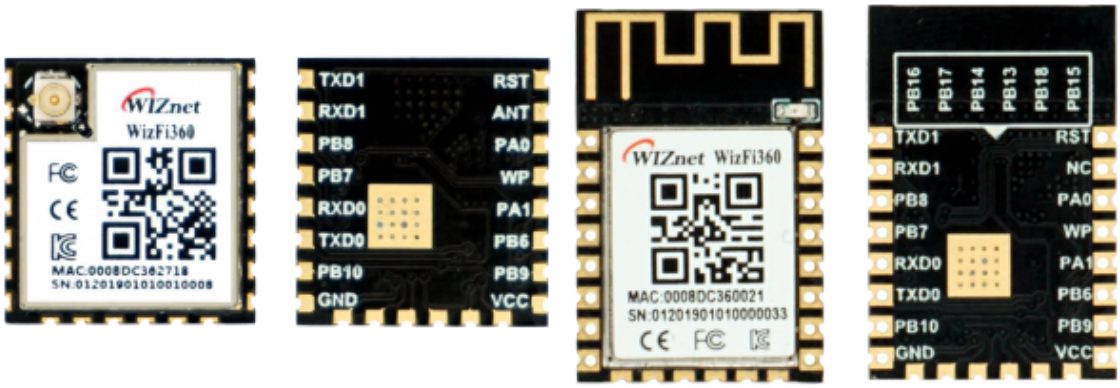
Getting Started

#Overview

WizFi360-CON **WizFi360-PA**



WizFi360 is a low cost and low-power consumption industrial-grade WiFi module. It is compatible with IEEE802.11 b/g/n standard and supports SoftAP, Station and SoftAP+Station modes. The serial port baud rate can be up to 2Mbps, which can meet the requirement of various applications.

#Features

- WiFi 2.4G, 802.11 b/g/n
- Support Station / SoftAP / SoftAP+Station operation modes
- Support “Data pass-through” and “AT command data transfer” mode
- Support serial AT command configuration
- Support TCP Server / TCP Client / UDP operating mode
- Support configuration of operating channel 0 ~ 13
- Support auto 20MHz / 40MHz bandwidth
- Support WPA_PSK / WPA2_PSK encryption
- Serial port baud rate up from 600bps to 2Mbps with 16 common values
- Support up to 5 TCP / UDP links
- Obtaining IP address automatically from the DHCP server (Station mode)
- DHCP service for Wireless LAN clients (AP mode)
- Support DNS for communication with servers by domain name
- Support “Keep-Alive” to monitor TCP connection
- Support “Ping” for monitoring network status
- Built-in SNTP client for receiving the network time
- Support built-in unique MAC address and user configurable
- Support firmware upgrade by UART Download / OTA (via WLAN)
- Industrial grade (operating temperature range: -40 ° C ~ 85 ° C)
- CE, FCC certification

#Details

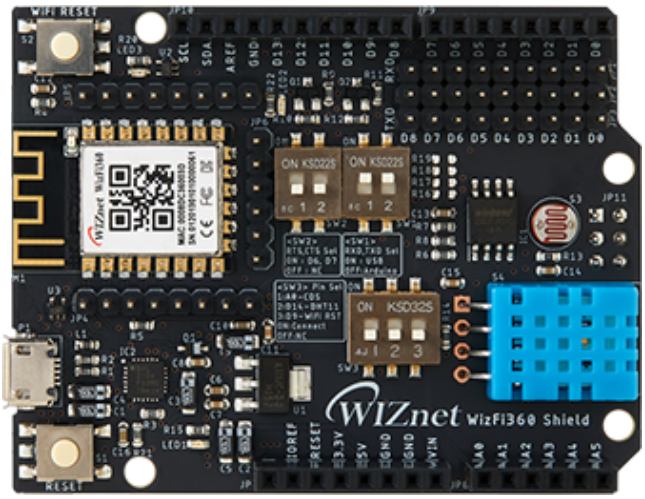
- Documents
  - Datasheet : A brief introduce about WizFi360. These documents describe the Features, package information and etc.
  - AT Instruction set: It describes the AT Command of WizFi360 and you can confirm the return message by AT command of WizFi360.
  - AT command Example: It includes the Basic example using AT command and it is examples of TCP, UDP, SSL use.
  - Quick Start Guide: The examples using the WizFi360-EVB or WizFi360io, it describes the scenarios in order from Hardware setting to example of AT command use

#Evaluation Boards

[WizFi360-EVB-Mini]

- NodeMCU Pin-Compatible

[WizFi360-EVB-Shield]



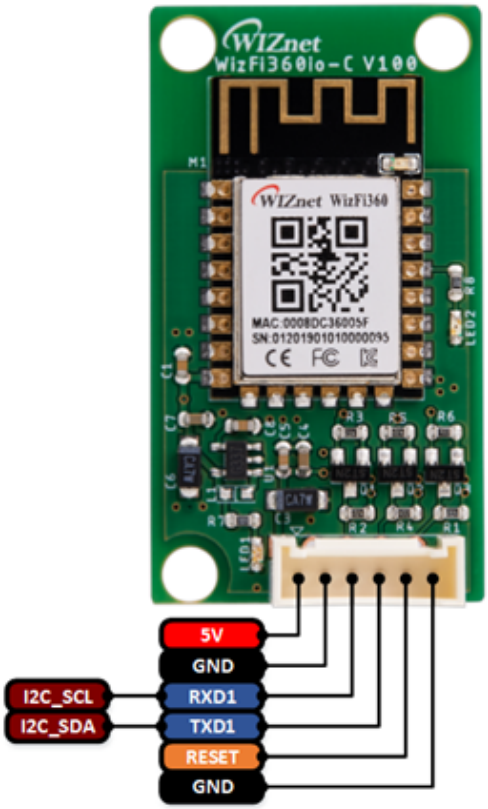
Arduino Pin-Compatible

#Custom Modules

[WizFi360io-H]

Similar to XBee pinout

[WizFi360io-C]



5V operation

Resources

DataSheet provides a brief introduce about WizFi360. These documents describe the Features,package information and etc.

Program Guide provides the AT Instruction set and AT command examles. The AT Instruction document include the description of AT command and the AT command examples document include TCP/UDP/SSL connection result and AT command used example.

AT Command Comparison Sheet provides comparison with ESP8266 AT command.

#DataSheet

#WizFi360

- WizFi360 Datasheet v1.05
- 2019-12-11 V1.05
    - Edited “5. Pin Definitions”
        - No.11 & No.19

▶ Old Datasheet History

#WizFi360-EVB-Shield

WizFi360-EVB-Shield Datasheet

Version	Date	Decription
V 1.1.2	SEP2019	Initial Release

#WizFi360-EVB-Mini

WizFi360-EVB-Mini Datasheet

Version	Date	Decription
V 1.0.0	SEP2019	Initial Release

#WizFi360io Series

#WizFi360io-C

WizFi360io-C Datasheet

Version	Date	Decription
V 1.0.2	SEP2019	Initial Release

#WizFi360io-H

WizFi360io-H Datasheet

Version	Date	Decription
V 1.0.1	AUG2019	Initial Release

#Certification

#CE

WizFi360 Certification

Date	Decription
AUG2019	Initial Release

#FCC

WizFi360 Certification

Date	Decription
AUG2019	Initial Release

#KC

WizFi360-PA_EMC Certification

WizFi360-CON_EMC Certification

Date	Decription
NOV2019	Initial Release

#J-MIC(TELEC)

WizFi360-PA_EMC Certification

WizFi360-CON_EMC Certification

Date	Decription
NOV2019	Initial Release

#RoHS

SGS RoHS Test Report

Date	Decription
NOV2019	Initial Release

#RoHS

SGS REACH SVHC Test Report

Date	Decription
NOV2019	Initial Release

#Reliability Test Report

KORAS Operation Test - Korean

KORAS Operation Test - English

KORAS Operation Test - Korean

KORAS Operation Test - Korean

Date	Decription
NOV2019	Initial Release

#Program Guide

#AT Instruction set

The AT Instruction set include the description of AT command. WizFi360 basically provides a command compatible with ESP8266. In addition, commands are provided for using MQTT and Azure IoT Hub.

- AT Instruction set V1.0.7 - English
- AT Instruction set V1.0.7 - Korean

Version	Date	Decription
V 1.0.7	MAR2020	Add AT+WIZ_NETCONFIG, Modify AT+CWSTARTSMART Modify minor typos and error

▶ Old Datasheet History

#AT Command Comparison Sheet

The AT Command Comparison Sheet provides comparison with ESP8266 AT command.

AT Command Comparison Sheet V1.0.1

Version	Date	Decription
V1.0.0	AUG2019	Initial Release
V1.0.1	OCT2019	Modify AT+CIPSSLCCONF Add AT+AZCON, AT+AZSET and AT+CASEND command

#AT Command examples

The AT command examples document include TCP/UDP/SSL connection result and AT command used example.

- AT Command Examples V1.0.3 - English
- AT Command Examples V1.0.3 - Korean

Version	Date	Decription
V1.0.3	OCT2019	Add “Auto TCP Connection on Transparent Mode”

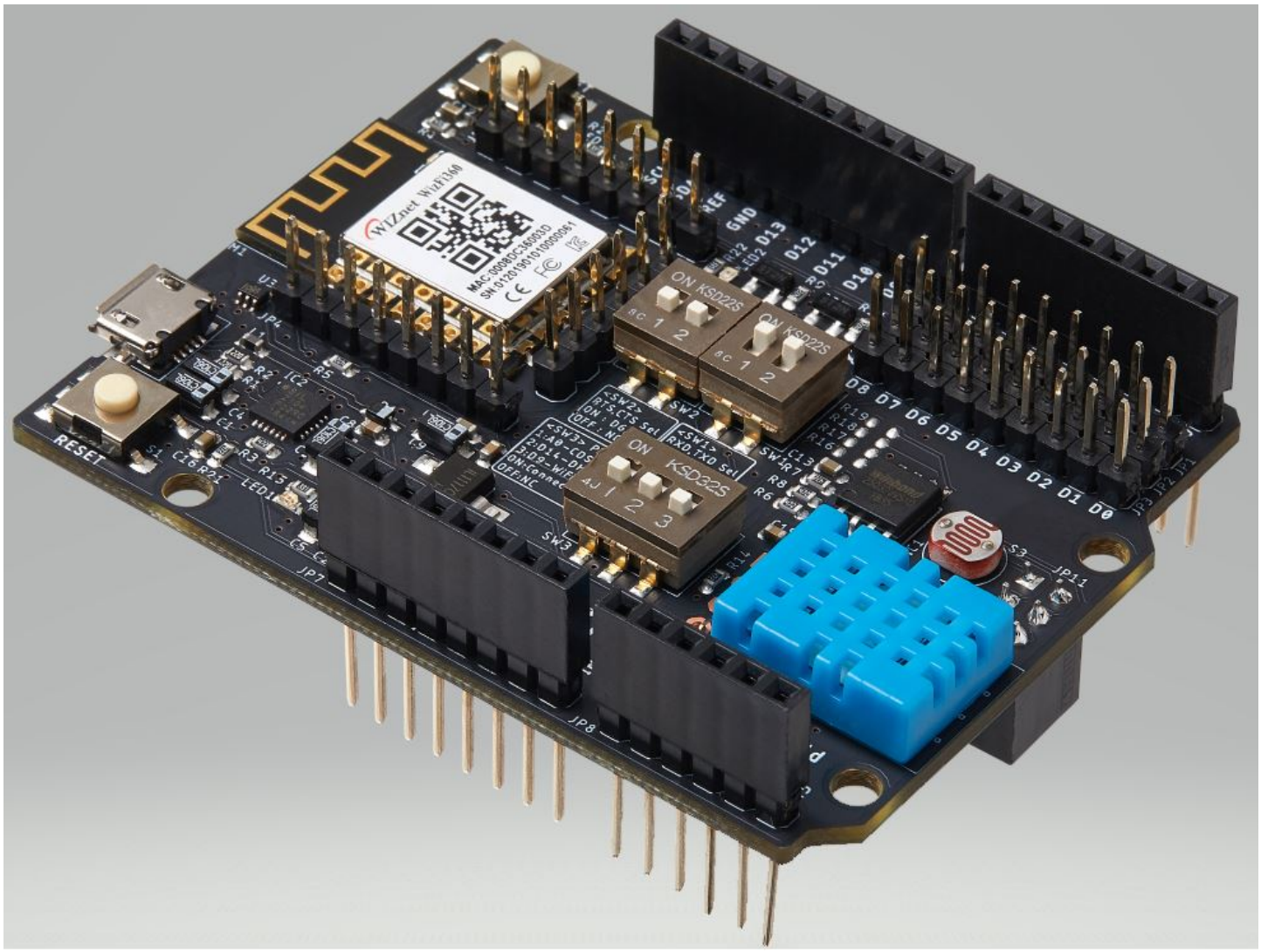
▶ Old Datasheet History

#Hardware Design Guide

Hardware Design Guide

Version	Date	Decription
V1.0.3	NOV2019	Initial Release

WizFi360-EVB-Shield



#Overview

This document describes WizFi360-EVB-Shield. WizFi360-EVB-Shield is a development board for experiment, test and verification of WizFi360. WizFi360-EVB-Shield can also be used as an Arduino shield. WizFi360 is a low cost and low-power consumption industrial-grade WiFi module. It is compatible with IEEE802.11 b/g/n standard and supports SoftAP, Station and SoftAP+Station modes. The serial port baud rate can be up to 2Mbps, which can meet the requirement of various applications.

#Features

- WizFi360
  - WiFi 2.4G, 802.11 b/g/n
  - Support Station / SoftAP / SoftAP+Station operation modes
  - Support “Data pass-through” and “AT command data transfer” mode
  - Support serial AT command configuration
  - Support TCP Server / TCP Client / UDP operating mode
  - Support configuration of operating channel 0 ~ 13
  - Support auto 20MHz / 40MHz bandwidth
  - Support WPA_PSK / WPA2_PSK encryption
  - Serial port baud rate up from 600bps to 2Mbps with 16 common values
  - Support up to 5 TCP / UDP links
  - Obtaining IP address automatically from the DHCP server (Station mode)
  - DHCP service for Wireless LAN clients (AP mode)
  - Support DNS for communication with servers by domain name
  - Support “Keep-Alive” to monitor TCP connection
  - Support “Ping” for monitoring network status
  - Built-in SNTP client for receiving the network time
  - Support built-in unique MAC address and user configurable
  - Support firmware upgrade by UART Download / OTA (via WLAN)
  - Industrial grade (operating temperature range: -40 ° C ~ 85 ° C)
  - CE, FCC certification
- ETC
  - Built-in UART to USB chip
    - CP2104-GM
    - Micro USB B Type Connector
  - UART Selector

        - JP1, JP2, JP3
        - 2.54mm Pin Header
    - Built-in Sensors
        - Temprature/Humidity Sensor: DHT11
        - CDS Sensor: GL5537
    - Built-in Tact Switchs
        - System Reset Switch: S1
        - WiFi Reset Switch: S2
    - Built-in LED Indicators
        - D13 LED
    - Built-in Level Shifters
        - The voltage of the RXD/TXD signal changes according to the main board platform voltage.
    - Built-in DIP Switchs
        - UART RXD/TXD Selector: SW1
        - UART RTS/CTS Selector: SW2
        - Sensor/RESET Pins Selector: SW3

#Quick Start Guide

Quick Start Guide

#Datasheet

Download

#Technical Reference

#Ref Schematic & Other Board Schematics

Link to Github

#Library

Link to Github

#ETC

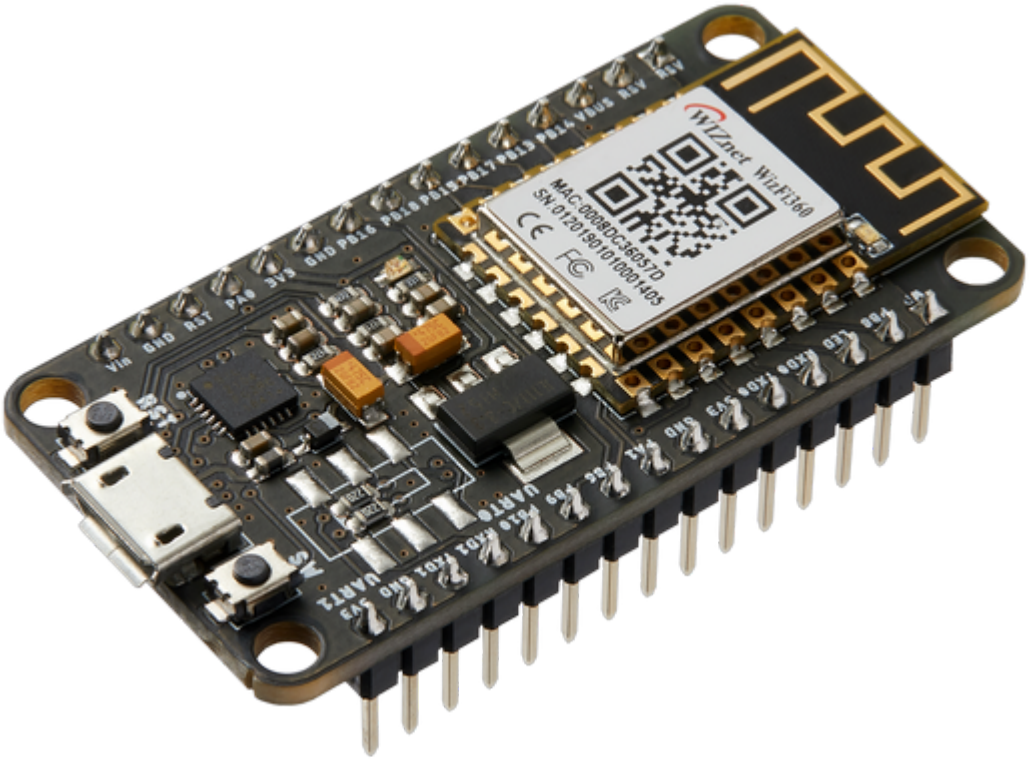
#Serial Port Driver

CP210x USB to UART Bridge VCP Drivers

#Sensor Datasheets

- DHT11 Humidity & Temperature Sensor Datasheet
- GL5537 Datasheet

WizFi360-EVB-Mini



#Overview

This document describes WizFi360-EVB-Mini. WizFi360-EVB-Mini is a compact development board for experiment, test and verification of WizFi360. WizFi360-EVB-Mini is the same form factor as the NodeMCU V2. WizFi360 is a low cost and low-power consumption industrial-grade WiFi module. It is compatible with IEEE802.11 b/g/n standard and supports SoftAP, Station and SoftAP+Station modes. The serial port baud rate can be up to 2Mbps, which can meet the requirement of various applications.

#Feature

- WizFi360
  - WiFi 2.4G, 802.11 b/g/n
  - Support Station / SoftAP / SoftAP+Station operation modes
  - Support “Data pass-through” and “AT command data transfer” mode
  - Support serial AT command configuration
  - Support TCP Server / TCP Client / UDP operating mode
  - Support configuration of operating channel 0 ~ 13
  - Support auto 20MHz / 40MHz bandwidth
  - Support WPA_PSK / WPA2_PSK encryption
  - Serial port baud rate up from 600bps to 2Mbps with 16 common values
  - Support up to 5 TCP / UDP links
  - Obtaining IP address automatically from the DHCP server (Station mode)
  - DHCP service for Wireless LAN clients (AP mode)
  - Support DNS for communication with servers by domain name
  - Support “Keep-Alive” to monitor TCP connection
  - Support “Ping” for monitoring network status
  - Built-in SNTP client for receiving the network time
  - Support built-in unique MAC address and user configurable
  - Support firmware upgrade by UART Download / OTA (via WLAN)
  - Industrial grade (operating temperature range: -40 ° C ~ 85 ° C)
  - CE, FCC certification
- ETC
  - Built-in UART to USB chip

- CP2104-GM
- Micro USB B Type Connector

#Quick Start Guide

Quick Start Guide

#Datasheet

Download

#Technical Reference

#Ref Schematic & Other Board Schematics

Link to Github

#Library

Link to Github

#ETC

#Serial Port Driver

CP210x USB to UART Bridge VCP Drivers

WizFi360io

#Overview

This page describes the io interface boards using WizFi360. WizFi360io series boards can be customized according to customers' requests.

#WizFi360io-C

WizFi360io-C cable connector type io interface board. The operating voltage of the WizFi360io-C and the UART voltage are 5V. WizFi360io-C can connect the UART interface to the connector. The part name of the connector mounted on WizFi360io-C is “SMW200-06”. The cable connector that can be connected to “SMW200-06” is “SMH200-06”.

#WizFi360io-H

WizFi360io-H is a pin header type io interface board. The form factor of WizFi360io is an Xbee interface module. 2.00mm pin header is used, similar to the Xbee pin layout. But it is not exactly compatible.

#Features

- WizFi360
  - WiFi 2.4G, 802.11 b/g/n
  - Support Station / SoftAP / SoftAP+Station operation modes
  - Support “Data pass-through” and “AT command data transfer” mode
  - Support serial AT command configuration
  - Support TCP Server / TCP Client / UDP operating mode
  - Support configuration of operating channel 0 ~ 13
  - Support auto 20MHz / 40MHz bandwidth
  - Support WPA_PSK / WPA2_PSK encryption
  - Serial port baud rate up from 600bps to 2Mbps with 16 common values
  - Support up to 5 TCP / UDP links
  - Obtaining IP address automatically from the DHCP server (Station mode)
  - DHCP service for Wireless LAN clients (AP mode)
  - Support DNS for communication with servers by domain name
  - Support “Keep-Alive” to monitor TCP connection
  - Support “Ping” for monitoring network status
  - Built-in SNTP client for receiving the network time
  - Support built-in unique MAC address and user configurable
  - Support firmware upgrade by UART Download / OTA (via WLAN)
  - Industrial grade (operating temperature range: -40 ° C ~ 85 ° C)
  - CE, FCC certification
- **WizFi360io-C**
  - 5V Operating Voltage
    - Built-in LDO
    - WizFi360 operates at 3.3V
  - 5V voltage level UART
    - Built-in Level Shifters
    - WizFi360 operates at 3.3V
  - SMW200-06 Mounted
    - Cable side connector is SMH200-06
- **WizFi360io-H**
  - 3.3V Operating Voltage
    - Xbee Form Factor
    - 2.00mm Pin Header

#Quick Start Guide

Quick Start Guide

#Datasheet

WizFi360io-C

WizFi360io-H

#Technical Reference

#Ref Schematic & Other Board Schematics

Link to Github

#Library

Link to Github

#ETC

#WizFi360io-C Connector Datasheet

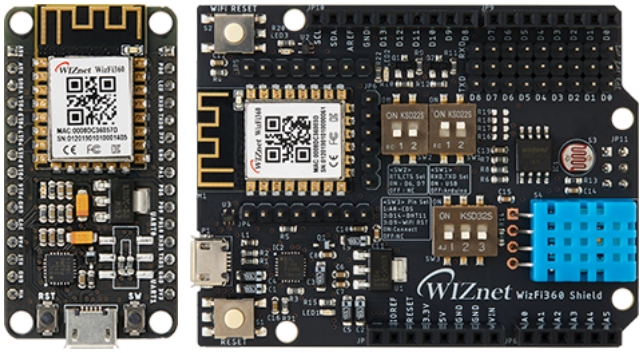
SMW200-06

Quickstart Guide

#Environment setting

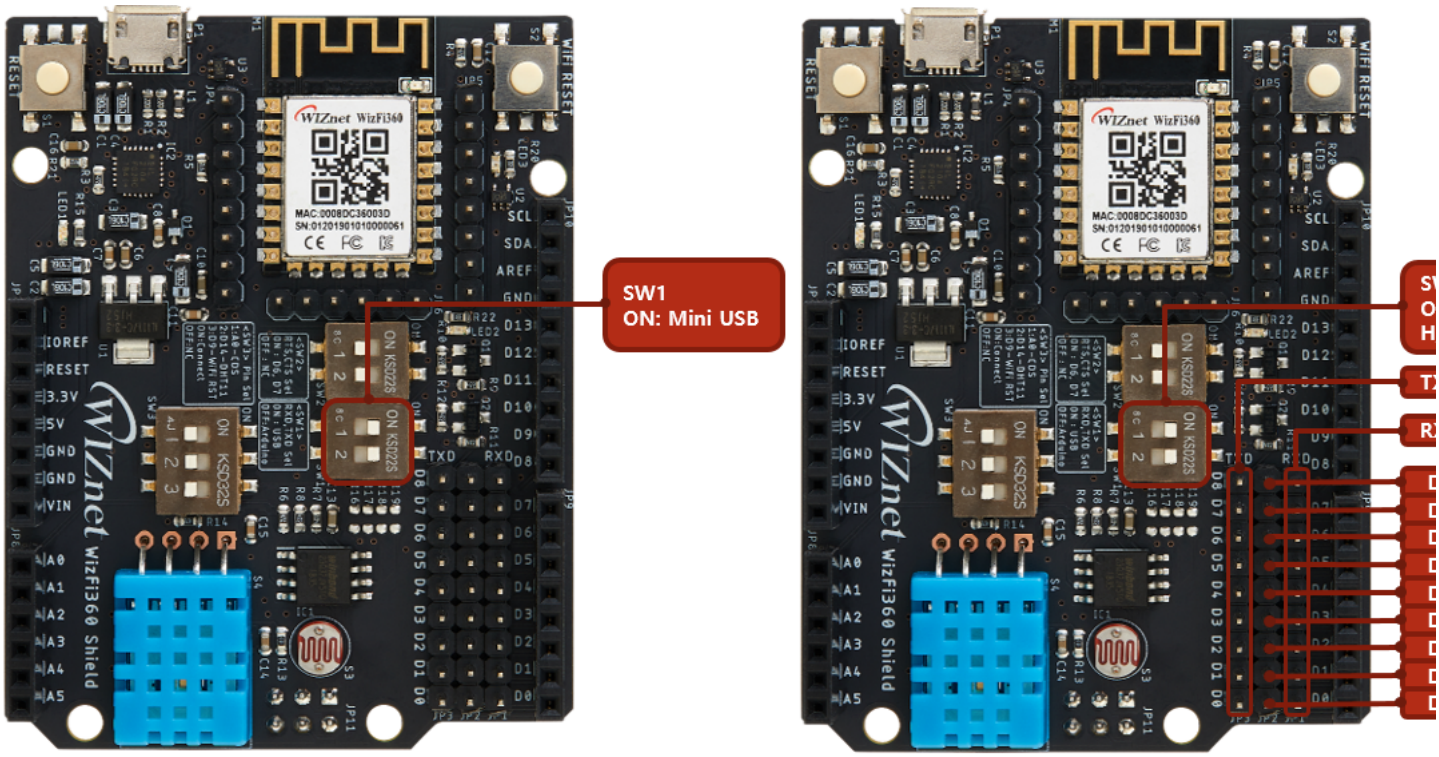
There are two types of WizFi360-EVB, which can be classified as ‘Shield’ and ‘Mini’ versions. WizFi360 is controlled by sending AT command through UART. WizFi360-EVB-Mini and WizFi360-EVB-Shield uses UART1 by connecting Micro USB.

WizFi360-EVB-Mini **WizFi360-EVB-Shield**

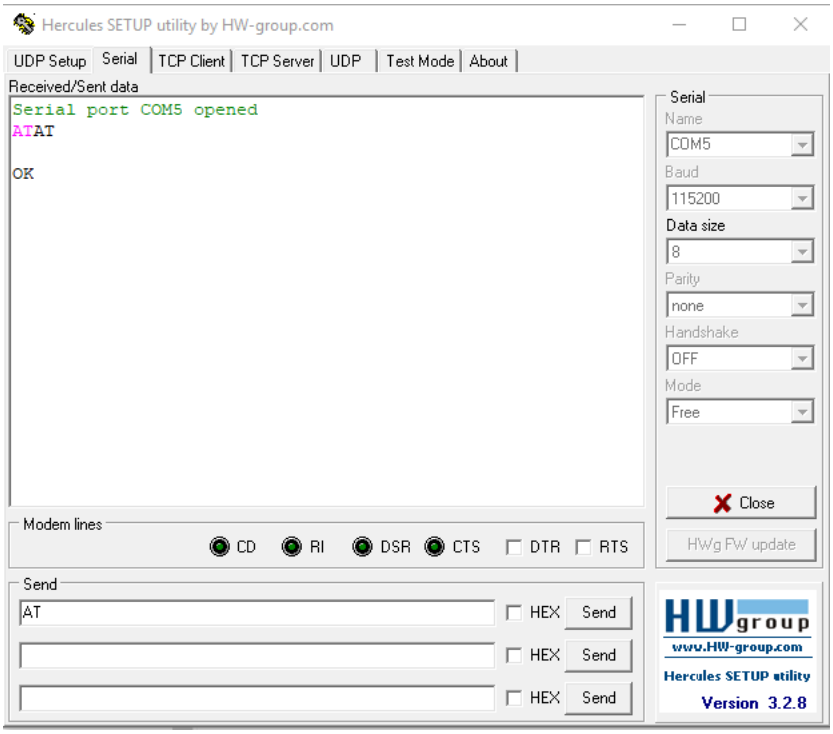


In addition, WizFi360-EVB-Shield is Pin compatible with Arduino. So you can use UART in two cases. If you use Micro USB, turn SW1 ON and connect Micro USB. In case of using Arduino H/W compatible pin, turn SW1 OFF and select RXD / TXD pin for Arduino using jumper cap. See the figure below.

Micro USB **Arduino H/W compatible pin**



If WizFi360-EVB-Shield or WizFi360-EVB-Mini is connected to a PC with a Mini USB Cable, run the serial program for UART communication on the PC. Open Port in Serial Program and input AT. If WizFi360 return OK, AT command can be used.



#TCP CLient Example

A brief AT command describe for operating of WizFi360 as TCP Client in single connection mode. If you need AT command example of another mode, see the documentation called "AT command examples"

Open TCP Server from PC which is connected to the same WiFi AP that WizFi360 is connected to. (*IP : 192.168.10.100 Port : 5000*)

Copy

```
//Set WizFi Station mode
AT+CWMODE_CUR=1 //Station mode, in case of AT+CWMODE_DEF, it is stored in flash.
OK

//Set Single Connection Mode
AT+CIPMUX=0
OK

//Set DHCP enable
AT+CWDHCP_CUR=1,1
OK

//Get possible WiFi AP list for WizFi360 connection
AT+CWLAP
+CWLAP:(3,"ssid",-5,"mac address",1,1)

//Connect to WiFi AP
AT+CWJAP_CUR="ssid","password"
WIFI CONNECTED
WIFI GOT IP

OK

//Query WizFi360 device' IP address
AT+CIPSTA_CUR?
+CIPSTA_CUR:ip:"192.168.10.13"
+CIPSTA_CUR:gateway:"192.168.10.1"
+CIPSTA_CUR:network:"255.255.255.0"

OK

//Connect to TCP server as client
AT+CIPSTART="TCP","192.168.10.100",5000    //protocol, server IP, port
CONNECT

OK

//WizFi360 sends data to the TCP Server
AT+CIPSEND=10    //set data length, such as 10 bytes.
OK
>  // return ">" to begin receiving of serial data.
1234567890    //enter the data, no <CR><LF>

Recv 10 bytes

SEND OK

//When WizFi360 received data from TCP Server, it will prompt message below:
+IPD,10:1234567890

//End the TCP Connection
AT+CIPCLOSE
CLOSED

OK
```

important

In normal transmission mode, there are three commands to send data. (For more details, refer to AT Instruction set.) 1. AT+CIPSEND 2. AT+CIPSENDBUF 3. AT+CIPSENDEX

In AT+CIPSEND and AT+CIPSENDEX, If the data is entered more than the length set (n) :

- the system will send the first n bytes and discard exceeded data.

In AT+CIPSENDBUF, If the data is entered more than the length set (n) :

- the system will reply busy and send the first n bytes
- and discard exceeded data.

Firmware Upgrade Guide

#There are three methods to update firmware: Tera Term(Serial Program), Cloud or Upgrade Tool.

#Using Tera Term (Serial)

#Required hardware and software

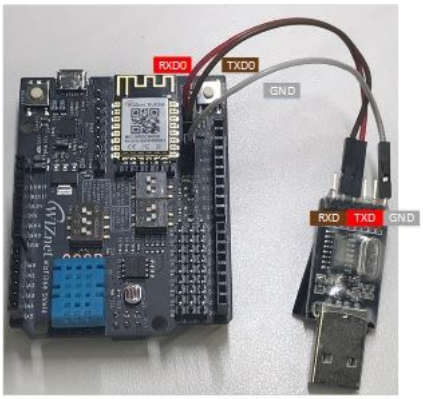
- WizFi360-EVB
- Tera Term
- USB cable or TTL-to-USB module
- WizFi360 Firmware .img file

#How to download using Serial

#Step 1

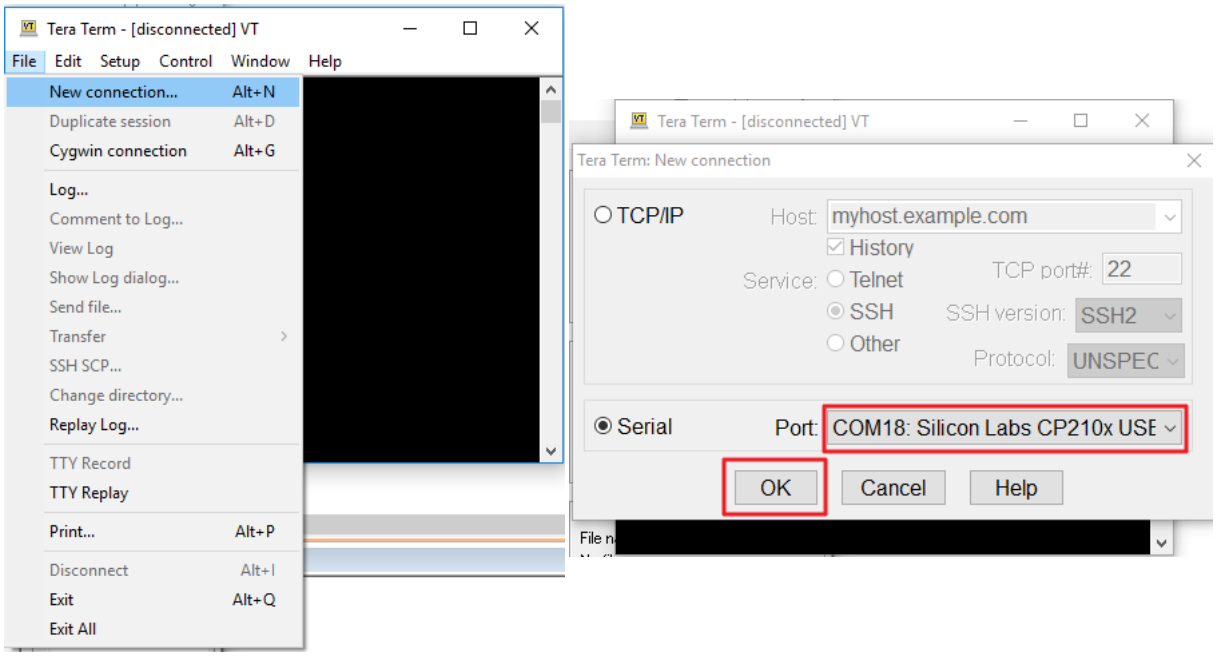
Update Firmware by communicating with Tera Term through Serial. Use UART0 or UART1 for serial communication. In case of UART1 use USB cable and in case of UART0 use TTL to USB module.

UART1 **UART0**



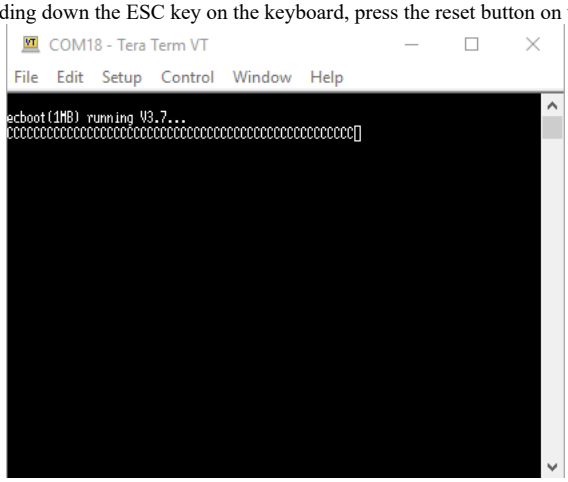
#Step 2

Run the Tera Term, go to ‘File - New connection’ menu, select ‘Serial’, and choose the port to which WizFi360 is connected. Set speed to 115200bps.



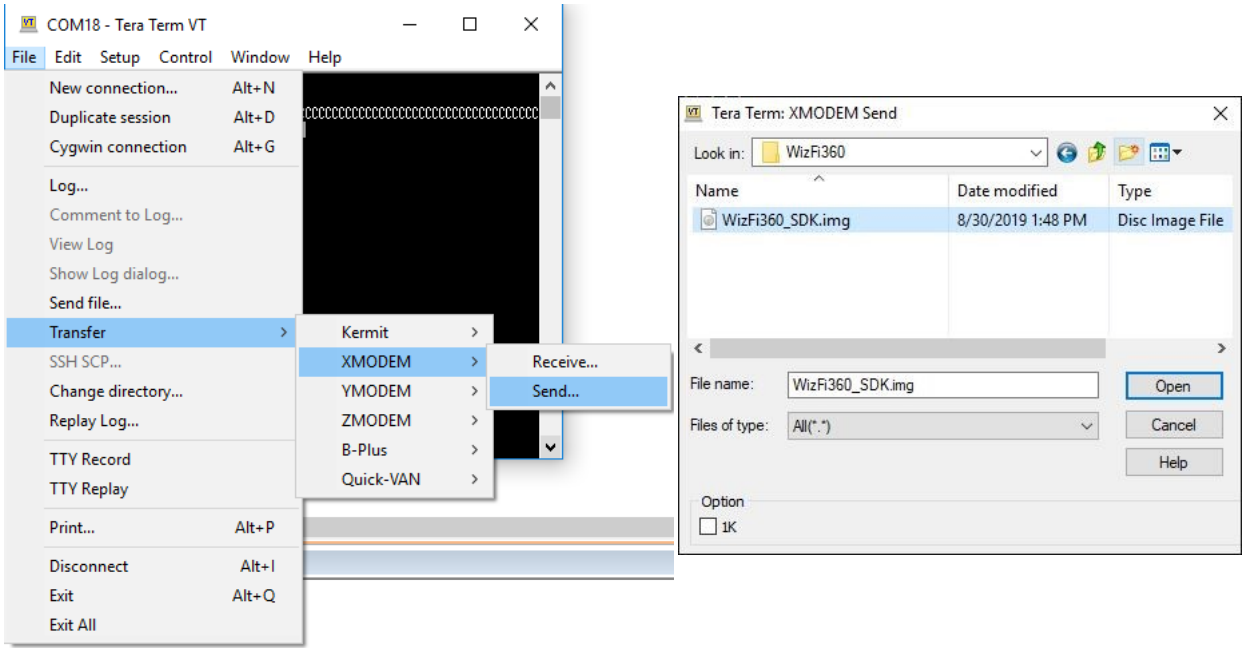
#Step 3

While holding down the ESC key on the keyboard, press the reset button on the WizFi360 to enter the Boot mode. When boot mode is entered, the below screen is displayed.



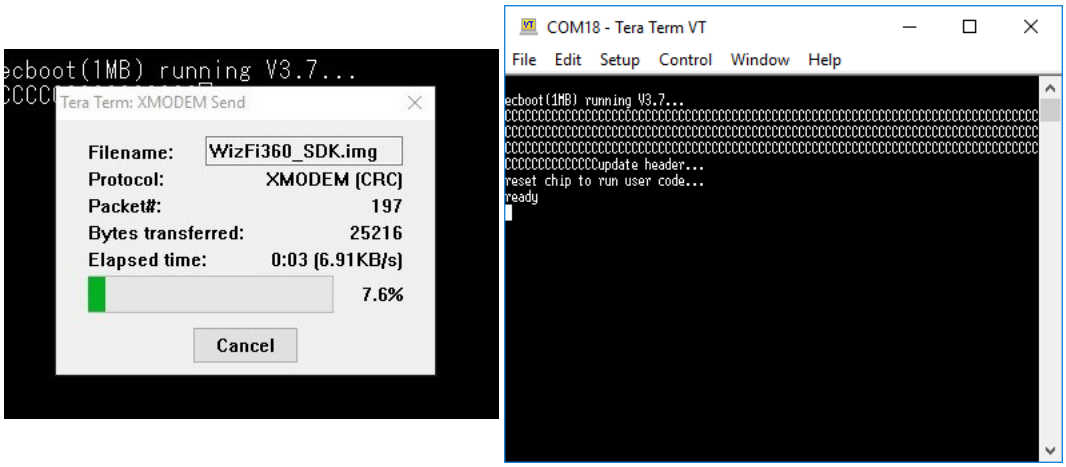
#Step 4

Go to 'File - Transfer - XMODEM - Send' menu, select the binary file and click OK button to proceed the firmware update.



#Step 5

When the firmware update is completed, 'ready' is displayed on the terminal.



#Using Cloud (Wi-Fi)

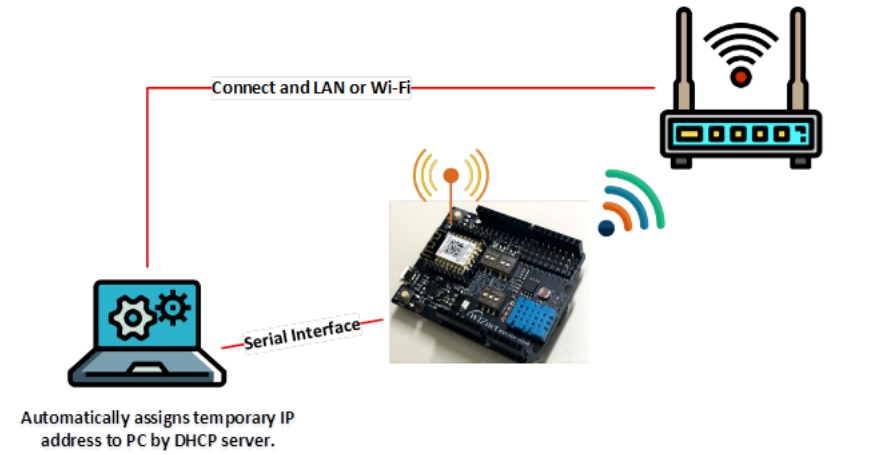
#Required hardware & software

- WizFi360-EVB
- USB cable or TTL-to-USB module

#How to upgrade using Cloud

#Step 1

Update Firmware by connecting to Cloud through WiFi. Connect PC, WizFi360, and Router as shown below. The router must be connected to the Internet, and the WizFi360 and WizFi360 and PC are assigned an IP by DHCP Server of router. To use AT Command, serial communication is used UART0 or UART1. In case of UART1 use USB cable and in case of UART0 use TTL to USB module.



#Step 2

Use the AT command to upgrade the firmware like below.

```
(10:09:15.104) ready<CR><LF>
(10:09:21.898) AT+CWMODE_CUR=1<CR><LF>
(10:09:22.039) AT+CWMODE_CUR=1<CR><LF>
(10:09:22.039) <CR><LF>
(10:09:22.039) OK<CR><LF>
(10:09:22.932) AT+CWLAP<CR><LF>
(10:09:23.081) AT+CWLAP<CR><LF>
(10:09:24.769) +CWLAP:(3,"##WIZnet_irina",-53,"64:e5:99:b8:60:5a",1,1)<CR><LF>
(10:09:24.854) +CWLAP:(4,"DIR-815_Wiznet",-59,"14:d6:4d:cf:fa:a7",1,1)<CR><LF>
(10:09:24.854) +CWLAP:(3,"Matthew2.4",-61,"88:36:6c:be:8a:3a",2,1)<CR><LF>
(10:09:24.854) +CWLAP:(0,"iptime",-69,"88:36:6c:7d:79:44",4,1)<CR><LF>
(10:09:24.854) +CWLAP:(3,"Dap",-57,"88:36:6c:e8:32:9e",3,1)<CR><LF>
(10:09:24.854) +CWLAP:(4,"Teddy_AP",-38,"90:9f:33:21:e5:00",5,1)<CR><LF>
(10:09:24.854) +CWLAP:(0,"WizFi630S_AP_57985E",-57,"00:08:dc:57:98:5e",6,0)<CR><LF>
(10:09:24.854) +CWLAP:(4,"DLINK-IPv6",-51,"cc:b2:55:d2:19:a2",7,1)<CR><LF>
(10:09:24.854) +CWLAP:(3,"WIZnet_Scott",-44,"18:a6:f7:56:73:2e",6,1)<CR><LF>
(10:09:24.854) +CWLAP:(3,"rena",-53,"88:36:6c:6e:ac:28",9,1)<CR><LF>
(10:09:24.854) +CWLAP:(3,"wizms1",-61,"e8:fc:af:fd:a2:85",11,1)<CR><LF>
(10:09:24.854) +CWLAP:(0,"iptime",-71,"64:e5:99:a4:e3:6c",11,1)<CR><LF>
(10:09:24.854) +CWLAP:(0,"ESP_577CC7",-85,"2e:f4:32:57:7c:c7",11,0)<CR><LF>
(10:09:24.854) <CR><LF>
(10:09:24.854) OK<CR><LF>
(10:09:25.523) AT+CWJAP_CUR="Teddy_AP","12345678"<CR><LF>
(10:09:25.669) AT+CWJAP_CUR="Teddy_AP","12345678"<CR><LF>
(10:09:29.122) WIFI CONNECTED<CR><LF>
(10:09:29.122) WIFI GOT IP<CR><LF>
(10:09:29.122) <CR><LF>
(10:09:29.122) OK<CR><LF>
(10:09:38.905) AT+CIUPDATE<CR><LF>
(10:09:39.052) AT+CIUPDATE<CR><LF>
(10:09:39.161) +CIPUPDATE:1<CR><LF>
(10:09:39.781) +CIPUPDATE:2<CR><LF>
(10:09:39.781) +CIPUPDATE:3<CR><LF>
(10:09:54.538) +CIPUPDATE:4<CR><LF>
(10:09:54.645) <CR><LF>
(10:09:54.645) OK<CR><LF>
(10:10:01.939) <CR><LF>
(10:10:01.939) ready<CR><LF>
```

#Using WizFi Upgrade Tool

#Required hardware & software

- WizFi360-EVB
- WizFi360 Upgrade Tool
- USB Cable or TTL to USB module
- WizFi360 Firmware: WizFi360_SDK.img

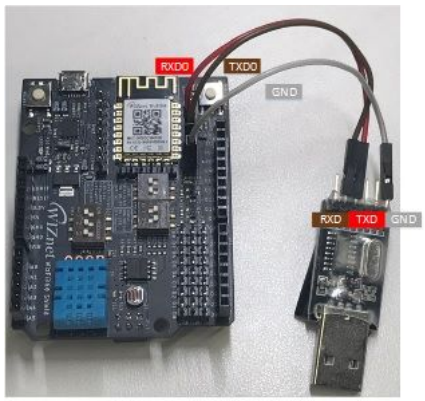
#How to upgrade using Upgrade Tool (Serial)

#Step 1

Update Firmware by communicating with Upgrade tool through Serial. Use UART0 or UART1 for serial communication. In case of UART1 use USB cable and in case of UART0 use TTL to USB module.

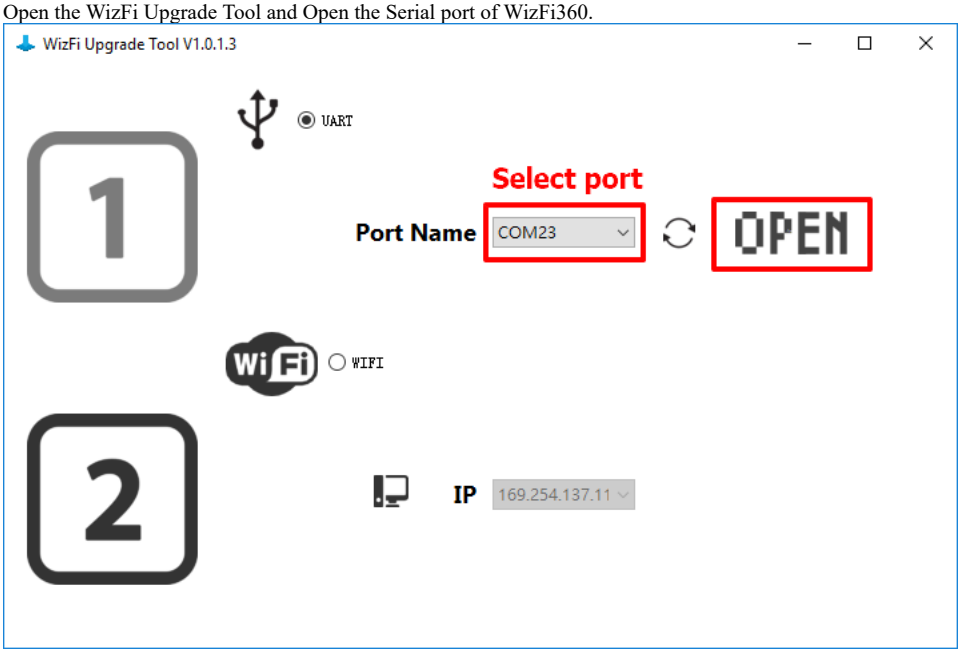
UART1	UART0

UART1 **UART0**



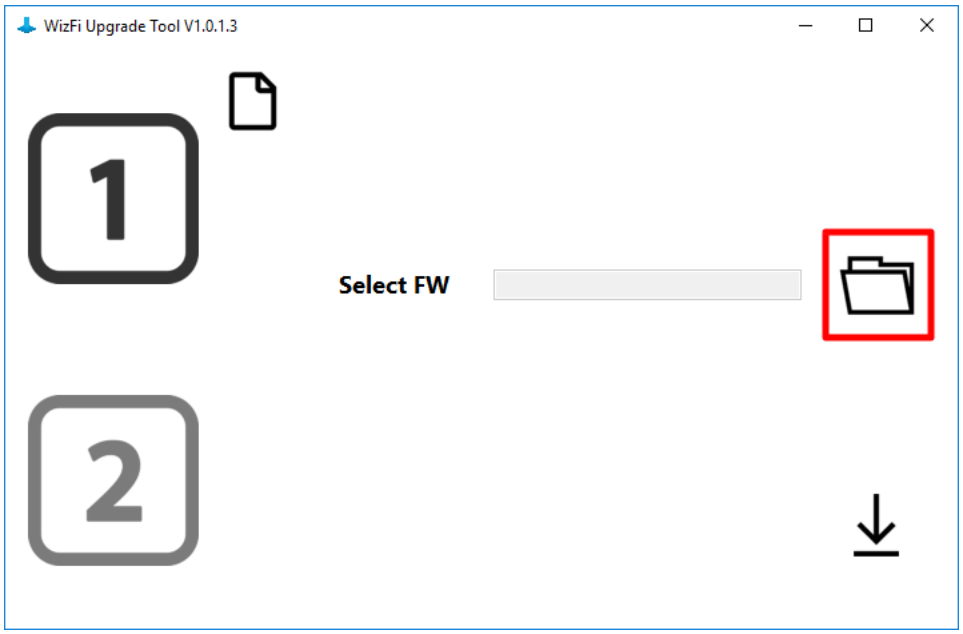
#Step 2

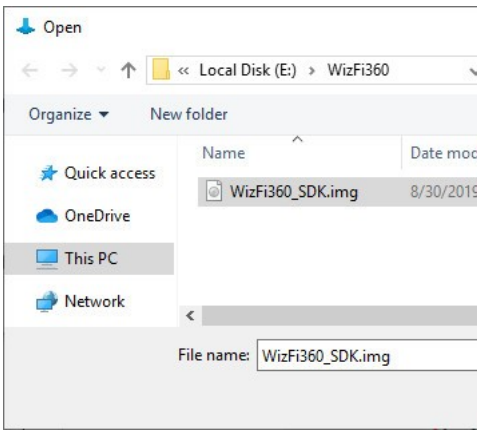
Open the WizFi Upgrade Tool and Open the Serial port of WizFi360.



#Step 3

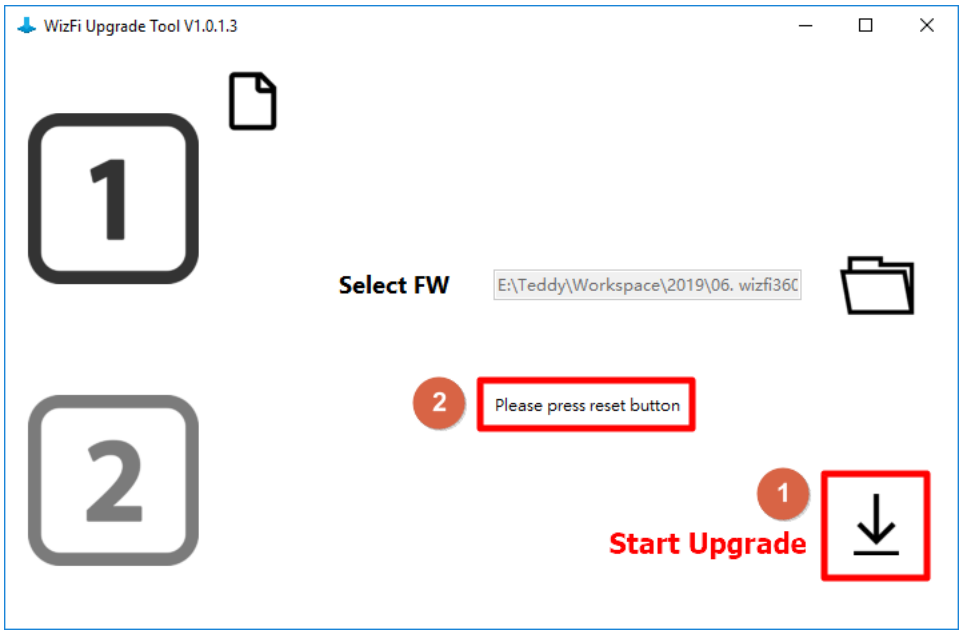
Click the folder icon to select the WizFi360 Firmware binary file. (WizFi360_SDK.img)

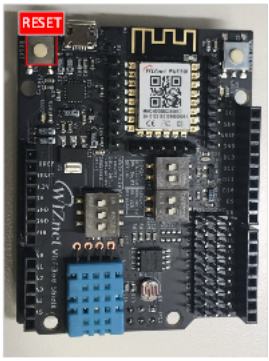




#Step 4

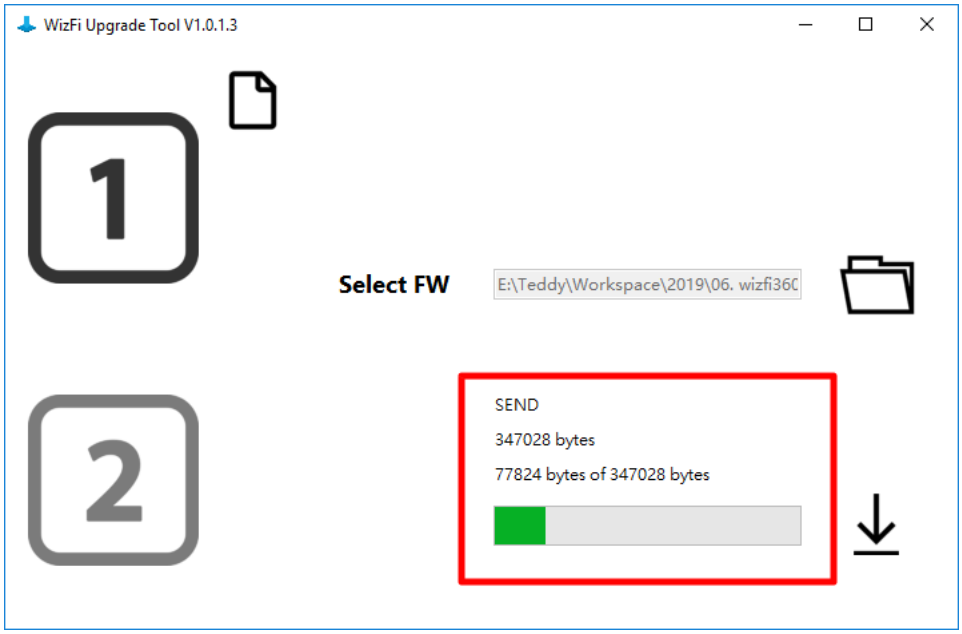
Click the Start Upgrade button (arrow icon) and check the "Please Press reset button message."

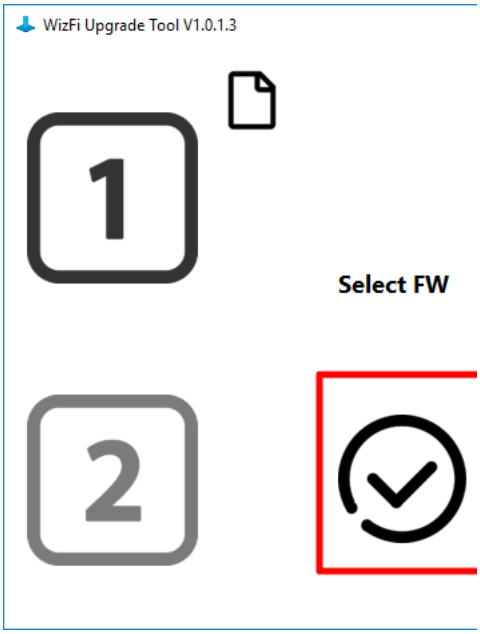




#Step 5

Firmware upload is progressed and uploaded firmware size is displayed.

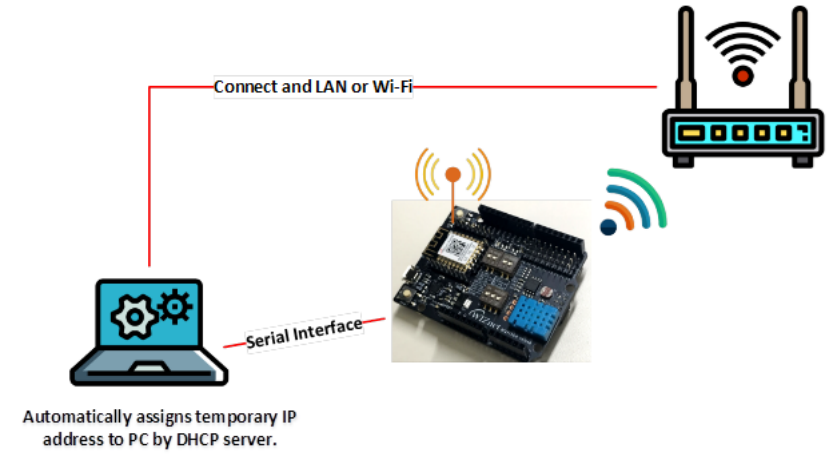




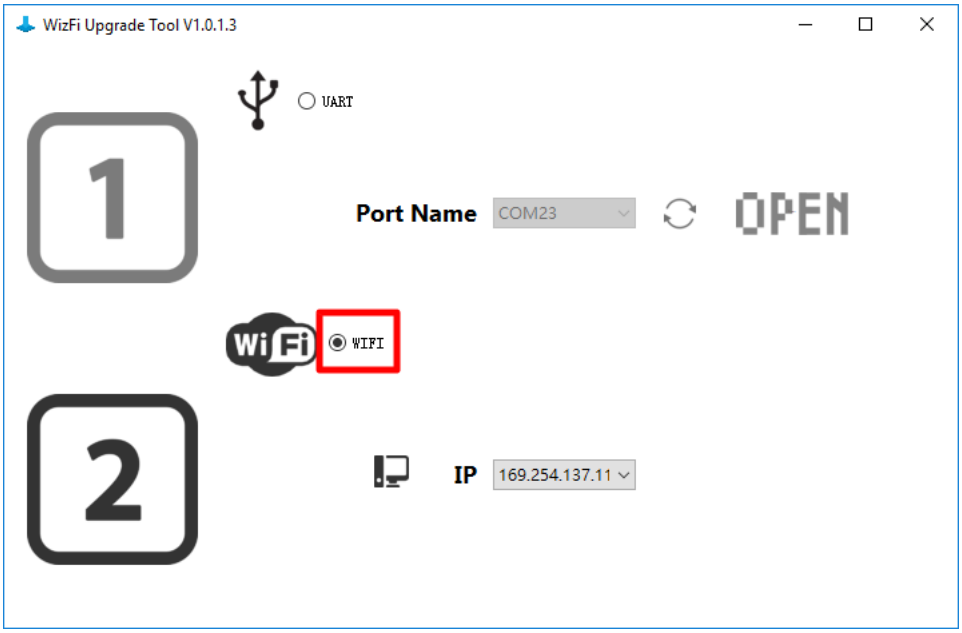
#How to upgrade using Upgrade Tool (Wi-Fi)

#Step 1

Update Firmware by communicating with Upgrade Tool through Wi-Fi. Select Firmware binary file to update in Upgrade Tool, and update firmware using AT Command (AT + CIUPDATE) in WizFi360. To use AT Command, serial communication is used UART0 or UART1. In case of UART1 use USB cable and in case of UART0 use TTL to USB module.



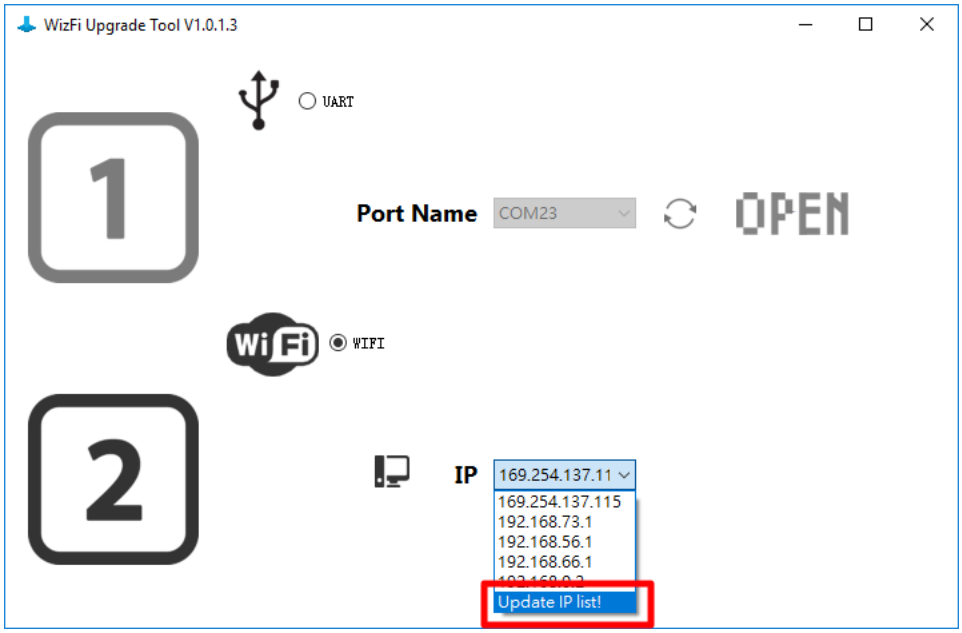
#Step 2

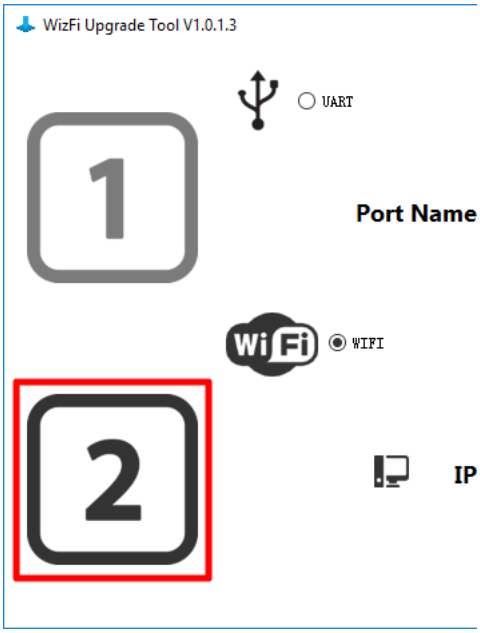


Click the WiFi option of the Upgrade Tool.

#Step 3

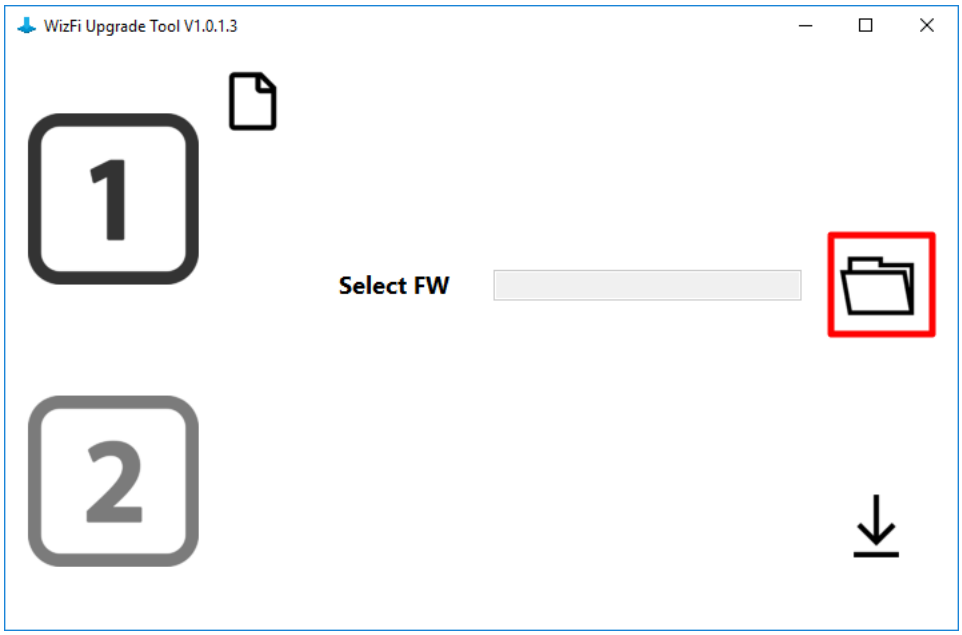
Click IP list and select IP of PC.





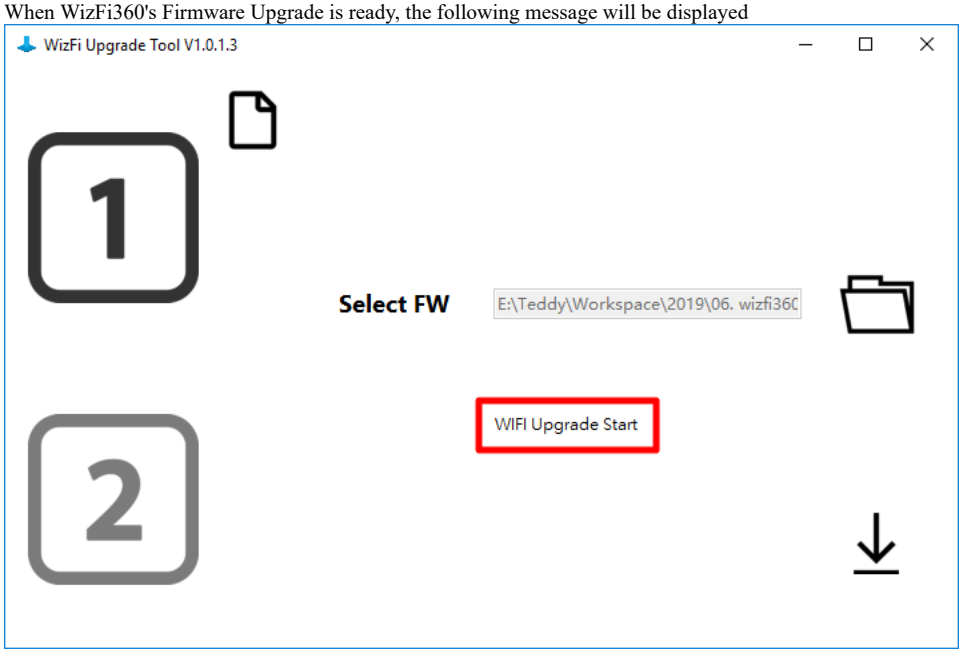
#Step 4

In Select FW, click the folder icon to select the WizFi360 Firmware binary file. (WizFi360_SDK.img)



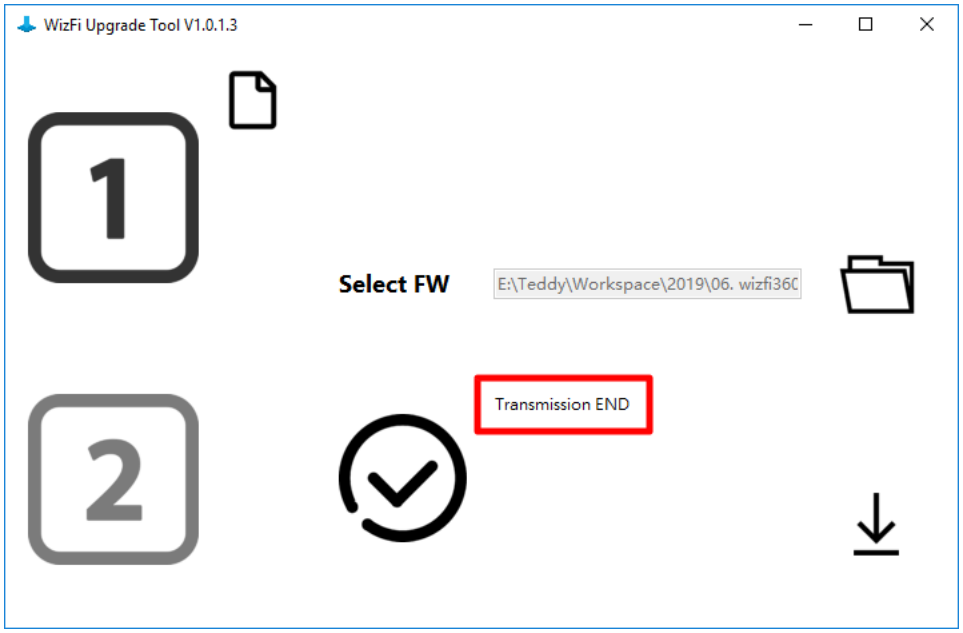
#Step 5

When WizFi360's Firmware Upgrade is ready, the following message will be displayed



#Step 6

Use the AT command to upgrade the firmware.




```
AT+CIUPDATE="http://192.168.0.2:8080/WizFi36
AT+CIUPDATE="http://192.168.0.2:8080/WizFi36
+CIPUPDATE:1
+CIPUPDATE:2
+CIPUPDATE:3
+CIPUPDATE:4

OK

ready
```

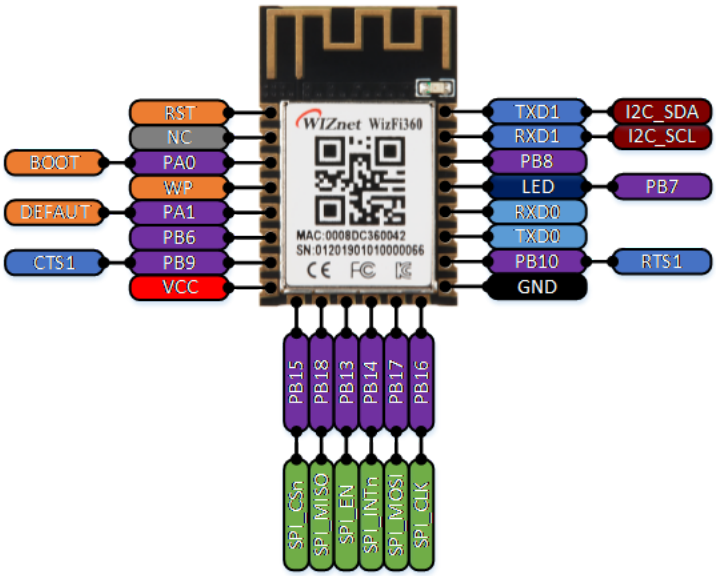
SPI Guide

#Introduction

WizFi360 operates in SPI slave mode and can be controlled via AT commands. In order to communicate with the MCU, the SPI pins must be connected and set the SPI_EN(PB13) pin to Low for SPI. Refer to Figure 1. WizFi360 Pinout to locate the SPI pins. Set the SPI_INT(PB14) pin to low when the user needs to send data from WizFi360 so that the SPI master can read the data.

#Pinout

SPI pins are from PB13 to PB17 in the below Figure WizFi360 Pinout.



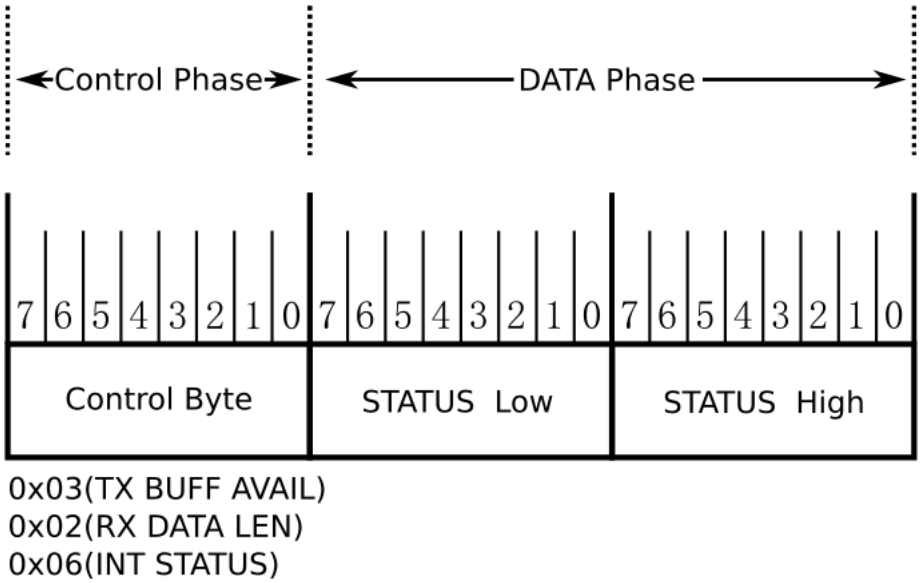
#SPI Frame Format

WizFi360 is controlled by the SPI frame format sent from the SPI master. The SPI frame is controlled by CSn and composed of SPI Control Frame, AT CMD Frame, and DATA Frame . DATA Frame composed RX DATA Frame and TX DATA Frame. Users can select the default status, buffer save size, CMD, DATA SEND, and DATA RECEIVE during the control phase.

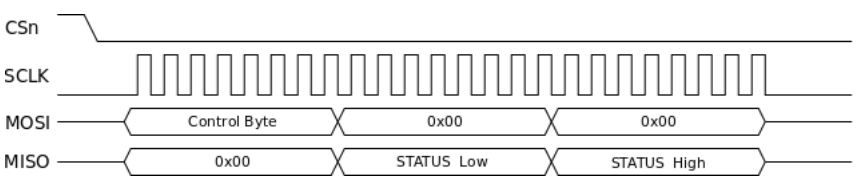
#SPI Control Frame

TX BUFF AVAIL, RX DATA LEN, and INT STATUS must be read before users write or read data into WizFi360. The SPI Control Frame sends 1Byte of control byte and reads 2Byte of status data.

