

User Guide

MTDuino-SFM2CWW001

M0 and SigFox connectivity module





Index

1. General Description	2
2. Technical Specifications	2
Power	2
Input and Output	3
Communication	4
Pin Assignment	6
3. Hardware Introduction	7
a. sch	7
b. top	9
c. bottom	9
d. gerber	10
4. Software Introduction	12
4.3 Installing Arduino IDE	13
4.3.1 Download the Arduino Software (IDE)	13
4.3.2 Install Arduino IDE	14
4.3.3 Install Arduino M0+ driver	16
4.3.4 Check Hardware	18
4.3.5 Upload sketch	19
4.3.6 Confirm Upload	24
4.4 Sigfox Backend - Active	26
4.5 Sigfox Backend – Data Receive	28
5. Example Code	30
5.1 Sigfox Terminal Send AT	30
5.2 Sigfox Send	32
NOTE	33

Important Notice:

And reserves the right to alter the hardware, software, and/or specifications detailed here at any time without notice.



1. General Description

The MTDuino-SigFox is powered by Atmel's SAMD21 MCU, featuring a 32-bit ARM Cortex® M0 core. With the addition of the M0 board, the Arduino family becomes larger with a new member providing increased performance.

It is based on the Microchip SAMD21 and a UPLYNX-M-RCZx SigFox module.

2. Technical Specifications

Microcontroller	
Board Power Supply (USB/VIN)	ATSAMD21G18, ARM Cortex-M0+, 48pins LQFP
Supported Batteries(*)	5V
Circuit Operating Voltage	Li-polymer battery
Digital I/O Pins	3.3V
PWM Output	20
Analog Input Pins	12
Analog Output Pins	6 (ADC 8/10/12 bit)
DC Current per I/O Pin	1 (DAC 10 bit)
Flash Memory	7 mA
SRAM	256 KB
Clock Speed	32 KB
Antenna power	32.768 kHz (RTC), 48 MHz
Carrier frequency	0.3dB @ 868M, 1.3 dB @ 915M
Working region	868 MHz, 915MHz
Lenght	World Wide(WW)
Width	67 mm
	25 mm

Power

The MTDuino-SigFox can be powered via the micro USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from Lithium-battery. The MTDuino-SigFox will automatically detect which power sources are available and choose which one to use according to the following priority:

- External power: Li-polymer battery
- Target USB

The MTDuino-SigFox is a Constant-Voltage (CV) and Constant-Current (CC) type charging IC for linear charging of single-cell Li-ion batteries and Li-polymer batteries.



Input and Output

Each of the 14 digital i/o pins on the MTDuino-SigFox can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 3.3 volts. 7mA as maximum DC current for I/O pins and an internal pull-up resistor (disconnected by default) of 20-60 kOhms. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data using the ATSAMD21G18 hardware serial capability. Note that on the M0, the SerialUSB class refers to USB (CDC) communication; for serial on pins 0 and 1, use the Serial5 class.
- TWI: SDA and SCL. Support TWI communication using the Wire library.
- PWM: Pins 2 to 13 Provide 8-bit PWM output with the analogWrite() function. The resolution of the PWM can be changed with the analogWriteResolution() function. Note1 The pins 4 and 10 can not be used simultaneously as PWM. Note2 The pins 5 and 12 can not be used simultaneously as PWM.
- Analog Inputs: A0-A5. The M0 has 6 analog inputs, labeled A0 through A5. Pins A0-A5 appear in the same locations as on the Uno; Each analog input provides 12 bits of resolution (i.e. 4096 different values). By default the analog inputs measure from ground to 3.3 volts, though it is possible to change the upper end of their range using the AREF pin and the analogReference() function.
- DAC: pin A0 provides true analog outputs with 10-bits resolution (1023 levels) with the analogWrite() function. This pin can be used to create an audio output using the Audio library.
- Reset: Bring this line LOW to reset the microcontroller. This is typically used to add a reset button when shields are used that block the one already present on the board.

Communication

The MTDuino-SigFox has a number of facilities for communicating with a computer, with another Arduino or other microcontrollers, and with different devices like phones, tablets, cameras and so on. The SAMD21 provides one hardware UART and three hardware USARTs for 3.3V serial communication. The Arduino software includes a serial monitor allowing simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATSAMD21G18 chip and USB connection to the computer (but not for serial communication on pins 0 and 1). The Native USB port is connected to the SAMD21. It allows for serial (CDC) communication over USB. This provides a serial connection to the Serial Monitor or other applications on your computer. The SAMD21 also supports TWI and SPI communication. The Arduino software includes a Wire library to simplify use of the TWI bus. For SPI communication, you can use the SPI library on the board.



Programming

The MTduino-SigFox can be programmed with the Arduino software. Uploading sketches to the SAMD21 is different from how it works with the AVR microcontrollers found in other Arduino boards: the flash memory needs to be erased before being re-programmed. Upload operation is managed by a dedicated ROM area on the SAMD21. USB port: To use this port, select "Arduino M0 (Native USB Port)" as your board in the Arduino IDE. The Native USB port is connected directly to the SAMD21. Connect the M0 Native USB port (the one closest to the reset button) to your computer.

Antenna

Manufacturer: Molex

Part Number: 105262-0002

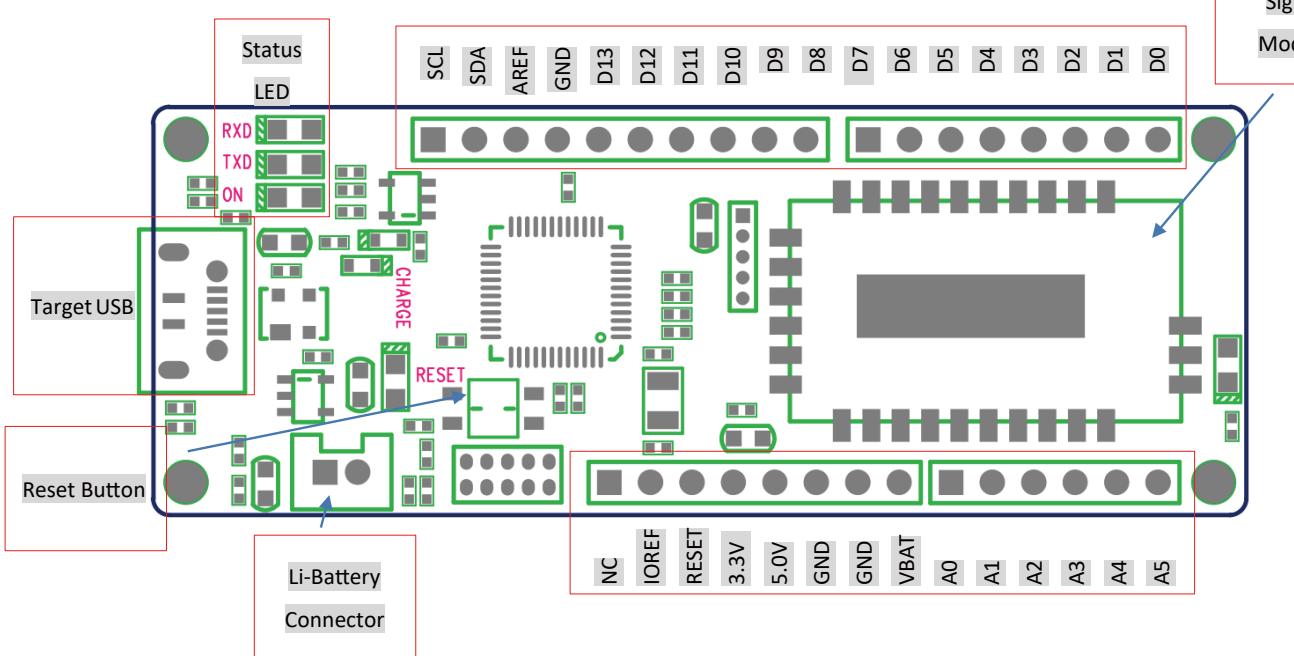
Description: Antennas ISM 868/915 MHz ANTENNA 150MM



