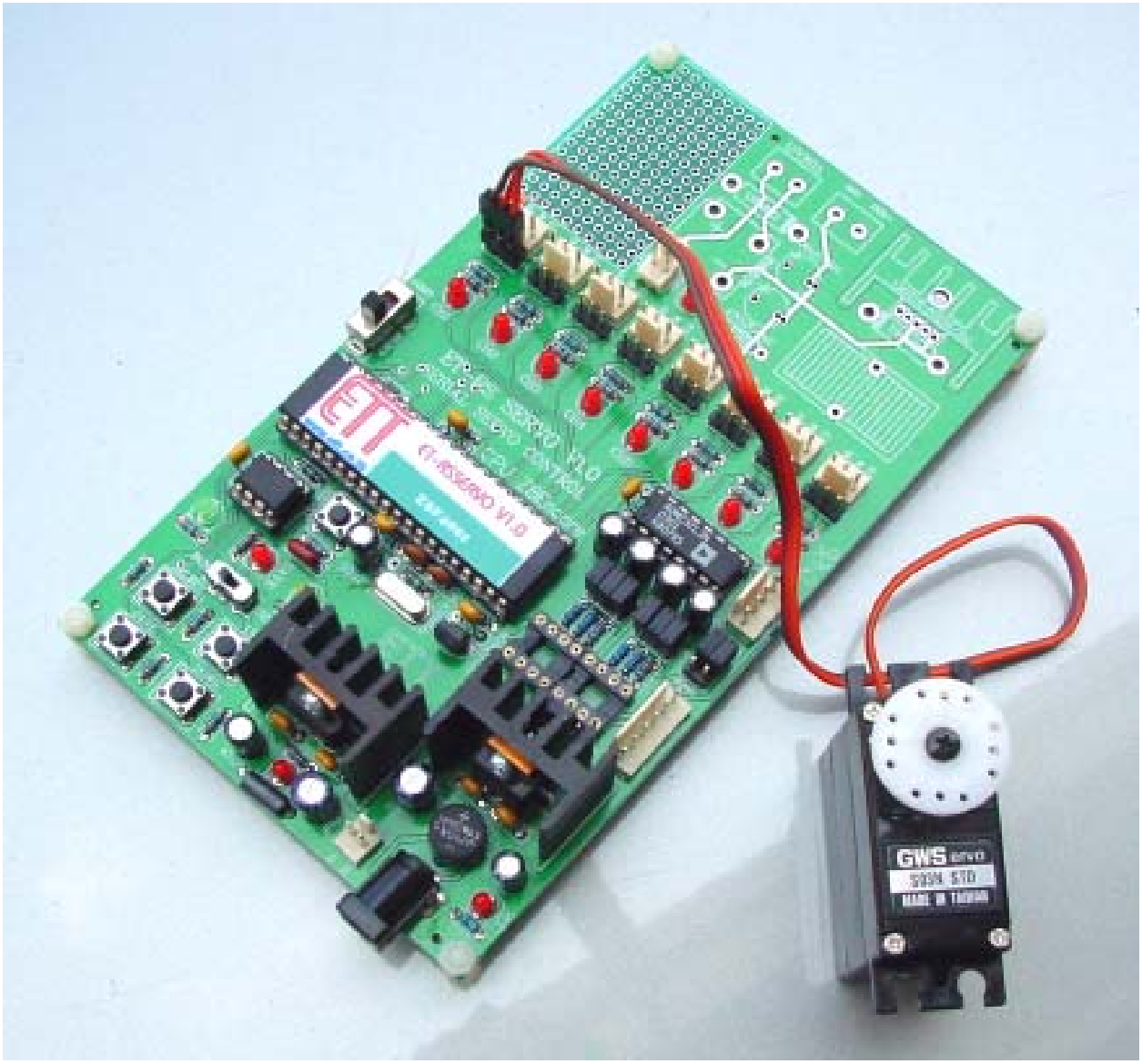
# คู่มือการใช้งานบอร์ด

# *ET-RS SERVO V1.0*

## ควบคุม SERVO MOTOR




ETT
www.ett.co.th

บริษัท อีทีที จำกัด ETT CO., LTD.

1112/96-98 ถนนสุขุมวิท แขวงพระโขนง เขตคลองเตย กรุงเทพฯ 10110 http://www.etteam.com
1112/96-98 Sukhumvit Rd., Phrakanong Klongtoey Bangkok 10110 http://www.ett.co.th
Tel : 02-7121120 Fax : 02-3917216 email : sale@etteam.com

# คำนำ

หนังสือคู่มือเล่มนี้ จัดทำขึ้นเพื่อใช้สำหรับเป็นคู่มือประกอบการใช้งานบอร์ด ควบคุม Servo motor รุ่น ET-RS SERVO V1.0 ที่ทางบริษัท ETT เป็นผู้จัดทำขึ้น โดยลักษณะของตัวบอร์ดจะใช้ MCU ตระกูล Z8Encore! เบอร์ Z8F4801 เป็นตัวประมวลผล และ รับข้อมูลจาก คอมพิวเตอร์ หรือ MCU ตระกูลอื่นๆ ที่ต่อจากภายนอกผ่าน พอร์ต RS232,422,485 ของตัวบอร์ด เข้ามาประมวลผลเพื่อส่งไปควบคุม Servo motor อีกต่อหนึ่ง โดยในบอร์ดควบคุม Servo motor นี้สามารถต่อ Servo motor ได้ทั้งหมด 8 ตัว และ ยังสามารถนำบอร์ดรุ่นเดียวกันนี้มาต่อขนานกันได้อีกเพื่อเพิ่มจำนวน Servo motor มีฟังก์ชันการทำงานต่างๆ ไว้ให้ผู้ใช้สำหรับใช้ควบคุม Servo motor ได้อย่างสะดวก อาทิ เช่น ฟังก์ชันการ Calibrate หาตำแหน่ง ค่า Min และ Max ของ Servo motor ที่ผู้ใช้นำมาทดลอง พร้อมกับแสดงตำแหน่งการหมุนผ่านทางโปรแกรม Procomm หรือ Hyper Terminal เป็นต้น

เนื้อหาภายในคู่มือนี้จะอธิบายถึงการทำงานและขอบเขตการทำงานของบอร์ด,การใช้งานในฟังก์ชันต่างๆที่บอร์ดนี้สามารถทำได้ รวมทั้ง ตัวอย่างโปรแกรม ในการส่งผ่านข้อมูลเพื่อใช้ควบคุม Servo motor ผ่านทาง RS232 , RS422/485 ของไมโครคอนโทรลเลอร์อีกตัวหนึ่ง ผู้เขียนหวังเป็นอย่างยิ่งว่าคู่มือเล่มนี้จะช่วยให้ท่านสามารถใช้งานบอร์ดควบคุม Servo motor ได้เป็นอย่างดี

**ทีมงานอีทีที**

# สารบัญ

# ET-RS SERVO V1.0

## คุณสมบัติและลักษณะโครงสร้างของบอร์ด ET-RS SERVO V1.0

**- ลักษณะทั่วไปของบอร์ด**

บอร์ด ET-RS SERVO V1.0 เป็นบอร์ดสำหรับคอนโทรล Servo Motor โดยการรับคำสั่งควบคุมมาจากภายนอกบอร์ด ผ่านระบบการสื่อสารแบบอนุกรม RS232,RS422 และ RS485 ตามความสะดวกของผู้ใช้ที่จะเลือกใช้งาน โดยคำส่งที่ส่งมาให้ตัวบอร์ดจะมีด้วยกัน 3 ไบต์ ได้แก่ Sync Byte จะมีค่าเป็น $FF_H$ ตามด้วยไบต์ของแชนแนล และไบต์ของตำแหน่ง Servo

| Byte1 | Byte2 | Byte3 |
|---|---|---|
| [Sync byte ($FF_H$)] | [ID Address (0-$F_H$) + Channel servo (1-$8_H$)] | [Position (50-250)] |

ในบอร์ดรุ่นนี้จะใช้ไมโครคอนโทรลเลอร์ตระกูล Z8Encore! เบอร์ Z8F4801 เป็นตัวประมวลผลในการรับคำสั่งที่ส่งมาจากภายนอกเข้ามาประมวลผลแล้วจึงส่งพัลซ์ออกไปควบคุม Servo ตามที่ผู้ใช้ต้องการ โดยBaud Rate ในการรับส่งข้อมูลนั้นจะถูกกำหนดไว้ที่ 9600 Hz/s ตายตัว

บอร์ดคอนโทรล Servo Motor รุ่นนี้จะออกแบบวงจรให้มีส่วนของแชนแนลที่ใช้สำหรับต่อ Servo ออกไปใช้งานได้ถึง 8 แชนแนล ซึ่งผู้ใช้สามารถเลือกใช้งานแชนแนลไหนก็ได้ และยังมีในส่วนของแหล่งจ่ายไฟเสริมที่ผู้ใช้สามารถต่อจากภายนอกเข้ามาได้ สำหรับเลี้ยง Servo โดยตรง ซึ่งจะมีประโยชน์โดยผู้ใช้จะสามารถนำ Servo ที่กินกระแสมากมาต่อทดลองได้ (ไม่เกิน3A) เมื่อไฟเลี้ยง Servo จากบอร์ดไม่พียงพอ และยังมี $E^2$PROMPT ไว้สำหรับเก็บค่า Step การหมุน,เก็บค่า Min,Max และ ID Address ของบอร์ดไว้ได้เพื่อใช้งานในครั้งต่อไป โดยผู้ใช้จะได้ไม่ต้องเสียเวลาในการ Setup ตัวบอร์ดและตัว Servo อีก ซึ่งในการ Setup ตัว Servo ก่อนการใช้งานนี้จะมีประโยชน์มาก คือเป็นการป้องกันความเสียหายที่จะเกิดขึ้นกับตัว Servo ในกรณีที่เราจ่ายพัลซ์ให้ตัว Servo เกินจุดที่ Servo จะหมุนไปได้

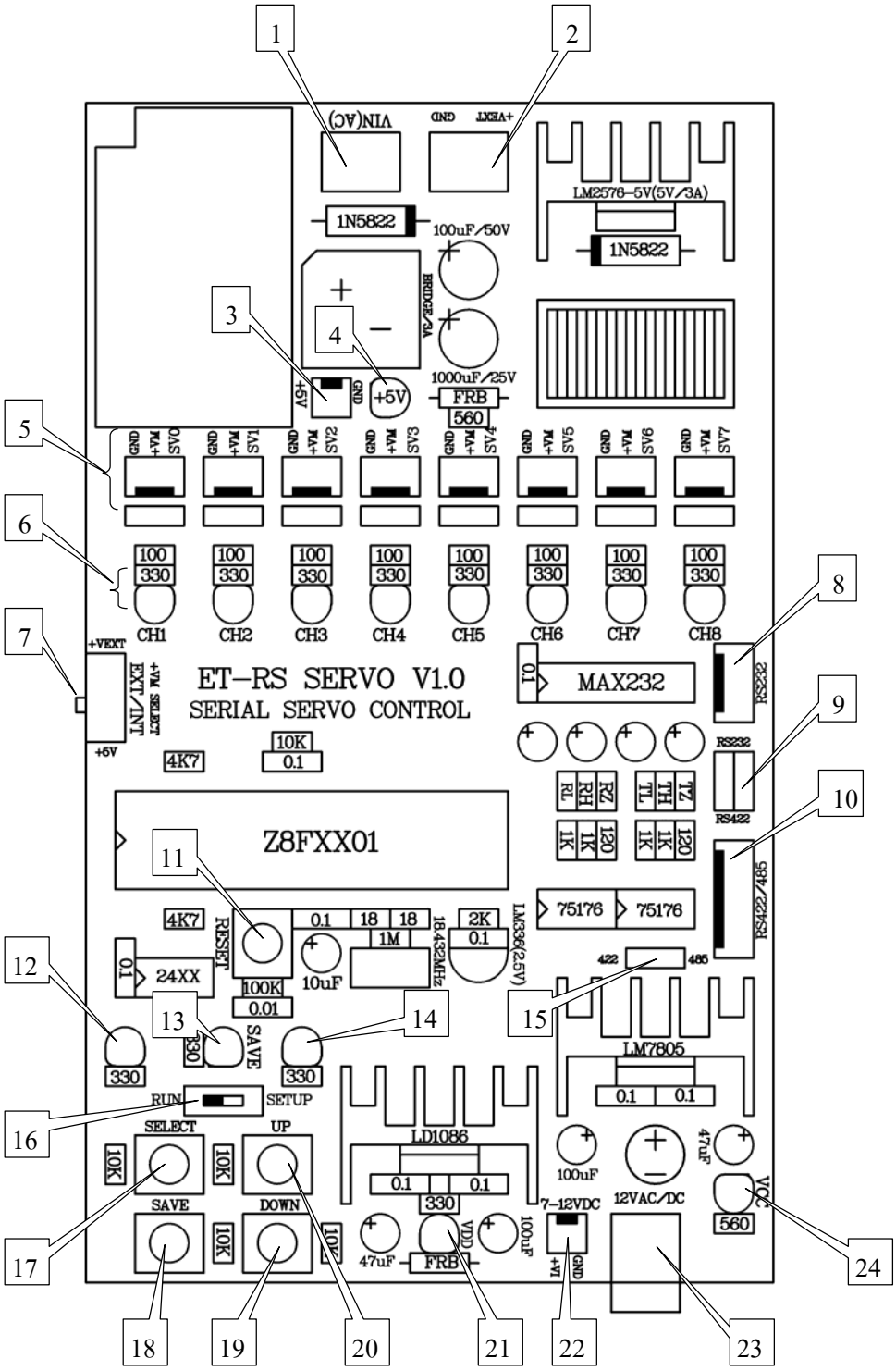
ในส่วนของการส่งผ่านคำสั่งมาควบคุม Servo นั้นตัวบอร์ดจะออกแบบให้มีการสื่อสารแบบ อนุกรม โดยสามารถต่อใช้งานได้ทั้งแบบ RS232 ,RS422 และ RS485 โดยในส่วนของ RS485 นี้ผู้ใช้สามารถที่จะนำบอร์ดรุ่นเดียวกันนี้มาต่อขนานกันได้ถึง 16 บอร์ด โดยจะต้องกำหนดค่า ID Address ของแต่ละบอร์ดให้ต่างกัน แล้วสามารถใช้ตัวส่งคำสั่งเพียงตัวเดียว เลือกส่งคำสั่งมาควบคุม Servo ที่ต่ออยู่ในบอร์ดไหนก็ได้

