Автоматическая заправочная станция

на базе контроллера ESP32 LILYGO T-SIM7070G

Оглавление

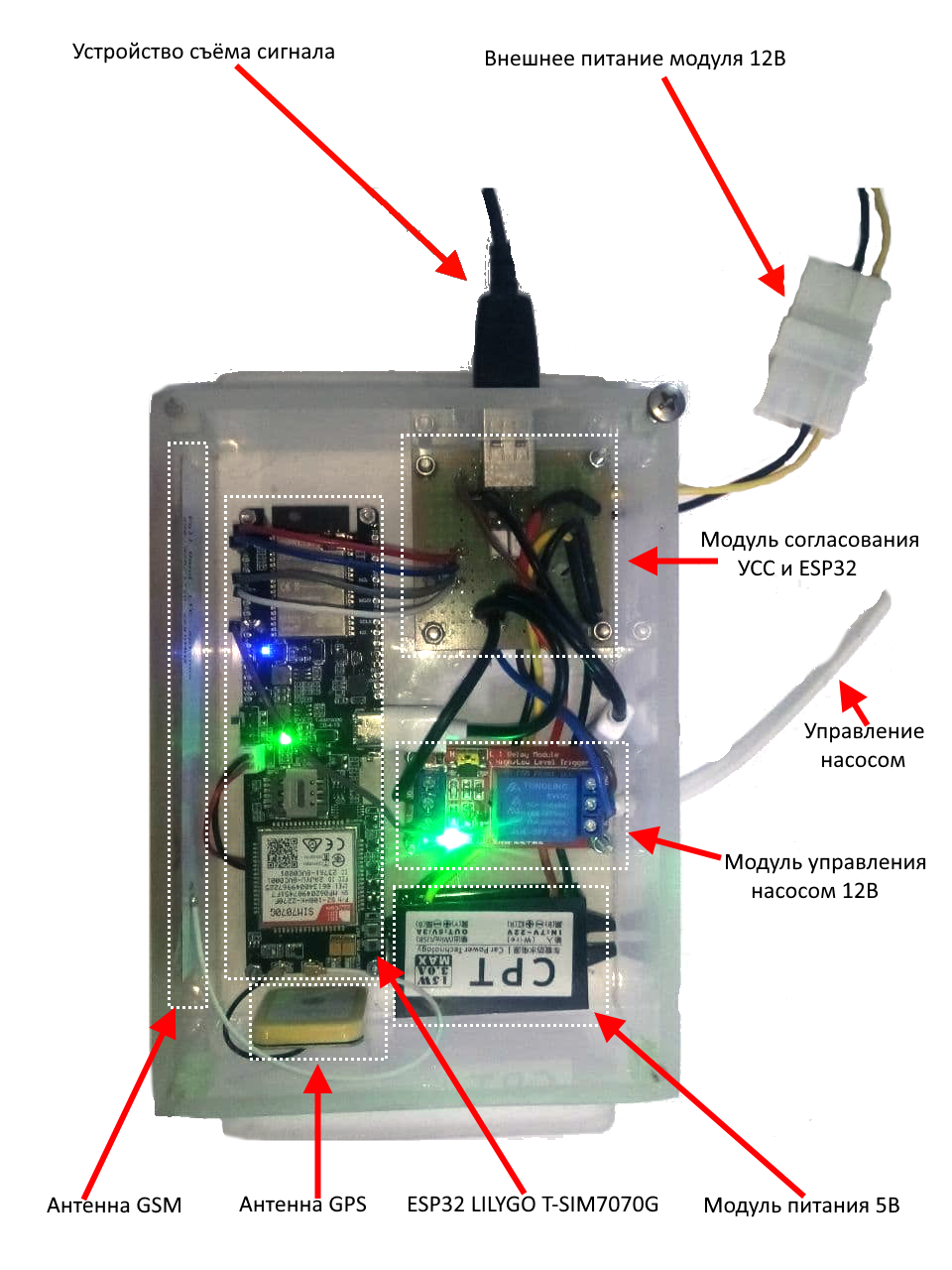
[Состав и расположение модулей АЗС 3](#_Toc92624785)