- 0x03(TX BUFF AVAIL) : checks whether the peer buffer is ready to write date before transmission.
- 0x02(RX DATA LEN) : reads the data length accumulated in the peer buffer before the data is received.
- 0x06(INT STATUS) : reads the interrupt status of slave.



SPI Control Frame

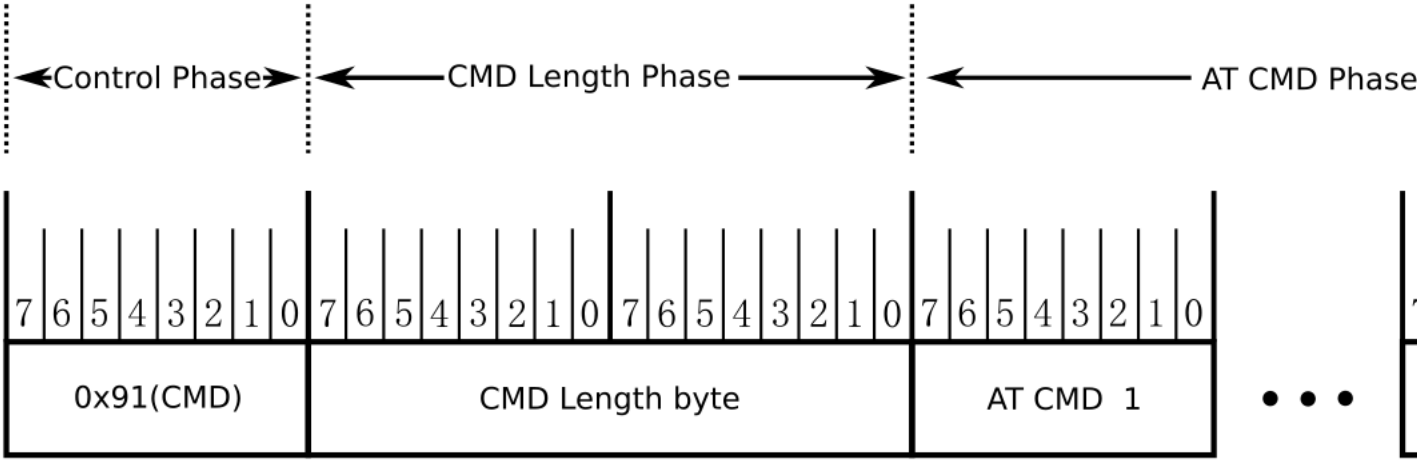


SPI Timing Graph (SPI Control Frame)

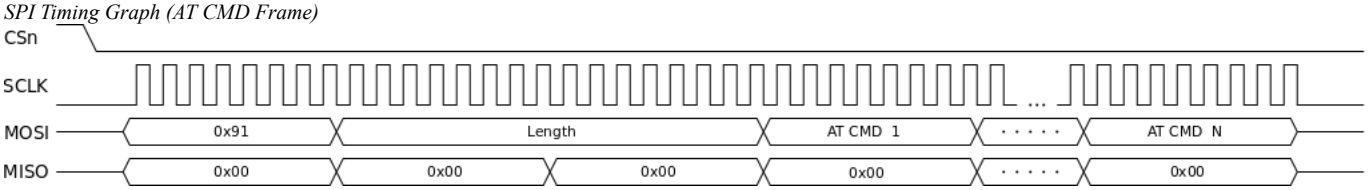
#AT CMD Frame

The AT CMD frame reads the TX BUFF AVAIL from the **SPI Control Frame** and sets the Control Byte as 0x91 during the Control Phase if 0x0002 or bit 2 is high. Then the CMD length is set in units of 4bytes and AT CMD messages are included in the data for transmission. AT CMD reply uses the **RX Data Frame** method when receiving data. Please refer to the AT instruction set for more details on AT-CMD.

AT CMD Frame



SPI Timing Graph (AT CMD Frame)

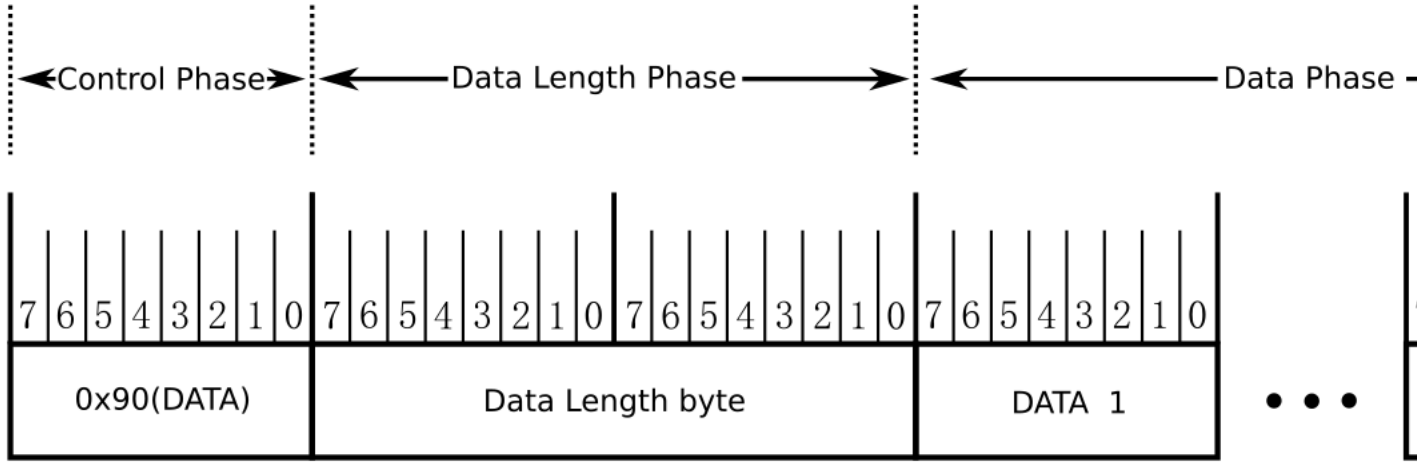


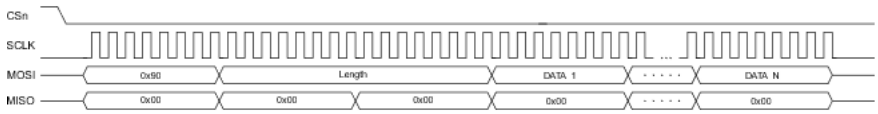
#Data Frame

#TX Data Frame

AT+CIPSEND, AT+CIPSENDEX, and AT+CIPSENDBUF must be transmitted from the **AT CMD Frame** and users must follow the next steps to prepare TCP or UDP data transmission in DATA trans mode. The TX data frame reads the TX BUFF AVAIL from the **SPI Control Frame** and sets the Control Byte as 0x90 during the Control Phase if 0x0002 or bit 2 is high. Then the CMD length is set in units of 4bytes and DATA messages are included in the data for transmission. DATA reply uses the **RX Data Frame** method when receiving data.

TX Data Frame

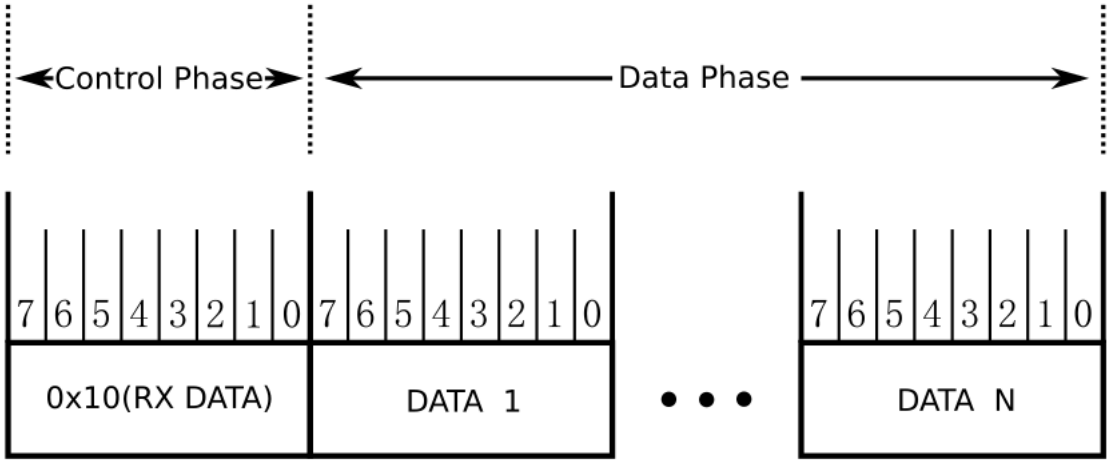




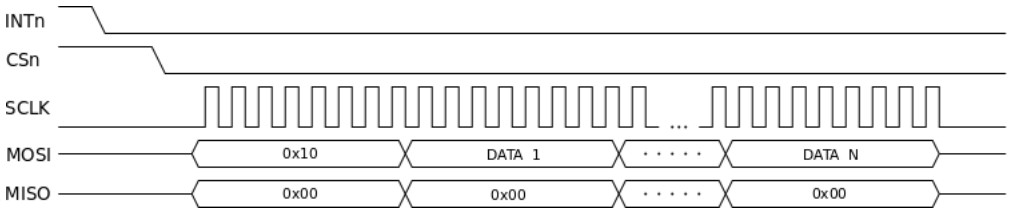
SPI Timing Graph (TX DATA Frame)

#RX Data Frame

When a reply or data is received after the **AT CMD Frame** is transmitted, check whether the interrupt pin is low or not. If the interrupt pin is low, users read the value of INT STATUS using the SPI Control Frame. If the value of INT STATUS is 0x0002 or bit 2 is high, users read the value of RX DATA LEN using the **SPI Control Frame**. And if the value of RX Data Len is not zero, users set the Control Byte as 0x10 during the control phase and read data. The total data count is the value of RX DATA LEN.



RX Data Frame

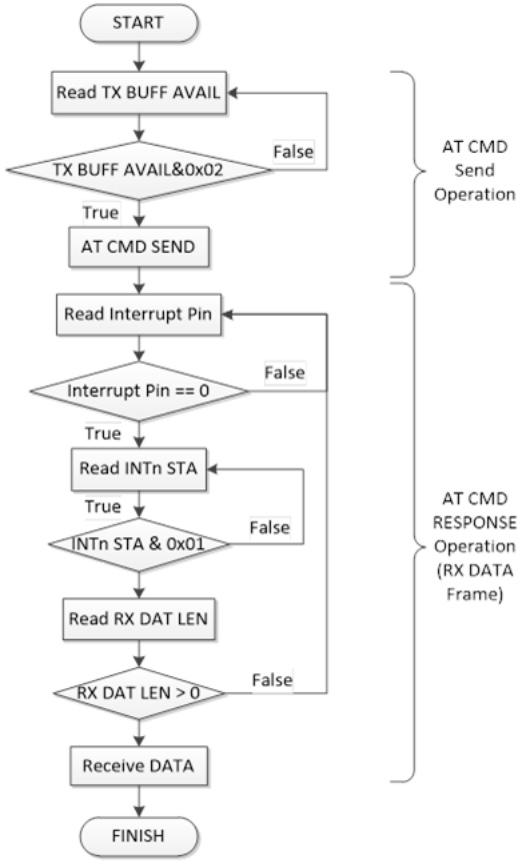


SPI Timing Graph (RX Data Frame)

#Operation

#AT CMD Operation

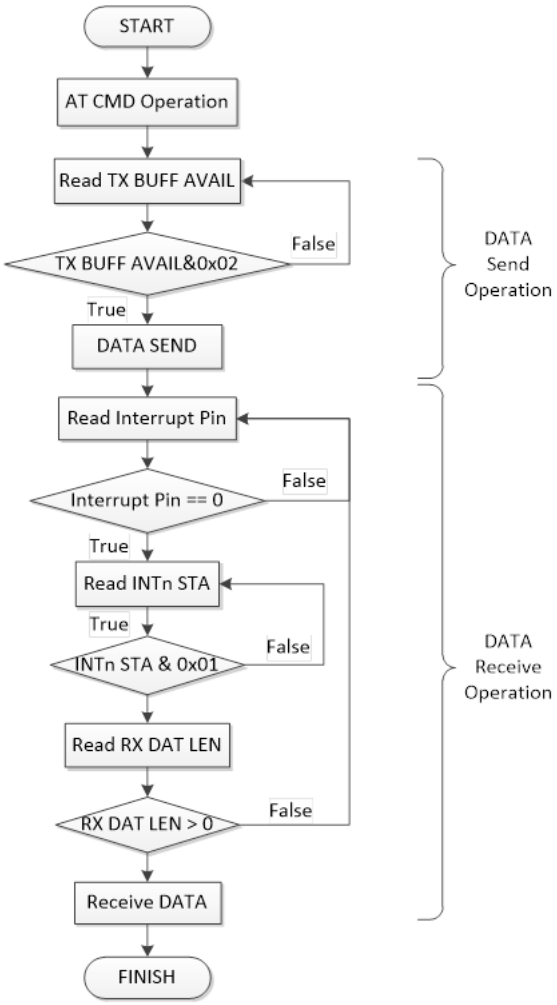
Use AT CMD to set WizFi360 or follow the steps below to set SEND mode and request data.



AT CMD Operation Flowchart

#Data Operation

Data can be sent if AT+CIPSEND, AT+CIPSENDEX, OR AT+CIPSENDBUF is entered in AT CMD or in DATA TRANS mode.



DATA Operation Flowchart

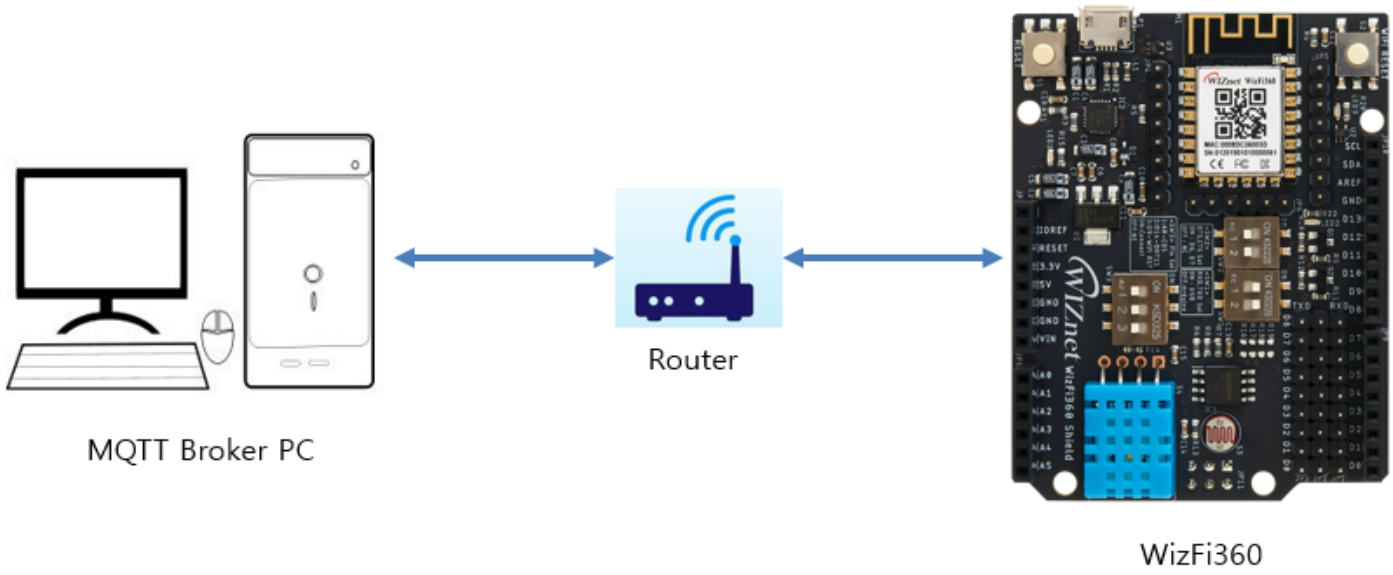
MQTT Client

#Introduction

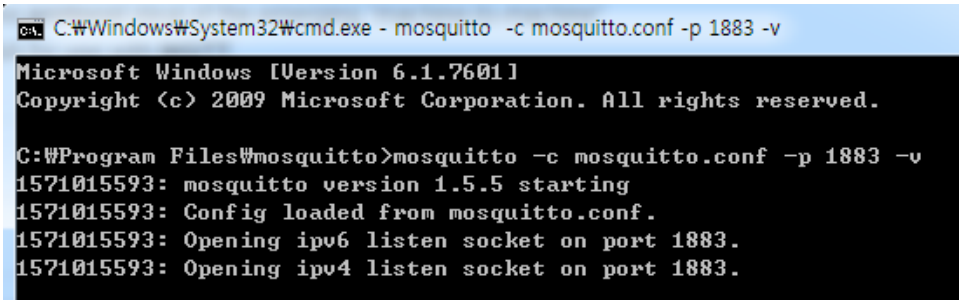
MQTT is a light weight messaging protocol for use on top of the TCP/IP protocol. It is designed for connections with remote locations where a “small code footprint” is required or the network bandwidth is limited.It is a communication protocol which almost can link all networked objects with the external, and it is used as a sensor. https://en.wikipedia.org/wiki/MQTT

#Local Test Environment

Brokers acting as Mqtt Servers mainly use the Mosquitto program. Download and install the Mosquitto tool from the following link. https://mosquitto.org Connect your Mosquitto-installed PC and WizFi360 to the same Router (AP).



Open a command prompt window and navigate to the path where Mosquitto is installed. (cd C: \ Program Files \ mosquitto) Then run the following command: (mosquitto -c mosquitto.conf -p 1883 -v) 1883 is the port number, and Mqtt usually uses the 1883 port number.



#WizFi360 AT Commands

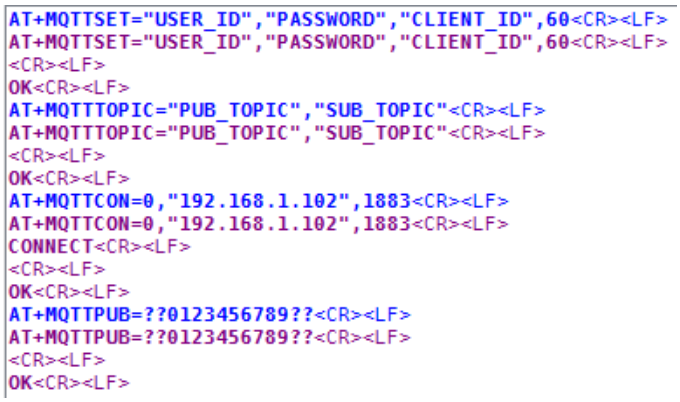
Use following commands to connect to Wi-Fi AP.

#Station Mode

AT Command	Terminal

AT Command	Terminal
AT AT+CWMODE_CUR=1 AT+CWDHCP_CUR=1,1 AT+CWLAP AT+CWJAP_CUR=”ssid”,”password” AT+CIPSTA_CUR?	

#MQTT Connect & Data Publish

AT Command	Terminal
AT+MQTTSET="USER_ID","PASSWORD","CLIENT_ID",60 AT+MQTTTOPIC="PUB_TOPIC","SUB_TOPIC" AT+MQTTCON=0,"192.168.1.102",1883 AT+MQTTPUB=”0123456789”	

Hardware Design Guide

#Overview

If you are designing hardware using the WizFi360 please refer to this document.

This document includes a reference circuit diagram and a PCB guide.

#Pin definition

Figure 1. WizFi360 Pinout

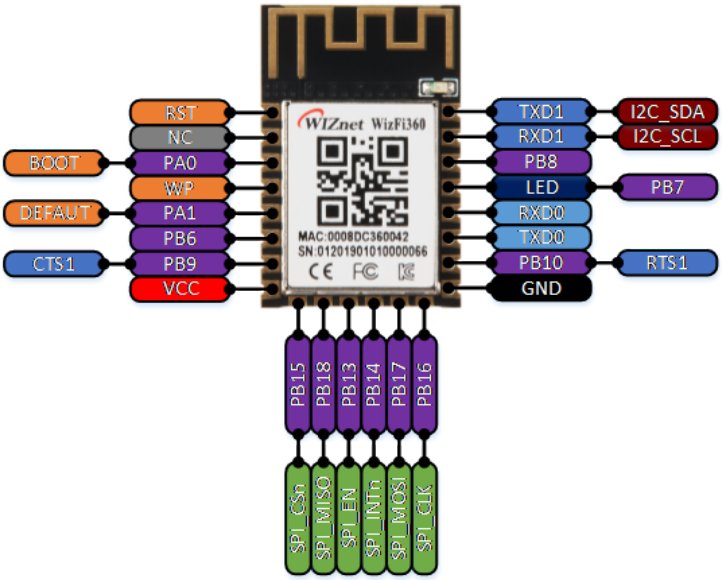


Table 1. Pin Definitions

Pin Name	Type	Pin Function
RST	I	Module Reset Pin (Active Low)
NC	-	WizFi360-PA Reserved
ANT	O	WizFi360-CON ANT pin for external antenna
PA0	I/O	BOOT Pin (Active low) When power on or reset is low, it operates in Boot mode. In the normal operating mode, this pin can be controlled by AT command.
WP	I	WAKEUP Pin (Active High) If the wake-up pin is high in Standby mode, the WizFi360 is reset to the normal operating mode.
PA1	I	Pull down over 3s for taking effect. UART1's current parameter changes to default value (please refer to the AT+UART_CUR command in WizFi360 AT command manual).
PB6	I/O	This pin can be controlled by AT command.
PB9	I	CTS Pin of UART1 If you don't use the CTS function, this pin can be controlled by AT command.
VCC	P	Power Pin (Typical Value 3.3V)
PB15	I/O	This pin can be controlled by AT command.
PB18	I/O	This pin can be controlled by AT command.
PB13	I/O	This pin can be controlled by AT command.
PB14	I/O	This pin can be controlled by AT command.
PB17	I/O	This pin can be controlled by AT command.
PB16	I/O	This pin can be controlled by AT command.
GND	I/O	Ground Pin
PB10	O	RTS Pin of UART1 If you don't use the RTS function, this pin can be controlled by AT command.
TXD0	O	TXD Pin of UART0
RXD0	I	RXD Pin of UART0
PB7	O	LED Light output (Active low). Go to Low while each TX/RX packet and then back to high. Note: It has been connected to onboard LED for WizFi360-PA
PB8	I/O	This pin can be controlled by AT command.
RXD1	I	RXD Pin of UART1
TXD1	O	TXD Pin of UART1

important

UART1 is used for AT command and data communication. UART0 is used for debugging and firmware upgrade.

#Initial value of GPIO Pins

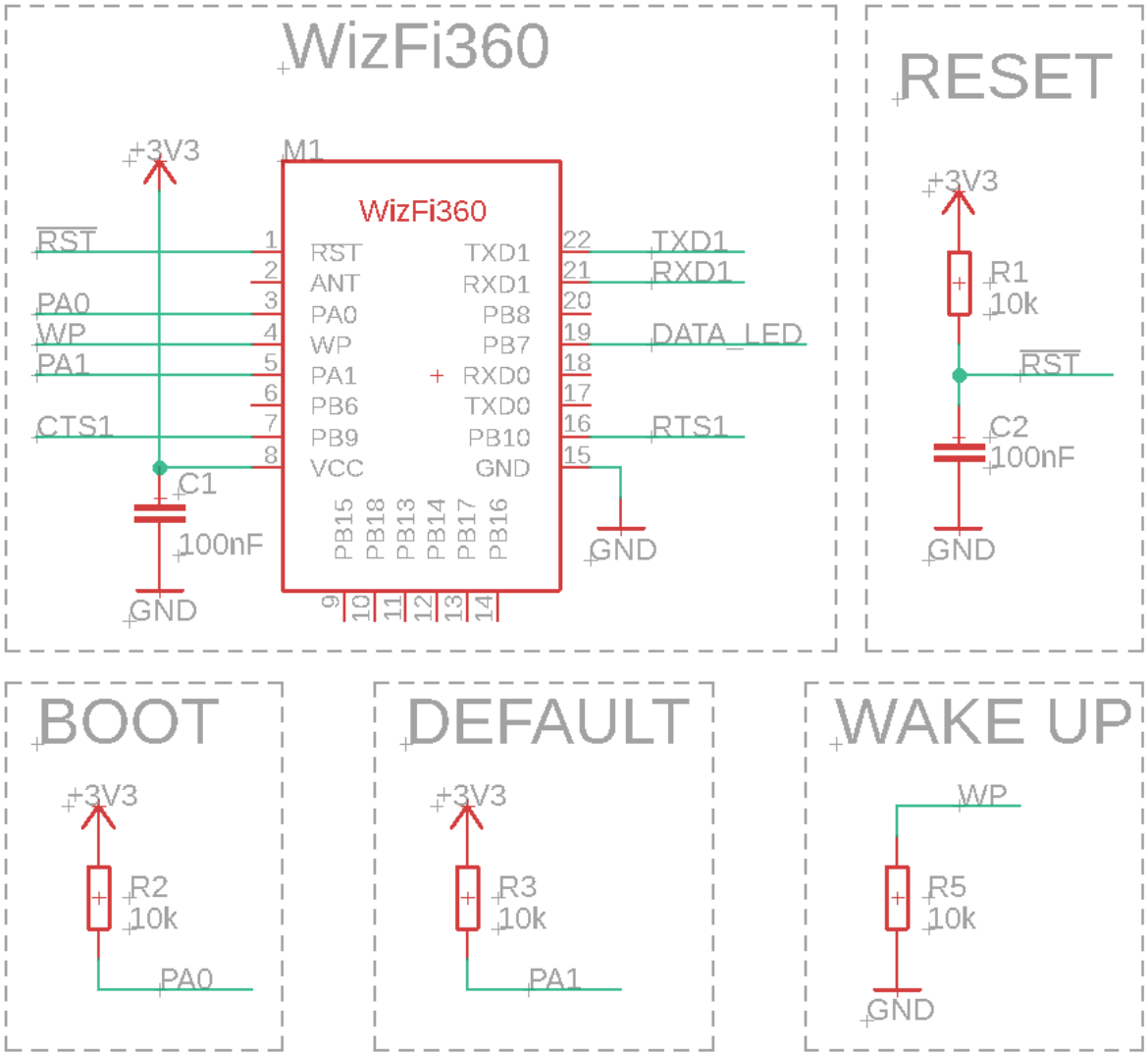
This is the initial value of GPIO when using AT command to use GPIO on WizFi360.

Pin Name	Type	Value	Pull up/down
PA0	I/O	High	Pull up
PB6	I/O	Low	Pull down
PB9	I/O	Low	Pull down
PB15	I/O	High	Pull down
PB18	I/O	High	Pull down
PB13	I/O	High	Pull down
PB14	I/O	High	Pull down
PB17	I/O	High	Pull down
PB16	I/O	High	Pull down
PB10	I/O	Low	Pull down
PB07	I/O	High	Pull down
PB08	I/O	High	Pull down

#Circuit

#System

The WizFi360 has a very simple circuit. You can connect power to the WizFi360 and send and receive data through UART1. Please pay attention to four pins.

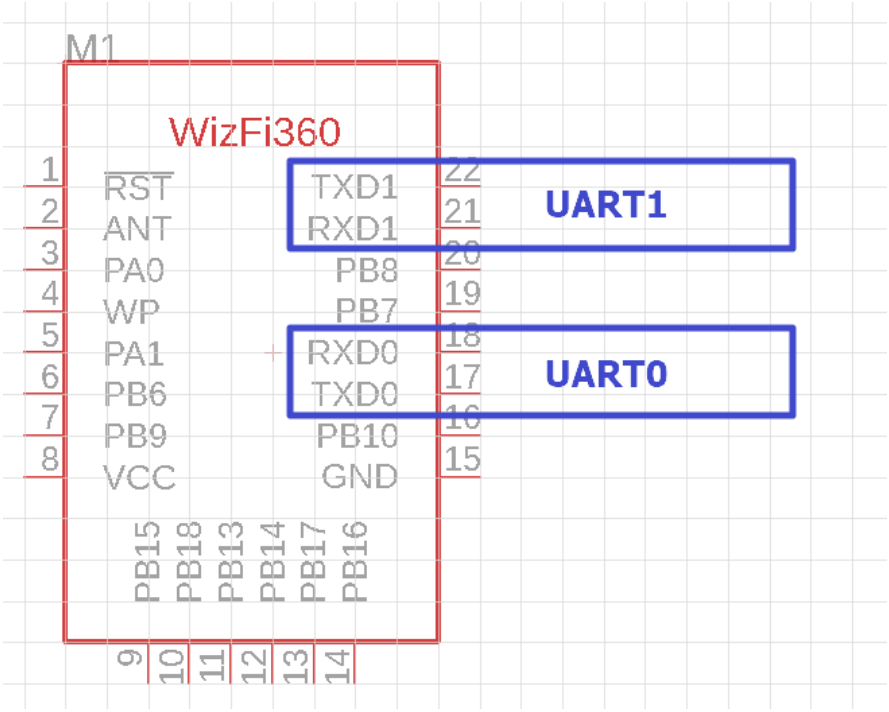


- **Reset**
  Reset circuit offers to design with RC circuit. WizFi360 reset automatically by low level power. If RESET pin controlled by external circuit, the WizFi360 will reset when the level is below 2.0V. The low level needs to last more than 100µs.
- **PA0**
  PA0 circuit offers to design 10k pull-up. PA0 is used as a boot pin, but it's use unlikely for normal users. This pin is used at the factory stage. (Module production)
- **PA1**
  PA1 circuit offers to design 10k pull-up. If PA1 is Low for 3 seconds, UART1's current parameter changes to default value (please refer to the AT+UART_CUR command in WizFi360 AT command manual).
- **WP**
  WP circuit offers to design user configuration. You must control this pin if you are using standby mode. If this pin is high in Standby mode, the WizFi360 is reset to the normal operating mode.

#Power

WizFi360 requires the use of a power supply capable of supplying 3.0V to 3.6V and more than 500mA. Because WizFi360 operates normally from 3.0V to 3.6V, it consumes up to 230mA of instantaneous current. The wiring width should not be less than 30mil. The power stabilizing capacitor (100nF) should be placed close to the VCC pin.

#UART

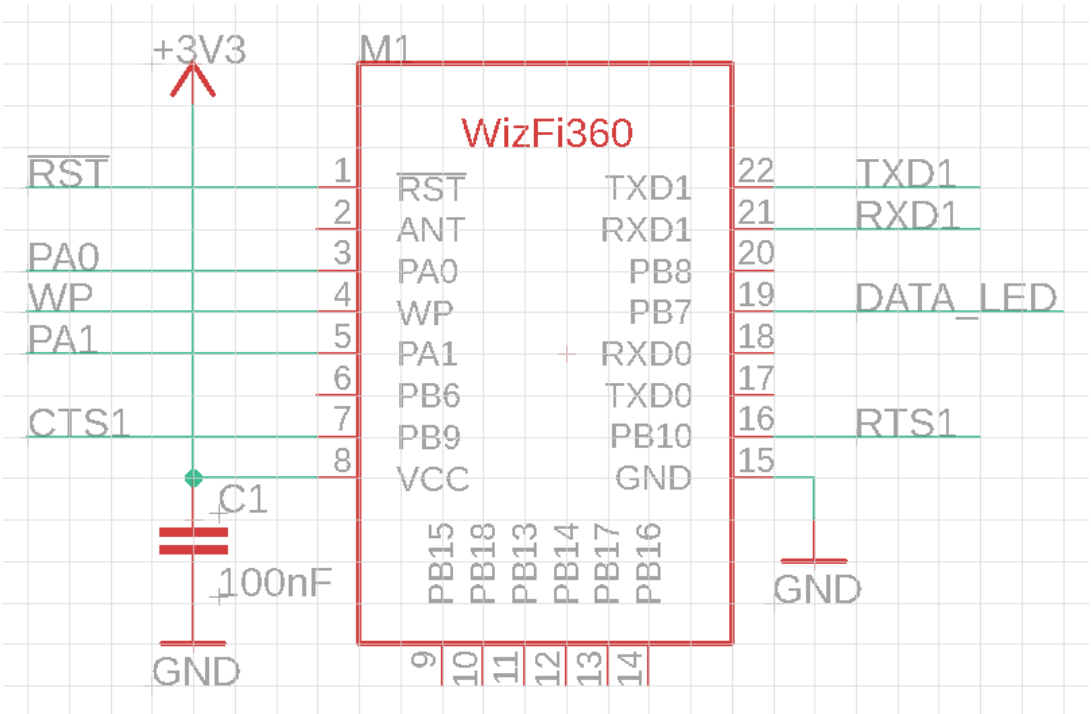


- UART1
  UART1 is the main communication UART. AT command communication is possible with UART1 and data communication is possible.
- UART0
  UART0 is not available to normal users. This UART is used at the factory stage (Module production) and intended for internal firmware developers of the WizFi360.

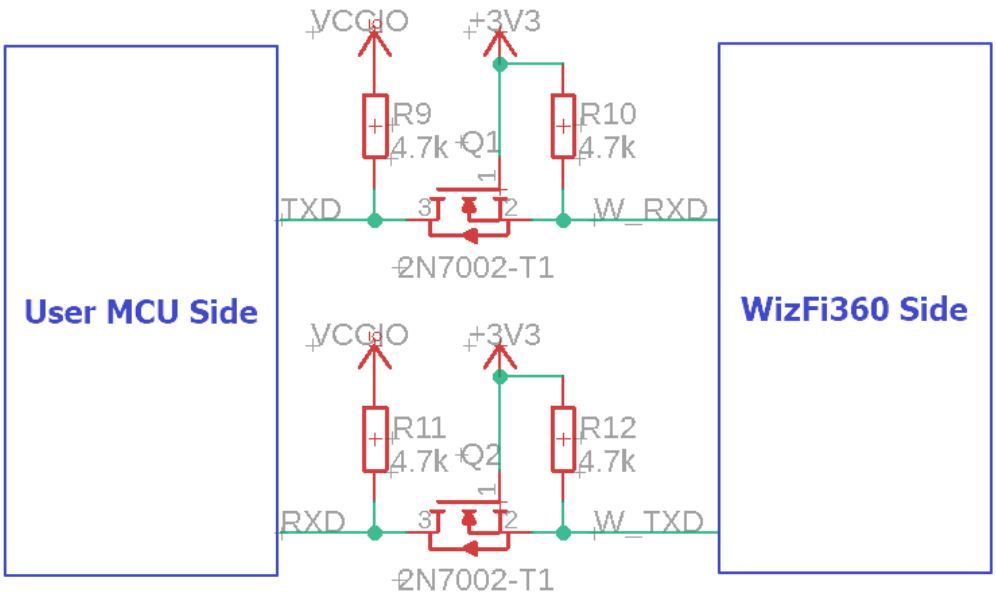
#ETC

This session is an additional circuit guide for using the WizFi360. You don't have to keep this session. But if you need it, you design it.

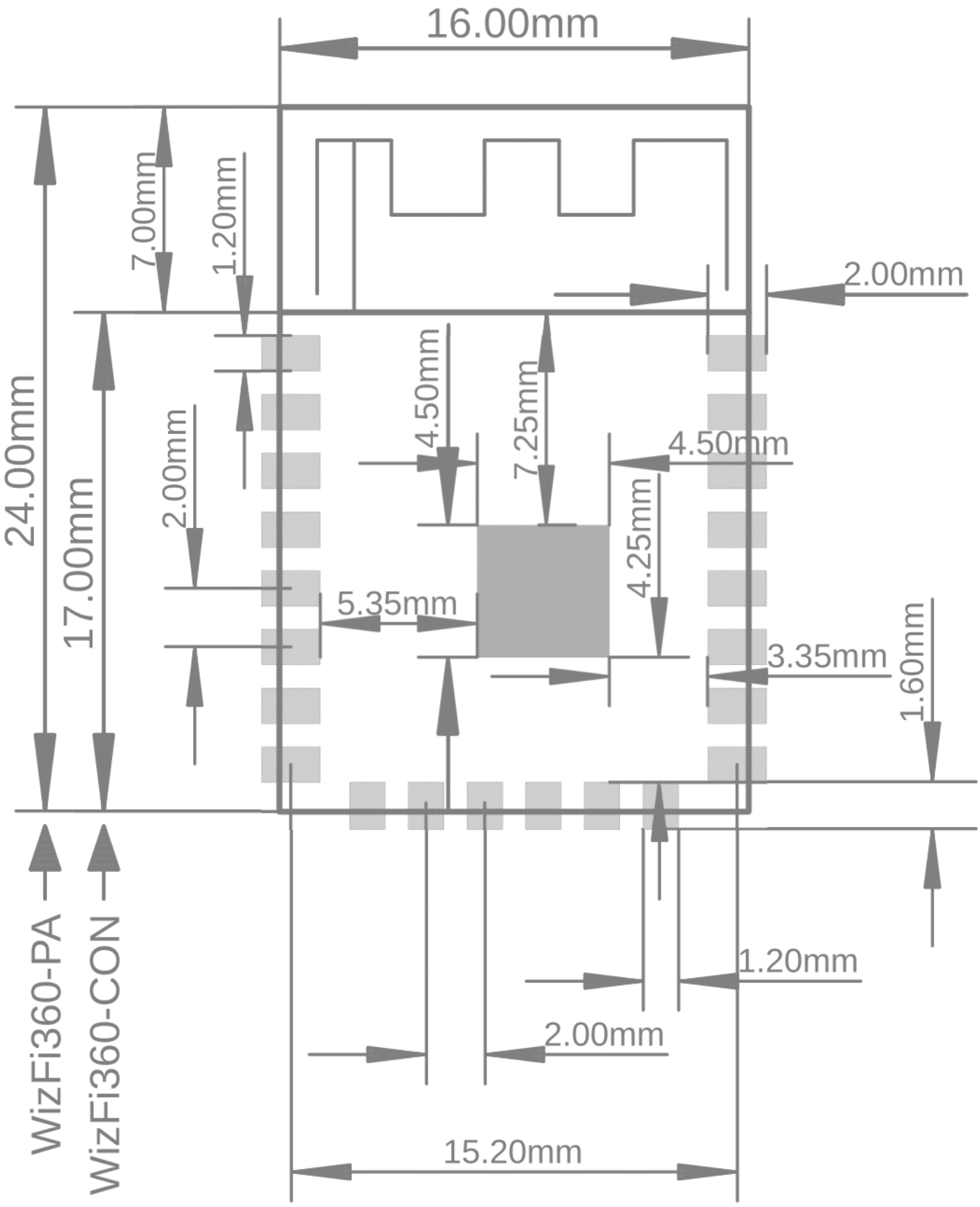
- **UART Flow Control**
  If you want to use UART Flow Control, you need to design a circuit as shown in Figure 3. PB9 is CTS1, PB10 is RTS1.



- **UART Level Shifter**
  The UART voltage on the WizFi360 is 3.3V. However, your MCU may not have a voltage of 3.3V. If so you need a Level Shifter to connect the WizFi360 to your MCU. You can design a Level Shifter circuit by referring to Figure 4. Connect your MCU's UART voltage to the VCCIO at Figure 4.

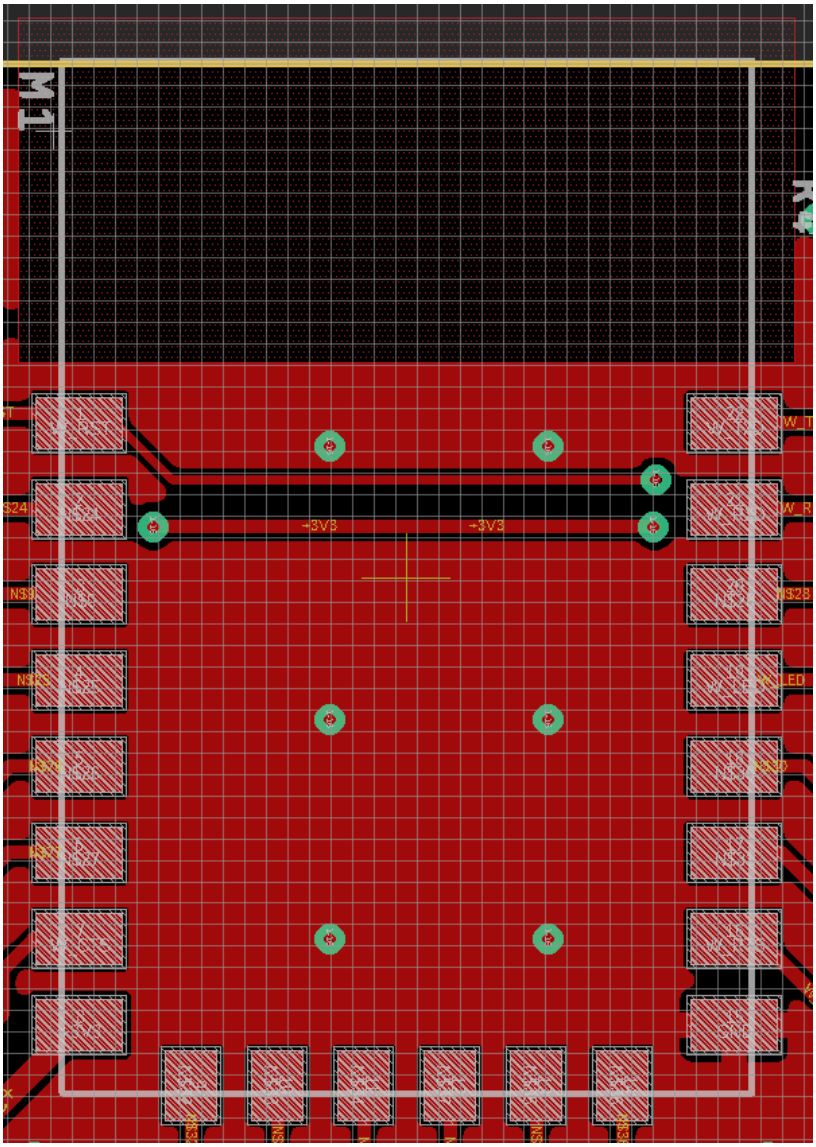


#PCB Footprint

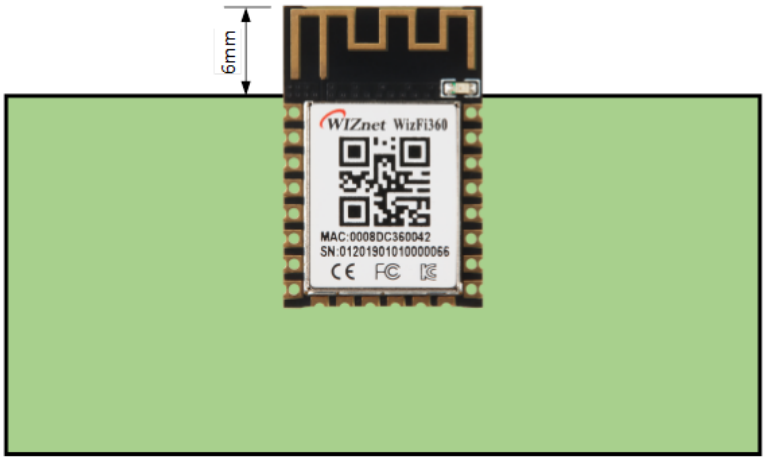


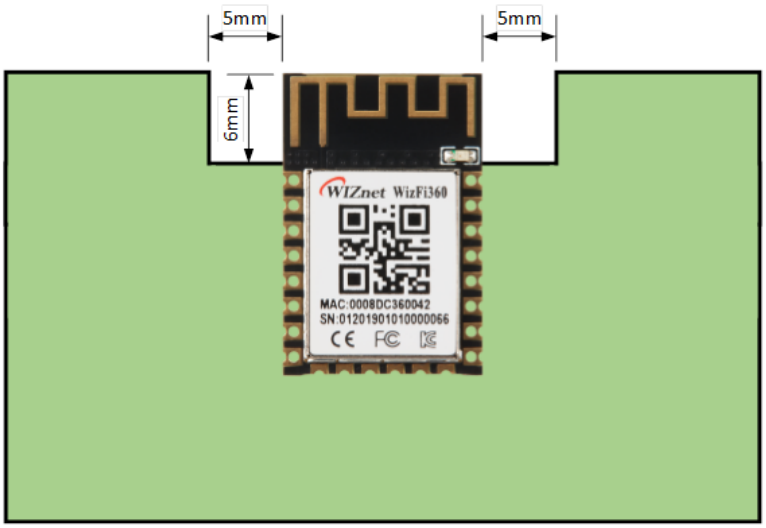
#PCB Layout

- Power wiring width should not be less than 30mil.
- Except for the antenna portion of WizFi360, the bottom layer of the shield can must have a GND plane.



- Figures. 6 and Figures. 7 are 2 antenna placement which can best performance of antenna. We suggest customers to choose one of these 2 modes to design the placement. For the second placement mode, PCB antenna should be at least 5.0mm from both sides of the bottom board.




5mm
5mm
6mm
WIZnet WizFi360
MAC:0008DC360042
SN:01201901010000066

HTTP Client using WizFi360

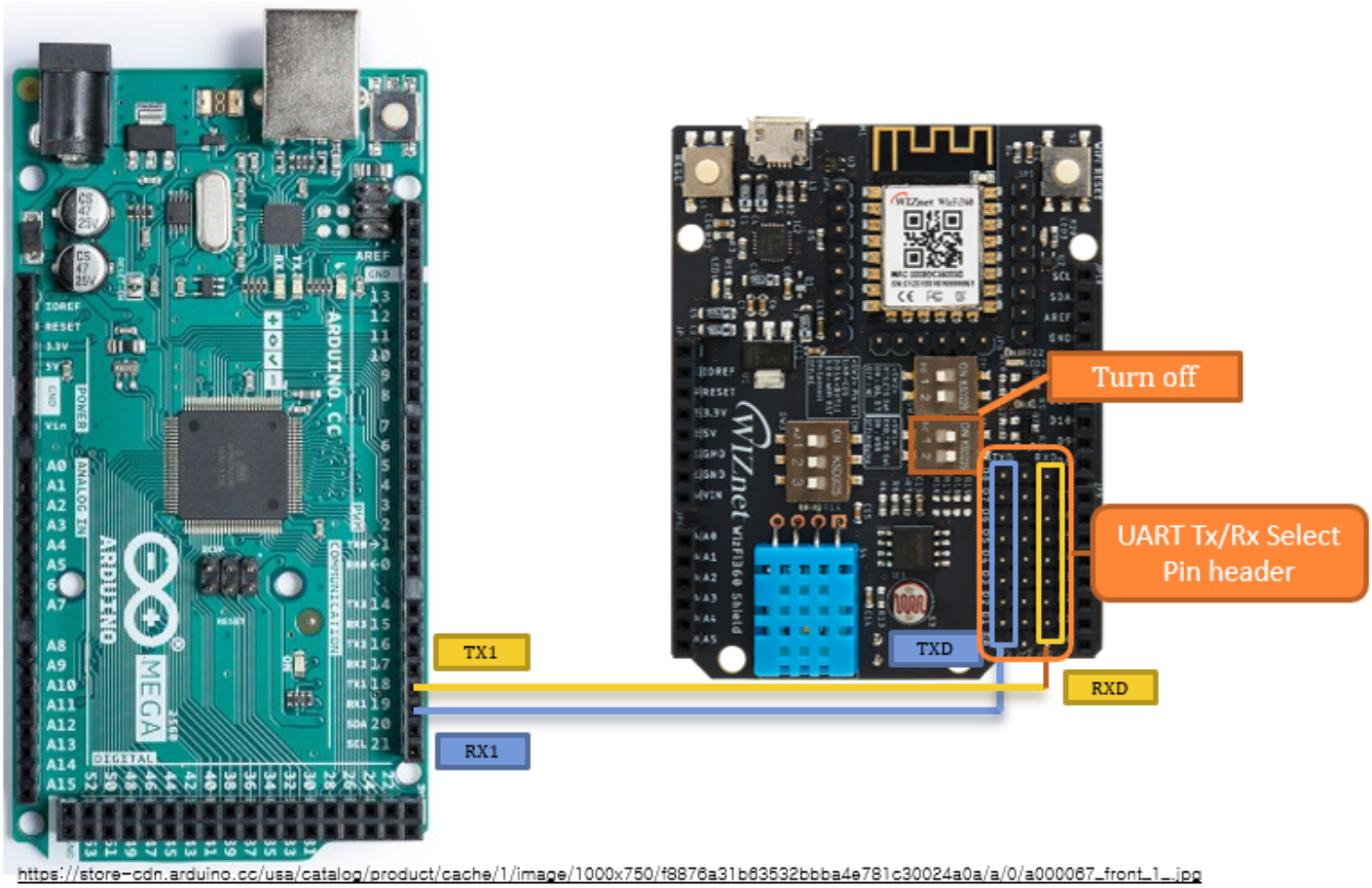
#Introduction

HTTP(Hypertext transfer protocol) is a communication protocol for exchanging data between server and client on the Web. HTTP exchanges data in the form that the client sends the request and the server sends the reply to the request. At this time, Server transmits response such as HTML, JSON, etc., and uses port 80. HTTP is a TCP-based protocol. WizFi360 can communicate the HTTP communication by creating TCP Server. The HTTP protocol must be implemented by the user.

#Hardware environment

This document uses Arduino Mega2560 and WizFi360-EVB-Shield. To communicate with the WizFi360-EVB-Shield using UART1 in the Arduino Code, connect the TX1 and RX1 pins of Arduino to the RXD and TXD pins of the WizFi360-EVB-Shield. In the WizFi360-EVB-Shield, change the RXD / TXD Selector to OFF to enable UART communication via the pin, not the USB.

You can also use D0 ~ D8 Pin as TXD / RXD Pin by using UART Select Pin header. If you use it, you can connect without Wiring by placing WizFi360-EVB-Shield on Arduino.



https://store-cdn.arduino.cc/usa/catalog/product/cache/1/image/1000x750/f8876a31b63532bbba4e781c30024a0a/a/0/a000067_front_1_.jpg

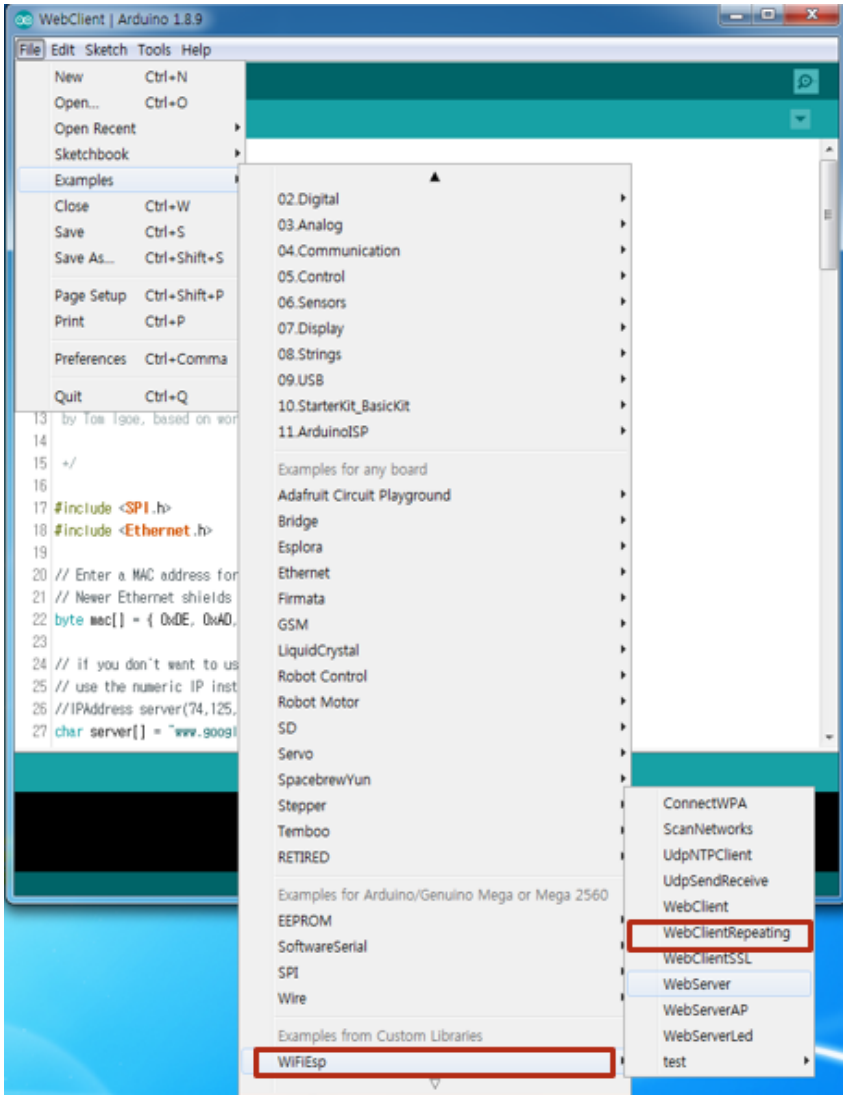
#Library download

Download the library from Github and copy it in the “libraries” folder inside your sketchbook.

#Execution

#Open example

Launch Arduino and Open the WebClientRepeating example as shown below.



#Code

Update your AP credentials (SSID and password) to connect WizFi360. Serial1 is the serial communicating with Arduino and WizFi360 and should have the same baud rate as WizFi360. The factory default baud rate for WizFi360 is 115200, and if you have not changed it, enter 115200. Enter the address of the HTTP server to connect to server. You can enter ip address or domain address. In this case, connect to arduino.cc.

```
19 char ssid[] = "wizms1";          // your network SSID (name)
20 char pass[] = "maker0701";       // your network password
21 int status = WL_IDLE_STATUS;     // the Wifi radio's status
22
23 char server[] = "arduino.cc";
24
25 unsigned long lastConnectionTime = 0;         // last time you connecte
26 const unsigned long postingInterval = 10000L; // delay between updates,
27
28 // Initialize the Ethernet client object
29 WiFiEspClient client;
30
31 void setup()
32 {
33   // initialize serial for debugging
34   Serial.begin(115200);
35   // initialize serial for ESP module
36   Serial1.begin(115200);
37   // initialize ESP module
38   WiFi.init(&Serial1);
```

In this guide, Arduino (WizFi360EVB) connects to Web Server and sends the following data. Request /ascillogo.txt using the GET method.

```
// if there's a successful connection
if (client.connect(server, 80)) {
  Serial.println("Connecting...");

  // send the HTTP PUT request
  client.println(F("GET /asciilogo.txt HTTP/1.1"));
  client.println(F("Host: arduino.cc"));
  client.println("Connection: close");
  client.println();
```

#Result

Receive the following data from the server.

```
[WiFiEsp] Connecting to arduino.cc
Connecting...
HTTP/1.1 200 OK
Server: nginx
Date: Fri, 26 Jul 2019 05:23:50 GMT
Content-Type: text/plain
Content-Length: 2263
Last-Modified: Wed, 02 Oct 2013 13:46:47 GMT
Connection: close
Vary: Accept-Encoding
ETag: "524c23c7-8d7"
Accept-Ranges: bytes


          `::::,`                    .::::.
        .::::::::::::`              ::::::::::::     TM
      `::::::::::::::::`           ::::::::::::::::
     ::::::::::::::::::::        `::::::::::::::::::::
    ::::::::::::::::::::::       .::::::::::::::::::::::
   ::::::::::`   `::::::::::     ,::::::::::.`   .::::::::::
  .::::::::,         ::::::::   .::::::::          ::::::::
  ::::::::             ::::::::  ::::::::,           ::::::::.
 ,::::::               ::::::::.::::::::`             ::::::::
 ::::::.                ::::::::::::`      ```       ::::::::`
 ::::::                  ::::::::::,       :::       .::::::
`::::::                  `:::::::::        :::        ::::::
,::::::`    `,,,,,,,,,     ::::::::      ..,:::,,,     ::::::
::::::`    .::::::::::       ::::::,     ::::::::::     ::::::
::::::`    .::::::::::      `::::::      ::::::::::     ::::::
.::::::.                    ::::::::.        :::        ::::::
 ::::::                    ::::::::::        :::        ::::::
 ::::::                   .::::::::::       :::       ::::::,
 ::::::::                `::::::::::::                 ::::::
 `::::::,              .:::::::::::::::               ::::::::
  ::::::::           ::::::::.  ::::::::            ::::::::
   ::::::::`       .::::::::,    ::::::::         ::::::::
    ::::::::::,:::::::::::::      ::::::::::::,::::::::::
    `::::::::::::::::::::.         ::::::::::::::::::::
      ::::::::::::::::::           ::::::::::::::::::
```

HTTP Server using WizFi360

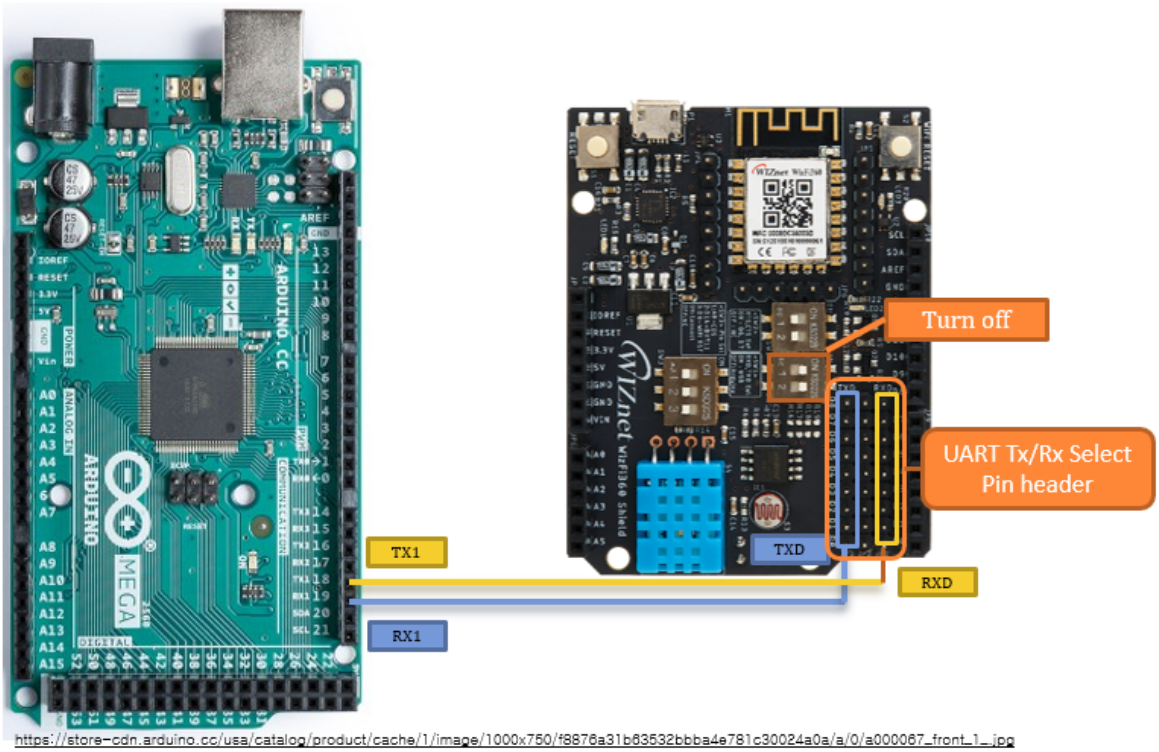
#Introduction

HTTP(Hypertext transfer protocol) is a communication protocol for exchanging data between server and client on the Web. HTTP exchanges data in the form that the client sends the request and the server sends the reply to the request. At this time, Server transmits response such as HTML, JSON, etc., and uses port 80. HTTP is a TCP-based protocol. WizFi360 can communicate the HTTP communication by creating TCP Server. The HTTP protocol must be implemented by the user.

#Hardware environment

This document uses Arduino Mega2560 and WizFi360-EVB-Shield. To communicate with the WizFi360-EVB-Shield using UART1 in the Arduino Code, connect the TX1 and RX1 pins of Arduino to the RXD and TXD pins of the WizFi360-EVB-Shield. In the WizFi360-EVB-Shield, change the RXD / TXD Selector to OFF to enable UART communication via the pin, not the USB.

You can also use D0 ~ D8 Pin as TXD / RXD Pin by using UART Select Pin header. If you use it, you can connect without Wiring by placing WizFi360-EVB-Shield on Arduino.



https://store-cdn.arduino.cc/usa/catalog/product/cache/1/image/1000x750/f8876a31b63532bbba4e781c30024a0a/a/0/a000067_front_1_.jpg

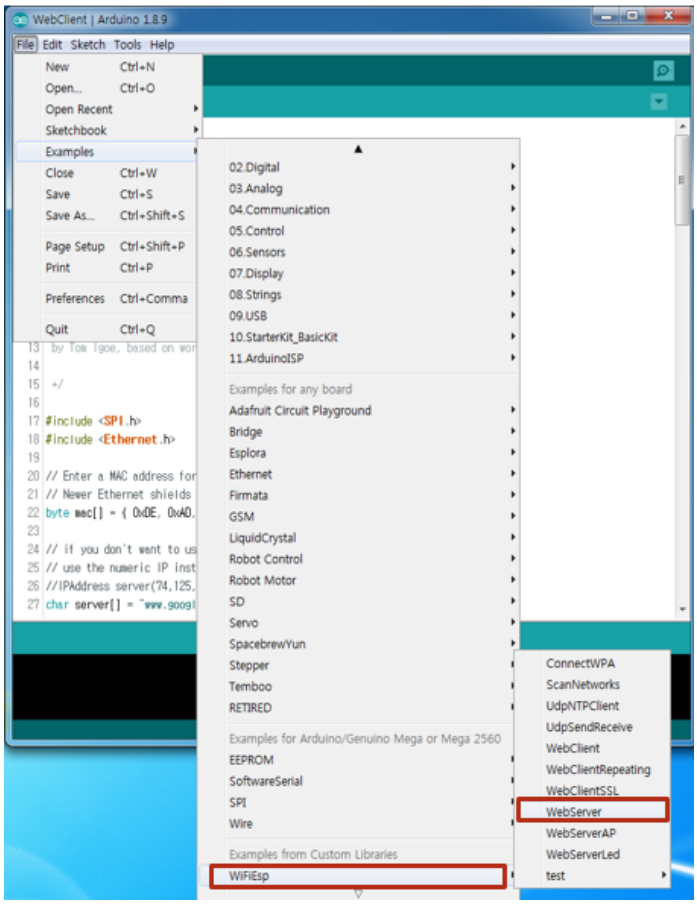
#Library download

Download the library from Github and copy it in the “libraries” folder inside your sketchbook.

#Execution

#Open example

Launch Arduino and Open the WebServer example as shown below.



#Code

Update your AP credentials (SSID and password) to connect WizFi360. Serial1 is the serial communicating with Arduino and WizFi360 and should have the same baud rate as WizFi360. The factory default baud rate for WizFi360 is 115200, and if you have not changed it, enter 115200. Enter the address of the HTTP server to connect to server. You can enter ip address or domain address. In this case, connect to arduino.cc.

```
// Emulate Serial1 on pins 6/7 if not present
#ifndef HAVE_HWSERIAL1
#include "SoftwareSerial.h"
SoftwareSerial Serial1(6, 7); // RX, TX
#endif

char ssid[] = "wizms1";          // your network SSID (name)
char pass[] = "maker0701";       // your network password
int status = WL_IDLE_STATUS;     // the Wifi radio's status
int reqCount = 0;                // number of requests received

WiFiEspServer server(80);


void setup()
{
  // initialize serial for debugging
  Serial.begin(115200);
  // initialize serial for ESP module
  Serial1.begin(115200);
  // initialize ESP module
  WiFi.init(&Serial1);

```

Arduino (WizFi360-EVB-Shield) transmits the following HTML data when Web client connects. If you want to send other data, you can modify it below.

WebServer | Arduino 1.8.10

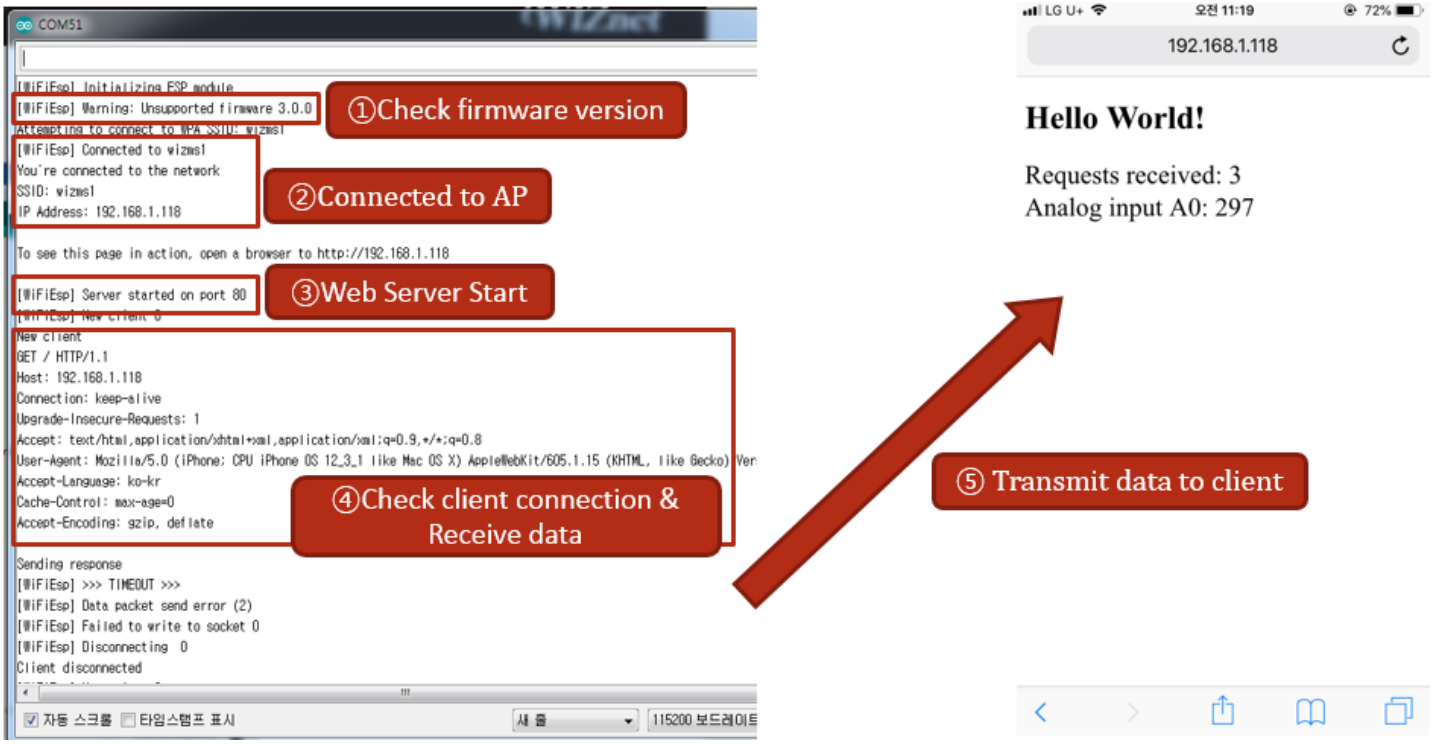
File Edit Sketch Tools Help

WebServer

```
82          client.print(
83            "HTTP/1.1 200 OK\r\n"
84            "Content-Type: text/html\r\n"
85            "Connection: close\r\n"  // the connection will be c
86            "Refresh: 20\r\n"        // refresh the page automat
87            "\r\n");
88          client.print("<!DOCTYPE HTML>\r\n");
89          client.print("<html>\r\n");
90          client.print("<h1>Hello World!</h1>\r\n");
91          client.print("Requests received: ");
92          client.print(++reqCount);
93          client.print("<br>\r\n");
94          client.print("Analog input A0: ");
95          client.print(analogRead(0));
96          client.print("<br>\r\n");
97          client.print("</html>\r\n");
98          break;
99        }
```

#Result

1. Check the firmware version. The current firmware version is 3.0.0.
2. Attempted to connect to the AP, succeeded, and assigned a virtual IP address of 192.168.1.118.
3. TCP Server was created and WebServer was opened.
4. Check the connection of the client and receive the Get method data from the client
5. HTML data sent to client.



Throughput

#Test Environment

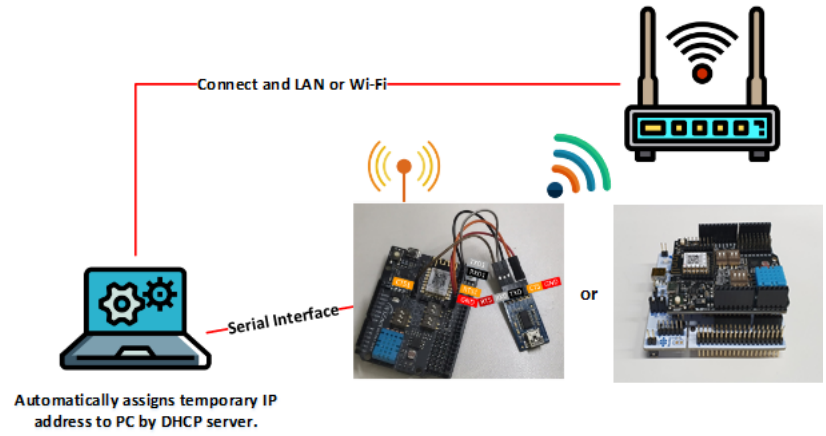
To UART throughput test, it controls using CTS / RTS and WizFi360 control software are required.

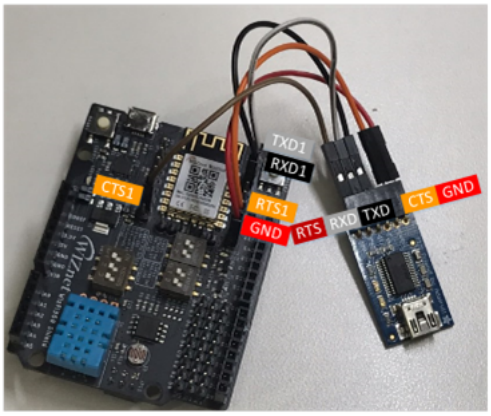
- WizFi360 EVB or WizFi360io
- STM32Fxxx EVB(NUCLEO-F401RE)
- PC
- Serial Tool

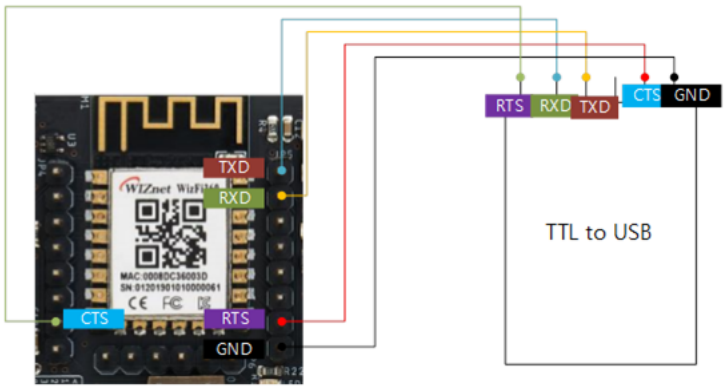
YAT Serial Tool(Data Mode)

- WizFi360 Control Software(Command Mode)
- 1Mbyte data file
- WiFi Router(exclude when it use in softAP mode)

When data mode uses, it sets RTS/CTS in flow control the using the YAT Serial Tool and it sets DTR as Data Read signal. When command mode uses, it sets the AT+CIPSENDBUF=2048 as maximum length of the data to be transmitted and it sends data of 2048 length. Repeat the previous operation.







#Serial command usage

#Station Mode

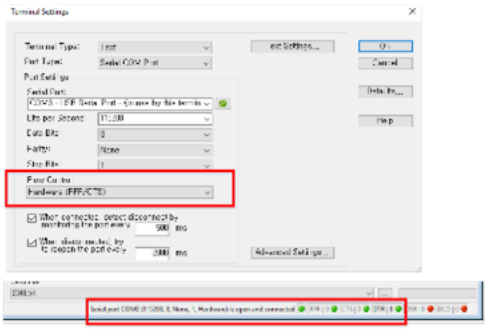
AT Command	Terminal

AT Command	Terminal
AT AT+CWMODE_CUR=1 AT+CWDHCP_CUR=1,1 AT+CWLAP AT+CWJAP_CUR="wizms1","maker0701" AT+CIPSTA_CUR?	AT<CR><LF> <CR><LF> OK<CR><LF> AT+CWMODE_CUR=1<CR><LF> <CR><LF> OK<CR><LF> AT+CWDHCP_CUR=1,1<CR><LF> <CR><LF> OK<CR><LF> AT+CWLAP<CR><LF> +CWLAP:(4,"DIR-815_Wiznet",-59,"[redacted]",1)<CR><LF> +CWLAP:(0,"ESP_574935",-71,"[redacted]",1)<CR><LF> +CWLAP:(3,"##WIZnet_irina",-46,"[redacted]",1)<CR><LF> +CWLAP:(3,"Matthew2.4",-63,"[redacted]",2)<CR><LF> +CWLAP:(3,"rena",-46,"[redacted]",3)<CR><LF> +CWLAP:(0,"iptime",-67,"[redacted]",4)<CR><LF> +CWLAP:(3,"Dap",-63,"[redacted]",5)<CR><LF> +CWLAP:(0,"ESP_577CC7",-67,"[redacted]",6)<CR><LF> +CWLAP:(3,"wizms1",-63,"[redacted]",6)<CR><LF> +CWLAP:(0,"Wizfi360",-69,"[redacted]",6)<CR><LF> +CWLAP:(4,"DLINK-IPv6",-55,"[redacted]",10)<CR><LF> +CWLAP:(0,"iptime",-59,"[redacted]",11)<CR><LF> +CWLAP:(3,"WIZnet_Scott",-51,"[redacted]",11)<CR><LF> +CWLAP:(0,"WizFi360_A1B2D1",-69,"[redacted]",11)<CR><LF> +CWLAP:(3,"Teddy_AP",-57,"[redacted]",13)<CR><LF> <CR><LF> OK<CR><LF> AT+CWJAP_CUR="wizms1","maker0701"<CR><LF> WIFI DISCONNECT<CR><LF> WIFI CONNECTED<CR><LF> WIFI GOT IP<CR><LF> <CR><LF> OK<CR><LF> AT+CIPSTA_CUR?<CR><LF> +CIPSTA_CUR:ip:"192.168.1.120"<CR><LF> +CIPSTA_CUR:gateway:"192.168.1.1"<CR><LF> +CIPSTA_CUR:netmask:"255.255.255.0"<CR><LF> <CR><LF> OK<CR><LF>

#UART CTS/RTS Setting

AT Command	Terminal
AT+CWUART_CUR = 115200,8,1,0,1	AT+UART_CUR=115200,8,1,0,1<CR><LF> <CR><LF> OK<CR><LF>

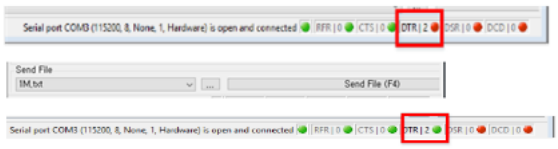
Terminal setting

AT Command	Terminal
1. Pressing Ctrl+Shift+S and Open the Terminal Settings window 2. You have to change the Hardware(RFR/CTS) in Flow Control 3. If you can see under the terminal window that the CTS/DTR is green	

#TCP Client Mode

AT Command	Terminal
AT+CIPSTART="TCP","192.168.100.27",5001 AT+CIPMODE=1 AT+CIPSEND	AT+CIPSTART="TCP","192.168.100.27",5001<CR><LF> CONNECT<CR><LF> <CR><LF> OK<CR><LF> AT+CIPMODE=1<CR><LF> <CR><LF> OK<CR><LF> AT+CIPSEND<CR><LF> <CR><LF> >

Terminal setting

AT Command	Terminal
1. When DTR is red, it sends the 1M.txt 2. If you click the DTR, it changes the DTR is green and it is sending the data through Serial	

#TCP Client / Command mode

AT Command	Terminal

AT Command	Terminal

AT+CIPSTART=”TCP”,”192.168.100.27”,5001
AT+CIPMODE=0
AT+CIPSENDBUF=2048
Send the 2048byte data * 512times = 1Mbyte
	int8_t deviceTestThroughput_WizFi360(char *data, int len)
	{
	int8_t ret = RET_NOK;
	int cnt;
	int segid = 0;
	
	for(cnt = 0; cnt < (len / 4); cnt++) // 2k * 512 = 1M
	{
	if(ATCmdParser_send("AT+CIPSENDBUF=%d", len)&& ATCmdParser_recv("OK") && ATCmdParser_re
	{
	if(ATCmdParser_send("%s", data) && ATCmdParser_recv("%d,SEND OK", &segid))
	{
	ret = RET_OK;
	}
	else
	{
	printf("Write data : failed\r\n");
	}
	}
	else
	{
	printf("Set buffer : failed\r\n");
	}
	}
	
	return ret;
	}

#Result of UART Throughput

PC sends the 1Mbyte through serial of WizFi360(UART1) and WizFi360 send the data to TCP Server.

