หุ่นยนต์เลียนแบบการก้าวเดินของมนุษย์

Humanoid Walking Robot

จักรพงษ์ เครือจันต๊ะ
Chakkaphong Khrueachanta

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงงาน2
ภาควิชาวิศวกรรมอิเล็กทรอนิกส์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ.2565

หุ่นยนต์เลียนแบบการก้าวเดินของมนุษย์

Humanoid Walking Robot

โดย

จักรพงษ์ เครือจันต๊ะ

อาจารย์ที่ปรึกษา

ผศ.ดร.สุเมฆ วิศยทักษิณ

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงงาน 2

ภาควิชาวิศวกรรมอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ.2565

รายงานวิชา PROJECT 2 ปีการศึกษา 2565

ภาควิชา วิศวกรรมอิเล็กทรอนิกส์

คณะ วิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง หุ่นยนต์เลียนแบบการก้าวเดินของมนุษย์

Humanoid Walking Robot

ผู้จัดทำ นายจักรพษ์ เครือจันต๊ะ รหัสประจำตัว 62010101

รายงานนี้ผ่านการตรวจสอบโดยอาจารย์ที่ปรึกษาแล้ว

Sumet Wisat

(ผศ.ดร.สุเมฆ วิศยทักษิณ)

อาจารย์ที่ปรึกษา

หัวข้อโครงงาน หุ่นยนต์เลียนแบบการก้าวเดินของมนุษย์
นักศึกษา นายจักรพษ์ เครือจันต๊ะ รหัสประจำตัว 62010101
ปริญญา วิศวกรรมศาสตร์บัณฑิต
ภาควิชา วิศวกรรมอิเล็กทรอนิกส์
ปีการศึกษา 2565
อาจารย์ที่ปรึกษาโครงการ ผศ.ดร.สุเมฆ วิศยทักษิณ

บทคัดย่อ

โปรเจคนี้เป็นการสร้างหุ่นยนต์ที่เลียนแบบมนุษย์ โดยเริ่มจากการก้าวเดิน โดยจะมีอยู่ 2 ส่วนคือส่วนของ หุ่นยนต์และส่วนของรีโมท ที่ใช้ในการควบคุมองศาของ Servo แต่ล่ะตัว เพื่อให้เกิดท่าทางในการก้าวเดินแบบมนุษย์ ซึ่งเหตุผลที่ทำโปรเจคนี้ขึ้นมา เพราะปัจจุบันในประเทศไทย มีการแข่งขันหุ่นยนต์ ในช่วงชั้นของประถมศึกษา ถึงมัธยมศึกษาในรายการต่างๆ ทั้งระดับภาค ระดับประเทศ เพื่อคัดตัวแทนไปแข่งขันระดับนานาชาติที่ต่างประเทศ ซึ่งพอเข้าไปในการแข่งขันระดับนานาชาติแล้ว ประเภทของหุ่นยนต์ที่เป็นกฎในการแข่งขัน จะถูกปรับระดับให้ยากขึ้น ซึ่งในช่วงปัจจุบัน หุ่นยนต์ Humanoid กำลังเป็นที่นิยม และใช้เป็นกฎการแข่งขัน ในหลายๆรายการ ซึ่งในประเทศไทย ยังไม่ค่อยมีการศึกษา และสร้างหุ่นยนต์ Humanoid ขึ้นมาใช้งานเองอย่างจริงจัง โปรเจคนี้จึงเป็นการเริ่มศึกษา เพื่อยกระดับความสามารถของหุ่นยนต์ ที่นักเรียนไทยสร้างขึ้นได้ และสามารถแข่งขันกับนานาชาติได้อย่างภาคภูมิ

Project Title	Humanoid Walking Robot
Student	Mr. Chakkaphong Khrueachanta Student ID 62010101
Degree	Bachelor of Engineering
Program	Electronics Engineering
Year	2022
Project Advisor	Asst. Prof. Sumek Wisayataksin, Ph.D

ABSTRACT

This project is about creating a robot that imitates humans, starting with walking. There will be two parts: the part of the robots and the remote parts used to control the angle of each servo to achieve a human walking posture. The reason for this project is that nowadays, in Thailand, there is a robot competition for elementary and secondary schools in various programs at the regional and national levels. The purpose is to select representatives to compete in international competitions abroad, which are challenging enough to qualify for the international competition. The types of robots that rule the competition will be leveled up to be more difficult. Currently, humanoid robots are becoming popular and are used as the rules of the competition in many items, which have not yet been studied in Thailand. Creating a humanoid robot for serious use, therefore, begins to study to enhance the abilities of the robot created by Thai students and to compete internationally with pride.

กิตติกรรมประกาศ

โครงงานครั้งนี้สามารถสำเร็จลุล่วงได้ด้วยดี โดยได้รับความกรุณาจากผศ.ดร.สุเมฆ วิศยทักษิณ และรุ่นพี่ที่ให้การปรึกษาความรู้เกี่ยวกับเทคนิคการเขียนโค้ด คำแนะนำในการทำโครงงานแต่ละขั้นตอน การจัดหาอุปกรณ์ รวมทั้งให้คำแนะนำปรึกษาในการแก้ไขปัญหาที่เกิดขึ้นระหว่างการทำงานมาโดยตลอดจนโครงงานนี้เสร็จสมบูรณ์ ผู้จัดทำโครงงานขอกราบขอบพระคุณเป็นอย่างสูงสุด

จักรพงษ์ เครือจันต๊ะ

สารบัญ

สารบัญรูปภาพ

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

ปัจจุบันในประเทศไทย มีการแข่งขันหุ่นยนต์ ในช่วงชั้นของประถมศึกษา ถึงมัธยมศึกษาในรายการต่างๆ ทั้งระดับภาค ระดับประเทศ เพื่อคัดตัวแทนไปแข่งขันระดับนานาชาติที่ต่างประเทศ ซึ่งพอเข้าไปในการแข่งขันระดับนานาชาติแล้ว ประเภทของหุ่นยนต์ที่เป็นกฎในการแข่งขัน จะถูกปรับระดับให้ยากขึ้น ซึ่งในช่วงปัจจุบัน หุ่นยนต์ Humanoid กำลังเป็นที่นิยม และใช้เป็นกฎการแข่งขัน ในหลายๆรายการ ซึ่งในประเทศไทย ยังไม่ค่อยมีการศึกษา และสร้างหุ่นยนต์ Humanoid ขึ้นมาใช้งานเองอย่างจริงจัง

1.2 วัตถุประสงค์

1.เพื่อศึกษาการออกแบบ หุ่นยนต์ Humanoid
2.เพื่อศึกษาการใช้ใช้งานโมดูลหลายๆตัวให้ทำงานพร้อมกัน
3.เพื่อศึกษาการเขียน Code สั่งให้โมดูลทำงาน
4.เพื่อเก็บข้อมูล และนำไปพัฒนาต่อในอนาคต

1.3 ขอบเขต

1.หุ่นยนต์ Humanoid ที่สามารถก้าวเดินได้

1.4 ระยะเวลา

ตั้งแต่วันที่ 9 มกราคม 2566 ถึง 31 มีนาคม 2566

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.ทักษะการออกแบบที่เพิ่มขึ้น
2.ทักษะการเขียนโปรแกรมที่สับซ้อนมากยิ่งขึ้น
3.การใช้โปรแกรมการออกแบบวงจร

บทที่ 2
เอกสารเกี่ยวข้อง

2.1 บอร์ด Esp32

รูปที่ 1 Esp32 38 pin

ESP32 เป็นชื่อของไอซีไมโครคอนโทรลเลอร์ที่รองรับการเชื่อมต่อ WiFi และ Bluetooth 4.2 BLE ในตัว ผลิตโดยบริษัท Espressif จากประเทศจีน โดยราคา ณ ที่เขียนบทความอยู่นี้ มีราคาไม่เกิน 500 บาท (บอร์ดพัฒนาสำเร็จรูป) โดยตัวไอซี ESP32 มีสเปคโดยละเอียด ดังนี้

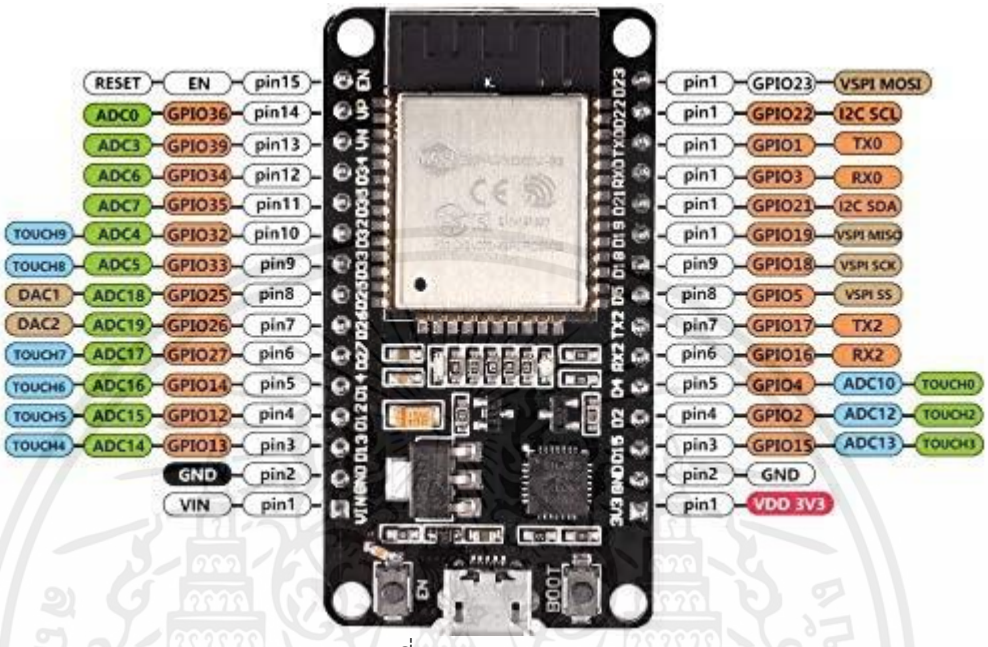
- ซีพียูใช้สถาปัตยกรรม Tensilica LX6 แบบ 2 แกนสมอง สัญญาณนาฬิกา 240MHz
- มีแรมในตัว 512KB
- รองรับการเชื่อมต่อรอมภายนอกสูงสุด 16MB
- มาพร้อมกับ WiFi มาตรฐาน 802.11 b/g/n รองรับการใช้งานทั้งในโหมด Station softAP และ Wi-Fi direct
- มีบลูทูธในตัว รองรับการใช้งานในโหมด 2.0 และโหมด 4.0 BLE
- ใช้แรงดันไฟฟ้าในการทำงาน 2.6V ถึง 3V
- ทำงานได้ที่อุณหภูมิ -40°C ถึง 125°C
-

นอกจากนี้ ESP32 ยังมีเซ็นเซอร์ต่าง ๆ มาในตัวด้วย ดังนี้

- วงจรกรองสัญญาณรบกวนในวงจรขยายสัญญาณ
- เซ็นเซอร์แม่เหล็ก

- เซ็นเซอร์สัมผัส (Capacitive touch) รองรับ 10 ช่อง
- รองรับการเชื่อมต่อคลิสตอล 32.768kHz สำหรับใช้กับส่วนวงจรนับเวลาโดยเฉพาะ

ขาใช้งานต่าง ๆ ของ ESP32 รองรับการเชื่อมต่อบัสต่าง ๆ ดังนี้



รูปที่ 2 Pinout esp32

- มี GPIO จำนวน 32 ช่อง
- รองรับ UART จำนวน 3 ช่อง
- รองรับ SPI จำนวน 3 ช่อง
- รองรับ I2C จำนวน 2 ช่อง
- รองรับ ADC จำนวน 12 ช่อง
- รองรับ DAC จำนวน 2 ช่อง
- รองรับ I2S จำนวน 2 ช่อง
- รองรับ PWM / Timer ทุกช่อง
- รองรับการเชื่อมต่อกับ SD-Card

นอกจากนี้ ESP32 ยังรองรับฟังก์ชั่นเกี่ยวกับความปลอดภัยต่าง ๆ ดังนี้

- รองรับการเข้ารหัส WiFi แบบ WEP และ WPA/WPA2 PSK/Enterprise
- มีวงจรเข้ารหัส AES / SHA2 / Elliptical Curve Cryptography / RSA-4096 ในตัว
- ในด้านประสิทธิ์ภาพการใช้งาน ตัว ESP32 สามารถทำงานได้ดี โดยรับ – ส่ง ข้อมูลได้ความเร็วสูงสุดที่ 150Mbps เมื่อเชื่อมต่อแบบ 11n HT40 ได้ความเร็วสูงสุด 72Mbps เมื่อเชื่อมต่อแบบ

11n HT20 ได้ความเร็วสูงสุดที่ 54Mbps เมื่อเชื่อมต่อแบบ 11g และได้ความเร็วสูงสุดที่ 11Mbps เมื่อเชื่อมต่อแบบ 11b

- เมื่อใช้การเชื่อมต่อผ่านโปรโตคอล UDP จะสามารถรับ – ส่งข้อมูลได้ที่ความเร็ว 135Mbps ในโหมด Sleep ใช้กระแสไฟฟ้าเพียง 2.5uA

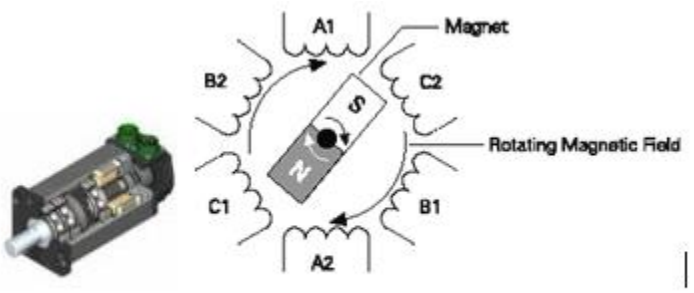
2.2 Servo Motor

รูปที่ 3 Servo motor

เซอร์โวมอเตอร์ (Servo Motor) เป็นมอเตอร์ที่มีการควบคุมการเคลื่อนที่ของมัน (State) ไม่ว่าจะเป็นระยะ ความเร็ว มุมการหมุน โดยใช้การควบคุมแบบป้อนกลับ (Feedback control) เป็นอุปกรณ์ที่สามารถควบคุมเครื่องจักรกล หรือระบบการทำงานนั้นๆ ให้เป็นไปตามความต้องการ เช่น ควบคุมความเร็ว (Speed), ควบคุมแรงบิด (Torque), ควบคุมแรงตำแหน่ง (Position), ระยะทางในการเคลื่อนที่(หมุน) (Position Control) ของตัวมอเตอร์ได้ ซึ่งมอเตอร์ทั่วไปไม่สามารถควบคุมในลักษณะงานเบื้องต้นได้ โดยให้ผลลัพธ์ตามความต้องการที่มีความแม่นยำสูง

2.2.1 หลักการทำงานของเซอร์โวมอเตอร์

การทำงานของเซอร์โวมอเตอร์ชนิดนี้จะคล้ายกับการทำงานของซิงโครนัสมอเตอร์ 3 เฟส กล่าวคือเมื่อมีการควบคุมให้คอนโทรลเลอร์จ่ายกระแสไฟฟ้าเข้าไปยังขดลวดที่สเตเตอร์ แกนเหล็กของสเตเตอร์จะกลายเป็นแม่เหล็กไฟฟ้า และหมุนเคลื่อนที่ด้วยความเร็วที่แปรผันตามความถี่ ซึ่งเรียกว่า ความเร็วซิงโครนัส (synchronous speed) หรือความเร็วสนามแม่เหล็กหมุน และจะดูดให้โรเตอร์ซึ่งเป็นแม่เหล็กถาวรหมุนเคลื่อนที่ตาม

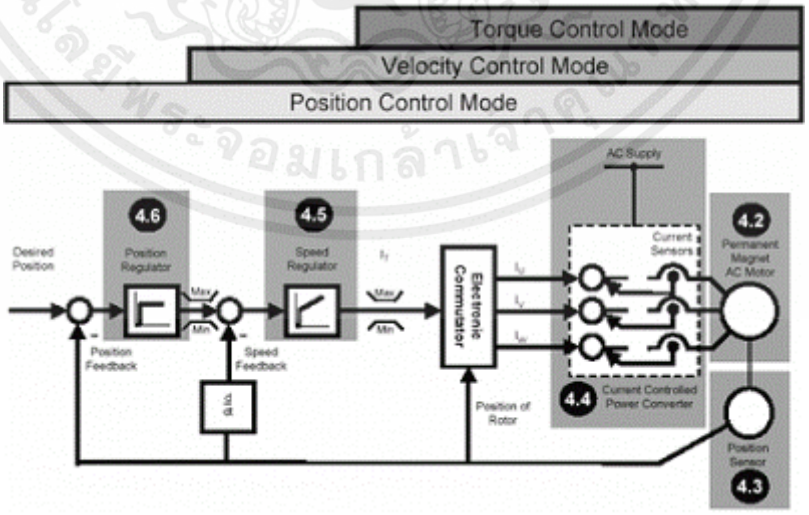


รูปที่ 4 หลักการทำงานของเซอร์โวมอเตอร์

จากลักษณะโครงสร้างของโรเตอร์และหลักการทำงานที่เหมือนกับซิงโครนัสมอเตอร์ซึ่งเป็นมอเตอร์แบบเอซี แต่ไม่มีแปรงถ่าน (Brushless) ไม่มีซี่คอมมิวเตอรเตอร์ จึงทำให้มอเตอร์ชนิดนี้มีชื่อเรียกขานแตกต่างกันออกไป เช่น เรียกทับศัพท์ว่า Permanent Magnet Synchronous Motor(PMSM) ซึ่งหมายถึงซิงโครนัสมอเตอร์ที่ไม่มีแปรงถ่าน บ้างก็เรียกว่าเอซีเซอร์โวมอเตอร์ (AC Servo motor) หรือบ้างก็เรียกสั้นๆย่อๆว่า AC Brushless หรือ Brushless Motor เป็นต้น

2.2.2 โครงสร้างของระบบควบคุมเซอร์โวมอเตอร์

ลักษณะของระบบควบคุมเซอร์โวมอเตอร์จะเป็นระบบควบคุมแบบลูปปิด(Closed loop control) ซึ่งประกอบด้วย 3 โหมดการควบคุมคือ โหมดการควบคุมแรงบิด (Torque Control Mode) ซึ่งอยู่วงรอบหรือลูปในสุด โหมดการควบคุมอัตราเร่ง(Velocity Control Mode) และโหมดการควบคุมตำแหน่ง(Position Control Mode) ซึ่งอยู่ลูปด้านนอกสุด โดยมีองค์ประกอบที่สำคัญๆ ดังรูป

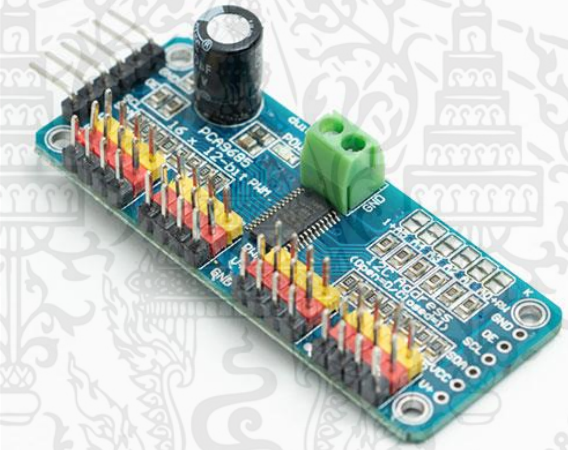


รูปที่ 5 โครงสร้างของระบบควบคุมเซอร์โวมอเตอร์

1. เซอร์โวมอเตอร์ (Servo Motor) (ตำแหน่ง 4.2)
2. ชุดควบคุมการขับเคลื่อนเซอร์โว (Servo Drive,Servo Amplifier หรือบ้างก็เรียกว่า servo controller) (ตำแหน่ง 4.4, 4.5, 4.6)
3. อุปกรณ์ป้อนกลับ (Feedback Device เช่น Speed encoder และ Position Sensor) (ตำแหน่ง 4.3)

2.3 PCA9685

ไอซี PCA9685 หรือก็คือ PWM LED controller จาก NXP Semiconductors การควบคุมการทำงานของเซอร์โวจะใช้สัญญาณ PWM ขับ เมื่อไอซี สร้างสัญญาณ PWM จะทำให้สามารถขับเซอร์โวได้



รูปที่ 6 PCA9685

2.3.1 Pinouts

- GND กราวด์ของโมดูล
- VCC ไฟเลี้ยงโมดูล ข้อระวังต้องตรงกับ Logic Level ที่ใช้ หาก Arduino ใช้ 5V หากใช้ NodeMCU ก็ 3.3V และไฟที่ขานี้ห้ามเกิน 5V
- V+ เป็นไฟเสริม คือหากใช้ขับเซอร์โว 16 ตัว ไฟ VCC จะไม่พอ จ่ายไฟภายนอกเข้าไปตรงนี้ ตัวโมดูลทำ Screw Terminal มาให้ใช้งาน ใส่ 5V จากแหล่งกระแสสูง ๆ ที่จะพอขับเซอร์โวทั้ง 16 ตัว
- SDA สายสัญญาณ SDA ของการสื่อสารแบบ I2C
- SCL สายสัญญาณ SCL ของการสื่อสารแบบ I2C

