**HOME AUTOMATION USING IOT**

**ABSTRACT**

Nowadays Technology keeps on upgrading. Home security is essential for occupant’s convenience and protection. Security systems are being preferred over manual system. With the rapid increase in the number of users of internet over the past decade has made Internet a part and parcel of life, and IOTs is the latest and emerging internet technology. Home Appliances control of smart security system using IOTs uses computers or mobile devices to control basic home functions and features through internet from anywhere around the world. This security system differs from other system by allowing the user to operate the system from anywhere around the world through internet connection. With the help of Arduino microcontroller as an embedded device, security system design was constructed with the help of many sensors like PIR sensor, Temperature Sensor and electromagnetic switch.

**ACRONYMS**

* + 1. **IOT-** The Internet of Things (IoT) is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.
    2. **Cloud computing-** Cloud computing is a general term for the delivery of hosted services over the internet. Cloud computing enables companies to consume a compute resource, such as a virtual machine (VMs), storage or an application, as a utility -- just like electricity -- rather than having to build and maintain computing infrastructures in house.
    3. **Arduino-** Arduino is an open-source electronics platform based on easy-to-use hardware and software. [Arduino boards](https://www.arduino.cc/en/Main/Products) are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board.
    4. **Blynk app-** Blynk is a Platform with iOS and Android apps to control Arduino, Raspberry Pi and the likes over the Internet. It's a digital dashboard where you can build a graphic interface for your project by simply dragging and dropping widgets. It's really simple to set everything up and you'll start tinkering in less than 5 mins. Blynk is not tied to some specific board or shield. Instead, it's supporting hardware of your choice. Whether your Arduino or Raspberry Pi is linked to the Internet over Wi-Fi, Ethernet or this new ESP8266 chip, Blynk will get you online and ready for the **Internet Of Your Things**.

**INTRODUCTION**

Safety and security of any living or working place is one of the most primary concerns. The advancement of technology has increased the safety and security of people along with their belongings. One of the reasons for the rise of the smart home is the increasing risk of burglary and robbery and the busy lifestyle. The busy lifestyle of people is leading to the necessity of controlling the devices at home remotely and increasing the necessity of keeping surveillance over their homes. With advancement of internet technology lifestyle of every person is changing constantly. So here is a method proposed to make a home secured with the help of IOT and along with some sensors.

The Internet of things can be defined as connecting the various types of objects like smart phones, personal computer and Tablets to internet, which brings in very newfangled type of communication between things and people and also between things [2]. With the introduction of IoTs, the research and development of home automation are becoming popular in the recent days. Many of the devices are controlled and monitored for helps the human being. Additionally various wireless technologies help in connecting from remote places to improve the intelligence of home environment. An advanced network of IoT is being formed when a human being is in need of connecting with other things. IoTs technology is used to come in with innovative idea and great growth for smart homes to improve the living standards of life. Internet helps us to bring in with immediate solution for many problems and also able to connect from any of the remote places which contributes to overall cost reduction and energy consumption

In recent years, there has been a growing interest among consumers in the smart home concept. Home automation system represents and reports the status of the connected devices in an intuitive, user-friendly interface allowing the user to interact and control various devices with the touch of a few buttons. Some of the major communication technologies used by today’s home automation system include Bluetooth, Wi-MAX and Wireless LAN (Wi-Fi), ZigBee, and Global System for Mobile Communication (GSM) .Here we are using Wi-Fi module. It offers the user complete access control of the appliances through a remote interface. Automation is the use of control systems and information technology to control equipment, industrial machinery and processes, reducing the need for the human intervention

The wide variety of potential IoT applications needs a software development environment that ties together the applications, the command, control and routing processing and the security of the node and system. While the importance of software in MCU solutions has increased during the past few years, for MCUs supporting the IoT, even more software, tools and enablement will be needed. A broad ecosystem with easily accessible support is key to enabling the development of embedded processing nodes and IoT applications.

**PROBLEM STATEMENT**

Nowadays IoT is everywhere in the world to make the smarter world. Due to IoT we can see many smart devices around us. Many people, including myself, hold the view that cities and the world itself will be overlaid with sensing and actuation, many embedded in “things” creating what is referred to as a smart world. For example, today many buildings already have sensors for attempting to save energy, home automation; cars, taxis, and traffic lights have devices to try and improve safety and transportation; people have smart phones with sensors for running many useful apps; industrial plants are connecting to the Internet; and healthcare services are relying on increased home sensing to support remote medicine and wellness. One possibility is a global sensing and actuation utility connected to the Internet. Electricity and water are two utilities that can be used for a myriad of purposes. Sensing and actuation in the form of an IoT platform will become a utility. IoT will not be seen as individual systems, but as a critical, integrated infrastructure upon which many applications and services can run. Some applications will be personalized such as digitizing daily life activities, others will be city-wide such as efficient, delay-free transportation, and others will be worldwide such as global delivery systems. In cities perhaps there will be no traffic lights and even 3D transportation vehicles. Smart buildings will not only control energy or security, but integrate personal comfort, energy savings, security and health and wellness aspects into convenient and effective spaces. Individuals may have patches of bionic skin with sensing of physiological parameters being transmitted to the cloud which houses his digital health, and to the surrounding smart spaces for improved comfort, health, efficiency, and safety. In fact, smart watches, phones, body nodes, and clothes will act as personalized input to optimize city-wide services benefiting both the individual and society [10]. Ten “critical” trends and technologies impacting IT for the next five years were laid out by Gartner and among them the Internet of Things [5]. All of these things have an IP address and can be tracked. The Internet is expanding into enterprise assets and consumer items such as cars and televisions. The problem is that most enterprises and technology vendors have yet to explore the possibilities of an expanded Internet and are not operationally or organizationally ready. Gartner identifies four basic usage models that are:

* Emerging
* Manage
* Monetize
* Operate
* Extend

These can be applied to people, things, information, and places, and therefore the so called “Internet of Things” will be succeeded by the “Internet of Everything.”

**LITERATURE SURVEY**

Consequently, we will often be implicitly linked into the new utility. Some examples of new services include immediate and continuous access to the right information for the task at hand, be it, traveling to work or a meeting, exercising, shopping, socializing, or visiting a doctor. Sometimes these activities will be virtual activities, or even include the use of avatars or robots. Many outputs and displays for users may be holographic. Credit cards should disappear and biometrics like voice or retinas will provide safe access to buildings, ATMs, and transportation systems. A sensing and actuation utility will not only exist in public spaces, but also extend into the home, apartments, and condominiums. Here people will be able to run health, energy, security, and entertainment apps on the infrastructure. Installing and running new apps will be as easy as plugging in a new toaster into the electric utility. One app may help monitor and control heart rate, another perform financial and investments services, another automatically ordering food and wine, or even predicting a impending medical problem that should be addressed early to mitigate or even avoid the problem. Humans will often be integral parts of the IoT system. The Industrial Internet is also a form of IoT where the devices (things) are objects in manufacturing plants, dispatch centers, process control industries, etc[11].

According to Jayavardhana[12], the term Internet of Things was first coined by Kevin Ashton in 1999 in the context of supply chain management. However, in the past decade, the definition has been more inclusive covering wide range of applications like healthcare, utilities, transport, etc. Although the definition of ‘Things’ has changed as technology evolved, the main goal of making a computer sense information without the aid of human intervention remains the same. A radical evolution of the current Internet into a Network of interconnected objects that not only harvests information from the environment (sensing) and interacts with the physical world (actuation/ command/control), but also uses existing Internet standards to provide services for information transfer, analytics, applications, and communications. Fueled by the prevalence of devices enabled by open wireless technology such as Bluetooth, radio frequency identification (RFID), Wi-Fi, and telephonic data services as well as embedded sensor and actuator nodes, IoT has stepped out of its infancy and is on the verge of transforming the current static Internet into a fully integrated Future Internet. The Internet revolution led to the interconnection between people at an unprecedented scale and pace. The next revolution will be the interconnection between objects to create a smart environment. John A. Stankovic vision saying [10], Many technical communities are vigorously pursuing research topics that contribute to the IoT. Today, as sensing, communication, and control become ever more sophisticated and ubiquitous, there is significant overlap in these communities; sometimes from slightly different perspectives. More cooperation between communities is encouraged. To provide a basis for discussing open research problems in IoT, a vision for how IoT could change the world in the distant future. In 2013, Salah Addin Ahmed developed Smart GSM Based Home Automation System. In recent years, there has been a growing interest among consumers in the smart home concept. Smart homes contain multiple, connected devices such as home entertainment consoles, security systems, lighting, access control systems and surveillance. Intelligent home automation system is incorporated into smart homes to provide comfort, convenience, and security to home owners. Home automation system represents and reports the status of the connected devices in an intuitive, user-friendly interface allowing the user to interact and control various devices with the touch of a few buttons. Some of the major communication technologies used by today’s home automation system include Bluetooth, WiMAX and Wireless LAN (Wi-Fi), ZigBee, and Global System for Mobile Communication (GSM). All GSM is one of the most widely used cellular technologies in the world. With the increase in the number of GSM subscribers, research and development is heavily supported in further investigating the GSM implementation [1]. In 2014, Nikhil Singh, Shambhu Shankar Bharti, Rupal Singh, Dushyant Kumar Singh, developed Remotely Controlled Home Automation System. They made server and android based Home automation system. The program is designed such that if no one is at home then all home appliances automatically switched off. For designed purpose we have used Proteus Design Suite. The design consist a simple home automation design having a “motion sensor” for counting number of people inside home, “a speed controlled fan” ,”a light bulb”, ”a LCD” for displaying status of home appliances. “a microcontroller ic” for controlling devices “a port” for connection purpose.

**Home Automation System (HAS) using Android forMobile PhoneSharon Panth , Mahesh Jivani**

Automation of the surrounding environment of a modern human being allows increasing his work efficiency and comfort. There has been a significant development in the area of an individual’s routine tasks and those can be automated. In the present times, we can find most of the people clinging to their mobile phones and smart devices throughout the day. Hence with the help of his companion – a mobile phone, some daily household tasks can be accomplished by personifying the use of the mobile phone. Analyzing the current smart phone market, novice mobile users are opting for Android based phones. It has become a second name for a mobile phone in layman terms. Home Automation System (HAS) has been designed for mobile phones having Android platform to automate an 8 bit Bluetooth interfaced microcontroller which controls a number of home appliances like lights, fans, bulbs and many more using on/off relay. This paper presents the automated approach of controlling the devices in a household that could ease the tasks of using the traditional method of the switch. The most famous and efficient technology for short range wireless communication- Bluetooth is used here to automate the system. The HAS system for Android users is a step towards the ease of the tasks by controlling one to twenty four different appliances in any home environment.

**Energy Efficient IoT-Based Smart Home - Laila Salman, Safa Salman, Saeed Jahangirian, Mehdi Abraham, Fred German, Charlotte Blair, Peter Krenz ANSYS Inc. Canonsburg, PA, USA**

Smart Home technology is the future of residential related technology which is designed to deliver and distribute number of services inside and outside the house via networked devices in which all the different applications & the intelligence behind them are integrated and interconnected. These smart devices have the potential to share information with each other given the permanent availability to access the broadband internet connection. Hence, Smart Home Technology has become part of IoT (Internet of Things). In this work, a home model is analysed to demonstrate an energy efficient IoT based smart home.

