

# Proyecto Final de Ingeniería Automática

## Servidor Web mediante el uso del ESP32-CAM

Autores:

Gilberto **Angles Geroy**

Marcos Daniel **Arribas Bayolo**

Miguel Alejandro **Cabrera Santos**

Cuarto Ing. Automática

Santa Clara

2024

Universidad Central “Marta Abreu” de Las Villas

## Introducción

El presente proyecto se centra en la implementación de un sistema de transmisión de video en tiempo real mediante un servidor web alojado en un microcontrolador ESP32-CAM. Este dispositivo, gracias a su capacidad de procesamiento y conectividad Wi-Fi integrada, se presenta como una herramienta ideal para aplicaciones de monitoreo remoto.

El objetivo principal de este proyecto es diseñar y configurar una cámara, que permita supervisar de forma remota a través de dispositivos móviles o computadoras conectados a la red distintas locaciones o personas. En el mundo ya varias personas han realizado proyectos con esta placa ESP-32 CAM, muchos de ellos disponibles en YouTube, pero este proyecto busca ser implementado en un mayor número de tareas, ya que contará con una IP estática establecida para el servidor web lo que hace que pueda ser comercializado ya que el servidor web será creado en la red local con un host de servidor en http://IP-ESP32-CAM. La solución propuesta busca ser económica, eficiente y de fácil implementación, aprovechando las capacidades del ESP32-CAM para capturar y transmitir imágenes en tiempo real.

Este trabajo abarca el diseño del servidor web, la configuración del módulo ESP32-CAM para el Streaming de video y la optimización del sistema para asegurar una transmisión estable y de calidad. Además, se consideran aspectos de seguridad y usabilidad, esenciales para garantizar un monitoreo confiable y seguro.

El proyecto también plantea su implementación en diversos ámbitos, destacando su utilidad en sistemas de videovigilancia para el monitoreo de espacios tanto interiores como exteriores. Además, este sistema resulta ideal para aplicaciones de seguridad en entornos donde se requiere una vigilancia continua, ya sea para observar el comportamiento de personas, el movimiento de animales domésticos o de granja, o para proteger áreas sensibles frente a intrusos.

## Desarrollo

### **Marco teórico:**

**1. ESP32-CAM**

* Arquitectura y características técnicas del ESP32-CAM.
* Módulo de cámara OV2640: especificaciones y funcionamiento.
* Capacidades de conectividad Wi-Fi y Bluetooth.
* Gestión de recursos limitados en sistemas embebidos.

**2. Transmisión de Video en Tiempo Real (Streaming)**

* Conceptos básicos de transmisión de video.
* Protocolos de transmisión (HTTP, RTSP, MJPEG).
* Compresión y calidad de video en sistemas de bajo consumo.

**3. Servidores Web Embebidos**

* Funcionamiento de servidores web en microcontroladores.
* Configuración de servidores web ligeros (ESPAsyncWebServer, WebServer).
* Manejo de rutas, solicitudes HTTP y transmisión de datos.

**4. Redes Inalámbricas y Comunicación IoT**

* Funcionamiento de redes Wi-Fi.
* Conceptos de IoT (Internet of Things).
* Seguridad en redes inalámbricas (encriptación, autenticación).

**5. Sistemas de Monitoreo Remoto**

* Aplicaciones de los sistemas de videovigilancia.
* Requisitos de seguridad y privacidad en sistemas de monitoreo.

