INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING
IJTE

ISSN: 2057-5688

SMART OTP BASED DEVICE CONTROL SYSTEM

[1] MR.C.P. SUDHAKAR, [2]B AKHIL, [3]P NITHISH, [4]Y NARASIMHA, [5]G HARIKRISHNA, [6]J SAI CHAITHANYA

[1]Assistant Professor, Dept. Of EEE, DRK INSTITUTE OF SCIENCE & TECHNOLOGY, Hyderabad,

[2,3,4,5,6]BTech Student, Dept. Of EEE, DRK INSTITUTE OF SCIENCE & TECHNOLOGY, Hyderabad

Abstract: *In this project, we are going to make a smart OTP-based locking system. This smart lock can generate a new password every time you unlock it, which further enhances your security level. This new device is much safer than the traditional key-based system and electronic wireless lock system. If you are still using the key-based system, you are likely to land in a big problem if your key gets lost or stolen. The electronic wireless lock system is not safe either. You might forget the password and there is also a high risk being hacked. The main objective of this project is to set up a basic devices control system in an Arduino Uno with a 4x4 keypad, 16x2 LCD and Relay, DC fan, Led, every time we enter a random password it checks with the controller whether the OTP is correct or not. If it is a correct OTP, the Fans & lights ON. Once that is ready, I have used an ESP8266 which gets connected to the WiFi network and sends string values to the Twilit API.*

I. INTRODUCTION

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, sometimes with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale.

Physically embedded systems range from portable devices such as digital watches and MP3 players, to large

ISSN: 2057-5688

stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single micro controller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not an exactly defined term, as many systems have some element of profitability. For example, Handheld computers share some elements with embedded systems — such as the operating systems and microprocessors which power them — but are not truly embedded systems, because they allow different applications to be load and peripherals to be connected.

An embedded system is some combination of computer hardware and software, either fixed in capability or programmable, that is specifically designed for a particular kind of application device. Industrial machines, automobiles, medical equipment, cameras, household appliances, airplanes, vending machines, and toys (as well as the more obvious cellular phone and PDA) are among the myriad possible hosts of an embedded system. Embedded systems that are programmable are provided with a programming interface, and embedded systems programming is a specialized occupation. Certain operating systems or language platforms are tailored for the embedded market, such as Embedded Java and Windows XP Embedded. However, some low-end consumer products use very inexpensive microprocessors and limited storage, with the application and operating system both part of a single program. The program is written permanently into the system's memory in this case, rather than being loaded into RAM (random access memory), as programs on a personal computer are.

II Literature survey

This section contains the Literature review and the theoretical details about the project. Infrared Optical Wireless Communication for Smart Door Locks Using Smart phones With the recent rapid advancements in the Internet of Things (loT), one of the applications being developed is that of smart door lock (SDL) systems. SDL are intended to over high security, easy access and easy sharing. Unlike existing SDL solutions that mostly use bio metrics or crunched RF spectrum, we uniquely propose to use Infrared (IR) optical wireless signal (OWS) using IR light

ISSN: 2057-5688

emitting diode (LED) of smart phones. We designed and developed a complete system of Android smart phone app including physical layer encoding, a cloud server and programmable hardware prototypes using Arduino as well as Raspberry Pi. Opt lock includes multi-level security schemes including user registration, authentication and authorization using one-time-password (OTP). This extensive experiments show 100 accuracy with 1.33 Kbps of average data rate is achieved up to 20 meters of distance between a smart phone and a lock. It allows convenient remote access, easy access control and sharing as well as high security.

III HARDWARE COMPONENTS

POWER SUPPLY UNIT

The power supply for this system is shown below.

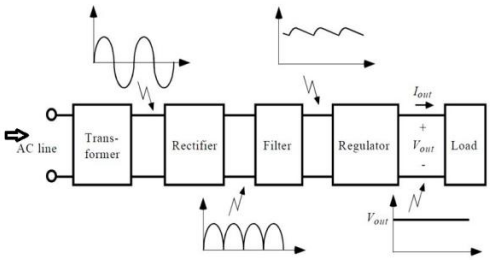


Figure.4.1.power supply

Transformer:

Transformer is a static device used to convert the voltage from one level to another level without change its frequency. There are two types of transformers

1. Step-up transformer
2. Step-down transformer

Step-up transformer converts low voltage level into high voltage level without change its frequency.