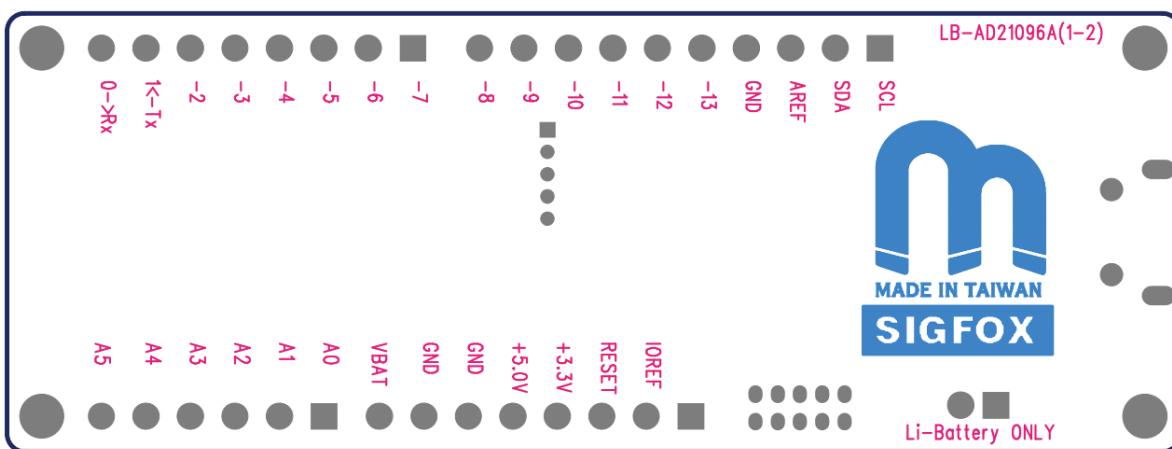
MTDuino-SFM2CWW001



TOP



BOTTOM





Pin Assignment

*	+	-----	-----	-----	-----	-----	-----
*	+	-----	-----	-----	-----	-----	-----
*		-----	-----	-----	-----	-----	-----
*	+	-----	-----	-----	-----	-----	-----
*		Digital Low					
*	+	-----	-----	-----	-----	-----	-----
*	0	0 -> RX	PA11	EIC/EXTINT[11] ADC/AIN[19]	PTC/X[3]	*SERCOM0/PAD[3]	
*	1	1 <- TX	PA10	EIC/EXTINT[10] ADC/AIN[18]	PTC/X[2]	*SERCOM0/PAD[2]	
*	2	2	PA14	EIC/EXTINT[14]		SERCOM2/PAD[2]	
*	3	~3	PA09	EIC/EXTINT[9] ADC/AIN[17]	PTC/X[1]	SERCOM0/PAD[1]	
*	4	~4	PA08	EIC/NMI ADC/AIN[16]	PTC/X[0]	SERCOM0/PAD[0]	
*	5	~5	PA15	EIC/EXTINT[15]		SERCOM2/PAD[3]	
*	6	~6	PA20	EIC/EXTINT[4]	PTC/X[8]	SERCOM5/PAD[2]	
*	7	7	PA21	EIC/EXTINT[5]	PTC/X[9]	SERCOM5/PAD[3]	
*	+	-----	-----	-----	-----	-----	-----
*		Digital High					
*	+	-----	-----	-----	-----	-----	-----
*	8	~8	PA06	EIC/EXTINT[6] ADC/AIN[6] AC/AIN[2]	PTC/Y[4]	SERCOM0/PAD[2]	
*	9	~9	PA07	EIC/EXTINT[7] ADC/AIN[7] AC/AIN[3]	PTC/Y[5]	SERCOM0/PAD[3]	
*	10	~10	PA18	EIC/EXTINT[2]	PTC/X[6]	+SERCOM1/PAD[2]	
*	11	~11	PA16	EIC/EXTINT[0]	PTC/X[4]	+SERCOM1/PAD[0]	
*	12	~12	PA19	EIC/EXTINT[3]	PTC/X[7]	+SERCOM1/PAD[3]	
*	13	~13	PA17 LED	EIC/EXTINT[1]	PTC/X[5]	+SERCOM1/PAD[1]	
*	+	-----	-----	-----	-----	-----	-----
*		Analog Connector					
*	+	-----	-----	-----	-----	-----	-----
*	14	A0	PA02 A0	EIC/EXTINT[2] *ADC/AIN[0]	DAC/VOUT	PTC/Y[0]	
*	15	A1	PB08 A1	EIC/EXTINT[8] *ADC/AIN[2]		PTC/Y[14]	SERCOM4/PAD[0]
*	16	A2	PB09 A2	EIC/EXTINT[9] *ADC/AIN[3]		PTC/Y[15]	SERCOM4/PAD[1]
*	17	A3	PA04 A3	EIC/EXTINT[4] *ADC/AIN[4]	AC/AIN[0]	PTC/Y[2]	SERCOM0/PAD[0]
*	18	A4	PA05 A4	EIC/EXTINT[5] *ADC/AIN[5]	AC/AIN[1]	PTC/Y[5]	SERCOM0/PAD[1]
*	19	A5	PB02 A5	EIC/EXTINT[2] *ADC/AIN[10]		PTC/Y[8]	SERCOM5/PAD[0]
*	+	-----	-----	-----	-----	-----	-----
*		Wire					
*	+	-----	-----	-----	-----	-----	-----
*	20	SDA	PA22 SDA	EIC/EXTINT[6]		PTC/X[10]	*SERCOM3/PAD[0]
*	21	SCL	PA23 SCL	EIC/EXTINT[7]		PTC/X[11]	*SERCOM3/PAD[1]

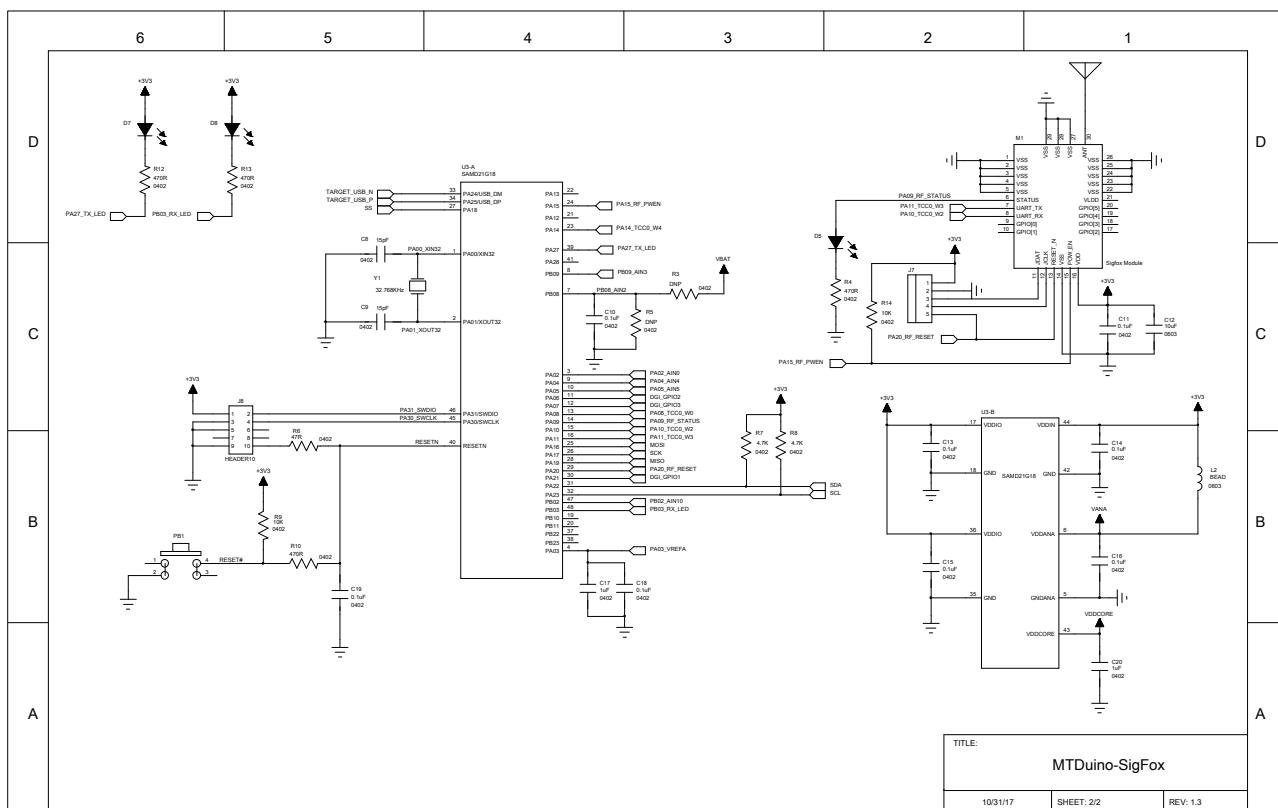
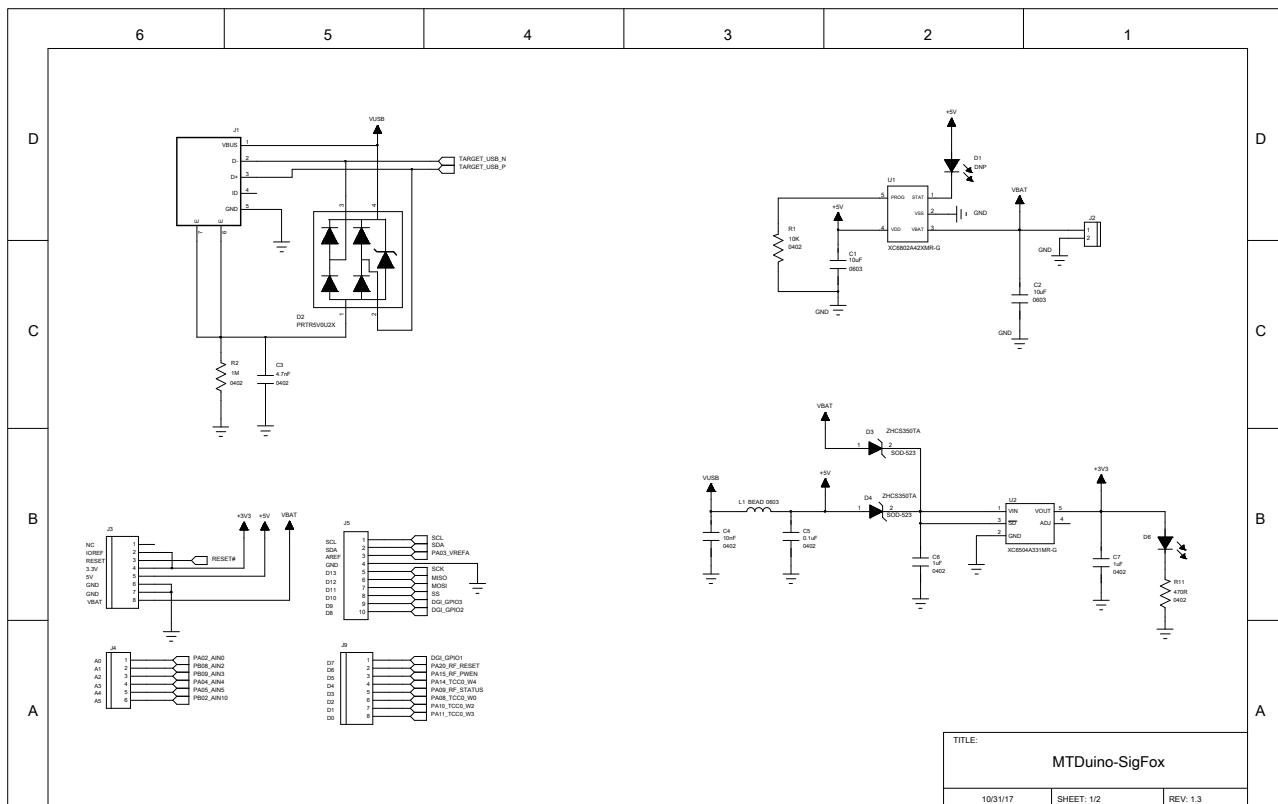
Reference

- . <https://store.arduino.cc/usa/arduino-m0>
- . <http://www.mouser.tw/ProductDetail/Molex/105262-0002/?qs=lQwVXVmkiFIBK%2fDeZK%252bySg>