- OE คือ Output Enable เป็นการกำหนดว่าจะใช้งาน Output ทั้ง 16 ช่องหรือไม่ โดยทำงานแบบ Active LOW หากใส่สัญญาณ LOW เข้าไป Output ก็จะทำงาน แต่หากใส่ HIGH เข้าไปจะไม่มี Output ทั้งนี้ ขานี้เป็นแค่ Optional เวลาใช้งานให้ปล่อยลอยไว้.
- Output ขา Output ทั้งหมดตัวโมดูลจะถูกออกแบบมาให้วางเป็นชุด ๆ คือมี PWM ตรงกลาง เป็น V+ และ GND ตามลำดับ ซึ่งก็จะตรงตามขาของเซอร์โว นำมาเสียบได้ทันที มีทั้งหมด 16 ชุด 16 Output ตั้งแต่ 0 ถึง 15

2.3.2 I2C Address

โมดูล I2C นั้นสามารถพ่วงกันหลาย ๆ ตัวได้ โดยที่แต่ละตัวนั้นต้องมี Address ไม่ซ้ำกัน PCA9685 นั้นก็ทำได้เหมือนกัน แต่การตั้งค่า Address นั้นต้องใช้การ Hardware Setting คือที่มุมขวาบนจะมีช่อง Jumper ว่าง ๆ มา A0 ถึง A5 ตรงนี้เป็นตัวตั้ง I2C Address ซึ่งมีให้ตั้งได้ตามตาราง แต่จะสามารถ Multiple Driver ได้แค่ 62 Outputs เท่านั้น คือแค่ 4 โมดูล 0=ปล่อยว่าง 1=บัดกรีเชื่อมกัน

A5	A4	A3	A2	A1	A0	ADDR	A5	A4	A3	A2	A1	A0	ADDR	A5	A4	A3	A2	A1	A0	ADDR	A5	A4	A3	A2	A1	A0	ADDR
0	0	0	0	0	0	0x40	0	1	0	0	0	0	0x50	1	0	0	0	0	0	0x60	1	1	0	0	0	0	0x70
0	0	0	0	0	1	0x41	0	1	0	0	0	1	0x51	1	0	0	0	0	1	0x61	1	1	0	0	0	1	0x71
0	0	0	0	1	0	0x42	0	1	0	0	1	0	0x52	1	0	0	0	1	0	0x62	1	1	0	0	1	0	0x72
0	0	0	0	1	1	0x43	0	1	0	0	1	1	0x53	1	0	0	0	1	1	0x63	1	1	0	0	1	1	0x73
0	0	0	1	0	0	0x44	0	1	0	1	0	0	0x54	1	0	0	1	0	0	0x64	1	1	0	1	0	0	0x74
0	0	0	1	0	1	0x45	0	1	0	1	0	1	0x55	1	0	0	1	0	1	0x65	1	1	0	1	0	1	0x75
0	0	0	1	1	0	0x46	0	1	0	1	1	0	0x56	1	0	0	1	1	0	0x66	1	1	0	1	1	0	0x76
0	0	0	1	1	1	0x47	0	1	0	1	1	1	0x57	1	0	0	1	1	1	0x67	1	1	0	1	1	1	0x77
0	0	1	0	0	0	0x48	0	1	1	0	0	0	0x58	1	0	1	0	0	0	0x68	1	1	1	0	0	0	0x78
0	0	1	0	0	1	0x49	0	1	1	0	0	1	0x59	1	0	1	0	0	1	0x69	1	1	1	0	0	1	0x79
0	0	1	0	1	0	0x4A	0	1	1	0	1	0	0x5A	1	0	1	0	1	0	0x6A	1	1	1	0	1	0	0x7A
0	0	1	0	1	1	0x4B	0	1	1	0	1	1	0x5B	1	0	1	0	1	1	0x6B	1	1	1	0	1	1	0x7B
0	0	1	1	0	0	0x4C	0	1	1	1	0	0	0x5C	1	0	1	1	0	0	0x6C	1	1	1	1	0	0	0x7C
0	0	1	1	0	1	0x4D	0	1	1	1	0	1	0x5D	1	0	1	1	0	1	0x6D	1	1	1	1	0	1	0x7D
0	0	1	1	1	0	0x4E	0	1	1	1	1	0	0x5E	1	0	1	1	1	0	0x6E	1	1	1	1	1	0	0x7E
0	0	1	1	1	1	0x4F	0	1	1	1	1	1	0x5F	1	0	1	1	1	1	0x6F	1	1	1	1	1	1	0x7F

รูปที่ 7 I2C Address

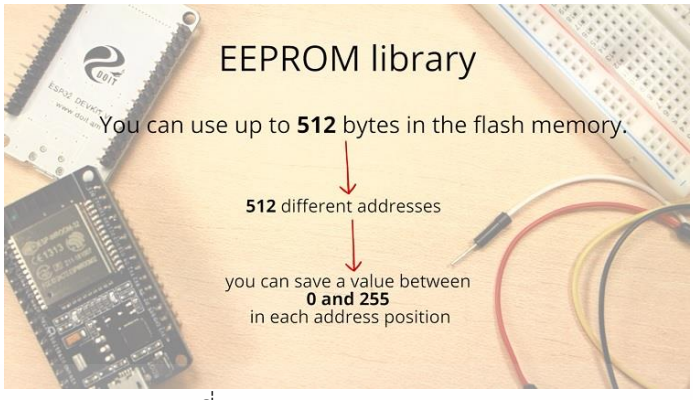
2.3.3 Library PCA9685

Adafruit_PWMServo (Adafruit Github) หรือ Adafruit_PWMServo (Mirror Fitrox GDrive)

2.4 EEPROM Esp32 library

การเขียนและอ่านค่าจากหน่วยความจำ flash memory ของ Node32S โดยใช้ Arduino IDE เราจะใช้ EEPROM Libraryของ ESP32ซึ่งการใช้งานก็จะคล้ายๆกันกับ Library ที่ใช้กับ Arduino

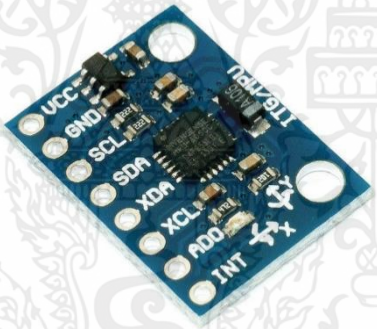
EEPROM Library ของ Node32S นั้นเราสามารถใช้การ flash memory ได้สูงสุด 512 bytesซึ่งหมายความว่าเราสามารถบันทึกค่าได้ 512 addresses ที่ค่าแตกต่างกันได้ โดยที่ค่าที่สามารถบันทึกได้มีค่าตั้งแต่ 0 – 255 ในแต่ละตำแหน่ง



รูปที่ 8 EEPROM Esp32 library

2.5 Gyro sensor MPU6050

MPU6050 เป็น 9DOF IMU คือ **9** **D**egrees **O**f **F**reedom **I**nertial **M**easurement **U**nit ซึ่งคำว่า 9 แกนนี้มาจากที่มันสามารถวัดความเร่ง (Accelerometer)ได้ 3 แกน (X, Y, Z) มีไจโรสโคป (Gyroscope) 3 แกน (X, Y, Z) และวัดความเข้มของสนามแม่เหล็ก (Magnetometer)ได้ 3 แกน (X, Y, Z) อีกทั้งยังมี Digital Motion Processor ซึ่งเป็นหัวใจสำคัญ MPU6050 เชื่อมต่อกับบอร์ดพัฒนาด้วยการสื่อสารแบบ I2C



รูปที่ 9 Gyro sensor MPU6050

2.5.1 Library Gyro sensor MPU6050

MPU6050_light ที่เป็น Library สำหรับ MPU6050 ที่ใช้งานได้ง่าย มีขนาดเล็ก และสามารถใช้งานได้ดี

2.6 Esp-NOW

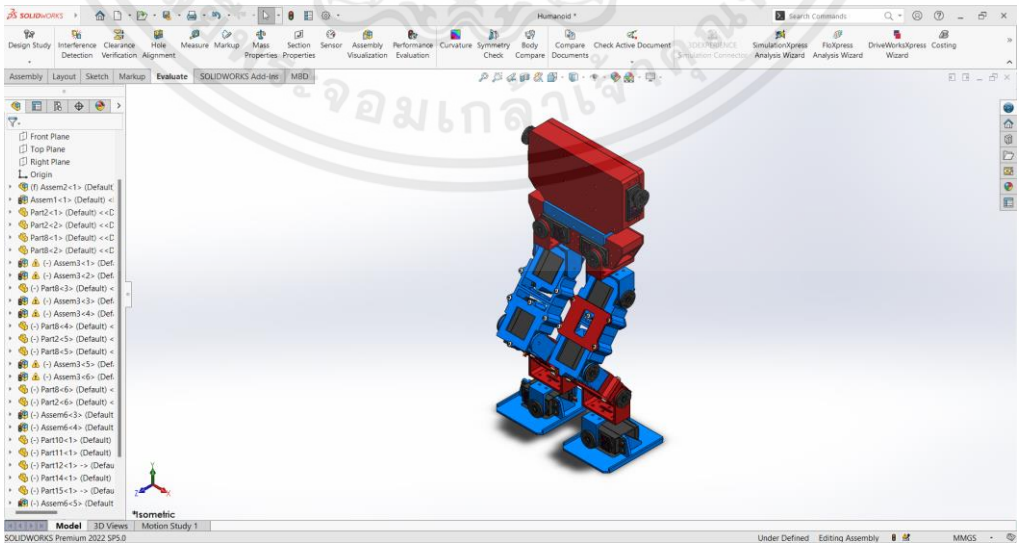
ESP-NOW คือการสื่อสารแลกเปลี่ยนข้อมูลระหว่างบอร์ด ESP32 ESP8266 สามารถเขียนโปรแกรมควบคุมได้ด้วย Arduino IDE การสื่อสารแบบ ESP-NOW เป็นโปรโตคอลที่พัฒนาโดย Espressif สำหรับส่งข้อมูลขนาดเล็กแบบประหยัดพลังงานด้วยความถี่ 2.4G ทำให้อุปกรณ์คุยกันได้โดยตรงแบบไม่ต้องผ่านตัวกลาง สามารถสื่อสารได้ทั้งแบบ อุปกรณ์และอุปกรณ์ หรือแบบหลายอุปกรณ์แบบเครือข่ายได้พร้อมกัน

รูปที่ 10 Esp-NOW

2.6.1 ความสามารถของ ESP-NOW

- สื่อสารแบบไร้สายความถี่ 2.4G แบบประหยัดพลังงาน
- สร้างเครือข่ายการรับส่งสัญญาณแบบไร้สาย
- จ่ายไฟแล้วเชื่อมต่ออัตโนมัติ ไม่ต้องตั้งค่าใหม่
- เข้า/ถอดรหัสข้อมูล สื่อสารระหว่างบอร์ด
- ส่งแพ็คเกจข้อมูลสูงสุดครั้งละ 250 ไบต์
- เชื่อมต่อได้สูงสุด 20 โหนด
- มีฟังก์ชั่นตรวจสอบสถานะการรับและส่งข้อมูล
- ระยะส่ง ESP32/ESP8266 ประมาณ 100-200 เมตร ขึ้นกับอุปกรณ์และสภาพแวดล้อม

2.7 โปรแกรม Solidwork



รูปที่ 11 โปรแกรม Solidwork

SOLIDWORKS ซอฟต์แวร์ออกแบบสำหรับงานด้านวิศวกรรม เช่น ชิ้นส่วน อุปกรณ์เครื่องจักร เป็นต้น โดย เป็นซอฟต์แวร์ในตระกูล CAD (Computer Aided Design and Drafting) ซึ่งสามารถจำลองเป็นโมเดลสามมิติได้ และมีความละเอียดกว่า CAD ทั่วไปในการวิเคราะห์ส่วนต่าง ๆ เช่น ความแข็งแรง อุณหภูมิ อายุการใช้งาน เป็นต้น

Solution ที่โดดเด่นของ SOLIDWORKS

- สามารถสร้างชิ้นส่วนที่มีความซับซ้อน ออกแบบชิ้นส่วนได้อย่างรวดเร็ว มีประสิทธิภาพและง่ายต่อการใช้งาน
- ง่ายต่อการค้นหาและใช้ประโยชน์จากข้อมูลทางด้านวิศวกรรมที่มีอยู่แล้ว เพื่อทำให้การพัฒนาผลิตภัณฑ์เป็นไปอย่างรวดเร็ว
- บันทึกชิ้นส่วนที่ใช้งานบ่อย คุณสมบัติและแม่แบบให้ง่ายต่อการเข้าถึงจากภายในโปรแกรม SOLIDWORKS
- สร้างภาพวาด 2D ที่ใช้สำหรับการผลิตเพื่อใช้ในการสื่อสารว่างานออกแบบนี้จะใช้ในการผลิตและประกอบอย่างไร
- การตรวจสอบต้นทุนการผลิตอย่างอัตโนมัติในระหว่างการออกแบบเพื่อที่จะช่วยในการควบคุมต้นทุนการผลิตได้อย่างมีประสิทธิภาพ

2.8 โปรแกรม Arduino

รูปที่ 12 โปรแกรม Arduino

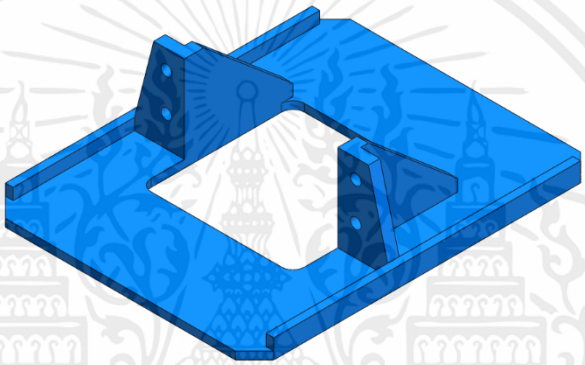
เป็นซอร์ฟแวร์ที่ใช้ในการโปรแกรมคำสั่งจากโค้ดผู้เขียนไปยังบอร์ไมโครคอนโทรลเลอร์ให้ทำตามคำสั่งโดยในตัวโปรแกรมจะมีไลบราลี ที่ทำหน้าที่เป็นการเรียกฟังก์ชันจากผู้พัฒนาฟังก์ชันเกี่ยวกับเรื่องต่างๆให้สามารถนำมาเขียนได้ง่ายขึ้น และมีตัวอย่างโค้ดในการเขียนโปรแกรม เพื่อลองนำไปใช้และประยุกต์

บทที่ 3
การดำเนินงาน

3.1 การออกแบบชิ้นส่วนหุ่นยนต์

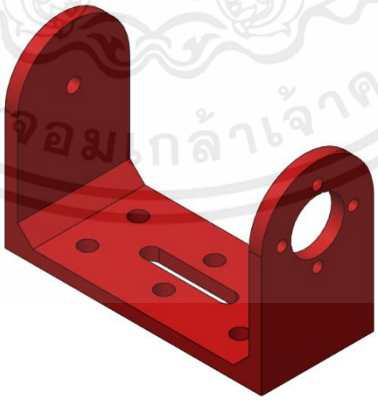
ในการออกแบบหุ่นยนต์ จะทำการสร้างโมเดล 3D ในโปรแกรม Solidwork จากนั่นจะใช้เครื่อง 3D printer ในการสร้างชิ้นงานจริง

3.1.1 โมเดล 3D ของชิ้นงาน



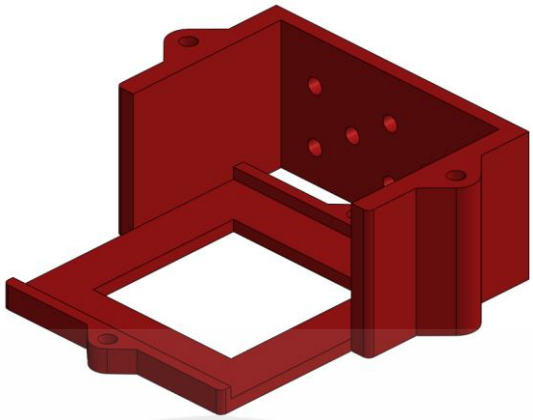
รูปที่ 13 Part 1

เป็นส่วนเท้าของหุ่นยนต์ ตรงกลางออกแบบ เพื่อให้สามารถขันน็อตยึด Servo motor กับเท้าของหุ่นยนต์ได้



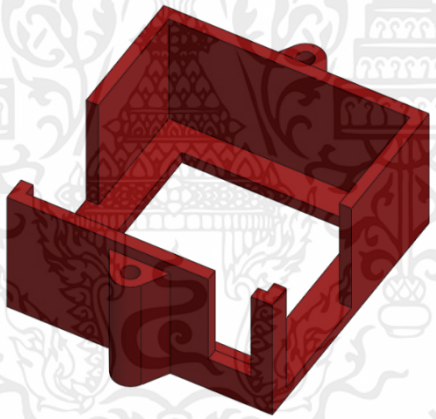
รูปที่ 14 Part 2

เป็นส่วนข้อต่อ Servo motor แต่ละ โดยจะออกแบบให้มีแขน 2 ข้าง ซึ่ง Servo motor ที่ซื้อมาจะมีตัวยึดแค่ข้างเดียว จึงได้ทำแบบเจาะรูที่ ด้านของ servo motor อีกข้างเพื่อเพิ่มความแข็งแรงในการยึดข้อต่อ

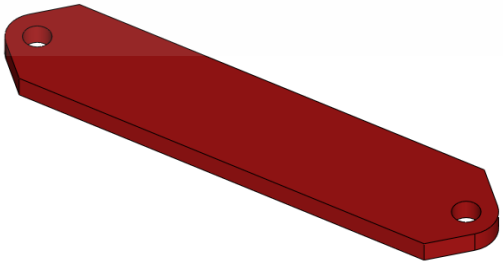


รูปที่ 15 Part 3

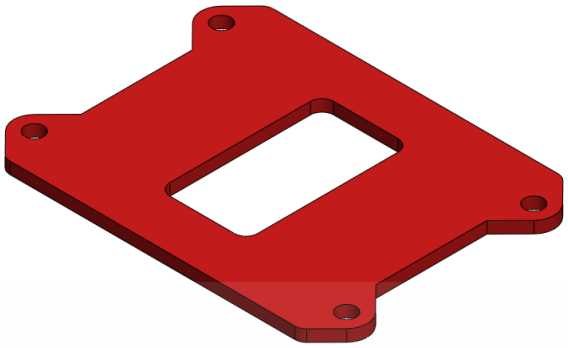
เป็นส่วนที่ออกแบบมาเพื่อยึด servo motor เป็นฝาด้านล่าง เนื่องจากตัวยึดเดิมของ Servo motor นั่น ถ้าออกจากออกแบบตัวยึด จะต้องทำเป็นขนาดที่ทำใหญ่ ซึ่งจะทำให้ขนาดโดยรวมของหุ่นยนต์ใหญ่เพิ่มขึ้น จึงได้ทำการออกแบบตัวยึดใหญ่ ให้เป็นกล่องครอบ Servo motor แบบ ผาบน และผาล่าง และมีชิ้นยาวอีกชิ้น เพื่อเสริมให้ฝาทั้งสอง แนบสนิทกันพอดี



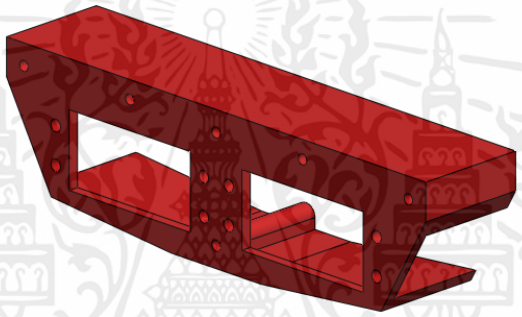
รูปที่ 16 Part 4



รูปที่ 17 Part 5

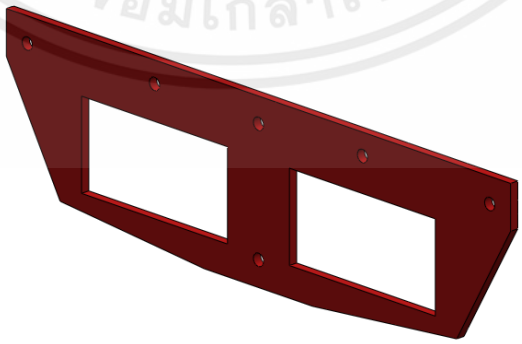


รูปที่ 18 Part 6



รูปที่ 19 Part 7

เป็นส่วนสะโพกของหุ่นยนต์ ออกให้สามารถ ใส่ servo motor 2 ตัว ที่เป็นส่วนข้อต่อของขาหนีบ จะมีส่วนคือส่วนกล่อง และส่วนฝา เพื่อให้ง่ายต่อการแก้ไข้ ให้น็อตในการยึดติดชิ้นส่วนเข้าด้วยกันออกแบบให้ท้าย servo ทะลุออก เพื่อที่จะยึดแขนแบบ 2 จุดได้

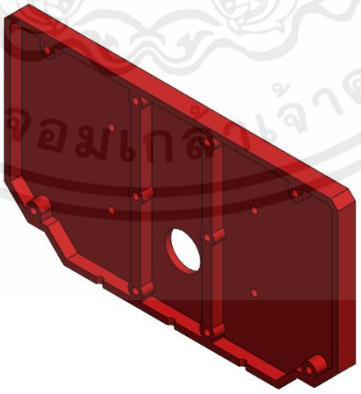


รูปที่ 20 Part 8

รูปที่ 21 Part 9

เป็นส่วนที่ยึดชิ้นส่วนระหว่างสะโพก กับช่วงตัวด้านบน ใช้การออกแบบที่ว่า ให้มีน็อตลายออกไปได้ทั้งบนและล่าง เพื่อที่ในการประกอบ จะไม่ได้ใช้กาวในการยึดติดชิ้นส่วน