Several Multiphysics simulations were carried out focusing on the kitchen of the home model. A motion sensor with a surveillance camera was used as part of the home security system. Coupled with the home light and HVAC control systems, the smart system can remotely control the lighting and heating or cooling when an occupant enters or leaves the kitchen.

**OBJECTIVES**

* To provide security, comfort and energy efficiency and convenience at all times
* Enhance the security system
* Functionality and flexibility is given to home appliances
* Provide a smart home at low cost

**METHODOLOGY**

**HARDWARE AND SOFTWARE REQUIRED**

Hardware Requirements:

* Microcontroller
* PIR Sensor
* Electromagnetic Switch
* temperature sensor
* PC
* Relay

Software Requirements:

* Embedded C
* Blynk Widget

**BLOCK DIAGRAM**

**BLYNK APP**

**CLOUD**

**BUZZER**

**LED**

**FAN**

**WATER PUMP**

**4-channel relay**

**PC**

**ARDUINO**

**ARDUINO**

**Overview**

Arduino is an open-source electronics platform based on easy-to-use hardware and software. [Arduino boards](https://www.arduino.cc/en/Main/Products) are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the [Arduino programming language](https://www.arduino.cc/en/Reference/HomePage) (based on [Wiring](http://wiring.org.co/)), and [the Arduino Software (IDE)](https://www.arduino.cc/en/Main/Software), based on [Processing](https://processing.org/).

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of [accessible knowledge](http://forum.arduino.cc/) that can be of great help to novices and experts alike.

Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IoT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and eventually adapt them to their particular needs. The [software](https://www.arduino.cc/en/Main/Software), too, is open-source, and it is growing through the contributions of users worldwide.

Arduino is a computer hardware and software company, project, and user community that designs and manufactures [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware) and [software](https://en.wikipedia.org/wiki/Open-source_software), which are licensed under the [GNU Lesser General Public License](https://en.wikipedia.org/wiki/GNU_Lesser_General_Public_License) (LGPL) or the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL),[[1]](https://en.wikipedia.org/wiki/Arduino#cite_note-1) permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as [do-it-yourself](https://en.wikipedia.org/wiki/Do-it-yourself) kits.

The project's board designs use a variety of microprocessors and controllers. These systems provide sets of digital and analog [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O) pins that may be interfaced to various expansion boards ("shields") and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus ([USB](https://en.wikipedia.org/wiki/USB)) on some models, for loading programs from personal computers. The microcontrollers are mainly programmed using a dialect of features from the programming languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B). In addition to using traditional compiler toolchains, the Arduino project provides an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) based on the [Processing](https://en.wikipedia.org/wiki/Processing_(programming_language)) language project.

The Arduino project started in 2005 as a program for students at the [Interaction Design Institute Ivrea](https://en.wikipedia.org/wiki/Interaction_Design_Institute_Ivrea) in [Ivrea](https://en.wikipedia.org/wiki/Ivrea), Italy,[[2]](https://en.wikipedia.org/wiki/Arduino#cite_note-kushner-2) aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using [sensors](https://en.wikipedia.org/wiki/Sensor) and [actuators](https://en.wikipedia.org/wiki/Actuator). Common examples of such devices intended for beginner hobbyists include simple [robots](https://en.wikipedia.org/wiki/Robot), [thermostats](https://en.wikipedia.org/wiki/Thermostat), and [motion detectors](https://en.wikipedia.org/wiki/Motion_detector).

Arduino/Genuino Uno is a microcontroller board based on the ATmega328P ([datasheet](http://www.atmel.com/Images/doc8161.pdf)). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worring too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

### **Why Arduino?**

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications. The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step by step instructions of a kit, or sharing ideas online with other members of the Arduino community.

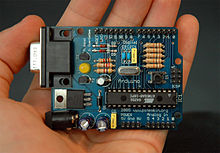
There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

* **Inexpensive** - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50
* **Cross-platform** - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
* **Simple, clear programming environment** - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
* **Open source and extensible software** - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.
* **Open source and extensible hardware** - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the [breadboard version of the module](https://www.arduino.cc/en/Main/Standalone) in order to understand how it works and save money.

**Hardware**

Arduino is [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware). The hardware reference designs are distributed under a [Creative Commons](https://en.wikipedia.org/wiki/Creative_Commons) Attribution Share-Alike 2.5 license and are available on the Arduino website. Layout and production files for some versions of the hardware are also available. The source code for the IDE is released under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), version 2.[[8]](https://en.wikipedia.org/wiki/Arduino#cite_note-8) Nevertheless an official [Bill of Materials](https://en.wikipedia.org/wiki/Bill_of_Materials) of Arduino boards has never been released by the staff of Arduino.

Although the hardware and software designs are freely available under [copyleft](https://en.wikipedia.org/wiki/Copyleft) licenses, the developers have requested that the name "Arduino" be [exclusive to the official product](https://en.wikipedia.org/wiki/Generic_trademark) and not be used for derived works without permission. The official policy document on use of the Arduino name emphasizes that the project is open to incorporating work by others into the official product.[[9]](https://en.wikipedia.org/wiki/Arduino#cite_note-AutoF7-44-9) Several Arduino-compatible products commercially released have avoided the *Arduino* name by using *-duino* name variants.[[10]](https://en.wikipedia.org/wiki/Arduino#cite_note-freeduino-10)

[](https://en.wikipedia.org/wiki/File:Arduino316.jpg)

An early Arduino board[[11]](https://en.wikipedia.org/wiki/Arduino#cite_note-11) with an [RS-232](https://en.wikipedia.org/wiki/RS-232) [serial](https://en.wikipedia.org/wiki/Serial_communication) interface (upper left) and an Atmel ATmega8 microcontroller chip (black, lower right); the 14 digital I/O pins are at the top, the 6 analog input pins at the lower right, and the power connector at the lower left.

An Arduino board consists of an [Atmel](https://en.wikipedia.org/wiki/Atmel) 8-, 16- or 32-bit AVR [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) (although since 2015 other makers' microcontrollers have been used) with complementary components that facilitate programming and incorporation into other circuits. An important aspect of the Arduino is its standard connectors, which let users connect the CPU board to a variety of interchangeable add-on modules termed *shields*. Some shields communicate with the Arduino board directly over various pins, but many shields are individually addressable via an [I²C](https://en.wikipedia.org/wiki/I%C2%B2C)[serial bus](https://en.wikipedia.org/wiki/Serial_bus)—so many shields can be stacked and used in parallel. Before 2015, Official Arduinos had used the Atmel [megaAVR](https://en.wikipedia.org/wiki/MegaAVR) series of chips, specifically the ATmega8, ATmega168, [ATmega328](https://en.wikipedia.org/wiki/ATmega328), ATmega1280, and ATmega2560. In 2015, units by other producers were added. A handful of other processors have also been used by Arduino compatible devices. Most boards include a 5 V [linear regulator](https://en.wikipedia.org/wiki/Linear_regulator) and a 16 MHz [crystal oscillator](https://en.wikipedia.org/wiki/Crystal_oscillator) (or [ceramic resonator](https://en.wikipedia.org/wiki/Ceramic_resonator) in some variants), although some designs such as the LilyPad run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions. An Arduino's microcontroller is also pre-programmed with a [boot loader](https://en.wikipedia.org/wiki/Boot_loader) that simplifies uploading of programs to the on-chip [flash memory](https://en.wikipedia.org/wiki/Flash_memory), compared with other devices that typically need an external [chip programmer](https://en.wikipedia.org/wiki/Programmer_(hardware)). This makes using an Arduino more straightforward by allowing the use of an ordinary computer as the programmer. Currently, optiboot bootloader is the default bootloader installed on Arduino UNO.[[12]](https://en.wikipedia.org/wiki/Arduino#cite_note-12)

At a conceptual level, when using the Arduino integrated development environment, all boards are programmed over a serial connection. Its implementation varies with the hardware version. Some serial Arduino boards contain a level shifter circuit to convert between [RS-232](https://en.wikipedia.org/wiki/RS-232) logic levels and [transistor–transistor logic](https://en.wikipedia.org/wiki/Transistor%E2%80%93transistor_logic) (TTL) level signals. Current Arduino boards are programmed via [Universal Serial Bus](https://en.wikipedia.org/wiki/Universal_Serial_Bus) (USB), implemented using USB-to-serial adapter chips such as the [FTDI](https://en.wikipedia.org/wiki/FTDI) FT232. Some boards, such as later-model Uno boards, substitute the FTDI chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, [Bluetooth](https://en.wikipedia.org/wiki/Bluetooth) or other methods, when used with traditional microcontroller tools instead of the Arduino IDE, standard AVR [in-system programming](https://en.wikipedia.org/wiki/In-system_programming) (ISP) programming is used.

[](https://en.wikipedia.org/wiki/File:UnoConnections.jpg)

An official Arduino Uno R2 with descriptions of the I/O locations

The Arduino board exposes most of the microcontroller's I/O pins for use by other circuits. The *Diecimila*,[[a]](https://en.wikipedia.org/wiki/Arduino#cite_note-N10000-13) *Duemilanove*,[[b]](https://en.wikipedia.org/wiki/Arduino#cite_note-N2009-14) and current *Uno*[[c]](https://en.wikipedia.org/wiki/Arduino#cite_note-N1-15) provide 14 digital I/O pins, six of which can produce [pulse-width modulated](https://en.wikipedia.org/wiki/Pulse-width_modulation) signals, and six analog inputs, which can also be used as six digital I/O pins. These pins are on the top of the board, via female 0.1-inch (2.54 mm) headers. Several plug-in application shields are also commercially available. The Arduino Nano, and Arduino-compatible Bare Bones Board[[13]](https://en.wikipedia.org/wiki/Arduino#cite_note-16) and Boarduino[[14]](https://en.wikipedia.org/wiki/Arduino#cite_note-17) boards may provide male header pins on the underside of the board that can plug into solderless [breadboards](https://en.wikipedia.org/wiki/Breadboard).

Many Arduino-compatible and Arduino-derived boards exist. Some are functionally equivalent to an Arduino and can be used interchangeably. Many enhance the basic Arduino by adding output drivers, often for use in school-level education, to simplify making buggies and small robots. Others are electrically equivalent but change the form factor, sometimes retaining compatibility with shields, sometimes not. Some variants use different processors, of varying compatibility.

## **Digital Pins**

The pins on the Arduino can be configured as either inputs or outputs. This document explains the functioning of the pins in those modes. While the title of this document refers to digital pins, it is important to note that vast majority of Arduino (Atmega) analog pins, may be configured, and used, in exactly the same manner as digital pins.

