FPV RC CAR

WiFi Controlled Robot with Live Video Stream

Edgar Gutierrez

ECE 528L - Robotics and Embedded Systems

Instructor: Aaron Nanas

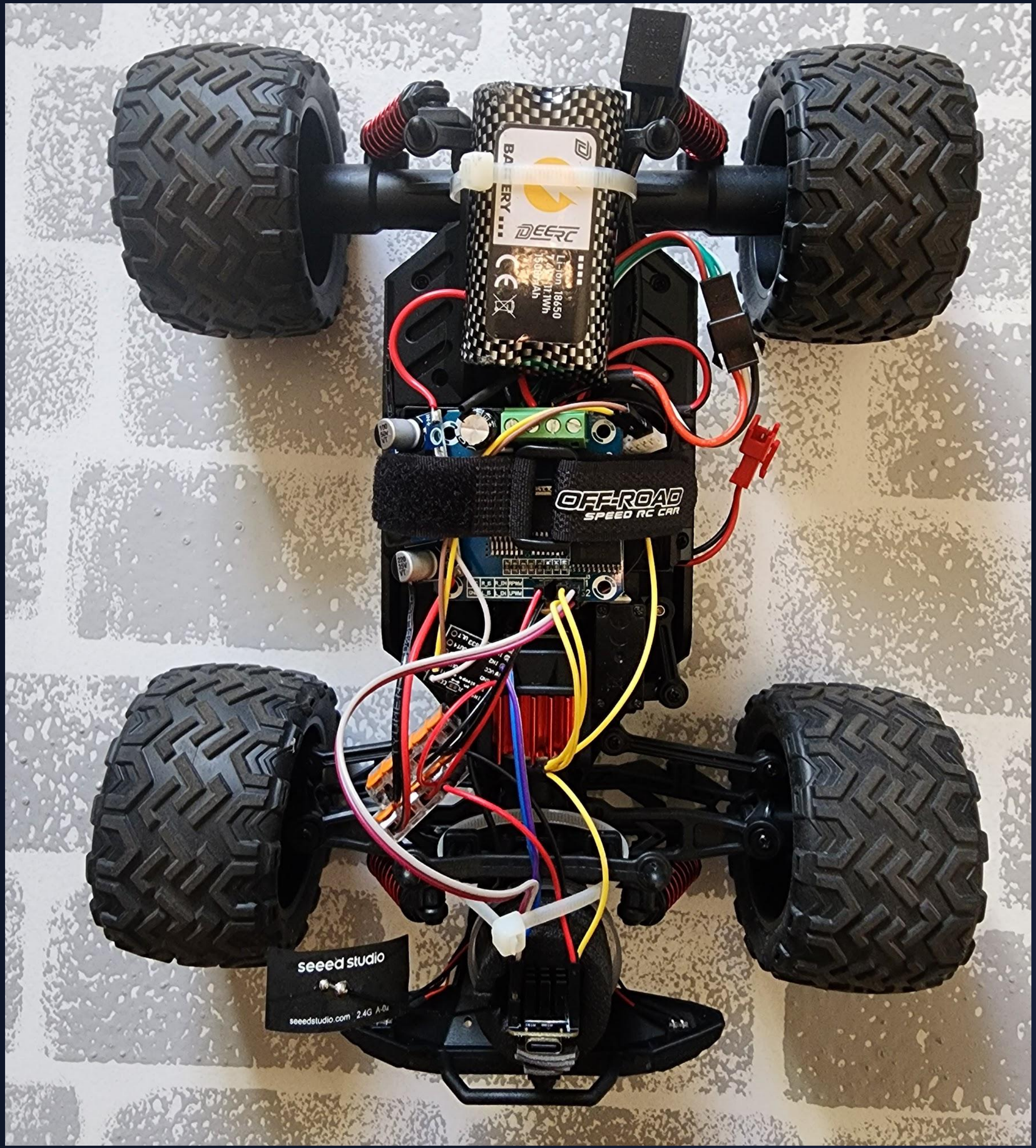
PROJECT OBJECTIVE

GOAL

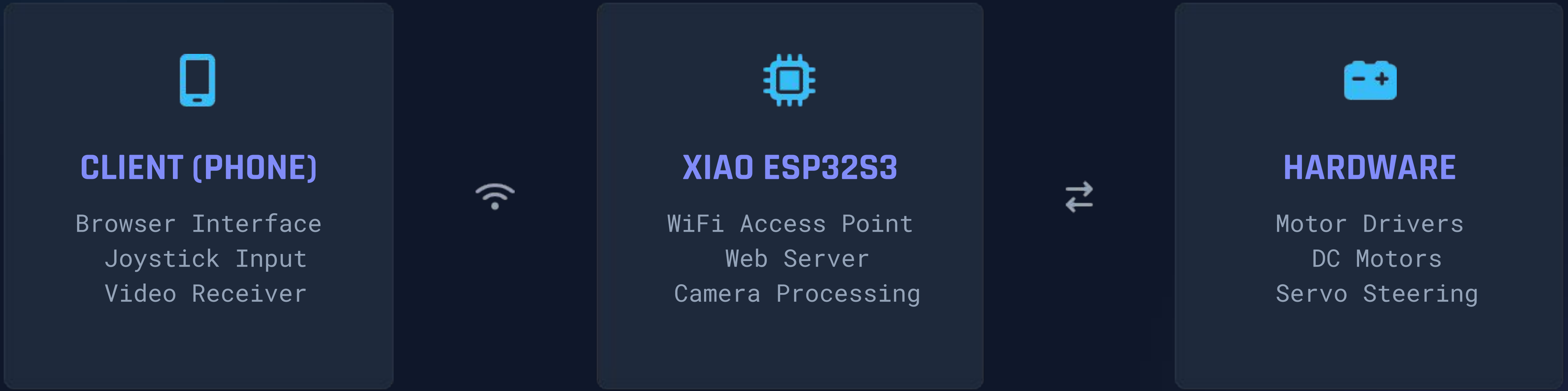
To design and implement a First-Person View (FPV) Remote-Controlled Car that integrates embedded systems concepts for motor control, wireless communication, and real-time video transmission.

KEY FEATURES

- Real-time MJPEG Video Streaming
- Browser-based Wireless Control (WiFi)
- PWM Motor Speed Control
- Low-latency Response



SYSTEM ARCHITECTURE



CLIENT (PHONE)