Step-down transformer converts high voltage level into low voltage level without change its frequency.

In this project we using step-down transformer which converts 230V AC to 12V AC [or] 230V AC to 5V as shown below.

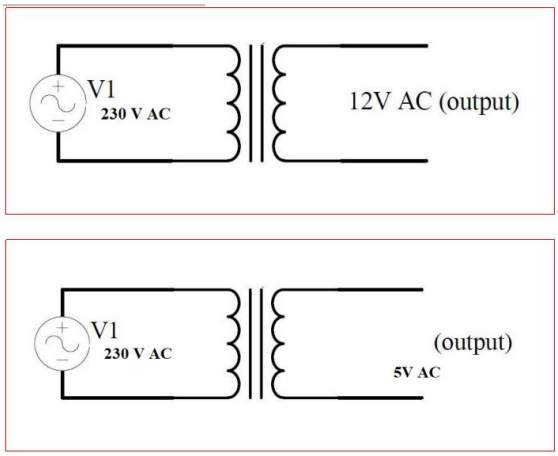


Figure.4.2.Transformers

Diodes:

Diodes allow electricity to flow in only one direction. The arrow of the circuit

INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING
IJTE
ISSN: 2057-5688

symbol shows the direction in which the current can flow. Diodes are the electrical version of a valve and early diodes were actually called valves.

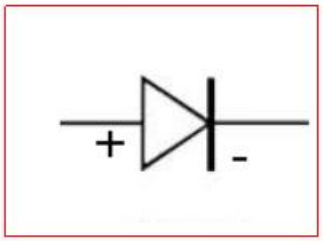


Figure.4.3. Diode Symbol

A **diode** is a device which only allows current to flow through it in one direction. In this direction, the diode is said to be 'forward-biased' and the only effect on the signal is that there will be a voltage loss of around 0.7V. In the opposite direction, the diode is said to be 'reverse-biased' and no current will flow through it.

Rectifier

The purpose of a rectifier is to convert an AC waveform into a DC waveform (OR) Rectifier converts AC current or voltages into DC current or voltage. There are two different rectification circuits, known as **'half-wave'** and **'full-wave' rectifiers**. Both use components called **diodes** to convert **AC into DC**.

The Half-wave Rectifier

The half-wave rectifier is the simplest type of rectifier since it only uses one diode, as shown in figure.

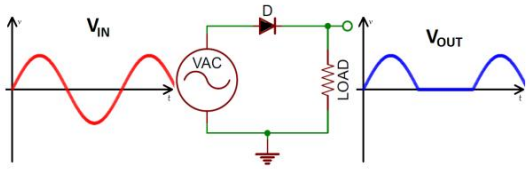


Figure.4.4Half Wave Rectifier

Figure 2 shows the AC input waveform to this circuit and the resulting output. As you can see, when the AC input is positive, the diode is forward-biased and lets the current through. When the AC input is negative, the diode is reverse-biased and the diode does not let any current through, meaning the output is 0V. Because there is a 0.7V voltage loss across the diode, the peak output voltage will be 0.7V less than Vs.