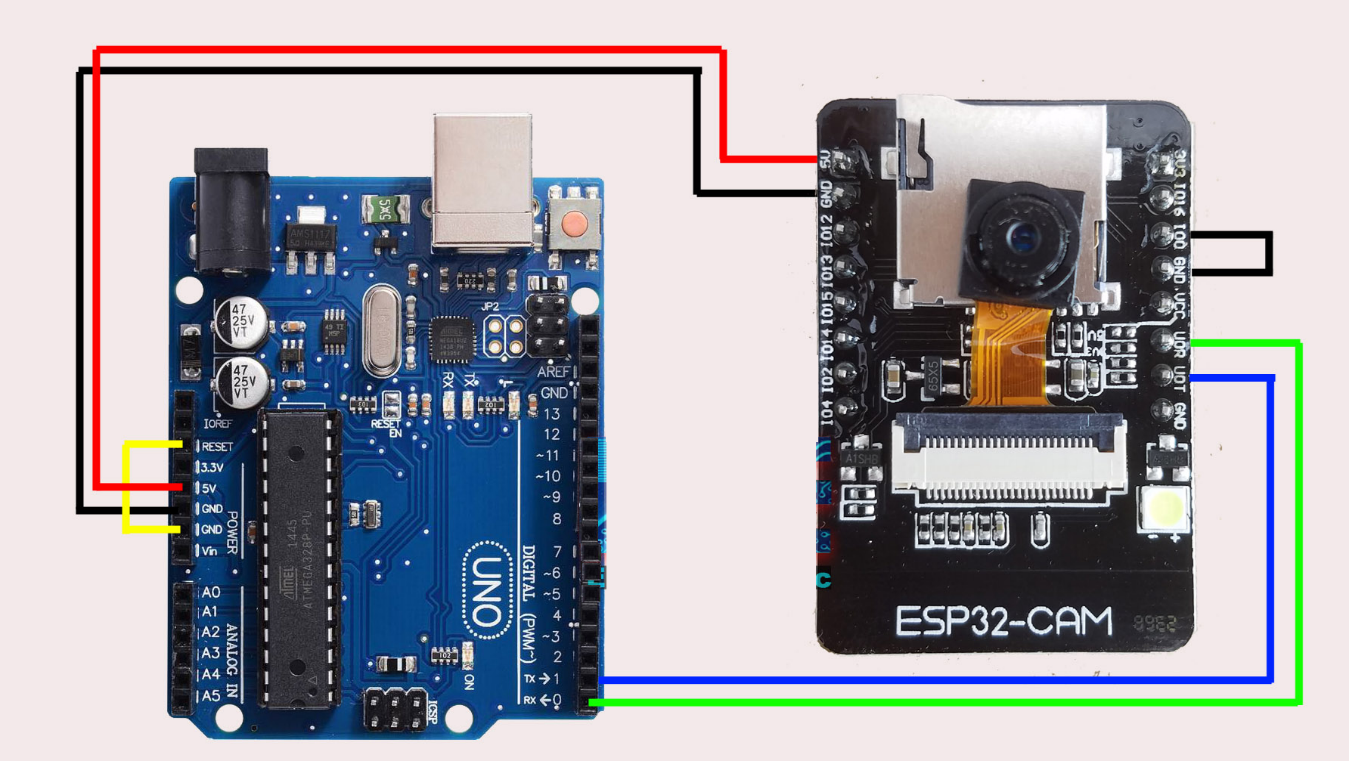
**6. Optimización de Recursos en Dispositivos de Bajo Consumo**

* Gestión de memoria y procesamiento en microcontroladores.
* Técnicas de ahorro energético.
* Estrategias para mejorar la estabilidad del streaming.

### Descripción detallada del diseño hardware y software

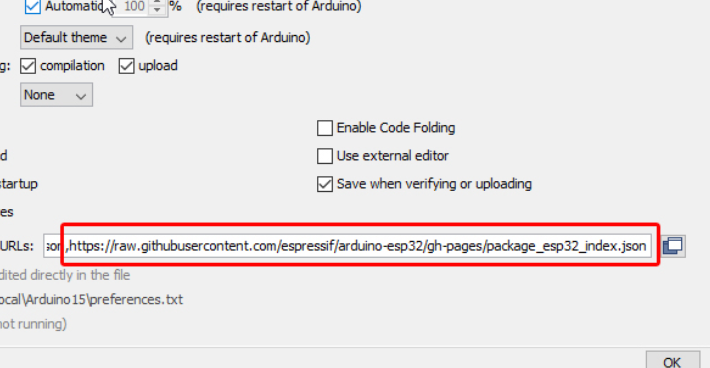
**Conexión del ESP32 con Arduino Uno o Arduino Nano:**

