(https://www.dfrobot.com/product-1674.html)

Introduction

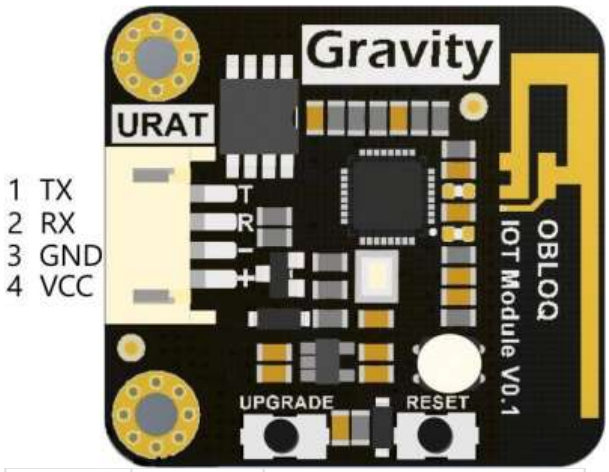
The core of IoT is user experience. It integrates embedded systems technology, mobile technology and network technology in one complete package to create a smart system. It unifies hardware, software and network services in a simple but effective way. Gravity: UART OBLOQ module is a WiFi to Serial device. It mainly faces to non-professional developers, supports standard MQTT protocol IoT service such as Microsoft Azure IoT. Using the OBLOQ module you can quickly build an array of IoT applications without having a complex background knowledge of IoT. In this way, makers can really focus on the purpose of creating interesting things on the IoT. OBLOQ is designed based on ESP8266 WiFi chip, it has compact size, low price, simple interface, plug and play feature, and it works stably under the 3.3V ~ 5V control system.

Specifications

- Supply Voltage: 3.3~5.0V
- Operating Current: <240mA
- Interface Type: Gravity UART 4PIN
- Baud rate: 9600 bps
- Wireless mode: IEEE802.11b/g/n
- Encryption type: WPA WPA2/WPA2–PSK
- Wireless frequency: 2.4GHz
- Product Size: 35mm * 32mm / 1.38inch * 1.26inch
- Built-in agreement: TCP/IP Protocol stack

- Weight:16g

Board Overview



Num	Label	Description
1	TX	Serial port sender
2	RX	Serial port receiver
3	GND	GND
4	VCC	Power + (3.3/5V)

Arduino OBLOQ Tutorial

The OBLOQ module has two basic functions: send data to IoT and receive IoT data. Here's an experimental demonstration of OBLOQ connecting Azure IOT device and sending data to Azure IoT cloud.

- The Arduino reads data from temperature sensor LM35 and sends the data to the Azure IOT device via the OBLOQ module.

Requirements

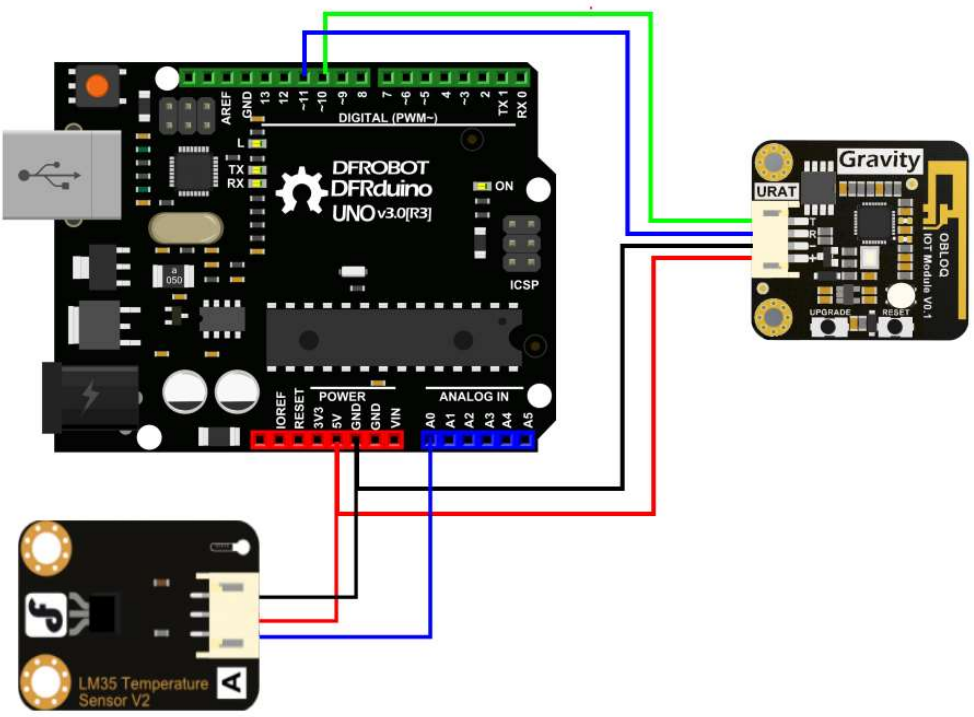
- **Hardware**
  - DFRduino UNO R3 (https://www.dfrobot.com/product-838.html) (or similar) x 1
  - Gravity: Analog LM35 Temperature Sensor For Arduino (https://www.dfrobot.com.cn/goods-74.html) x1
  - 1 x OBLOQ module
  - some connecting wires
- **Software**
  - Arduino IDE, Click to Download Arduino IDE from Arduino®

