## A MINI PROJECT REPORT ON

“LISTENING BUG USING IC 741”

## A Project report submitted in partial fulfilment of requirement for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**BY**

**NAME REGISTER NUMBER**

**A. Sai Likhitha (17091A04E9)**

**M. Sasikala (17091A04G4)**

**S. Asma (17091A0414 )**

**K. Siva Bharath (17091A04H5)**

**Under the guidance of**

**Dr. N. RAMANJANEYULU M.Tech.,Ph.D**

**Assoc. Professor, ECE Department**



### ESTD -1995

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY**

(AUTONOMOUS)

Accredited by NAAC of UGC, New Delhi with ‘A+’ Grade Approved by AICTE, Affiliated to J.N.T.U.A- Anantapuramu,

Nandyal-518501

## Year: 2020-2021

RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY

Autonomous

***Affiliated to J.N.T.U.A - Anantapuramu, Approved by A.I.C.T.E., New Delhi,***

***Accredited By N.B.A. & NAAC with ‘A+’ Grade -New Delhi,***

***NANDYAL –518501, KurnoolDist. A.P.***



**CERTIFICATE**

**This is to certify that the dissertation entitled “TRESPASSERS ALTERING**

**SYSTEM USING ESP32-CAM” that is being submitted by V.P.VEENA LAKSHMI (18091A04N6), E.SHAMAN (18091A04J3), A.RAJENDRA KUMAR (18091A04F1), G.GURU KAVYA SREE (18091A0458) under the guidance of Dr.N.RAMANJANEYULU** M.Tech., Ph.D., MIETE **for Mini Project of the award of B.Tech Degree in ELECTRONICS AND COMMUNICATION ENGINEERING in the RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY, Nandyal (Affiliated to J.N.T.U.A, Anantapuramu) is a record of bonafide work carried out by them under our guidance and supervision.**

**Head of the Department: Project Guide:**

### Dr. K.MALLIKARJUNADr.N.RAMANJANEYULU

M.E. Ph.D, MISTE, MIETE M.Tech., Ph.D., MIETE

**Professor and H.O.D Professor in ECE**

**Signature of the External Examiner**

**Date:**

CANDIDATE’S DECLARATION

We hereby declare that the work done in this Project titled “**TRESPASSERS ALERTING CONTROL SYSTEM USING ESP32-CAM**” submitted towards completion of mini project in IV Year I Semester of B. Tech (ECE) at the **Rajeev Gandhi Memorial College of Engineering & Technology**, Nandyal. It is an authentic record our original work done under the guidance of **Dr. N.RAMANJANEYULU, Assoc. Professor**, Dept. of ECE, RGMCET, Nandyal. We have not submitted the matter embodied in this project for the award of any other Degree in any other institutions.

**SUBMITTED BY:**

**V.P.Veena Lakshmi 18091A04N6**

**E.Shaman 18091A04J3**

**A.Rajendra Kumar 18091A04F1**

**G.Guru Kavya Sree 18091A0458**

**Place: Nandyal Date:**

**ACKNOWLEDGEMENT**

We manifest our heartier thankfulness pertaining to your contentment over our project guide **Dr. N.RAMANJANEYULU** **GARU,** Assistant Professor of Electronics and Communication Engineering department, with whose adroit concomitance the excellence has been exemplified in bringing out this project to work with artistry.

We express our gratitude to **Dr.K.MALLIKARJUNA****GARU,** Head of the Department of Electronics and communication Engineering department, all teaching and non-teaching staff of the Electronic and communication engineering department of Rajeev Gandhi memorial College of Engineering and Technology for providing continuous encouragement and cooperation at various steps of our project.

Involuntarily, we are perspicuous to divulge our sincere gratefulness to our Principal, **Dr. T. JAYACHANDRA PRASAD GARU,** who has been observed posing valiance in abundance towards our individuality to acknowledge our project work tangentially.

At the outset we thank our honorable **Chairman Dr. M. SANTHI RAMUDU GARU,** for providing us with exceptional faculty and moral support throughout the course.

Finally, we extend our sincere thanks to all the **Staff Members** of ECE Department who have co-operated and encouraged us in making our project successful.

Whatever one does, whatever one achieves, the first credit goes to the **Parents** be it not for their love and affection, nothing would have been responsible. We see in every good that happens to us their love and blessings.

**SUBMITTED BY:**

**V.P.Veena Lakshmi 18091A04N6**

**E.Shaman 18091A04J3**

**A.Rajendra Kumar 18091A04F1**

**G.Guru Kavya Sree 18091A0458**

The issue of security is very prominent in every organization especially such as bank. Therefore, we intend to aid insecurity of an organization by bringing Trespassers alerting control system.

CCTV Cameras are playing vital role in security system, but they are not helping in real time. Robbers are using these real time disadvantages of cameras as their advantages and escaping from the security system. For Example: If the robber breaks the cameras or cover their faces with masks or with any equipment, he can do his work without any fear or any interruptions.

Trespassers alerting control system alerts the administrator when the device detects any person. ESP32 used in the system is connected with camera module and windows sensors. This device output will trigger the alarm when it detects any person. When the trespassers enter the room either by windows or by doors, they can be detected by PIR sensors and camera. Either of the detection can trigger the alarm simultaneously the device will send the message to the respective administrator.

.

**CHAPTER -1**

**INTRODUCTION**

In this project we are going to make a motion sensor detector with photo capture using an ESP32-CAM. The ESP32-CAM is initially in deep sleep mode. When your IR sensor detects motion, it wakes up ESP32-CAM from the deep sleep mode with external wake up enabled. When motion is detected, the PIR motion sensor sends a signal to wake up the ESP-32. It goes back to deep sleep mode until a new signal from the PIR motion sensor is received. When PIR sensor wakes up ESP32 –CAM it captures a picture and send to the administrator along with buzzer sound.

The ESP-32 does the above job. It receives the signals from the sensors, and this signal is operated under the control of software called Arduino (IDE). Also in addition the pictures captured by OV2640 cam module will be sending to the administrator through a Telegram platform along with a message ‘MOTION DETECTED’ and also triggers the buzzer thus makes this system a very user friendly.

The most basic definition of any security system is found in its name. It is literally a means or method by which something is secured through a system of interworking components and devices. In this instance, we’re talking about security systems, which are networks of integrated electronic devices working together with a telegram to protect against burglars.

Whenever the administrator want to check the room at anytime the administrator can give a command photo through the telegram then the ESP32-CAM will wake up from the deep sleep mode and OV2640 camera module will capture the picture of the room instantly and sends it to the administrator’s telegram by this way we are able to check the room at any time and we can able to prevent burglars/robbers looting the organizations.

In this project the administrator can also have option like

1. Flash.
2. Buzzon.
3. Buzzoff.
4. Photo.

These are the commands which can be used by the administrator for different purpose.

**1.1 Working of different commands:**

**1) Flash:** This flash command is used to invert the state of the flash of the ESP32-CAM whenever the motion is detected and also when the ESP32-CAM wakes up.

**2) Buzzon:** Buzzon command is used to trigger the buzzer only when motion is detected by ESP32-CAM.

**3) Buzzoff:** Buzzoff command is used to turn off the buzzer it entirely operated by the administrator buzzer is in alarming state i.e.buzzon state until the administrator gives the buzzoff command.

**4) Photo:** Photo command is the command which is used by the administrator whenever the administrator want to check the room then the administrator uses this command. Hence, by using this command the administrator gets the picture captured by the ESP32-CAM module instantly without triggering the buzzer.

