Miniproject FPV Remote Control Tank with WiFi

จัดทำโดย

นายชวาทิก	ธรฤทธิ์	6610110066	Section 2
นายภัทรพล	กิจเจริญ	6610110230	Section 2
นายศรัณย์กร	ชัยสุนทรานนท์	6610110289	Section 2

ระดับชั้นอุดมศึกษาปีที่ 3

เสนอ

รศ.ดร. ทวีศักดิ์ เรืองพีระกุล

รศ.ดร. ปัญญยศ ไชยกาฬ

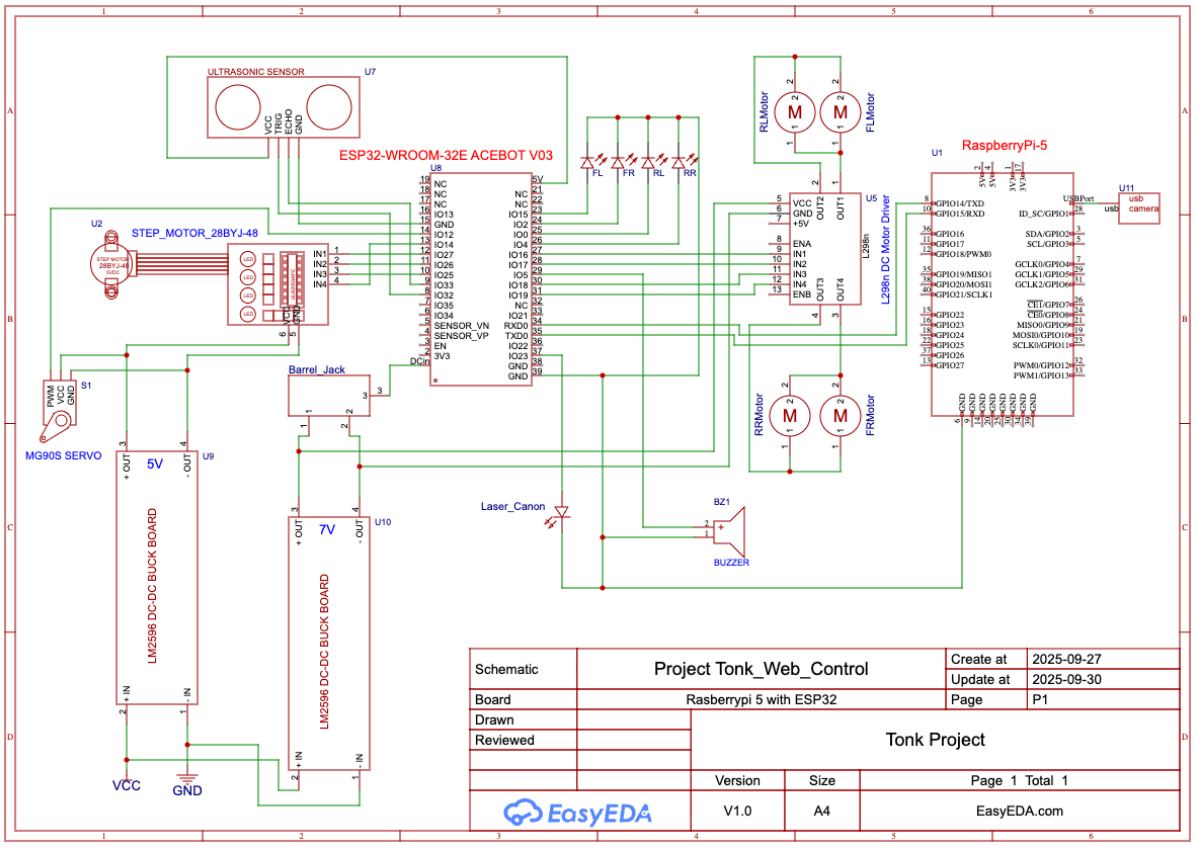
ผศ.ดร. วชรินทร์ แก้วอภิชัย

รายงานฉบับนี้เป็นส่วนหนึ่งของรายวิชา Embedded System

รหัสวิชา 240-319

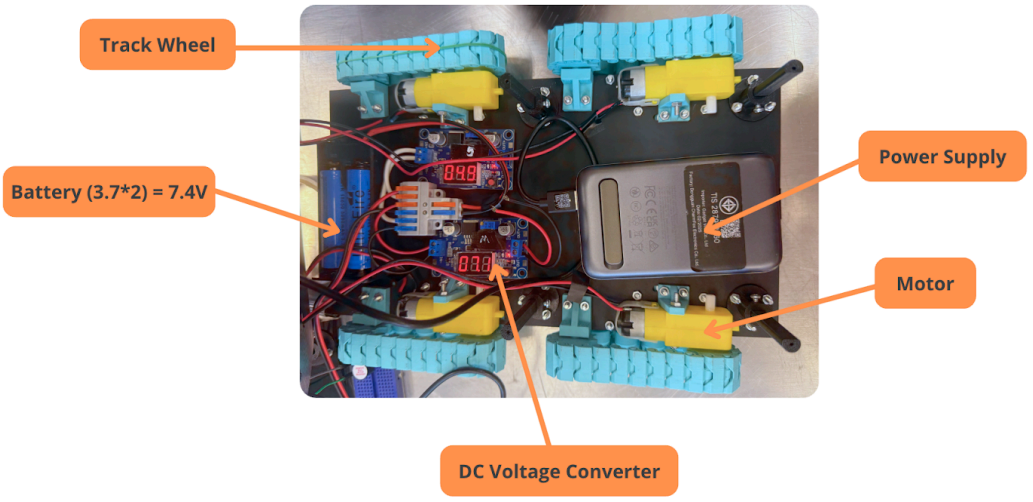
ภาคเรียนที่ 2 ปีการศึกษา 2568

1. รูป Schematic diagram ของการเชื่อมต่อวงจร

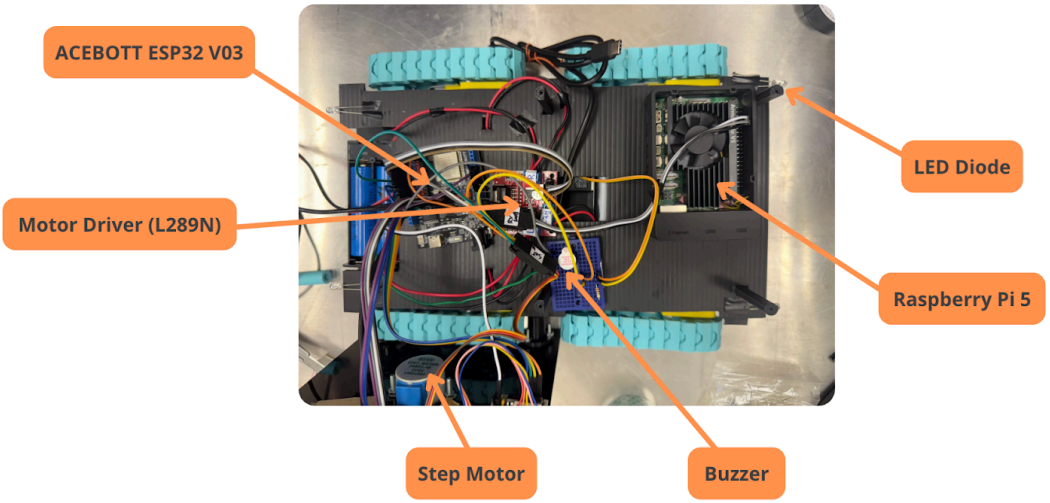


2. รูปการต่อวงจรจริง

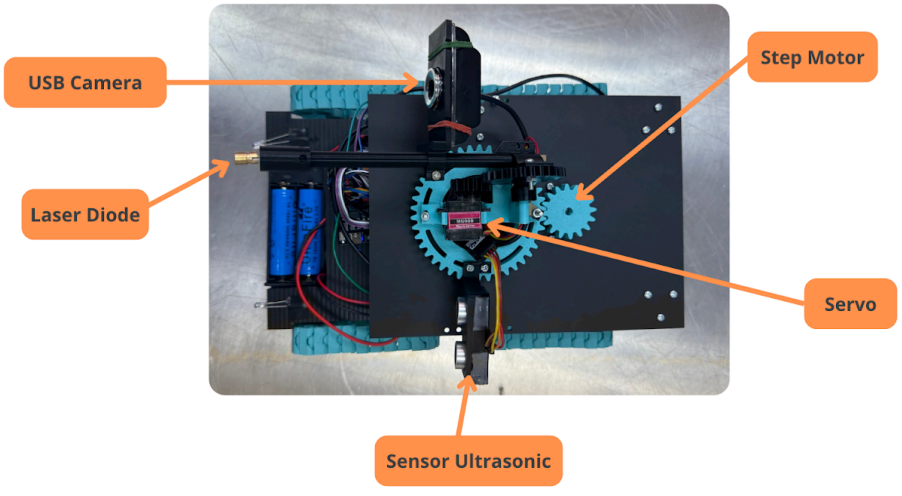
รูปผลงานที่สมบูรณ์



ชั้นที่ 1



ชั้นที่ 2



ชั้นที่ 3

3. อธิบายการทำงานของโปรแกรมอย่างละเอียด

web.py :

- **ส่วนการตั้งค่า :** ทำการกำหนดค่าต่างๆที่ต้องใช้ เช่น port 8080, resolution 640x480, framerate 24, baudrate 115200, serial port '/dev/ttyAMA0' (สำหรับ ESP32 บน Pi) ฯลฯ
- **การสตรีมวิดีโอ :** ใช้ Class StreamingOutput() ทำหน้าที่เป็น Buffer กลาง ที่เก็บข้อมูลล่าสุดจากกล้อง ไว้ในตัวแปร frame และใช้ฟังก์ชัน stream_camera() เพื่อทำการวนอ่านค่าจากกล้องด้วยคำสั่ง