วิชา Data Communication Laboratory

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 4 Protocol Design (Bus Communication)

วัตถุประสงค์

1. เพื่อให้เข้าใจหลักการสื่อสารผ่านระหว่าง Arduino ด้วยโมดูล CAN Bus
2. สามารถเขียนโปรแกรมเพื่อติดต่อสื่อสารระหว่าง Arduino ด้วยการเชื่อมต่อแบบ Bus Communication
3. สามารถเขียนโปรแกรมประยุกต์เพื่อสร้างโปรโตคอลติดต่อสื่อสารระหว่างคอมพิวเตอร์ และ Arduino ได้

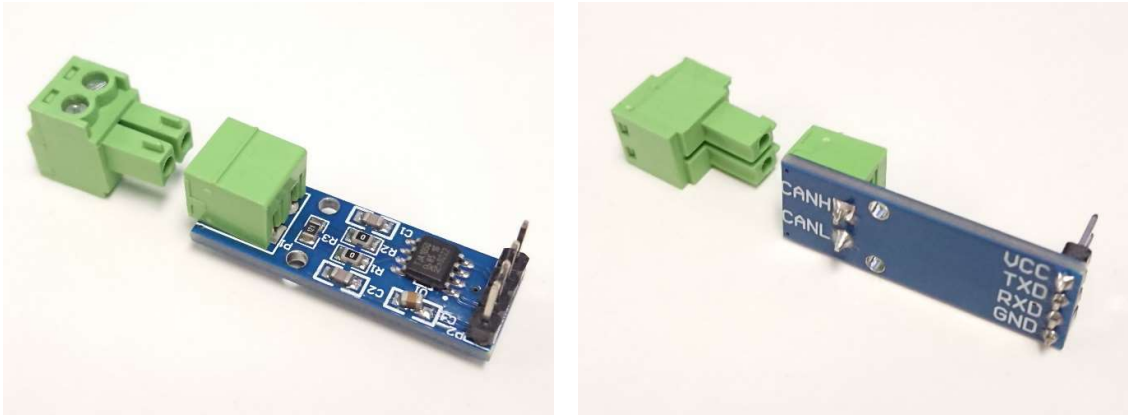
CAN Bus

CAN ย่อมาจาก Controller Area Network พัฒนาโดย Robert Bosch ในปี ค.ศ. 1983 เป็นสถาปัตยกรรมการสื่อสารผ่าน Serial Data Bus นิยมเรียกเป็น CAN Bus ใช้สำหรับสื่อสารระหว่างคอมพิวเตอร์ตั้งแต่ 2 เครื่องขึ้นไป ถูกนำมาใช้ในอุตสาหกรรมต่างๆ ต่อมา International Organization for Standardization กำหนดเป็นมาตรฐานชุด IEEE 11898 ภายหลังเริ่มนำมาใช้งานในอุตสาหกรรมรถยนต์ จนเป็นระบบสื่อสารหลักของส่วนต่างๆ ภายในรถยนต์

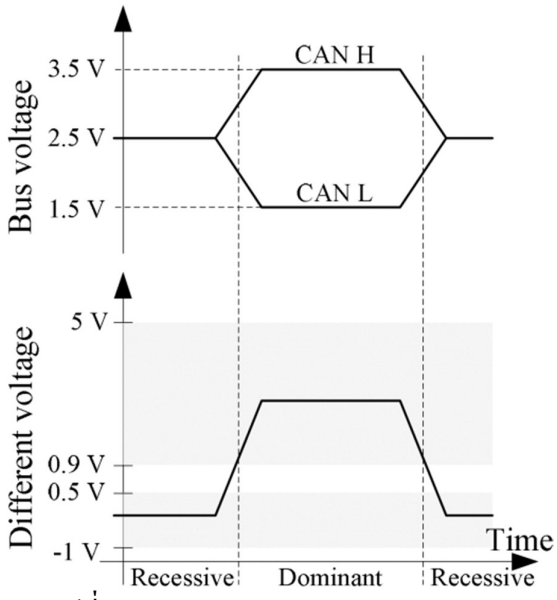
CAN Module

Module CAN Bus interface เป็น โมดูลสำหรับแปลงสัญญาณ UART เป็น สัญญาณ CAN bus โดยมี IC TJA1050 เป็นชิปที่ทำหน้าที่ในการแปลงสัญญาณ

