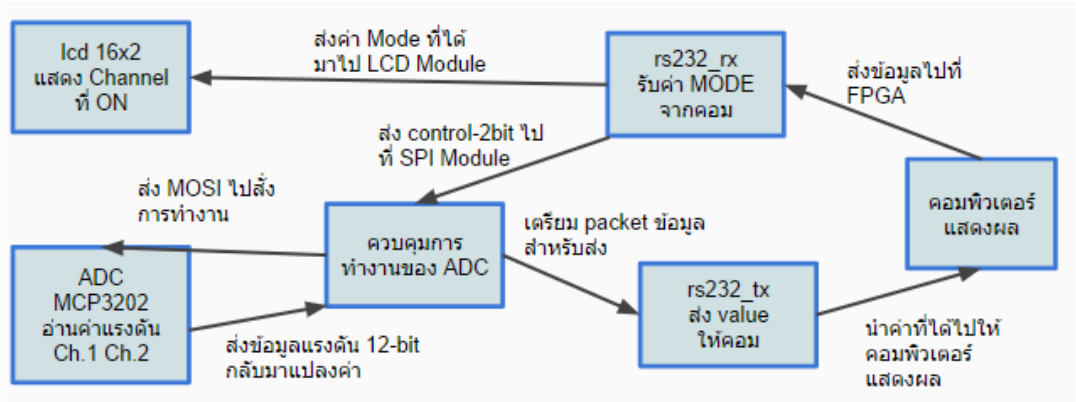
บทที่ 3

ขั้นตอนการทำงานและแนวคิดของ Hardware and software

ภาพรวมของระบบ



รูปภาพ 3.1 ภาพการทำงานโดยรวมของระบบ

การส่งข้อมูลจาก Computer เข้าบอร์ด FPGA

ในการส่งและรับข้อมูลระหว่าง computer และเข้าบอร์ด FPGA นั้นได้จะทำการกำหนด Protocol โดยสำหรับการส่งจะกำหนดไว้ดังนี้

[start byte] ใช้เป็น เลข 23ในHexหรือส่ง # จากคอม	[Mode byte] ใช้เลข 0 ถึง 3 ใน การกำหนด

รูปภาพ 3.2 ภาพของ Packet ที่จะทำการส่งไปให้ FPGA

โดยที่ Mode Byte นั้นจะแบ่งเป็น 4 คำสั่งหลักๆดังนี้

1. X”30” หรือ 0 ascii จะเข้าสู่โหมด OFF
2. X”31” หรือ 1 ascii จะเข้าสู่โหมด SingleCh1
3. X”32” หรือ 2 ascii จะเข้าสู่โหมด SingleCh2
4. X”33” หรือ 3 ascii จะเข้าสู่โหมด DualChannel

และเงื่อนไขเพิ่มเติมคือในตัว Control Module จะมีตัว Timer สำหรับตรวจจับเวลาที่ข้อมูลส่งมาจาก Computer หากตัวบอร์ด ไม่ได้รับ Start Byte ในเวลาที่กำหนดไว้ (3วินาที) ตัวบอร์ดจะเข้าสู่โหมด

Sleep คือไม่ทำการอ่านค่าจากทั้ง 2-Channel จนกว่าจะทำการเสียบสายแล้ว Reconnect ตัวบอร์ดกับ Computer

การส่งข้อมูลจาก Pc เข้าบอร์ด FPGA นั้นจะไม่สนใจที่ State ของ FPGA แต่จะได้รับ ผลกระทบมาจากโหมด ของ pc ณ ขณะนั้นๆแทน

โดย pc จะส่ง การตั้งค่าโหมดให้กับบอร์ด เมื่อ

1. input ที่ได้จากบอร์ดนั้น mode ไม่ตรงกับที่ pc ต้องการ pc จะทำการ ส่งตัวset ค่าโหมดให้กับบอร์ด และ ไม่สนใจค่าที่อ่านได้
2. เมื่อ วนครบรอบจำนวน หนึ่ง จะส่งการตั้งค่าโหมดไป เพื่อให้บอร์ดได้รับทราบว่า pc ยังรับข้อมูลอยู่

Mode	Output to FPGA [1byte/1package]
0	[start][mode]
1	[start][mode]
2	[start][mode]
3	[start][mode]

ตารางที่ 2

ตารางแสดง output จาก pc สู่ FPGA ตาม ค่าmode ต่างๆ ของ PC

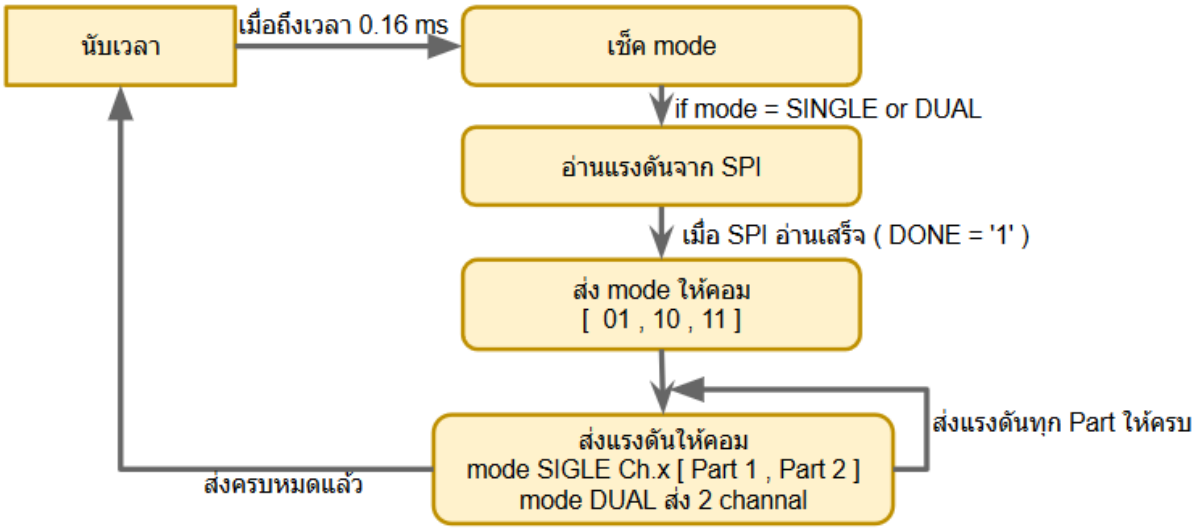
การส่งข้อมูล จาก บอร์ด FPGA เข้าเครื่อง Computer

ในการส่งข้อมูล จาก บอร์ด FPGA เข้าเครื่อง Pc นั้นจะขึ้นอยู่กับ Stage ณ ขณะนั้น ดังตารางข้างล่างนี้

Mode		Output to pc [1Byte/package]
Off	0	None
Single Ch	1	[Mode][value_1][value_2]
	2	[Mode][value_1][value_2]
Dual ch	3	[Mode][value_1_ch1][value_2_ch1] [value_1_ch2][value_2_ch2]