camera.read() จากนั้นทำการแปลงสีด้วยคำสั่ง cv2.cvtColor ต่อมาใช้ฟังก์ชัน PIL.Image.fromarray() เพื่อสร้างวัตถุภาพขึ้นมา แล้วจึงบันทึกข้อมูลของภาพที่บีบอัดแล้วลงใน Output Buffer ด้วยคำสั่ง output.write(buffer.getvalue()) และใช้ time.sleep(1.0 /FRAMERATE) เพื่อควบคุมให้มีการถ่ายภาพด้วยอัตราเฟรมที่ต้องการ (24 FPS)
- **Serial Communication :** จะมี 2 ฟังก์ชันหลักๆ คือ read_serial_data_thread() เป็นเธรดเพื่อทำหน้าที่อ่านค่าจาก Serial โดยจะใช้ ser.in_waiting > 0 เพื่อตรวจสอบว่ามีข้อมูลใหม่เข้ามาจาก Arduino หรือไม่ และใช้ ser.readline().decode('utf-8').strip() เพื่ออ่านข้อมูลเป็นบรรทัด (จนกว่าจะเจอ \n) และทำการตรวจสอบ ว่าขึ้นต้นด้วยคำว่า "Dist:" หรือไม่ ถ้าใช่ก็จะทำการจะแยกค่าตัวเลขออกมา และเก็บไว้ในตัวแปร Global ที่ชื่อว่า ultrasonic_distance และฟังก์ชัน parse_tank_command(path: str) เป็นเธรดเพื่อทำหน้าที่ถอดรหัสคำสั่ง ที่มาจาก URL (HTTP GET) โดยจะรับ URL Path ที่มี Query String (เช่น /tank_command?cmd=FR:5;LR:0;...) แล้วใช้ urllib.parse.parse_qs() เพื่อแยกพารามิเตอร์ cmd ออกมา จากนั้นทำการแยกสตริงคำสั่งย่อยด้วยตัวแบ่ง ; และ : เพื่อให้ได้คู่ Key:Value (เช่น FR เป็น 5, LR เป็น 0) แล้วจึงคืนค่าเป็น Dictionary ที่มีค่าคำสั่งควบคุมเป็นตัวเลข (Integers)