CAN จะใช้คู่สัญญาณสองเส้นคือ CANH และ CANL เมื่อ CAN ส่ง Logic High ขา CANH และ CANL จะมีแรงดันไฟฟ้าอยู่ที่ 2.5V (ความต่างของระดับแรงดันฟ้าเป็น 0V) และเมื่อ CAN ส่ง Logic Low ขา CANH จะมีแรงดันไฟฟ้าอยู่ที่ 3.5V และ CANL จะมีแรงดันไฟฟ้าอยู่ที่ 1.5V เมื่อหาความต่างของระดับแรงดันไฟฟ้าทั้ง 2 ขา จะพบว่ามี Voltage อยู่ที่ 2.0V



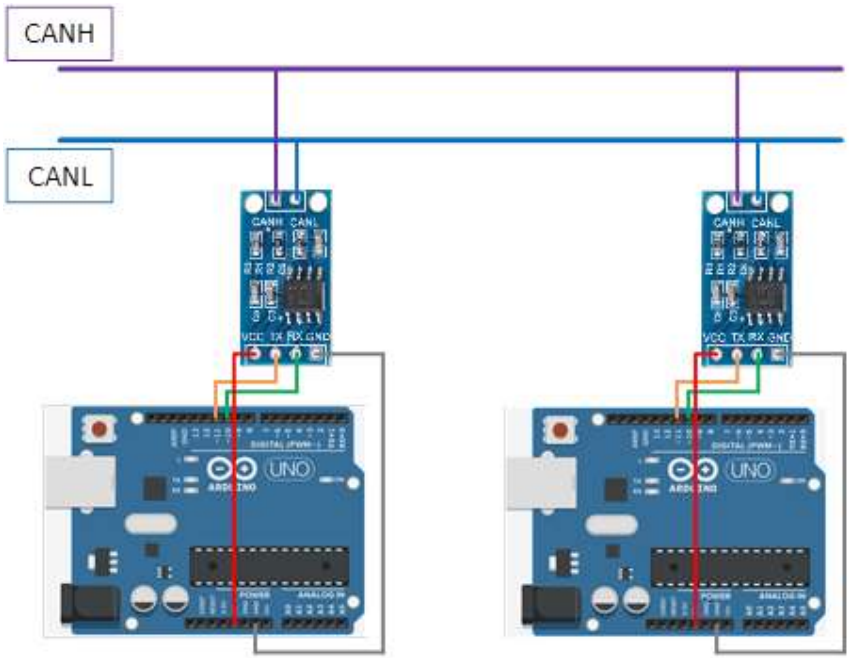
รูปที่ 4.1 CAN Module



รูปที่ 4.2 ระดับแรงดันสัญญาณของ CAN

การทดลองที่ 4.1 การสื่อสารอนุกรมระหว่าง Arduino ด้วยโมดูล CAN Bus

1. ทำการเชื่อมต่อ Arduino UNO R3 กับ CAN Bus ดังรูปที่ 4.3
2. โดยเชื่อมต่อขาระหว่าง Arduino UNO กับ โมดูล CAN Bus ดังตารางที่ 4.1 และ ขาระหว่างโมดูล CAN Bus ด้วย CANH กับ CANH และ CANL กับ CANL
3. เปิดโปรแกรม Arduino แล้วพิมพ์โปรแกรมตามรูปที่ 4.4 แล้วเลือก Upload โปรแกรมลง Arduino UNO
4. เปิด Serial Monitor แล้วตั้งค่า baud rate เป็น 115200 baud
5. สังเกตผลที่ได้ และทำความความเข้าใจโปรแกรมตัวอย่าง