[Схема модуля согласования УСС и ESP32 4](#_Toc92624786)

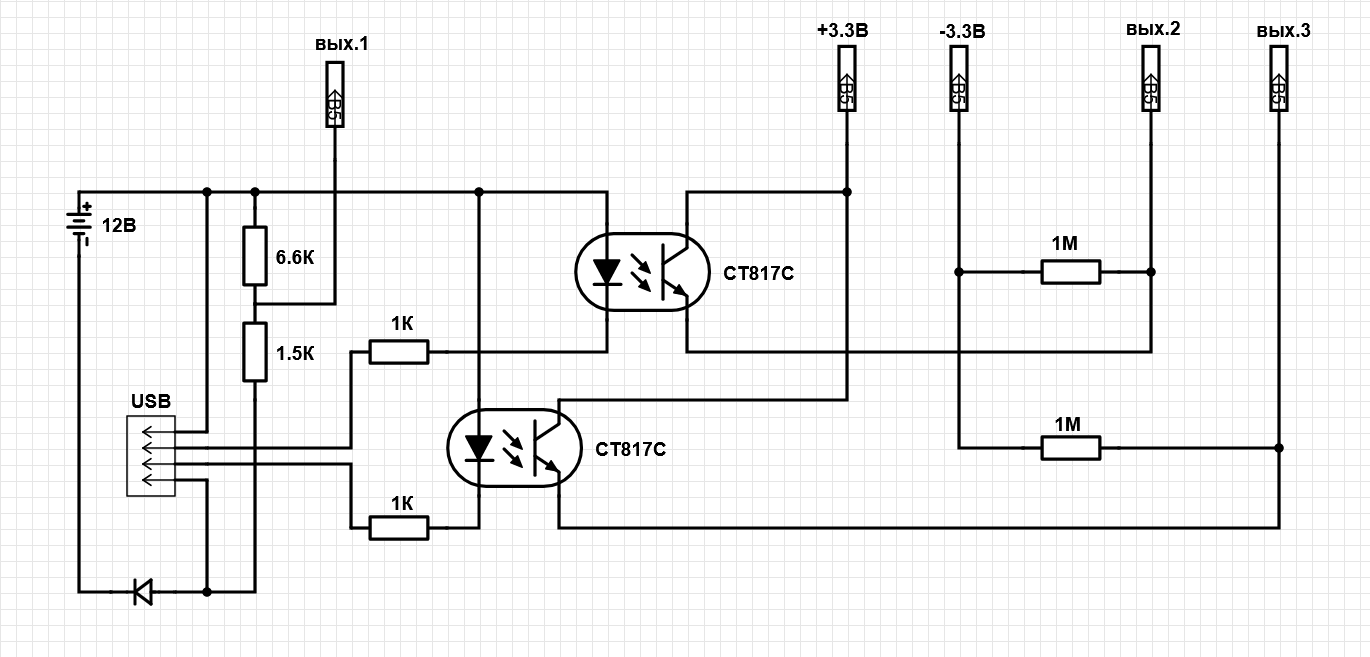
[Блок схема микроконтроллера ESP32 5](#_Toc92624787)

[Листинг кода ESP32 6](#_Toc92624788)

# Состав и расположение модулей АЗС



# Схема модуля согласования УСС и ESP32



Ссылка на электронную версию:

<https://www.digikey.com/schemeit/project/заправочная-станция-модуль-согласования-усс-и-esp32-83c7fa265f3d4c3e819c508f970e47d7>