Browser Interface
Joystick Input
Video Receiver

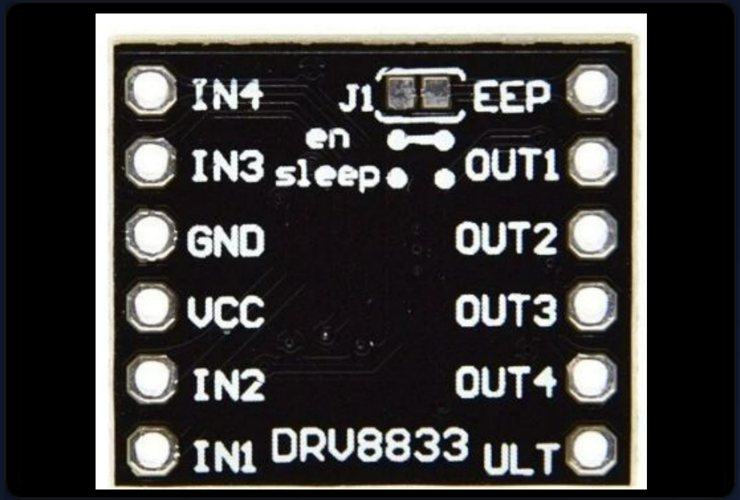
XIAO ESP32S3

WiFi Access Point
Web Server
Camera Processing

HARDWARE

Motor Drivers
DC Motors
Servo Steering

HARDWARE COMPONENTS



CONTROLLER

Seeed Studio XIAO ESP32S3 Sense

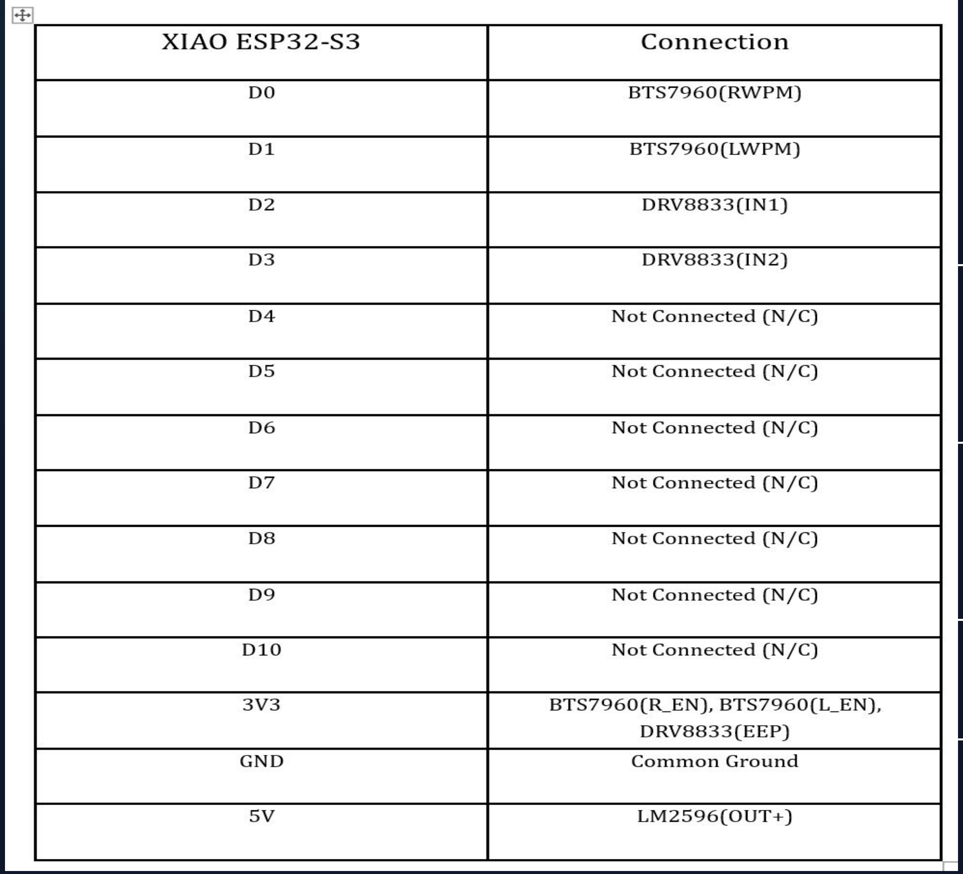
CHASSIS

Amazon RC Car

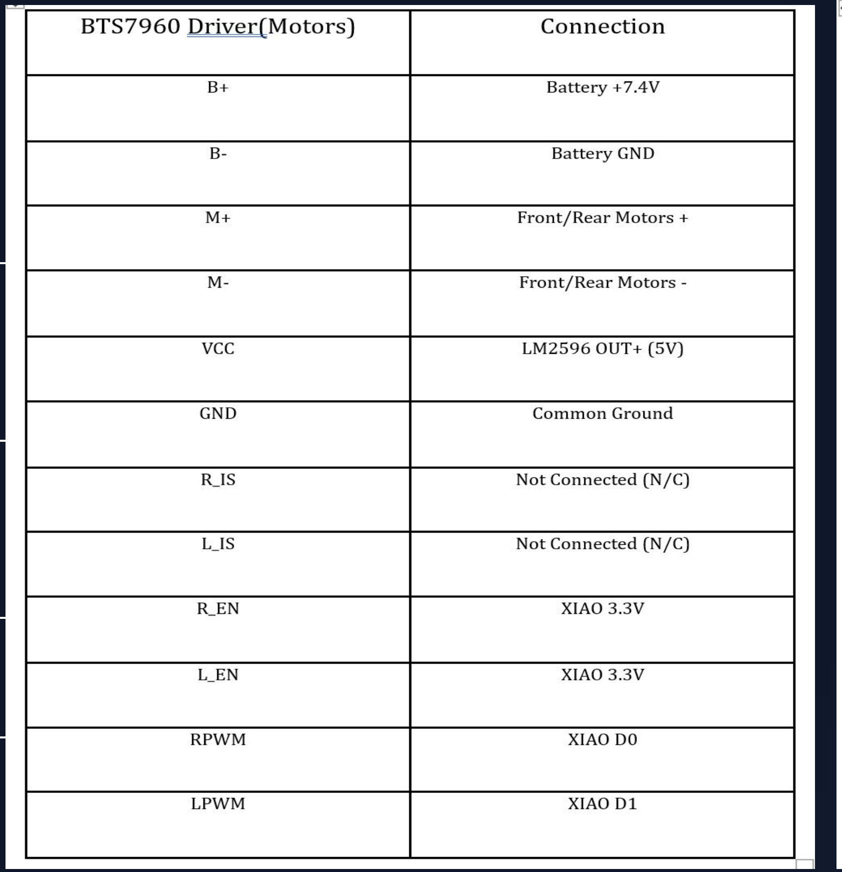
ACTUATION

Motor Drivers:DRV8833 / BTS7960