ตารางที่ 3

ตาราง แสดง รูปแบบ output จาก FPGA สู่ Pc ตาม stage ณ ขณะนั้นๆ



รูปภาพ 3.3 แสดงขั้นตอนการส่งข้อมูลจากบอร์ด FPGA เข้าเครื่อง Computer

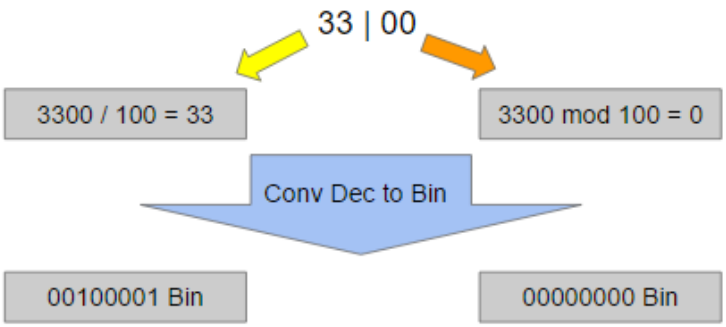
หลักการแปลงค่าเพื่อส่งกลับไปที่คอมพิวเตอร์

ข้อมูลที่ได้จาก SPI จะเป็น Binary 12-bit ซึ่งสามารถอ่าค่าได้ตั้งแต่ 0-4095 เราจะนำค่าส่วนนี้มาแปลงเป็นค่าแรงดันหลัก mV โดยใช้วิธีเทียบบัญญัติไตรยางค์ โดยค่าแรงดันที่สามารถอ่านได้คือ 3300 mV โดยจะใช้วิธีแปลงดังนี้

ค่าแรงดัน (mV) = (12-bit data convert to integer * 3300) / 4095

สมมุติให้ค่าแรงดันที่ได้จาก SPI คือ 4095 จะได้ค่าแรงดันที่แปลงแล้วเป็น (4095*3300) / 4095 = 3300 mV

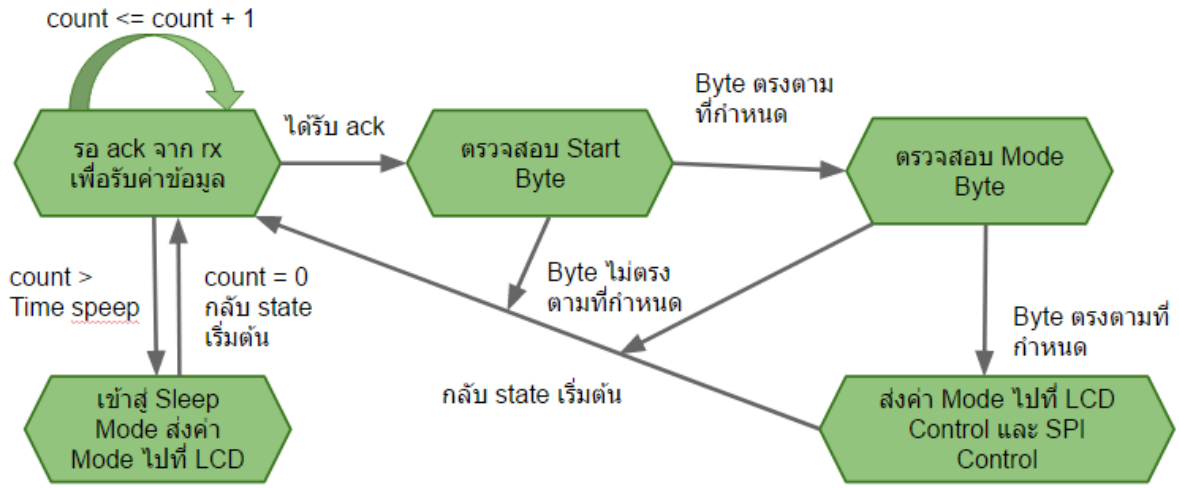
และนำข้อมูลที่ได้มาแบ่งออกเป็นข้อมูล 8-bit 2 package ส่งไปทีละ package โดยมีหลักการดังนี้ สมมุติให้ค่าแรงดันที่ได้คือ 3300 mV จะทำการแปลงค่าแล้วแบ่งข้อมูลเป็น 2 ส่วน 8-bit แรกได้จากการนำค่าแรงดันมาหารด้วย 100 และแปลงให้เป็น binary 8-bit แล้วส่งไปที่ Tx เพื่อส่งข้อมูลส่วนแรกมาเก็บที่ pc ก่อน ตามด้วยส่วนที่ 2 ได้จากการนำค่าแรงดันที่อ่านได้มา Mod ด้วย 100 และแปลงให้เป็น binary 8-bit แล้วส่งไปที่ Tx เช่นเดียวกับส่วนแรก



