

วิชา Data Communication Laboratory
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 3 Ring Communication

วัตถุประสงค์

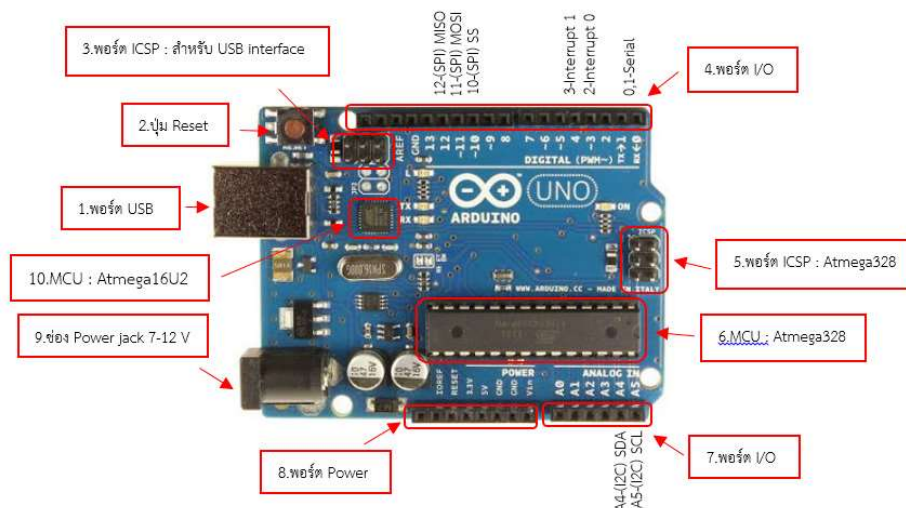
1. เพื่อให้เข้าใจหลักการสื่อสารผ่านพอร์ทอนุกรมของ Arduino
2. สามารถเขียนโปรแกรมเพื่อติดต่อสื่อสารระหว่าง Arduino ด้วยการเชื่อมต่อแบบ Ring Communication
3. สามารถเขียนโปรแกรมประยุกต์เพื่อติดต่อสื่อสารระหว่างคอมพิวเตอร์ และ Arduino ได้

การเขียนโปรแกรมลงบน Arduino

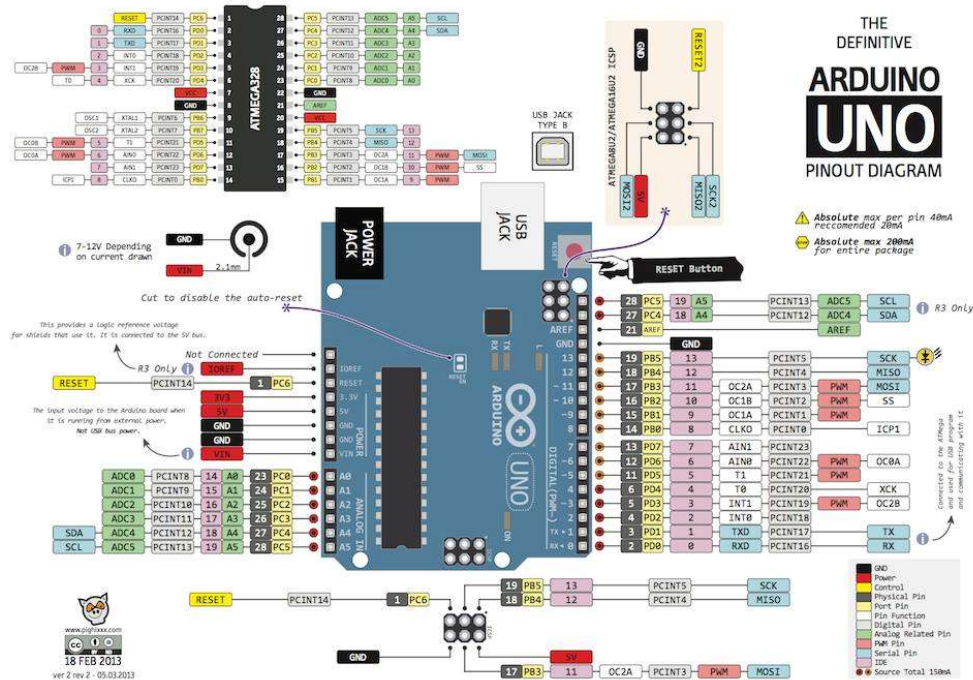
Arduino อ่านว่า (อา-ดู-อิ-โน้ หรือ อาดูยโน้) เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ Open Source คือ มีการเปิดเผยข้อมูลทั้งด้าน Hardware และ Software โดยสามารถใช้งานได้ง่ายและสามารถใช้ในติดต่อสื่อสารกับอุปกรณ์ภายนอกผ่านทางขา I/O ของบอร์ดได้ง่าย โดยขา I/O ของ บอร์ด ประกอบด้วย (โครงสร้างบอร์ด Arduino UNO R3 แสดงในรูปที่ 3.1 และ 3.2)

