

Arduino to ThingSpeak via SIM800 No Wi-Fi

sending data from an Arduino microcontroller to the ThingSpeak platform using a GPRS module, specifically the SIM800.

 Difficulté Facile

 Durée 6 heure(s)

 Catégories Électronique

 Coût 20 USD (\$)

Sommaire

Introduction

Étape 1 - About the Project

Étape 2 - Get PCBs For Your Projects Manufactured

Étape 3 - Networking testing

Étape 4 - TCP/IP Communication

Étape 5 - Client communication in non-transparent mode

Étape 6 - Client communication in transparent transmission mode

Étape 7 - HTTP Communication

Étape 8 - Establishing Communication Between Arduino Nano, SIM800, and ThingSpeak (Serial Monitor Data Captured During Project Execution).

Commentaires

Introduction

Sending data from an Arduino microcontroller to the ThingSpeak platform using a GPRS module is not a novel project concept. While it may appear outdated in numerous regions, owing to the rapid evolution of communication technologies such as 2G, 3G, 4G, 5G, and the potential for 6G, this is not the case in my country. In India, 2G technologies remain prevalent, and major network operators have confirmed their intent to sustain 2G services.

Matériaux

Outils

Étape 1 - About the Project

The legacy project documents that were originally designed with the SIM800 module may necessitate slight adjustments. This project will prove invaluable to those who continue to rely on 2G and GPRS technology. It offers essential support and guidance for individuals who intend to persist with these communication methods.

Sending data from an Arduino microcontroller to the ThingSpeak platform using a GPRS module, specifically the SIM800, is a fundamental concept. The crucial aspect is that this communication method operates independently of Wi-Fi, constituting an IoT connectivity solution that relies on GPRS for data transmission.

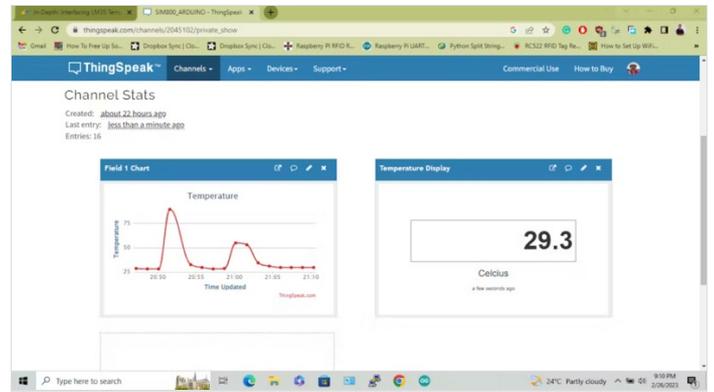
1 / 2

In this project, LM35 temperature sensor data is being transmitted to the ThingSpeak platform through an Arduino Nano and a SIM800 module. The SIM800 module is leveraged to establish a GPRS connection, facilitating the transmission of data to ThingSpeak at specified intervals.

To ensure a reliable connection between ThingSpeak and the hardware, users must configure the SIM800 module to establish a connection with their mobile network. This configuration encompasses setting the Access Point Name (APN) specific to their mobile carrier. It's important to note that the specific AT commands for this configuration may vary based on the user's chosen mobile network provider.

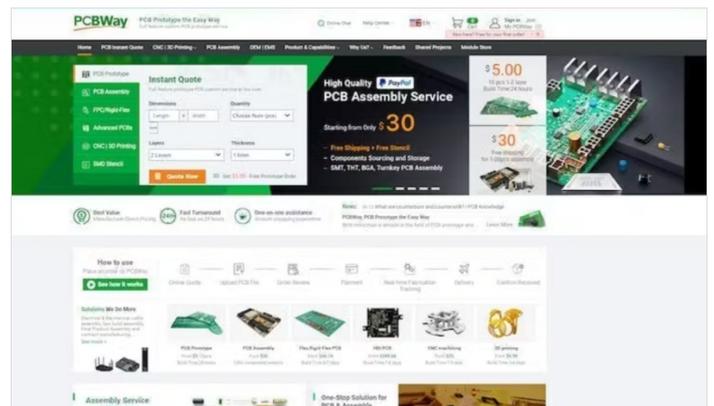
For this project, I utilized the services of the network provider **Airtel** to establish the connection.

