3G Shield / 3G Module

บน Platform Arduino

Thaieasyelec

enable your design

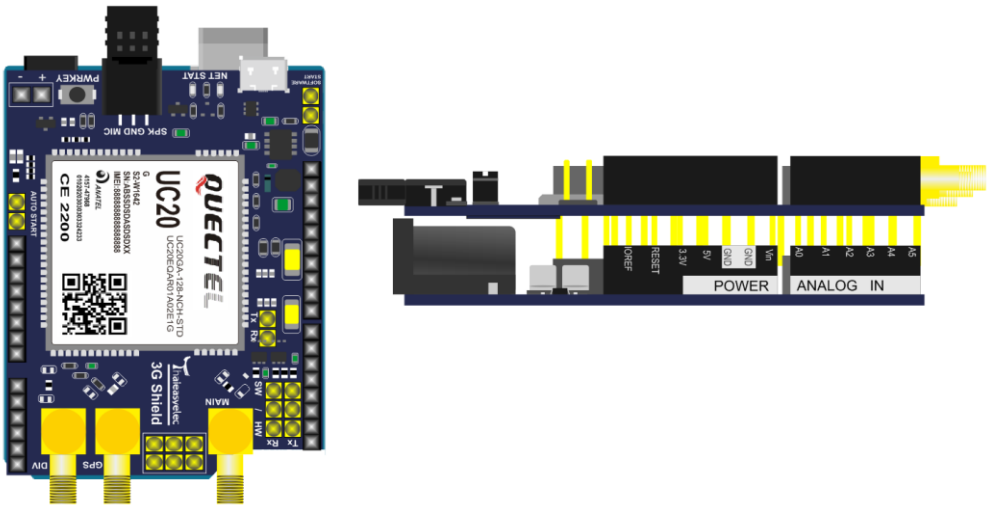
ประวัติการเปลี่ยนเวอร์ชั่น

เวอร์ชั่น	วันที่	การเปลี่ยนแปลง
1.0	21/03/2559	เวอร์ชั่นแรก
1.1	26/07/2559	เพิ่มชุดคำสั่ง MQTT และ Audio

สารบัญ

วิธีเชื่อมต่อ 3G Shield / 3G Module กับ Arduino UNO R3

3G Shield

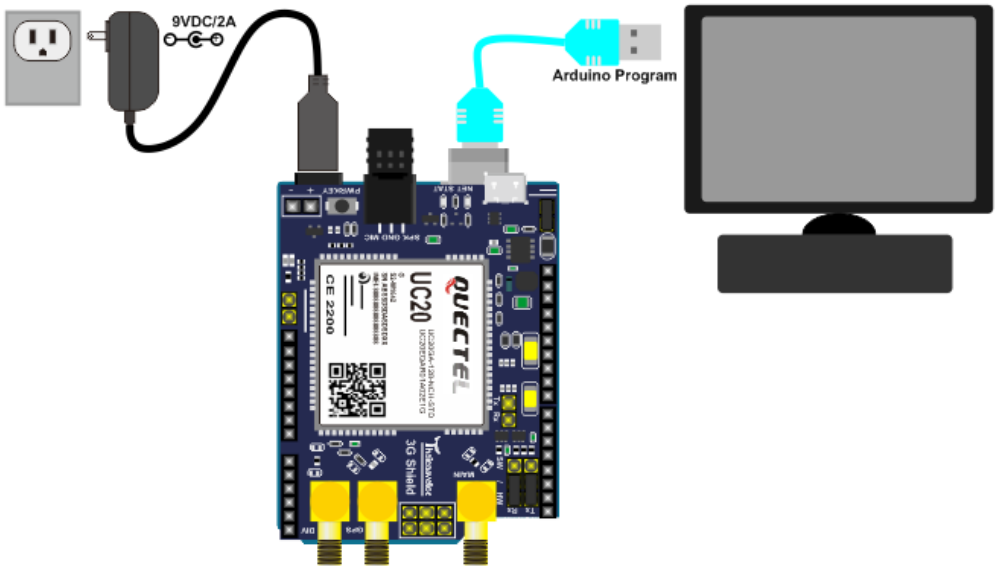


การจ่ายไฟให้กับ 3G Shield

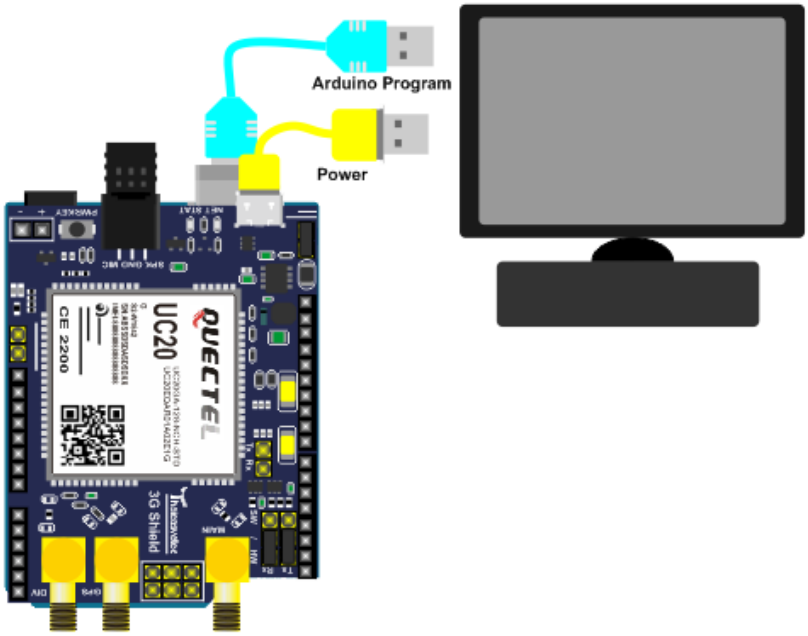
การจ่ายไฟเลี้ยงให้กับ 3G Shield สามารถจ่ายไฟได้ 2 ช่องทางได้แก่

- จ่ายไฟเลี้ยงผ่านทาง DC Jack ของ บอร์ด Arduino ซึ่งทางเราแนะนำให้ใช้กับภาคจ่ายไฟ 9VDC 2A (บน 3G Shield จะมีวงจร Switching ที่รองรับการจ่ายไฟได้ตั้งแต่ 5 – 12 VDC แต่เนื่องจากการจ่ายไฟทาง DC Jack ของ Arduino บอร์ด Arduino จะนำไฟฟ้าจากส่วนนี้ไปใช้บนบอร์ดด้วย 9VDC 2A จึงเป็นค่าที่เหมาะสมเพราะจะทำให้ Regulator ที่อยู่บน Arduino ไม่ร้อนเกินไป)

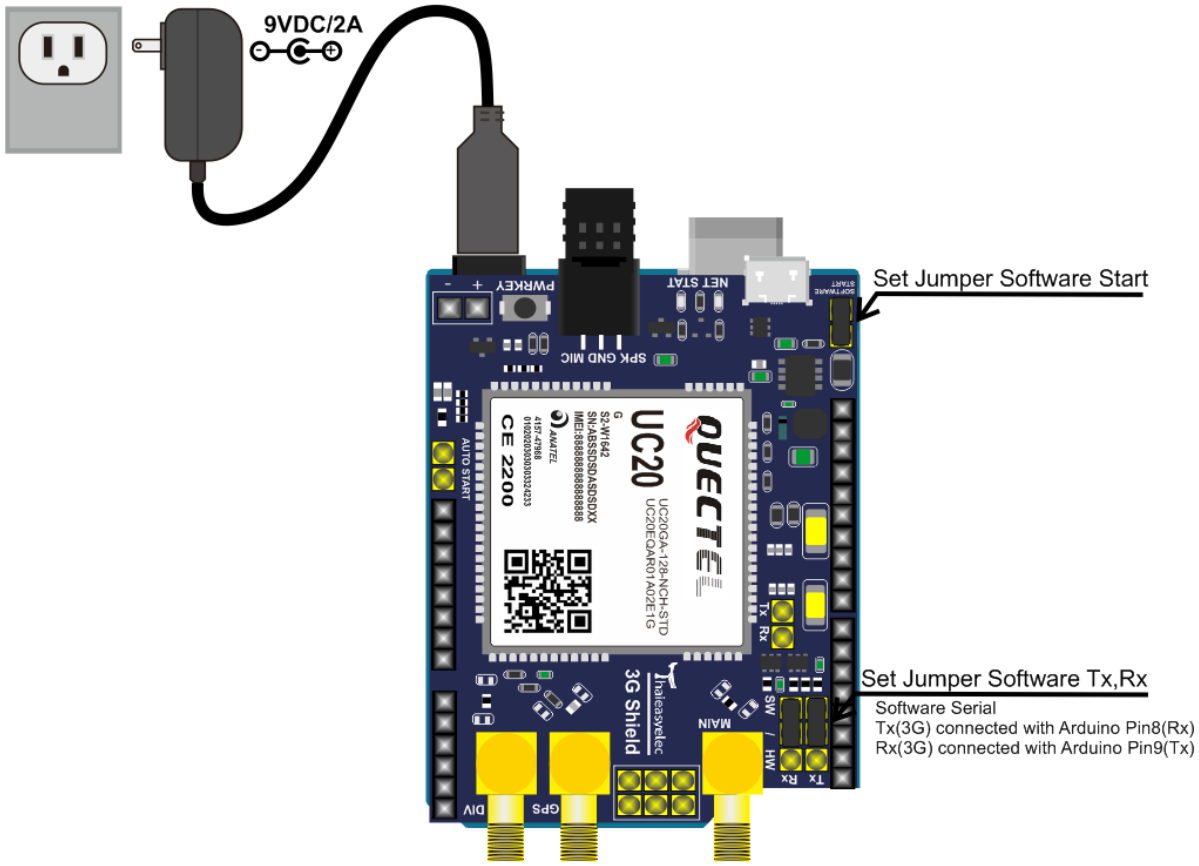
หมายเหตุ : บน 3G Shield ในสภาวะปรกติจะกินกระแสไม่มากนัก แต่ถ้าหากนำโมดูลไปใช้ในสภาพแวดล้อมที่มีสัญญาณน้อย ช่วงเวลาที่โมดูลค้นหาสัญญาณ จะใช้กระแสสูงกว่าปรกติ จึงควรจัดหา แหล่งจ่ายไฟที่สามารถจ่ายกระแสได้ประมาณ 2A



- จ่ายไฟเลี้ยงให้กับ 3G Shield ผ่านทาง USB Port ของ 3G Shield



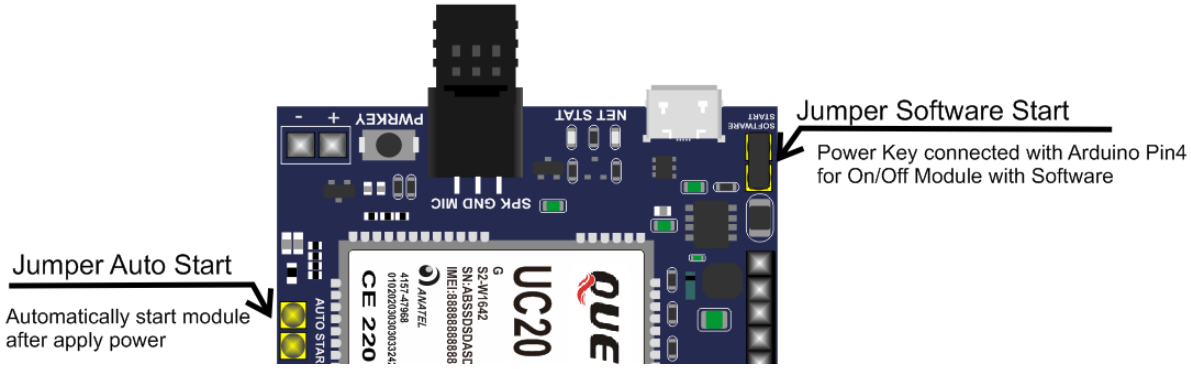
การเซ็ต Jumper



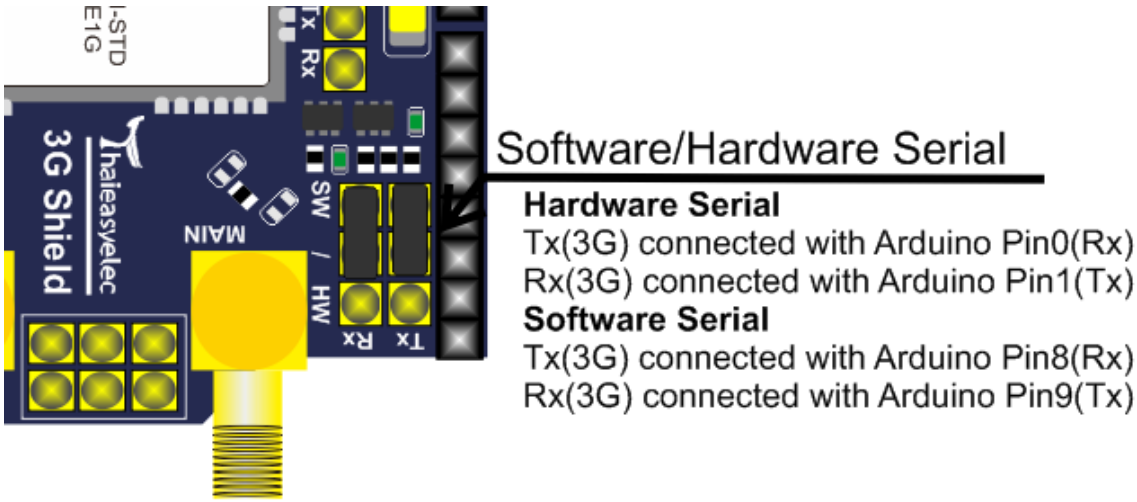
- **Jumper Start**

  Jumper ที่ใช้สำหรับ Start หรือ สั่งงานให้ 3G on ขึ้นมาจะมีอยู่ด้วยกัน 2 ตำแหน่ง ได้แก่