3. Hardware Introduction

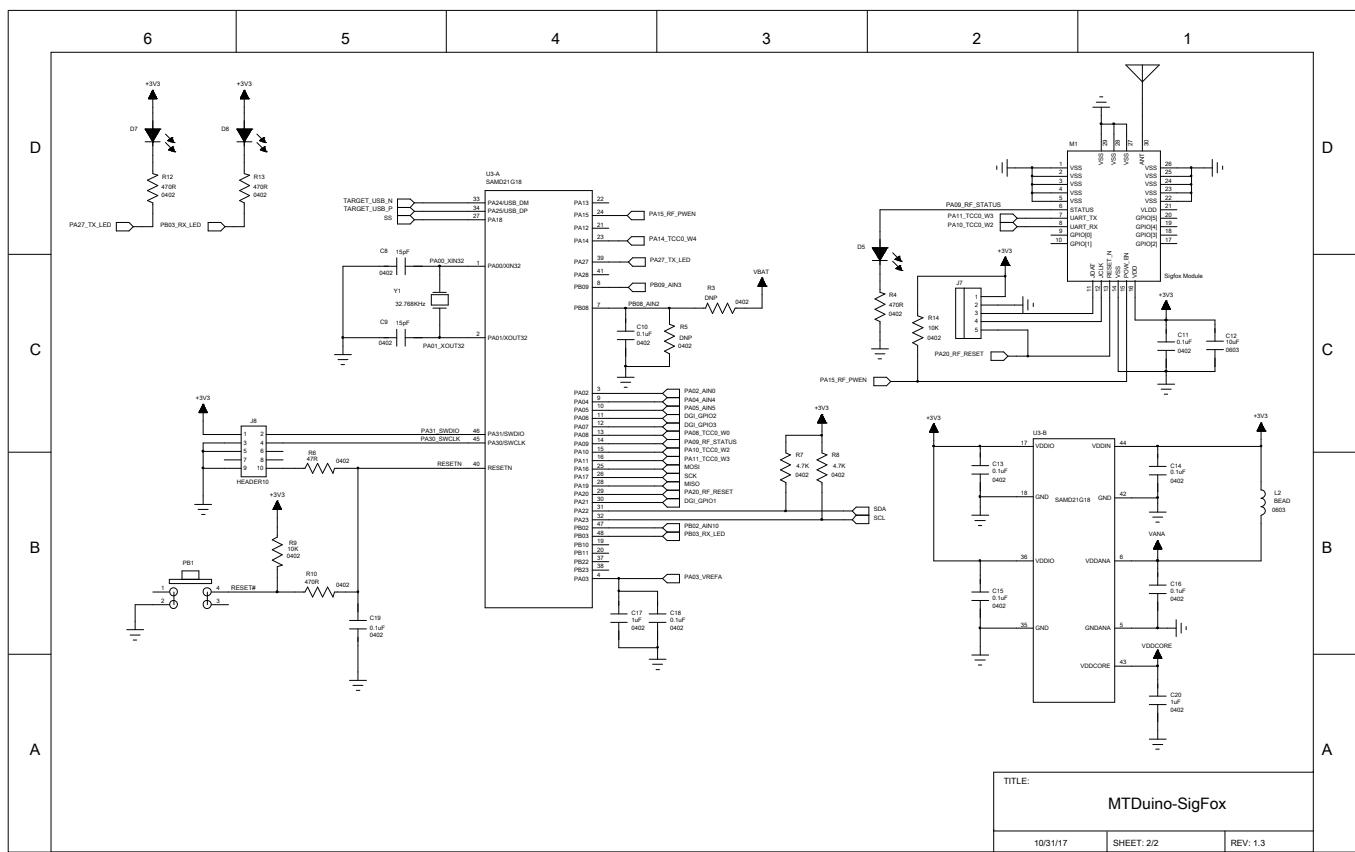
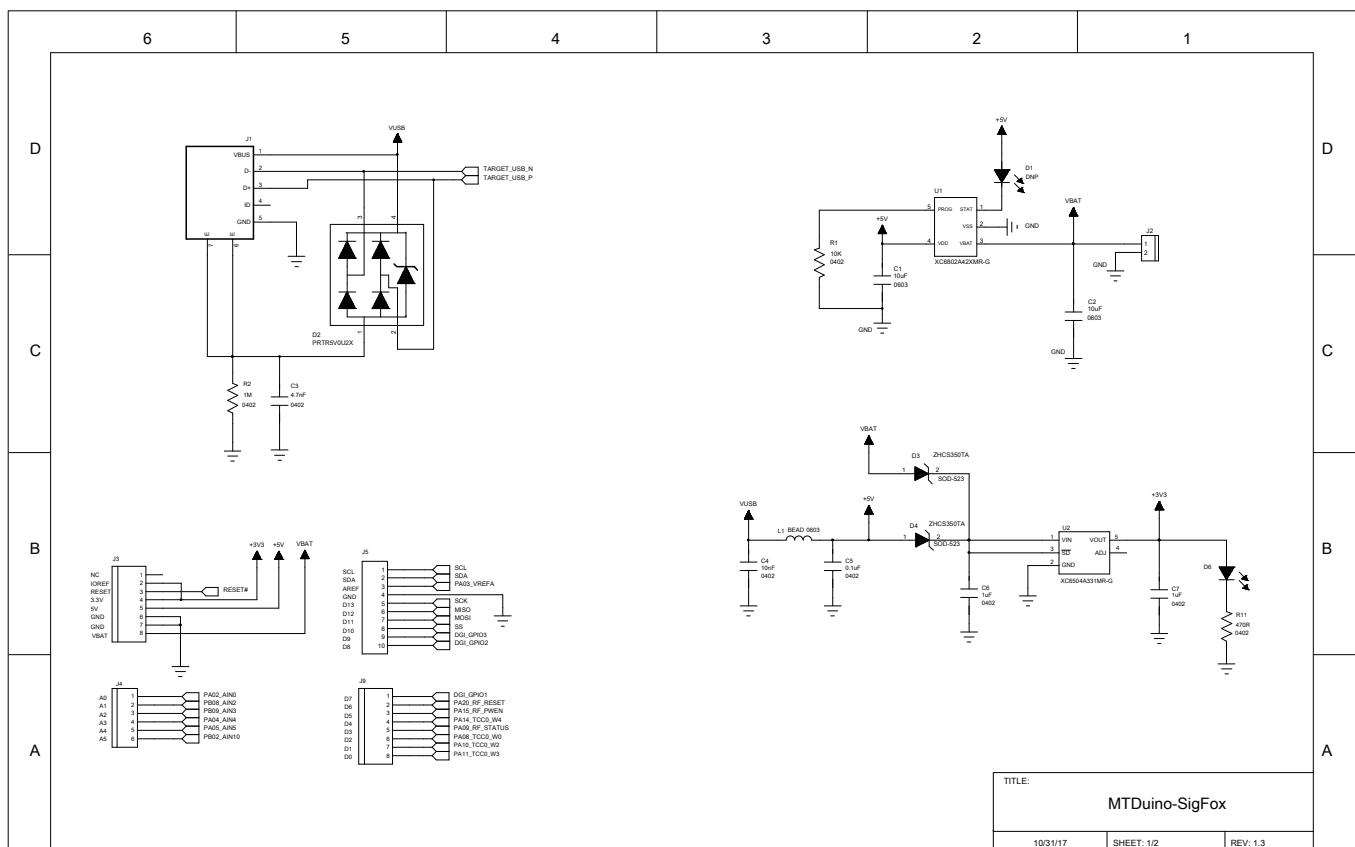
a. Sch



MTDuino-SFM2CWW001

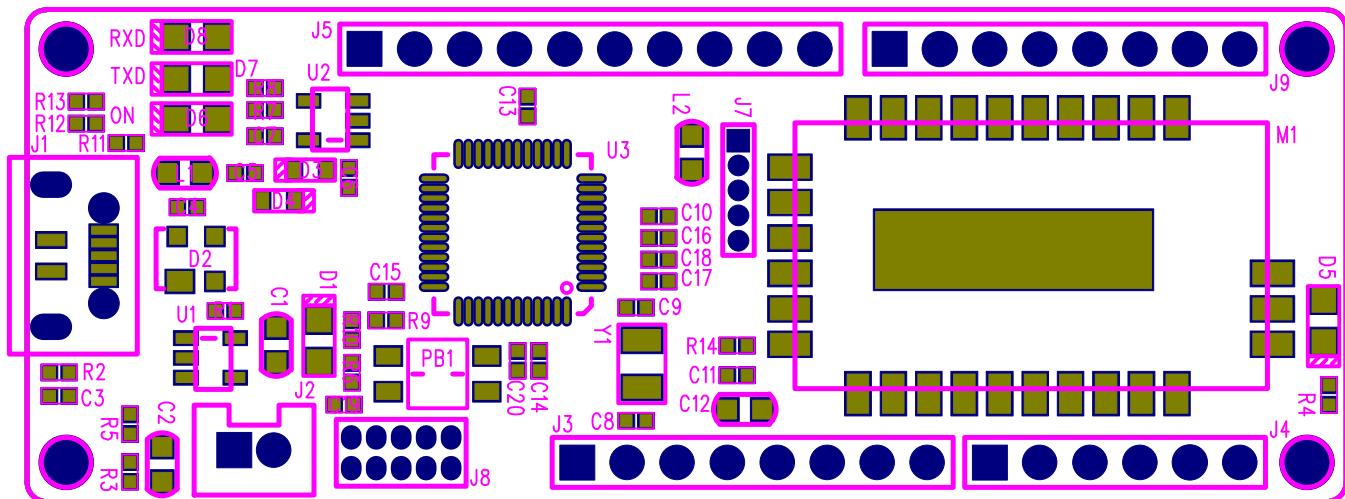


c, Sch

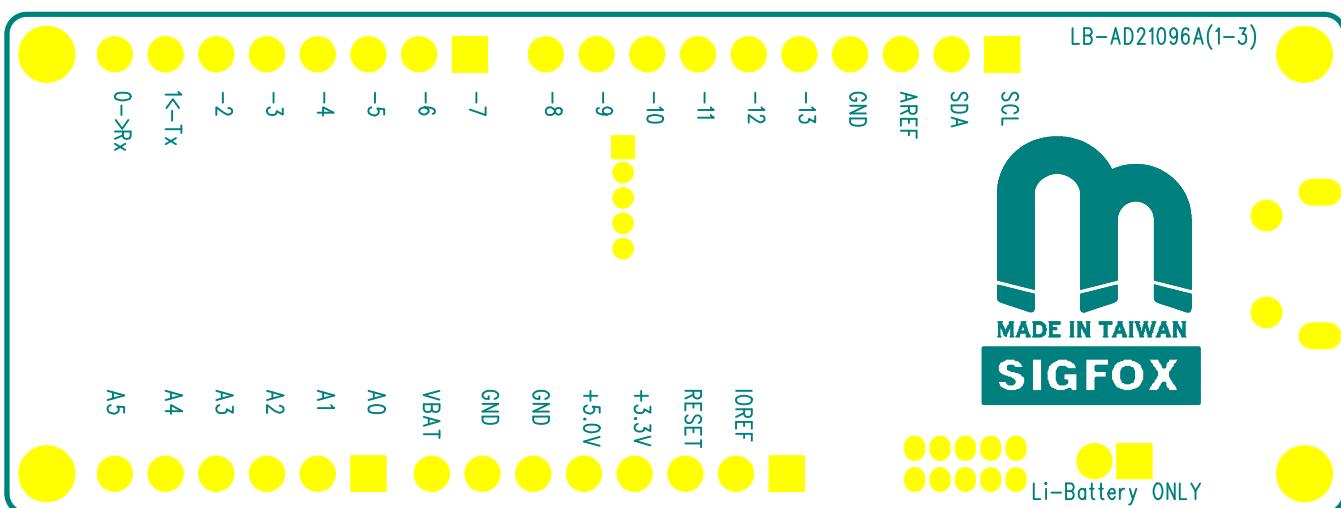




b. TOP



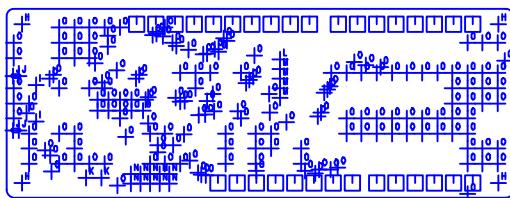
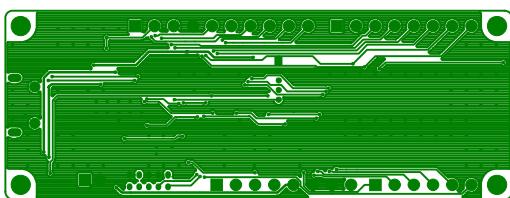
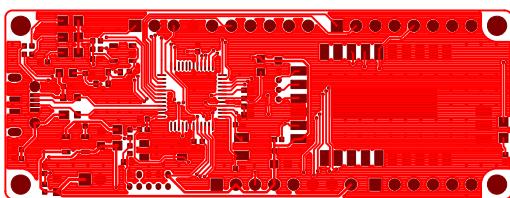
c. Bottom



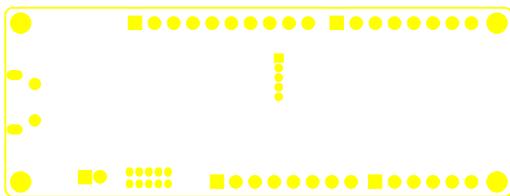
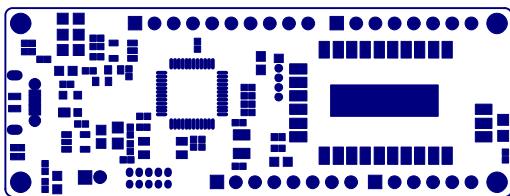
MTDuino-SFM2CWW001



d. Gerber



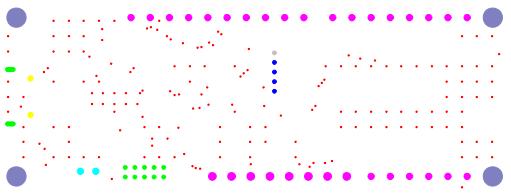
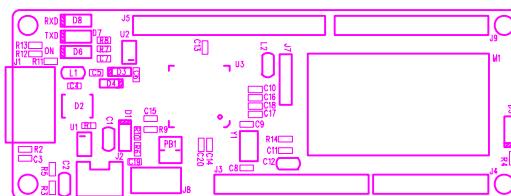
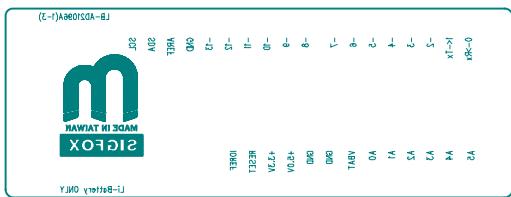
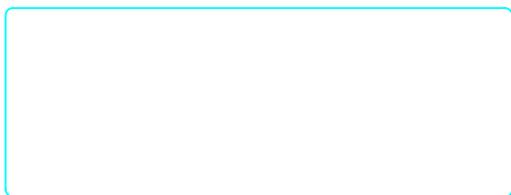
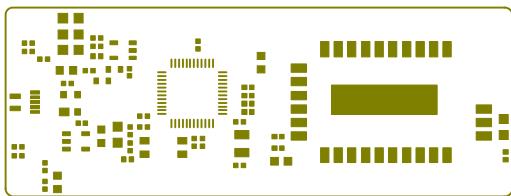
SIZE	QTY	SYM	PLATED	TOL
0.9398	32	□	YES	??x
2.54	4	+H	YES	+/-0.0
0.7	2	+I	YES	+/-0.0
0.5 x 1.3	2	+J	YES	+/-0.0
0.8128	2	+K	YES	+/-0.0
0.6	1	+L	YES	+/-0.0
0.55	4	+M	YES	+/-0.0
0.5	10	+N	YES	+/-0.0
0.2032	179	+O	YES	+/-0.0



MTDuino-SFM2CWW001



d. Gerber





4. Introduction



4.1 Overview

The MTDuino is an easy to use powerful single board. The MTDuino is open-source, which means hardware is reasonably priced and development software is free.

