Wido-WIFI IoT Node SKU:DFR0321

From Robot Wiki

Contents

Introduction

Wido is an Arduino compatible WIFI IoT Node development board, which integrates with WG1300 WIFI solution. The microcontroller of Wido is ATMEL ATmega32U4.

Classic Application

- M2M Sensor Node development
- Toys
- Gaming
- Smart Home Device

Specification

- Power Supply range: 5v or 7-12v
- Arduino Leonardo Compatible
- Integrate with WG1300 WIFI chip and support 2.4GHz IEEE 802.11 b/g network
- WIFi and MicroSD driven by SPI port
- On board 2.4G PCB Antenna
- Driver pins:
  - WIFI Module-D7(IRQ),D5(VBAT),D10(CS),D14(MISO),D15(SCK),D16(MOSI)
  - MicroSD-D4(CS),D14(MISO),D15(SCK),D16(MOSI)

PinOut

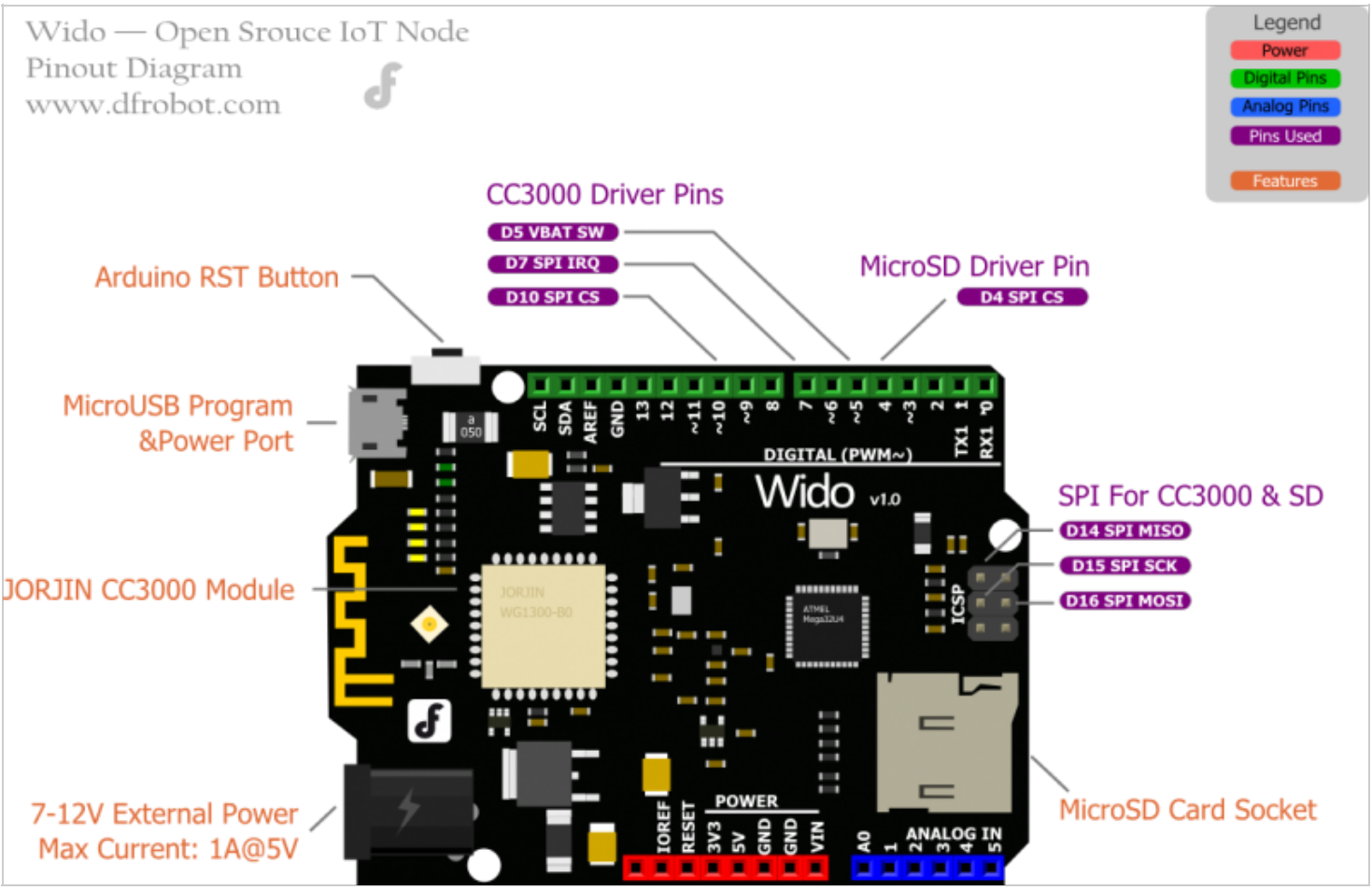

Fig1: Wido Pin Out

Get Started with Wido

First of all, we will bring you a step by step tutorial to lead finish the Wido router connection configuration and make it work as a TCP client connected to the local server.