Baud rate	Data mode		Command mode	
	Time	Speed(bit/s)	Time	Speed(bit/s)
115200	123s	66K	93.9s	87.2K
921600	16.3s	502K	14.0s	585.1K
1000000	14.9s	550K	13.0s	630.2K
1250000	12.7s	645K	11.0s	744.7K
1500000	10.5s	780K	10.0s	819.2K
2000000	9.7s	845K	8.0s	1.0M

We measured the time from the start of data transfer to the end of data transfer using the wireshark tool, see Appendix 1.

#Appendix 1

Item	Detail
Baudrate	115 200
Data mode	3823 122.860907 192.168.100.27 192.168.100.28 TCP 54 5001 → 52161 [ACK] Seq=1 Ack=1023025 Wi 3824 122.865190 192.168.100.28 192.168.100.27 TCP 490 52161 → 5001 [PSH, ACK] Seq=1023025 Ack= 3825 122.906828 192.168.100.27 192.168.100.28 TCP 54 5001 → 52161 [ACK] Seq=1 Ack=1023461 Wi 3826 122.912979 192.168.100.28 192.168.100.27 TCP 594 52161 → 5001 [PSH, ACK] Seq=1023461 Ack= 3827 122.958830 192.168.100.27 192.168.100.28 TCP 54 5001 → 52161 [ACK] Seq=1 Ack=1024001 Wi
Command mode	3 3.351673 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [ACK] Seq=1 Ack=2 Win=6144 Len=102 4 3.351673 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [PSH, ACK] Seq=1025 Ack=2 Win=6144 5 3.351752 192.168.0.4 192.168.0.2 TCP 54 8000 → 57187 [ACK] Seq=2 Ack=2049 Win=64512 Len 6 3.372523 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [ACK] Seq=2049 Ack=2 Win=6144 Len= 7 3.372524 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [PSH, ACK] Seq=3073 Ack=2 Win=6144
	1534 14.330917 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [PSH, ACK] Seq=1045505 Ack=2 Wi 1535 14.330971 192.168.0.4 192.168.0.2 TCP 54 8000 → 57187 [ACK] Seq=2 Ack=1046529 Win=645 1536 14.351216 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [ACK] Seq=1046529 Ack=2 Win=614 1537 14.351217 192.168.0.2 192.168.0.4 TCP 1078 57187 → 8000 [PSH, ACK] Seq=1047553 Ack=2 Wi 1538 14.351273 192.168.0.4 192.168.0.2 TCP 54 8000 → 57187 [ACK] Seq=2 Ack=1048577 Win=645
Baudrate	921 600
Data mode	2547 16.217822 192.168.100.28 192.168.100.27 TCP 1078 52165 → 5001 [ACK] Seq=1022161 Ack 2548 16.217860 192.168.100.27 192.168.100.28 TCP 54 5001 → 52165 [ACK] Seq=1 Ack=10231 2549 16.317138 192.168.100.28 192.168.100.27 TCP 870 52165 → 5001 [PSH, ACK] Seq=10231 2550 16.357729 192.168.100.27 192.168.100.28 TCP 54 5001 → 52165 [ACK] Seq=1 Ack=10240

Item	Detail

Command mode

```
3 1.958011    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [ACK] Seq=1 Ack=2 Win=6144 Len=102
4 1.958012    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=1025 Ack=2 Win=6144
5 1.958109    192.168.0.4  192.168.0.2  TCP      54 8000 → 60368 [ACK] Seq=2 Ack=2049 Win=64512 Len
6 1.979981    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [ACK] Seq=2049 Ack=2 Win=6144 Len=
7 1.979981    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=3073 Ack=2 Win=6144
1537 11.937349    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=1045505 Ack=2 Wi
1538 11.937412    192.168.0.4  192.168.0.2  TCP      54 8000 → 60368 [ACK] Seq=2 Ack=1046529 Win=645
1539 11.955972    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [ACK] Seq=1046529 Ack=2 Win=614
1540 11.955972    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=1047553 Ack=2 Wi
1541 11.956032    192.168.0.4  192.168.0.2  TCP      54 8000 → 60368 [ACK] Seq=2 Ack=1048577 Win=645
```

Baudrate 1 000 000

Data mode

```
3868 14.774321    192.168.100.28  192.168.100.27  TCP    490 58128 → 5001 [PSH, ACK] Seq=102312
3869 14.815213    192.168.100.27  192.168.100.28  TCP     54 5001 → 58128 [ACK] Seq=1 Ack=10235
3870 14.819495    192.168.100.28  192.168.100.27  TCP    494 58128 → 5001 [PSH, ACK] Seq=102356
3871 14.859281    192.168.100.27  192.168.100.28  TCP     54 5001 → 58128 [ACK] Seq=1 Ack=10240
```

Command mode

```
3 2.492951    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [ACK] Seq=1 Ack=2 Win=6144 Len=102
4 2.496860    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [PSH, ACK] Seq=1025 Ack=2 Win=6144
5 2.496929    192.168.0.4  192.168.0.2  TCP      54 8000 → 63635 [ACK] Seq=2 Ack=2049 Win=64512 Len
6 2.506899    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [ACK] Seq=2049 Ack=2 Win=6144 Len=
7 2.507544    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [PSH, ACK] Seq=3073 Ack=2 Win=6144
1533 10.470341    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [PSH, ACK] Seq=1045505 Ack=2 Wi
1534 10.470400    192.168.0.4  192.168.0.2  TCP      54 8000 → 63635 [ACK] Seq=2 Ack=1046529 Win=645
1535 10.485628    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [ACK] Seq=1046529 Ack=2 Win=614
1536 10.485628    192.168.0.2  192.168.0.4  TCP    1078 63635 → 8000 [PSH, ACK] Seq=1047553 Ack=2 Wi
1537 10.485694    192.168.0.4  192.168.0.2  TCP      54 8000 → 63635 [ACK] Seq=2 Ack=1048577 Win=645
```

Baudrate 1 250 000

Data mode

```
2863 12.592400    192.168.100.28  192.168.100.27  TCP    490 58136 → 5001 [PSH, ACK] Seq=1023185 Ac
2864 12.631883    192.168.100.27  192.168.100.28  TCP     54 5001 → 58136 [ACK] Seq=1 Ack=1023621 W
2865 12.633959    192.168.100.28  192.168.100.27  TCP    434 58136 → 5001 [PSH, ACK] Seq=1023621 Ac
2866 12.674856    192.168.100.27  192.168.100.28  TCP     54 5001 → 58136 [ACK] Seq=1 Ack=1024001 W
```

Command mode

```
3 3.351673    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [ACK] Seq=1 Ack=2 Win=6144 Len=102
4 3.351673    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [PSH, ACK] Seq=1025 Ack=2 Win=6144
5 3.351752    192.168.0.4  192.168.0.2  TCP      54 8000 → 57187 [ACK] Seq=2 Ack=2049 Win=64512 Len
6 3.372523    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [ACK] Seq=2049 Ack=2 Win=6144 Len=
7 3.372524    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [PSH, ACK] Seq=3073 Ack=2 Win=6144
1534 14.330917    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [PSH, ACK] Seq=1045505 Ack=2 Wi
1535 14.330971    192.168.0.4  192.168.0.2  TCP      54 8000 → 57187 [ACK] Seq=2 Ack=1046529 Win=645
1536 14.351216    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [ACK] Seq=1046529 Ack=2 Win=614
1537 14.351217    192.168.0.2  192.168.0.4  TCP    1078 57187 → 8000 [PSH, ACK] Seq=1047553 Ack=2 Wi
1538 14.351273    192.168.0.4  192.168.0.2  TCP      54 8000 → 57187 [ACK] Seq=2 Ack=1048577 Win=645
```

Baudrate 1 500 000

Data mode

```
2242 10.389973    192.168.100.28  192.168.100.27  TCP    490 65021 → 5001 [PSH, ACK] Seq=1023074
2244 10.430942    192.168.100.27  192.168.100.28  TCP     54 5001 → 65021 [ACK] Seq=1 Ack=1023510
2245 10.445897    192.168.100.28  192.168.100.27  TCP    545 65021 → 5001 [PSH, ACK] Seq=1023510
2247 10.486915    192.168.100.27  192.168.100.28  TCP     54 5001 → 65021 [ACK] Seq=1 Ack=1024001
```

Command mode

```
3 1.958011    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [ACK] Seq=1 Ack=2 Win=6144 Len=102
4 1.958012    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=1025 Ack=2 Win=6144
5 1.958109    192.168.0.4  192.168.0.2  TCP      54 8000 → 60368 [ACK] Seq=2 Ack=2049 Win=64512 Len
6 1.979981    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [ACK] Seq=2049 Ack=2 Win=6144 Len=
7 1.979981    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=3073 Ack=2 Win=6144
1537 11.937349    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=1045505 Ack=2 Wi
1538 11.937412    192.168.0.4  192.168.0.2  TCP      54 8000 → 60368 [ACK] Seq=2 Ack=1046529 Win=645
1539 11.955972    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [ACK] Seq=1046529 Ack=2 Win=614
1540 11.955972    192.168.0.2  192.168.0.4  TCP    1078 60368 → 8000 [PSH, ACK] Seq=1047553 Ack=2 Wi
1541 11.956032    192.168.0.4  192.168.0.2  TCP      54 8000 → 60368 [ACK] Seq=2 Ack=1048577 Win=645
```

Baudrate 2 000 000

Data mode

```
6316 9.646387    192.168.100.28  192.168.100.27  TCP    490 65031 → 5001 [PSH, ACK] Seq=1023245 Ac
6317 9.686546    192.168.100.27  192.168.100.28  TCP     54 5001 → 65031 [ACK] Seq=1 Ack=1023681 W
6318 9.690489    192.168.100.28  192.168.100.27  TCP    374 65031 → 5001 [PSH, ACK] Seq=1023681 Ac
6319 9.731538    192.168.100.27  192.168.100.28  TCP     54 5001 → 65031 [ACK] Seq=1 Ack=1024001 W
```

Item	Detail
Command mode	3 2.492951 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [ACK] Seq=1 Ack=2 Win=6144 Len=102 4 2.496860 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [PSH, ACK] Seq=1025 Ack=2 Win=6144 5 2.496929 192.168.0.4 192.168.0.2 TCP 54 8000 → 63635 [ACK] Seq=2 Ack=2049 Win=64512 Len 6 2.506899 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [ACK] Seq=2049 Ack=2 Win=6144 Len= 7 2.507544 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [PSH, ACK] Seq=3073 Ack=2 Win=6144 1533 10.470341 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [PSH, ACK] Seq=1045505 Ack=2 Wi 1534 10.470400 192.168.0.4 192.168.0.2 TCP 54 8000 → 63635 [ACK] Seq=2 Ack=1046529 Win=645 1535 10.485628 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [ACK] Seq=1046529 Ack=2 Win=614 1536 10.485628 192.168.0.2 192.168.0.4 TCP 1078 63635 → 8000 [PSH, ACK] Seq=1047553 Ack=2 Wi 1537 10.485694 192.168.0.4 192.168.0.2 TCP 54 8000 → 63635 [ACK] Seq=2 Ack=1048577 Win=645

W600-SDK

#Overview

WizFi360 is based on the W600 chip which features ARM Cortex-M3 with 1MB on-chip flash and freeRTOS kernel. This page includes the software development kit (SDK) files, documents, and usage method. This SDK has integrated WizFi360 hardware driver (BSP), RTOS, TCP/IP protocol, Wi-Fi Protocol and other public function modules.

#Software SDK

Item	Detail
Official SDK	Contact us
Official SDK from chip vendor	Download link
KEIL Flash Driver	Download link

#Software SDK User Manual

#Basic

Item	Description
WizFi360_W600_SDK_Guide_v0.0.1_E	Describes the basic way to download and compile of W600 SDK in English.
WizFi360_W600_SDK_Guide_v0.0.1_K	Describes the basic way to download and compile of W600 SDK in Korean.
SDK User Manual V1.1	Describes the functions and usage of W600 SDK.
SDK DEMO User Guide V0.5	Describes the demo code of SDK for developers.
SDK GCC Compiling Guide V1.2	Describes building in linux environment
AT Command V1.0.3	Describes the WM's AT instruction communication protocol.

#Advanced

Item	Description
Firmware Generation Guide V1.2	Describes the firmware's format, storage address and image generation.
Firmware Update Guide V1.2	Describes how to update the firmware.
QFLASH Management Guide V1.2	Describes the management of QFLASH intergrated in chip.
Parameter Space Guide V1.2	Describes how to deal with QFLASH management, key parameter space, system parameter space and user parameter space.
SECBOOT Function Guide V1.1	Describes SECBOOT functions and usages for designer and developer.
SWD Debugging Guide V1.4	Describes chip on-line debug configuration under KEIL IDE.

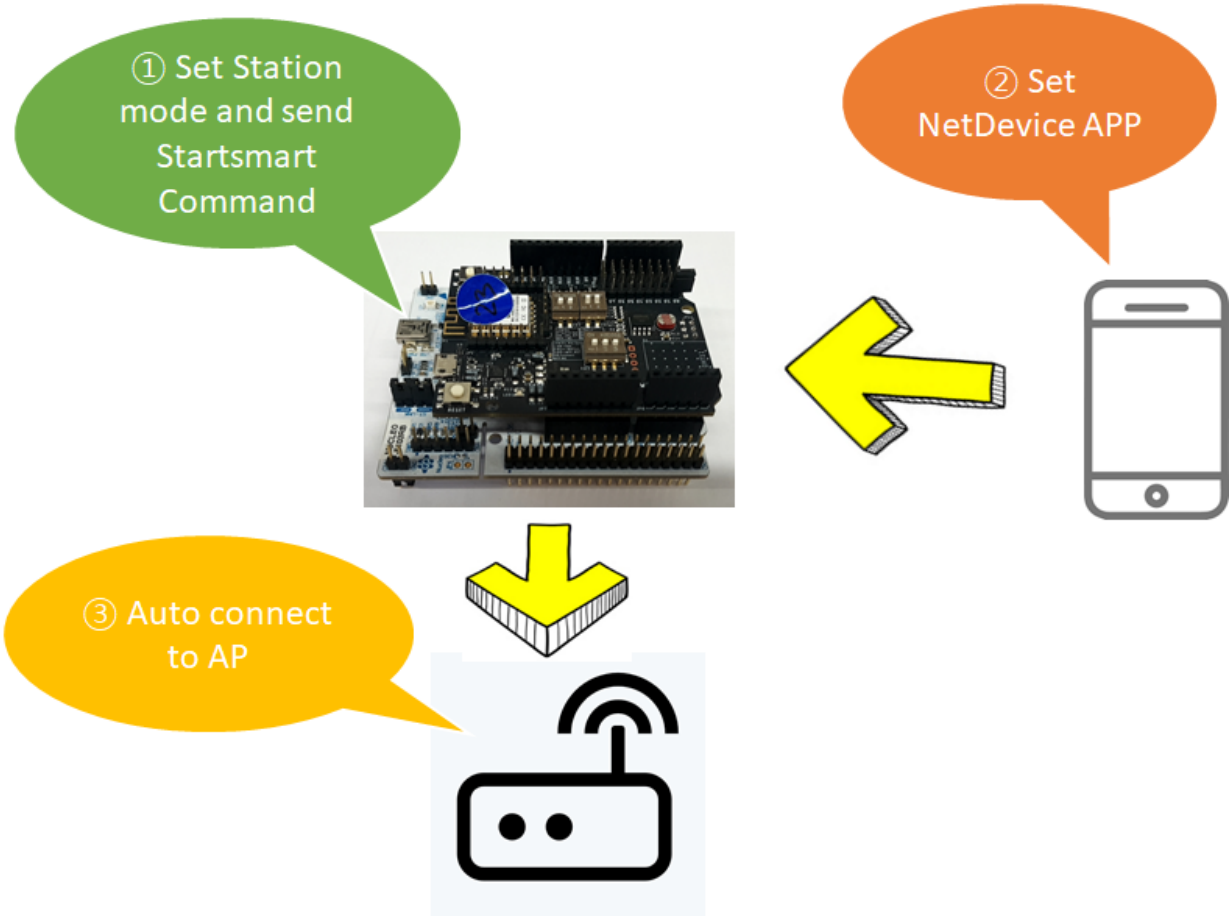
#Chip vendor original page

WinnerMicro W600 download Page

Airkiss Configuration Guide

#Introduction

In this guide we introduce Airkiss method to configure WizFi360.



#Airkiss sequence

1. Set module to station mode using AT+CWMODE_DEF=1
2. Airkiss start using AT+CWSTARTSMART=2
3. Launch Android app and input WiFi AP' SSID and password.
4. Check AP connection status
5. Start WizFi360 Server mode

important

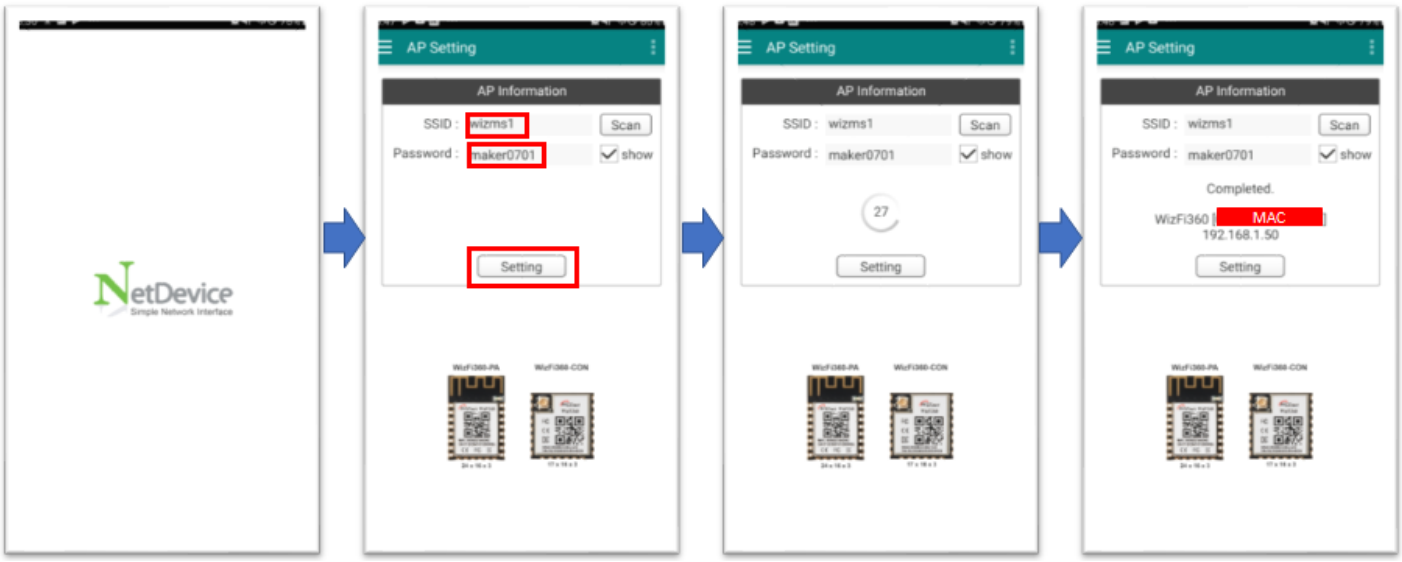
• SmartConfig is only available in the Station mode. (AT+CWMODE_CUR=1)
• The message "smart get wifi info" means that SmartConfig has successfully acquired the AP information. WizFi360 will try to connect to the target AP.
• The message "smartconfig connected wifi" is printed if the connection is successful. Use command AT+CWSTOPSMART to stop SmartConfig before running other commands. Please make sure that you do not execute other commands during SmartConfig.
• The LED(PB_07) is blinking fast during running Oneshot via Airkiss.
The LED(PB_07) is blinking slowly during running Oneshot via WebServer.• SmartConfig

#Android Application

#Connect with AirKiss

Prior configuring WizFi360 module it is necessary to set settings in Android application. This procedure is simple and contains following steps:

1. Launch application on smartphone which is connected to AP.
2. Input SSID and password, then press "Setting" button.
3. Wait while WizFi360 is connected to AP.
4. Upon completion app will display MAC and IP addresses.

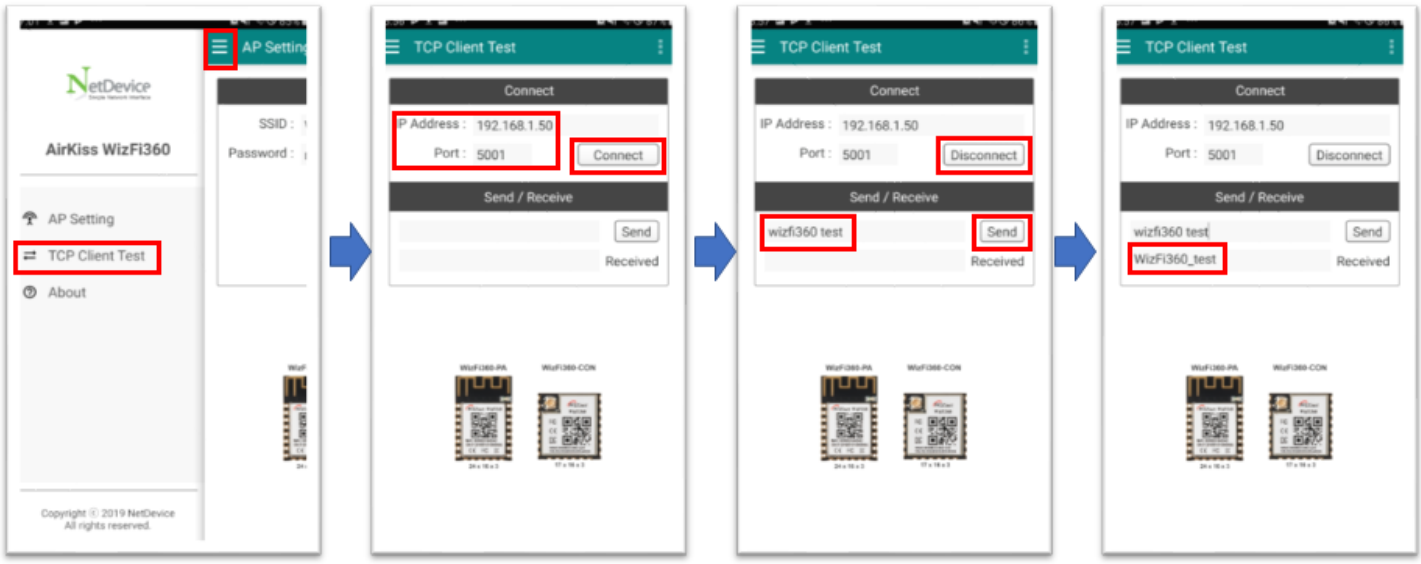


#TCP Client Test

It is possible to test TCP connection between WizFi360 and smartphone, where smartphone will be TCP client and WizFi360 will be used as server.

To conduct test follow below procedures:

1. Press "Menu" button on top left side of application. Then press "TCP Client Test"
2. Check IP address and port, then click "Connect" button. When application is connected, button will be changed to "Disconnect".
3. Input a message and press "Send" button.
4. App will display message on the screen when it is received from server.



Program WizFi360 using Arduino IDE

#Introduction

It is possible to program WizFi360 using Arduino IDE. In this guide we will show how to do it.

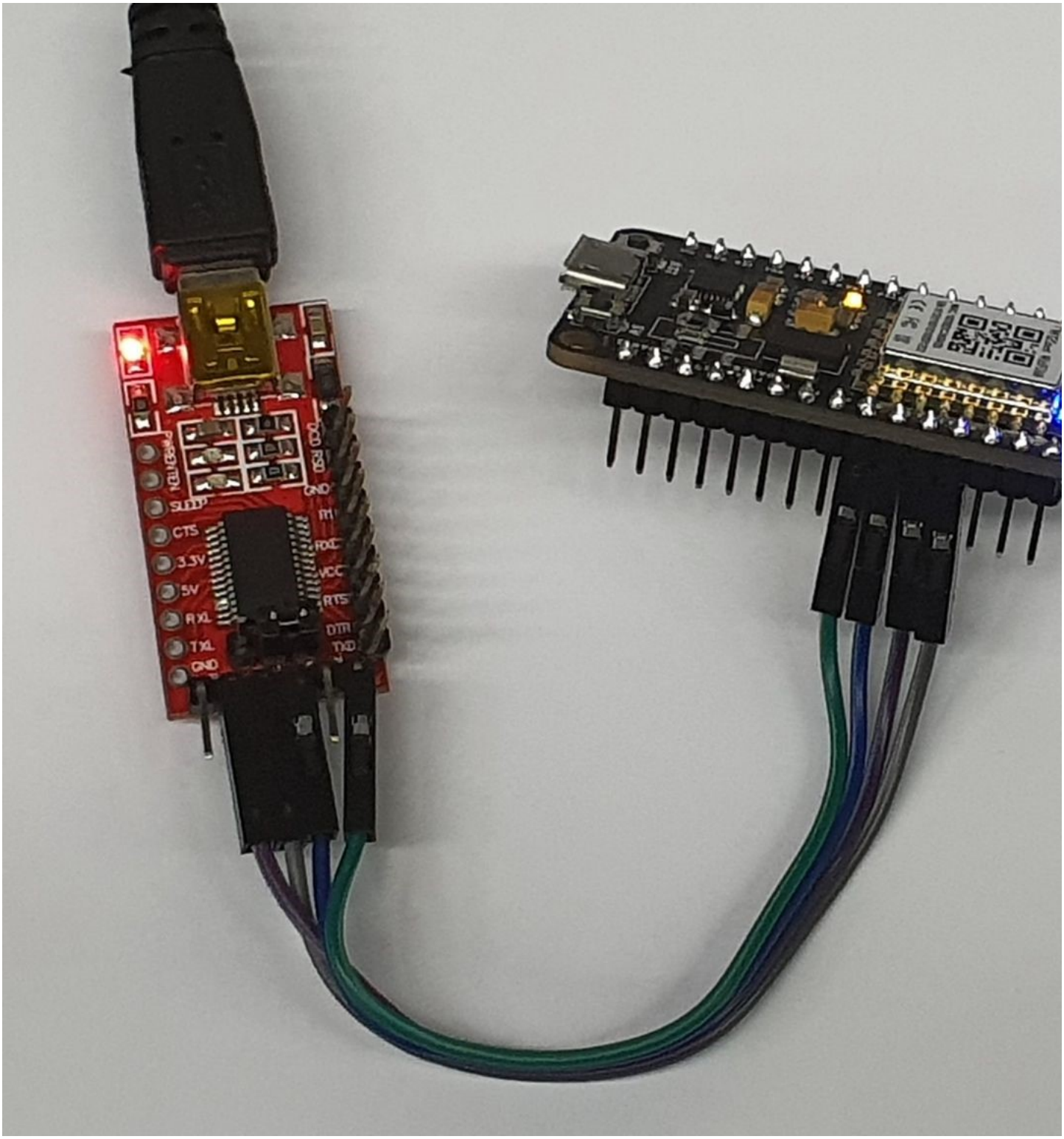
#Requirements

#Hardware

- WizFi360-EVB-mini or WizFi360-EVB-Shield
- USB cable
- Notebook or Desktop PC
- UartToUsb

Connect UartToUsb to WizFi360 evaluation board as below:

- WizFi360 RXD0 → UartToUSB TX
- WizFi360 TXD0 → UartToUSB RX
- WizFi360 3v3 → UartToUSB VCC
- WizFi360 GND → UartToUSB GND



#Software

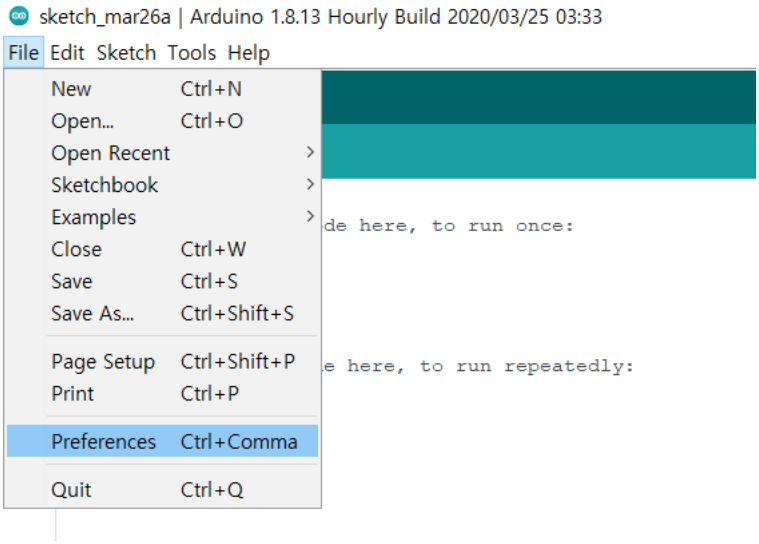
- Arduino IDE

important

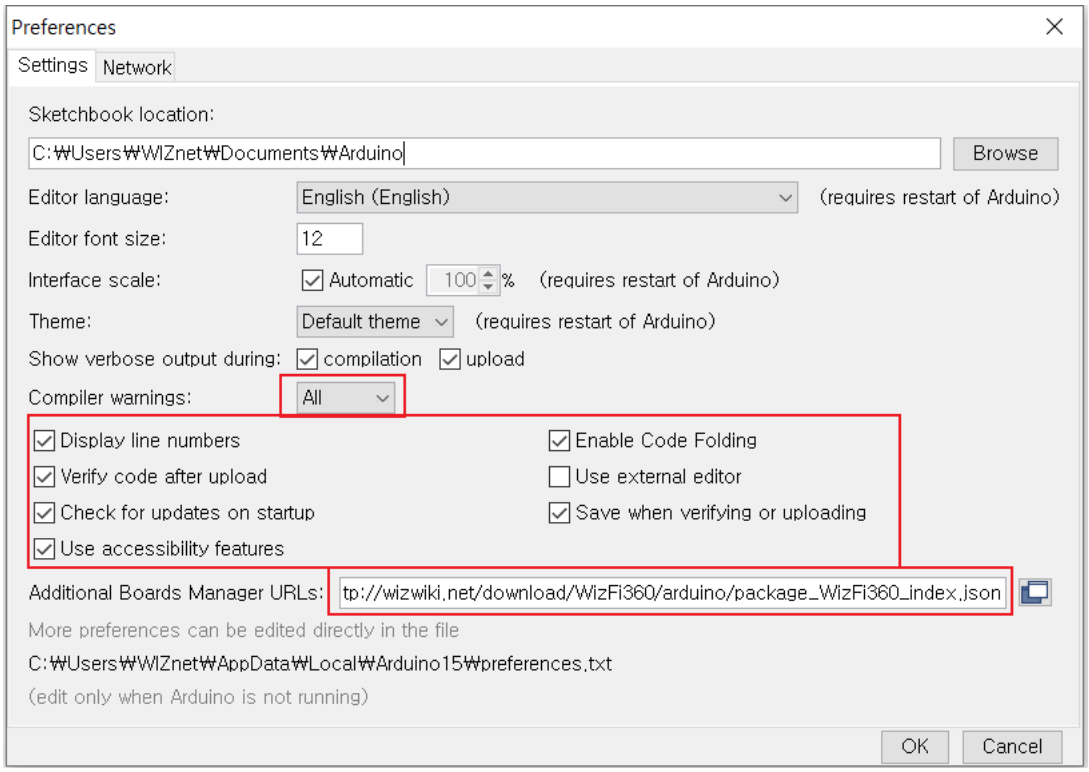
For this guide hourly build of Arduino IDE was used. Arduino IDE version was "LAST UPDATE 25 March 2020 23:34:57 GMT" at the moment of this guide creation.

#Instructions

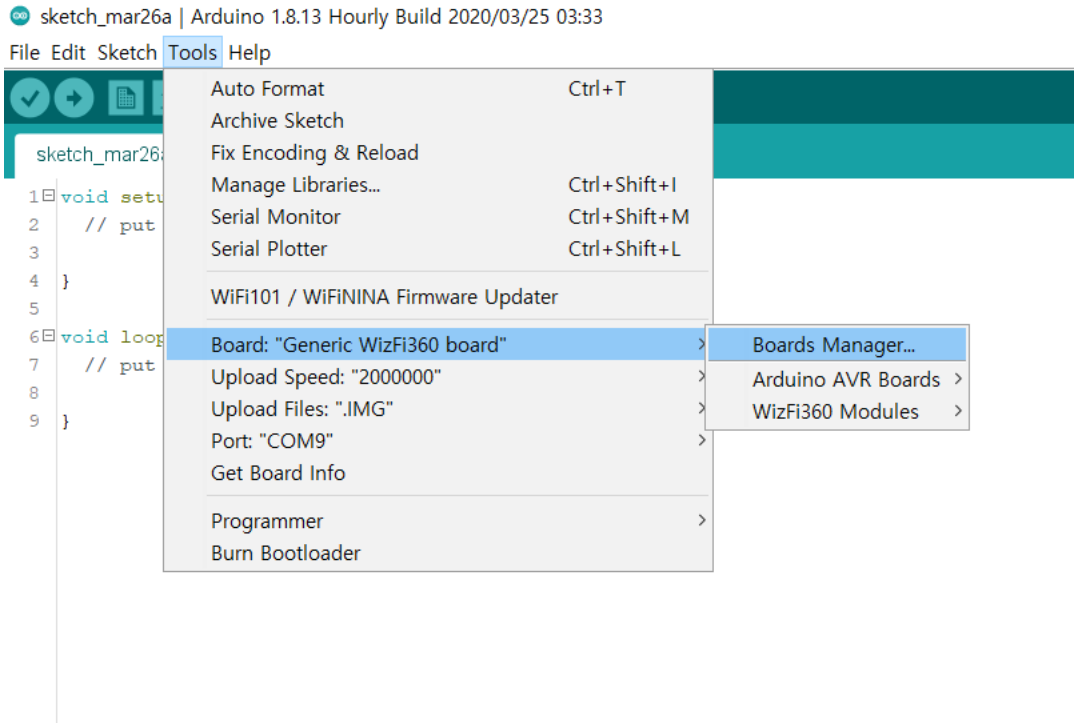
1. Open Preferences in Arduino IDE.



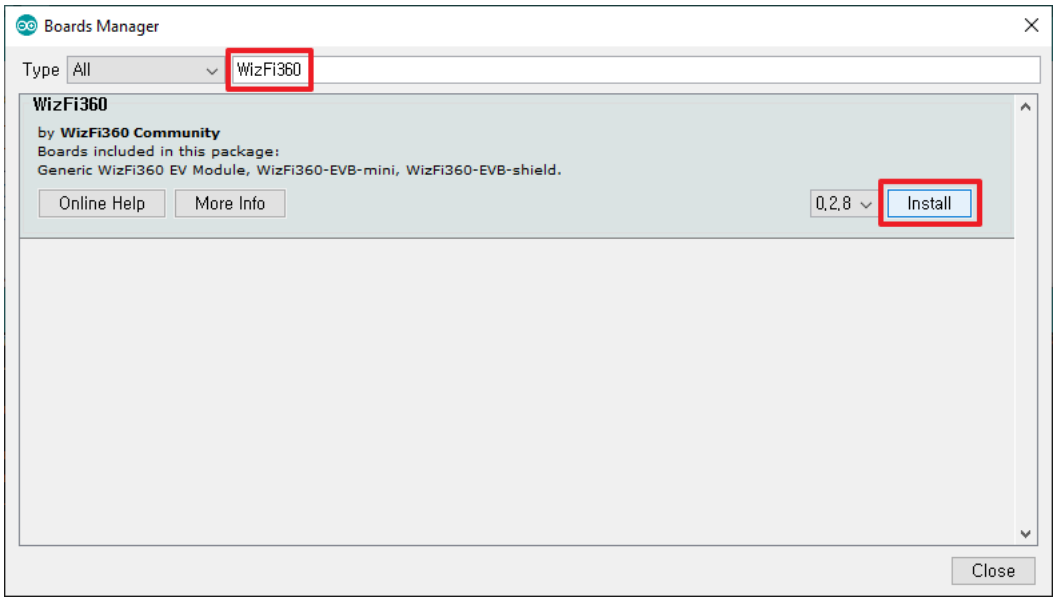
2. In preferences window set compiler warnings to "All". Also check all items except "Use external editor".
   Add http://wizwiki.net/download/WizFi360/arduino/package_WizFi360_index.json to board manager URL.



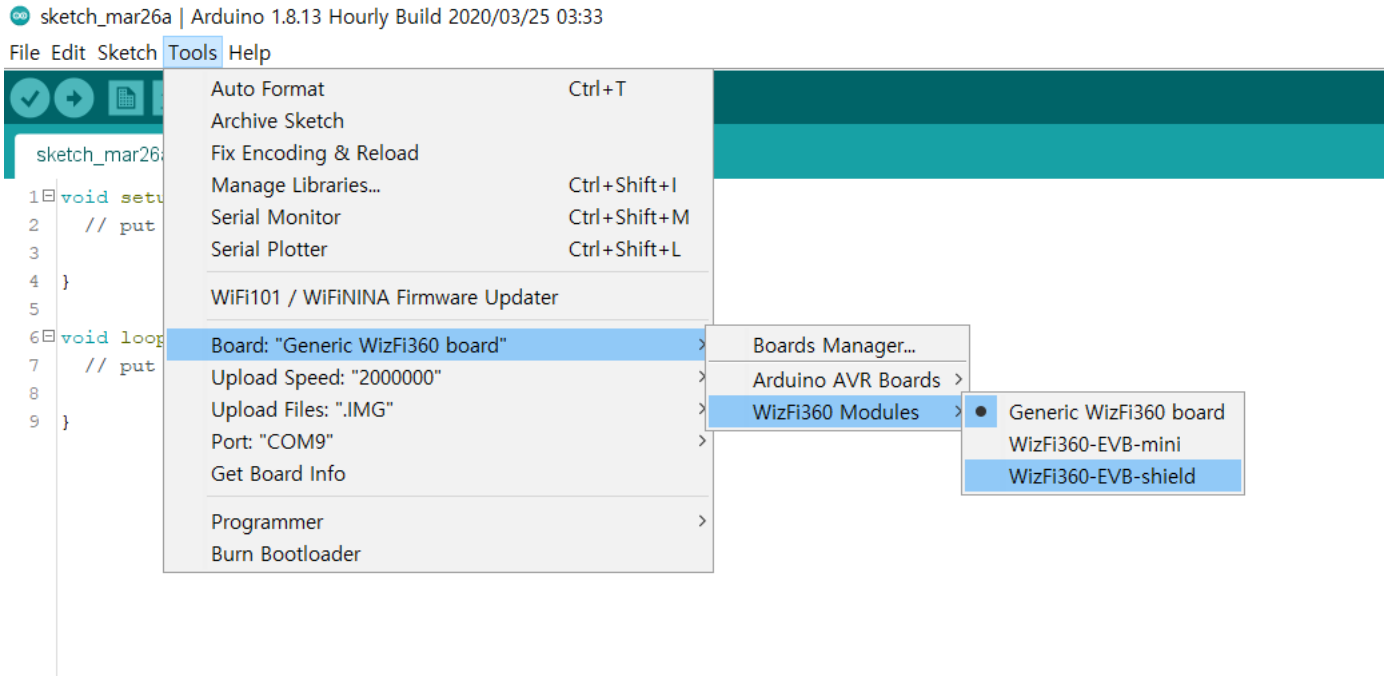
3. Open board manager from Tools menu.



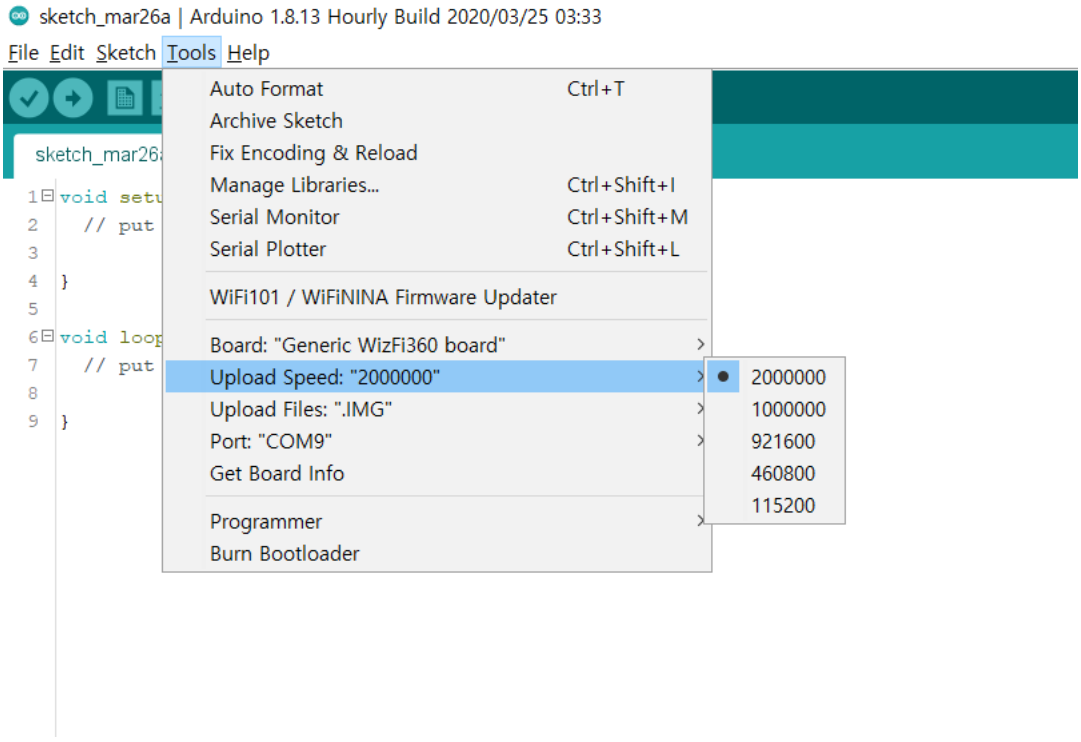
4. Search for WizFi360 in boards manager and install latest version.



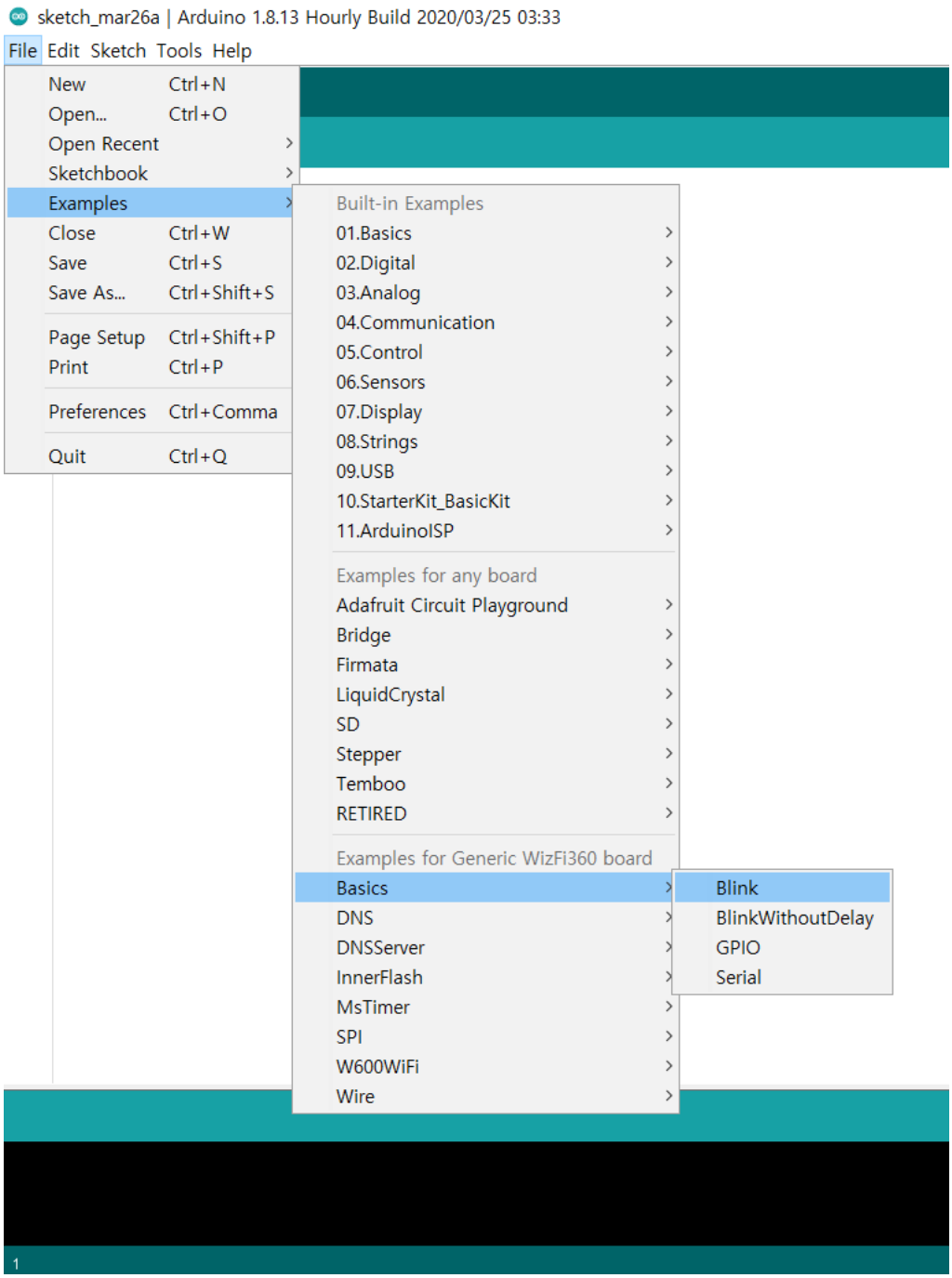
5. Select WizFi360-EVB-Mini or Shield in Tools -> Board menu.



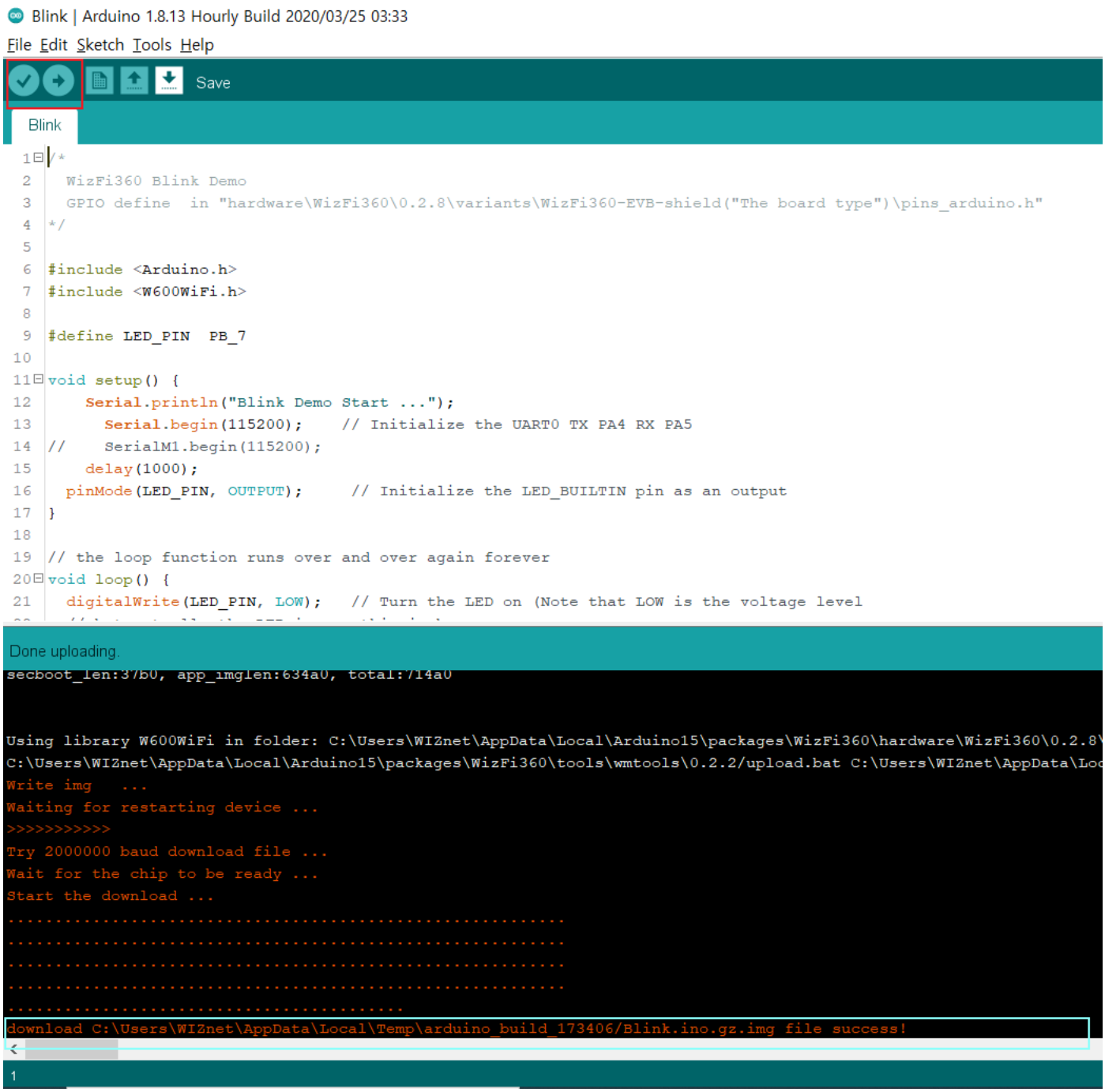
6. Set upload speed to 2000000 and check COM port.



7. Open example from File -> Examples



8. Compile and upload.



Congratulations! Process is completed!

WizFi360 MQTT AT Command를 이용하여 Azure IoT Hub에 연동

#시작하기 전에

Azure Portal에 Login을 합니다. 계정이 없는 경우, 계정 생성 후에 Login을 진행합니다.

※ 본 문서는 **체험 계정**으로 진행합니다.

Azure Portal을 사용하여 IoT Hub 만들기 등 앞선 일련의 과정에 대하여 Azure Cloud 서비스 소개를 참조하시기 바랍니다.

- [MS] Azure Portal을 사용하여 IoT Hub 만들기
- Azure Portal을 사용하여 Blob Storage 만들기
- Azure Portal을 사용하여 Stream Analytics 만들기
- Azure Portal을 사용하여 Stream Analytics 작업 입 · 출력 구성 및 변환 Query 정의

#소개

Microsoft Azure Service에 **WizFi360**을 **연동**하여, Data를 Cloud로 전송하고, Monitoring을 할 수 있습니다.

Data 통신은 다음과 같은 구조로 이루어집니다.

MQTT AT Command를 이용하여, IoT Hub Service 연결 및 Data 송신을 합니다.

IoT Hub로 송신이 된 Data는 Stream Analytics를 통하여 Data 저장소 Blob Storage로 저장이 됩니다.

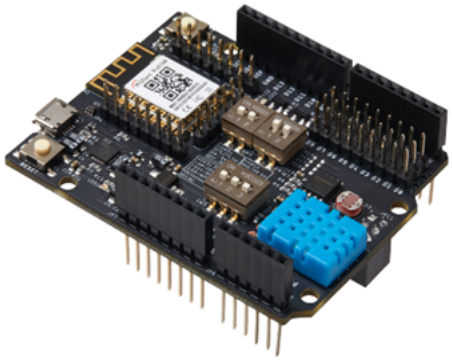
본 문서는 WizFi360 MQTT AT Command 이용하여 Microsoft Azure Service 연결 방법에 대한 Guide를 제공합니다.

#Step 1: 필수 구성 요소

본 문서를 따라하기에 전에 다음 항목이 준비되어야 합니다.

#Hadrware

- Desktop or Laptop Computer
- WizFi360-EVB-Shield
- Micro 5 Pin USB Cable



#Software

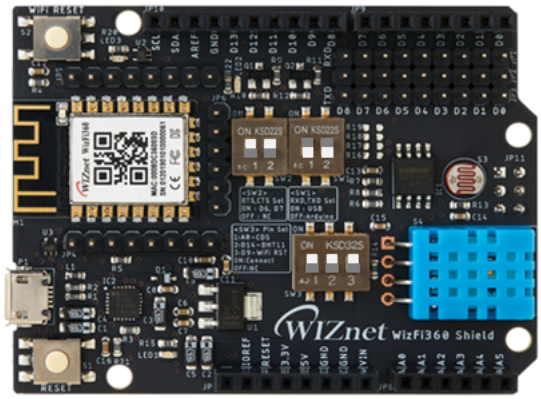
- Preferred Serial Terminal (TeraTerm, Hercules, etc . . .)

#Step 2: Device 준비

#1. Hardware 준비

WizFi360-EVB-Shield는 **Standalone Mode**로 사용되어 집니다. 따라서 WizFi360-EVB-Shield의 DIP Switch를 다음과 같이 설정이 필요합니다.

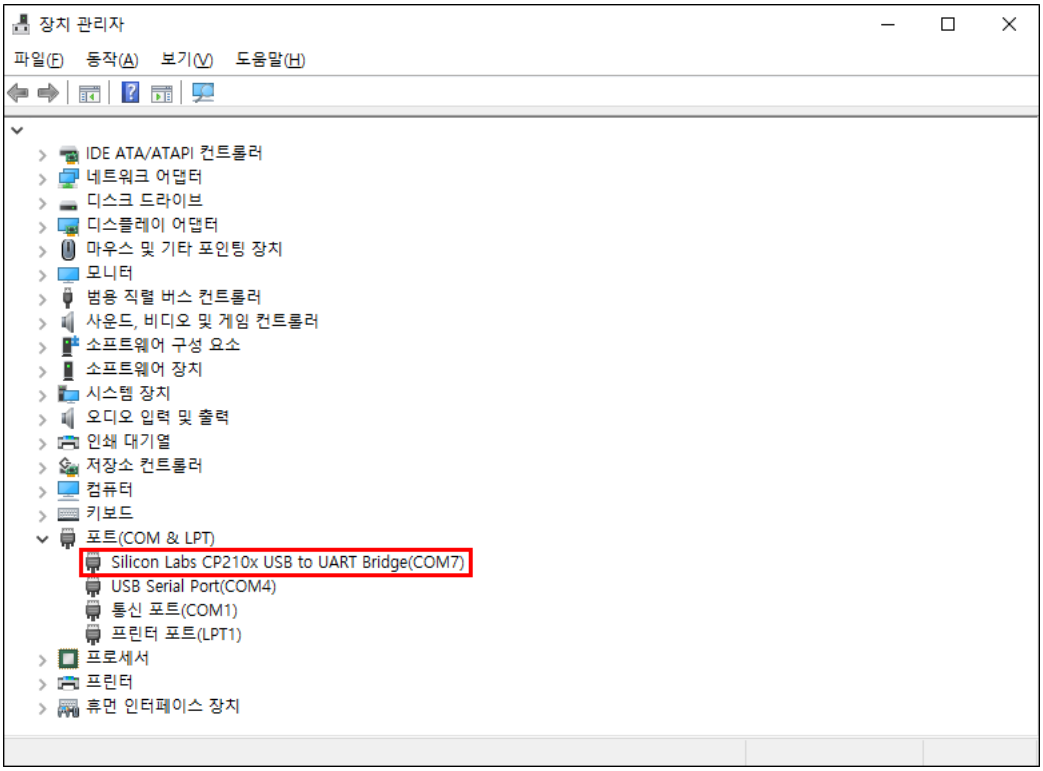
- SW1 : On
- SW2 : Off
- SW3 : Off



#2. Device 연결

Hardware 설정 후, Micro 5 Pin USB Cable을 이용하여 WizFi360-EVB-Shield를 Desktop 혹은 Laptop Computer와 연결을 합니다.

장치 관리자에서 WizFi360-EVB-Shield와 연결된 **COM Port**를 확인 할 수 있습니다.



장치 관리자에서 COM Port를 확인 할 수 없는 경우, 다음 Link에서 Driver를 Downlonad하여 설치하시기 바랍니다.

- CP210x USB to UART Bridge VCP Drivers

#Step 3: 동작 예제

WizFi360의 조금 더 상세한 **사용 방법**은 **Quick Start Guide**, **AT Command**는 **AT Instruction Set**을 참고 바랍니다.

#1. Mode 설정

Command	**Response**
AT+CWMODE_CUR=1	OK

#2. DHCP 설정

Command	**Response**
AT+CWDHCP_CUR=1,1	OK

#3. AP 접속

Command	**Response**
AT+CWJAP_CUR="**ssid**","**password**"	WIFI CONNECTED WIFI GOT IP

Example :
AT+CWJAP_CUR="**wiznet**","**0123456789**" OK

#4. MQTT 연결 설정

Command

AT+MQTTSET="**iot_hub_host_name**/**device_id**/?api-version=2018-06-30","**sas_token**","**device_id**",60

Example :
AT+MQTTSET="**MyWizFi360IoTHub.azure-devices.net/MyWizFi360IoTDevice**/?api-version=2018-06-30","**SharedAccessSignature sr=MyWizFi360IoTHub.az**
devices.net%2Fdevices%2FMyWizFi360IoTDevice&sig=t3R9nDS7ezMGBdb%2FNd5ktb3xQx5jx4NC02n325vRA6c%3D&se=1611895717","**MyWizFi360IoT**

SAS Token 생성은 다음을 참고 바랍니다.

- Device Explorer를 사용하여 SAS Token 생성하기
- Azure IoT Explorer를 사용하여 SAS Token 생성하기

#5. Topic 설정

Command	**Response**
AT+MQTTTOPIC="devices/**device_id**/messages/events/",devices/**device_id**/messages/devicebound/#	OK
Example : AT+MQTTTOPIC="devices/**MyWizFi360IoTDevice**/messages/events/","devices/**MyWizFi360IoTDevice**/messages/devicebound/#"	

#6. Broker 연결

Command	**Response**
AT+MQTTCON=1,"**iot_hub_host_name**",8883	CONNECT
Example : AT+MQTTCON=1,"**MyWizFi360IoTHub.azure-devices.net**",8883	OK

#7. Publish Message

Note :

Data를 Publish 할 때, **Stream Analytics 실행** 중이어야 Blob Storage로 Data가 전달됩니다.

Command	**Response**
AT+MQTTPUB="publish_data"	OK
Example : AT+MQTTPUB="{"deviceId":"MyWizFi360IoTDevice","temperature":21.97,"humidity":43.58}"	

Note :

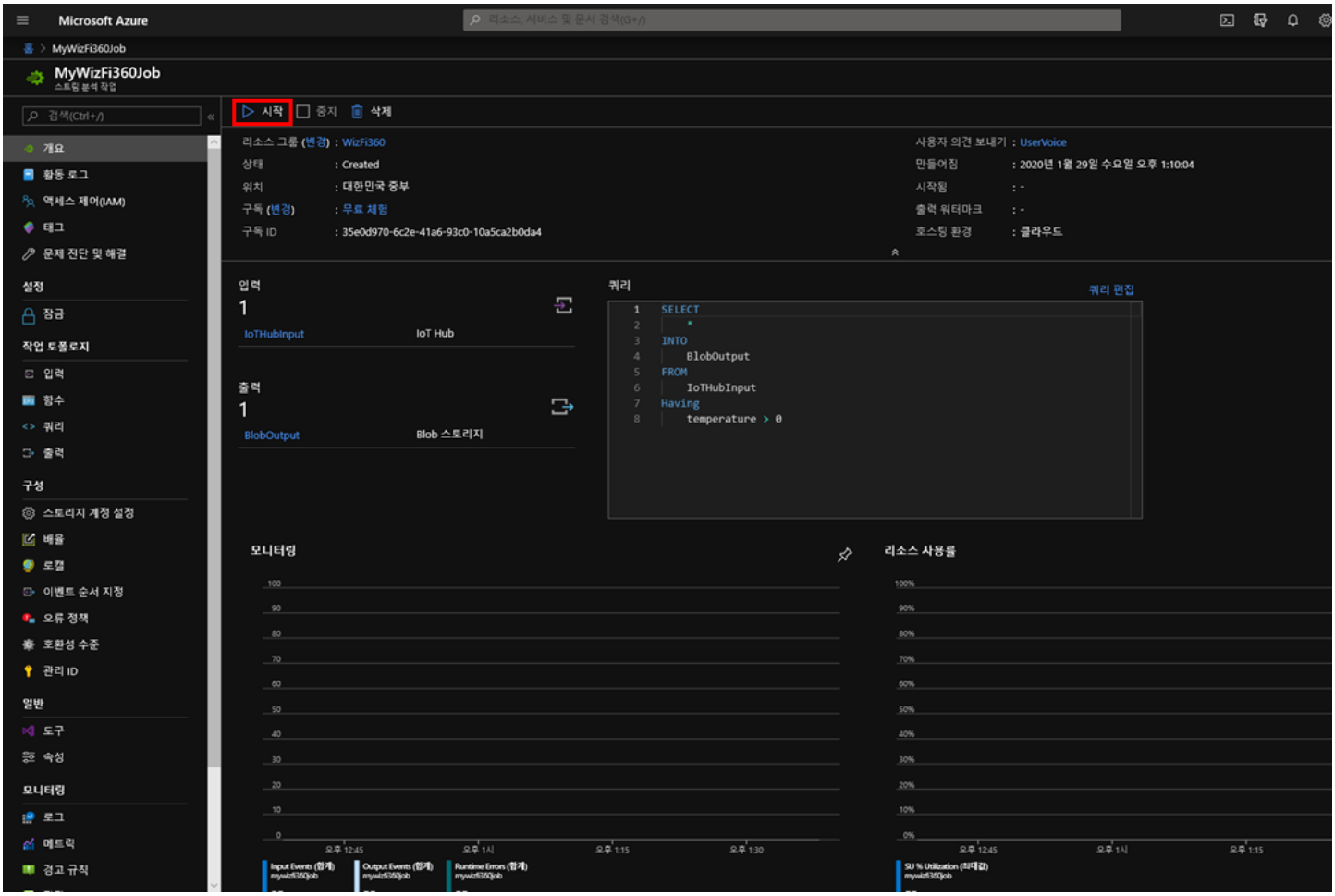
Publish Data는 어떤 형태라도 가능하지만, Azure Guide의 Stream Analytics에서 작업 입력 구성 설정 중,

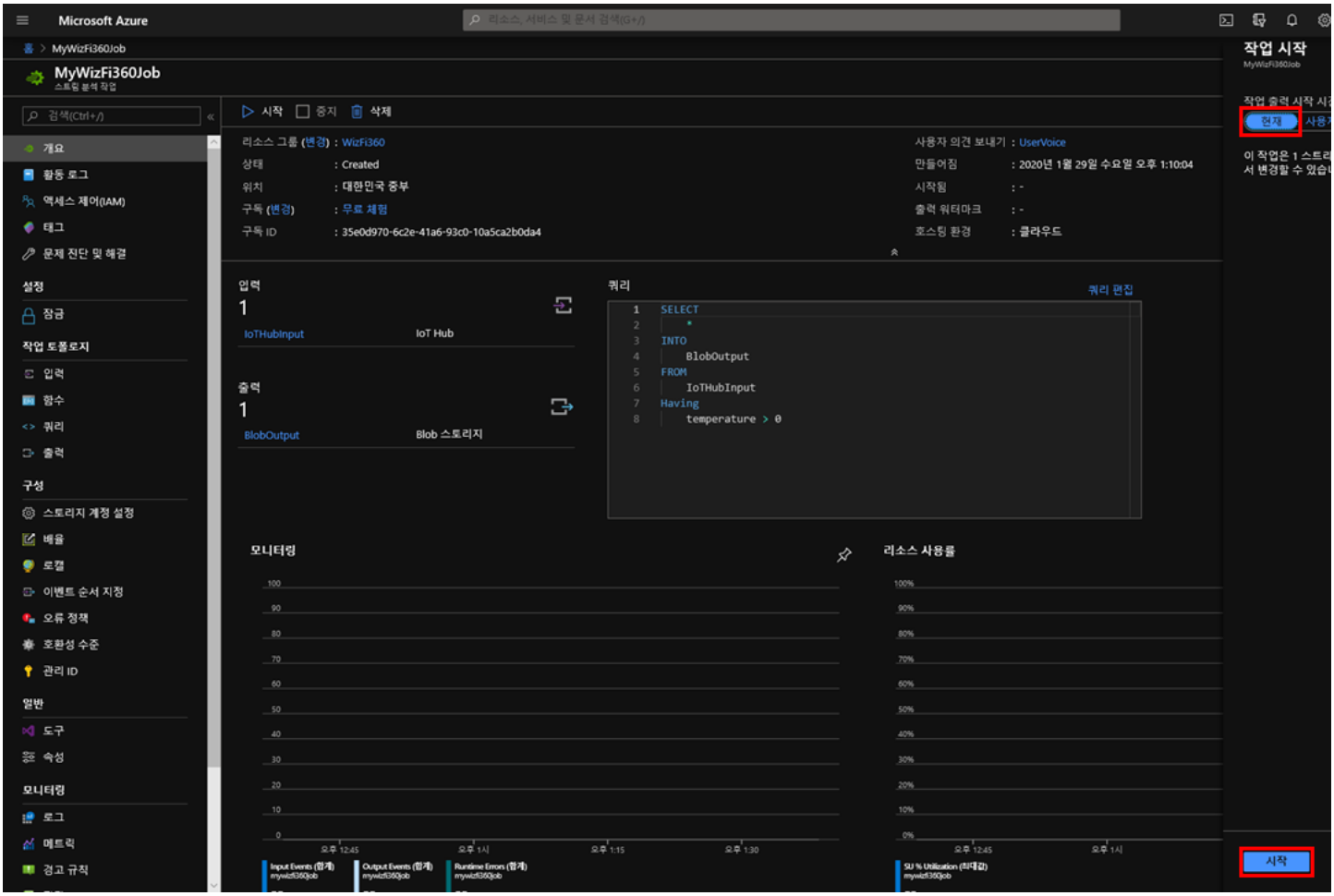
Event Serialization 형식이 기본 JSON 형태이므로 맞춰주어야 합니다.

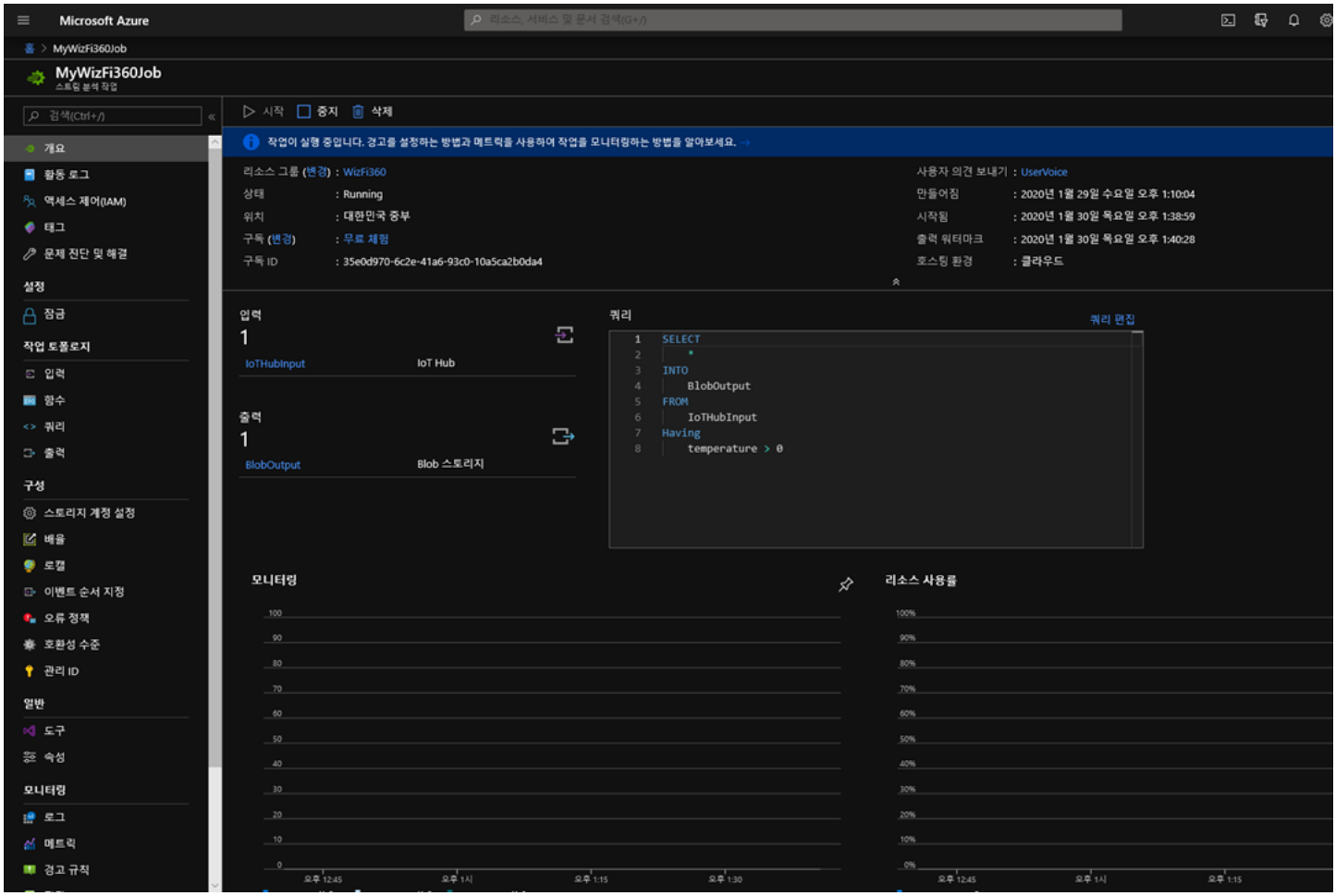
#Step 4: 동작 예제 결과

#1. Stream Analytics 작업 시작

Stream Analytics에서 **개요** > **시작** > **현재** > **시작**을 선택합니다.