1. ขารองรับสัญญาณ Analog (A0-A5) จำนวน 6 ขา
2. ขารองรับสัญญาณ Digital (0-13) จำนวน 14 ขา ซึ่งบางขาสามารถโปรแกรมให้สร้างสัญญาณ Pulse Width Modulation (PWM) (3, 5-6, 9-11) จำนวน 6 ขา

การเขียนโปรแกรมควบคุมการทำงานของบอร์ด Arduino สามารถเขียนผ่านทาง Arduino IDE ภาษาที่ใช้จะเป็น ภาษา C ที่มีโครงสร้างการเขียนพื้นฐานที่แตกต่างจากภาษา C ทั่วไป คือ จะมีการทำงานแบบวนทำซ้ำ (loop) ตั้งแต่โปรแกรมบรรทัดแรก จนถึง บรรทัดสุดท้าย ใน loop () หลังจากนั้นจะวนกลับมาเริ่มทำงานบรรทัดแรกอีกครั้ง



รูปที่ 3.1 บอร์ด Arduino UNO R3 (<http://www.myarduino.net>)



รูปที่ 3.2 โครงสร้างบอร์ด Arduino UNO R3 (<http://www.myarduino.net>)

โครงสร้างโปรแกรมจะแบ่งเป็น 2 ส่วน ประกอบด้วย

1. ส่วนหัวโปรแกรม ซึ่งคล้ายกับภาษา C ทั่วไป ประกอบด้วย 프리โปรเซสเซอร์ไดเรกทีฟ (Preprocessing Directives) ตัวแปรชนิดโกลบอล (Global Variable) เป็นต้น ซึ่งในเขียนการโปรแกรม Arduino พื้นฐานหากไม่ใช้ก็ไม่จำเป็นต้องมี
2. ส่วนตัวโปรแกรมต้องประกอบด้วยฟังก์ชันหลัก 2 ฟังก์ชัน ได้แก่
 - 1) ส่วน `setup()` เป็นส่วนตั้งการทำงาน เช่น ตั้งค่าการสื่อสารระหว่างบอร์ด Arduino กับคอมพิวเตอร์ผ่านทาง Serial communication หรือ ตั้งค่ารูปแบบของขา I/O เป็นต้น
 - 2) ส่วน `loop()` เป็นส่วนการทำงานของโปรแกรมจริงที่วนทำซ้ำตลอด