- **Web Server Handler (Class StreamingHandler) :** คลาสนี้จะเป็นคลาสที่จัดการคำขอทั้งหมดที่มาจากเว็บเบราว์เซอร์ผ่านเมธอด do_GET() ตัวอย่าง เช่น
    - **/index.html :** ทำหน้าที่แสดงหน้าควบคุมหลัก โดยใช้คำสั่ง with open('index_fixed.html', 'rb') as f: อ่านเนื้อหาไฟล์ และส่งกลับไปด้วยคำสั่ง self.wfile.write(content)
    - **/gunshot.mp3 :** ทำหน้าที่เปิดไฟล์เสียง (ในที่นี้ คือเสียงปืน) ด้วยคำสั่ง with open(self.path[1:], 'rb') as f: แล้วส่งกลับเนื้อหาไฟล์ด้วย Header Content-Type: audio/mpeg
    - **/tank_command?cmd=... :** ทำหน้าที่รับ และส่งคำสั่งควบคุมไปยังESP32 โดยใช้คำสั่ง command_data = parse_tank_command(self.path) เพื่อแปลงพารามิเตอร์เป็น Dictionary, command_str = f"FR:{...};LR:{...};..." เพื่อจัดรูปแบบสตริงและคำสั่ง ser.write(message_to_send.encode('utf-8')) ส่งผ่าน Serial ไปยัง ESP32
    - **/stream.mjpg :** ทำหน้าที่ส่งข้อมูลวิดีโอแบบ MJPEG (Motion JPEG) ด้วยคำสั่ง self.send_header('Content-Type', 'multipart/x-mixed-replace; boundary=FRAME') โดยใช้ output.condition.wait() เพื่อรอเฟรมล่าสุด และใช้ self.wfile.write(frame) ส่งข้อมูลภาพ JPEG ออกไปเป็นส่วนๆ

