Experiment 7

Aim: To implement and study ESP32-CAM module.

Theory: The ESP32-CAM is a low-cost, compact microcontroller with an integrated camera module and built-in WiFi and Bluetooth capabilities. It is widely used for IoT applications such as surveillance, facial recognition, and image processing. The ESP32-CAM lacks a USB interface, so it requires an FTDI programmer for flashing firmware and serial communication.

The ESP32-CAM supports 802.11 b/g/n WiFi, which allows it to connect to wireless networks and transmit data to cloud services. The WiFi module is configured in station mode (STA mode), where the device connects to a router and gains internet access.

The OV2640 camera sensor in the ESP32-CAM captures images in various resolutions, including QVGA (320x240), SVGA (800x600), and UXGA (1600x1200). The captured image can be processed and stored locally or sent to a remote cloud server for further analysis.

To send images to the cloud, the ESP32-CAM follows these steps:

1. Capture Image: The camera module takes a picture when triggered by software or an external event.
2. Convert Image to Base64 (Optional): Since image files are binary, they can be encoded to Base64 for easy transmission via HTTP.
3. Establish a WiFi Connection: The ESP32-CAM connects to a pre-configured WiFi network.
4. Upload Image using HTTP or MQTT: The image is sent to a cloud server via REST API (HTTP POST request) or an IoT messaging protocol like MQTT.

Popular cloud services that support image upload from ESP32-CAM include:

- Google Drive or Firebase Storage: Using Firebase SDK or REST API.
- AWS S3: Sending images using pre-signed URLs.
- Thingspeak or Adafruit IO: For IoT-based applications with image data logging.
- Custom Web Server: Deploying a Flask/Django server that receives and processes images.

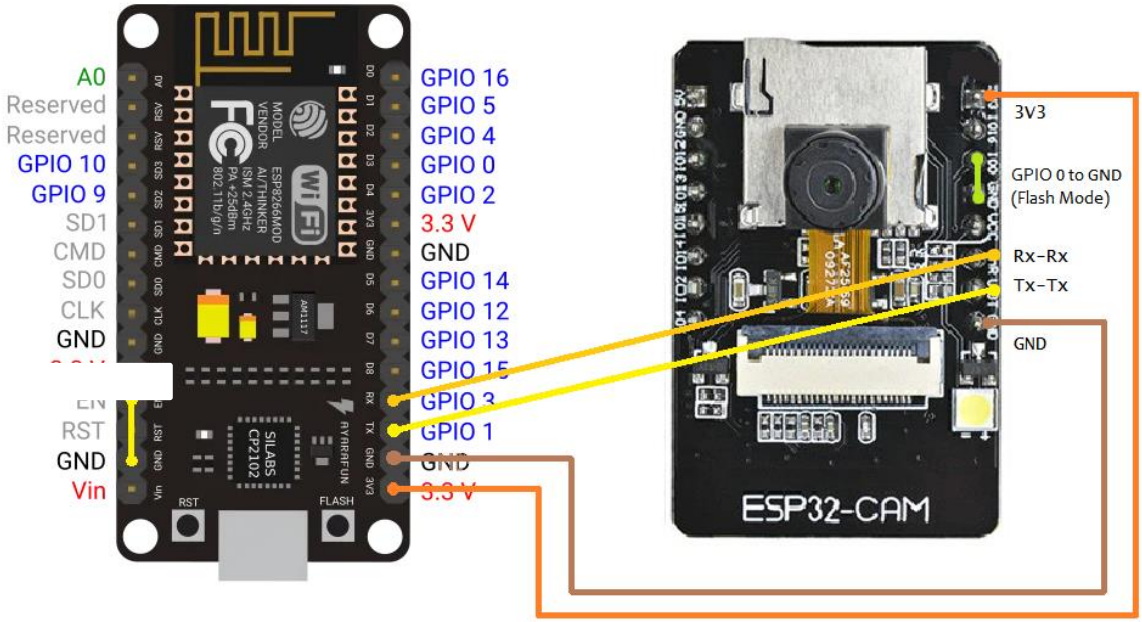
The ESP32-CAM can be programmed using the Arduino IDE with the following libraries:

- WiFi.h – for network connectivity
- ESP32HTTPClient.h – for HTTP communication
- FS.h & SPIFFS.h – for temporary file storage
- ArduinoJson.h – for handling JSON responses if needed

A simple workflow for programming ESP32-CAM:

1. Install the ESP32 board package in Arduino IDE.
2. Write a sketch to initialize the camera, connect to WiFi, and send the image.
3. Compile and upload the code via an FTDI programmer.
4. Monitor serial output to debug the process.
5.

Observations:



Result: Successfully implemented and studied ESP32-CAM module.