คำสั่งเบื้องต้นในการเขียนโปรแกรมเพื่อติดต่อกับ I/O

1. **`pinMode(pin, mode)`** ใช้กำหนด mode การทำงานให้กับขา I/O ที่ต้องการใช้
 เมื่อ `pin` : หมายเลขขา I/O ของบอร์ดที่ต้องการกำหนด
`mode` : เลือกเป็น INPUT, OUTPUT หรือ INPUT_PULLUP
2. **`digitalWrite(pin, logic)`** ใช้ส่งข้อมูล Digital ไปยังขา I/O ที่ต้องการ
 เมื่อ `pin` : หมายเลขขา I/O ของบอร์ดที่ต้องการใช้ส่งข้อมูล
`logic` : ข้อมูลที่ต้องการส่ง HIGH หรือ LOW
3. **`digitalRead(pin)`** ใช้อ่านค่าข้อมูล Digital จากขา I/O ที่ต้องการ
 เมื่อ `pin` : หมายเลขขา I/O ของบอร์ดที่ต้องการใช้รับข้อมูลหลังจาก Return ค่า โดยมีสถานะข้อมูล เป็น HIGH หรือ LOW

4. **analogWrite(pin, duty)** ใช้ส่งข้อมูล Analog PWM ไปยังขา I/O ที่ต้องการ
 เมื่อ pin : หมายเลขขา I/O ของบอร์ดที่ต้องการใช้ส่งข้อมูล
 duty : ความกว้างของ Active pulse (PWM duty cycle) ตั้งแต่ 0 – 255 (8 bits)
 หมายเหตุ ความถี่ของสัญญาณ PWM ของขา 3, 9-11 ประมาณ 490 Hz
 ความถี่ของสัญญาณ PWM ของขา 5-6 ประมาณ 980 Hz
5. **analogRead(pin)** ใช้อ่านค่าข้อมูล Analog จากขา I/O ที่ต้องการ
 เมื่อ pin : หมายเลขขา I/O ของบอร์ดที่ต้องการใช้รับข้อมูลหลังจาก Return ค่า
 หมายเหตุ เนื่องจากส่วนแปลงสัญญาณจาก Analog-to-Digital (ADC) ของบอร์ด Arduino UNO R3 มีความเร็วที่ไม่สูงมาก อยู่ที่ประมาณ 80 KHz (เมื่อไม่มีคำสั่งอื่นๆ) และ มีความละเอียดอยู่ที่ 0-1024 (10 bits) รับสัญญาณ Analog อยู่ใน ช่วง 0-5 Volt (ไม่สามารถให้สัญญาณติดลบได้) ทำให้ความละเอียดของสัญญาณที่อ่านได้ อยู่ที่ 5V / 1024 หน่วย หรือ 0.0049V / หน่วย โดยความละเอียดนี้สามารถปรับเปลี่ยนได้โดยใช้คำสั่ง analogReference() บอร์ด Arduino จะใช้เวลาในการอ่านข้อมูล Analog (รับข้อมูล Analog และแปลงเป็น Digital) ประมาณ 100 ms (0.0001 s) ทำให้อัตราการอ่านข้อมูลสูงสุดได้เพียง 10,000 ครั้ง / วินาที เท่านั้น
 ถ้าขา Analog ไม่มีการเชื่อมต่อกับอุปกรณ์ใด จะทำให้ค่าที่อ่านได้จากคำสั่ง analogRead() เป็นค่าไม่แน่นอน
6. **analogReference (Type)** ใช้ในการตั้งค่า Voltage อ้างอิงสูงสุดของ Analog input เมื่อกำหนดค่า Type เป็น

DEFAULT	: จะใช้ค่า 5V สำหรับบอร์ดที่จ่ายไฟ 5V และ 3.3V สำหรับบอร์ดที่จ่ายไฟ 3.3V
INTERNAL	: จะใช้ค่าอ้างอิงภายในบอร์ด 1.1V สำหรับ ATmega168 / ATmega328 และ 2.56V volts สำหรับ the ATmega8 (ไม่ใช้กับบอร์ด Arduino Mega)
INTERNAL1V1	: จะใช้ค่าอ้างอิงภายในบอร์ด 1.1V สำหรับบอร์ด Arduino Mega เท่านั้น
INTERNAL2V56	: จะใช้ค่าอ้างอิงภายในบอร์ด 2.56V สำหรับบอร์ด Arduino Mega เท่านั้น
EXTERNAL	: จะใช้ค่าอ้างอิงภายนอกบอร์ดที่รับมาจากขา AREF (0 - 5V)