Communication between the hardware components, specifically the SIM800 module, Arduino Nano, and ThingSpeak platform, relies entirely on AT commands. To ensure successful project implementation and effectively troubleshoot any issues that may arise, users must possess a basic understanding of SIM800 AT commands. This knowledge is crucial for configuring, managing, and diagnosing the communication process and resolving potential challenges during the project.



Étape 2 - Get PCBs For Your Projects Manufactured

You must check out PCBWAY for ordering PCBs online for cheap! You get 10 good-quality PCBs manufactured and shipped to your doorstep for cheap. You will also get a discount on shipping on your first order. Upload your Gerber files onto PCBWAY to get them manufactured with good quality and quick turnaround time. PCBWay now could provide a complete product solution, from design to enclosure production. Check out their online Gerber viewer function. With reward points, you can get free stuff from their gift shop. Also, check out this useful blog on PCBWay Plugin for KiCad from here. Using this plugin, you can directly order PCBs in just one click after completing your design in KiCad.



Étape 3 - Networking testing

The table provided below lists several common AT commands that serve to swiftly and effectively verify the proper functioning of the SIM800C's AT serial communication and network connection. Before proceeding with the steps outlined for the network communication demonstration, it is advisable to conduct a straightforward network test. This preliminary test ensures that the intended network connection is in working order and fully operational.

Commands	Description
AT	AT Test Command
ATE	ATE1: Turns on echoing; ATE0: Turns off e'choing.
AT+CSQ	Network signal quality query, returns a signal value
AT+CGMR	Request firmware version
AT+CREG?	Request network registration status
AT+CGATT?	Check for GPRS attachment service status

```
AT+CGMR
Revision:1418B08SIM800C24_BT

OK
AT
OK
AT+CPIN?
+CPIN: READY

OK
AT+CSQ
+CSQ: 25,0

OK
AT+CREG?
+CREG: 0,1

OK
AT+CGATT?
+CGATT: 1

OK
```

Étape 4 - TCP/IP Communication

The SIM800 serial module's TCP/IP application offers two connection modes, selectable through the AT command `AT CIPMUX=<n>`.

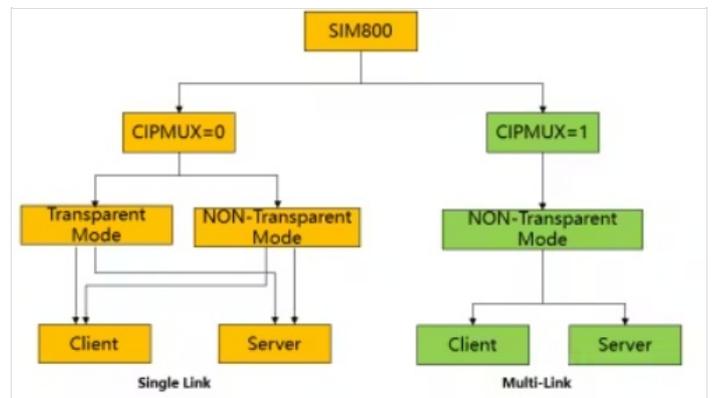
- When `AT CIPMUX` is set to 0 (`AT+CIPMUX=0`), it operates in single-link mode.
- When `AT CIPMUX` is set to 1 (`AT+CIPMUX=1`), it operates in multi-link mode.

By default, the module is configured in single-link mode.

In single-link mode, the SIM800 serial module can function in both transparent and non-transparent transmission modes. In both of these modes, the module can be configured as either a TCP/UDP client or a TCP server.

In multi-link mode, the SIM800 serial module operates solely in non-transparent mode. In this mode, it can serve as a TCP/UDP client, allowing for the establishment of a maximum of 6 connections. It can also be configured as a TCP server, with support for 5 TCP/UDP clients. SIM800C TCP/IP operates with a multi-client architecture by default, enabling up to five sockets for TCP or UDP connections.

In the upcoming demonstrations, our focus will be on the client communication capabilities of the SIM800C module. Specifically, we will explore its operation in single-link non-transparent mode and transparent mode.