  - Jumper Auto Start ใช้สำหรับตั้งค่าให้ 3G shield Power On ทันทีเมื่อจ่ายไฟให้กับโมดูล
  - Jumper Software Start ใช้สำหรับสั่งงาน Power On โมดูลด้วย Software โดยขา Software Start จะต่ออยู่กับ Arduino ที่ Pin4

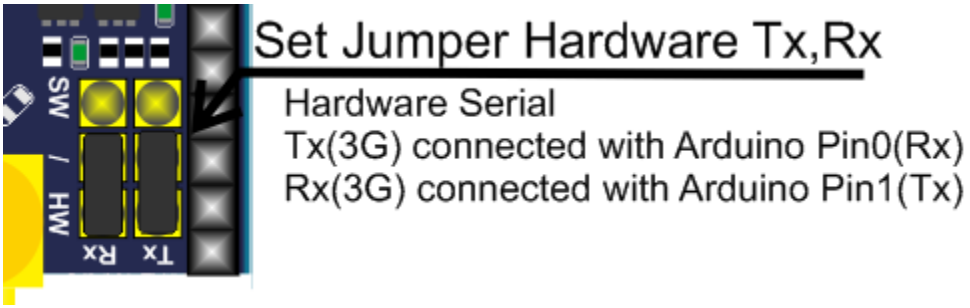


- **Jumper Serial**

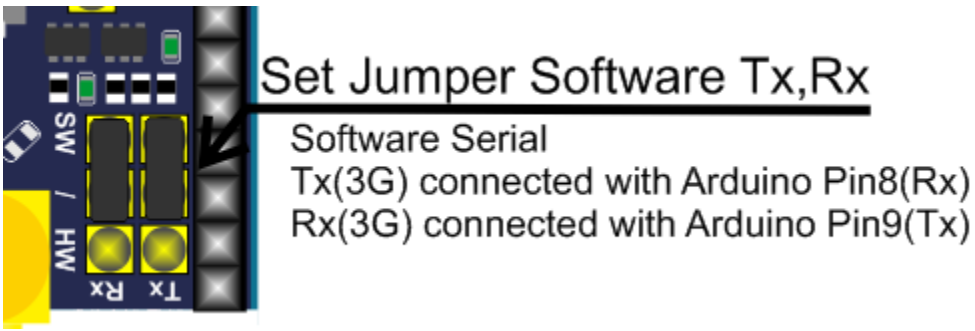


Jumper Serial ใช้สำหรับกำหนด ช่องทางของ Serial ที่ใช้เชื่อมต่อกับตัว 3G Module บน 3G Shield ซึ่งสามารถกำหนดการเชื่อมต่อได้ 2 ช่องทางได้แก่

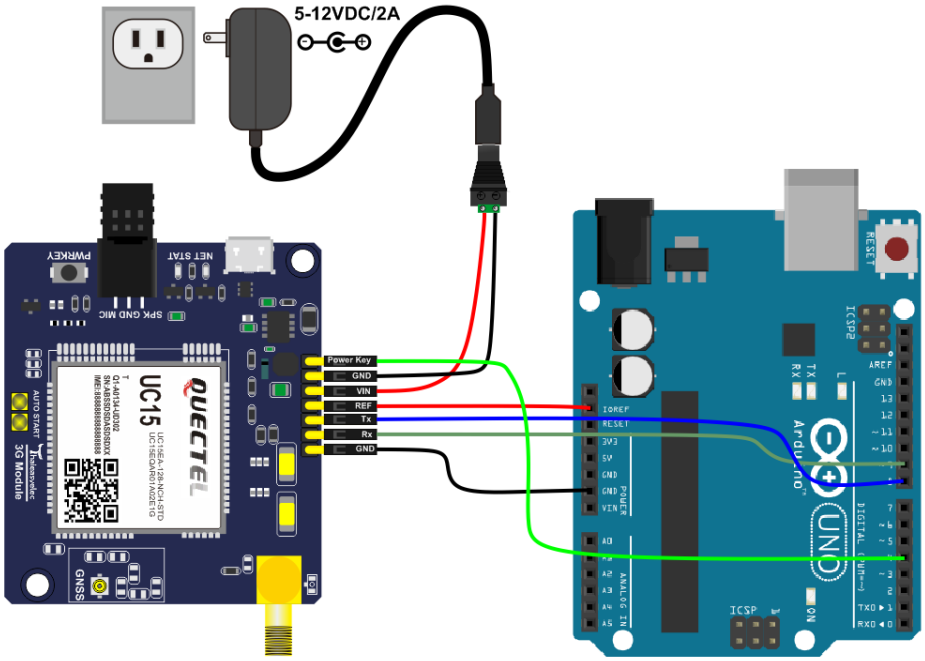
- เชื่อมต่อ 3G Shield กับขา Hardware Serial (Arduino Pin 0,1)



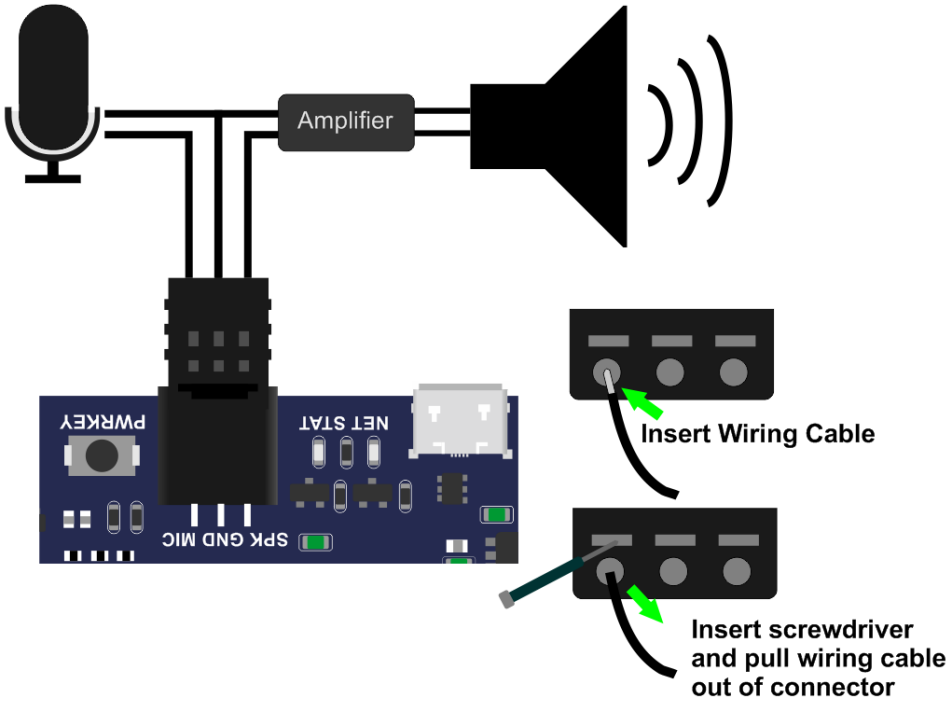
- เชื่อมต่อ 3G Shield กับขา Software Serial (Arduino Pin 8,9)



3G Module



วิธีเชื่อมต่อ 3G Shield / 3G Module กับ Mic และ Speaker



การติดตั้งไลบรารีสำหรับใช้งาน 3G Shield / 3G Module กับ Arduino

- ดาวน์โหลดไลบรารี AltSoftSerial จาก
  http://www.pjrc.com/teensy/td_libs_AltSoftSerial.html
  หรือ
  https://github.com/PaulStoffregen/AltSoftSerial
- แตกไฟล์ AltSoftSerial-master.zip และแก้ไขชื่อจาก AltSoftSerial-master เป็น AltSoftSerial
- คัดลอกไดเรกทอรี่ AltSoftSerial ไปไว้ในไดเรกทอรี่ libraries ภายในไดเรกทอรี่ของ Arduino IDE
- ดาวน์โหลดไลบรารี TEE_UC20_Shield จาก http://www.thaieasyelec.com/
- แตกไฟล์ TEE_UC20_Shield.7z
- คัดลอกไดเรกทอรี่ TEE_UC20_Shield ไปไว้ในไดเรกทอรี่ libraries ภายในไดเรกทอรี่ของ Arduino IDE

ใช้งาน AT command โดยใช้ Arduino เป็น USB-to-Serial

การใช้งาน 3G Shield / 3G Module สามารถสั่งงานและสื่อสารกับโมดูลด้วยการใช้ AT Command ผ่านทางช่อง Serial UART และ USB ซึ่งหากเป็นสั่งงานโมดูลจากเครื่องคอมพิวเตอร์จะสามารถใช้งานผ่านทาง USB ได้อย่างสะดวก โดยมีพอร์ต USB Modem สำหรับการสั่งงานและสื่อสารกับเครือข่าย และ พอร์ต USB AT สำหรับการสั่งงานเพียงอย่างเดียว แต่ถ้าหากเป็นการสั่งงานจากไมโครคอนโทรลเลอร์แล้วต้องใช้งานผ่านทางช่อง Serial UART

นอกจากนี้จากคำสั่ง AT Command บางคำสั่งที่สั่งผ่าน USB จะไม่ส่งผลถึงการทำงานบนช่อง Serial UART ยกตัวอย่างเช่นคำสั่งเปลี่ยนค่า Baud Rate ของการสื่อสาร การใช้คำสั่งนี้จะส่งผลกับค่า Baud Rate เฉพาะช่องที่ได้รับคำสั่งนี้เท่านั้น คือ สั่งบน USB ก็จะเป็นการเปลี่ยน Baud Rate ของ USB แต่หากสั่งบน Serial UART ก็จะเปลี่ยน Baud Rate เฉพาะพอร์ต Serial UART เพียงอย่างเดียว

หากเราไม่มีโมดูล USB-to-Serial เราสามารถใช้ Arduino ทำหน้าที่แทน USB-to-Serial ในการส่งคำสั่ง AT Command ให้กับ 3G Shield / 3G Module ผ่านทาง Serial UART ได้เช่นกัน

วิธีใช้งาน

- เชื่อมต่อ 3G Shield / 3G Module เข้ากับบอร์ด Arduino
- เขียนโปรแกรมแล้วอัพโหลดลงใน Arduino ดังนี้

```
// !!อย่าลืมสั่ง Power On Module
#include <AltSoftSerial.h>
AltSoftSerial mySerial;
void setup() {
     Serial.begin(9600);
     mySerial.begin(9600);
     //mySerial.begin(115200);
}
void loop() {
     if (mySerial.available()) {
          Serial.write(mySerial.read());
     }
     if (Serial.available()) {
          mySerial.write(Serial.read());
     }
}
```

- การทำงานของ Code

Serial คือ ออบเจคต์ของช่อง Hardware Serial บนบอร์ด Arduino ซึ่งเชื่อมต่ออยู่กับชิพ ATmega16U2 บนบอร์ด ทำหน้าที่เป็น USB-to-Serial สำหรับอัพโหลดโปรแกรม Arduino และใช้รับ-ส่งข้อมูลกับเครื่องคอมพิวเตอร์

mySerial คือ ออบเจคต์ของ Software Serial ที่เราสร้างขึ้น ทำให้ Arduino มีพอร์ต Serial เพิ่มขึ้นมาโดยกระบวนการสื่อสารจัดการด้วยโค้ดของ Software สำหรับเชื่อมต่อกับอุปกรณ์อื่นๆ ที่สื่อสารผ่าน Serial UART ได้ ในที่นี้เรานำ Software Serial มาเชื่อมต่อกับ 3G Shield / 3G Module

ข้อจำกัด คือ ไลบรารี Software Serial บางตัวอาจจะรองรับ Baud Rate ได้ไม่สูงนัก มีการทำงานที่ช้าและใช้ทรัพยากรบนไมโครคอนโทรลเลอร์เพิ่มขึ้นในการประมวลผลเมื่อเทียบกับการใช้งาน Hardware Serial ซึ่งมีวงจร Hardware ในการทำงานจริง แต่ Software Serial ช่วยให้เราสามารถเพิ่มช่องทางสื่อสารให้กับ Arduino ที่มีช่อง Serial น้อยไปไม่เพียงพออย่าง Arduino Uno ซึ่งมี Hardware Serial เพียงช่องเดียว

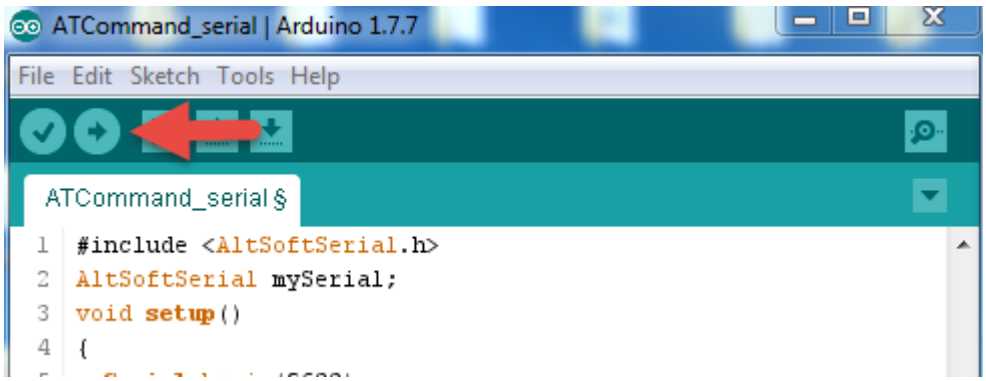
จากตัวอย่างโค้ดภายในฟังก์ชัน loop() หากมีข้อมูลส่งเข้ามาทาง Software Serial ด้วย mySerial.available() จะรับค่าด้วย mySerial.read() แล้วส่งออกไปที่ Hardware Serial ด้วย Serial.write()

และหากมีข้อมูลส่งเข้ามาทาง Hardware Serial ด้วย Serial.available() จะรับค่าด้วย Serial.read() แล้วส่งออกไปที่ Software Serial ด้วย mySerial.write() เช่นกัน