**CHAPTER-2**

**CIRCUIT AND COMPONENT DESCRIPTION**

**2.1 BLOCK DIAGRAM**

The below figure shows systematic arrangement of various components of the trespassers alerting controlling system using ESP32-CAM based on residents.

PIR SENSOR

AITHINKER ESP32-CAM

MESSAGE

BUZZER

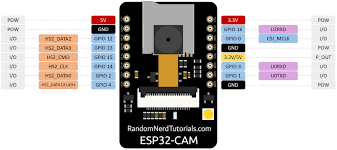
FTDI

**Fig.2.1: Block diagram of Trespassers alerting control system**

**2.2 HARDWARE COMPONENT DESCRIPTION**

**2.2.1 ESP32-CAM:**

The ESP32-CAM WIFI Module Bluetooth with OV2640 Camera Module 2MP For Face Recognization has a very competitive small size camera Module that can operate independently has a minimum system with Foot print of only 40\*27 mm, a deep sleep current of up to 6mA and is widely used in various IOT Applications. It is suitable for home smart devices, industrial wireless control and wireless monitoring.

****

**Fig.2.2: ESP32-CAM**

This Module adopts a DIP Package and can be directly inserted into the back plane to realize rapid production of products, providing customers with high reliability connection mode, which is convenient for various IOT hardware terminals. ESP means Extrasensory Perception also called as sixth sense includes claimed reception of information not gained through the recognized physical senses, but sensed with the mind. The term was adopted by Duke University.

**2.2.1.1 SPECIFICATIONS:**

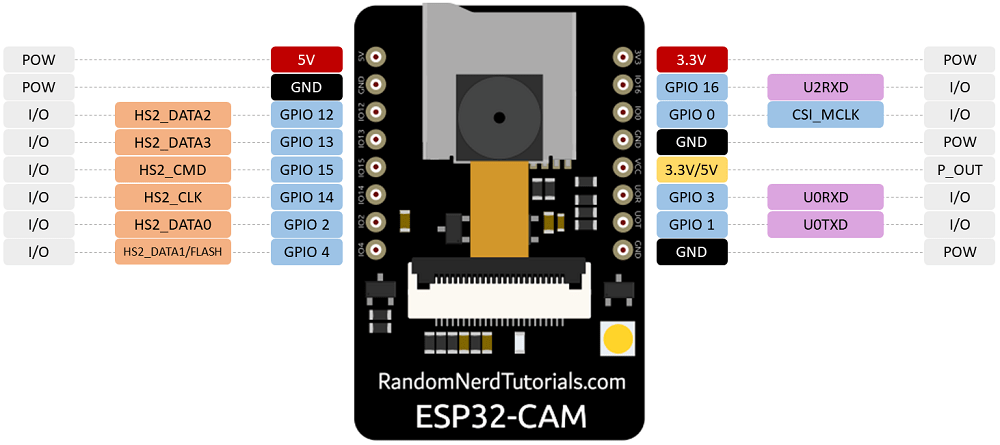
|  |  |
| --- | --- |
| * External Storage | Micro SD Card slot up to 4Gb |
| * Operating Voltage Range | 2.2-3.6 V |
| * Digital I/O Pins | 16 |
| * Package | DIP 16 |
| * SPI Flash | Default 32Mbit |
| * Image output format | JPEG (only supported by OV2640), BMP, GRAYSCALE |
| * Antenna form | onboard PCB antenna , gain 2dBi |
| * Support interface | UART, SPI, I2C, PWM,ADC,DAC |
| * RAM | internal520KB+external 4M PSRAM |
| * IO port | 9 |
| * Clock Speed | 160MHz |
| * Spectrum range | 2412 ~ 2484MHz |
| * MISC | Reset Button |
| * Antenna Type | IPEX and PCB |
| * Current Dissipation (mA) | 180 310 |

**Table.2.1: Specifications of ESP32-CAM**

**2.2.1.2 POWER SUPPLY:**

ESP32 CAM requires 310mA and 5V if you’re using the flash LED as well. If not 180mA should work fine. This part really depends on how you want to power it up. The board uses either batteries or AC current with a proper voltages and regulators in place. In the ESP32 all ground pins should be connected together.

**2.2.1.3: PIN DESCRIPTION:**



**Fig2.3: Pin description of ESP32-CAM**

1. **Power Pins:**

The ESP32-CAM comes with three GND pins (coloured in black colour) and two power pins (coloured with red colour) 3.3V and 5V.

Power can be supplied to ESP32-CAM through the 3.3V or 5V pins. It is advisable to **power the ESP32-CAM through the 5V pin to reduce errors.**

### Power output pin:

### There’s also the pin labeled on the silkscreen as VCC (colored with a yellow rectangle). You should not use that pin to power the ESP32-CAM. That is an output power pin. It can either output 5V or 3.3V. The ESP32-CAM outputs 3.3V whether it is powered with 5V or 3.3V.3) Serial Pins:

GPIO1 and GPIO3 are the serial pins (TX and RX, respectively). Because the ESP32-CAM doesn’t have a built-in programmer, you need to use these pins to communicate with the board and upload code.

**GPIO1** and **GPIO3** are used to connect other peripherals like outputs or sensors after uploading the code.

**GPIO 0** determines whether the ESP32 is in flashing mode or not. This GPIO is internally connected to a pull-up 10k Ohm resistor. When GPIO 0 is connected to GND, the ESP32 goes into flashing mode and you can upload code to the board.

**GPIO 13** PIR sensor out pin is connected to GPIO13 pin of ESP32 and vcc pin is connected to 3.3v.

Buzzer is connected to **GPIO15** pin. The ESP32-CAM has a very bright built-in LED that can work as a flash when taking photos. That LED is internally connected to **GPIO 4**.

All these GPIOs are RTC and support ADC: GPIOs 2, 4, 12, 13, 14, and 15.

## Flashlight (GPIO 4):

The ESP32-CAM has a very bright built-in LED that can work as a flash when taking photos. That LED is internally connected to GPIO 4.

1. **GPIO 33 – Built-in Red LED:**

Next to the RST button, there’s an on-board red LED. That LED is internally connected to GPIO 33. You can use this LED to indicate that something is happening.

For example, if the Wi-Fi is connected, the LED is red or vice-versa. That LED works with inverted logic, so you send a LOW signal to turn it on and a HIGH signal to turn it off.

1. **Camera Connections:**

The connections between the camera and the ESP32-CAM AI-Thinker as follows:

|  |  |
| --- | --- |
| **OV2640 CAMERA** | **ESP32** |
| D0 | GPIO 5 |
| XCLK | GPIO 0 |

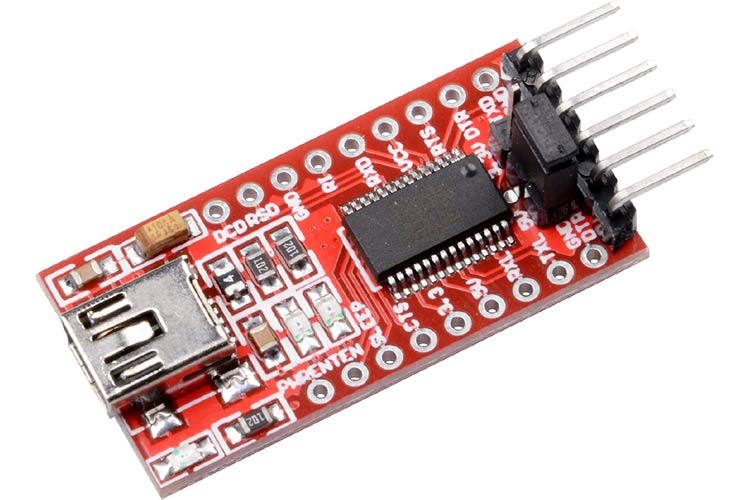
1. **Ground Pins:**

There are 3 ground pins on the ESP32-CAM, any of which can be used to ground your circuit.