Hardware Requirement

- Wido 1unit
- MicroUSB Cable 1unit

Detail Steps

1. Install the Arduino library (https://github.com/Lauren-ED209/Adafruit_CC3000_Library/archive/master.zip) to your Arduino IDE. This library for Wido is forked from Adafruit. They've finished an awesome project for this CC3000 development. Based on this library, we updated the pin configuration and extended some application sample codes. For more details from Github (https://github.com/Lauren-ED209/Adafruit_CC3000_Library)

2. Open the sample code, which is named buildtest.

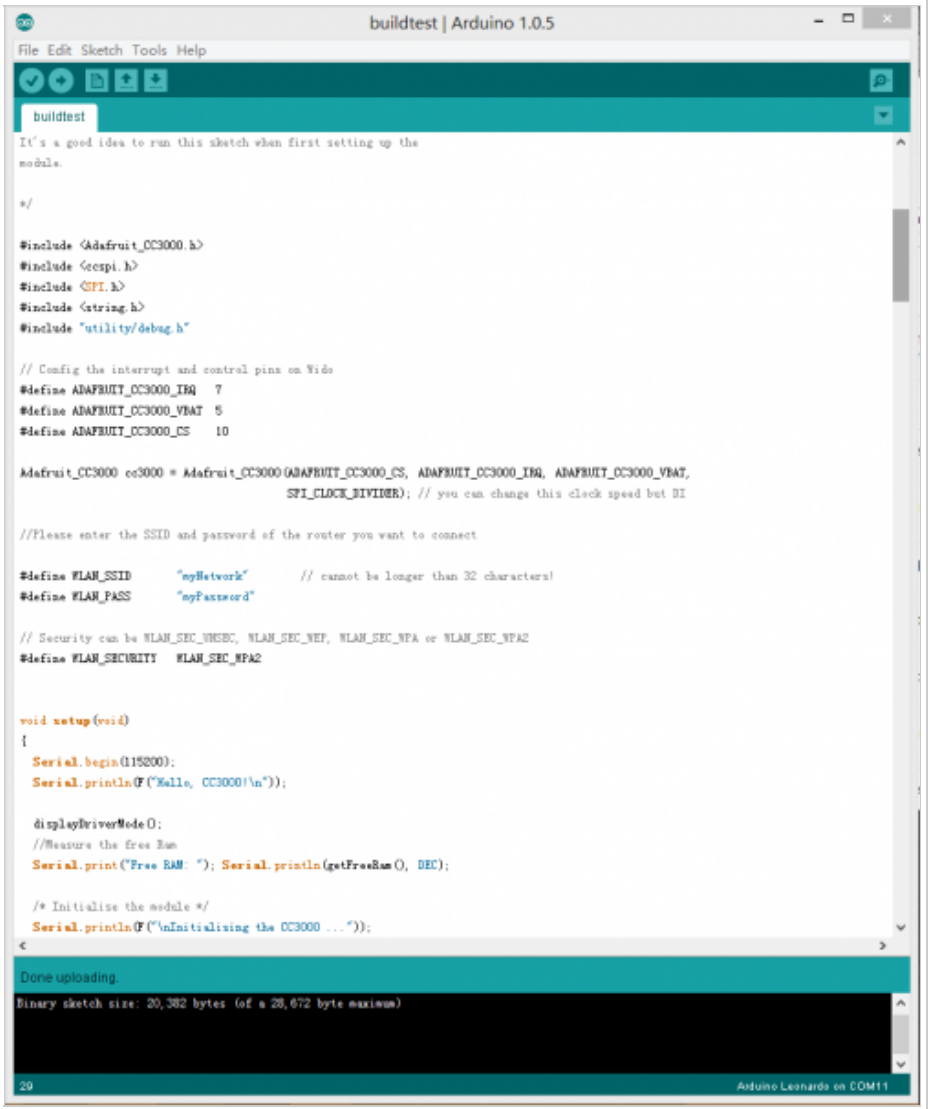

Fig2: buildtest

3. Upload the sample code to Wido and check the Serial monitor after programming.

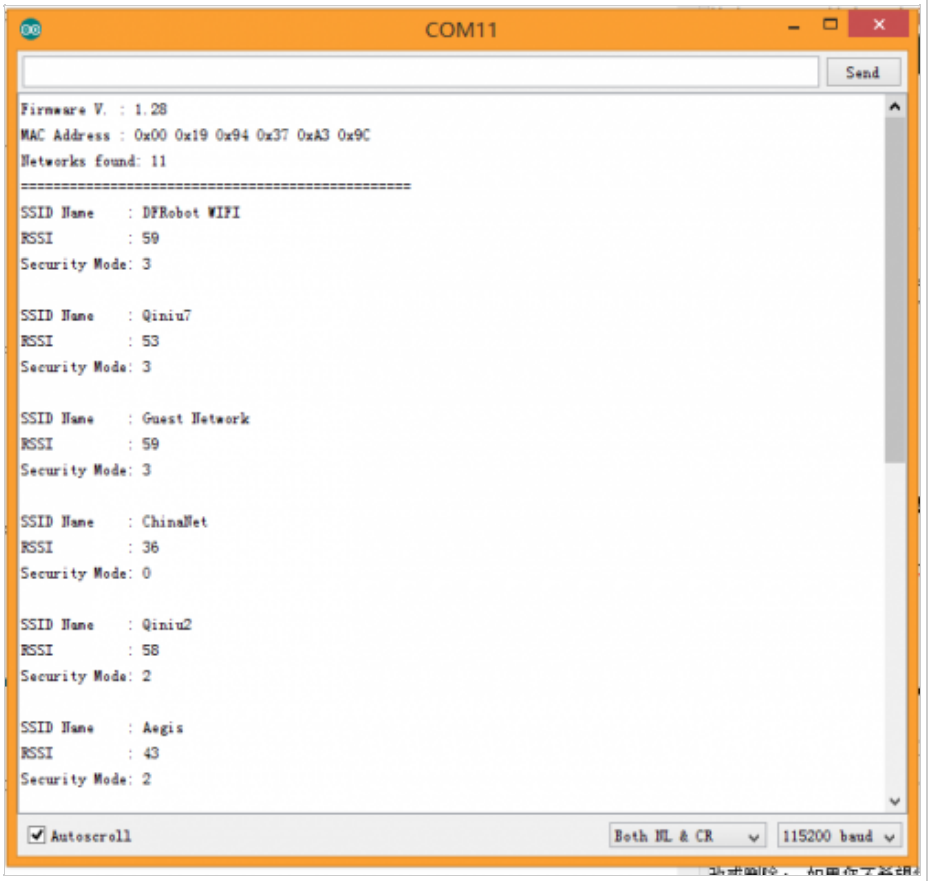

Fig3: Scan the Routers

You will see the information printed including the MAC address and local router information detected by Wido.

4. Update the SSID and password configuration in your code!

?

```
1//Please enter the SSID and password of the router you want to connect
2#define WLAN_SSID       "myNetwork"        // cannot be longer than 32 characters!
3#define WLAN_PASS       "myPassword"
```