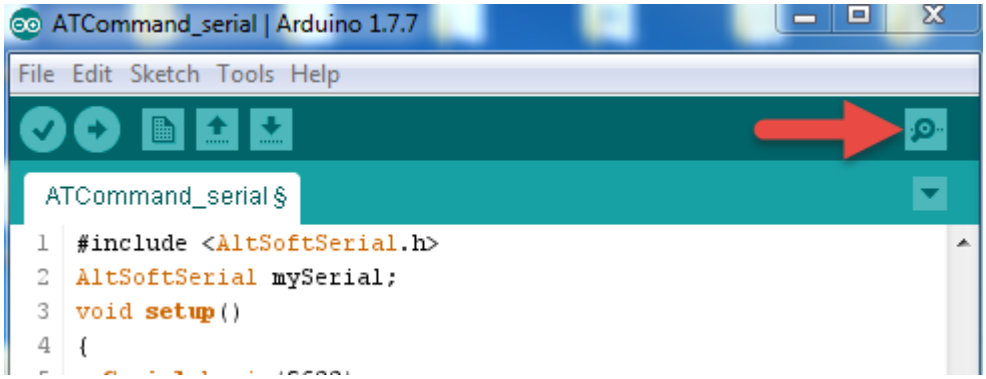
เสมือนกับเป็นการนำเอา Usb-to-Serial ของ Arduino ต่อเข้ากับ Serial UART ของ 3G Shield / 3G Module

หมายเหตุ!!! ที่ mySerial.begin(115200) จะต้องตั้งค่า Baud rate ให้ตรงกับที่ 3G Shield / 3G Module โดยค่า Default Baud Rate จะเป็น 115200 หรือ 9600 ขึ้นกับเฟิร์มแวร์ที่ติดมากับโมดูล

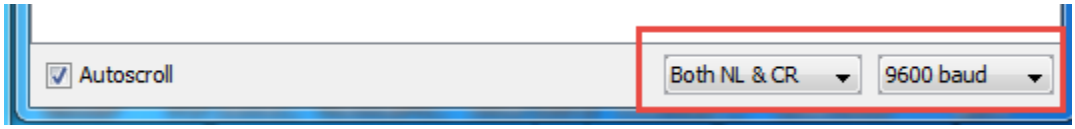
- อัพโหลดโปรแกรมลงบนบอร์ด Arduino



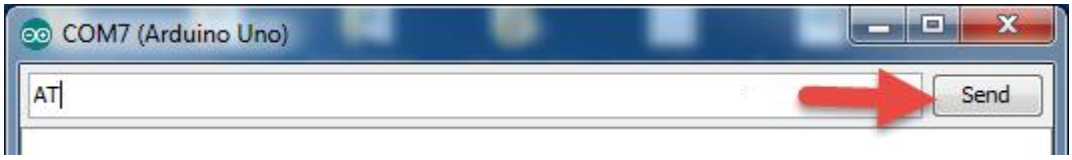
- เปิดSerial Monitor



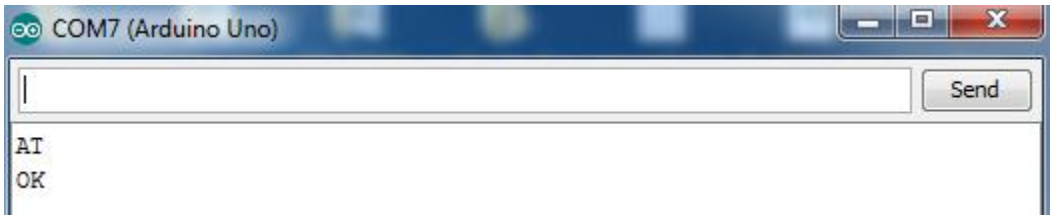
- ตั้งค่า Baud rate และ NL/CR



- ทดลองพิมพ์คำสั่ง AT



- โมดูลจะตอบ OK กลับมาแสดงว่าสื่อสารกันได้



การตั้งค่า AT Command ให้กับ 3G Shield / 3G Module ก่อนใช้งานร่วมกับไลบรารี TEE_UC20_Shield

หมายเหตุ!!! 3G Shield ได้มีการตั้งค่าเหล่านี้มากจากการผลิตแล้ว ไม่จำเป็นต้องตั้งค่าใหม่

- ปิด Echo

```
ATE0
```

- ตั้งค่าพอร์ต URC ไปที่ UART1 พอร์ต URC ใช้แสดงผลและสถานะการทำงานต่างๆ ที่โมดูลตอบกลับมาในขณะทำงานหรือมีข้อผิดพลาด เช่น ข้อความ RING เมื่อมีสายโทรเข้า ข้อความตอบกลับเมื่อใช้งาน USSD เป็นต้น

```
AT+QURCCFG="urcport","uart1"
```

- เปลี่ยน Baud Rate เป็น 9600

```
AT+IPR=9600
AT+IFC=0,0
```

- บันทึกค่าคอนฟิกไว้ในหน่วยความจำของโมดูล เมื่อเปิดขึ้นมาโมดูลจะใช้ค่าที่กำหนดไว้

```
AT&W
```

การตั้งค่า UART สำหรับใช้งานไลบรารี TEE_UC20_Shield

การใช้งานไลบรารี TEE_UC20_Shield ร่วมกับ AltSoftSerial

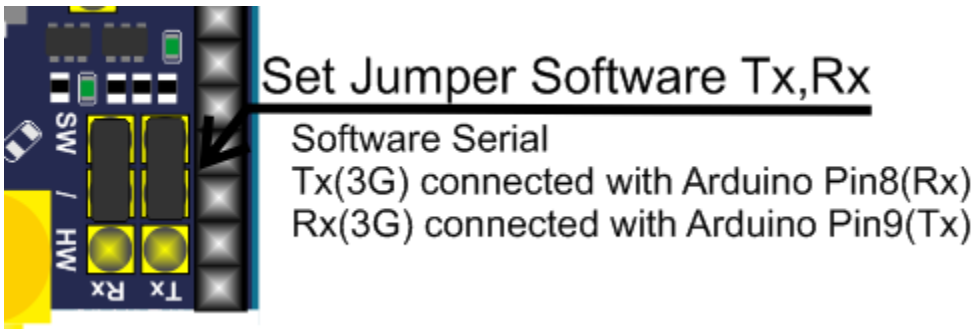
AltSoftSerial เป็นไลบรารีสำหรับใช้งานขา GPIO ที่กำหนดให้ทำงานเป็น Software Serial ซึ่งสามารถทำงานใน Baud Rate ที่สูงกว่าไลบรารี SoftwareSerial ปกติที่มากับตัว Arduino IDE โดยสามารถใช้กับบอร์ดได้หลายรุ่น โดย AltSoftSerial กำหนดให้ใช้กับขาสัญญาณของบอร์ดรุ่นต่างๆ ดังนี้

Board	Transmit Pin	Receive Pin	Unusable PWM
Teensy 3.0 / 3.1 / 3.2	21	20	22
Teensy 2.0	9	10	(none)
Teensy++ 2.0	25	4	26, 27
Arduino Uno, Duemilanove, LilyPad, Mini (& other ATMEGA328)	9	8	10
Arduino Leonardo, Yun, Micro	5	13	(none)
Arduino Mega	46	48	44, 45
Wiring-S	5	6	4
Sanguino	13	14	12

อ้างอิงจาก http://www.pjrc.com/teensy/td_libs_AltSoftSerial.html

3G Shield ได้ออกแบบให้สามารถเลือกใช้งาน Software Serial โดยเลือกจัมพ์เปอร์ไปที่ SW ทำให้ขา Tx และ Rx เชื่อมต่อกับขา 8 และ 9 ของ Arduino Uno ทำให้ใช้งานร่วมกับ AltSoftSerial ได้

หมายเหตุ!!! Software Serial ควรใช้ Baud Rate ในช่วง 9600 – 57600 bps



ตัวอย่างการใช้งาน

```
// !!อย่าลืมสั่ง Power On Module
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
AltSoftSerial mySerial;
void setup(){
   Serial.begin(9600);
   gsm.begin(&mySerial,9600);
}
```

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

การใช้งานไลบรารี TEE_UC20_Shield ร่วมกับ SoftwareSerial

SoftwareSerial เป็นไลบรารีพื้นฐานที่ติดมากับ Arduino IDE และเป็นที่นิยมในการนำไปเชื่อมต่ออุปกรณ์กับ Arduino ผ่านทาง Serial Port สามารถเลือกกำหนด GPIO ได้หลายขาสัญญาณทำให้ยืดหยุ่นต่อการใช้งานแต่ใน Baud Rate ที่สูงกว่า 19200 bps จะพบว่ามีความผิดพลาดของข้อมูลที่รับเข้ามาค่อนข้างมาก

ตัวอย่างการใช้งาน

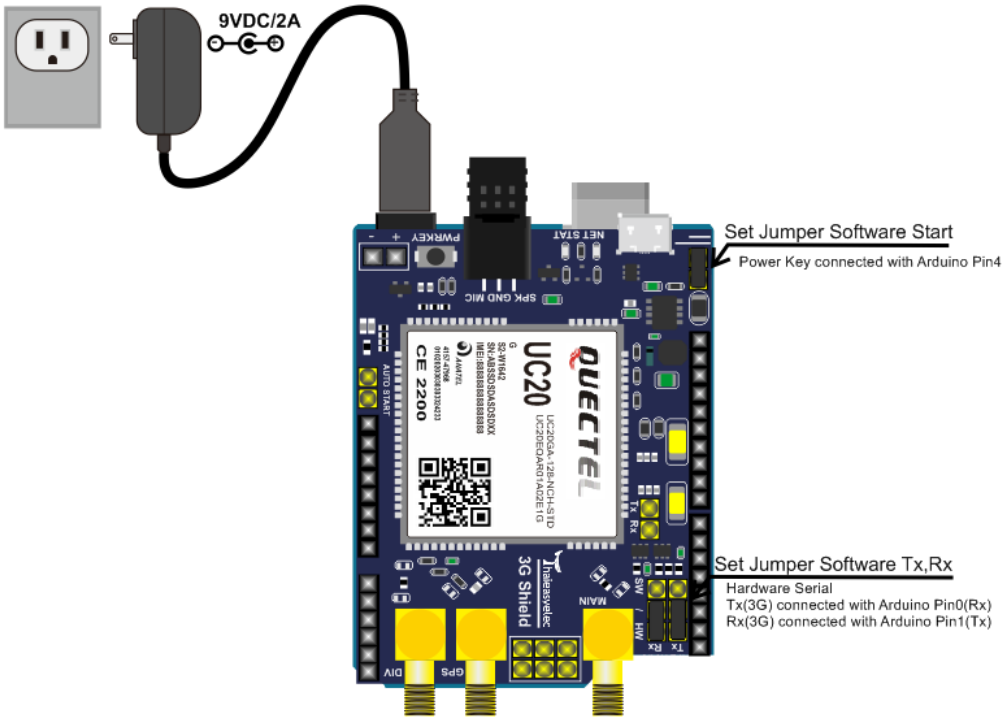
```
// !!อย่าลืมสั่ง Power On Module
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
SoftwareSerial mySerial(8,9);
void setup(){
     Serial.begin(9600);
     gsm.begin(&mySerial,9600);
}
void loop()
{
   if (gsm.available())
   {
     Serial.write(gsm.read());
   }
   if (Serial.available())
   {
     gsm.write(Serial.read());
   }
}
```

การใช้งานไลบรารี TEE_UC20_Shield ร่วมกับ Hardware Serial

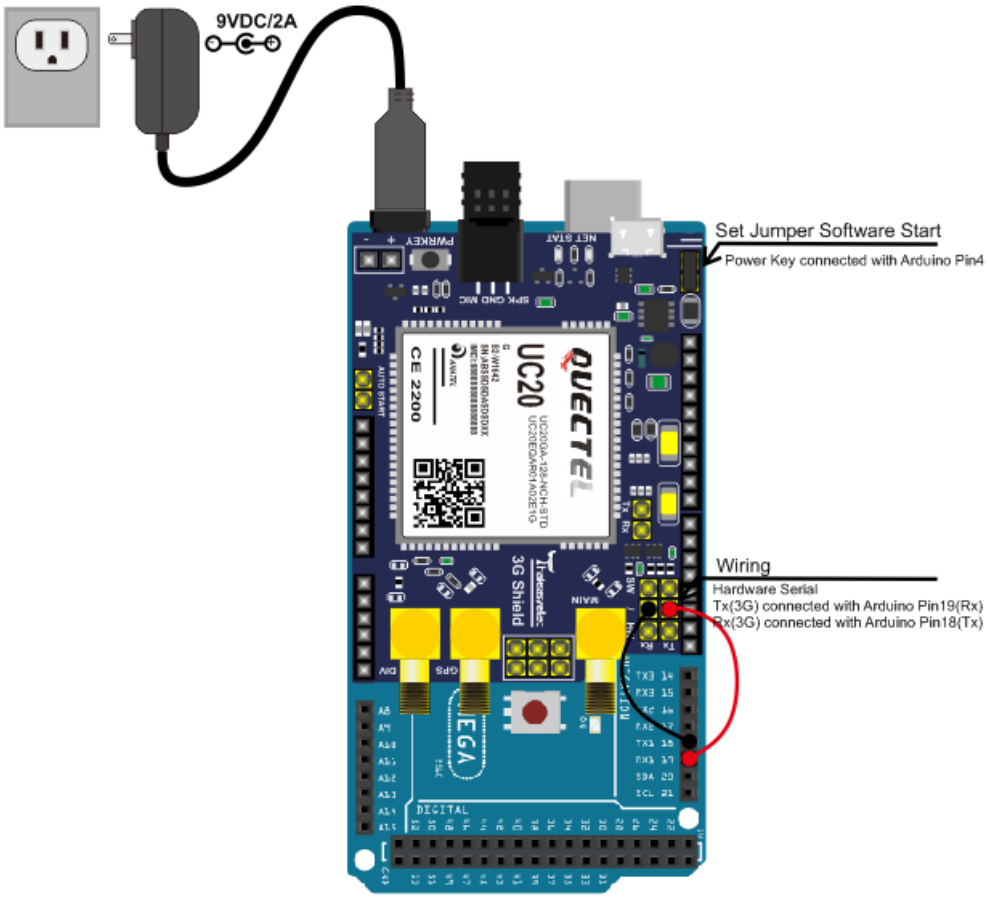
Hardware Serial คือ Serial ที่เป็น Hardware ของ Arduino โดยตรง สามารถทำงานได้ดีและมีประสิทธิภาพที่สุด แต่เนื่องจาก Arduino บางบอร์ดอย่าง Uno มี Hardware Serial เพียงแค่ช่องเดียวและใช้งานเป็นพอร์ตหลักสำหรับโปรแกรม การใช้งานจึงค่อนข้างลำบากเนื่องจากหากมี Hardware อื่นมาต่อพ่วง