* Conectar el pin de Reset del Arduino al GND.
* Conectar el pin IO0 del ESP32-CAM al pin GND, para esto puedes usar un cable jumper tipo hembra a hembra.
* Conectar los pines 5V y GND del ESP32-CAM a los pines 5V y GND del Arduino.
* Conectar el pin de Recepción (RX) del ESP32-CAM al pin RX del Arduino.
* Conectar el pin de Transmisión (TX) del ESP32-CAM al pin TX del Arduino.
* La conexión del ESP32-CAM con el Arduino está completa.

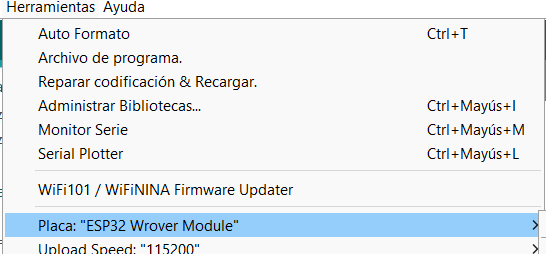


**Instalación del Administrador de Placas del ESP32-CAM:**

* Asegurarse de tener instalada la última versión del Arduino IDE en tu computadora. Copiar el siguiente enlace:  
  <https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json>
* Abrir el Arduino IDE, hacer clic en el menú Archivo y luego en Preferencias, y pegar la URL copiada.

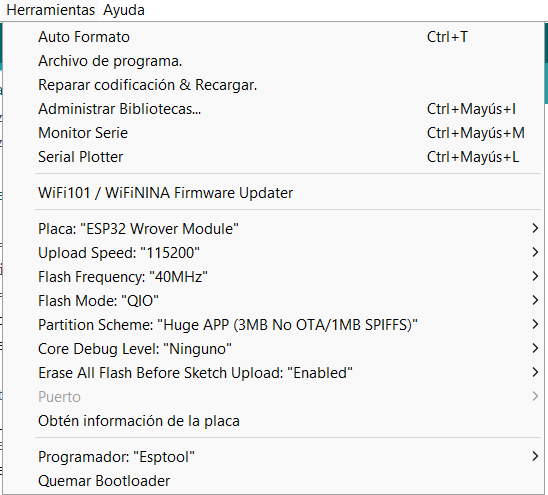


* Ahora, clic en el menú Herramientas, seleccionar Placa y luego Gestor de placas. Buscar ESP32 e instalar (esto puede tardar varios minutos).
* Verificar si la placa deseada se ha instalado correctamente. Como puedes ver, el ESP32 Wrover Module está disponible.