5. Then upload the sample sketch again. And after several seconds. You will see the effect, like the picture attached.

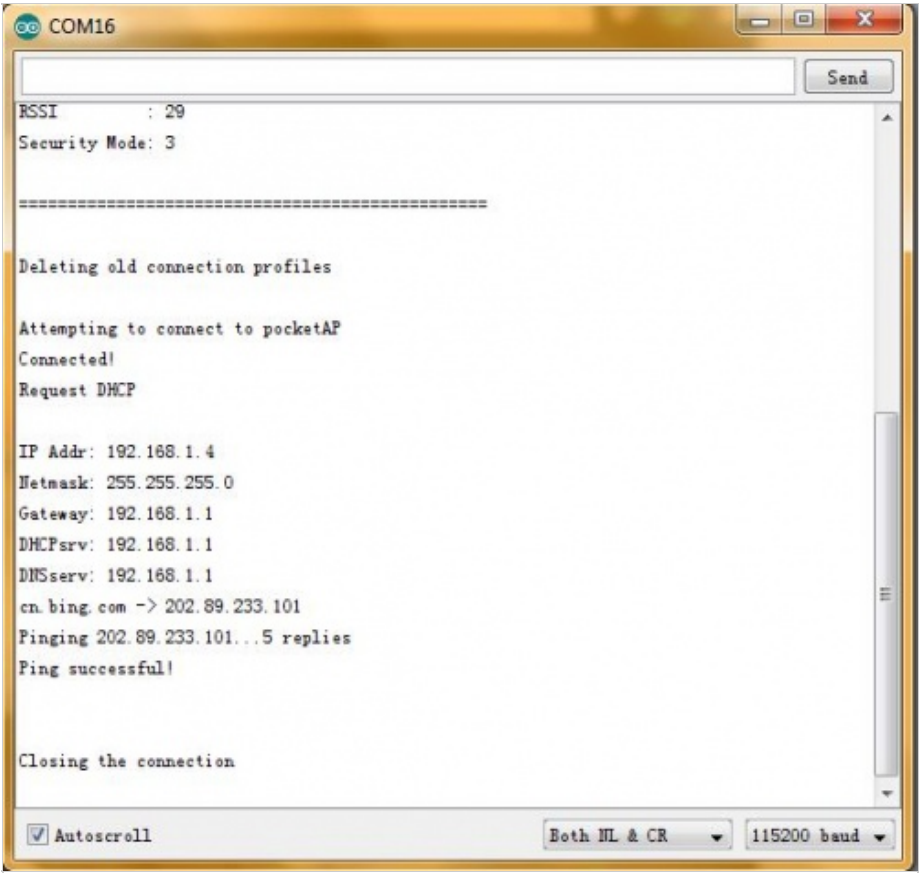

Fig4: Ping bing.com

Connecting to the local server

Software Tools

- TCP Server Tool (https://github.com/Lauren-ED209/Wido-OpenSource-IOT-Node/raw/master/Tools/USR-TCP232-Test.exe), used to create a local TCP server from your PC

Steps

1. Open the Tool above. Config the port number, and click listening to wait for the client connection.

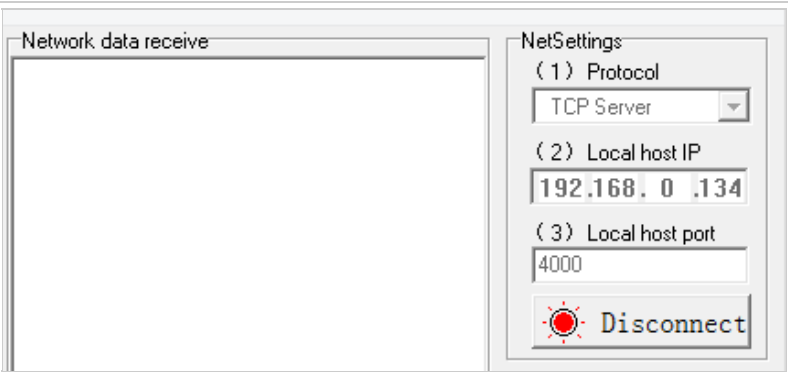

Fig1: Open TCP Server

2. Open the sketch named "Wido2LocalTcpServer", and config the TCP server address and port according to your tool setup.

```
?
1/* Set the target ip address and connection port */
2  uint32_t ip = WiDo.IP2U32(192,168,0,134);
3  tcpClient = WiDo.connectTCP(ip, 9000);
```