ที่ขา 0 (Rx) และ 1 (Tx) ของ Uno อาจทำให้ไม่สามารถอัพโหลดโปรแกรมลงบอร์ด Arduino ได้ และพอร์ตนี้มักนิยมใช้ในการรับ-ส่งข้อมูลกับคอมพิวเตอร์หรือใช้เพื่อการดีบักแก้ปัญหาหรือดูสถานะการทำงานของโปรแกรม สำหรับการใช้งานกับ Arduino Uno จึงแนะนำให้ใช้ Software Serial จะสะดวกมากกว่า การใช้งาน Hardware Serial แนะนำให้ใช้กับบอร์ด Arduino ที่มี Hardware Serial มากกว่า 1 ช่อง เช่น Arduino Mega 2560 Arduino Due Arduino Leonardo (Hardware Serial แยกกับช่องอัพโปรแกรม USB CDC)

- การต่อใช้งาน Hardware Serial กับ Arduino Uno และ Leonardo



- การต่อใช้งาน Hardware Serial กับ Arduino Mega 2560



ตัวอย่างการใช้งาน กับ Arduino Mega

```
// !!อย่าลืมสั่ง Power On Module
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
void setup()
{
      Serial.begin(9600);
      gsm.begin(&Serial1,9600);
}
void loop()
{
    if (gsm.available())
    {
      Serial.write(gsm.read());
    }
    if (Serial.available())
    {
      gsm.write(Serial.read());
    }
}
```

Note!!!

```
gsm.begin(&Serial,9600);      //สำหรับใช้งานกับ Serial Pin 0, 1 ของ Uno
gsm.begin(&Serial1,9600);     //สำหรับใช้งานกับ Serial1 Pin 19, 18 ของ Mega 2560
gsm.begin(&Serial2,9600);     //สำหรับใช้งานกับ Serial2 Pin 17, 16 ของ Mega 2560
gsm.begin(&Serial3,9600);     //สำหรับใช้งานกับ Serial3 Pin 15, 14 ของ Mega 2560
```

การใช้งานไลบรารี TEE_UC20_Shield

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
```

ฟังก์ชัน begin()

void begin(SerialPort, baudrate) คือ ฟังก์ชันสำหรับตั้งค่าเริ่มต้นให้โมดูลใช้ไลบรารีของ Serial จาก AltSoftSerial SoftwareSerial หรือ HardwareSerial โดยในส่วนหัวต้นโปรแกรมก่อนเข้า setup() เราประกาศตัวแปรสร้างออบเจคต์ของ Serial ตัวอย่างเช่น

`AltSoftSerial mySerial;` เมื่อเรียกใช้ AltSoftSerial หรือ

`SoftwareSerial mySerial(8,9);` เมื่อเรียกใช้ Software Serial

และกำหนดค่า Baud Rate ในการสื่อสารกับ 3G Shield / 3G Module (สามารถดูตัวอย่างใช้งานจากหัวข้อการตั้งค่า UART สำหรับใช้งานไลบรารี TEE_UC20_Shield)

ตัวแปร Event_debug

Event_debug เป็นตัวแปรสำหรับกำหนดแอดเดรสของฟังก์ชันที่มารองรับการแสดงผลการดีบักของไลบรารี ซึ่งฟังก์ชันที่จะมารับข้อมูลดีบักจะอยู่ในรูปแบบ void functionname(String data) โดย functionname คือค่าที่กำหนดให้ตัวแปร Event_debug และข้อมูลจะถูกส่งมายังตัวแปร data ของฟังก์ชันดังกล่าว

ตัวอย่าง

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
AltSoftSerial mySerial;
void debug(String data){
     Serial.println(data);
}
void setup(){
     Serial.begin(9600);
     gsm.begin(&mySerial,9600);
     gsm.Event_debug = debug;
}
void loop(){

}
```

ฟังก์ชัน SetPowerKeyPin()

void SetPowerKeyPin(int pin) คือ ฟังก์ชันกำหนดค่าขา (Pin) ที่ใช้สั่งเปิด-ปิด 3G Shield / 3G Module โดยกำหนดหมายเลขขาที่ต้องการใช้งานลงในพารามิเตอร์ pin โดยปกติแล้วหากไม่มีการตั้งค่าจากฟังก์ชันนี้จะใช้ขา Digital 4 ของ Arduino

ฟังก์ชัน PowerOn()

void PowerOn() คือ ฟังก์ชันเปิดใช้งาน 3G Shield / 3G Module โดยใช้ขาสัญญาณของ Arduino ที่เชื่อมต่อกับ Power Key ของ 3G Module ปกติเป็นดิจิตอลขา 4 สามารถเปลี่ยนขาได้จากฟังก์ชัน SetPowerKeyPin() การทำงานของฟังก์ชันนี้จะสั่งทริกสัญญาณไปเปิดให้ 3G Shield / 3G Module ทำงาน แต่หากกำลังทำงานอยู่แล้วเป็นการสั่งให้ปิด 3G Shield / 3G Module แล้วเปิดขึ้นมาใหม่

ฟังก์ชัน PowerOff()

void PowerOff() คือ ฟังก์ชันปิดการทำงานของ 3G Shield / 3G Module โดยใช้ขาของ Arduino ที่เชื่อมต่อกับขา Power Key ของ 3G Shield / 3G Module สามารถเปลี่ยนขาได้จากฟังก์ชัน SetPowerKeyPin()

ฟังก์ชัน WaitReady()

bool WaitReady() คือ ฟังก์ชันรอให้ 3G Shield / 3G Module พร้อมเมื่อเปิดใช้งาน เนื่องจากการเปิดโมดูลขึ้นมาจะใช้เวลาสำหรับ Initial การทำงานต่างๆ เช่น การจับสัญญาณเข้ากับเครือข่าย การตรวจสอบ SIM Card เป็นต้น โดยฟังก์ชันนี้จะให้ค่าตอบกลับ Boolean เป็น false ออกมาเมื่อโมดูลพร้อมทำงานแล้ว

ฟังก์ชัน GetOperator

String GetOperator() คือ ฟังก์ชันตรวจสอบว่าตอนนี้ 3G Shield / 3G Module กำลังเชื่อมต่อกับเครือข่ายของผู้ให้บริการใด โดยโดยนี้จะตอบกลับข้อมูล String เป็นชื่อของผู้ให้บริการออกมา

ฟังก์ชัน SignalQuality()

unsigned char SignalQuality() คือ ฟังก์ชันสำหรับตรวจสอบคุณภาพสัญญาณ ที่ 3G Module ได้รับจากผู้ให้บริการเครือข่าย โดยฟังก์ชันนี้จะตอบกลับค่าความแรงของสัญญาณออกมาเป็น ตัวเลขที่มีความหมายดังนี้

ฟังก์ชัน Return	Signal Quality
0	-113 dBm or less
1	-111 dBm
2...30	-109... -53 dBm
31	-51 dBm or greater
99	Not known or not detectable

การใช้งานไลบรารี TEE_UC20_Shield โทรออกและรับสาย

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "call.h"
```

สร้างออบเจคต์ CALL

```
CALL call;
```

ฟังก์ชัน Call()

unsigned char Call(String call_number) คือ ฟังก์ชันโทรออกโดยใส่ String ของหมายเลขที่ต้องการโทรออกไปลงในพารามิเตอร์ call_number โดยฟังก์ชันจะตอบกลับค่าผลการทำงานของออกมาเป็นตัวเลขซึ่งมีความหมายดังนี้

ฟังก์ชัน Return	Description
0	Timeout
1	Ok
2	No Carrier
3	Busy

ฟังก์ชัน HangUp()

bool HangUp() คือ ฟังก์ชันวางสายโทรศัพท์ โดยฟังก์ชันจะตอบกลับเป็น true เมื่อวางสายสำเร็จ

ฟังก์ชัน DisconnectExisting()

bool DisconnectExisting() คือ ฟังก์ชันยกเลิกการทำงานทั้งหมดเกี่ยวกับการ Call โดยฟังก์ชันจะตอบกลับเป็น true เมื่อทำงานสำเร็จ

ฟังก์ชัน Answer()

bool Answer() คือ ฟังก์ชันรับสายโทรศัพท์ โดยฟังก์ชันจะตอบกลับมาเป็น true เมื่อรับสายได้

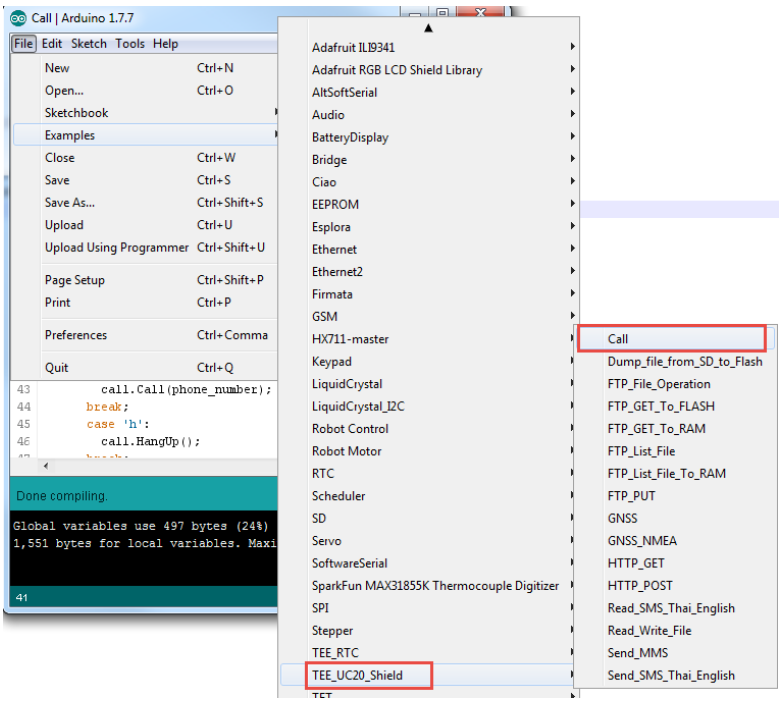
ฟังก์ชัน CurrentCallsMe

String CurrentCallsMe() คือ ฟังก์ชันแสดงค่าหมายเลขโทรที่กำลังโทรเข้ามาในขณะนั้น โดยฟังก์ชันจะตอบกลับเป็น String ของหมายเลขที่กำลังโทรศัพท์เข้ามา

ฟังก์ชัน WaitRing

bool WaitRing() คือ ฟังก์ชันตรวจจับสัญญาณ RING เมื่อมีสายโทรเข้ามา โดยฟังก์ชันจะตอบกลับมาเป็น true เมื่อมีสายโทรเข้ามาแล้วโมดูลตรวจจับ RING ได้

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module สำหรับการโทรเข้าและรับสายได้จาก File > Examples > TEE_UC20_Shield > Call



การใช้งานไลบรารี TEE_UC20_Shield รับ-ส่งข้อความแบบ SMS

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "sms.h"
```

สร้างออบเจคต์ SMS

```
SMS sms;
```

ฟังก์ชัน DefaultSetting()

void DefaultSetting() คือ ฟังก์ชันกำหนดค่าต่างๆ ที่สำหรับใช้งาน SMS เช่น ด้วยคำสั่ง AT+CMGF, AT+CSMP, AT+CSCS เป็นต้น

ฟังก์ชัน Start()

void Start(String rx_number) คือ ฟังก์ชันเริ่มต้นส่ง SMS และกำหนดหมายเลขโทรศัพท์ปลายทางที่ต้องการส่ง SMS โดยกำหนดเป็น String ของหมายเลขโทรศัพท์ที่ต้องการส่ง SMS

ฟังก์ชัน Send() และ Sendln()

void Send(String data) และ void Sendln(String data) คือ ฟังก์ชันส่งข้อความที่ต้องการไปยังผู้รับโดยกำหนด ข้อความที่จะส่งลงในพารามิเตอร์ String data