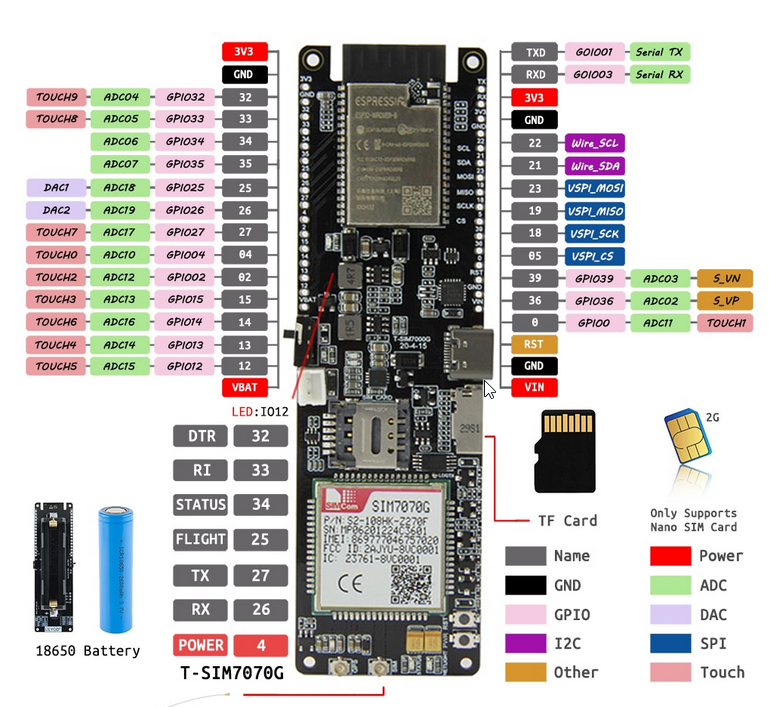
# ESP32 LILYGO T-SIM7070G

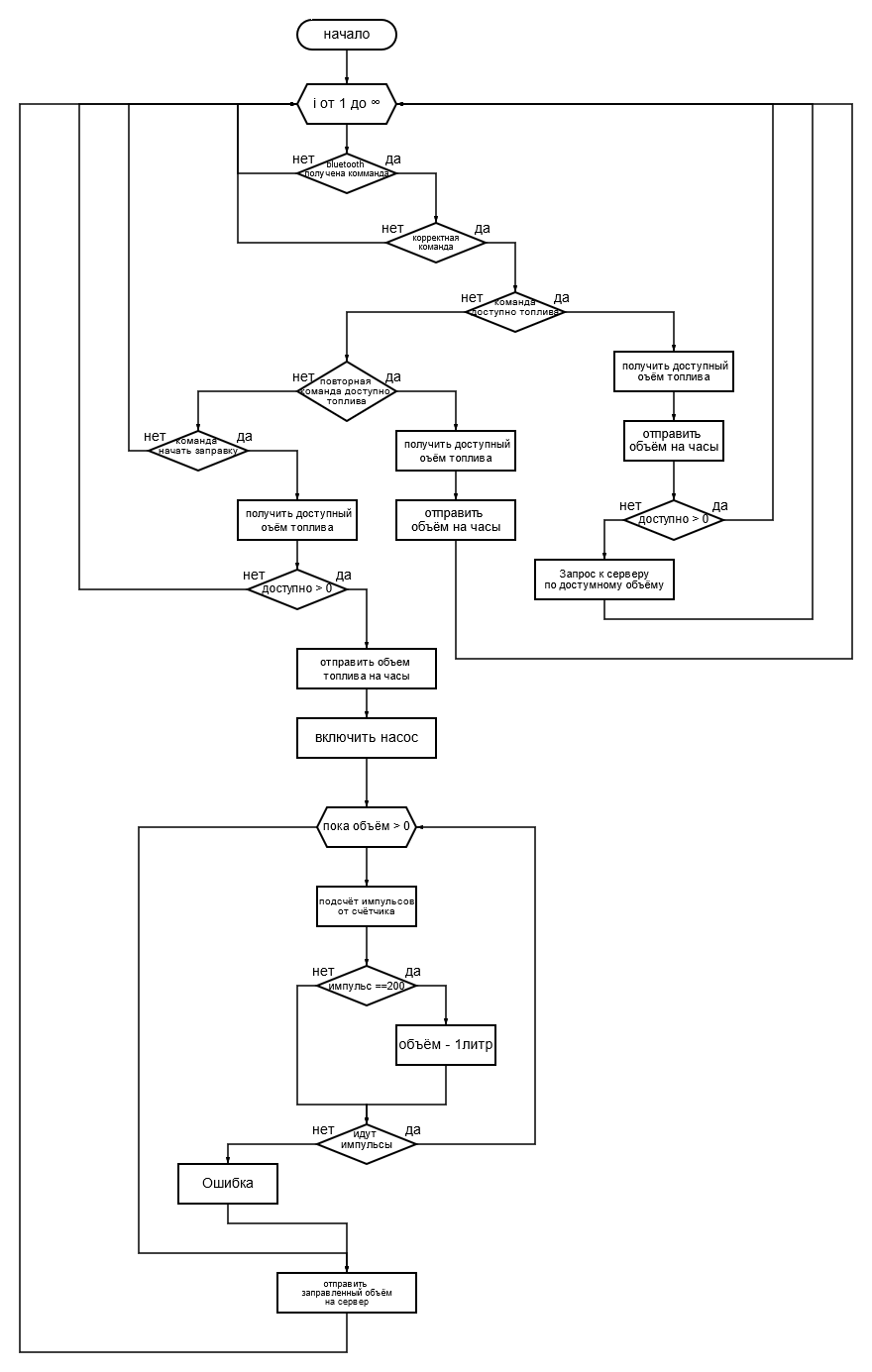
Плата на базе микроконтроллера ESP32, содержит на борту GSM, GPRS, Wi-Fi, Bluetooth и сетевой модуль SIM7070G поддерживающий протоколы Cat-M/ Cat-NB/GPRS/EDGE, может работать GSM частототах 850/900/1800/1900