(https://www.arduino.cc/en/Main/Software)

- **Create Azure IOT device**

    - Register a device in the IoT hub for your device (https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-raspberry-pi-web-simulator-get-started)
    - Make a note of the primary key of the connection string.
    - Device Explorer tool and how to use it (https://github.com/Azure/azure-iot-sdk-csharp/tree/master/tools/DeviceExplorer#download)

Connection Diagram



- TX-Pin10,
- RX-Pin11
- A0-LM35
- GND-GND
- VCC-VCC

Sample code

- Functions realized: Arduino reads temperature data from LM35, and sends the data to Azure IoT device via OBLOQ module.

```
#include <Arduino.h>
#include <SoftwareSerial.h>

//Initialize OBLOQ
Obloq olq(&softSerial, "SeiferNet", "7106230000", "iot_s1.dfrobot.com","1883", "HJ-LCn_HS7

//wifi SSID and password
#define WIFISSID "DFRobot-guest"
#define WIFIPWD  "dfrobot@2017"

//Azure IOT device connection string, this string need modification when connect to differ
const String connectionString = "HostName=dfrobot.azure-devices.cn;DeviceId=temperature;Sh
const String separator = "|";
bool pingOn = true;
bool createIoTClientSuccess = false;
static unsigned long previoustGetTempTime = 0;
static unsigned long pingInterval = 2000;
static unsigned long sendMessageInterval = 10000;
unsigned long previousPingTime = 0;
String receiveStringIndex[10] = {};

//Disconnected wifi detection variable
bool wifiConnect = false;
bool wifiAbnormalDisconnect = false;

SoftwareSerial softSerial(10,11);

enum state{
    WAIT,
    PINGOK,
    WIFIOK,
    AZURECONNECT
}obloqState;

/*****************************************************************************
Function    : sendMessage
Description : send message via serial port
Params      : message : message content, it is a string
Return      : None
*****************************************************************************
void sendMessage(String message)
{
    softSerial.print(message+"\r");
}

/*****************************************************************************
Function    : ping
Description : Check if the serial communication is normal. When Obloq serial port receives
```

```
Return      : None
*****************************************************************************************
void ping()
{
    String pingMessage = F("|1|1|");
    sendMessage(pingMessage);
}

/*****************************************************************************************
Function    : connectWifi
Description : Connect wifi, once successfully connected Obloq returns: |2|3|ip|\r
Params      : ssid : wifi SSID
Params      : pwd :  wifi password
Return      : None
*****************************************************************************************
void connectWifi(String ssid,String pwd)
{
    String wifiMessage = "|2|1|" + ssid + "," + pwd + separator;
    sendMessage(wifiMessage);
}

/*****************************************************************************************
Function    : createIoTClient
Description : Create device client handle, once successfully created Obloq returns: |4|2|1
Params      : connecttionString  device connection string
Return      : None
*****************************************************************************************
void createIoTClient(String connecttionString)
{
    String azureConnectMessage = "|4|2|1|" + connecttionString + separator;
    sendMessage(azureConnectMessage);
}

/*****************************************************************************************
Function    : subscribeDevice
Description : Subscribe to device. Obloq returns message content after the device receives
Params      : None
Return      : None
*****************************************************************************************
void subscribeDevice()
{
    String subscribeDeviceMessage = "|4|2|2|";
    sendMessage(subscribeDeviceMessage);
}

/*****************************************************************************************
Function    : unsubscribeDevice
Description : Unsubscribe to device. Once successfully unsubscribed, Obloq returns: |4|2|6
Params      : None
Return      : None
*****************************************************************************************
void unsubscribeDevice()
{
    String unsubscribeDeviceMessage = "|4|2|6|";
    sendMessage(unsubscribeDeviceMessage);
```

```
}


/*******************************************************************************
Function    : publish
Description : Publish message. Before this command you must create device client handle fi
Params      : message: message content to publish, once successfully sent, OBloq returns :
Return      : None
*******************************************************************************/
void publish(String message)
{
    String publishMessage = "|4|2|3|" + message + separator;
    sendMessage(publishMessage);
}

/*******************************************************************************
Function    : disconnect
Description : destroy device client handle: |4|2|4|1|\r
Params      : None
Return      : None
*******************************************************************************/
void distoryIotClient()
{
    String distoryIotClientMessage = "|4|2|4|";
    sendMessage(distoryIotClientMessage);
}

/*******************************************************************************
Function    : recreateIoTClient
Description : Recreate device client handle, once created successfully, Obloq returns : |4
Params      : None
Return      : None
*******************************************************************************/
void recreateIoTClient()
{
    String recreateIoTClientMessage = "|4|2|5|";
    sendMessage(recreateIoTClientMessage);
}

/*******************************************************************************
Function    : splitString
Description : split a string
Params      : data:  save the cutted char array
Params      : str: the string that will be cutted
Params      : delimiters: the delimiter that cut string
Return      : None
*******************************************************************************/
int splitString(String data[],String str,const char* delimiters)
{
  char *s = (char *)(str.c_str());
  int count = 0;
  data[count] = strtok(s, delimiters);
  while(data[count]){
    data[++count] = strtok(NULL, delimiters);
  }
```

```
  return count;
}

/*************************************************************************************
Function    : handleUart
Description : handle data received by serial port
Params      : None
Return      : None
*************************************************************************************
void handleUart()
{
    while(softSerial.available() > 0)
    {
        String receivedata = softSerial.readStringUntil('\r');
        const char* obloqMessage = receivedata.c_str();
        if(strcmp(obloqMessage, "|1|1|") == 0)
        {
            Serial.println("Pong");
            pingOn = false;
            obloqState = PINGOK;
        }
        if(strcmp(obloqMessage, "|2|1|") == 0)
        {
            if(wifiConnect)
            {
                wifiConnect = false;
                wifiAbnormalDisconnect = true;
            }
        }
        else if(strstr(obloqMessage, "|2|3|") != 0 && strlen(obloqMessage) != 9)
        {
            Serial.println("Wifi ready");
            wifiConnect = true;
            if(wifiAbnormalDisconnect)
            {
                wifiAbnormalDisconnect = false;
                createIoTClientSuccess = true;
                return;
            }
            obloqState = WIFIOK;
        }
        else if(strcmp(obloqMessage, "|4|2|1|1|") == 0)
        {
            Serial.println("Azure ready");
            createIoTClientSuccess = true;
            obloqState = AZURECONNECT;
        }
    }
}

/*************************************************************************************
Function    : sendPing
Description : Check if serial port is communicating normally
Params      : None
Return      : None
```