**2.2.1.4 USB Over current Protection:**

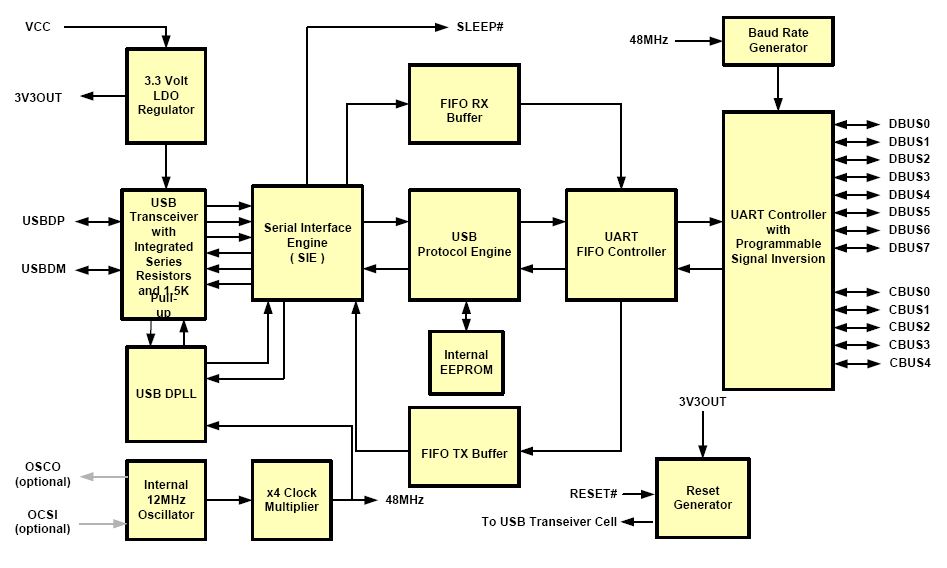
ESP32-CAM doesn’t come with a USB connector, so you need either a CP2102 or a FT23RL USB to TTL Serial Converter to connect it to your PC. When using Wi-Fi the ESP32 can use more current than is supplied via USB through these devices resulting in the module crashing and rebooting. FTDI programmer to upload code through the U0T and U0R pins (serial pins).If you use a FT232 with 5V and 3.3V, set the voltage jumper to 3.3V. You might be able to use the 5V from the side of the board to power the 5V pin on the camera.

**2.2.2 FTDI:**



**Fig 2.4: FT232RL USB TO TTL Converter.**

The **FT232RL** USB to TTL 3.3V/5V FTDI Serial Adapter Module is a popular module that is used to connect a TTL serial communicating device to a PC via a USB mini port. This converter module has options for different output voltage options which are set by the jumper on the board. FT232RL USB to TTL Serial Adapter Module can be used to upgrade any **legacy peripherals to USB.**

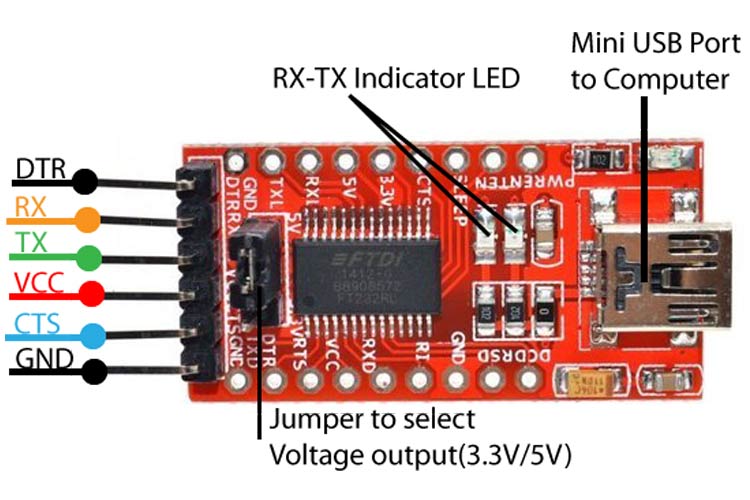


**Fig. Block diagram of FT232**

**2.2.2.1 Specifications:**

1. Operating Voltage: 5V/3.3V DC.
2. Max Current Draw: 5V - 500mA; 3.3V - 50mA.
3. Connector: Mini USB.
4. Fully integrated 1024 bit EEPROM storing device descriptors and CBUS I/O configuration.
5. Data transfer rates from 300 baud to 3 Mbaud (RS422, RS485, and RS232) at TTL levels.
6. 128 byte receive buffer and 256 bytes transmit buffer.
7. Transmit and receive LED drive signals.
8. Fully integrated clock generation with no external crystal required

**2.2.2.2 Pin description:**

****

**Fig 2.5: Pin description of FT232RL USB TO TTL Converter.**

The FT232RL USB to TTL Converter has 6 pins. The table below depicts all the pin types along with the function of each pin.

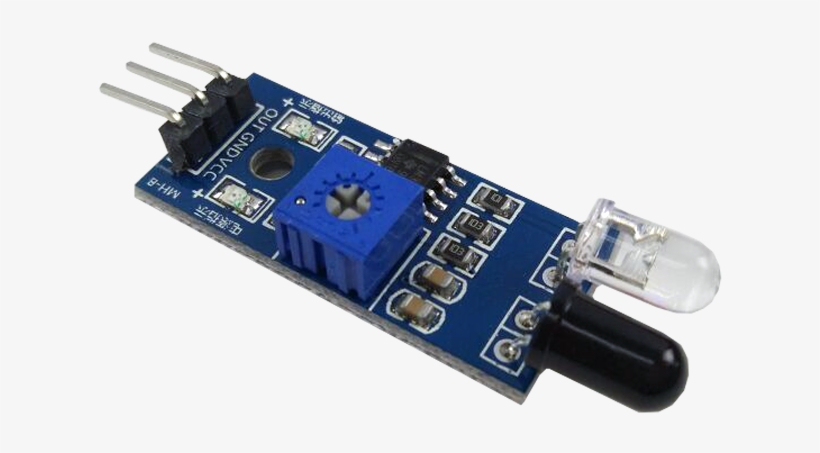
|  |  |
| --- | --- |
| **Pin Type** | **Description** |
| DTR | Data Terminal Ready(Output used for flow control) |
| RX | Serial Data Receiver |
| TX | Serial Data Transmit |
| VCC | Power Input |
| CTS | Clear to send(Input used for flow control) |
| GND | Ground |

**Table.2.2: Specifications of FT232RL USB TO TTL Converter**

**2.2.2.3: Applications of FT232RL USB TO TTL Converter**

Below are the applications of FT232RL USB TO TTL Converter:

1. Programming AVR, ARM-based microcontrollers/microprocessors.
2. USB to RS232/RS422/RS485 Converters
3. USB Smart Card Readers
4. For serial communication with GPS devices
5. Serial terminal on devices like Raspberry Pi

**2.2.3 IR SENSOR:** 

**Fig.2.6: IR Sensor**

An infrared sensor is an electronic device that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. These types of sensors measures only infrared radiation, rather than emitting it that is called as a passive IR sensor. Usually in the infrared spectrum, all the objects radiate some form of thermal radiations. These types of radiations are invisible to our eyes that can be detected by an infrared sensor. The emitter is simply an IR LED and the detector is simply an IR photodiode which is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode The resistances and these output voltages, change in proportion to the magnitude of the IR light received.