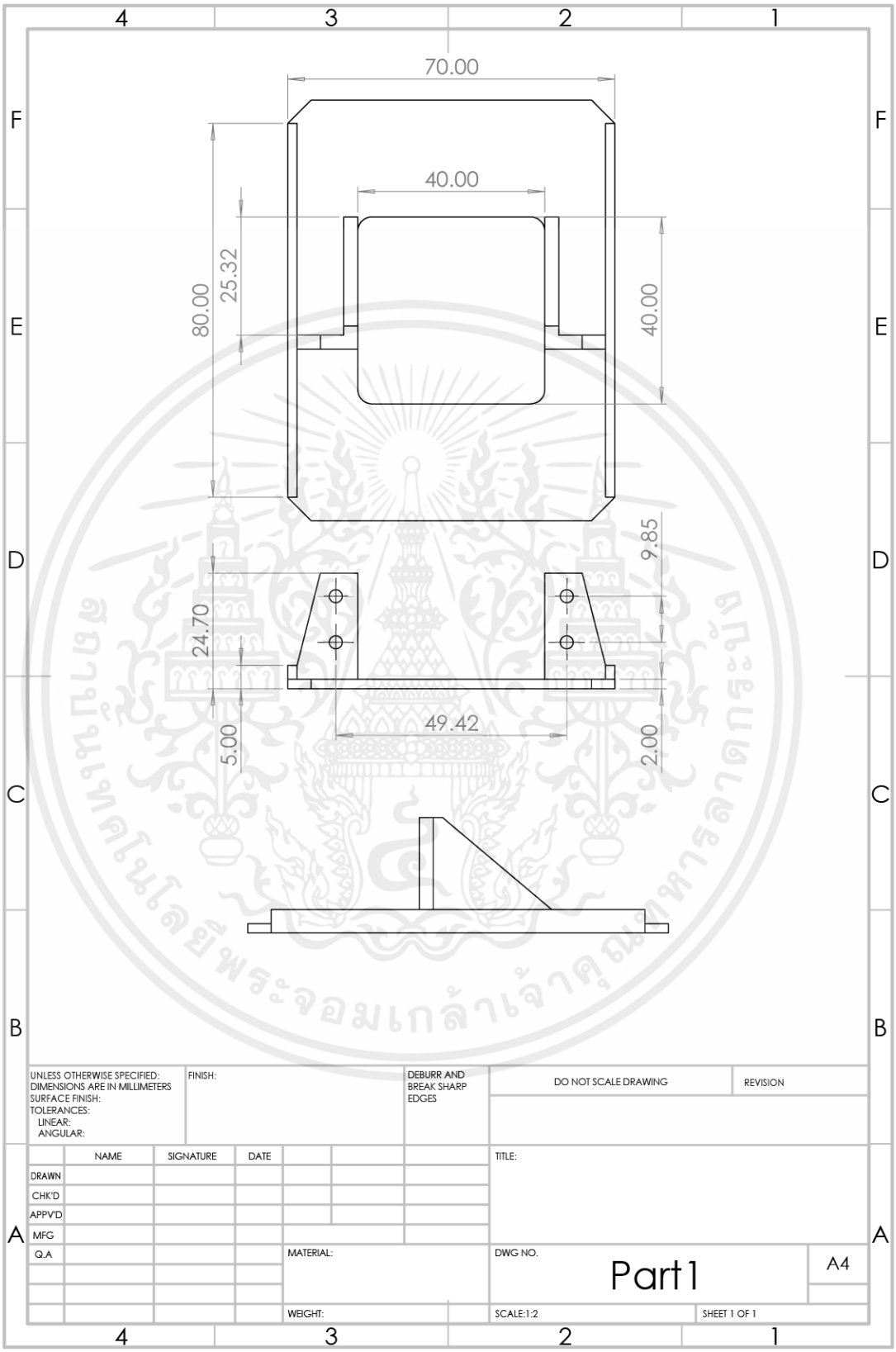
รูปที่ 22 Part 10



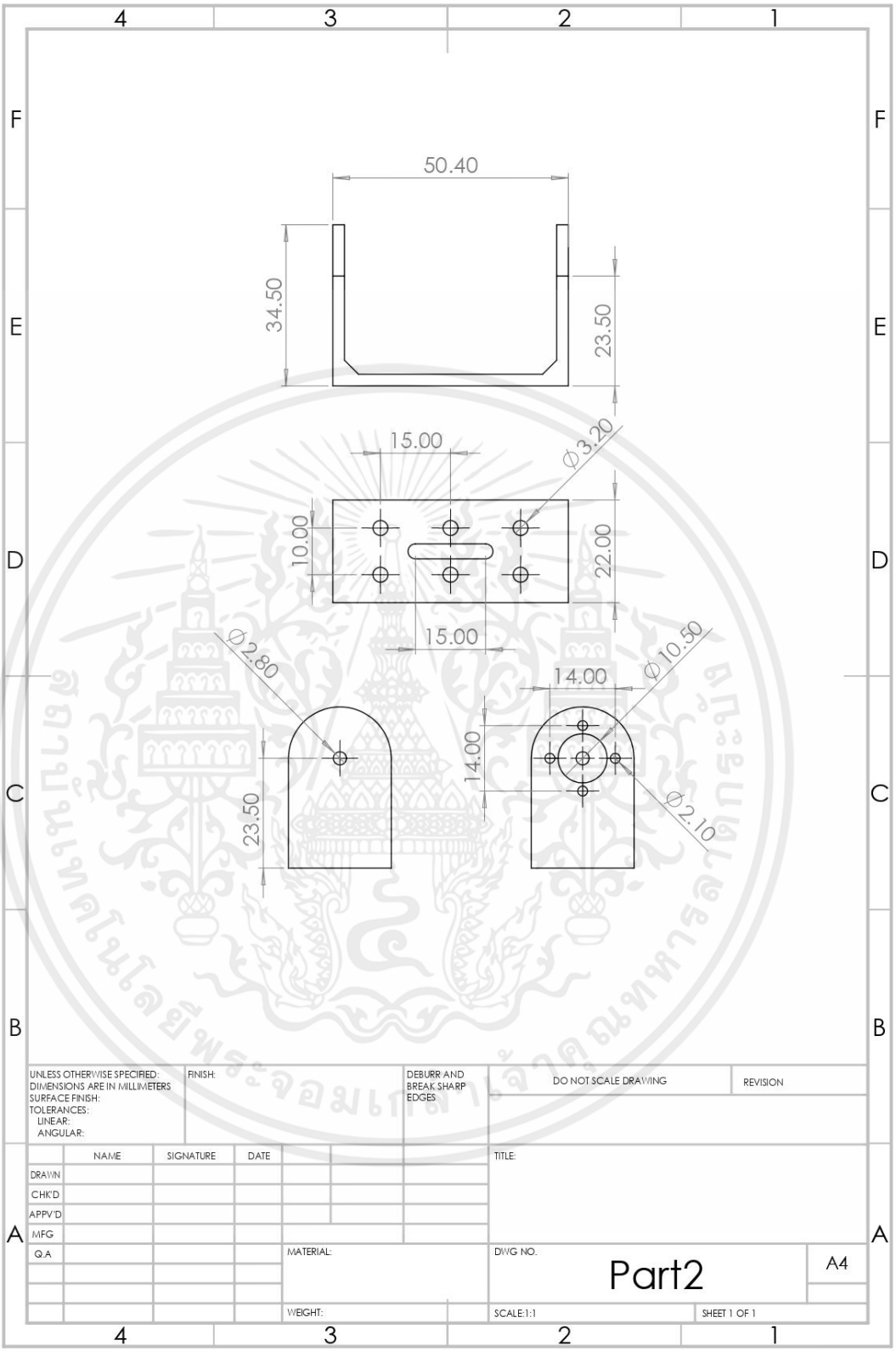
รูปที่ 23 Part 10

ส่วนลำตัวด้านบน หลักออกแบบให้หลักการเดียวกับการออกแบบ สะโพก มีสองชิ้นสองเป็นฝาปิด และใช้น็อตในการยึดติด ข้างใน ใส่ servo ได้ 2 ตัว เป็นข้อต่อของไหล ด้านหลังจะมีที่ว่าง และจุดยึด เพื่อใช้ในการยึดบอร์ด