### **Properties of Pins Configured as INPUT**

Arduino (Atmega) pins default to inputs, so they don't need to be explicitly declared as inputs with pinMode() when you're using them as inputs. Pins configured this way are said to be in a **high-impedance state**. Input pins make extremely small demands on the circuit that they are sampling, equivalent to a series resistor of 100 megohm in front of the pin. This means that it takes very little current to move the input pin from one state to another, and can make the pins useful for such tasks as implementing [a capacitive touch sensor](http://www.arduino.cc/playground/Code/CapacitiveSensor), reading an LED as a [photodiode](http://www.arduino.cc/playground/Learning/LEDSensor), or reading an analog sensor with a scheme such as [RCTime.](https://www.arduino.cc/en/Tutorial/RCtime)

This also means however, that pins configured as pinMode(pin, INPUT) with nothing connected to them, or with wires connected to them that are not connected to other circuits, will report seemingly random changes in pin state, picking up electrical noise from the environment, or capacitively coupling the state of a nearby pin.

### **Pullup Resistors with pins configured as INPUT**

Often it is useful to steer an input pin to a known state if no input is present. This can be done by adding a pullup resistor (to +5V), or a pulldown resistor (resistor to ground) on the input. A 10K resistor is a good value for a pullup or pulldown resistor.

### **Properties of Pins Configured as INPUT\_PULLUP**

There are 20K pullup resistors built into the Atmega chip that can be accessed from software. These built-in pullup resistors are accessed by setting the pinMode() as INPUT\_PULLUP. This effectively inverts the behavior of the INPUT mode, where HIGH means the sensor is off, and LOW means the sensor is on.

The value of this pullup depends on the microcontroller used. On most AVR-based boards, the value is guaranteed to be between 20kΩ and 50kΩ. On the Arduino Due, it is between 50kΩ and 150kΩ. For the exact value, consult the datasheet of the microcontroller on your board.

When connecting a sensor to a pin configured with INPUT\_PULLUP, the other end should be connected to ground. In the case of a simple switch, this causes the pin to read HIGH when the switch is open, and LOW when the switch is pressed.

The pullup resistors provide enough current to dimly light an LED connected to a pin that has been configured as an input. If LEDs in a project seem to be working, but very dimly, this is likely what is going on.

The pullup resistors are controlled by the same registers (internal chip memory locations) that control whether a pin is HIGH or LOW. Consequently, a pin that is configured to have pullup resistors turned on when the pin is an INPUT, will have the pin configured as HIGH if the pin is then switched to an OUTPUT with pinMode(). This works in the other direction as well, and an output pin that is left in a HIGH state will have the pullup resistors set if switched to an input with pinMode().

### **Properties of Pins Configured as OUTPUT**

Pins configured as OUTPUT with pinMode() are said to be in a low-impedance state. This means that they can provide a substantial amount of current to other circuits. Atmega pins can source (provide positive current) or sink (provide negative current) up to 40 mA (milliamps) of current to other devices/circuits. This is enough current to brightly light up an LED (don't forget the series resistor), or run many sensors, for example, but not enough current to run most relays, solenoids, or motors.

Short circuits on Arduino pins, or attempting to run high current devices from them, can damage or destroy the output transistors in the pin, or damage the entire Atmega chip. Often this will result in a "dead" pin in the microcontroller but the remaining chip will still function adequately. For this reason it is a good idea to connect OUTPUT pins to other devices with 470Ω or 1k resistors, unless maximum current draw from the pins is required for a particular

Technical specifications

|  |  |
| --- | --- |
| Microcontroller | [ATmega328P](http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf) |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limit) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| PWM Digital I/O Pins | 6 |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 20 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328P) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328P) |
| EEPROM | 1 KB (ATmega328P) |
| Clock Speed | 16 MHz |
| LED\_BUILTIN | 13 |
| Length | 68.6 mm |
| Width | 53.4 mm |
| Weight | 25 g |
|  |  |

**ATmega328p Microcontroller**

# Introduction

The Atmel® picoPower® ATmega328/P is a low-power CMOS 8-bit microcontroller based on the AVR® enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega328/P achieves throughputs close to 1MIPS per MHz. This empowers system designer to optimize the device for power consumption versus processing speed. Freatures High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller Family Advanced RISC Architecture

* 131 Powerful Instructions
* Most Single Clock Cycle Execution
* 32 x 8 General Purpose Working Registers
* Fully Static Operation
* Up to 20 MIPS Throughput at 20MHz
* On-chip 2-cycle Multiplier

High Endurance Non-volatile Memory Segments

* 32KBytes of In-System Self-Programmable Flash program

**Memory**

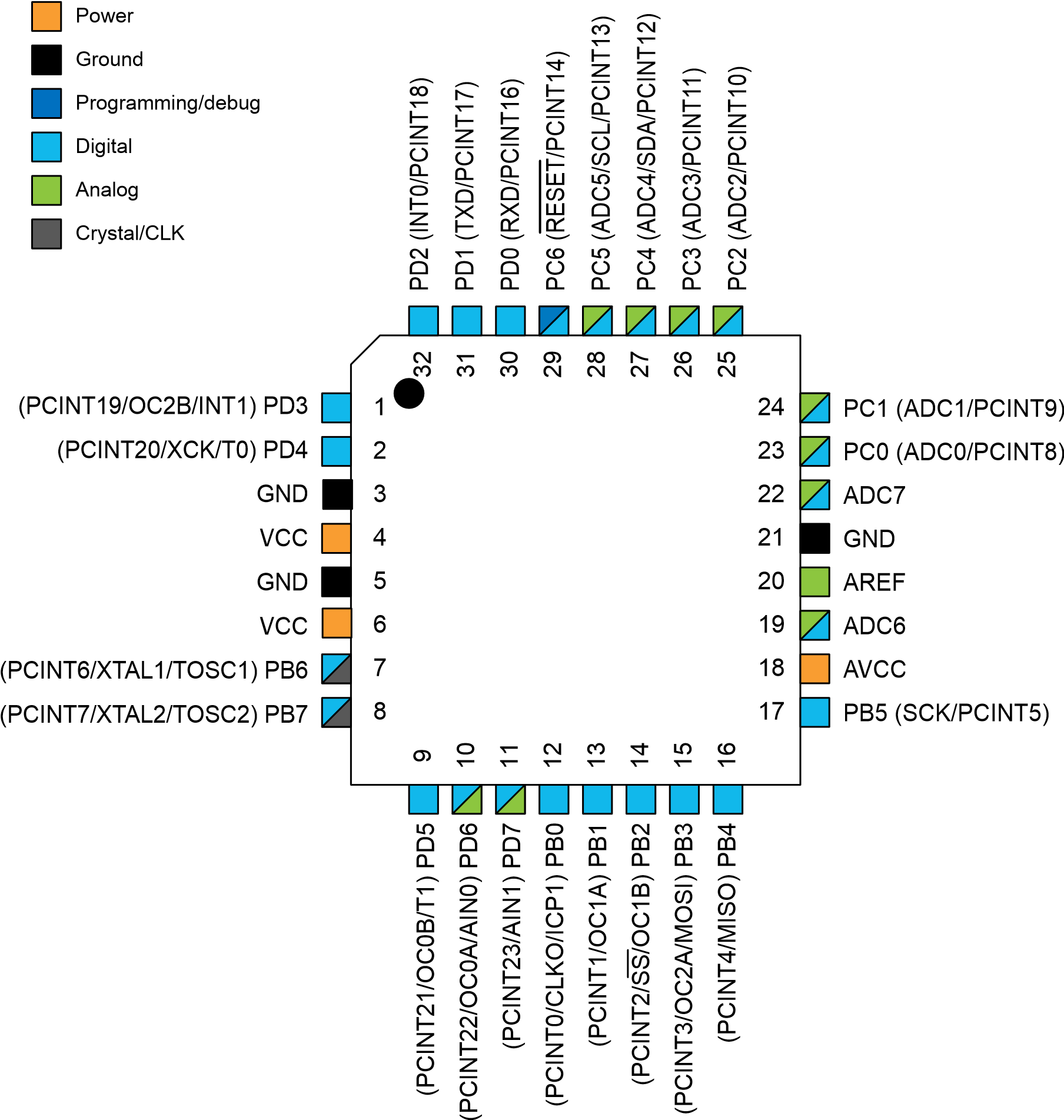
* 1KBytes EEPROM
* 2KBytes Internal SRAM
* Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
* Data Retention: 20 years at 85°C/100 years at 25°C(1)
* Optional Boot Code Section with Independent Lock Bits
* In-System Programming by On-chip Boot Program
* True Read-While-Write Operation
* Programming Lock for Software Security
* Atmel® QTouch® Library Support
* Capacitive Touch Buttons, Sliders and Wheels
* QTouch and QMatrix® Acquisition
* Up to 64 sense channels
* Peripheral Features
* Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
* One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
* Real Time Counter with Separate Oscillator
* Six PWM Channels
* 8-channel 10-bit ADC in TQFP and QFN/MLF package
* Temperature Measurement
* 6-channel 10-bit ADC in PDIP Package
* Two Master/Slave SPI Serial Interface
* One Programmable Serial USART
* One Byte-oriented 2-wire Serial Interface (Philips I2C compatible)
* Programmable Watchdog Timer with Separate On-chip Oscillator
* One On-chip Analog Comparator
* Interrupt and Wake-up on Pin Change
* Special Microcontroller Features
* Power-on Reset and Programmable Brown-out Detection
* Internal Calibrated Oscillator
* External and Internal Interrupt Sources
* Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
* I/O and Packages
* 23 Programmable I/O Lines
* 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
* Operating Voltage:
* 1.8 - 5.5V
* Temperature Range:
* -40°C to 105°C
* Speed Grade:
* 0 - 4MHz @ 1.8 - 5.5V
* 0 - 10MHz @ 2.7 - 5.5V
* 0 - 20MHz @ 4.5 - 5.5V
* Power Consumption at 1MHz, 1.8V, 25°C
* Active Mode: 0.2mA
* Power-down Mode: 0.1μA
* Power-save Mode: 0.75μA (Including 32kHz RTC)