Скетчи рекомендованные производителем доступны с GitHub: https://github.com/orgs/Xinyuan-LilyGO/repositories

Напряжение питания 5В, ток потребления в пике не более 100мА

Внешний вид и распиновка:



Блок схема микроконтроллера ESP32 

Редактор: https://programforyou.ru/block-diagram-redactor

# Листинг кода ESP32

#include "BluetoothSerial.h"

#include "esp\_bt\_device.h"

#include <WiFiMulti.h>

const String systemId **=** "0"**;**

const int batteryVoltagePin **=** 36**;**

const int portPin1 **=** 34**;**

const int portPin2 **=** 35**;**

const int portTrigger **=** 27**;**

const int pulsePerLiter **=** 200**;** //number of pulses per liter

const int preStopTrigger **=** 25**;** //min value = 0; max value "pulsePerLiter"; 1 = 10ml

int pulse **=** 0**;**

int level **=** 0**;**

int portValue **=** 0**;**

int waitingSignal **=** 15**;** //waiting for signal in seconds

const char**\*** ssid **=** "wifi"**;**

const char**\*** password **=** "0123456789"**;**

int retryFindWiFi **=** 5**;**

const uint16\_t port **=** 80**;**

const char **\*** host **=** "erpelement.ru"**;** // ip or dns

BluetoothSerial SerialBT**;**

WiFiClient client**;**

WiFiMulti WiFiMulti**;**

String incoming**;**

int LED\_BUILTIN **=** 12**;**

struct Command

**{**

String GetfuelVolume **=** "0"**;**

String GetfuelVolumeRetry **=** "1"**;**

String StartFuelFill **=** "2"**;**

**};**

struct ClientFuelInfo

**{**

String id**;**

int fuel**;**

**};**

struct Command Command**;**

struct ClientFuelInfo ClientFuelInfo**[**1000**];**

void setup**()** **{**

pinMode**(**batteryVoltagePin**,** INPUT**);**

pinMode**(**portPin1**,** INPUT**);**

pinMode**(**portPin2**,** INPUT**);**

pinMode**(**portTrigger**,** OUTPUT**);**

digitalWrite**(**portTrigger**,** LOW**);**

pinMode **(**LED\_BUILTIN**,** OUTPUT**);**// задаем контакт подключения светодиода как выходной