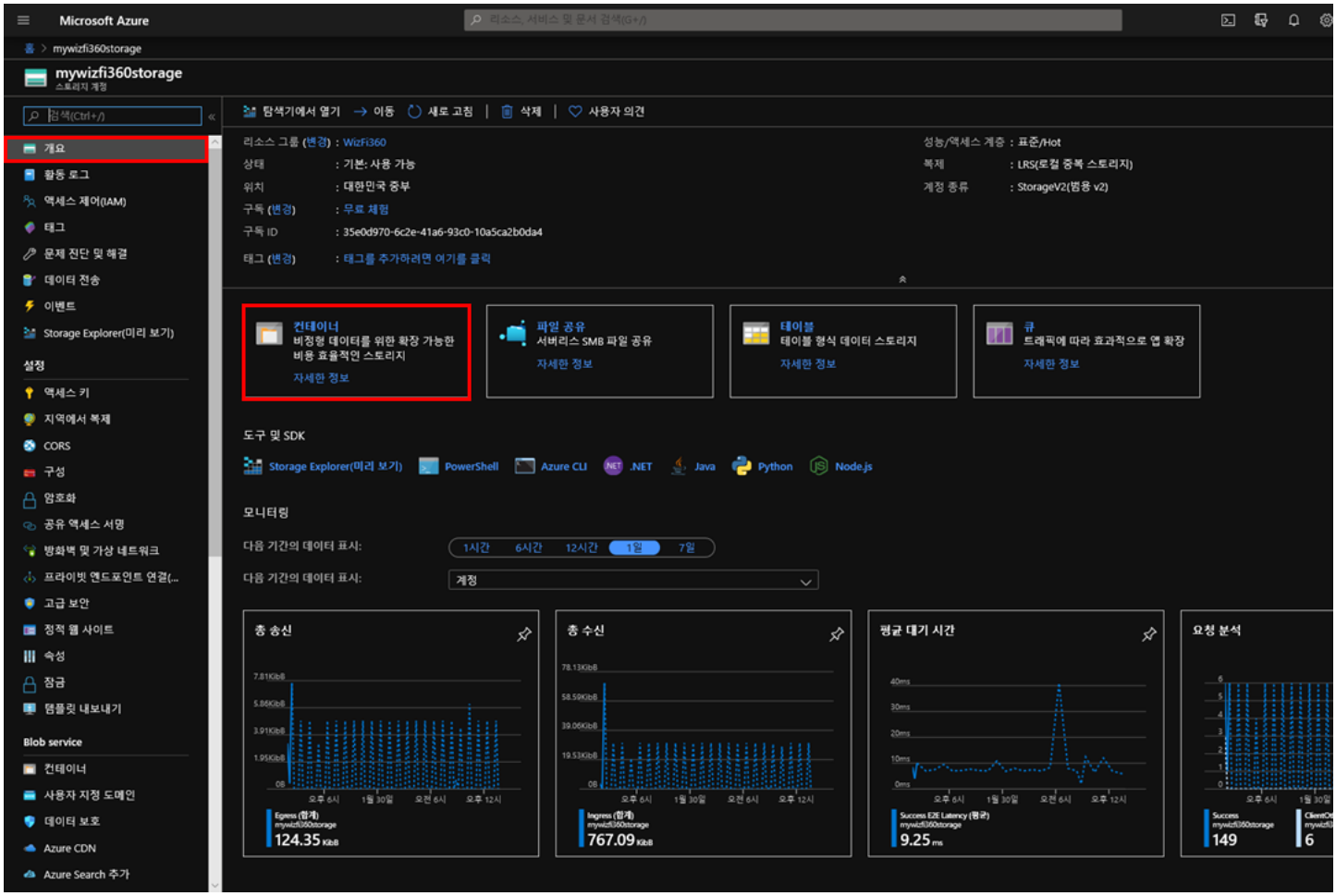




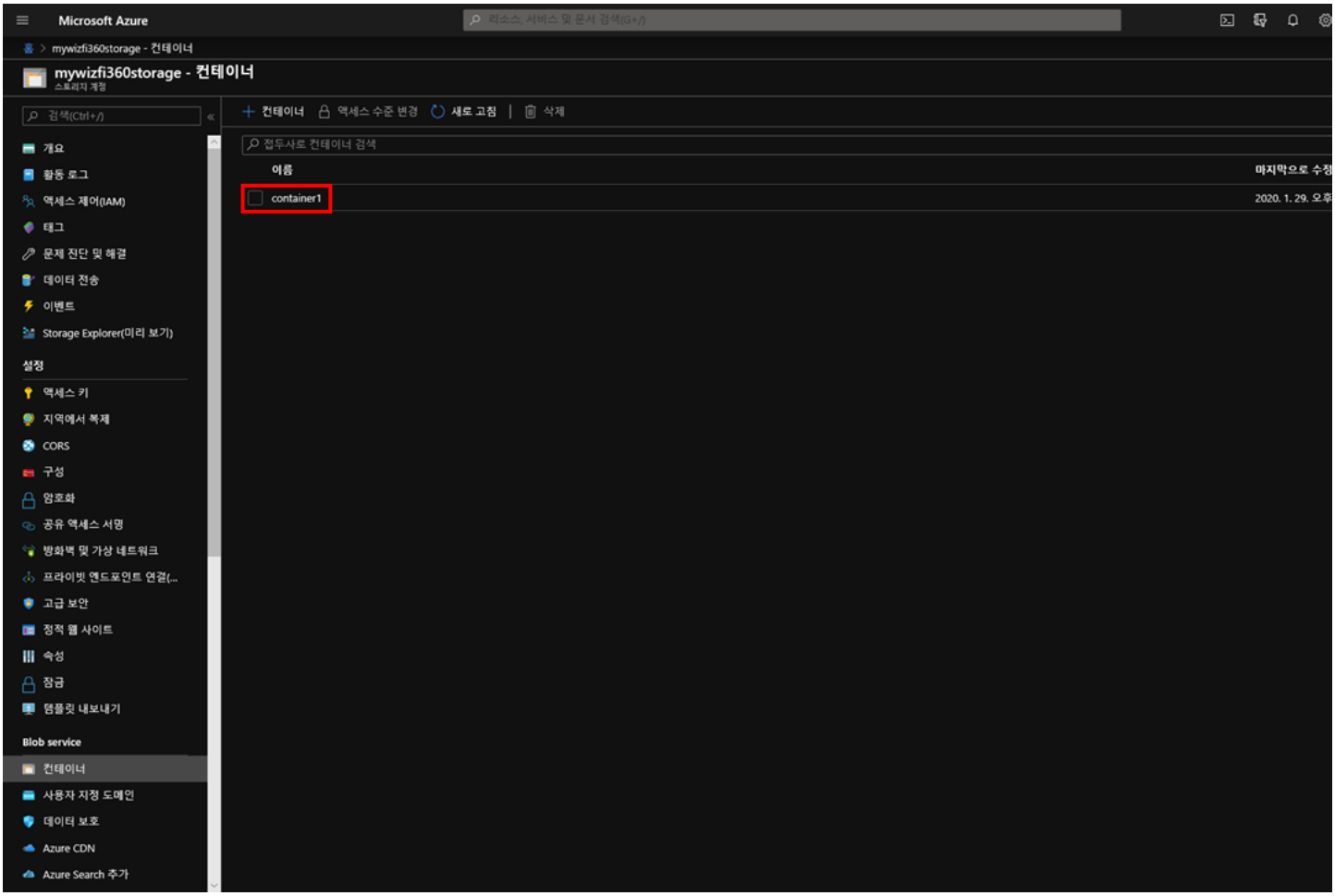


#2. 출력 확인

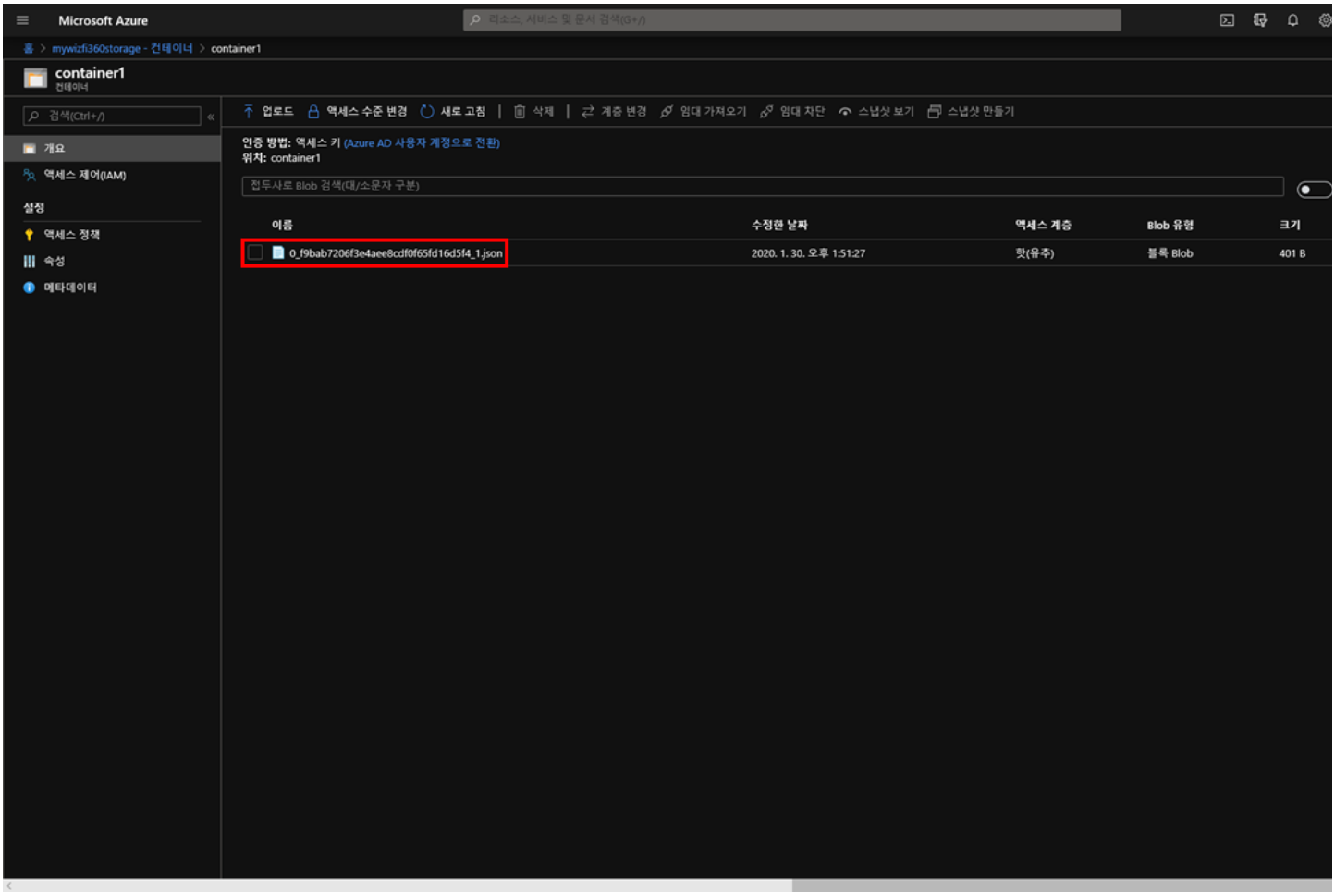
1) Blob Storage에서 **개요** > **컨테이너**를 선택합니다.



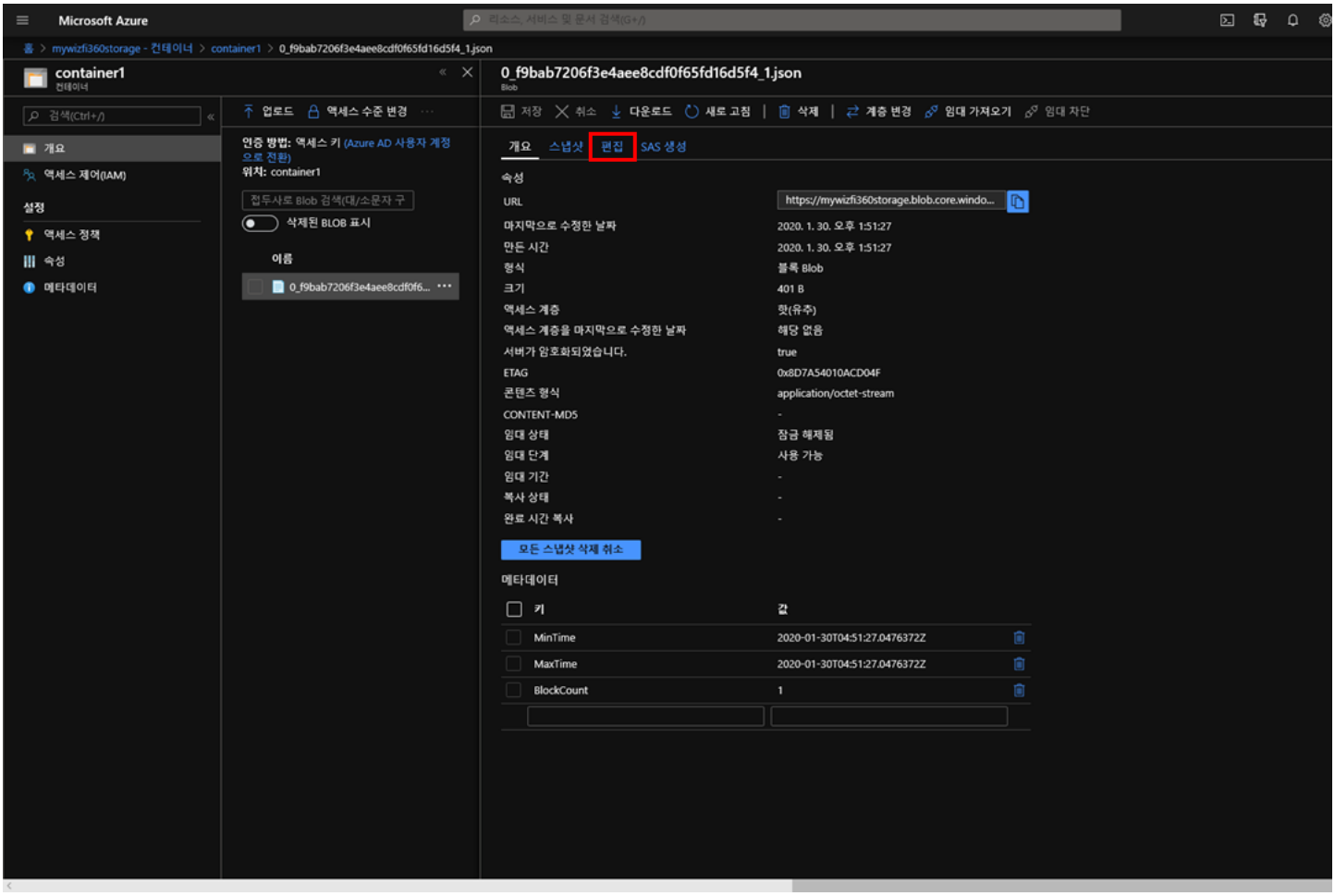
2) 목록에서 Container를 선택하여 엽니다.

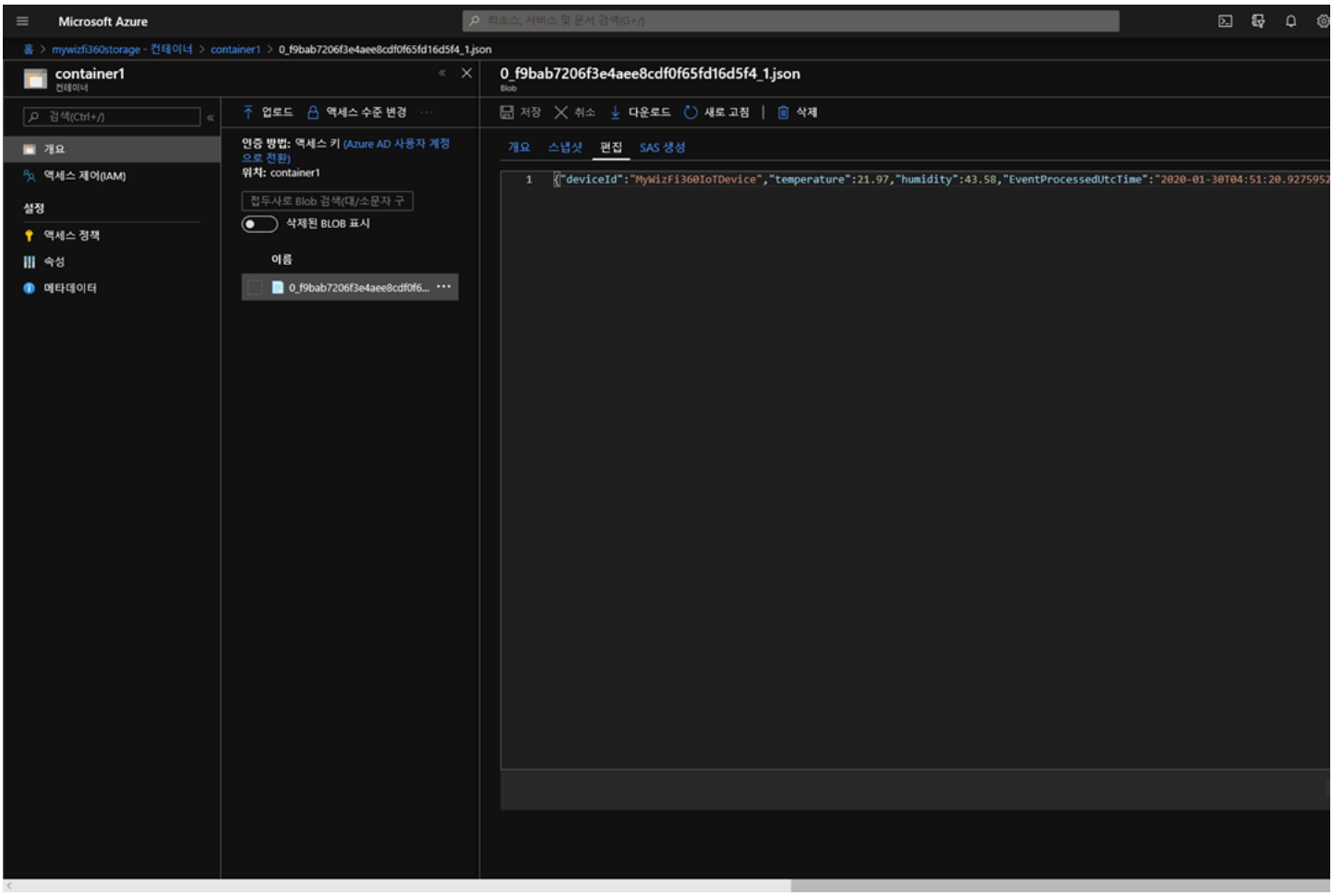


3) 목록에서 Blob을 선택하여 엽니다.



4) **편집**을 눌러 WizFi360으로부터 수신한 Data를 확인합니다.





#더 보기

1. Azure Portal을 사용하여 Stream Analytics 작업 입 · 출력 구성 및 변환 Query 정의
2. WizFi360 MQTT AT Command를 이용한 Azure IoT Hub 연동 예제

- Mbed Example
- Arduino Example

Connect to Azure IoT Hub using WizFi360 MQTT AT Command

#Getting started

Login to **Azure Portal**.

※ In this guide we will procedd with **free account**. To learn how to create IoT Hub please refer to Azure Cloud Service Introduction.

- [MS] Create IoT Hub using Azure Portal
- Create Blob storage using Azure Portal
- Create Stream Analytics using Azure Portal
- Setup Queries in Stream Analytics using Azure Portal

#Introduction

It is possible to connect to **Microsoft Azure Service** using **WizFi360**, send data to cloud and monitor current status.

Data communication will be established as following.

MQTT AT Commands will be used to connect to IoT Hub Service & send data.

Data sent to IoT Hub will be saved in Blob Storage through Stream Analytics.

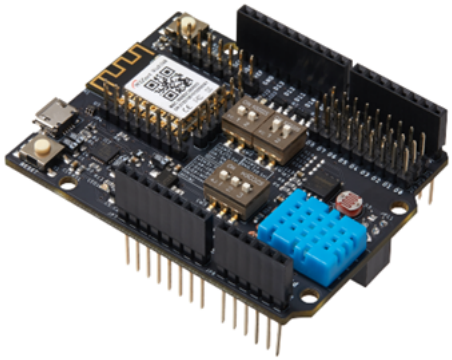
In this guide we will use WizFi360 to connect to Microsoft Azure Services using MQTT AT Command.

#Step 1: Required items

Items below are required for this guide.

#Hadrware

- Desktop or Laptop Computer
- WizFi360-EVB-Shield
- Micro 5 Pin USB Cable



#Software

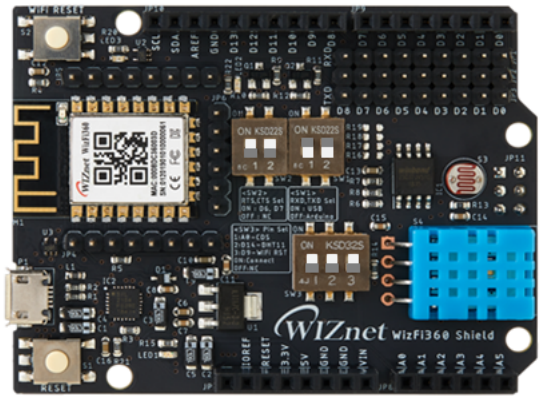
- Preferred Serial Terminal (TeraTerm, Hercules, etc . . .)

#Step 2: Device preparation

#1. Hardware preparation

WizFi360-EVB-Shield will be used in **Standalone mode**. Therefore DIP Switch and jumper cables shall be connected as following:

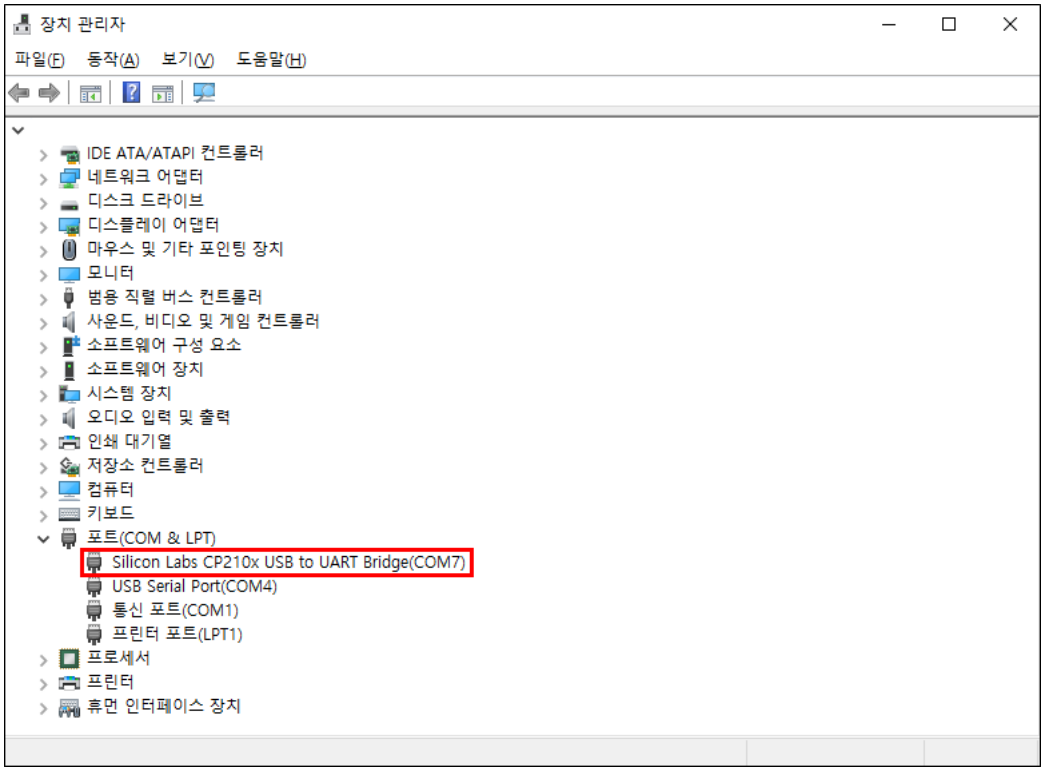
- SW1 : On
- SW2 : Off
- SW3 : Off



#2. Device connection

After connecting hardware, connect WizFi360-EVB-Shield to Desktop or Laptop using USB Cable.

Check **COM Port** from **Device Manager**.



If COM port cannot be found in Device manager, check drivers from link below.

- CP210x USB to UART Bridge VCP Drivers

#Step 3: Sample application

To learn more about **WizFi360** please refer to **Quick Start Guide**, **AT Instruction Set**.

#1. Mode setting

Command	Response
AT+CWMODE_CUR=1	OK

#2. DHCP setting

Command	Response
AT+CWDHCP_CUR=1,1	OK

#3. Connect to AP

Command	Response
AT+CWJAP_CUR="**ssid**","**password**"	WIFI CONNECTED WIFI GOT IP

Example :
AT+CWJAP_CUR="**wiznet**","**0123456789**" OK

#4. MQTT Connection setting

Command
AT+MQTTSET="**iot_hub_host_name**/**device_id**/?api-version=2018-06-30","**sas_token**","**device_id**",60

Example :
AT+MQTTSET="**MyWizFi360IoTHub.azure-devices.net**/**MyWizFi360IoTDevice**/?api-version=2018-06-30","**SharedAccessSignature sr=MyWizFi360IoTHub.az**
devices.net%2Fdevices%2FMyWizFi360IoTDevice&sig=t3R9nDS7ezMGBdb%2FNd5ktb3xQx5jx4NC02n325vRA6c%3D&se=1611895717","**MyWizFi360IoT**

For **SAS Token creation** please refer below.

- Create SAS Token using Device Explorer
- Create SAS Token using Azure IoT Explorer

#5. Topic Setting

Command	**Response**
AT+MQTTTOPIC="devices/**device_id**/messages/events/",devices/**device_id**/messages/devicebound/#	OK
Example : AT+MQTTTOPIC="devices/**MyWizFi360IoTDevice**/messages/events/","devices/**MyWizFi360IoTDevice**/messages/devicebound/#"	

#6. Connect to Broker

Command	**Response**
AT+MQTTCON=1,"**iot_hub_host_name**",8883	CONNECT
Example : AT+MQTTCON=1,"**MyWizFi360IoTHub.azure-devices.net**",8883	OK

#7. Publish Message

Note :

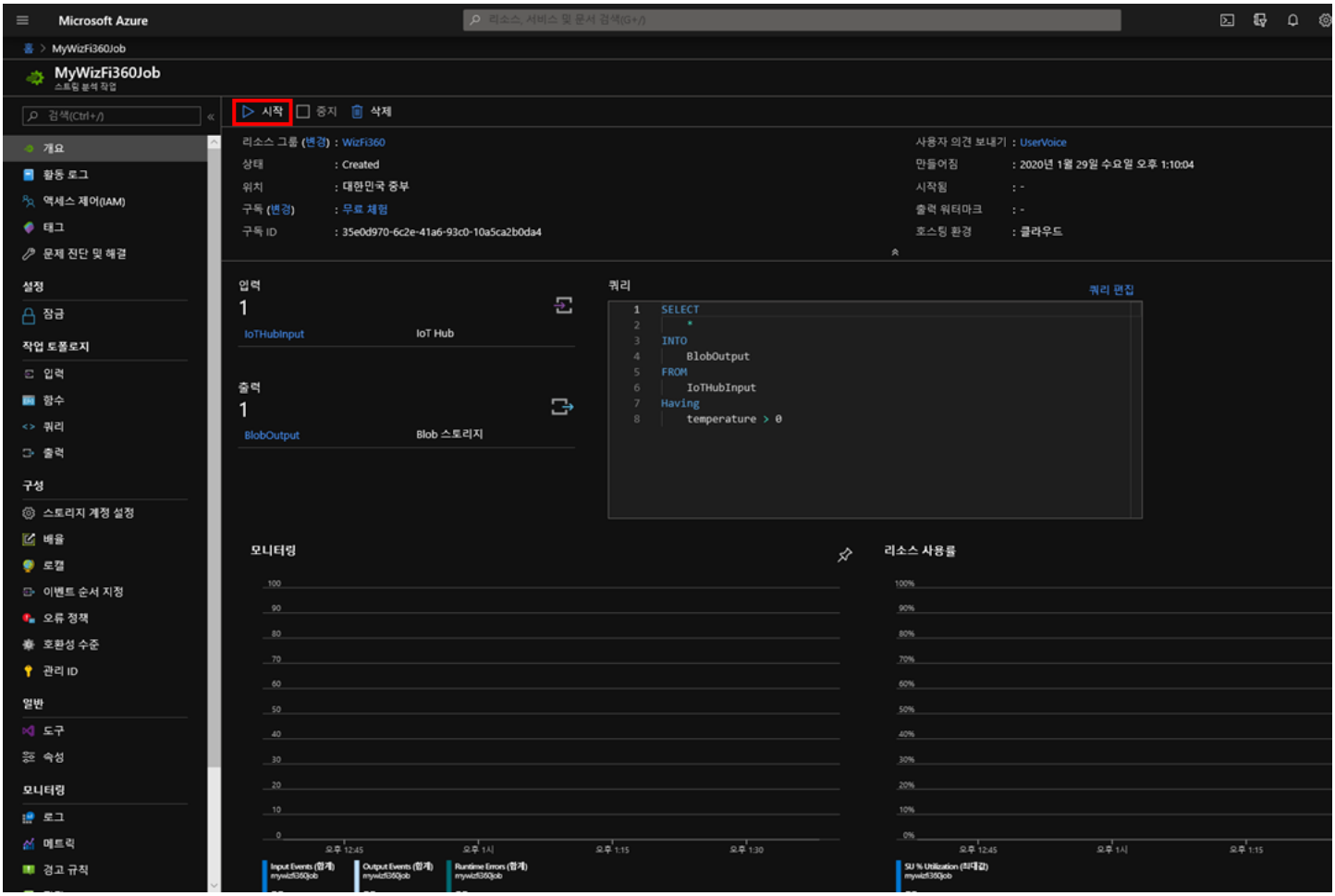
When publishing data, **Start Stream Analytics** to forward data to Blob Storage.

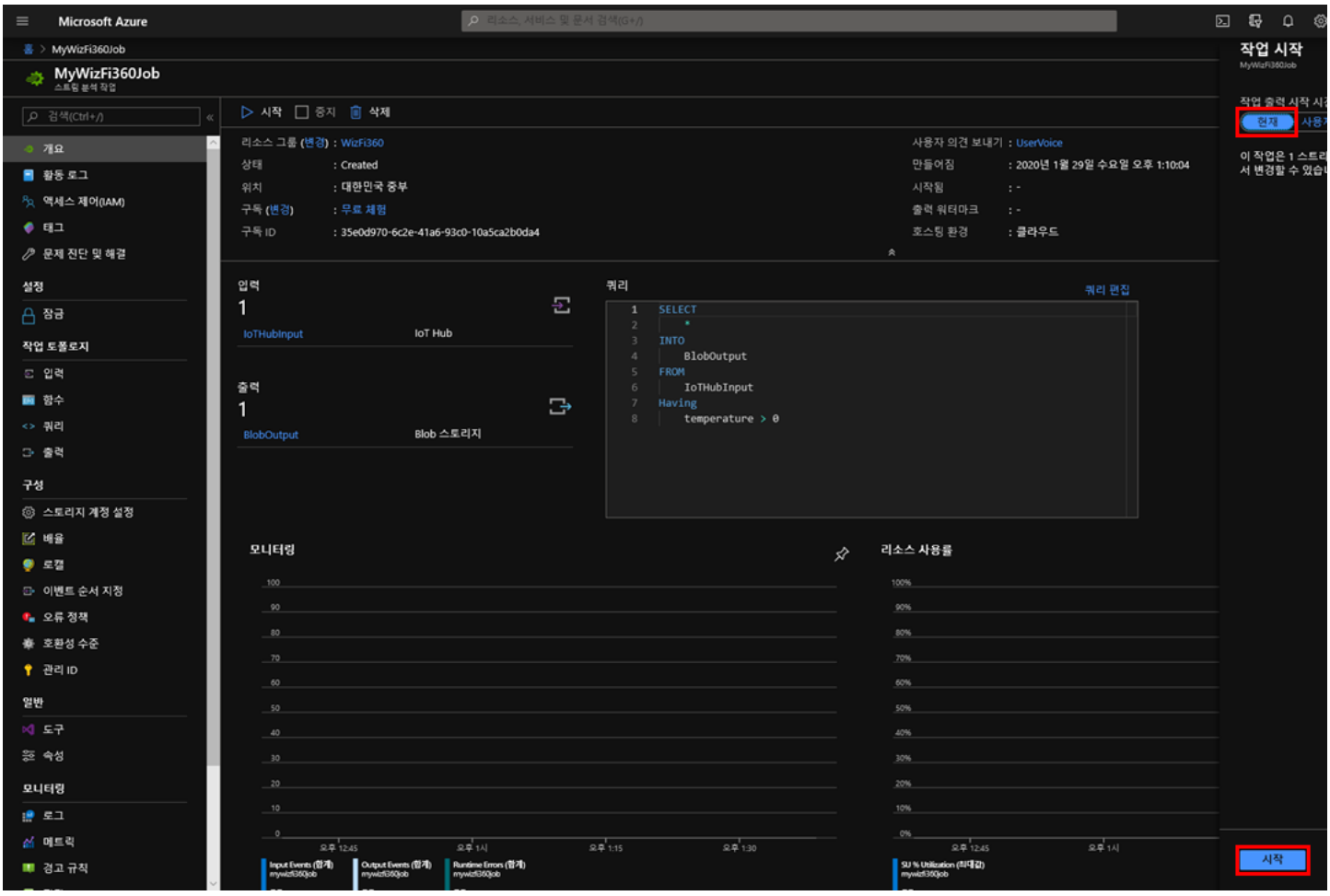
Command	**Response**
AT+MQTTPUB="publish_data"	OK
Example : AT+MQTTPUB="{"deviceId":"MyWizFi360IoTDevice","temperature":21.97,"humidity":43.58}"	

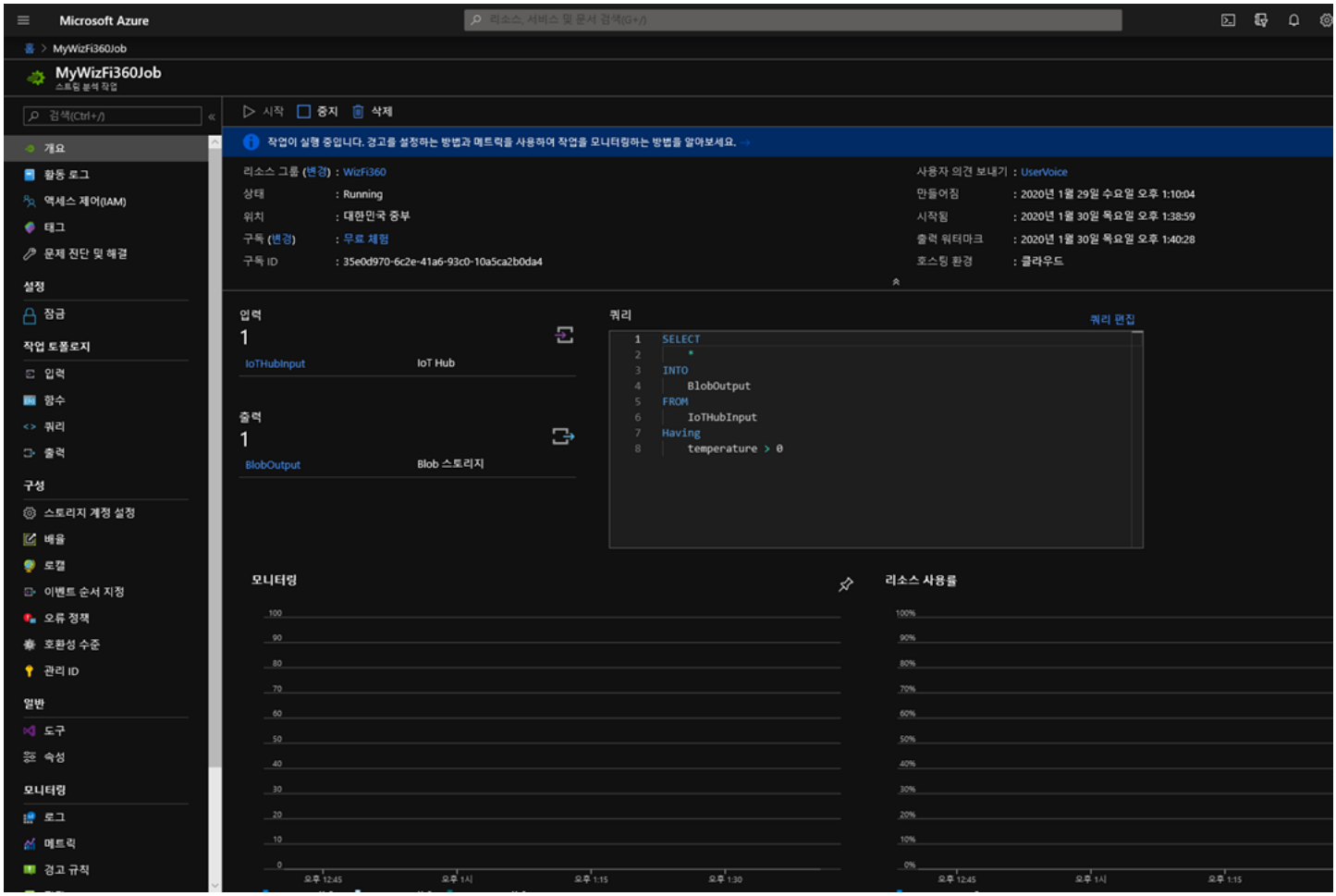
#Step 4: Results

#1. Start job in Stream Analytics

Select in Stream Analytics **Introduction** > **Start** > **Current** > **Start**.

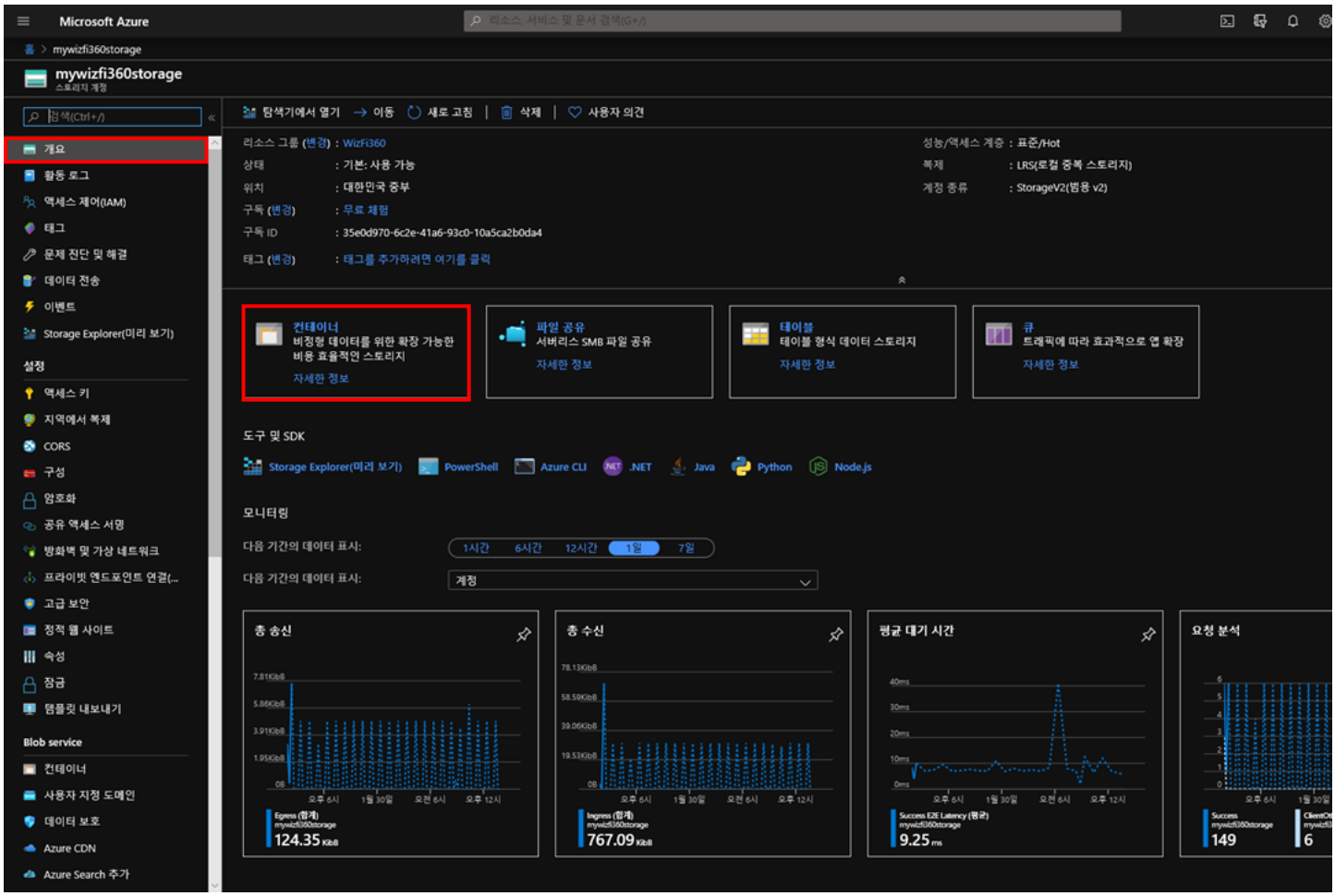

Microsoft Azure
MyWizFi360Job
MyWizFi360Job
스트림 분석 작업
시작
중지
삭제
리소스 그룹 (변경) : WizFi360
상태 : Created
위치 : 대한민국 중부
구독 (변경) : 무료 체험
구독 ID : 35e0d970-6c2e-41a6-93c0-10a5ca2b0da4
사용자 의견 보내기 : UserVoice
만들어짐 : 2020년 1월 29일 수요일 오후 1:10:04
시작됨 : -
출력 워터마크 : -
호스팅 환경 : 클라우드
입력
1
IoTHubInput IoT Hub
출력
1
BlobOutput Blob 스토리지
쿼리
쿼리 편집
SELECT
*
INTO
BlobOutput
FROM
IoTHubInput
Having
temperature > 0
모니터링
리소스 사용률
개요
활동 로그
액세스 제어(IAM)
태그
문제 진단 및 해결
설정
잠금
작업 토폴로지
입력
함수
쿼리
출력
구성
스토리지 계정 설정
배율
로캘
이벤트 순서 지정
오류 정책
호환성 수준
관리 ID
일반
도구
속성
모니터링
로그
메트릭
경고 규칙


Microsoft Azure
MyWizFi360Job
MyWizFi360Job
스트림 분석 작업
시작
중지
삭제
리소스 그룹 (변경) : WizFi360
상태 : Created
위치 : 대한민국 중부
구독 (변경) : 무료 체험
구독 ID : 35e0d970-6c2e-41a6-93c0-10a5ca2b0da4
사용자 의견 보내기 : UserVoice
만들어짐 : 2020년 1월 29일 수요일 오후 1:10:04
시작됨 : -
출력 워터마크 : -
호스팅 환경 : 클라우드
입력
1
IoTHubInput IoT Hub
출력
1
BlobOutput Blob 스토리지
쿼리
쿼리 편집
SELECT
*
INTO
BlobOutput
FROM
IoTHubInput
Having
temperature > 0
모니터링
리소스 사용률
작업 시작
MyWizFi360Job
현재
시작
개요
활동 로그
액세스 제어(IAM)
태그
문제 진단 및 해결
설정
잠금
작업 토폴로지
입력
함수
쿼리
출력
구성
스토리지 계정 설정
배율
로캘
이벤트 순서 지정
오류 정책
호환성 수준
관리 ID
일반
도구
속성
모니터링
로그
메트릭
경고 규칙

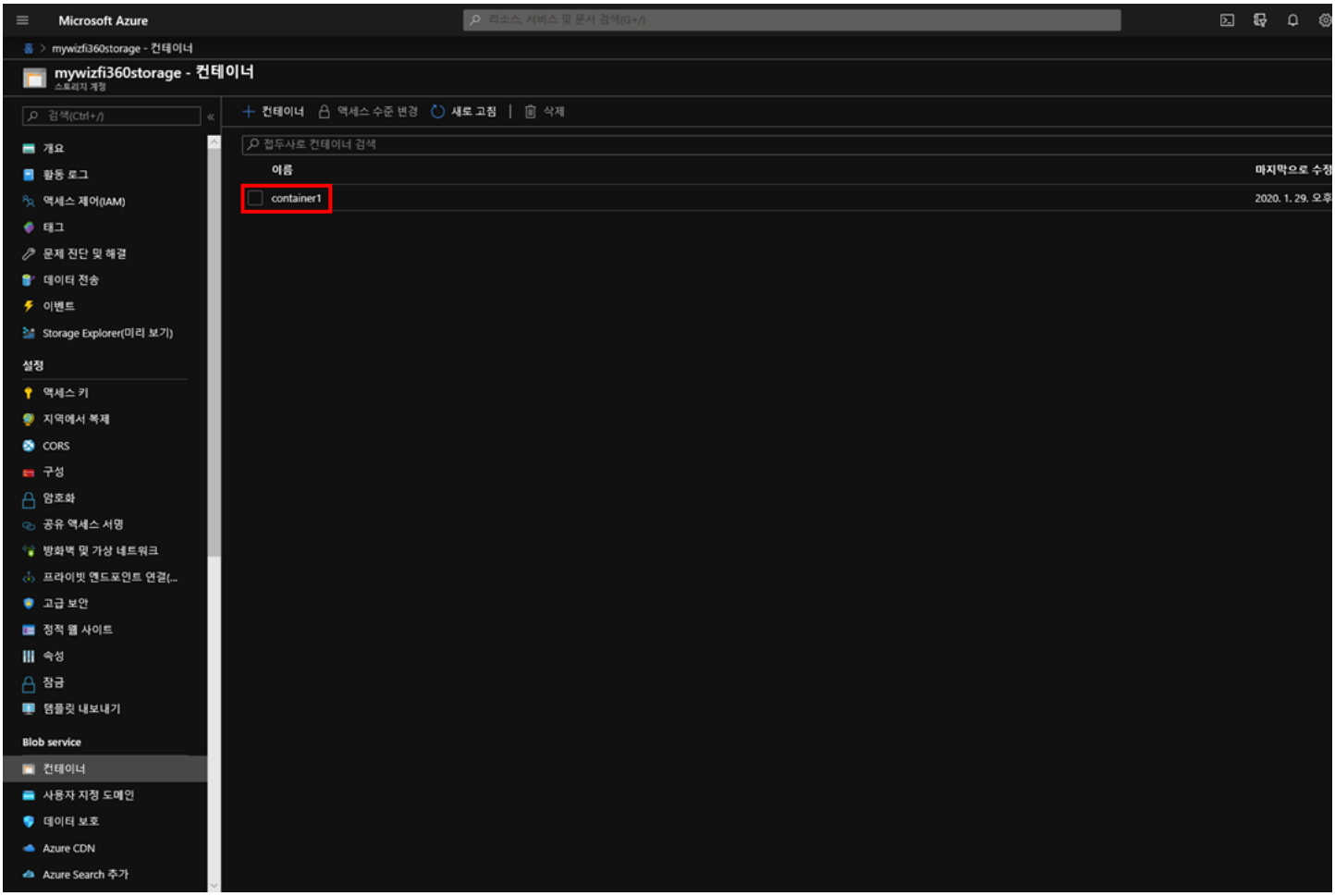


#2. Check output

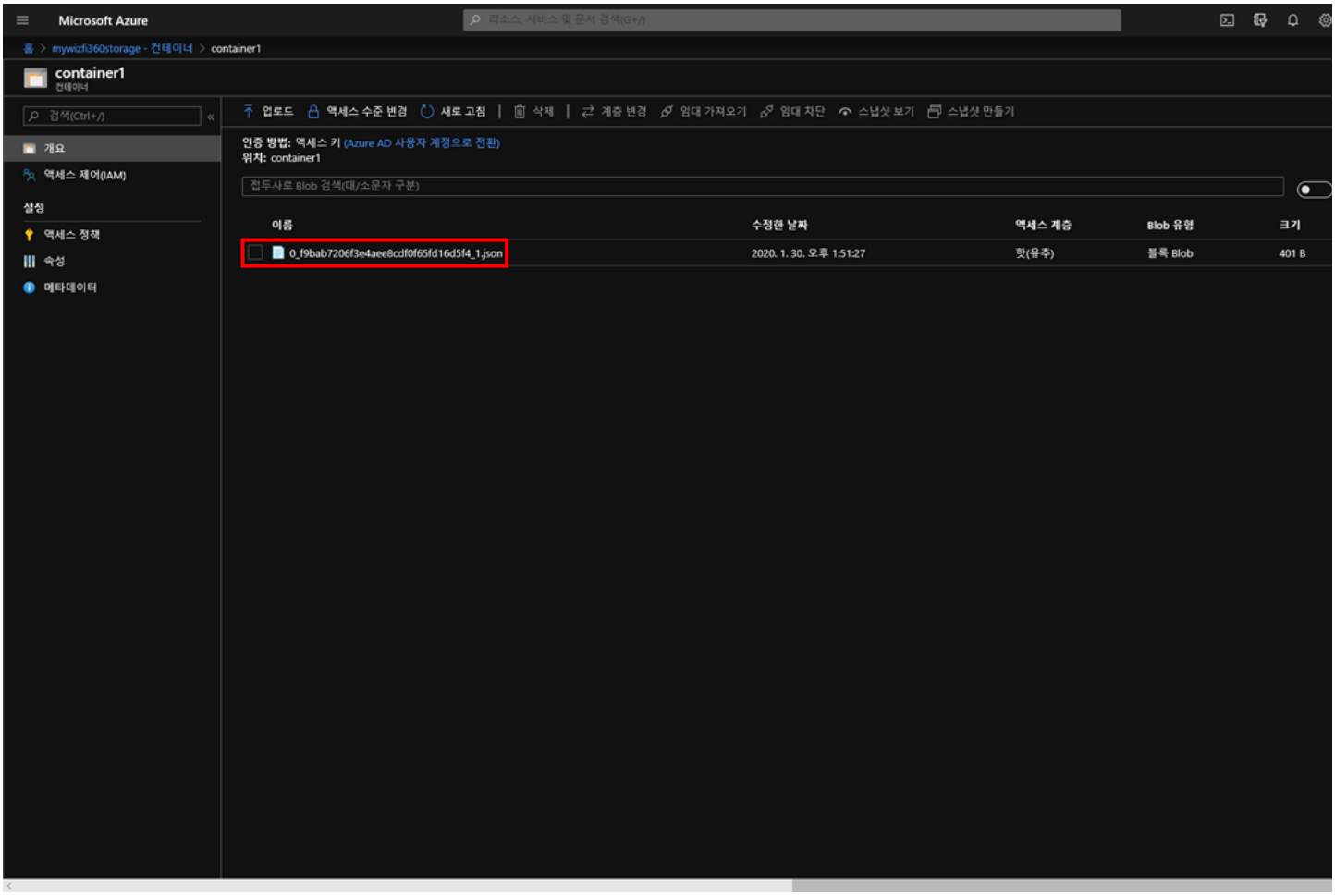
1) Select in Blob Storage **Introduction** > **Container**.



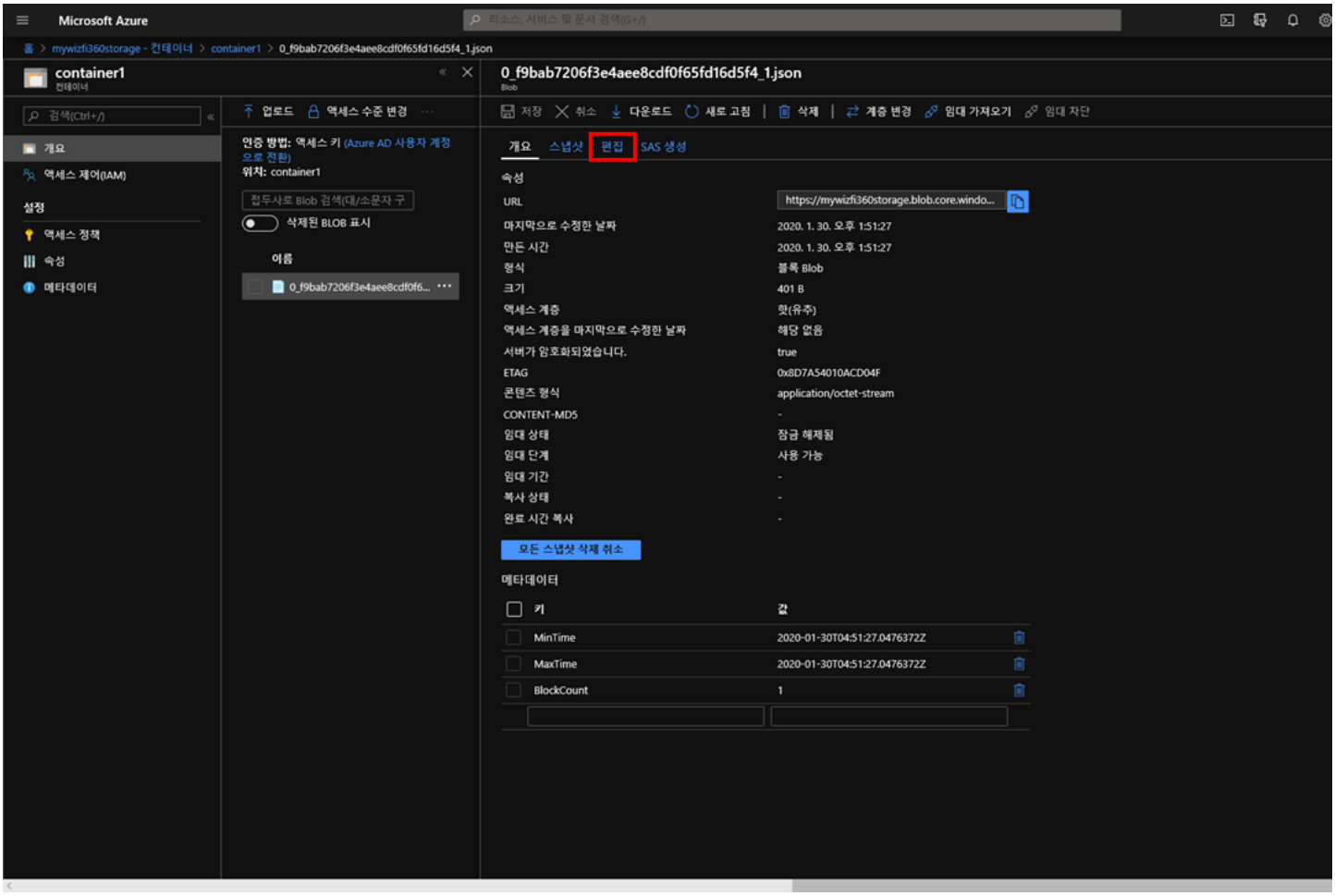
2) Select Container in List.

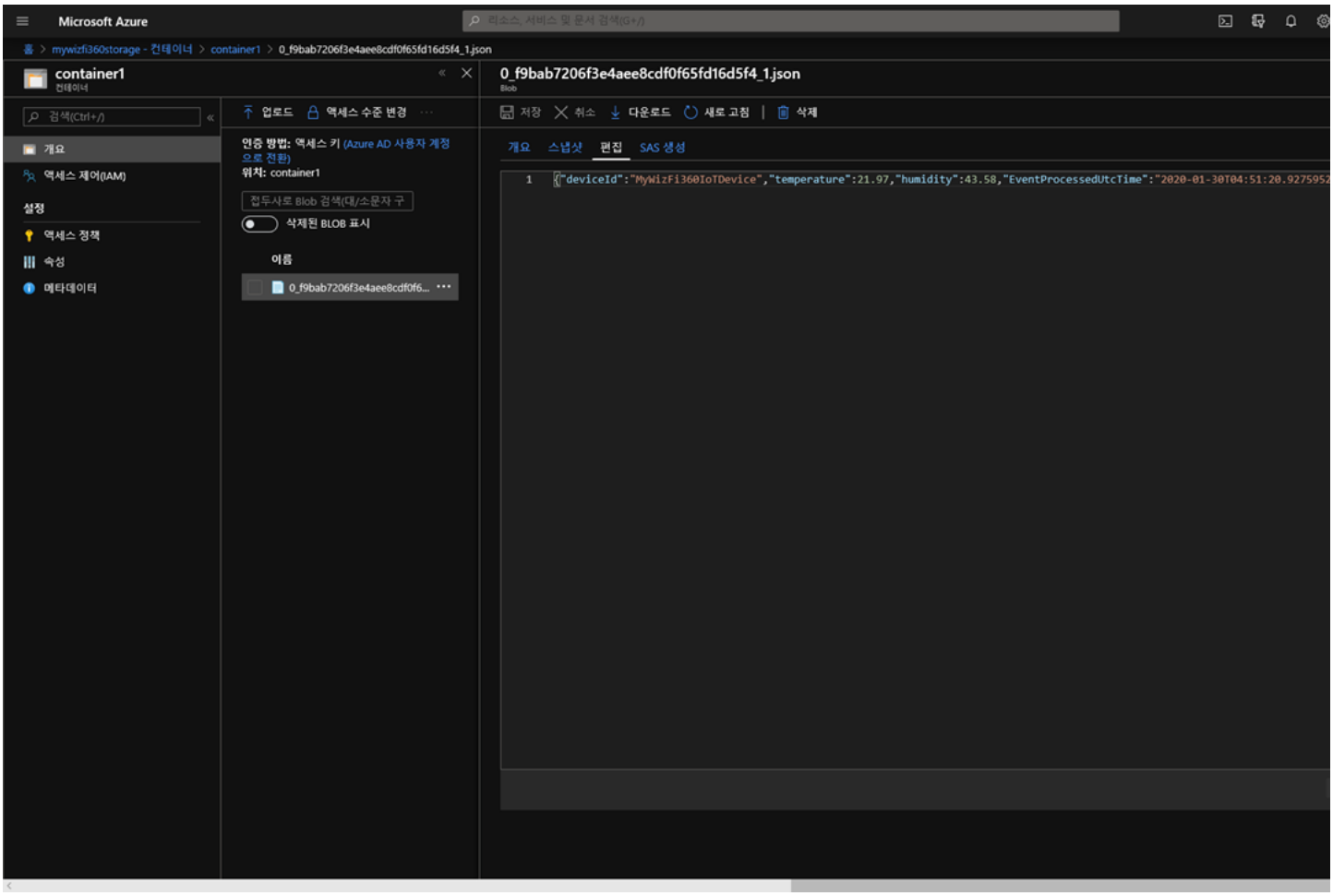


3) Select Blob in list.



4) Click **Edit** to check data received from WizFi360.


Microsoft Azure
리소스, 서비스 및 문서 검색(G+/)
홈 > mywizfi360storage - 컨테이너 > container1 > 0_f9bab7206f3e4aee8cdf0f65fd16d5f4_1.json
container1
컨테이너
0_f9bab7206f3e4aee8cdf0f65fd16d5f4_1.json
Blob
검색(Ctrl+/)
개요
액세스 제어(IAM)
설정
액세스 정책
속성
메타데이터
업로드
액세스 수준 변경
저장
취소
다운로드
새로 고침
삭제
계층 변경
임대 가져오기
임대 차단
인증 방법: 액세스 키 (Azure AD 사용자 계정으로 전환)
위치: container1
접두사로 Blob 검색(대/소문자 구
삭제된 BLOB 표시
이름
0_f9bab7206f3e4aee8cdf0f6...
개요
스냅샷
편집
SAS 생성
속성
URL
https://mywizfi360storage.blob.core.windo...
마지막으로 수정한 날짜
2020. 1. 30. 오후 1:51:27
만든 시간
2020. 1. 30. 오후 1:51:27
형식
블록 Blob
크기
401 B
액세스 계층
핫(유추)
액세스 계층을 마지막으로 수정한 날짜
해당 없음
서버가 암호화되었습니다.
true
ETAG
0x8D7A54010ACD04F
콘텐츠 형식
application/octet-stream
CONTENT-MD5
-
임대 상태
잠금 해제됨
임대 단계
사용 가능
임대 기간
-
복사 상태
-
완료 시간 복사
-
모든 스냅샷 삭제 취소
메타데이터
키
값
MinTime
2020-01-30T04:51:27.0476372Z
MaxTime
2020-01-30T04:51:27.0476372Z
BlockCount
1


Microsoft Azure
리소스, 서비스 및 문서 검색(G+/)
홈 > mywizfi360storage - 컨테이너 > container1 > 0_f9bab7206f3e4aee8cdf0f65fd16d5f4_1.json
container1
컨테이너
0_f9bab7206f3e4aee8cdf0f65fd16d5f4_1.json
Blob
검색(Ctrl+/)
개요
액세스 제어(IAM)
설정
액세스 정책
속성
메타데이터
업로드
액세스 수준 변경
저장
취소
다운로드
새로 고침
삭제
인증 방법: 액세스 키 (Azure AD 사용자 계정으로 전환)
위치: container1
접두사로 Blob 검색(대/소문자 구
삭제된 BLOB 표시
이름
0_f9bab7206f3e4aee8cdf0f6...
개요
스냅샷
편집
SAS 생성
1 {"deviceId":"MyWizFi360IoTDevice","temperature":21.97,"humidity":43.58,"EventProcessedUtcTime":"2020-01-30T04:51:20.9275952

WizFi360 Azure AT Command를 이용하여 Azure IoT Hub에 연결

#시작하기 전에

#Hardware Requirement

- Desktop or laptop computer
- MicroUSB 케이블
- WizFi360-EVB-Shield

#Software Requirement

- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Azure IoT Explorer

  Azure Portal에 Login을 합니다. 계정이 없는 경우, 계정 생성 후에 Login을 진행합니다.

  ※ 본 문서는 **체험 계정**으로 진행합니다.

  Azure Portal을 사용하여 IoT Hub 만들기 등 앞선 일련의 과정에 대하여 Azure Cloud 소개를 참조하시기 바랍니다.

  - Azure Portal을 사용하여 IoT Hub 만들기
  - Azure Portal을 사용하여 Blob Storage 만들기
  - Azure Portal을 사용하여 Stream Analytics 만들기
  - Azure Portal을 사용하여 Stream Analytics 작업 입 · 출력 구성 및 변환 Query 정의

#소개

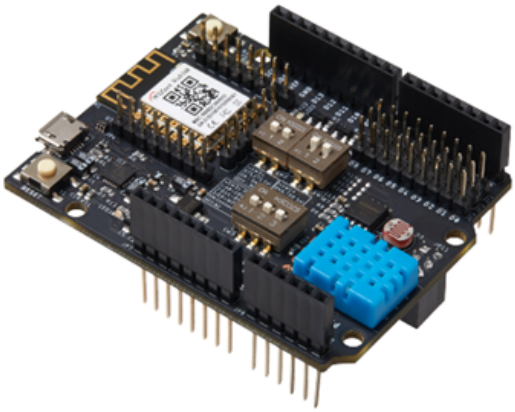
Microsoft Azure 는 Microsoft 의 클라우드 컴퓨팅 서비스입니다. Microsoft Azure 의 서비스에 WizFi360 을 연동하여 데이터를 클라우드로 전송하고, 모니터링 할 수 있습니다.

본 문서에서는 WizFi360 이용하여 MS Azure Services에 연결 방법에 대한 가이드를 제공합니다. 이 프로세스는 다음 단계로 구성됩니다.

- Azure IoT Hub 준비
- IoT 디바이스 등록
- Azure IoT와 연결 및 데이터 통신

Azure IoT Hub 준비와 IoT 디바이스 등록 과정 대해 Azure Cloud 소개 참조하시기 바랍니다.

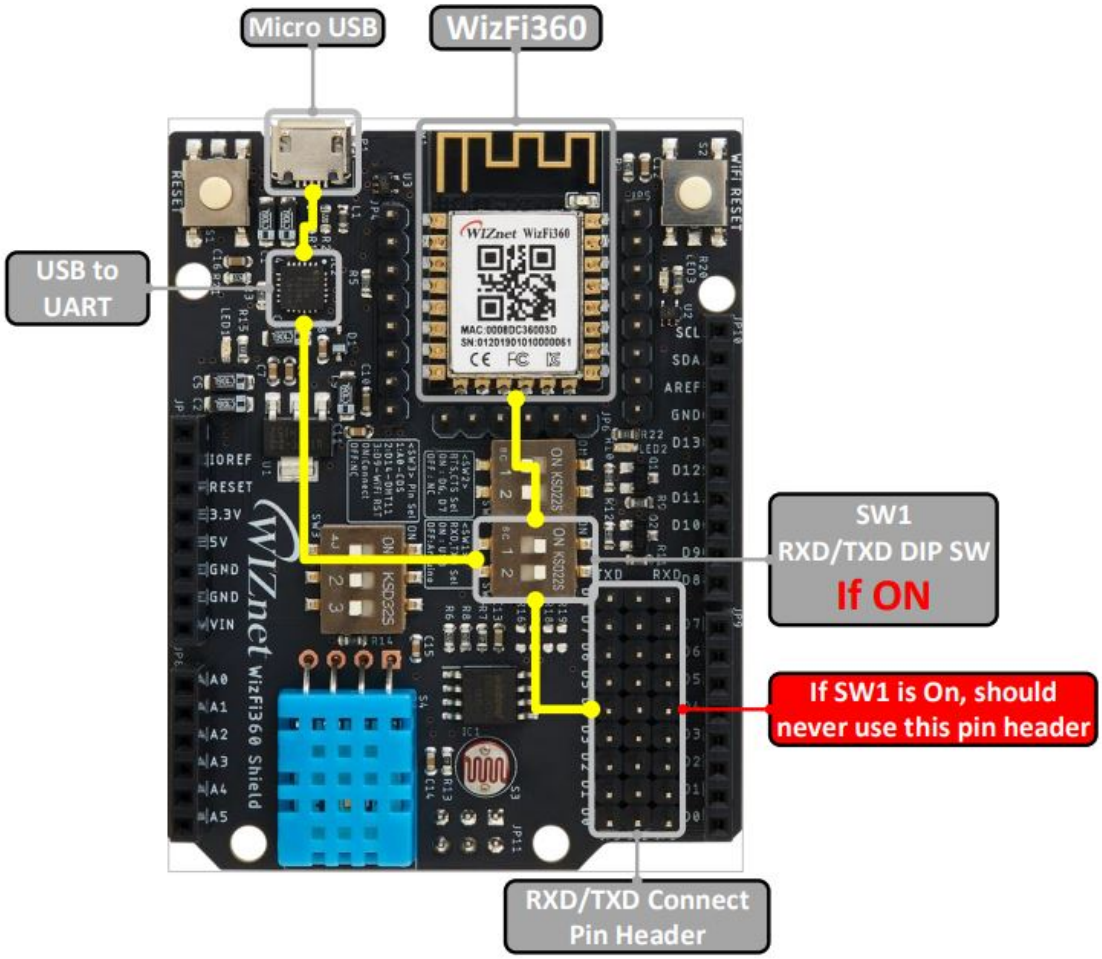
WiFi모듈 테스트를 위해 WizFi360-EVB-Shield Evaluation 보드를 사용되었습니다.



#디바이스 준비

#하드웨어 설정

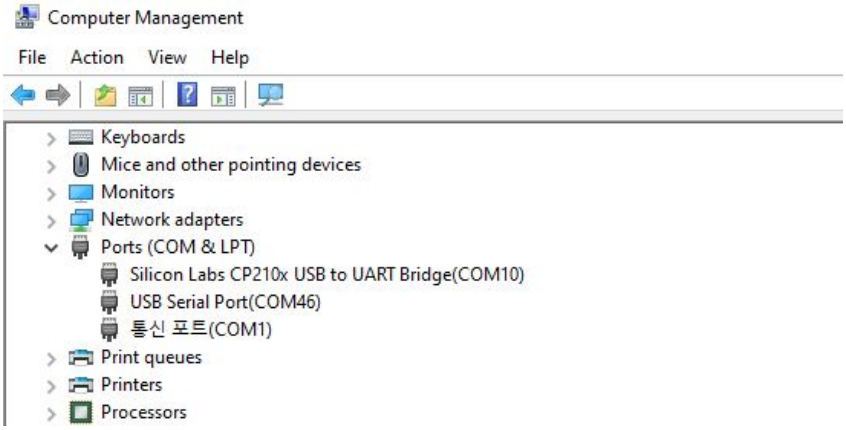
본 문서에서 WizFi360-EVB-Shield가 standalone mode에서 사용됩니다. 따라서 UART를 위해 MicroUSB를 사용할겁니다. MicroUSB 사용하는경우 SW1을 ON 시키고 MicroUSB 연결해야됩니다.



#디바이스 연결

하드웨어 설정 후 MicroUSB 이용하여 PC와 연결합니다. PC운영체제에서 보드와 연결된 COM 포트를 확인할 수 있습니다.

윈도우 운영체제의 경우, 장치 관리자(Device Manager)에서 COM 포트를 확인할 수 있습니다.



장치 관리자에서 COM 포트를 확인할 수 없는 경우, 다음 링크에서 드라이버를 다운로드하여 설치하시기 바랍니다.

- Silicon Labs CP210x USB to UART Driver

#AT 명령어

#1. Set current WiFi mode (not saved in flash)

AT Command: AT+CWMODE_CUR Syntax:

Type	Command	Response
Query	AT+CWMODE_CUR?	+CWMODE:<mode> OK
Set	AT+CWMODE_CUR=<mode>	OK

Defined values:

Mode	Value
1	Station mode
2	SoftAP mode (factory default)
3	Station+SoftAP mode

#2. Enable DHCP

AT Command: AT+CWDHCP_CUR Syntax:

Type	Command	Response
Query	AT+CWDHCP_CUR?	+CWDHCP_CUR:<para> OK
Set	AT+CWMODE_CUR=<para>,<en>	OK

Defined values:

Parameter	Value
0	SoftAP DHCP 와 Station DHCP 를 disable 한다.
1	SoftAP DHCP 는 enable 하고 Station DHCP 는 disable 한다.
2	2: SoftAP DHCP 는 disable 하고 Station DHCP 는 enable 한다.
3	SoftAP DHCP 와 Station DHCP 를 enable 한다. (factory default)

#3. List available APs

AT Command: AT+CWLAP Syntax:

Type	Command	Response
Query	AT+CWLAP	+CWLAP:([<ecn>,<ssid>,<rssi>,<mac>,<channel>,<wps>])

Defined values:

Parameter	Value
<ecn>	0: Open 1: WEP 2: WPA_PSK 3: WPA2_PSK 4:WPA_WPA2_PSK
<ssid>	string parameter. AP의 ssid
<rssi>	signal strength
<mac>	string parameter. AP의 mac
<wps>	0: WPS는 disable된다 1: WPS는 enable된다

#4. Connect to AP

AT Command: AT+CWJAP_CUR Syntax:

Type	Command	Response
Set	AT+CWJAP_CUR=<ssid>,<pwd>,[<bssid>]	+CWJAP_CUR:<ssid>,<bssid>,<channel>,<rssi> OK

Defined values:

Parameter	Value
<ssid>	string parameter. Target AP의 ssid. MAX: 32 bytes
<pwd>	string parameter. Target AP의 password. MAX: 64-byte ASCII
<bssid>	string parameter, target AP 의 MAC address, 같은 SSID 를 가진 여러 개의 AP 들이 있을 때 사용된다.

#5. Azure IoT Hub configuration set

AT Command: AT+AZSET

Syntax:

Type	Command	Response
Set	AT+AZSET=<iothub_name>,<device_id>,<device_key>	OK

Defined values:

Parameter	Value
<hub ID>	string parameter. IoT Hub의 ID
<device ID>	string parameter. IoT Device의 ID
<key>	string parameter, IoT Device의 Key

#6. Set MQTT Topic

AT Command: AT+MQTTTOPIC

Syntax:

Type	Command	Response
Set	AT+MQTTTOPIC=<publish topic>,<subscribe topic>,<subscribe topic2>,<subscribe topic3>	OK

Defined values:

Parameter	Value
<publish topic>	string parameter, WizFi360 이 publish 하는 topic
<subscribe topic>	string parameter, WizFi360 이 subscribe 하는 topic
<subscribe topic2>	string parameter, WizFi360 이 subscribe 하는 topic
<subscribe topic3>	string parameter, WizFi360 이 subscribe 하는 topic

Note:

- 이 command 는 broker 에 연결하기전에 설정되어야 합니다.
- <subscribe topic2> 와 <subscribe topic3>는 Firmware v1.0.5.0 이후 version 부터 사용가능 합니다.

#7. Connect to Azure

AT Command: AT+AZCON

Syntax:

Type	Command	Response
Set	AT+AZCON	CONNECT OK

Note: • 이 command 를 전송하기전에 AT+AZSET command 와 AT+MQTTTOPIC command 를 설정합니다. • Connect 이후 AT+MQTTPUB command 를 통해 Azure Sever 에 데이터를 전송합니다. • 자세한 내용은 https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support 를 참조하세요.

#8. Publish a message

AT Command: AT+MQTTPUB

Syntax:

Type	Command	Response
Set	AT+MQTTPUB=<message>	OK

Note: • 이 command 는 MQTT 가 연결되어 있을 때 사용됩니다. • Publish 한 data 의 topic 은 AT+MQTTTOPIC command 에 의해 결정되며, 사용자는 broker 에 연결하기전에 topic 을 설정합니다.

#동작 예제

#시리얼 터미널 연결 및 실행

시리얼 터미널 프로그램을 실행하여 디바이스 연결 단계에서 확인한 보드의 COM 포트와 Baudrate 115200을 선택하여 시리얼 포트를 연결합니다.

디버그 메시지 출력용 시리얼 포트 설정 정보: 115200-8-N-1, None.

#WizFi360모듈의 WiFi 설정

#1. Set Wifi station mode

Command	Response
AT+CWMODE_CUR=1 // station mode	OK

#2. Set DHCP Enable

Command	Response
AT+CWDHCP_CUR=1,1 // DHCP enable on Station mode	OK

#3. Get possible Wi-Fi AP List for connection

Command	Response
AT+CWLAP	+CWLAP : (3,"ssid",-57,"mac address",1,1) // encryption method, ssid, rssi, mac address, channel, wps

#4. Connect to Wi-Fi AP

Command	Response
AT+CWJAP_CUR="ssid","password"	WIFI CONNECTED WIFI GOT IP

#5. Query WizFi360 device' IP address

Command	Response
AT+CIPSTA_CUR?	+CIPSTA_CUR:ip:”192.168.10.13” +CIPSTA_CUR:gateway:”192.168.10.1” +CIPSTA_CUR:network:”255.255.255.0”

#다음 AT 명령을 실행하여 Azure 서비스에 연결

#1. Set Azure connection

Command	Response
AT+AZSET="iothub_name","device_id","device_key"	OK

#2. Set MQTT Topic

Command	Response
AT+MQTTTOPIC="/devices/{device_id}/messages/events/","/devices/{device_id}/messages/devicebound/#"	OK
Example AT+MQTTTOPIC="/devices/testDevice/messages/events/","/devices/testDevice/messages/devicebound/#"	

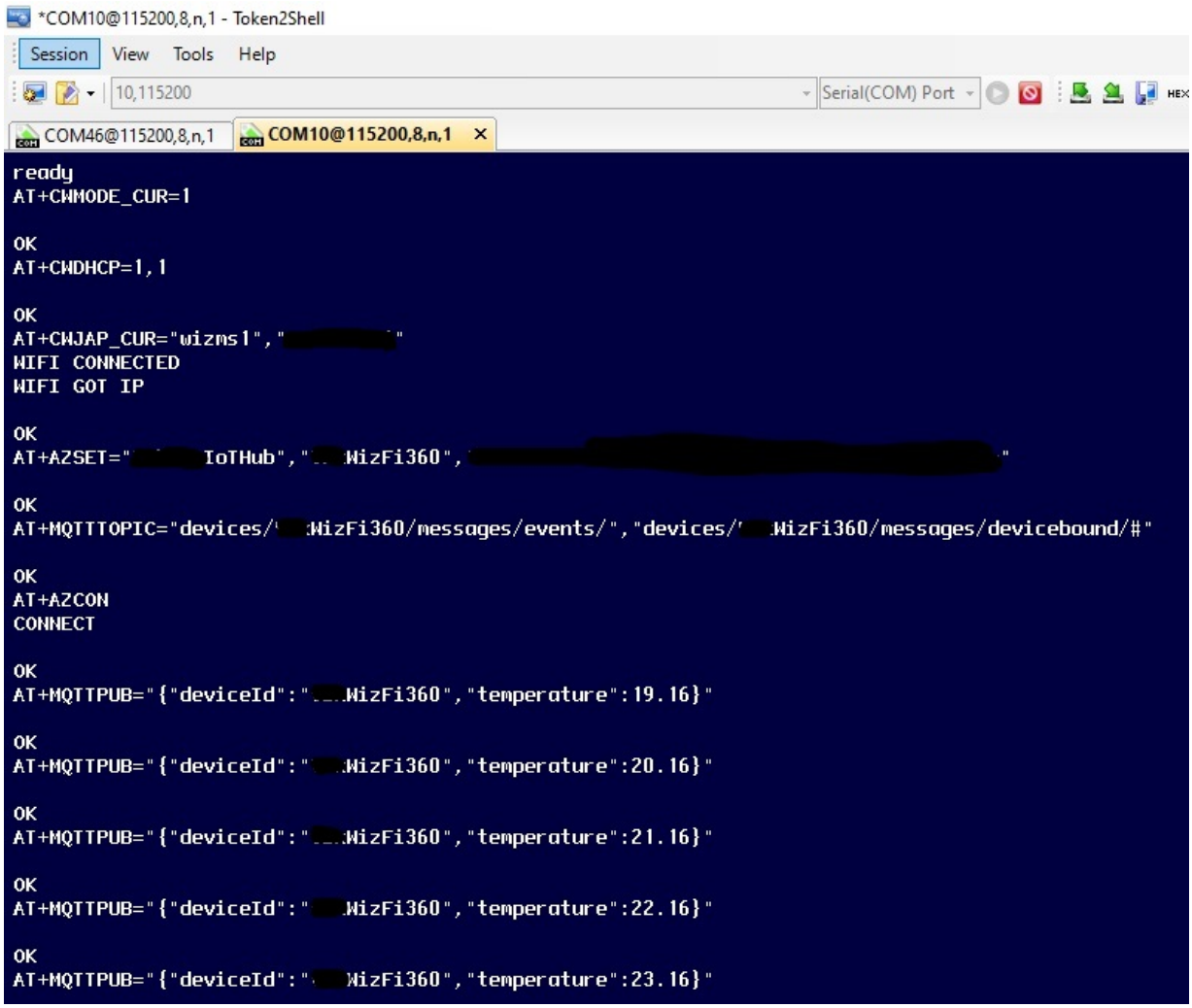
> Note: MQTT Topic follows the rules defined in Azure IoT Hub. Refer to document: Communicate with your IoT hub using the MQTT protocol: Using the MQTT protocol directly (as a device)

#3. Connect to Azure

Command	Response
AT+AZCON	OK

#4. Publish data

Command	Response
AT+MQTTPUB="{"deviceId":"WizFi360","temperature":28.16,"humidity":46.04}"	OK

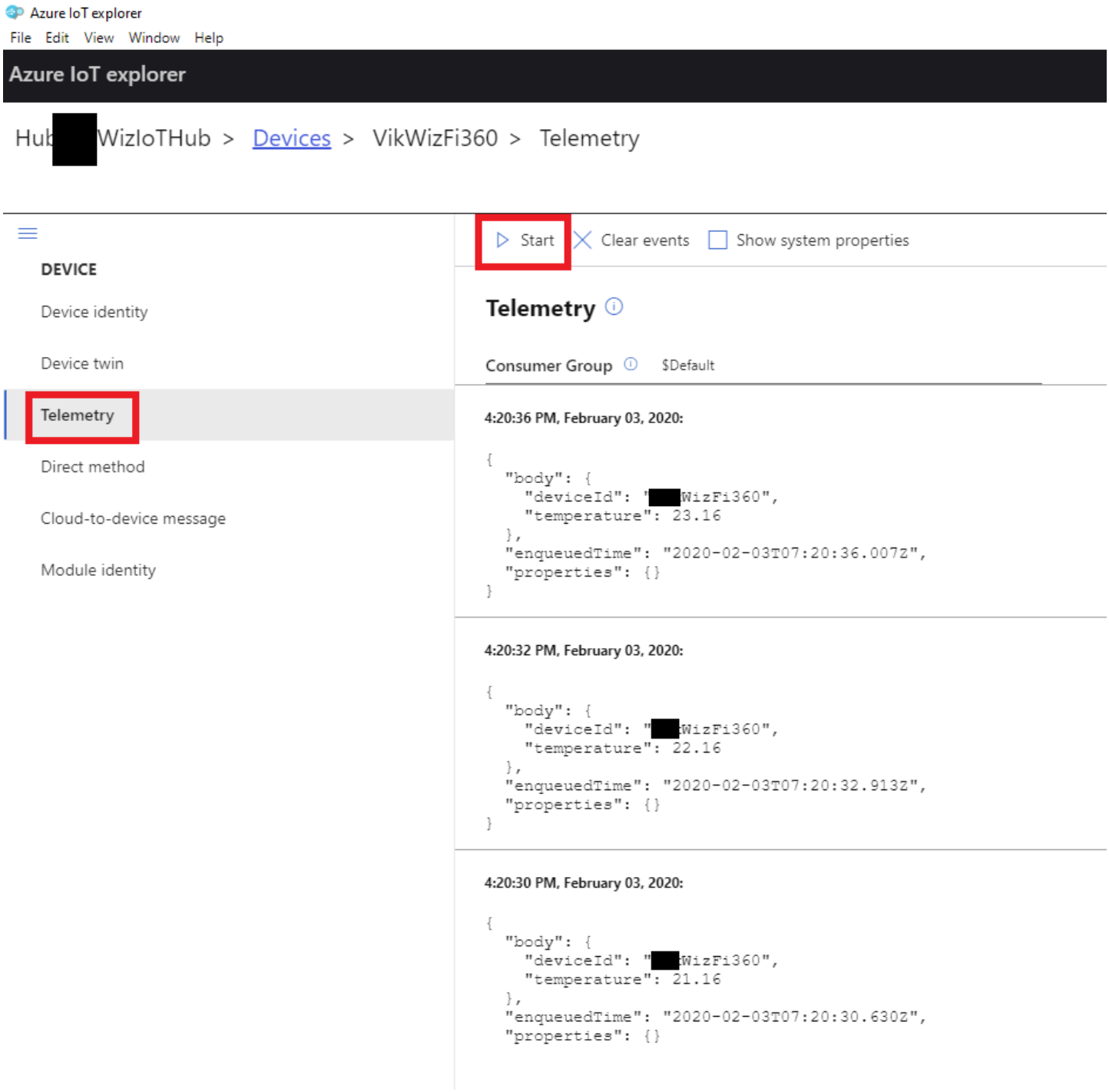


#동작 예제 결과

1. IoT Explorer 에서 Telemetry Section안에 "Start" 버튼을 누릅니다.

   MQTTPUB 명령을 통해 메시지를 보내기 전에 "Start" 버튼을 눌러야 합니다.