Étape 5 - Client communication in non-transparent mode

Client communication in non-transparent mode refers to the way the SIM800C module interacts with remote servers or devices when it acts as a client, transmitting data using a specific protocol such as TCP or UDP. In non-transparent mode, the module sends and receives data through AT commands and does not directly pass data between the microcontroller and the remote server.

This mode provides control over the data transmission process, allowing you to send and receive data, manage connections, and configure communication settings using AT commands. It is suitable for applications where you need fine-grained control over the communication process and want to ensure data integrity.

To use the SIM800C in client communication in non-transparent mode, you will typically configure the module using appropriate AT commands and establish connections with remote servers or devices for data exchange.

```

AT+CGATT?
+CGATT: 1
OK
AT+CSTT="CMNET"
OK
AT+CIICR
OK
AT+CIFSR
10.203.242.202
AT+CIPSTART="TCP","118.190.93.84",2317
OK
CONNECT OK
AT+CIPSEND=28
> Hello,Vaveshare TCP/IP test
SEND OK
Hello,Vaveshare TCP/IP test AT+CIPCLOSE
CLOSE OK
AT+CIPSTART="UDP","118.190.93.84",2317
OK
CONNECT OK
AT+CIPSEND=28
> Hello,Vaveshare TCP/IP test
SEND OK
Hello,Vaveshare TCP/IP test AT+CIPCLOSE
CLOSE OK
AT+CIPSHUT
SHUT OK
    
```

AT Commands	Description	Return
AT+CGATT?	Check GPRS attachment service status	OK
AT+CSTT="CMNET"	Set APN	OK
AT+CIICR	Establish a wireless connection	OK
AT+CIFSR	Retrieve the local IP address	OK
AT+CIPSTART="TCP","118.190.93.84",2317	Establish a TCP client connection	OK
AT+CIPSTART="UDP","118.190.93.84",2317	Establish a UDP client connection	OK
AT+CIPSEND=<string length>	Send a specified string length	OK
AT+CIPCLOSE	Close the connection	OK
AT+CIPSHUT	Close PDP context connection	OK

Étape 6 - Client communication in transparent transmission mode

Client communication in transparent transmission mode refers to the SIM800C module's ability to act as a client while allowing data to flow directly between the microcontroller (e.g., an Arduino) and a remote server or device. In this mode, the SIM800C module operates as a transparent bridge, forwarding data between the microcontroller and the remote server without the need for explicit AT commands to send or receive each piece of data.

This mode simplifies data transfer by treating the SIM800C module as a transparent conduit. Data sent by the microcontroller is transmitted to the remote server without manual packetization, and data received from the server is forwarded to the microcontroller without manual processing.

Using the SIM800C in transparent transmission mode is advantageous when you want to streamline data transfer and reduce the complexity of managing data packets and AT commands for each communication task. It's particularly useful for applications where data throughput and efficiency are essential.

```

AT+CGATT?
+CGATT: 1
OK
AT+CIPMODE=1
OK
AT+CSTT="CMNET"
OK
AT+CIICR
OK
AT+CIFSR
10.101.78.188
AT+CIPSTART="TCP","118.190.93.84",2317
OK
CONNECT
Hello,Vaveshare TCP/IP test
Hello,Vaveshare TCP/IP test
Hello,Vaveshare TCP/IP test
Hello,Vaveshare TCP/IP test
AT
    
```

AT Commands	Description	Return
AT+CGATT?	Check GPRS attachment service status	OK
AT+CIPMODE=1	Set to transparent mode	OK
AT+CSTT="CMNET"	Set APN	OK
AT+CIICR	Establish a wireless connection	OK
AT+CIFSR	Retrieve the local IP address	OK
AT+CIPSTART="TCP","118.190.93.84",2317	Establish a TCP client connection	OK
AT+CIPSTART="UDP","118.190.93.84",2317	Establish a UDP client connection	OK
AT+CIPSEND=<string length>	Send a specified string length	OK
AT+CIPCLOSE	Close the connection	OK
AT+CIPSHUT	Close PDP context connection	OK

Étape 7 - HTTP Communication

This chapter provides an overview of the HTTP communication capabilities of the SIM800C module, focusing on HTTP GET and HTTP POST methods. For in-depth information about HTTP and FTP (File Transfer Protocol) communication with the SIM800C module, please refer to the "SIM800C Series_IP_Application_Note." This additional resource will provide comprehensive details and guidelines for utilizing these communication functions effectively.