 หมายเหตุ ถ้าเลือกใช้ Type = EXTERNAL จากขา AREF ต้องตั้งค่า ก่อนเรียกใช้ analogRead() มิฉะนั้นจะทำให้เกิดการเชื่อม Active Reference Voltage (Internal Reference) กับ External Reference (AREF) ทำให้ Microcontroller ของบอร์ดพังได้ เพื่อป้องกันการตั้งค่าผิดพลาด สามารถต่อเชื่อมตัวต้านทาน 5K ที่ขา AREF เพื่อลดผลกระทบได้
7. **delay (time)** ใช้ในการหยุดโปรแกรมตามเวลาที่กำหนด โดยวัดเป็น Milliseconds (unsigned long)
8. **Millis ()** จะให้ค่าเวลาหน่วย Miliseconds (unsigned long) ตั้งแต่บอร์ดเริ่มรัน โปรแกรมปัจจุบัน
 Description ซึ่งจะวนกลับไป 0 หลังจากเวลาผ่านไปประมาณ 50 วัน
9. **Sin (rad)** ให้ค่า sine ของมุมที่กำหนด (rad) หน่วยเป็น Radian ซึ่งมีค่าอยู่ระหว่าง -1 ถึง 1

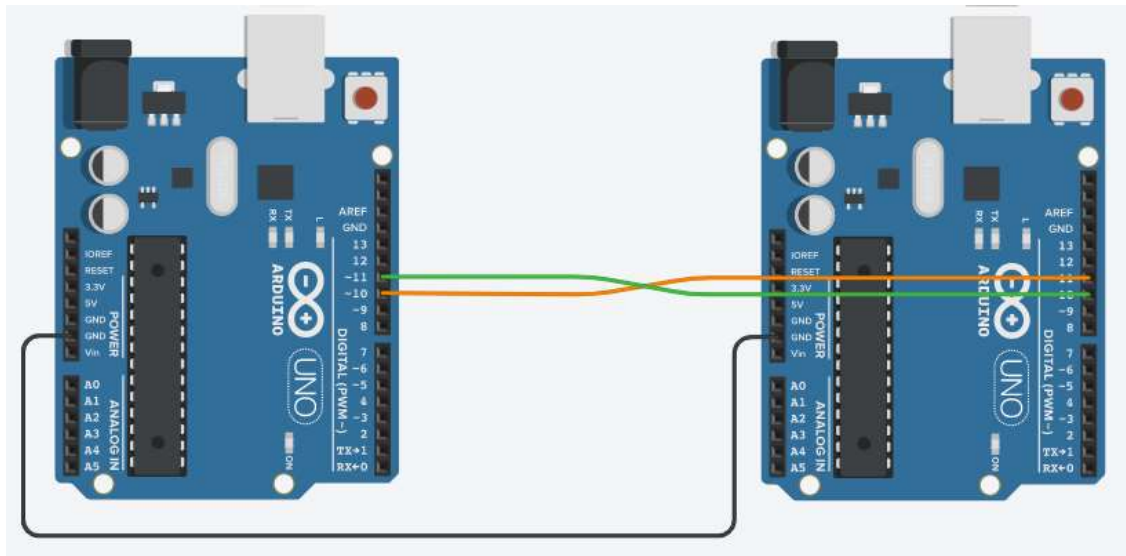
การสื่อสารอนุกรมระหว่าง Arduino

บอร์ด Arduino UNO จะมีส่วนติดต่อสื่อสารระหว่างพอร์ตอนุกรมด้วย UART ซึ่งมีขนาด 64 ไบต์ โดยในการสื่อสารใช้ขา 0 เป็นส่วนที่รับข้อมูล (Rx) และใช้ขา 1 เป็นส่วนที่ส่งข้อมูล (Tx) โดยทั่วไปมักใช้งานสำหรับการเขียนโปรแกรม หรือติดต่อสื่อสารกับคอมพิวเตอร์ ซึ่งทำให้ไม่สามารถนำมาใช้ในการสื่อสารแบบอนุกรม กับอุปกรณ์อื่นได้