digitalWrite **(**LED\_BUILTIN**,** LOW**);**

Serial**.**begin**(**2000000**);** // Запускаем последовательный монитор

Serial**.**println**(**"\n"**);**

WiFiMulti**.**addAP**(**ssid**,** password**);**

getFuelInfo**(**"Start"**);**

SerialBT**.**begin**(**"АЗС"**);** // Задаем имя вашего устройства Bluetooth

SerialBT**.**register\_callback**(**callback**);**

printDeviceAddress**();**

**}**

void loop**()** **{**

delay**(**50**);** // millisecond timeout

digitalWrite **(**LED\_BUILTIN**,** LOW**);**

delay**(**50**);** // millisecond timeout

digitalWrite **(**LED\_BUILTIN**,** HIGH**);**

**if** **(**SerialBT**.**available**())** // Проверяем, не получили ли мы что-либо от Bluetooth модуля

**{**

Serial**.**setTimeout**(**100**);** // 100 millisecond timeout

incoming **=** SerialBT**.**readString**();**

Serial**.**println**(**"<<< " **+** incoming**);**

int str\_len **=** incoming**.**length**()** **+** 1**;**

char char\_array**[**str\_len**];**

incoming**.**toCharArray**(**char\_array**,** str\_len**);**

char **\***str**;**

char **\***p **=** char\_array**;**

int curr**=**0**;**

String deviceUniqueID**=**""**,** command**=**""**,** value**=**""**;**

**while** **((**str **=** strtok\_r**(**p**,** "|"**,** **&**p**))** **!=** **NULL)**

**{**

**if(**curr **==** 0**)** deviceUniqueID **=** String**(**str**);**

**if(**curr **==** 1**)** command **=** String**(**str**);**

**if(**curr **==** 2**)** value **=** String**(**str**);**

curr**++;**

**}**

**if** **(**curr **==** 3 **)**

**{**

**if(**command **==** Command**.**GetfuelVolume**)**

**{**

int fuelVolume **=** GetFuelVolume**(**deviceUniqueID**);**

SerialBT**.**println**(**deviceUniqueID **+** "|0|" **+** fuelVolume**);**

Serial**.**println**(**">>> " **+** deviceUniqueID **+**"|0|"**+** fuelVolume**);**

**if(**fuelVolume **==** 0**)**

**{**

getFuelInfo**(**deviceUniqueID**);**

**}**

**}**

**if(**command **==** Command**.**GetfuelVolumeRetry**)**

**{**

int fuelVolume **=** GetFuelVolume**(**deviceUniqueID**);**

SendCommand**(**deviceUniqueID**,** command**,** String**(**fuelVolume**));**

**}**

**if(**command **==** Command**.**StartFuelFill**)**

**{**

**for** **(**int i**=**0**;** i **<** **(sizeof** ClientFuelInfo**/sizeof** ClientFuelInfo**[**0**]);** i**++)**

**{**

**if(**ClientFuelInfo**[**i**].**id **==** deviceUniqueID **&&** ClientFuelInfo**[**i**].**fuel **>** 0**)**

**{**

SerialBT**.**println**(**ClientFuelInfo**[**i**].**id **+**"|"**+**Command**.**StartFuelFill**+**"|"**+** ClientFuelInfo**[**i**].**fuel**);**