รูปภาพ 3.4 วิธีการแปลงข้อมูล 12-bit ให้เป็น 2-byte package

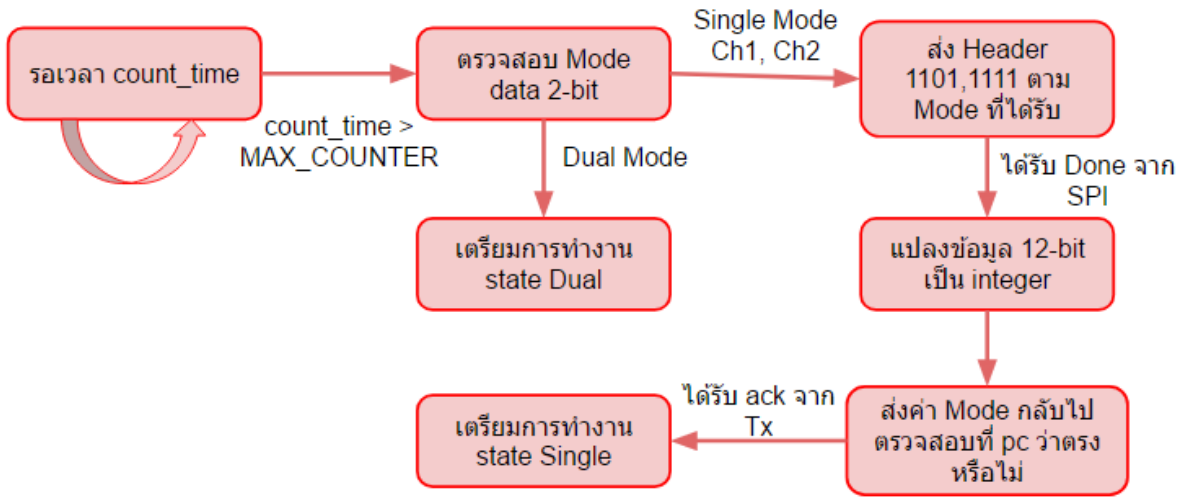
ข้อตกลงเกี่ยวกับ State และเงื่อนไข

State การตรวจสอบการเชื่อมต่อของ pc และ FPGA

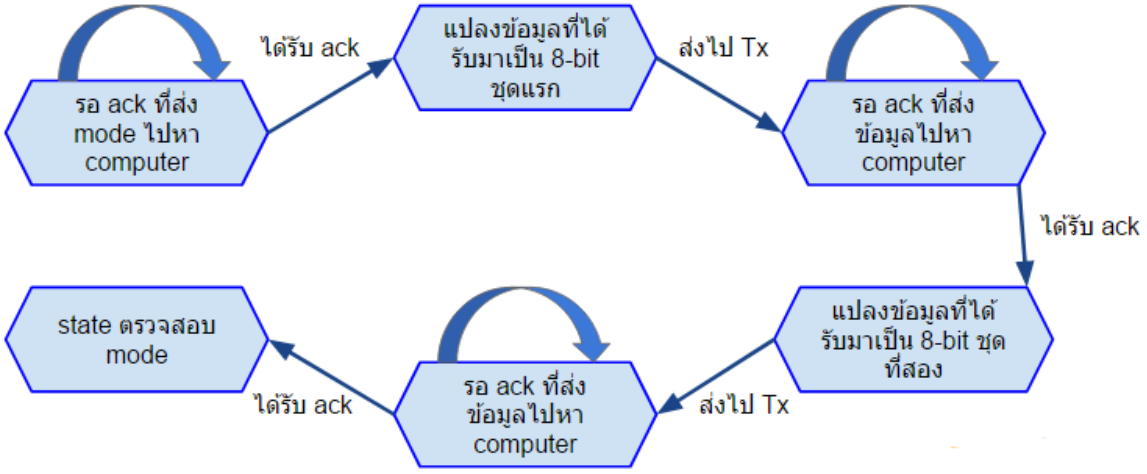


รูปภาพ 3.5 state การทำงานส่วนของ Control Module

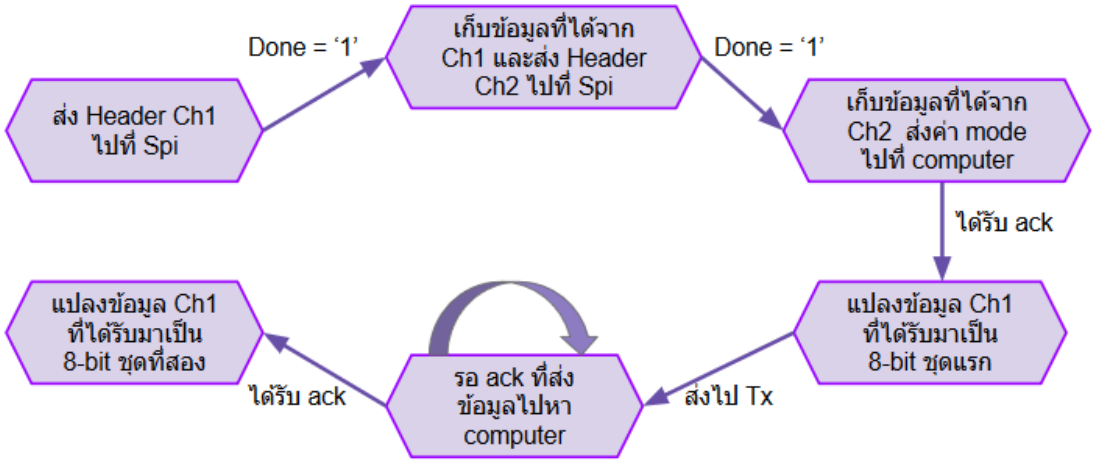
State การรับ-ส่งข้อมูลที่ได้จาก MCP3202

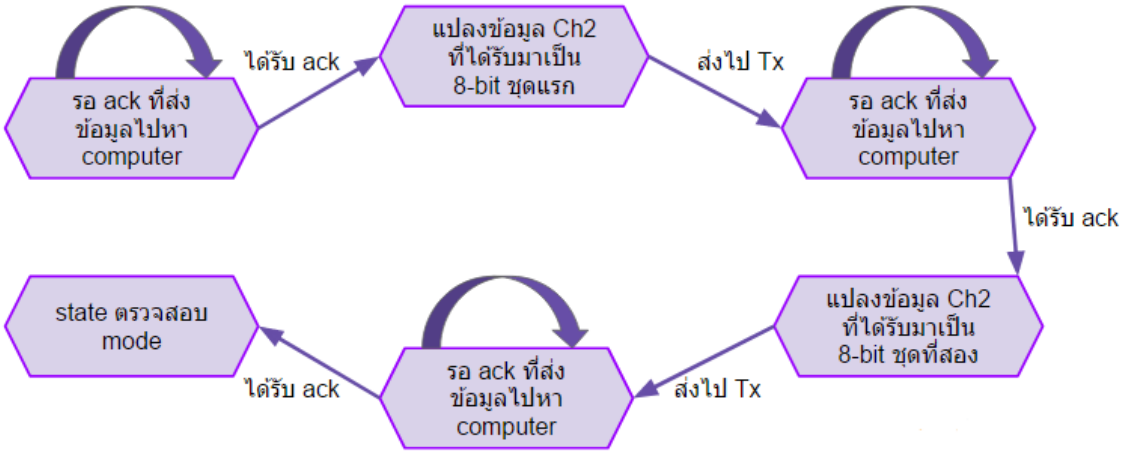


รูปภาพ 3.6 state การทำงานส่วนของ send_voltage_value



รูปภาพ 3.7 state การทำงานส่วนของ send_voltage_value (Single mode)





รูปภาพ 3.8 state การทำงานส่วนของ send_voltage_value (Dual mode)

การคำนวนข้อมูลที่จะส่งจากบอร์ด FPGA เข้าเครื่อง Computer

WAIT_COUNT_MAX (cycle) 1/2 SPI clk period	เวลาที่ใช้ใน 1 รอบการทำงาน		จำนวนรอบการทำงานใน 1 วินาที
100	3456 cycle	69.14 us	14451
50	1756 cycle	35.14 us	28409
25	906 cycle	18.14 us	54945

ตาราง 4 แสดงเวลาที่ SPI ใช้ในการทำงาน 1รอบ โดยที่ SPI clk period ที่แตกต่างกัน

สูตรการคำนวนเวลาที่ tx ใช้ส่งข้อมูล = n x [package size = 10bit] / baud rate

MODE	จำนวน Package	เวลาที่ใช้ในการส่ง		
		บอร์ดเรท 9600	บอร์ดเรท 115200	บอร์ดเรท 460800
OFF	0	-	-	-