จึงได้มีการพัฒนา SoftwareSerial Library ซึ่งช่วยให้ Arduino สามารถใช้ขาที่รองรับสัญญาณ Digital สื่อสารแบบอนุกรมแทนได้ โดยความเร็วสูงสุดของการใช้งาน SoftwareSerial คือ 57,600 bps

การทดลองที่ 3.1 การสื่อสารอนุกรมระหว่าง Arduino ด้วย SoftwareSerial

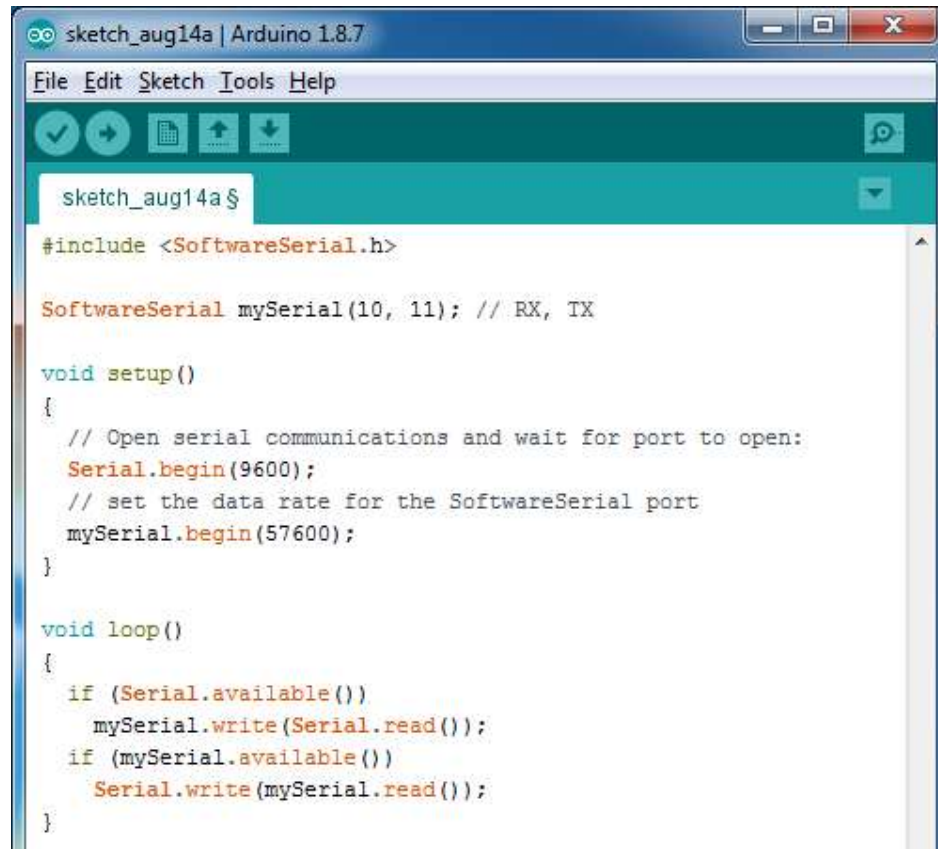
1. ทำการเชื่อมต่อ Arduino UNO R3 ดังรูปที่ 3.3 โดยเชื่อมต่อขาดังตารางที่ 3.1
2. เปิดโปรแกรม Arduino แล้วพิมพ์โปรแกรมตามรูปที่ 3.4 แล้วเลือก Verify 
3. คอสาข Upload
4. เลือก Com Port ที่เป็น Arduino/Genuino Uno ดังรูปที่ 3.5 แล้วเลือก Upload 
5. เปิด Serial Monitor แล้วตั้งค่าเป็น Newline / 9600 baud ดังรูปที่ 3.6
6. ทดลองพิมพ์แล้วส่ง (กด Send)