## **Pin Configurations**

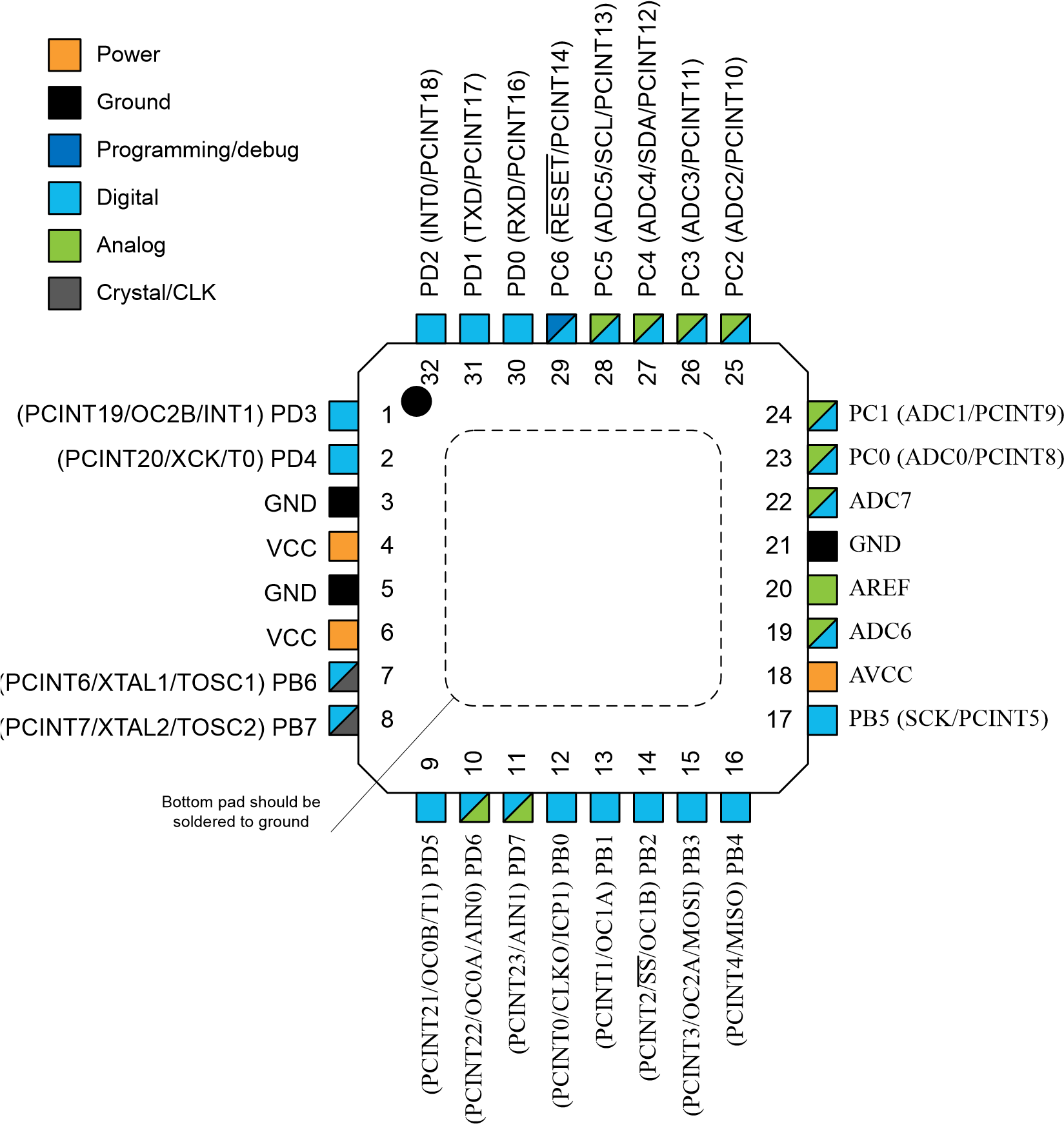
#### 

 28-pin MLF Top View

#### Figure 5-3. 32-pin TQFP Top View



#### Figure 5-4. 32-pin MLF Top View



### **Pin Descriptions**

### **VCC**

Digital supply voltage.

**GND**

Ground.

**Port B (PB[7:0]) XTAL1/XTAL2/TOSC1/TOSC2**

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

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

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

If the Internal Calibrated RC Oscillator is used as chip clock source, PB[7:6] is used as TOSC[2:1] input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

**Port C (PC[5:0])**

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC[5:0] output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

**PC6/RESET**

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

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in the *Alternate Functions of Port C* section.

**Port D (PD[7:0])**

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

**AVCC**

AVCC is the supply voltage pin for the A/D Converter, PC[3:0], and PE[3:2]. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC[6:4] use digital supply voltage, VCC.

**AREF**

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

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

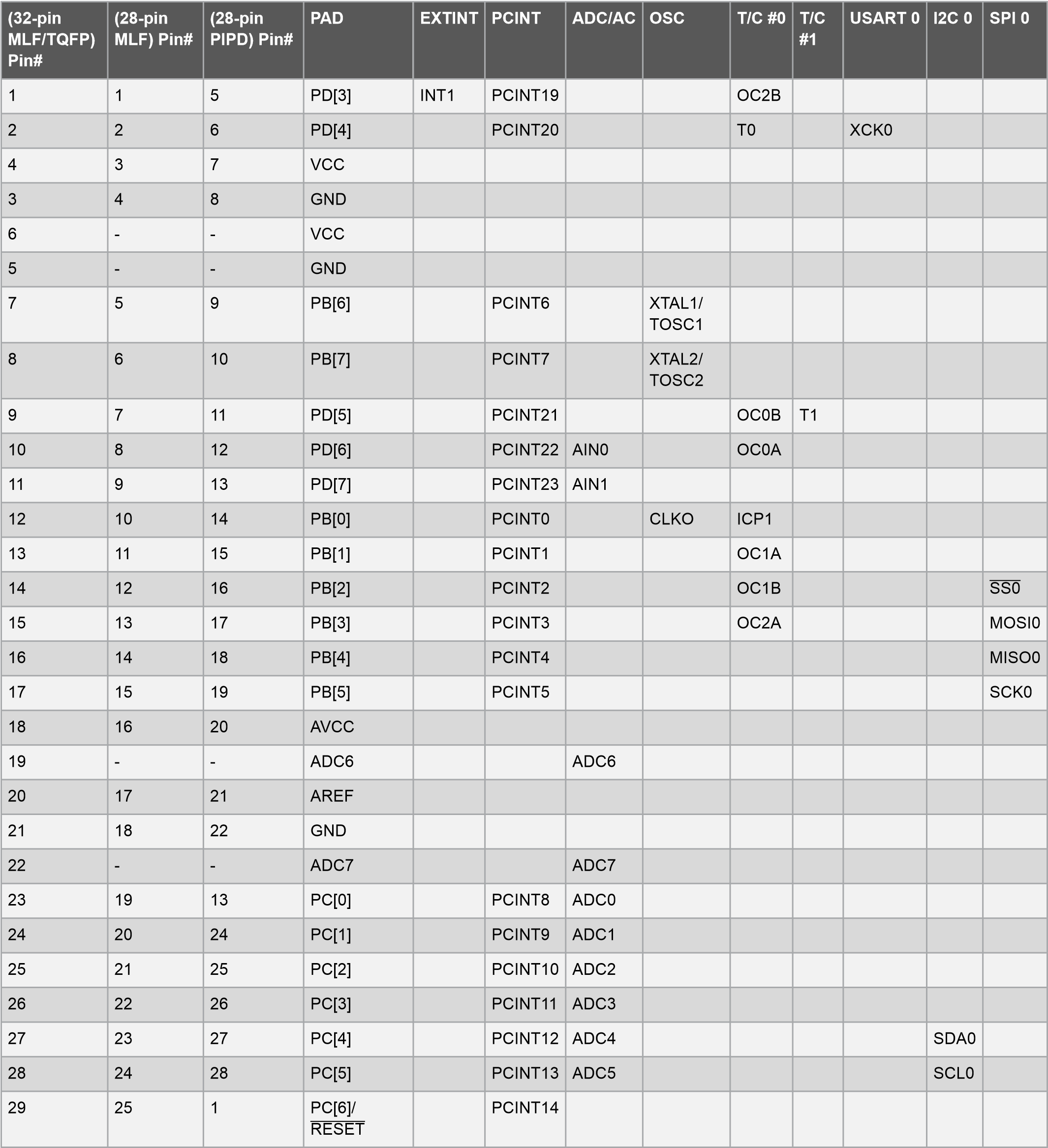
In the TQFP and VFQFN package, ADC[7:6] serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

## 6.I/O Multiplexing

Each pin is by default controlled by the PORT as a general purpose I/O and alternatively it can be assigned to one of the peripheral functions.

The following table describes the peripheral signals multiplexed to the PORT I/O pins.

Table 6-1. PORT Function Multiplexing



**Software development**

**Programming**

The Arduino/Genuino Uno can be programmed with the ([Arduino Software](https://www.arduino.cc/en/Main/Software) (IDE)). Select "Arduino/Genuino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the [reference](https://www.arduino.cc/en/Reference/HomePage) and [tutorials](https://www.arduino.cc/en/Tutorial/HomePage).

The ATmega328 on the Arduino/Genuino Uno comes preprogrammed with a [bootloader](https://www.arduino.cc/en/Hacking/Bootloader?from=Tutorial.Bootloader) that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol ([reference](http://www.atmel.com/Images/doc2525.pdf), [C header files](http://www.atmel.com/dyn/resources/prod_documents/avr061.zip)).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using [Arduino ISP](https://www.arduino.cc/en/Main/ArduinoISP) or similar; see [these instructions](https://www.arduino.cc/en/Hacking/Programmer) for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available in the Arduino repository. The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then rese ing the 8U2.

On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use [Atmel's FLIP software](http://www.atmel.com/products/microcontrollers/default.aspx) (Windows) or the [DFU programmer](http://dfu-programmer.github.io/) (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See [this user-contributed tutorial](http://forum.arduino.cc/index.php/topic,111.0.html) for more information.

**Warnings**

The Arduino/Genuino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

**Differences with other boards**

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

**Power**

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

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

Vin. The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

5V.This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND. Ground pins.

IOREF. This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

**Memory**

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the [EEPROM library](https://www.arduino.cc/en/Reference/EEPROM)).

**Input and Output**

Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode()](https://www.arduino.cc/en/Reference/PinMode), [digitalWrite()](https://www.arduino.cc/en/Reference/DigitalWrite), and [digitalRead()](https://www.arduino.cc/en/Reference/DigitalRead) functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

* Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
* PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
* SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference()function.  
There are a couple of other pins on the board:

* AREF. Reference voltage for the analog inputs. Used with analogReference().
* Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

**Communication**

Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, [on Windows, a .inf file is required](https://www.arduino.cc/en/Guide/Windows#toc4). The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A [Software Serial library](https://www.arduino.cc/en/Reference/SoftwareSerial) allows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the [documentation](https://www.arduino.cc/en/Reference/Wire) for details. For SPI communication, use the [SPI library](https://www.arduino.cc/en/Reference/SPI).

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see [this forum thread](http://forum.arduino.cc/index.php/topic,22974.0.html) for details.

Arduino programs may be written in any [programming language](https://en.wikipedia.org/wiki/Programming_language) with a compiler that produces binary machine code. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio, which can be used for programming Arduino.

The Arduino project provides the Arduino [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE), which is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application written in the programming language [Java](https://en.wikipedia.org/wiki/Java_(programming_language)). It originated from the IDE for the languages [*Processing*](https://en.wikipedia.org/wiki/Processing_(programming_language)) and [*Wiring*](https://en.wikipedia.org/wiki/Wiring_(development_platform)). It was created for people with no profound knowledge of electronics. It includes a code editor with features such as [syntax highlighting](https://en.wikipedia.org/wiki/Syntax_highlighting), [brace matching](https://en.wikipedia.org/wiki/Brace_matching), cutting-pasting and searching-replacing text, and automatic indenting, and provides simple one-click mechanism to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a series of menus.