**2.2.3.1 Pin Configuration:**

|  |  |
| --- | --- |
| **Pin Name** | **Description** |
| VCC | Power Supply Input |
| GND | Power Supply Ground |
| OUT | Active High Output |

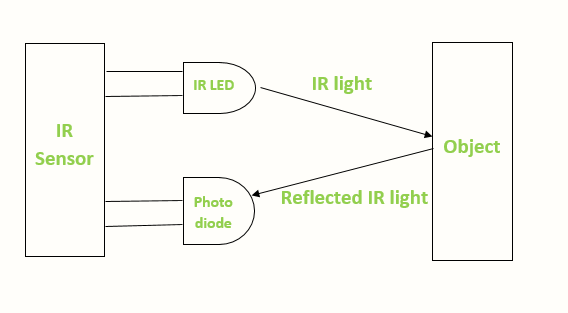
**Table.2.2: Pin Configuration of IR Sensor**

**2.2.3.2 Working:**

An IR sensor consists of two parts, the emitter circuit and the receiver circuit. This is collectively known as a photo-coupler or an opt coupler.

The emitter is an IR LED and the detector is an IR photodiode. The IR photodiode is sensitive to the IR light emitted by an IR LED. The photo-diode’s resistance and output voltage change in proportion to the IR light received. This is the underlying working principle of the IR sensor.

The type of incidence can be direct incident or indirect incidence. In direct incidence, the IR LED is placed in front of a photodiode with no obstacle in between. In indirect incidence, both the diodes are placed side by side with an opaque object in front of the sensor. The light from the IR LED hits the opaque surface and reflects back to the photodiode.



**Fig.2.7: Working of IR Sensor**

**2.2.3.3 Application:**

1. Obstacle Detection.
2. Industrial safety devices.
3. Wheel encoder.
   * + 1. **Features:**
4. The operating voltage is 5VDC.
5. I/O pins – 3.3V & 5V.
6. Mounting hole.
7. The range is up to 20 centimeters.
8. The supply current is 20mA.
9. The range of sensing is adjustable.
10. Fixed ambient light sensor.
    * 1. **Buzzer:**

An audio signaling device like a beeper or buzzer may be electromechanical or [piezoelectric](https://www.elprocus.com/what-is-a-piezoelectric-material-working/) or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.



**2.2.4.1 Types of Buzzer:** Buzzers come in a variety of construction, size, and specifications. Different types and sizes of buzzers are used for different applications. Based on construction, there are the following kinds of buzzers:

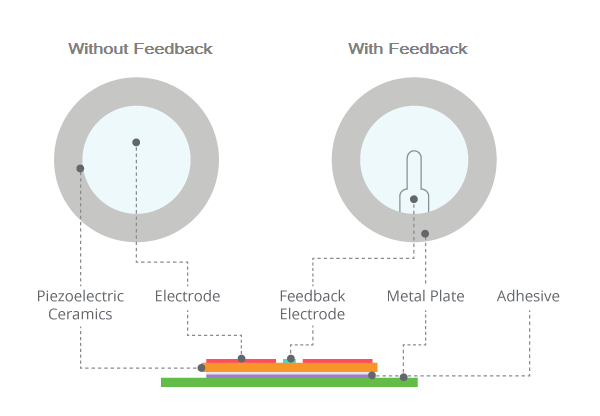
1. Piezoelectric buzzers
2. Magnetic buzzers
3. Electromagnetic buzzers
4. Mechanical buzzers
5. Electromechanical buzzers

Piezoelectric and magnetic buzzers are most commonly used in electronic applications. The buzzers are designed to be used as a transducer or indicator in any circuit.

1. **Piezoelectric buzzers:**

The piezo buzzers have a wide operating voltage that ranges from 3V to 250V. Most of the piezo buzzers used in electronic circuits have an operating voltage between 3V and 12V. These buzzers have high sound pressure levels. They have very low current consumption. Higher is the frequency/tone of the piezo buzzer; the lower is its current consumption. The buzzers used in electronic applications have current consumption as low as 30 mA. The piezo buzzers have a large footprint and are preferred to be used in cost-sensitive electronic applications.

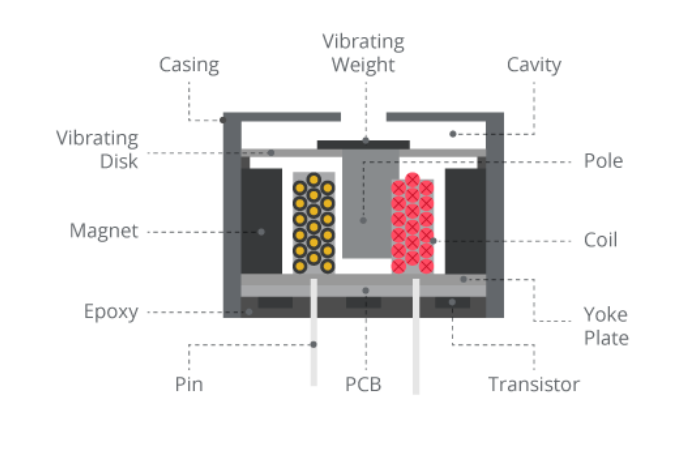
Many piezo buzzers have a feedback line. In such buzzers, the piezoelectric element is divided into two electrically isolated parts. When the main piezo element is actuated, it squeezes the feedback component, producing a feedback voltage. The feedback signal is usually applied to a transistor/OP-AMP circuit, which blocks or amplifies the current supply to the piezoelectric element.



**Fig.2.9:** **Construction of piezoelectric buzzer**

1. **Magnetic buzzers:**

In a magnetic buzzer, there is a ferromagnetic disc that is attached to a pole. There are magnets around the pole which maintains the disc in a rest position. There is a coil below the ferromagnetic disc that acts as an electromagnet. When current is supplied to the coil, the disc is attracted to the coil. When there is no current in the coil, the disc returns to its rest position. A weight above it controls the vibrations of the disc. When an oscillating signal is applied to the coil, the electromagnetic field generated from it also fluctuates, causing vibrations in the ferromagnetic disk. This way, the sound of the frequency same as the frequency of the applied oscillating signal is produced by the magnetic buzzer. Magnetic buzzers come in both transducer as well as indicator configurations.

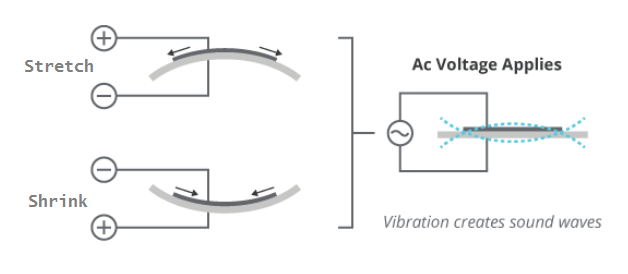
**Fig.2.10: Construction of magnetic buzzer**

The magnetic buzzers have a narrow operating voltage that ranges from 1V to 16V. These buzzers produce lower-rated frequencies have low sound pressure levels. These have a slightly greater current consumption in comparison to piezo buzzers. Their current consumption can be as high as 100 mA. These buzzers have a small footprint and are generally used in high-end consumer applications.

