Fatima Jinnah Women University

Opening Portals of Excellence Through Higher Education

PROJECT PROGRESS

EMBEDDED SYSTEMS

SUBMITTED TO

SIR HAROON WASIM

SUBMITTED BY

LAIBA SOHAIL (2021-BSE-016)
TANZEELA ASGHAR (2021-BSE-032)
NEHA AMJAD (2021-BSE-024)

SEMESTER

VI-A

DATE

JUNE 10, 2024

“Report Writing”

Motion detecting system:

1. Introduction:

In the context of this project, we have the “Motion Detection System” that seeks to develop a surveillance system with the ability to detect motion through the ESP32-CAM microcontroller module, and the PIR (Passive Infrared) sensor. It is for this reason that the system can be applied in various fields that include home security, wildlife surveillance, and even in business processes

2. Components Used:

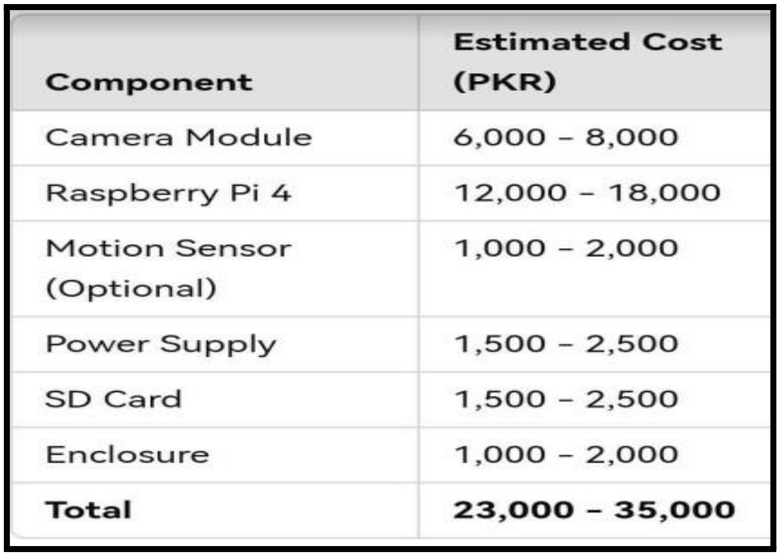
- **ESP32-CAM:** More specifically, it operates as both the Key Microcontroller and the camera unit.
- **PIR Sensor:** Detects motion by sensing changes in thermography also known as infrared radiation.
- **FTDI Programmer**: This is the board that is used for interfacing with the ESP32-CAM where the programming is done.
- **MicroSD Card:** Serves as a storage platform for images or videos captured with the use of ESP32-CAM.
- **5V Power Bank**:Supplies power to the ESP32-CAM.
- **OV2640 Camera**: Used in the ESP32-CAM module to take pictures for distribution Referenced in the code for the collection of images.
- **Jumper Wires:** Used in joining or interconnecting interfaces of various parts of a system.
- **Breadboard:** Offers an area to test and categorize links.

Images:

3. System Architecture:

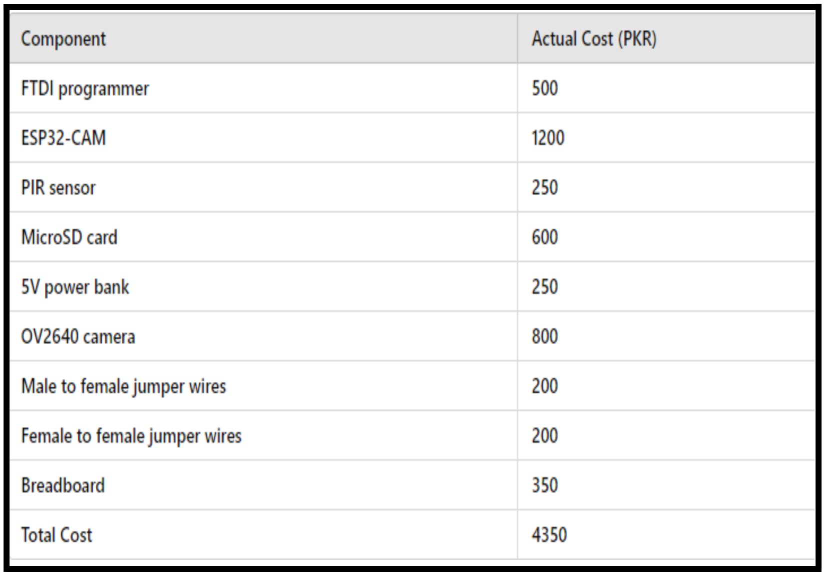
The main trunk of this system is the ESP32-CAM microcontroller. It communicates with the PIR sensor, which is used for determining motion occurrences. In this case, once motion is identified, the ESP32-CAM employs the OV2640 camera to take images. All images are taken and stored in the MicroSD card for later reactivation and evaluation.