- **/get_distance :** ทำหน้าที่แสดงค่าระยะทาง Ultrasonic ด้วยคำสั่ง self.wfile.write(str(ultrasonic_distance).encode('utf-8')) เพื่อส่งค่า Global Variable ที่ได้รับจากเธรด Serial Reader

- **/get_time :** ทำหน้าที่แสดงเวลาปัจจุบัน (ไทย) โดยกำหนด now_thailand = datetime.now(THAILAND_TIMEZONE), time_str = now_thailand.strftime("%H:%M:%S") เพื่อส่งสตริงเวลาที่จัดรูปแบบแล้ว

- **Initialization and Cleanup :** ฟังก์ชั่นเหล่านี้ มีขึ้นเพื่อเริ่มต้น และจบการทำงานโปรแกรมได้อย่างราบรื่น เช่น

  - **main() :** เป็นฟังก์ชันเริ่มต้นของโปรแกรม โดยจะทำการตั้งค่า Signal Handlers (cleanup_gpio) เพื่อให้โปรแกรมปิดตัวได้อย่างปลอดภัยเมื่อถูกขัดจังหวะ เช่น Ctrl+C เปิด Serial Port ด้วยคำสั่ง ser = serial.Serial() (ถ้าเปิดสำเร็จ จะเริ่ม serial_reader_thread ทันที) จากนั้นทำการใช้ cv2.VideoCapture(CAMERA_INDEX) เพื่อเปิดใช้งานกล้อง และตั้งค่าความละเอียด แล้วจึงใช้คำสั่ง ThreadedHTTPServer เพื่อเริ่มทำงานบนเธรดแยก และใช้ streaming_thread เพื่อดึงภาพจากกล้อง และอัปเดต Buffer

  - **cleanup_gpio() :** เป็นฟังก์ชันนี้ที่จะทำงานเมื่อโปรแกรมปิดตัวลง (เช่น ผู้ใช้กด Ctrl+C) โดยมีหน้าที่ปิดการเชื่อมต่อ Serial โดยใช้ ser.close() เพื่อเคลียร์สัญญาณ GPIO และปิดเธรดอื่นๆ ที่ทำงานอยู่ และออกจากโปรแกรมอย่างปลอดภัย

index.html :

- เป็นส่วนหน้าจอผู้ใช้งาน (User Interface) ที่ทำงานบนเว็บเบราว์เซอร์ ทำหน้าที่หลักในการแสดงผลวิดีโอ, แสดงสถานะ, และรับ Input จากผู้ใช้เพื่อแปลงเป็นคำสั่ง HTTP แล้วส่งไปยัง web.py

- JavaScript ที่ฝังอยู่ใน index.html มีขึ้นเพื่อจัดการการสื่อสารกับ Server ผ่าน Endpoint ที่กำหนดไว้ใน web.py เช่น

  - **Video Display :** document.getElementById('videoFeed').src = "/stream.mjpg"; โดยไฟล์ web.py จะตอบกลับคำขอ /stream.mjpg ด้วยข้อมูลภาพ JPEG ต่อเนื่อง ทำให้เบราว์เซอร์แสดงเป็นภาพวิดีโอ
    - FR : ควบคุม Forward, Reverse
    - LR : ควบคุม Left, Right
    - UD : ควบคุมป้อมปืน Up, Down
    - TLR : ควบคุมป้อมปืน Left, Right
    - FC : ควบคุมการยิง
    - LC : ควบคุมการเปิด/ปิดไฟรถ

