International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 07 | Jul 2025 www.irjet.net p-ISSN: 2395-0072

# Web-based Video Streaming using ESP32-CAM and FTDI Module

**Tanmay Khaladkar**

*Tanmay Khaladkar, Department of Mechatronics Engineering,*
*Symbiosis Skills & Professional University, Pune, Maharashtra, India*

---

**Abstract -** *The ESP32-CAM module interfaced with an FTDI programmer is used in this paper to present a web-based video streaming system. The goal is to develop a small, affordable surveillance or remote monitoring system. The Arduino IDE is used to program the ESP32-CAM module, and an onboard micro web server is used to connect to a web browser. The ESP32-CAM is programmed and debugged using the FTDI module. The system exhibits real-time video streaming without requiring expensive infrastructure or third-party cloud services. This paper presents the performance analysis, connection diagrams, and implementation details.*

***Key Words***: ESP32-CAM, FTDI Module, Web Streaming, Surveillance System, IoT Camera, Real-time Video, Embedded Web Server

## 1.INTRODUCTION

The ESP32-CAM module offers a workable solution for low-cost surveillance in light of the growing demand for small and reasonably priced monitoring systems. The ESP32-CAM is a single board that combines a camera, Wi-Fi, and microcontroller functions. It can be readily programmed via USB by connecting it to an FTDI programmer. The creation of a video streaming system with these elements is described in detail in this paper. Applications include remote equipment monitoring, wildlife observation, and home security.

### 1.1 Motivation and Objective

The ESP32-CAM module offers a workable solution for low-cost surveillance in light of the growing demand for small and reasonably priced monitoring systems. Applications include remote equipment monitoring, wildlife observation, and home security.

### 1.2 Overview of ESP32-CAM and FTDI Interface

The ESP32-CAM is a single board that integrates microcontroller, Wi-Fi, and camera capabilities. Connecting it to an FTDI programmer via USB makes programming it simple. This paper provides a detailed description of how to create a video streaming system with these components.

## 2. SYSTEM COMPONENTS AND CONNECTIONS

The hardware parts used include the FTDI programmer, the ESP32-CAM module, and the jumper wires that are needed. You can connect the FTDI to the ESP32-CAM using the TX-RX, GND, and 5V lines. To get to programming mode, pull GPIO0 low. You can program the ESP32-CAM with the Arduino IDE and the right libraries. The system also has a local Wi-Fi network for streaming. A connection diagram shows how everything is set up.

The ESP32-CAM module is a small, inexpensive development board that comes with an integrated OV2640 camera, Wi-Fi and Bluetooth support, and can be used as a microcontroller. This development module is ideal for low power image processing applications and/or streaming video wirelessly. The ESP32-CAM module is capable of handling various resolutions and has GPIO pins to connect sensors and other devices. You can program the ESP32-CAM using a third-party FTDI programmer via the Arduino IDE. To put the module into program mode, pull GPIO0 to GND, and it requires 5V of power. Once everything is set up, it can act as a light-weight webserver streaming MJPEG video in real time over a local Wi-Fi network. Many people enjoy using the ESP32-CAM for home automation, surveillance, and the Internet of Things (IOT.).

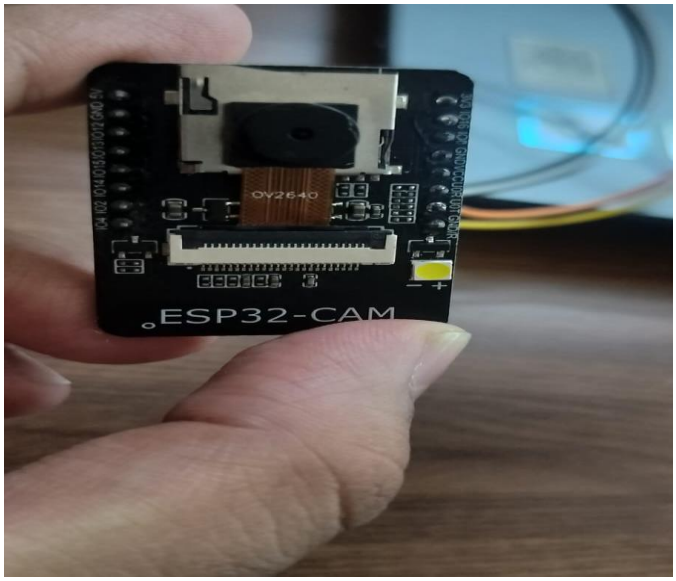


**figure -1**: ESP 32 Cam Module

The FTDI module is commonly used as a USB-to-Serial converter to upload programs into the development boards like the ESP32-CAM. The FTDI module acts as a reliable device to interface between the computer and a microcontroller in a UART protocol. The FTDI modules are typically just a cheap board that includes TX (Transmitter), RX (Receiver), 5V, 3.3V, and GND (ground) pins for easy interfacing. FTDI modules can be less overwhelming when working with serial communication while you are developing or debugging. Based on the module, this FTDI modules allows for voltage selection (3.3V or 5V) based on the onboard jumpers or solder pads. For uploading code, you need to appropriately wire the FTDI module and ground IO0 (RESET). Double check the TX and RX pins are not wired in reverse.