Estimated Cost:



Component	Estimated Cost (PKR)
Camera Module	6,000 – 8,000
Raspberry Pi 4	12,000 – 18,000
Motion Sensor (Optional)	1,000 – 2,000
Power Supply	1,500 – 2,500
SD Card	1,500 – 2,500
Enclosure	1,000 – 2,000
Total	**23,000 – 35,000**

Actual Cost:



Component	Actual Cost (PKR)
FTDI programmer	500
ESP32-CAM	1200
PIR sensor	250
MicroSD card	600
5V power bank	250
OV2640 camera	800
Male to female jumper wires	200
Female to female jumper wires	200
Breadboard	350
Total Cost	4350

Hardware connections:

1. ESP32-CAM to FTDI Programmer:

Interface the FTDI programmer with the ESP32-CAM by enabling the PROGRAMMING mode on the board.

- **ESP32-CAM**:
  - **GND**: Set for Connection to FTDI GND
  - **5V**: Connect to FTDI VCC (5V)
  - **U0R**: Connect to FTDI TX
  - **U0T**: Regarding the serial connection to FTDI RX-connect it

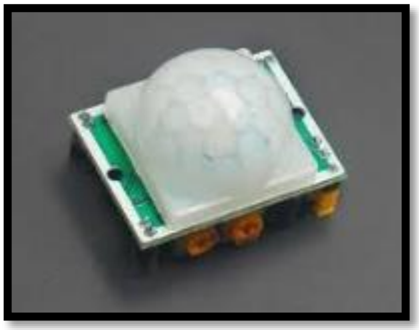
  - **IO0**: GND must be connected to a terminal for its programming mode.

FTDI Programmer:

- **GND**: Connect to ESP32-CAM GND
- **VCC (5V)**: Connect to ESP32-CAM 5V
- **TX**: Connect to ESP32-CAM U0R
- **RX**: Connect to ESP32-CAM U0T

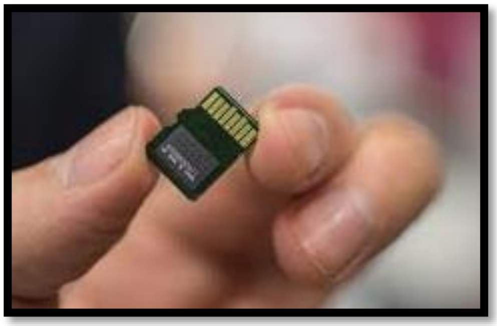
While uploading the code for the current design, ensure that IO0 is connected to the GND pin of the Arduino module. After uploading, removal of IO0 from GND and then resetting of ESP32-CAM is required..

2. PIR Sensor to ESP32-CAM:



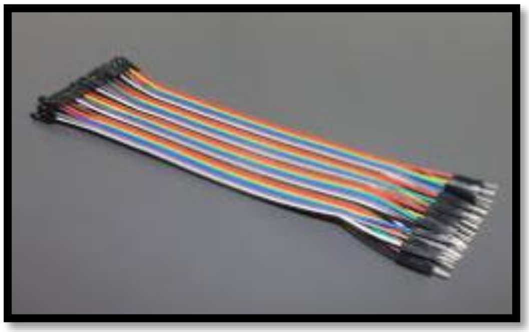
- **PIR Sensor**:
  - **VCC**: Connect to ESP32-CAM 3.3V or 5V
  - **GND**: The second step is to connect the ESP32-CAM ground pin or GND.
  - **OUT**: Set up the wire regarding ESP32-CAM GPIO pin (for instance GPIO-13)

3. MicroSD Card:



- The MicroSD card slot is separate from the ESP32-CAM module but is integrated into the mainboard, specifically into the ESP32-SOLO-1 module. Place the MicroSD card into the slot located on the board of the ESP32-CAM.