SINGLE	3	3.125 ms	260.4 us	65.14 us
DUAL	5	5.208 ms	434.0 us	108.51 us

ตาราง 5 แสดงเวลาที่ tx ใช้ในการส่งข้อมูลต่อ 1รอบ ใน MODE ต่างๆ

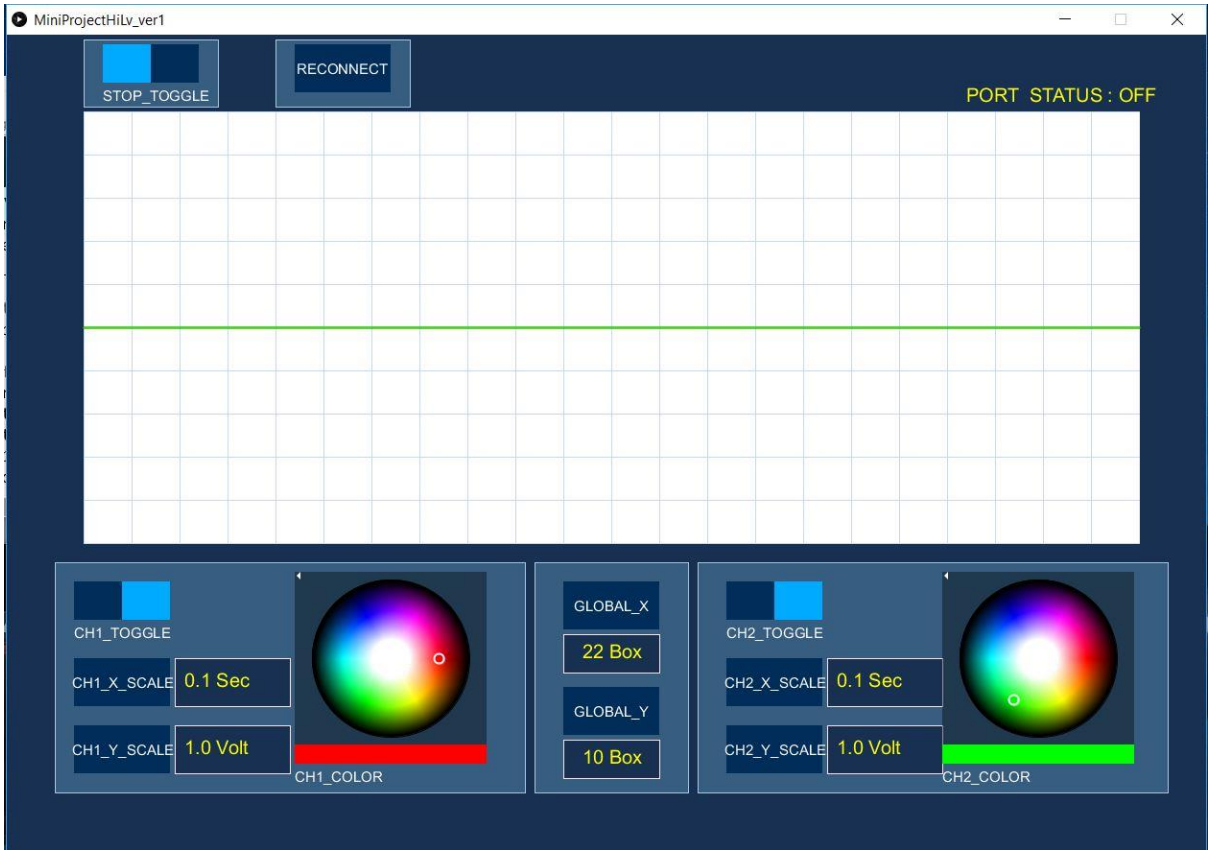
MODE	จำนวน Package	เวลาที่ใช้ในการส่ง		Total	Free Time
		½ SPI clock 25	บอร์ดเรท 460800		
OFF	0	-	-	0 us	160 us
SINGLE	3	18.14 us	65.14 us	82.28 us	77.72 us
DUAL	5	36.28 us	108.51 us	144.79 us	15.21 us

ตาราง 6 แสดงเวลาที่ใช้ในการวัดแรงดันจนถึงการส่งข้อมูลเสร็จเรียบร้อยใน MODE ต่างๆ

ในการกำหนดจำนวนครั้งในการวัดและส่งข้อมูลใน 1วินาที จะอ้างอิงจาก MODE ที่ใช้เวลามากที่สุด คือ MODE DUAL โดยใช้เวลาในการส่งถึง 144.79 us ต่อ 1ครั้ง ทำให้สามารถส่งข้อมูลได้มากถึง 6906 ข้อมูล ต่อวินาที แต่ว่า ในการใช้งานจริงจะใช้การส่งข้อมูล 6250 ข้อมูลต่อวินาที ทำให้ข้อมูลแต่ละข้อมูลมีระยะห่าง = 160 us

การออกแบบ Software และการทำงาน

ความสามารถในการใช้งานของ Software ที่ต้องการ คือ สามารถ นำข้อมูลจาก บอร์ด FPGA มาเพื่อแสดงผล ได้ บนจอ PC ผ่าน App ที่ได้ ทำมา โดย ที่ มีการแสดง สถาณะของการเชื่อมต่อ การหยุดภาพกราฟที่อ่านมาได้ , การกดเชื่อมต่อ กับบอร์ด , การ เปลี่ยนสีของเส้นแสดงแต่ละ ชาแนลได้ การ ปรับขนาดสเกล ในแกน y ที่เป็นขนาดแรงดันที่อ่านได้ หรือจะเป็นแกน x ว่า 1 ช่องที่ถูกแสดงผลออกมา เทียบได้กับเวลาเท่าไหร่ รวมถึง สามารถกำหนดว่าจะมีจำนวน colum และ row เท่าไหร่ อีกด้วย
โดยข้อมูลที่จะได้รับมานั้นจะได้รับมาทุกๆ 0.00016 วินาที หรือ 0.16 ms / 1 ข้อมูล ซึ่งท้ายที่สุดแล้ว ได้ผลลัพธ์ GUI ออกมาใน ลักษณะ นี้