2. 터미널에서 MQTTPUB command으로 수신한 데이터를 확인 할 수 있습니다.



#다음 단계

1. Azure Portal을 사용하여 Stream Analytics 작업 입 · 출력 구성 및 변환 Query 정의
2. WizFi360 Azure AT Command를 이용한 Azure IoT Hub 연동 예제
   - Mbed Example
   - Arduino Example

Connect to Azure IoT Hub using WizFi360 Azure AT Command

#Getting Started

#Hardware Requirement

- Desktop or laptop computer
- MicroUSB cable
- WizFi360-EVB-Shield

#Software Requirement

- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Azure IoT Explorer

  Login to **Azure Portal**.

  ※ In this guide we will procedd with **free account**. To learn how to create IoT Hub please refer to Azure Cloud Service Introduction.

  - [MS] Create IoT Hub using Azure Portal
  - Create Blob storage using Azure Portal
  - Create Stream Analytics using Azure Portal
  - Setup Queries in Stream Analytics using Azure Portal

#Introduction

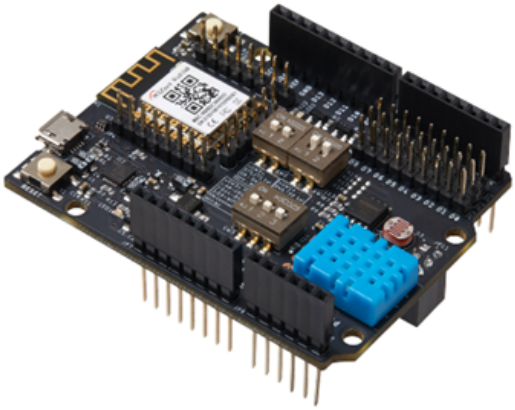
Microsoft Azure is cloud computing service. Using WizFi360 we can connect to Azure services, transmit data and monitor status.

In this document we will guide how to connect Arduino Mega 2560 + WizFi360 to MS Azure Services. Thic process consists of following steps:

- Azure IoT Hub preparation
- IoT device registration
- Connect to Azure IoT and transmit data

For Azure IoT Hub setup and IoT device creation please refer to [Azure Cloud Introduction].

For this guide WizFi360-EVB-Shield Evaluation board was used.

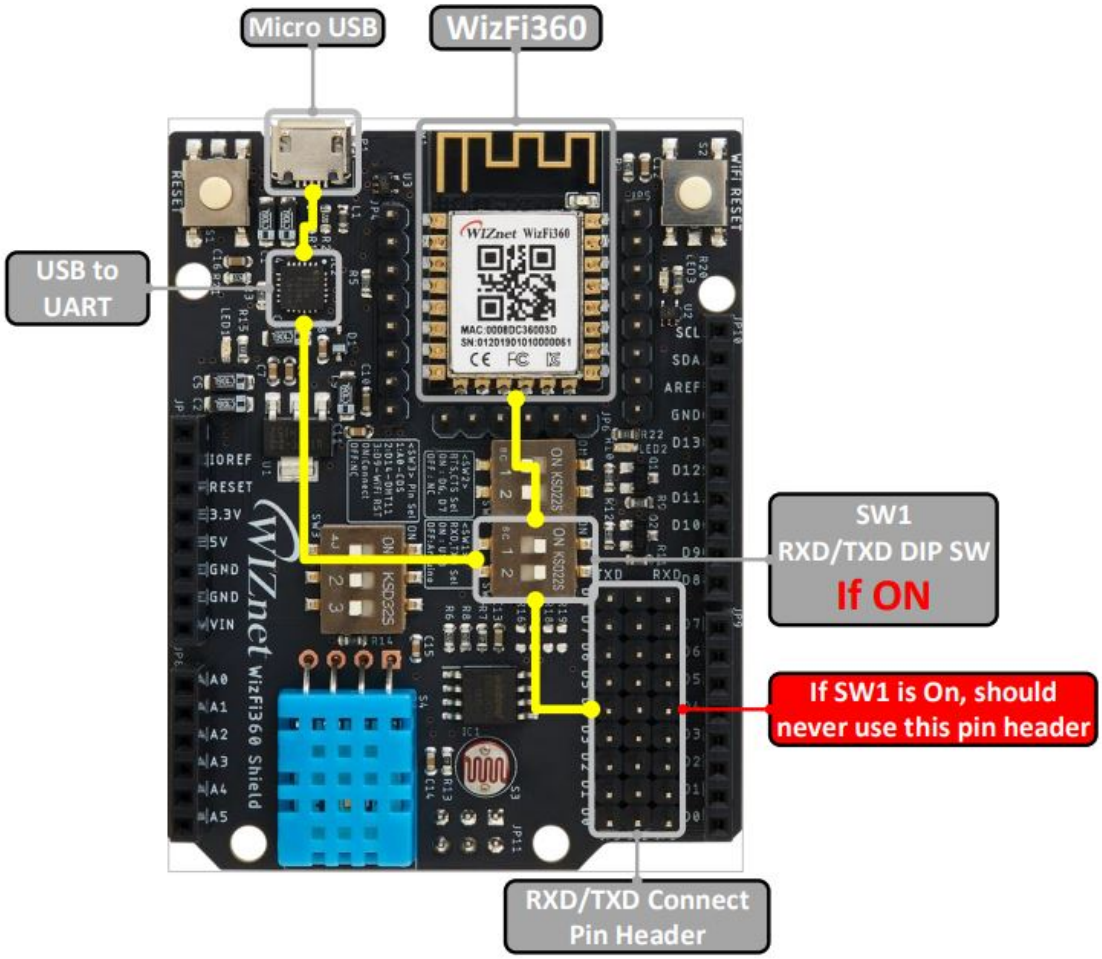


#Device preparation

#Hardware configuration

WizFi360-EVB-Shield will be used in **Standalone mode**. Therefore DIP Switch and jumper cables shall be connected as following:

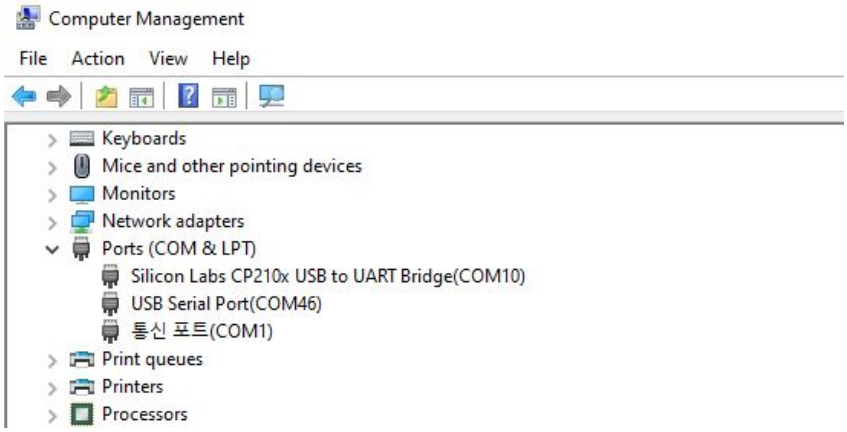
- SW1 : On
- SW2 : Off
- SW3 : Off



#Device connection

After connecting hardware, connect WizFi360-EVB-Shield to Desktop or Laptop using USB Cable.

Check **COM Port** from **Device Manager**.



If COM port cannot be found in Device manager, check drivers from link below.

- Silicon Labs CP210x USB to UART Driver

#AT Commands

#1. Set current WiFi mode (not saved in flash)

AT Command: AT+CWMODE_CUR Syntax:

Type	Command	Response
Query	AT+CWMODE_CUR?	+CWMODE:<mode> OK
Set	AT+CWMODE_CUR=<mode>	OK

Defined values:

Mode	Value
1	Station mode
2	SoftAP mode (factory default)
3	Station+SoftAP mode

#2. Enable DHCP

AT Command: AT+CWDHCP_CUR Syntax:

Type	Command	Response
Query	AT+CWDHCP_CUR?	+CWDHCP_CUR:<para> OK
Set	AT+CWMODE_CUR=<para>,<en>	OK

Defined values:

Parameter	Value
0	DIsable SoftAP DHCP & Station DHCP.
1	Enable SoftAP DHCP & Disable Station DHCP.
2	Disable SoftAP DHCP & enable Station DHCP.
3	Enable SoftAP DHCP & Station DHCP. (factory default)

#3. List available APs

AT Command: AT+CWLAP Syntax:

Type	Command	Response
Query	AT+CWLAP	+CWLAP:([<ecn>,<ssid>,<rssi>,<mac>,<channel>,<wps>])

Defined values:

Parameter	Value
<ecn>	0: Open 1: WEP 2: WPA_PSK 3: WPA2_PSK 4:WPA_WPA2_PSK
<ssid>	string parameter. AP ssid
<rssi>	signal strength
<mac>	string parameter. AP mac
<wps>	0: Disable WPS 1: Enable WPS

#4. Connect to AP

AT Command: AT+CWJAP_CUR Syntax:

Type	Command	Response
Set	AT+CWJAP_CUR=<ssid>,<pwd>,[<bssid>]	+CWJAP_CUR:<ssid>,<bssid>,<channel>,<rssi> OK

Defined values:

Parameter	Value
<ssid>	string parameter. Target AP의 ssid. MAX: 32 bytes
<pwd>	string parameter. Target AP의 password. MAX: 64-byte ASCII
<bssid>	string parameter, target AP' MAC address, used in case if there are several APs with same SSID.

#5. Azure IoT Hub configuration set

AT Command: AT+AZSET

Syntax:

Type	Command	Response
Set	AT+AZSET=<iothub_name>,<device_id>,<device_key>	OK

Defined values:

Parameter	Value

Parameter	Value
<hub ID>	string parameter. IoT Hub ID
<device ID>	string parameter. IoT Device ID
<key>	string parameter, IoT Device Key

#6. Set MQTT Topic

AT Command: AT+MQTTTOPIC

Syntax:

Type	Command	Response
Set	AT+MQTTTOPIC=<publish topic>,<subscribe topic>,<subscribe topic2>,<subscribe topic3>	OK

Defined values:

Parameter	Value
<publish topic>	string parameter, topic where WizFi360 will publish
<subscribe topic>	string parameter, topic where WizFi360 will subscribe
<subscribe topic2>	string parameter, topic where WizFi360 will subscribe
<subscribe topic3>	string parameter, topic where WizFi360 will subscribe

Note:

- This command shall be set before connecting to broker.
- <subscribe topic2> & <subscribe topic3> are available in Firmware v1.0.5.0 and later.

#7. Connect to Azure

AT Command: AT+AZCON

Syntax:

Type	Command	Response
Set	AT+AZCON	CONNECT OK

Note: • Before sending this ocmmand, AT+AZSET command & AT+MQTTTOPIC command shall be set. • After connection data can be sent to Azure server using AT+MQTTPUB command. • For more details please refer to https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support.

#8. Publish a message

AT Command: AT+MQTTPUB

Syntax:

Type	Command	Response
Set	AT+MQTTPUB=<message>	OK

Note: • This command shall be used when MQTT connection is established. • Topic where data will be published is set with AT+MQTTTOPIC command, user shall define topic before connecting to broker.

#Running sample code

#Connect in serial terminal & code execution

For connection use following configuration in terminal: 115200-8-N-1, None.

#WiFi setting

#1. Set Wifi station mode

Command	Response
AT+CWMODE_CUR=1 // station mode	OK

#2. Set DHCP Enable

Command	Response
AT+CWDHCP_CUR=1,1 // DHCP enable on Station mode	OK

#3. Get possible Wi-Fi AP List for connection

Command	Response
AT+CWLAP	+CWLAP : (3,"ssid",-57,"mac address",1,1) // encryption method, ssid, rssi, mac address, channel, wps

#4. Connect to Wi-Fi AP

Command	Response
AT+CWJAP_CUR="ssid","password"	WIFI CONNECTED WIFI GOT IP

#5. Query WizFi360 device' IP address

Command	Response
AT+CIPSTA_CUR?	+CIPSTA_CUR:ip:”192.168.10.13” +CIPSTA_CUR:gateway:”192.168.10.1” +CIPSTA_CUR:network:”255.255.255.0”

#Connect to Azure services using next AT commands

#1. Set Azure connection

Command	Response
AT+AZSET="iothub_name","device_id","device_key"	OK

#2. Set MQTT Topic

Command	Response
AT+MQTTTOPIC="/devices/{device_id}/messages/events/","/devices/{device_id}/messages/devicebound/#"	OK
Example AT+MQTTTOPIC="/devices/testDevice/messages/events/","/devices/testDevice/messages/devicebound/#"	

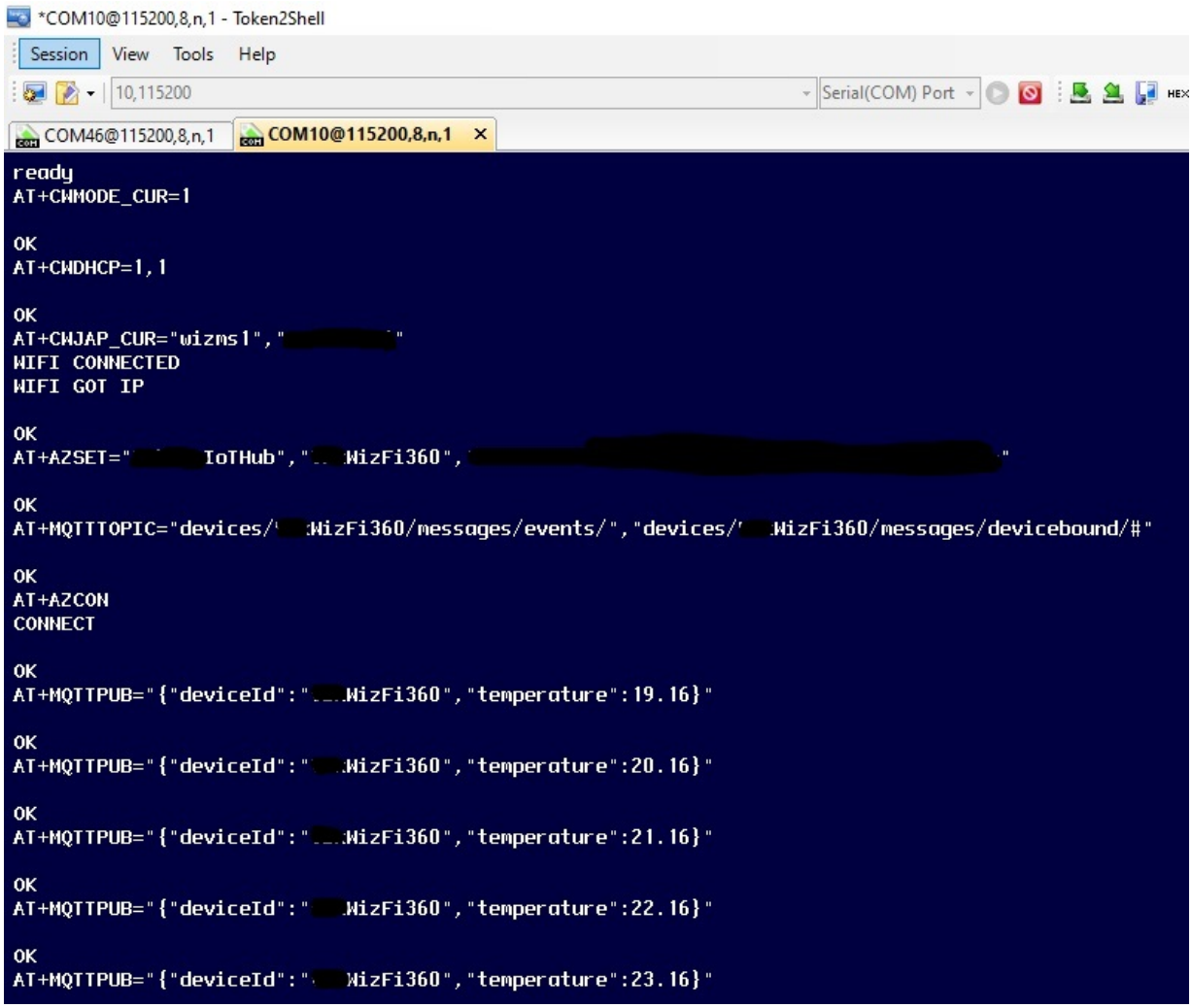
> Note: MQTT Topic follows the rules defined in Azure IoT Hub. Refer to document: Communicate with your IoT hub using the MQTT protocol: Using the MQTT protocol directly (as a device)

#3. Connect to Azure

Command	Response
AT+AZCON	OK

#4. Publish data

Command	Response
AT+MQTTPUB="{"deviceId":"WizFi360","temperature":28.16,"humidity":46.04}"	OK



#Results

1. Press "Start" button in Telemetry Section in IoT Hub Explorer.

   "Start" button shall be pressed before sending data using MQTTPUB commands.

2. Check data sent with MQTTPUB command.

Azure IoT explorer

File Edit View Window Help

Azure IoT explorer

Hub[redacted]WizIoTHub > Devices > VikWizFi360 > Telemetry

DEVICE

- Device identity
- Device twin
- Telemetry
- Direct method
- Cloud-to-device message
- Module identity

Start | Clear events | Show system properties

Telemetry

Consumer Group $Default

4:20:36 PM, February 03, 2020:


```
{
  "body": {
    "deviceId": "[redacted]WizFi360",
    "temperature": 23.16
  },
  "enqueuedTime": "2020-02-03T07:20:36.007Z",
  "properties": {}
}
```


4:20:32 PM, February 03, 2020:


```
{
  "body": {
    "deviceId": "[redacted]WizFi360",
    "temperature": 22.16
  },
  "enqueuedTime": "2020-02-03T07:20:32.913Z",
  "properties": {}
}
```


4:20:30 PM, February 03, 2020:


```
{
  "body": {
    "deviceId": "[redacted]WizFi360",
    "temperature": 21.16
  },
  "enqueuedTime": "2020-02-03T07:20:30.630Z",
  "properties": {}
```

WizFi360 MQTT AT Command를 이용한 Azure IoT Hub 연동 예제

#시작하기 전에

> **Azure Portal**에 Login을 합니다. 계정이 없는 경우, 계정 생성 후에 Login을 진행합니다.
>
> ※ 본 문서는 **체험 계정**으로 진행합니다.
>
> Azure Portal을 사용하여 IoT Hub 만들기 등 앞선 일련의 과정에 대하여 Azure Cloud 서비스 소개를 참조하시기 바랍니다.

- [MS] Azure Portal을 사용하여 IoT Hub 만들기
- Azure Portal을 사용하여 Blob Storage 만들기
- Azure Portal을 사용하여 Stream Analytics 만들기
- Azure Portal을 사용하여 Stream Analytics 작업 입 · 출력 구성 및 변환 Query 정의
- WizFi360 MQTT AT Command를 이용하여 Azure IoT Hub에 연동

#소개

Microsoft Azure Service에 **WizFi360**을 **연동**하여, Data를 Cloud로 전송하고, Monitoring을 할 수 있습니다.

Data 통신은 다음과 같은 구조로 이루어집니다.

MQTT AT Command를 이용하여, IoT Hub Service 연결 및 Data 송신을 합니다.

IoT Hub로 송신이 된 Data는 Stream Analytics를 통하여 Data 저장소 Blob Storage로 저장이 됩니다.

본 문서는 Mbed 기반 WizFi360 MQTT AT Command 이용한 Microsoft Azure Service 연동 예제에 대하여 Guide를 제공합니다.

#Step 1: 필수 구성 요소

본 문서를 따라하기 전에 다음 항목이 준비되어야 합니다.

#Hadrware

- Desktop or Laptop Computer
- WizFi360-EVB-Shield
- NUCLEO-L476RG
- Mini USB Cable

#Software

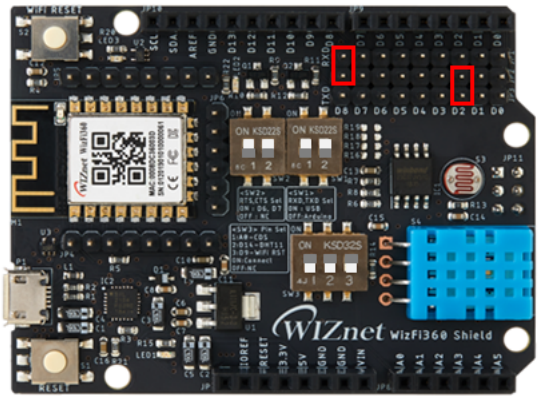
- Mbed Studio
- Preferred Serial Terminal (TeraTerm, Hercules, etc . . .)

#Step 2: Device 준비

#1. Hardware 준비

WizFi360-EVB-Shield는 NUCLEO-L476RG와 결합을 하여 사용되어 집니다. 따라서 WizFi360-EVB-Shield의 DIP Switch 및 UART Jumper Pin을 다음과 같이 설정이 필요합니다.

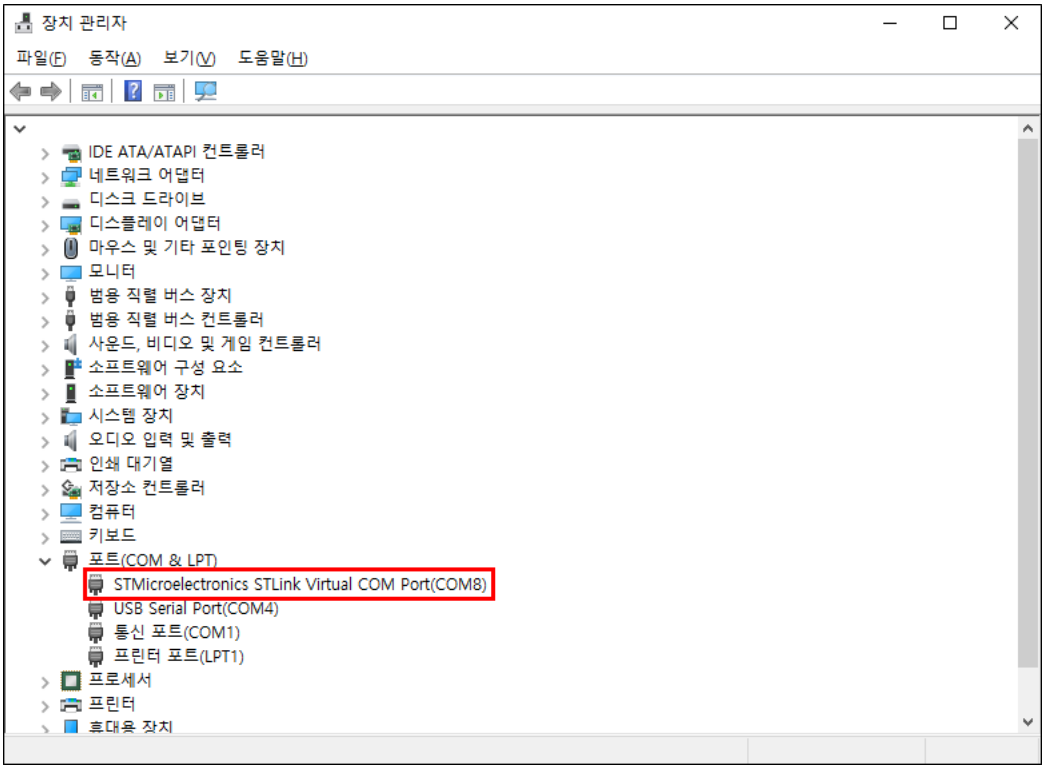
- SW1 : Off
- SW2 : Off
- SW3 : On
- D2 : UART Tx
- D8 : UART Rx



#2. Device 연결

Hardware 설정 후, Mini USB Cable을 이용하여 NUCLEO-L476RG를 Desktop 혹은 Laptop Computer와 연결을 합니다.

장치 관리자에서 NUCLEO-L476RG와 연결된 **COM Port**를 확인 할 수 있습니다.



장치 관리자에서 COM Port를 확인 할 수 없는 경우, 다음 Link에서 Driver를 Downlonad하여 설치하시기 바랍니다.

- ST-LINK, ST-LINK/V2, ST-LINK/V2-1 USB driver

#Step 3: 동작 예제

#1. 예제 Download 및 실행

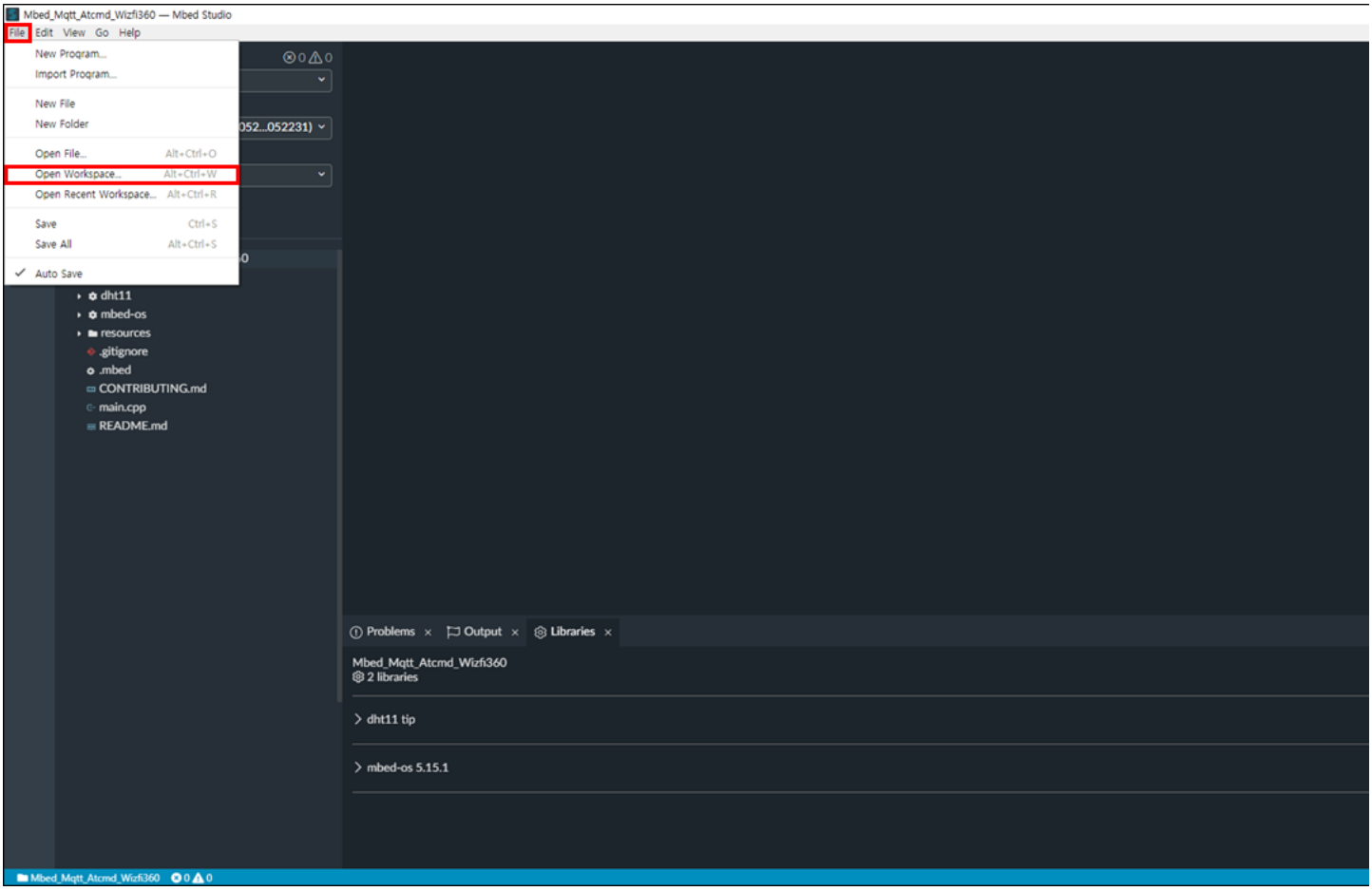
예제 Download를 한 후, **File** > **Open Workspace**을 선택하여 **Project 실행**합니다.

예제는 다음 경로에 위치하고 있는 Project를 참고 바랍니다.

- **samples/Wi-Fi/Mbed_Mqtt_Atcmd_Wizfi360**
- **samples/Wi-Fi/Mbed_Mqtt_Atcmd_Wizfi360_Azure_C_Shared_Utility**

Mbed_Mqtt_Atcmd_Wizfi360 경우에는 Device Explorer 혹은 Azure IoT Explorer와 같은 Tool을 이용하여 **별도의 SAS Token 생성 과정이 필요**합니다.

Mbed_Mqtt_Atcmd_Wizfi360_Azure_C_Shared_Utility 경우에는 Azure IoT Common Library인 azure_c_shared_utility Library를 Porting을 한 Project이므로 **별도의 SAS Token 생성 과정이 필요 없으며, Project 내부에서 SAS Token 생성 과정을 처리**합니다.

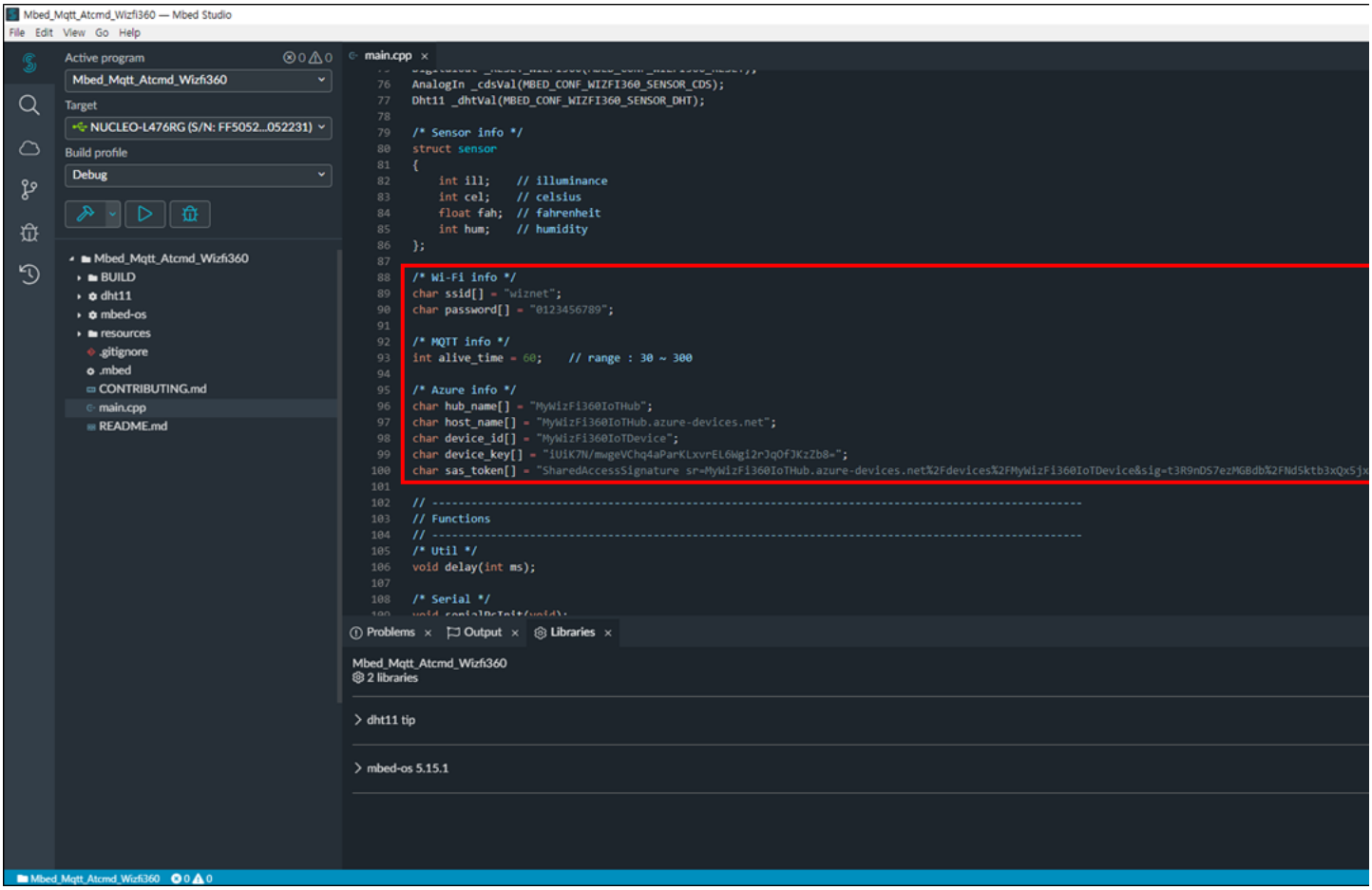


#2. Parameter 값 수정

Azure IoT Hub에 연결을 하기 위하여, 다음의 Parameter를 수정합니다.

Copy

```
/* Wi-Fi info */
char ssid[] = "xxxxxxxxxx";
char password[] = "xxxxxxxxxx";

/* MQTT info */
int alive_time = xx;     // range : 30 ~ 300

/* Azure info */
char hub_name[] = "xxxxxxxxxx";
char host_name[] = "xxxxxxxxxx";
char device_id[] = "xxxxxxxxxx";
char device_key[] = "xxxxxxxxxx";
char sas_token[] = "xxxxxxxxxx";
```



SAS Token 생성은 다음을 참고 바랍니다.

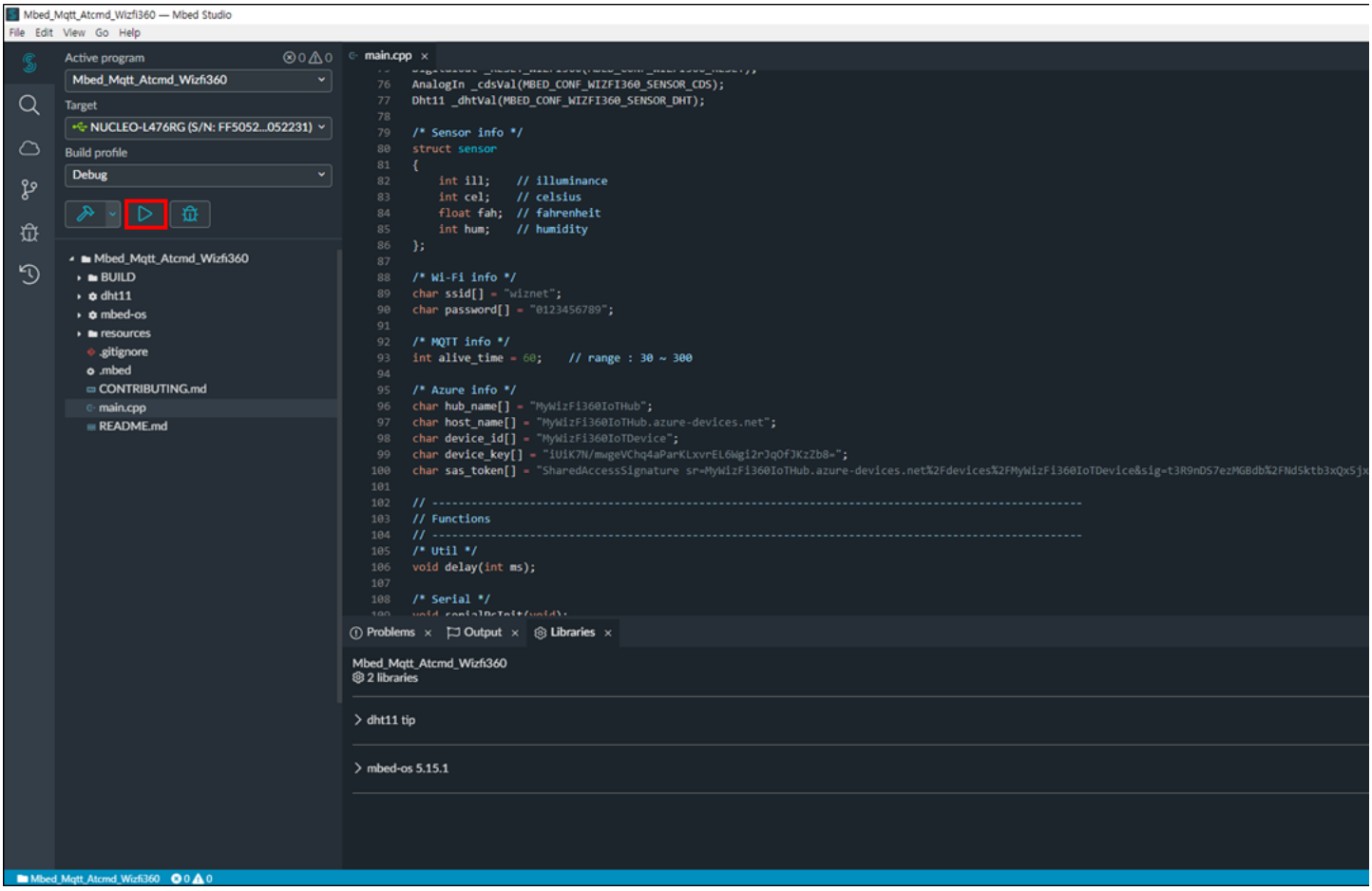
- Device Explorer를 사용하여 SAS Token 생성하기
- Azure IoT Explorer를 사용하여 SAS Token 생성하기

#3. Project Build 및 Run

Run Program을 눌러 Project Build 및 Run을 합니다.

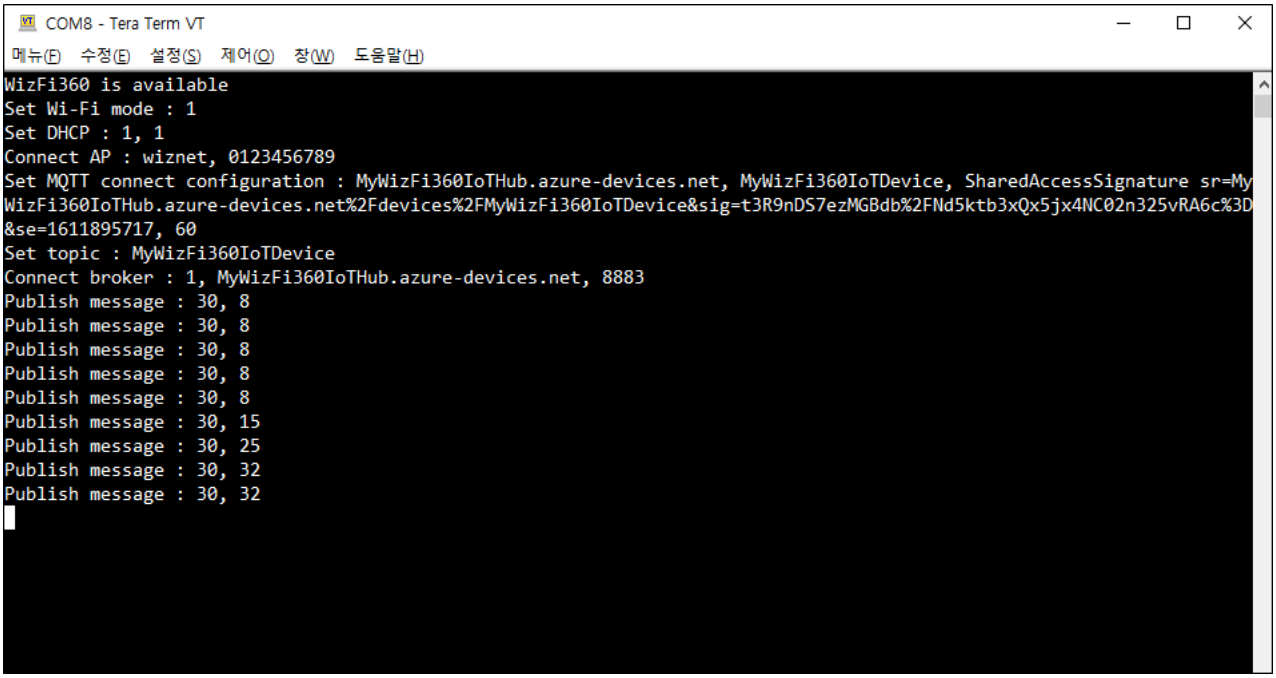
Note :

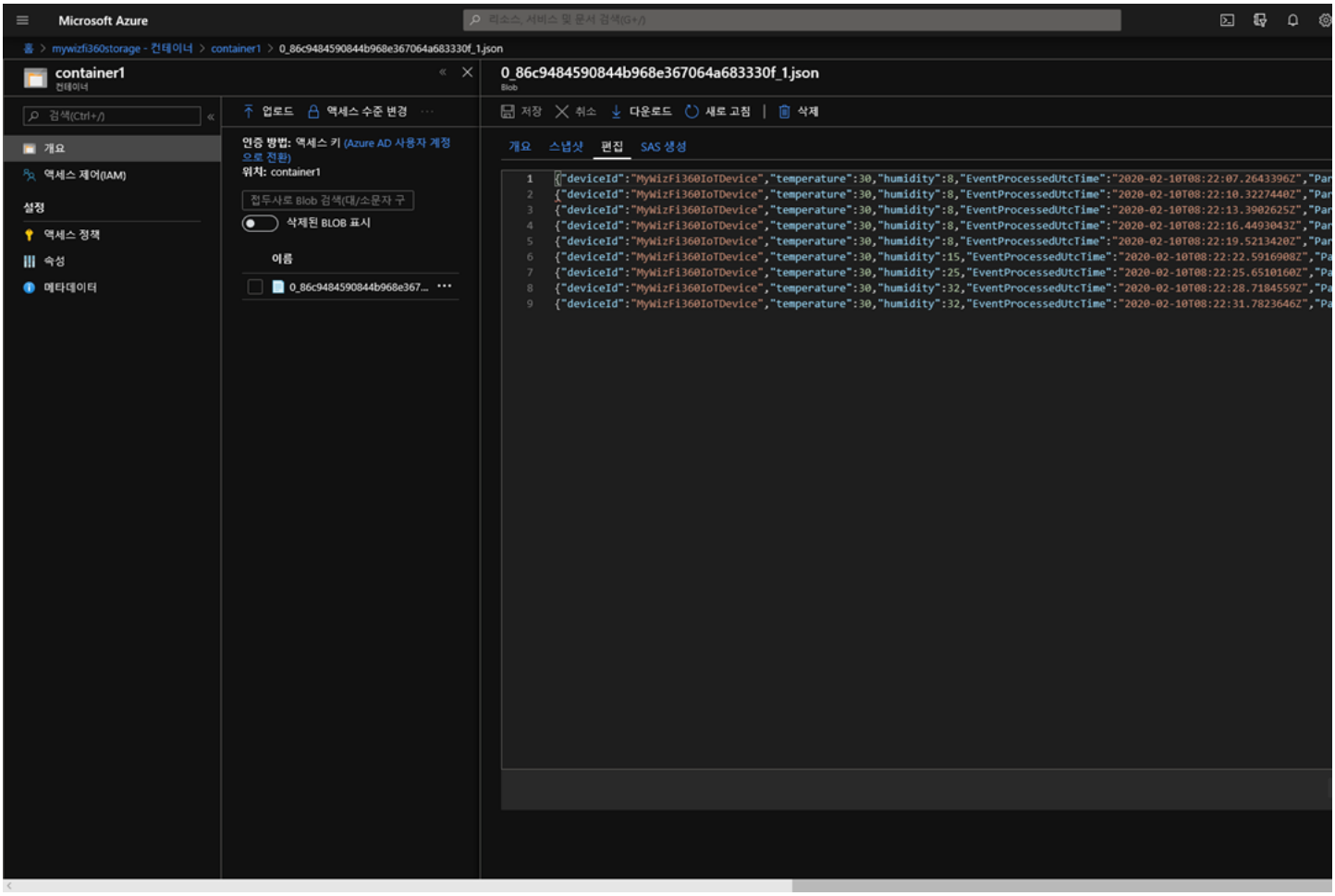
Stream Analytics 실행 중이어야 Blob Storage로 Data가 전달됩니다.



#Step 4: 동작 예제 결과

Terminal Program으로 WizFi360에서 Azure IoT Hub로 전송하는 Data, Blob Storage에서 WizFi360으로부터 수신한 Data를 확인 할 수 있습니다.





#더 보기

- WizFi360 MQTT AT Command를 이용하여 Azure IoT Hub에 연동

Connect to Azure IoT Hub using WizFi360 MQTT AT Command

#Getting started

Login to **Azure Portal**.

※ In this guide we will procedd with **free account**. To learn how to create IoT Hub please refer to Azure Cloud Service Introduction.

- [MS] Create IoT Hub using Azure Portal
- Create Blob storage using Azure Portal
- Create Stream Analytics using Azure Portal
- Setup Queries in Stream Analytics using Azure Portal
- Connect to Azure IoT Hub using WizFi360 MQTT AT Command

#Introduction

It is possible to connect to **Microsoft Azure Service** using **WizFi360**, send data to cloud and monitor current status.

Data communication will be established as following.

MQTT AT Commands will be used to connect to IoT Hub Service & send data.

Data sent to IoT Hub will be saved in Blob Storage through Stream Analytics.

In this guide we will use WizFi360 with Mbed to connect to Microsoft Azure Services using MQTT AT Command.

#Step 1: Required items

Items below are required for this guide.

#Hadrware

- Desktop or Laptop Computer
- WizFi360-EVB-Shield
- NUCLEO-L476RG
- Mini USB Cable

#Software

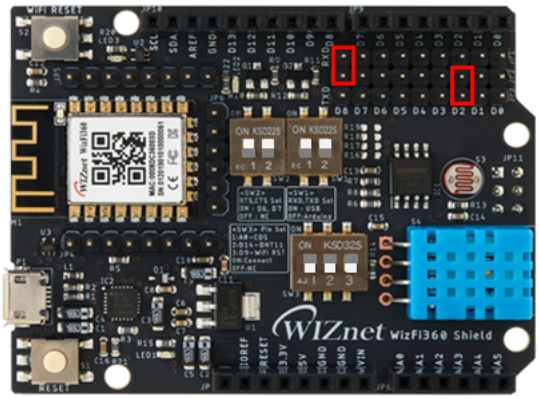
- Mbed Studio
- Preferred Serial Terminal (TeraTerm, Hercules, etc . . .)

#Step 2: Device preparation

#1. Hardware preparation

WizFi360-EVB-Shield will be installed on top of NUCLEO-L476RG. Therefore DIP Switch and jumper cables shall be connected as following:

- SW1 : Off
- SW2 : Off
- SW3 : On
- D2 : UART Tx
- D8 : UART Rx



#2. Device connection

After connecting hardware, connect NUCLEO-L476RG to Desktop or Laptop using USB Cable.

Check **COM Port** from **Device Manager**.

If COM port cannot be found in Device manager, check link below and follow instructions.

- ST-LINK, ST-LINK/V2, ST-LINK/V2-1 USB driver

#Step 3: Sample application

#1. Code Download & Execution

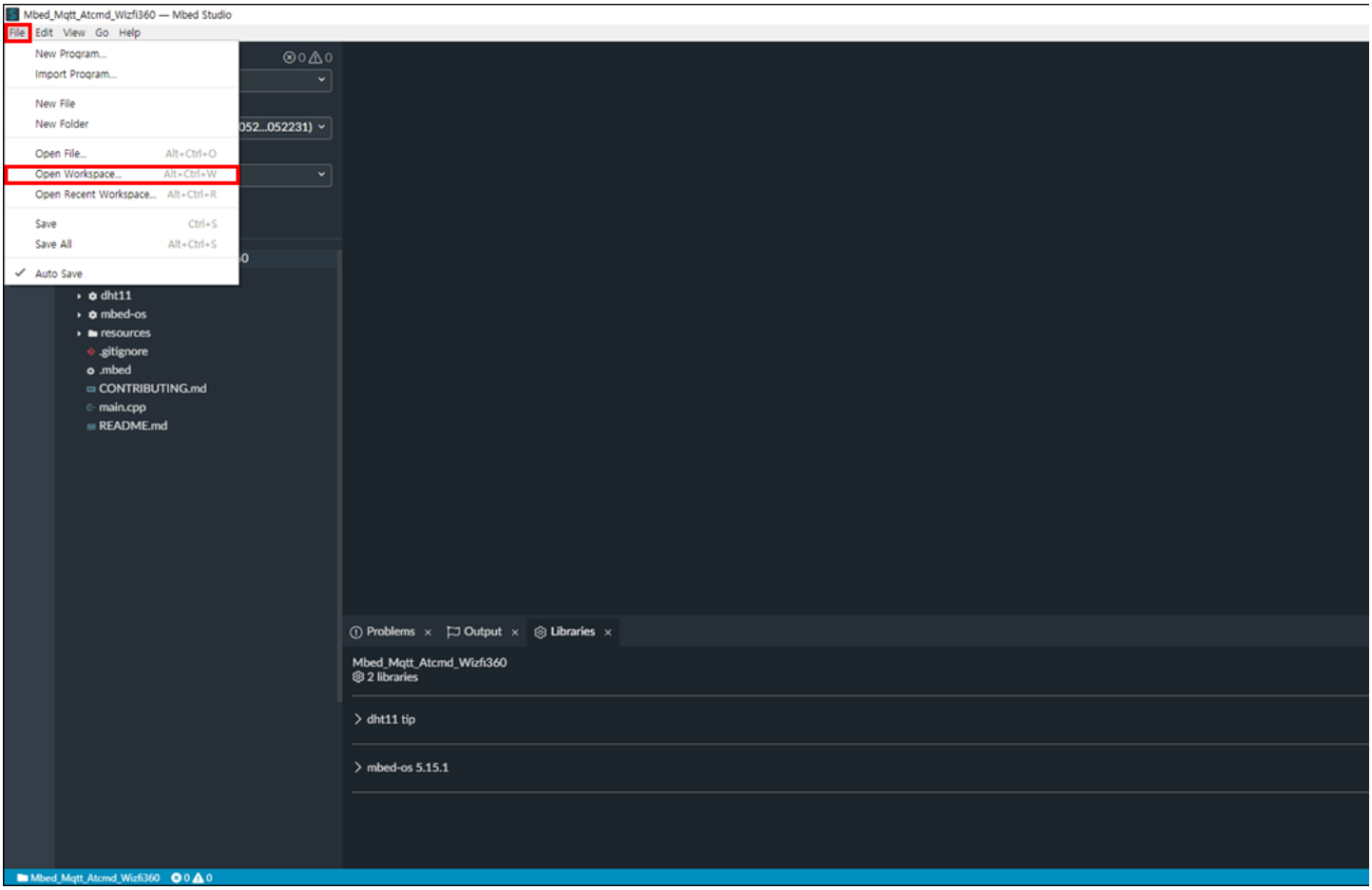
After code download, open project by selecting **File** > **Open Workspace**.

Sample code is stored in following path.

- **samples/Wi-Fi/Mbed_Mqtt_Atcmd_Wizfi360**
- **samples/Wi-Fi/Mbed_Mqtt_Atcmd_Wizfi360_Azure_C_Shared_Utility**

In case of **Mbed_Mqtt_Atcmd_Wizfi360** SAS Token was created using Device Explorer or Azure IoT Explorer.

In case of **Mbed_Mqtt_Atcmd_Wizfi360_Azure_C_Shared_Utility** there is no need to create SAS Token separately as azure_c_shared_utility Library was ported into Project.

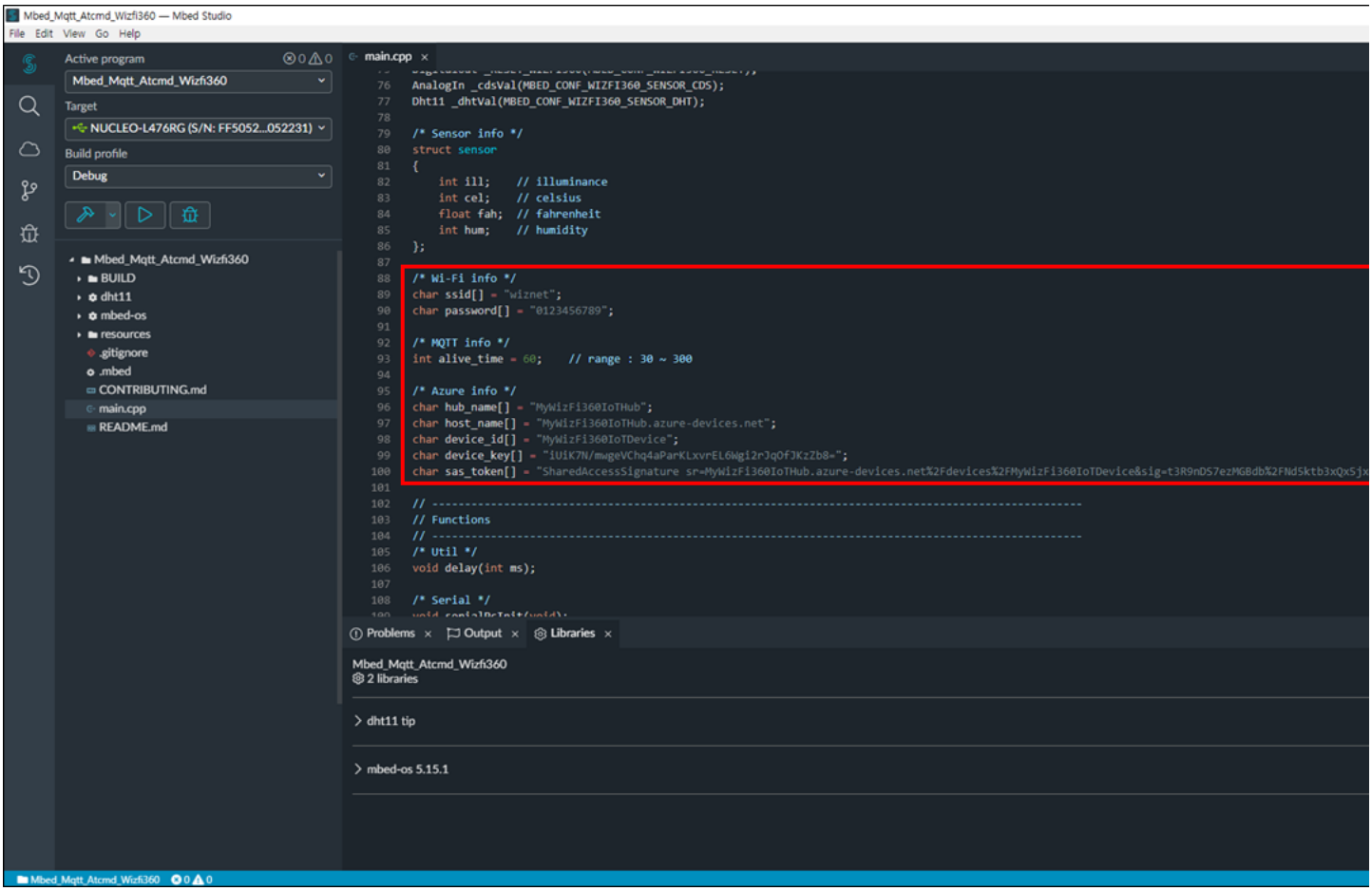


#2. Parameter update

To connect to Azure IoT Hub, update next parameters.

Copy

```
/* Wi-Fi info */
char ssid[] = "xxxxxxxxxx";
char password[] = "xxxxxxxxxx";

/* MQTT info */
int alive_time = xx;     // range : 30 ~ 300

/* Azure info */
char hub_name[] = "xxxxxxxxxx";
char host_name[] = "xxxxxxxxxx";
char device_id[] = "xxxxxxxxxx";
char device_key[] = "xxxxxxxxxx";
char sas_token[] = "xxxxxxxxxx";
```



For **SAS Token creation** refer to below.

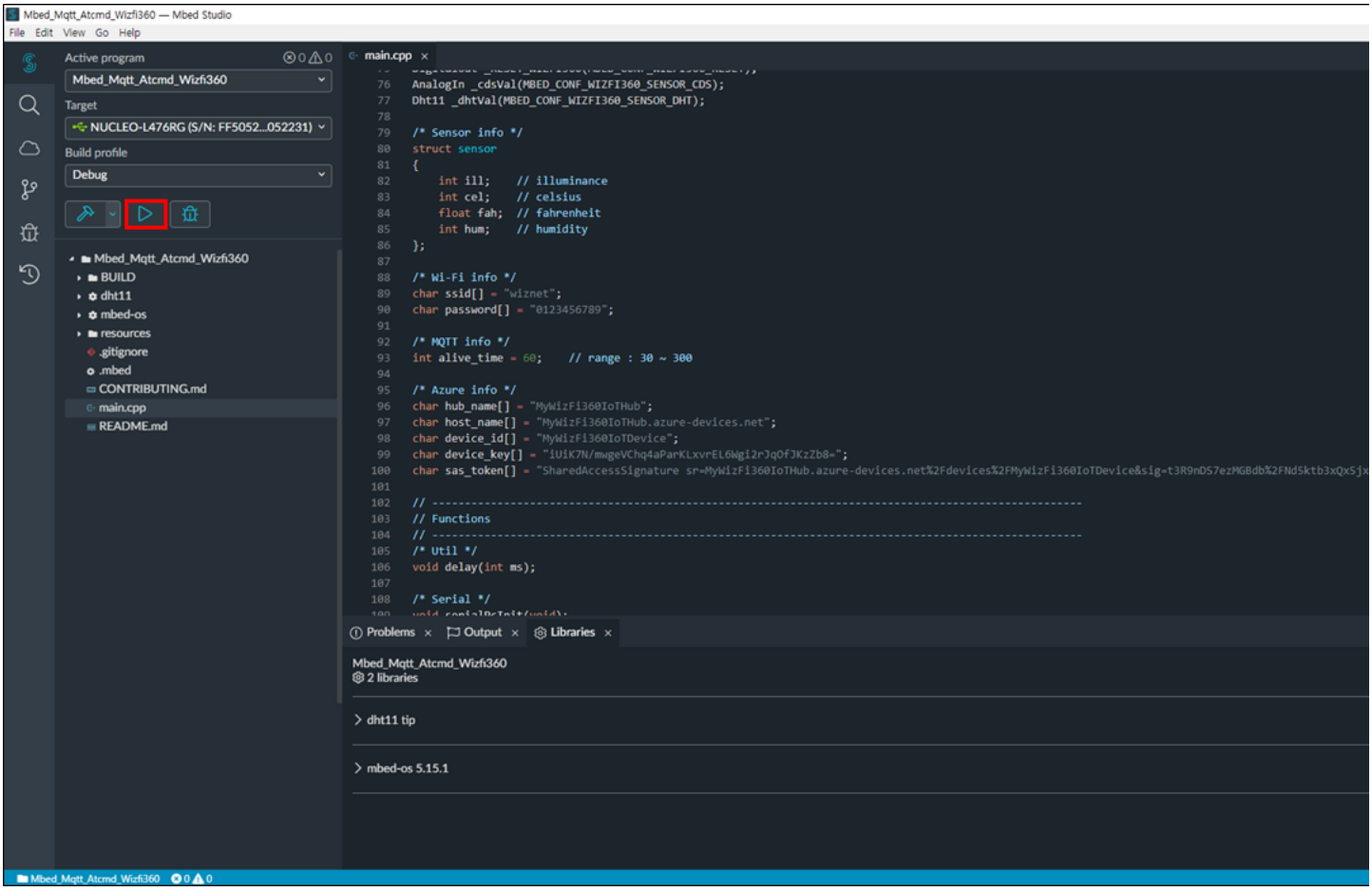
- Create SAS Token using Device Explorer
- Create SAS Token using Azure IoT Explore

#3. Project Build & Run

Press **Run Program** to Project Build & Run.

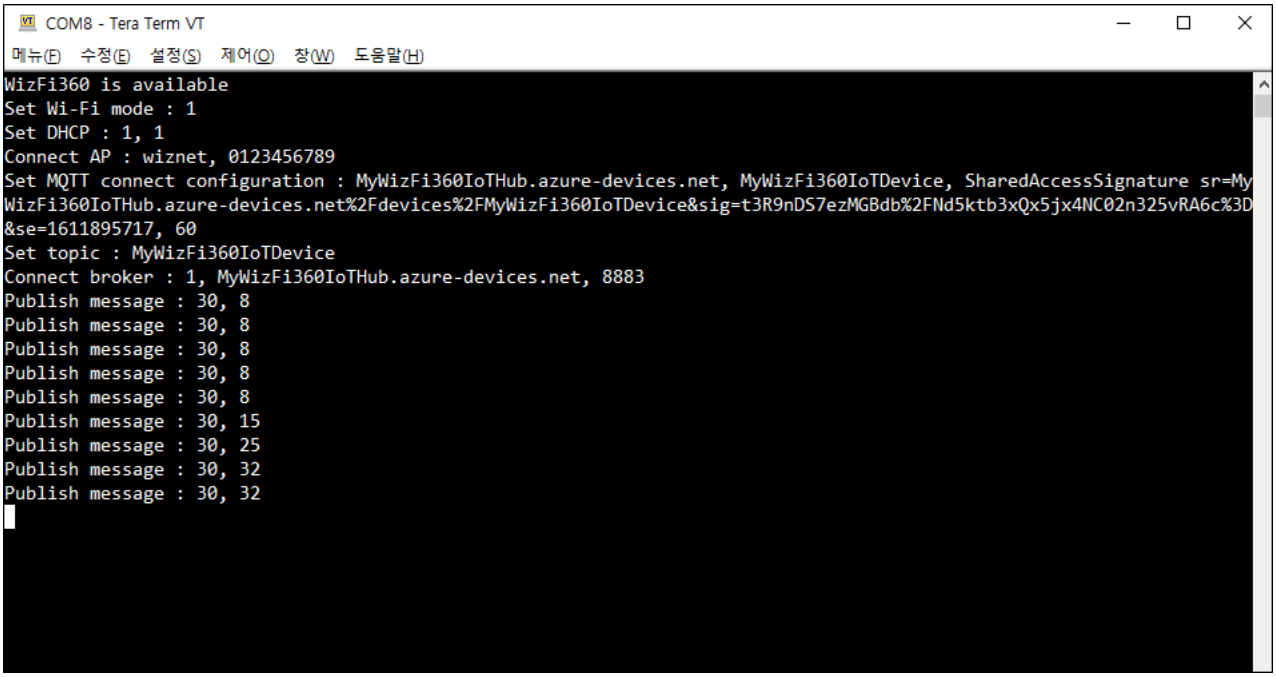
Note :

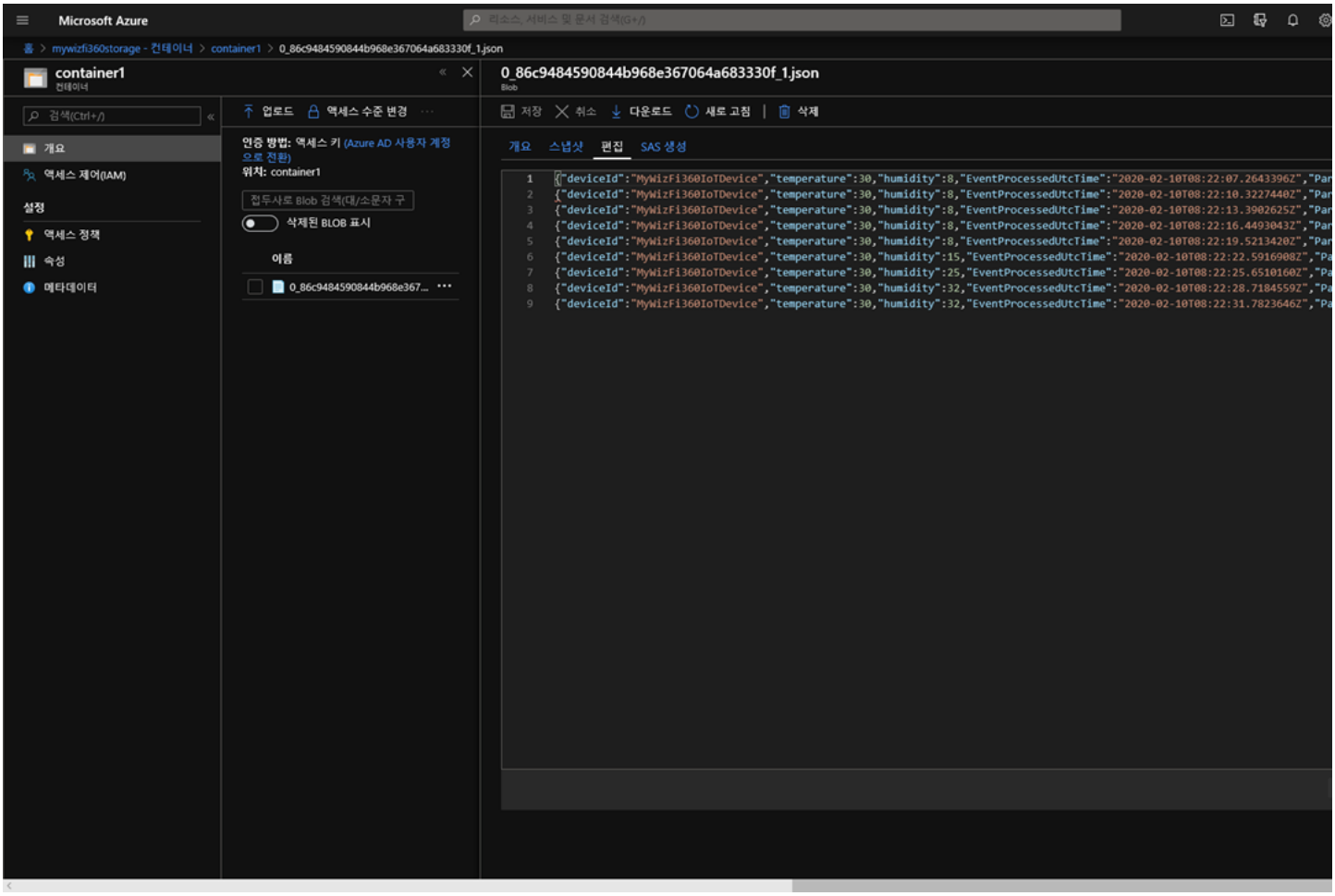
Start Stream Analytics in order to forward data to Blob Storage.



#Step 4: Results

In terminal program we can check data sent from WizFi360 to Azure IoT Hub. In Blob Storage we can check received data.





#References

- Connect to Azure IoT Hub using WizFi360 MQTT AT Command

NUCLEO-L476RG + WizFi360 Azure AT Command를 이용하여 Azure IoT Hub에 연결

#시작하기 전에

#Hardware Requirement

- NUCLEO-L476RG
- Desktop or laptop computer
- USB 케이블
- WizFi360-EVB-Shield

#Software Requirement

- Microsoft Azure Account (Azure 구독이 아직 없는 경우 체험 무료계정을 만듭니다.)
- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Azure IoT Explorer
- MBED Studio / MBED Online Compiler

#소개

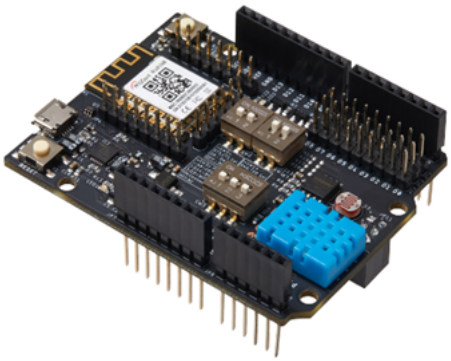
Microsoft Azure 는 Microsoft 의 클라우드 컴퓨팅 서비스입니다. Microsoft Azure 의 서비스에 WizFi360 을 연동하여 데이터를 클라우드로 전송하고, 모니터링 할 수 있습니다.

본 문서에서는 NUCLEO-L476G + WizFi360 이용하여 MS Azure Services에 연결 방법에 대한 가이드를 제공합니다. 이 프로세스는 다음 단계로 구성됩니다.

- Azure IoT Hub 준비
- IoT 디바이스 등록
- Azure IoT와 연결 및 데이터 통신

Azure IoT Hub 준비와 IoT 디바이스 등록 과정 대해 Azure Cloud 소개 참조하시기 바랍니다.

WiFi모듈 테스트를 위해 WizFi360-EVB-Shield Evaluation 보드를 사용되었습니다.

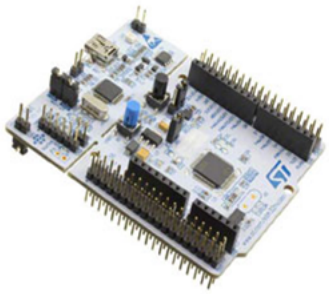


#디바이스 준비

#하드웨어 설정

WizFi360-EVB-Shield는 NUCLEO-L476RG와 결합을 하여 사용되어 집니다. 따라서 WizFi360-EVB-Shield의 DIP Switch 및 UART Jumper Pin을 다음과 같이 설정이 필요합니다.

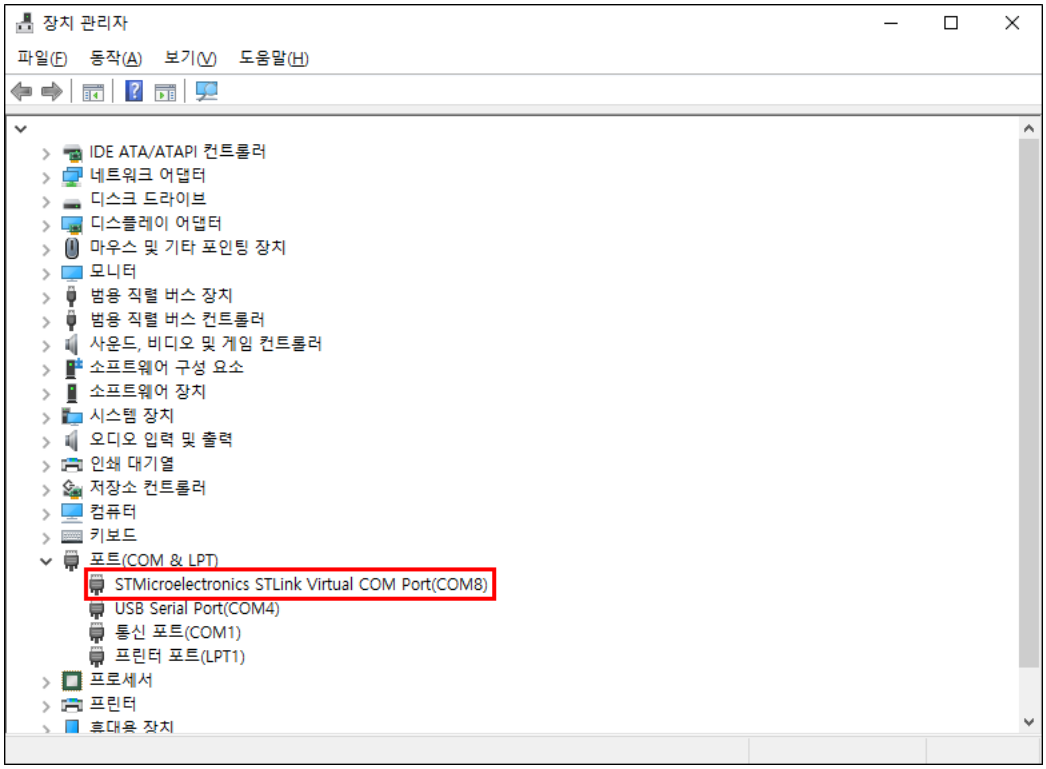
- SW1 : Off
- SW2 : Off
- SW3 : On
- D2 : UART Tx
- D8 : UART Rx



#디바이스 연결

Hardware 설정 후, Mini USB Cable을 이용하여 NUCLEO-L476RG를 Desktop 혹은 Laptop Computer와 연결을 합니다.