- SIM800 Series_AT Command Manual_V1.10
- SIM800 Series_TCPIP_Application Note_V1.02

```

AT+CGATT?
+CGATT: 1
OK
AT+SAPBR=3,1,"Contype","GPRS"
OK
AT+SAPBR=3,1,"APN","CMNET"
OK
AT+SAPBR=1,1
OK
AT+SAPBR=2,1
+SAPBR: 1,1,"10.190.55.120"
OK
AT+HTTPIPINIT
OK
AT+HTTPIPARA="CID",1
OK
AT+HTTPIPARA="URL","www.sim.com"
OK
AT+HTTPIPACT=0
OK
+HTTPIPACT: 0,200,69
AT+HTTPIPREAD
+HTTPIPREAD: 69
<script language=javascript>location.href='espcms/index.php'</script>
OK
AT+HTTPIPTERM
OK
AT+SAPBR=0,1
OK
    
```

AT Commands	Description	Return
AT+SAPBR=3,1,"Contype","GPRS"	Configure bearer profile 1	OK
AT+SAPBR=3,1,"APN","CMNET"	Configure the bearer profile APN	OK
AT+SAPBR=1,1	Open a GPRS context	OK
AT+SAPBR=2,1	Query a GPRS context	OK
AT+SAPBR=0,1	Close the GPRS context	OK
AT+HTTPIPINIT	Check the HTTP connection status	OK
AT+HTTPIPARA="CID",1	Set parameters for HTTP session	OK
AT+HTTPIPARA="URL","www.sim.com"	Set parameters for HTTP session	OK
AT+HTTPIPACT=0	GET session start	OK
AT+HTTPIPREAD	Read the data of the HTTP server	OK
AT+HTTPIPTERM	End HTTP service	OK

```

AT+CGATT?
+CGATT: 1
OK
AT+SAPBR=3,1,"Contype","GPRS"
OK
AT+SAPBR=3,1,"APN","CMNET"
OK
AT+SAPBR=1,1
OK
AT+SAPBR=2,1
+SAPBR: 1,1,"10.166.0.146"
OK
AT+HTTPIPINIT
OK
AT+HTTPIPARA="CID",1
OK
AT+HTTPIPARA="URL","www.sim.com"
OK
AT+HTTPIPDATA=100,10000
DOWNLOAD
OK
AT+HTTPIPACT=1
OK
+HTTPIPACT: 1,200,69
AT+HTTPIPREAD
OK
AT+SAPBR=0,1
OK
    
```

AT Commands	Description	Return
AT+SAPBR=3,1,"Contype","GPRS"	Configure bearer profile 1	OK
AT+SAPBR=3,1,"APN","CMNET"	Configure the bearer profile APN	OK
AT+SAPBR=1,1	Open a GPRS context	OK
AT+SAPBR=2,1	Query a GPRS context	OK
AT+SAPBR=0,1	Close the GPRS context	OK
AT+HTTPIPINIT	Check the HTTP connection status	OK
AT+HTTPIPARA="CID",1	Set parameters for HTTP session	OK
AT+HTTPIPARA="URL","www.sim.com"	Set parameters for HTTP session	OK
AT+HTTPIPDATA=100,10000	Upload 100 bytes data, waiting for 10000ms	OK
AT+HTTPIPACT=0	GET session start	OK
AT+HTTPIPREAD	Read the data of the HTTP server	OK
AT+HTTPIPTERM	End HTTP service	OK