รูปที่ 4.3 การเชื่อมต่อระหว่าง Arduino UNO 2 บอร์ด ด้วยโมดูล CAN Bus

ตารางที่ 4.2 การเชื่อมต่อขาระหว่าง Arduino UNO กับ โมดูล CAN Bus

ขา Arduino	ขาโมดูล CAN Bus
5V	VCC
11 (TX)	TXD
10 (RX)	RXD
GND	GND

Lab_4_1_CANbus_Tx

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11);

void setup()
{
  //Serial.begin(115200);
  mySerial.begin(57600);
}

void flushRx()
{
  while(mySerial.available())
    uint8_t tmp = mySerial.read();
}

void loop()
{
  char myString [] = "Computer Engineering";
  for (int i=0; myString[i] != '\0'; i++)
  {
    mySerial.write(myString[i]);
    delay(10);
    flushRx();
  }
  delay(500);
  mySerial.write('\n');
}
```

Lab_4_1_CANbus_Rx

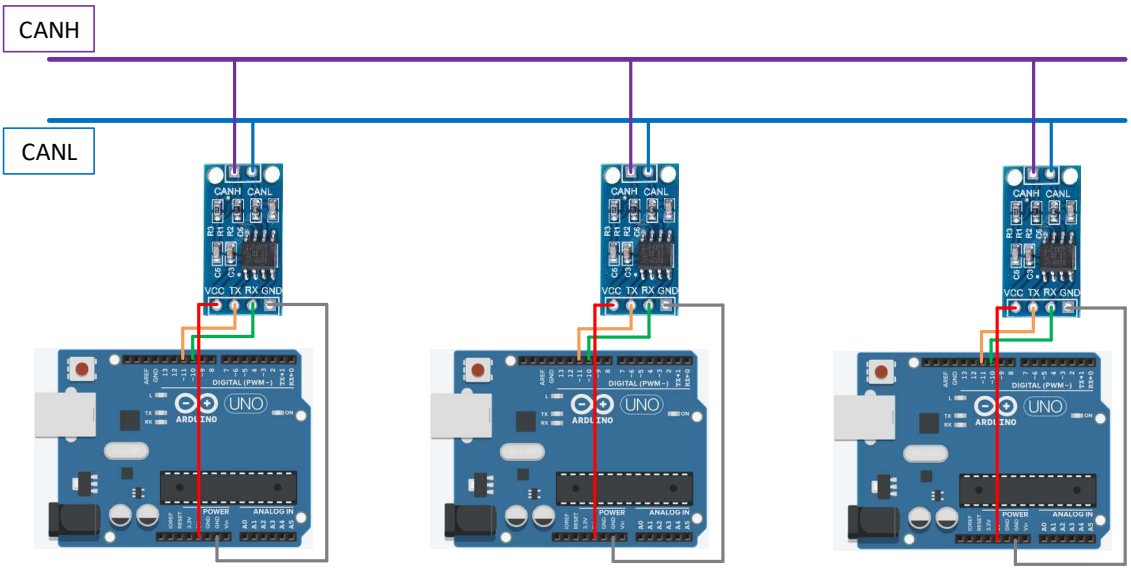
```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11);