This guide is for students who are confronting the MTDuino for the first time.

The Arduino project was started in Italy to develop low cost hardware for interaction design. An overview is on the Wikipedia entry for Arduino. The Arduino home page is

<http://www.arduino.cc/>

4.2 What You Need

1. MTDuino board
2. Micro USB cable
3. Host PC running the Arduino development environment. For Windows, Mac and Linux



4.3 Installing Arduino IDE

Getting started of the Arduino web site, <http://arduino.cc/en/Guide/HomePage>

4.3.1 Download the Arduino Software (IDE)

Get the latest version from the download page. You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually.

Download the Arduino IDE



The screenshot shows the Arduino Software (IDE) download page. On the left, there's a large teal circle with a white infinity symbol containing a minus sign (-) on the left and a plus sign (+) on the right. To its right, the text "ARDUINO 1.8.5" is displayed. Below it, a paragraph explains what the Arduino Software (IDE) is and how it runs on Windows, Mac OS X, and Linux. It also mentions that it can be used with any Arduino board and refers to the "Getting Started" page for installation instructions. On the right side, there are download links for different platforms: "Windows Installer" (highlighted with an orange border), "Windows ZIP file for non admin install", "Windows app" (Requires Win 8.1 or 10, with a Get button), "Mac OS X" (10.7 Lion or newer), "Linux 32 bits", "Linux 64 bits", "Linux ARM", and links for "Release Notes", "Source Code", and "Checksums (sha512)".

Contribute to the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)

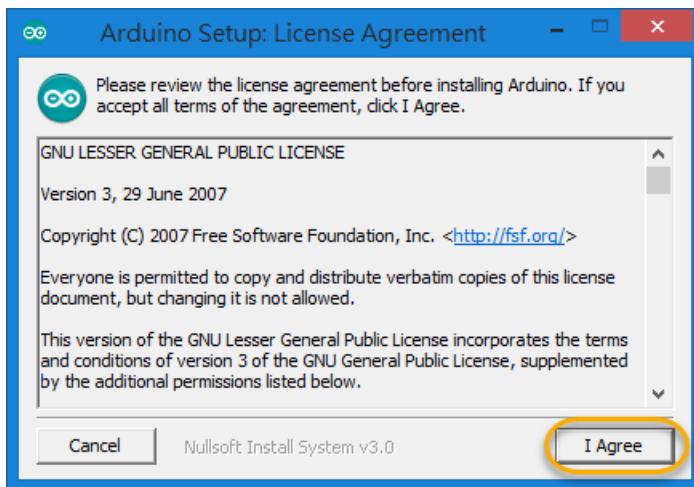


The screenshot shows the Arduino contribution page. At the top, there's a cartoon illustration of three simple electronic components: a red square (representing a microcontroller), a grey rectangle (representing a memory chip), and a blue circle (representing a capacitor). To the right of the illustration, text reads: "SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED 21,622,731 TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!" Below this, there are six circular buttons with contribution amounts: "\$3", "\$5", "\$10", "\$25", "\$50", and "OTHER". At the bottom, there are two buttons: "JUST DOWNLOAD" (in a yellow-bordered box) and "CONTRIBUTE & DOWNLOAD" (in a dark green box).

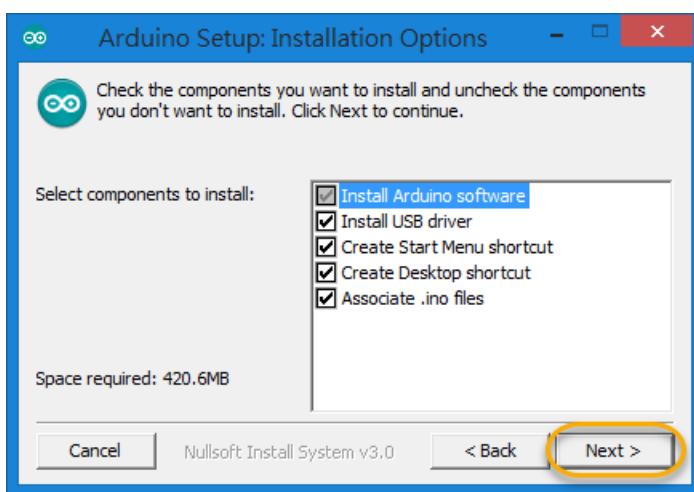


4.3.2 Install Arduino IDE

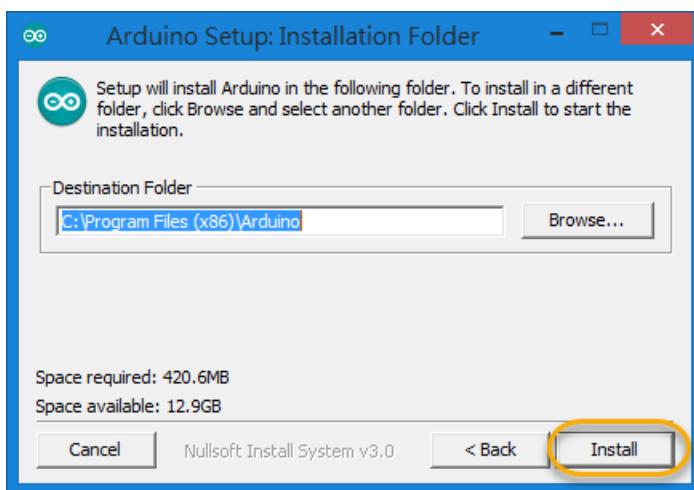
Step.1 Double click “.exe” and Click “I Agree”



Step.2 Click “Next”

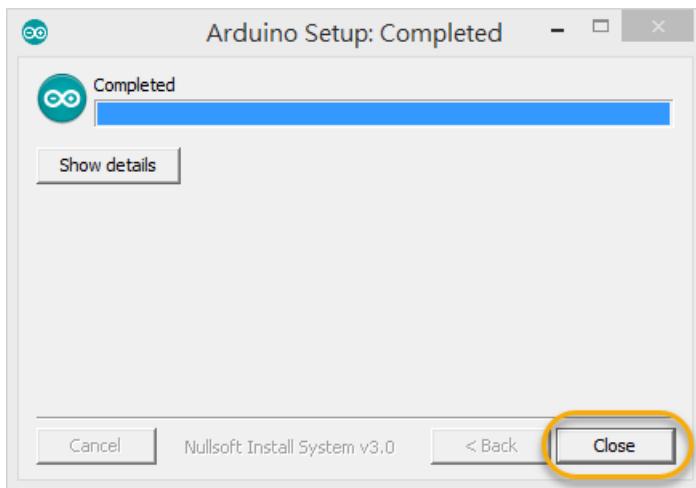


Step.3 Click “Install”

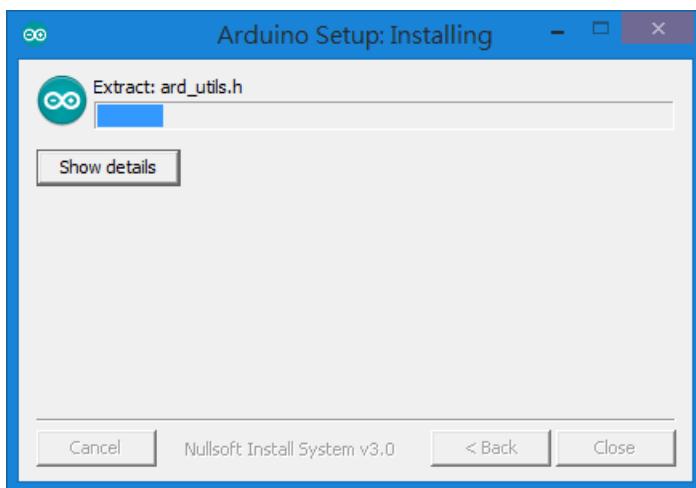




Wait for completed



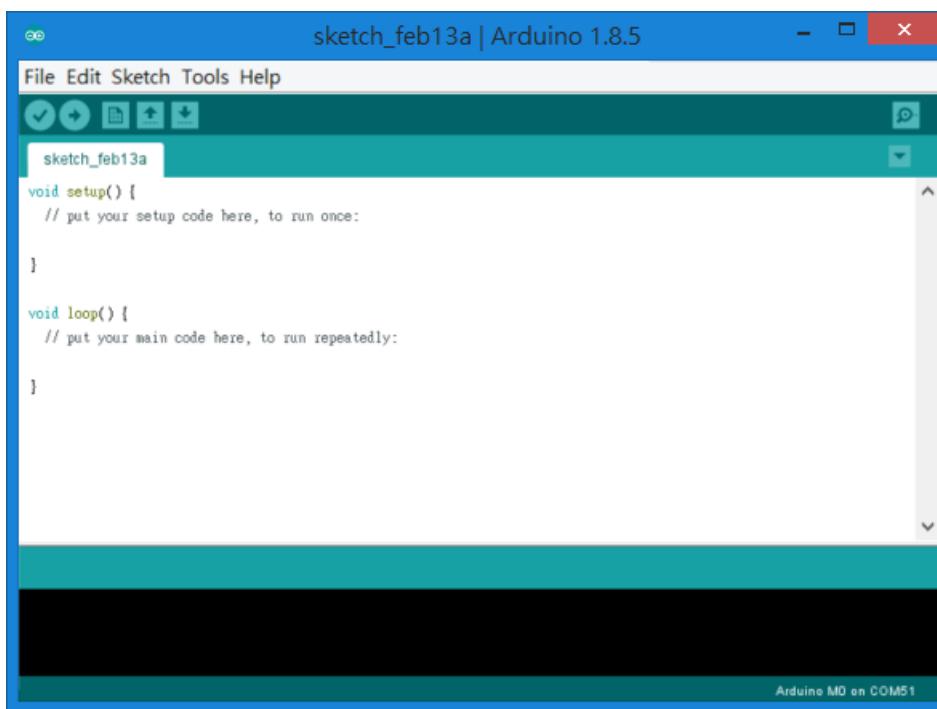
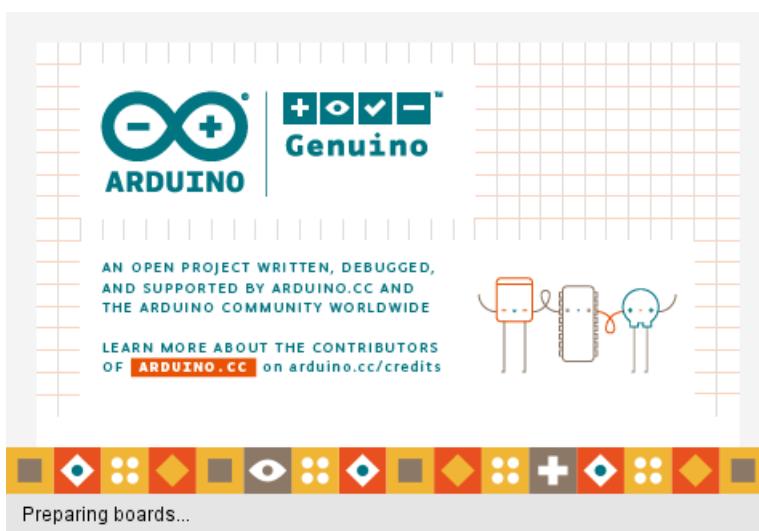
Step.4 Completed and click “Close”