A program written with the IDE for Arduino is called a "sketch".[[40]](https://en.wikipedia.org/wiki/Arduino#cite_note-43) Sketches are saved on the development computer as files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules to organize code. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two functions, for starting the sketch and the main programs loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive) program with the [GNU toolchain](https://en.wikipedia.org/wiki/GNU_toolchain), also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal coding that is loaded into the Arduino board by a loader program in the board's firmware

A minimal Arduino C/C++ sketch, as seen by the Arduino IDE programmer, consist of only two functions:

* *setup()*: This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
* *loop()*: After setup() is called, this function is called repeatedly by a program loop in the main program. It controls the board until it is powered off or is reset.

Most Arduino boards contain a [light-emitting diode](https://en.wikipedia.org/wiki/Light-emitting_diode) (LED) and a load resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions.[[44]](https://en.wikipedia.org/wiki/Arduino#cite_note-Blink_Tutorial-47) A typical program for a beginning Arduino programmer blinks an LED repeatedly. This program is usually loaded in the Arduino by the manufacturer. In the Arduino environment, a user might write such a program as shown:

#define LED\_PIN 13 // Pin number attached to LED.

void setup() {

pinMode(LED\_PIN, OUTPUT); // Configure pin 13 to be a digital output.

}

void loop() {

digitalWrite(LED\_PIN, HIGH); // Turn on the LED.

delay(1000); // Wait 1 second (1000 milliseconds).

digitalWrite(LED\_PIN, LOW); // Turn off the LED.

delay(1000); // Wait 1 second.

}

This program uses the functions *pinMode()*, *digitalWrite()*, and *delay()*, which are provided by the internal libraries included in the IDE environment.

**RELAY**

The diagram shows an inner section diagram of a relay. An iron core is surrounded by a control coil. As shown, the power source is given to the electromagnet through a control switch and through contacts to the load. When current starts flowing through the control coil, the electromagnet starts energizing and thus intensifies the magnetic field. Thus the upper contact arm starts to be attracted to the lower fixed arm and thus closes the contacts causing a short circuit for the power to the load. On the other hand, if the relay was already de-energized when the contacts were closed, then the contact move oppositely and make an open circuit.

As soon as the coil current is off, the movable armature will be returned by a force back to its initial position. This force will be almost equal to half the strength of the magnetic force. This force is mainly provided by two factors. They are the spring and also gravity.

Relays are mainly made for two basic operations. One is low voltage application and the other is high voltage. For low voltage applications, more preference will be given to reduce the noise of the whole circuit. For high voltage applications, they are mainly designed to reduce a phenomenon called arcing.

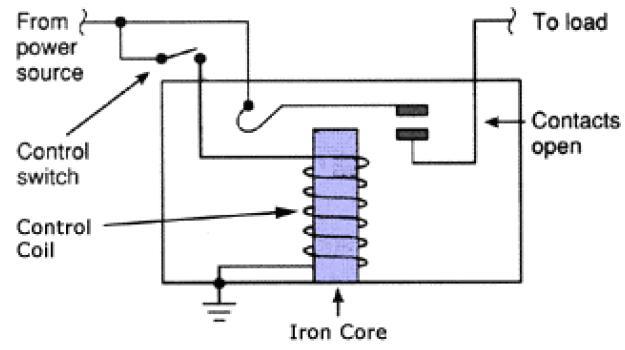


Fig: Working of relay

The basics for all the relays are the same. Take a look at a 4 – pin relay shown below. There are two colours shown. The green colour represents the control circuit and the red colour represents the load circuit. A small control coil is connected onto the control circuit. A switch is connected to the load. This switch is controlled by the coil in the control circuit. Now let us take the different steps that occur in a relay.

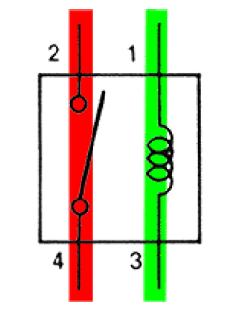


Fig: Relay Operation

As shown in the circuit, the current flowing through the coils represented by pins 1 and 3 causes a magnetic field to be aroused. This magnetic field causes the closing of the pins 2 and 4. Thus the switch plays an important role in the relay working. As it is a part of the load circuit, it is used to control an electrical circuit that is connected to it. Thus, when the relay in energized the current flow will be through the pins 2 and 4.

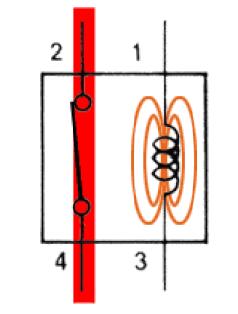
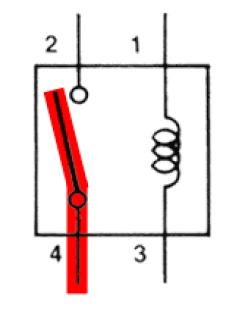


Fig: Energized Relay ON

As soon as the current flow stops through pins 1 and 3, the switch opens and thus the open circuit prevents the current flow through pins 2 and 4. Thus the relay becomes de-energized and thus in off position.



**Fig: De-Energized Relay OFF**

**3.3.2 TYPES OF RELAY**

There are some basic relay circuits that must be kept in our mind. They are

* **Voltage Suppression Relays**

As relays are used in industrial purposes very often, they are mostly controlled with the help of computers. But when relays are controlled with such devices, there will surely be the presence of semi-conductors like transistors. This will in turn cause the presence of voltage spikes. As a result, it is really necessary to introduce voltage suppression devices, otherwise they will clearly destroy the transistors.

This voltage suppression can be introduced in two ways. Either the computer provides the suppression or the relay provides the suppression. If the relay provides the suppression they are called voltage-suppression relays.  In relays voltage suppression is provided with the help of resistors of high value and even diodes and capacitors. Out of these diodes and resistors are more commonly used. Whatever device is used, it will be clearly stated in the relay. Take a look at the diagram of a voltage suppressed relay with the help of a diode.

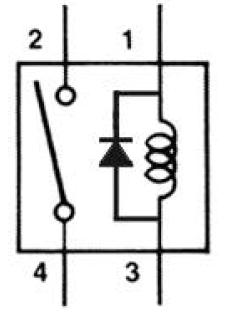


Fig: voltage suppression relay using diode

* **De-spiking Diode Relays**

A diode in the reverse-biased position is connected in parallel with the relay coil. As there is no flow of current due to such a connection, an open circuit of the relay will cause the current to stop flowing through the coil. This will have effect on the magnetic field. The magnetic field will be decreased instantly. This will cause the rise of an opposite voltage with very high reverse polarity to be induced. This is mainly caused because of the magnetic lines of force that cut the armature coil due to the open circuit. Thus the opposite voltage rises until the diode reaches 0.7 volts. As soon as this cut-off voltage is achieved, the diode becomes forward-biased. This causes a closed circuit in the relay, causing the entire voltage to pass through the load. The current thus produced will be flowing through the circuit for a very long time. As soon as the voltage is completely drained, this current flow will also stop. Take a look at the figure given below.

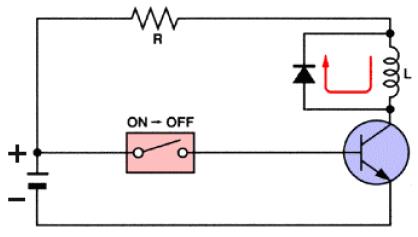


Fig: De-spiking diode relays

* **De-spiking Resistor Relays**

A resistor is almost efficient as that of a diode. It can not only suppress the voltage spikes efficiently, but also allows the entire current to flow through it when the relay is in the on position. Thus the current flow through it will also be very high. To reduce this, the value of the resistance should be as high as 1 Kilo Ohm. But, as the value of the resistors increases the voltage spiking capability of the relay decreases.  Take a look at the circuit diagram below to understand more.

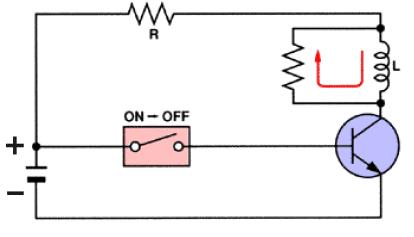


Fig: De-spiking resistor relays

Here is a detailed list of the different types of relays.

### 1. Latching Relay

Latching relays are also called impulse relays. They work in the bistable mode, and thus have two relaxing states. They are also called keep relays or stay relays because as soon as the current towards this relay is switched off, the relay continues the process that it was doing in the last state. This can be achieved only with a solenoid which is operating in a ratchet and cam mechanism.  It can also be done by an over-centre spring mechanism or a permanent magnet mechanism in which, when the coil is kept in the relaxed point, the over-centre spring holds the armature and the contacts in the right spot. This can also be done with the help of a remanent core.

In the ratchet and cam method, power consumption occurs only for a particular time. Hence it is more advantageous than the others.

### 2. Reed Relay

These types of relays have been given more importance in the contacts. In order to protect them from atmospheric protection they are safely kept inside a vacuum or inert gas.  Though these types of relays have a very low switching current and voltage ratings, they are famous for their switching speeds.

### 3. Polarized Relay

This type of relay has been given more importance on its sensitivity. These relays have been used since the invention of telephones. They played very important roles in early telephone exchanges and also in detecting telegraphic distortion. The sensitivity of these relays are very easy to adjust as the armature of the relay is placed between the poles of a permanent magnet.

### 4. Buchholz Relay

This relay is actually used as a safety device. They are used for knowing the amount of gas present in large oil-filled transformers. They are designed in such a way that they produce a warning if it senses either the slow production of gas or fast production of gas in the transformer oil.

### 5. Overload protection Relay

As the name implies, these relays are used to prevent the electric motors from damage by over current and short circuits. For this the heating element is kept in series with the motor. Thus when over heat occurs the bi-metallic strip connected to the motor heats up and in turn releases a spring to operate the contacts of the relay.

### 6. Mercury Wetted Relay

This relay is almost similar to the reed relay explained earlier. The only difference is that instead of inert gases, the contacts are wetted with mercury. This makes them more position sensitive and also expensive. They have to be vertically mounted for any operation. They have very low contact resistance and so can be used for timing applications. Due to these factors, this relay is not used frequently.

### 7. Machine Tool Relay

This is one of the most famous industrial relay. They are mainly used for the controlling of all kinds of machines. They have a number of contacts with easily replaceable coils. This enabkes them to be easily converted from NO contact to NC contact. Many types of these relays can easily be setup in a control panel. Though they are very useful in industrial applications, the invention of PLC has made them farther away from industries.

### 8. Contacor Relay

This is one of the most heavy load relay ever used. They are mainly used in switching electric motors. They have a wide range of current ratings from a few amps to hundreds. The contacts of these relays are usually made with alloys containing a small percentage of silver. This is done so as to avoid the hazardous effects of arcing. These type of relays are mainly categorized in the rough use areas. So, they produce loud noises while operated and hence cannot be used in places where noise is a problem.

### 9. Solid State relay

SSR relays, as its name implies are designed with the help of solid state components. As they do not have any moving objects in their design they are known for their high reliability.