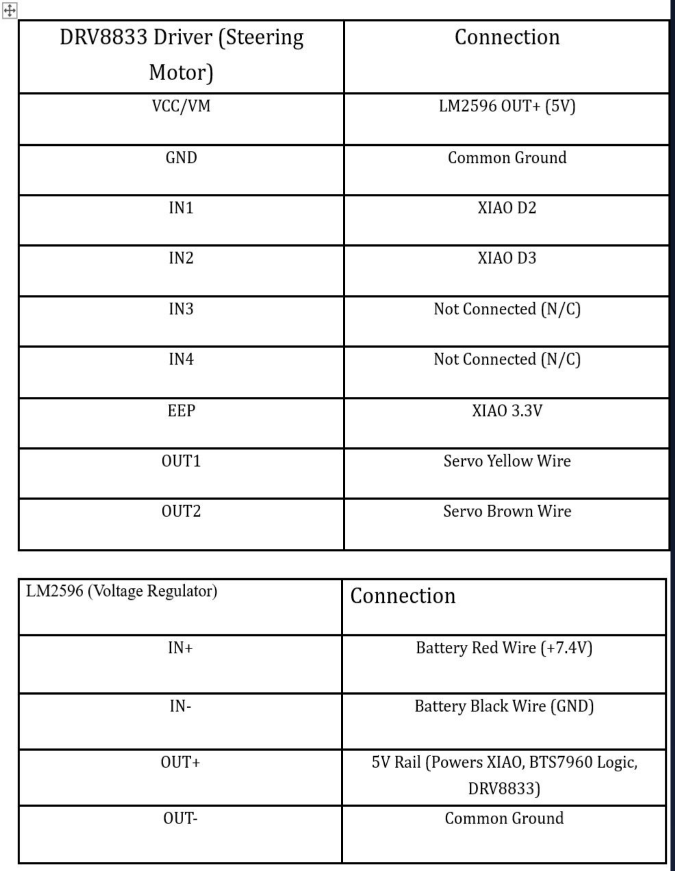
PINOUT CONFIGURATION



XIAO ESP32-S3	Connection
D0	BTS7960(RWPM)
D1	BTS7960(LWPM)
D2	DRV8833(IN1)
D3	DRV8833(IN2)
D4	Not Connected (N/C)
D5	Not Connected (N/C)
D6	Not Connected (N/C)
D7	Not Connected (N/C)
D8	Not Connected (N/C)
D9	Not Connected (N/C)
D10	Not Connected (N/C)
3V3	BTS7960(R_EN), BTS7960(L_EN), DRV8833(EEP)
GND	Common Ground
5V	LM2596(OUT+)