นอกจากส่วนประกอบต่างๆของบอร์ดที่ได้กล่าวไปแล้ว ในส่วนของการแสดงผล จะแสดงผลการทำงานในแต่ละโหมดการทำงานด้วย LED ทั้งหมดซึ่งสามารถเข้าใจได้ง่าย และยังแสดงผลการทำงานเป็นข้อความให้เห็นในบางโหมดการทำงานได้ด้วย โดยผู้ใช้จะต้อง ต่อสายจาก RS232,422 หรือ 485 ของบอร์ดไปเข้าที่Port COM1 หรือ COM2 ของ เครื่อง PC แล้วจึงใช้โปรแกรม PROCOMM หรือ Hyper Terminal เปิดดูข้อความเหล่านั้นได้ โดยกำหนด Baud Rate ของโปรแกรม PROCOMM หรือ Hyper Terminalไปที่ 9600 bit/s

**- คุณสมบัติของบอร์ด ET-RS SERVO V1.0**

- สามารถต่อ Servo motor ได้ 8 ตัว/บอร์ด
- ทำการรับส่งข้อมูลแบบอนุกรมโดยใช้การสื่อสารด้วย RS232 ,422 ,485
- ใช้ MCU Z8Encore! เบอร์ Z8F4801 ในการประมวลผล และรับส่งข้อมูลจากภายนอก
- แสดงผลการทำงานในโหมดฟังก์ชันต่างๆด้วย LED
- แสดงตำแหน่งการหมุนของ Servo Motor ผ่านทาง โปรแกรม Procomm หรือ Hyper Terminal
- Step ในการหมุนสามารถกำหนดได้ในช่วง 50-250 Step
- ความละเอียดต่อการหมุน 1 Step = 10 us
- Step ต่ำสุดคือ 50 จะจ่าย Pulse ได้กว้าง 500us และ Step สูงสุดคือ 250 จ่าย Pulse ได้กว้าง 2.5 ms
- Broad Rate ในการรับส่งข้อมูลผ่านทาง RS 232 ,422 ,485 = 9600 bit/s
- สามารถ Save ค่า Min และ Max ของ Servo Motor ในแต่ละแชนแนลการทำงานได้
- สามารถบันทึก Step การหมุน และ Delay Time ในแต่ละ Step แต่ละแชนแนลได้
- สามารถ Run Step ที่บันทึกเก็บไว้ภายในตัวบอร์ดได้ เพื่อดูรูปแบบการหมุนของ Servo
- สามารถหาตำแหน่ง Center ของ Servo motor แต่ละตัวได้อัตโนมัติ
- มีโหมดการทำงานสำหรับให้ผู้ใช้ Calibrate หาค่า Min และ Max ของ Servo Motor
- มีโหมดสำหรับ คอนโทรล Servo motor โดยใช้ไมโครคอนโทรลเลอร์ตระกูลต่างๆ เขียนโปรแกรมควบคุมผ่านทาง พอร์ต Uart ด้วยการสื่อสารแบบ RS232,422,485 ได้
- มีโหมด สำหรับคอนโทรล Servo motor ครั้งละ1 แชนแนล โดยใช้โปรแกรมสำเร็จรูป ET-RSS V1.00 ผ่านทางหน้าจอของเครื่อง PC ได้
- ในส่วนของการสื่อสารแบบ RS485 สามารถที่จะนำบอร์ดรุ่นเดียวกันนี้ต่อขนานกันได้ 16 บอร์ด แล้วเลือกควบคุม Servo ที่ต่ออยู่ของแต่ละบอร์ดได้ โดยการกำหนด ID Address ที่ต่างกันให้แต่ละบอร์ดซึ่งค่า ID Address ที่สามารถกำหนดได้จะอยู่ในช่วง $0_H$- $F_H$
- สามารถต่อแหล่งจ่ายไฟจากภายนอกได้ทั้ง AC และ DC (9-12V) เพื่อใช้ในการขับ Servo Motor ในกรณีที่แหล่งจ่ายไฟเลี้ยงของบอร์ดจ่ายกระแสไปขับ Servo Motor ได้ไม่เพียงพอ

**- โครงสร้างและส่วนประกอบของบอร์ด ET-RS SERVO V1.0**




**รูป แสดงส่วนประกอบของบอร์ด ET-RS SERVO V1.0**

*หน้าที่ของส่วนประกอบต่างๆ*

- หมายเลข **1** : เป็นขั้วสำหรับใช้ต่อแรงดัน VAC (9-12V) จากภายนอกเข้ามายังบอร์ด เพื่อเป็นไฟเลี้ยงให้กับ Servo กรณีที่ ไฟเลี้ยงภายในบอร์ดจ่ายกระแสให้ Servo ไม่เพียงพอ
- หมายเลข **2** : มีหน้าที่เช่นเดียวกับขั้วหมายเลข1 แต่ขั้วนี้จะรับแรงดัน VDC(9-12V) ดังนั้นในการต่อผู้ใช้จะต้องเลือกว่าจะจ่ายแรงดัน AC หรือ DC อย่างใดอย่างหนึ่งเท่านั้น และต่อให้ถูกขั้ว ระหว่างขั้ว AC และ DC ด้วย
- หมายเลข **3** : ขั้วสำหรับใช้ต่อแรงดัน VDC +5V ที่เกิดจากการป้อนแรงดันจากภายนอกเข้ามาทางขั้วหมายเลข 1 หรือ หมายเลข 2 ไปใช้งานอื่นๆ ตามความต้องการของผู้ใช้ หรือในกรณีที่ ไม่ได้ต่อไฟเข้าที่ขั้วหมายเลข 1 หรือ 2 อาจจะต่อไฟ VDC +5 V เข้าที่ขั้วนี้เพื่อเลี้ยง Servo โดยตรงเลยก็ได้
- หมายเลข **4** : LED(RED) แสดงการทำงานของภาคจ่ายไฟที่รับแรงดัน Input เข้ามาทางขั้วหมายเลข 1(AC) หรือ ขั้วหมายเลข 2(DC)
- หมายเลข **5** : เป็นขั้วสำหรับใช้ต่อ Servo Motor โดยจะมี 2 ลักษณะให้เลือกใช้ ซึ่งขั้วต่อนี้จะมีด้วยกัน 3 ขา ประกอบด้วย GND ,+VCC,Control โดยสามารถต่อ Servo ได้ทั้งหมด 8 ตัวที่ควบคุมได้เป็นอิสระกัน
- หมายเลข **6** : LED(RED) แสดงการทำงานของ Servo Motor แต่ละตัว ซึ่งมีทั้งหมด 8 แชนแนล โดยถ้ามอเตอร์ที่ต่ออยู่แชนแนลไหนถูกเลือกให้ทำงาน LED ของแชนแนลนั้นก็จะติดค้างอยู่ จนกว่าจะมีการเปลี่ยนแชนแนลใหม่ LED แชนแนลเก่าก็จะดับลง แชนแนลใหม่ที่เลือกก็จะติดแทน
- หมายเลข **7** : สวิตช์ทำหน้าที่ในการเลือกแหล่งจ่ายไฟให้กับ Servo Motor คือ เลื่อน SW มาทาง INT เป็นการเลือกแหล่งจ่ายจากไฟเลี้ยงของบอร์ด ให้กับ Servo ถ้า เลื่อน SW มาทาง EXT จะเป็นการเลือกแหล่งจ่ายที่ต่อเข้ามาทางขั้วหมายเลข 1 หรือ หมายเลข 2 ให้กับ Servo Motor
- หมายเลข **8** : พอร์ต RS232 ใช้ติดต่อสื่อสารแบบอนุกรมกับ PC หรือ MCU เพื่อที่จะรับส่งข้อมูลในการคอนโทรล Servo ให้ทำงานตามที่เราต้องการจากภายนอกบอร์ดได้
- หมายเลข **9** : Jumper สำหรับเลือกการสื่อสารของบอร์ด ว่าผู้ใช้ต้องการเลือกการสื่อสารในการรับส่งข้อมูลด้วย RS232 หรือ RS422 /485
- หมายเลข **10** : พอร์ต RS422 และ RS485 ใช้ติดต่อสื่อสารแบบอนุกรมกับ PC หรือ MCU เพื่อที่จะคอนโทรล Servo ให้ทำงานตามที่เราต้องการจากภายนอกบอร์ดได้
- หมายเลข **11** : สวิตช์สำหรับ Reset MCU ในกรณีที่ บอร์ดทำงานผิดพลาด หรือต้องการให้ MCU เริ่มต้นการทำงานใหม่
- หมายเลข **12** : LED(Green) แสดงการทำงานของ Mode RUN

- หมายเลข **13** : LED(RED) แสดงการทำงานของ SW. Save คือ เมื่อกด SW. Save LED นี้จะติดและดับลง (เมื่ออยู่ใน Mode Setup) รวมทั้งจะติดพร้อมกับ LED หมายเลข 12 ด้วยเมื่อเข้าสู่โหมด RUN เพื่อแสดงการทำงานในส่วนของ การรับส่ง Command จากบอร์ดไมโครคอนโทรลเลอร์ที่อยู่ภายนอก
- หมายเลข **14** : LED(Yellow) แสดงการทำงานของโหมด Setup รวมทั้งจะติดพร้อมกับ LED หมายเลข 12 ด้วยเมื่อเข้าสู่โหมด RUN เพื่อแสดงการทำงานในส่วนของการคอนโทรล Servo ด้วยโปรแกรม ET-RSS V1.00 ผ่านทาง COM Port ของ PC
- หมายเลข **15** : Jumper สำหรับเลือกการสื่อสารของบอร์ดว่าต้องการรับส่งข้อมูลด้วย RS422 หรือ RS485 ในการกำหนด Jumper นี้ จะมีผลเมื่อเลือก Jumper หมายเลข 9 มาทางด้าน RS422
- หมายเลข **16** : สวิตช์เลื่อน สำหรับเลือกโหมดการทำงาน คือ เลื่อนมาทาง SETUP เป็นการกำหนดค่า Min,Max ให้กับ Servo Motor ในแต่ละแชนแนล และกำหนด ID Address ให้กับบอร์ด เลื่อนมาทาง RUN จะเป็นการคอนโทรล Servo Motor โดยรับคำสั่งผ่านทางพอร์ตอนุกรม RS232,422,485
- หมายเลข **17** : สวิตช์ SELECT เมื่ออยู่ในโหมด SETUP จะใช้เลือกแชนแนลของ Servo ที่เราต้องการจะกำหนดค่า Min , Max รวมทั้งเลือกการ Set ID Address ให้กับบอร์ด และเมื่ออยู่ในโหมด RUN สวิตช์นี้จะทำหน้าที่ให้ผู้ใช้เลือก Run Servo ครั้งละแชนแนลโดยใช้โปรแกรม ET-RSS V1.00
- หมายเลข **18** : สวิตช์ Save ทำหน้าที่ Save ค่า Min ,Max และค่า ID Address เมื่ออยู่ในโหมด Setup และ Save ค่าตำแหน่ง Servo เมื่ออยู่ในโหมด Run เก็บไว้ใน $E^2$Prompt
- หมายเลข **19** : สวิตช์ DOWN ถ้าอยู่ในโหมด SETUP จะทำหน้าที่ลดค่า Step การหมุนของ Servo ลงมาครั้งละ 1 Step (จาก Step 250-50) หรือทำการ ลดค่า IDAddress ลงมาครั้งละ 1 Step (จากค่า $F_H$-0 ) , ถ้าอยู่ในโหมด RUN สวิตช์นี้จะทำหน้าที่ให้ผู้ใช้เลือก Run Servo โดยใช้ MCU เป็นตัวส่ง Command จากภายนอกบอร์ดมาควบคุม
- หมายเลข **20** : สวิตช์ UP ถ้าอยู่ในโหมด SETUP จะทำหน้าที่เพิ่มค่า Step การหมุนของ Servo ขึ้นมาครั้งละ 1 Step (จาก Step 50-250) หรือทำการ เพิ่มค่า ID Address ขึ้นมาครั้งละ 1 Step (จากค่า 0-$F_H$ ) , ถ้าอยู่ในโหมด RUN สวิตช์นี้จะทำหน้าที่ให้ผู้ใช้เลือกสำหรับ Run Step ของ Servo ที่ได้ Save ไว้ใน $E^2$Prompt
- หมายเลข **21** : LED(RED) แสดงการทำงานของแหล่งจ่ายไฟเลี้ยง MCU 3.3 V
- หมายเลข **22** : ขั้วสำหรับต่อแรงดัน VDC +5V ที่เกิดจากไฟเลี้ยงบอร์ด ไปใช้งานส่วนอื่นๆได้
- หมายเลข **23** : Jack สำหรับต่อแรงดันไฟ 9-12VAC/DC จากภายนอกเข้ามาเพื่อเลี้ยงวงจร
- หมายเลข **24** : LED(RED) แสดงการทำงานของแหล่งจ่ายไฟเลี้ยงวงจร 5 V

**การทำงานของบอร์ด ET-RS SERVO V1.0**

ในการทำงานของบอร์ดนั้น เมื่อเราจ่ายไฟให้กับบอร์ดที่ขั้วหมายเลข 23 ตัวโปรแกรมที่เราเขียนไว้ใน MCU ก็จะเริ่มทำงานโดยการส่งพัลซ์ที่มีค่าความกว้าง 1.5 ms ออกไปยังขั้วต่อ Servo แชนแนล 1-8 ซึ่งค่าความกว้างของพัลซ์นี้ จะเป็นค่าที่ทำให้ Servo หมุนมาอยู่ที่ตำแหน่ง Center ดังนั้น เมื่อผู้ใช้นำ Servo มาต่อที่แชนแนล 1-8 Servoทุกตัวก็จะหมุนมาอยู่ที่ตำแหน่ง Center อัตโนมัติ จากนั้นให้ผู้ใช้ทำการเลื่อน SW.หมายเลข 16 เพื่อเลือกโหมดการทำงานโดยโหมดการทำงานจะแบ่งออกเป็น 2 โหมดใหญ่ๆ ดังนี้