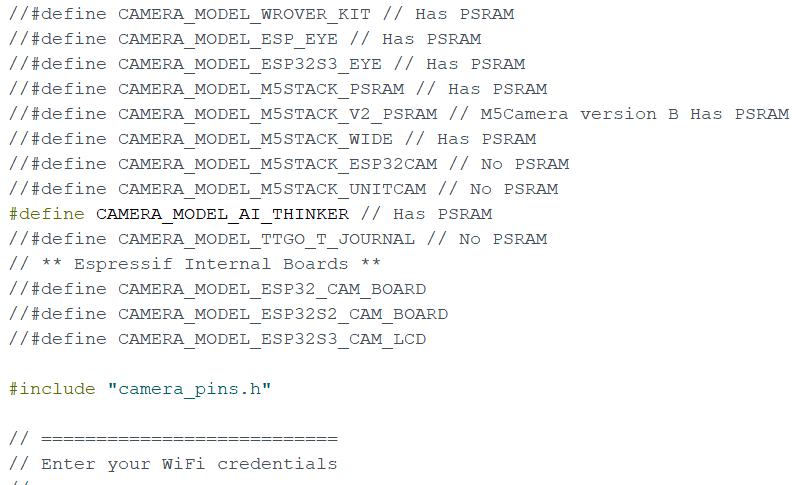
**Configuración del ESP32-CAM:**

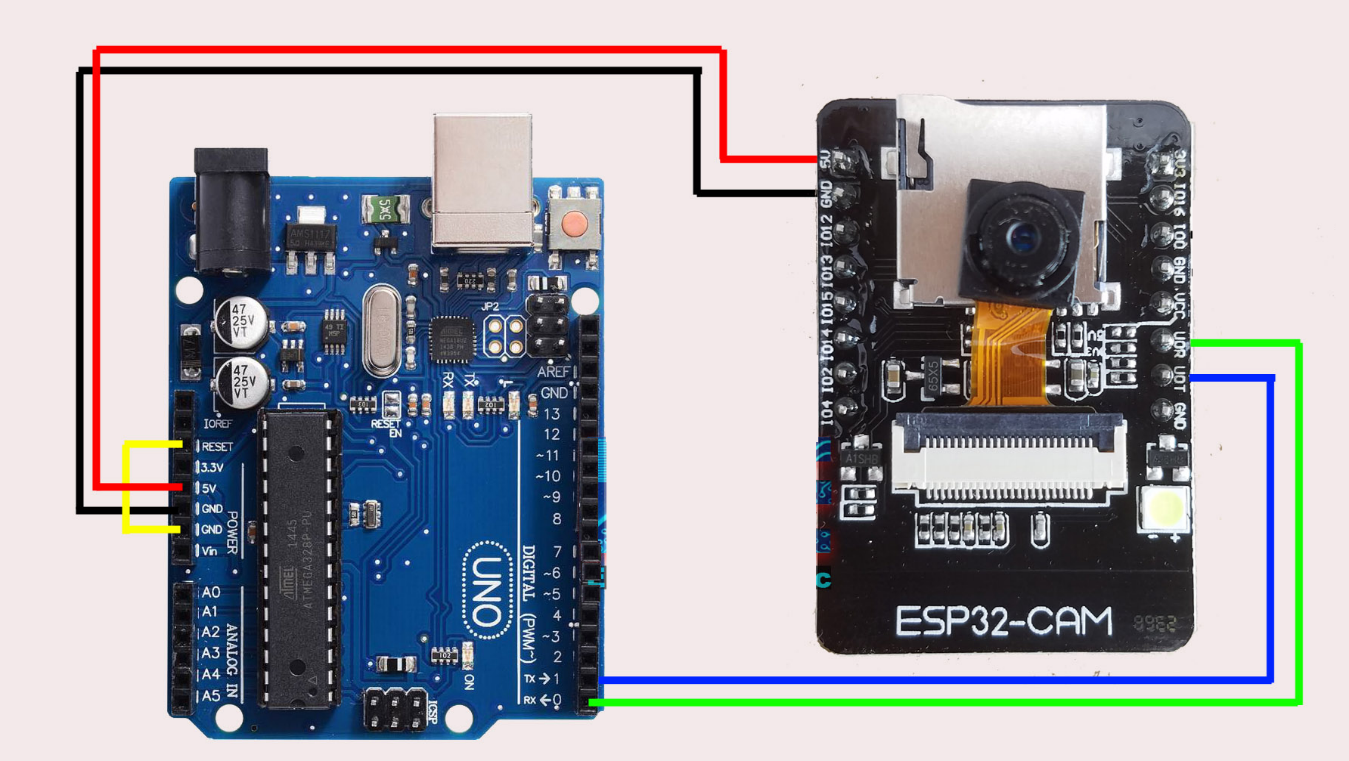
* En la barra superior seleccionar el menú Herramientas.
* Dirigirse a Flash Mode y seleccionar QIO.
* Seleccionar la Frecuencia de Flash como 40MHz.
* Seleccionar el Esquema de Particiones como Huge APP.
* Configurar la Velocidad de Carga en 115200.
* Finalmente, seleccionar el programador Esptool.
* Como puedes ver, el Puerto: “COM10” indica que la placa Arduino está conectada a la laptop mediante un cable USB.