4.3.3 Install Arduino M0+ driver

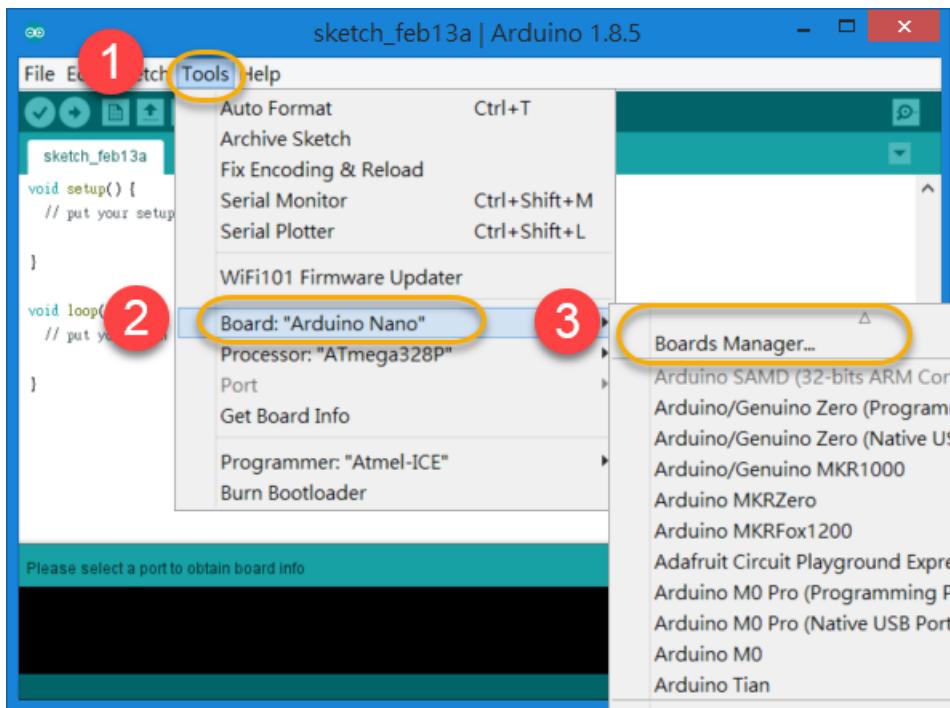
Step.1 Double click "Arduino IDE" icon



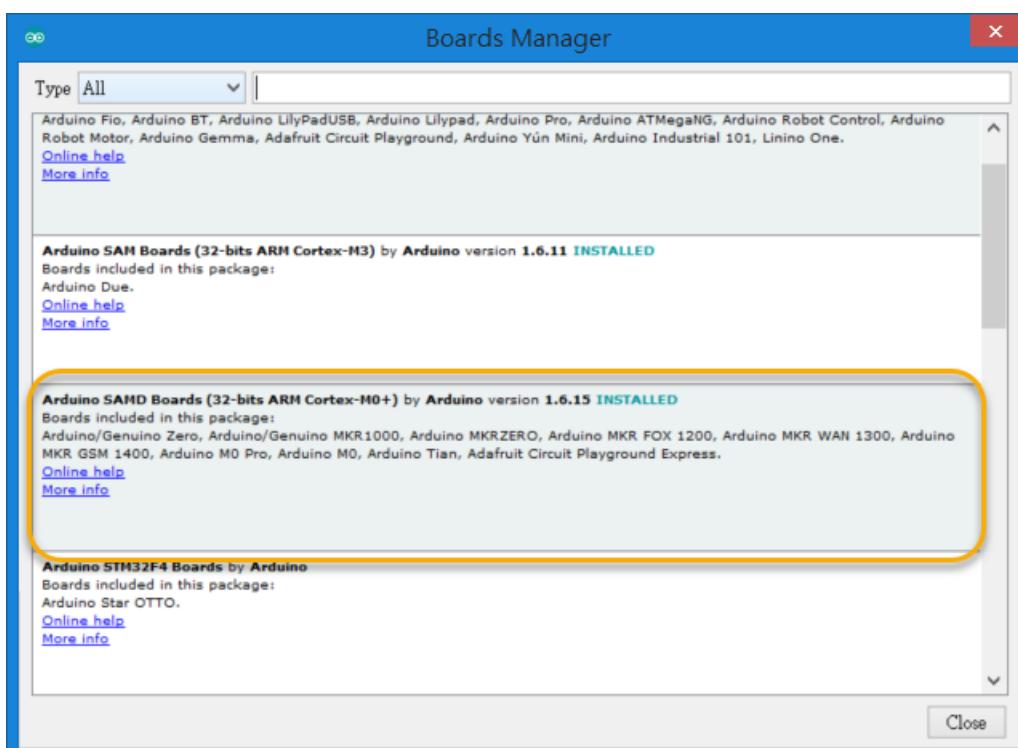
MTDuino-SFM2CW001



Step.2 Select Tools→Board: “xxxxxxxx” → Boards Manager



Step.3 Please install “Arduino SAMD Board (32bits ARM Cortex-M0+)”



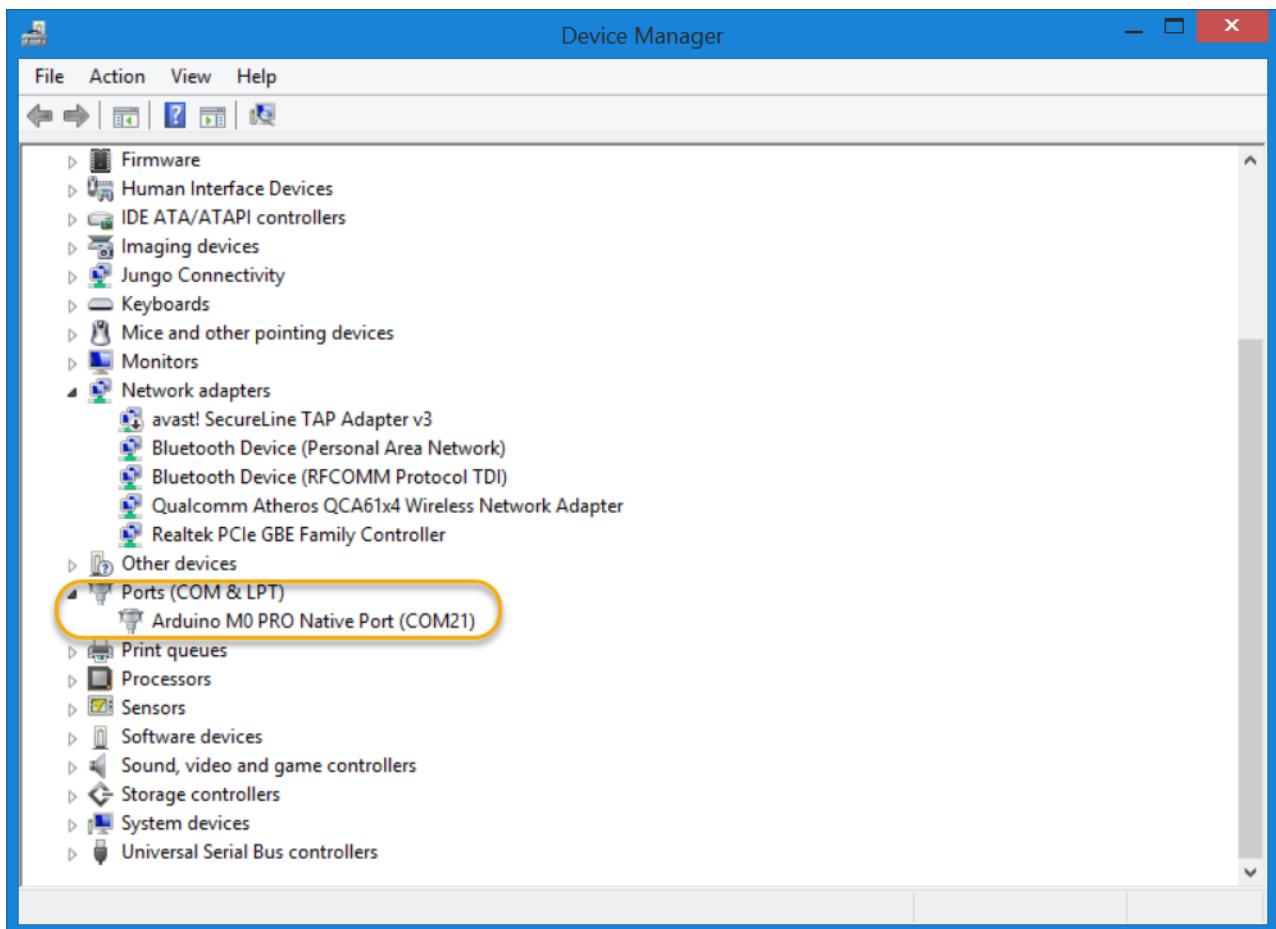


4.3.4 Check Hardware

Step.1 Plug in MTduino device.

Step.2 Open Device management and confirm ComPort

If correct, it will show as figure.