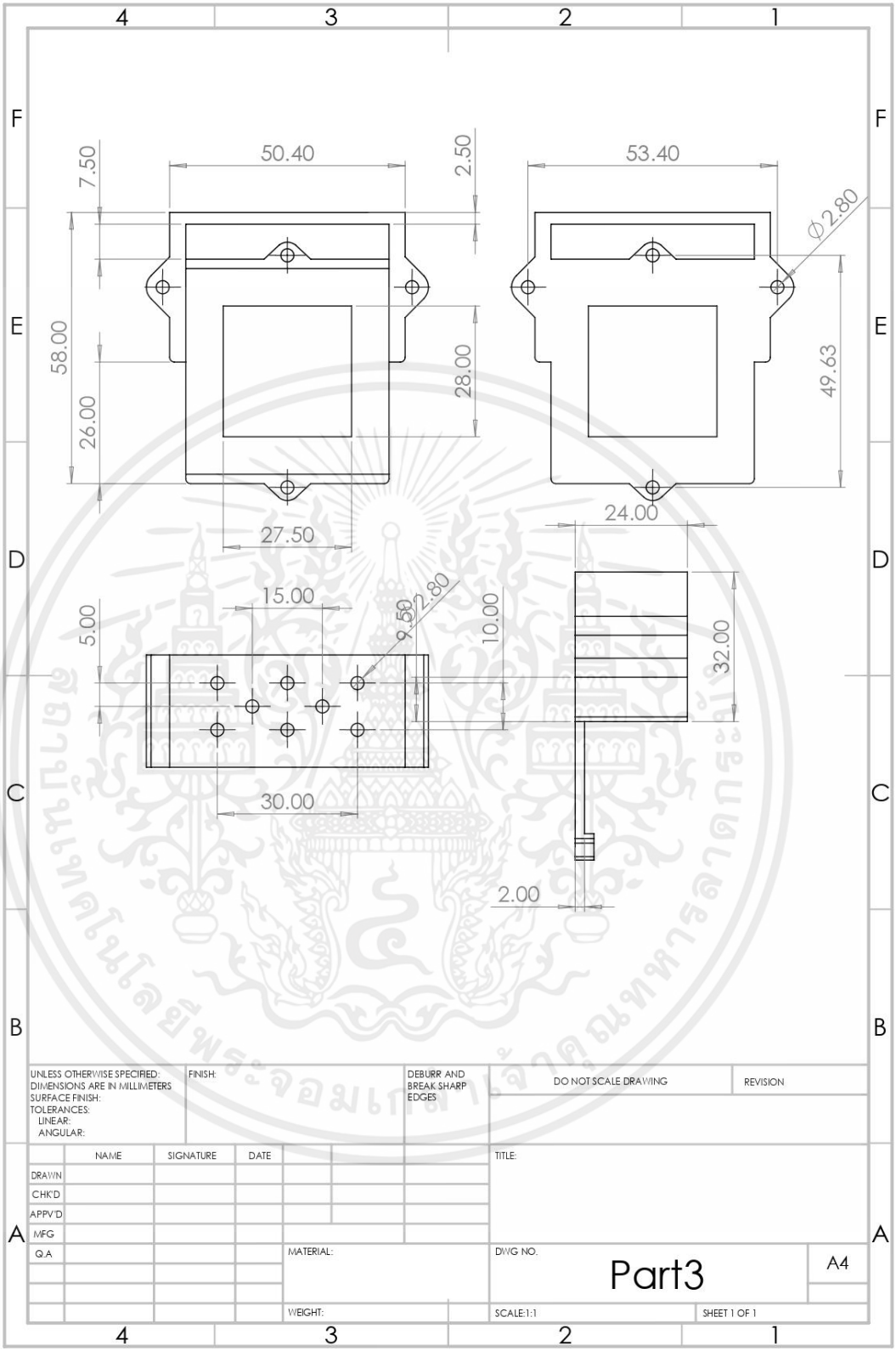
3.1.2 Drawing ของชิ้นงาน



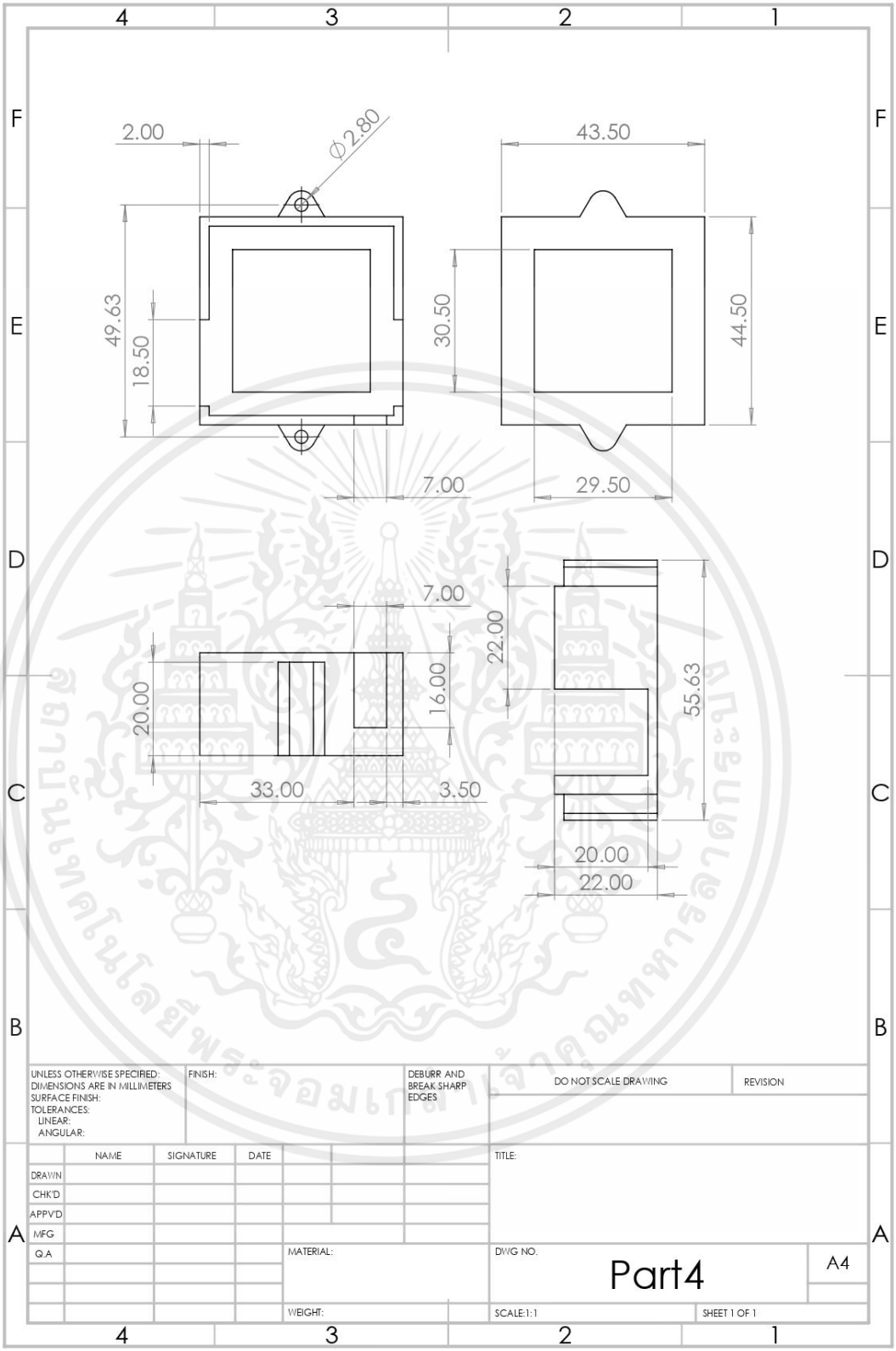
รูปที่ 24 Drawing part 1



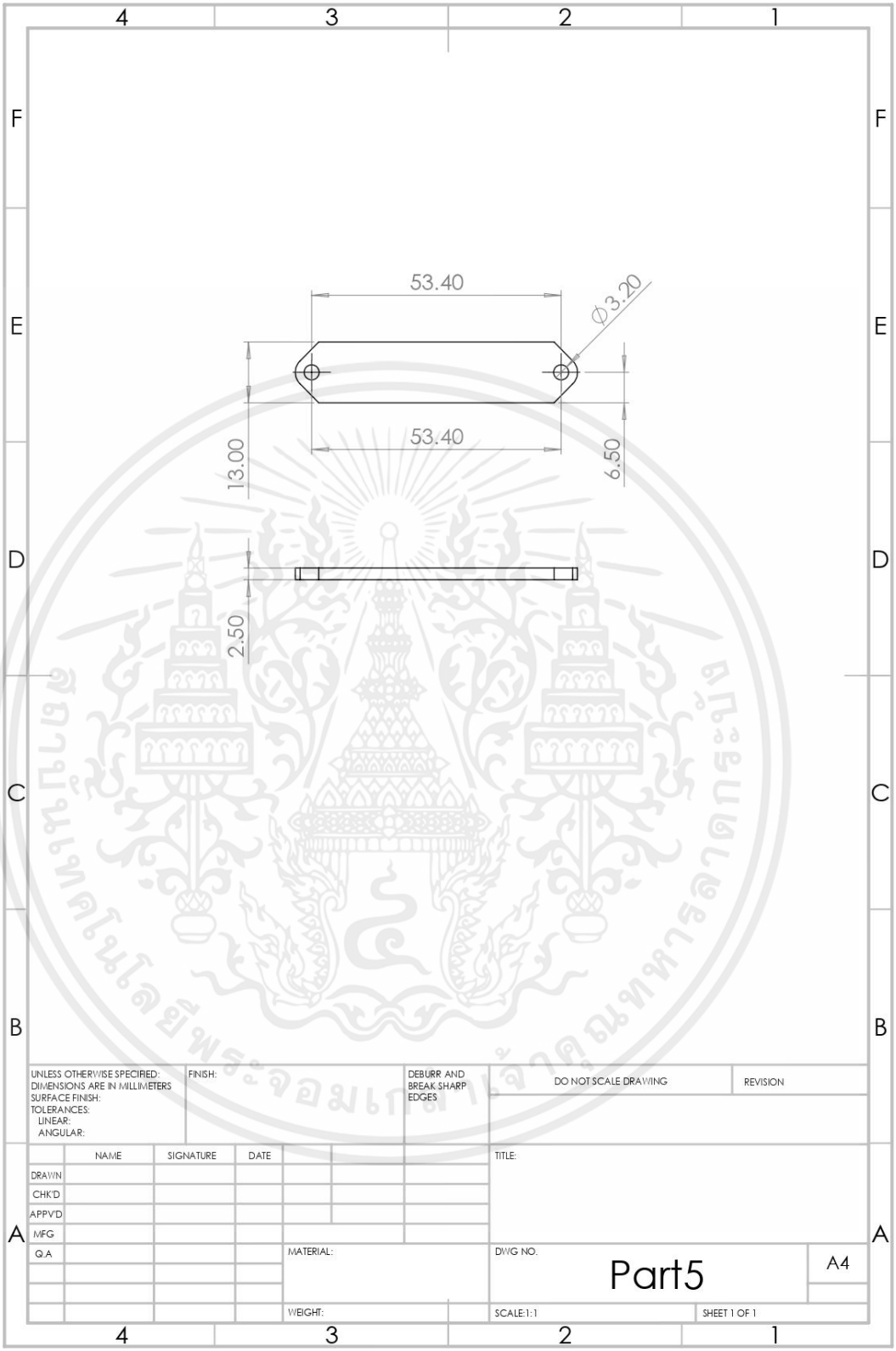
รูปที่ 25 Drawing part 2



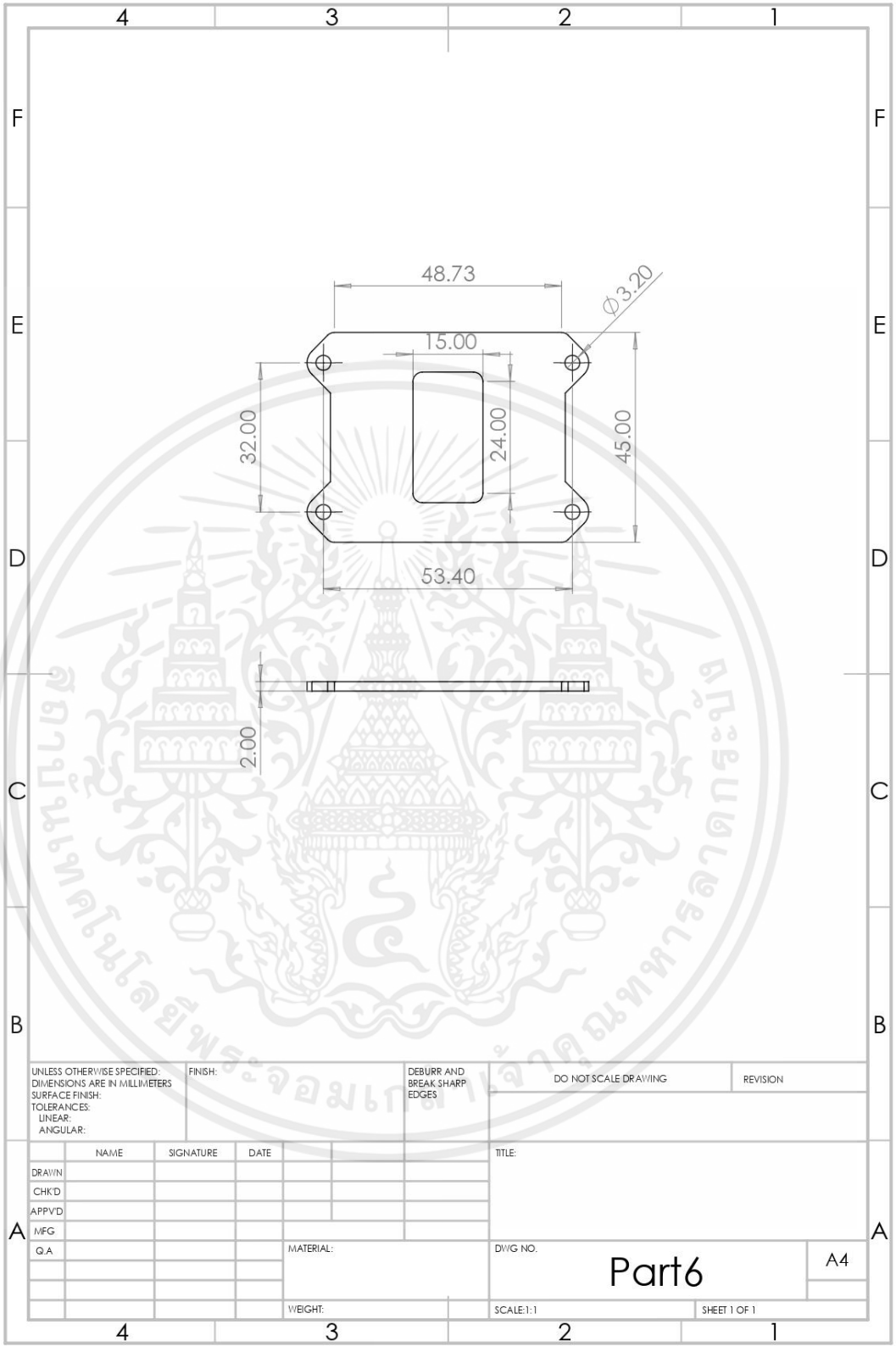
รูปที่ 26 Drawing part 3



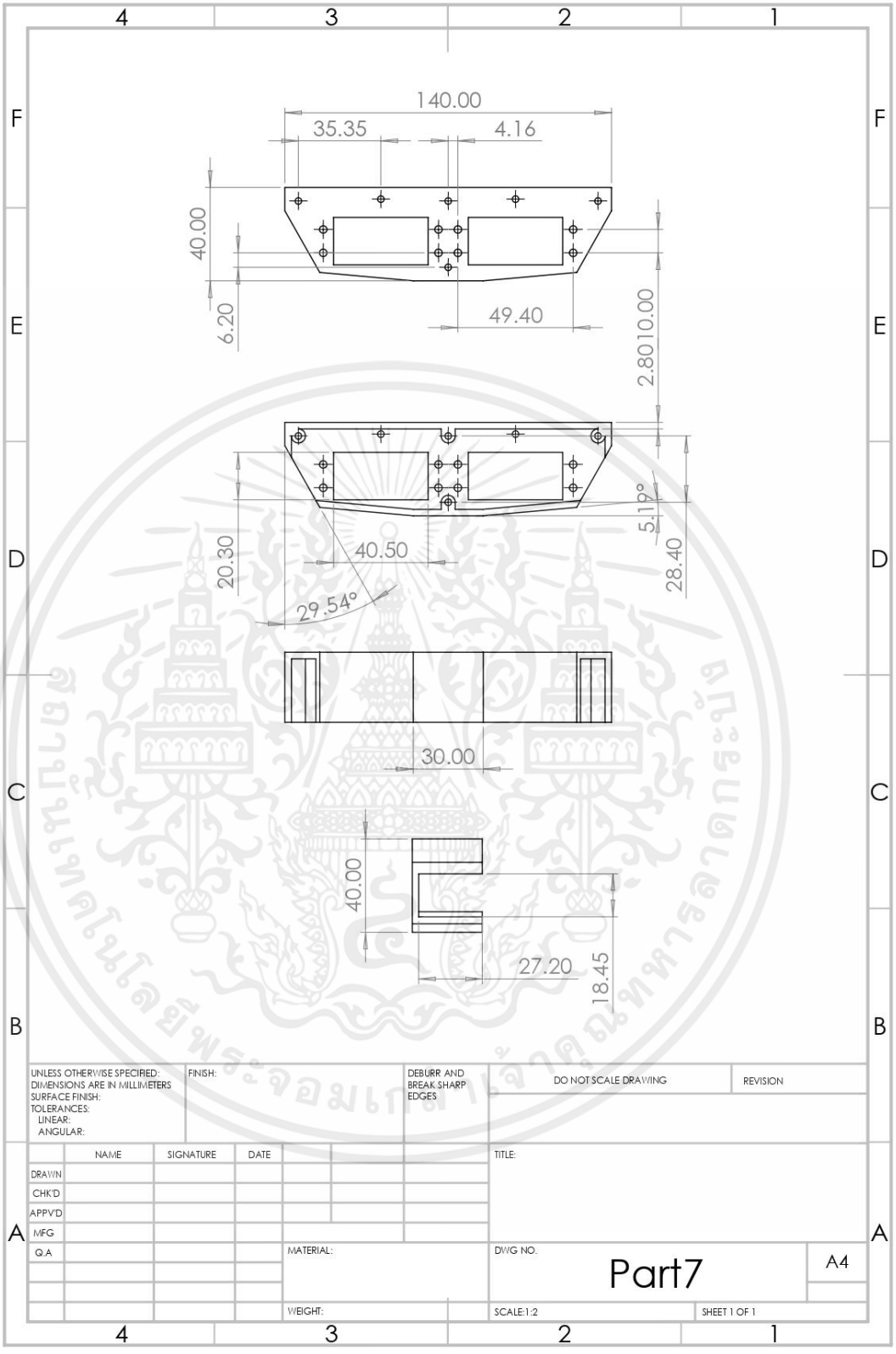
รูปที่ 27 Drawing part 4



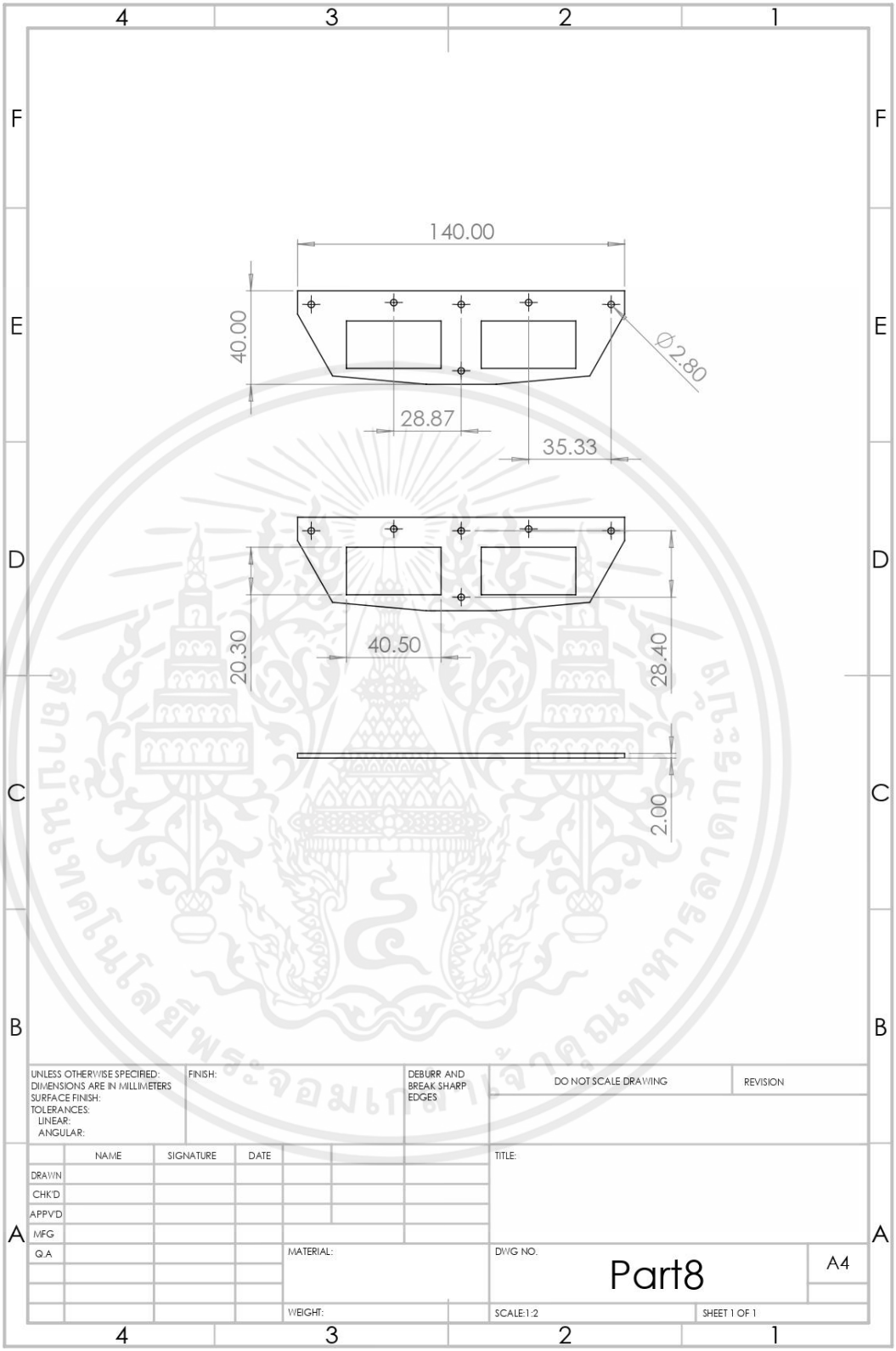
รูปที่ 28 Drawing part 5



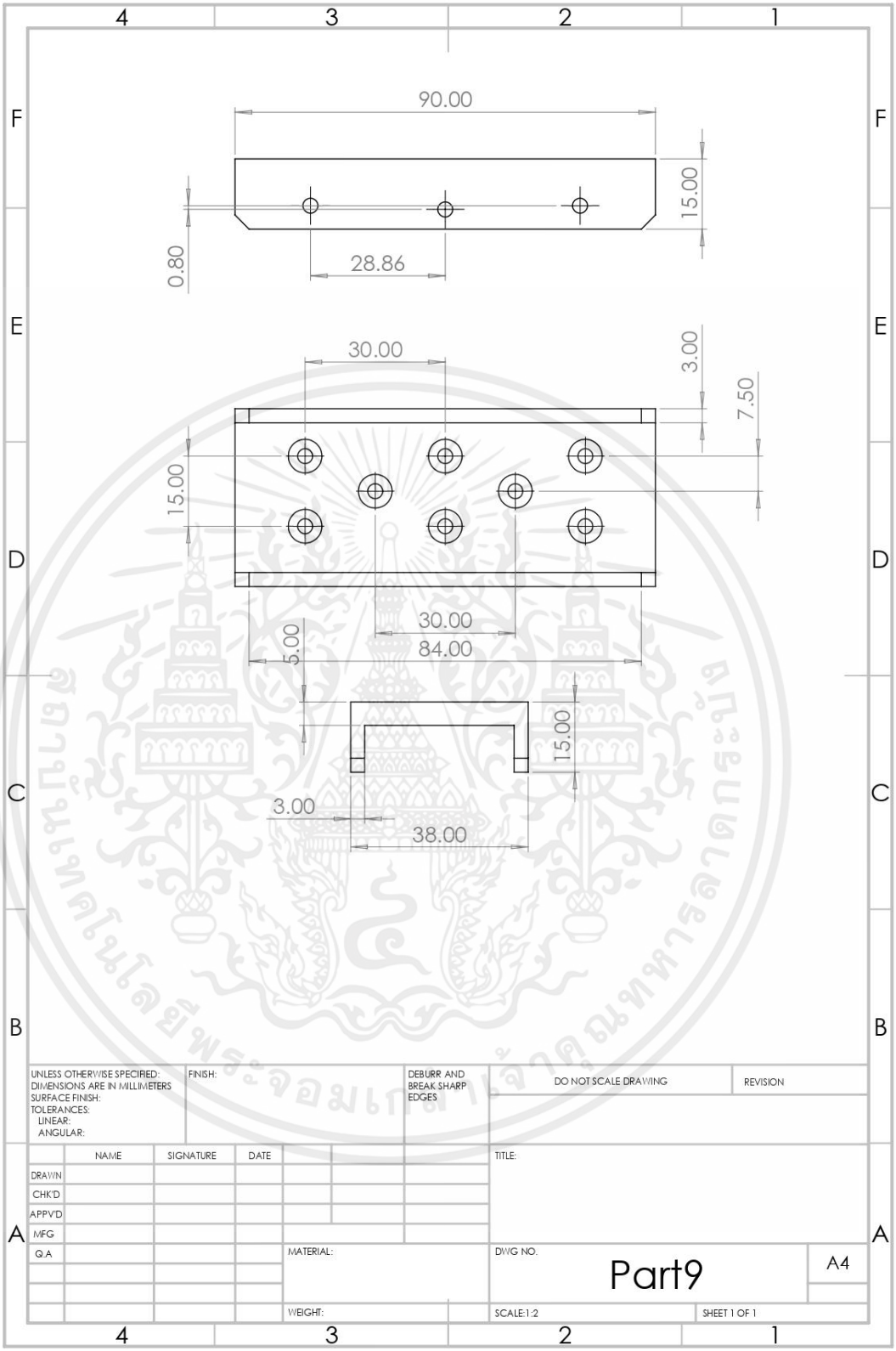
รูปที่ 29 Drawing part 6



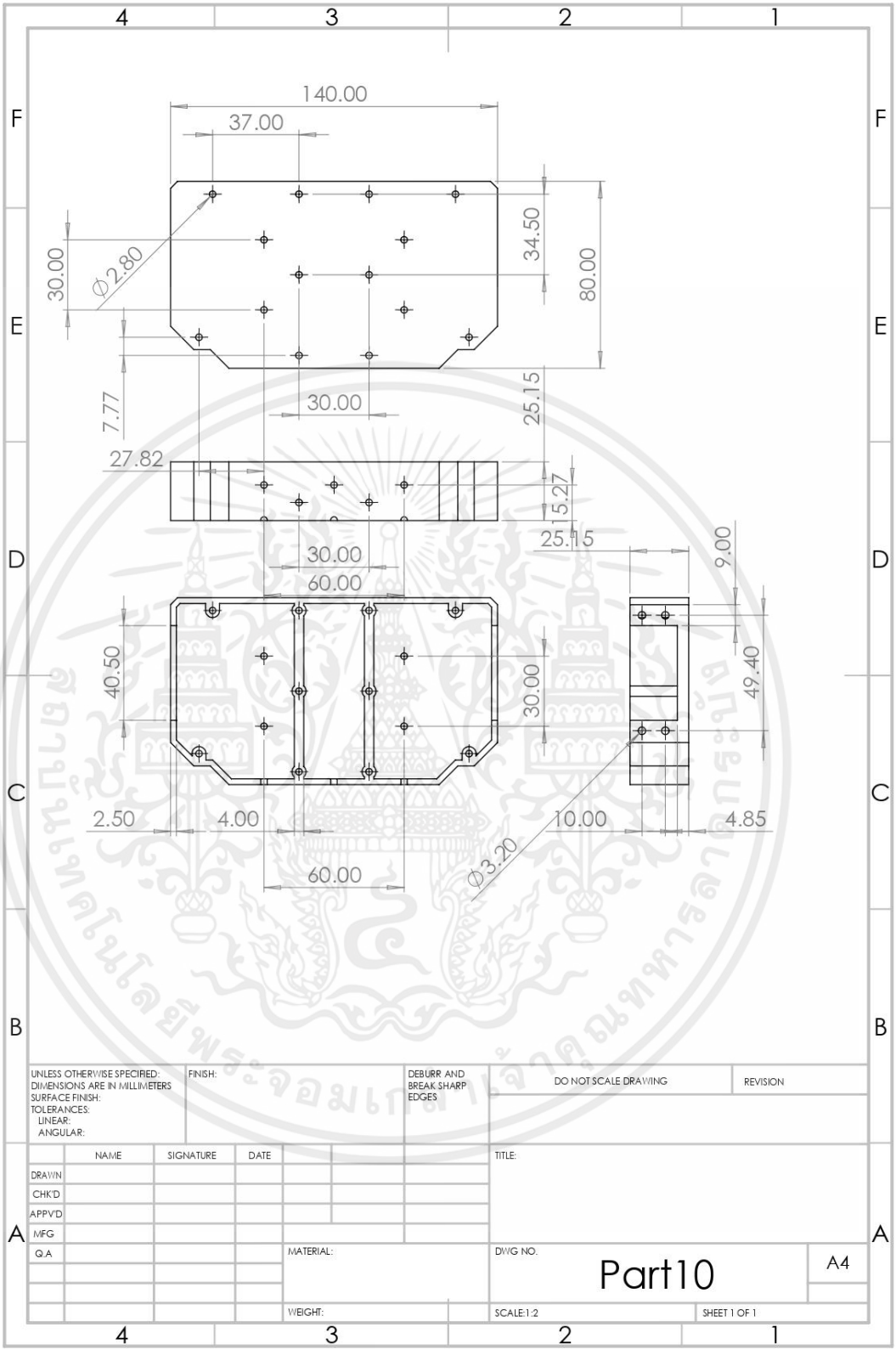
รูปที่ 30 Drawing part 7



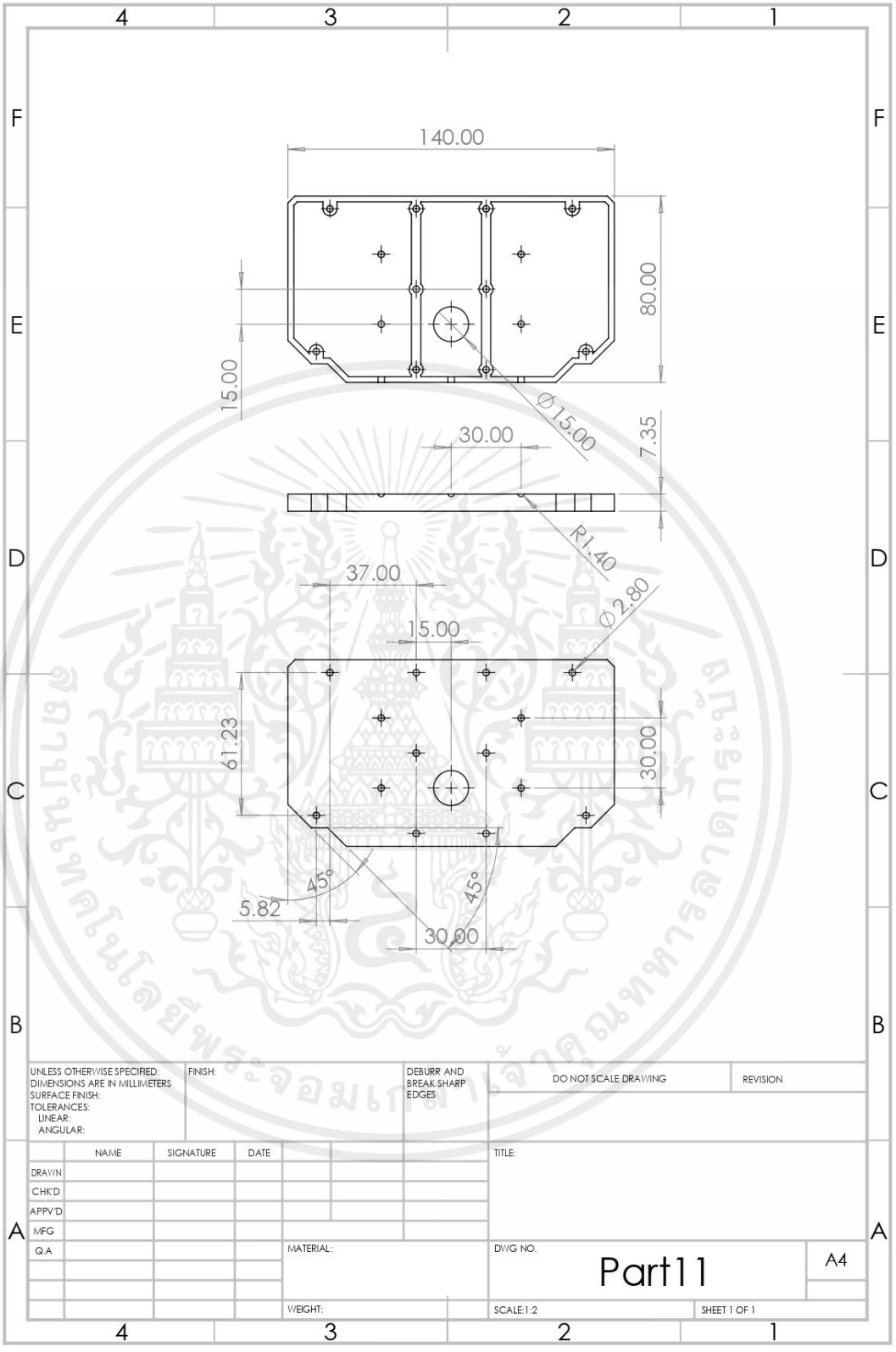
รูปที่ 31 Drawing part 8



รูปที่ 32 Drawing part 9

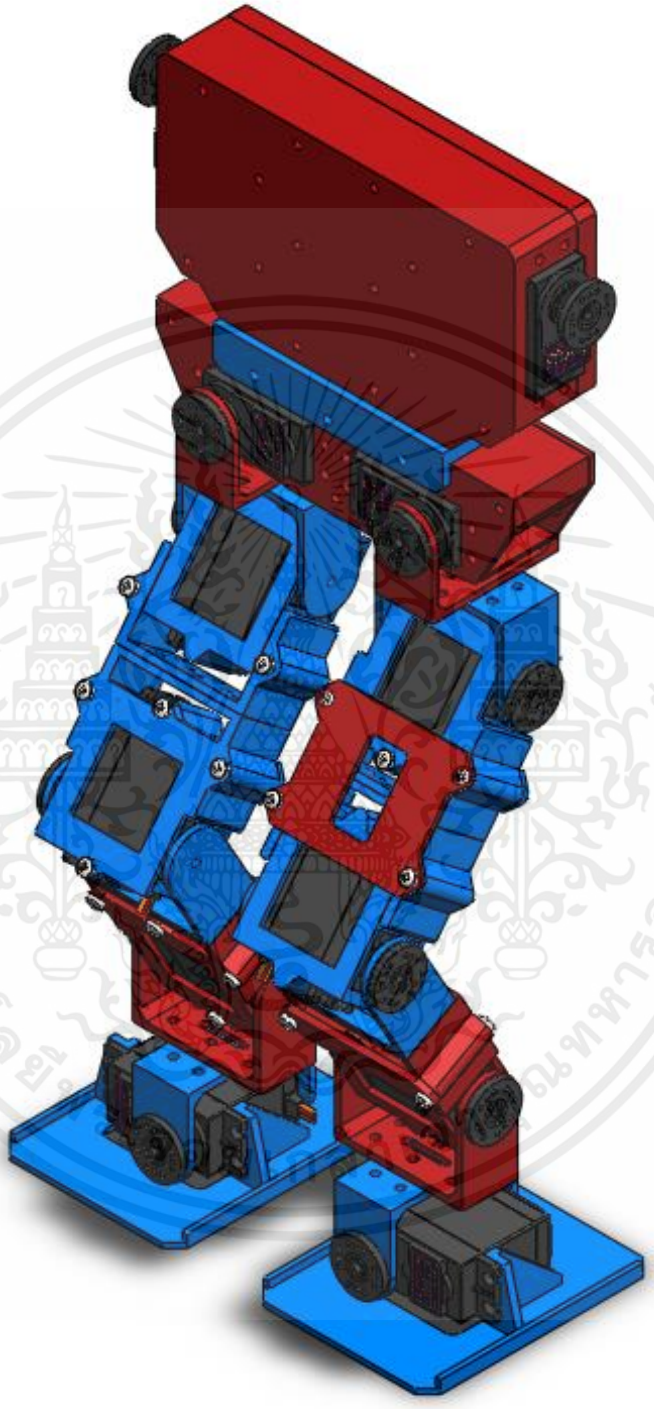


รูปที่ 33 Drawing part 10



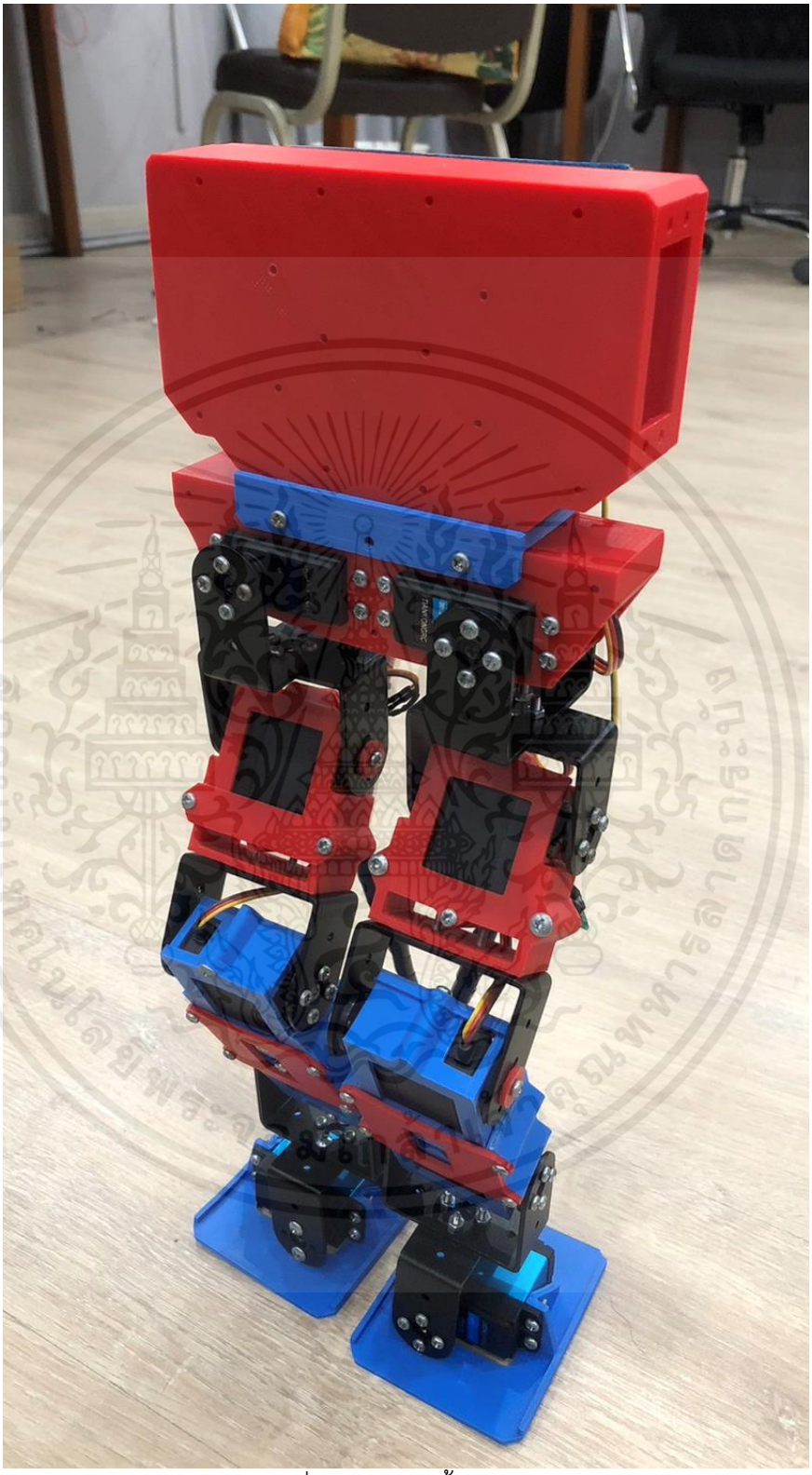
รูปที่ 34 Drawing part 11

3.1.3 3D model หุ่นยนต์



รูปที่ 35 3D model หุ่นยนต์

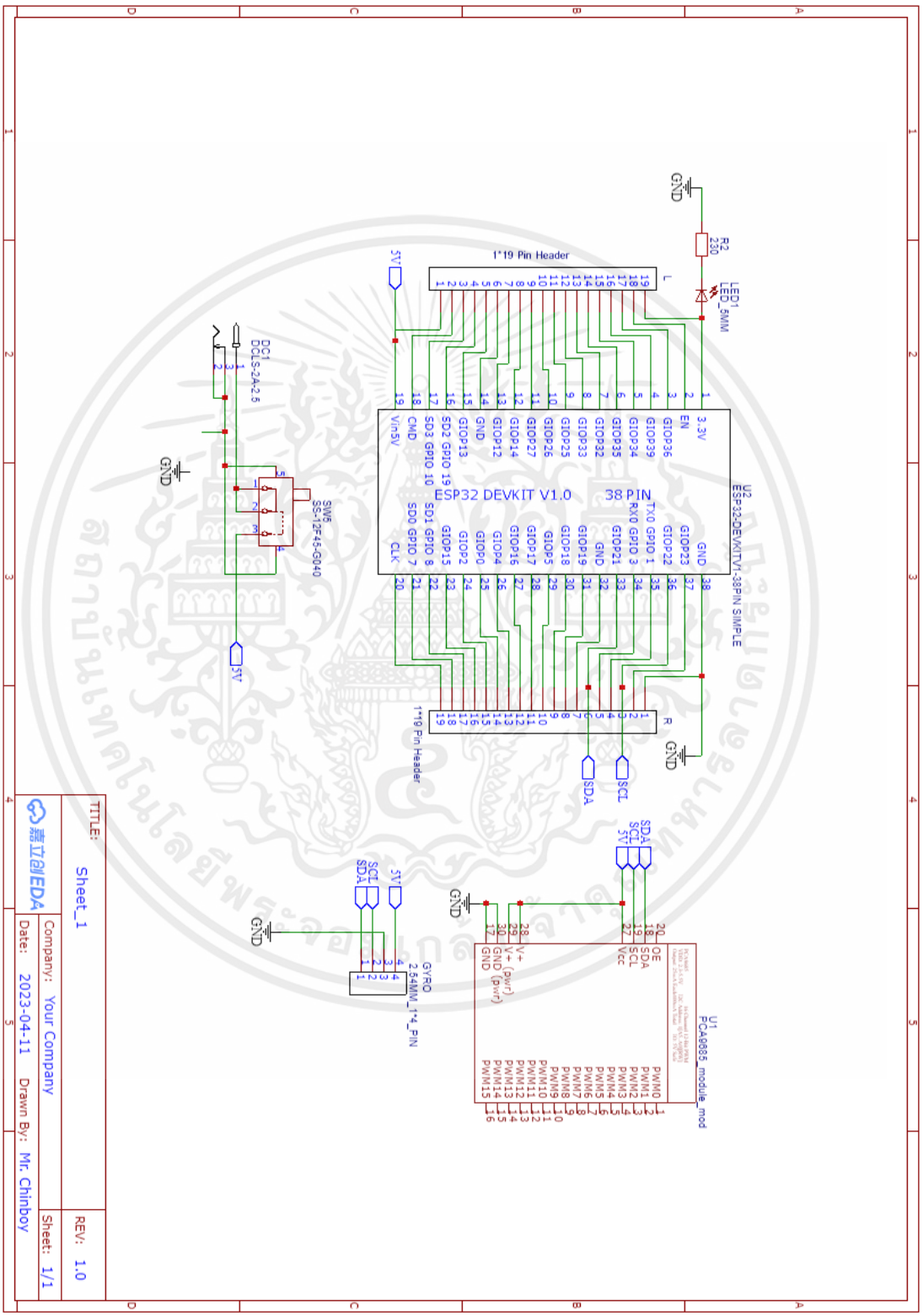
3.1.4 หุ่นยนต์ชิ้นงานจริง



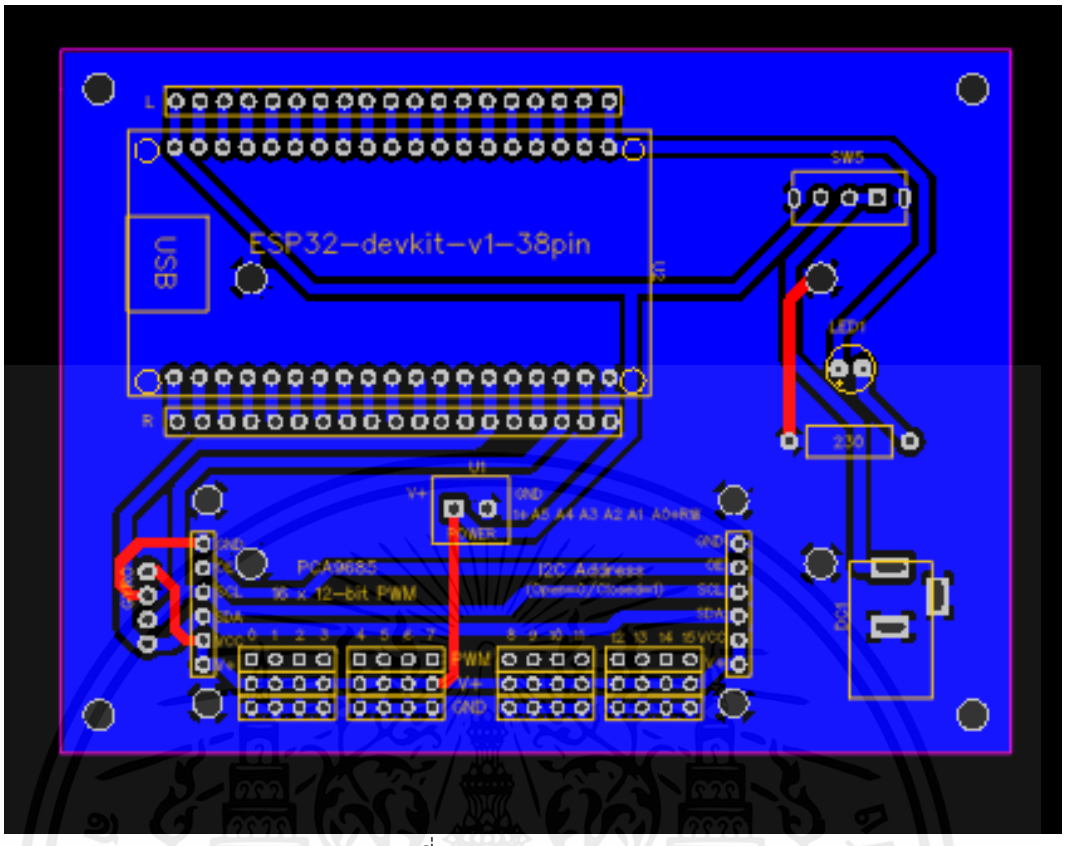
รูปที่ 36 หุ่นยนต์ชิ้นงานจริง

3.2 การออกแบบ PCB

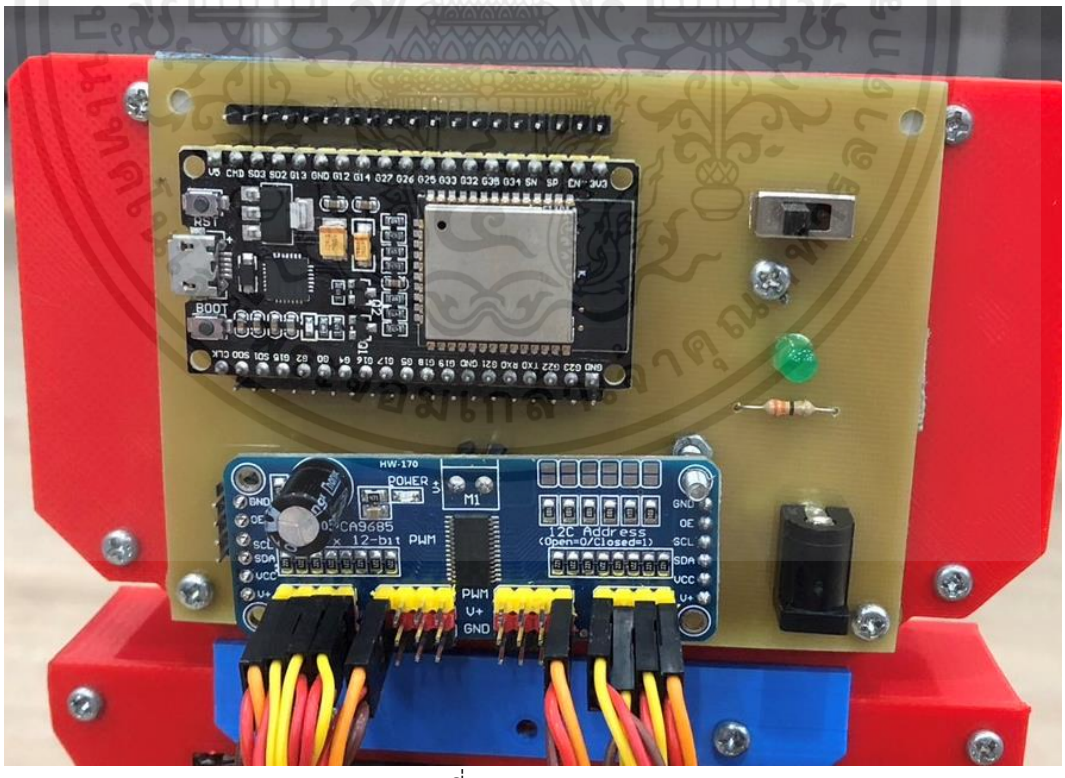
3.2.1 ออกแบบ PCB Mainboard



รูปที่ 37 Schematic mainboard

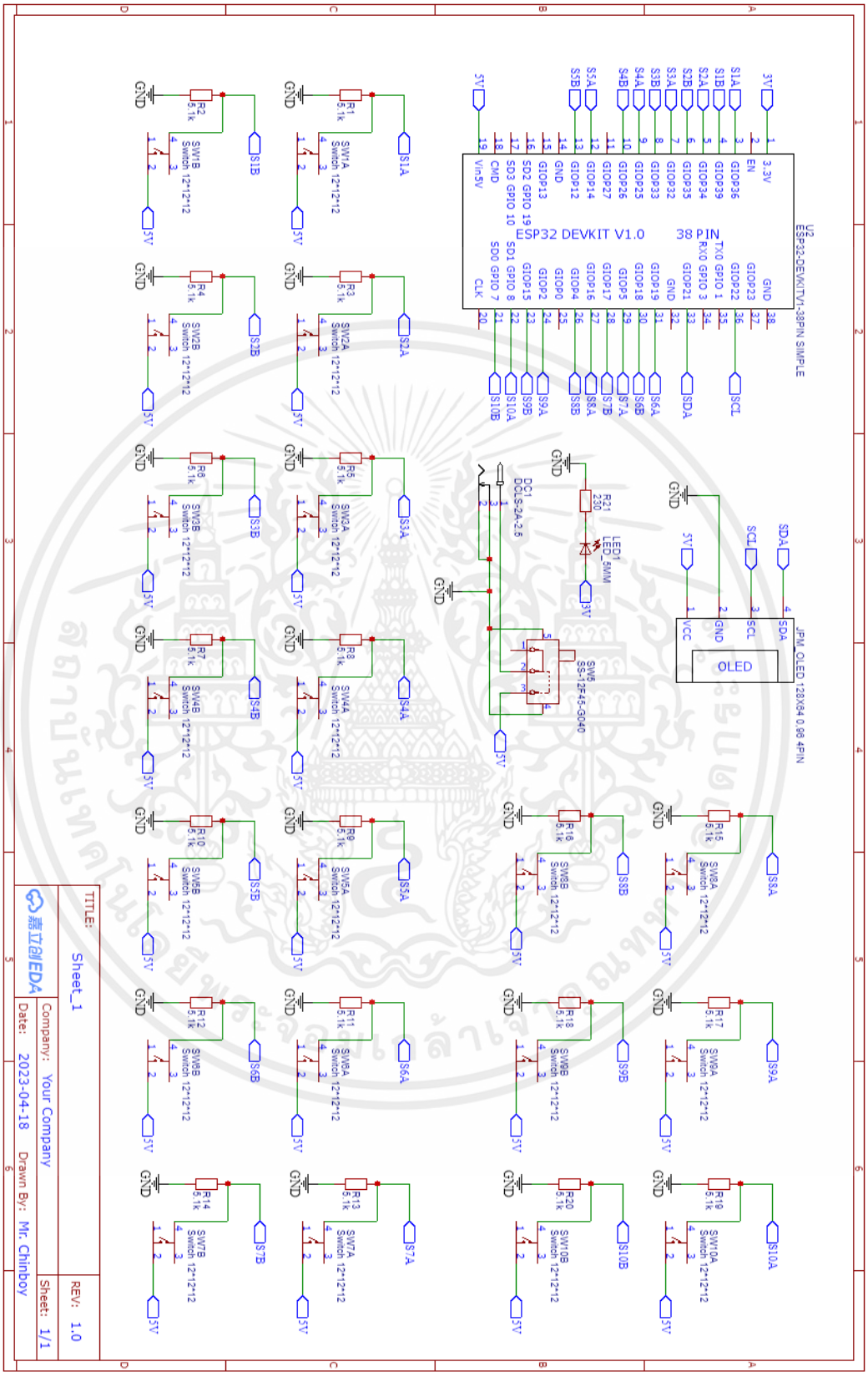


รูปที่ 38 PCB mainboard

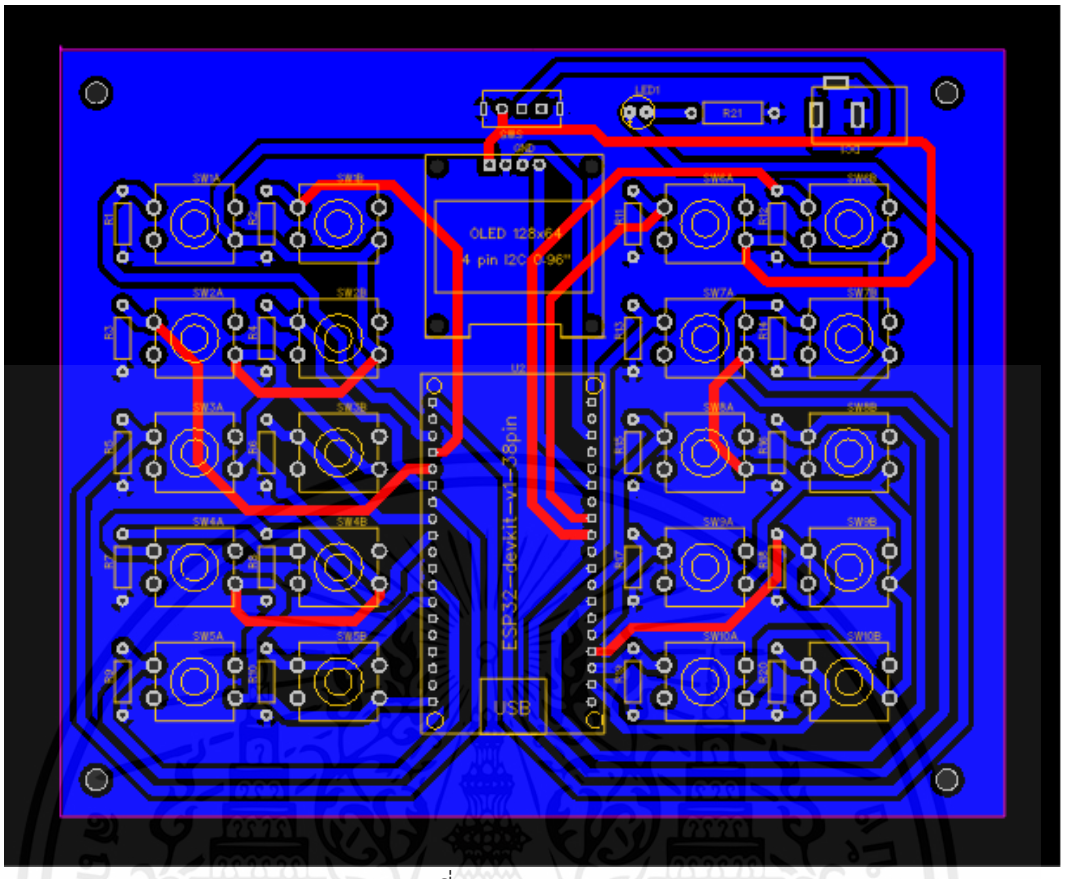


รูปที่ 39 Mainboard

3.2.2 ออกแบบ PCB Remote



รูปที่ 40 Schematic Remote

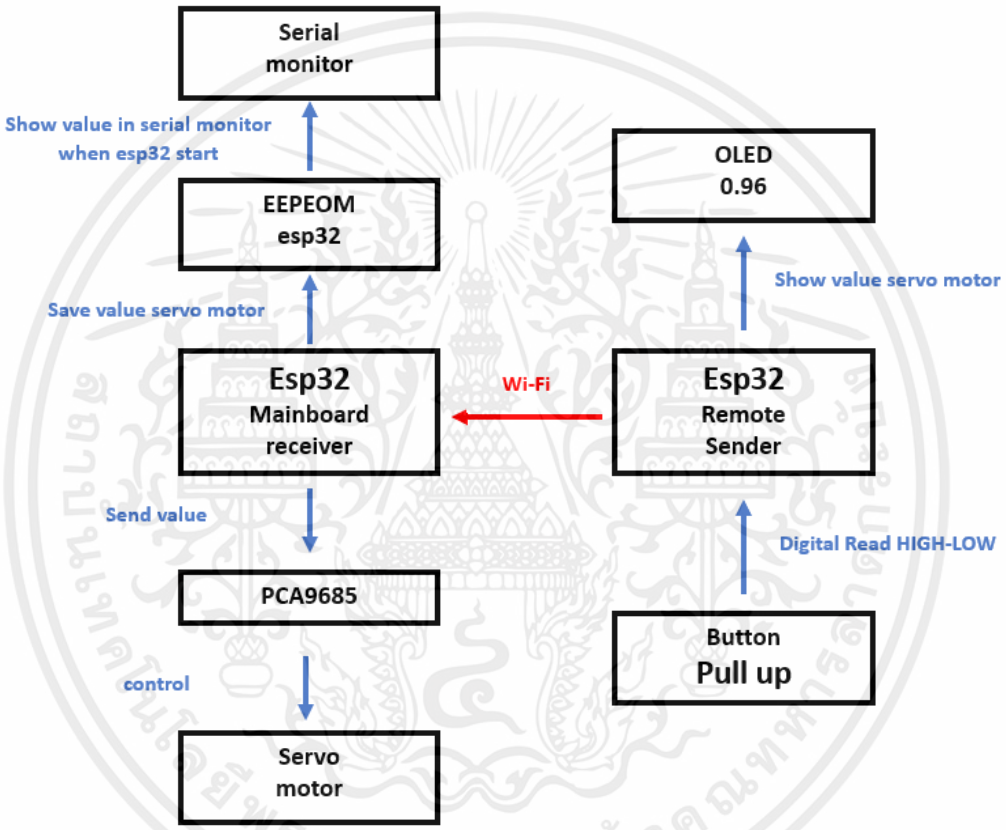


รูปที่ 41 PCB Remote

รูปที่ 42 Remote

3.3 หลักการทำงาน

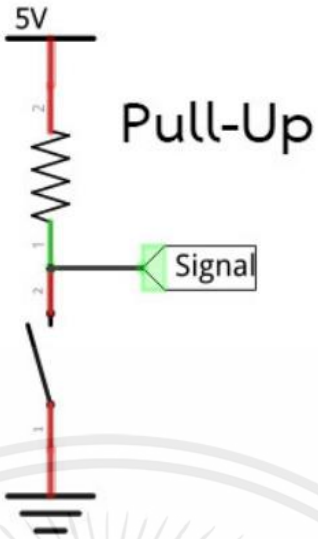