รูปภาพ 3.9

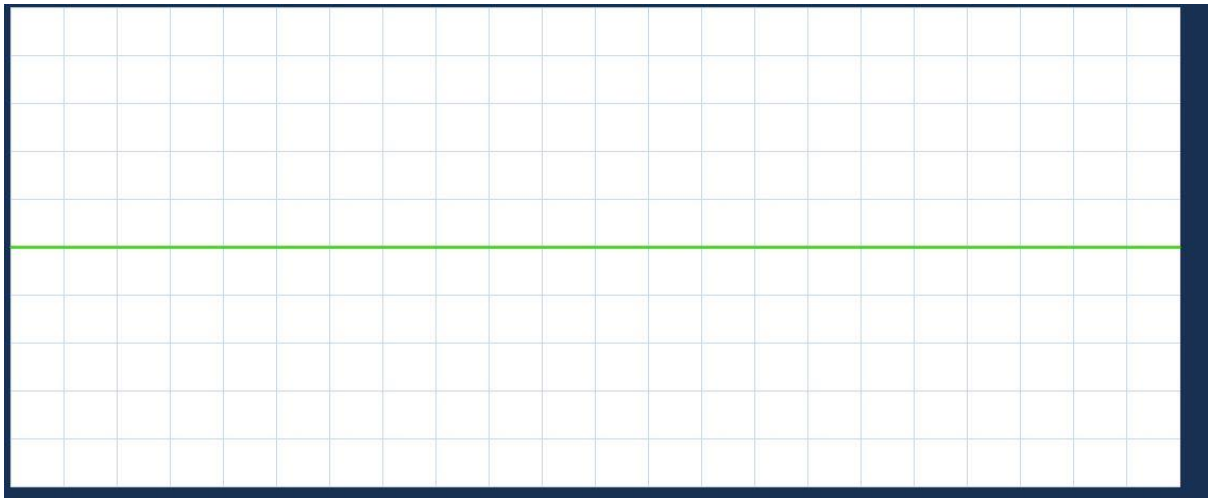
รูป ภาพหลัก GUI

จากภาพ จะเห็นได้ว่า มีส่วนประกอบหลัก ๆ ดังนี้ คือ

รูปภาพ 3.10

รูปอธิบายโปรแกรมส่วนหัว

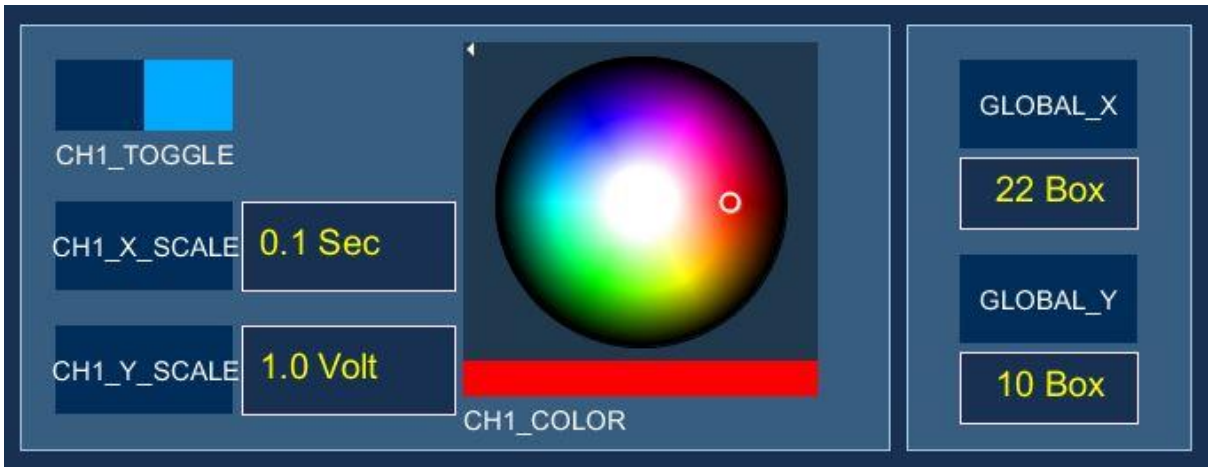
ทางด้านซ้ายมือ มีปุ่มสำหรับหยุดการ อัพเดทค่า กราฟ และ ปุ่มสำหรับติดต่อกับบอร์ดส่วนทางขวามือ มี การแสดงสถานะ ของการเชื่อมต่ออยู่



รูปภาพ 3.11

รูปอธิบายโปรแกรมส่วน การแสดงกราฟ

ตรงกลางหน้าจอ แอป จะมี กราฟ แสดงข้อมูลที่ได้รับมา โดยจะมี ช่อง colum และ row ที่สามารถ ปรับได้ ซึ่ง รายละเอียด จะอธิบายต่อภายหลัง



รูปภาพ 3.12

รูปอธิบายโปรแกรมส่วน การตั้งค่าต่างๆ

ต่อมาจะมีส่วนที่ใช้สำหรับ การปรับค่า ของ ch ซึ่ง ch 1 จะอยู่ทางซ้าย ส่วน ch2 จะอยู่ทางขวา ส่วน การปรับค่าจำนวน colum , row จะอยู่ตรงกลาง ซึ่ง ในการตั้งค่าของ แต่ละ ch นั้นจะมีการ ปรับได้ดังนี้ คือ การเปิดปิด การอ่านชาแนล การตั้งค่าสีของเส้น การตั้งค่าสเกล แกน x ว่า ใน 1 colum จะเท่ากับ กี่ sec หรือ การตั้งค่าสเกลแกน y ว่าใน 1 row นั้นหมายถึง กี่ Volt ซึ่งรายละเอียดการคิด จะอธิบายต่อภายหลัง

การทำตาราง **Colum , Row**

ใช้วิธีการ กำหนดเป็นค่าคงที่ ตามที่ต้องการก่อนโดยที่ค่าคงที่นั้นจะเป็นจำนวน Colum , Row ก่อน แล้วจากนั้น นำค่าเหล่านั้น ไปทำการวาดเป็นตาราง เมื่อ โปรแกรมสั่งให้วาดใหม่ โดยมีโค้ด ประมาณนี้

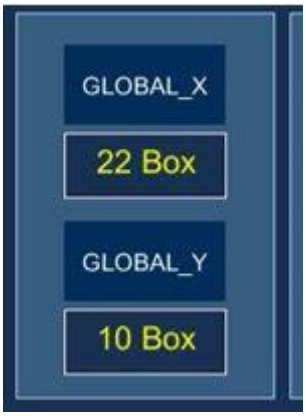
```
for(int colum = 0 ; colum < max_colum ; colum ++){
    line(x_app+80+(colum*multiple_of_colum), y_app+80, x_app+80+(colum*multiple_of_colum),
y_app+529);

  }
  for(int row = 0 ; row <= max_row ; row ++){
    line(x_app+80 , y_app+80+(row*multiple_of_row), x_app+1180, y_app+80+(row*multiple_of_row));
  }
```

โดย ที่ max_colum , max_row นั้นสามารถเปลี่ยนแปลงได้ และค่าตัวคูณ หรือ multiple_of_colum, multiple_of_row จะเปลี่ยนไปพร้อมกับ max_colum, max_row ตามที่ได้คำนวนไว้แล้ว ดังนี้

```
multiple_of_colum = 1100/max_colum;
multiple_of_row = 450/max_row;
```

โดยที่ 1100 นั้นมาจาก ความกว้างของกราฟเพื่อหาอัตราส่วน ที่จะต้องถูกกระจายไปตาม colum ส่วน 450 นั้นมาจากความสูงของกราฟ เพื่อนำมาหาอัตราส่วนที่จะต้องกระจายไปใน แต่ละ row นั้นเอง ดังรูปตัวอย่างข้างล่างนี้



รูปภาพ 3.13

การตั้งค่าสเกล colum,row ในตัวอย่างการคำนวนสเกล_1

รูปภาพ 3.14

การตั้งค่าสเกล colum,row ในตัวอย่างการคำนวนสเกล_2

ตารางแสดงจำนวนrow และ colum ทั้งหมดที่มี

Row	Colum

6	11
10	22
14	44

ตาราง 7
ตารางแสดงจำนวนrow และ colum ทั้งหมดที่มี

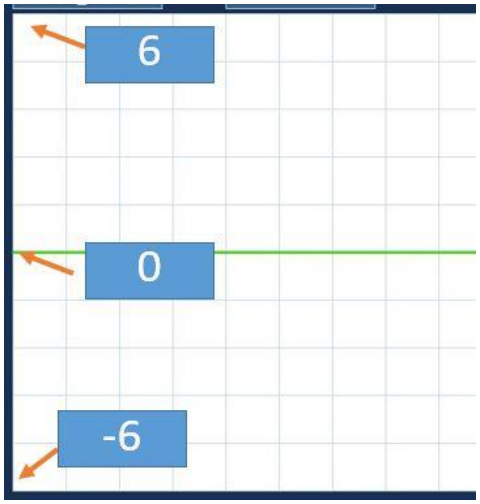
การทำ Scaling ในแกน Y ให้ให้ตามที่ตั้งไว้