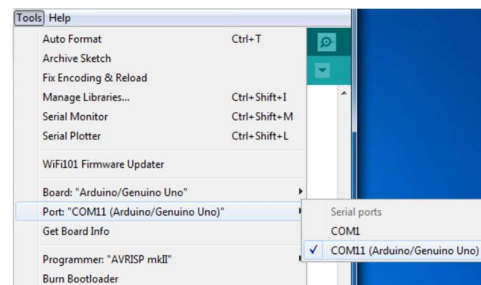
รูปที่ 3.3 การเชื่อมต่อระหว่าง Arduino UNO 2 บอร์ด

ตารางที่ 3.1 การเชื่อมต่อขาระหว่าง Arduino UNO 2 บอร์ด

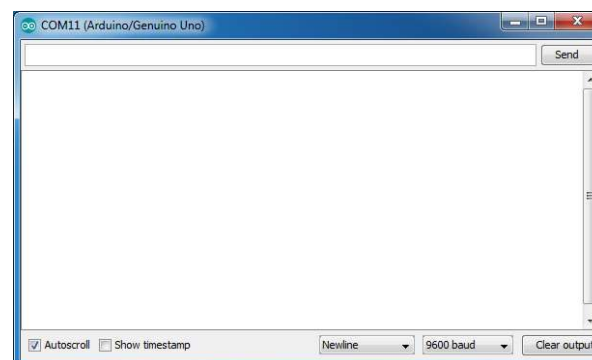
ขา Arduino 1 (ซ้าย)	ขา Arduino 2 (ขวา)
11 (TX)	10 (RX)
10 (RX)	11 (TX)
GND	GND



รูปที่ 3.4 ตัวอย่างโปรแกรมการใช้งาน SoftwareSerial



รูปที่ 3.5 การเลือก Com Port ของโปรแกรม Arduino



รูปที่ 3.6 การตั้งค่า Serial Monitor

การทดลองที่ 3.2 การเขียนโปรแกรม Chat messenger ระหว่าง Arduino

1. จากการทดลองที่ 3.1 ให้ปรับโปรแกรมเพื่อแสดงผลให้เป็นลักษณะ Chat messenger บน Serial Monitor โดยกำหนดให้ Arduino 1 (ซ้าย) เป็น A และ Arduino 2 (ขวา) เป็น B
2. ตัวอย่างการทำงาน

2.1. เมื่อเริ่มต้น Run Arduino

Serial Monitor ของ Arduino A	บน Serial Monitor ของ Arduino B

2.2. เมื่อ A พิมพ์ข้อความ Hello A แล้วส่ง (กด Send)

บน Serial Monitor ของ Arduino A	บน Serial Monitor ของ Arduino B
Me : Hello B	A : Hello B

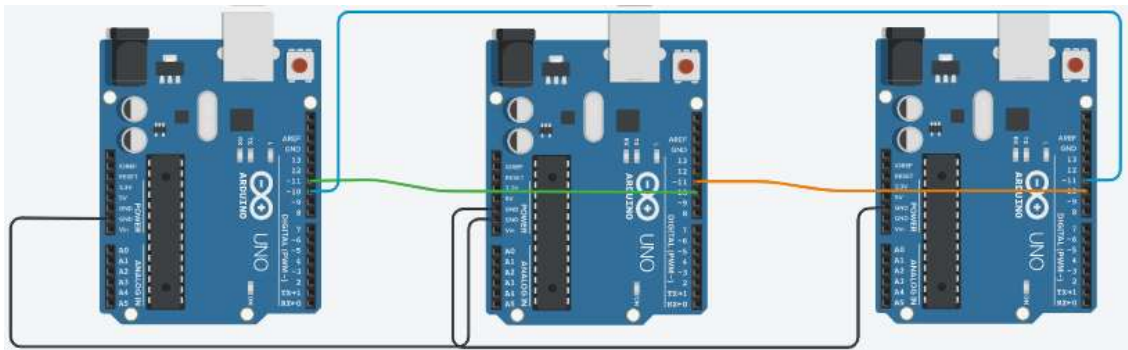
2.3. เมื่อ B พิมพ์ข้อความ Hi แล้วส่ง (กด Send)