ในโปรเจคนี้ จะแบ่งการทำงานได้เป็น 2 ส่วนคือ ส่วนของ Mainboard มีหน้าที่ในการสั่งการทำงานที่ควบคุม servo motor 10 ตัว และ ส่วนของ Remote ที่จะค่อยส่งค่าองศาของ servo motor เพื่อใช้ในจัดท่าของหุ่นยนต์ แต่ละท่าๆ เพื่อให้หุ่นยนต์ก้าวเดินได้ ซึ่งทั้งสองบอร์ด สื่อสารกันผ่าน Wi-Fi โดยที่ mainboard เป็นตัวรับ และ Remote เป็นตัวส่งสัญญาณ ให้กับ address ของ Mainboard



รูปที่ 43 ไดอะแกรมการทำงาน

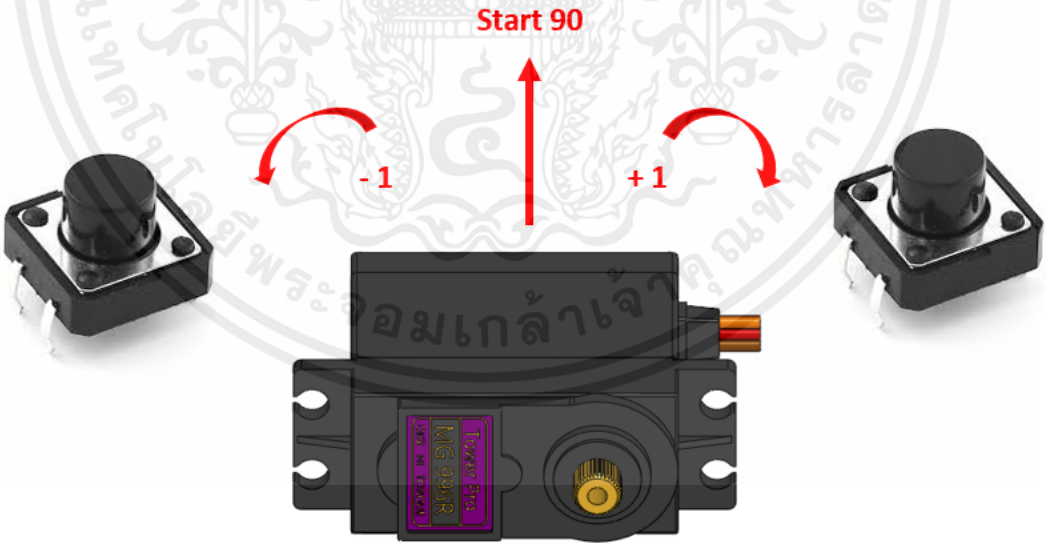
3.3.1 Esp32 Remote Sender

ในส่วนนี้จะเป็นส่วนของรีโมทการควบคุมการทำงานของหุ่นยนต์ โดยที่รีโมทจะมีปุ่ม button ทั้งหมด 20 ปุ่ม เป็นการต่อแบบ pull up บอร์ด esp32 จะอ่านค่าแบบดิจิตอลได้เป็น 1 เมื่อทำการกดปุ่ม ใช้ 2 ปุ่มเพื่อส่งค่าสำหรับควบคุมการทำงานของ servo motor 1 ตัว โดยที่เมื่อกดปุ่มแรก จะทำให้ค่าองศาเพิ่ม และถ้ากดอีกปุ่ม จะทำให้ค่าองศาลดลง



รูปที่ 44 การต่อ button แบบ pull up

ในส่วนของค่าองศาของ servo motor จะทำการสร้าง Array ขึ้นมาทั้งหมด 10 ช่อง A[10] โดยเริ่มใช้ Array ช่องที่ 1 เพิ่มไม่ให้เกิดความสับสน ซึ่ง Array ทุกตัวจะถูกตั้งค่าเริ่มต้นไว้ที่ 90 เพื่อให้ servo motor อยู่ตรงกลาง จะทำให้หมุนซ้าย หมุนขวา ได้เท่ากัน จากนั่นจะทำการสร้างฟังก์ชั่น For loop ในการอ่านค่า button แต่ละตัว เมื่อมีการกด button ก็จะเกิดการเปลี่ยนค่าของตัว Array ที่จับคู่กับ button นั่นๆ แล้วทำการส่งค่า ผ่าน Wi-Fi ไปให้ Mainboard

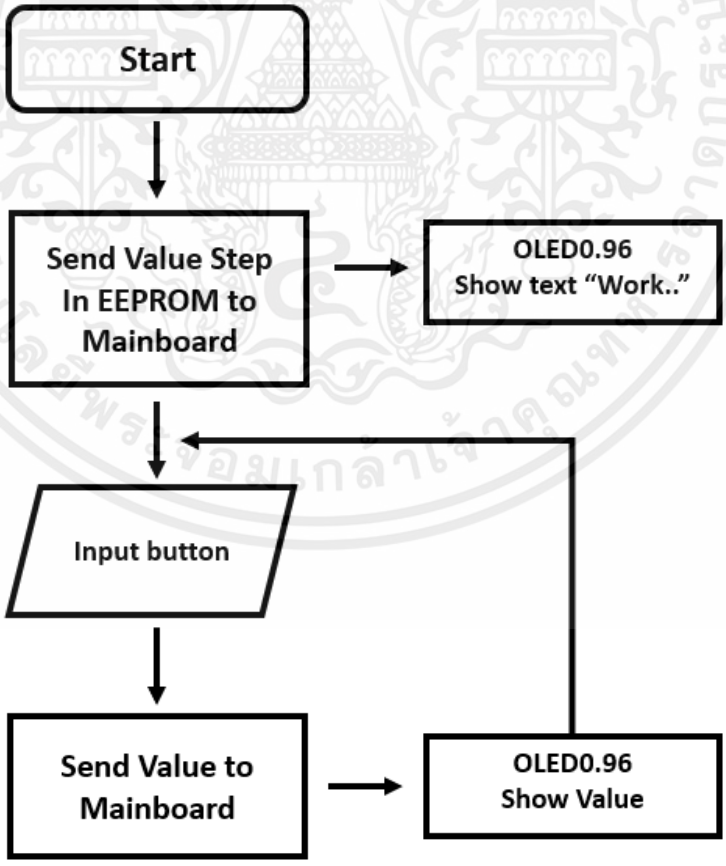


รูปที่ 45 การทำงาน servo motor ร่วมกับ button

และในทำนองเดียวกัน เมื่อทำการส่งค่า servo motor ไปแล้ว ค่านั่นก็จะถูกแสดงให้เห็นทาง OLED ของ บอร์ด Remote ด้วย

รูปที่ 46 OLED 0.96

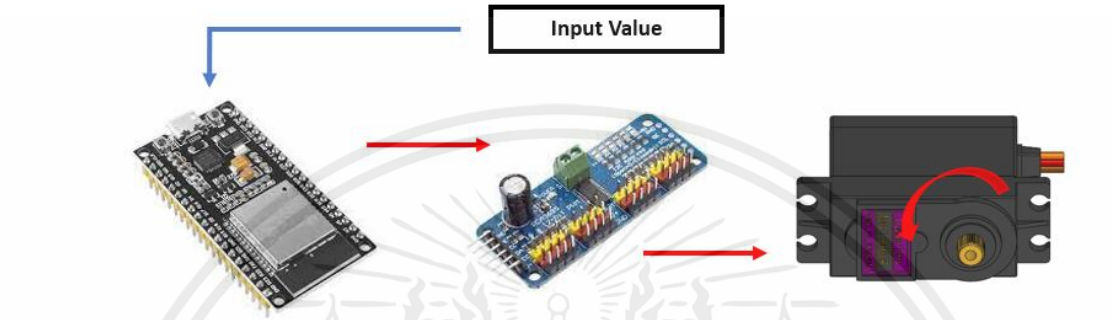