**Fig -2**: FTDI Module

## 3. SOFTWARE IMPLEMENTATION

The ESP32-CAM module can also stream MJPEG video as a web server. The Arduino sketch contains the code to initialize the Wi-Fi credentials, sets up the camera, and launches the web server. The streaming IP address can be used by any device connected on the same network to access the stream. There are a variety of resolutions (QQVGA to SVGA) that you can use for performance tuning. Controls for the camera (capture and pan/tilt) and status data (presumably for debug/diagnostics) are shown on the webpage as well.

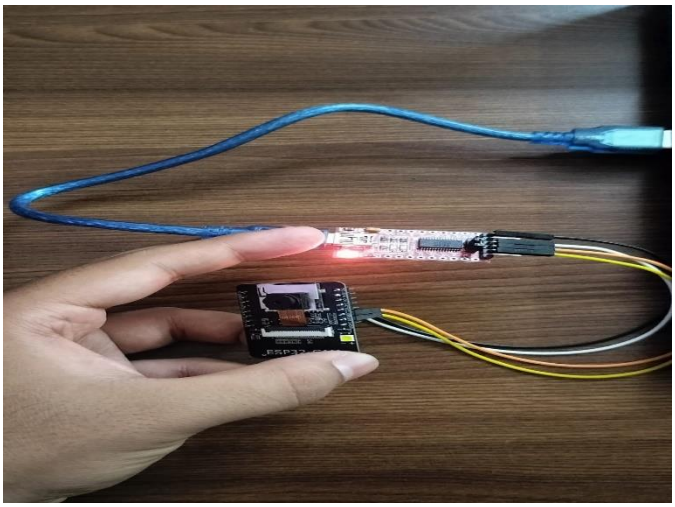

**Fig -3:**Connection Digram

## 4. RESULTS AND ANALYSIS

The prototype was tested over a local Wi-Fi network. The ESP32-CAM was able to stream video (live video) at frame rates of 8–15 fps; this depends on resolution and signal strength. In optimal network conditions, latency was not an issue. This performed acceptable for portable battery-based systems because it is low power. For aspects of monitoring, stream quality was acceptable.

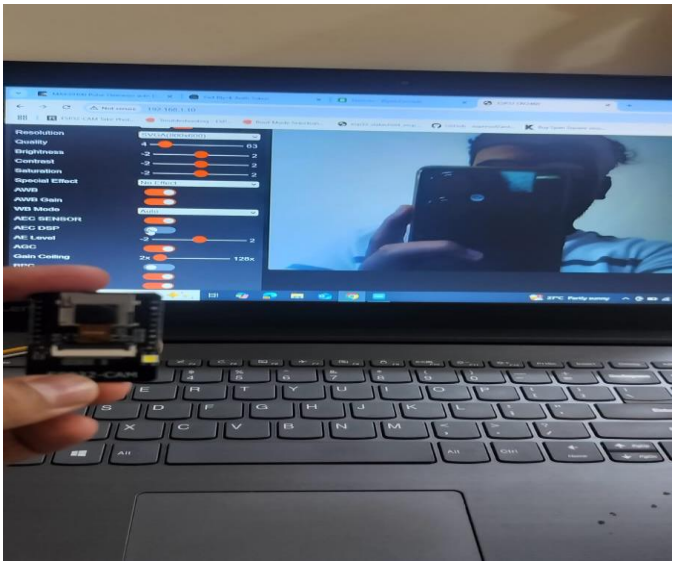

**Fig -4:** Result

## 5. CONCLUSION

This paper shows that ESP32-CAM and FTDI can be used to create an effective and inexpensive video surveillance system. The system works without external storage or cloud services, is small, and is very easy to program. The design adds value for small scale commercial, DIY, or educational use. Future improvements could include

motion detection alerts, cloud integration, and port operation and battery optimization.

## REFERENCES

[1] Espressif Systems, "ESP32-CAM Datasheet," 2021

[2] Arduino IDE – https://www.arduino.cc/en/software

[3] Random Nerd Tutorials, "ESP32-CAM Video Streaming Webserver"

[4] R. Nicole, "Real-time Embedded Camera Systems," Journal of IoT, 2020