1.โหมด SETUP โดยในโหมดนี้ก็จะแบ่งเป็น
   1.1) การกำหนดค่า Min และ Max ให้กับ Servo แต่ละตัวในแต่ละแชนแนล
   1.2) การกำหนดค่า ID Address ให้กับบอร์ดโดยเลือกได้ตั้งแต่ 0-F (ฐาน16)
2.โหมด RUN โดยในโหมดนี้จะเป็นการสื่อสารผ่าน RS232,422,485 แบ่งเป็น
   2.1) การคอนโทรล Servo ผ่านทาง MCU ตระกูลต่างๆ
   2.2) การคอนโทรล Servo ครั้งละ 1 แชนแนล จากหน้าจอ PC โดยใช้โปรแกรมสำเร็จรูป ET-RSS V1.00
   2.3) การ RUN Step Servo ที่ Save ไว้ใน $E^2$Prompt โดยตรงภายในบอร์ด

ซึ่งในการใช้งานบอร์ดครั้งแรกจะต้องเลื่อน SW.มายังตำแหน่ง SETUP (LEDสีเหลืองจะติดแสดงการทำงานของโหมดนี้) เพื่อทำการกำหนด ค่า Min และ Max ให้กับ Servo และค่า ID Address ให้กับบอร์ด สาเหตุที่ต้องกำหนดค่า Min และ Max ให้กับ Servo ก็เพื่อป้องกัน Servo เสียหายในกรณีที่เรากำหนด Step การหมุนเกินระดับที่ Servo จะหมุนได้ซึ่งก็จะทำให้เฟืองภายใน Servo เสียหายได้ และทุกๆครั้งที่เราเปลี่ยน Servo ไปต่อในแชนแนลอื่นที่ยังไม่ได้กำหนดค่า Min,Max ให้กับ Servo ตัวที่เราจะใช้งาน ก็ควรจะกำหนดใหม่ เพราะระดับค่า Min,Max ของ Servo แต่ละตัวจะไม่เท่ากัน แต่ถ้าเรายังคงใช้แชนแนลเดิมกับตัว Servo ที่เราได้ Save ค่าทั้ง 2 นี้ไว้แล้วก็ไม่จำเป็นต้องกำหนดใหม่ ส่วนค่า ID Address ของบอร์ดกำหนดเพียงครั้งแรกครั้งเดียวก็ได้ หรือจะเปลี่ยนก็ได้ขึ้นอยู่กับผู้ใช้งานแต่ต้องจำค่า ID Address นี้ไว้ด้วย เพื่อนำไปใช้ในโหมด RUN ในการส่ง Command ควบคุม Servo ในทุกๆครั้งที่เราเข้ามาในโหมด SETUP นี้ แชนแนล 1 จะถูกเลือกเสมอ สังเกตที่ LED CH1 จะติด ถ้าผู้ใช้จะ SETUP แชนแนลอื่นก็ให้ทำการเปลี่ยนแชนแนลได้โดยกดที่ SW. SELECT ไปเรื่อยๆ แชนแนลก็จะเปลี่ยนไปและวนกลับมาเริ่มที่แชนแนล 1 ใหม่ และเราสามารถดูได้ว่าขณะที่เรากด SW. UP หรือ DOWN นั้น ตำแหน่งของ Servo เปลี่ยนไปอยู่ที่ตำแหน่งเท่าใหร่ โดยการ ต่อสาย RS232,422 หรือ 485 ของบอร์ดเข้าที่คอมพอร์ตของ PC แล้วเปิดโปรแกรม Procomm หรือ Hyper ขึ้นมา(Set Baud Rate = 9600) เราก็จะเห็นค่าของ Step เปลี่ยนไปตามการกดสวิตช์ของเรา

หลังจากที่เรา Set ค่าให้กับบอร์ดเรียบร้อยแล้วก็ให้เลื่อนสวิตช์หมายเลข 16 มายังโหมด RUN LED สีเขียวจะติดแสดงการทำงานในโหมดนี้ จากนั้น LED สีแดงก็จะติดตามมา นั่นคือทุกๆครั้งที่เราเข้ามาในโหมดนี้ มันจะถูกกำหนดให้เข้าสู่โหมดย่อย ในโหมดของการคอนโทรล Servo ผ่านทาง MCU โดยอัตโนมัติ และผู้ใช้สามารถเปลี่ยนให้เข้าสู่การทำงานอีก2โหมดย่อยได้คือ การคอนโทรล Servo โดยใช้โปรแกรม ET-RSS V1.00ผ่านทาง Comport ของ PC โดยการ กด SW.SELECT (LEDสีเหลืองกับเขียวจะติดแสดงการทำงานในโหมดย่อยนี้) และ การคอนโทรล Servo โดยการ Run Step การหมุนของ Servo ที่ Save ไว้ใน $E^2$Prompt โดยการกด SW. UP (LED สีเขียว สีแดง และสีเหลือง จะติดพร้อมกัน3 ดวงแสดงการทำงานในส่วนนี้) สรุปก็คือ ในโหมด RUN นี้เราจะใช้ SW 3 ตัวในการเปลี่ยนโหมดย่อยดังนี้

- *Switch UP* สำหรับเลือกโหมด คอนโทรล Servo โดยการ RUN Step การหมุนของ Servo ที่ทำการ Save ไว้ใน $E^2$Prompt (LED เขียว,แดง , เหลือง ติด)
- *Switch DOWN* สำหรับเลือกโหมดคอนโทรล Servo โดยใช้ MCUจากภายนอกในการส่ง Command Code มาควบคุม Servo (LED เขียว,แดง ติด)
- *Switch SELECT* สำหรับเลือกโหมดคอนโทรล Servo โดยใช้โปรแกรม ET-RSS V1.00 ในการควบคุมการหมุนของ Servo (LED เขียว,เหลือง ติด) และในโหมดนี้สามารถใช้ SW.Save save ค่าตำแน่งการหมุนของ Servo แต่ละแชนแนลเก็บไว้ได้ เพื่อทดสอบ Step การหมุนของ Servo ในเบื้องต้นได้

ไม่ว่าผู้ใช้จะทำงานอยู่ในโหมดการทำงานใดก็ตามสามารถเปลี่ยนโหมดการทำงานไปอีกโหมดหนึ่งได้โดยการเลื่อนสวิตช์ไปยังตำแหน่งของโหมดการทำงานนั้นได้ โดยไม่ต้องทำการ Reset MCU ใหม่ ส่วนในโหมดการทำงานย่อยก็เช่นกันสามารถเปลี่ยนโหมดการทำงานได้แต่ต้องเลื่อนสวิตช์(หมายเลข16) เปลี่ยนโหมดการทำงานหลักที่เราจะใช้งานเสียก่อนแล้วจึงเปลี่ยนการทำงานในโหมดย่อยนั้นๆได้ โดยการกดสวิตช์ตามที่กล่าวไปข้างต้น

ในบอร์ดนี้ยังมีภาคจ่ายไฟเสริม สำหรับให้ผู้ใช้ป้อนไฟ DC หรือ AC 9-12 V จากภายนอกเข้ามาเพื่อใช้สำหรับเลี้ยง Servo โดยตรง เมื่อไฟเลี้ยงบอร์ดไม่สามารถจ่ายไฟเลี้ยงให้กับ Servo ได้เพียงพอ ซึ่งจะช่วยให้ผู้ใช้สามารถนำ Servo ที่ใช้กระแสสูงมาต่อทดลองใช้งานได้แต่ต้องไม่เกิน 3 A เนื่องจากภาคจ่ายไฟเสริมนี้ได้ออกแบบให้ทนกระแสได้ประมาณนี้ เมื่อผู้ใช้จะใช้ภาคจ่ายไฟเสริมจะต้อง เลื่อนสวิตช์หมายเลข 7 มาทางด้าน EXT ซึ่งจะเป็นการตัดไฟเลี้ยง Servo จากตัวบอร์ดทิ้ง และจะต่อไฟเลี้ยง Servo เข้ากับ ชุดแหล่งจ่ายไฟเสริมแทน จากนั้นจึงทำการป้อนไฟเข้าที่ขั้วหมายเลข 1(AC) หรือ หมายเลข 2 (DC)

**<u>หมายเหตุ</u>** แหล่งจ่ายไฟเสริมนี้จะเป็น Option พิเศษของบอร์ด โดยปกติจะไม่ต่ออุปกรณ์ในภาคนี้ให้ไป ถ้าผู้ใช้ต้องการใช้งานโดยไม่สั่งให้ประกอบ Option นี้เพิ่ม อาจจะใช้แหล่งจ่ายไฟ DC หรือ แบตเตอรี่ +5 V มาต่อเข้าที่คอนเนคเตอร์ หมายเลข3 แทนก็ได้ ตามขั้วที่แสดงบนบอร์ด

## การใช้งานบอร์ด ET-RS SERVO V1.0

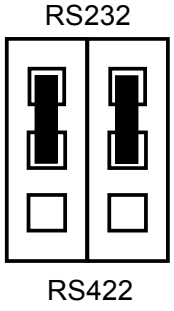
ในการเริ่มต้นใช้งานบอร์ดครั้งแรกผู้ใช้จะต้องทำการ SETUP Servo ในแชนแนลที่นำ Servo มาต่อ (ให้ Save ค่า Min,Max) และ SETUP ID Address ให้กับบอร์ดด้วย โดยเลือกไปที่โหมด SETUP มิฉะนั้นเวลาเข้าสู่โหมด RUN จะทำให้การส่งคำสั่งควบคุม Servo ผิดพลาด หรือไม่ทำงานได้ เนื่องจากในโหมด RUN นี้โปรแกรมจะต้องไปอ่านค่า Min,Max ,ID Address มาทำการเปรียบเทียบกับคำสั่งที่ผู้ใช้ส่งเข้ามาถ้าถูกต้องถึงจะทำงานถูกต้อง ถ้าไม่ทำการ Save ค่าเหล่านี้ไว้ก่อน จะทำให้การหมุนของ Servo ผิดพลาดได้

ในการใช้งานครั้งต่อๆไปในส่วนของ ID Address ของบอร์ดไม่จำเป็นต้อง SETUP ใหม่ก็ได้แต่ผู้ใช้ต้องจำได้ว่ากำหนด ID Address ของบอร์ดไว้ที่ค่าเท่าไหร่($0_H$-$F_H$) แต่ถ้าลืมก็ให้ทำการ SETUP ใหม่ได้ เพราะค่า ID Address นี้จะนำมาใช้เมื่อผู้ใช้จะทำการส่งคำสั่งมาควบคุม Servo โดยผ่านทาง MCU ซึ่งจะต้องส่งค่า ID Address นี้มาด้วยเสมอ โดยต้องส่งมาคู่กับค่าของแชนแนลซึ่ง ค่า ID Address จะกำหนดให้เป็น 4 บิทบนและ ค่าแชนแนลที่เลือกจะเป็น 4 บิทล่าง เช่น ถ้ากำหนดให้ ID Address = 2 และเลือกควบคุม Servo ที่แชนแนล 1 ก็ให้ส่งข้อมูลเป็น $21_H$เป็นต้น (ดูรายระเอียดได้จากการใช้งานในโหมด RUN )

ส่วนการใช้งานครั้งต่อ ๆไปใน การ SETUP ค่า Min,Max ให้กับ Servo นั้น ถ้าผู้ใช้ยังใช้ Servo ตัวที่เคย SETUP ไว้แล้วในแชนแนลนั้นๆ และยังไม่มีการเปลี่ยนแปลงค่าที่ SETUP ไว้ (ยังไม่มีการ Save ค่าใหม่ทับลงไป) ก็ไม่จำเป็นต้องทำการ SETUP ใหม่ สามารถเข้าสู่โหมด RUN ได้เลย แต่ถ้านำ Servo ไปต่อในแชนแนลอื่นที่ไม่ใช่แชนแนลเดิมที่เคยต่อ ก็ควรจะ SETUP ค่าใหม่เสมอมิฉะนั้นก็จะเกิดความผิดพลาดในการควบคุมตำแหน่งการหมุนของ Servo ได้

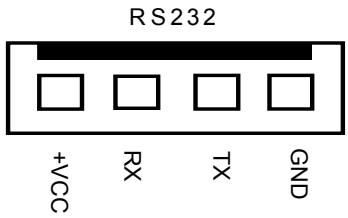
**1. การกำหนด JUMPER สำหรับใช้งานขั้วต่อ RS232 และ RS422 /485**

- ***การใช้งานขั้วต่อ RS232*** ในการใช้งานขั้วต่อ RS232 นั้นให้ผู้ใช้ทำการกำหนด Jumper หมายเลข 9 มาทางด้าน RS232 ดังรูป



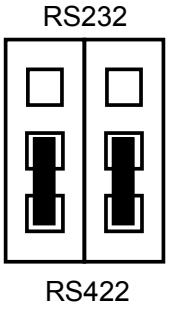

**รูป แสดงการกำหนด Jumper RS232/RS422**

และการจัดเรียงขาของขั้วต่อ RS232 จะแสดงดังรูป

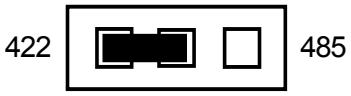



**รูป แสดงขั้วต่อสัญญาณ RS232**

- ***การใช้งานขั้วต่อ* RS422** ในการใช้งานขั้วต่อ RS422 นั้นให้ผู้ใช้ทำการกำหนด Jumper หมายเลข 9 มาทางด้าน RS422 และกำหนด Jumper หมายเลข 15 มาทางด้าน 422 ดังรูป



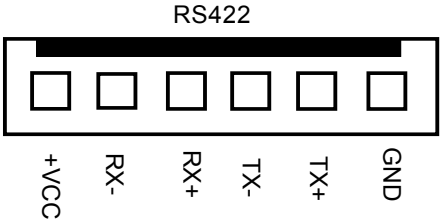

ก) JUMPER RS232/RS422




ข) JUMPER 422/485

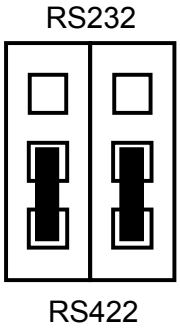
**รูป แสดงการกำหนด Jumper RS232/RS422 และ 422/485**