บน Serial Monitor ของ Arduino A	บน Serial Monitor ของ Arduino B
Me : Hello B B : Hi	A : Hello B Me : Hi

3. เชิญอาจารย์ตรวจการทดลอง

การทดลองที่ 3.3 การสื่อสารรูปแบบ Ring Communication ระหว่าง Arduino เบื้องต้น

1. ทำการเชื่อมต่อ Arduino UNO R3 ดังรูปที่ 3.7 โดยเชื่อมต่อขา ดังตารางที่ 3.2



รูปที่ 3.7 การเชื่อมต่อ Ring Communication ระหว่าง Arduino UNO 3 บอร์ด

ตารางที่ 3.2 การเชื่อมต่อขา ระหว่าง Arduino UNO 3 บอร์ด

ขา Arduino 1 (ซ้าย)	ขา Arduino 2 (กลาง)	ขา Arduino 3 (ขวา)
11 (TX)	10 (RX)	
	11 (TX)	10 (RX)
10 (RX)		11 (TX)
GND	GND	

2. กำหนดให้เฟรมข้อมูลที่ส่งระหว่าง Arduino เป็นดังนี้

ขนาด	1 byte	1 byte	n byte	1 byte
ชนิดข้อมูล	ผู้รับ	ผู้ส่ง	ข้อความ	\0

3. กำหนดให้การทำงานของ

3.1. การสื่อสารลักษณะ Ring Communication เบื้องต้น (ไม่จำเป็นต้องทำเป็น Token Ring)

3.2. เมื่อเริ่มต้นโปรแกรม Arduino ทุกตัวจะต้องกำหนด ID ผู้รับเป็นตัวอักษรภาษาอังกฤษ

3.3. การส่งข้อความ ต้องป้อนผู้รับ (A, B, C, ..., Z) / ตามด้วย : / ตามด้วยข้อความ ตัวอย่างเช่น

1) เมื่อ A ต้องการส่งข้อความ Hello ให้ C ป้อน

C:Hello

2) เมื่อ B ต้องการส่งข้อความ Hi ให้ C

C:Hi

3.4. เมื่อมีข้อมูลที่ได้รับจาก SoftwareSerial ให้พิจารณาดังนี้

1) หากผู้รับในเฟรมข้อมูลตรงกับ ID ที่กำหนดไว้ ให้แสดงผลว่าใครส่งข้อความมา

2) หากผู้รับในเฟรมข้อมูลไม่ตรงกับ ID ที่กำหนดไว้ ให้ส่งข้อมูลทั้งหมดออกทาง SoftwareSerial

4. ตัวอย่างผลการทำงานบน Serial Monitor ของ Arduino

4.1. เมื่อเริ่มต้น Run Arduino

Arduino A	Arduino B	Arduino C
Enter ID :	Enter ID :	Enter ID :

4.2. เมื่อป้อน A, B, C ตามลำดับ

Arduino A	Arduino B	Arduino C
Enter ID : Your ID : A	Enter ID : Your ID : B	Enter ID : Your ID : C

4.3. เมื่อ A ต้องการส่งข้อความ Hello ให้ C | A ต้องป้อน C:Hello แล้วส่ง (กด Send)

Arduino A	Arduino B	Arduino C
Enter ID : Your ID : A Me:Hello	Enter ID : Your ID : B	Enter ID : Your ID : C A :Hello

4.4. เมื่อ C ต้องการส่งข้อความ Hi ให้ A | C ต้องป้อน A:Hi แล้วส่ง (กด Send)

Arduino A	Arduino B	Arduino C
Enter ID : Your ID : A Me:Hello C :Hi	Enter ID : Your ID : B	Enter ID : Your ID : C A :Hello Me:Hi

5. เชิญอาจารย์ตรวจการทดลอง

ลายเซ็นอาจารย์ผู้ตรวจการทดลอง