ปัญหานี้เกิดมาจาก การที่ เราสามารถตั้งสเกลในแกน Y ว่า ใน 1row นั้นจะหมายถึง กี่ Volt ซึ่งในที่นี้เรากำหนดไว้ดังนี้

ความหมายใน 1 row
0.5 Volt
1 Volt
2 Volt
3 Volt

เราก็จะเจอปัญหาที่ว่า Input ที่เราจะได้รับมาจาก บอร์ดนั้น ไม่ได้ทำการสเกลมาให้เหมาะสม หรือก็คือเป็นค่าดิบที่อ่านได้โดยตรงนั้นเอง แต่ว่า controlP5 นั้นจะช่วยในการการ Chart ให้เราโดยอัตโนมัติ โดยเทียบกับ max min ของ chart ตามที่กำหนดไว้แต่แรก

หรือก็คือ 6 และ - 6 ในที่นี้ ซึ่งก็ทำให้เราจำเป็นต้องมากำหนด ขนาดให้กับค่าที่ได้รับมาใหม่ตามสเกลที่เราต้องการ



รูปภาพ 3.15

รูปการแสดงค่าที่ controlP5 จะนำไป ทำเป็นกราฟ

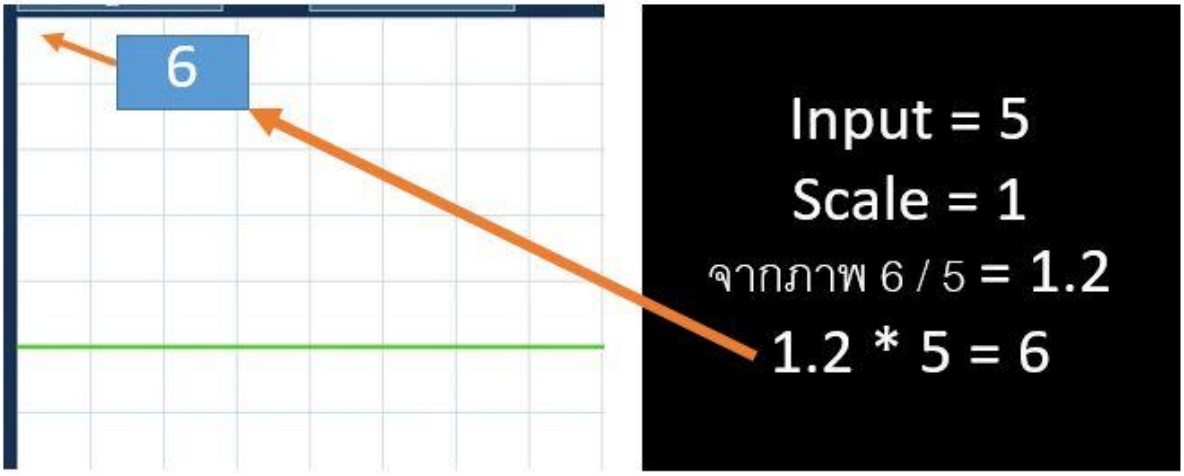
แต่ถ้าหากเราตีความตามที่เราตั้งค่าสเกลไว้ เช่น 1 ช่อง เท่ากับ 1 Volt จากภาพก็จะพบว่า มันควรจะสูงสุดที่ 5 Volt เท่านั้นหรือก็คือ 5 Volt ของ input ที่เข้ามาจะหมายถึง 6 Volt เวลาเราจะทำการเพิ่มค่าให้กับ ControlP5 นั้นเอง หรือก็คือ การคำนวนจะได้ว่า

input_to_controlP5 = (range_y_max / ((max_row/2)*scale_y)) * input

หรือถ้าตามตัวอย่างด้านบนก็คือ

input_to_controlP5 = (6/5) * input

ซึ่งการคำนวนมาจากแนวคิด ได้ว่า ถ้า Y_max คือค่า max และมีจำนวนช่องเท่ากับ row จะหมายความว่า ใน 1 ช่อง จะมีค่า คือ Y_max/row ซึ่งเป็นการกระจายเท่ากันจากนั้น นำไปดูว่า ค่าใน 1 ช่องนั้น เป็นกี่เท่าของสเกล ที่เราจัดการอยู่ เช่นสเกล เป็น 1 ก็ คือ (Y_max/row)/1 = r ที่เมื่อนำไปคูณกับ input จะได้ ค่าที่อยู่ในการพล๊อต chart ได้ ดังตัวอย่างรูปด้านล่าง เมื่อกำหนด สเกลเป็น 1 และ input = 5



รูปภาพ 3.16

รูปตัวอย่างการสเกลแกนY

ซึ่งดังตัวอย่างนั้นจะเห็นได้ว่า ถ้า input = 5 นั้น input ที่ controlP5 จะนำไปใช้งานจะได้ค่า เป็น 6 ก็จะเกิดจุดบน ตำแหน่ง สูงสุด ซึ่งคนที่ดูก็จะเข้าใจได้ว่าเป็น 5 เพราะคนใช้งานกำหนดไว้ว่า 1 ช่องคือ 1 volt และบนสุดนั้น อยู่บนช่อง ที่ 5 นั้นเอง

การทำ Scaling ในแกน X ให้ให้ตามที่ตั้งไว้

การ scaling ในแกน x หรือแกน เวลานั้น เกิดจากการที่เราต้องการจะ ทำการให้ผู้ใช้งานสามารถกำหนดได้ว่าใน 1 ช่อง ที่ปรากฏนั้นหมายถึง กี่วินาที นั้นเอง ซึ่ง เนื่องจากเราได้กำหนดอัตราการส่งค่าจาก บอร์ดเป็นค่าคงที่ หรือ เท่ากับ 0.16 ms/1package นั้นจึงทำให้เราจำเป็นต้องมาเปลี่ยนวิธีการเก็บ หรือจำนวน Buffer ให้สอดคล้องกันแทน ซึ่ง controlP5 นั้น จะมีการใช้ การเก็บ Buffer เอาไว้ ก่อนนำไปแสดงผล แต่ เราจะไม่สามารถกำหนดได้ว่าจะให้มัน map อย่างไร controlP5 จะทำแบบอัตโนมัติโดยการ นำค่าใน Buffer ไป map กับ ความกว้างของ Chart หรือความกว้างในแกน x แบบอัตโนมัติ และทำการวาดกราฟ ตามจุด Buffer ที่ได้ map ไว้แล้ว ทันที ซึ่งในความหมายที่ว่ามานี้ หมายความว่าทุกๆ ครั้งที่ มีการอัพเดท ค่าใน บัฟเฟอร์ หรือพูดอีกอย่างคือ
มีการอ่านค่าจากบอร์ดคำนวนและแปลงเสร็จแล้ว อัพเดทเข้าไปใน Buffer ซึ่ง controlP5 จะทำการเพิ่มไปที่ตัวท้ายสุดและ นำตัวแรกออก นั้น จะเกิดการขยับของกราฟ ได้