4.3.5 Upload sketch

Step.1 Open Sample code “SIGFOX_Terminal_SendAT.ino”

```

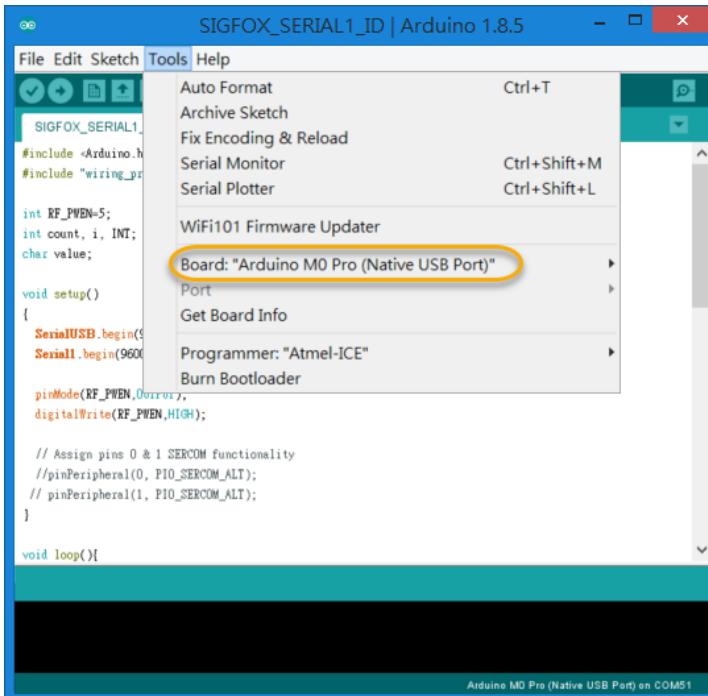
SIGFOX_Terminal_SendAT | Arduino 1.8.5
File Edit Sketch Tools Help
SIGFOX_Terminal_SendAT
#include <Arduino.h>
#include "wiring_private.h"
int LED_POW=3;           // required before wiring_private.h
int RF_PWN=5;            // pinPeripheral() function
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);      //Terminal的速率
    Serial1.begin(9600);        //sigfox的速率
    pinMode(LED_POW,OUTPUT);
    pinMode(RF_PWN,OUTPUT);
    digitalWrite(LED_POW,HIGH);
    digitalWrite(RF_PWN,HIGH);  //因應第一版的sigfox的power_en沒接電,所以將其腳位接出來的pin拉高
}

void loop()
{
    INT = SerialUSB.available(); //Serial.available()檢查是否Serial port有接收到數值,若有接收到了
}

```

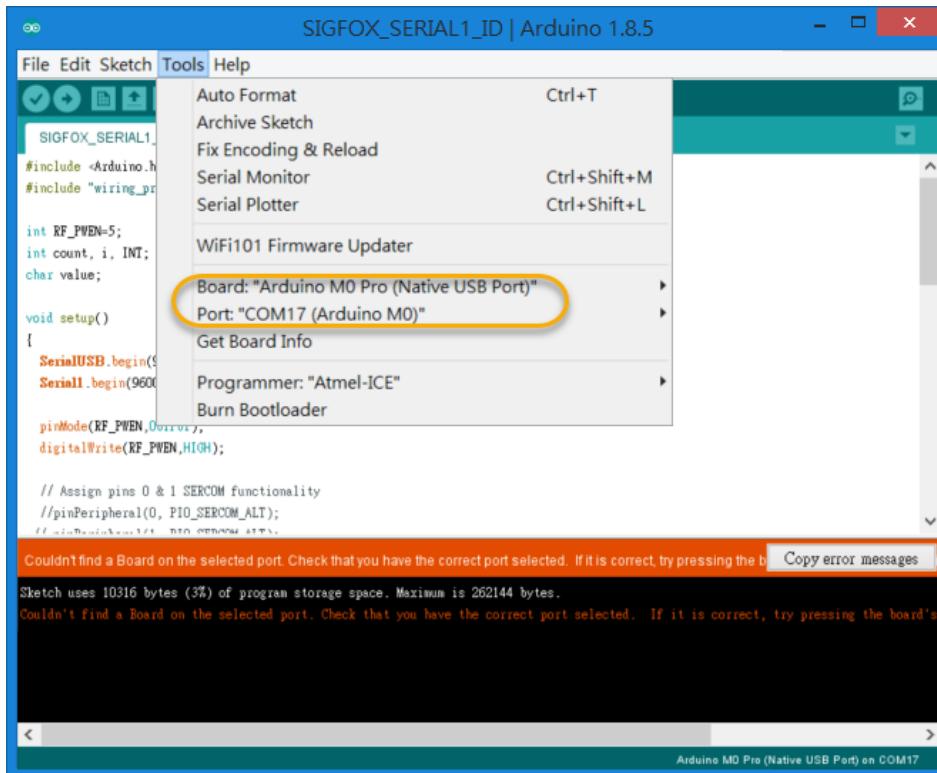
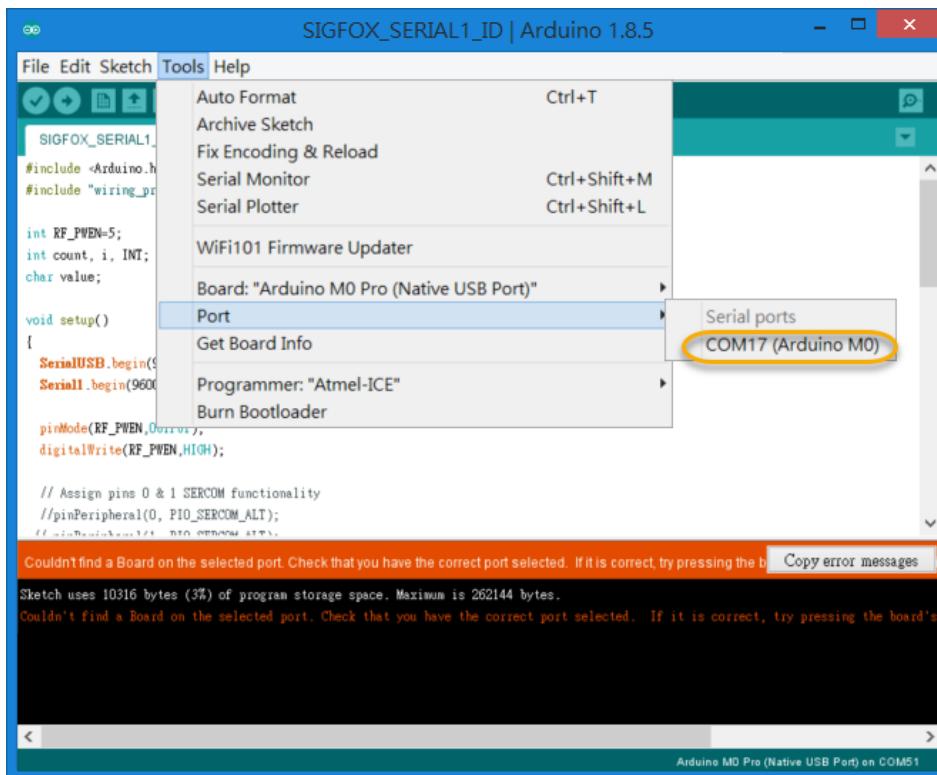
Step.2 Select Board: “Arduino M0 Pro(Native USB Port)“



MTDuino-SFM2CW001



Step.3 Select Port



MTDuino-SFM2CW001



Step.4 Click "Verify"

```
SIGFOX_SERIAL1_ID | Arduino 1.8.5
File Edit Sketch Tools Help
Verify
SIGFOX_SERIAL1_ID
#include <Arduino.h> // required before wiring_private.h
#include "wiring_private.h" // pinPeripheral() function

int RF_PWN=5; //控制端
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);
    Serial1.begin(9600);

    pinMode(RF_PWN,OUTPUT);
    digitalWrite(RF_PWN,HIGH);

    // Assign pins 0 & 1 SERCOM functionality
    //pinPeripheral(0, PIO_SERCOM_ALT);
    // pinPeripheral(1, PIO_SERCOM_ALT);
}

void loop(){}
```

Arduino M0 Pro (Native USB Port) on COM51

```
SIGFOX_SERIAL1_ID | Arduino 1.8.5
File Edit Sketch Tools Help
Verify
SIGFOX_SERIAL1_ID
#include <Arduino.h> // required before wiring_private.h
#include "wiring_private.h" // pinPeripheral() function

int RF_PWN=5; //控制端
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);
    Serial1.begin(9600);

    pinMode(RF_PWN,OUTPUT);
    digitalWrite(RF_PWN,HIGH);

    // Assign pins 0 & 1 SERCOM functionality
    //pinPeripheral(0, PIO_SERCOM_ALT);
    // pinPeripheral(1, PIO_SERCOM_ALT);
}

void loop(){
    //SerialUSB.print("AT$ID?\r"); // AT 指令
    //Serial1.print("AT$ID?\r");
    //delay(100);

    INT = SerialUSB.available();
```

Compiling sketch...

Arduino M0 Pro (Native USB Port) on COM51

MTDuino-SFM2CW001



Mighty Net



The screenshot shows the Arduino IDE interface with the title bar "SIGFOX_SERIAL1_ID | Arduino 1.8.5". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for Open, Save, Print, and Upload. The code editor contains the following sketch:

```
#include <Arduino.h> // required before wiring_private.h
#include "wiring_private.h" // pinPeripheral() function

int RF_Pwen=5; //控制端
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);
    Serial1.begin(9600);

    pinMode(RF_Pwen,OUTPUT);
    digitalWrite(RF_Pwen,HIGH);

    // Assign pins 0 & 1 SERCOM functionality
    //pinPeripheral(0, PIO_SERCOM_ALT);
    //pinPeripheral(1, PIO_SERCOM_ALT);
}

Done compiling.
```

The serial monitor window displays the compilation message: "Done compiling." followed by "Archiving built core (caching) in: C:\Users\user\AppData\Local\Temp\arduino_cache_864371\core\core_arduino. Sketch uses 10316 bytes (3%) of program storage space. Maximum is 262144 bytes." A yellow box highlights this message.

At the bottom, it says "Arduino M0 Pro (Native USB Port) on COM51".

Step.5 Click "Upload"

The screenshot shows the Arduino IDE interface with the title bar "SIGFOX_SERIAL1_ID | Arduino 1.8.5". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for Open, Save, Print, and Upload, with the Upload icon circled in yellow. The code editor contains the same sketch as the previous screenshot. The serial monitor window displays the compilation message: "Done compiling." followed by "Archiving built core (caching) in: C:\Users\user\AppData\Local\Temp\arduino_cache_864371\core\core_arduino. Sketch uses 10316 bytes (3%) of program storage space. Maximum is 262144 bytes." A yellow box highlights this message.

At the bottom, it says "Arduino M0 Pro (Native USB Port) on COM51".

MTDuino-SFM2CWW001



The screenshot shows the Arduino IDE interface with the sketch titled "SIGFOX_SERIAL1_ID". The code includes #include <Arduino.h>, #include "wiring_private.h", and defines RF_PVEN=5 as a control pin. The setup() function initializes SerialUSB and Serial1 at 9600 baud and sets RF_PVEN to HIGH. The serial monitor window shows "Compiling sketch...".

```
#include <Arduino.h> // required before wiring_private.h
#include "wiring_private.h" // pinPeripheral() function

int RF_PVEN=5; //控制端
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);
    Serial1.begin(9600);

    pinMode(RF_PVEN,OUTPUT);
    digitalWrite(RF_PVEN,HIGH);

    // Assign pins 0 & 1 SERCOM functionality
    //pinPeripheral(0, PIO_SERCOM_ALT);
    //pinPeripheral(1, PIO_SERCOM_ALT);
}

Compiling sketch...
```

The screenshot shows the Arduino IDE interface with the sketch titled "SIGFOX_SERIAL1_ID". The code is identical to the one in the previous screenshot. The serial monitor window shows "Done uploading." and "Sketch uses 10316 bytes (3%) of program storage space. Maximum is 262144 bytes." The status bar indicates "Arduino M0 Pro (Native USB Port) on COM19".

```
#include <Arduino.h> // required before wiring_private.h
#include "wiring_private.h" // pinPeripheral() function

int RF_PVEN=5; //控制端
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);
    Serial1.begin(9600);

    pinMode(RF_PVEN,OUTPUT);
    digitalWrite(RF_PVEN,HIGH);

    // Assign pins 0 & 1 SERCOM functionality
    //pinPeripheral(0, PIO_SERCOM_ALT);
    //pinPeripheral(1, PIO_SERCOM_ALT);

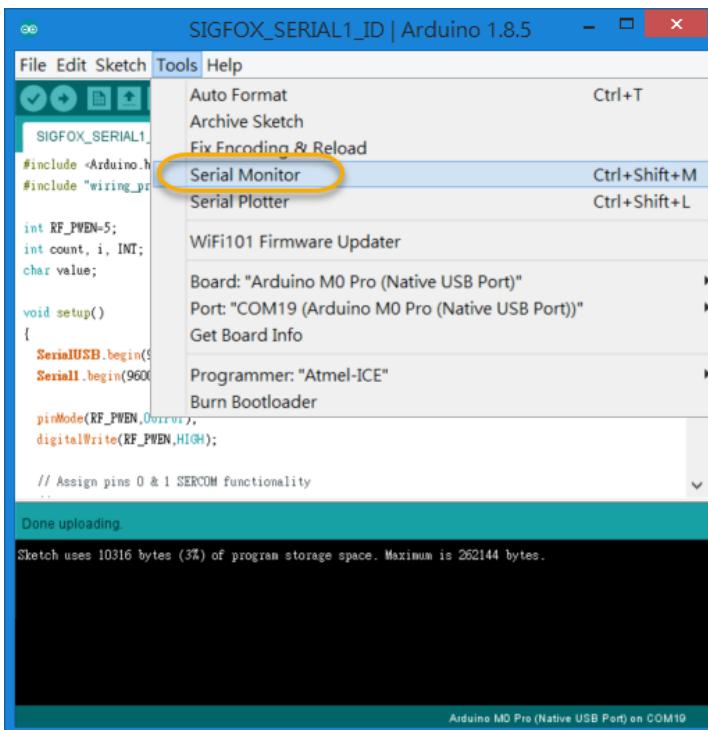
Done uploading.

Sketch uses 10316 bytes (3%) of program storage space. Maximum is 262144 bytes.
```