**2.2.4.2 Working:**

The working principle of a buzzer depends on the theory that, once the voltage is given across a piezoelectric material, then a pressure difference is produced. A piezo type includes piezo crystals among two conductors.

Once a potential disparity is given across these crystals, then they thrust one conductor & drag the additional conductor through their internal property. So this continuous action will produce a sharp sound signal. A piezoelectric buzzer operates on the principle of the piezoelectric effect. The main component of a piezoelectric buzzer is a piezoelectric element. The element is composed of a piezoelectric ceramic and a metal plate. Both the piezoelectric disc and metal plate are held together with an adhesive. The piezoceramic disc has electrodes attached to it. The piezoelectric disc expands and contracts diametrically when an alternating current is applied to it. This produces vibrations in the piezoelectric element and generating the sound of a particular frequency or range of frequencies.

**[](https://b3van8qm1o7ou9d3b48qdhsg-wpengine.netdna-ssl.com/wp-content/uploads/2020/09/TCH07-01.png)**

**Fig.2.11: Working principle of Buzzer.**

**2.2.4.3 Applications:**

1. Alarms / warning devices / automobile alarms.
2. Pest deterrents.
3. Computer devices.
4. Telephones.
5. Toys.
   * + 1. **Features:**

1) Rated Voltage: 6V DC.

2) Operating Voltage: 4-8V DC.

3) Rated current: <30mA.

4) Sound Type: Continuous Beep.

5) Resonant Frequency: ~2300 Hz.

6) Small and neat sealed package.

7) Breadboard and Perf board friendly.

**2.2.5 Jumper Wires:**

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering.Jumper wires typically come in three versions:

* 1. Male-to-male.
  2. Male-to-female.
  3. Female-to-female.



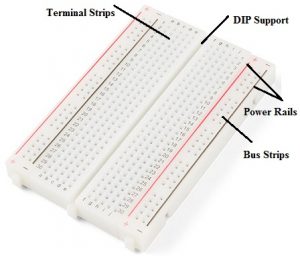
**Fig.2.12: Real View of Jumper Wires**

The different between each is in the end point of the wire. Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into. Male-to-male jumper wires are the most common and are used to connect two ports on a breadboard, a female-to-female wire is what you’ll need.

Female to female jumper wires for connecting standard **2.54 Mm header** pin and also relimate connector of different type of development board. Pack consist 40 pcs of jumper wire in 10 different colors.

**2.2.6 Breadboard:**

A breadboard is used to build and test circuits quickly before finalizing any circuit design. The breadboard has many holes into which circuit components like ICs and resistors can be inserted.



**Fig.2.13: Real view of breadboard**

The breadboard contains spring clip contacts typically arranged in matrices with certain blocks of clips already wired together. The components and jump wires (assorted wire lengths with pins at both ends) are plugged into the clips to create the circuit patterns. The boards also typically include metal strips along the side that are used for common power rails and signal buses.

Breadboards are designed to work with through-hole electroniccomponents. These components have long metal leads that are designed to be inserted through holes in a printed circuit board (PCB) that are plated with a thin copper coating, which allows the components' leads to be soldered to the board.

**CHAPTER-3**

**SOFTWARE DESCRIPTION**

**3.1 MAJOR SOFTWARE REQUIRED:**

For the software development process, the group followed the following procedures:

1. The initial values are set and the I/O ports are initialized.
2. Each sensor reading is set to the driest state to obtain uniform for

Calibration.

1. After calibration, the program reads the value of the sensors one at a

Time, populating each array index.

1. On each array index, the program checks the value returned by the

Sensor and analyses the state of soil.

1. If the soil is dry, the driver circuit activates and the solenoid

Valve opens. If the soil is not dry, the program proceeds to check the

Net array index, and;

1. The program also checks if the end of array index is reached. The

Program runs on a loop –checking the array indices one at a time.

ESP32 contains one or two low-power Xtensa® 32-bit LX6 microprocessor with the following features:

1) 7-stage pipeline to support the clock frequency of up to 240 MHz (160 MHz for ESP32-S0WD)

2)16/24-bit Instruction Set provides high code-density

3) Support for Floating Point Unit

4) Support for DSP instructions, such as a 32-bit multiplier, a 32-bit divider, and a 40-bit MAC

5) Support for 32 interrupt vectors from about 70 interrupt sources

6) The single-/dual-CPU interfaces include:

• Xtensa RAM/ROM Interface for instructions and data

• Xtensa Local Memory Interface for fast peripheral register access

• External and internal interrupt sources

7) JTAG for debugging

**3.2 ARDUINO SOFTWARE (IDE):**

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any ESP32 board. For latest software: https://www.arduino.cc/en/Main/Software.

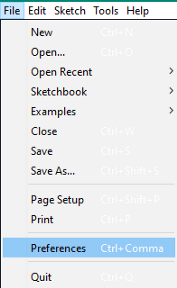
* Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students with or without a background in electronics and programming.
* Arduino is an open-source prototyping platform based on easy-to use hardware and software.
* ESP board can read the inputs- motion detected by a sensor, a figure on button or a message to it from FTDI – and turn it into an output – activating the buzzer, publishing the image online and many more.
* 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 (based on Writing), and the Arduino Software (IDE), based on Processing.
* **Cross-platform:** The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller system is 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.
* **Open source and extensible hardware:** The plans of the ESP32 board are published under a Creative Common license, so experienced circuit designers can make their own version of the module, extending it and improving it.
* **Open source and extensible software:** The Arduino software is published as open source tool and the language can be expanded through C++ libraries

**3.3 Steps for using Arduino IDE:**

**Step 1: Download**[**Arduino IDE**](https://downloads.arduino.cc/arduino-1.8.15-windows.exe)

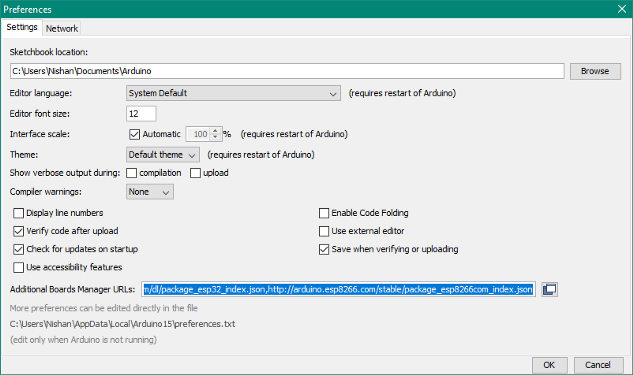
In this tutorial, we assume that using an Arduino Uno also need a standard USB cable (A plug to B plug the kind you would connect to a USB printer, for example