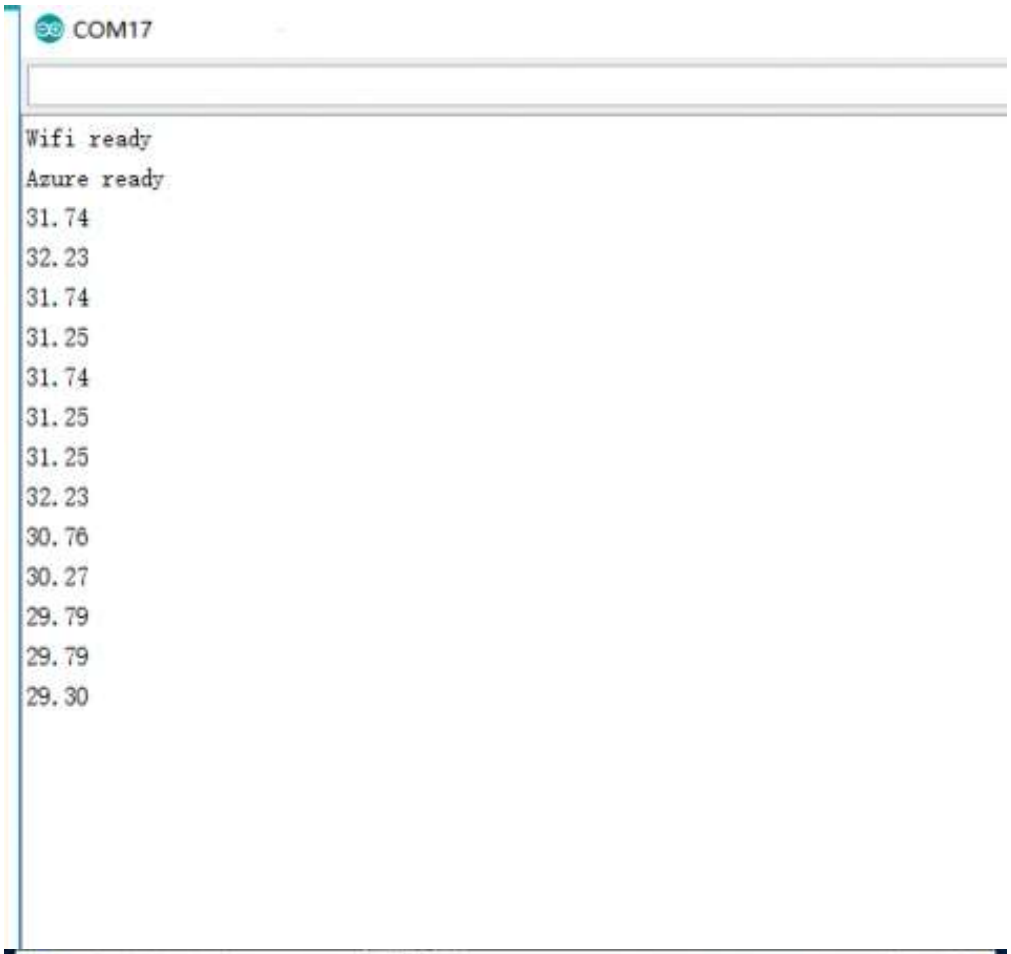
```
Return      : None
*****************************************************************************************
void sendPing()
{
    if(pingOn && millis() - previousPingTime > pingInterval)
    {
        previousPingTime = millis();
        ping();
    }
}

/*****************************************************************************************
Function    : execute
Description : send different command according to different status.
Params      : None
Return      : None
*****************************************************************************************
void execute()
{
    switch(obloqState)
    {
        case PINGOK: connectWifi(WIFISSID,WIFIPWD); obloqState = WAIT; break;
        case WIFIOK: createIoTClient(connectionString);obloqState = WAIT; break;
        case AZURECONNECT : obloqState = WAIT; break;
        default: break;
    }
}

/*****************************************************************************************
Function    : getTemp
Description : Get temperature measured by LM35
Params      : None
Return      : None
*****************************************************************************************
float getTemp()
{
    uint16_t val;
    float dat;
    val=analogRead(A0);//Connect LM35 on Analog 0
    dat = (float) val * (5/10.24);
    return dat;
}

/*****************************************************************************************
Function    : checkWifiState
Description : Callback function of receiving message
Params      : message: message content string that received
Return      : None
*****************************************************************************************
void checkWifiState()
{
    static unsigned long previousTime = 0;
    if(wifiAbnormalDisconnect && millis() - previousTime > 60000)  // reconnect once per m
    {
        previousTime = millis();
        createIoTClientSuccess = false;
```