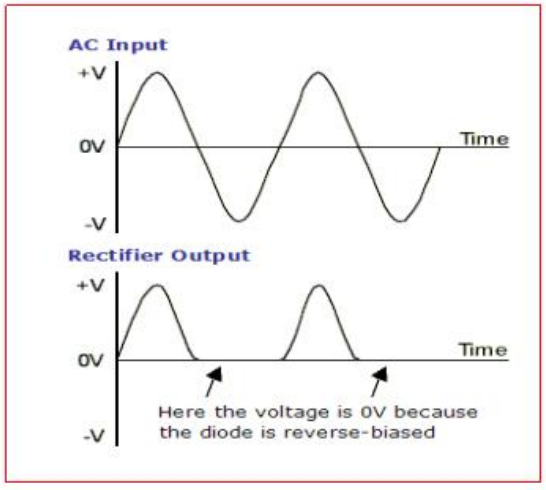


Figure.4.5Half-Wave Rectification

ISSN: 2057-5688

While the output of the half-wave rectifier is DC (it is all positive), it would not be suitable as a power supply for a circuit. Firstly, the output voltage continually varies between 0V and Vs-0.7V, and secondly, for half the time there is no output at all.

The Full-wave Bridge Rectifier

The circuit in figure 3 addresses the second of these problems since at no time is the output voltage 0V. This time four diodes are arranged so that both the positive and negative parts of the AC waveform are converted to DC. The resulting waveform is shown in figure 4.

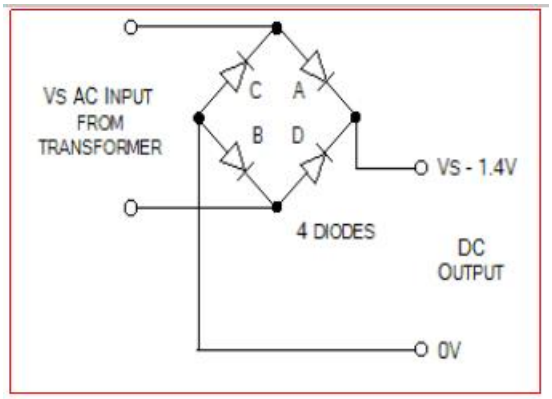


Figure.4.6. Full-Wave Rectifier

IV SOFTWARES

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a micro-controller) and a ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

The key features are:

- Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
- You can control your board functions by sending a set of instructions to the micro-controller on the board via Arduino IDE (referred to as uploading software).
- Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
- Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.
- Finally, Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

ISSN: 2057-5688

After learning about the main parts of the Arduino UNO board, we are ready to learn how to set up the Arduino IDE. Once we learn this, we will be ready to upload our program on the Arduino board.

Arduino data types:

Data types in C refers to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in the storage and how the bit pattern stored is interpreted.

The following table provides all the data types that you will use during Arduino programming.

Void:

The void keyword is used only in function declarations. It indicates that the function is expected to return no information to the function from which it was called.

Example:

```
Void Loop ( )
{
// rest of the code
}
```

Boolean:

A Boolean holds one of two values, true or false. Each Boolean variable occupies one byte of memory.

Example:

Boolean state= false ; // declaration of variable with type boolean and initialize it with false.

Boolean state = true ; // declaration of variable with type boolean and initialize it with false.

Char:A data type that takes up one byte of memory that stores a character value. Character literals are written in single quotes like this: 'A' and for multiple characters, strings use double quotes: "ABC".

However, characters are stored as numbers. You can see the specific encoding in the ASCII chart. This means that it is possible to do arithmetic operations on characters, in which the ASCII value of the character is used. For example, 'A' + 1 has the value 66, since the ASCII value of the capital letter A is 65.

Example:

Char chr_a = ‘a’ ;//declaration of variable with type char and initialize it with character a.

ISSN: 2057-5688

Char chr_c = 97 ;//declaration of variable with type char and initialize it with character 97

Unsigned char:

Unsigned char is an unsigned data type that occupies one byte of memory. The unsigned char data type encodes numbers from 0 to 255.

Example:

Unsigned Char chr_y = 121 ; // declaration of variable with type Unsigned char and initialize it with character y

Byte:

A byte stores an 8-bit unsigned number, from 0 to 255.

Example:

byte m = 25 ;//declaration of variable with type byte and initialize it with 25

int:

Integers are the primary data-type for number storage. **int** stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

The **int** size varies from board to board. On the Arduino Due, for example, an **int** stores a 32-bit (4-byte) value. This yields a range of -2,147,483,648 to 2,147,483,647 (minimum value of -2^31 and a maximum value of (2^31) - 1).