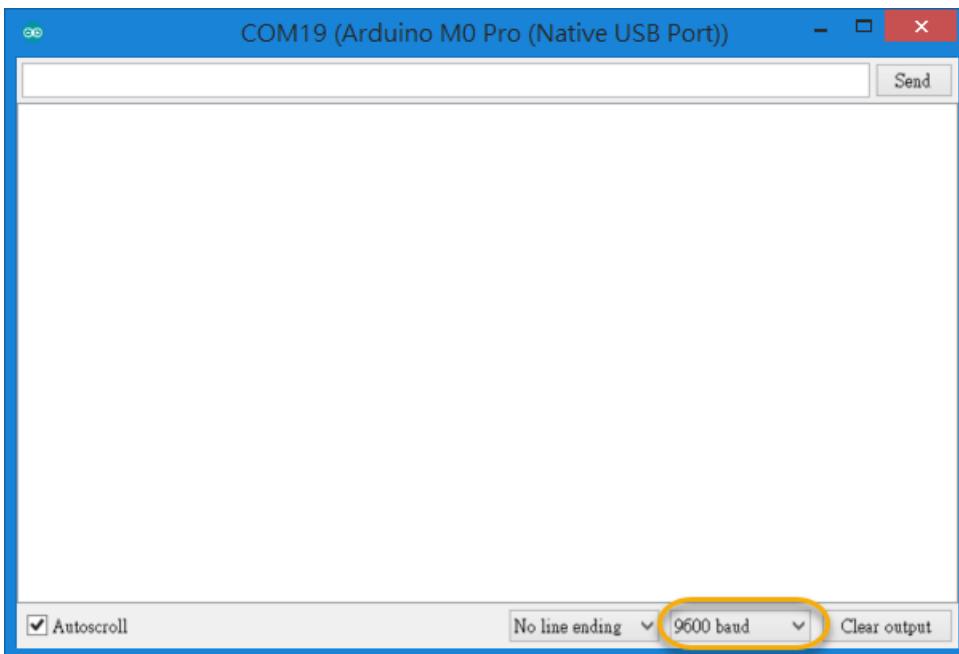


4.3.6 Confirm Upload

Step.1 Open Tools→Serial Monitor



Step.2 Select UART Baudrate →9600bps





Step.3 Send Command

Command	Description	Value
AT\$ID?	Get device ID	return ID
AT\$PAC?	Get device PAC	return PAC
AT\$SF=frame	Send payload data, 1 to 12 bytes	Frame: data bytes (0,1,2,3...C,D,E,F) to be sent, 12 byte maximum

Example:

ID: 4DB0E0

PAC: 787393AFA1404518

```

    COM19 (Arduino M0 Pro (Native USB Port)) - ×
    | Send
    AT$ID?
    000000000000000000000000000000004DB0E0
    AT$PAC?
    787393AFA140451800000000000000000000
    AT$SF=1234ABCDEF
    SFX_ERR: 0x31
    OK

    ✓ Autoscroll No line ending 9600 baud Clear output
  
```

If success, you can see the message.



4.4 Sigfox Backend - Active

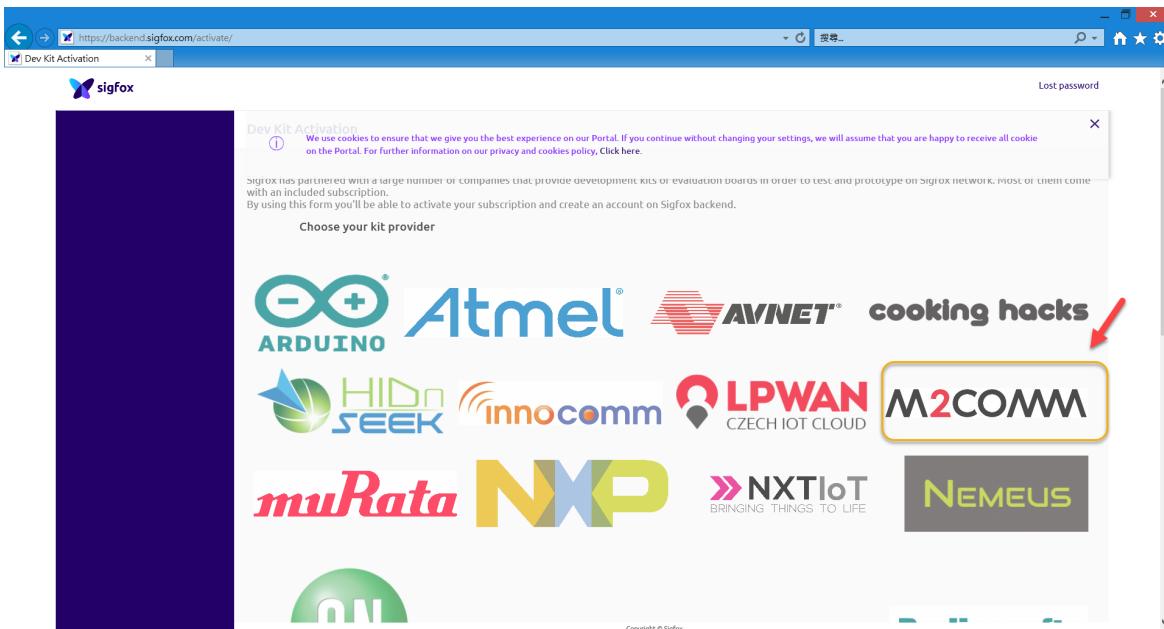
When you first time to use Sigfox device, you need to active

Step.1 Open website

<https://backend.sigfox.com/activate/>



Step.2 Click "M2COMM"



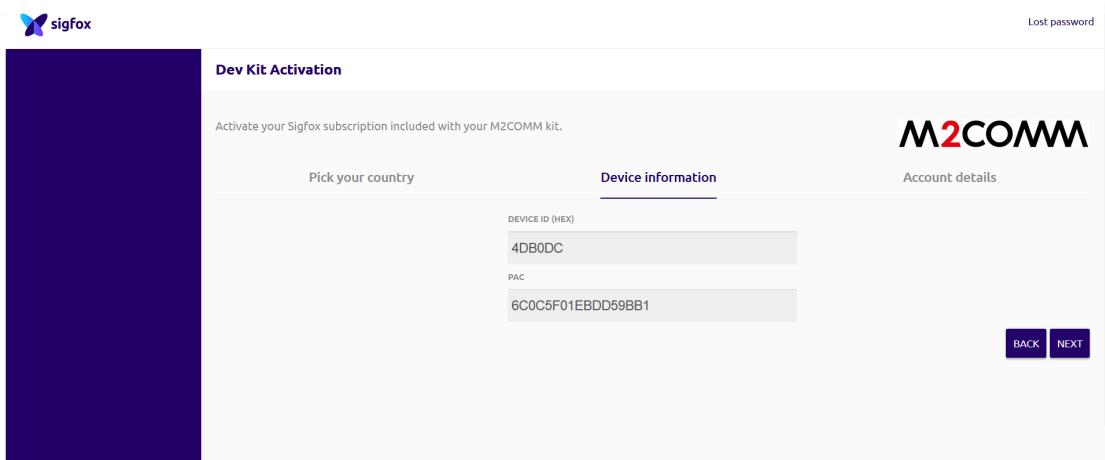


Step.3 Click "Singapore UnaBiz"



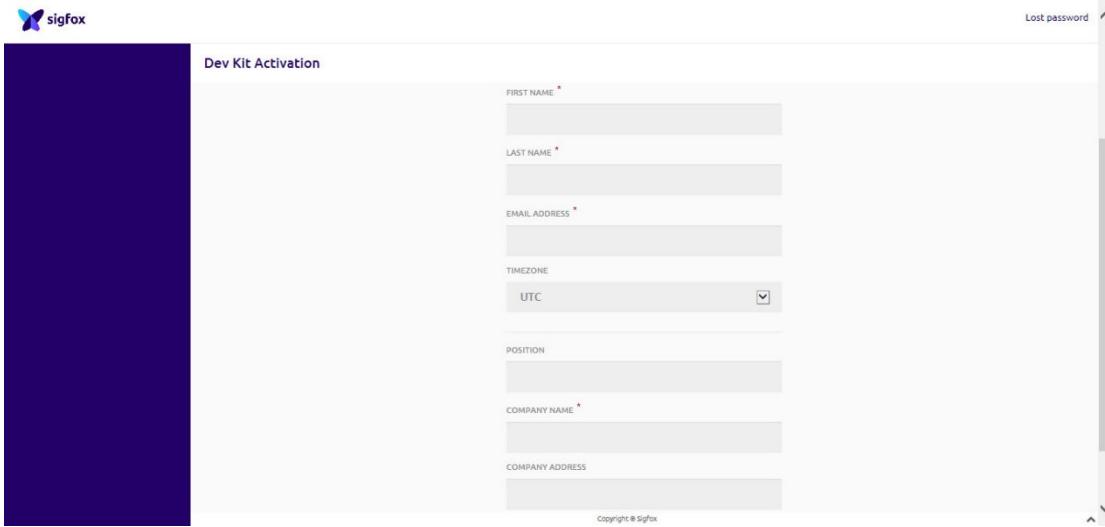
The screenshot shows the 'Dev Kit Activation' page from Omantel. It features a grid of logos for various connectivity partners. The 'UnaBiz' logo is highlighted with a red border and a red arrow pointing to it. Other logos visible include SimpleCell, WorldNET, cellnex, IoT SWEDEN, arqiva, IDEO, and sigfox.

Step.4 Enter ID and PAC information



The screenshot shows the 'Dev Kit Activation' page for M2COMM. The 'Device information' section is active. The 'DEVICE ID (HEX)' field contains '4DB0DC' and the 'PAC' field contains '6C0C5F01EBDD59BB1'. Navigation buttons 'BACK' and 'NEXT' are at the bottom right.