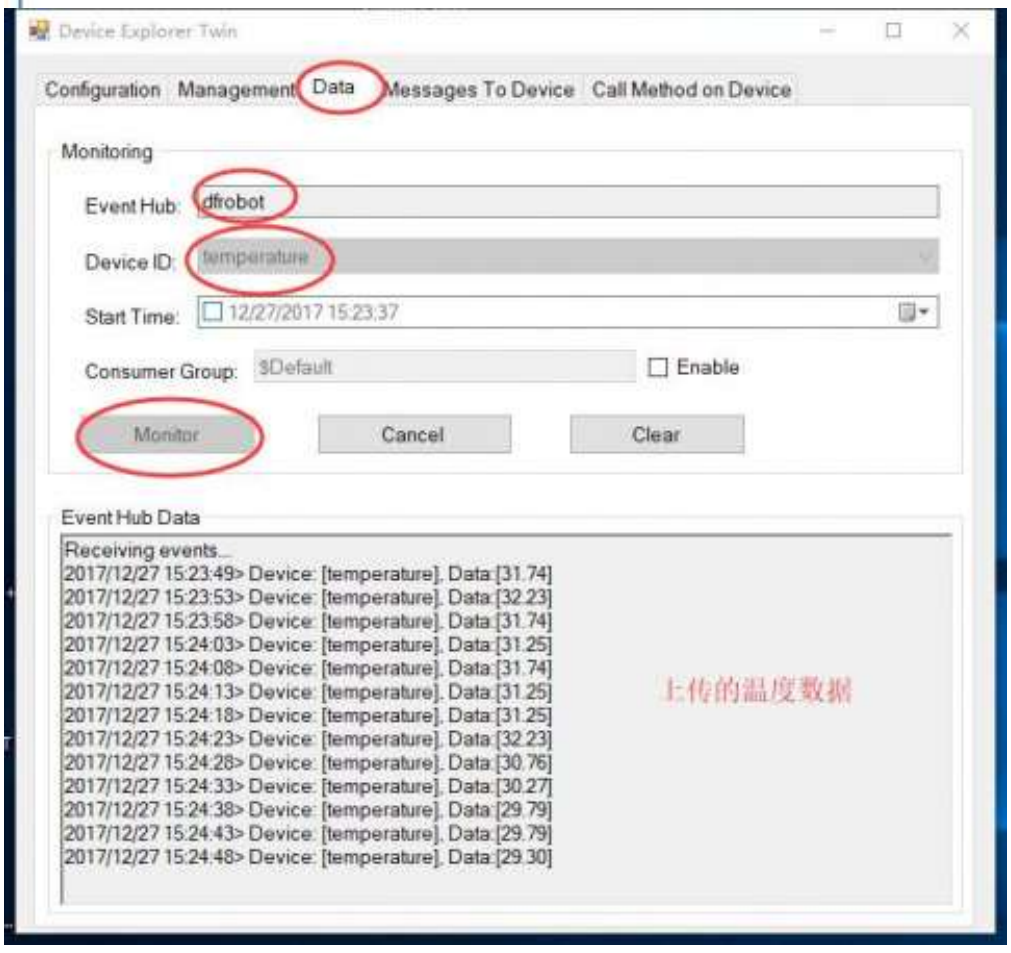
```
        connectWifi(WIFISSID,WIFIPWD);
    }
}

void setup()
{
    Serial.begin(9600);
    softSerial.begin(9600);
}

void loop()
{
    sendPing();
    execute();
    handleUart();
    checkWifiState();
    //Sending data every 5 seconds
    if(createIoTClientSuccess && millis() - previoustGetTempTime > 5000)
    {
        previoustGetTempTime = millis();

        //Get data from temperature sensor
        float temperature = getTemp();
        publish((String)temperature);
        Serial.println(temperature);
    }
}
```