และการจัดเรียงขาของขั้วต่อ RS422 จะแสดงดังรูป

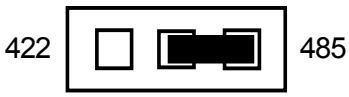



**รูป แสดงขั้วต่อสัญญาณ RS422**

- ***การใช้งานขั้วต่อ RS485*** ในการใช้งานขั้วต่อ RS485 นั้นให้ผู้ใช้ทำการกำหนด Jumper หมายเลข 9 มาทางด้าน RS422 และกำหนด Jumper หมายเลข 15 มาทางด้าน 485 ดังรูปด้านล่าง และการจัดเรียงขาของขั้วต่อ RS485 จะเหมือนกับ RS422 ทุกประการสำหรับในบอร์ดนี้




ก) JUMPER RS232/RS422



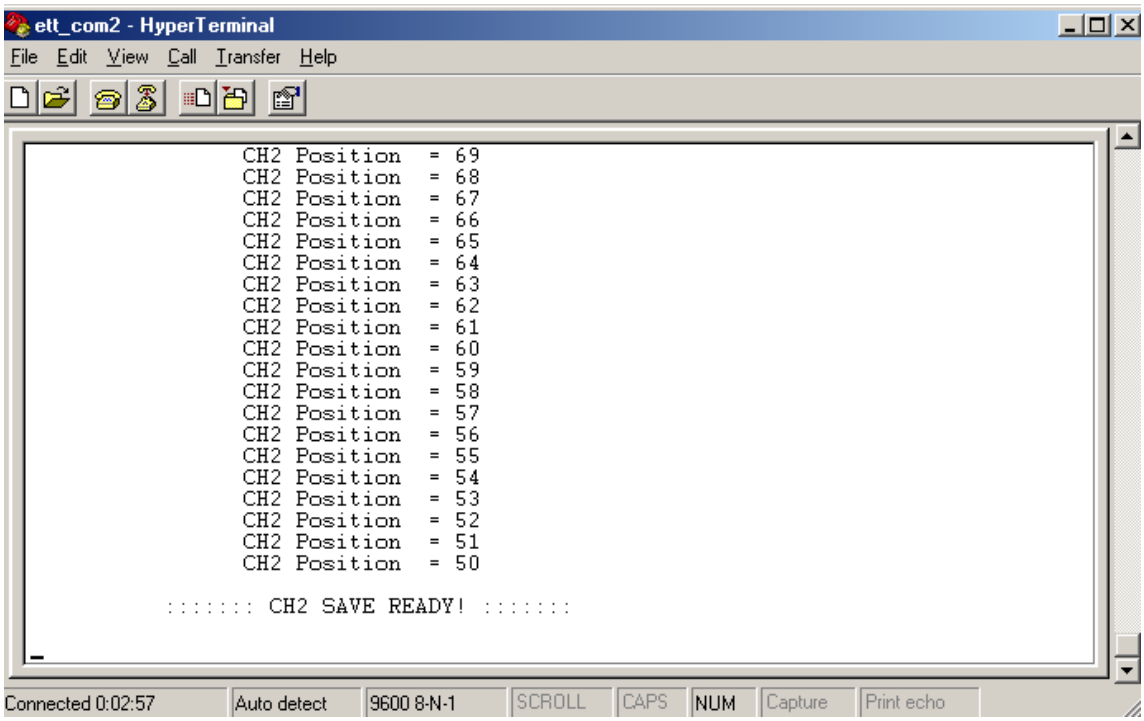

ข) JUMPER 422/485

**รูป แสดงการกำหนด Jumper RS232/RS422 และ 422/485**

**2. การใช้งานโหมด SETUP**

ต่อสายสื่อสารข้อมูลเข้าที่ขั้วต่อ RS232 หรือ RS422/485 ตามที่ผู้ใช้สะดวก จากนั้นนำอีกด้านหนึ่งต่อเข้าที่ COM1 หรือ COM2 ของ PC แล้วเปิดโปรแกรม Procomm หรือ Hyper ขึ้นมารอไว้ ( กำหนด Baud Rate ไว้ที่ 9600 และกำหนด COM Port ที่ต่อให้ถูกต้อง ) เพื่อจะใช้ดูตำแหน่งของ Servo ที่เราทำการ Setup จากนั้นต่อไฟเข้าบอร์ดที่ขั้วหมายเลข 23 แล้วให้ผู้ใช้เลื่อนสวิตช์หมายเลข 16 มาที่ตำแหน่ง SETUP LED สีเหลืองจะติดแสดงการทำงานในโหมดนี้ ในขณะเดียวกัน แชนแนล 1 จะถูกเลือกอัตโนมัติสังเกตุ LED CH1 จะติด และ Servo ที่ต่อไว้ไม่ว่าแชนแนลใดก็ตาม จะหมุนมาอยู่ที่ตำแหน่ง Center อัตโนมัติโดยมีค่าอยู่ที่ 150 เป็นตำแหน่ง Center ซึ่งทุกๆครั้งที่เรากด สวิตช์ RESET และเลือกทำงานในโหมด SETUP จะทำให้ Servo หมุนมาอยู่ที่ตำแหน่ง Center เสมอ แต่ถ้าเปลี่ยนโหมดทำงานเฉยๆไม่ได้กด RESET Servo จะยังคงรักษาตำแหน่งเดิมล่าสุดไว้ จากนั้นให้ผู้ใช้กดสวิตช์ SELECT (หมายเลข17) เลือกแชนแนลที่ต่อ Servo อยู่เพื่อจะทำการ Setup Servo ในแชนแนลนั้น สมมติว่าเราเลือก Setup CH2 ต่อไปให้กดสวิตช์ UP หรือ DOWN Servo ก็จะหมุนตามการกดสวิตช์ของผู้ใช้ และจะแสดงตำแหน่งการหมุนของ Servo ในแต่ละ Step ของการกดสวิตช์ ออกทาง Procomm หรือ Hyper ที่ผู้ใช้ได้เปิดไว้ในตอนแรก ถ้าผู้ใช้ต้องการ Save ค่า Min ก่อนก็กดสวิตช์ DOWN ลงมาเรื่อยๆ จากนั้นให้จับ Servo ดูถ้ารู้สึกว่าเหมือน Servo จะหมุนฝืนเฟืองที่อยู่ภายในหรือมีเสียงคราง จากเฟืองออกมา ก็ให้กดสวิตช์ UP กลับมาซัก 1-2 Step แล้วจึงทำการกดสวิตช์ SAVE (หมายเลข18 ) LED SAVE สีแดงก็จะติดและดับแสดงว่า Save ค่า Min เรียบร้อยแล้ว หรือจะดูที่หน้าจอของ Procomm ก็ได้จะมีข้อความขึ้นว่า “ ::::::: CH2 SAVE READY! ::::::: ” จากนั้นก็ทำการ Save ค่า Max ต่อโดยกดสวิตช์ UP ขึ้นไปเรื่อยๆ ซึ่งตอนนี้ Servo จะหมุนกลับทางมาอีกด้านหนึ่ง เมื่อรู้สึกว่าหมุนฝืนเฟืองแล้ว ก็ให้ทำเหมือนในตอนแรก แล้วจึงกดสวิตช์ SAVE ค่า Max ไว้ ถ้าต่อ Servo ไว้มากกว่า 1 ตัวก็ให้ทำการเปลี่ยนแชนแนลไปยัง Servo ตัวนั้น แล้วทำการ Save ค่า Min และ Max ของ Servo ที่จะใช้งานในแต่ละแชนแนลทุกตัวตามวิธีที่กล่าวมาข้างต้นให้เรียบร้อย ในการสังเกตว่า Servo หมุนมาใกล้ถึงจุด Min หรือ Max หรือยังอาจจะสังเกตได้อีกวิธีหนึ่งคือ โดยปกติ Servo ที่ยังไม่ได้ดัดแปลงจะหมุนได้ 180 องศา จากที่เรากล่าวไปแล้วข้างต้นว่าเมื่อเรานำ Servo มาต่อเข้ากับบอร์ด Servo จะหมุนมาอยู่ที่จุด Center อัตโนมัติ ดังนั้นเมื่อเรา

กดสวิตช์ UP หรือ DOWN Servo จะหมุนมาทางด้านซ้ายและขวา ได้ไม่เกิน 90 องศา จากจุด Center ผู้ใช้ก็จำจุด Center ไว้และให้สังเกต Servo ว่าหมุนห่างออกจากจุด Center ใกล้ถึง 90 องศา หรือยัง ถ้าใกล้แล้วก็ทำการ Save ค่าเก็บไว้ได้

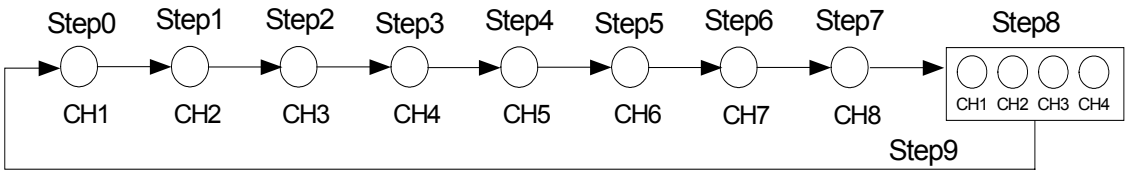



**รูป แสดงหน้าจอในขณะที่กดสวิตช์ UP, DOWN เพื่อหาค่า Min,Max ของ Servo ใน CH2**

จากรูปตำแหน่งของ Servo นั้นจะแสดงเป็นเลขฐาน10 เสมอ เวลาผู้ใช้จะนำค่าตำแหน่งเหล่านี้ไปเขียนส่งข้อมูลมาควบคุม Servo ด้วยภาษา Assambly จะต้องแปลงเป็นฐาน16 เสียก่อน

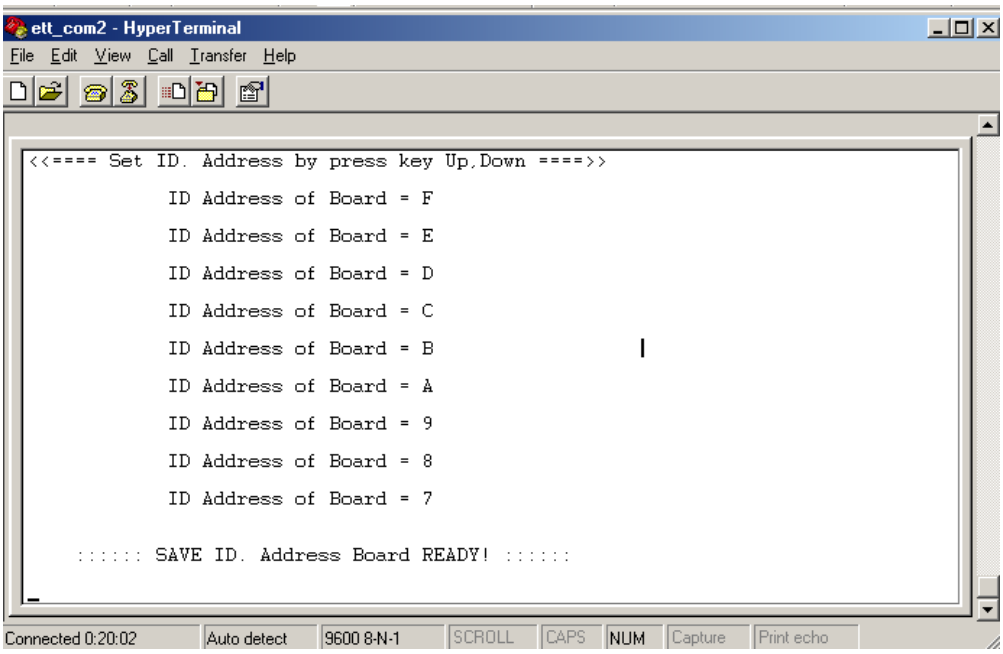
ที่กล่าวไปข้างต้นเป็นการกำหนดค่า Min และ Max ให้กับ Servo ในแต่ละแชนแนล ต่อไปจะเป็นการกำหนดค่า ID Address ให้กับบอร์ด ซึ่งค่า ID Address นี้ จะถูกนำไปใช้ประโยชน์ในการสื่อสารด้วย RS485 ซึ่งจะทำให้สามารถนำบอร์ดนี้มากกว่าหนึ่งบอร์ดมาต่อขนานกันได้ และใช้ตัวส่งข้อมูลเพียงตัวเดียวในการเลือกควบคุม Servo ในแต่ละบอร์ด โดยในแต่ละบอร์ดจะต้องกำหนดค่า ID Address ให้แตกต่างกัน ดังนั้น ถึงผู้ใช้จะไม่ได้ใช้การสื่อสารด้วย RS485 แต่ก็จะต้องกำหนดค่า ID Address ให้กับบอร์ดเช่นกัน เพราะเวลาส่งคำสั่งมาควบคุม Servo ตัวบอร์ดจะทำการตรวจสอบ ID Address ที่ส่งเข้ามาก่อนเสมอ

การกำหนดค่า ID Address มีวิธีเหมือนกับกำหนดค่า Min,Max ที่กล่าวไปข้างต้น คือ ให้กดสวิตช์ SELECT (หมายเลข 17) ไปเรื่อยๆ จนLED แชนแนล 1 ถึง แชนแนล 4 ติดพร้อมกัน นั่นแสดงว่าตอนนี้อยู่ในส่วนการทำงานของ การกำหนดค่า ID Address โดยสเต็ปการทำงานของสวิตช์ SELECT แสดงดังรูป