  - **Fetching Status Data :** ในโค้ด JavaScript จะใช้ฟังก์ชัน setInterval() เพื่อเรียก Endpoint ข้อมูลสถานะเป็นระยะ (50 ms) ตัวอย่างเช่น
    - /get_distance : เพื่อดึงข้อมูลระยะทาง Ultrasonic
    - /get_time : เพื่อดึงข้อมูลเวลาปัจจุบัน

mainESP32.ino :

- **Includes :** ทำการ Import Library ที่จำเป็นในการทำงาน เช่น Stepper.h และESP32Servo.h

- **Global Variables :** เพิ่มตัวแปรต่างๆที่ใช้สำหรับการควบคุมรถ เช่น forwardReverse (FR), leftRight (LR), upDown (UD), turretLeftRight(TLR), fireCannon (FC) เป็นต้น

- **Pin Configuration :** ทำการกำหนด Pin ควบคุมต่างๆ ดังนี้
    - ควบคุม Motors : IN1, IN2, IN3, IN4 ใช้ Pin 16, 17, 18, 19 ตามลำดับ
    - ควบคุม Step Motor : IN1, IN2, IN3, IN4 ใช้ Pin 14, 27, 26, 25 ตามลำดับ
    - ควบคุม Servo : ใช้ Pin 12
    - รับค่า Sensor Ultrasonic : TRIG Pin 32, ECHO Pin 33
    - ควบคุมการยิงเลเซอร์ : ใช้ Pin 23
    - ควบคุมเสียงการยิง (Buzzer) : ใช้ Pin 5
    - ควบคุมไฟ LED หน้า, หลังรถ : ใช้ Pin 0, 2 ตามลำดับ

- **setup() :**
    - Serial.begin(115200) : เริ่มการสื่อสาร Serial ด้วย Baud Rate ที่ตรงกับ web.py
    - pinMode(..., OUTPUT/INPUT) : กำหนดขาต่างๆ เป็น Input/Output
    - myStepper.setSpeed(turnSpeed) : ตั้งความเร็วพื้นฐานของ Stepper Motor
    - servo1.attach(servoPin) : เชื่อมต่อวัตถุ Servo กับขาที่กำหนด

- **setMotorSpeed(int motorNum, int speed) :**
    - ใช้เพื่อควบคุมความเร็วของรถ โดยจะควบคุมแต่ละมอเตอร์ผ่านคำสั่ง analogWrite() โดยใช้ค่าตั้งแต่ -255 ถึง 255

- **updateMotors(int forwardReverse, int leftRight) :**
    - รับค่า Throttle และ Steer มา จากนั้นใช้ constrain() เพื่อจำกัดค่า Input ให้อยู่ในช่วง –7 ถึง 7 แล้วใช้ map() เพื่อแปลงช่วงค่า Input (–7 ถึง 7) ไปเป็นช่วงค่า PWM ที่มอเตอร์ต้องการ (–255 ถึง 255) และใช้หลักการ "Mixing" (leftSpeed = mappedThrottle + mappedSteer;) เพื่อคำนวณความเร็วที่เหมาะสมสำหรับมอเตอร์แต่ละข้าง ทำให้รถถังสามารถเลี้ยวขณะเคลื่อนที่ไปข้างหน้าหรือถอยหลังได้

- **checkAndFireCannon() :** ทำหน้าที่ควบคุมอุปกรณ์เลเซอร์ และ Buzzer เพื่อจำลองการยิง
    - จะทำงานเมื่อตัวแปร fireCannon ถูกตั้งค่าเป็น 1
    - มีการใช้ตัวแปร lastFired และ reloadDuration (500 ms) เพื่อจัดการ Cool-down Time (เวลาหน่วงในการยิงซ้ำ)
    - เมื่อสั่งยิง จะทำการสั่ง digitalWrite(laserPin, HIGH) และ digitalWrite(buzzerPin, HIGH) เป็นเวลาสั้นๆ เพื่อให้เกิดแสงและเสียง แล้วจึงสั่งปิด (LOW)