**Step 2:** **in your Arduino IDE, go to File > Preferences**

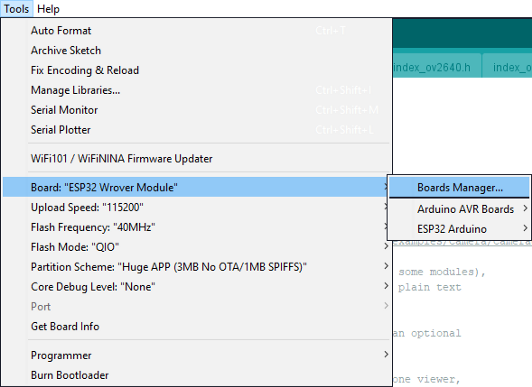
****

**Step 3: Connection to board**

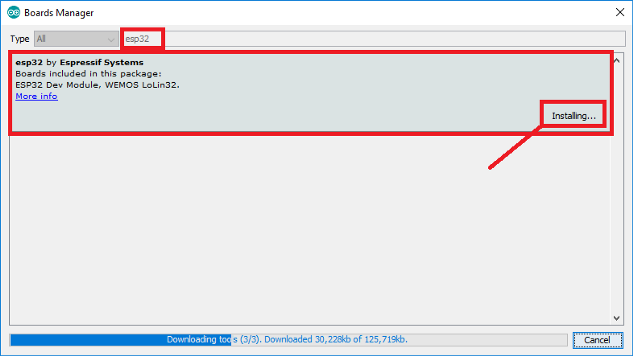
Enter https://dl.espressif.com/dl/package\_esp32\_index.json,http://arduino.esp8266.com/stable/package\_esp8266com\_index.json into the “Additional Board Manager URLs” field as shown in the figure below. Then, click the “OK” button.



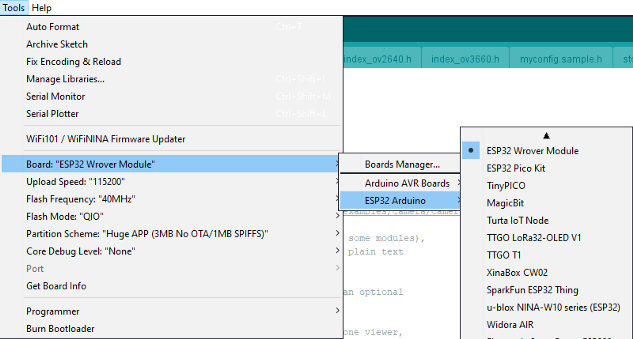
**Step 4:**Open the Boards Manager. Go to **Tools** > **Board** > **Boards Manager…**



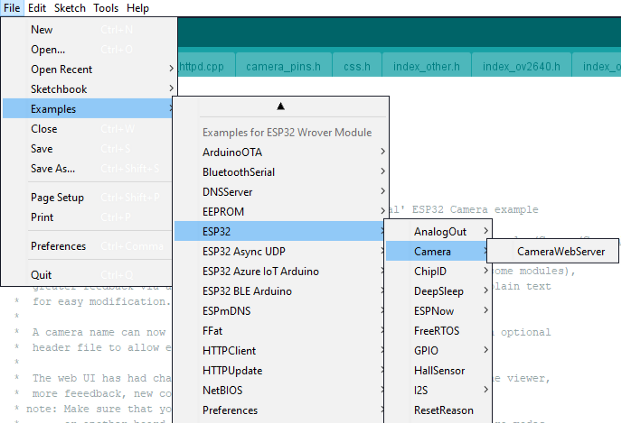
**Step 5: Search for ESP32 and press install button for the “ESP32 by Express if Systems“**



**Step 6:**After that you can set **ESP32 Wrover Module** as your board. Also set upload speed as **115200**, Flash Frequency to 40MHz, Flash Mode to QIO, Partition Scheme to **Huge App.**

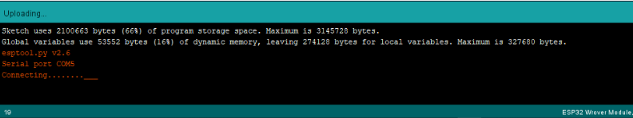
****

**Step 7:**After that you should be able to access ESP32 examples, open the Camera Web Server example as follows.

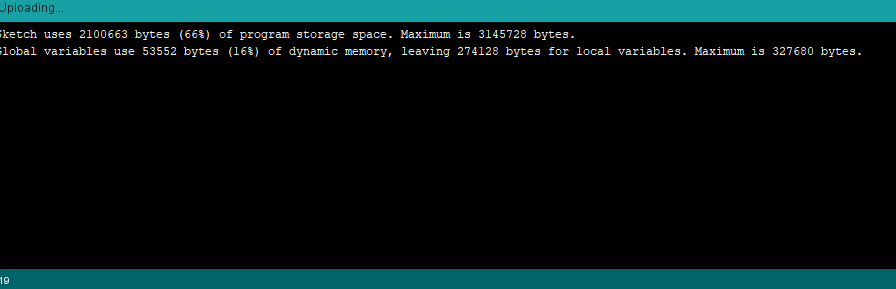


**Step-8:** **Select camera mode**

Uncomment #defineCAMERA\_MODEL\_AI\_THINKER**.** Now you can compiler and upload your code. When you are uploading your code, some information will pop up in the output.

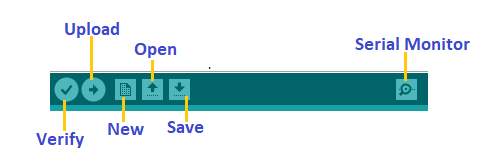


When it says it’s connecting.., a good idea to press the reset button on the ESP32 CAM module if it takes a long time. Usually you don’t have to do this but in some cases I had to do it. So keep that in mind as well. Reset button is not so hard to locate because it is the only button on the board.



**Step-9: Select the Serial port and upload code**

Open Serial Monitor using Tools > Serial Monitor**,**Press Reset button once.



**Fig: Uploading the program**

**Verify**- Used to check if there is any compilation error.

**Upload**- Used to upload a program to the ESP32 board.

**New**-Shortcut used to create new sketch.

**Open**-Used directly to open one of the example sketch.

**Save**- Used to save your sketch.

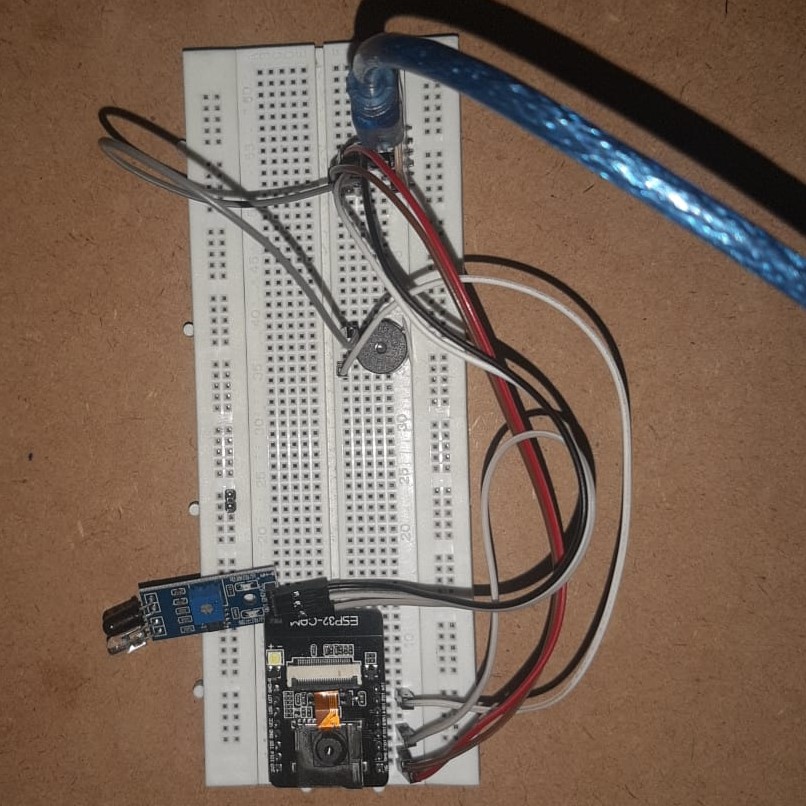
**Serial Monitor**- Serial monitor used to receive serial data from the board and send serial data to board.