**Carga del Programa al ESP32-CAM:**

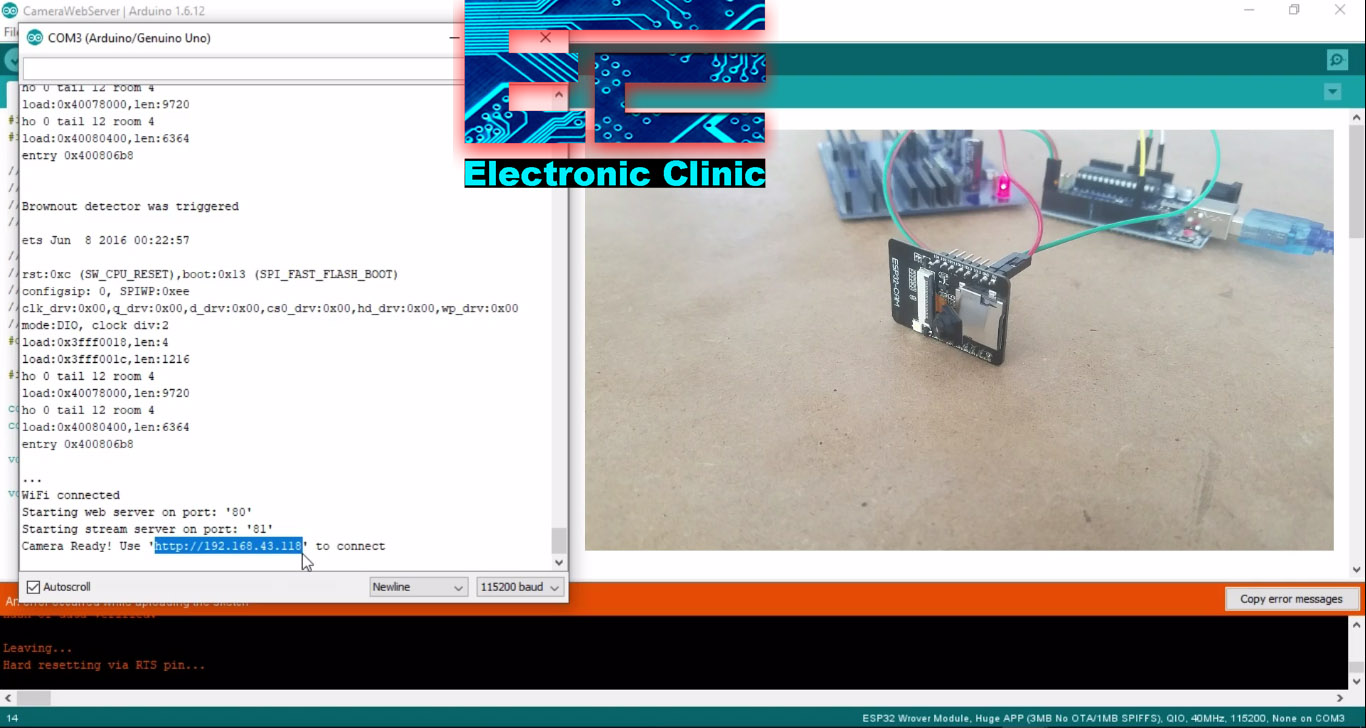
* Haz clic en Archivo > Ejemplos > ESP32 > Camera > CameraWebServer.
* Abre el programa CameraWebServer.
* Ingresa tu SSID y contraseña de WiFi.
* Selecciona el modelo de tu cámara.
* Todo está listo, ahora haz clic en el botón de Subir.
* Una vez cargado el programa, retira el cable que conecta el pin IO0 con GND.





**Ejecución del Streaming:**

* Abrir el Monitor Serial.
* Presionar el botón de reset, ubicado en la parte trasera del ESP32-CAM.
* En el Monitor Serial, se podrá ver que el módulo ESP32-CAM se ha conectado a la red WiFi, y mostrará la dirección IP local.
* Copiar el enlace, abrir un navegador de internet y pegarlo.
* Finalmente, hacer clic en el botón Start Stream y ¡listo!