Serial**.**println**(**">>> " **+** ClientFuelInfo**[**i**].**id **+**"|"**+**Command**.**StartFuelFill**+**"|"**+** ClientFuelInfo**[**i**].**fuel**);**

int fuelBegin **=** ClientFuelInfo**[**i**].**fuel**;**

digitalWrite**(**portTrigger**,** HIGH**);**

uint32\_t tmr **=** micros**();**

**while(**ClientFuelInfo**[**i**].**fuel **>** 0**)**

**{**

portValue **=** analogRead**(**portPin1**);**

**if(**level **==** 0 **&&** portValue **==** 4095**)**

**{**

level **=** 1**;**

pulse**++;**

tmr **=** micros**();**

**}**

**if(**level **==** 1 **&&** portValue **<** 4095**)**

**{**

level **=** 0**;**

pulse**++;**

tmr **=** micros**();**

**}**

**if(**pulse **==** pulsePerLiter **||** **(**ClientFuelInfo**[**i**].**fuel **==** 1 **&&** pulse **==** **(**pulsePerLiter **-** preStopTrigger**)))**

**{**

pulse **=** 0**;**

ClientFuelInfo**[**i**].**fuel**--;**

Serial**.**println**(**">>> " **+** ClientFuelInfo**[**i**].**id **+** "|" **+** Command**.**StartFuelFill**+**"|" **+** ClientFuelInfo**[**i**].**fuel**);**

SerialBT**.**println**(**ClientFuelInfo**[**i**].**id **+** "|" **+** Command**.**StartFuelFill **+** "|" **+** ClientFuelInfo**[**i**].**fuel**);**

**}**

**if(**micros**()** **-** tmr **>** waitingSignal**\***1000000**)**

**{**

Serial**.**println**(**"Выход. Нет ответа от счётчика"**);**

**break;**

**}**

**}**

digitalWrite**(**portTrigger**,** LOW**);**

Serial**.**println**(**">>> " **+** ClientFuelInfo**[**i**].**id **+**"|"**+**Command**.**StartFuelFill**+**"|"**+** ClientFuelInfo**[**i**].**fuel**);**

SerialBT**.**println**(**ClientFuelInfo**[**i**].**id **+**"|"**+**Command**.**StartFuelFill**+**"|"**+** ClientFuelInfo**[**i**].**fuel**);**

setFuelInfo**(**ClientFuelInfo**[**i**].**id**,** fuelBegin **-** ClientFuelInfo**[**i**].**fuel**);**

ClientFuelInfo**[**i**].**fuel**=**0**;**

**break;**

**}**

**}**

**}**

**}**

**}**

**}**

void SendCommand**(**String deviceUniqueID**,** String command**,** String value**)**

**{**

SerialBT**.**println**(**deviceUniqueID **+** "|" **+** command **+** "|" **+** value**);**

Serial**.**println**(**">>> " **+** deviceUniqueID **+** "|" **+** command **+** "|" **+** value**);**

**}**

/\*Возвращает доступный объём топлива из массива\*/

int GetFuelVolume**(**String deviceUniqueID**)**

**{**

**for** **(**int i **=** 0**;** i **<** **(sizeof** ClientFuelInfo**/sizeof** ClientFuelInfo**[**0**]);** i**++)**

**{**

**if(**ClientFuelInfo**[**i**].**id **==** deviceUniqueID **&&** ClientFuelInfo**[**i**].**fuel **>** 0**)**

**{**

**return** ClientFuelInfo**[**i**].**fuel**;**

**}**

**}**

**return** 0**;**

**}**

void getFuelInfo**(**String id**)**

**{**

String response **=** GetRestResponse**(**"api/system/getFuelInfo.php?systemId=" **+** systemId **+** "&id=" **+** id **+** "&batteryVoltage=" **+** analogRead**(**batteryVoltagePin**));**