และเมื่อทำการ จัดท่าของหุ่นยนต์ในการก้าวเกินได้แล้ว เราจะนำค่านั่นๆที่ทำเป็น library มาใส่ไว้ใน Void setup ของ esp32 Remote และทำการส่งค่าเหล่านั่น ไปให้กับตัว mainboard ทุกครั้งที่ทำการเริ่มใช้งาน จะทำให้เรา สามารถทดสอบการทำงานของหุ่นยนต์ได้



รูปที่ 47 ไดอะแกรมการทำงานของ Remote

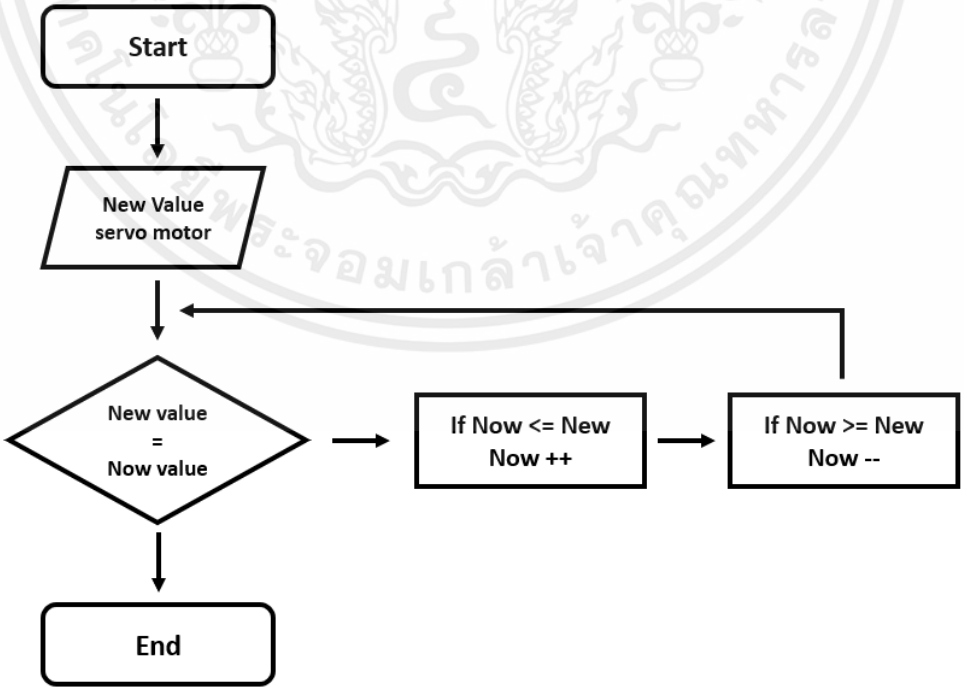
3.3.2 Esp32 Mainboard receiver

ส่วนของ Mainboard จะมีหน้าที่ในการรับค่า และทำการส่งค่าเหล่านั่นไปให้ PCA9685 เพื่อทำการสั่งการทำงานของ servo motor ทั้ง 10 ตัว จะมี Void loop ในการสั่งให้ servo motor ทำงานอยู่ตลอดเวลา เมื่อค่าองศามีการเปลี่ยนแปลง servo motor ก็จะทำงานให้องศาตรงกับค่าที่รับมา



รูปที่ 48 การทำงานของ Mainboard

เมื่อ servomotor ทำงานนั่น จะมีฟังก์ชั่นในการทำงาน โดยการที่จะสร้างเป็น For loop ทำการเปรียบเทียบค่าตำแหน่งของ servo motor ปัจจุบัน และค่าองศาที่ส่งมา หากไม่เท่ากัน จะทำการเพิ่มองศาที 1 แล้ววน loop แบบนี้จนกว่าค่าองศาจะเท่ากัน โดยที่ใน For loop จะมี delay ใส่ไว้เพื่อที่เราจะสามารถควบคุมความเร็วในการหมุนของ Servo motor ได้

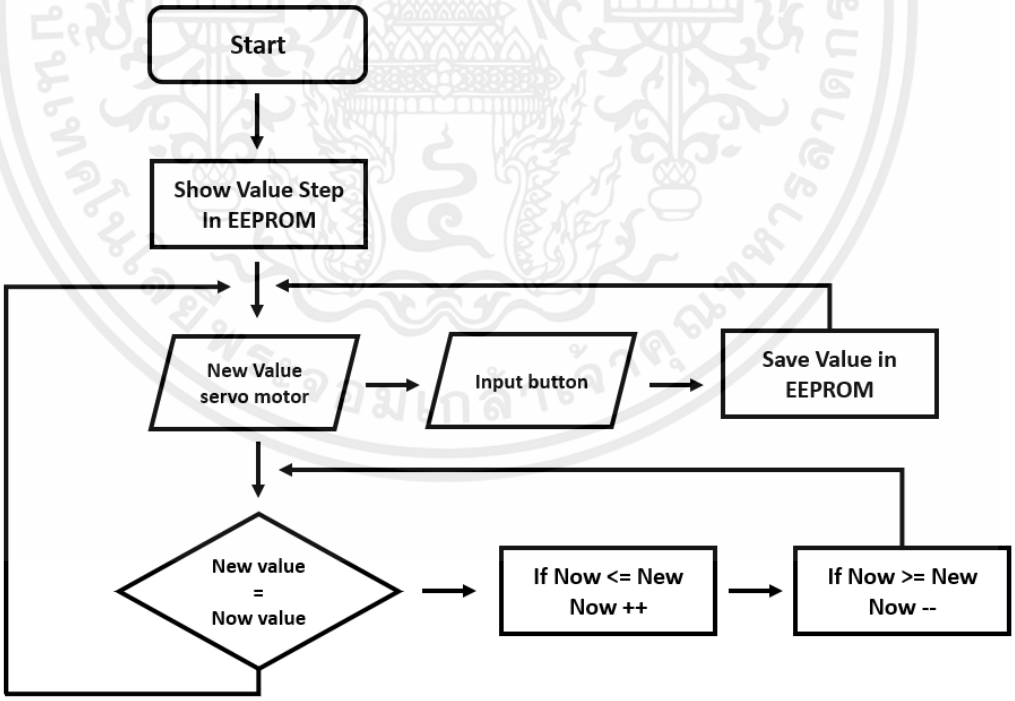


รูปที่ 49 flowchart การเพิ่มลด องศาของ servo motor

ในส่วนของ Mainboard จะมีฟังก์ชั่นในการช่วยจดจำค่าของ servo motor โดยเมื่อต้องการบันทึกค่า ให้กดปุ่ม button ที่อยู่กับ mainboard จากนั่นจะทำการบันทึกค่าของ servo ทั้ง 10 ตัว และค่า Gyro sensor ไว้ใน EEPROM ไว้เป็น Stepซึ่งจะไม่หายเมื่อทำการหยุดการทำงานของบอร์ด esp32

รูปที่ 50 การบันทึกค่า ลง EEPROM

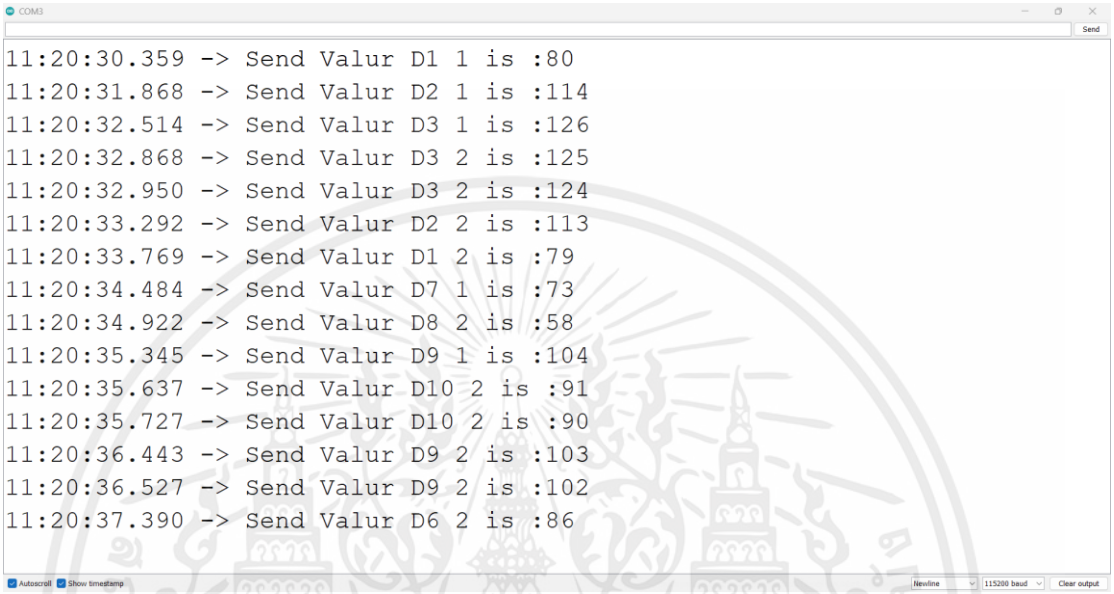
และเมื่อทำการเริ่มต้นการทำงานของ Mainboard ใหม่ จะทำการแสดงค่าที่บันทึกไว้ผ่านทาง Serial monitor ก่อน ที่จะเข้าสู่ loop การรับค่าและสั่งการทำงานของ servo motor