**รูป แสดง Step การทำงานของการกดสวิตช์ SELECT**

หลังจากที่ LED ทั้ง 4 แชนแนลติดแล้ว ซึ่ง LED ทั้ง 4 แชนแนลนี้ จะแสดงค่าเป็นรหัสเลขฐาน16 โดย บิต CH1 ถือเป็นบิตต่ำสุด และ CH4 เป็นบิตสูงสุด ดังนั้นค่ารหัสประจำแต่ละบิตจะไล่เป็น 1,2,4,8 ซึ่งจะเห็นว่าขณะที่ LED ทั้ง 4 แชนแนลติดหมด นั่นก็คือ รหัส $F_H$ ถ้าผู้ใช้ต้องการกำหนดค่า ID Address ที่รหัสนี้ก็ให้กด SW. SAVE ได้เลย LED SAVE ก็จะกระพริบ 1ครั้งแสดงว่า Save เรียบร้อย ถ้าต้องการเปลี่ยน ID Address เป็นรหัสอื่นก็ให้กดสวิตช์ DOWN เพื่อลดค่า ID Address ลง หรือกดสวิตช์ UP เพื่อเพิ่มค่า ID Address ขึ้น ค่ารหัสที่แสดงที่ LED ก็จะเปลี่ยนค่าไปโดยลดค่าลงมา หรือเพิ่มค่าขึ้นเรื่อยๆ ตามการกดสวิตช์ของผู้ใช้ซึ่งค่า ID Address จะสามารถเลือกได้อยู่ในช่วง $0_H$ - $F_H$ เมื่อเลือกค่าที่ต้องการได้แล้วให้กด สวิตช์ SAVE ก็เป็นอันเรียบร้อย สำหรับการทำงานในโหมด SETUP ในกรณีที่ผู้ใช้อ่านค่ารหัส ID Address จาก LEDไม่เป็น ก็ให้ดูที่หน้าต่างของโปรแกรม Procomm หรือ Hyper ที่ได้เปิดไว้ในตอนแรก ซึ่งจะแสดงค่า ID Address ให้เห็นตาม Step ที่กดไป ดังรูปด้านล่าง




**รูป แสดงหน้าจอขณะทำการกำหนดค่า IP Address โดยรูปนี้กำหนด ID Address ที่ 7**

สังเกตว่าในส่วนของค่า ID Address ที่แสดงให้เห็นที่หน้าจอ หรือที่อ่านค่าจาก LED นั้นจะเป็นเลขฐาน 16 เสมอ เวลานำค่าไปใช้งานในการเขียนโปรแกรมจะต้องระวังด้วย เนื่องจากมันจะแตกต่างจากค่าตำแหน่งของ Servo ที่จะแสดงเป็นเลขฐาน 10 ส่วนค่า ID Address จะแสดงเป็นเลขฐาน 16 โดยเฉพาะถ้าเขียนโปรแกรมควบคุม Servo ด้วยภาษา Assambly จะต้องแปลงให้อยู่ในรูปฐาน16 ให้เหมือนกันก่อน

**3. การใช้งานโหมด RUN**

หลังจากที่ได้ทำการ Setup เรียบร้อยแล้ว ให้เลื่อนสวิตช์หมายเลข 16 มาทางด้าน RUN เพื่อเริ่มต้นการควบคุม Servo จากที่กล่าวไปข้างต้นว่าในโหมดนี้จะแบ่งการทำงานออกเป็น 3 โหมดย่อย คือ *การคอนโทรล Servo ผ่านทางPC ด้วยโปรแกรม ET-RSS V1.00 ,การคอนโทรล Servo โดยการ Run Step การหมุนของ Servo ที่ Save ไว้ใน E²Prompt* และโหมดสุดท้ายคือ *การคอนโทรล Servo ด้วย MCU* ซึ่งการคอนโทรลด้วยวิธีนี้ จะสามารถควบคุมให้ Servo หมุนได้มากกว่าหนึ่งตัวและทำงานเป็น Loop ได้ตามลักษณะโปรแกรมที่ผู้ใช้เขียนขึ้น

***3.1 โหมดการคอนโทรล Servo โดยใช้ MCU เป็นตัวส่ง Command Code***

ในการทำงานโหมดนี้ผู้ใช้สามารถเขียนโปรแกรมส่งข้อมูลจาก MCU ที่ต่ออยู่ภายนอกบอร์ดตระกูลใดก็ได้ผ่านทาง Function Uart ของ MCU มาควบคุม Servo ที่ต่ออยู่บนบอร์ดนี้ โดยจะต้องส่งข้อมูลแบบอนุกรมมาชุดละ 3 Byte ต่อการทำให้ Servo หมุนไป 1 ตำแหน่ง โดยข้อมูลที่จะต้องส่งมาไบต์แรกคือ Sync byte ซึ่งมีค่าเป็น $FF_H$ (255) เท่านั้น ไบต์ที่ 2 คือ ค่าของ ID Address และค่าของ แชนแนล โดยกำหนดให้ค่า ID Address อยู่ที่ 4 บิตบน และค่าของแชนแนลอยู่ที่ 4 บิตล่าง ไบต์ที่ 3 คือตำแหน่งของ Servo ที่จะให้ Servo หมุนไป มีค่าตั้งแต่ 50-250 (ฐาน10) Step ซึ่งแต่ละ Step ที่เปลี่ยนแปลงไปจะมีความละเอียดอยู่ที่ 10us

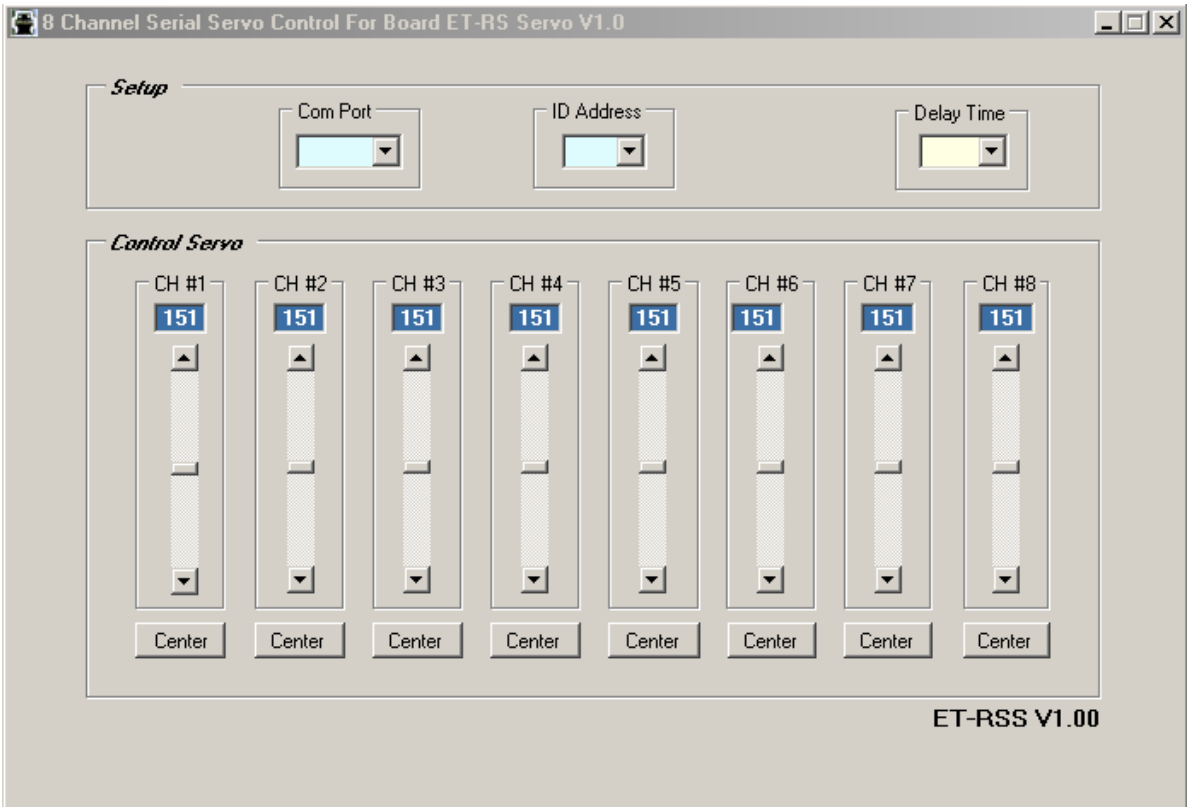
เริ่มต้นให้ผู้ใช้กดสวิตช์ DOWN เพื่อเข้ามาทำงานในโหมดนี้ หรือเลื่อนสวิตช์ หมายเลข16 มาที่ตำแหน่ง RUN ในครั้งแรกก็จะเข้ามาในโหมดนี้โดยอัตโนมัติเช่นกัน ซึ่งจะสังเกตเห็นว่า LED สีเขียวและแดงจะติด แสดงว่าเข้ามาทำงานในโหมดนี้แล้ว จากนั้นก็ต่อสายสื่อสารเข้ากับขั้ว RS232 หรือ RS422/485 ของบอร์ด ET-RS SERVO V1.0 ขึ้นอยู่กับผู้ใช้จะเลือกใช้งาน และต่อสายอีกด้านหนึ่งเข้ากับบอร์ดตัวส่งข้อมูลของท่าน โดยถ้าเลือกการส่งแบบไหนก็จะต้องต่อให้เหมือนกันด้วย เช่นต้องการสื่อสารโดยใช้ RS232 ก็จะต้องต่อที่ขั้วต่อ RS232 ของทั้งบอร์ดส่งและบอร์ดรับให้ตรงกัน และต้องจำไว้อย่างหนึ่งว่า จะต้องสลับสายของขั้วต่อให้ถูกต้อง หลักการง่ายๆก็คือ ให้ต่อขา RX ของบอร์ดแรกเข้ากับขา TX ของบอร์ดที่2 และต่อขา RX ของบอร์ดที่ 2 เข้ากับขา TX ของบอร์ดแรก ดูการต่อได้จากภาคผนวกท้ายเล่ม

หลังจากที่ต่อสายเรียบร้อยแล้วก็ให้ทำการรันโปรแกรมที่เขียนไว้ในบอร์ดส่งได้ Servo ก็จะหมุนไปตามโปรแกรมที่คุณเขียนถ้าโปรแกรมนั้นถูกต้อง ถ้า Servo ไม่หมุนตามที่ต้องการแสดงว่าการส่งข้อมูลผิดพลาด ให้ลอง Reset MCU ของบอร์ดส่งดู และถ้ายังไม่ทำงานอีก ก็ลอง Reset MCU ของบอร์ดคอนโทรล Servo นี้ดูโดยกดที่สวิตช์หมายเลข 11 จากนั้นก็ให้เข้ามาอยู่ในโหมดย่อยนี้อีกครั้ง แล้วจึงทำการ Reset MCU ของบอร์ดส่งดูใหม่อีกครั้ง ซึ่งถ้าโปรแกรมที่ผู้ใช้เขียนขึ้นไม่ผิดพลาดก็จะต้องทำงานได้…

***3.2 โหมดการคอนโทรล Servo ผ่านทาง PC โดยใช้โปรแกรม ET-RSS V1.00***

การใช้งานในโหมดนี้ ก่อนอื่นให้ต่อสายสื่อสารข้อมูลเข้าที่ COM1 หรือ COM2 ของ PC และอีกด้านหนึ่งต่อเข้าที่บอร์ด ET-RS SERVO V1.0 ที่จุด RS232 หรือ RS 422/485 ก็ได้แล้วแต่ผู้ใช้จะสะดวกแต่การใช้งาน RS422/485 ผู้ใช้จะต้องมีชุดแปลงจาก RS232 ไปเป็น RS422/485 ด้วยถึงจะใช้งานได้ จากนั้นให้

เลื่อนสวิตช์หมายเลข 16 มาทางตำแหน่ง RUN แล้วให้ผู้ใช้กดสวิตช์ SELECT จะเห็น LED สีเหลืองสว่างขึ้นคู่กับสีเขียว แสดงว่าเข้าสู่โหมดย่อยนี้เรียบร้อยแล้ว จากนั้นให้ทำการรันโปรแกรม ET-RSS.EXE ขึ้นมาจะแสดงดังรูป




**รูปหน้าต่างเมื่อ RUN โปรแกรม ET-RSS V1.00**

***การใช้งานโปรแกรม ET-RSS V1.00***

จากรูปข้างต้นจะมีส่วนควบคุมอยู่ 2 ส่วน คือ **Setup** และ **Control Servo** เมื่อทำการรันโปรแกรมขึ้นมาแต่ละครั้ง อันดับแรกผู้ใช้จะต้องทำการ Setup การใช้งานโปรแกรมเสียก่อนโดยเลือก Com Port ที่ผู้ใช้ต่อใช้งานอยู่ และเลือก ID Address ให้ตรงกับที่ผู้ใช้ได้ตั้งไว้ที่บอร์ดคอนโทรล Servo จากนั้นก็เลือกค่า Delay Time มาค่าใดค่าหนึ่งก็ได้ ซึ่งค่าที่เหมาะสมจะอยู่ในช่วง 10-20 โดยค่า Delay Time นี้จะไม่มีผลต่อการควบคุม Servo ครั้งละแชนแนล แต่จะมีผลเมื่อผู้ใช้ทำการ Save ค่า Step ในแต่ละแชนแนลเก็บไว้ เวลาเราทำการ RUN Step ที่ Save ไว้ ค่า Delay Time นี้จะเป็นตัวหน่วงค่าของ Step นั้นๆให้ค้างอยู่นานแค่ไหน

เมื่อ Setup เรียบร้อยแล้วผู้ใช้ก็สามารถเลื่อน Scroll Bar ควบคุม Servo ในแชนแนลที่ต้องการได้ ซึ่งถ้าผู้ใช้ทำการ Setup ไม่เรียบร้อย แล้วมาเลื่อน Scroll Bar เลยจะทำให้โปรแกรม Error ฉะนั้นต้องกลับไป Setupให้เรียบร้อยก่อนในทุกครั้งที่เปิดโปรแกรมขึ้นมา จากรูป Step เริ่มต้นจะถูกกำหนดไว้ที่ 151 เมื่อผู้ใช้ทำการเลื่อน Scroll Bar Step ที่แสดงก็จะเปลี่ยนแปลงตามไปด้วย โดย Step การหมุนจะอยู่ในช่วง 50-250 และถ้าต้องการให้ Servo หมุนมาอยู่ที่ตำแหน่ง Center ก็ให้คลิกที่ปุ่ม Center Servo ก็จะหมุนมาอยู่ที่ตำแหน่ง Center นั่นคือค่า Step = 150