장치 관리자에서 NUCLEO-L476RG와 연결된 **COM Port**를 확인 할 수 있습니다.



장치 관리자에서 COM Port를 확인 할 수 없는 경우, 다음 Link에서 Driver를 Downlonad하여 설치하시기 바랍니다.

- [ST-LINK, ST-LINK/V2, ST-LINK/V2-1 USB driver][Link-St_Link_St_Link_V2_St_Link_V2_1_Usb_Driver]

#AT 명령어

#1. Set current WiFi mode (not saved in flash)

AT Command: AT+CWMODE_CUR Syntax:

Type	Command	Response
Query	AT+CWMODE_CUR?	+CWMODE:<mode> OK
Set	AT+CWMODE_CUR=<mode>	OK

Defined values:

Mode	Value
1	Station mode
2	SoftAP mode (factory default)
3	Station+SoftAP mode

#2. Enable DHCP

AT Command: AT+CWDHCP_CUR Syntax:

Type	Command	Response
Query	AT+CWDHCP_CUR?	+CWDHCP_CUR:<para> OK
Set	AT+CWMODE_CUR=<para>,<en>	OK

Defined values:

Parameter	Value
0	SoftAP DHCP 와 Station DHCP 를 disable 한다.
1	SoftAP DHCP 는 enable 하고 Station DHCP 는 disable 한다.
2	2: SoftAP DHCP 는 disable 하고 Station DHCP 는 enable 한다.
3	SoftAP DHCP 와 Station DHCP 를 enable 한다. (factory default)

#3. List available APs

AT Command: AT+CWLAP Syntax:

Type	Command	Response
Query	AT+CWLAP	+CWLAP:([<ecn>,<ssid>,<rssi>,<mac>,<channel>,<wps>])

Defined values:

Parameter	Value
<ecn>	0: Open 1: WEP 2: WPA_PSK 3: WPA2_PSK 4:WPA_WPA2_PSK
<ssid>	string parameter. AP의 ssid
<rssi>	signal strength
<mac>	string parameter. AP의 mac
<wps>	0: WPS는 disable된다 1: WPS는 enable된다

#4. Connect to AP

AT Command: AT+CWJAP_CUR Syntax:

Type	Command	Response
Set	AT+CWJAP_CUR=<ssid>,<pwd>,[<bssid>]	+CWJAP_CUR:<ssid>,<bssid>,<channel>,<rssi> OK

Defined values:

Parameter	Value
<ssid>	string parameter. Target AP의 ssid. MAX: 32 bytes
<pwd>	string parameter. Target AP의 password. MAX: 64-byte ASCII
<bssid>	string parameter, target AP 의 MAC address, 같은 SSID 를 가진 여러 개의 AP 들이 있을 때 사용된다.

#5. Azure IoT Hub configuration set

AT Command: AT+AZSET

Syntax:

Type	Command	Response
Set	AT+AZSET=<iothub_name>,<device_id>,<device_key>	OK

Defined values:

Parameter	Value
<hub ID>	string parameter. IoT Hub의 ID
<device ID>	string parameter. IoT Device의 ID
<key>	string parameter, IoT Device의 Key

#6. Set MQTT Topic

AT Command: AT+MQTTTOPIC

Syntax:

Type	Command	Response
Set	AT+MQTTTOPIC=<publish topic>,<subscribe topic>,<subscribe topic2>,<subscribe topic3>	OK

Defined values:

Parameter	Value
<publish topic>	string parameter, WizFi360 이 publish 하는 topic
<subscribe topic>	string parameter, WizFi360 이 subscribe 하는 topic
<subscribe topic2>	string parameter, WizFi360 이 subscribe 하는 topic
<subscribe topic3>	string parameter, WizFi360 이 subscribe 하는 topic

Note:

- 이 command 는 broker 에 연결하기전에 설정되어야 합니다.
- <subscribe topic2> 와 <subscribe topic3>는 Firmware v1.0.5.0 이후 version 부터 사용가능 합니다.

#7. Connect to Azure

AT Command: AT+AZCON

Syntax:

Type	Command	Response
Set	AT+AZCON	CONNECT OK

Note: • 이 command 를 전송하기전에 AT+AZSET command 와 AT+MQTTTOPIC command 를 설정합니다. • Connect 이후 AT+MQTTPUB command 를 통해 Azure Sever 에 데이터를 전송합니다. • 자세한 내용은 https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support 를 참조하세요.

#8. Publish a message

AT Command: AT+MQTTPUB

Syntax:

Type	Command	Response
Set	AT+MQTTPUB=<message>	OK

Note: • 이 command 는 MQTT 가 연결되어 있을 때 사용됩니다. • Publish 한 data 의 topic 은 AT+MQTTTOPIC command 에 의해 결정되며, 사용자는 broker 에 연결하기전에 topic 을 설정합니다.

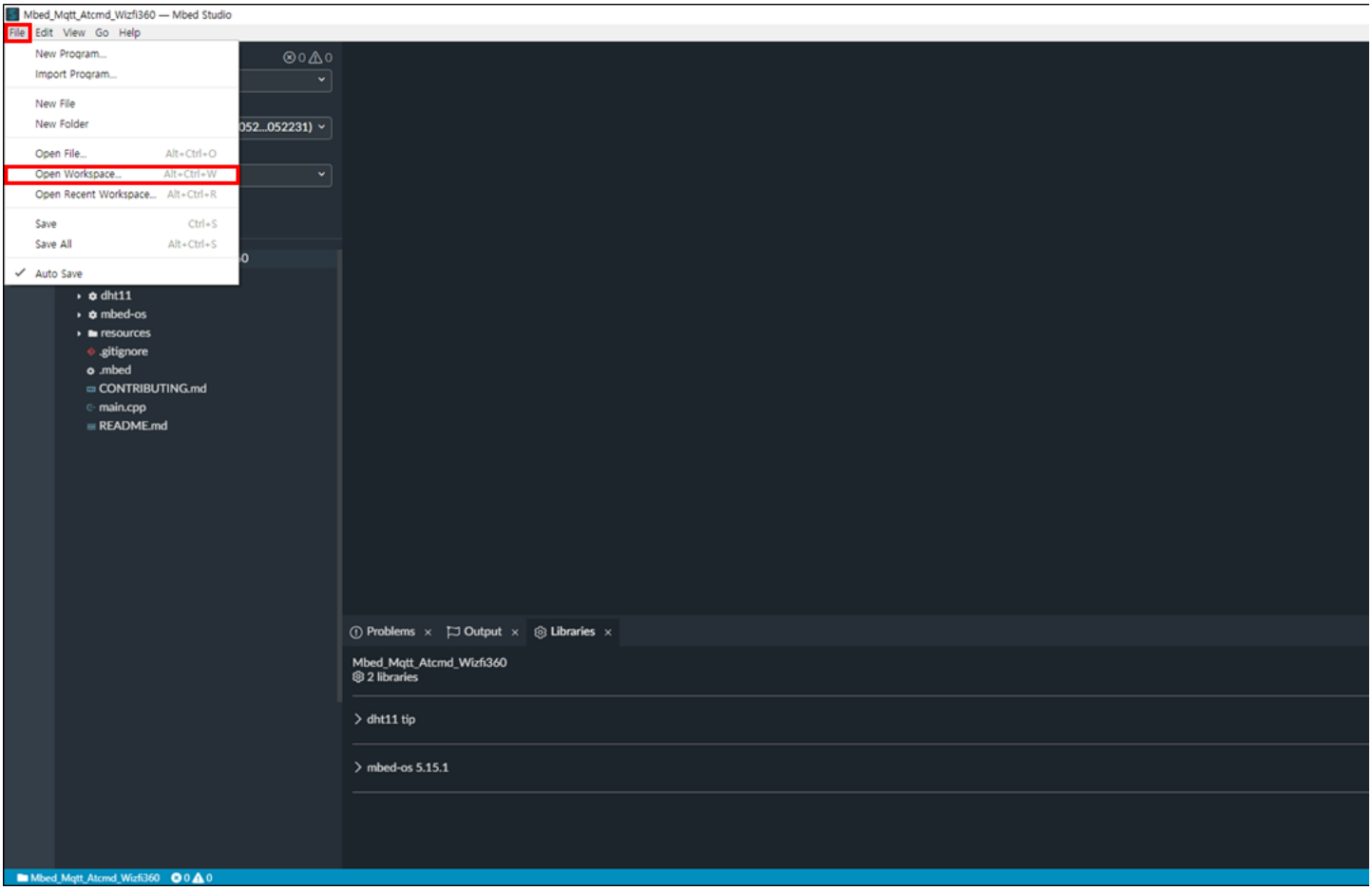
#동작 예제

#Mbed 예제 코드 다운로드

예제 Download를 한 후, **File** > **Open Workspace**을 선택하여 **Project 실행**합니다.

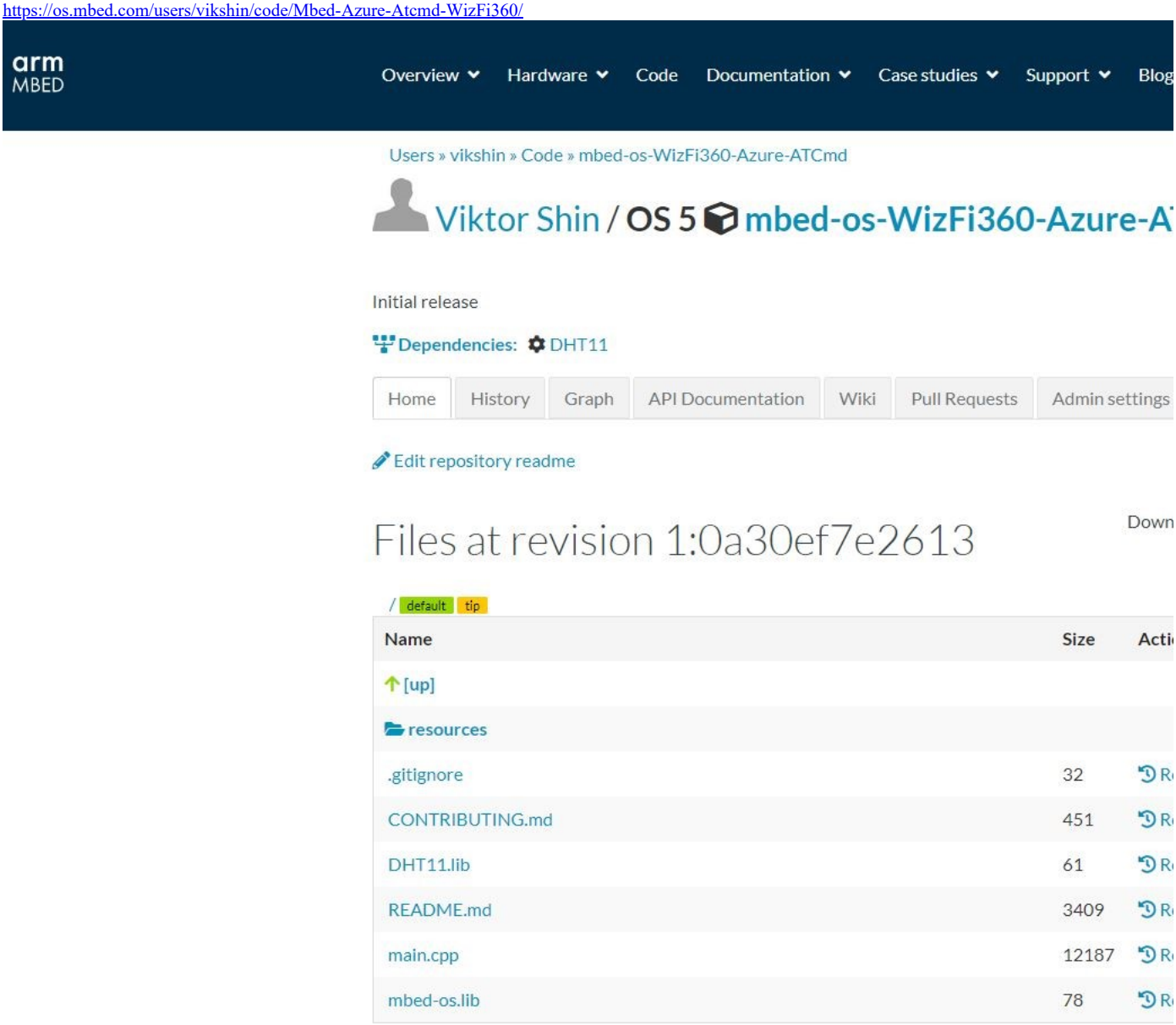
예제는 다음 경로에 위치하고 있는 Project를 참고 바랍니다.

samples/Wi-Fi/Mbed_Azure_Atcmd_Wizfi360



Online Compiler 경우 다음 링크에서 Compiler에 import을 할 수 있습니다.

https://os.mbed.com/users/vikshin/code/Mbed-Azure-Atcmd-WizFi360/



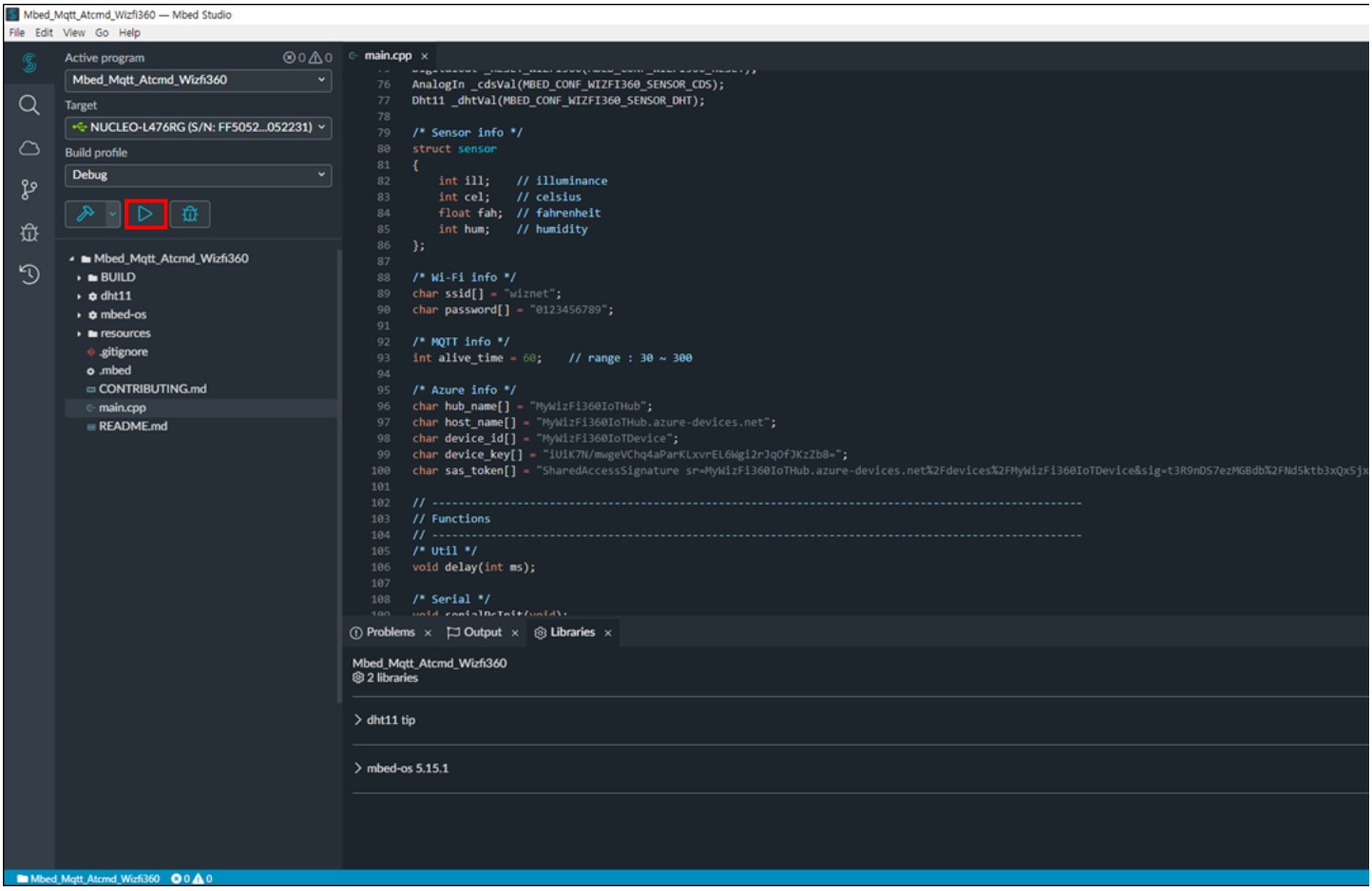
#Modify parameters

Azure IoT Hub 연결 위한 WiFi ssid, WiFi password, Hub ID, Device ID, Device Key 변경하여 테스트 해볼 수 있습니다.

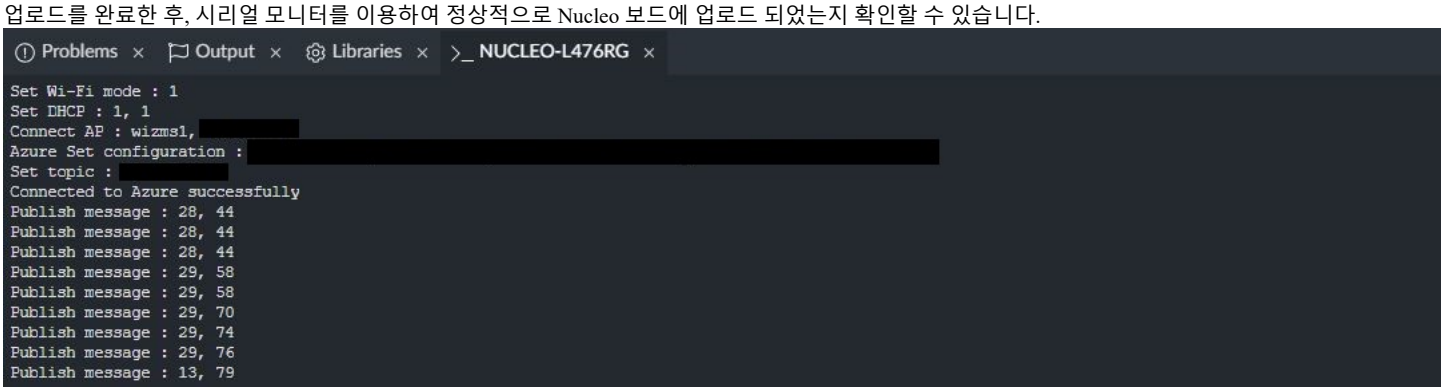
Copy

```
/* WiFi info */
char ssid[] = "XXXXXXXXXXXXXXXXXXXXXXXX";
char password[] = "XXXXXXXXXXXXXXXXXXXXXXXXX";

/* Azure info */
char hub_name[] = "XXXXXXXXXXXXXXXXXXXXXXXXX";
char device_id[] = "XXXXXXXXXXXXXXXXXXXXXXXXX";
char device_primary_key[] = "XXXXXXXXXXXXXXXXXXXXXXXXXX=";
```

Run Program을 눌러 Project Build 및 Run을 합니다.



업로드를 완료한 후, 시리얼 모니터를 이용하여 정상적으로 Nucleo 보드에 업로드 되었는지 확인할 수 있습니다.

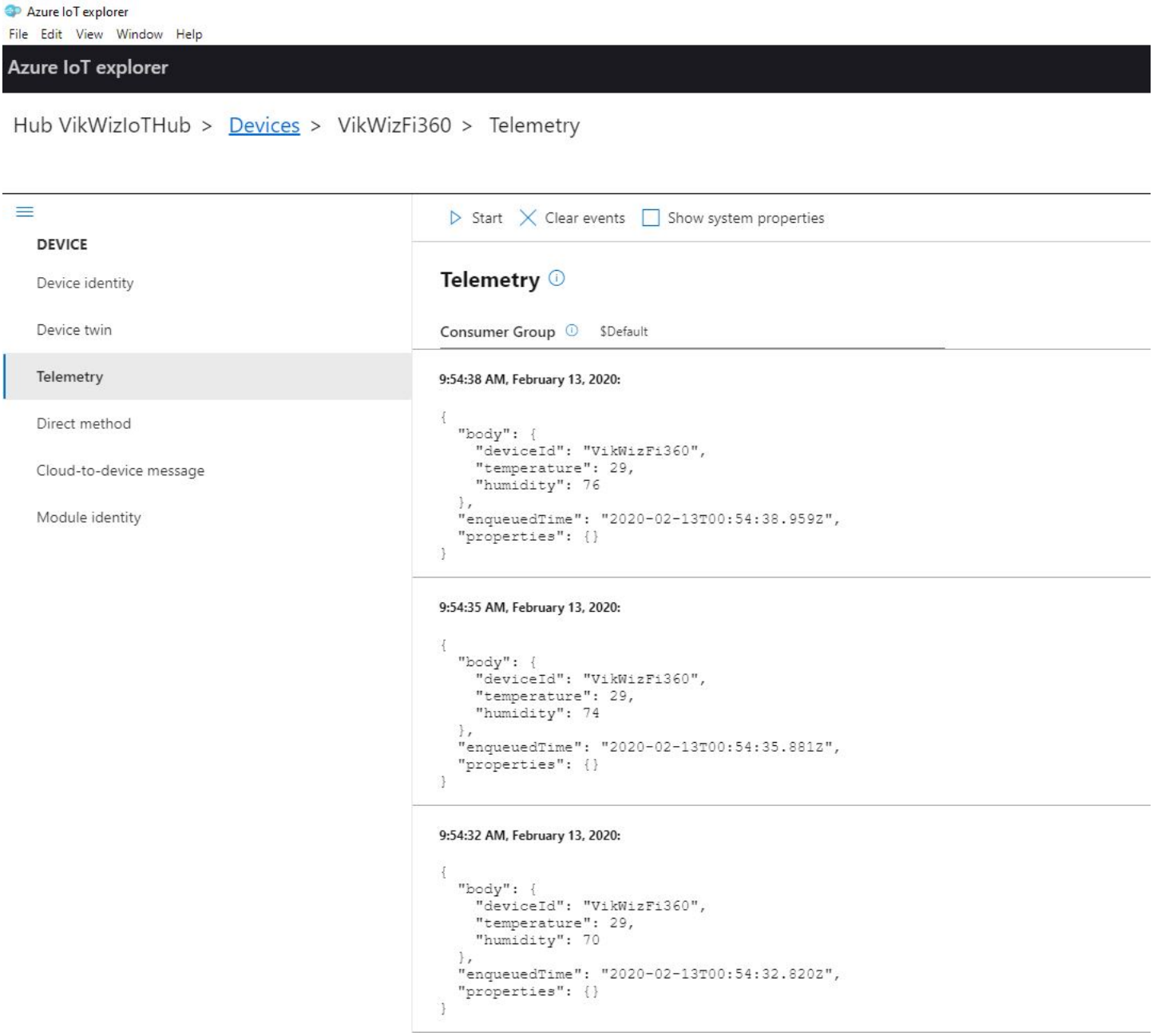


#동작 예제 결과

1. IoT Explorer 에서 Telemetry Section안에 "Start" 버튼을 누릅니다.

   MQTTPUB 명령을 통해 메시지를 보내기 전에 "Start" 버튼을 눌러야 합니다.

2. MQTTPUB command으로 수신한 데이터를 확인 할 수 있습니다.



#더 보기

WizFi360 Azure AT Command를 이용하여 Azure IoT Hub에 연결

Connect to Azure IoT Hub using NUCLEO-L476RG + WizFi360 Azure AT Command

#Getting Started

#Hardware Requirement

- NUCLEO-L476RG
- Desktop or laptop computer
- USB cable
- WizFi360-EVB-Shield

#Software Requirement

- Microsoft Azure Account (To create Azure account press here.)
- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Azure IoT Explorer
- MBED Studio / MBED Online Compiler

#Introduction

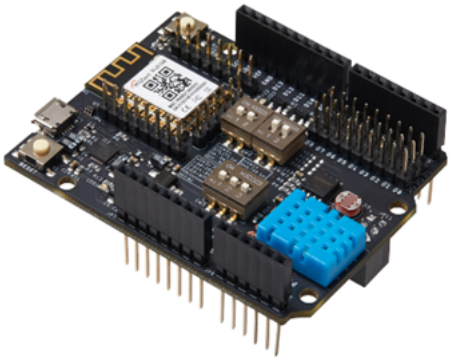
Microsoft Azure is cloud computing service. Using WizFi360 we can connect to Azure services, transmit data and monitor status.

In this document we will guide how to connect Arduino Mega 2560 + WizFi360 to MS Azure Services. Thic process consists of following steps:

- Azure IoT Hub preparation
- IoT device registration
- Connect to Azure IoT and transmit data

For Azure IoT Hub setup and IoT device creation please refer to [Azure Cloud Introduction].

For this guide WizFi360-EVB-Shield Evaluation board was used.

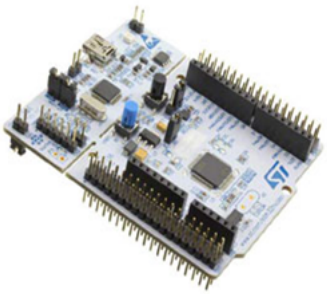


#Device preparation

#Hardware configuration

WizFi360-EVB-Shield will be installed on top of NUCLEO-L476RG. Therefore DIP Switch and jumper cables shall be connected as following:

- SW1 : Off
- SW2 : Off
- SW3 : On
- D2 : UART Tx
- D8 : UART Rx



#2. Device connection

After connecting hardware, connect NUCLEO-L476RG to Desktop or Laptop using USB Cable.

Check **COM Port** from **Device Manager**.

If COM port cannot be found in Device manager, check link below and follow instructions.

- [ST-LINK, ST-LINK/V2, ST-LINK/V2-1 USB driver][Link-St_Link_St_Link_V2_St_Link_V2_1_Usb_Driver]

#AT commands

#1. Set current WiFi mode (not saved in flash)

AT Command: AT+CWMODE_CUR Syntax:

Type	Command	Response
Query	AT+CWMODE_CUR?	+CWMODE:<mode> OK
Set	AT+CWMODE_CUR=<mode>	OK

Defined values:

Mode	Value
1	Station mode
2	SoftAP mode (factory default)
3	Station+SoftAP mode

#2. Enable DHCP

AT Command: AT+CWDHCP_CUR Syntax:

Type	Command	Response
Query	AT+CWDHCP_CUR?	+CWDHCP_CUR:<para> OK
Set	AT+CWMODE_CUR=<para>,<en>	OK

Defined values:

Parameter	Value
0	DIsable SoftAP DHCP & Station DHCP.
1	Enable SoftAP DHCP & Disable Station DHCP.
2	Disable SoftAP DHCP & enable Station DHCP.
3	Enable SoftAP DHCP & Station DHCP. (factory default)

#3. List available APs

AT Command: AT+CWLAP Syntax:

Type	Command	Response
Query	AT+CWLAP	+CWLAP:([<ecn>,<ssid>,<rssi>,<mac>,<channel>,<wps>])

Defined values:

Parameter	Value
<ecn>	0: Open 1: WEP 2: WPA_PSK 3: WPA2_PSK 4:WPA_WPA2_PSK
<ssid>	string parameter. AP ssid
<rssi>	signal strength
<mac>	string parameter. AP mac
<wps>	0: Disable WPS 1: Enable WPS

#4. Connect to AP

AT Command: AT+CWJAP_CUR Syntax:

Type	Command	Response
Set	AT+CWJAP_CUR=<ssid>,<pwd>,[<bssid>]	+CWJAP_CUR:<ssid>,<bssid>,<channel>,<rssi> OK

Defined values:

Parameter	Value
<ssid>	string parameter. Target AP ssid. MAX: 32 bytes
<pwd>	string parameter. Target AP password. MAX: 64-byte ASCII
<bssid>	string parameter, target AP' MAC address, used in case if there are several APs with same SSID.

#5. Azure IoT Hub configuration set

AT Command: AT+AZSET

Syntax:

Type	Command	Response
Set	AT+AZSET=<iothub_name>,<device_id>,<device_key>	OK

Defined values:

Parameter	Value
<hub ID>	string parameter. IoT Hub ID
<device ID>	string parameter. IoT Device ID
<key>	string parameter, IoT Device Key

#6. Set MQTT Topic

AT Command: AT+MQTTTOPIC

Syntax:

Type	Command	Response
Set	AT+MQTTTOPIC=<publish topic>,<subscribe topic>,<subscribe topic2>,<subscribe topic3>	OK

Defined values:

Parameter	Value
<publish topic>	string parameter, topic where WizFi360 will publish
<subscribe topic>	string parameter, topic where WizFi360 will subscribe
<subscribe topic2>	string parameter, topic where WizFi360 will subscribe
<subscribe topic3>	string parameter, topic where WizFi360 will subscribe

Note:

- This command shall be set before connecting to broker.
- <subscribe topic2> & <subscribe topic3> are available in Firmware v1.0.5.0 and later.

#7. Connect to Azure

AT Command: AT+AZCON

Syntax:

Type	Command	Response
Set	AT+AZCON	CONNECT OK

Note: • Before sending this ocmmand, AT+AZSET command & AT+MQTTTOPIC command shall be set. • After connection data can be sent to Azure server using AT+MQTTPUB command. • For more details please refer to https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support.

#8. Publish a message

AT Command: AT+MQTTPUB

Syntax:

Type	Command	Response
Set	AT+MQTTPUB=<message>	OK

Note: • This command shall be used when MQTT connection is established. • Topic where data will be published is set with AT+MQTTTOPIC command, user shall define topic before connecting to broker.

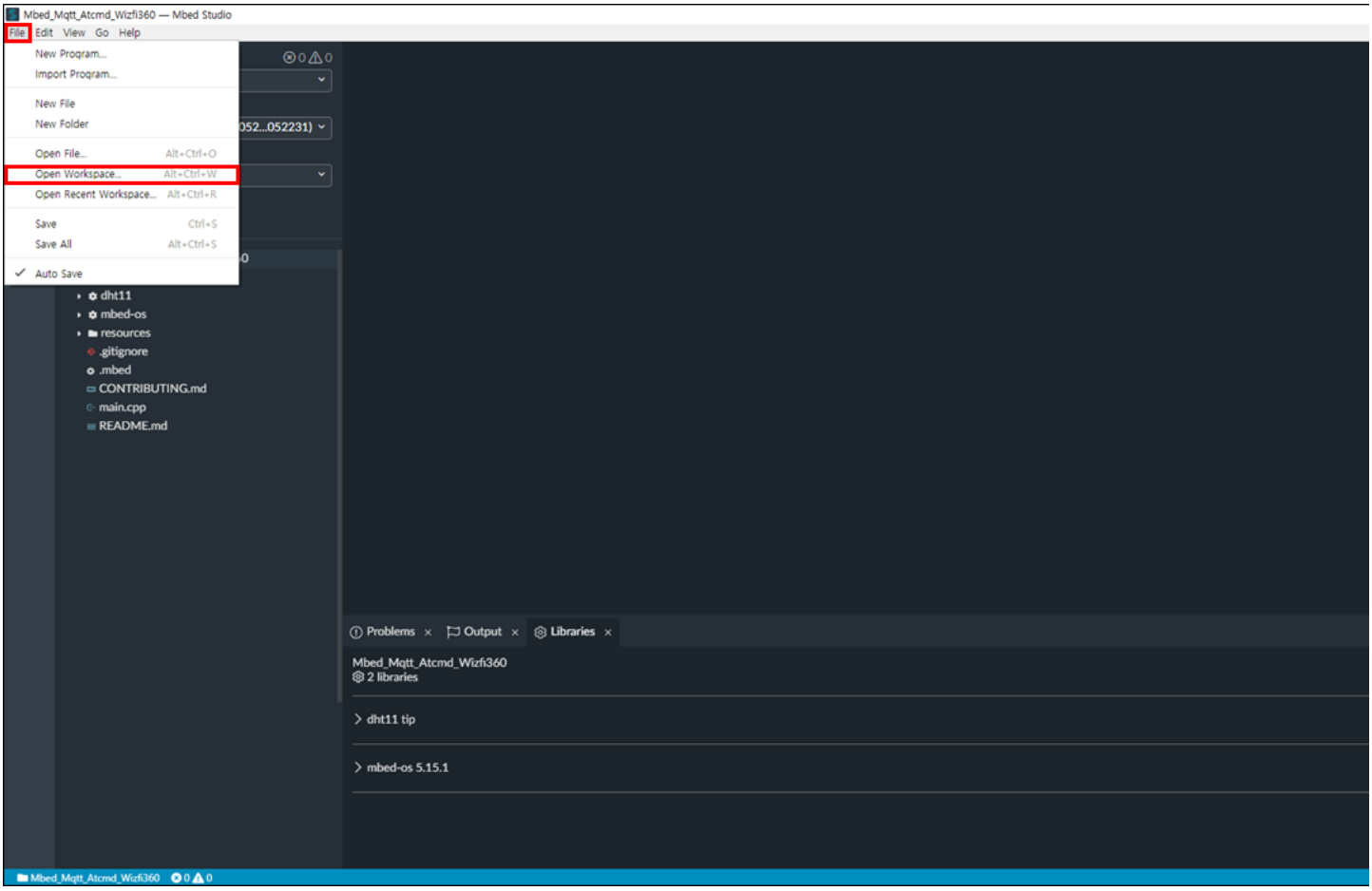
#Running sample code

#Mbed sample code download

After **Sample Download**, open and launch project in **File** > **Open Workspace**.

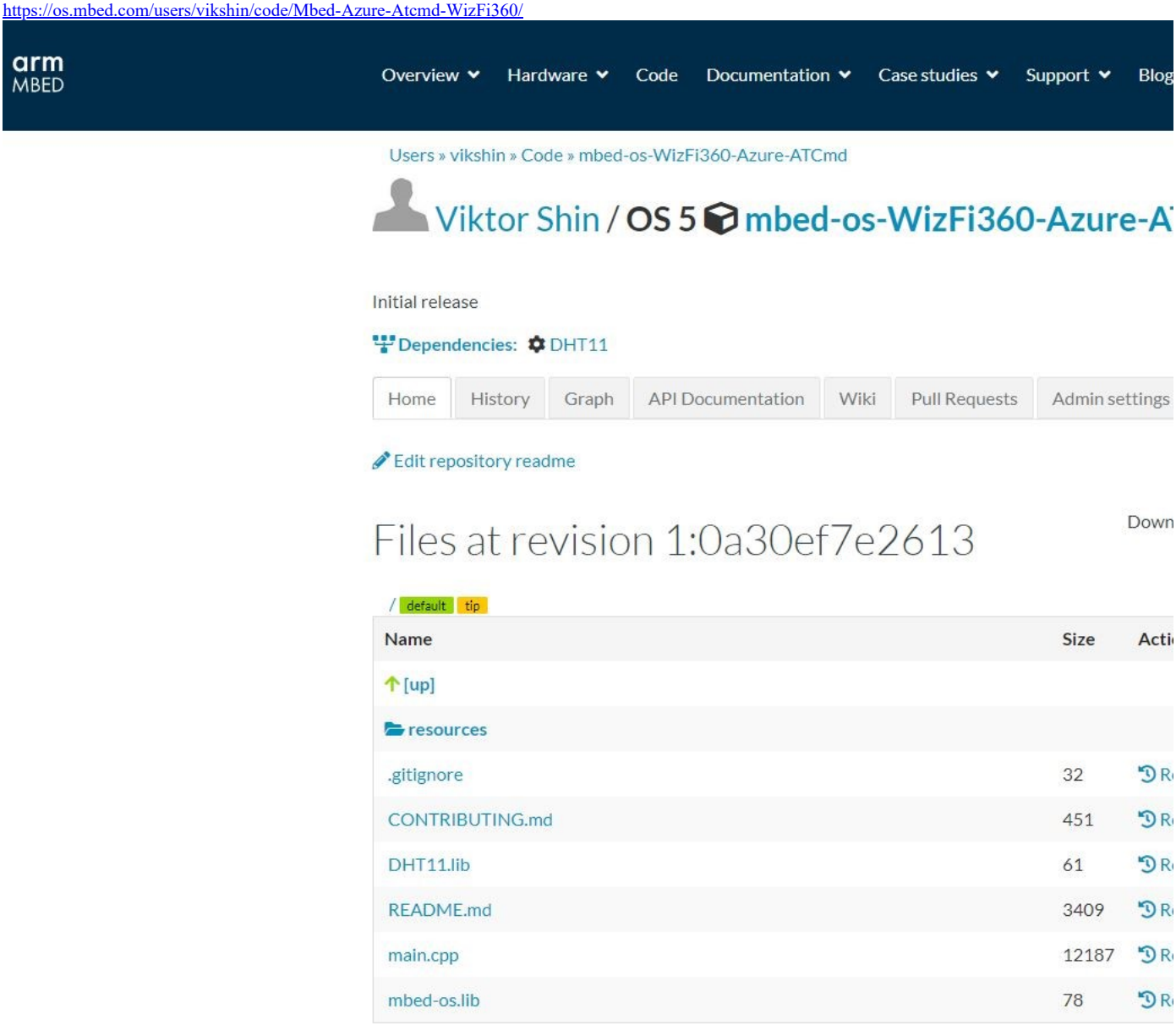
Sample code can be found at following path.

samples/Wi-Fi/Mbed_Azure_Atcmd_Wizfi360



Sample code also can be imported into Online Compiler from following link:

https://os.mbed.com/users/vikshin/code/Mbed-Azure-Atcmd-WizFi360/



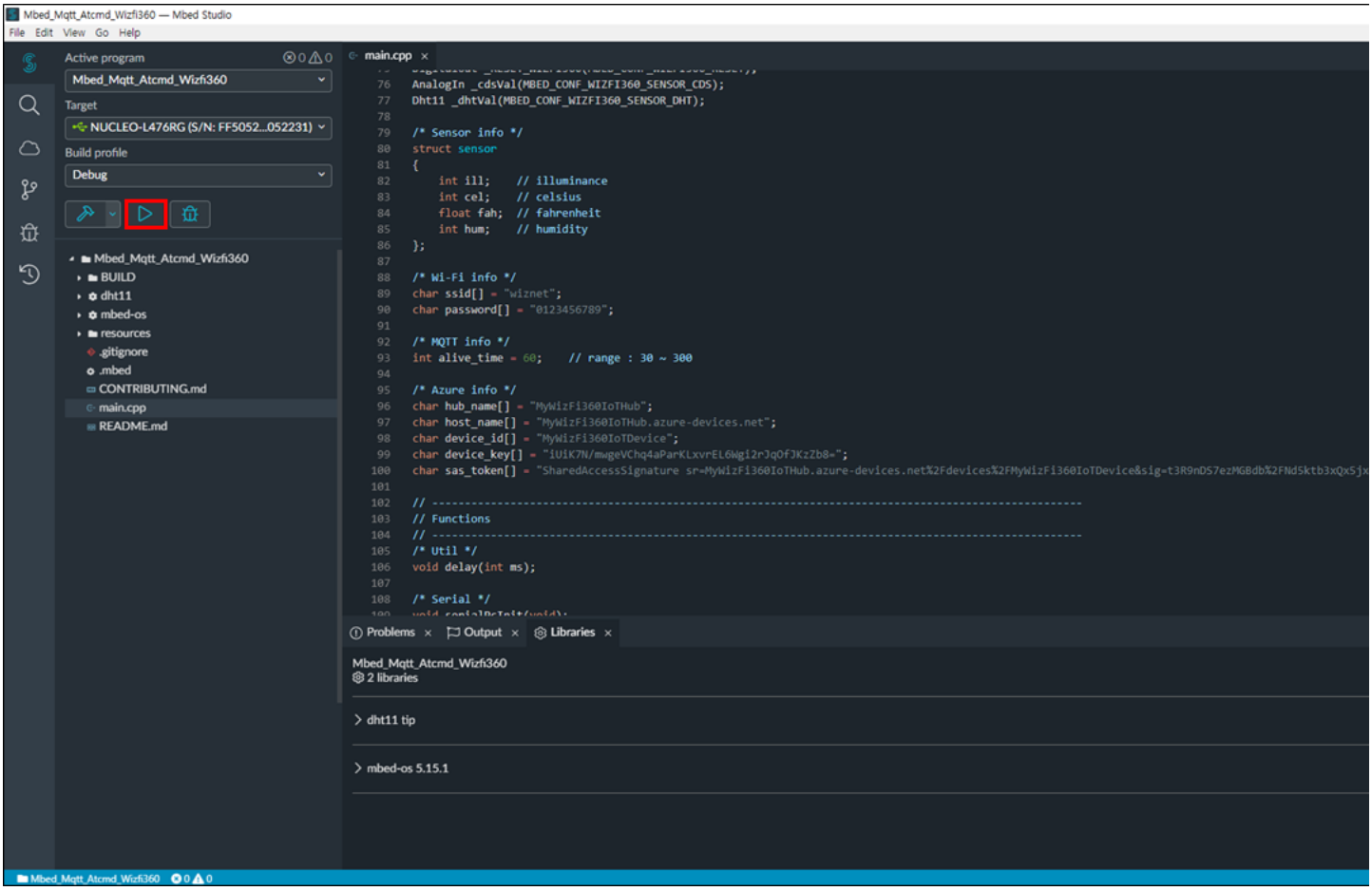
#Modify parameters

Update your credentials (WiFi ssid, WiFi pwd, Hub ID, Device ID, Device Key) in order to connect to Azure IoT Hub.

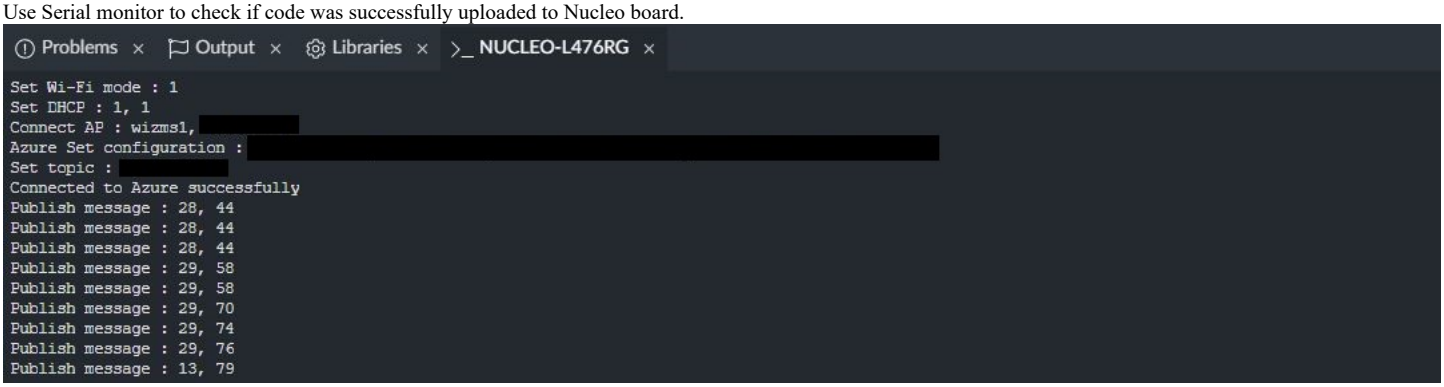
Copy

```
/* WiFi info */
char ssid[] = "XXXXXXXXXXXXXXXXXXXXXXXX";
char password[] = "XXXXXXXXXXXXXXXXXXXXXXXXXX";

/* Azure info */
char hub_name[] = "XXXXXXXXXXXXXXXXXXXXXXXXX";
char device_id[] = "XXXXXXXXXXXXXXXXXXXXXXXXX";
char device_primary_key[] = "XXXXXXXXXXXXXXXXXXXXXXXXXX=";
```

Press **Run Program** to build and run project.



Use Serial monitor to check if code was successfully uploaded to Nucleo board.



#Results

1. Press "Start" button in Telemetry Section in IoT Hub Explorer.

   "Start" button shall be pressed before sending data using MQTTPUB commands.

2. Check data sent with MQTTPUB command.

Azure IoT explorer

File Edit View Window Help

Azure IoT explorer

Hub VikWizIoTHub > Devices > VikWizFi360 > Telemetry

DEVICE

- Device identity
- Device twin
- Telemetry
- Direct method
- Cloud-to-device message
- Module identity

Start | Clear events | Show system properties

Telemetry

Consumer Group $Default

9:54:38 AM, February 13, 2020:


```
{
  "body": {
    "deviceId": "VikWizFi360",
    "temperature": 29,
    "humidity": 76
  },
  "enqueuedTime": "2020-02-13T00:54:38.959Z",
  "properties": {}
}
```


9:54:35 AM, February 13, 2020:


```
{
  "body": {
    "deviceId": "VikWizFi360",
    "temperature": 29,
    "humidity": 74
  },
  "enqueuedTime": "2020-02-13T00:54:35.881Z",
  "properties": {}
}
```


9:54:32 AM, February 13, 2020:


```
{
  "body": {
    "deviceId": "VikWizFi360",
    "temperature": 29,
    "humidity": 70
  },
  "enqueuedTime": "2020-02-13T00:54:32.820Z",
  "properties": {}
}
```

Arduino Mega 2560 + WizFi360 Azure AT Command를 이용하여 Azure IoT Hub에 연결

#시작하기 전에

#Hardware Requirement

- Arduino Mega 2560 board
- Desktop or laptop computer
- USB 케이블
- WizFi360-EVB-Shield

#Software Requirement

- MS Azure Account (Azure 구독이 아직 없는 경우 체험 무료계정을 만듭니다.)
- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Azure IoT Explorer
- Arduino IDE

#소개

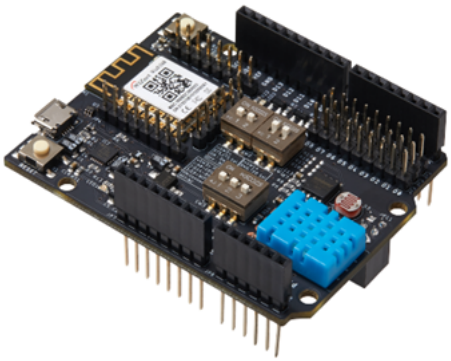
Microsoft Azure 는 Microsoft 의 클라우드 컴퓨팅 서비스입니다. Microsoft Azure 의 서비스에 WizFi360 을 연동하여 데이터를 클라우드로 전송하고, 모니터링 할 수 있습니다.

본 문서에서는 Arduino Mega 2560 + WizFi360 이용하여 MS Azure Services에 연결 방법에 대한 가이드를 제공합니다. 이 프로세스는 다음 단계로 구성됩니다.

- Azure IoT Hub 준비
- IoT 디바이스 등록
- Azure IoT와 연결 및 데이터 통신

Azure IoT Hub 준비와 IoT 디바이스 등록 과정 대해 Azure Cloud 소개 참조하시기 바랍니다.

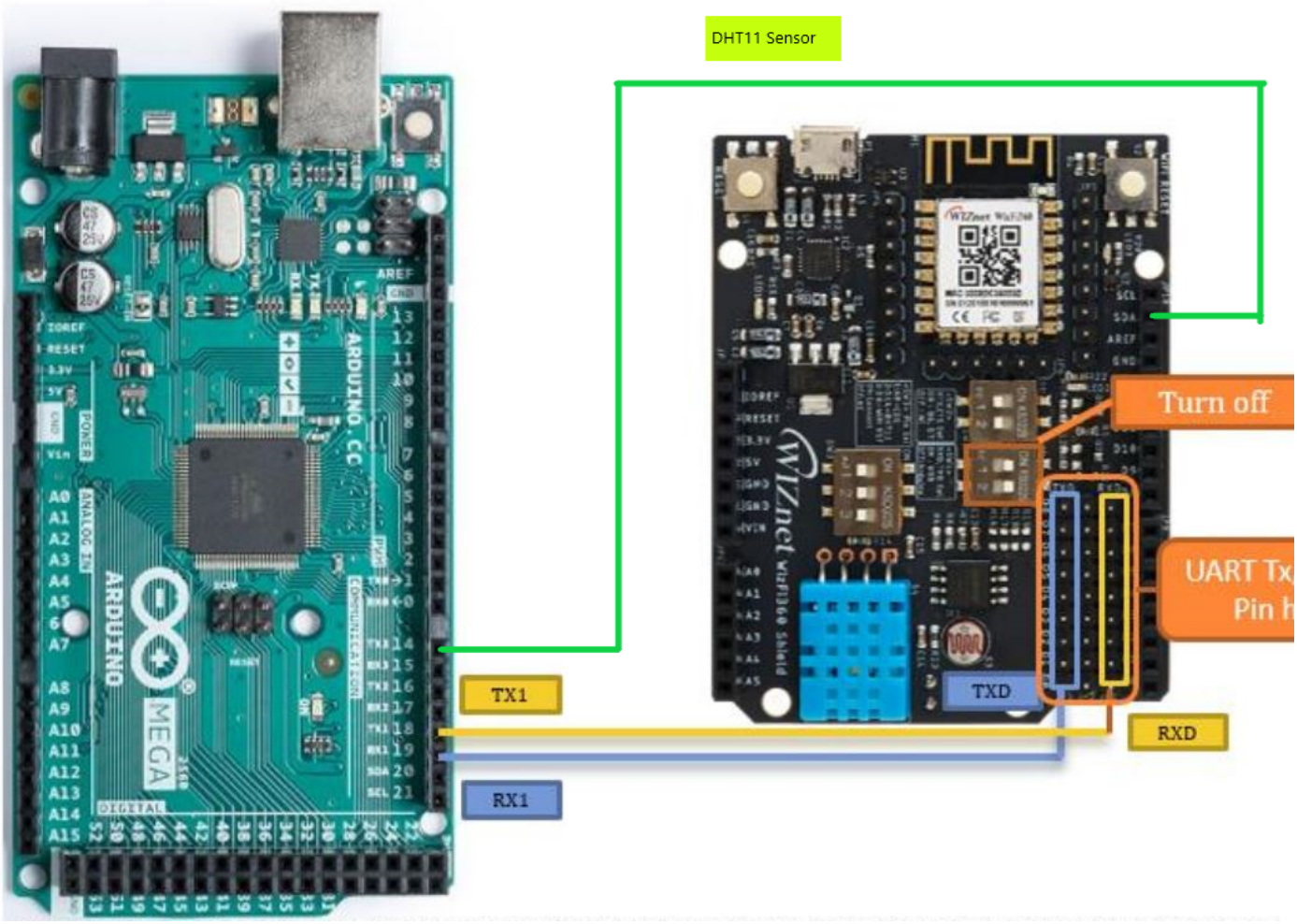
WiFi모듈 테스트를 위해 WizFi360-EVB-Shield Evaluation 보드를 사용되었습니다.



#디바이스 준비

#하드웨어 설정

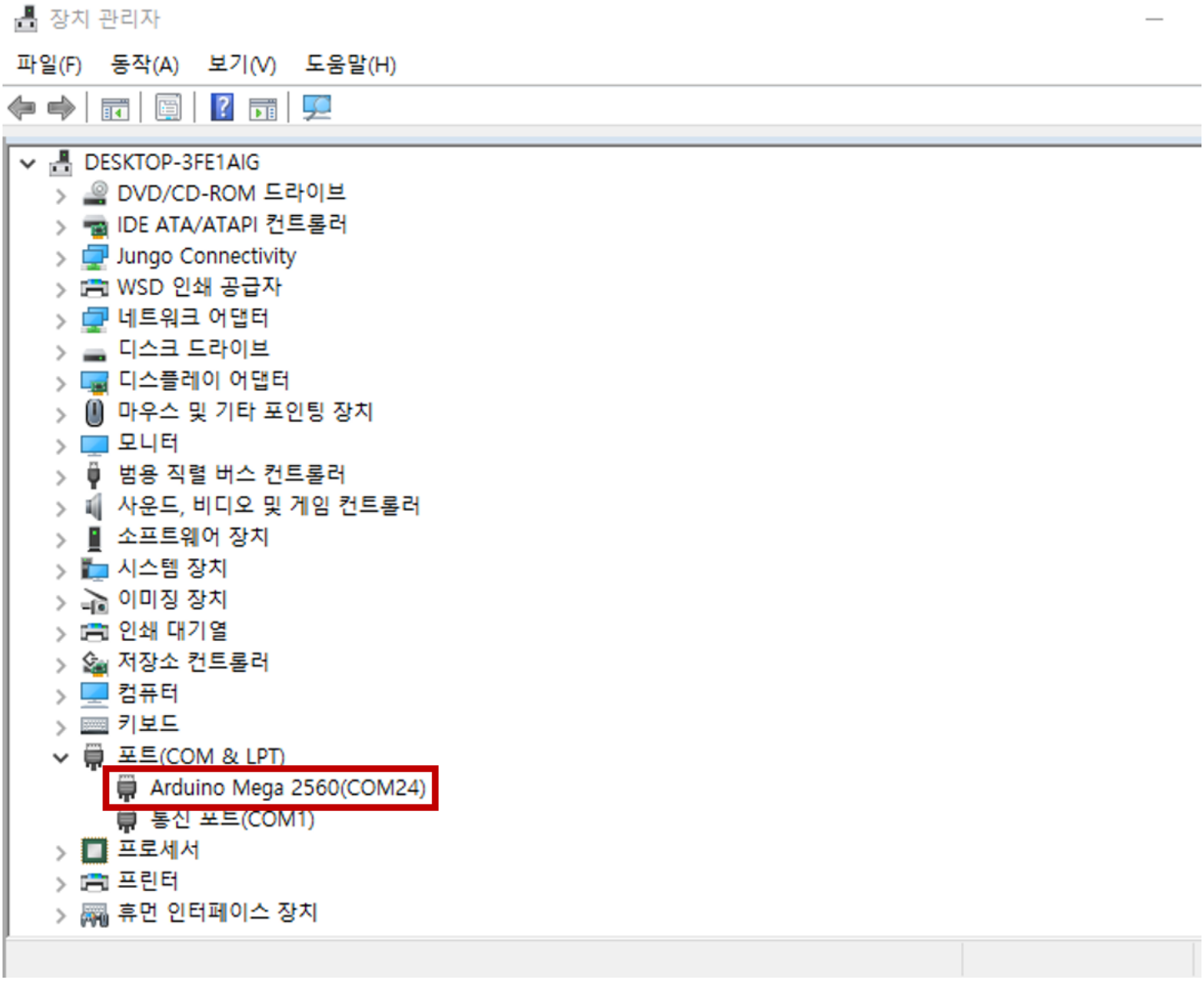
이 문서에서는 Arduino Mega2560 과 WizFi360-EVB-Shield 를 사용합니다. Arduino Code 에서 UART1 을 사용하여 WizFi360-EVB-Shield 와 통신하기 위해, Arduino 의 TX1, RX1 Pin 과 WizFi360-EVB-Shield 의 RXD, TXD pin 을 연결합니다. WizFi360-EVB-Shield 에서 RXD/TXD Selector 를 OFF 로 변경하여 USB 가 아닌 Pin 을 통해 UART 통신을 하도록 합니다.



WizFi360-EVB-Shield에 있는 DHT11 센서 사용을 위해 Arduino Mega D14 pin과 EVB D14 pin 연결해야 됩니다.

#디바이스 연결

하드웨어 설정 후 USB 커넥터를 이용하여 Arduino Mega2560 Rev3 보드와 PC를 연결합니다. PC 운영체제 장치 관리자에서 장치와 연결된 COM 포트를 확인할 수 있습니다.



Arduino IDE를 정상적으로 설치하면, 위와 같이 장치 관리자에서 COM 포트를 확인할 수 있습니다.

#AT 명령어

#1. Set current WiFi mode (not saved in flash)

AT Command: AT+CWMODE_CUR Syntax:

Type	Command	Response
Query	AT+CWMODE_CUR?	+CWMODE:<mode> OK
Set	AT+CWMODE_CUR=<mode>	OK

Defined values:

Mode	Value
1	Station mode
2	SoftAP mode (factory default)
3	Station+SoftAP mode

#2. Enable DHCP

AT Command: AT+CWDHCP_CUR Syntax:

Type	Command	Response
Query	AT+CWDHCP_CUR?	+CWDHCP_CUR:<para> OK
Set	AT+CWMODE_CUR=<para>,<en>	OK

Defined values:

Parameter	Value
0	SoftAP DHCP 와 Station DHCP 를 disable 한다.
1	SoftAP DHCP 는 enable 하고 Station DHCP 는 disable 한다.

Parameter	Value
2	2: SoftAP DHCP 는 disable 하고 Station DHCP 는 enable 한다.
3	SoftAP DHCP 와 Station DHCP 를 enable 한다. (factory default)

#3. List available APs

AT Command: AT+CWLAP Syntax:

Type	Command	Response
Query	AT+CWLAP	+CWLAP:([<ecn>,<ssid>,<rssi>,<mac>,<channel>,<wps>])

Defined values:

Parameter	Value
<ecn>	0: Open 1: WEP 2: WPA_PSK 3: WPA2_PSK 4:WPA_WPA2_PSK
<ssid>	string parameter. AP의 ssid
<rssi>	signal strength
<mac>	string parameter. AP의 mac
<wps>	0: WPS는 disable된다 1: WPS는 enable된다

#4. Connect to AP

AT Command: AT+CWJAP_CUR Syntax:

Type	Command	Response
Set	AT+CWJAP_CUR=<ssid>,<pwd>,[<bssid>]	+CWJAP_CUR:<ssid>,<bssid>,<channel>,<rssi> OK

Defined values:

Parameter	Value
<ssid>	string parameter. Target AP의 ssid. MAX: 32 bytes
<pwd>	string parameter. Target AP의 password. MAX: 64-byte ASCII
<bssid>	string parameter, target AP 의 MAC address, 같은 SSID 를 가진 여러 개의 AP 들이 있을 때 사용된다.

#5. Azure IoT Hub configuration set

AT Command: AT+AZSET

Syntax:

Type	Command	Response
Set	AT+AZSET=<iothub_name>,<device_id>,<device_key>	OK

Defined values:

Parameter	Value
<hub ID>	string parameter. IoT Hub의 ID
<device ID>	string parameter. IoT Device의 ID
<key>	string parameter, IoT Device의 Key

#6. Set MQTT Topic

AT Command: AT+MQTTTOPIC

Syntax:

Type	Command	Response
Set	AT+MQTTTOPIC=<publish topic>,<subscribe topic>,<subscribe topic2>,<subscribe topic3>	OK

Defined values:

Parameter	Value
<publish topic>	string parameter, WizFi360 이 publish 하는 topic
<subscribe topic>	string parameter, WizFi360 이 subscribe 하는 topic
<subscribe topic2>	string parameter, WizFi360 이 subscribe 하는 topic
<subscribe topic3>	string parameter, WizFi360 이 subscribe 하는 topic

Note:

- 이 command 는 broker 에 연결하기전에 설정되어야 합니다.
- <subscribe topic2> 와 <subscribe topic3>는 Firmware v1.0.5.0 이후 version 부터 사용가능 합니다.

#7. Connect to Azure

AT Command: AT+AZCON

Syntax:

Type	Command	Response
Set	AT+AZCON	CONNECT OK

Note: • 이 command 를 전송하기전에 AT+AZSET command 와 AT+MQTTTOPIC command 를 설정합니다. • Connect 이후 AT+MQTTPUB command 를 통해 Azure Sever 에 데이터를 전송합니다. • 자세한 내용은 https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support 를 참조하세요.

#8. Publish a message

AT Command: AT+MQTTPUB

Syntax:

Type	Command	Response
Set	AT+MQTTPUB=<message>	OK

Note: • 이 command 는 MQTT 가 연결되어 있을 때 사용됩니다. • Publish 한 data 의 topic 은 AT+MQTTTOPIC command 에 의해 결정되며, 사용자는 broker 에 연결하기전에 topic 을 설정합니다.

#동작 예제

#Arduino 예제 코드 다운로드

다음 링크에서 Arduino 예제 코드를 다운로드한 후, ino 확장자의 프로젝트 파일을 실행 시킵니다.

예제는 다음 경로에 위치하고 있는 Project를 참고 바랍니다.

samples/Wi-Fi/Arduino_Azure_Atcmd_Wizfi360

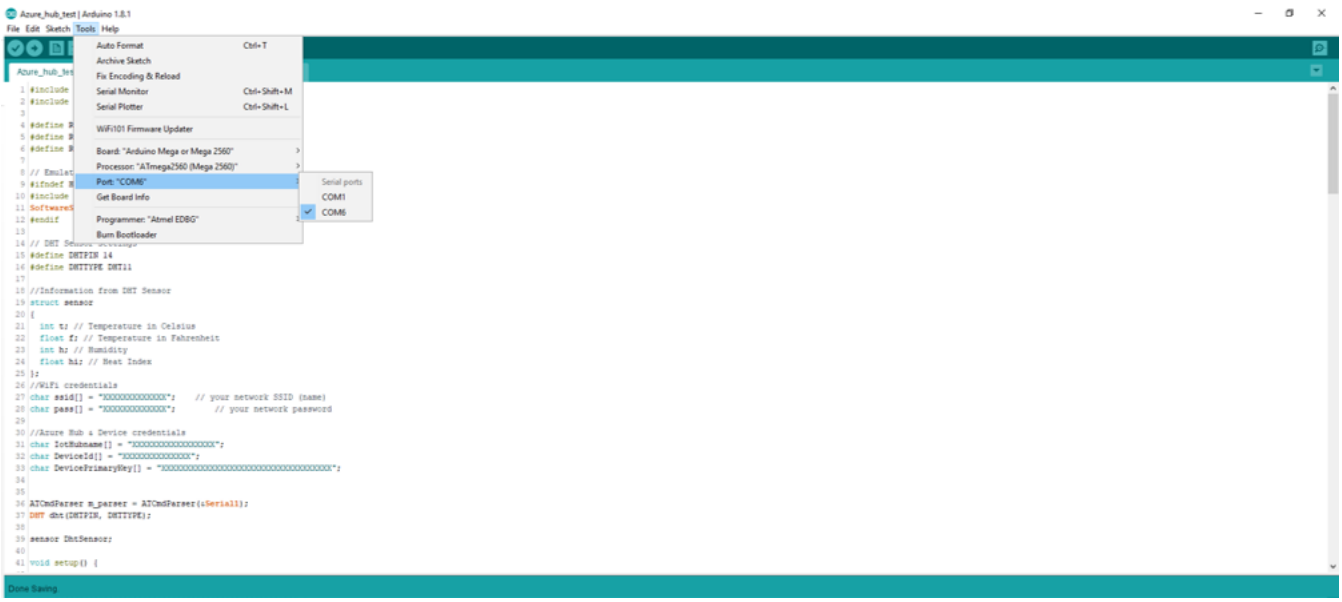
#Modify parameters

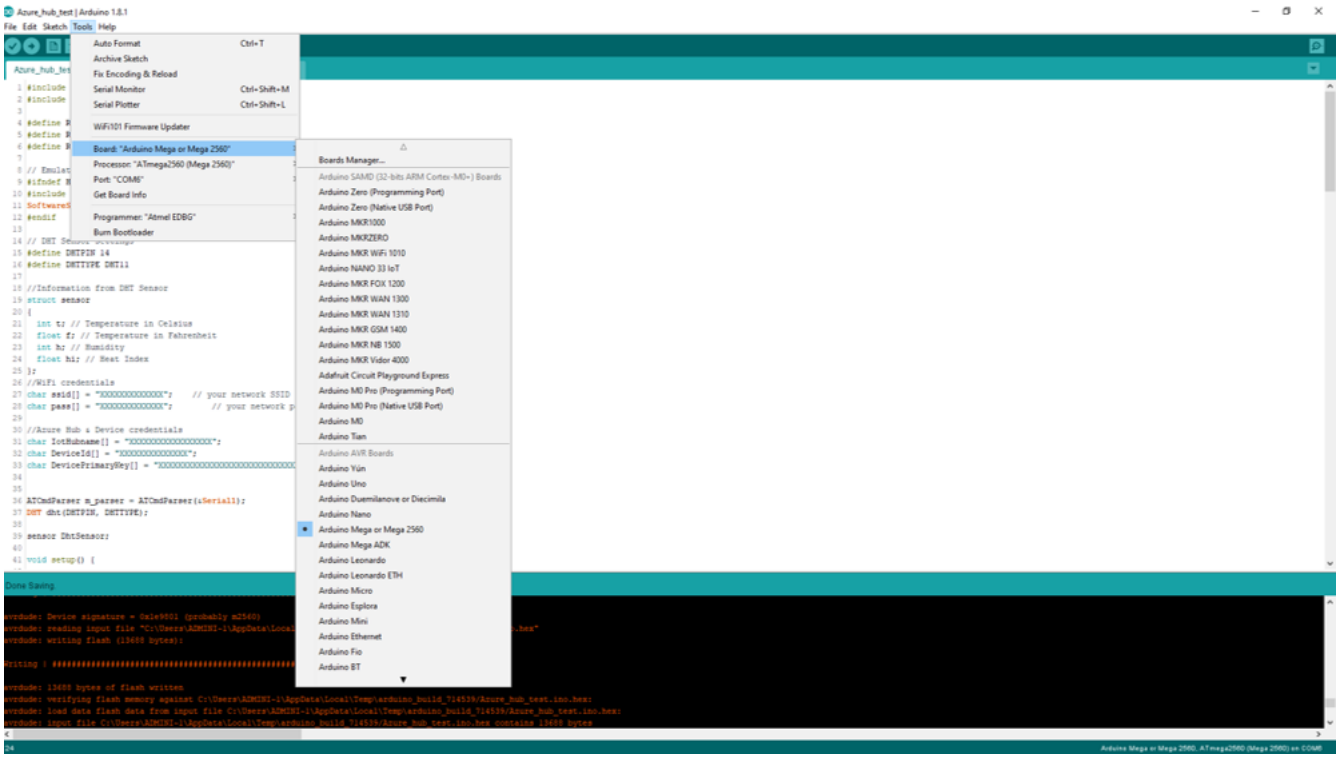
Azure IoT Hub 연결 위한 WiFi ssid, WiFi pwd, Hub ID, Device ID, Device Key 변경하여 테스트 해볼 수 있습니다.

Copy

```
//WiFi credentials
char ssid[] = "XXXXXXXXXXXXXXXXXX";     // your network SSID (name)
char pass[] = "XXXXXXXXXXXXXXXXXX";         // your network password

//Azure Hub & Device credentials
char IotHubname[] = "XXXXXXXXXXXXXXX";
char DeviceId[] = "XXXXXXXXXXXXX";
char DevicePrimaryKey[] = "XXXXXXXXXXXXXXXXXX";
```

다음 그림과 같이 Arduino Mega2560 보드와 포트를 선택하고, 컴파일을 수행합니다.





컴파일이 완료 되면 다음과 같이 업로드를 수행하여 최종적으로 보드에 업로드를 수행 합니다. 업로드가 정상적으로 완료되면 'avrdude done. Thank you.' 메시지를 확인 할 수 있습니다.

```
업로드 완료.

Writing | ################################################## | 100% 1.80s

avrdude: 11228 bytes of flash written
avrdude: verifying flash memory against C:\Users\WIZnet\AppData\Local\Temp\arduino_build_528741/WIoT-WM01_Arduino_Ping.ino.hex:
avrdude: load data flash data from input file C:\Users\WIZnet\AppData\Local\Temp\arduino_build_528741/WIoT-WM01_Arduino_Ping.ino.hex:
avrdude: input file C:\Users\WIZnet\AppData\Local\Temp\arduino_build_528741/WIoT-WM01_Arduino_Ping.ino.hex contains 11228 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 1.42s

avrdude: verifying ...
avrdude: 11228 bytes of flash verified

avrdude done.  Thank you.
```

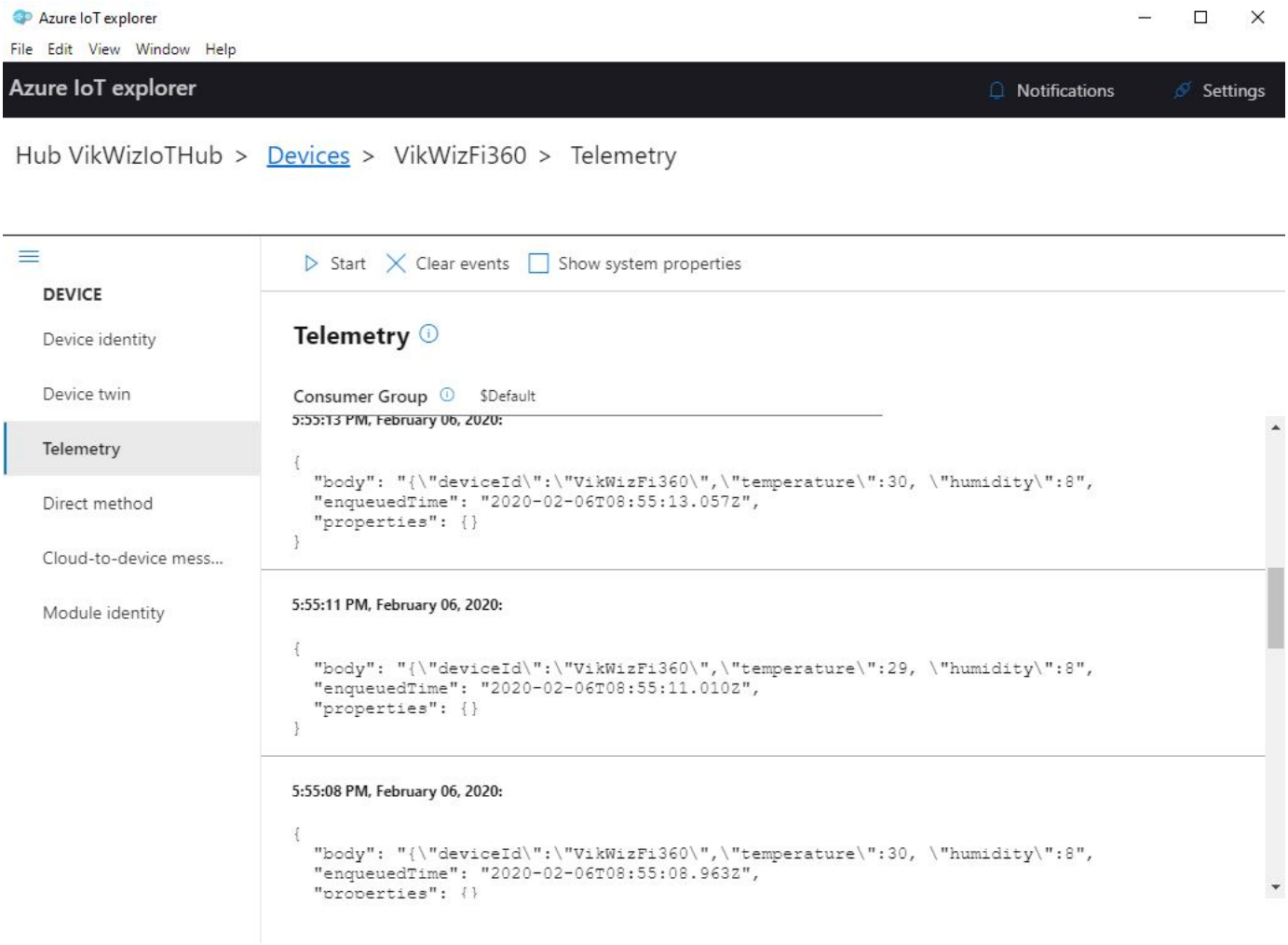
업로드를 완료한 후, 시리얼 모니터를 이용하여 정상적으로 Arduino Mega2560 보드에 업로드 되었는지 확인할 수 있습니다.

#동작 예제 결과

1. IoT Explorer 에서 Telemetry Section안에 "Start" 버튼을 누릅니다.

   MQTTPUB 명령을 통해 메시지를 보내기 전에 "Start" 버튼을 눌러야 합니다.

2. MQTTPUB command으로 수신한 데이터를 확인 할 수 있습니다.


Azure IoT explorer
File Edit View Window Help
Azure IoT explorer
Notifications
Settings
Hub VikWizIoTHub > Devices > VikWizFi360 > Telemetry
DEVICE
Device identity
Device twin
Telemetry
Direct method
Cloud-to-device mess...
Module identity
Start
Clear events
Show system properties
Telemetry
Consumer Group
$Default
5:55:13 PM, February 06, 2020:
{
"body": "{\"deviceId\":\"VikWizFi360\",\"temperature\":30, \"humidity\":8",
"enqueuedTime": "2020-02-06T08:55:13.057Z",
"properties": {}
}
5:55:11 PM, February 06, 2020:
{
"body": "{\"deviceId\":\"VikWizFi360\",\"temperature\":29, \"humidity\":8",
"enqueuedTime": "2020-02-06T08:55:11.010Z",
"properties": {}
}
5:55:08 PM, February 06, 2020:
{
"body": "{\"deviceId\":\"VikWizFi360\",\"temperature\":30, \"humidity\":8",
"enqueuedTime": "2020-02-06T08:55:08.963Z",
"properties": {}

#다음 단계

Connect to Azure IoT Hub using Arduino Mega 2560 + WizFi360 Azure AT Command

#Getting Started

#Hardware Requirement

- Arduino Mega 2560 board
- Desktop or laptop computer
- USB cable
- WizFi360-EVB-Shield

#Software Requirement

- MS Azure Account (To create Azure account press here.)
- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Azure IoT Explorer
- Arduino IDE

#Introduction

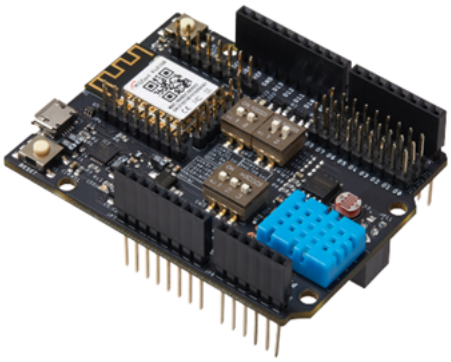
Microsoft Azure is cloud computing service. Using WizFi360 we can connect to Azure services, transmit data and monitor status.

In this document we will guide how to connect Arduino Mega 2560 + WizFi360 to MS Azure Services. Thic process consists of following steps:

- Azure IoT Hub preparation
- IoT device registration
- Connect to Azure IoT and transmit data

For Azure IoT Hub setup and IoT device creation please refer to Azure Cloud Introduction.