Example:

int counter = 32 ;// declaration of variable with type int and initialize it with 32.

Unsigned int:

Unsigned int (unsigned integers) are the same as int in the way that they store a 2 byte value. Instead of storing negative numbers, however, they only store positive values, yielding a useful range of 0 to 65,535 (2^16) - 1). The Due stores a 4 byte (32-bit) value, ranging from 0 to 4,294,967,295 (2^32 - 1).

Example:

Unsigned int counter= 60 ; // declaration of variable with type unsigned int and initialize it with 60.

Word:

On the Uno and other ATMEGA based boards, a word stores a 16-bit unsigned number. On the Due and Zero, it stores a 32-bit unsigned number.

Example

ISSN: 2057-5688

word w = 1000 ;//declaration of variable with type word and initialize it with 1000.

Long:

Long variables are extended size variables for number storage, and store 32 bits (4 bytes), from 2,147,483,648 to 2,147,483,647.

Example:

Long velocity= 102346 ;//declaration of variable with type Long and initialize it with 102346

Unsigned long: Unsigned long variables are extended size variables for number storage and store 32 bits (4 bytes). Unlike standard longs, unsigned longs will not store negative numbers, making their range from 0 to 4,294,967,295 (2^32 - 1).

Example:

Unsigned Long velocity = 101006 ;// declaration of variable with type Unsigned Long and initialize it with 101006.

Short:

A short is a 16-bit data-type. On all Arduino (At Mega and ARM based), a short stores a 16-bit (2-byte) value. This yields a range of -32,768 to 32,767 (minimum value of -2^15 and a maximum value of (2^15) - 1).

Example:

short val= 13 ;//declaration of variable with type short and initialize it with 13

Float:

Data type for floating-point number is a number that has a decimal point. Floating-point numbers are often used to approximate the analog and continuous values because they have greater resolution than integers.

Floating-point numbers can be as large as 3.4028235E+38 and as low as 3.4028235E+38. They are stored as 32 bits (4 bytes) of information.

Example:

float num = 1.352;//declaration of variable with type float and initialize it with 1.352.

Double:

On the Uno and other ATMEGA based boards, Double precision floating-point number occupies four bytes. That is, the double implementation is exactly the same as the float, with no gain in precision.

ISSN: 2057-5688

On the Arduino Due, doubles have 8-byte (64 bit) precision.

Example:

double num = 45.352 ;// declaration of variable with type double and initialize it with 45.352.

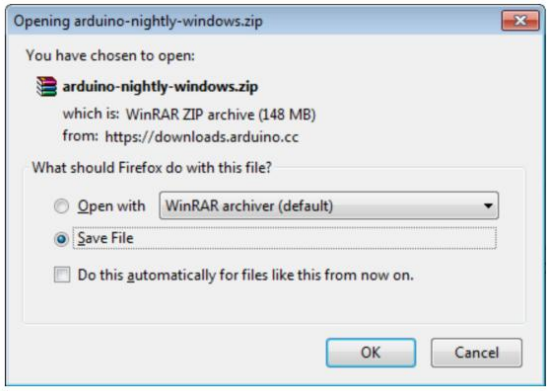
In this section, we will learn in easy steps, how to set up the Arduino IDE on our computer and prepare the board to receive the program via USB cable.

Step 1: First you must have your Arduino board (you can choose your favorite board) and a USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind you would connect to a USB printer as shown in the following image.

Figure 5.1: USB Cable

Step 2: Download Arduino IDE Software.

You can get different versions of Arduino IDE from the Download page on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



Step 3: Power up your board.