Send() คือ ส่งข้อความต่อกันไปเรื่อยๆ ในบรรทัดเดิม

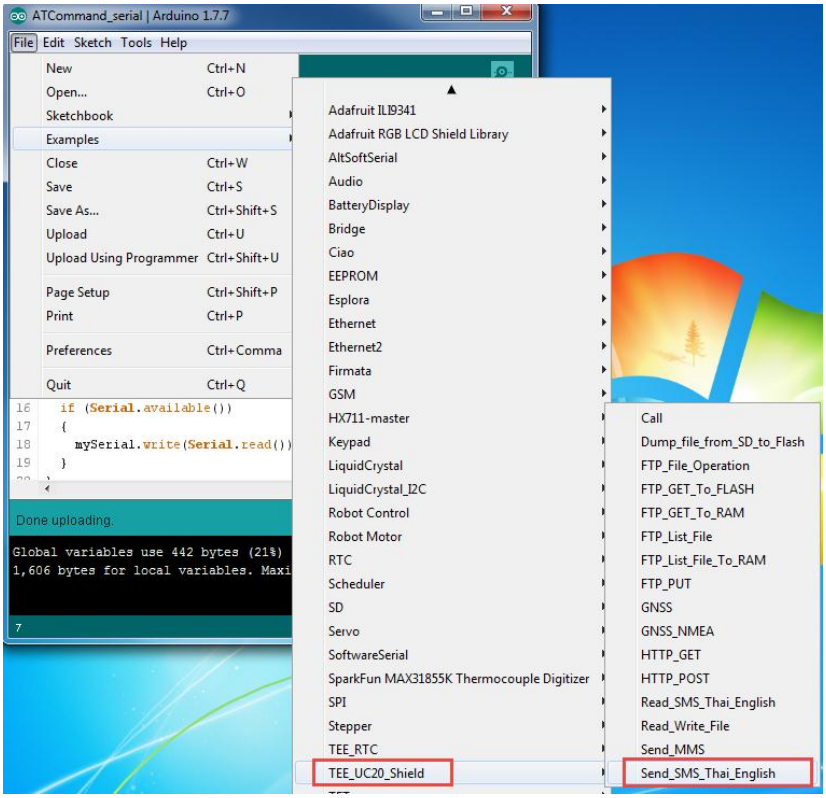
Sendln() คือ ส่งข้อความพร้อมส่งคำสั่งให้ขึ้นบรรทัดใหม่

ฟังก์ชัน Stop

void Stop() คือ ฟังก์ชันสิ้นสุดการส่งข้อความ

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module สำหรับการส่ง SMS ได้จาก

File > Examples > TEE_UC20_Shield > Send_SMS_Thai_English



ฟังก์ชัน IndexNewSMS()

unsigned char IndexNewSMS() คือ ฟังก์ชันแสดงค่าตำแหน่ง SMS ล่าสุดที่เข้ามาในกล่องข้อความ โดย ฟังก์ชันจะตอบกลับเป็นหมายเลข Index ของข้อความล่าสุด

ฟังก์ชัน ReadSMS()

String ReadSMS(int index) คือ ฟังก์ชันอ่าน SMS ในกล่องข้อความโดยกำหนดพารามิเตอร์ int index เป็นหมายเลขเพื่ออ่าน SMS ที่เก็บอยู่ใน Index ที่ต้องการอ่าน และฟังก์ชันจะตอบกลับมาเป็น String ข้อความที่ SMS ที่เก็บอยู่ใน Index นั้นออกมา

ตัวแปร SMSInfo

String SMSInfo เป็นตัวแปรที่เก็บรายละเอียดของ SMS ที่กำลังอ่านอยู่

ฟังก์ชัน ConvertStrUnicodeToTIS620()

String ConvertStrUnicodeToTIS620(String data) คือ ฟังก์ชันแปลงตัวอักษรจาก Unicode ให้เป็น TIS620 เพื่อแสดงข้อความที่เป็นภาษาไทยบนหน้าต่าง Serial Monitor ได้ โดยฟังก์ชันจะตอบกลับออกมาเป็น String

ฟังก์ชัน ConvertStrUnicodeToUTF8()

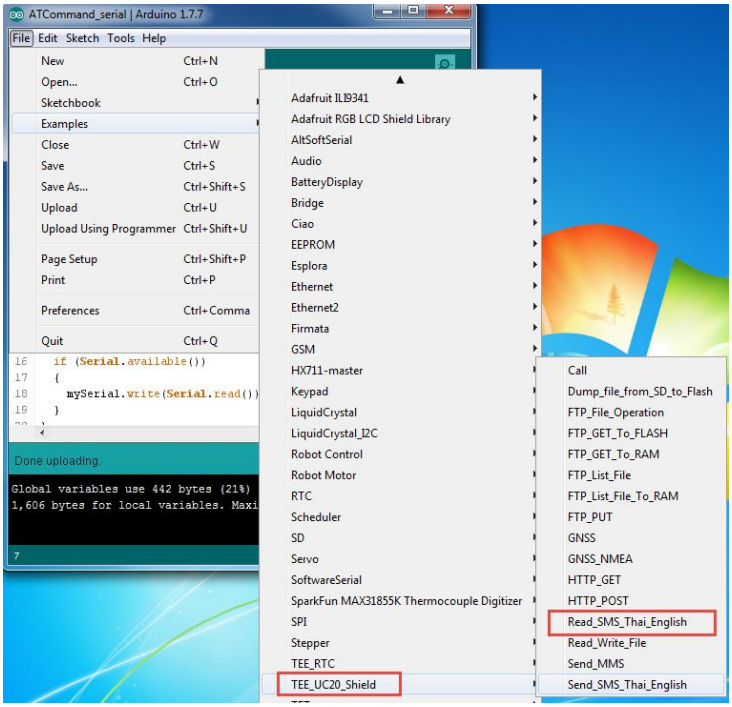
String ConvertStrUnicodeToUTF8(String data) คือ ฟังก์ชันแปลงตัวอักษรจาก Unicode ให้เป็น UTF-8 เพื่อนำข้อความไปใช้ทำเงื่อนไขเปรียบเทียบ โดยฟังก์ชันตอบกลับออกมาเป็น String

ฟังก์ชัน DeleteSMS()

bool DeleteSMS(int index) คือ ฟังก์ชันลบ SMS ในกล่องข้อความโดยกำหนด Index ของข้อความที่ต้องการลบผ่านพารามิเตอร์ int index และเมื่อฟังก์ชันทำงานสำเร็จจะตอบกลับเป็น true ออกมา

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module สำหรับการรับ SMS ได้จาก

File > Examples > TEE_UC20_Shield > Read_SMS_Thai_English



การใช้งานไลบรารี TEE_UC20_Shield ในการจัดการไฟล์

ภายใน 3G Shield / 3G Module มีส่วนสำหรับเก็บข้อมูลให้ใช้งานอยู่ภายในทั้งแบบชั่วคราวและถาวร ซึ่งเปิดให้ผู้ใช้สามารถนำข้อมูลเข้าไปเก็บบันทึกหรือพักข้อมูลก่อนนำไปใช้งานต่างๆ ได้ ยกตัวอย่างเช่น เก็บรูปภาพก่อนส่ง MMS หรือพักข้อมูลจากที่ดาวน์โหลดจาก Internet

ส่วนสำหรับเก็บข้อมูลของ 3G Shield และ 3G Module จะถูกแบ่งออกเป็น 2 ส่วนได้แก่

UFS (User File Storage directory) คือ ส่วนเก็บข้อมูลถาวร บันทึกข้อมูลลงใน Flash Memory หากเก็บข้อมูลไว้ในส่วน UFS ข้อมูลจะไม่สูญหายเมื่อไม่มีไฟฟ้าเลี้ยงโมดูล

ใน 3G Shield ซึ่งใช้ UC20 มีพื้นที่ใช้งานประมาณ 60 MB

ใน 3G Module ซึ่งใช้ UC15 มีพื้นที่ใช้งานประมาณ 70 MB

RAM (Random Access Memory) คือ ส่วนเก็บข้อมูลชั่วคราว สามารถเข้าถึงข้อมูลได้รวดเร็วกว่า UFS แต่ข้อมูลจะสูญหายเมื่อไม่มีไฟฟ้าเลี้ยงโมดูล

ใน 3G Shield ซึ่งใช้ UC20 มีพื้นที่ใช้งานประมาณ 2.5 MB

ใน 3G Module ซึ่งใช้ UC15 มีพื้นที่ใช้งานประมาณ 3 MB

หมายเหตุ!!! พื้นที่ว่างในใช้งานขึ้นอยู่กับปัจจัยหลายอย่าง ทั้งการเปิดใช้งานคุณสมบัติต่างๆ ของโมดูลซึ่งจะลดพื้นที่ของ RAM ลง และเฟิร์มแวร์ของโมดูลเวอร์ชั่นใหม่ๆ ซึ่งอาจใช้ UFS ของโมดูลมากกว่าเดิมในการเก็บส่วนที่รองรับฟีเจอร์การทำงานที่เพิ่มมากขึ้น ในการใช้งานทั่วไปแนะนำให้ใช้ UFS และคอยตรวจสอบพื้นที่ว่างบนโมดูลไม่ให้เก็บข้อมูลจนเต็ม ในส่วนของ RAM ไม่แนะนำให้ใช้งานจนเต็มเพราะจะส่งผลต่อการทำงานและสมรรถภาพโดยรวมของการทำงานของโมดูล

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "File.h"
```

สร้างออบเจคต์ UC_FILE

```
UC_FILE file;
```

ฟังก์ชัน begin()

void begin() คือ ฟังก์ชันกำหนดค่าเริ่มต้นการใช้งานเกี่ยวกับไฟล์

ฟังก์ชัน GetSpace()

long GetSpace(String pattern) คือ ฟังก์ชันแสดงขนาดหน่วยความจำทั้งหมด สามารถเลือกแสดงขนาดของหน่วยความจำในแต่ละส่วนโดยกำหนดพารามิเตอร์ลงในตัวแปร String pattern เช่น แ ส ด ง หน่วยความจำทั้งหมดของ UFS = file.GetSpace(“UFS”);

แสดงหน่วยความจำทั้งหมดของ RAM = file.GetSpace(“RAM”);

และฟังก์ชันจะตอบกลับขนาดของหน่วยความจำทั้งหมดเป็นจำนวนเต็มแบบ long

ฟังก์ชัน GetFreeSpace()

long GetFreeSpace(String pattern) คือ ฟฟังก์ชันแสดงขนาดหน่วยความจำที่ว่าง สามารถเลือกแสดงขนาดของหน่วยความจำในแต่ละส่วนโดยกำหนดพารามิเตอร์ลงในตัวแปร String pattern เช่น

แสดงหน่วยความจำที่ว่างของ UFS = file.GetFreeSpace (“UFS”);

แสดงหน่วยความจำที่ว่างของ RAM = file.GetFreeSpace(“RAM”);

และฟังก์ชันจะตอบกลับขนาดของหน่วยความจำที่ว่างเป็นจำนวนเต็มแบบ long

ฟังก์ชัน List()

List(String pattern) คือ ฟังก์ชันแสดงรายชื่อไฟล์ทั้งหมดในหน่วยความจำ สามารถเลือกแสดงไฟล์ของหน่วยความจำในแต่ละส่วนโดยกำหนดพารามิเตอร์ลงในตัวแปร String pattern เช่น

แสดงรายชื่อไฟล์ของ UFS = file. List (“UFS”);

แสดงรายชื่อไฟล์ของ RAM = file. List (“RAM”);

โดยฟังก์ชันนี้จะแสดงรายชื่อไฟล์ในตัวแปรที่เรากำหนด

ตัวแปร ListOutput

ListOutput เป็นตัวแปรสำหรับกำหนดตำแหน่งของฟังก์ชันที่มารองรับการแสดงผลของฟังก์ชัน List() ซึ่งฟังก์ชันที่จะมารับข้อมูลจะอยู่ในรูปแบบ void functionname(String data) โดย functionname คือ ค่าที่กำหนดให้ตัวแปร ListOutput และข้อมูลจะถูกส่งมายังตัวแปร data ของฟังก์ชันดังกล่าว

ฟังก์ชัน Open()

int Open(String pattern, String fn) คือ ฟังก์ชันเปิดไฟล์ขึ้นมาใช้งานเพื่ออ่านหรือเขียน สามารถเลือกตำแหน่งที่อยู่ของไฟล์โดยกำหนดพรารามิเตอร์ลงในตัวแปร String pattern และกำหนดชื่อไฟล์ได้จากพารามิเตอร์ String fn และเมื่อเปิดไฟล์สำเร็จฟังก์ชันตอบกลับเป็นหมายเลข Handle Number ออกมาเพื่อให้ใช้อ่านหรือเขียนไฟล์ต่อไป หากไม่สามารถเปิดไฟล์ได้ ฟังก์ชันตอบกลับค่าเป็น -1 ออกมา

ฟังก์ชัน Close()

bool Close(int handle) คือ ฟังก์ชันปิดไฟล์หลังจากอ่านหรือเขียนไฟล์แล้ว โดยการปิดไฟล์จะใช้หมายเลข Handle Number ที่ได้จากฟังก์ชัน Open() ในการระบุไฟล์ที่ต้องการปิด และเมื่อปิดไฟล์สำเร็จฟังก์ชันจะตอบกลับเป็น true ออกมา

ฟังก์ชัน BeginWrite()

BeginWrite(int handle,int size) คือ ฟังก์ชันเริ่มเขียนไฟล์ที่เปิดด้วยฟังก์ชัน Open() เอาไว้ โดยนำเอาหมายเลข Handel Number ที่ได้จากการเปิดไฟล์ใส่ลงในพารามิเตอร์ int handle และการเขียนไฟล์ต้องกำหนดขนาดของข้อมูลที่ต้องการลงในพารามิเตอร์ int size ฟังก์ชัน BeginWrite จะตอบกลับเป็นค่า true ออกมา

ฟังก์ชัน Write()

Write(char data) คือ ฟังก์ชันเขียนไฟล์โดยเขียนข้อมูลลงทีละ 1 ไบต์

ฟังก์ชัน Print()

Print(String data) คือ ฟังก์ชันเขียนไฟล์โดยส่งข้อมูลที่ต้องการเขียนลงไปเป็นข้อความ

ฟังก์ชัน Println()

Println(String data) คือ ฟังก์ชันเขียนไฟล์โดยส่งข้อมูลที่ต้องการเขียนลงไปเป็นข้อความและปิดท้ายด้วยการขึ้นบรรทัดใหม่ (0x0A,0x0D)

ฟังก์ชัน WaitFinish()

WaitFinish() คือ ฟังก์ชันรอให้การเขียนไฟล์สิ้นสุดลง

ฟังก์ชัน Seek()

Seek(int handle,long start_at) คือ ฟังก์ชันเลื่อนไปยังตำแหน่งที่ต้องการอ่านหรือเขียนไฟล์โดยกำหนดไฟล์ที่ต้องการด้วย Handle Number จากพารามิเตอร์ int handle และกำหนดตำแหน่งที่ต้องการเลื่อนไปอ่านหรือเขียนไฟล์จากพารามิเตอร์ long start_at

ตัวอย่างการใช้ฟังก์ชัน Open(), BeginWrite(), Print(), WaitFinish(), Close()