Upload the full sample after TCP/Router configuration.

```
?
1 #include <Adafruit_CC3000.h>
2 #include <ccspi.h>
3 #include <SPI.h>
4 #include <string.h>
5 #include "utility/debug.h"
6
7 #define WiDo_IRQ   7
8 #define WiDo_VBAT  5
9 #define WiDo_CS    10
10Adafruit_CC3000 WiDo = Adafruit_CC3000(WiDo_CS, WiDo_IRQ, WiDo_VBAT,
11                                        SPI_CLOCK_DIVIDER); // you can change this clock speed
12
13#define WLAN_SSID       "myNetwork"           // cannot be longer than 32 characters!
14#define WLAN_PASS       "myPassword"
15
16// Security can be WLAN_SEC_UNSEC, WLAN_SEC_WEP, WLAN_SEC_WPA or WLAN_SEC_WPA2
17#define WLAN_SECURITY   WLAN_SEC_WPA2
18#define TIMEOUT_MS  1000
19
20void setup(){
21
22  Serial.begin(115200);
23  /* Initialise the module */
24  Serial.println(F("\nInitialising the CC3000 ..."));
25  if (!WiDo.begin())
26  {
27    Serial.println(F("Unable to initialise the CC3000! Check your wiring?"));
28    while(1);
29  }
30
31  /* NOTE: Secure connections are not available in 'Tiny' mode!
32     By default connectToAP will retry indefinitely, however you can pass an
33     optional maximum number of retries (greater than zero) as the fourth parameter.
34  */
35
36  Serial.println(F("Connecting Router/AP"));
37  if (!WiDo.connectToAP(WLAN_SSID, WLAN_PASS, WLAN_SECURITY)) {
38    Serial.println(F("Failed!"));
39    while(1);
40  }
41
42  Serial.println(F("Router/AP Connected!"));
43
44  /* Wait for DHCP to complete */
45  Serial.println(F("Request DHCP"));
46  while (!WiDo.checkDHCP())
47  {
48    delay(100); // ToDo: Insert a DHCP timeout!
49  }
50}
51
```

```
void loop(){

  static Adafruit_CC3000_Client tcpClient;
  static unsigned long heartRate = millis();

  if(!tcpClient.connected()){
    Serial.println("Try to connect the Local Server");
    tcpClient.close();

    /* Set the target ip address and connection port */
    uint32_t ip = WiDo.IP2U32(192,168,0,134);
    tcpClient = WiDo.connectTCP(ip, 4000);

    if(!tcpClient.connected()){
      Serial.println(F("Couldn't connect to server! Make sure TCP Test Tool is running on the server."));
      while(1);
    }
  }
  else if(millis() - heartRate > 1000){
    heartRate = millis();  // Update time stamp of the microcontroller system

    char clientString[30];
    sprintf(clientString, "%s%d%s", "Wido heartRate: ",heartRate/1000," s\r\n");

    Serial.println(clientString);
    tcpClient.fastrprintln(clientString);
  }

  /* Read data until either the connection is closed, or the timeout is reached. */
  unsigned long lastRead = millis();
  while (tcpClient.connected() && (millis() - lastRead < TIMEOUT_MS)) {
    while (tcpClient.available()) {
      char c = tcpClient.read();
      Serial.print(c);
      lastRead = millis();

      // Disable sending message for a moment
      heartRate = millis();
    }
  }
}
```

3. Open the serial monitor. After connecting the router, your Wido will start to upload data to the TCP server tool!

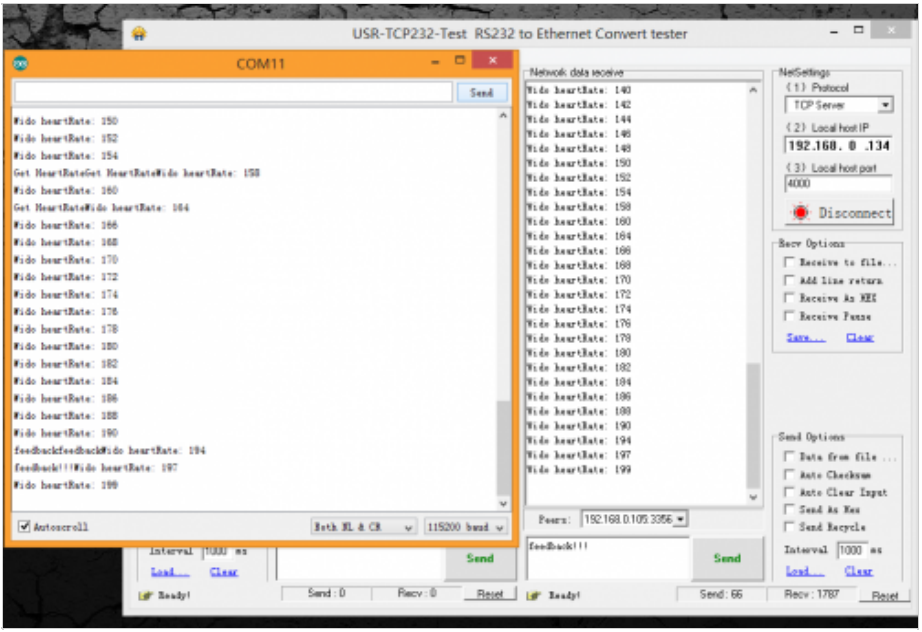

Fig1: Open TCP Server

Finish the local server connection and communication now. You get the access to build a WIFI controlled robot with Wido or some local cloud service project. But it's not enough for most of WIFI application!