- **Ultrasonic Sensor :**
    - microsecondsToCentimeters() : ทำหน้าที่คำนวณระยะทางจากเวลาที่คลื่นเสียงเดินทางไปกลับ
    - readDistance() : ควบคุมขา Trig เพื่อส่งคลื่นเสียง และใช้ pulseIn เพื่อวัดเวลาที่คลื่นสะท้อนกลับมาที่ขา Echo

- **checkLight() :**
    - ทำการตรวจสอบค่า lightControl ถ้าเป็น 1 (HIGH) จะสั่งเปิดไฟด้วย digitalWrite(..., HIGH); หรือถ้าเป็น 0 (LOW) ก็จะสั่งปิดไฟด้วยdigitalWrite(..., LOW);

- **parseCommand(String dataString) :** ทำหน้าที่ประมวลผลคำสั่งที่มาจาก Python
    - อันดับแรก ทำการรับสตริงข้อมูลทั้งหมด เช่น "FR:5;LR:0;UD:10;TLR:0;FC:0"
    - ใช้ indexOf(';') และ indexOf(':') เพื่อ แยกส่วน คำสั่งออกเป็น Key (เช่น "FR") และ Value (เช่น "5")
    - ใช้ key.equalsIgnoreCase("...") เพื่อตรวจสอบชื่อคำสั่ง
    - ใช้ valueStr.toInt() เพื่อ แปลงค่าที่เป็น String ให้เป็น Integer และนำไปเก็บในตัวแปร Global ที่เกี่ยวข้อง (forwardReverse, leftRight, ฯลฯ)

- **loop() :**

    - **ตรวจสอบ Serial Input :**
        - if (Serial.available()) : หากมีข้อมูลเข้ามา จะเรียกฟังก์ชัน Serial.readStringUntil('\n') เพื่ออ่านคำสั่งทั้งหมด
        - เรียก parseCommand(commandString) เพื่ออัปเดตตัวแปรควบคุมทั้งหมด

    - **ควบคุมการขับเคลื่อนและป้อมปืน :**
        - DC Motors : เรียกใช้ updateMotors(forwardReverse, leftRight) เพื่อสั่งการมอเตอร์ในการเคลื่อนที่ของรถ
        - Servo (Tilt) : ตรวจสอบค่า upDown เพื่อเพิ่มหรือลด องศาของ servoDeg และใช้ servo1.write(servoDeg) โดยมี การจำกัดองศา ระหว่าง 0 ถึง 180 องศา
        - Stepper (Pan) : ตรวจสอบค่า turretLeftRight และสั่ง myStepper.step(stepsToTake) หรือ myStepper.step(-stepsToTake) เพื่อหมุนป้อมปืนไปทางซ้าย หรือขวา

    - **ควบคุมการยิง :**
        - เรียก checkAndFireCannon() เพื่อจัดการการยิง และการหน่วงเวลา

    - **ส่งข้อมูลเซนเซอร์ Ultrasonic :**
        - ใช้คำสั่ง long distance = readDistance(); เพื่อเก็บค่าระยะที่ได้
        - แล้วส่งค่าระยะทางกลับไปยัง web.py ในรูปแบบ Serial.print("Dist:"); Serial.println(distance); ซึ่งจะไปแยกใน web.py เพื่อนำไปใช้แสดงผล

    - **ควบคุมสถานะการเปิดปิดไฟหน้า, หลังรถ :**
        - เรียกใช้ checkLight(); เพื่อควบคุมสถานะของไฟ

อุปกรณ์ที่ใช้

- Board ACEBOTT ESP32 x 1
- Raspberry Pi 5 (8 GB) x 1
- Powerbank (10000 mAh) x 1
- Lithium-ion Batteries (Series 7.2 V (3.2V*2)) x 2
- DC Motors 5 V x 4
- USB Camera x 1
- DC-DC Step Down Converter x 2
- Motor Driver L289N x 1
- Step Motor 28BYJ-48 x 1
- Servo Motor (90 Degree) x 1
- Ultrasonic Sensor x 1
- Laser Diode 5 V x 1
- LED Diode (FL, FR, RL, RR) x 4
- Buzzer x 1

3D printed part

- Track และจุดยึด Chasis ต่างๆ (Polymaker PLA Lite pro) x 34
- เกียร์ทด สำหรับหมุนป้อม (Bambu Lab PETG HF) x 1
- Frame ป้อมปืน (Esun PLA+) x 4