```
void write_file(String file_name,String data){
     int handle = file.Open(UFS,file_name);
     if(handle!=-1){
          if(file.BeginWrite(handle,data.length())){
               file.Print(data);
               file.WaitFinish();
          }
     }
     file.Close(handle);
}
```

ฟังก์ชัน Read()

Read(int handle,int buf_size,char *buf) คือ ฟังก์ชันอ่านไฟล์ที่เปิดเอาไว้ด้วยฟังก์ชัน Open() สามารถกำหนดไฟล์ที่ต้องการอ่านจากหมายเลข Handle Number โดยใช้พารามิเตอร์ int handle ในการอ่านไฟล์ต้องกำหนดขนาดของข้อมูลที่ต้องการอ่านออกมาโดยในพารามิเตอร์ int buf_size และกำหนดตำแหน่งของบัฟเฟอร์ที่จะนำมาเก็บข้อมูลที่อ่านออกมาเก็บไว้ในพารามิเตอร์ char *buf

ฟังก์ชัน ReadFile()

ReadFile(String pattern,String file_name) คือ ฟังก์ชันสำหรับอ่านไฟล์ที่อำนวยความสะดวกมากยิ่งขึ้นโดยรวมเอาฟังก์ชัน Open() และ Read() ไว้ในฟังก์ชันเดียว โดยกำหนดตำแหน่งที่อยู่ของไฟล์จากพารามิเตอร์ String pattern และชื่อไฟล์จากพารามิเตอร์ String file_name

ตัวแปร DataOutput

DataOutput เป็นตัวแปรสำหรับกำหนดแอดเดรสของฟังก์ชันที่มารองรับข้อมูลที่ได้จากการอ่านไฟล์ของฟังก์ชัน ReadFile() ซึ่งฟังก์ชันที่จะมารับข้อมูลจะอยู่ในรูปแบบ void functionname(char data) โดย functionname คือค่าที่กำหนดให้ตัวแปร DataOutput และข้อมูลจะถูกส่งมายังตัวแปร data ของฟังก์ชันดังกล่าว

ตัวอย่างการใช้ฟังก์ชัน ReadFile()

```
void data_out(char data){
     Serial.write(data);
}
void read_file(String pattern,String file_name){
     file.DataOutput =  data_out;
     file.ReadFile(pattern,file_name);
}
```

ฟังก์ชัน Delete()

Delete(String pattern,String fn) คือ ฟังก์ชันลบไฟล์โดยสามารถกำหนดที่อยู่ของไฟล์ที่ต้องการลบจากพารามิเตอร์ String pattern และกำหนดชื่อของไฟล์ที่ต้องการลบจากพารามิเตอร์ String fn ตัวอย่างเช่น

```
file.Delete(“UFS”,”test”);          //ลบไฟล์ชื่อ test ที่อยู่บน UFS
file.Delete(“RAM”,"*");             //ลบทุกไฟล์ที่อยู่บน RAM
```

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module สำหรับการอ่านและเขียนไฟล์ได้จาก

File > Examples > TEE_UC20_Shield > Read_Write_File

File > Examples > TEE_UC20_Shield > Dump_file_from_SD_to_Flash

การใช้งานไลบรารี TEE_UC20_Shield ในการเชื่อมต่อ Internet

การเชื่อมต่อ Internet เป็นส่วนสำคัญที่ต้องนำไปเป็นพื้นฐานในการใช้งานหัวข้ออื่นต่อไป เช่น การส่ง MMS การใช้ FTP การใช้ SMTP และการใช้งานเป็น Web client เป็นต้น ในส่วนนี้จึงเป็นเนื้อหาสำหรับใช้ 3G Shield / 3G Module เชื่อมต่อกับเครือข่าย Internet ของผู้ให้บริการรายต่างๆ ที่มีในไทยปัจจุบัน โดยแต่ละเครือข่ายในประเทศไทยจะใช้พารามิเตอร์การตั้งค่าการเชื่อมต่อไม่เหมือนกันดังตารางนี้

ตารางพารามิเตอร์การเชื่อมต่อของผู้ให้บริการเครือข่ายต่างๆ

ผู้ให้บริการ AIS

Parameter	Internet	MMS
APN	internet	multimedia
MMSC	-	http://mms.mobilelife.co.th
MMS Proxy	-	203.170.229.34
MMS Port	-	8080
User	-	-
Password	-	-
APN Type	default	mms

ผู้ให้บริการ DTAC

Parameter	Internet	MMS
APN	www.dtac.co.th	mms
MMSC	-	http://mms2.dtac.co.th/8002
MMS Proxy	-	203.155.200.133
MMS Port	-	8080
User	-	-
Password	-	-
APN Type	default	mms

ผู้ให้บริการ TRUE

Parameter	Internet	MMS
APN	internet	hmms
MMSC	-	http://mms.trueworld.net:8002
MMS Proxy	-	10.4.7.39
MMS Port	-	8080

User	true	true
Password	true	true
APN Type	default	mms

ผู้ให้บริการ TOT

Parameter	Internet	MMS
APN	internet	mms
MMSC	-	http://mmsc.tot3g.net:8002
MMS Proxy	-	10.218.24.83
MMS Port	-	8080
User	-	-
Password	-	-
APN Type	default	mms

ผู้ให้บริการ My by Cat

Parameter	Internet	MMS
APN	internet	ผู้ให้บริการยังไม่เปิดให้ใช้บริการ
MMSC	-	-
MMS Proxy	-	-
MMS Port	-	-
User	-	-
Password	-	-
APN Type	default	-

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "internet.h"
```

สร้างออบเจคต์ INTERNET

```
INTERNET net;
```

ฟังก์ชัน Configure()

bool INTERNET::Configure(String apn,String user,String password) คือ ฟังก์ชันตั้งค่าการเชื่อมต่อ Internet โดยมีพารามิเตอร์ดังต่อไปนี้ String apn, String user, String password ผู้ใช้งานต้องเลือกตั้งค่าพารามิเตอร์ตามตารางของผู้ให้บริการในแต่ละค่าย

ฟังก์ชัน Connect()

bool Connect() คือ ฟังก์ชันสั่งให้ 3G Shield / 3G Module เชื่อมต่อเข้ากับเครือข่าย Internet ของผู้ให้บริการตามพารามิเตอร์ที่กำหนดไว้ก่อนหน้าจากฟังก์ชัน Configure() เมื่อเชื่อมต่อสำเร็จฟังก์ชันจะตอบกลับมาเป็นค่า true

ฟังก์ชัน DisConnect()

bool DisConnect() คือ ฟังก์ชันสั่งให้ 3G Shield / 3G Module ยกเลิกการเชื่อมต่อกับเครือข่าย Internet โดยเมื่อยกเลิกการเชื่อมต่อสำเร็จ ฟังก์ชันจะตอบกลับมาเป็นค่า true

ฟังก์ชัน GetIP()

String INTERNET::GetIP() คือ ฟังก์ชันแสดงค่า IP Address ของตัว 3G Module ที่ได้รับจากผู้ให้บริการหลังจากการเชื่อมต่อกับเครือข่าย Internet ได้แล้ว โดยฟังก์ชันจะตอบกลับออกมาเป็นค่า String ip

การใช้งานไลบรารี TEE_UC20_Shield ในการส่ง MMS

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "internet.h"
#include "File.h"
#include "mms.h"
```

สร้างออบเจคต์ INTERNET UC_FILE และ MMS

```
INTERNET net;
UC_FILE file;
MMS mms;
```

ฟังก์ชัน SetMMSC()

bool SetMMSC(String data) คือ ฟังก์ชันตั้งค่า MMSC ให้ผู้ใช้ตั้งค่า MSSC ตามที่ผู้ให้บริการเครือข่ายเป็นผู้กำหนด สามารถดูข้อมูลได้จากตารางของเครือข่ายต่างๆ ในหัวข้อการเชื่อมต่อ Internet

ฟังก์ชัน SetProxy()

bool SetProxy(String ip,String port) คือ ฟังก์ชันตั้งค่า IP Address ของ Proxy และ Port ที่ใช้ในการส่ง ให้ผู้ใช้ตั้งค่าตามที่ผู้ให้บริการเครือข่ายเป็นผู้กำหนด สามารถดูข้อมูลได้จากตารางของเครือข่ายต่างๆ ในหัวข้อการเชื่อมต่อ Internet

ฟังก์ชัน Title()

bool Title(String title) คือ ฟังก์ชันตั้งชื่อหัวข้อของ MMS

ฟังก์ชัน SendTo()

SendTo(String receive) คือ ฟังก์ชันตั้งค่าหมายเลขโทรศัพท์ของผู้รับ MMS

ฟังก์ชัน AddFile()

bool AddFile(String pattern,String Filename) คือ ฟังก์ชันเพิ่มไฟล์จาก UFS หรือ RAM เข้าไปสร้างรายการไฟล์สำหรับส่ง MMS

ฟังก์ชัน ListMMSFile

String ListMMSFile() คือ ฟังก์ชันแสดงรายชื่อไฟล์ที่ถูกเพิ่มเข้ามาไว้ในรายการเพื่อส่ง MMS โดยฟังก์ชันตอบกลับชื่อของไฟล์ทั้งหมดออกมาเป็น String

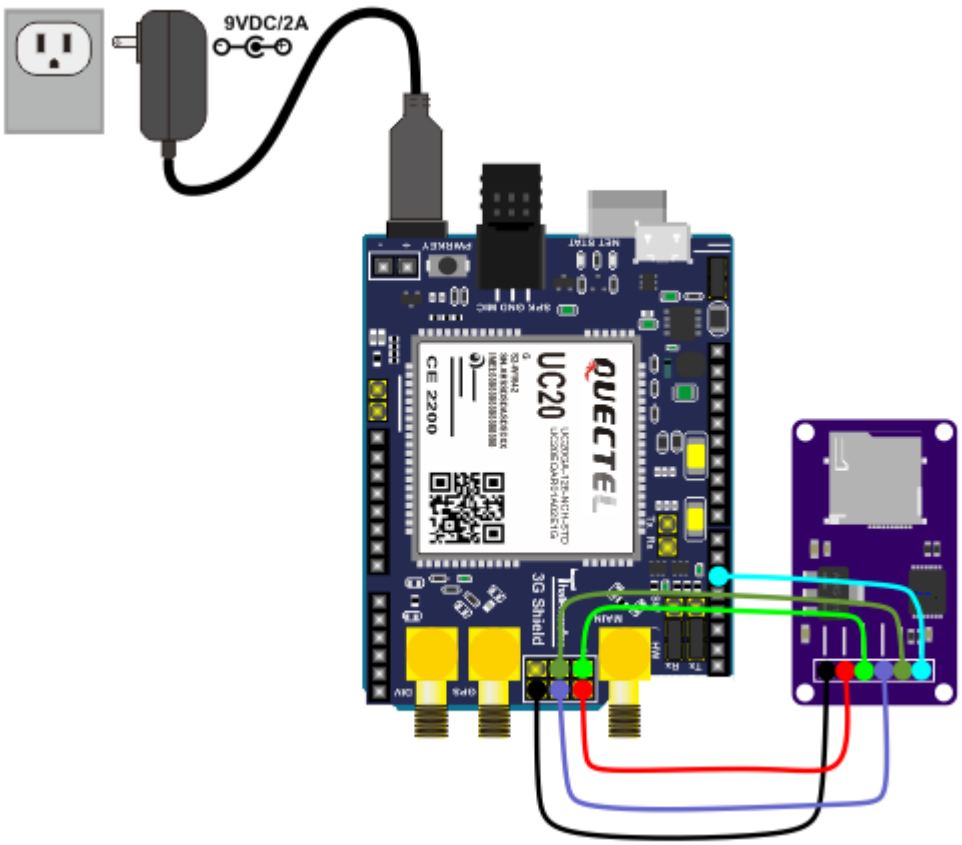
ฟังก์ชัน Send()

String Send() คือ ฟังก์ชันส่ง MMS ไปยังหมายเลขโทรศัพท์ที่กำหนดไว้ ฟังก์ชันตอบกลับผลการส่ง MMS ออกมาเป็น String

ฟังก์ชัน Clear()

bool Clear() คือ ฟังก์ชันล้างการตั้งค่าต่างๆ รวมทั้งไฟล์ที่เพิ่มไว้สำหรับส่ง MMS

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module ในการส่ง MMS ได้จาก
File > Examples > TEE_UC20_Shield > Send_MMS



ภาพวิธีต่อโมดูล SD Card เพื่อทดสอบตัวอย่างการส่ง MMS

การเตรียม SD Card

ข้อมูลใน SD Card ให้ใส่ไฟล์รูปภาพนามสกุล .jpg เป็นภาพอะไรก็ได้ตั้งชื่อว่า pic.jpg โปรแกรมตัวอย่างจะอ่านรูปภาพจาก SD Card ไปเก็บบนหน่วยความจำของ 3G Shield / 3G Module แล้วส่ง SMS รูปภาพนี้ไปยังหมายเลขโทรศัพท์ของผู้รับ

การใช้งานไลบรารี TEE_UC20_Shield ในการสื่อสาร TCP

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "internet.h"
#include "tcp.h"
```

สร้างออบเจคต์ INTERNET และ TCP

```
INTERNET net;
TCP tcp;
```

ฟังก์ชัน Open()

bool Open(String ip_url,String port) คือ ฟังก์ชันเริ่มต้นการเชื่อมต่อ TCP กับ Server โดยผู้ใช้สามารถกำหนด IP Address หรือ URL ของ Server ได้จากพารามิเตอร์ String ip_url และกำหนด TCP Port ในการสื่อสารได้จากพารามิเตอร์ String port ฟังก์ชันจะตอบกลับเป็น true เมื่อสามารถเชื่อมต่อกับ Server ได้สำเร็จและตอบกลับเป็น false หากเชื่อมต่อไม่สำเร็จ