Result demostration

After successfully uploading the program to Arduino, reset both OBLOQ module and Arduino. When not connected to the wifi, OBLOQ shows red light, but once connected, OBLOQ will show a green light.

Use the Tool Device Explorer under the "Data" menu, then select the device that you are going to send message to. Then simply monitor it. You will see the uploaded message in the EventHub Data window.


COM17
Wifi ready
Azure ready
31.74
32.23
31.74
31.25
31.74
31.25
31.25
32.23
30.76
30.27
29.79
29.79
29.30


Device Explorer Twin
Configuration
Management
Data
Messages To Device
Call Method on Device
Monitoring
Event Hub:
dfrobot
Device ID:
temperature
Start Time:
12/27/2017 15:23:37
Consumer Group:
$Default
Enable
Monitor
Cancel
Clear
Event Hub Data
Receiving events...
2017/12/27 15:23:49> Device: [temperature], Data:[31.74]
2017/12/27 15:23:53> Device: [temperature], Data:[32.23]
2017/12/27 15:23:58> Device: [temperature], Data:[31.74]
2017/12/27 15:24:03> Device: [temperature], Data:[31.25]
2017/12/27 15:24:08> Device: [temperature], Data:[31.74]
2017/12/27 15:24:13> Device: [temperature], Data:[31.25]
2017/12/27 15:24:18> Device: [temperature], Data:[31.25]
2017/12/27 15:24:23> Device: [temperature], Data:[32.23]
2017/12/27 15:24:28> Device: [temperature], Data:[30.76]
2017/12/27 15:24:33> Device: [temperature], Data:[30.27]
2017/12/27 15:24:38> Device: [temperature], Data:[29.79]
2017/12/27 15:24:43> Device: [temperature], Data:[29.79]
2017/12/27 15:24:48> Device: [temperature], Data:[29.30]
上传的温度数据

More Documents

Sample code (https://github.com/DFRobot/OBLOQExamples)

Get **Gravity: UART OBLOQ IoT Module** (https://www.dfrobot.com/product-1674.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/index.php?route=information/distributorslogo)

Turn to the Top