ในกรณีที่ผู้ใช้ Save ค่า Min และ ค่า Max( ตอน Setup Servo ) ของแชนแนลใดก็ตามไว้ไม่ถึงตำแหน่ง 50 และ 250 เวลาเลื่อน Scroll Bar เกินค่า Min และ Max ไป Servo จะไม่หมุนต่อไปจะหยุดอยู่ที่ตำแหน่งของค่า Min และMax นั้นๆ จนกว่าผู้ใช้จะเลื่อน Scroll Bar มาอยู่ในย่านของค่า Min และ Max นั้นอีกครั้ง Servo ถึงจะเกิดการเปลี่ยนแปลงขึ้นอีกครั้ง และสิ่งที่ต้องคำนึงถึงอีกประการคือ ค่าตำแหน่งของ Servo ที่เราคอนโทรลที่ Scroll Bar จะถูกส่งไปควบคุม Servo ก็ต่อเมื่อ ค่าของ Scroll Bar เกิดการเปลี่ยนแปลงขึ้นเท่านั้น ถ้าค่าของ Scroll Bar คงที่ไม่มีการเปลี่ยนแปลงก็จะไม่มีการส่งค่า Step การหมุนไปควบคุม Servo แต่ตัวบอร์ดจะนำค่าล่าสุดก่อนหน้านี้ที่ส่งไปให้มาควบคุม Servo

ในส่วนของโหมดนี้นอกจากจะใช้หา Step การหมุนของ Servo ในแต่ละแชนแนลได้แล้วยังสามารถ Save ค่า Step ของแต่ละแชนแนล เก็บไว้ได้เพื่อใช้ RUN ดูตัวอย่างการหมุนของ Servo เป็น Pattern เบื้องต้นได้ ในการ Save ค่า Step แต่ละ Step ที่จะใช้งานเก็บไว้ทำการ RUN ผู้ใช้สามารถเปลี่ยนค่า Delay Time ของแต่ละ Step ได้ หรือจะใช้ค่า Delay Time ทุก Step เท่ากันหมดก็ได้ ในการเปลี่ยนค่า Delay Time จะต้องเปลี่ยนก่อนที่จะเลื่อน Scroll Barเสมอ ค่า Delay Time ที่กำหนดถึงจะถูกนำไปใช้งานได้ถูกต้องตรงกับ Step ที่ผู้ใช้ต้องการ เช่น ต้องการ Save ค่าที่ทำให้ Servo แชนแนล 1 หมุนมาอยู่ที่ตำแหน่ง 100 และหลังจากหมุนแล้วให้ Delay ไว้ 15 ก่อนที่จะไปทำงานใน Step ต่อไป ก็ให้ทำดังนี้ เลือกค่า Delay Time มาอยู่ที่เลข 15 จากนั้นก็เลื่อน Scroll Bar CH1 ไปที่ตำแหน่ง 100 แล้วจึงกด สวิตช์ Save ที่ตัวบอร์ด 1 ครั้ง LED Save ที่บอร์ดก็จะติดและดับ นั่นแสดงว่า ได้ Save Step แรกเรียบร้อยแล้ว จากนั้นก็ทำในลักษณะนี้ต่อไปในการ Save Step อื่น ถ้าเราให้ Delay Time เท่ากันหมดทุก Step ก็ไม่ต้องไปเปลี่ยนค่า Delay Time ให้เลื่อน Scroll Bar หาตำแหน่งที่ต้องการแล้วกด สวิตช์ Save ได้เลย โดยค่า Step ค่า แชนแนล และค่า Delay Time จะถูกเก็บไว้ใน $E^2$Prompt ซึ่งจะมีเนื้อที่เก็บ Command ประมาณ 13.5 Kbyte ในการกดสวิตช์ Save นั้นให้กดแล้วรีบปล่อยมิฉะนั้นจะเป็นการ Save Step เดิมซ้ำหลายครั้ง ส่งผลให้เวลา RUN จะทำให้เสียเวลาในการอ่าน Step เดิมหลายครั้ง ซึ่งจะเป็นการเพิ่ม Delay ไปในตัวอีกลักษณะหนึ่ง ซึ่งถ้าเรา Set ค่า Delay Time ไว้เหมาะสมแล้วก็ไม่ควรใช้การ Save ตำแหน่งเดิมทับอีกเพราะจะทำให้เสียเวลา

ในกรณีที่เรา Save ค่าตำแหน่งพลาดไม่ตรงกับที่เราต้องการเราไม่สามารถย้อนกลับมาลบออก เพื่อ Save ค่าใหม่ได้ แต่การ Save จะเป็นในลักษณะต่อกันไปเรื่อยๆ ค่าเก่าที่เราไม่ต้องการจะยังถูกบันทึกอยู่และเวลา RUN ก็จะถูกนำมา RUN ด้วย ดังนั้นถ้า Save ผิดก็ควรจะออกจากโหมดนี้และเข้ามาในโหมดนี้ใหม่ แล้วจึงเริ่มทำการ Save ใหม่ทั้งหมดตั้งแต่แรกอีกครั้ง ค่าเก่าที่เคย Save ไว้ก็จะถูกค่าใหม่นี้ทับลงไปทั้งหมด ในทุกๆครั้งที่เราออกจากโหมดนี้แล้วกลับเข้ามาใหม่ และมีการกดสวิตช์ Save ขึ้น ค่าที่เรา Save เก็บไว้ก่อนหน้านี้ก็จะหายไปหมด จะเหลือแค่ค่าใหม่ที่ถูก Save ลงไป ที่จะสามารถถูก RUN ขึ้นมาได้ ดังนั้นจึงควรระวังด้วย ถ้าเรายังต้องการใช้งาน Step เดิมที่บันทึกเก็บไว้อยู่ และในการ Save นี้ในบางครั้งอาจจะทำให้บอร์ด Error ได้เวลาทำการ RUN Step ที่ Save ซึ่งจะทำให้ไม่สามารถใช้งานสวิตช์ต่างๆบนบอร์ดได้ก็ให้ผู้ใช้ทำการกดสวิตช์ Reset MCU ใหม่

### *3.3 การ RUN STEP การหมุนของ Servo ที่ Save ไว้ใน $E^2$Prom*

ในโหมดนี้จะทำหน้าที่ในการ RUN Step ของ Servo ที่ผู้ใช้ได้ทำการ Save ไว้ในหัวข้อที่ 3.2 ซึ่งการใช้งานนั้นผู้ใช้จะต้องทำการ Save ค่าตำแหน่งของ Servo ที่ต้องการ เก็บไว้ใน $E^2$Prom เรียบร้อยแล้วตามที่ได้กล่าวไปในหัวข้อ 3.2 จากนั้นให้ผู้ใช้ทำการกดสวิตช์ UP จะสังเกตเห็นว่า LED ทั้ง 3 ดวง คือ เขียว ,แดง , เหลือง จะติดขึ้นมา แสดงว่า เข้าสู่การทำงานในส่วนนี้เรียบร้อยแล้ว Servo ก็จะหมุนไปตาม Step ที่เราได้โปรแกรมไว้ และเมื่อ RUN มาถึง Step สุดท้ายที่เราได้โปรแกรมไว้ ตัวบอร์ดก็จะวนกลับไป RUN เริ่มต้นที่ Step แรกอีกครั้งเป็นแบบนี้ไปเรื่อยๆ จนกว่าเราจะออกจากโหมดนี้ โดยการกดสวิตช์เปลี่ยนโหมดการทำงานไปเป็นโหมดอื่น Servo ที่ RUN อยู่ก็จะหยุดหมุน และไปทำงานตามคำสั่งของโหมดใหม่ต่อไป ทุกครั้งที่เราเข้ามาในโหมดนี้ Step การหมุนของ Servo ที่ถูก Save ไว้ใน $E^2$Prom ก็จะถูก RUN ให้ทำงานเสมอ

## 4. วิธีและหลักการในการเขียนโปรแกรมควบคุม Servo ให้กับบอร์ด ET-RS SERVO V1.0

ในการเขียนโปรแกรมส่งข้อมูลออกไปควบคุม Servo นั้น จะเป็นการส่งข้อมูลออกครั้งละ 3 Byte ต่อการทำให้ Servo นั้นหมุนไป 1 ครั้ง ดังนั้นผู้ใช้จะต้องกำหนดรูปแบบที่แน่นอนในการที่จะให้ Servo หมุนไป และทำการส่งข้อมูลออกไปเป็นชุดๆเพื่อให้ Servo เกิดการเคลื่อนไหวอย่างต่อเนื่องและเป็นไปตามรูปแบบที่ผู้ใช้กำหนด สำหรับ Baud Rate ที่จะต้องกำหนดให้ MCU ในการรับส่งข้อมูลนั้นคือ 9600 โดยผู้ใช้จะต้องเขียนโปรแกรม กำหนด Baud Rate ค่านี้ ให้กับ MCU ที่เป็นตัวส่งข้อมูลของท่านเสมอ ไม่ว่าจะใช้ MCU เบอร์ใดตระกูลใดก็ตาม มิเช่นนั้นแล้วจะทำให้การส่งข้อมูลออกไปไม่ตรงกับบอร์ด ET-RS SERVO V1.0 ได้ โดย Baud Rate นี้จะเป็นค่าที่ตายตัวไม่สามารถที่จะเปลี่ยนเป็นค่าอื่นได้

***- ขั้นตอนในการเขียนโปรแกรม***

ก่อนอื่นขอทำความเข้าใจกับผู้อ่านก่อนว่า ตัวอย่างโปรแกรมที่เราจะอ้างถึงนี้จะเป็นตัวอย่างสำหรับใช้กับ MCU Z8Encore! โดยอ้างอิงกับบอร์ดรุ่น CP-JRZ8F02 V1.0 หรือ CP-JRZ8F01 V1.0/EXP ที่ทางบริษัท ETT เป็นผู้จัดทำขึ้นเท่านั้น ซึ่งจะพัฒนาโปรแกรมด้วยภาษา C แต่ผู้ใช้สามารถนำตัวอย่างโปรแกรมเหล่านี้ ไปดัดแปลงใช้กับไมโครคอนโทรลเลอร์ในตระกูลอื่นได้ซึ่งจะไม่ยากนัก เพียงแต่ผู้ใช้ต้องเข้าใจวิธีการเขียนโปรแกรม รับ-ส่ง ข้อมูล หรือ การใช้งาน UART Port ของ MCU ในตระกูลนั้นๆก็เพียงพอ

1) ทำการกำหนดค่าเริ่มต้น Set MCU ให้ทำงานในฟังก์ชันของการรับส่งข้อมูล(Uart) และกำหนด Baud Rate ในการรับส่งที่ 9600 bit/s
2) เขียนโปรแกรมย่อยในส่วนของ Delay Time , การเช็คบิต Flag สำหรับการรับ , การเช็คบิต Flag สำหรับการส่ง
3) เขียนโปรแกรมย่อยในส่วนของการ ตรวจสอบ Sync Byte ซึ่ง Sync Byte ที่จะตรวจสอบจะใช้ค่า $FF_H$ เท่านั้น โดยหลักในการเขียนคือ
   - เรียกโปรแกรมย่อยตรวจสอบ Flag การส่ง เมื่อพร้อม ก็ส่ง Sync Byte $FF_H$ ออกไปให้บอร์ดคอนโทรล
   - เรียกโปรแกรมย่อยตรวจสอบ Flag การรับ เมื่อพร้อม ก็ให้อ่าน ข้อมูลจากบอร์ดคอนโทรลเข้ามา

- นำข้อมูลที่อ่านมาตรวจสอบว่า ใช่ $FF_H$ หรือไม่ ( ทางบอร์ดคอนโทรล Servo เมื่อได้รับ Sync Byte แล้ว ก็จะส่งค่า Sync Byte นั้นกลับมาให้บอร์ดตัวส่งอีกครั้งเสมอ) ถ้าใช่ก็ให้จบในส่วนของการส่ง Sync Byte แต่ถ้าไม่ใช่ก็ให้วนกลับไปส่ง Sync Byte ต่อ จนกว่าจะใช่

4) หลังจากที่ Set MCU และเขียนโปรแกรมย่อยในส่วนต่างๆไว้เรียบร้อยแล้ว ต่อไปในส่วนของโปรแกรมหลักก็ให้เริ่มทำการส่งข้อมูลได้ตามนี้

- เรียกโปรแกรมย่อยในส่วนของการส่ง Sync Byte (เป็น Byte ที่ 1)
- เรียกโปรแกรมย่อยตรวจสอบ Flag การส่ง แล้วจึงส่งค่า ID Address (4บิตบน) และค่าแชนแนลของ Servo (4บิตล่าง) ออกไปให้บอร์ดคอนโทรล (เป็น Byte ที่2)
- เรียกโปรแกรมย่อยตรวจสอบ Flag การส่ง แล้ว ส่งค่า ตำแหน่งที่จะให้ Servo หมุน ออกไปยังบอร์ดคอนโทรล (เป็น Byte ที่ 3)

ใน Step ที่กล่าวนี้เป็นการส่งข้อมูลออกไปควบคุม Servo ให้หมุนเพียง 1 ครั้งเท่านั้น ต่อไปเราจะมาดูตัวอย่างโปรแกรมการควบคุม Servo 2 ตัวให้ทำงานในลักษณะเป็น Loop ตามรูปแบบที่วางไว้

**Example**

ในตัวอย่างนี้ได้กำหนดให้โปรแกรมสามารถใช้การสื่อสารแบบ RS232,RS422,RS485 ได้กับบอร์ด CP-JRZ8F02 V1.0 ในแชนแนล 0 ส่วนบอร์ดรุ่น CP-JRZ8F01 V1.0/EXP สามารถใช้การสื่อสารแบบ RS232 แชนแนล 0 ได้เพียงอย่างเดียว