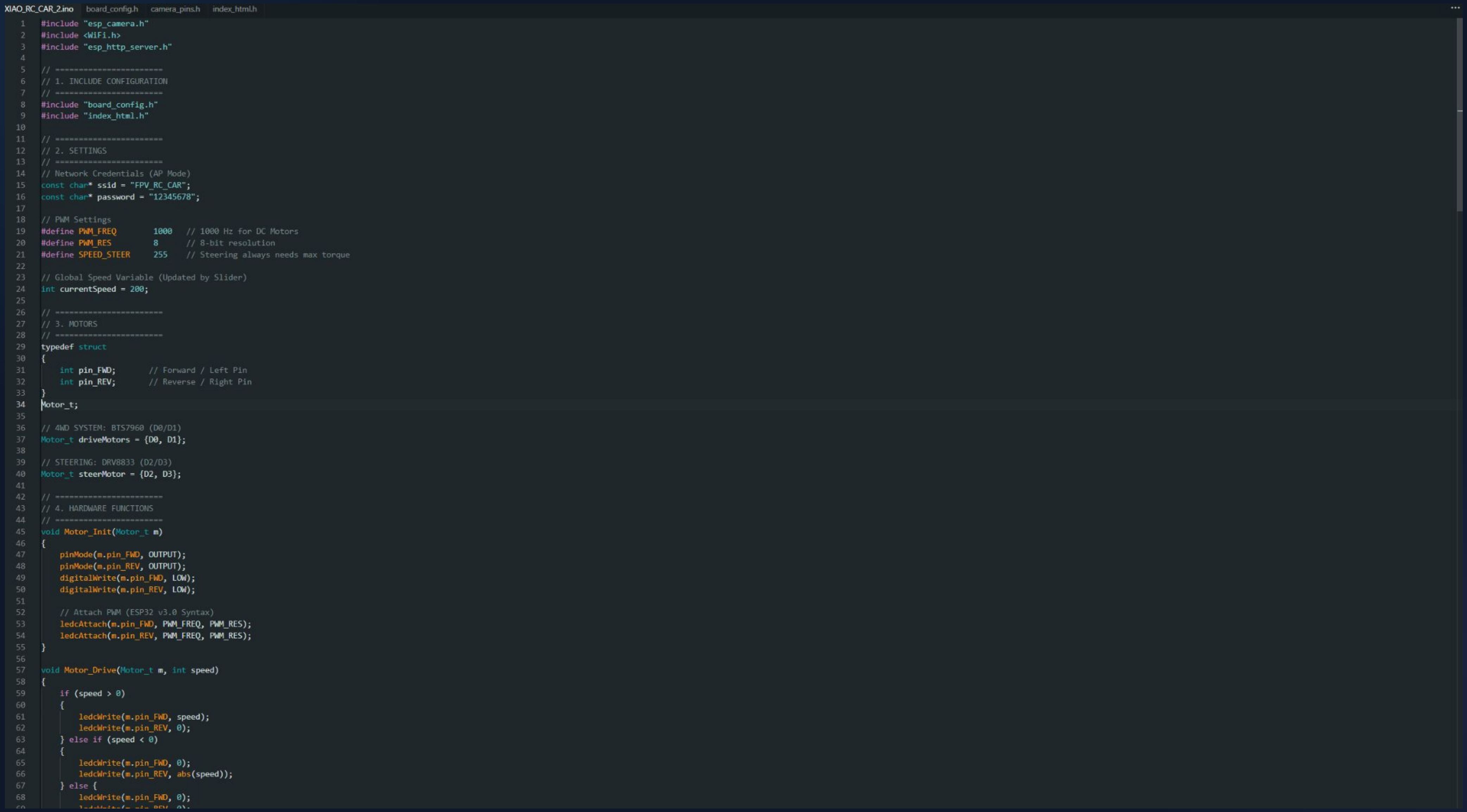
BTS7960 Driver(Motors)	Connection
B+	Battery +7.4V
B-	Battery GND
M+	Front/Rear Motors +
M-	Front/Rear Motors -
VCC	LM2596 OUT+ (5V)
GND	Common Ground
R_IS	Not Connected (N/C)
L_IS	Not Connected (N/C)
R_EN	XIAO 3.3V
L_EN	XIAO 3.3V
RPWM	XIAO D0
LPWM	XIAO D1



DRV8833 Driver (Steering Motor)	Connection
VCC/VM	LM2596 OUT+ (5V)
GND	Common Ground
IN1	XIAO D2
IN2	XIAO D3
IN3	Not Connected (N/C)
IN4	Not Connected (N/C)
EEP	XIAO 3.3V
OUT1	Servo Yellow Wire
OUT2	Servo Brown Wire