To upload the Arduino compile the code first, if there is no errors click the upload button. Wait a few seconds; if the upload successful, the message “Done Uploading” will appear in the status bar and the LED on the board will blink for one second.

**CHAPTER-4**

**PROJECT DESCRIPTION**

**4.1 HARDWARE CONNECTION DIAGRAM:**



**Fig.4.1: Overview of Trespassers alerting control system using ESP32**

**4.2 OPERATION OF CIRCUIT DIAGRAM:**

1. To upload code to the ESP32-CAM board, connect it to your computer using an FTDI programmer. Many FTDI programmers have a jumper that allows you to select 3.3V or 5V. Make sure the jumper is in the right place to select 5V. RX and TX pins should be connected.
2. PIR sensor out pin is connected to GPIO13 pin of ESP32 and vcc pin is connected to 3.3v.
3. Buzzer is connected to GPIO15 pin.
4. FTDI vcc is connected to 5v for the power supply.

**4.3 ADVANTAGES:**

1. PIR sensor detects motion reliably in indoors as well as in dark or day.
2. ESP32 has faster WI-FI, more GPIOs adds an extra CPU code.
3. FTDI used to detect low power.
4. Buzzer size is small, consumes low energy and sound pressure is high.

**4.4 ESP32-CAM PROGRAMMING CODE:**

#include <WiFi.h>

#include <WiFiClientSecure.h>

#include "soc/soc.h"

#include "soc/rtc\_cntl\_reg.h"

#include "esp\_camera.h"

#include <UniversalTelegramBot.h>

#include <ArduinoJson.h>

#include <Wire.h>

// Replace with your network credentials

const char\* ssid = "WiFiname";

const char\* password = "WiFiPassword";

// Use @myidbot to find out the chat ID of an individual or a group

// Also note that you need to click "start" on a bot before it can

// message you

String chatId = "-1001772078058";

// Initialize Telegram BOT

String BOTtoken ="2118707404:AAG9fosRsFlG3A3qik75k3iqx3DIzxW6kBk";

bool sendPhoto = false;

WiFiClientSecure clientTCP;

UniversalTelegramBot bot (BOTtoken, clientTCP);

//CAMERA\_MODEL\_AI\_THINKER

#define PWDN\_GPIO\_NUM 32

#define RESET\_GPIO\_NUM -1

#define XCLK\_GPIO\_NUM 0

#define SIOD\_GPIO\_NUM 26

#define SIOC\_GPIO\_NUM 27

#define Y9\_GPIO\_NUM 35

#define Y8\_GPIO\_NUM 34

#define Y7\_GPIO\_NUM 39

#define Y6\_GPIO\_NUM 36

#define Y5\_GPIO\_NUM 21

#define Y4\_GPIO\_NUM 19

#define Y3\_GPIO\_NUM 18

#define Y2\_GPIO\_NUM 5

#define VSYNC\_GPIO\_NUM 25

#define HREF\_GPIO\_NUM 23

#define PCLK\_GPIO\_NUM 22

#define FLASH\_LED\_PIN 4

bool flashState = LOW;

#define buzzer 2

// Motion Sensor

bool motionDetected = false;

int botRequestDelay = 1000; // mean time between scan messages

long lastTimeBotRan; // last time messages' scan has been done

void handleNewMessages (int numNewMessages);

String sendPhotoTelegram ();

// Indicates when motion is detected

static void IRAM\_ATTR detectsMovement (void \* arg){

Serial.println("MOTION DETECTED!!!");

motionDetected = true;

}

void setup(){

WRITE\_PERI\_REG (RTC\_CNTL\_BROWN\_OUT\_REG, 0);

Serial.begin(115200);

pinMode(FLASH\_LED\_PIN, OUTPUT);

pinMode (buzzer, OUTPUT);

digitalWrite(buzzer, LOW);

digitalWrite(FLASH\_LED\_PIN, flashState);

WiFi.mode(WIFI\_STA);

Serial.println();

Serial.print ("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

clientTCP.setCACert(TELEGRAM\_CERTIFICATE\_ROOT); // Add root certificate for api.telegram.org

while (WiFi.status () != WL\_CONNECTED) {

Serial.print (".");

delay (500);

}

Serial.println ();

Serial.print ("ESP32-CAM IP Address: ");

Serial.println (WiFi.localIP ());

camera\_config\_t config;

config.ledc\_channel = LEDC\_CHANNEL\_0;

config.ledc\_timer = LEDC\_TIMER\_0;

config.pin\_d0 = Y2\_GPIO\_NUM;

config.pin\_d1 = Y3\_GPIO\_NUM;

config.pin\_d2 = Y4\_GPIO\_NUM;

config.pin\_d3 = Y5\_GPIO\_NUM;

config.pin\_d4 = Y6\_GPIO\_NUM;

config.pin\_d5 = Y7\_GPIO\_NUM;

config.pin\_d6 = Y8\_GPIO\_NUM;

config.pin\_d7 = Y9\_GPIO\_NUM;

config.pin\_xclk = XCLK\_GPIO\_NUM;

config.pin\_pclk = PCLK\_GPIO\_NUM;

config.pin\_vsync = VSYNC\_GPIO\_NUM;

config.pin\_href = HREF\_GPIO\_NUM;

config.pin\_sscb\_sda = SIOD\_GPIO\_NUM;

config.pin\_sscb\_scl = SIOC\_GPIO\_NUM;

config.pin\_pwdn = PWDN\_GPIO\_NUM;

config.pin\_reset = RESET\_GPIO\_NUM;

config.xclk\_freq\_hz = 20000000;

config.pixel\_format = PIXFORMAT\_JPEG;

//init with high specs to pre-allocate larger buffers

if (psramFound()){

config.frame\_size = FRAMESIZE\_UXGA;

config.jpeg\_quality = 10; //0-63 lower number means higher quality

config.fb\_count = 2;

} else {

config.frame\_size = FRAMESIZE\_SVGA;

config.jpeg\_quality = 12; //0-63 lower number means higher quality

config.fb\_count = 1;

}

// camera init

esp\_err\_t err = esp\_camera\_init(&config);

if (err! = ESP\_OK) {

Serial.printf ("Camera init failed with error 0x%x", err);

delay(1000);

ESP.restart();

}

// Drop down frame size for higher initial frame rate

sensor\_t \* s = esp\_camera\_sensor\_get();

s->set\_framesize(s, FRAMESIZE\_CIF); //UXGA|SXGA|XGA|SVGA|VGA|CIF|QVGA|HQVGA|QQVGA

// PIR Motion Sensor mode INPUT\_PULLUP

//err = gpio\_install\_isr\_service (0);

err = gpio\_isr\_handler\_add (GPIO\_NUM\_13, &detectsMovement, (void \*) 13);

if (err! = ESP\_OK) {

Serial.printf ("handler add failed with error 0x%x \r\n", err);