ตัวอย่างโปรแกรมนี้จะใช้ในการควบคุม Servo Motor 2 ตัวด้วยกัน โดยต่อ Servo Motor เข้ากับบอร์ด ET-RS SERVO V1.0 ที่ CH1 และ CH2 แล้ว ทำการ Save หาค่า Min และ Max ของทั้ง 2 แชนแนลให้เรียบร้อย รวมทั้งกำหนด ID Address ของบอร์ดคอนโทรล ET-RS SERVO V1.0 เป็น 0 จากนั้นทำการRun โปรแกรมที่บอร์ดตัวส่งสำหรับส่ง Command ได้ โดยที่บอร์ดรับ Command (ET-RS SERVO V1.0) จะต้อง Set ให้อยู่ในโหมดของการรับคำสั่งจาก MCU รอไว้เสียก่อนมิเช่นนั้นจะทำให้การ รับ-ส่ง ผิดพลาดได้ ซึ่งแก้ไขได้โดยทำการ Reset MCU ของบอร์ดตัวส่งใหม่

```
#include <stdio.h>
#include <ez8.h>
  //-------------------------------------- Delay  ------------------------------------//
   void delay (int count)
   {  int  i , j  ;
      for ( i = 0 ; i <= count ; i++ )
       for ( j = 0 ; j <= count ; j++ )          ;
   }
```

```
//-------------------------------------- Check Receiver data ----------------------------------------//
      void recev_uart()
       {
         PDOUT = 0x00                        ; // For Control RS485 ;PD6 = 0:ทำหน้าที่รับข้อมูล
         while (! ( U0STAT0 & 0x80 ) ) { ; }  // เช็คสถานะการรับข้อมูล บิต RDA=1 คือพร้อมรับ
      }
 //---------------------------------- Check Transmission data ----------------------------------//
      void transmit()
       {
         PDOUT = 0xFF                        ; // For Control RS485 ; PD6 = 1 :ทำหน้าที่ส่งข้อมูล
         while (! ( U0STAT0 & 0x04 ) ) { ; } //เช็คสถานะการส่งข้อมูล บิต TDRE=1 คือพร้อมส่ง
       }
//---------------------------------------- Sent Sync Byte ----------------------------------------//
       void sync()
       {
         do
          {
            transmit()                       ;  //เช็คสถานะการส่ง
            U0D = 0xFF                       ; // ส่ง Sync Byte FF_H
            delay(10)                        ; // delay bit control PD6 for RS485(ถ้าใช้)
            recev_uart()                     ; //รับการตอบรับ Sync Byte ค่า FF_Hจากบอร์ดรับ
            delay(10)                        ; // delay bit control PD6 for RS485(ถ้าใช้)
         } while (U0D !=  0xFF)              ; // ตรวจสอบข้อมูลที่รับมาว่าใช่ FF หรือไม่
       }

//-------------------------------------------- Main ----------------------------------------------//
       main()
         {
           unsigned char ch1 , ch2    ;
           unsigned  int  x           ;
```

```
        ch1 = 0x01                          ; // IP Address 0 + Channel 1
        ch2 = 0x02                          ; // IP Address 0 + Channel 2
//------------------------------------ Initial Transmission Uart0 ------------------------------------//
        U0BRH  = 0x00                       ;
        U0BRL  = 120                        ; // Set Baud Rate = 9600 Kb.
        PAAF   = 0x30                       ; // Set Alternate Function PA4-5 for Uart 0
        U0CTL0 = 0xC0                       ; // Control Register Uart0 Function
        PDDD   = 0x00                       ; //  Port C = OUTPUT For Ctrl. RS485
        While(1)
        {
//====================== Step 1Turn Left ========================//
         for (x = 50 ; x <= 250 ; x = x+2)      // Loop สำหรับเพิ่มค่าตำแหน่งให้ครั้งละ2 Step
          {
        //------------------------------ Servo CH1 Operate ------------------------------//
            sync()                          ; // ส่ง Sync Byte
            transmit()                      ; // Check Buffer สำหรับการส่งว่างหรือไม่
            U0D = ch1                       ; // ส่ง IP Address และ ค่าแชนแนล(0x01)
            transmit()                      ; // Check Buffer สำหรับการส่งว่างหรือไม่
            U0D = x                         ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
            delay(250)                      ; // หน่วงเวลาการส่งข้อมูลในแต่ละStep
        //------------------------------ Servo C2 Operate ------------------------------//
            sync()                          ; // ส่ง Sync Byte
            transmit()                      ;
            U0D = ch2                       ; // ส่ง IP Address และ ค่าแชนแนล(0x02)
            transmit()                      ;
            U0D = x                         ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
            delay(250)                      ;
          }
            delay(500)                      ;
```

```
//====================== Step2 Turn Right ========================//
        for (x = 250 ; x >= 50 ; x = x-10)        // Loop สำหรับลดค่าตำแหน่งลงครั้งละ10 Step
         {
        //----------------------------- Servo CH1 Operate -----------------------------//
             sync()                              ; // ส่ง Sync Byte
             transmit()                          ; // Check Buffer สำหรับการส่งว่างหรือไม่
             U0D = ch1                           ; // ส่ง IP Address และ ค่าแชนแนล(0x01)
             transmit()                          ; // Check Buffer สำหรับการส่งว่างหรือไม่
             U0D = x                             ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
             delay(250)                          ; // หน่วงเวลาการส่งข้อมูลในแต่ละStep
       //----------------------------- Servo CH2 Operate -----------------------------//
             sync()                              ;  // ส่ง Sync Byte
             transmit()                          ;
             U0D = ch2                           ; // ส่ง IP Address และ ค่าแชนแนล(0x02)
             transmit()                          ;
             U0D = x                             ;  // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
             delay(250)                          ;
         }
             delay(500)                          ;

//====================== Step3 Turn Left ========================//
        //----------------------------- Servo CH1 Operate -----------------------------//
             sync()                              ; // ส่ง Sync Byte
             transmit()                          ; // Check Buffer สำหรับการส่งว่างหรือไม่
             U0D = ch1                           ; // ส่ง IP Address และ ค่าแชนแนล(0x01)
             transmit()                          ; // Check Buffer สำหรับการส่งว่างหรือไม่
             U0D = 250                             ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
             delay(900)                          ; // หน่วงเวลาการส่งข้อมูลในแต่ละStep
```

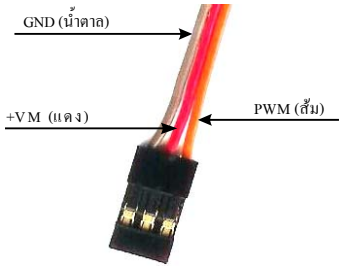
```
//------------------------------------ Servo CH2 Operate ------------------------------------//
            sync()                          ; // ส่ง Sync Byte
            transmit()                      ;
            U0D = ch2                       ; // ส่ง IP Address และ ค่าแชนแนล(0x02)
            transmit()                      ;
            U0D = 250                       ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
            delay(900)                      ;

//====================== Step4 Turn Right ========================//
        //----------------------------- Servo CH1 Operate -----------------------------//
            sync()                          ; // ส่ง Sync Byte
            transmit()                      ; // Check Buffer สำหรับการส่งว่างหรือไม่
            U0D = ch1                       ; // ส่ง IP Address และ ค่าแชนแนล(0x01)
            transmit()                      ; // Check Buffer สำหรับการส่งว่างหรือไม่
            U0D = 50                        ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
            delay(900)                      ; // หน่วงเวลาการส่งข้อมูลในแต่ละStep
      //----------------------------- Servo CH2 Operate -----------------------------//
            sync()                          ; // ส่ง Sync Byte
            transmit()                      ;
            U0D = ch2                       ; // ส่ง IP Address และ ค่าแชนแนล(0x02)
            transmit()                      ;
            U0D = 50                        ; // ส่งค่าตำแหน่งที่จะให้ Servo หมุน
            delay(900)                      ;
        }
    }
```

จากตัวอย่างโปรแกรมข้างต้นถ้า Servo ที่นำมาต่อเป็นของยี่ห้อ GWS เมื่อทำการ RUN โปรแกรม จะเห็นว่า Servo จะหมุนทวนเข็มนาฬิกาก่อน จากนั้นก็จะหมุนกลับคือ ตามเข็มนาฬิกาสลับกันไปในลักษณะนี้เรื่อยๆ ซึ่งในตัวอย่างโปรแกรมนี้จะเห็นว่ามีรูปแบบการส่งข้อมูลอยู่ 2 ลักษณะคือ ใน Stepที่ 1 กับ 2 นั้นจะเป็นการส่งข้อมูลในลักษณะ ให้เพิ่มค่าขึ้นครั้งละ 2 Step ( Servo หมุนซ้าย ) และลดค่าลงครั้งละ 10 Step ( Servo หมุนขวา) ตามลำดับ จนถึงค่าตำแหน่งที่เราต้องการ การส่งข้อมูลในลักษณะนี้จะช่วยให้เรากำหนดความเร็ว ในการหมุนได้ คือถ้ากำหนดให้ Servo มีการเปลี่ยนแปลงครั้งละหลาย Step Servo ก็จะหมุนเร็วขึ้น และในการเปลี่ยนแปลงของ Step ที่กำหนดจะต้องกำหนดค่า Delay ให้เหมาะสมด้วยเพื่อให้การหมุนของ Servo ไม่ผิดพลาด และการส่งข้อมูลอีกลักษณะหนึ่งก็คือ ใน Step ที่ 3 กับ 4 จะเป็นการตำแหน่งเริ่มต้นและตำแหน่งสุดท้ายให้กับ Servo ตรงๆเลย ซึ่งการส่งข้อมูลในลักษณะนี้จะทำให้ Servo หมุนด้วยความเร็วสูงสุด ดังนั้นค่า Delay Time จะต้องกำหนดมากขึ้นดังนั้นถ้ารันโปรแกรมดูแล้ว จะสังเกตเห็นว่า เมื่อโปรแกรมทำงานใน Step ที่ 1 และ 2 แล้วจะเห็นเหมือนว่า Servo หมุน 2 ตัวพร้อมกันเนื่องจากค่า Delay Time ต่ำ แต่ถ้าโปรแกรมทำงานใน Step ที่ 3 และ 4 จะเห็นว่า Servo ในแชนแนล 1 จะหมุนไปก่อน ส่วนแชนแนล 2 จะหมุนตามไปที่หลังเนื่องจากค่า Delay Time ระหว่างแชนแนลมากนั่นเอง และการที่ต้องกำหนดค่า Delay Time นี้ให้มากหน่อยก็เพื่อให้ Servo หมุนมาถึงตำแหน่งที่เรากำหนดไว้จริงๆ ถ้า Delay Time น้อยไปจะทำให้ Servo หมุนมาไม่ถึงตำแหน่งที่เราต้องการ

*\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/----- **ET-RS SERVO V1.0** -----\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/*

## *GWS SERVO SPECIFICATIONS*

## STANDARD SERIES

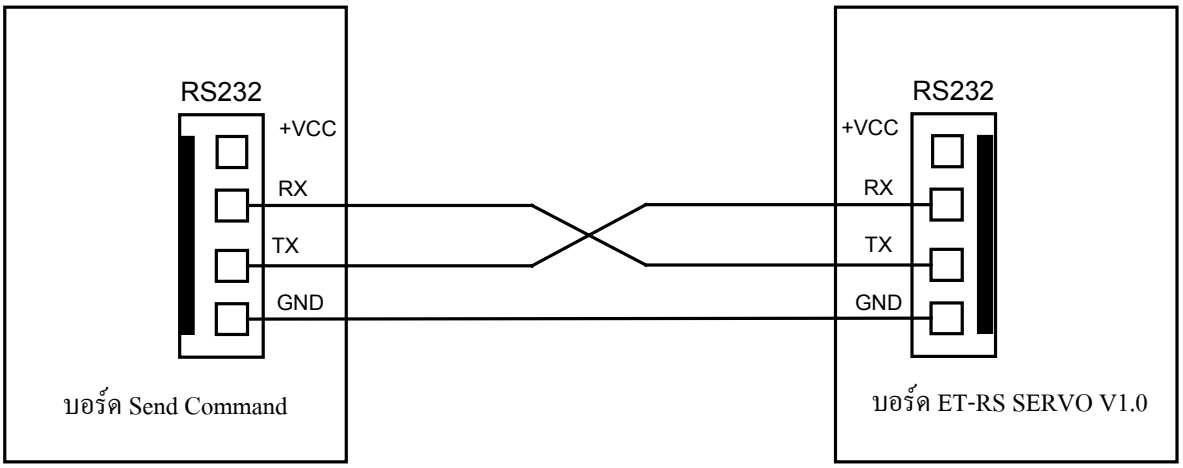




*Connector Servo*

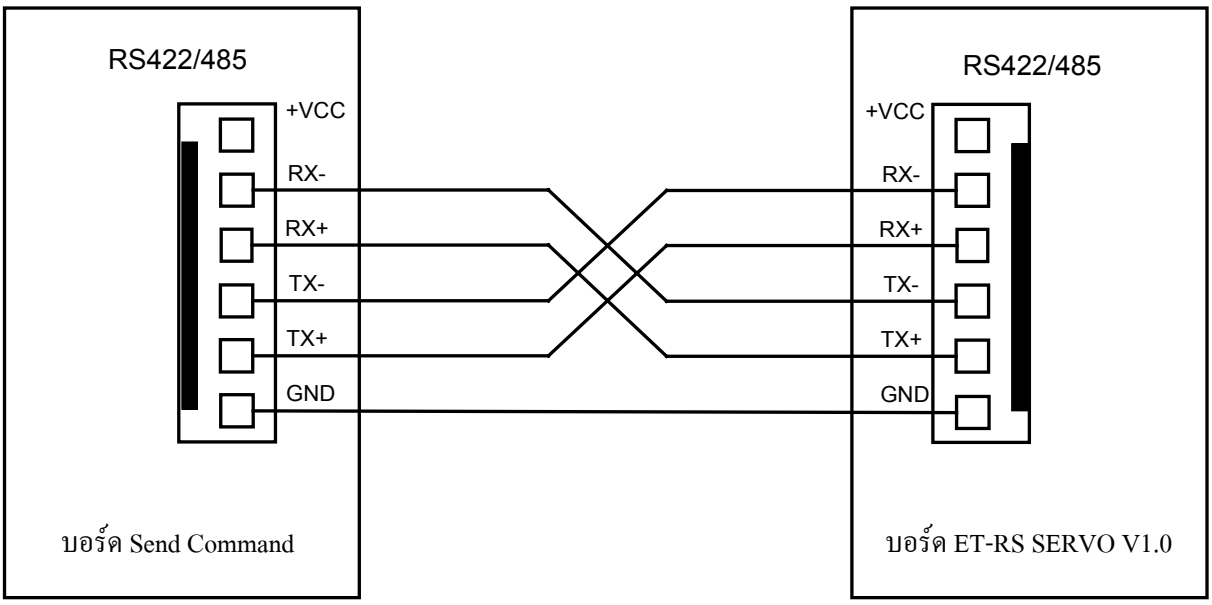
| Model | STD | 2BB | MG | Size ( L x W x H ) mm/in | Weight | | 4.8V | | | 6V | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | g | oz | Speed (sec/60°) | Torque kg-cm | Torque oz-in | Speed (sec/60°) | Torque kg-cm | Torque oz-in |
| S03N | ■ | ■ | | 39.5x20.0x35.6 1.56x0.79x1.40 | 41.0 | 1.45 | 0.23 | 3.40 | 47 | 0.18 | 4.00 | 56 |
| S03NF | ■ | ■ | | | | | 0.18 | 2.80 | 39 | 0.15 | 3.20 | 44 |
| S03NXF | ■ | ■ | | | | | 0.15 | 2.20 | 31 | 0.12 | 2.45 | 34 |
| S03N 2BBMG | | | ■ | 40.6x20.0x38 1.60x0.79x1.50 | 64 | 2.26 | 0.23 | 3.40 | 47 | 0.18 | 4.20 | 58 |
| S03NF 2BBMG | | | ■ | | | | 0.18 | 2.80 | 39 | 0.15 | 3.40 | 47 |
| S03NXF2BBMG | | | ■ | | | | 0.15 | 2.50 | 35 | 0.12 | 3.00 | 42 |
| S03T | ■ | ■ | | 39.5x20.0x39.6 1.56x0.79x1.56 | 46.0 | 1.62 | 0.33 | 7.20 | 100 | 0.27 | 8.00 | 111 |
| S03TF | ■ | ■ | | | | | 0.27 | 5.80 | 81 | 0.22 | 6.50 | 90 |
| S03TXF | ■ | ■ | | | | | 0.21 | 5.00 | 69 | 0.17 | 6.20 | 86 |
| S03T 2BBMG | | | ■ | 40.6x20.0x42.8 1.60x0.79x1.70 | 73 | 2.57 | 0.33 | 7.40 | 103 | 0.27 | 8.60 | 119 |
| S03TF 2BBMG | | | ■ | | | | 0.27 | 6.00 | 83 | 0.22 | 6.95 | 97 |
| S03TXF 2BBMG | | | ■ | | | | 0.21 | 5.60 | 78 | 0.17 | 6.40 | 89 |