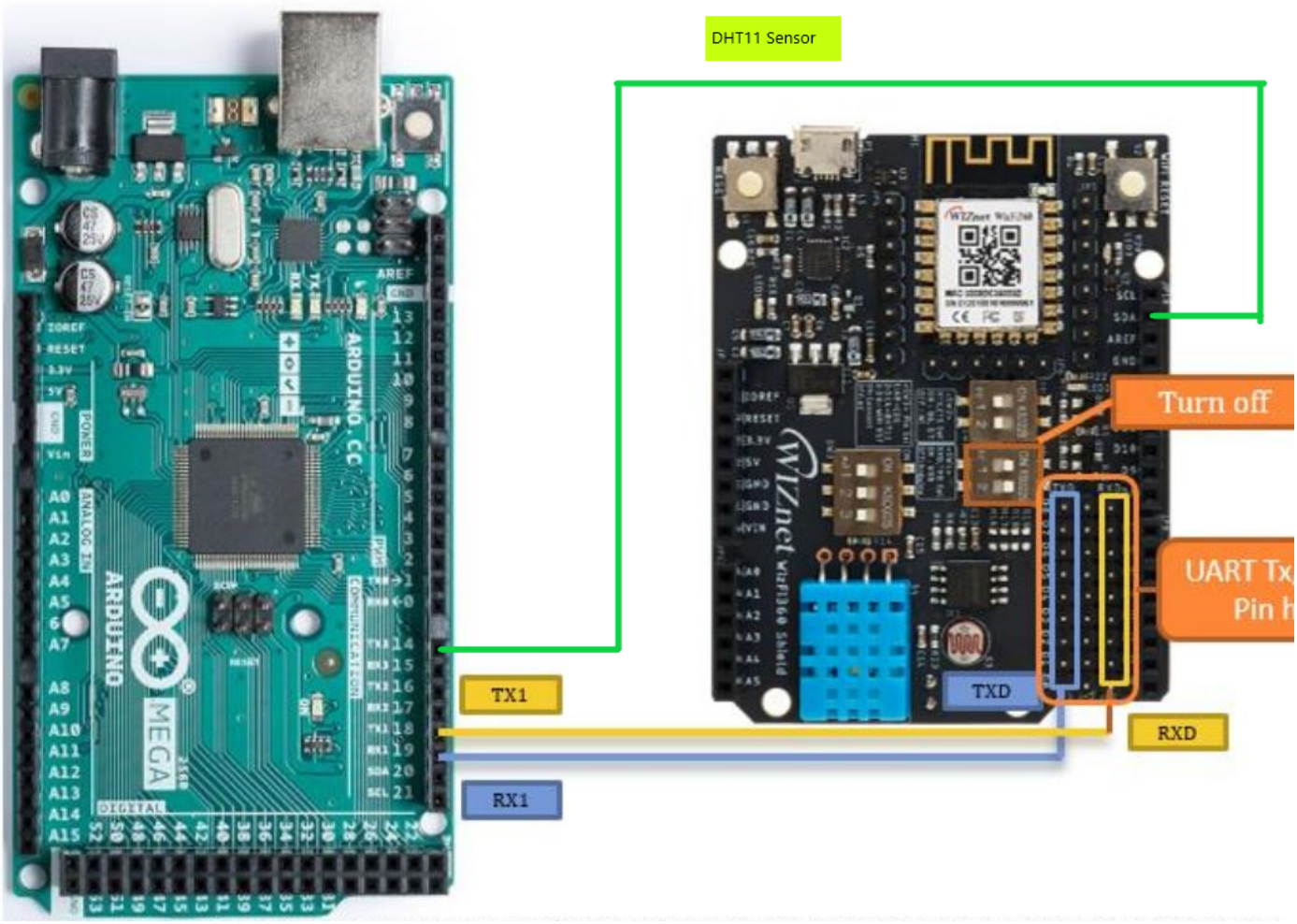
For this guide WizFi360-EVB-Shield Evaluation board was used.



#Device preparation

#Hardware configuration

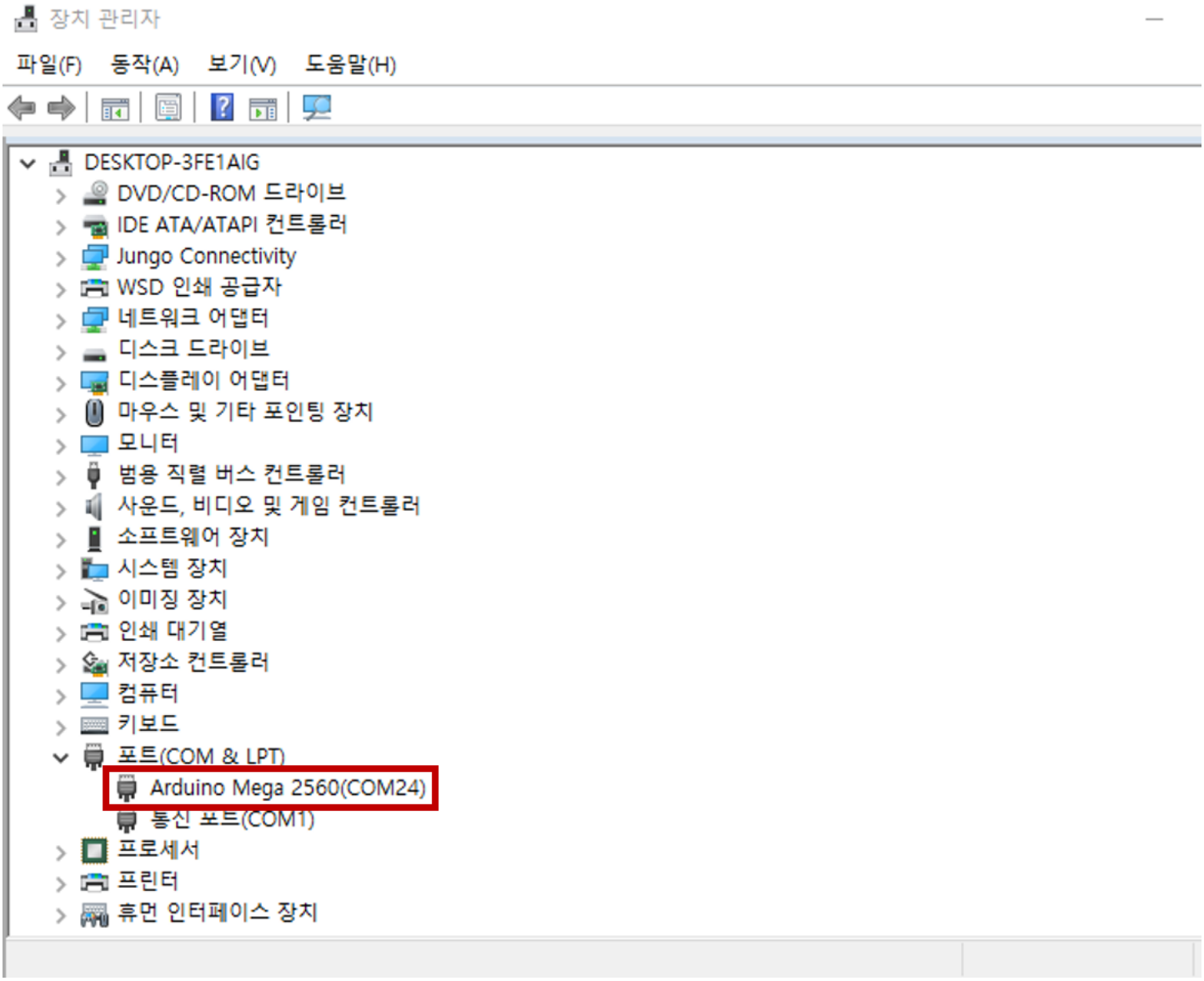
In this guide we will use Arduino Mega2560 and WizFi360-EVB-SHield. In Arduino Code we will use UART1 for ocmmunication with WizFi360-EVB-Shield. To do so we need to connect TX1, RX1 pins of Arduino board with RXD and TXD pins of WizFi360-EVB-Shield. Also, switch RXD/TXD Selector OFF in order to enable communication through pin not USB.



To use DHT11 sensor from WizFi360-EVB-Shield connect D14 pin on EVB board to D14 pin on Arduino Mega.

#Device connection

After configuring hardware connect Arduino Mega2560 board with PC. CHeck port number in Device Manager.



If everything setup correctly, then COM port can be checked in Device manager as shown on picture above.

#AT Commands

#1. Set current WiFi mode (not saved in flash)

AT Command: AT+CWMODE_CUR Syntax:

Type	Command	Response
Query	AT+CWMODE_CUR?	+CWMODE:<mode> OK
Set	AT+CWMODE_CUR=<mode>	OK

Defined values:

Mode	Value
1	Station mode
2	SoftAP mode (factory default)
3	Station+SoftAP mode

#2. Enable DHCP

AT Command: AT+CWDHCP_CUR Syntax:

Type	Command	Response
Query	AT+CWDHCP_CUR?	+CWDHCP_CUR:<para> OK
Set	AT+CWMODE_CUR=<para>,<en>	OK

Defined values:

Parameter	Value
0	DIsable SoftAP DHCP & Station DHCP.
1	Enable SoftAP DHCP & Disable Station DHCP.
2	Disable SoftAP DHCP & enable Station DHCP.

Parameter	Value
3	Enable SoftAP DHCP & Station DHCP. (factory default)

#3. List available APs

AT Command: AT+CWLAP Syntax:

Type	Command	Response
Query	AT+CWLAP	+CWLAP:([<ecn>,<ssid>,<rssi>,<mac>,<channel>,<wps>])

Defined values:

Parameter	Value
<ecn>	0: Open 1: WEP 2: WPA_PSK 3: WPA2_PSK 4:WPA_WPA2_PSK
<ssid>	string parameter. AP의 ssid
<rssi>	signal strength
<mac>	string parameter. AP의 mac
<wps>	0: Disable WPS 1: Enable WPS

#4. Connect to AP

AT Command: AT+CWJAP_CUR Syntax:

Type	Command	Response
Set	AT+CWJAP_CUR=<ssid>,<pwd>,[<bssid>]	+CWJAP_CUR:<ssid>,<bssid>,<channel>,<rssi> OK

Defined values:

Parameter	Value
<ssid>	string parameter. Target AP' ssid. MAX: 32 bytes
<pwd>	string parameter. Target AP' password. MAX: 64-byte ASCII
<bssid>	string parameter, target AP' MAC address, used in case if there are several APs with same SSID.

#5. Azure IoT Hub configuration set

AT Command: AT+AZSET

Syntax:

Type	Command	Response
Set	AT+AZSET=<iothub_name>,<device_id>,<device_key>	OK

Defined values:

Parameter	Value
<hub ID>	string parameter. IoT Hub ID
<device ID>	string parameter. IoT Device ID
<key>	string parameter, IoT Device Key

#6. Set MQTT Topic

AT Command: AT+MQTTTOPIC

Syntax:

Type	Command	Response
Set	AT+MQTTTOPIC=<publish topic>,<subscribe topic>,<subscribe topic2>,<subscribe topic3>	OK

Defined values:

Parameter	Value
<publish topic>	string parameter, topic where WizFi360 will publish
<subscribe topic>	string parameter, topic where WizFi360 will subscribe
<subscribe topic2>	string parameter, topic where WizFi360 will subscribe
<subscribe topic3>	string parameter, topic where WizFi360 will subscribe

Note:

- This command shall be set before connecting to broker.
- <subscribe topic2> & <subscribe topic3> are available in Firmware v1.0.5.0 and later.

#7. Connect to Azure

AT Command: AT+AZCON

Syntax:

Type	Command	Response
Set	AT+AZCON	CONNECT OK

Note: • Before sending this ocmmand, AT+AZSET command & AT+MQTTTOPIC command shall be set. • After connection data can be sent to Azure server using AT+MQTTPUB command. • For more details please refer to https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support.

#8. Publish a message

AT Command: AT+MQTTPUB

Syntax:

Type	Command	Response
Set	AT+MQTTPUB=<message>	OK

Note: • This command shall be used when MQTT connection is established. • Topic where data will be published is set with AT+MQTTTOPIC command, user shall define topic before connecting to broker.

#Running sample code

#Download sample Arduino code

Please download Arduino sample code from link below and open ino project file.

Sample can be found at below path.

samples/Wi-Fi/Arduino_Azure_Atcmd_Wizfi360

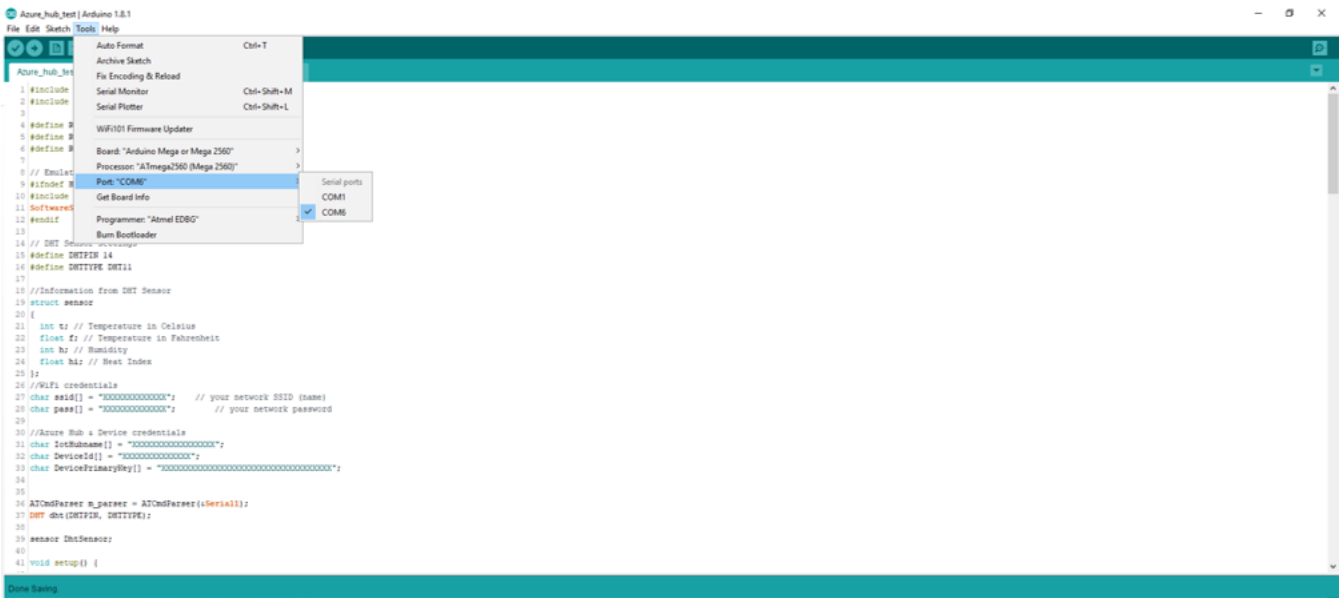
#Modify parameters

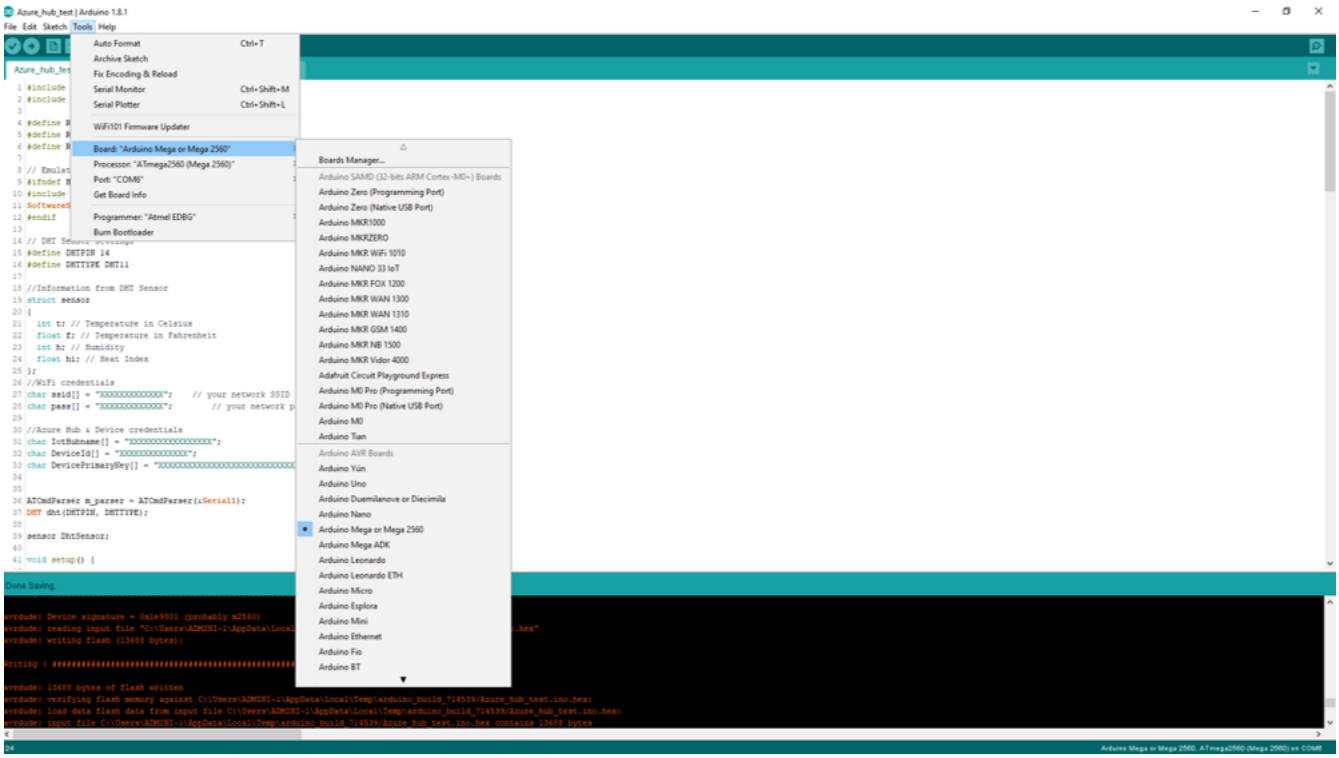
Update your credentials (WiFi ssid, WiFi pwd, Hub ID, Device ID, Device Key) in order to connect to Azure IoT Hub.

Copy

```
//WiFi credentials
char ssid[] = "XXXXXXXXXXXXXXXXXX";     // your network SSID (name)
char pass[] = "XXXXXXXXXXXXXXXXXX";         // your network password

//Azure Hub & Device credentials
char IotHubname[] = "XXXXXXXXXXXXXXX";
char DeviceId[] = "XXXXXXXXXXXXX";
char DevicePrimaryKey[] = "XXXXXXXXXXXXXXXXXX";
```

As shown on below picture, check your board & port, then compile code.





When compile is done, upload to board. When upload is done, you will see 'avrdude done. Thank you.' message.

```
업로드 완료.

Writing | ################################################## | 100% 1.80s

avrdude: 11228 bytes of flash written
avrdude: verifying flash memory against C:\Users\WIZnet\AppData\Local\Temp\arduino_build_528741/WIoT-WM01_Arduino_Ping.ino.hex:
avrdude: load data flash data from input file C:\Users\WIZnet\AppData\Local\Temp\arduino_build_528741/WIoT-WM01_Arduino_Ping.ino.hex:
avrdude: input file C:\Users\WIZnet\AppData\Local\Temp\arduino_build_528741/WIoT-WM01_Arduino_Ping.ino.hex contains 11228 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 1.42s

avrdude: verifying ...
avrdude: 11228 bytes of flash verified

avrdude done.  Thank you.
```

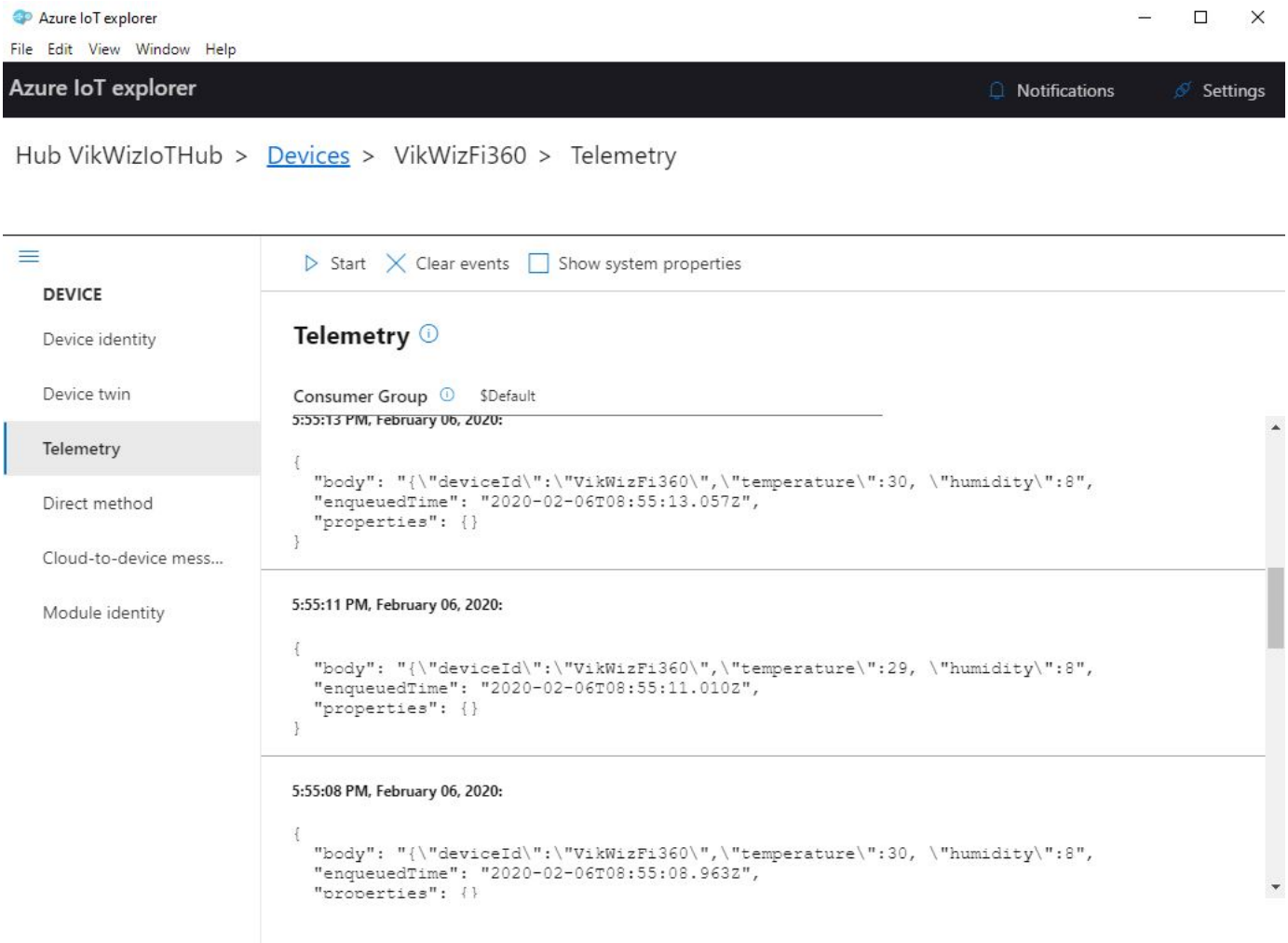
Use Serial monitor to check if code was successfully uploaded to Adruino Mega2560 board.

#Results

1. Press "Start" button in Telemetry Section in IoT Hub Explorer.

   "Start" button shall be pressed before sending data using MQTTPUB commands.

2. Check data sent with MQTTPUB command.


Azure IoT explorer
File Edit View Window Help
Azure IoT explorer
Notifications
Settings
Hub VikWizIoTHub > Devices > VikWizFi360 > Telemetry
Start
Clear events
Show system properties
DEVICE
Device identity
Device twin
Telemetry
Direct method
Cloud-to-device mess...
Module identity
Telemetry
Consumer Group
$Default
5:55:13 PM, February 06, 2020:
{
"body": "{\"deviceId\":\"VikWizFi360\",\"temperature\":30, \"humidity\":8",
"enqueuedTime": "2020-02-06T08:55:13.057Z",
"properties": {}
}
5:55:11 PM, February 06, 2020:
{
"body": "{\"deviceId\":\"VikWizFi360\",\"temperature\":29, \"humidity\":8",
"enqueuedTime": "2020-02-06T08:55:11.010Z",
"properties": {}
}
5:55:08 PM, February 06, 2020:
{
"body": "{\"deviceId\":\"VikWizFi360\",\"temperature\":30, \"humidity\":8",
"enqueuedTime": "2020-02-06T08:55:08.963Z",
"properties": {}

WizFi360 MQTT AT Command를 이용한 Azure IoT Hub 연동 예제

#시작하기 전에

Azure Portal에 Login을 합니다. 계정이 없는 경우, 계정 생성 후에 Login을 진행합니다.

※ 본 문서는 **체험 계정**으로 진행합니다.

Azure Portal을 사용하여 IoT Hub 만들기 등 앞선 일련의 과정에 대하여 Azure Cloud 서비스 소개를 참조하시기 바랍니다.

- [MS] Azure Portal을 사용하여 IoT Hub 만들기
- Azure Portal을 사용하여 Blob Storage 만들기
- Azure Portal을 사용하여 Stream Analytics 만들기
- Azure Portal을 사용하여 Stream Analytics 작업 입 · 출력 구성 및 변환 Query 정의
- WizFi360 MQTT AT Command를 이용하여 Azure IoT Hub에 연동

#소개

Microsoft Azure Service에 **WizFi360**을 **연동**하여, Data를 Cloud로 전송하고, Monitoring을 할 수 있습니다.

Data 통신은 다음과 같은 구조로 이루어집니다.

MQTT AT Command를 이용하여, IoT Hub Service 연결 및 Data 송신을 합니다.

IoT Hub로 송신이 된 Data는 Stream Analytics를 통하여 Data 저장소 Blob Storage로 저장이 됩니다.

본 문서는 Arduino 기반 WizFi360 MQTT AT Command 이용한 Microsoft Azure Service 연동 예제에 대하여 Guide를 제공합니다.

#Step 1: 필수 구성 요소

본 문서를 따라하기 전에 다음 항목이 준비되어야 합니다.

#Hadrware

- Desktop or Laptop Computer
- WizFi360-EVB-Shield
- Arduino Mega 2560
- USB Type-B Cable

#Software

- Arduino IDE
- Preferred Serial Terminal (TeraTerm, Hercules, etc . . .)

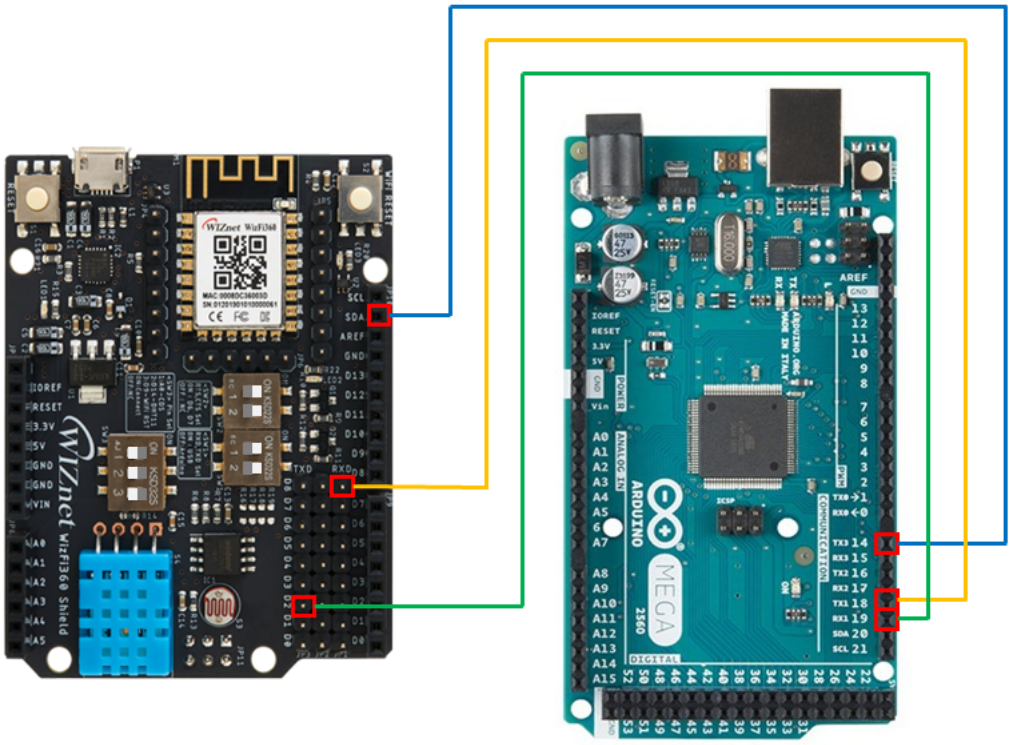
#Step 2: Device 준비

#1. Hardware 준비

WizFi360-EVB-Shield는 Arduino Mega 2560과 결합을 하여 사용되어 집니다. 따라서 WizFi360-EVB-Shield의 DIP Switch 설정 및 Jumper Cable 연결이 다음과 같이 필요합니다.

- SW1 : Off
- SW2 : Off
- SW3 : On
- WizFi360-EVB-Shield : D8 - Arduino Mega 2560 : 18

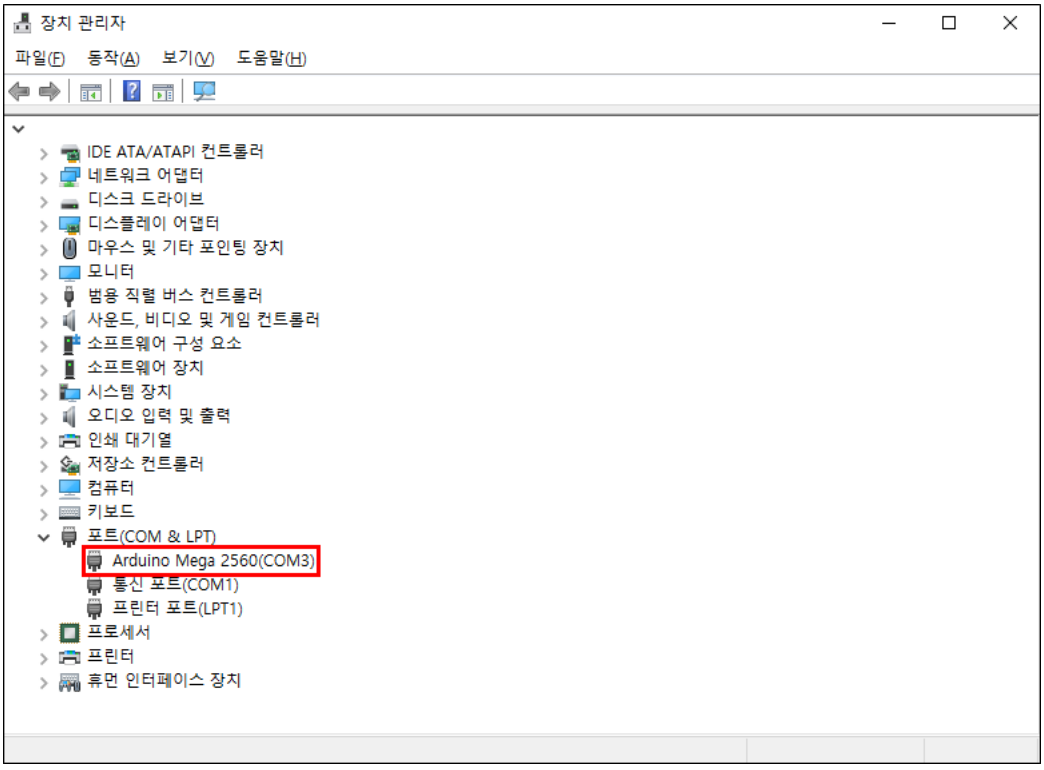
- WizFi360-EVB-Shield : D2 - Arduino Mega 2560 : 19
- WizFi360-EVB-Shield : SDA - Arduino Mega 2560 : 14



#2. Device 연결

Hardware 설정 후, USB Type-B Cable을 이용하여 Arduino Mega 2560을 Desktop 혹은 Laptop Computer와 연결을 합니다.

장치 관리자에서 Arduino Mega 2560과 연결된 **COM Port**를 확인 할 수 있습니다.



Arduino IDE를 정상적으로 설치를 하였다면 위와 같이 장치 관리자에서 COM 포트를 확인할 수 있습니다.

장치 관리자에서 COM Port를 확인 할 수 없는 경우, 다음 Link의 설명에 따라 설정 바랍니다.

- Manually install Drivers on Windows

#Step 3: 동작 예제

#1. 예제 Download 및 실행

예제 Download를 한 후, .ino 확장자를 선택하여 **Project 실행**합니다.

예제는 다음 경로에 위치하고 있는 Project를 참고 바랍니다.

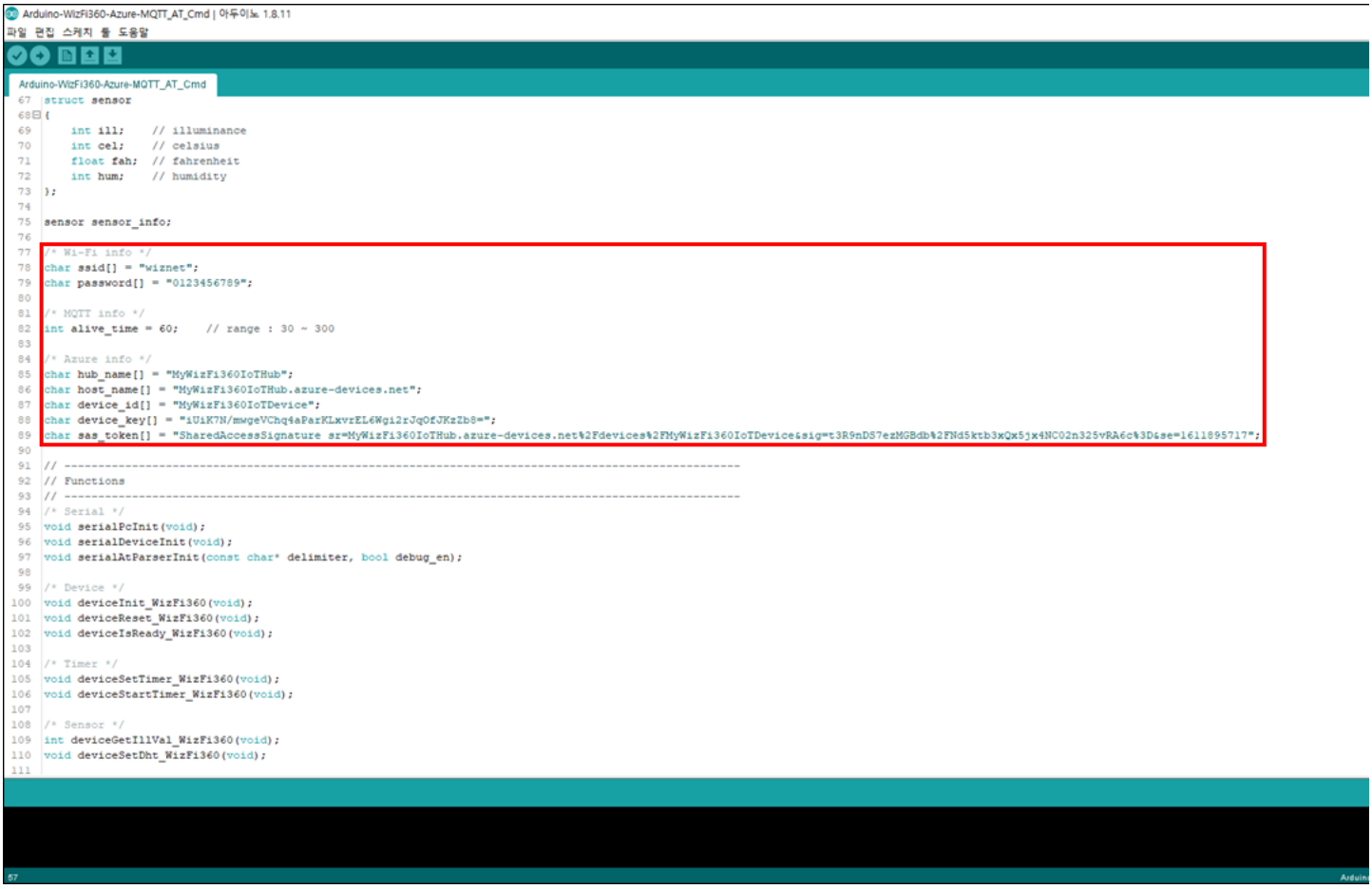
- **samples/Wi-Fi/Arduino_Mqtt_Atcmd_Wizfi360**

#2. Parameter 값 수정

Azure IoT Hub에 연결을 하기 위하여, 다음의 Parameter를 수정합니다.

Copy

```
/* Wi-Fi info */
char ssid[] = "xxxxxxxxxx";
char password[] = "xxxxxxxxxx";

/* MQTT info */
int alive_time = xx;     // range : 30 ~ 300

/* Azure info */
char hub_name[] = "xxxxxxxxxx";
char host_name[] = "xxxxxxxxxx";
char device_id[] = "xxxxxxxxxx";
char device_key[] = "xxxxxxxxxx";
char sas_token[] = "xxxxxxxxxx";
```



SAS Token 생성은 다음을 참고 바랍니다.

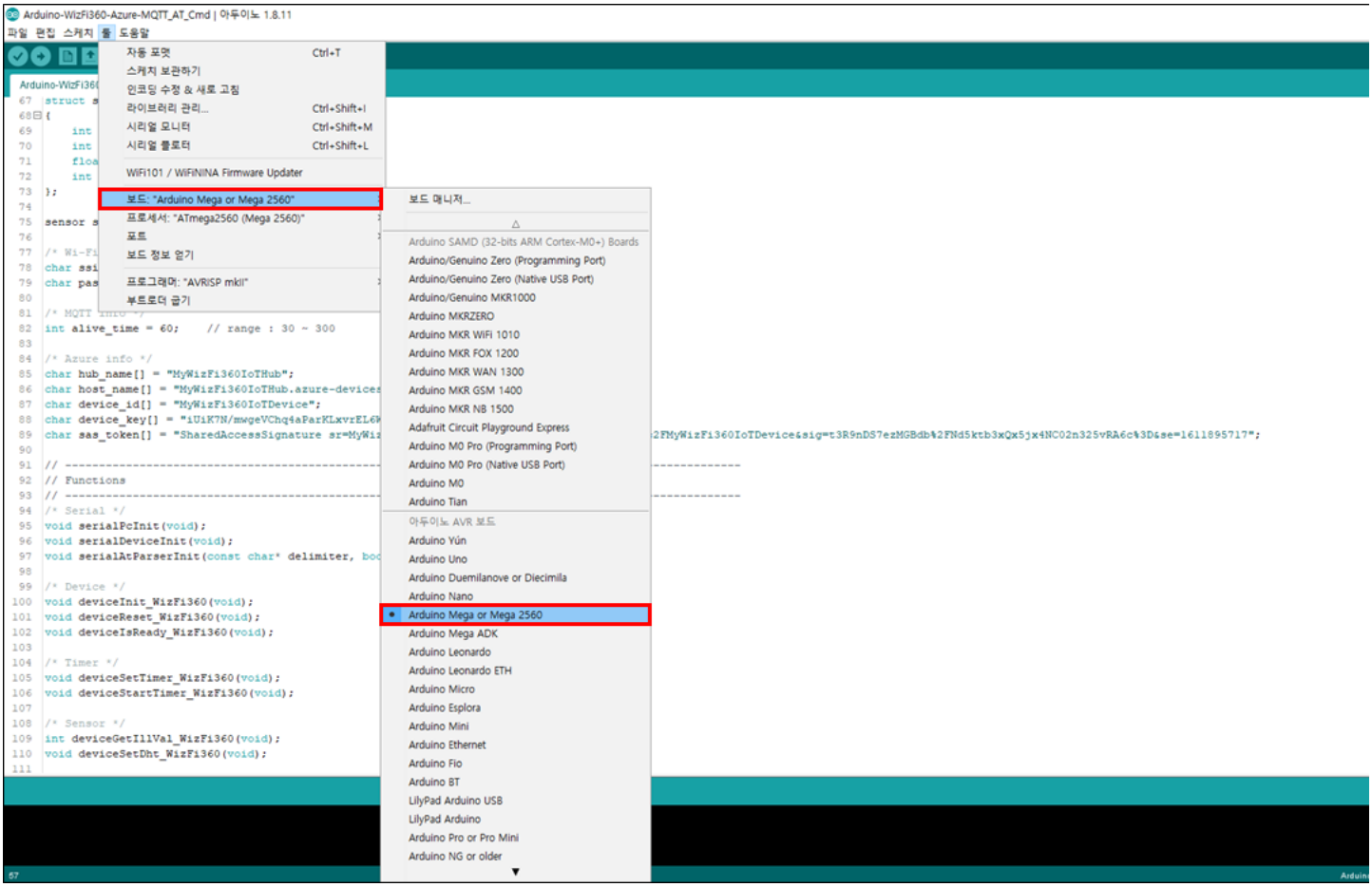
- Device Explorer를 사용하여 SAS Token 생성하기
- Azure IoT Explorer를 사용하여 SAS Token 생성하기

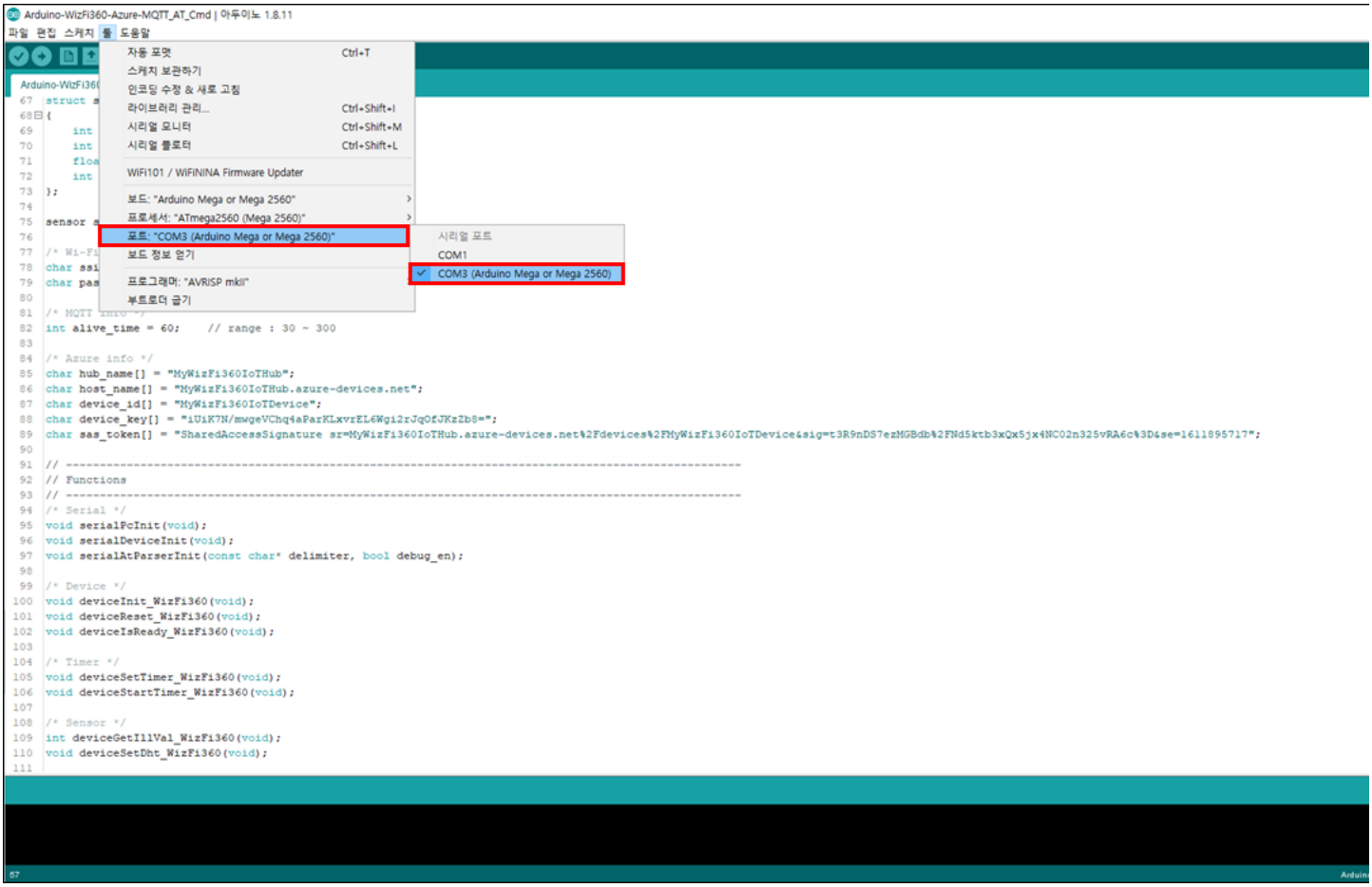
#3. Project Compile 및 Upload

사용하는 Board, COM Port 선택 후, Compile 및 Upload를 수행합니다.

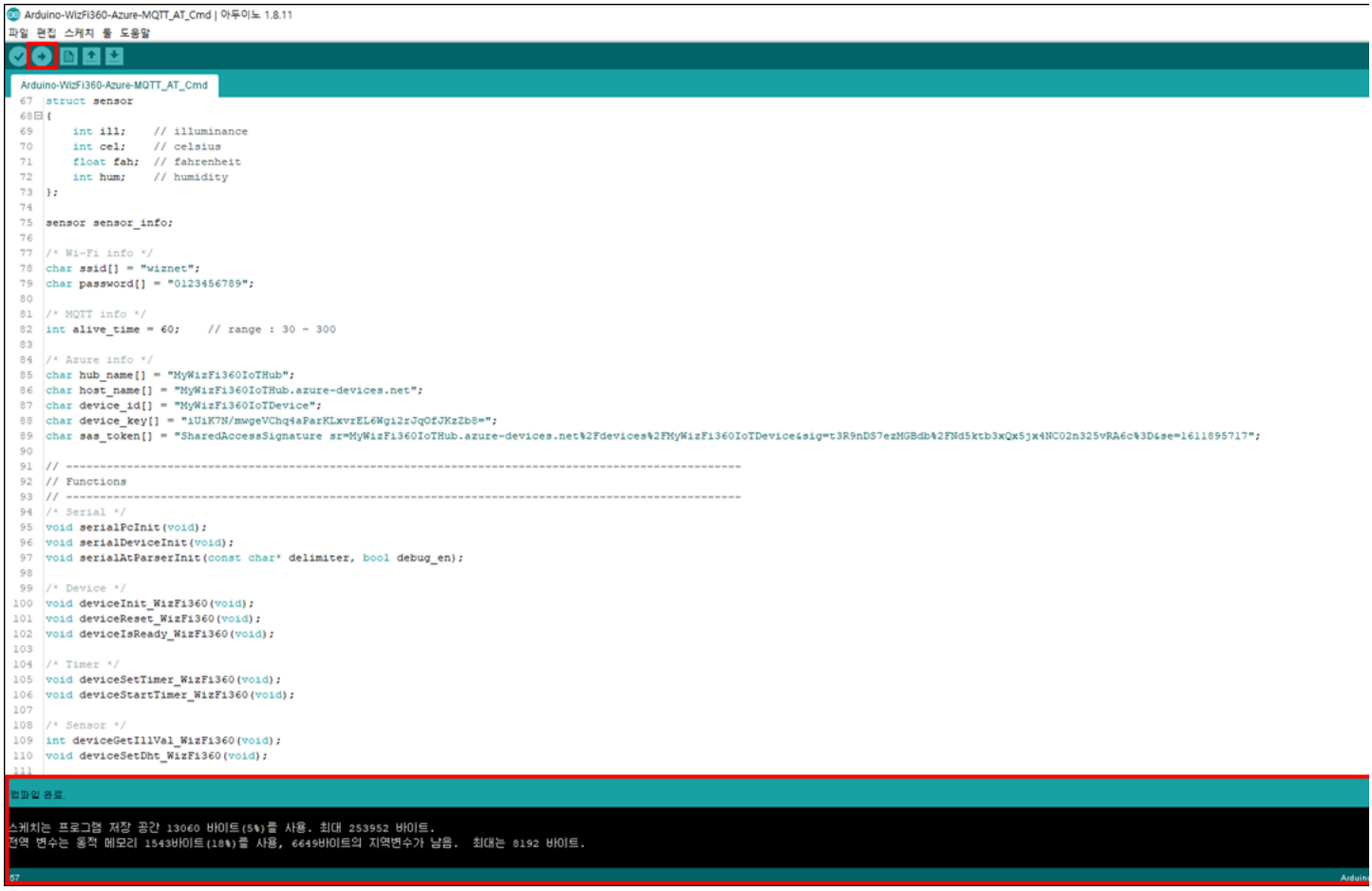
Note :

Stream Analytics 실행 중이어야 Blob Storage로 Data가 전달됩니다.



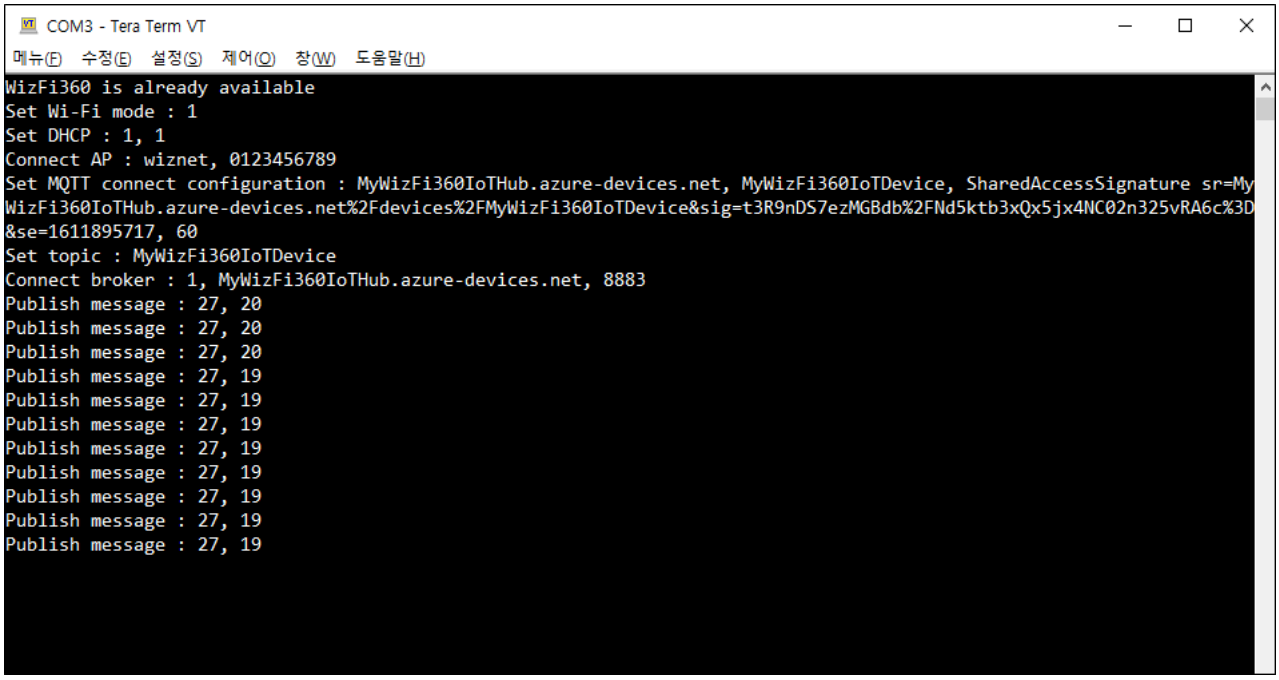


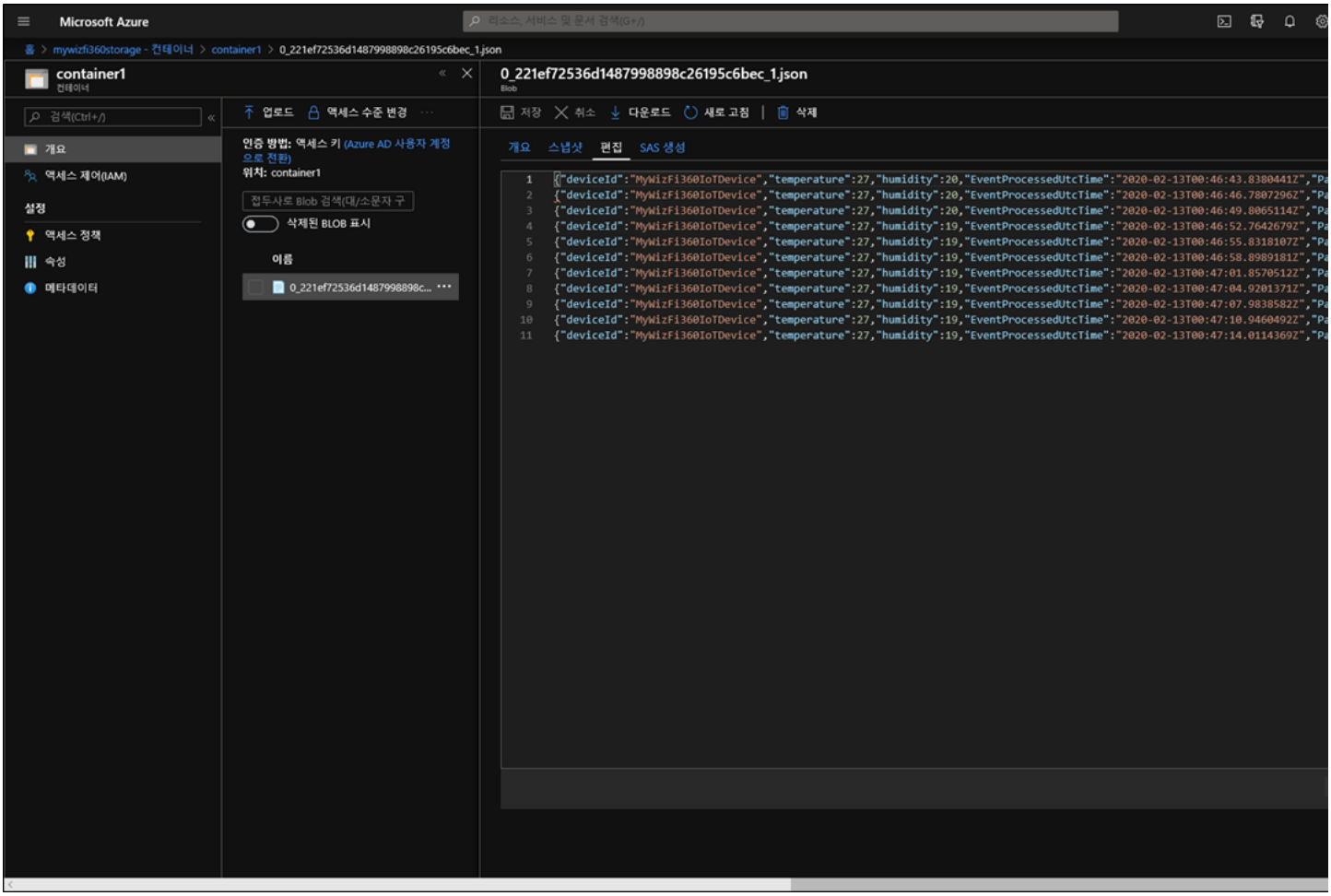

```
struct s
{
    int
    int
    floa
    int
};

sensor s

/* Wi-Fi
char ssi
char pas

/* MQTT info */
int alive_time = 60;    // range : 30 ~ 300

/* Azure info */
char hub_name[] = "MyWizFi360IoTHub";
char host_name[] = "MyWizFi360IoTHub.azure-devices.net";
char device_id[] = "MyWizFi360IoTDevice";
char device_key[] = "iUiK7N/mwgeVChq4aParKLxvrEL6Wgi2rJqOfJKzZb8=";
char sas_token[] = "SharedAccessSignature sr=MyWizFi360IoTHub.azure-devices.net%2Fdevices%2FMyWizFi360IoTDevice&sig=t3R9nDS7ezMGBdb%2FNd5ktb3xQx5jx4NC02n325vRA6c%3D&se=1611895717";

// ----------------------------------------------------------------------------------------
// Functions
// ----------------------------------------------------------------------------------------
/* Serial */
void serialPcInit(void);
void serialDeviceInit(void);
void serialAtParserInit(const char* delimiter, bool debug_en);

/* Device */
void deviceInit_WizFi360(void);
void deviceReset_WizFi360(void);
void deviceIsReady_WizFi360(void);

/* Timer */
void deviceSetTimer_WizFi360(void);
void deviceStartTimer_WizFi360(void);

/* Sensor */
int deviceGetIllVal_WizFi360(void);
void deviceSetDht_WizFi360(void);
```



#Step 4: 동작 예제 결과

Terminal Program으로 WizFi360에서 Azure IoT Hub로 전송하는 Data, Blob Storage에서 WizFi360으로부터 수신한 Data를 확인 할 수 있습니다.





#더 보기

- WizFi360 MQTT AT Command를 이용하여 Auzre IoT Hub에 연동

Connect to Azure IoT Hub using WizFi360 MQTT AT Command

#Getting started

Login to **Azure Portal**.

※ In this guide we will procedd with **free account**. To learn how to create IoT Hub please refer to Azure Cloud Service Introduction.

- [MS] Create IoT Hub using Azure Portal
- Create Blob storage using Azure Portal
- Create Stream Analytics using Azure Portal
- Setup Queries in Stream Analytics using Azure Portal
- Connect to Azure IoT Hub using WizFi360 MQTT AT Command

#Introduction

It is possible to connect to **Microsoft Azure Service** using **WizFi360**, send data to cloud and monitor current status.

Data communication will be established as following.

MQTT AT Commands will be used to connect to IoT Hub Service & send data.

Data sent to IoT Hub will be saved in Blob Storage through Stream Analytics.

In this guide we will use WizFi360 with Arduino to connect to Microsoft Azure Services using MQTT AT Command.

#Step 1: Required items

Items below are required for this guide.

#Hadrware

- Desktop or Laptop Computer
- WizFi360-EVB-Shield
- Arduino Mega 2560
- USB Type-B Cable

#Software

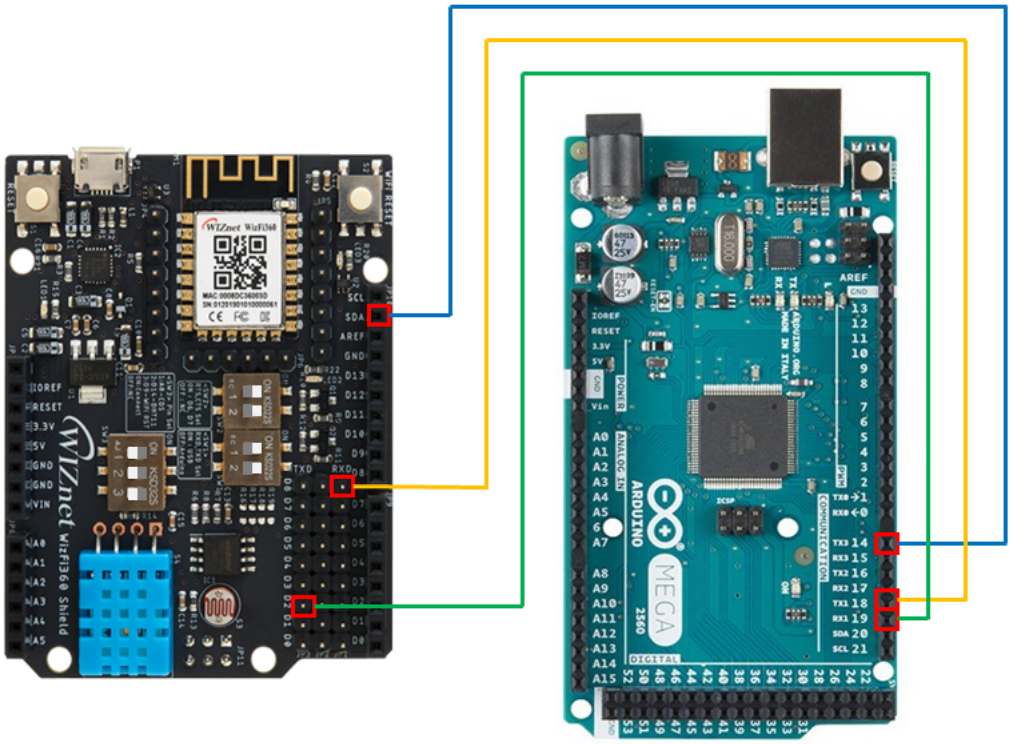
- Arduino IDE
- Preferred Serial Terminal (TeraTerm, Hercules, etc . . .)

#Step 2: Device preparation

#1. Hardware preparation

WizFi360-EVB-Shield will be installed on top of Arduino Mega 2560. Therefore DIP Switch and jumper cables shall be connected as following:

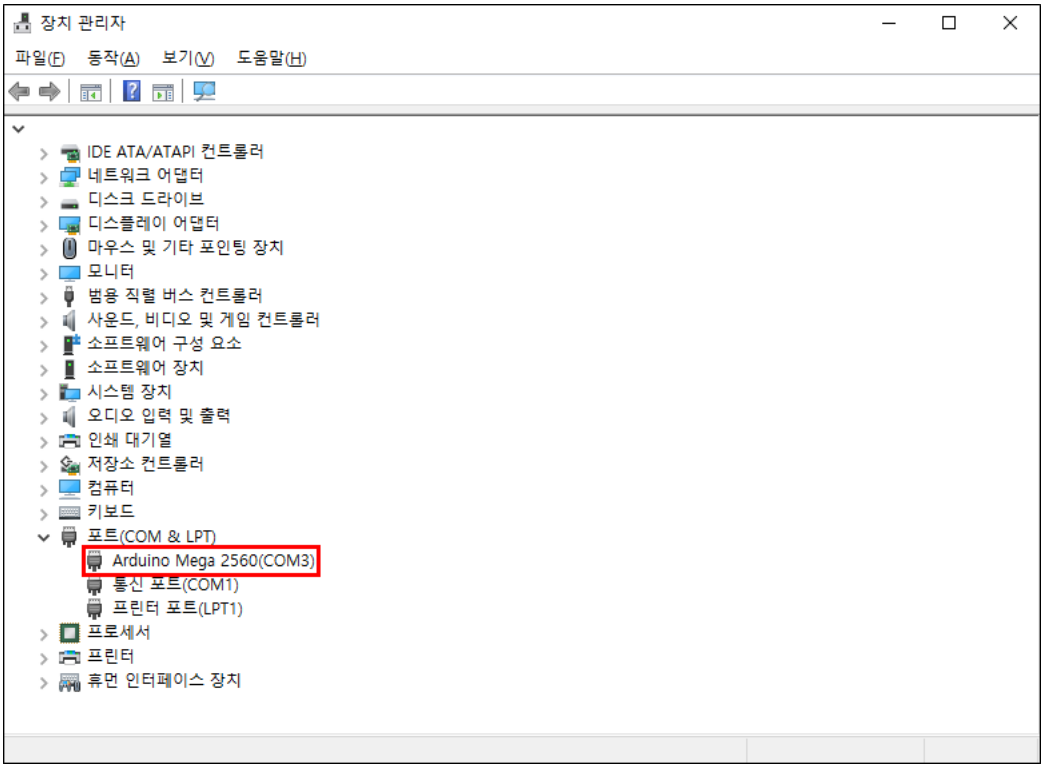
- SW1 : Off
- SW2 : Off
- SW3 : On
- WizFi360-EVB-Shield : D8 - Arduino Mega 2560 : 18
- WizFi360-EVB-Shield : D2 - Arduino Mega 2560 : 19
- WizFi360-EVB-Shield : SDA - Arduino Mega 2560 : 14



#2. Device connection

After connecting hardware, connect Arduino Mega2560 to Desktop or Laptop using USB Type-B Cable.

Check **COM Port** from **Device Manager**.



Please check port number in Arduino IDE as shown on picture above.

If COM port cannot be found in Device manager, check link below and follow instructions.

- Manually install Drivers on Windows

#Step 3: Sample application

#1. Code Download & Execution

After **Code Download** open .ino **Project**.

Sample code is stored in following path.

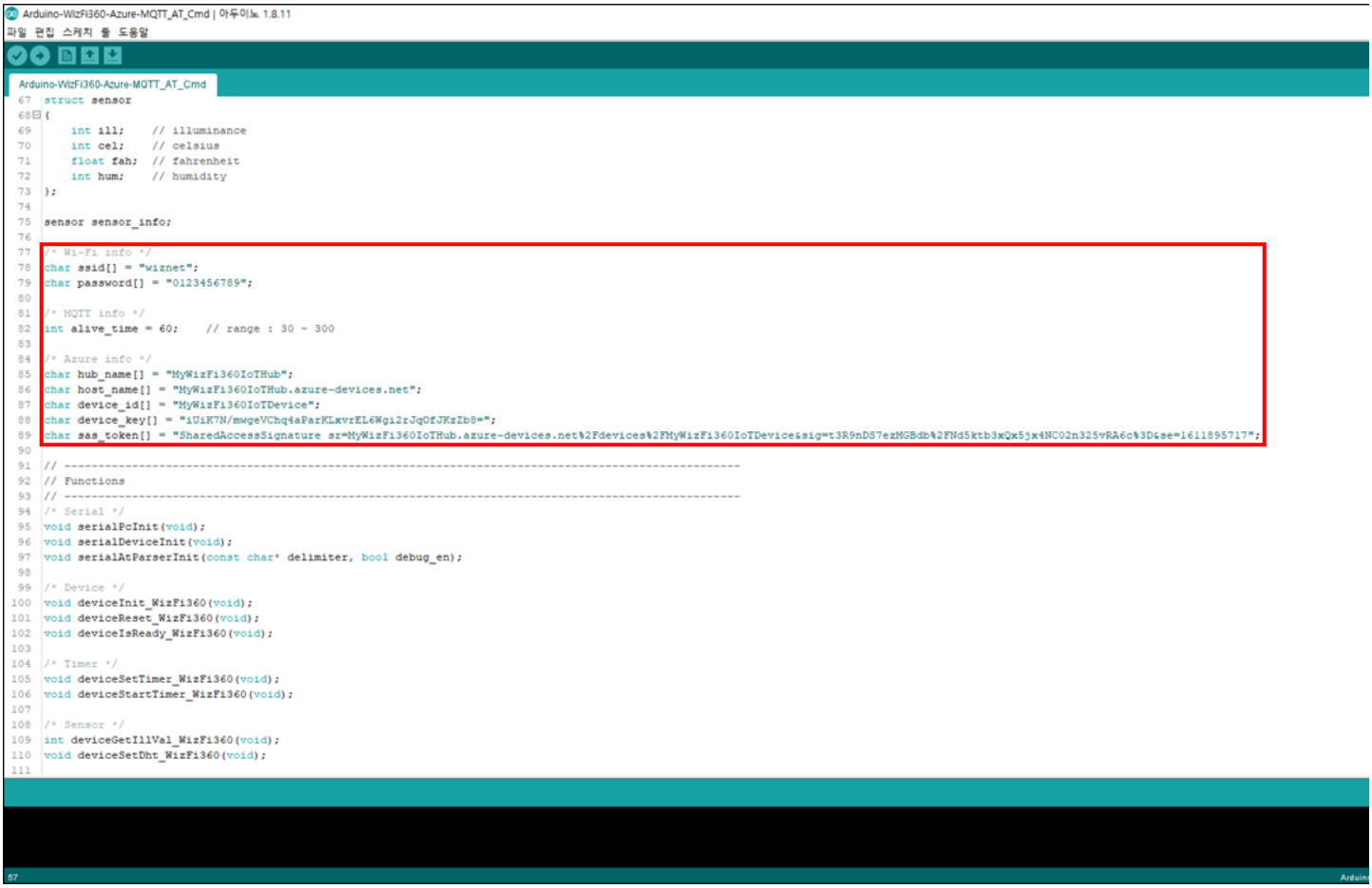
- **samples/Wi-Fi/Arduino_Mqtt_Atcmd_Wizfi360**

#2. Parameter update

To connect to Azure IoT Hub, update next parameters.

Copy

```
/* Wi-Fi info */
char ssid[] = "xxxxxxxxxx";
char password[] = "xxxxxxxxxx";

/* MQTT info */
int alive_time = xx;    // range : 30 ~ 300

/* Azure info */
char hub_name[] = "xxxxxxxxxx";
char host_name[] = "xxxxxxxxxx";
char device_id[] = "xxxxxxxxxx";
char device_key[] = "xxxxxxxxxx";
char sas_token[] = "xxxxxxxxxx";
```



For **SAS Token creation** refer to below.

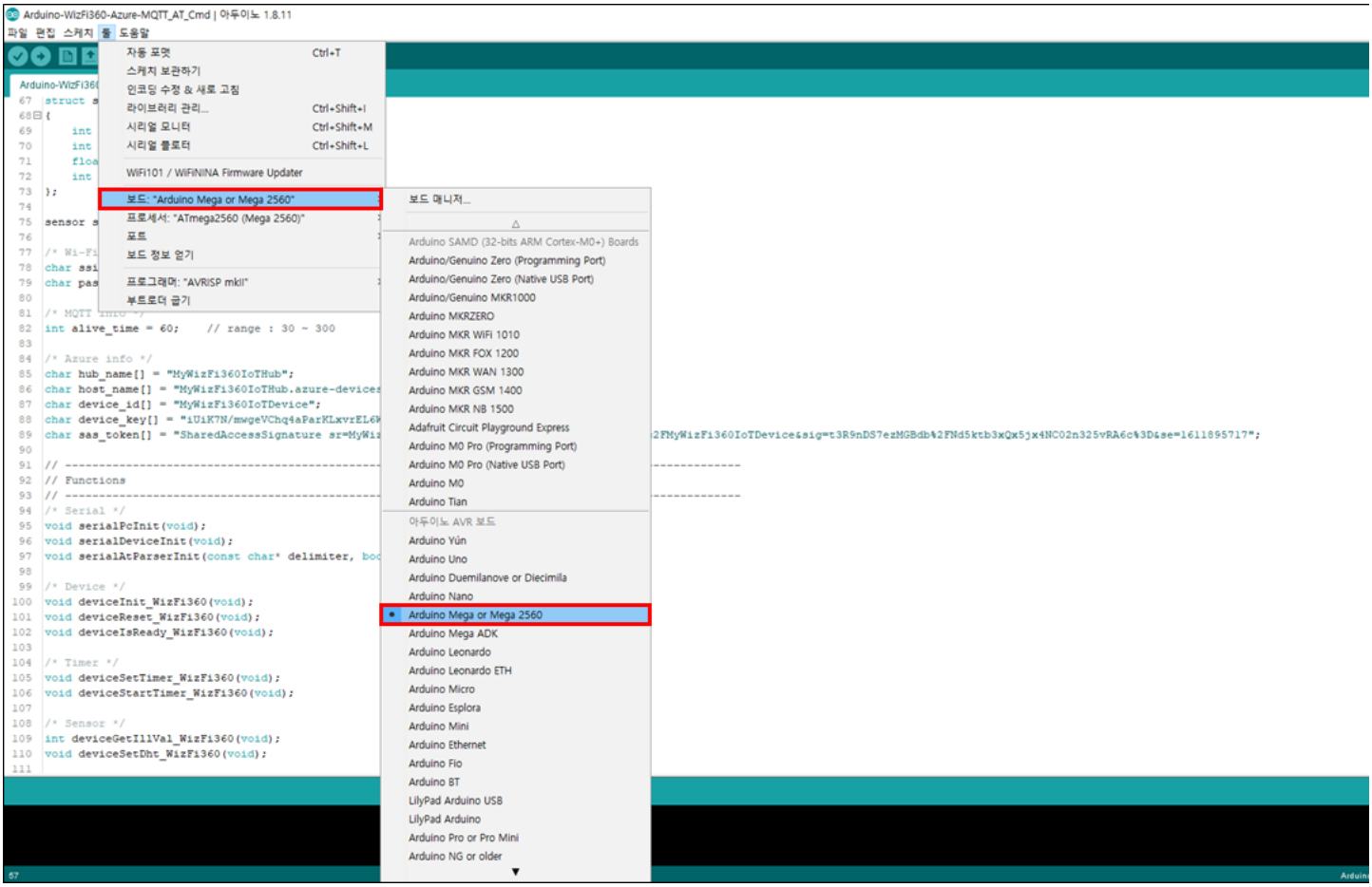
- Create SAS Token using Device Explorer
- Create SAS Token using Azure IoT Explore

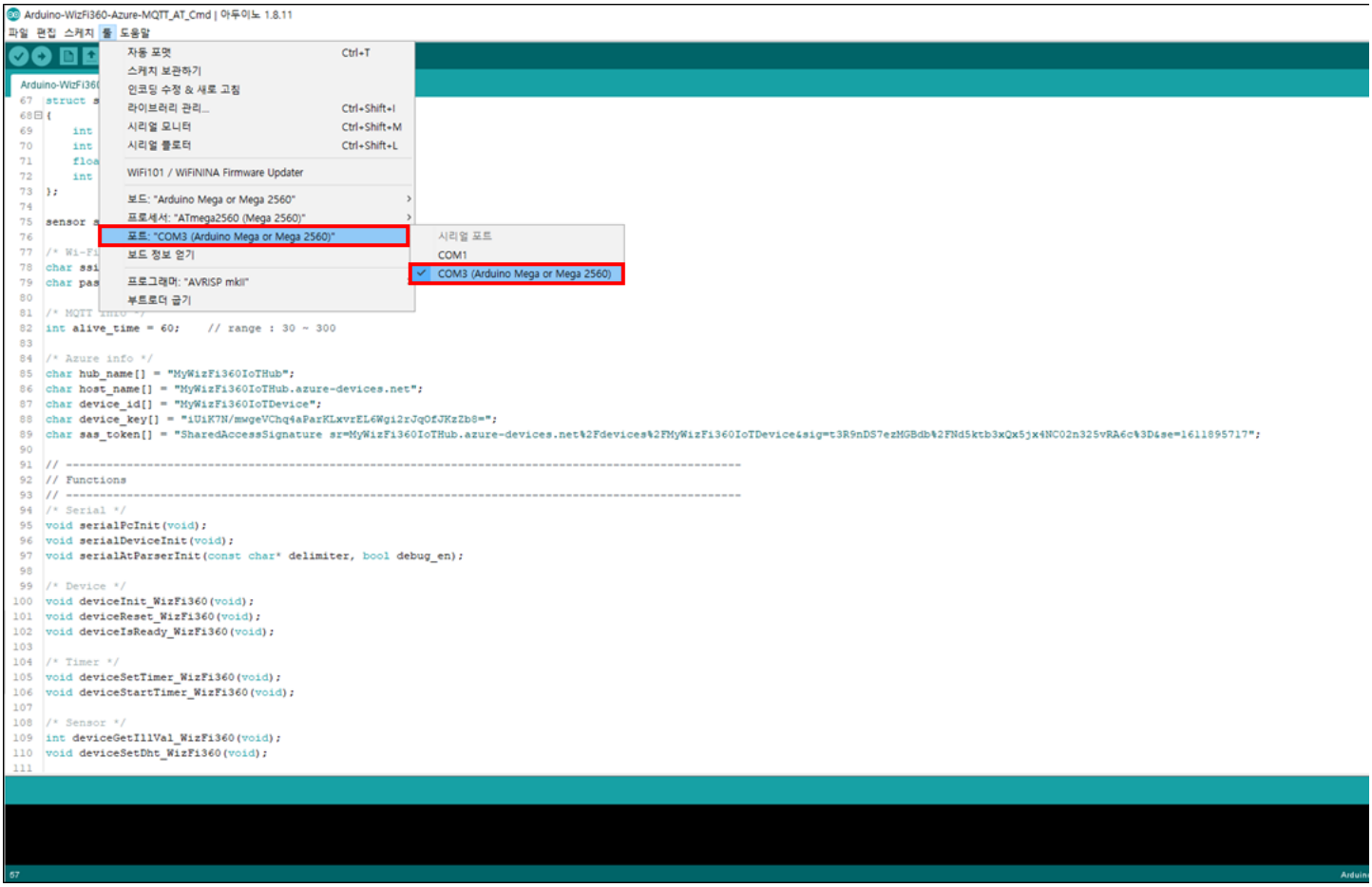
#3. Project Compile & Upload

Select corresponding Board, COM Port, then Compile & Upload.