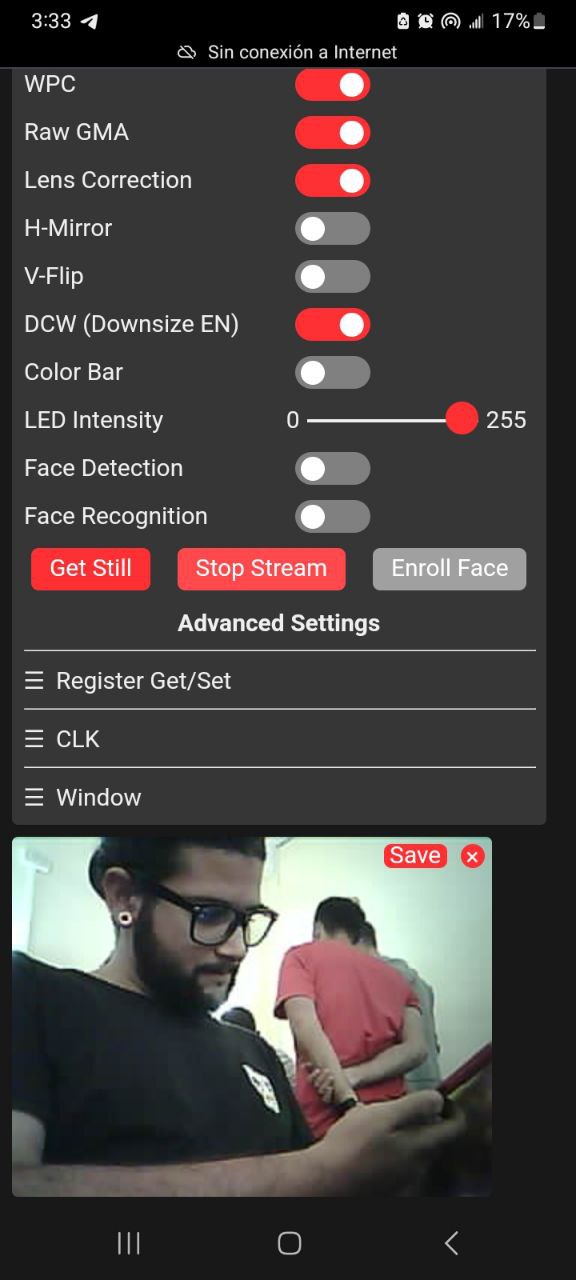


## Resultados

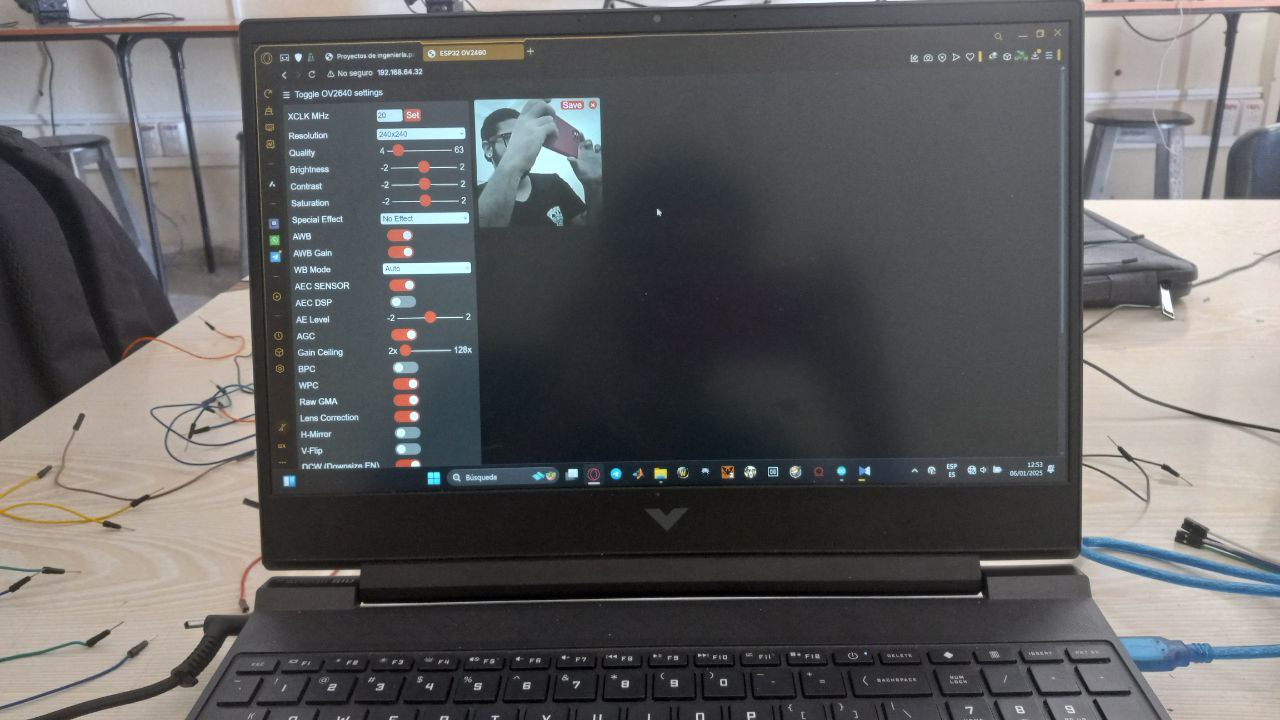
A continuación, se presentan varias imágenes que evidencian el correcto funcionamiento del sistema de monitoreo implementado con el ESP32-CAM. En estas fotos se puede observar la transmisión en tiempo real del video capturado por la cámara, visualizado a través de un navegador web mediante la dirección IP estática que se estableció en el código que se encuentra en el repositorio de GitHub. Y es precisamente esta característica la que lo sitúa por encima de otros que han sido desarrollados previamente, ya que una dirección IP estática produce que se pueda comercializar a un mayor número de sectores, tanto privados como empresariales. En ningún momento se debe utilizar el IDE de Arduino para conocer la IP del servidor web, por ello se crea una aplicación que redirige al usuario a la página donde se encontrará alojado el servidor web dentro de la red local de su propietario.

Además de estas características se cuenta con resultados que demuestran la estabilidad de la conexión Wi-Fi con la que se establece la conexión del ESP32 al punto de acceso, la calidad de imagen obtenida (la cual puede ser fácilmente variada por los usuarios) y la eficacia del sistema. Estas pruebas validan el cumplimiento de los objetivos propuestos, destacando la viabilidad del uso del ESP32-CAM en aplicaciones de monitoreo remoto.