Connecting to Xively

Xively (formerly Cosm) is a Platform as a Service that provides everything you need to simpify and accelerate the creation of compelling connected products and solutions. In this section, we will bring your sensor to the **cloud**. ☺

Steps

1. Create your own Xively account and login the develop page. Create the a new device. xively.com (https://xively.com/)

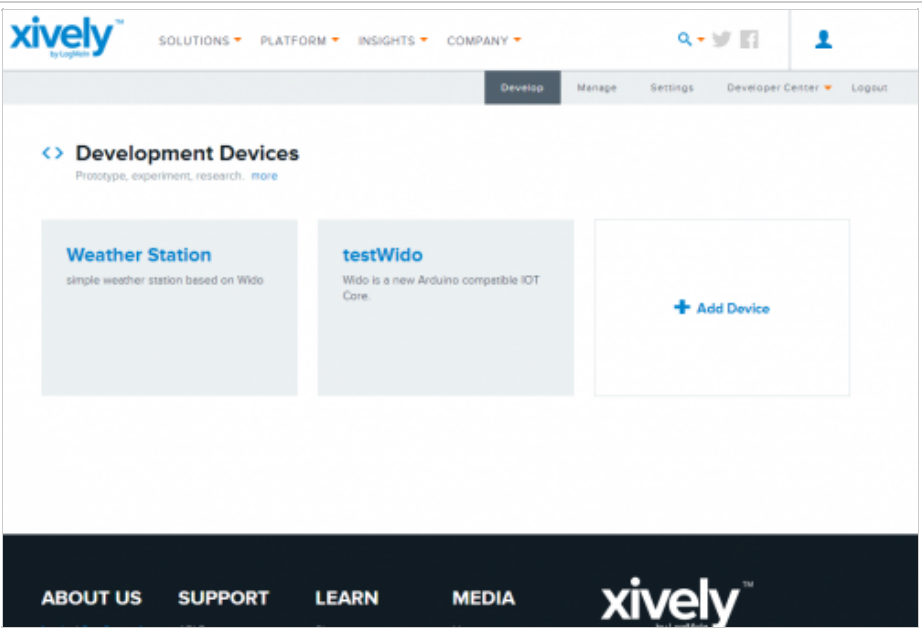

Fig1: Create Xively Account

2. Click and enter your device.

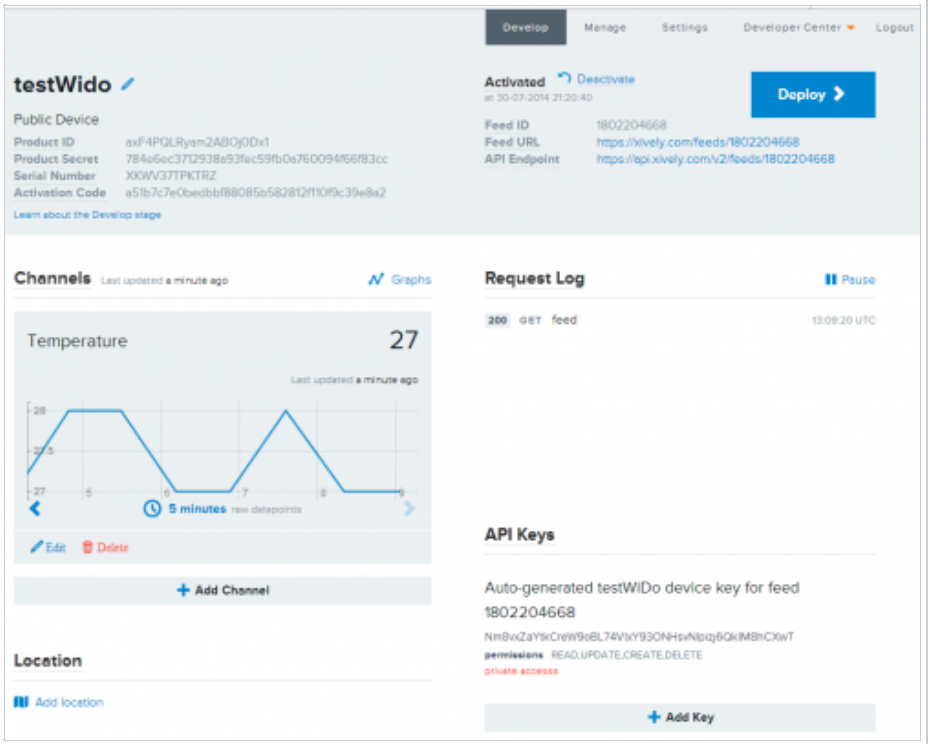

Fig2: Open the device page

3. Open the example code named "Wido2Xively" included in the Arduino library. Modify the info below in the sample code according to the device information from the step 2.

```
?
1#define WEBSITE  "api.xively.com"
2#define API_key  "Nm8vxZaYtkCreW9oBL74VIxY93ONHsvNlpizj6QkIM8hxxxx"  // Check your API Key from device page
3#define feedID  "180220xxxx"                                        // Check your feedID
```