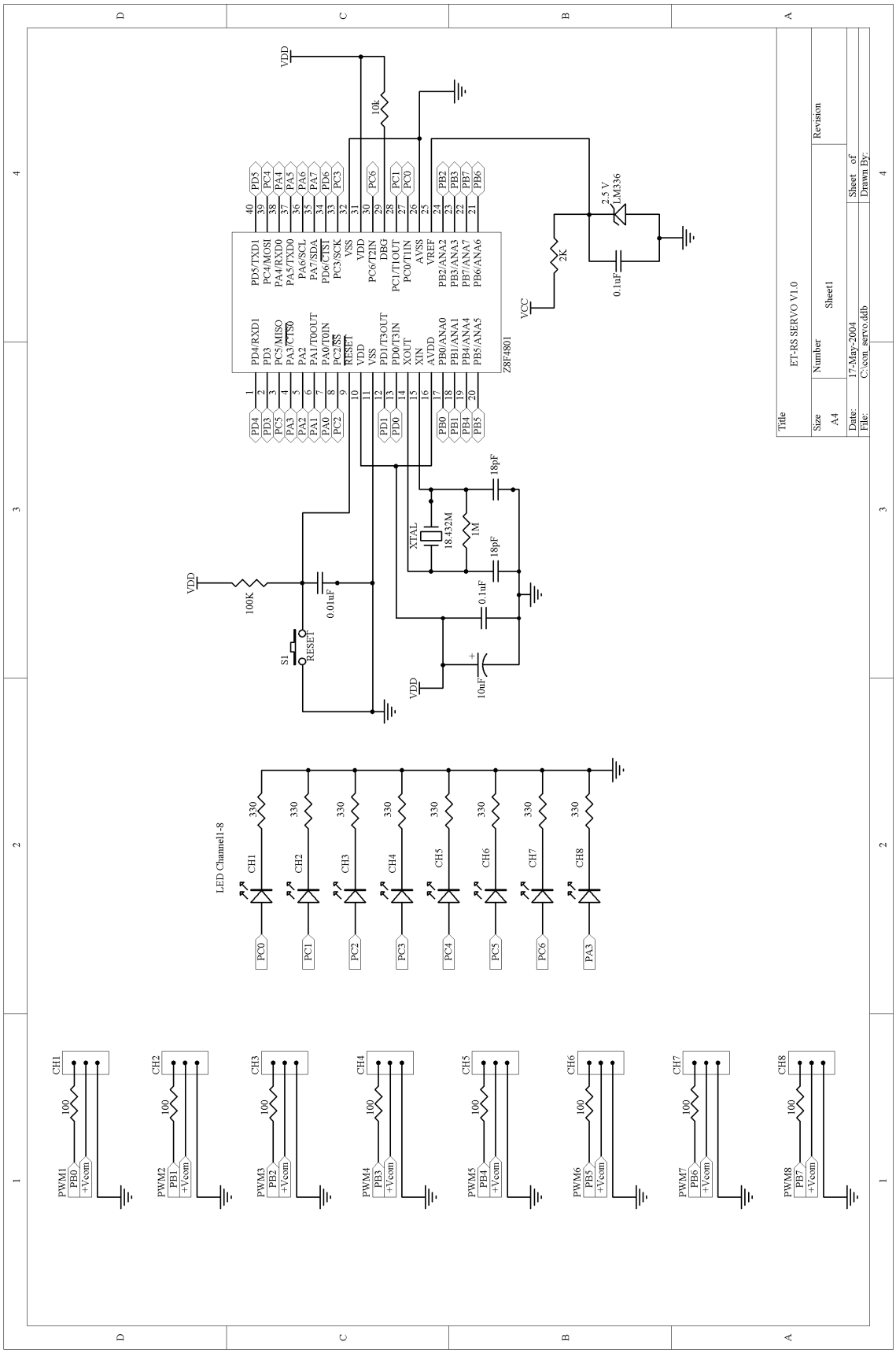
**STD = Oiliness Bearing  2BB = 2 Ball Bearings  MG = Metal Gear with Ball Bearings**

## *การต่อสายในการ รับ –ส่ง ข้อมูลระหว่างบอร์ด โดยใช้การสื่อสารแบบ RS232*

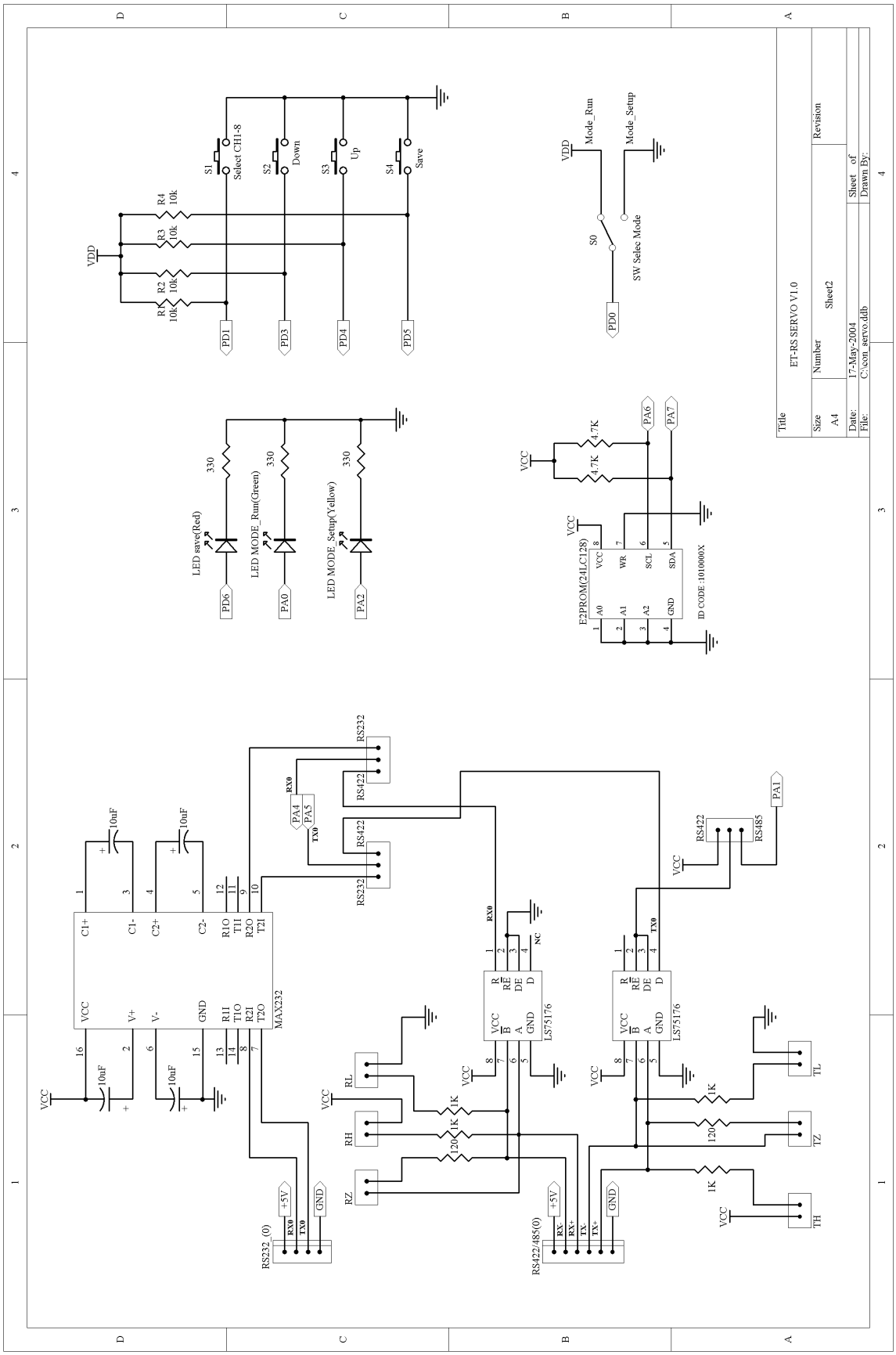



## *การต่อสายในการ รับ –ส่ง ข้อมูลระหว่างบอร์ด โดยใช้การสื่อสารแบบ RS422/485*

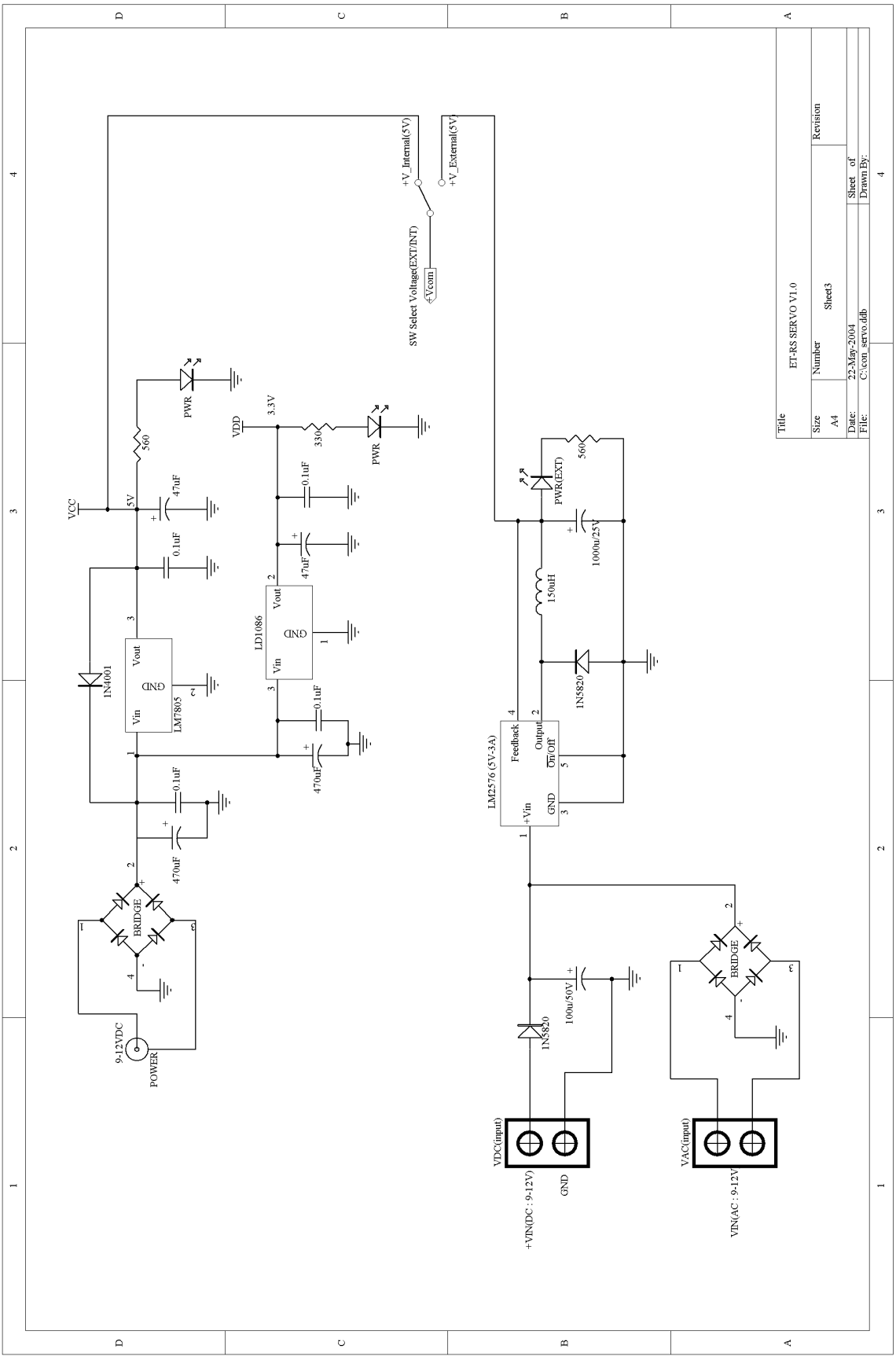





**รูป วงจรบอร์ด ET-RS SERVO V1.0 (Sheet1)**




**รูป วงจรบอร์ด ET-RS SERVO V1.0 (Sheet2)**




**รูป วงจรบอร์ด ET-RS SERVO V1.0 (Sheet3)**