Étape 8 - Establishing Communication Between Arduino Nano, SIM800, and ThingSpeak (Serial Monitor Data Captured During Project Execution).

```

OK
AT+CIICR<CR>
OK
AT+CIFSR<CR>
100.78.215.26
AT+CIPSPRT=0<CR>
OK
AT+CIPSTART="TCP","api.thingspeak.com","80"<CR>
OK

CONNECT OK<CR>AT+CIPSEND<CR>GET https://api.thingspeak.co
m/update?api_key=5XC1TCVONJVK1PNN&field1=20.00&field2=10.00
GET https://api.thingspeak.com/update?api_key=5XC1TCVONJVK1PN
N&<SUB>

SEND OK
6
CLOSED
<CR>AT+CIPSHUT<CR>Temperature = 20.00 °C
Humidity = 10.00 %

SHUT OK
AT<CR>
OK
AT+CPIN?<CR>
+CPIN: READY

OK
AT+CREG?<CR>
+CAT+CSTT="airtelgprs.com"<CR>
OK
AT+CIICR<CR>
OK
AT+CIFSR<CR>
100.90.199.122
AT+CIPSPRT=0<CR>
OK
AT+CIPSTART="TCP","api.thingspeak.com","80"<CR>
OK

CONNECT OK<CR>AT+CIPSEND<CR>GET https://api.thingspeak.co
m/update?api_key=5XC1TCVONJVK1PNN&field1=20.00&field2=10.00
GET https://api.thingspeak.com/update?api_key=5XC1TCVONJVK1PN
N&<SUB>

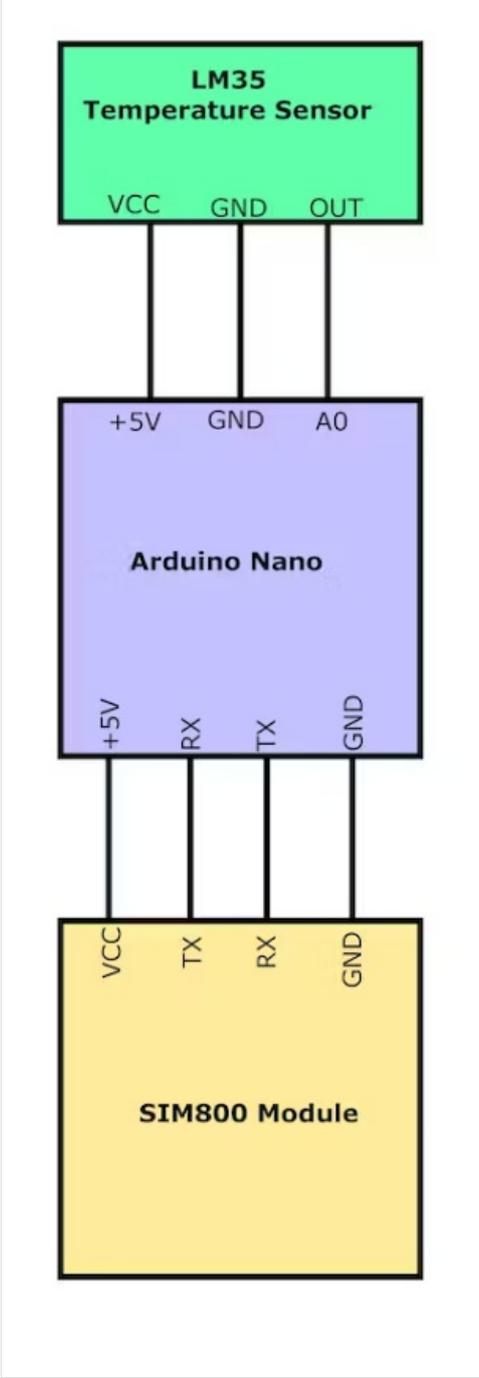
SEND OK
7
CLOSED
<CR>AT+CIPSHUT<CR>Temperature = 20.00 °C
Humidity = 10.00 %

SHUT OK
AT<CR>
OK
AT+CPIN?<CR>
+CPIN: READY

OK
AT+CREG?<CR>
+CAT+CSTT="airtelgprs.com"<CR>
OK
AT+CIICR<CR>
OK
AT+CIFSR<CR>
100.102.108.63
AT+CIPSPRT=0<CR>
OK
AT+CIPSTART="TCP","api.thingspeak.com","80"<CR>
OK

CONNECT OK<CR>AT+CIPSEND<CR>GET https://api.thingspeak.co
m/update?api_key=5XC1TCVONJVK1PNN&field1=20.00&field2=10.00
GET https://api.thingspeak.com/update?api_key=5XC1TCVONJVK1PN
N&<SUB>

```



SEND OK
8
CLOSED