The sample code:

```
?
1  /*************************************************
2   * This is an example for the DFRobot Wido - Wifi Integrated IoT lite sensor and control node
3   *
4   * Designed specifically to work with the DFRobot Wido products:
5   *
6   *
7   * The main library is forked from Adafruit
8   *
9   * Written by Lauren
10  * BSD license, all text above must be included in any redistribution
11  *
12  *************************************************/
13 /*
14 This example code is used to connect the Xively cloud service.
15
16 The device required is just:
17
18 1. LM35 low cost temperature sensor or any device you used to upload data
19 2. And Wido
20
21 */
22
23
```

```
#include <Adafruit_CC3000.h>
#include <ccspi.h>
#include <SPI.h>

#define Wido_IRQ   7
#define Wido_VBAT  5
#define Wido_CS    10
Adafruit_CC3000 Wido = Adafruit_CC3000(Wido_CS, Wido_IRQ, Wido_VBAT,
SPI_CLOCK_DIVIDER); // you can change this clock speed

#define WLAN_SSID       "myNetwork"           // cannot be longer than 32 characters!
#define WLAN_PASS       "myPassword"
// Security can be WLAN_SEC_UNSEC, WLAN_SEC_WEP, WLAN_SEC_WPA or WLAN_SEC_WPA2
#define WLAN_SECURITY   WLAN_SEC_WPA2

#define IDLE_TIMEOUT_MS  2000
#define TCP_TIMEOUT      3000

#define WEBSITE  "api.xively.com"
#define API_key  "Nm8vxZaYtkCreW9oBL74VIxY93ONHsvNlpizj6QkIM8hxxxx"  // Update Your API Key
#define feedID  "180220xxxx"                                         // Update Your own feedID

void setup(){

  Serial.begin(115200);
  Serial.println(F("Hello, CC3000!\n"));

  /* Initialise the module */
  Serial.println(F("\nInitialising the CC3000 ..."));
  if (!Wido.begin())
  {
    Serial.println(F("Unable to initialise the CC3000! Check your wiring?"));
    while(1);
  }

  /* Attempt to connect to an access point */
  char *ssid = WLAN_SSID;             /* Max 32 chars */
  Serial.print(F("\nAttempting to connect to "));
  Serial.println(ssid);

  /* NOTE: Secure connections are not available in 'Tiny' mode!
   By default connectToAP will retry indefinitely, however you can pass an
   optional maximum number of retries (greater than zero) as the fourth parameter.
   */
  if (!Wido.connectToAP(WLAN_SSID, WLAN_PASS, WLAN_SECURITY)) {
    Serial.println(F("Failed!"));
    while(1);
  }

  Serial.println(F("Connected!"));

  /* Wait for DHCP to complete */
  Serial.println(F("Request DHCP"));
  while (!Wido.checkDHCP())
  {
    delay(100); // ToDo: Insert a DHCP timeout!
  }

}
```

```
83
84 uint32_t ip = 0;    // Store Xively ip address
85 float temp = 0;     // Store temporary sensor data for post
86
87 void loop(){
88
89   static Adafruit_CC3000_Client WidoClient;
90   static unsigned long RetryMillis = 0;  // timer stamp for building the connection
91   static unsigned long uploadtStamp = 0; // timer stamp for posting data to service
92   static unsigned long sensortStamp = 0; // timer stamp for reading data to LM35
93
94   // Apply for the connection with the cloud service
95
96   if(!WidoClient.connected() && millis() - RetryMillis > TCP_TIMEOUT){
97     // Update the time stamp
98     RetryMillis = millis();
99
100    Serial.println(F("Try to connect the cloud server"));
101
102    //Get Xively IOT Server IP
103    ip = Wido.IP2U32(216,52,233,120);
104    WidoClient = Wido.connectTCP(ip, 80);
105  }
106
107  // After building the connection with the service
108  // Post the sensor data to Xively
109  if(WidoClient.connected() && millis() - uploadtStamp > 2000){
110    uploadtStamp = millis();
111    // If the device is connected to the cloud server, upload the data every 2000ms.
112
113    // Prepare JSON for Xively & get length
114    int length = 0;
115    // JSON beginning
116    String data_start = "";
117    data_start = data_start + "\n"
118      + "{\"version\":\"1.0.0\",\"datastreams\" : [ ";
119    // JSON for temperature & humidity
120    String data_temperature = "{\"id\" : \"Temperature\",\"current_value\" : \""
121      + String(int(temp)) + "\"}]}";
122    // Get length
123    length = data_start.length() + data_temperature.length();
124
125    Serial.println(F("Connected to Xively server."));
126
127    // Send headers
128    Serial.print(F("Sending headers"));
129    WidoClient.fastrprint(F("PUT /v2/feeds/"));
130    WidoClient.fastrprint(feedID);
131    WidoClient.fastrprintln(F(".json HTTP/1.0"));
132    Serial.print(F("."));
133    WidoClient.fastrprintln(F("Host: api.xively.com"));
134    Serial.print(F("."));
135    WidoClient.fastrprint(F("X-ApiKey: "));
136    WidoClient.fastrprintln(API_key);
137    Serial.print(F("."));
138    WidoClient.fastrprint(F("Content-Length: "));
139    WidoClient.println(length);
140    Serial.print(F("."));
141    WidoClient.fastrprint(F("Connection: close"));
```

```
142    Serial.println(F(" done."));
143
144    // Send data
145    Serial.print(F("Sending data"));
146    WidoClient.fastrprintln(F(""));
147    WidoClient.print(data_start);
148    Serial.print(F("."));
149    WidoClient.print(data_temperature);
150    Serial.print(F("."));
151    WidoClient.fastrprintln(F(""));
152    Serial.println(F(" done."));
153
154    /* Get the http page info
155    Serial.println(F("Reading answer..."));
156    while (WidoClient.connected()) {
157      while (WidoClient.available()) {
158        char c = WidoClient.read();
159        Serial.print(c);
160      }
161    }
162    */
163    delay(1000);             // Wait for 1s to finish posting the data stream
164    WidoClient.close();      // Close the service connection
165    RetryMillis = millis();  // Reset the timer stamp for applying the connection with the service
166  }
167
168  //Realtime update the latest sensor data from LM35 once per 100ms and convert the unit (degree)
169  if(millis() - sensortStamp > 100){
170    sensortStamp = millis();
171    // read the LM35 sensor value and convert to the degrees every 100ms.
172
173    int reading = analogRead(0);
174    temp = reading *0.0048828125*100;
175    Serial.print(F("Real Time Temp: "));
176    Serial.println(temp);
177  }
178
179}
```