ผมจึงมีแนวคิดที่ว่า ถ้าหาก เราทำการ กำหนด ให้ 1 ช่อง colum นั้นมีจำนวนบัฟเฟอร์ ที่เมื่อนำไป คูณกับ ค่าคงที่ของการอัพเดทได้ แล้วจะได้ ค่ามาค่าหนึ่งซึ่งมีความหมายว่า ใน1 ช่องที่มีบัฟเฟอร์เท่านี้นั้นจะหมายถึง กี่วินาที ดังสมการนี้

เวลาใน 1 colum = จำนวน buffer ใน 1 colum * time_update

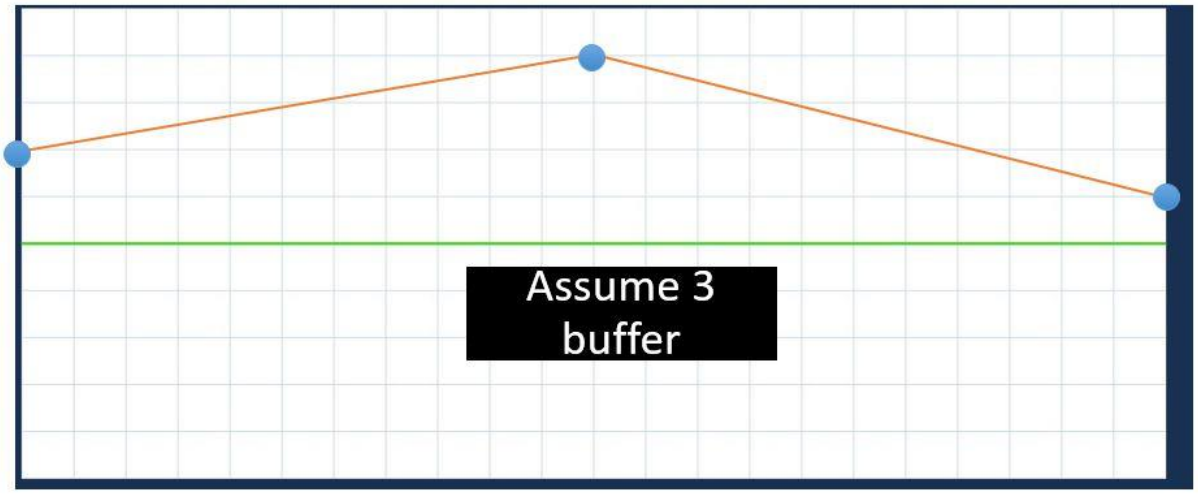
ซึ่งถ้าหากนำมาคิดกลับกันแล้วหากเราต้องการ กำหนด เวลาใน 1 colum ตามที่เราต้องการได้ นั้นก็มีแต่ต้องแก้จำนวน Buffer ใน 1 colum เท่านั้น จากนั้นจึงได้สูตรคำนวน นี้มา

buffer_per_colum = time_per_colum / time_update

แต่เนื่องจาก มันไม่ได้มีเพียง colum เดียวจึงจำเป็นต้อง คูณด้วยจำนวน colum ทั้งหมดเพื่อให้กลายเป็น จำนวน buffer ทั้งหมดที่ต้องใช้

buffer_all = (time_per_colum / time_update) * max_colum

เมื่อได้ค่าจำนวน buffer ที่ต้องใช้งานแล้วก็นำไปอัพเดทให้ controlP5 ทุกครั้งที่มีการเปลี่ยนแปลง scale นี้



รูปภาพ 3.17

ตัวอย่างการแสดงผล เมื่อมี 3 Buffer โดยให้จุดแทน ตำแหน่งบัฟเฟอร์

และถ้าหากใช้การสมสมุติตามรูปข้างต้นนี้ เราจะได้ ว่า มี 22 colum และ มีจำนวน buffer = 3 ถ้าให้อัตราการส่งเท่ากับ 0.16ms แล้วละก็ จะได้ว่า เรากำหนดให้ 1 ช่อง = (3/22)*0.16 ms

1 colum = 0.0218 ms นั้นเอง (ที่ต้องหาร 22 เพราะ 3 นั้นเป็นจำนวน buffer ทั้งหมด แต่เราต้องการ จำนวน buffer / 1 colum)