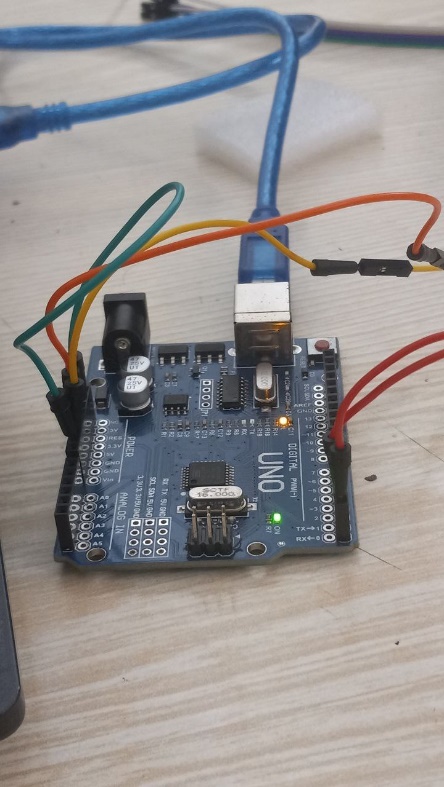
**Imágenes tomadas desde el móvil:**

**Imagen tomada de la PC:**

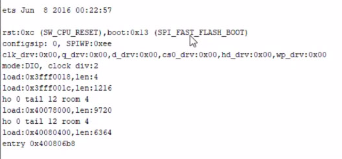
****

**Imágenes del hardware:**

** **

## Discusión

Es fundamental destacar que pueden surgir diversos problemas durante la implementación del sistema. Entre los inconvenientes más comunes se encuentran una conexión inadecuada o inestable del microcontrolador ESP32-CAM, fallos en la inicialización de la cámara debido a configuraciones incorrectas o posibles defectos en el hardware, y errores relacionados con la carga o ejecución del firmware. Además, pueden presentarse interrupciones en la transmisión de video causadas por una señal Wi-Fi débil o interferencias en la red. En algunos casos específicos, se puede observar el siguiente mensaje de error



Al investigar en foros especializados y diversas comunidades de desarrolladores enfocados en aplicaciones de IoT, se ha identificado que varios problemas asociados al funcionamiento del ESP32-CAM pueden estar relacionados directamente con fallos en el propio hardware. Entre estos inconvenientes destacan fallas en el suministro de energía, que pueden generar reinicios inesperados o afectar la estabilidad del dispositivo, y defectos en la PSRAM de la placa, los cuales pueden provocar errores en la transmisión de datos o en la carga de imágenes. Además, se ha reportado que una soldadura deficiente en el módulo que proporciona conectividad Wi-Fi al ESP32-CAM puede ocasionar desconexiones intermitentes o pérdida total de señal. No obstante, a pesar de estos posibles inconvenientes, se ha observado que en ciertas ocasiones la configuración del dispositivo ha funcionado de manera óptima, sin presentar fallos, lo que sugiere que estos problemas no son constantes y pueden depender de factores específicos del entorno o de la calidad del ensamblaje del hardware.

## Conclusiones

La implementación de un sistema de monitoreo remoto utilizando el microcontrolador ESP32-CAM ha demostrado ser una solución eficiente, económica y funcional para el desarrollo de un baby monitor. El dispositivo logró transmitir video en tiempo real a través de un servidor web, cumpliendo satisfactoriamente con los objetivos planteados. La integración de hardware y software permitió comprobar la capacidad del ESP32-CAM para manejar tareas de transmisión de datos de forma estable, aprovechando su conectividad Wi-Fi.

Asimismo, el proyecto puso en práctica conceptos fundamentales de sistemas embebidos, redes inalámbricas y servidores web, consolidando los conocimientos adquiridos en la asignatura de Ingeniería Automática. La experiencia obtenida durante el desarrollo del proyecto resalta la versatilidad de este microcontrolador para aplicaciones de bajo costo relacionadas con la seguridad y monitoreo.

## Recomendaciones

Para futuros trabajos de mejora e implementación, se sugieren las siguientes recomendaciones:

1. Optimización de la Calidad de Video: Explorar técnicas de compresión de video y ajuste dinámico de la resolución para mejorar la calidad de imagen sin comprometer el rendimiento del sistema.
2. Implementación de Seguridad: Incorporar medidas de seguridad, como autenticación de usuarios y cifrado de datos, para proteger la transmisión de video y evitar accesos no autorizados.
3. Alimentación Autónoma: Integrar sistemas de alimentación mediante baterías recargables o paneles solares para hacer el dispositivo más independiente y portátil.
4. Detección de Movimiento: Añadir sensores de movimiento o implementar algoritmos de reconocimiento para alertar al usuario en caso de actividad inusual.
5. Integración con Sistemas Domóticos: Conectar el sistema a plataformas de domótica (como Google Home, Home Assistant y demás) para ampliar sus funcionalidades dentro de entornos inteligentes.

Estas mejoras permitirían ampliar las capacidades del sistema, ofreciendo una solución más robusta, segura y versátil para aplicaciones de monitoreo.