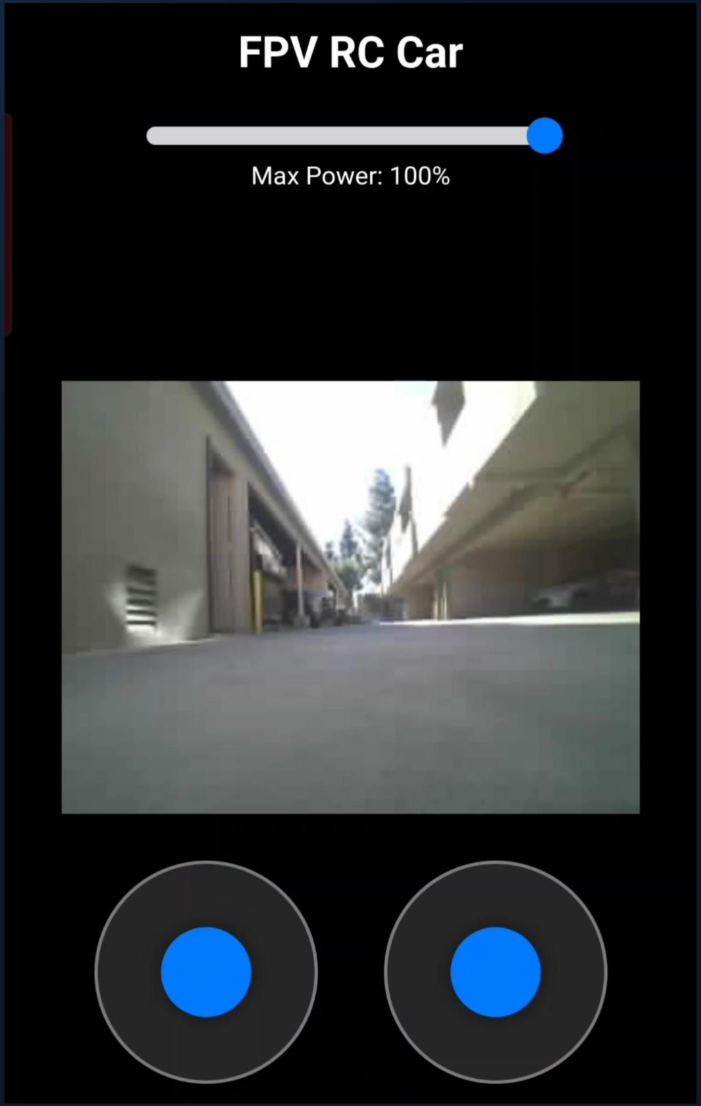
LM2596 (Voltage Regulator)	Connection
IN+	Battery Red Wire (+7.4V)
IN-	Battery Black Wire (GND)
OUT+	5V Rail (Powers XIAO, BTS7960 Logic, DRV8833)
OUT-	Common Ground