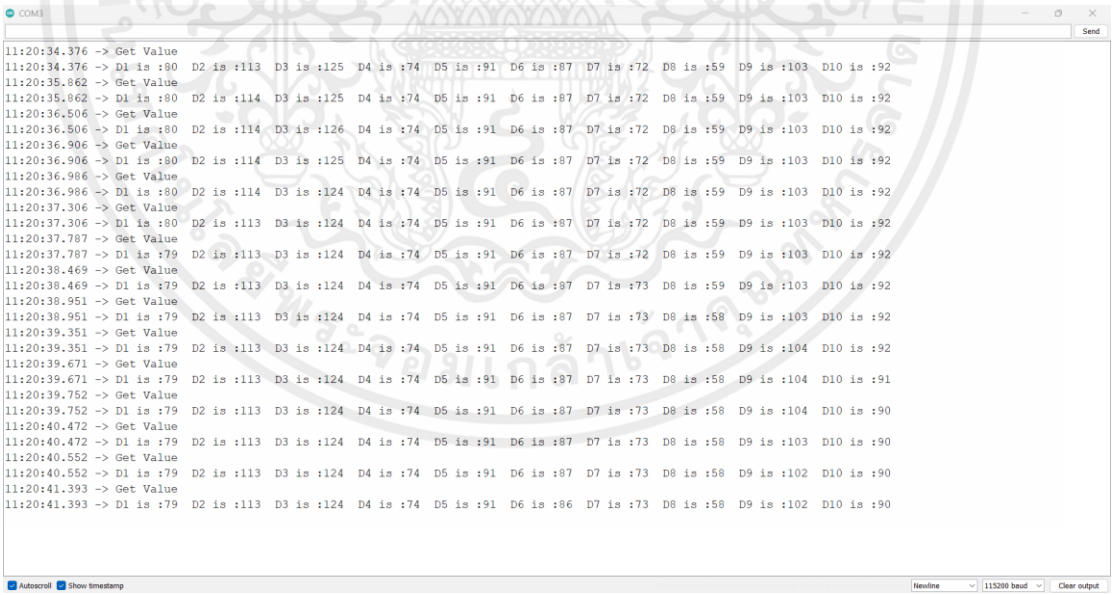
รูปที่ 51 ไดอะแกรมการทำงานของ Mainboard

บทที่ 4
ผลการทดลอง

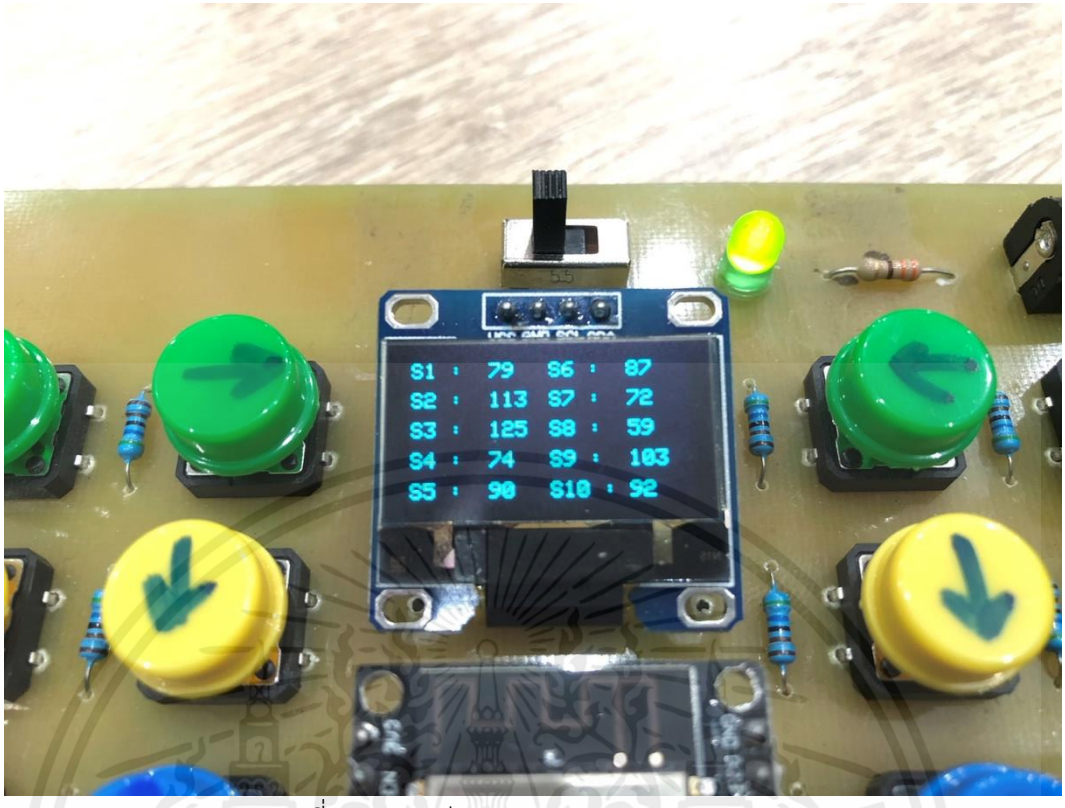
4.1 การส่งค่า servo ระหว่างบอร์ด



รูปที่ 52 Serial monitor Remote

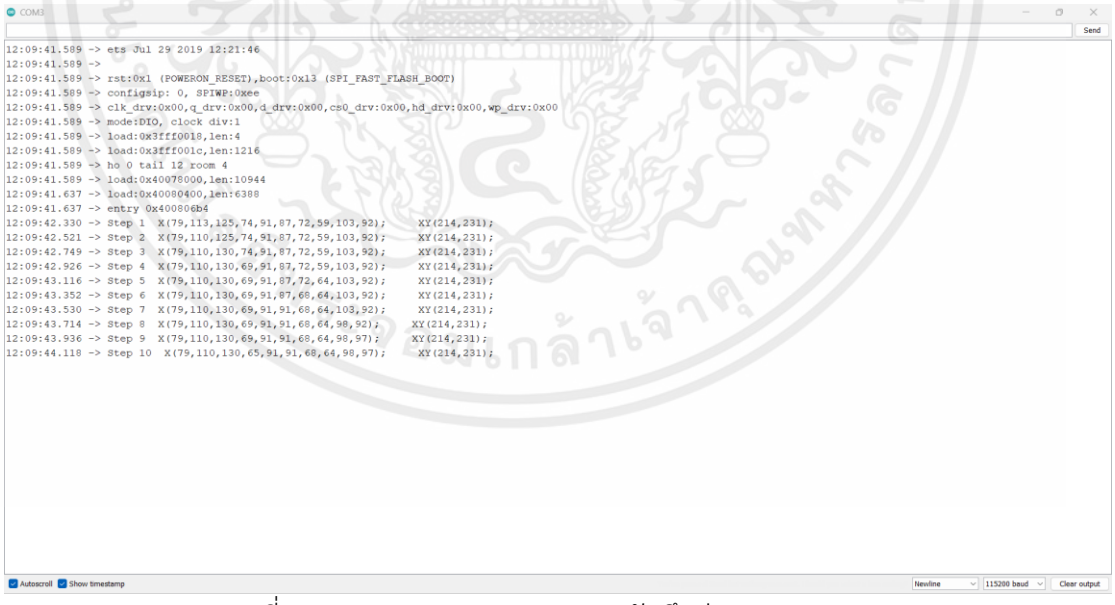


รูปที่ 53 Serial monitor Mainboard



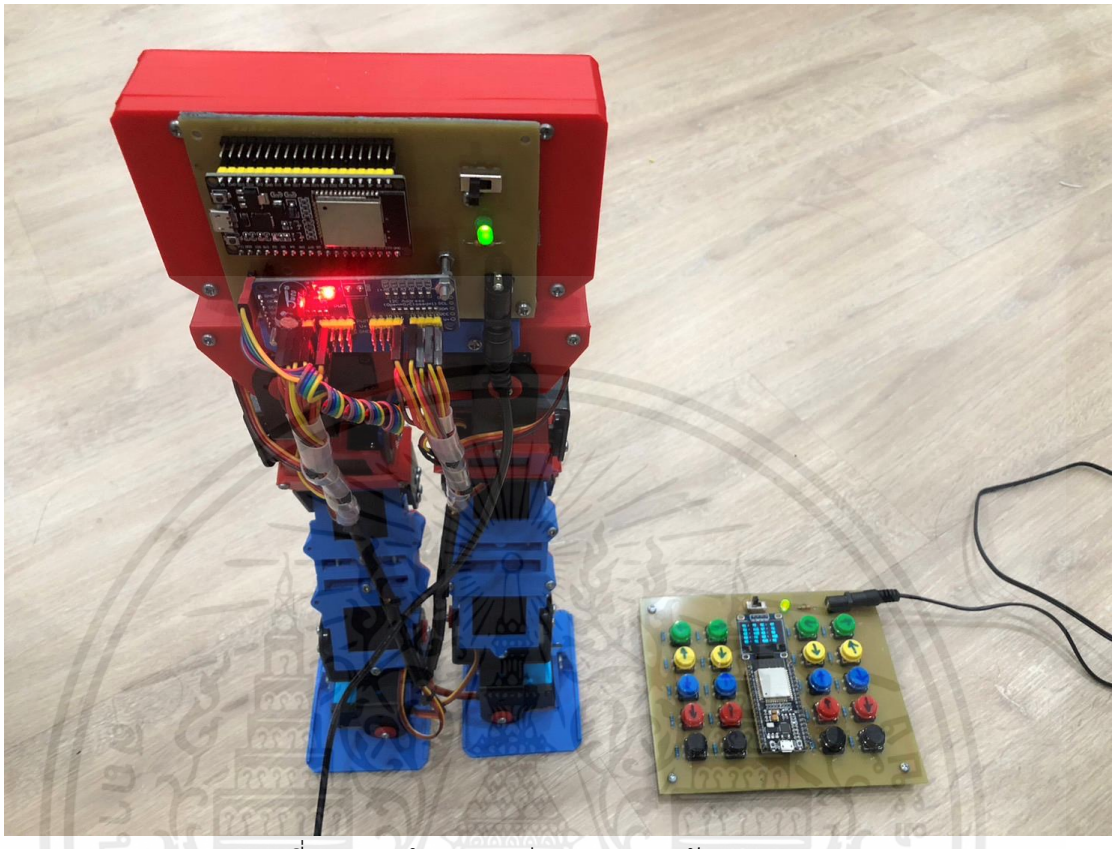
รูปที่ 54 แสดงค่า servo ออกทาง OLED 0.96

4.2 การส่งค่า servo ลงใน EEPROM และ แสดงบน Serial monitor



รูปที่ 55 Serial monitor แสดงการบันทึกค่าจาก EEPROM

4.3 การทดสอบการทำงาน ระหว่างรีโมท กับหุ่นยนต์



รูปที่ 56 การทำงานระหว่าง Remote กับ Mainboard

ESP32_Sender_board | Arduino 1.8.19 (Windows Store 1.8.57.0)

File Edit Sketch Tools Help

ESP32_Sender_board §

```
112  X(79, 130, 125, 83, 76, 82, 42, 59, 76, 85);      //XY(238, 13);
113  X(79, 130, 125, 83, 76, 82, 42, 46, 76, 85);      //XY(241, 14);
114  X(79, 130, 125, 83, 76, 82, 42, 46, 96, 85);      //XY(240, 14);
115  X(79, 120, 125, 83, 76, 82, 42, 46, 96, 85);      //XY(22, 6);
116  X(79, 120, 125, 83, 76, 82, 50, 46, 96, 85);      //XY(28, 6);
117  X(79, 120, 125, 80, 76, 82, 50, 46, 96, 85);      //XY(24, 6);
118  X(79, 120, 125, 80, 76, 82, 70, 46, 96, 85);      //XY(32, 6);
119  X(79, 120, 125, 74, 76, 82, 70, 46, 96, 85);      //XY(23, 7);
120  X(79, 113, 125, 74, 76, 82, 72, 46, 96, 85);      //XY(33, 7);
121  X(79, 113, 125, 74, 76, 82, 72, 59, 96, 85);      //XY(33, 7);
122  X(79, 113, 125, 74, 76, 82, 72, 59, 103, 85);      //XY(32, 7);
123  X(79, 113, 125, 72, 76, 82, 72, 59, 103, 85);      //XY(18, 7);
124  X(79, 113, 125, 74, 76, 82, 72, 59, 103, 85);      //XY(22, 7);
125  X(79, 113, 125, 74, 76, 82, 72, 59, 103, 75);      //XY(22, 10);
126  X(79, 113, 125, 74, 90, 82, 72, 59, 103, 75);      //XY(21, 1);
127  X(79, 113, 125, 74, 90, 82, 72, 59, 103, 82);      //XY(20, 254);
128  X(79, 113, 125, 74, 90, 95, 72, 59, 103, 82);      //XY(17, 3);
129  X(79, 113, 125, 74, 90, 87, 72, 59, 103, 82);      //XY(20, 0);
130  X(79, 113, 125, 74, 90, 87, 72, 59, 103, 92);      //XY(23, 254);//
131  X(79, 130, 125, 83, 76, 82, 42, 46, 96, 72);      //XY(240, 13);
132  X(79, 130, 125, 83, 81, 82, 42, 46, 96, 72);      //XY(241, 7);
133  X(79, 130, 125, 83, 81, 82, 42, 46, 96, 85);      //XY(241, 255);
```

No changes necessary for Auto Format.

รูปที่ 57 ตัวอย่าง code library การเดินของหุ่นยนต์

บทที่ 5
สรุปผลการทดลอง

เริ่มเป้าหมายในการทำหุ่นยนต์ ในโปรเจคนี้คือทำทั้งตัว มีแขนและสามารถหยิบจับวัตถุได้ แต่เนื่องจาก Servo motor ที่ใช้ ไม่ได้มีประสิทธิภาพสูง และเป็นของราคาถูก จึงไม่สามารถรับน้ำหนักของหุ่นยนต์ได้เยอะ จึงได้ทำการตัดส่วนแขน และส่วนหัวออกไป เพื่อให้ servo motor ยังสามารถทำได้ไง น้ำหนักโดยรวมของหุ่นยนต์ อยู่ที่ 1.2 กิโลกรัม มีส่วนสูง 32 เซนติเมตร และด้วยเหตุผลเดียวกัน ทำให้ไม่สามารถนำ Gyro sensor เข้ามาช่วยในการจัดสมดุลขอลหุ่นยนต์ได้ เนื่องจากพอเกิดการขยับของ servo ตัวใดตัวหนึ่งขึ้น จะทำให้ servo ตัวอื่นๆมีการสั่นขึ้นเล็กน้อยเนื่องด้วย แรงบิตของ servo ที่ใช้ไม่มากพอ

เนื่องด้วยเหตุผลเดียวกับที่กล่าวมา จึงไม่สามารถใช้แบตเตอร์รี่กับหุ่นยนต์ได้ เพราะจะเป็นการเพิ่มน้ำหนัก และเพิ่มภาระให้หุ่นยนต์เพิ่ม หรืออาจจะซื้อแบตเตอร์รี่ที่มีขนาดเล็ก ให้ไฟที่ต้องการได้ แต่ราคาแบตเตอร์ก็สูงตาม จึงเปลี่ยนเป็น Adapter 5V 4A แทน

จากการทดลองการจัดท่าทางเพื่อ ทำ library ในการก้าวเดินของหุ่นยนต์พบว่า เป็นการยากที่จะจัดท่าทาง โดยการปรับเพิ่มองศาของ Servo แต่ล่ะตัว และจะต้องใช้เวลานาน ในการศึกษา และปรับปรุงให้หุ่นยนต์ มีการก้าวเดินที่สมดุลมากที่สุด

จากการทำโปรเจคนี้ เมื่อรวมค่าใช้จ่ายในการซื้ออุปกรณ์ทั้งหมดแล้ว อยู่ที่ประมาณ 6000 บาท ซึ่งหากจะทำต่อแล้วเปลี่ยนอุปกรณ์ให้มีประสิทธิภาพที่ดีขึ้นแล้วสามารถนำไปใช้ในการแข่งขันได้ จ่ายใช้จ่ายจะเพิ่มเป็นประมาณ 12000 ซึ่งราคาข้องตัวหุ่นยนต์ที่นำเข้าจากต่างประเทศ ราคาเริ่มต้นที่ 30000 บาท

เอกสารอ้างอิง

[1] artronshop. “Esp32 เบื้องต้น” [ออนไลน์], แหล่งที่มา: https://www.artronshop.co.th/article/51/esp32 สืบค้นเมื่อ 17 มกราคม 2566.

[2] แสงชัย มิเตอร์. “เซอรโวมอเตอร์คืออะไร” [ออนไลน์],แหล่งที่มา : https://www.sangchaimeter.com/support_detail/servo-motor สืบค้นเมื่อ 17 มกราคม 2566.

[3] AranaCorp’s. “Using a PCA9685 module with Arduino”[ออนไลน์],แหล่งที่มา : https://www.aranacorp.com/en/using-a-pca9685-module-with-arduino/ สืบค้นเมื่อ 25 มกราคม 2566.

[4] Random Nerd. “ESP32 Flash Memory – Store Permanent Data”[ออนไลน์], แหล่งที่มา : https://randomnerdtutorials.com/esp32-flash-memory/ สืบค้นเมื่อ 19 กุมภาพันธ์ 2566.

[5] ฮาร์ดแวร์ปอนด์. “MPU6050: โมดูลสำหรับกำหนดตำแหน่งกับ Arduino” [ออนไลน์], แหล่งที่มา : https://www.hwlibre.com/th/mpu6050/ สืบค้นเมื่อ 19 กุมภาพันธ์ 2566.

[6] Random Nerd. “Getting Started with ESP-NOW” [ออนไลน์], แหล่งที่มา : https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/ สืบค้นเมื่อ 9 มีนาคม 2566.

[7] Applicad Public Company Limited. “SOLIDWORKS” [ออนไลน์], แหล่งที่มา: https://www.applicadthai.com/solidworks-package/?ref=adwords&gad=1&gclid=CjwKCAjw3ueiBhBmEiwA4BhspBC7T4GGL0L8-6Z-XaXwOsRnfcLrirWniwAtfOaXyVYFT2-BktrolRoCqB0QAvD_BwE สืบค้นเมื่อ 25 มีนาคม 2566.

[8] https://www.ai-corporation.net/. “Arduino IDE คืออะไร” [ออนไลน์], แหล่งที่มา: https://www.ai-corporation.net/2021/11/18/what-is-arduino-ide/ สืบค้นเมื่อ 27 มีนาคม 2566.