### 10. Solid State Contactor Relay

These relays combine both the features of solid state relays and contactor relays. As a result they have a number of advantages. They have a very good heat sink and can be designed for the correct on-off cycles. They are mainly controlled with the help of PLC, micro-processors or microcontrollers.

**METHODOLOGY**

The various sensors are connected to the microcontroller. All these parameters are monitored on the Blynk app and then various devices can also be turned on accordingly. if we want to turn on the light and fan we can turn it on from anywhere. The water pump can also be turned on through the internet. If the PIR sensor gets activated thief is detected so the buzzer gets activate, and we can take the required action of informing to the police.

**INTERNET OF THINGS**

IoT (Internet of Things) is an advanced automation and analytics system which exploits networking, sensing, big data, and artificial intelligence technology to deliver complete systems for a product or service. These systems allow greater transparency, control, and performance when applied to any industry or system.

IoT systems have applications across industries through their unique flexibility and ability to be suitable in any environment. They enhance data collection, automation, operations, and much more through smart devices and powerful enabling technology.

IoT systems allow users to achieve deeper automation, analysis, and integration within a system. They improve the reach of these areas and their accuracy. IoT utilizes existing and emerging technology for sensing, networking, and robotics.

IoT exploits recent advances in software, falling hardware prices, and modern attitudes towards technology. Its new and advanced elements bring major changes in the delivery of products, goods, and services; and the social, economic, and political impact of those changes.

**4.1 IOT FEATURES**

The most important features of IoT include artificial intelligence, connectivity, sensors, active engagement, and small device use. A brief review of these features is given below −

* **AI** − IoT essentially makes virtually anything “smart”, meaning it enhances every aspect of life with the power of data collection, artificial intelligence algorithms, and networks. This can mean something as simple as enhancing your refrigerator and cabinets to detect when milk and your favorite cereal run low, and to then place an order with your preferred grocer.
* **Connectivity** − New enabling technologies for networking, and specifically IoT networking, mean networks are no longer exclusively tied to major providers. Networks can exist on a much smaller and cheaper scale while still being practical. IoT creates these small networks between its system devices.
* **Sensors** − IoT loses its distinction without sensors. They act as defining instruments which transform IoT from a standard passive network of devices into an active system capable of real-world integration.
* **Active Engagement** − Much of today's interaction with connected technology happens through passive engagement. IoT introduces a new paradigm for active content, product, or service engagement.
* **Small Devices** − Devices, as predicted, have become smaller, cheaper, and more powerful over time. IoT exploits purpose-built small devices to deliver its precision, scalability, and versatility.

**4.2 IOT ADVANTAGES**

The advantages of IoT span across every area of lifestyle and business. Here is a list of some of the advantages that IoT has to offer −

* **Improved Customer Engagement** − Current analytics suffer from blind-spots and significant flaws in accuracy; and as noted, engagement remains passive. IoT completely transforms this to achieve richer and more effective engagement with audiences.
* **Technology Optimization** − The same technologies and data which improve the customer experience also improve device use, and aid in more potent improvements to technology. IoT unlocks a world of critical functional and field data.
* **Reduced Waste** − IoT makes areas of improvement clear. Current analytics give us superficial insight, but IoT provides real-world information leading to more effective management of resources.
* **Enhanced Data Collection** − Modern data collection suffers from its limitations and its design for passive use. IoT breaks it out of those spaces, and places it exactly where humans really want to go to analyze our world. It allows an accurate picture of everything.

**4.3 IOT SOFTWARE**

IoT software addresses its key areas of networking and action through platforms, embedded systems, partner systems, and middleware. These individual and master applications are responsible for data collection, device integration, real-time analytics, and application and process extension within the IoT network. They exploit integration with critical business systems (e.g., ordering systems, robotics, scheduling, and more) in the execution of related tasks.

## Data Collection

This software manages sensing, measurements, light data filtering, light data security, and aggregation of data. It uses certain protocols to aid sensors in connecting with real-time, machine-to-machine networks. Then it collects data from multiple devices and distributes it in accordance with settings. It also works in reverse by distributing data over devices. The system eventually transmits all collected data to a central server.

## Device Integration

Software supporting integration binds (dependent relationships) all system devices to create the body of the IoT system. It ensures the necessary cooperation and stable networking between devices. These applications are the defining software technology of the IoT network because without them, it is not an IoT system. They manage the various applications, protocols, and limitations of each device to allow communication.

## Real-Time Analytics

These applications take data or input from various devices and convert it into viable actions or clear patterns for human analysis. They analyze information based on various settings and designs in order to perform automation-related tasks or provide the data required by industry.

## Application and Process Extension

These applications extend the reach of existing systems and software to allow a wider, more effective system. They integrate predefined devices for specific purposes such as allowing certain mobile devices or engineering instruments access. It supports improved productivity and more accurate data collection.

**4.4 IOT TECHNOLOGY AND PROTOCOLS**

IoT primarily exploits standard protocols and networking technologies. However, the major enabling technologies and protocols of IoT are RFID, NFC, low-energy Bluetooth, low-energy wireless, low-energy radio protocols, LTE-A, and WiFi-Direct. These technologies support the specific networking functionality needed in an IoT system in contrast to a standard uniform network of common systems.

## NFC and RFID

RFID (radio-frequency identification) and NFC (near-field communication) provide simple, lowenergy, and versatile options for identity and access tokens, connection bootstrapping, and payments.

* RFID technology employs 2-way radio transmitter-receivers to identify and track tags associated with objects.
* NFC consists of communication protocols for electronic devices, typically a mobile device and a standard device.

## Low-Energy Bluetooth

This technology supports the low-power, long-use need of IoT function while exploiting a standard technology with native support across systems.

## Low-Energy Wireless

This technology replaces the most power hungry aspect of an IoT system. Though sensors and other elements can power down over long periods, communication links (i.e., wireless) must remain in listening mode. Low-energy wireless not only reduces consumption, but also extends the life of the device through less use.

## Radio Protocols

ZigBee, Z-Wave, and Thread are radio protocols for creating low-rate private area networks. These technologies are low-power, but offer high throughput unlike many similar options. This increases the power of small local device networks without the typical costs.

## LTE-A

LTE-A, or LTE Advanced, delivers an important upgrade to LTE technology by increasing not only its coverage, but also reducing its latency and raising its throughput. It gives IoT a tremendous power through expanding its range, with its most significant applications being vehicle, UAV, and similar communication.

## WiFi-Direct

WiFi-Direct eliminates the need for an access point. It allows P2P (peer-to-peer) connections with the speed of WiFi, but with lower latency. WiFi-Direct eliminates an element of a network that often bogs it down, and it does not compromise on speed or throughput.

**4.5 BLYNK APP**

Blynk is a toolset for all makers, badass inventors, designers, teachers, nerds and geeks who would love to use their smartphones to control electronics like Arduino, RaspberryPi and similar ones. We’ve done all the hard work of establishing internet connection, building an app and writing hardware code.

With Blynk, you simply snap together an amazing interface from various widgets we provide, upload the example code to your hardware and enjoy seeing first results in under 5 minutes! It works perfectly for newbie makers and saves tons of time for evil geniuses.

Blynk will work with all popular boards and shields. We wanted to give you full freedom when deciding how to plug Blynk into your existing or new project. You will also enjoy the convenience of Blynk Cloud. Which is, by the way is free and open-source.

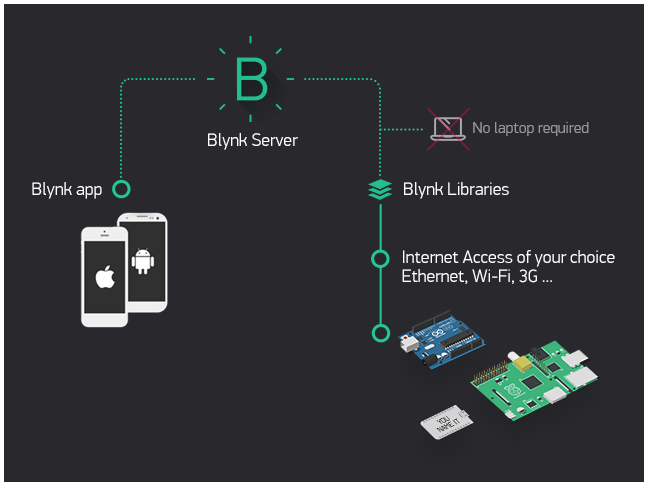
Imagine a prototyping board on your smartphone where you drag and drop buttons, sliders, displays, graphs and other functional widgets. And in a matter of minutes these widgets can control Arduino and get data from it.

Blynk is not an app that works only with a particular shield. Instead, it's been designed to support the boards and shields you are already using. And it works on iOs and Android.

**UPD:** Blynk also works over USB. This means you can tinker with the app by connecting it to your laptop or desktop while waiting for some internet shield to arrive.

**Blynk works over the Internet.**So the one and only requirement is that your hardware can talk to the Internet.

No matter what type of connection you choose - Ethernet, Wi-Fi or maybe this new ESP8266 everyone is talking about – Blynk libraries and example sketches will get you online, connect to Blynk Server and pair up with your smartphone.

****

**FIG:** Blynk architecture

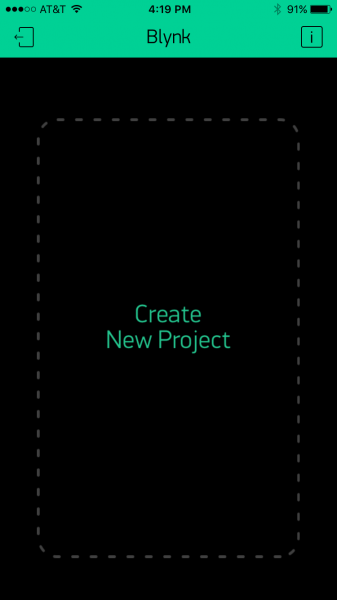
Currently, Blynk libraries work with:

* USB
* Ethernet shield
* WiFi shield
* Arduino with Ethernet
* Arduino YÚN (testing in progress)
* ESP8266
* Raspberry Pi (Blynk will communicate with Pi's GPIOs)
* more Arduino compatible shields and boards (this list will be updated as we test the compatibility)

It's not that easy to take Arduino out of your home network, so we've built a Blynk server. It handles all the authentication and communication, and also keeps an eye on your board while the smartphone is offline. Blynk server runs on Java and is open-source. You will be able to run it locally if you really need to. Messaging between mobile apps , Blynk Server and Arduino is based on a simple, lightweight and fast binary protocol over TCP/IP sockets.

**4.5.1 CREATING A PROJECT IN BLYNK APP**

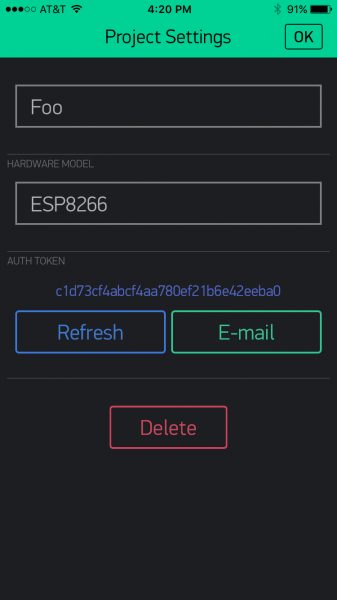
After downloading the app, create an account and log in. Welcome to Blynk!