Code



```cpp
#include "esp_camera.h"
#include <WiFi.h>
#include "esp_http_server.h"

// =======================
// 1. INCLUDE CONFIGURATION
// =======================
#include "board_config.h"
#include "index_html.h"

// =======================
// 2. SETTINGS
// =======================
// Network Credentials (AP Mode)
const char* ssid = "FPV_RC_CAR";
const char* password = "12345678";

// PWM Settings
#define PWM_FREQ        1000   // 1000 Hz for DC Motors
#define PWM_RES         8      // 8-bit resolution
#define SPEED_STEER     255    // Steering always needs max torque

// Global Speed Variable (Updated by Slider)
int currentSpeed = 200;

// =======================
// 3. MOTORS
// =======================
typedef struct
{
    int pin_FWD;       // Forward / Left Pin
    int pin_REV;       // Reverse / Right Pin
}
Motor_t;

// 4WD SYSTEM: BTS7960 (D0/D1)
Motor_t driveMotors = {D0, D1};

// STEERING: DRV8833 (D2/D3)
Motor_t steerMotor = {D2, D3};

// =======================
// 4. HARDWARE FUNCTIONS
// =======================
void Motor_Init(Motor_t m)
{
    pinMode(m.pin_FWD, OUTPUT);
    pinMode(m.pin_REV, OUTPUT);
    digitalWrite(m.pin_FWD, LOW);
    digitalWrite(m.pin_REV, LOW);

    // Attach PWM (ESP32 v3.0 Syntax)
    ledcAttach(m.pin_FWD, PWM_FREQ, PWM_RES);
    ledcAttach(m.pin_REV, PWM_FREQ, PWM_RES);
}

void Motor_Drive(Motor_t m, int speed)
{
    if (speed > 0)
    {
        ledcWrite(m.pin_FWD, speed);
        ledcWrite(m.pin_REV, 0);
    } else if (speed < 0)
    {
        ledcWrite(m.pin_FWD, 0);
        ledcWrite(m.pin_REV, abs(speed));
    } else {
        ledcWrite(m.pin_FWD, 0);
        ledcWrite(m.pin_REV, 0);
```

WEB CONTROL INTERFACE



FPV RC Car

Max Power: 100%

TOUCH INTERACTION

The interface is built using HTML5 & JavaScript, optimized for mobile touchscreens.

- **Dual Joysticks:** Left stick for Throttle (Forward/Back), Right stick for Steering (Left/Right).
- **Speed Slider:** Real-time adjustment of maximum PWM duty cycle (100-255).

LATENCY OPTIMIZATION

FRAME CONFIGURATION

To ensure low latency over the SoftAP network, the camera is configured for speed over resolution.

```
config.frame_size = FRAMESIZE_QVGA;
config.jpeg_quality = 15;
config.fb_count = 2;
```

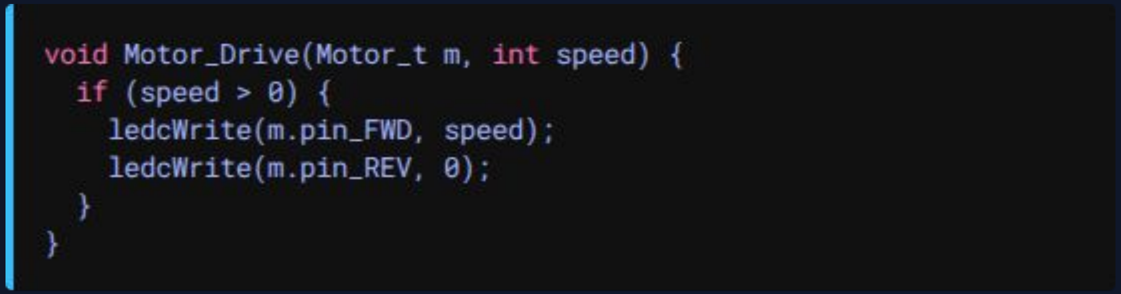
GRAB MODE

The CAMERA_GRAB_LATEST mode is crucial. It discards older buffered frames, ensuring the user always sees the most current reality, reducing "video lag."

```
config.grab_mode = CAMERA_GRAB_LATEST;
```

MOTOR CONTROL LOGIC

- **PWM Implementation:** Uses ESP32 ledc peripherals.
- **Frequency:** 1000 Hz (Standard for DC Motors).
- **Resolution:** 8-bit (0-255 values).



```
void Motor_Drive(Motor_t m, int speed) {
  if (speed > 0) {
    ledcWrite(m.pin_FWD, speed);
    ledcWrite(m.pin_REV, 0);
  }
}
```

DIFFERENTIAL & STEERING

The system supports independent control of drive and steering motors.

- **Drive Motors:** Controlled by variable speed (slider * joystick input).
- **Steering:** Uses max torque (255) to ensure wheels turn fully against friction.