void setup()
{
  Serial.begin(115200);
  mySerial.begin(57600);
}

void loop()
{
  if (mySerial.available())
    Serial.write(mySerial.read());
}
```

รูปที่ 4.4 ตัวอย่างโปรแกรมการใช้งานโมดูล CAN Bus ผ่าน SoftwareSerial

เนื่องจาก การเขียนข้อมูลลงใน Bus มีโอกาสที่ Serial ของ Arduino จะได้รับค่าที่ตัวเองส่งได้ ดังนั้นหากต้องการเคลียร์ค่าใน Buffer ให้ทำการอ่านค่าจาก Hardware Buffer (ฟังก์ชัน `flushRx()`)

การทดลองที่ 4.2 การสื่อสารอนุกรมระหว่าง Arduino มากว่า 2 บอร์ด ด้วยโมดูล CAN Bus

1. ทำการเชื่อมต่อ Arduino UNO R3 กับ CAN Bus ดังรูปที่ 4.5
2. โดยเชื่อมต่อขาระหว่าง Arduino UNO กับ โมดูล CAN Bus ดังตารางที่ 4.1 และ ขาระหว่างโมดูล CAN Bus ด้วย CANH กับ CANH และ CANL กับ CANL
3. พิมพ์โปรแกรมสำหรับส่งข้อมูล (CANbus_Tx) แล้ว Upload โปรแกรมลง Arduino UNO R3 บอร์ดซ้าย / พิมพ์โปรแกรมสำหรับรับข้อมูล (CANbus_Rx) แล้ว Upload โปรแกรมลง Arduino Arduino UNO R3 บอร์ดกลาง และบอร์ดขวา
4. เปิด Serial Monitor แล้วตั้งค่า baud rate เป็น 115200 baud
5. สังเกตผลที่ได้จาก Serial Monitor บอร์ดกลาง และบอร์ดขวา และทำความความเข้าใจ
6. พิมพ์โปรแกรมสำหรับส่งข้อมูล (CANbus_Tx) แล้ว Upload โปรแกรมลง Arduino UNO R3 บอร์ดกลาง
7. สังเกตผลที่ได้จาก Serial Monitor บอร์ดขวา และทำความความเข้าใจ



รูปที่ 4.5 การเชื่อมต่อระหว่าง Arduino UNO 3 บอร์ด ด้วยโมดูล CAN Bus

การทดลองที่ 4.3

1. จากการทดลองที่ 4.2 ออกแบบโปรแกมในการรับ-ส่งไฟล์ข้อมูล โดยมีข้อกำหนดต่อไปนี้
   1.1. การสื่อสารลักษณะ Bus Comunication (ด้วยโมดูล CAN Bus)
   1.2. เฟรมข้อมูลที่ใช้สื่อสารมีลักษณะดังนี้
      1) ประกอบด้วย Flag (เปิด-ปิด) Header (ผู้รับ-ผู้ส่ง) และ Trailer (Error Checking) **<u>เป็นอย่างน้อย</u>**
      2) สามารถเลือกได้ระหว่าง Fixed-Size Framing กับ Variable-Size Framing
         หากกำหนดเป็น Variable-Size Framing ต้องมี Character-oriented protocol
   1.3. มี Flow & Error Control ในการความคุมการส่งข้อมูล
   1.4. เมื่อเริ่มต้นโปรแกรม Arduino ทุกตัวจะต้องกำหนด ID ผู้รับเป็นตัวอักษรภาษาอังกฤษ
   1.5. เลือกผู้รับ และไฟล์ที่จะส่งข้อมูลได้ & เลือกรับชื่อไฟล์ได้

1.6. บอร์ดที่ไม่ได้เป็นผู้รับ หรือ ผู้ส่ง ให้แสดงข้อมูลจริงที่ได้รับแต่ละเฟรม

1.7. ตัวอย่างโปรแกรมการรับ-ส่งไฟล์ข้อมูล

Sender (A)	**Receiver (C)**

Enter ID : //Type A
My ID is : A
//Type C
Reciver is : C
//C:\dir\data.txt
Send file : **C:\dir\data.txt**
	Enter ID : //Type C
	My ID is : C
	
	Sender Send : data.txt
	//C:\commu.txt
	Save as : **C:\commu.txt**

Send frame : 0 Data : xxxxxxxxxxxxx	Receive frame Header : CA Frame No. : 0 Data : xxxxxxxxxxxxx Checking : ???? Received Send ACK1
Receive frame Header : AC ACK No. : 1 Checking : ???? Received Send frame : 1 Data : yyyyyyyyyyyyy	
/*Disconnected Cable*/	Receive frame Header : CA Frame No. : 1 Data : yyyyyyyyyyyyy Checking : ???? Received Send ACK0
Timeout Retransmit frame 1 Send frame : 1 Data : yyyyyyyyyyyyy	
	Receive frame Header : CA Frame No. : 1 Data : yyyyyyyyyyyyy Checking : ???? Reject Send ACK0
Receive frame Header : AC ACK No. : 0 Checking : ???? Received	

1.8. ให้นักศึกษาออกแบบ Frame ข้อมูลที่ใช้ในการส่งไฟล์ ให้เหมาะสม พร้อมแสดงรูปแบบของ Frame

1.9. ให้นักศึกษาเขียนผังงาน (Flow Chart) โปรแกมในการรับ-ส่งไฟล์ข้อมูล