | **Parameter** | **Test Conditions** | | **Min** | **Typ** | **Max** | **Units** |
| VDD | Supply Voltage |  | | 2.5 |  | 5.5 | V |
| PO | Output Power | THD+N = 10%, f = 1KHz, RL = 4Ω | VDD = 5.0V |  | 3.2 |  | W |
| VDD = 3.6V |  | 1.6 |  |
| VDD = 3.2V |  | 1.3 |  |
| THD+N = 1%, f = 1KHz, RL = 4Ω | VDD = 5.0V |  | 2.5 |  | W |
| VDD = 3.6V |  | 1.3 |  |
| VDD = 3.2V |  | 0.85 |  |
| THD+N = 10%, f = 1KHz, RL = 8Ω | VDD = 5.0V |  | 1.8 |  | W |
| VDD = 3.6V |  | 0.9 |  |
| VDD = 3.2V |  | 0.6 |  |
| THD+N = 1%, f = 1KHz, RL = 8Ω | VDD = 5.0V |  | 1.4 |  | W |
| VDD = 3.6V |  | 0.72 |  |
| VDD = 3.2V |  | 0.45 |  |
| THD+N | Total Harmonic Distortion Plus Noise | VDD = 5.0V, PO = 1W, RL = 8Ω | f = 1kHz |  | 0.15 |  | % |
| VDD = 3.6V, PO = 0.1W, RL = 8Ω |  | 0.11 |  |
| VDD = 5.0V, PO = 0.5W, RL = 4Ω | f = 1kHz |  | 0.15 |  | % |
| VDD = 3.6V, PO = 0.2W, RL = 4Ω |  | 0.11 |  |
| GV | Closed Loop Gain | VDD = 3V to 5V | |  | 24 |  | dB |
| PSRR | Power Supply Ripple Rejection | VDD = 5.0V, Inputs AC-Grounded with CIN = 0.47µF | f = 100Hz |  | -59 |  | dB |
| f = 1kHz |  | -58 |  |
| CS | Crosstalk | VDD = 5.0V, PO = 0.5W, RL = 8Ω, GV = 20db | f = 1kHz |  | -95 |  | dB |
| SNR | Signal-to-Noise Ratio | VDD = 5.0V, VORMS = 1V, GV = 20db | f = 1kHz |  | 80 |  | dB |
| VN | Output Noise | VDD = 5.0V, Inputs AC-Grounded with CIN = 0.47µF | No A-Weighting |  | 100 |  | µV |
| A-Weighting |  | 150 |  |
| Dyn | Dynamic Range | VDD = 5.0V, THD = 1% | f = 1kHz |  | 90 |  | dB |
| η | Efficiency | RL = 8Ω, THD = 10% | f = 1kHz |  | 87 |  | % |
| RL = 4Ω, THD = 10% |  | 83 |  |
| IQ | Quiescent Current | VDD = 5.0V | No Load |  | 16 |  | mA |
| VDD = 3.6V |  | 10 |  |
| VDD = 3.0V |  | 8 |  |
| IMUTE | Muting Current | VDD = 5.0V | VMUTE = 0.3V |  | 3.5 |  | mA |
| ISD | Shutdown Current | VDD = 2.5V to 5.5V | VSD = 0.3V |  | < 1 |  | µA |
| RDS(ON) | Static Drain-to-Source On-State Resistor | IDS = 500mA, VGS = 5V | PMOS |  | 180 |  | mΩ |
| NMOS |  | 140 |  |
| fSW | Switching Frequency | VDD = 3.0V to 5.0V | |  | 260 |  | kHz |
| VOS | Output Offset Voltage | VIN = 0V, VDD = 5.0V | |  | 10 |  | mV |
| VIH | Enable Input High Voltage | VDD = 5.0V | | 1.5 | 1.4 |  | V |
| VIL | Enable Input Low Voltage | VDD = 5.0V | |  | 0.7 | 0.4 |
| VIH | MUTE Input High Voltage | VDD = 5.0V | | 1.5 | 1.4 |  | V |
| VIL | MUTE Input Low Voltage | VDD = 5.0V | |  | 0.7 | 0.4 |
| OTP | Over Temperature Protection | No Load, Junction Temperature | VDD = 5.0V |  | 140 |  | V |
| OTH | Over Temperature Hysterisis |  | 30 |  | V |

**APPENDIX 2**

**TOUCH SCREEN CONTROLLER (XPT2046) PIN CONFIGURATION**

|  |  |  |
| --- | --- | --- |
| **PIN**  **Number** | **NAME** | **DESCRIPTION** |
| 13 | BUSY | Busy Output. This output is high impedance when CS is high. |
| 14 | DIN | Serial Data Input. If CS is low, data is latched on the rising edge of DCLK. |
| 15 | ——  CS | Chip Select Input. Controls conversion timing and enables the serial input/output |
| 16 | DCLK | External Clock Input. This clock runs the SAR conversion process and synchronizes |
| 1 | Vcc | Power Supply |
| 2 | XP | XP Position Input |
| 3 | YP | YP Position Input |
| 4 | XN | XN Position Input |
| 5 | YN | YN Position Input |
| 6 | Gnd | Ground |
| 7 | VBAT | Battery Monitor Input |
| 8 | AUX | Auxiliary Input to ADC |
| 9 | Vref | Voltage Reference Input/output |
| 10 | IOVDD | Digital I/O Power Supply |
| 11 | PENIRQ | Pen Interrupt |
| 12 | DOUT | Serial Data Output. Data is shifted on the  falling edge of DCLK. This output is high |