CONCLUSION

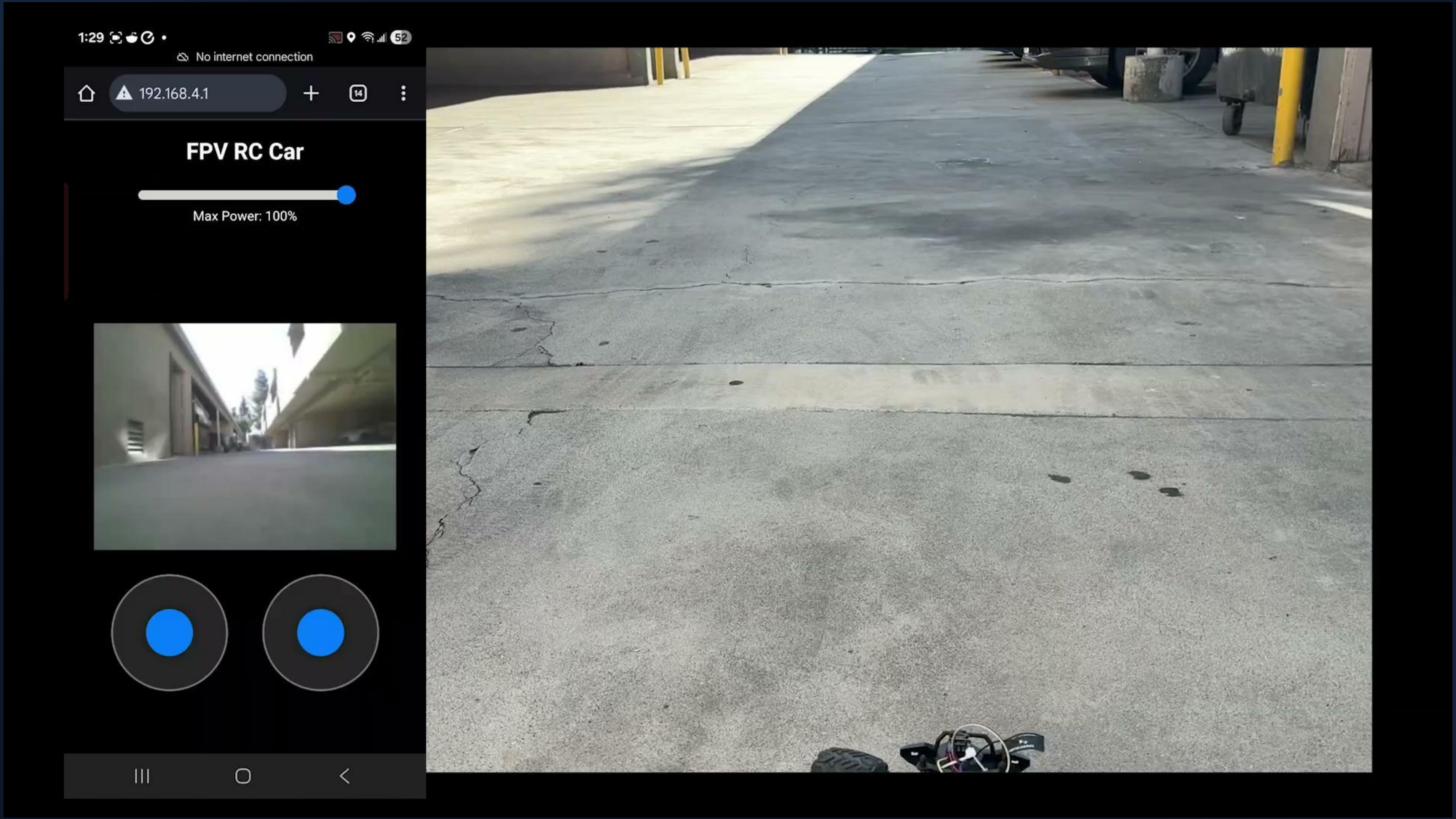
The project successfully demonstrates a functional FPV RC car. By leveraging the dual-core capabilities of the ESP32S3 and optimizing the video buffer pipeline, we achieved a responsive, low-latency control system suitable for real-time navigation.

✓ WIFI AP STABLE

LOW LATENCY STREAM

SMOOTH CONTROL

Demo



1:29

No internet connection

192.168.4.1

FPV RC Car

Max Power: 100%

QUESTIONS?

Thank you for your attention.