4. Then Wido will upload the sensor data to the cloud once every 2s. You could check the Request Log and the Channel info from the device page now.

Adafruit library key features

The Adafruit library for CC3000 is really good and extending lots of feature for the WG1300. This library is also modified based on the TI smartlink solution.

Here're some simple introduction for the functions commonly used!

1. Trick for saving the programming space. The ATmega32U4 programming space is limited. So call the feature is really useful for your program.

?
```
1#define CC3000_TINY_DRIVER
```

The code above will launch the tiny driver function.

2. Initialise the module.

```
?
1if (!cc3000.begin())
2{
3  Serial.println(F("Unable to initialise the CC3000! Check your wiring?"));
4  while(1);
5}
```

3. Setup the router connection!

```
?
1if (!cc3000.connectToAP(WLAN_SSID, WLAN_PASS, WLAN_SECURITY)) {
2  Serial.println(F("Failed!"));
3  while(1);
4}
```

4. Finish and get the DHCP info from the router/AP

```
?
1while (!cc3000.checkDHCP())
2  {
3    delay(100); // ToDo: Insert a DHCP timeout!
4  }
```

Part 2-4 the key steps to access the network.

Documents

- Wido v1.0 Schematic (https://github.com/Lauren-ED209/Wido-OpenSource-IOT-Node/raw/master/DFR0321_V1.0_Schematic.pdf)
- WIFI chip Datasheet (https://github.com/Lauren-ED209/Wido-OpenSource-IOT-Node/raw/master/Reference/WG1300%20datasheet.pdf)
- Github (https://github.com/Lauren-ED209/Wido-OpenSource-IOT-Node)
- Arduino library (https://github.com/Lauren-ED209/Adafruit_CC3000_Library)