[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/blynk-blank.PNG)

You’ll also need to install the **Blynk Arduino Library**, which helps generate the firmware running on your ESP8266. Download the latest release from Blynk’s GitHub repo, and follow along with the directions there to install the required libraries.

### Create a Blynk Project

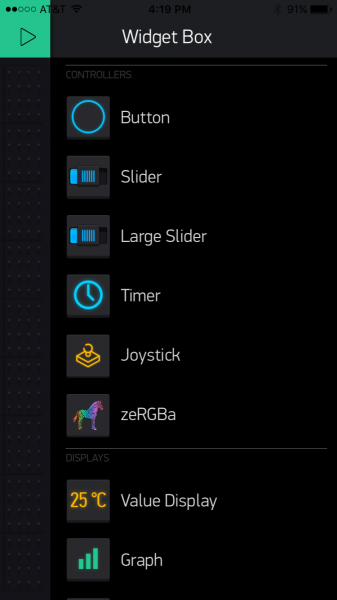
Next, click the “Create New Project” in the app to create a new Blynk app. Give it any name you please, just make sure the “Hardware Model” is set to **ESP8266**.

[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/Blynk-new.PNG)

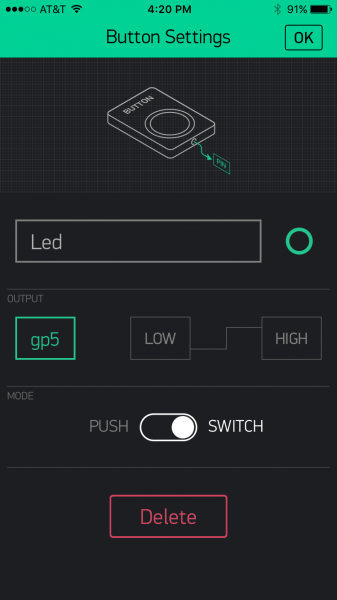
The **Auth Token** is very important – you’ll need to stick it into your ESP8266’s firmware. For now, copy it down or use the “E-mail” button to send it to yourself.

### Add Widgets to the Project

Then you’ll be presented with a blank new project. To open the widget box, click in the project window to open.

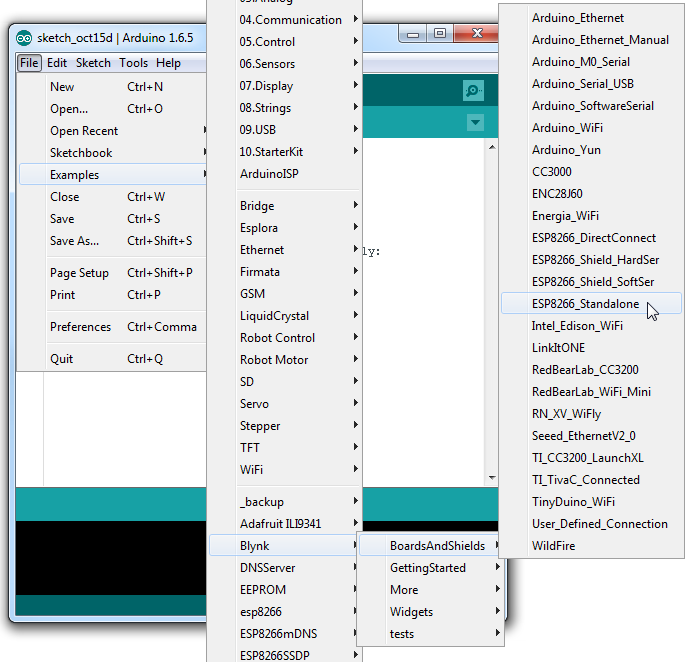
[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/Blynk-widgetBox.PNG)

Add a **Button**, then click on it to change its settings. Buttons can toggle outputs on the ESP8266. Set the button’s output to **gp5**, which is tied to an LED on the Thing Dev Board. You may also want to change the action to “Switch.”

[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/Blynk-Button.PNG)

### Upload the Blynk Firmware

Now that your Blynk project is set up, open Arduino and navigate to the **ESP8266\_Standalone** example in the **File** > **Examples** > **Blynk** > **BoardsAndShields** menu.

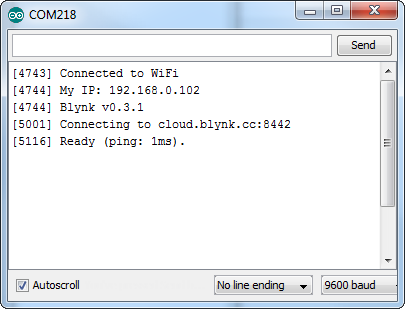
[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/Blynk-example.png)

Before uploading, make sure to paste your **authoriazation token** into the auth[] variable. Also make sure to **load your WiFi network settings into the** Blynk.begin(auth, "ssid", "pass") function.

Then upload!

### Run the Project

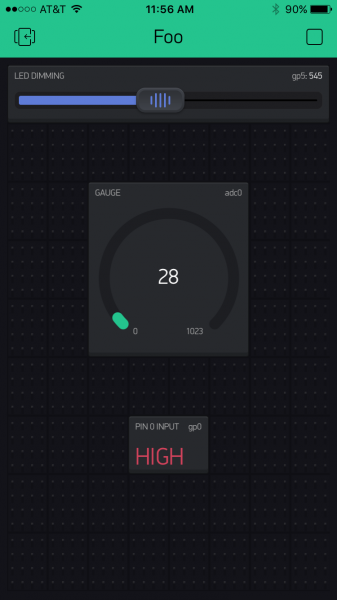
After the app has uploaded, open the serial monitor, setting the baud rate to 9600. Wait for the “Ready (ping: xms).” message.

[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/Blynk-Serial.png)

Then click the “Run” button in the top right corner of the Blynk app. Press the button and watch the LED!

[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/esp8266-blynk.jpg)

Then add more widgets to the project. They should immediately work on the ESP8266 without uploading any new firmware.

[](https://cdn.sparkfun.com/assets/learn_tutorials/4/4/5/Blynk-fullExample.PNG)

You can add analog output sliders, digital input monitors, and analog input gauges.

**ARCHITECTURE DIAGRAM (HOW TO ACESS VALUES FROM THE CLOUD AND VIEW IN BLYNK APP)**

USER LOGIN

CHECK

FILE SHARE (UPLOAD)

UNAUTHORISED

GENERATAE KEY

ENCRYPTION

FILE SEARCH (DOWNLOAD)

END PROCESS

DECRYPTION

To download a file from cloud the user logins the systems checks weather he is an authorised user or not. If he is an authorised user then he can share file i.e. he can upload the values to the cloud and then the key generation for encryption and when we search for a specific file i.e. want to download a file the decryption process takes place and then we can access the values.

**Performance Analysis**

The sprawling nature of IoT requires comprehensive management of the entire network, wired and wireless, right to the edge as devices – smart and not so smart – seek access and data transfer to core network components. This is why the switch is key. All the connected devices and sensors are transmitting data on the network, but sending data from devices straight to the data centre can be inefficient, cause bottlenecks on the network, and impact performance.

An intelligent network needs to extend functionality right to the edge so data can be analysed and processed on the way to the core, or from device to device. To manage the increased flow of IoT traffic, switches at the edge of the network will need to offer enhanced security and integrated analytics.

Not every device is smart. Poorly secured 'smart' devices such as smart watches and activity trackers pose a threat to essential network security – as do traditional 'dumb' devices such as door locks.

Simply monitoring and controlling the flow of packets to and from IoT devices is not enough to guarantee security. All devices right out to the network edge must be made smarter by the network management and the switches on the network.

**3.4 Logical database requirements**

Cloud Storage is a service that allows to save data on offsite storage system managed by third-party and is made accessible by a **web services API.**

## Storage Devices

Storage devices can be broadly classified into two categories:

* Block Storage Devices
* File Storage Devices

### Block Storage Devices

The **block storage devices** offer raw storage to the clients. These raw storage are partitioned to create volumes.

### File Storage Devices

The **file Storage Devices** offer storage to clients in the form of files, maintaining its own file system. This storage is in the form of Network Attached Storage (NAS).

## Cloud Storage Classes

Cloud storage can be broadly classified into two categories:

* Unmanaged Cloud Storage
* Managed Cloud Storage

### Unmanaged Cloud Storage

Unmanaged cloud storage means the storage is preconfigured for the customer. The customer can neither format, nor install his own file system or change drive properties.

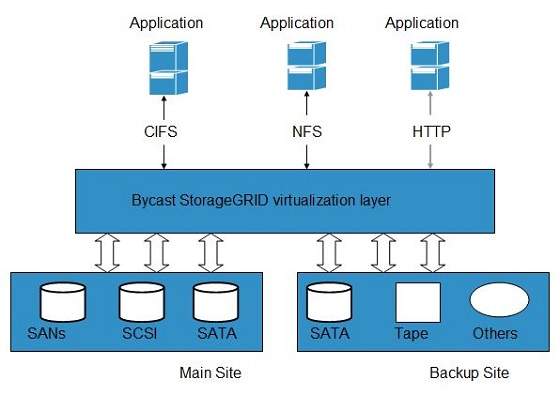
### Managed Cloud Storage

Managed cloud storage offers online storage space on-demand. The managed cloud storage system appears to the user to be a raw disk that the user can partition and format.

## Creating Cloud Storage System

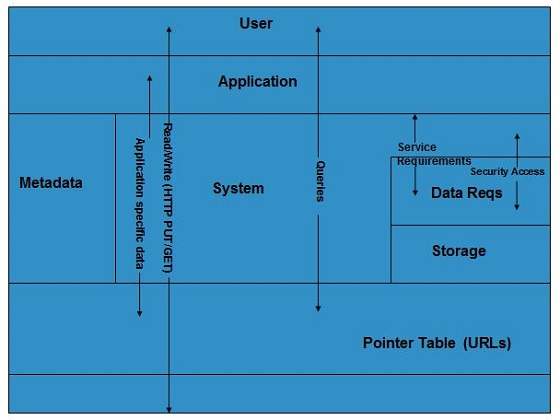
The cloud storage system stores multiple copies of data on multiple servers, at multiple locations. If one system fails, then it is required only to change the pointer to the location, where the object is stored.

To aggregate the storage assets into cloud storage systems, the cloud provider can use storage virtualization software known as **StorageGRID.** It creates a virtualization layer that fetches storage from different storage devices into a single management system. It can also manage data from **CIFS** and **NFS** file systems over the Internet. The following diagram shows how StorageGRID virtualizes the storage into storage clouds:



## Virtual Storage Containers

The **virtual storage containers** offer high performance cloud storage systems. **Logical Unit Number (LUN)** of device, files and other objects are created in virtual storage containers. Following diagram shows a virtual storage container, defining a cloud storage domain:



## Challenges

Storing the data in cloud is not that simple task. Apart from its flexibility and convenience, it also has several challenges faced by the customers. The customers must be able to:

* Get provision for additional storage on-demand.
* Know and restrict the physical location of the stored data.
* Verify how data was erased.
* Have access to a documented process for disposing of data storage hardware.
* Have administrator access control over data.