ภาคผนวก

โปรแกรม Mainboard

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>
#include <Bounce2.h>
#include <esp_now.h>
#include <WiFi.h>
#include <EEPROM.h>
#include <MPU6500_WE.h>
#define EEPROM_SIZE 512
#define MPU6500_ADDR 0x68

Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();
MPU6500_WE myMPU6500 = MPU6500_WE(MPU6500_ADDR);

int D[14] = {0, 79, 117, 125, 74, 91, 87, 68, 59, 103, 92};
int val;
int SW;
int I = 0;
int Num;
int O;
int N;
int A[2];
int G;

typedef struct struct_message {
  int S[12] = {0, 79, 117, 125, 74, 91, 87, 68, 59, 103, 92};

} struct_message;
struct_message myData;

#define SERVOMIN  550
#define SERVOMAX  2450
#define SERVO_FREQ 50 // Analog servos run at ~50 Hz updates

#define servo01 0
#define servo02 1
#define servo03 2
#define servo04 3
#define servo05 4
#define servo06 11
#define servo07 12
#define servo08 13
#define servo09 14
#define servo10 15

int servo1PPos, servo2PPos, servo3PPos, servo4PPos, servo5PPos, servo6PPos, servo7PPos, servo8PPos, servo9PPos, servo10PPos;
int degree_1, degree_2, degree_3, degree_4, degree_5, degree_6, degree_7, degree_8, degree_9, degree_10;

Bounce debouncer = Bounce();

void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&myData, incomingData, sizeof(myData));

  D[1] = myData.S[1];
  D[2] = myData.S[2];
  D[3] = myData.S[3];
  D[4] = myData.S[4];
  D[5] = myData.S[5];
  D[6] = myData.S[6];
  D[7] = myData.S[7];
  D[8] = myData.S[8];
  D[9] = myData.S[9];
  D[10] = myData.S[10];

  Serial.println("Get Value");

  for (int i = 1; i <= 9; i++) {
    Serial.print("D");
    Serial.print(i);
    Serial.print(" is :");
    Serial.print(D[i]);
    Serial.print("  ");
  }
  Serial.print("D");
  Serial.print(10);
  Serial.print(" is :");
  Serial.println(D[10]);


}

void setup() {

  Serial.begin(115200);
  pinMode(16, INPUT);
  EEPROM.begin(EEPROM_SIZE);
  WiFi.mode(WIFI_STA);
```

```
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  esp_now_register_recv_cb(OnDataRecv);

  pwm.begin();
  pwm.setOscillatorFrequency(27000000);  // The int.osc.
is closer to 27MHz
  pwm.setPWMFreq(SERVO_FREQ);  // Analog servos run
at ~50 Hz updates

  myMPU6500.autoOffsets();
  myMPU6500.enableGyrDLPF();
  myMPU6500.setGyrDLPF(MPU6500_DLPF_6);
  myMPU6500.setSampleRateDivider(5);
  myMPU6500.setGyrRange(MPU6500_GYRO_RANGE_250);
  myMPU6500.setAccRange(MPU6500_ACC_RANGE_2G);
  myMPU6500.enableAccDLPF(true);
  myMPU6500.setAccDLPF(MPU6500_DLPF_6);
  delay(200);

  I = EEPROM.read(0);
  N = I;
  Num = (I / 12);
  I = 1;

  for (int n = 1; n <= Num; n++) {
    Serial.print("Step ");
    Serial.print(n);
    Serial.print("  X(");
    for (int n = 1; n <= 9; n++) {
      O = EEPROM.read(I);
      Serial.print(O);
      Serial.print(",");
      I++;
    }
    O = EEPROM.read(I);
    Serial.print(O);
    Serial.print(");");
    Serial.print("      ");
    I++;

    G = EEPROM.read(I);
    Serial.print("XY(");
    Serial.print(G);
    Serial.print(",");
    I++;
    G = EEPROM.read(I);
    Serial.print(G);
    Serial.println(");");
    I++;

    delay(200);
  }
  I = 1;
}

void loop() {

  SW = digitalRead(16);

  if (SW == HIGH) {
    for (int n = 1; n <= 10; n++) {

      EEPROM.write(I, D[n]);
      Serial.print("D");
      Serial.print(n);
      Serial.print(" is :");
      Serial.println(D[n]);
      delay(10);
      Serial.println(I);
      I++;
      EEPROM.write(0, I);
      EEPROM.commit();

    }
    xyzFloat gValue = myMPU6500.getGValues();

    A[1] = gValue.x * 100;
    EEPROM.write(I, A[1]);
    Serial.print("X ");
    Serial.print(" is :");
    Serial.println(A[1]);
    delay(10);
    Serial.println(I);
    I++;
    EEPROM.write(0, I);
    EEPROM.commit();

    A[2] = gValue.y * 100;
    EEPROM.write(I, A[2]);
    Serial.print("Y ");
    Serial.print(" is :");
    Serial.println(A[2]);
```

```
    delay(10);
    Serial.println(I);
    I++;
    EEPROM.write(0, I);
    EEPROM.commit();
  }

  S5(D[1]);
  S4(D[2]);
  S3(D[3]);
  S2(D[4]);
  S1(D[5]);
  S10(D[6]);
  S9(D[7]);
  S8(D[8]);
  S7(D[9]);
  S6(D[10]);

}
void X(int s1, int s2, int s3, int s4, int s5, int s6, int s7, int
s8, int s9, int s10) {
  S5(s1);
  S4(s2);
  S3(s3);
  S2(s4);
  S1(s5);
  S10(s6);
  S9(s7);
  S8(s8);
  S7(s9);
  S6(s10);
}

void RUN(int x1, int x2, int x3, int x4, int x5, int x6, int x7,
int x8, int x9, int x10) {
  S1(x1);
  S2(x2);
  S3(x3);
  S4(x4);
  S5(x5);
  S6(x6);
  S7(x7);
  S8(x8);
  S9(x9);
  S10(10);
}

  void S1(int x) {
   servo1PPos = x;
   degree_1 = map(servo1PPos, 0, 180, SERVOMIN,
SERVOMAX);
   pwm.writeMicroseconds(servo01, degree_1);
   delay(10);
}

void S2(int x) {
   servo2PPos = x;
   degree_2 = map(servo2PPos, 0, 180, SERVOMIN,
SERVOMAX);
   pwm.writeMicroseconds(servo02, degree_2);
   delay(10);
}

void S3(int x) {
   servo3PPos = x;
   degree_3 = map(servo3PPos, 0, 180, SERVOMIN,
SERVOMAX);
   pwm.writeMicroseconds(servo03, degree_3);
   delay(10);
}

void S4(int x) {
   servo4PPos = x;
   degree_4 = map(servo4PPos, 0, 180, SERVOMIN,
SERVOMAX);
   pwm.writeMicroseconds(servo04, degree_4);
   delay(10);
}

void S5(int x) {
   servo5PPos = x;
   degree_5 = map(servo5PPos, 0, 180, SERVOMIN,
SERVOMAX);
   pwm.writeMicroseconds(servo05, degree_5);
   delay(10);
}

void S6(int x) {
   servo6PPos = x;
   degree_6 = map(servo6PPos, 0, 180, SERVOMIN,
SERVOMAX);
   pwm.writeMicroseconds(servo06, degree_6);
   delay(10);
}

void S7(int x) {
```

```
  servo7PPos = x;
  degree_7 = map(servo7PPos, 0, 180, SERVOMIN, SERVOMAX);
  pwm.writeMicroseconds(servo07, degree_7);
  delay(10);
}

void S8(int x) {
  servo8PPos = x;
  degree_8 = map(servo8PPos, 0, 180, SERVOMIN, SERVOMAX);
  pwm.writeMicroseconds(servo08, degree_8);
  delay(10);
}

void S9(int x) {
  servo9PPos = x;
  degree_9 = map(servo9PPos, 0, 180, SERVOMIN, SERVOMAX);
  pwm.writeMicroseconds(servo09, degree_9);
  delay(10);
}
void S10(int x) {
  servo10PPos = x;
  degree_10 = map(servo10PPos, 0, 180, SERVOMIN, SERVOMAX);
  pwm.writeMicroseconds(servo10, degree_10);
  delay(10);
}

void Walk() {

  X(79, 117, 125, 74, 91, 87, 68, 59, 103, 92); //XY(244,7);  //Start
  X(79, 117, 125, 74, 100, 87, 68, 59, 103, 92); //XY(253,253);
  X(79, 117, 125, 74, 100, 87, 68, 59, 103, 102); //XY(0,250);

  X(79, 127, 125, 74, 100, 87, 68, 59, 103, 102); //XY(249,249);   //Step1
  X(79, 127, 125, 85, 100, 87, 68, 59, 103, 102); //XY(251,250);
  X(79, 127, 125, 85, 91, 87, 68, 59, 103, 102); //XY(250,252);
  X(79, 127, 125, 85, 91, 87, 68, 59, 103, 91); //XY(2,1);
  X(79, 127, 125, 80, 91, 87, 68, 59, 103, 91); //XY(255,1);
  X(79, 127, 125, 80, 77, 87, 68, 59, 103, 91); //XY(0,12);
  X(79, 127, 125, 80, 77, 84, 68, 59, 103, 85);
  X(79, 127, 125, 76, 77, 84, 68, 59, 103, 85);
  X(79, 117, 125, 76, 77, 84, 68, 59, 103, 85);
  X(79, 117, 125, 74, 77, 84, 68, 59, 103, 85);

  X(79, 117, 125, 74, 77, 84, 58, 59, 103, 85); //XY(245,9); //Step2
  X(79, 117, 125, 74, 77, 84, 58, 59, 93, 85); //XY(245,8);
  X(79, 117, 125, 74, 77, 84, 58, 59, 93, 92); //XY(246,10);
  X(79, 117, 125, 74, 77, 84, 58, 59, 93, 83);      //XY(244, 12);
  X(79, 117, 125, 74, 91, 84, 58, 59, 93, 83);     //XY(0, 4);
  X(79, 117, 125, 74, 91, 84, 58, 59, 93, 92);     //XY(2, 0);
  X(79, 117, 125, 74, 91, 84, 58, 59, 96, 92);     //XY(252, 1);
  X(79, 117, 125, 74, 101, 84, 58, 59, 96, 92);     //XY(250, 253);
  X(79, 117, 125, 74, 101, 84, 58, 59, 96, 102);     //XY(252, 249);
  X(79, 117, 125, 74, 101, 84, 68, 59, 96, 102);     //XY(9, 247);
  X(79, 117, 125, 74, 101, 84, 68, 59, 103, 102);

  X(79, 117, 125, 74, 91, 84, 68, 59, 103, 85);   //End
  X(79, 117, 125, 74, 91, 84, 68, 59, 103, 92);
  X(79, 117, 125, 74, 91, 87, 68, 59, 103, 92);
}
```

โปรแกรม Remote

```
#include <esp_now.h>
#include <WiFi.h>
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128 // pixel ความกว้าง
#define SCREEN_HEIGHT 64 // pixel ความสูง

#define OLED_RESET     -1 //ขา reset เป็น -1 ถ้าใช้ร่วมกับขา
Arduino reset
Adafruit_SSD1306 OLED(SCREEN_WIDTH, SCREEN_HEIGHT,
&Wire, OLED_RESET);

uint8_t broadcastAddress[] = {0x94, 0xB5, 0x55, 0xF3,
0x6F, 0x70}; //94:B5:55:F3:6F:70


bool S1A, S1B, S2A, S2B, S3A, S3B, S4A, S4B, S5A, S5B, S6A,
S6B, S7A, S7B, S8A, S8B, S9A, S9B, S10A, S10B;
int S[11][3];
int State[11];
int D[12] = {0, 79, 113, 125, 74, 91, 87, 72, 59, 103, 92};
int N[12] = {0, 79, 113, 125, 74, 91, 87, 72, 59, 103, 92};

typedef struct struct_message {
  int S[12] = {0, 79, 113, 125, 74, 91, 87, 72, 59, 103, 92};
} struct_message;

struct_message myData;

esp_now_peer_info_t peerInfo;

void OnDataSent(const uint8_t *mac_addr,
esp_now_send_status_t status) {
  /*
    Serial.print("\r\nLast Packet Send Status:\t");
    Serial.println(status == ESP_NOW_SEND_SUCCESS ?
"Delivery Success" : "Delivery Fail");
  */
}

void setup() {
  Serial.begin(115200);

  WiFi.mode(WIFI_STA);
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  esp_now_register_send_cb(OnDataSent);

  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

  if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Failed to add peer");
    return;
  }

  if (!OLED.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // สั่ง
ให้จอ OLED เริ่มทำงานที่ Address 0x3C
    Serial.println("SSD1306 allocation failed");
  } else {
    Serial.println("ArdinoAll OLED Start Work !!!");
  }

  pinMode(39, INPUT);
  pinMode(23, INPUT);
  pinMode(34, INPUT);
  pinMode(35, INPUT);
  pinMode(32, INPUT);
  pinMode(33, INPUT);
  pinMode(25, INPUT);
  pinMode(26, INPUT);
  pinMode(14, INPUT);
  pinMode(12, INPUT);
  pinMode(19, INPUT);
  pinMode(18, INPUT);
  pinMode(5, INPUT);
  pinMode(17, INPUT);
  pinMode(16, INPUT);
  pinMode(4, INPUT);
  pinMode(2, INPUT);
  pinMode(15, INPUT);
  pinMode(13, INPUT);
  pinMode(27, INPUT);

  OLED.clearDisplay(); // ลบภาพในหน้าจอทั้งหมด
  OLED.setTextColor(WHITE, BLACK);  //กำหนดข้อความสีขาว
ฉากหลังสีดำ
```

```
  OLED.setCursor(15, 20); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(2); // กำหนดขนาดตัวอักษร
  OLED.print("Working..");
  OLED.display(); // สั่งให้จอแสดงผล

/*
  X(79, 117, 125, 74, 91, 87, 68, 59, 103, 92); //XY(244,7);  //Start
  X(79, 117, 125, 74, 100, 87, 68, 59, 103, 92); //XY(253,253);
  X(79, 117, 125, 74, 100, 87, 68, 59, 103, 102); //XY(0,250);

  for (int i = 1; i <= 5; i++) {
    X(79, 127, 125, 74, 100, 87, 68, 59, 103, 102); //XY(249,249);    //Step1
    X(79, 127, 125, 85, 100, 87, 68, 59, 103, 102); //XY(251,250);
    X(79, 127, 125, 85, 91, 87, 68, 59, 103, 102); //XY(250,252);
    X(79, 127, 125, 85, 91, 87, 68, 59, 103, 91); //XY(2,1);
    X(79, 127, 125, 80, 91, 87, 68, 59, 103, 91); //XY(255,1);
    X(79, 127, 125, 80, 77, 87, 68, 59, 103, 91); //XY(0,12);
    X(79, 127, 125, 80, 77, 87, 68, 59, 103, 85);
    X(79, 127, 125, 80, 77, 84, 68, 59, 103, 85);
    X(79, 127, 125, 76, 77, 84, 68, 59, 103, 85);
    X(79, 117, 125, 76, 77, 84, 68, 59, 103, 85);
    X(79, 117, 125, 74, 77, 84, 68, 59, 103, 85);

    X(79, 117, 125, 74, 77, 84, 58, 59, 103, 85); //XY(245,9); //Step2
    X(79, 117, 125, 74, 77, 84, 58, 59, 93, 85); //XY(245,8);
    X(79, 117, 125, 74, 77, 84, 58, 59, 93, 92); //XY(246,10);
    X(79, 117, 125, 74, 77, 84, 58, 59, 93, 83);     //XY(244, 12);
    X(79, 117, 125, 74, 91, 84, 58, 59, 93, 83);     //XY(0, 4);
    X(79, 117, 125, 74, 91, 84, 58, 59, 93, 92);     //XY(2, 0);
    X(79, 117, 125, 74, 91, 84, 58, 59, 96, 92);     //XY(252, 1);
    X(79, 117, 125, 74, 101, 84, 58, 59, 96, 92);     //XY(250, 253);
    X(79, 117, 125, 74, 101, 84, 58, 59, 96, 102);    //XY(252, 249);
    X(79, 117, 125, 74, 101, 84, 68, 59, 96, 102);     //XY(9, 247);
    X(79, 117, 125, 74, 101, 84, 68, 59, 103, 102);
  }
  X(79, 117, 125, 74, 91, 84, 68, 59, 103, 85);   //End
  X(79, 117, 125, 74, 91, 84, 68, 59, 103, 92);
  X(79, 117, 125, 74, 91, 87, 68, 59, 103, 92);
*/

  /*
    X(79, 113, 125, 74, 91, 87, 72, 59, 103, 92);
    X(79, 113, 125, 74, 105, 87, 72, 59, 103, 92);

    X(79, 113, 125, 74, 105, 87, 72, 59, 103, 105);

    X(86, 130, 125, 74, 105, 87, 72, 59, 103, 105);

    X(86, 130, 125, 88, 105, 87, 72, 59, 103, 105);

    X(86, 130, 125, 88, 112, 87, 72, 59, 103, 105);

    X(86, 130, 125, 88, 112, 87, 72, 59, 103, 99);

    X(86, 130, 125, 88, 89, 87, 72, 59, 103, 99);

    X(86, 130, 125, 88, 90, 87, 72, 59, 103, 97);

    X(79, 130, 125, 88, 90, 87, 72, 59, 103, 97);

    X(79, 130, 125, 80, 90, 87, 72, 59, 103, 97);

    X(79, 130, 125, 80, 90, 82, 72, 59, 103, 97);

    //X(79, 113, 125, 74, 91, 87, 72, 59, 103, 92);

    X(79, 130, 125, 80, 90, 82, 72, 59, 103, 95);

    X(79, 130, 125, 86, 90, 82, 72, 59, 103, 95);

    X(79, 130, 125, 86, 76, 82, 72, 59, 103, 95);  //


    X(79, 130, 125, 86, 76, 82, 35, 59, 103, 95);

    X(79, 130, 125, 86, 76, 82, 35, 59, 103, 85);

    X(79, 130, 125, 86, 76, 82, 35, 59, 76, 85);

    X(79, 130, 125, 83, 76, 82, 35, 59, 76, 85);
```

```
   X(79, 130, 125, 83, 76, 82, 42, 59, 76, 85);     //XY(238,
13);
   X(79, 130, 125, 83, 76, 82, 42, 46, 76, 85);     //XY(241,
14);
   X(79, 130, 125, 83, 76, 82, 42, 46, 96, 85);     //XY(240,
14);

   X(79, 120, 125, 83, 76, 82, 42, 46, 96, 85);     //XY(22,
6);
   X(79, 120, 125, 83, 76, 82, 50, 46, 96, 85);     //XY(28,
6);
   X(79, 120, 125, 80, 76, 82, 50, 46, 96, 85);     //XY(24,
6);
   X(79, 120, 125, 80, 76, 82, 70, 46, 96, 85);     //XY(32,
6);
   X(79, 120, 125, 74, 76, 82, 70, 46, 96, 85);     //XY(23,
7);
   X(79, 113, 125, 74, 76, 82, 72, 46, 96, 85);     //XY(33,
7);
   X(79, 113, 125, 74, 76, 82, 72, 59, 96, 85);     //XY(33,
7);
   X(79, 113, 125, 74, 76, 82, 72, 59, 103, 85);    //XY(32,
7);
   X(79, 113, 125, 72, 76, 82, 72, 59, 103, 85);    //XY(18,
7);
   X(79, 113, 125, 74, 76, 82, 72, 59, 103, 85);    //XY(22,
7);
   X(79, 113, 125, 74, 76, 82, 72, 59, 103, 75);    //XY(22,
10);
   X(79, 113, 125, 74, 90, 82, 72, 59, 103, 75);    //XY(21,
1);
   X(79, 113, 125, 74, 90, 82, 72, 59, 103, 82);    //XY(20,
254);
   X(79, 113, 125, 74, 90, 95, 72, 59, 103, 82);    //XY(17,
3);
   X(79, 113, 125, 74, 90, 87, 72, 59, 103, 82);    //XY(20,
0);
   X(79, 113, 125, 74, 90, 87, 72, 59, 103, 92);    //XY(23,
254);//


    X(79, 130, 125, 83, 76, 82, 42, 46, 96, 72);    //XY(240,
13);
    X(79, 130, 125, 83, 81, 82, 42, 46, 96, 72);    //XY(241,
7);
    X(79, 130, 125, 83, 81, 82, 42, 46, 96, 85);    //XY(241,
255);
  */
}

void loop() {

  S[1][1] = digitalRead(39);
  S[1][2] = digitalRead(23);
  S[2][1] = digitalRead(34);
  S[2][2] = digitalRead(35);
  S[3][1] = digitalRead(32);
  S[3][2] = digitalRead(33);
  S[4][1] = digitalRead(25);
  S[4][2] = digitalRead(26);
  S[5][1] = digitalRead(14);
  S[5][2] = digitalRead(12);
  S[6][1] = digitalRead(19);
  S[6][2] = digitalRead(18);
  S[7][1] = digitalRead(5);
  S[7][2] = digitalRead(17);
  S[8][1] = digitalRead(16);
  S[8][2] = digitalRead(4);
  S[9][1] = digitalRead(2);
  S[9][2] = digitalRead(15);
  S[10][1] = digitalRead(13);
  S[10][2] = digitalRead(27);

  OLEDS();

 for (int i = 1; i <= 11; i++) {
   if (S[i][1] == HIGH) {
     D[i]++;
     myData.S[i] = D[i];
     esp_err_t result = esp_now_send(broadcastAddress,
(uint8_t *) &myData, sizeof(myData));
     Serial.print("Send Valur D");
     Serial.print(i);
     Serial.print(" 1");
     Serial.print(" is :");
     Serial.println(D[i]);
   }
```

```
    else if (S[i][2] == HIGH) {
      D[i]--;
      myData.S[i] = D[i];
      esp_err_t result = esp_now_send(broadcastAddress,
(uint8_t *) &myData, sizeof(myData));
      Serial.print("Send Valur D");
      Serial.print(i);
      Serial.print(" 2");
      Serial.print(" is :");
      Serial.println(D[i]);
    }
  }
  delay(50 );

}

void OLEDS() {
  OLED.clearDisplay(); // ลบภาพในหน้าจอทั้งหมด
  OLED.setTextColor(WHITE, BLACK);  //กำหนดข้อความสีขาว
ฉากหลังสีดำ

  OLED.setCursor(0, 0); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S1 :  ");
  OLED.println(D[1]);

  OLED.setCursor(0, 14); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S2 :  ");
  OLED.println(D[2]);

  OLED.setCursor(0, 28); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S3 :  ");
  OLED.println(D[3]);

  OLED.setCursor(0, 42); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S4 :  ");
  OLED.println(D[4]);

  OLED.setCursor(0, 56); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S5 :  ");
  OLED.println(D[5]);

  OLED.setCursor(64, 0); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S6 :  ");
  OLED.println(D[6]);

  OLED.setCursor(64, 14); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S7 :  ");
  OLED.println(D[7]);

  OLED.setCursor(64, 28); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S8 :  ");
  OLED.println(D[8]);

  OLED.setCursor(64, 42); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S9 :  ");
  OLED.println(D[9]);

  OLED.setCursor(64, 56); // กำหนดตำแหน่ง x,y ที่จะแสดงผล
  OLED.setTextSize(1); // กำหนดขนาดตัวอักษร
  OLED.print("S10 : ");
  OLED.println(D[10]);

  OLED.display(); // สั่งให้จอแสดงผล
}

void SEND() {
  for (int i = 1; i <= 11; i++) {
    while (1) {
      if (N[i] > D[i]) {
        N[i]--;
        myData.S[i] = N[i];
        esp_err_t result =
esp_now_send(broadcastAddress, (uint8_t *) &myData,
sizeof(myData));
      }
      else if (N[i] < D[i]) {
        N[i]++;
        myData.S[i] = N[i];
        esp_err_t result =
esp_now_send(broadcastAddress, (uint8_t *) &myData,
sizeof(myData));
      }
      else if (N[i] == D[i]) {
        N[i] = D[i];
        myData.S[i] = N[i];
```

```
        esp_err_t                result                =
esp_now_send(broadcastAddress,  (uint8_t  *)  &myData,
sizeof(myData));
        break;
      }
      delay(45);
    }
  }
}

void X(int s1, int s2, int s3, int s4, int s5, int s6, int s7, int
s8, int s9, int s10) {
  D[1] = s1;
  D[2] = s2;
  D[3] = s3;
  D[4] = s4;
  D[5] = s5;
  D[6] = s6;
  D[7] = s7;
  D[8] = s8;
  D[9] = s9;
  D[10] = s10;

  SEND();
}
```