}

err = gpio\_set\_intr\_type (GPIO\_NUM\_13, GPIO\_INTR\_POSEDGE);

if (err! = ESP\_OK){

Serial.printf ("set intr type failed with error 0x%x \r\n", err);

}

}

void loop () {

if (sendPhoto) {

flashState =! flashState;

digitalWrite (FLASH\_LED\_PIN, flashState);

Serial.println ("Preparing photo");

sendPhotoTelegram ();

sendPhoto = false;

flashState = !flashState;

digitalWrite (FLASH\_LED\_PIN, flashState);

}

if (motionDetected) {

bot.sendMessage (chatId, "Motion detected!!", "");

Serial.println ("Motion Detected");

flashState = !flashState;

digitalWrite (FLASH\_LED\_PIN, flashState);

sendPhotoTelegram ();

digitalWrite (buzzer, HIGH);

motionDetected = false;

flashState = !flashState;

digitalWrite (FLASH\_LED\_PIN, flashState);

}

if (millis () > lastTimeBotRan + botRequestDelay) {

int numNewMessages = bot.getUpdates (bot.last\_message\_received + 1);

while (numNewMessages){

Serial.println ("got response");

handleNewMessages (numNewMessages);

numNewMessages = bot.getUpdates(bot.last\_message\_received + 1);

}

lastTimeBotRan = millis ();

}

}

String sendPhotoTelegram () {

const char\* myDomain = "api.telegram.org";

String getAll = "";

String getBody = "";

camera\_fb\_t \* fb = NULL;

fb = esp\_camera\_fb\_get ();

if(!fb) {

Serial.println("Camera capture failed");

delay (1000);

ESP.restart ();

return "Camera capture failed";

}

Serial.println ("Connect to " + String (myDomain));

if (clientTCP.connect (myDomain, 443)) {

Serial.println ("Connection successful");

String head = "--RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"chat\_id\"; \r\n\r\n" + chatId + "\r\n--RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"photo\"; filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";

String tail = "\r\n--RandomNerdTutorials--\r\n";

uint16\_t imageLen = fb->len;

uint16\_t extraLen = head.length () + tail.length ();

uint16\_t totalLen = imageLen + extraLen;

clientTCP.println ("POST /bot"+BOTtoken+"/sendPhoto HTTP/1.1");

clientTCP.println ("Host: " + String (myDomain));

clientTCP.println ("Content-Length: " + String (totalLen));

clientTCP.println ("Content-Type: multipart/form-data; boundary=RandomNerdTutorials");

clientTCP.println ();

clientTCP.print (head);

uint8\_t \*fbBuf = fb->buf;

size\_t fbLen = fb->len;

for (size\_t n=0;n<fbLen;n=n+1024) {

if (n+1024<fbLen) {

clientTCP.write (fbBuf, 1024);

fbBuf += 1024;

}

else if (fbLen%1024>0) {

size\_t remainder = fbLen%1024;

clientTCP.write (fbBuf, remainder);

}

}

clientTCP.print (tail);

esp\_camera\_fb\_return (fb);

int waitTime = 10000; // timeout 10 seconds

long startTimer = millis ();

boolean state = false;

while ((startTimer + waitTime) > millis ()) {

Serial.print (".");

delay (100);

while (clientTCP.available()) {

char c = clientTCP.read();

if (state==true) getBody += String(c);

if (c == '\n') {

if (getAll.length()==0) state=true;

getAll = "";

}

else if (c != '\r')

getAll += String(c);

startTimer = millis();

}

if (getBody.length()>0) break;

}

clientTCP.stop();

Serial.println(getBody);

}

else {

getBody="Connected to api.telegram.org failed.";

Serial.println("Connected to api.telegram.org failed.");

}

return getBody;

}

void handleNewMessages(int numNewMessages){

Serial.print("Handle New Messages: ");

Serial.println(numNewMessages);

for (int i = 0; i < numNewMessages; i++){

// Chat id of the requester

String chat\_id = String(bot.messages[i].chat\_id);

if (chat\_id != chatId){

bot.sendMessage(chat\_id, "Unauthorized user", "");

continue;

}

// Print the received message

String text = bot.messages[i].text;

Serial.println(text);

String fromName = bot.messages[i].from\_name;

if (text == "/flash") {

flashState = !flashState;

digitalWrite(FLASH\_LED\_PIN, flashState);

}

if (text == "/buzzon") {

digitalWrite (buzzer, HIGH);

}

if (text == "/buzzoff") {

digitalWrite(buzzer, LOW);

}

if (text == "/photo") {

sendPhoto = true;

Serial.println("New photo request");

}

if (text == "/start"){

String welcome = "Welcome to Trespassers alerting system.\n";

welcome += "/photo : takes a new photo\n";

welcome += "/flash : toggle flash LED\n";

welcome += "/buzzon : ON the buzzer\n";

welcome += "/buzzoff : OFF the buzzer\n";

welcome += "You'll receive a photo whenever motion is detected.\n";

bot.sendMessage (chatId, welcome, "Markdown");

}

}

}

**CHAPTER-5**

**CONCLUSION AND FUTURE SCOPE**

**5.1 CONCLUSION:**

A novel architecture for an economic Trespasser’s alerting control system is proposed and implemented in this paper. It gives basic idea of how to trespasser’s alerting system using ESP32-CAM and Arduino (IDE). The cost of this technology is very economical. This project uses low cost of the shelf components, and is based on Arduino platform which is FOSS (Free Open Source Software). So the overall implementation cost is very cheap and is affordable by a common person. This low cost system is designed to improve the living standard and complexity of identifying robbers. It provides accurate data and eliminating the error where possible.

The ESP32 contains one or two low-power Xtensa 32-bit LX6 microprocessor. Hence this will helps the system to work accurately and makes system faster. The OV2640 Camera Module is a precise small size, high sensitive and low voltage CMOS image sensor module for capturing and processing the image. It supports the image sizes VGA, CIF and any size scaling down from CIF to 40\*30. Because of all these advantages of OV2640 Camera Module will helps to capture a clear quality picture with CIF frame size of 352\*288 pixels which is standard resolution of CIF.

This system can also be easily operated by the users who didn’t have an idea of the Technology too. This indicates that the system is very friendly to the users. It has been successfully demonstrated that, this will serve as a device for providing security. It provides the user with efficient and reliable security system for organizations, malls and houses etc.

**5.2 FUTURE SCOPE:**

For future work, some recommendation can be made like, sending on alerting message and image captured by OV2640 camera module to many other social media websites like what’s app. We can also use other types of sensors other than IR sensor like PIR sensor, Microwave sensor, Tomographic sensor to cover the larger area of the room because this sensor are having wider angle and long distance than compare to IR sensor with appropriate interface with the ESP32-CAM and also by supplying suitable power supply. The whole system can be fabricated as economic commercial hardware package.

**REFERENCES**

* HTTPS://EN.WIKIPEDIA.ORG/WIKI/ARDUINO
* HTTPS://LEARN.ADAFRUIT.COM
* HTTPS://GITHUB.COM
* HTTPS://ELECTRONICSHUB.COM
* HTTPS://nishanc.medium.com/what-i-learned-about-esp32-cam-and-everything-you-need-to-know-12a9e520a0da
* HTTPS://WWW.ELECTRONICSFORU.COM
* <HTTPS://ELECTROINCS-PROJECT-HUB.COM>
* http://www.ftdichip.com/Documents/InstallGuides.htm