Step.5 Enter personal information



The screenshot shows the 'Dev Kit Activation' page for M2COMM. The 'Personal Information' section is active. The fields are as follows:

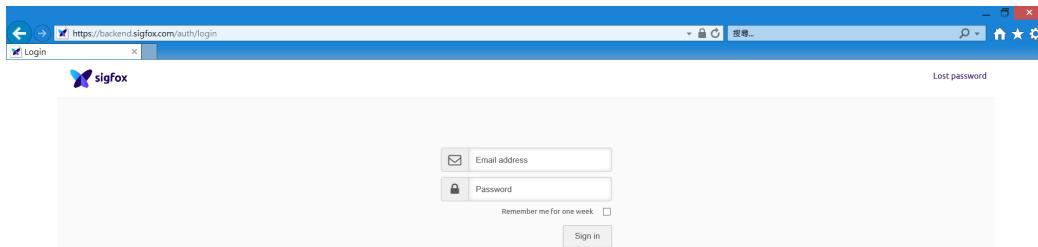
- FIRST NAME *
- LAST NAME *
- EMAIL ADDRESS *
- TIMEZONE: UTC
- POSITION
- COMPANY NAME *
- COMPANY ADDRESS



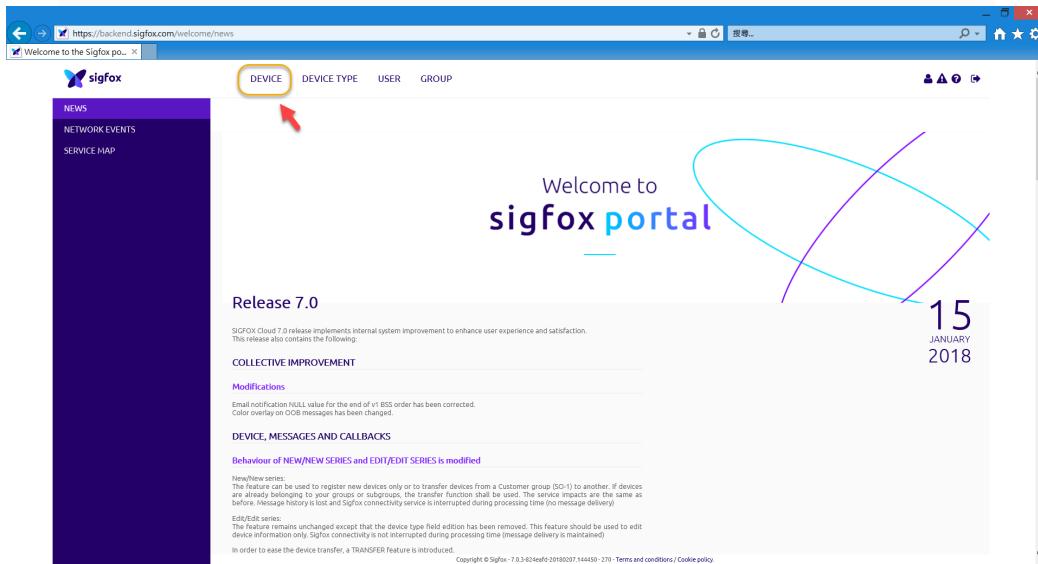
4.5 Sigfox Backend – Data Receive

Step.1 Open website, then enter “E-mail” and “Password”

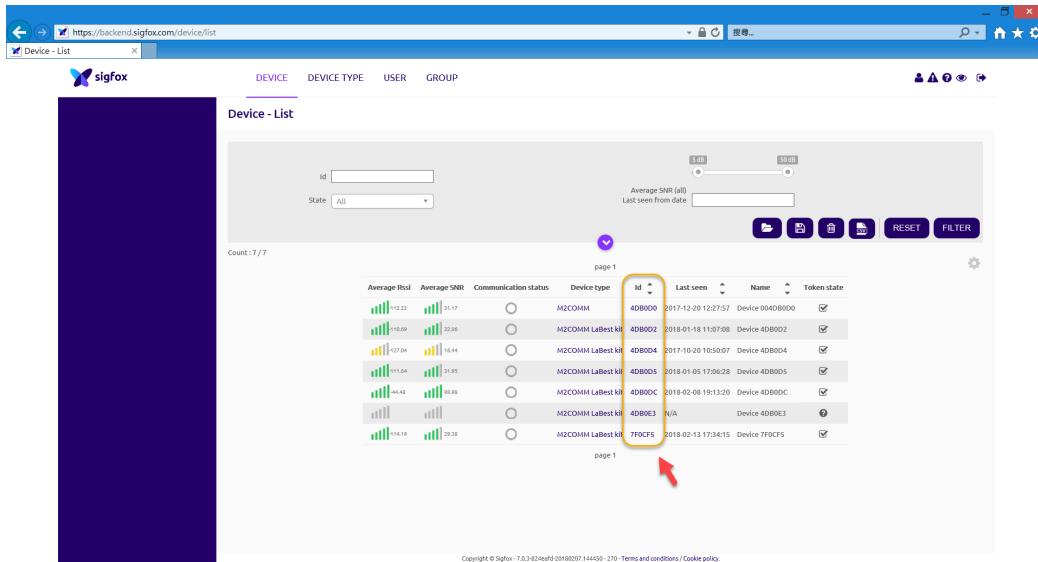
<https://backend.sigfox.com>



Step.2 Click “Device”



Step.3 Select ID and Click





Step.4 Click "Message"

The screenshot shows the 'Device 4DB0D2 - Information' page. The left sidebar has tabs for INFORMATION, LOCATION, MESSAGES (which is highlighted with a red box and arrow), EVENTS, STATISTICS, and EVENT CONFIGURATION. The main content area displays various device details such as Name: Device 4DB0D2, Protocol: V1, Last seen: 2018-01-18 11:07:08, and Modem configurations: RC1, RC2, RC4.

Step.5 Check Data

The screenshot shows the 'Device 4DB0D2 - Messages' page. The left sidebar has tabs for INFORMATION, LOCATION, MESSAGES (which is highlighted with a purple bar), EVENTS, STATISTICS, and EVENT CONFIGURATION. The main content area includes filters for 'From date' and 'To date', and buttons for 'RESET', 'FILTER', and a download icon. Below these are two tables: 'page 1' showing message details like Time, Data/Decoding, Elevation, Link quality, and Callbacks; and 'page 2' showing similar data for the next set of messages.



5. Example Code

5.1 Sigfox Terminal Send AT

Filename: SIGFOX_Terminal_SendAT.ino

This code is designed to send data to Sigfox backend by AT command via Terminal. Using command “AT\$SF” to make sure device is ready to connect Sigfox backend. It also can used AT command to get Sigfox registered information ID and PAC from Terminal, show on screen.

Code

```
#include <Arduino.h>
#include "wiring_private.h"
int LED_POW=3;
int RF_PWEN=5;
int count, i, INT;
char value;

void setup()
{
    SerialUSB.begin(9600);
    Serial1.begin(9600);
    pinMode(LED_POW,OUTPUT);
    pinMode(RF_PWEN,OUTPUT);
    digitalWrite(LED_POW,HIGH);
    digitalWrite(RF_PWEN,HIGH);
}

void loop()
{
    INT = SerialUSB.available(); // check Serial port

    if(INT>1)
    {
        for(int i = 0; i < INT; i++)
        {
            value = SerialUSB.read(); // Receive Serial port data
            Serial1.print(value);
        }
        Serial1.print("\r");
    }
    delay(100);
    count=Serial1.available();
    if(count>1)
    {
        for(int i = 0; i < count; i++)
        {
            value = Serial1.read();
            SerialUSB.print(value);
        }
    }
    delay(100);
}
```

// required before wiring_private.h
// pinPeripheral() function
// sigfox status LED pin D3
// sigfox power control pin D5

// Set Terminal Baudrate: 9600bps
// Set sigfox Baudrate:9600bps
// Set LED_POW pin as output
// Set RF_PWEN pin as output
// Set LED_POW to High
// Set RF_PWEN to High, and Sigfox module power ON

// check Serial port
// check Serial port

// Receive Serial port data



Result

```
ee COM21 (Arduino M0 Pro (Native USB Port)) - x
AT$PAC?
AT$ID?
00000000000000000000000000000004DB0D5

 Autoscroll No line ending 9600 baud Clear output
```

```
ee COM21 (Arduino M0 Pro (Native USB Port)) - x
AT$ID?
00000000000000000000000000000004DB0D5
AT$PAC?
484942FADDEB0FD10000000000000000
AT$SF=ABCDEF
SFX_ERR: 0x31
OK

 Autoscroll No line ending 9600 baud Clear output
```

Device 4DB0D5 - Messages

From date	<input type="text"/>	To date	<input type="text"/>	<input type="button" value="RESET"/>	<input type="button" value="FILTER"/>	<input type="button" value="CSV"/>
page 1						
Time	Data / Decoding	Location	Link quality	Callbacks		
2018-02-14 11:54:05	abcdef					



5.2 Sigfox Send

Filename: SIGFOX_Send_AT.ino

This code is designed to send data to Sigfox backend. It also can used AT command to get Sigfox registered information ID and PAC from Terminal, show on screen.

Code

```
#include <Arduino.h> // required before wiring_private.h
#include "wiring_private.h" // pinPeripheral() function
int LED_POW=3; // sigfox status LED pin D3
int RF_PWEN=5; // sigfox power control pin D5

unsigned long previousMillis = 0; // Time Counter
const long interval = 10000; // Unit: ms

void setup() {
    SerialUSB.begin(9600); // Set Terminal Baudrate: 9600bps
    Serial1.begin(9600); // Set sigfox Baudrate:9600bps
    pinMode(LED_POW,OUTPUT); // Set LED_POW pin as output
    pinMode(RF_PWEN,OUTPUT); // Set RF_PWEN pin as output
    digitalWrite(LED_POW,HIGH); // Set LED_POW to High
    digitalWrite(RF_PWEN,HIGH); // Set RF_PWEN to High, and Sigfox module power ON
}

void loop()
{
    unsigned long currentMillis = millis(); //sigfox AT Command

    if (currentMillis - previousMillis >= interval)
    {
        sigfox_atcommand_tx("AT$RCZ?\r"); //sigfox AT Command: get RCZ
        sigfox_atcommand_tx("AT$ID?\r"); //sigfox AT Command: Get ID
        sigfox_atcommand_tx("AT$PAC?\r"); //sigfox AT Command: Get PAC
        delay(100);

        previousMillis = currentMillis;
        sigfox_atcommand_tx("AT$SF=1234567890AB\r"); // Send Data to backend function
    }
}

void sigfox_atcommand_tx(char *wBuffer)
{
    Serial1.print(wBuffer); // Send Data to backend
    sigfox_atcommand_rx(); // Get feedback and show on Terminal function
}

void sigfox_atcommand_rx()
{
    char temp;
    while(Serial1.available() <= 0);
    while(Serial1.available() > 0)
    {
        temp = Serial1.read();
        SerialUSB.print(temp);
        delay(10);
    }
}
```



Result

COM21 (Arduino M0 Pro (Native USB Port))

```

484942FADDEB0FD10000000000000000000
AT$SF=1234567890AB
SFX_ERR: 0x31
OK
AT$RCZ?
4
AT$ID?
000000000000000000000000000000004DB0D5
AT$PAC?
484942FADDEB0FD10000000000000000000
AT$SF=1234567890AB
OK
AT$RCZ?
4
AT$ID?
000000000000000000000000000000004DB0D5
AT$PAC?
484942FADDEB0FD10000000000000000000
AT$SF=1234567890AB

```

Autoscroll No line ending 9600 baud Clear output

Device 4DB0D5 - Messages

From date	To date	RESET	FILTER	CSV
page 1				
Time	Data / Decoding	Location	Link quality	Callbacks
2018-02-14 11:51:58	1234567890ab			
2018-02-14 11:51:48	1234567890ab			
2018-02-14 11:51:38	1234567890ab			
2018-02-14 11:51:28	1234567890ab			
2018-02-14 11:51:18	1234567890ab			

NOTE

Command	Description	Value
AT\$SF=frame	AT\$SF=frameSend payload data, 1 to 12 bytes	Frame: data bytes (0,1,2,3...C,D,E,F) to be sent, 12 byte maximum