ฟังก์ชัน Close()

bool Close() คือ ฟังก์ชันปิดการเชื่อมต่อ TCP กับ Server ฟังก์ชันตอบกลับเป็น true เมื่อสามารถปิดการเชื่อมต่อกับ Server ได้สำเร็จ และตอบกลับเป็น false หากปิดการเชื่อมต่อไม่สำเร็จ

ฟังก์ชัน CheckConnection()

bool CheckConnection คือ ฟังก์ชันตรวจสอบว่ามีการเปิดการเชื่อมต่อ TCP กับ Server อยู่หรือไม่ โดยฟังก์ชันจะตอบกลับเป็น true เมื่อมีการสร้างการเชื่อมต่อกับ Server อยู่ และตอบกลับเป็น false หากไม่มีการเชื่อมต่อกับ Server อยู่ในขณะนั้น

ฟังก์ชัน StartSend

bool StartSend() คือ ฟังก์ชันเริ่มต้นการส่งส่งข้อมูลผ่าน TCP ไปยัง Server ฟังก์ชันจะตอบกลับ true เมื่อสามารถเชื่อมต่อกับ Server ได้สำเร็จ และ Return false เมื่อเชื่อมต่อไม่สำเร็จ

ฟังก์ชัน Write()

Write(char data) คือ ฟังก์ชันส่งข้อมูลทีละ 1 ไบต์

ฟังก์ชัน Print()

Print(String data) คือ ฟังก์ชันส่งข้อมูลเป็นข้อความ

ฟังก์ชัน Println()

Println(String data) คือ ฟังก์ชันส่งข้อมูลเป็นข้อความและปิดท้ายด้วยการขึ้นบรรทัดใหม่ (0x0A,0x0D)

ฟังก์ชัน StopSend()

bool StopSend() คือ ฟังก์ชันสิ้นสุดการส่งข้อมูลผ่าน TCP ไปยัง Server

ฟังก์ชัน ReceiveAvailable()

bool ReceiveAvailable() คือ ฟังก์ชันตรวจสอบว่ามีข้อมูลตอบกลับมาจาก Server หรือไม่ โดยฟังก์ชันจะได้ค่าเป็น true ออกมาเมื่อมีข้อมูลตอบกลับมา แต่ขณะที่ไม่มีข้อมูลตอบกลับมาจะได้ค่าเป็น False

ฟังก์ชัน ReadBuffer()

String ReadBuffer() คือ ฟังก์ชันสั่งให้ 3G Module อ่านข้อมูลที่ Server ตอบกลับมาซึ่งอยู่ในบัฟเฟอร์ส่งออกมาทาง UART ฟังก์ชันตอบกลับมาเป็นข้อมูลที่อยู่ในบัฟเฟอร์

ฟังก์ชัน Ping()

void Ping(unsigned char contextid,String ip_url) คือ ฟังก์ชันตรวจสอบการเชื่อมต่อโดยการส่ง Ping ไปยัง Server ใช้พารามิเตอร์ unsigned char contextid เพื่อกำหนดช่องในการเชื่อมต่อและพารามิเตอร์ String ip_url เพื่อกำหนด IP Address หรือ URL ของ Server

ฟังก์ชัน NTP()

String NTP(unsigned char contexid,String ip_url,String port) คือ ฟังก์ชันรับค่าเวลาจาก Server ใช้พารามิเตอร์ unsigned char contextid เพื่อกำหนดช่องในการเชื่อมต่อ พารามิเตอร์ String ip_url กำหนด IP Address หรือ URL ของ Server และพารามิเตอร์ String port เพื่อกำหนด Port ที่เชื่อมต่อ

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module ในการทำงาน TCP ได้จาก

File > Examples > TEE_UC20_Shield > TCP

File > Examples > TEE_UC20_Shield > Ping

File > Examples > TEE_UC20_Shield > NTP

การใช้งานไลบรารี TEE_UC20_Shield ทำ HTTP GET / HTTP POST

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "internet.h"
#include "File.h"
#include "http.h"
```

สร้าง Object INTERNET , UC_FILE , HTTP

```
INTERNET net;
UC_FILE file;
HTTP http;
```

ฟังก์ชัน begin()

bool begin(unsigned char context_ID) คือ ฟังก์ชันเริ่มต้นการใช้งาน HTTP โดยผู้ใช้สามารถกำหนด Context ID หรือตัวชี้ตำแหน่ง PDP (Packet Data Protocol) ได้จากพารามิเตอร์ unsigned char context_ID มีค่าตั้งแต่ 1 ถึง 16

ฟังก์ชัน url()

bool url(String url) คือ ฟังก์ชันกำหนด URL ที่ต้องการส่ง HTTP GET หรือ HTTP POST

ฟังก์ชัน get()

int get() คือ คำสั่งส่ง HTTP GET ไปยัง Server และฟังก์ชัน จะตอบกลับผลการทำงานของ HTTP GET ออกมาเป็นตัวเลขดังตารางแสดงค่า Return GET/POST

ฟังก์ชัน post()

int post() คือ ฟังก์ชันส่ง HTTP POST ไปยัง Server และฟังก์ชัน จะตอบกลับผลการทำงานของ HTTP GET ออกมาเป็นตัวเลขดังตารางแสดงค่า Return GET/POST

ตารางแสดงค่า Return GET/POST

<httprspcode>	Meaning
200	OK
403	Forbidden
404	Not found
409	Conflict
411	Length required
500	Internal Server error

ฟังก์ชัน ReadData()

void ReadData() คือ ฟังก์ชันให้อ่านข้อมูลที่ Server ตอบกลับมาออกทาง Serial UART ของ 3G Shield / 3G Module โดยไม่เก็บข้อมูลเอาไว้บน UFS หรือ RAM

ฟังก์ชัน SaveResponseToMemory()

bool SaveResponseToMemory(String pattern,String Filename) คือ ฟังก์ชันอ่านข้อมูลที่ Server ส่งค่ากลับมาเก็บลงในหน่วยความจำโดยสามารถเลือกที่เก็บข้อมูลได้ทั้งบน UFS และ RAM ด้วยพารามิเตอร์ String pattern และตั้งชื่อไฟล์ด้วยพารามิเตอร์ String Filename

จากนั้นหากต้องการนำข้อมูลออกมาให้ใช้ฟังก์ชัน ReadFile ในหัวข้อการจัดการไฟล์

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module การทำ HTTP GET / HTTP POST ได้จาก

File > Examples > TEE_UC20_Shield > HTTP_GET

File > Examples > TEE_UC20_Shield > HTTP_POST

การใช้งานไลบรารี TEE_UC20_Shield ในการทำ FTP

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "internet.h"
#include "File.h"
#include "ftp.h"
```

สร้างออบเจ็ค INTERNET UC_FILE FTP

```
INTERNET net;
UC_FILE file;
FTP ftp;
```

ฟังก์ชัน begin()

bool begin(unsigned char context_ID) คือ ฟังก์ชันเริ่มต้นการใช้งาน FTP โดยผู้ใช้สามารถกำหนด Context ID หรือตัวชี้ตำแหน่ง PDP (Packet Data Protocol) ได้จากพารามิเตอร์ unsigned char context_ID มีค่าตั้งแต่ 1 ถึง 16

ฟังก์ชัน SetUsernamePassword()

bool SetUsernamePassword(String user,String pass) คือ ฟังก์ชันกำหนดชื่อผู้ใช้ (Username) และรหัสผ่าน (Password) ที่ต้องการใช้ล็อกอินเข้าระบบของ FTP Server

ฟังก์ชัน SetFileType()

bool SetFileType(unsigned char type) คือ ฟังก์ชันตั้งค่าชนิดของไฟล์ที่ต้องการรับสั่ง FTP โดยสามารถกำหนดได้ 2 ชนิด คือ 0 = Binary และ 1 = ASCII

ฟังก์ชัน SetTransMode()

bool SetTransMode(unsigned char type) คือ ฟังก์ชันตั้งค่าการส่งข้อมูลกับ FTP Server โดยสามารถกำหนดได้ 2 ชนิดคือ 0 = Active mode และ 1 = Passive mode

ฟังก์ชัน SetTimeout()

SetTimeout(int t) คือ ฟังก์ชันกำหนดเวลา Timeout เมื่อเชื่อมต่อกับ FTP Server แล้วไม่มีการตอบสนอง สามารถกำหนด Timeout ได้ตั้งแต่ 20 ถึง 180 วินาที (ค่า default value = 90 วินาที)

ฟังก์ชัน LoginServer()

int LoginServer(String serv,int port) คือ ฟังก์ชันสั่งให้ 3G Module ลอกอินไปยัง FTP Server โดยกำหนด URL หรือ IP Address ของ Server จากพารามิเตอร์ String serv และกำหนด Port จากพารามิเตอร์ int port ฟังก์ชันจะตอบกลับผลของการลอกอินเป็นตัวเลข 0 = สำเร็จ สามารถดูตารางค่าตอบกลับเพิ่มเติมได้จากเอกสาร Quectel UC20 FTP AT Commands Manual V1.1 ในบทที่ 4

ฟังก์ชัน Logout()

int Logout() คือ ฟังก์ชันสั่งให้ 3G Module ลอกเอาต์ออกจาก FTP Server โดยฟังก์ชันตอบกลับผลของการลอกเอาต์ Logout เป็นตัวเลข สามารถดูตารางค่าตอบกลับได้จากเอกสาร Quectel UC20 FTP AT Commands Manual V1.1 ในบทที่ 4

ฟังก์ชัน SetPath()

int SetPath(String path) คือ ฟังก์ชันเข้าถึง Path หรือไดเรกทอรีของไฟล์ที่ต้องการเข้าถึงบน FTP Server ตามสิทธิ์การใช้งานของผู้ใช้ที่ลอกอินเข้าไป

ฟังก์ชัน List()

bool List(String path) คือ ฟังก์ชันแสดงรายชื่อไฟล์ที่อยู่ใน Path นั้นๆ แสดงผ่านทางตัวแปร ListOutput

ตัวแปร ListOutput

ListOutput เป็นตัวแปรสำหรับกำหนดตำแหน่งของฟังก์ชันที่มารองรับการแสดงผลของฟังก์ชัน List() ซึ่งฟังก์ชันที่จะมารับข้อมูลจะอยู่ในรูปแบบ void functionname(String data) โดย functionname คือ ค่าที่กำหนดให้ตัวแปร ListOutput และข้อมูลจะถูกส่งมายังตัวแปร data ของฟังก์ชันดังกล่าว

ฟังก์ชัน ListToMemory()

Int ListToMemory(String path,String pattern,String fn) คือ ฟังก์ชันสำหรับแสดงรายชื่อไฟล์บน FTP Server แล้วเก็บลงบน UFS หรือ RAM โดยกำหนด Path ที่ต้องการแสดงรายชื่อด้วยพารามิเตอร์ String path กำหนดตำแหน่งที่เก็บด้วยพารามิเตอร์ String pattern และกำหนดชื่อไฟล์ด้วยพารามิเตอร์ String fn

ฟังก์ชัน MakeFolder()

int MakeFolder(String name) คือ ฟังก์ชันสร้างโฟลเดอร์หรือไดเรกทอรีบน FTP Server สามารถกำหนดชื่อโฟลเดอร์ด้วยพารามิเตอร์ String name

ฟังก์ชัน RenameFolder()

int RenameFolder(String name_old,String name_new) คือ ฟังก์ชันเปลี่ยนชื่อโฟลเดอร์หรือไดเรกทอรีบน FTP Server สามารถกำหนดชื่อโฟลเดอร์ที่ต้องการเปลี่ยนด้วยพารามิเตอร์ String name_old และกำหนดชื่อใหม่ด้วยพารามิเตอร์ String name_new

ฟังก์ชัน DeleteFolder()

Int DeleteFolder(String name) คือ ฟังก์ชันลบโฟลเดอร์หรือไดเรกทอรีบน FTP Server สามารถกำหนดชื่อโฟลเดอร์ที่ต้องการลบด้วยพารามิเตอร์ String name

ฟังก์ชัน DeleteFile()

int DeleteFile(String name) คือ ฟังก์ชันลบไฟล์บน FTP Server สามารถกำหนดชื่อไฟล์ที่ต้องการลบได้ด้วยพารามิเตอร์ String name

ฟังก์ชัน put()

int put(String File,String pattern,String fn,int startpos,int uploadlen,int beof) คือ ฟังก์ชันอัพโหลดไฟล์ขึ้นไปบน FTP Server โดยมีพารามิเตอร์ดังต่อไปนี้

String File	คือ ชื่อไฟล์ปลายทางที่ต้องการอัพโหลดไว้บน FTP Server (ความยาวไม่เกิน 50 ไบต์)
String pattern	คือ ที่อยู่ของไฟล์ที่ต้องการอัพโหลดไปยัง Server สามารถเลือกได้จาก 3 แหล่งได้แก่ UFS RAM และ COM (Stream ข้อมูลจาก Serial UART)
String fn	คือ ชื่อไฟล์ที่อยู่ใน UFS หรือ RAM ที่ต้องการอัพโหลดไปยัง FTP Server
int startpos	คือ ตำแหน่งของไฟล์ที่เริ่มต้นอัพโหลด
int uploadlen	คือ จำนวนไบต์ที่ต้องการ Stream ข้อมูลจาก COM ไปยัง FTP Server
int beof	คือ ตัวกำหนดการอัพโหลดโดยกำหนดได้ 2 ค่า คือ
	0 เมื่ออัพโหลดจนครบตามจำนวนไบต์ที่กำหนดใน uploadlen แล้วจะยังไม่ปิดไฟล์สามารถสั่ง put() ต่อเนื่องเพื่อ Stream ข้อมูลเข้าไปในไฟล์เดิมได้
	1 เมื่ออัพโหลดจนครบตามจำนวนไบต์ที่กำหนดใน uploadlen แล้วจะปิดไฟล์สิ้นสุดการอัพโหลดทันที