**3.5 Design Constraints**

IoT introduces a deeper level of automation which can have control over critical systems, and systems impacting life and property. When these systems fail or malfunction, they can cause substantial damage; for example, if an IoT furnace control system experiences a glitch, it may fail in an unoccupied home and cause frozen pipes and water damage. This forces organizations to create measures against it.

IoT devices expose an entire network and anything directly impacted to the risk of attacks. Though those connections deliver powerful integration and productivity, they also create the perfect opportunity for mayhem like a hacked stove or fire safety sprinkler system. The best measures against this address the most vulnerable points, and provide custom protections such as monitoring and access privileges.

Some of the most effective measures against attacks prove simple −

* **Built-in Security** − Individuals and organizations should seek hardened devices, meaning those with security integrated in the hardware and firmware.
* **Encryption** − This must be implemented by the manufacturer and through user systems.
* **Risk Analysis** − Organizations and individuals must analyze possible threats in designing their systems or choosing them.
* **Authorization** − Devices, whenever possible, must be subject to privilege policies and access methods.

**3.6 Software System Attributes**

Data intensive applications on cloud requires high network bandwidth, which results in high cost. Low bandwidth does not meet the desired computing performance of cloud application.

**3.6.1 Reliability**

Every connected device creates opportunities for attackers. These vulnerabilities are broad, even for a single small device. The risks posed include data transfer, device access, malfunctioning devices, and always-on/always-connected devices.

The main challenges in security remain the security limitations associated with producing lowcost devices, and the growing number of devices which creates more opportunities for attacks.

**3.6.2 Availability**

Various Cloud Computing platforms are available today. The following table contains the popular Cloud Computing platforms:

|  |  |
| --- | --- |
| **SN** | **Platform Description** |
|  | **Salesforce.com**  This is a Force.com development platform. This provides a simple user interface and lets users log in, build an app, and push it in the cloud. |
|  | **Appistry**  The Appistry's CloudIQ platform is efficient in delivering a runtime application. This platform is very useful to create scalable and service oriented applications. |
|  | **AppScale**  The AppScale is an open source platform for App Engine of Google applications. |
|  | **AT&T**  The AT&T allows access to virtual servers and manages the virtualization infrastructure. This virtualization infrastructure includes network, server and storage. |
|  | **Engine Yard**  The Engine Yard is a rails application on cloud computing platform. |
|  | **Enomaly**  Enomaly provides the Infrastructure-as-a-Service platform. |
|  | **FlexiScale**  The FlexiScale offers a cloud computing platform that allows flexible, scalable and automated cloud infrastructure. |
|  | **GCloud3**  The GCloud3 offers private cloud solution in its platform. |
|  | **Gizmox**  The Gizmox Visual WebGUI platform is best suited for developing new web apps and modernize the legacy apps based on ASP.net, DHTML, etc. |
|  | **GoGrid**  The GoGrid platform allows the users to deploy web and database cloud services. |
|  | **Google**  The Google's App Engine lets the users build, run and maintain their applications on Google infrastructure. |
|  | **LongJump**  The LongJump offers a business application platform, a Platform-as-a-Service (PaaS). |
|  | **Microsoft**  The Microsoft Windows Azure is a cloud computing platform offering an environment to create cloud apps and services. |
|  | **OrangeScape**  OrangeScape offers a Platform-as-a-Service (Paas) for non-programmers. Building an app is as easy as spreadsheet. |
|  | **RackSpace**  The RackSpace provides servers-on-demand via a cloud-driven platform of virtualized servers. |
|  | **Amazon EC2**  The Amazon EC2 (Elastic Compute Cloud) lets the users configure and control computing resources while running them on Amazon environment. |

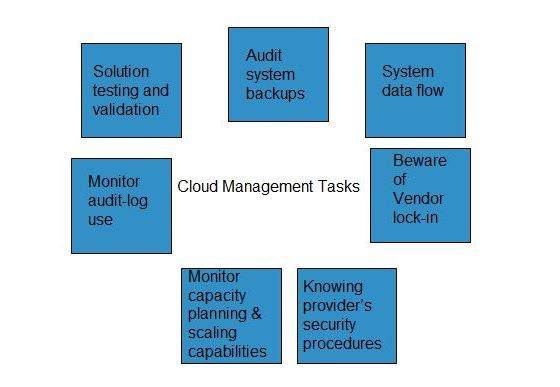
**3.6.3 Security**

Security issues plague IoT −

* **Unpredictable Behaviour** − The sheer volume of deployed devices and their long list of enabling technologies means their behaviour in the field can be unpredictable. A specific system may be well designed and within administration control, but there are no guarantees about how it will interact with others.
* **Device Similarity** − IoT devices are fairly uniform. They utilize the same connection technology and components. If one system or device suffers from a vulnerability, many more have the same issue.
* **Problematic Deployment** − One of the main goals of IoT remains to place advanced networks and analytics where they previously could not go. Unfortunately, this creates the problem of physically securing the devices in these strange or easily accessed places.
* **Long Device Life and Expired Support** − One of the benefits of IoT devices is longevity, however, that long life also means they may outlive their device support. Compare this to traditional systems which typically have support and upgrades long after many have stopped using them. Orphaned devices and abandonware lack the same security hardening of other systems due to the evolution of technology over time.
* **No Upgrade Support** − Many IoT devices, like many mobile and small devices, are not designed to allow upgrades or any modifications. Others offer inconvenient upgrades, which many owners ignore, or fail to notice.
* **Poor or No Transparency** − Many IoT devices fail to provide transparency with regard to their functionality. Users cannot observe or access their processes, and are left to assume how devices behave. They have no control over unwanted functions or data collection; furthermore, when a manufacturer updates the device, it may bring more unwanted functions.
* **No Alerts** − Another goal of IoT remains to provide its incredible functionality without being obtrusive. This introduces the problem of user awareness. Users do not monitor the devices or know when something goes wrong. Security breaches can persist over long periods without detection.

**3.6.4 Maintainability**

The cloud provider performs a number of tasks to ensure efficient use of cloud resources. Here, we will discuss some of them:



### Audit System Backups

It is required to audit the backups timely to ensure restoring of randomly selected files of different users. Backups can be performed in following ways:

* Backing up files by the company, from on-site computers to the disks that reside within the cloud.
* Backing up files by the cloud provider.

It is necessary to know if cloud provider has encrypted the data, who has access to that data and if the backup is taken at different locations then the user must know the details of those locations.

### Data Flow of the System

The managers are responsible to develop a diagram describing a detailed process flow. This process flow describes the movement of data belonging to an organization throughout the cloud solution.

### Vendor Lock-In Awareness and Solutions

The managers must know the procedure to exit from services of a particular cloud provider. The procedures must be defined to enable the cloud managers to export data of an organization from their system to another cloud provider.

### Knowing Provider’s Security Procedures

The managers should know the security plans of the provider for the following services:

* Multitenant use
* E-commerce processing
* Employee screening
* Encryption policy

### Monitoring Capacity Planning and Scaling Capabilities

The managers must know the capacity planning in order to ensure whether the cloud provider is meeting the future capacity requirement for his business or not.

The managers must manage the scaling capabilities in order to ensure services can be scaled up or down as per the user need.

### Monitor Audit Log Use

In order to identify errors in the system, managers must audit the logs on a regular basis.

### Solution Testing and Validation

When the cloud provider offers a solution, it is essential to test it in order to ensure that it gives the correct result and it is error-free. This is necessary for a system to be robust and reliable.

**3.6.5 Portability**

The whole model is portable and can be used in any part of the world. We can monitor a patients vital parameters from anywhere with internet.

**ADVANTAGES**

* The main advantages id physically challenged people can easily control the home appliances.
* Replace television, air condition remotes with a single mobile.
* High efficiency can be obtained.

DISADVANTAGES

* Requires high speed internet connectivity.

**CONCLUSION**

The home automation using Internet of Things has been experimentally proven to work satisfactorily by connecting simple appliances to it and the appliances were successfully controlled remotely through internet. The designed system not only monitors the sensor data, like temperature, gas, light, motion sensors, but also actuates a process according to the requirement, for example switching on the light when it gets dark or the fan and the water pump can also be turned on. It also stores the sensor parameters in the cloud in a timely manner. This will help the user to analyze the condition of various parameters in the home anytime anywhere.

**FUTURE WORK**

Using this system as framework, the system can be expanded to include various other options which could include home security feature like capturing the photo of a person moving around the house and storing it onto the cloud. This will reduce the data storage than using the CCTV camera which will record all the time and stores it. The system can be expanded for energy monitoring, or weather stations. This kind of a system with respective changes can be implemented in the hospitals for disable people or in industries where human invasion is impossible or dangerous, and it can also be implemented for environmental monitoring.

**REFERENCES**

[1] Teymourzadeh, Rozita, et al. "Smart GSM Based Home Automation System." *Systems, Process & Control (ICSPC), 2013 IEEE Conference on*. IEEE, 2013.

[2] Singh, Navab, et al. "Remotely controlled home automation system."*Advances in Engineering and Technology Research (ICAETR), 2014 International Conference on*. IEEE, 2014.

[3] PavithraD,IoT based Monitoring and Control System for Home Automation,Proceedings of 2015 Global Conference on Communication Technologies(GCCT 2015)

[4] Wenbo, Yan, Wang Quanyu, and GaoZhenwei. "Smart home implementation based on Internet and WiFi technology." *Control Conference (CCC), 2015 34th Chinese*. IEEE, 2015.

[5] Gurek, Alper, et al. "An android based home automation system." *High Capacity Optical Networks and Enabling Technologies (HONET-CNS), 2013 10th International Conference on*. IEEE, 2013.

[6] Bhide, VishwajeetHari, and SanjeevWagh. "i-learning IoT: An intelligent self-learning system for home automation using IoT." *Communications and Signal Processing (ICCSP), 2015 International Conference on*. IEEE, 2015.

[7] Gamba, Mattia, Alessandro Gonella, and Claudio E. Palazzi. "Design issues and solutions in a modern home automation system." *Computing, Networking and Communications (ICNC), 2015 International Conference on*. IEEE, 2015.

[8] Gubbi, Jayavardhana, et al. "Internet of Things (IoT): A vision, architectural elements, and future directions." *Future Generation Computer Systems* 29.7 (2013): 1645-1660.

[9] Karimi, Kaivan, and Gary Atkinson. "What the Internet of Things (IoT) needs to become a reality." *White Paper, FreeScale and ARM* (2013).

[10] Stankovic, John. "Research directions for the internet of things." *Internet of Things Journal, IEEE* 1.1 (2014): 3-9.F

[11] "Understanding the Internet of Things (IoT) “, July 2014. Gubbi, Jayavardhana, et al. "Internet of Things (IoT): A vision, architectural elements, and directions." *Future Generation Computer Systems* 29.7 (2013): 1645-1660.