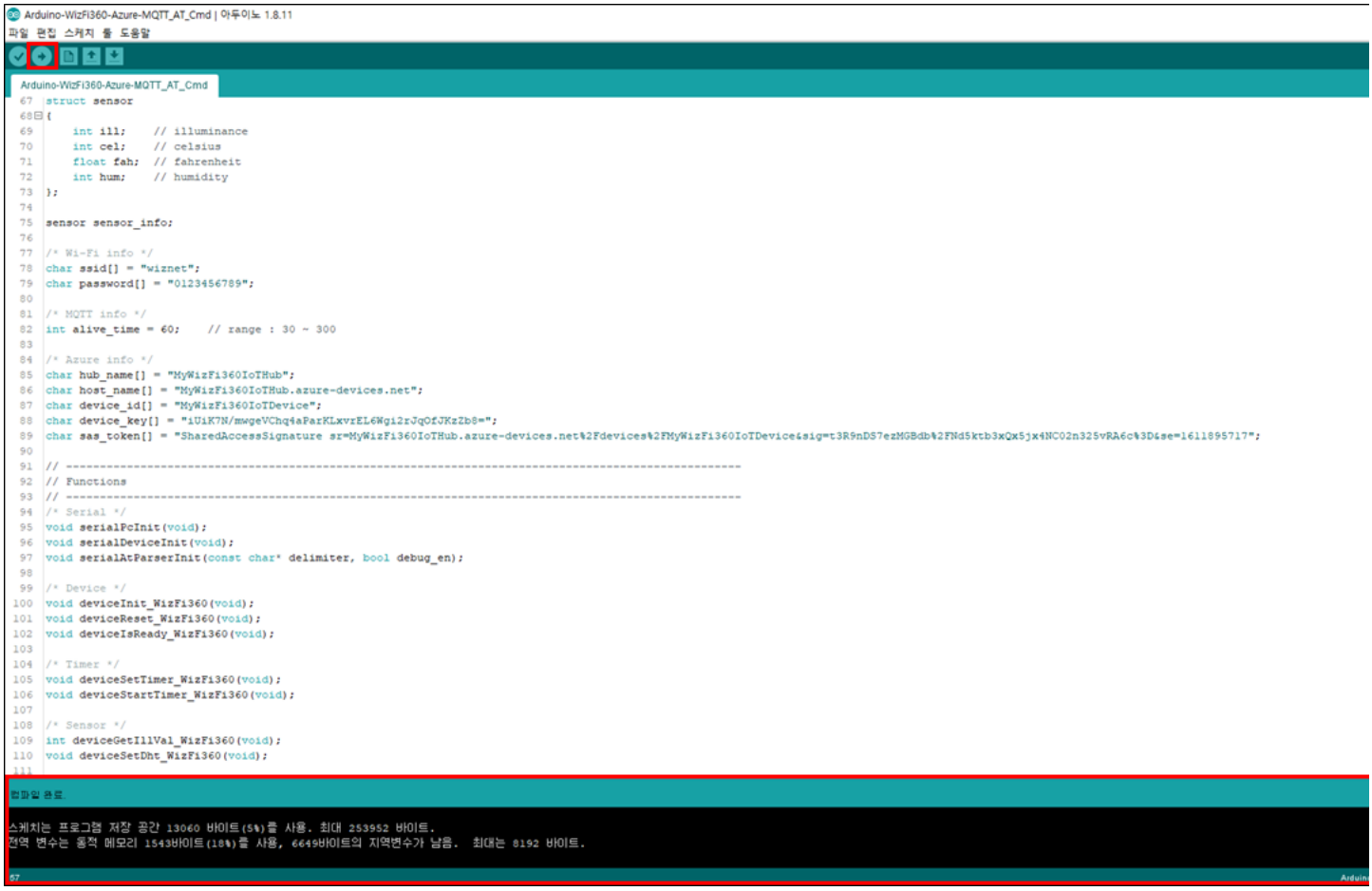
Note :

Start Stream Analytics in order to forward data to Blob Storage.



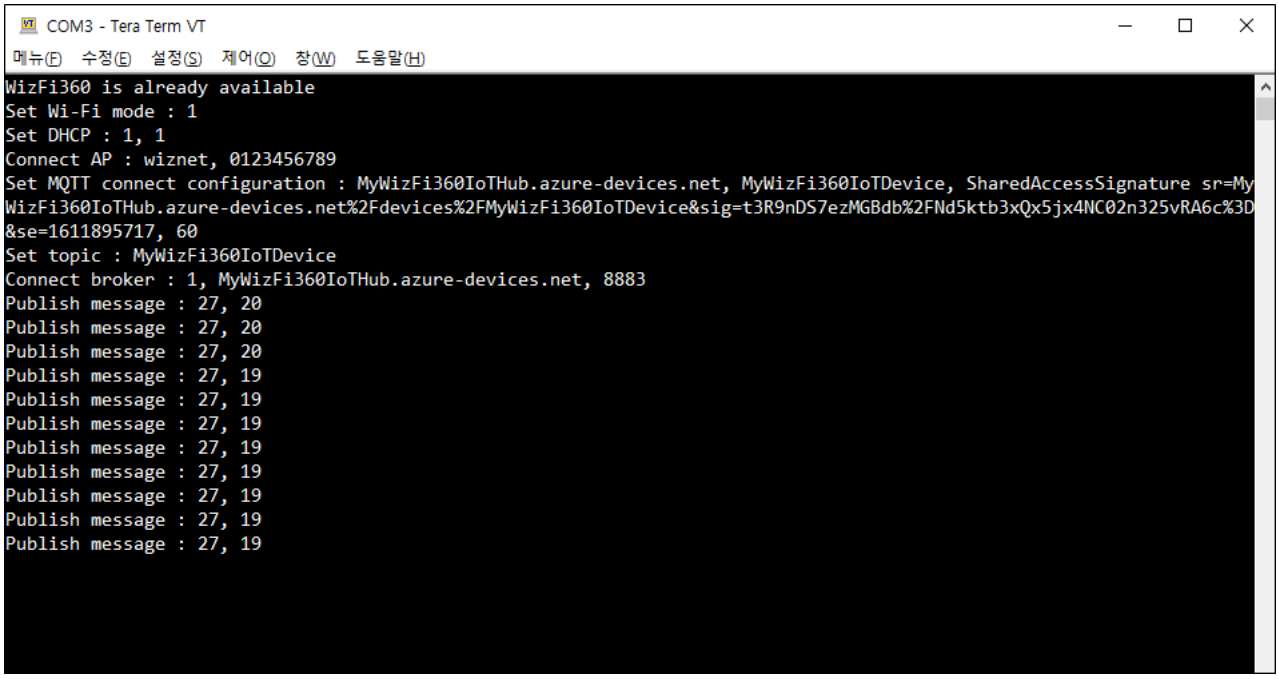


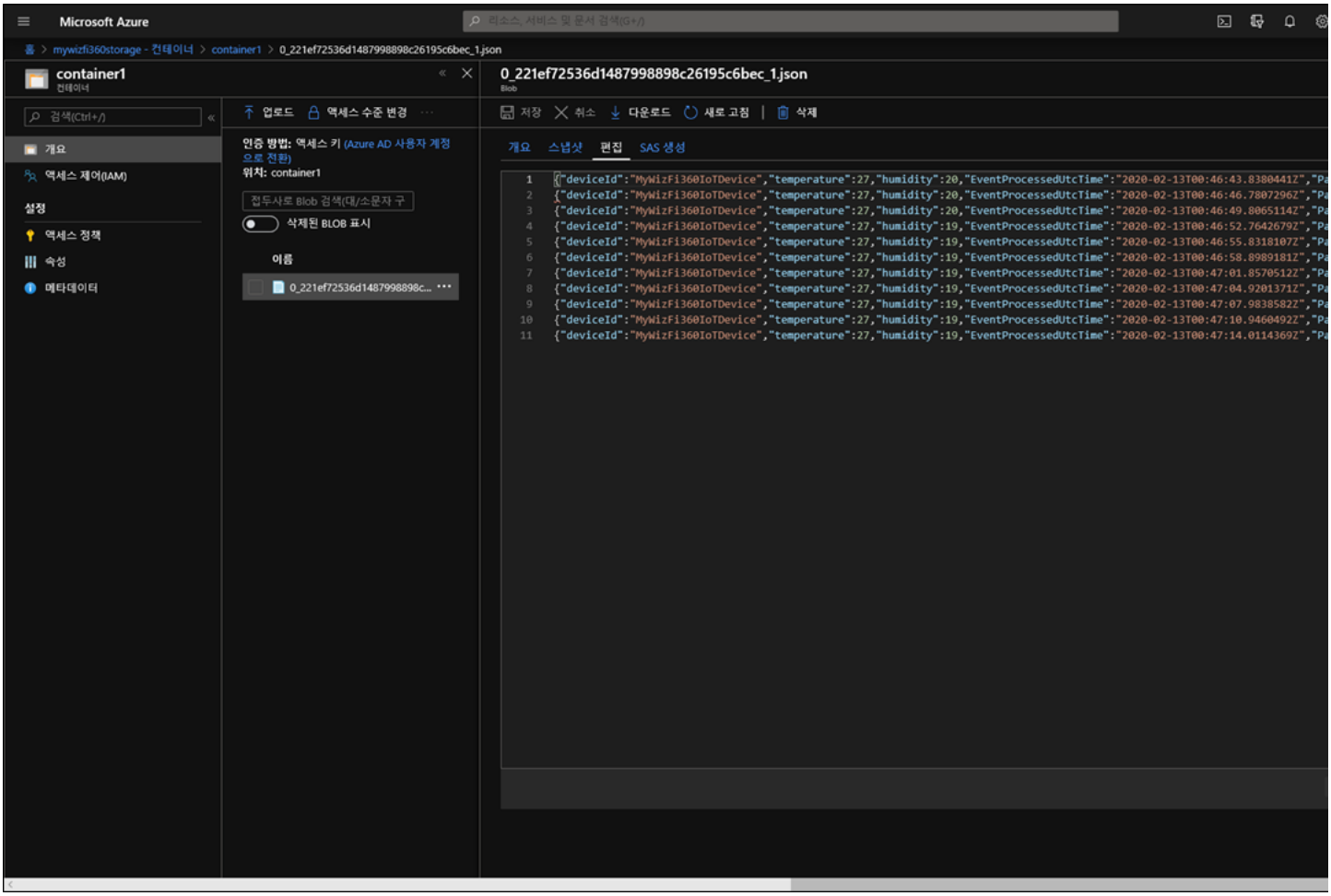

```
struct s
{
    int
    int
    floa
    int
};

sensor s

/* Wi-Fi
char ssi
char pas

/* MQTT info */
int alive_time = 60;    // range : 30 ~ 300

/* Azure info */
char hub_name[] = "MyWizFi360IoTHub";
char host_name[] = "MyWizFi360IoTHub.azure-devices.net";
char device_id[] = "MyWizFi360IoTDevice";
char device_key[] = "iUiK7N/mwgeVChq4aParKLxvrEL6Wgi2rJqOfJKzZb8=";
char sas_token[] = "SharedAccessSignature sr=MyWizFi360IoTHub.azure-devices.net%2Fdevices%2FMyWizFi360IoTDevice&sig=t3R9nDS7ezMGBdb%2FNd5ktb3xQx5jx4NC02n325vRA6c%3D&se=1611895717";

// ----------------------------------------------------------------------------------------
// Functions
// ----------------------------------------------------------------------------------------
/* Serial */
void serialPcInit(void);
void serialDeviceInit(void);
void serialAtParserInit(const char* delimiter, bool debug_en);

/* Device */
void deviceInit_WizFi360(void);
void deviceReset_WizFi360(void);
void deviceIsReady_WizFi360(void);

/* Timer */
void deviceSetTimer_WizFi360(void);
void deviceStartTimer_WizFi360(void);

/* Sensor */
int deviceGetIllVal_WizFi360(void);
void deviceSetDht_WizFi360(void);
```



#Step 4: Results

In terminal program we can check data sent from WizFi360 to Azure IoT Hub. In Blob Storage we can check received data.





#References

- Connect to Azure IoT Hub using WizFi360 MQTT AT Command

Connect WizFi360 to AWS IoT using AT Command

#Getting Started

#Hardware Requirement

- Desktop or laptop computer
- MicroUSB cable
- WizFi360-EVB-Shield

#Software Requirement

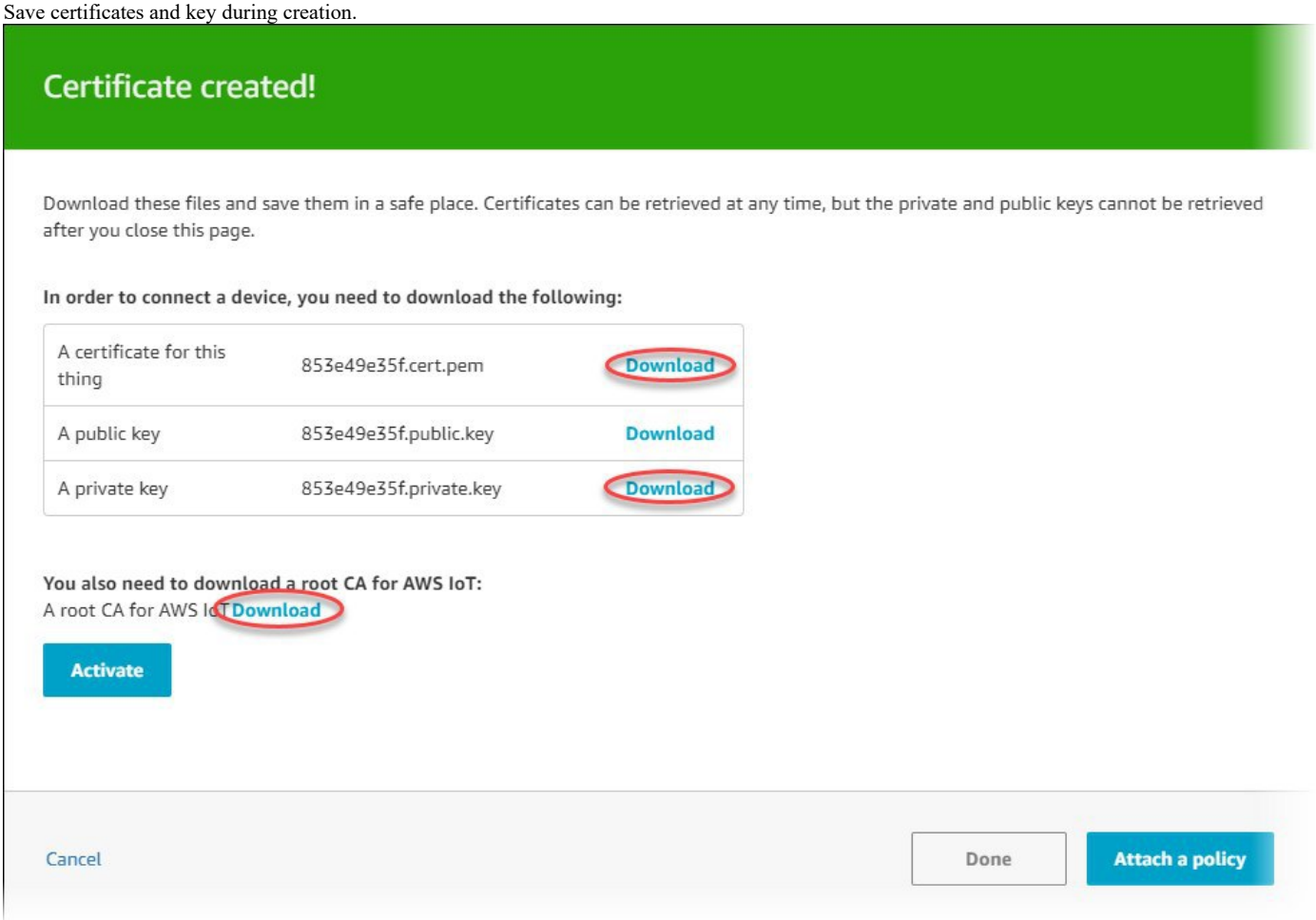
- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- AWS Console account
- Firmware v. 1.1.0.595 was used in this guide. Please contact us to receive this version.

▼ AWS IoT Core preparation before start

- **Sign to console**
- **Create a thing**
- **Create certificate**

  important

  Save certificates and key during creation.



  AWS root CA can be downloaded from **here**

- **Create a policy**
- **Attach policy to certificate**
- **Attach certificate to a thing**

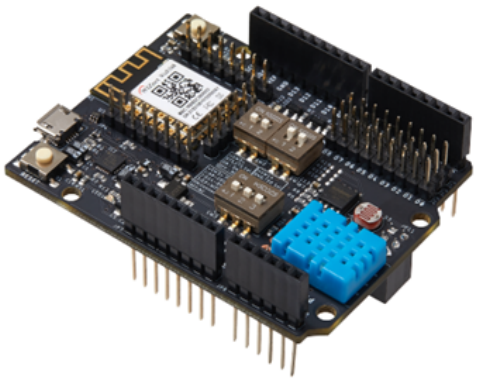
#Introduction

AWS IoT provides secure, bi-directional communication between Internet-connected devices such as sensors, actuators, embedded micro-controllers, or smart appliances and the AWS Cloud. It is possible to connect to AWS via WizFi360 and send data using MQTT.

In this document we will provide guide how to connect to AWS services. Process consists of following steps:

- Creation of AWS account
- Creation & configuration of Thing in IoT Core
- Connection & Message transfer

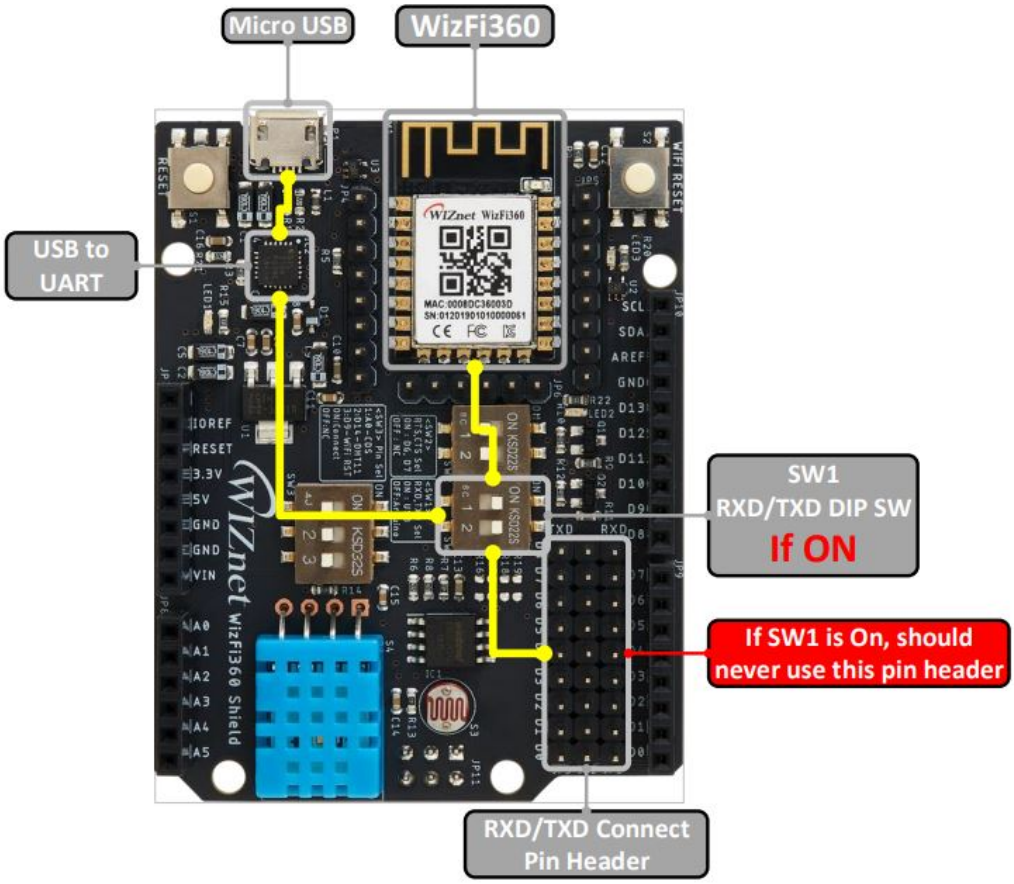
For this guide we used evaluation board WizFi360-EVB-Shield



#Device preparation

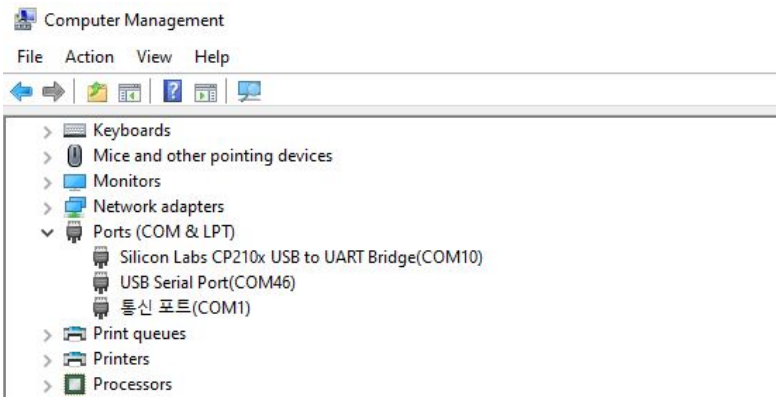
#Hardware setting

We are going to use WizFi360-EVB-Shield in standalone mode. MicroUSB cable will be used to connect through UART. Switch SW1 into ON position and connect MicroUSB.



#Device connection

Please check COM port number in Device Manager.



tip

If COM port cannot be found in Device Manager, please install drivers below.

- Silicon Labs CP210x USB to UART Driver

#AT Commands Description

Please refer to AT Instruction Set to find information about all AT Commands. Below we will describe commands created for AWS connection.

#1. Set SSL Certificate

AT Command: AT+CASEND

Syntax:

Type	Command	Response
Set	AT+CASEND=<parameter>	OK

Defined values:

Parameter	Value
<parameter>	0: delete certificate 1: generate certificate

In order to check current certificate enter command **AT+CASEND?**

#2. Set Private Key

AT Command: AT+AWSPKSEND

Syntax:

Type	Command	Response
Set	AT+AWSPKSEND=<parameter>	OK

Defined values:

Parameter	Value
<parameter>	0: delete private key 1: save new private key

In order to check current key enter command **AT+AWSPKSEND?**

#3. Set Certificate for Thing

AT Command: AT+CLICASEND

Syntax:

Type	Command	Response
Set	AT+CLICASEND=<parameter>	OK

Defined values:

Parameter	Value
<parameter>	0: delete Certificate 1: save new certificate for Thing

In order to check current certificate enter command **AT+CLICASEND?**

#4. Connect to AWS

AT Command: AT+AWSCON

Syntax:

Type	Command	Response
Set	AT+AWSCON="<Thing ARN>"	CONNECT OK

important

Before connection to AWS certificates, MQTTTOPIC and MQTTSET shall be set.
Otherwise AT+AWSCON will return error.

#Connection procedure

#Connect your device and launch terminal

For connection use following configuration in terminal: 115200-8-N-1, None.

#Connect WizFi360 to WiFi

Copy

```
// Set module to station mode
AT+CWMODE_CUR=1

// Get AP list
AT+CWLAP

// Connect to AP
AT+CWJAP_CUR="ssid","password"

// Query WizFi360 IP address
AT+CIPSTA_CUR?
```

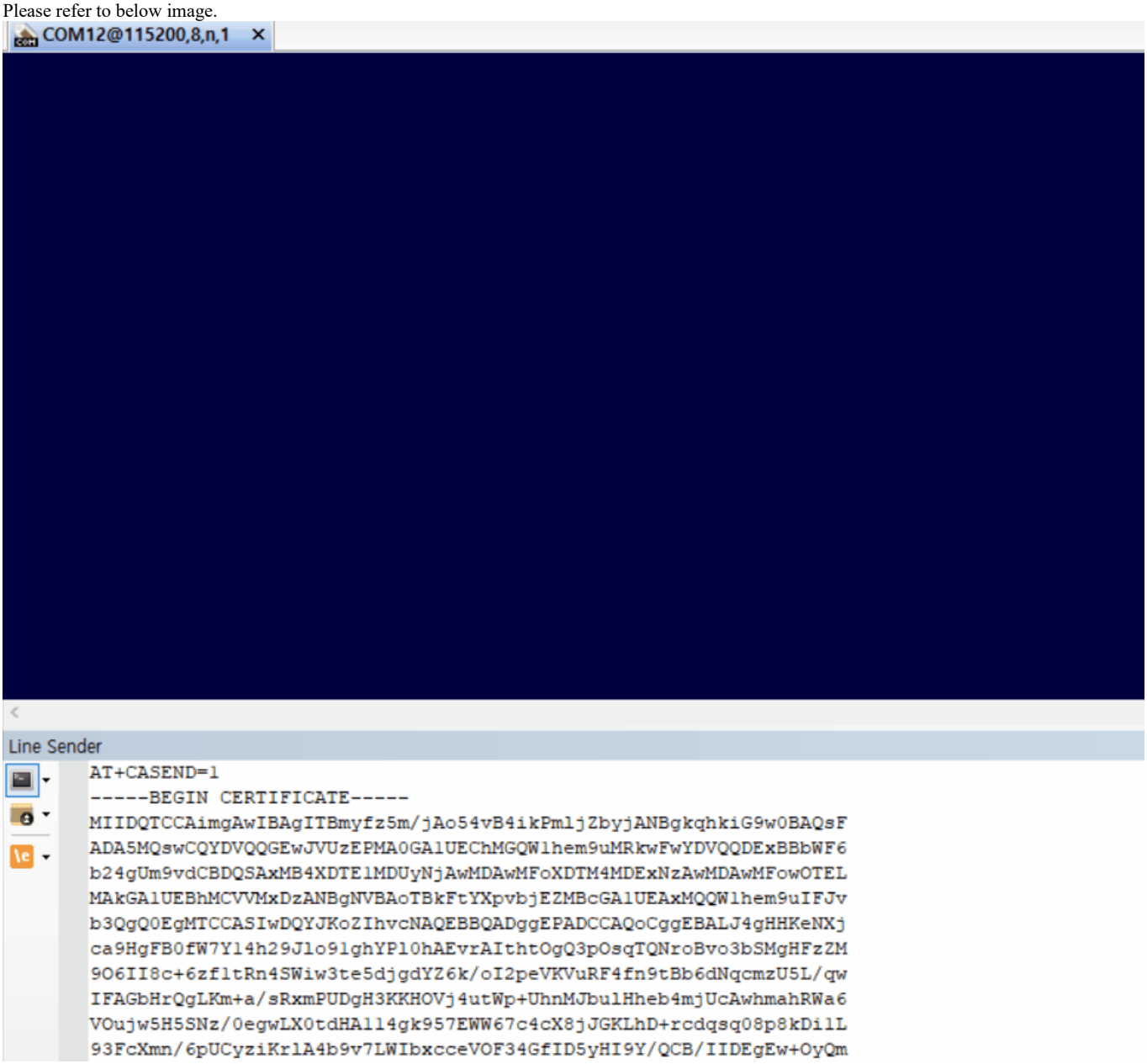
#Enter Certificate

Copy

```
//Enter AWS Root CA
AT+CASEND=1

//Enter Private key
AT+AWSPKSEND=1

//Enter Client ceritificate
AT+CLICASEND=1
```

When saving certificate or private key, all lines shall be sent one by one from "Begin certificate" line till "End certificate" line.

Please refer to below image.



#Connection to AWS

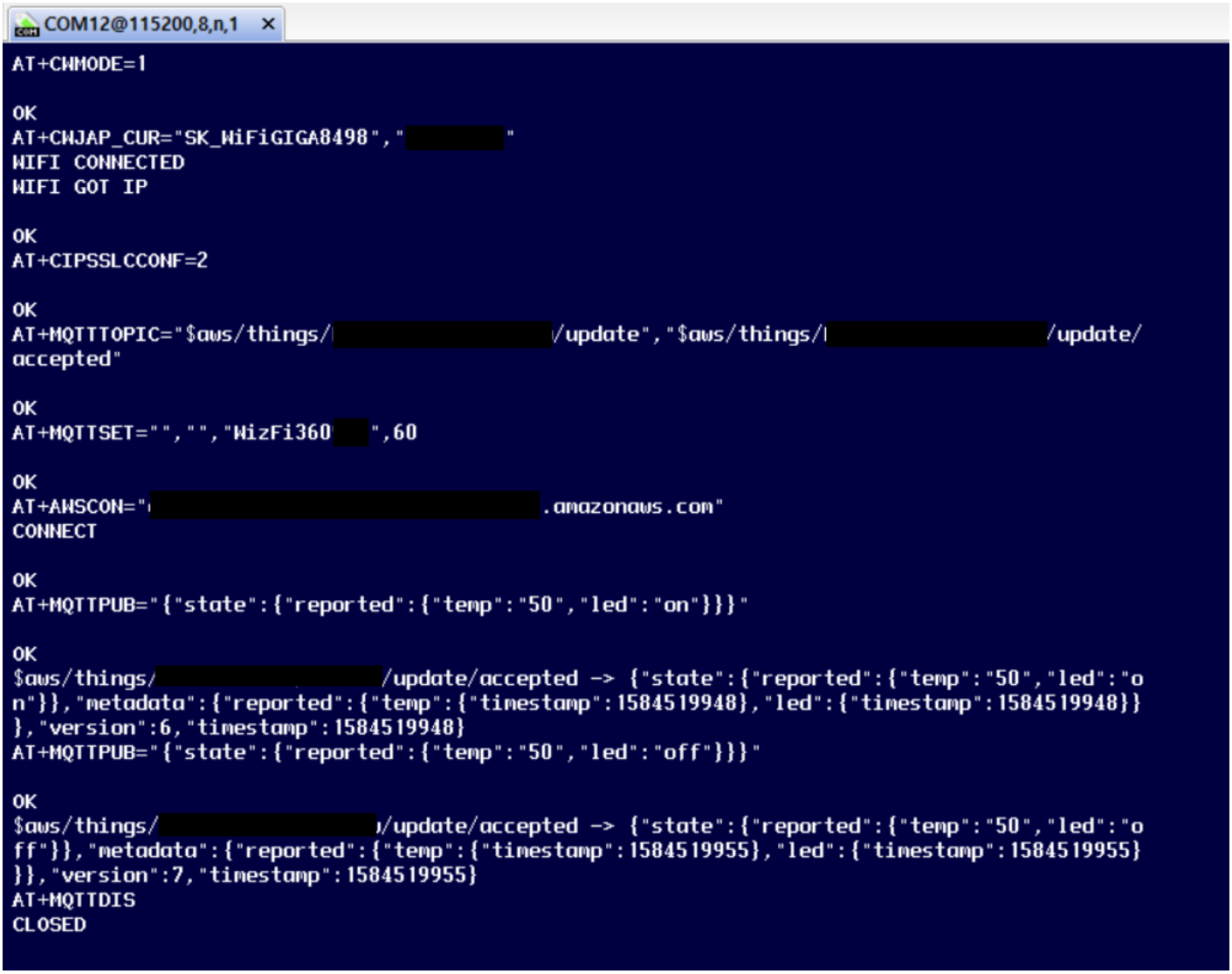
Copy

```
// Shadow update and accepted links  should be copied from "Interact" menu in AWS Console
```

```
AT+MQTTTOPIC="shadow_update_link","shadow_accepted_link"

// User & password can be empty, thing name shall be entered
AT+MQTTSET="","","thing_name",60

// Type your Rest API endpoint
AT+AWSCON="REST_API_endpoint"

// Publish message
AT+MQTTPUB="{"state":{"reported":{"temp":"40","led":"on"}}}"

// Disconnect from a broker
AT+MQTTDIS
```

▶ Can't find shadow links & Rest API endpoint? (Click here)

Below is screenshot from terminal



#Results

1. Results can be checked in AWS -> AWS IoT -> Manage -> Things -> Shadow.
2. Since we subscribed to "updated" topic, when MQTT message is sent we can see reply message instantly.

Shadow Document

Delete Edit

Last update: Mar 18, 2020 5:25:55 PM +0900

Shadow state:


```
{
  "reported": {
    "temp": "50",
    "led": "off"
  }
}
```


Metadata:


```
{
  "metadata": {
    "reported": {
      "temp": {
        "timestamp": 1584519955
      },
      "led": {
        "timestamp": 1584519955
      }
    }
  },
  "timestamp": 1584520988,
  "version": 7
}
```


Congratulations

WizFi360 is successfully connected to AWS!

WizFi360 AT Command를 이용하여 AWS IoT에 연결

#시작하기 전에

#Hardware Requirement

- Desktop or laptop computer
- MicroUSB 케이블
- WizFi360-EVB-Shield

#Software Requirement

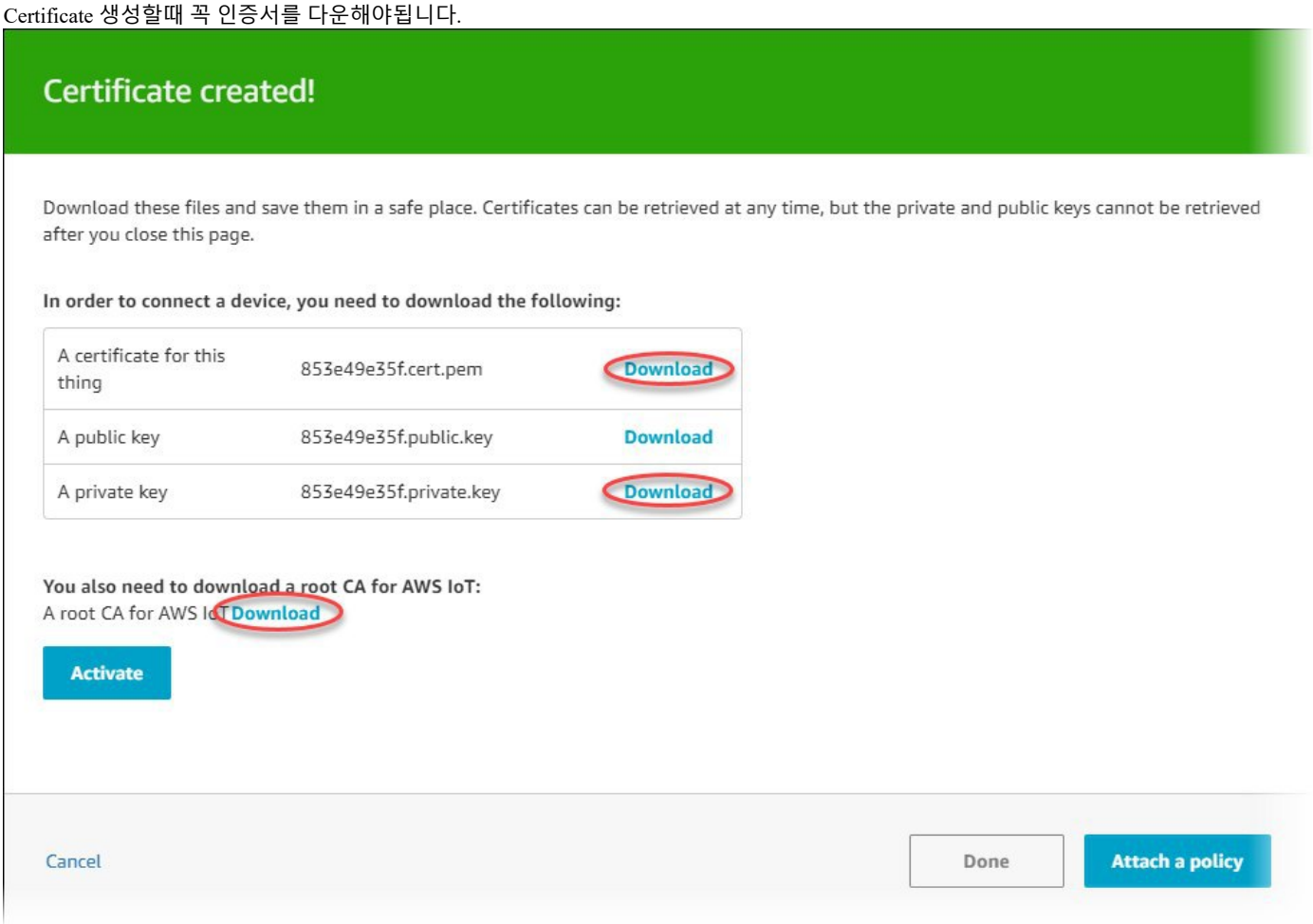
- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- AWS Console account
- 본 문서에서는 Firmware v.1.1.0.595사용되었습니다. Firmware 받으실려면 문의해주시기 바랍니다.

▼ 시작하기전에 AWS IoT Core 준비할것

- **AWS Console에 로그인**
- **사물 생성**
- **인증서 생성**

important

Certificate 생성할때 꼭 인증서를 다운해야됩니다.



AWS root CA **여기서** 다운로드 할 수 있습니다.

- **정책 생성**
- **디바이스 인증서에 정책 연결**
- **사물에 인증서 연결**

#소개

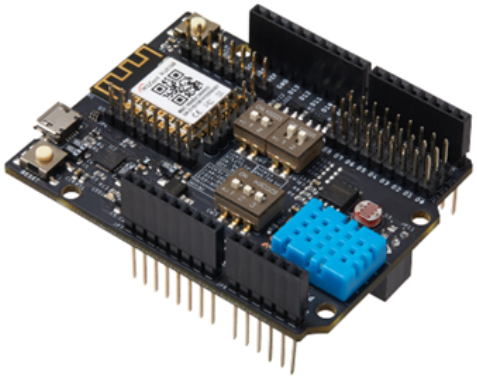
AWS IoT은(는) 인터넷 연결 제품(센서, 액추에이터, 내장형 마이크로 컨트롤러, 스마트 애플리케이션 등)과 AWS 클라우드 간에 안전한 양방향 통신을 제공합니다. AWS의 서비스에 WizFi360 을 연동하여 데이터를 클라우드로 전송하고, 모니터링 할 수 있습니다.

본 문서에서는 WizFi360 이용하여 AWS Services에 연결 방법에 대한 가이드를 제공합니다. 이 프로세스는 다음 단계로 구성됩니다.

- AWS Account 준비
- IoT 디바이스 등록

- AWS IoT와 연결 및 데이터 통신

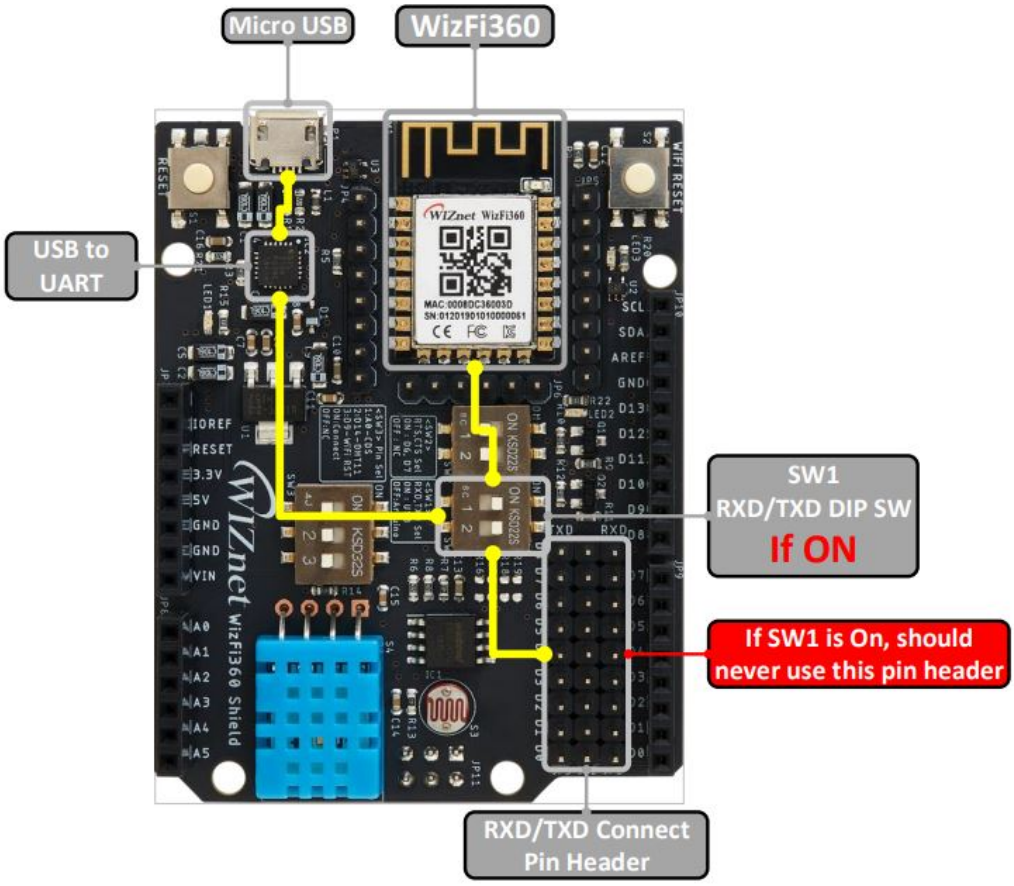
WiFi모듈 테스트를 위해 WizFi360-EVB-Shield Evaluation 보드를 사용되었습니다.



#디바이스 준비

#하드웨어 설정

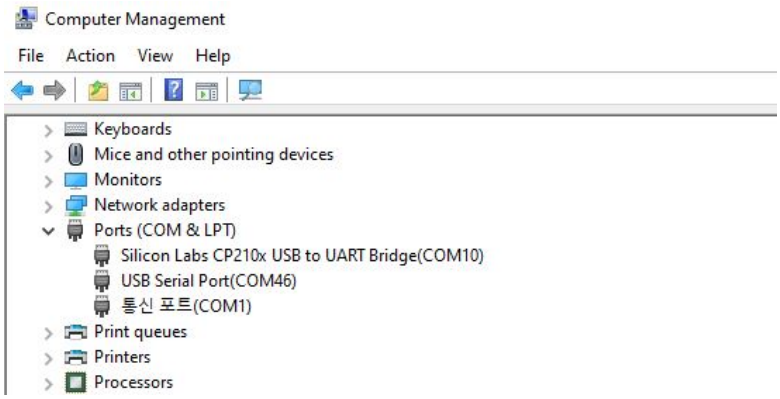
본 문서에서 WizFi360-EVB-Shield가 standalone mode에서 사용됩니다. 따라서 UART를 위해 MicroUSB를 사용할겁니다. MicroUSB 사용하는경우 SW1을 ON 시키고 MicroUSB 연결해야됩니다.



#디바이스 연결

하드웨어 설정 후 MicroUSB 이용하여 PC와 연결합니다. PC운영체제에서 보드와 연결된 COM 포트를 확인할 수 있습니다.

윈도우 운영체제의 경우, 장치 관리자(Device Manager)에서 COM 포트를 확인할 수 있습니다.



tip

장치 관리자에서 COM 포트를 확인할 수 없는 경우, 다음 링크에서 드라이버를 다운로드하여 설치하시기 바랍니다.

- Silicon Labs CP210x USB to UART Driver

#AT 명령어

WizFi360의 조금 더 상세한 **사용 방법**은 **Quick Start Guide**, **AT Command**는 **AT Instruction Set**을 참고 바랍니다.

#1. Set SSL Certificate

AT Command: AT+CASEND

Syntax:

Type	Command	Response
Set	AT+CASEND=<parameter>	OK

Defined values:

Parameter	Value
<parameter>	0: delete certificate 1: generate certificate

Certificate 확인하려면 **AT+CASEND?** 입력합니다.

#2. Set Private Key

AT Command: AT+AWSPKSEND

Syntax:

Type	Command	Response
Set	AT+AWSPKSEND=<parameter>	OK

Defined values:

Parameter	Value
<parameter>	0: delete private key 1: save new private key

Private Key 확인하려면 **AT+AWSPKSEND?** 입력합니다.

#3. Set Certificate for Thing

AT Command: AT+CLICASEND

Syntax:

Type	Command	Response
Set	AT+CLICASEND=<parameter>	OK

Defined values:

Parameter	Value
<parameter>	0: delete Certificate 1: save new certificate for Thing

Certificate 확인하려면 **AT+CLICASEND?** 입력합니다.

#4. Connect to AWS

AT Command: AT+AWSCON

Syntax:

Type	Command	Response
Set	AT+AWSCON="<Thing ARN>"	CONNECT OK

important

AWS에 접속하기 전에 certificates, MQTTTOPIC and MQTTSET 세팅 해야 됩니다.
그렇지 않으면 AT+AWSCON에서 오류가 반환됩니다.

#동작 예제

#모듈 연결 및 터미널 실행

터미널에서 다음 setting 해서 WizFi360에 연결합니다: 115200-8-N-1, None.

#WizFi360 WiFi설정

Copy

```
//  Mode 설정
AT+CWMODE_CUR=1

// AP list
AT+CWLAP

// AP 접속
AT+CWJAP_CUR="ssid","password"

// Query WizFi360 IP address
AT+CIPSTA_CUR?
```

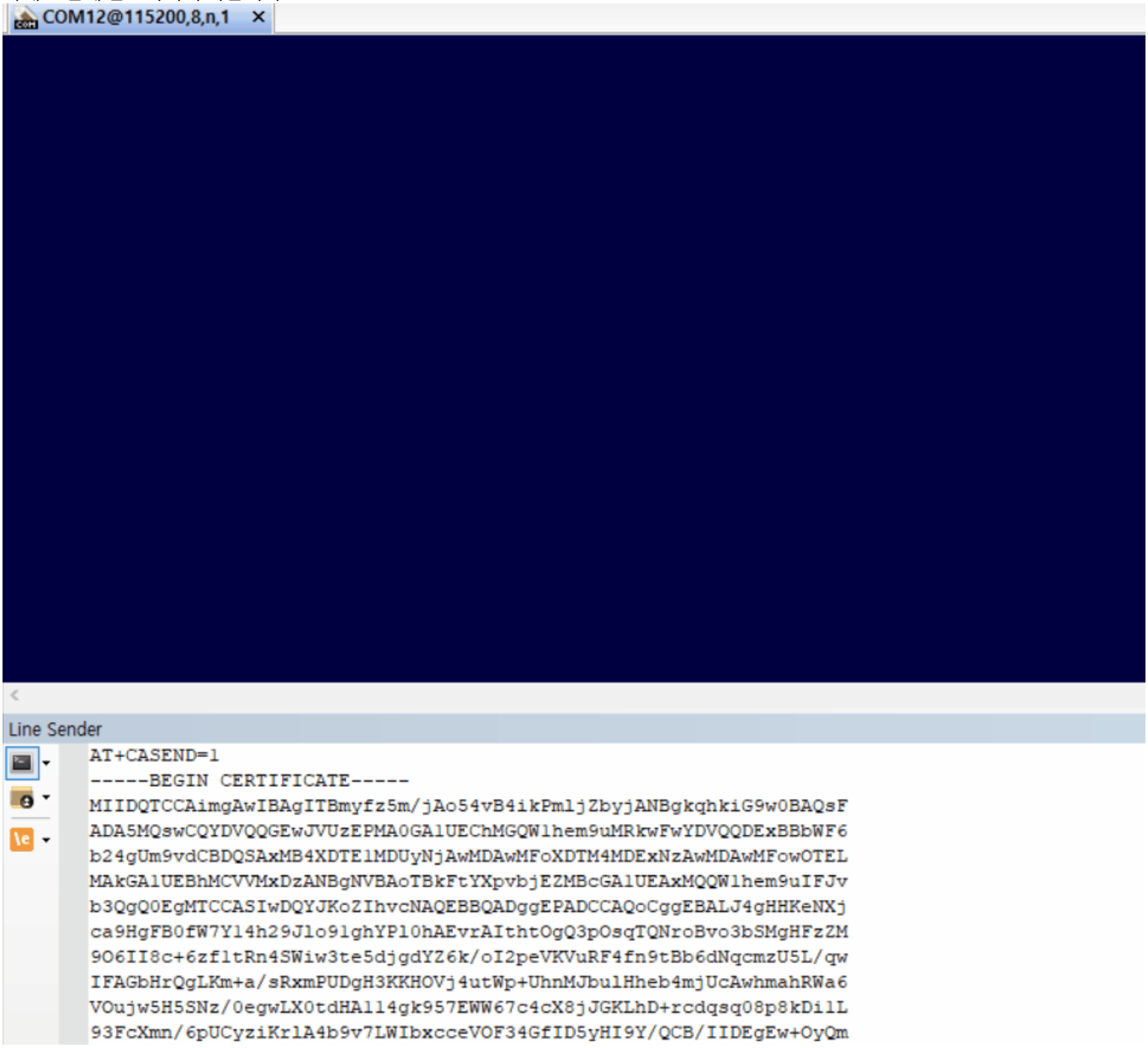
#인증서 저장

Copy

```
//AWS Root CA 저장
AT+CASEND=1

//Private key 저장
AT+AWSPKSEND=1

//Client ceritificate 저장
AT+CLICASEND=1
```

인증서 저장할때 Begin certificate 라인부터 End certificate 라인까지 복사해서 라인 하나씩 입력해야됩니다.

아래 그림에 참조하시기 바랍니다.



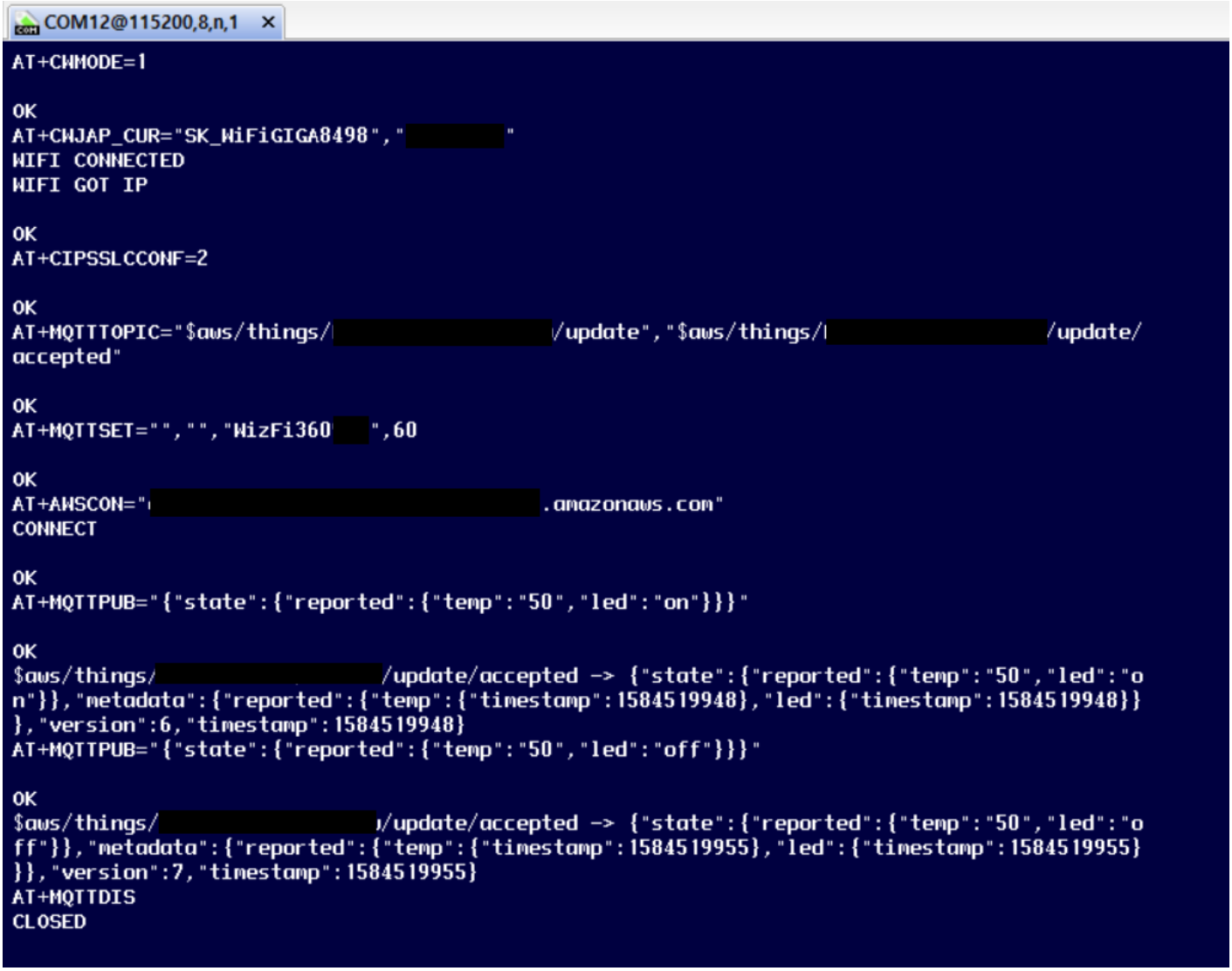
#AWS에 연결

Copy

```
// 사물 섀도우 업데이트 및 업데이트 승인 "상호 작용" 메뉴에서 복사합니다
AT+MQTTTOPIC="shadow_update_link","shadow_accepted_link"

// Username 및 password 입력 안해도 됩니다. ClientID는 사물이름입니다
AT+MQTTSET="","","thing_name",60

// Rest API 엔드포인트를 입력합니다
AT+AWSCON="REST_API_endpoint"

// Publish message
AT+MQTTPUB="{"state":{"reported":{"temp":"40","led":"on"}}}"

// Disconnect from a broker
AT+MQTTDIS
```

▶ 섀도우 링크 및 엔드포인트 못 찾으면 클릭

아래 터미널 screenshot 참조하시기 바랍니다.



#동작 예제 결과

1. 동작 예제 결과 AWS -> AWS IoT -> 관리 -> 사물 -> 섀도우 메뉴에서 확인 할 수 있습니다.
2. MQTTSET에서 "업데이트 승인" topic에 subscribe 되어서 MQTT 메시지 전송할 때 바로 결과 볼 수 있습니다.

Shadow Document

Delete Edit

Last update: Mar 18, 2020 5:25:55 PM +0900

Shadow state:


```
{
  "reported": {
    "temp": "50",
    "led": "off"
  }
}
```


Metadata:


```
{
  "metadata": {
    "reported": {
      "temp": {
        "timestamp": 1584519955
      },
      "led": {
        "timestamp": 1584519955
      }
    }
  },
  "timestamp": 1584520988,
  "version": 7
}
```


축하합니다

이제 AWS에 연결 및 MQTT 메시지 전송 마쳤습니다.

Location Tracker with WizFi360

#시작하기 전에

#Hardware Requirement

- Arduino Mega 2560 board
- Desktop and/or laptop computer
- USB 케이블
- WizFi360-EVB-Shield

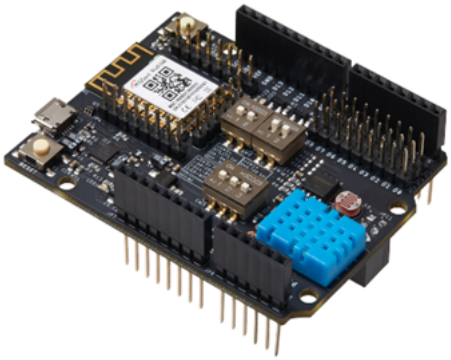
#Software Requirement

- Preferred Serial Terminal (TeraTerm, YAT, etc.)
- Arduino IDE
- ArduinoJson library (version 6.14.1 was used in this guide)

#소개

This guide was inspired by post on [https://www.instructables.com].

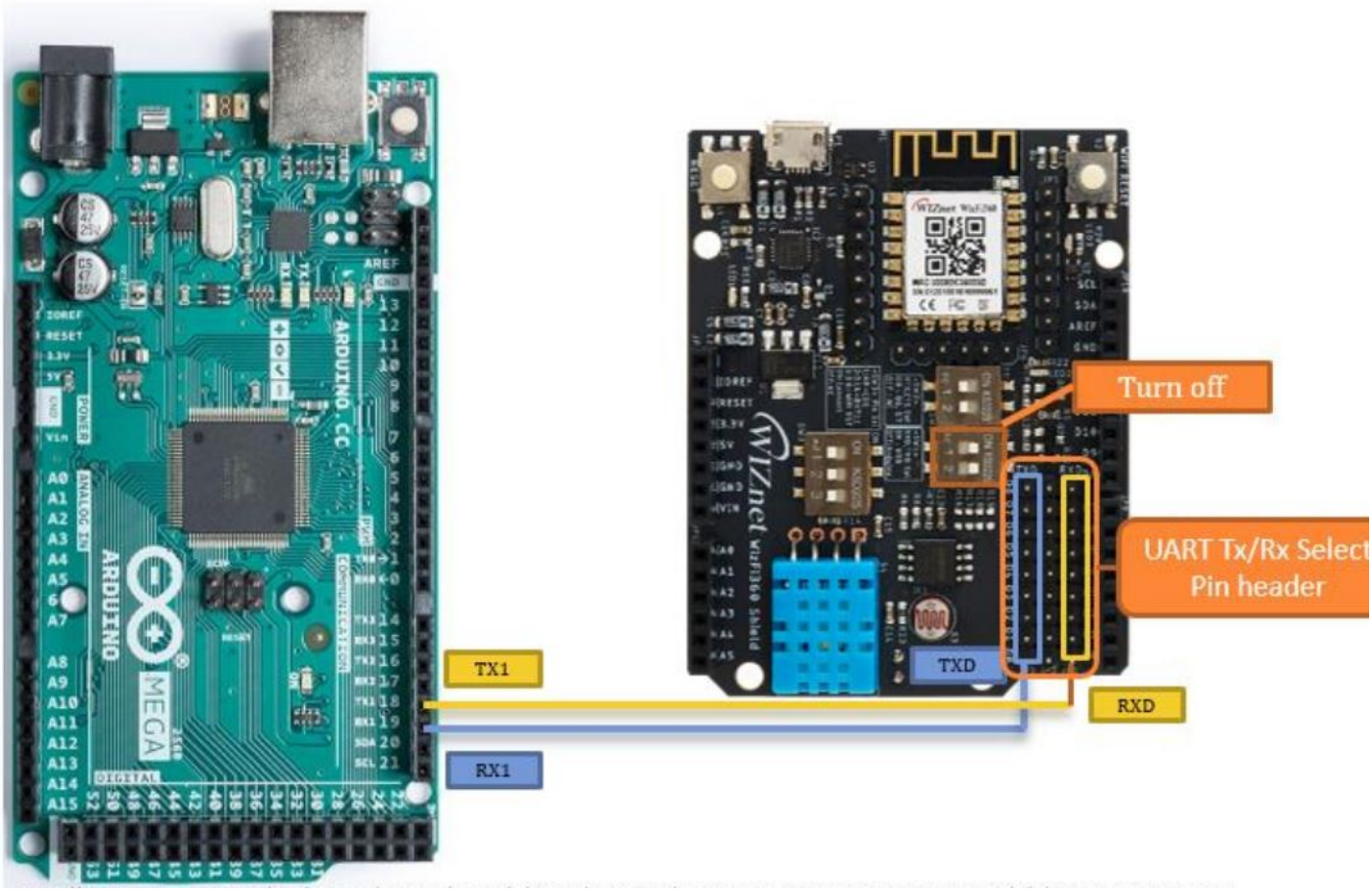
Using unwiredlabs geolocation API it is possible to get location coordinates without GPS and even display.



#디바이스 준비

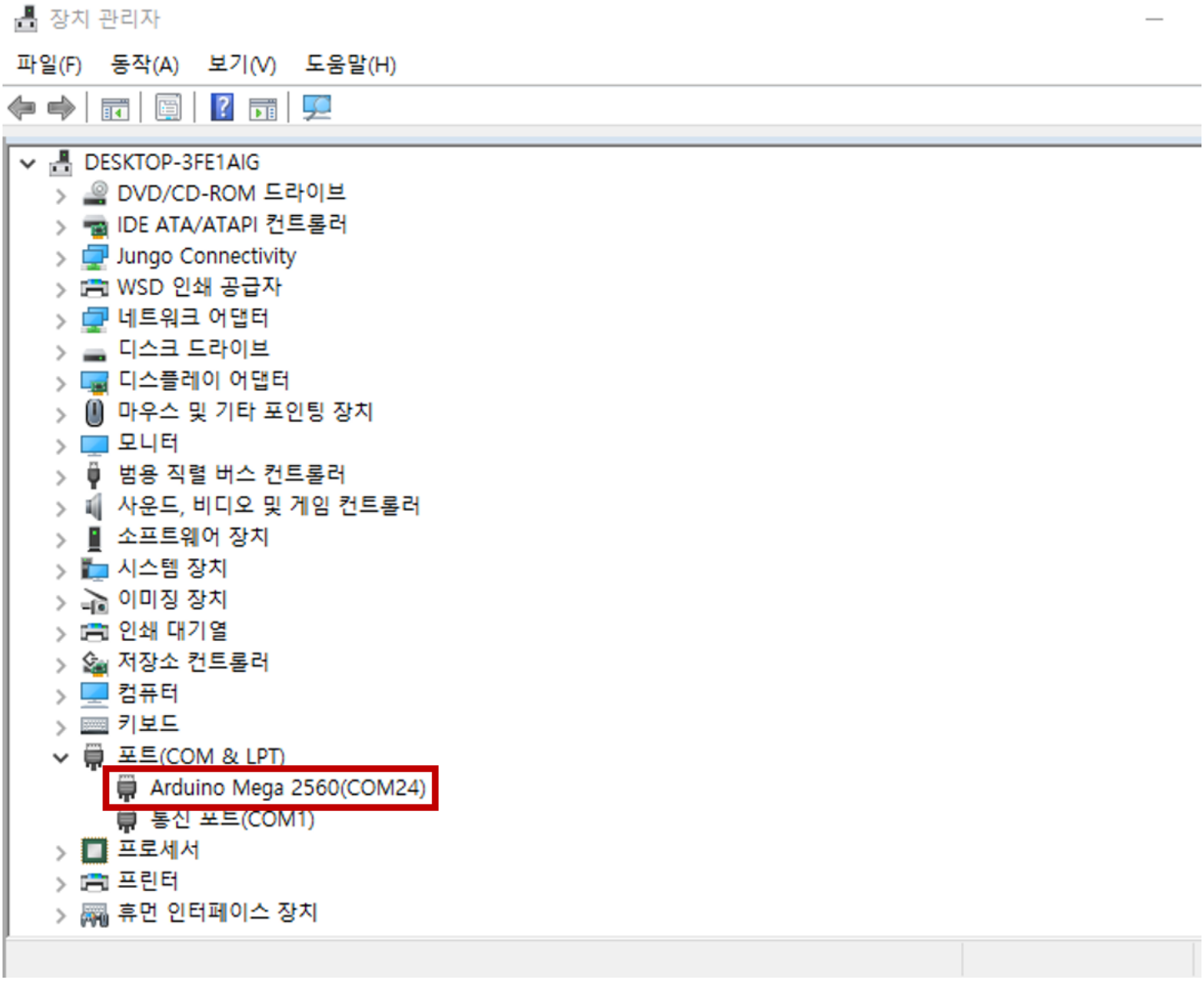
#하드웨어 설정

이 문서에서는 Arduino Mega2560 과 WizFi360-EVB-Shield 를 사용합니다. Arduino Code 에서 UART1 을 사용하여 WizFi360-EVB-Shield 와 통신하기 위해, Arduino 의 TX1, RX1 Pin 과 WizFi360-EVB-Shield 의 RXD, TXD pin 을 연결합니다. WizFi360-EVB-Shield 에서 RXD/TXD Selector 를 OFF 로 변경하여 USB 가 아닌 Pin 을 통해 UART 통신을 하도록 합니다.



#디바이스 연결

하드웨어 설정 후 USB 커넥터를 이용하여 Arduino Mega2560 Rev3 보드와 PC를 연결합니다. PC 운영체제 장치 관리자에서 장치와 연결된 COM 포트를 확인할 수 있습니다.



Arduino IDE를 정상적으로 설치하면, 위와 같이 장치 관리자에서 COM 포트를 확인할 수 있습니다.

#Unwired Labs Geolocation API

As mentioned before, in this guide we are going to use Unwired Labs (https://www.unwiredlabs.com/) Geolocation API. Please go to the website and sign up. On sign up page, fill in your name, email and use case. Select accountt type. For this guide free version was used, however please consider that there is limit for daily usage.

Please check your email with API token. Please copy it as it will be necessary for the code below.

#동작 예제

#Arduino 예제 코드 다운로드

다음 링크에서 Arduino 예제 코드를 다운로드한 후, ino 확장자의 프로젝트 파일을 실행 시킵니다.

예제는 다음 경로에 위치하고 있는 Project를 참고 바랍니다.

samples/Wi-Fi/Arduino_Azure_Atcmd_Wizfi360

#Modify parameters

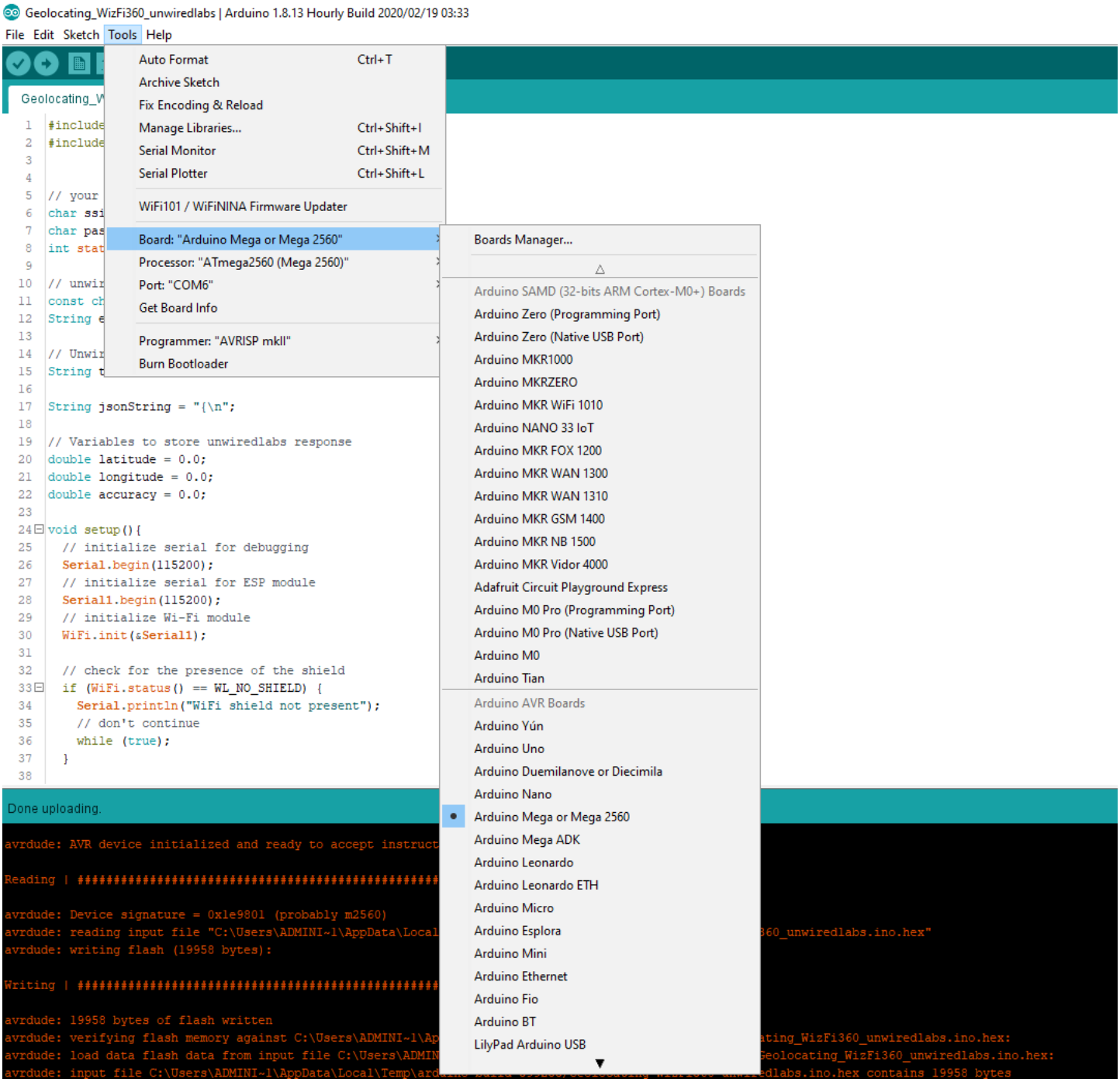
tip

Fill in your own WiFi/hotspot credentials. Also paste the API token that was received in email.

Copy

```
//WiFi credentials
char ssid[] = "XXXXXXXXXXXXXXXXX";     // your network SSID (name)
char pass[] = "XXXXXXXXXXXXXXXXX";         // your network password

// UnwiredLabs API_Token
String token = "XXXXXXXXXXXXXXXXXXXXX";
```

다음 그림과 같이 Arduino Mega2560 보드와 포트를 선택하고, 컴파일을 수행합니다.



컴파일이 완료 되면 다음과 같이 업로드를 수행하여 최종적으로 보드에 업로드를 수행 합니다. 업로드가 정상적으로 완료되면 'avrdude done. Thank you.' 메시지를 확인 할 수 있습니다.

```
Done uploading.

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude: Device signature = 0x1e9801 (probably m2560)
avrdude: reading input file "C:\Users\ADMINI~1\AppData\Local\Temp\arduino_build_399253/Geolocating_WizFi360_unwiredlabs.ino.hex"
avrdude: writing flash (19958 bytes):

Writing | ################################################## | 100% 3.20s

avrdude: 19958 bytes of flash written
avrdude: verifying flash memory against C:\Users\ADMINI~1\AppData\Local\Temp\arduino_build_399253/Geolocating_WizFi360_unwiredlabs.ino.hex:
avrdude: load data flash data from input file C:\Users\ADMINI~1\AppData\Local\Temp\arduino_build_399253/Geolocating_WizFi360_unwiredlabs.ino.hex:
avrdude: input file C:\Users\ADMINI~1\AppData\Local\Temp\arduino_build_399253/Geolocating_WizFi360_unwiredlabs.ino.hex contains 19958 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 2.50s

avrdude: verifying ...
avrdude: 19958 bytes of flash verified

avrdude done.  Thank you.
```

업로드를 완료한 후, 시리얼 모니터를 이용하여 정상적으로 Arduino Mega2560 보드에 업로드 되었는지 확인할 수 있습니다.

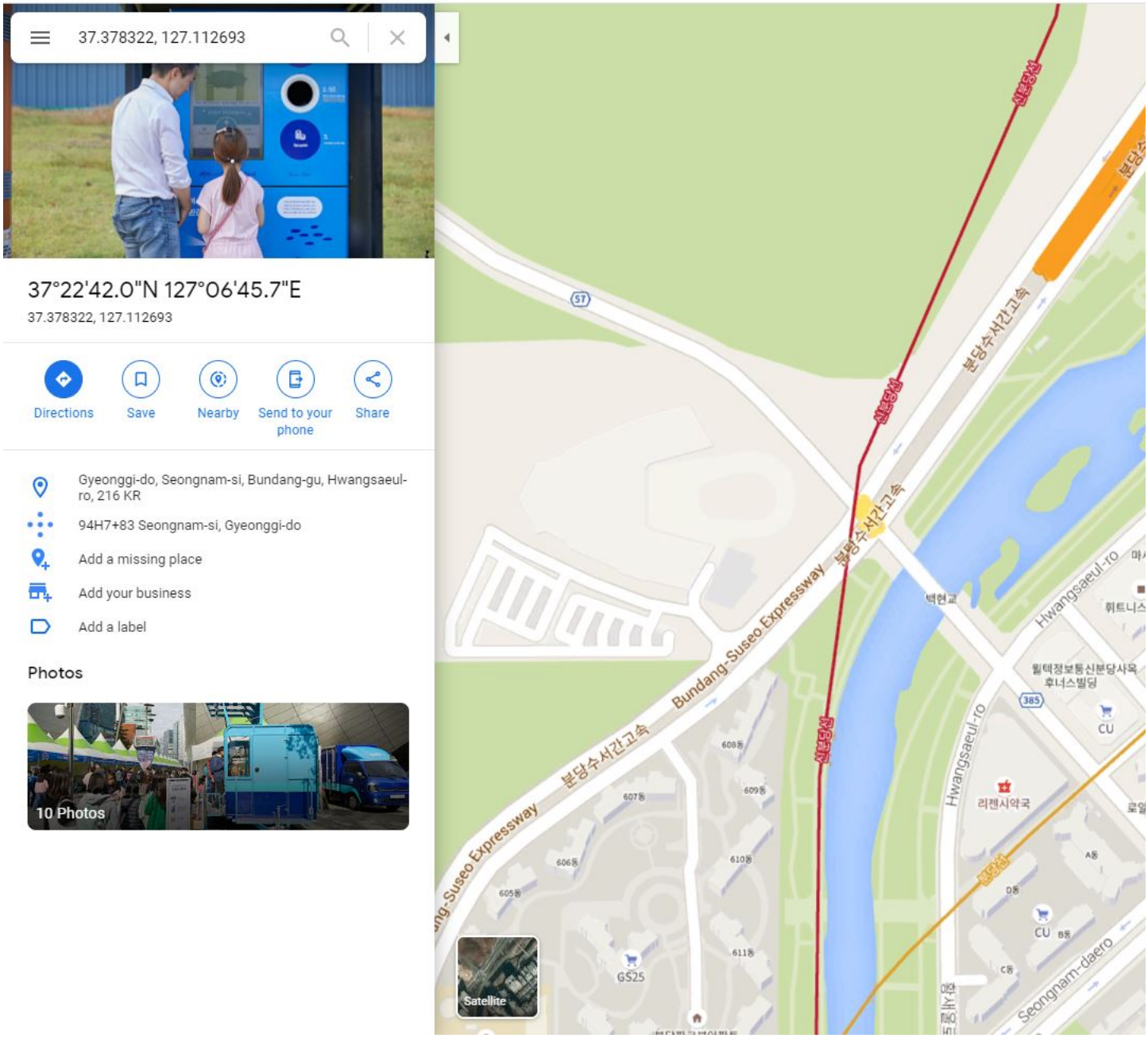
#5. 동작 예제 결과

1. After uploading code to Arduino board results can be checked on serial monitor.

COM6

```
[WiFiEsp] Initializing ESP module
Attempting to connect to WPA SSID: Dap
[WiFiEsp] Connected to Dap
You're connected to the network
SSID: Dap
IP Address: 192.168.0.122
Signal strength (RSSI):-502 dBm

scan done
9 networks found
Requesting URL: https://apl.unwiredlabs.com/v2/process.php
[WiFiEsp] Connecting to apl.unwiredlabs.com
Connected
Request sent
[WiFiEsp] TIMEOUT: 301
{"status":"ok","balance":29,"lat":37.37832407,"lon":127.11274002,"accuracy":40

Latitude = 37.378322
Longitude = 127.112693
Accuracy = 40.00
closing connection

[WiFiEsp] Disconnecting  3
```

2. Google Maps can be used to verify location coordinates.



#다음 단계

In next guide we will connect WizFi360 with Google Location API.