ฟังก์ชัน get()

int get(String File,String pattern,String fn,int startpos,int downloadlen) คือ ฟังก์ชันดาวน์โหลดไฟล์จาก FTP Server โดยมีพารามิเตอร์ดังต่อไปนี้

String File	คือ ชื่อไฟล์ที่ต้องการดาวน์โหลดลงจาก FTP Server (ความยาวไม่เกิน 50 ไบต์)
String pattern	คือ ที่อยู่ของไฟล์ที่ต้องการดาวน์โหลดจาก Server สามารถเลือกได้จาก 3 แหล่งได้แก่ UFS RAM และ COM (Stream ข้อมูลจาก Serial UART)
String fn	คือ ชื่อไฟล์ที่ต้องการเก็บไว้อยู่ใน UFS หรือ RAM
int startpos	คือ ตำแหน่งของไฟล์ที่เริ่มต้นดาวน์โหลด
int downloadlen	คือ จำนวนไบต์ที่ต้องการดาวน์โหลด

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module การทำ FTP ได้จาก

File > Examples > TEE_UC20_Shield > FTP_File_Operation

File > Examples > TEE_UC20_Shield > FTP_List_File

File > Examples > TEE_UC20_Shield > FTP_List_File_To_RAM

File > Examples > TEE_UC20_Shield > FTP_PUT

File > Examples > TEE_UC20_Shield > FTP_GET_To_FLASH

File > Examples > TEE_UC20_Shield > FTP_GET_To_RAM

การใช้งานไลบรารี TEE_UC20_Shield ทำงานกับ GNSS

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "gnss.h"
```

สร้างออบเจคต์ GNSS

```
GNSS gps;
```

ฟังก์ชัน Start()

bool Start() คือ ฟังก์ชันเปิดการใช้งาน GPS/GNSS บน 3G Module

ฟังก์ชัน Stop

bool Stop() คือ ฟังก์ชันปิดการใช้งาน GPS/GNSS บน 3G Module

ฟังก์ชัน GetPosition

String GetPosition() คือ ฟังก์ชันแสดงข้อมูลที่ได้รับจาก GPS/GNSS เมื่อโมดูลระบุตำแหน่งได้จะตอบกลับคำสั่งมีรูปแบบดังนี้

+QGPSLOC: <UTC>,<latitude>,<longitude>,<hdop>,<altitude>,<fix>,<cog>,<spkm>,<spkn>,<date>,<nsat>

Parameter	Meaning
<UTC>	UTC time. Format: hhmmss.sss (quoted from GPGGA sentence).
<latitude>	Latitude. Format: ddmm.mmmm N/S (quoted from GPGGA sentence). dd 00-89 (degree). mm.mmmm 00.0000-59.9999 (minute). N/S North latitude/ South latitude.
<longitude>	Longitude. Format: dddmm.mmmm E/W (quoted from GPGGA sentence). ddd 000-179 (degree).

	mm.mmmm 00.0000-59.9999 (minute). E/W East longitude/West longitude.
<hdop>	Horizontal precision, 0.5-99.9 (quoted from GPGGA sentence).
<altitude>	The altitude of the antenna away from the sea level (unit: m), accurate to one decimal place(quoted from GPGGA sentence).
<fix>	GNSS positioning mode (quoted from GNGSA/GPGSA). 2 2D positioning. 3 3D positioning.
<cog>	Ground heading based on true north. Format: ddd.mm (quoted from GPVT sentence). ddd 000-359 (degree). mm 00-59 (minute).
<spkm>	Speed over ground. Format: xxxx.x, unit: Km/h, accurate to one decimal place. (quoted from GPVTG sentence).
<spkn>	Speed over ground. Format: xxxx.x, unit: knots, accurate to one decimal place. (quoted from GPVTG sentence).
<date>	UTC date when positioning. Format: ddmmyy (quoted from GPRMC sentence).
<nsat>	Number of satellites, from 00 to 12 (the first 0 will also be transferred, quoted from GPGGA sentence).

ฟังก์ชัน EnableNMEA()

bool EnableNMEA() คือ ฟังก์ชันเปิดใช้งาน NMEA

ฟังก์ชัน DisableNMEA

bool DisableNMEA() คือ ฟังก์ชันปิดการใช้งาน NMEA

ฟังก์ชัน GetNMEA()

String GetNMEA(String nmea) คือ ฟังก์ชันดึงค่า NMEA ที่สนใจออกมาแสดง ผู้ใช้สามารถกำหนดชุดข้อมูล NMEA ที่เราสนใจลงในพารามิเตอร์ String nmea ฟังก์ชันตอบกลับมาเป็นชุดข้อมูลตามที่ต้องการออกมา ยกตัวอย่างเช่น ต้องการ NMEA เฉพาะส่วนของ GGA สามารถเรียกใช้ฟังก์ชัน GetNMEA(“GGA”)

การใช้งานไลบรารี TEE_UC20_Shield กับ MQTT

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include <AltSoftSerial.h>
#include "internet.h"
#include "uc_mqtt.h"
```

สร้างออบเจคต์ INTERNET , UCxMQTT

```
INTERNET net;
UCxMQTT mqtt;
```

ฟังก์ชัน ConnectMQTTServer()

bool ConnectMQTTServer(String web , String port) คือ ฟังก์ชันเชื่อมต่อไปยัง Server โดยมีพารามิเตอร์ดังต่อไปนี้

String web ใช้สำหรับกำหนด URL หรือ IP ของ MQTT Server

String port ใช้กำหนดหมายเลข Port ที่สื่อสารกับ MQTT Server

ฟังก์ชัน DisconnectMQTTServer()

bool DisconnectMQTTServer() คือ ฟังก์ชันสำหรับหยุดการเชื่อมต่อกับ MQTT Server

ฟังก์ชัน ConnectState()

bool ConnectState() คือ ฟังก์ชันสำหรับตรวจสอบการเชื่อมต่อกับ MQTT Server โดยหากสามารถเชื่อมต่อกับ MQTT Server สำเร็จจะ Return ค่าออกมาเป็น True

ฟังก์ชัน Connect()

unsigned char Connect(const char *id, const char *user, const char *pass); คือ ฟังก์ชันสำหรับเริ่มต้นการเชื่อมต่อกับ MQTT Broker โดยมีพารามิเตอร์ดังต่อไปนี้

const char *id	ใช้กำหนดชื่อที่ใช้ระบุตัวตนของ Node นั้นๆ
const char *user	ใช้กำหนด Username เพื่อ Login
const char *pass	ใช้กำหนด Password เพื่อ Login

ฟังก์ชัน Publish()

bool Publish(String topic,String payload, boolean retained); คือ ฟังก์ชันสำหรับ Publish ข้อมูลออกไป โดยมีพารามิเตอร์ดังต่อไปนี้

String topic	ใช้กำหนด topic ในการ publish
String payload	ข้อมูลที่ต้องการ publish
boolean retained	กำหนด Enable / Disable retrained

ฟังก์ชัน Subscribe()

bool Subscribe(const char* topic); คือ ฟังก์ชันสำหรับSubscribe Topic ที่ต้องการรับข้อมูล โดยมีพารามิเตอร์ดังต่อไปนี้

const char* topic	ใช้กำหนด topic ที่ต้องการ Subscribe

ฟังก์ชัน MqttLoop ()

void MqttLoop(); คือ ฟังก์ชันสำหรับรอรับข้อมูลจาก MQTT Broker และ ping กับ MQTT Broker เพื่อตรวจสอบสถานการ เชื่อมต่อ

ตัวแปร callback

callback เป็นตัวแปรสำหรับกำหนดแอดเดรสของฟังก์ชันที่มารองรับ เมื่อมีข้อมูลที่เรา Subscribe เอาไว้มีการ Publish ข้อมูลออกมา void functionname(String topic ,char *playload,unsigned char length) โดย functionname คือค่าที่กำหนดให้ตัวแปร callback และข้อมูลจะถูกส่งมายังตัวแปรต่างๆดังนี้

String topic	ชื่อ Topic ที่ได้รับมา
char *playload	ข้อมูลของ Topic ที่ได้รับมา
unsigned char length	ขนาดของข้อมูลที่ได้รับมา

สามารถดูตัวอย่างการใช้งาน 3G Shield / 3G Module กับ MQTT ได้จาก

File > Examples > TEE_UC20_Shield > MQTT

การใช้งานไลบรารี TEE_UC20_Shield กับ Audio

เรียกใช้งานไลบรารี TEE_UC20_Shield โดยเพิ่ม include ดังต่อไปนี้

```
#include "TEE_UC20.h"
#include "SoftwareSerial.h"
#include "call.h"
#include <AltSoftSerial.h>
#include "uc_audio.h"
#include "File.h"
#include "call.h"
```

สร้างออบเจคต์ FILE , UCxAUDIO,CALL

```
UC_FILE file;
UCxAUDIO audio;
CALL call;
```

ฟังก์ชัน StartRecord ()

bool StartRecord(bool ctrl,String pattern,String fn,unsigned char format,bool dlink) คือฟังก์ชันสำหรับบันทึกเสียงจาก Microphone หรือ บันทึกเสียงการสนทนา โดยมีพารามิเตอร์ดังต่อไปนี้

bool ctrl	กำหนด เริ่มต้นและหยุด บันทึกเสียง โดย true = เริ่มบันทึก false = หยุดบันทึก
String pattern	กำหนดแหล่งจัดเก็บ File เสียงที่บันทึก เช่น UFS , RAM
String fn	กำหนดชื่อ File ที่ต้องการบันทึก
unsigned char format	กำหนด Format ของ Sound ที่บันทึก 3 = AMR 13 = WAV_PCM13 14 = WAV_ALAW 15 = WAV_ULAW
bool dlink	กำหนดให้บันทึกเสียงจาก down-link หรือ up-link true = บันทึก down-link sound false = บันทึก up-link sound

ฟังก์ชัน StartWAVRecord ()

StartWAVRecord(String pattern,String fn,bool dlink) คือ ฟังก์ชันสำหรับบันทึกเสียงจาก Microphone หรือ บันทึกเสียงการสนทนา ให้เป็น File WAV โดยมีพารามิเตอร์ดังต่อไปนี้

String pattern กำหนดแหล่งจัดเก็บ File เสียงที่บันทึก เช่น UFS , RAM

String fn กำหนดชื่อ File ที่ต้องการบันทึก

bool dlink กำหนดให้บันทึกเสียงจาก down-link หรือ up-link

true= บันทึกdown-link sound

(บันทึกเสียงระหว่างสนทนา)

false = บันทึก up-link sound

(บันทึกเสียงจาก Microphone)

!!!อธิบายเพิ่มเติม คำสั่งบันทึกเสียง สามารถบันทึกจากแหล่งกำเนิดเสียงได้จาก 2 ทางคือ Microphone และเสียงระหว่างการสนทนา(โทรออก/รับสาย) โดยกำหนดจาก พารามิเตอร์ bool dlink ซึ่งหากกำหนดเป็น True ใช้สำหรับบันทึกเสียงขณะกำลังสนทนากับคู่สาย และ หากกำหนดเป็น False จะเป็นการบันทึกเสียงจาก Microphone โดยไม่ต้องมีการโทร เข้า/ออก (เหมือนเป็นเครื่องบันทึกเสียง)

ฟังก์ชัน StopRecord ()

StopRecord คือ ฟังก์ชันสำหรับสั่งให้ สิ้นสุดการบันทึกเสียง

ฟังก์ชัน PlayWAV ()

PlayWAV(String pattern,String fn, bool ulmute,bool dlmute) คือ ฟังก์ชันสำหรับสั่งให้เล่นเสียงจาก File WAV โดยมีพารามิเตอร์ดังต่อไปนี้

String pattern กำหนดแหล่งที่อยู่ File เสียงที่บันทึก เช่น UFS , RAM

String fn กำหนดชื่อ File ที่ต้องการเล่นเสียง

bool ulmute กำหนดให้ส่งเสียงออกไปทาง up-link หรือไม่

true= ส่งเสียงไปยัง up-link

false = ไม่ส่งเสียงไปยัง up-link

bool dlmute กำหนดให้ส่งเสียงออกไปทาง down-link หรือไม่

true= ส่งเสียงไปยัง down-link

false = ไม่ส่งเสียงไปยัง down-link

ฟังก์ชัน StopWAV ()

StopWAV() คือ ฟังก์ชันสำหรับสั่งให้หยุดเล่นเสียงจาก File WAV

ฟังก์ชัน PlayMP3 ()

PlayMP3(String pattern,String fn,bool repeat,unsigned char volumn) คือ ฟังก์ชันสำหรับสั่งให้เล่นเสียงจาก File mp3 โดยมีพารามิเตอร์ดังต่อไปนี้

String pattern	กำหนดแหล่งที่อยู่ File เสียงที่บันทึก เช่น UFS , RAM
String fn	กำหนดชื่อ File ที่ต้องการเล่นเสียง
bool repeat	กำนดป็น True เพื่อเล่น File ซ้ำ
unsigned char volumn	กำหนดระดับความดังของเสียงโดยกำหนดได้ 0 – 7 ระดับ

ฟังก์ชัน StopMP3 ()

StopMP3() คือ ฟังก์ชันสำหรับสั่งให้หยุดเล่นเสียงจาก File MP3

ฟังก์ชัน TextToSpeech()

TextToSpeech(String text) คือ ฟังก์ชันสำหรับสั่งให้โมดูลส่งเสียงออกมา ตามข้อความในรูปแบบ ASCII ที่กำหนดลงในพารามิเตอร์ String text

ฟังก์ชัน taskAudio()

taskAudio() คือ ฟังก์ชันสำหรับรอรับ Event ต่างๆที่ Module ตอบกลับมา

ฟังก์ชัน StatusPlay()

StatusPlay() คือ ฟังก์ชันสำหรับตรวจสอบสถานะว่าขณะนี้กำลังเล่น File เสียงอยู่หรือไม่ โดยถ้าหากกำลังเล่น File เสียงอยู่จะ return ค่าออกมาเป็น True