Acrylic sheet / Future board

- อคริลิค สีดำ ขนาด 3mm 60x40 cm x 2
- Future Board สีดำ ขนาด 29.7x42 cm x 1

Screws

- Screws M3 x 87
- Nut Screws M3 x 75

Main Function

ในโปรเจกต์นี้ จะใช้ไฟล์ index.html เป็นหน้าเว็บไซต์ที่สร้างขึ้นเพื่อเป็น ส่วนติดต่อผู้ใช้ (User Interface) โดยแสดงภาพ Live Video มีส่วนแสดงสถานะ (เช่น เวลา, ความหน่วง, ระยะห่าง ฯลฯ), ปุ่มควบคุมเสมือน (Joystick) และรองรับการควบคุมจาก PC และ Mobile ซึ่งจะถูกส่งเป็นคำสั่งไปยัง Python Web Server ที่รันอยู่บน Raspberry Pi จากนั้นทำการส่งข้อมูลผ่าน Protocal: UART โดยใช้สาย RX TX (Serial Communication) เพื่อไปสั่งการบอร์ด ESP32 ให้ทำงานตามที่กำหนด

Control/Functionality

- **Website :** ทำหน้าที่แสดงผลภาพ Live Video ค่าสถานะต่างๆ แบบเรียลไทม์ และสามารถคำสั่ง เช่น การควบคุมทิศทางการเคลื่อนที่ผ่าน Joystick, WASD เพื่อควบคุมรถถัง หรือการกดยิงกระสุนพลังงาน Photon ผ่าน การกด Click, กด Spacebar
- **Live Video** : สามารถแสดงภาพจากกล้องของรถถังได้แบบเรียลไทม์
- **Virtual Controller :** เป็น Joystick จำลอง 2 อันบนหน้าเว็บ อันแรกใช้สำหรับควบคุมการเคลื่อนที่ของรถถัง อีกอันใช้สำหรับควบคุมการหมุนของป้อมปืน และปุ่ม FIRE เพื่อยิงกระสุนพลังงาน Photon
- **Keyboard Input :** เมื่อเข้าเว็บไซต์บน PC จะสามารถควบคุมรถถัง ผ่านคีย์บอร์ดได้ โดย WASD จะใช้สำหรับเดินหน้า และถอยหลัง ส่วน IJMK ใช้สำหรับการหมุนป้อมปืน และการคลิกซ้าย หรือกดปุ่ม Spacebar จะเป็นการยิงกระสุนพลังงาน Photon
- **Mobile Input :** สามารถควบคุมการทำงานผ่านหน้าเว็บบนมือถือได้ โดยใช้ Virtual Controller
- **Controlled via Internet :** การทำงานของเว็บไซต์ จะทำงานอยู่บน Internet ทำให้สามารถควบคุมได้จากทุกที่ ที่มีสัญญาณ Internet

Data Transfer

- **UART :** มีการส่งข้อมูลจาก Raspberry Pi ไปยัง ESP32 ผ่าน Protocal : UART โดยใช้ขาสัญญาณ RX TX และ ใช้ GND เดียวกันร่วมกัน
- **HTTP :** เป็นโปรโตคอลที่ใช้ในการสื่อสารระหว่างเว็บ กับเซิร์ฟเวอร์ เพื่อจัดการการแสดงผลหน้าเว็บ HTML และใช้สำหรับ ส่งคำสั่งควบคุมจากเว็บไปยัง Raspberry Pi ซึ่งทำหน้าที่เป็น Server
- **HTML** (ร่วมกับ CSS) **:** ใช้สร้าง UI บนเว็บ ส่วน JavaScript ในเบราว์เซอร์รับ Input จากผู้ใช้แล้วส่งคำขอ HTTP (GET/POST ฯลฯ) ไปยัง Raspberry Pi ที่เปิด API ไว้เพื่อตอบสนองคำสั่งควบคุมอุปกรณ์
- **Global Private Network :** เป็นบริการที่สร้าง เครือข่ายส่วนตัวเสมือน (VPN) ข้ามอินเทอร์เน็ตสาธารณะ ทำให้ Raspberry Pi และอุปกรณ์ควบคุมต่างๆ สามารถติดต่อสื่อสารกันได้เสมือนอยู่ในเครือข่าย LAN เดียวกัน แม้จะอยู่คนละสถานที่ (ในที่นี้ ใช้บริการของ Zerotier)