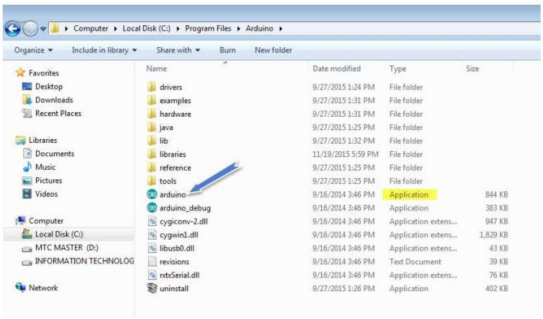
The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Dissimilar, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins

INTERNATIONAL JOURNAL OF TECHNO-ENGINEERING
IJTE
ISSN: 2057-5688

closest to the USB port. Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

Step 4: Launch Arduino IDE.

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double click the icon to start the IDE.

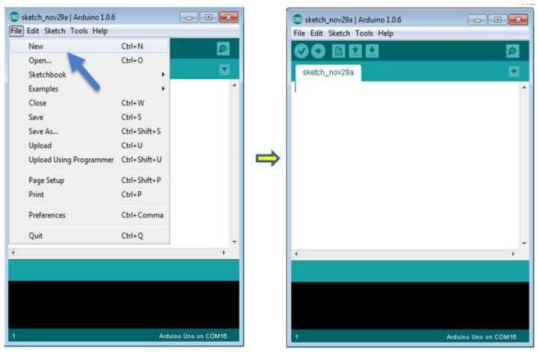


Step 5: Open your first project.

Once the software starts, you have two options:

- Create a new project.
- Open an existing project example.

To create a new project, select File --> New. To open



V RESULTS

Wi-Fi Security Control System controlled by Smart Devices Keeping access restricted to authorized employees alone is difficult in big apartment buildings, fraternities or for an owner who has multiple keys for each apartment, automobile or gate he owns. Security issues arise in the event of missing keys in addition to the expenses associated with key production, duplication, and distribution. Using current technology, this presentation will demonstrate a prototype for a novel device control system. This prototype is innovative because it combines new and existing technology to create something smarter and more efficient. In order to regulate any given system, we advise using smart devices. Any piece of machinery that replaces the

ISSN: 2057-5688

traditional key system with digital information, such a secret code, is said to have a digital devices control system. We propose a method in which the Central Control module is physically integrated into the Circuit board; this is necessary to avoid unnecessary complexity and to provide a more robust mechanism for the Circuit board as a whole. This system integrates technically with the home's LAN. This provides an additional layer of protection against unauthorized network access to the system. In addition, the suggested system has a significant advantage over current ones due to its simplicity and low infrastructure and planning needs.

VI Conclusion

The Smart devices control System will on the LEDS & FANS leading to a wide range of innovations in the world of Security systems wherever they may be. With its ease of installation and use, minimum complexity, wide applicability options, and strong feasibility, SLS guarantees a huge aspiring step forward into a better future control system. All of the above can't be considered authentic or even possible without considerately taking into account one of the most vital aspects to the innovation: security. Therefore, after examining the detailed evaluation and explanation of this phase, the project really tackles the security concerns to eliminate any worries which might cause a threat to the systems success and prosperity.

REFERENCES

[1] Kaustubh Dhondge Kaushik Ayinala Baek-Young Choi Sejun Song, "Infrared Optical Wireless Communication for Smart Door Locks Using Smart phones", 12th International Conference on Mobile Ad-Hoc and Sensor Networks, 2016.

[2] Abdallah Kassem and Sami El Murr, "A Smart control System using Wi-Fi Security", 3rd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA) 2016.

[3] 'Uno Revision 3", http://Arduino.cc/en/Main/arduinoBoardUno, 2016.

[4] M. Roland, "Software card emulation in NFC-enabled mobile phones: great advantage or security nightmare", in Fourth International Workshop on Security and Privacy in Spontaneous Interaction and Mobile Phone Use, 2012.

[5] Edoardo Persichetti,"Secure and Anonymous Hybrid Encryption from

ISSN: 2057-5688

Coding Theory", Springer-Verlag Berlin Heidelberg 2013.

[6] B. Rhodes, "Designing an access control system", https://ipvm.com/reports/designing- an-access-control system", 2015.

[7] "Access systems", https://WWW.security.honeywell.com/me/ documents/Access s Systems 2011.pdf, 2011.