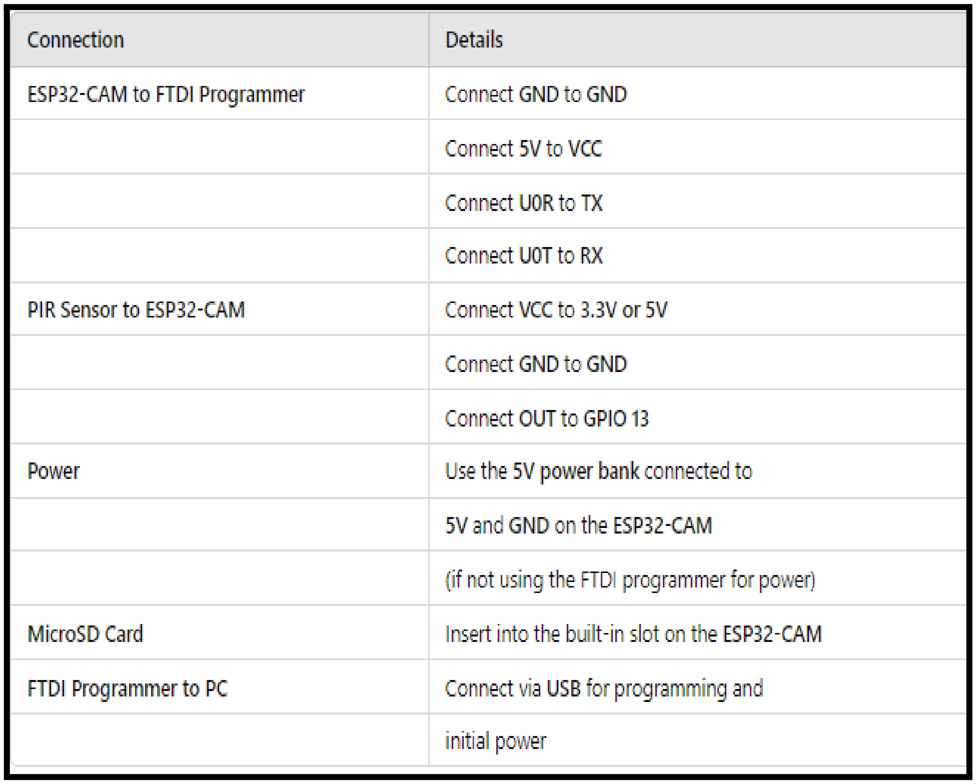
4. Power Supply:



- **5V Power Bank**:
  - Attach a 5V pin and GND to the power bank in order to give power to the ESP32-CAM module. However, use the FTDI programmer for power with the computer, while interfacing the microcontroller.

5. Connection with PC:

- The USB cable has to be connected from the FTDI programmer to the PC usually in a serial port. It will also enable, incorporating programming into the ESP32-CAM and powering the device during the development.



Connection	Details
ESP32-CAM to FTDI Programmer	Connect GND to GND
	Connect 5V to VCC
	Connect U0R to TX
	Connect U0T to RX
PIR Sensor to ESP32-CAM	Connect VCC to 3.3V or 5V
	Connect GND to GND
	Connect OUT to GPIO 13
Power	Use the 5V power bank connected to
	5V and GND on the ESP32-CAM
	(if not using the FTDI programmer for power)
MicroSD Card	Insert into the built-in slot on the ESP32-CAM
FTDI Programmer to PC	Connect via USB for programming and
	initial power

5. System Operation:

1. The PIR sensor continuously scans the raw environment for motion detection if it is present.
2. When the sensor picks a movement, the ESP32-CAM then pushes a button to turn on the OV2640 camera to take pictures.
3. The captured images are temporarily stored in the MicroSD card for later or further use or analysis.
4. Power to the system can be supplied through the FTDI programmer that is interfaced with the PC port or an external 5V power supply such as the power bank.

Code:

```
#include "esp_camera.h"

#include <FS.h>

#include "SD_MMC.h"

#include <WiFi.h>

// Camera pins configuration
```

```
#define PWDN_GPIO_NUM    -1
#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 PIR_PIN          13 // GPIO pin for PIR sensor
#define LED_PIN           4  // On-board LED
void setup() {
  Serial.begin(115200);
  pinMode(PIR_PIN, INPUT);
  pinMode(LED_PIN, OUTPUT);
  // Initialize the camera
  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;

if(psramFound()){

  config.frame_size = FRAMESIZE_UXGA;

  config.jpeg_quality = 10;

  config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    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);
    return;
  }
  // Start SD Card
  if(!SD_MMC.begin()){
  Serial.println("Card Mount Failed");
  return;
  }
}
void loop() {
  if(digitalRead(PIR_PIN) == HIGH) {
    digitalWrite(LED_PIN, HIGH);
    Serial.println("Motion detected");
    camera_fb_t * fb = NULL;
    // Take Picture with Camera
    fb = esp_camera_fb_get();
    if(!fb) {
      Serial.println("Camera capture failed");
      return;
    }
    // Path where new picture will be saved in SD Card
    String path = "/picture.jpg";
    fs::FS &fs = SD_MMC;
    Serial.printf("Picture file name: %s\n", path.c_str());
    File file = fs.open(path.c_str(), FILE_WRITE);
    if(!file){
      Serial.println("Failed to open file in writing mode");
    }
    else {
      file.write(fb->buf, fb->len); // payload (image), payload length
      Serial.printf("Saved file to path: %s\n", path.c_str());
    }
    file.close();
    esp_camera_fb_return(fb);
    digitalWrite(LED_PIN, LOW);
    delay(10000); // Delay to avoid multiple captures
  }
}
```