ซึ่งเราได้กำหนดไว้ดังนี้

ความหมายใน 1 colum
0.1 s
0.2 s
0.05 s
0.025 s
0.01 s
1 ms
0.5 ms
0.25 ms

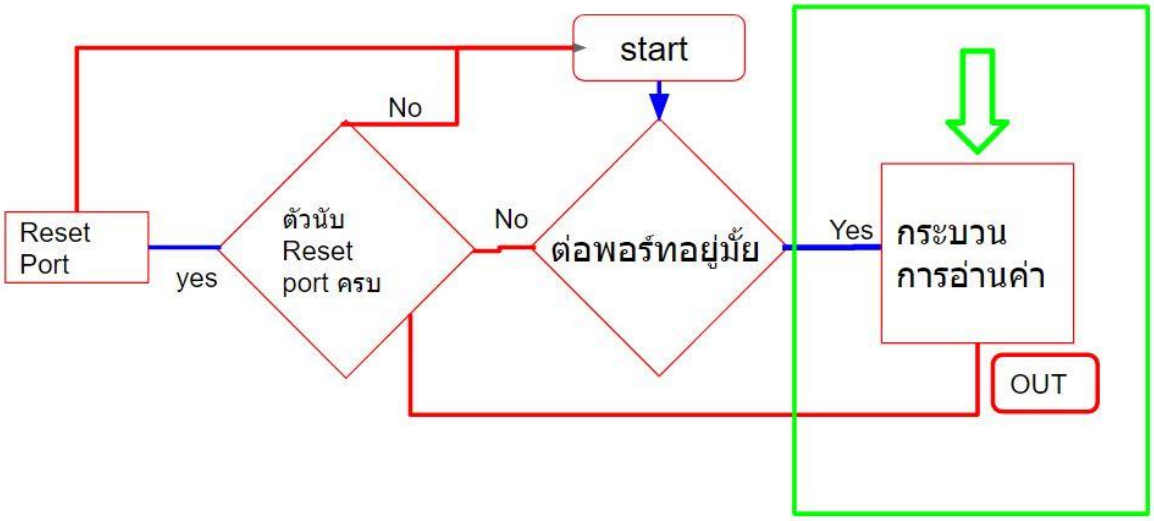
ขั้นตอนการทำงาน

การทำงานทางฝั่งคอมพิวเตอร์ในส่วนของการรับค่านั้น จะมีกระบวนการคือ

1. พอร์ต ได้ต่ออยู่มั้ย ซึ่งถ้าหากต่อ จะทำการ เช็คว่าพอร์ทที่ต่อนั้นเหมือนเดิมหรือไม่ และถ้าเหมือนเดิม จะไป ตั้งค่า time_out ที่จะเป็นตัว ตัดการเชื่อมต่อ ให้เริ่มต้นใหม่ ซึ่งหมายความว่าหากไม่เข้ากระบวนการนี้ เมื่อถึงระยะเวลาหนึ่ง pc จะทราบว่าพอร์ทไม่ได้ต่อและทำการ set port = null
2. ถ้าต่อพอร์ทอยู่ต่อมา ก็จะทำการเช็คว่า โหมด stop_update ทำงานมั้ยถ้าไม่ ก็จะเข้าสู่กระบวนการที่ 3
3. ถ้าหากมีข้อมูลมากกว่าค่า delay ที่กำหนดไว้(ขึ้นอยู่กับโหมดการทำงาน) ถ้าครบก็เข้าขั้นตอนที่ 4
4. ขั้นตอนที่4 ทำการ อ่านค่าที่ได้รับมา ถ้าหาก ว่าค่าที่ได้รับมามีค่าไม่ตรงกับโหมดปัจจุบัน (ขั้นตอนเช็คโหมด) จะทำการ ส่งการตั้งค่าโหมดกลับไปที่บอร์ด เซ็ทตัวนับการตั้งค่าเป็น 0 และทำการเคลียพอร์ท (ลบข้อมูลที่ค้างทั้งหมด) แต่ถ้าโหมดตรงแล้ว จะทำการอ่านค่าต่อๆไปเรื่อยๆตามโหมดที่กำหนดเอาไว้ และนำไปคำนวน ปรับสเกล ต่างๆ และ เพิ่มเข้าไปใน ControlP5
5. หากข้อมูลที่เข้ามายังมากกว่า 5 ให้กลับไปขั้นตอนที่ 4 ซึ่ง 5 มาจากความต้องการขั้นต่ำของโหมด 2 ว่าต้องใช้ 5 package ในการอัพเดทนั้นเอง

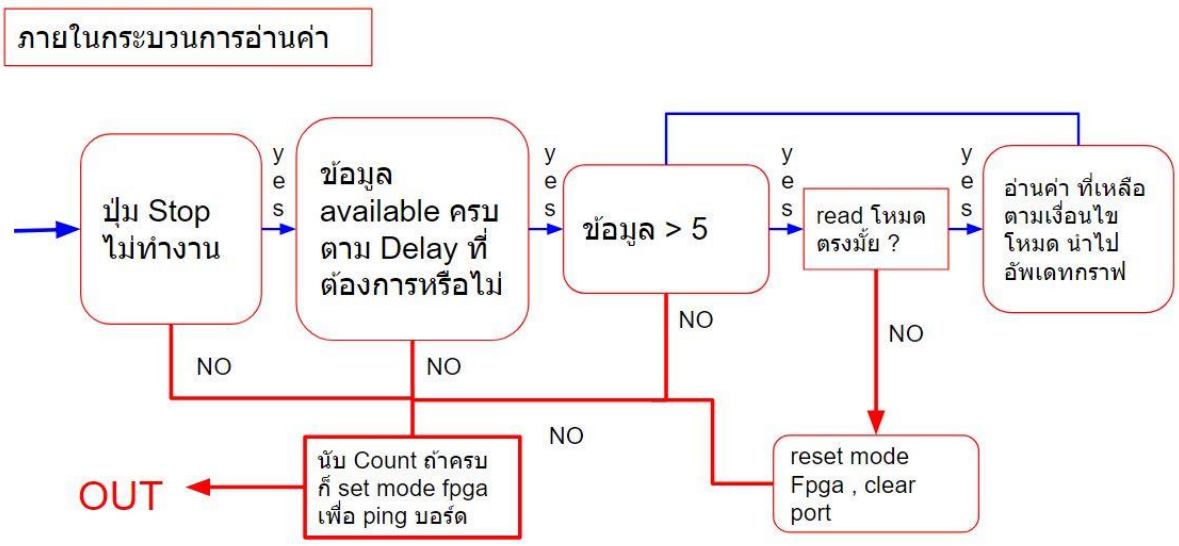
6. ต่อมา ถ้าผ่าน เงื่อนไขในข้อ 1 มาได้ ก็จะทำการ เพิ่มค่า confirm_connect ที่ถ้าหาก ครบตามจำนวนเงื่อนไขแล้วจะทำการ ส่งคำสั่ง set mode ให้บอร์ดเสมอ เพื่อเป็นการ บอกบอร์ดว่า ยังเชื่อมต่อกันอยู่
   (ถ้าบอร์ดไม่ได้รับการตั้งค่าโหมดในช่วงเวลาหนึ่งจะเกิดสถาณะ sleep)
7. หลังจากผ่านขั้นตอนข้างต้นด้านบนแล้ว จะมีการเช็ค time_out ถ้าหากมันมากกว่าเวลาหนึ่ง จะทำการ reset port = null เพราะมันหมายความว่า พอร์ท ที่ต่อนั้นได้หายไปแล้ว

รายละเอียดเพิ่มเติม ดูที่ภาคผนวค ในส่วนโค้ด การทำงาน



รูปที่

รูปภาพรวม ของการทำงานโปรแกรม



รูปที่

รูปแสดงกระบวนการของโปรแกรมสำหรับกรณี พบพอร์ท

การส่งคำสั่ง set mode นั้นจะ ใช้คำสั่ง myPort.Write('#') เป็นตัวกำหนดหัวก่อนจากนั้นตามด้วย myPort.write(str(mode)) เพื่อบอก สถาณะของคอม ณ ขณะนั้นๆ

ปัญหาพื้นฐาน

ปัญหา	ทางแก้
บอร์ดจะรู้ได้อย่างไรว่าpc ต่ออยู่	pc จะต้องทำการส่งค่า mode ให้กับ บอร์ดทุกๆ 3 วินาที ถ้าบอร์ด นับครบ 3 วิแล้วไม่มีการส่ง mode มาจะเข้าโหมด sleep หรือไม่อ่านไม่ส่งค่าใดๆ
pc จะรู้ได้อย่างไรว่า ต่อบอร์ดอยู่	มีตัวแปรคอยเช็ค ถ้าหากไม่มีการ reset ค่านี้ซึ่งการ reset จะต้องผ่านเงื่อนไขสำหรับเช็คการต่อบอร์ดอยู่ จะทำให้ pc นั้นตั้งค่า port = null เพื่อตัดการเชื่อมต่อกับพอร์ด และหยุดการอัพเดท
บอร์ดจะรู้หรือไม่ว่า pc อยู่ในโหมดเดียวกัน	รอ คำสั่งจาก คอม เท่านั้น ในระหว่างนั้นก็ส่งmode ปัจจุบันไปเรื่อยๆ

pc จะรู้ได้อย่างไรว่าบอร์ดอยู่โหมดไหน	ถ้า mode ที่ได้รับมาพร้อม value ไม่ตรงกับ mode ใน pc จะทิ้งค่าที่ได้รับมาและส่งคำสั่ง set mode ให้กับบอร์ด