**if(**response**.**length**()** **>** 0**)**

**{**

//set empty ClientFuelInfo array ToDo refactor

**for** **(**int i **=** 0**;** i **<** **(sizeof** ClientFuelInfo**/sizeof** ClientFuelInfo**[**0**]);** i**++)**

**{**

ClientFuelInfo**[**i**].**id **=** ""**;**

ClientFuelInfo**[**i**].**fuel **=** 0**;**

**}**

int i **=** 0**;**

String subStr **=** ""**;**

**for** **(**auto c **:** response**)**

**{**

**if(**c**==**';'**)**

**{**

String id**=**""**;**

String fuel**=**""**;**

int index **=** subStr**.**indexOf**(**'|'**);**

ClientFuelInfo**[**i**].**id **=** subStr**.**substring**(**0**,** index**);**

ClientFuelInfo**[**i**].**fuel **=** subStr**.**substring**(**index **+** 1**,**subStr**.**length**()).**toInt**();**

printf**(**"curr=%i id=%s fuel=%i\n"**,** i**,** ClientFuelInfo**[**i**].**id**.**c\_str**(),** ClientFuelInfo**[**i**].**fuel**);**

i**++;**

subStr **=** ""**;**

**}else{**

subStr **+=** c**;**

**}**

**}**

**}**

**}**

void setFuelInfo**(**String id**,** int fuel**)**

**{**

String response **=** GetRestResponse**(**"api/system/setFuelInfo.php?systemId=" **+** systemId **+** "&id=" **+** id **+** "&fuel=" **+** fuel**);**

**}**

void callback**(**esp\_spp\_cb\_event\_t event**,** esp\_spp\_cb\_param\_t **\***param**){**

**if(**event **==** ESP\_SPP\_SRV\_OPEN\_EVT**){**

Serial**.**println**(**"Client Connected"**);**

**}**

**if(**event **==** ESP\_SPP\_CLOSE\_EVT **){**

Serial**.**println**(**"Client disconnected"**);**

digitalWrite**(**LED\_BUILTIN**,** LOW**);**

**}**

**}**

void printDeviceAddress**()** **{**

Serial**.**print**(**"Bluetooth: "**);**

const uint8\_t**\*** point **=** esp\_bt\_dev\_get\_address**();**

**for** **(**int i **=** 0**;** i **<** 6**;** i**++)** **{**

char str**[**3**];**

sprintf**(**str**,** "%02X"**,** **(**int**)**point**[**i**]);**

Serial**.**print**(**str**);**

**if** **(**i **<** 5**){**

Serial**.**print**(**":"**);**

**}**

**}**

Serial**.**println**(**""**);**

**}**

String GetRestResponse**(**String getMethod**)**

**{**

String response **=** ""**;**

Serial**.**print**(**"Waiting for WiFi"**);**

int retry **=** 0**;**

**while(**WiFiMulti**.**run**()** **!=** WL\_CONNECTED**)** **{**

Serial**.**print**(**"."**);**

**if** **(**retry **>** retryFindWiFi**)**

**{**

Serial**.**println**(**"\nError Find WiFi"**);**

WiFi**.**mode**(**WIFI\_OFF**);**

**return** ""**;**

**}**

retry**++;**

delay**(**1000**);**

**}**

// begin region modem connect to 3g

**if** **(!**client**.**connect**(**"192.168.1.1"**,** 80**))**

**{**

Serial**.**println**(**"Connection modem failed."**);**

**}else{**

client**.**println**(**"GET /goform/setWanConnect?profile\_id=25099\_1&profile\_type=0 HTTP/1.1"**);**

client**.**println**(**"Host: " **+** String**(**"192.168.1.1"**)** **+** " \n\n"**);**

int currLoop **=** 0**;**

**while** **(!**client**.**available**()** **&&** currLoop**++** **<** 3000**)**

**{**

delay**(**10**);**

**}**

**if** **(**client**.**available**()** **>** 0**)**

**{**

**while(**client**.**available**()** **>** 0**)**

**{**

response **=** client**.**readStringUntil**(**'\n'**);**

**}**

Serial**.**println**(**"modem response: " **+** response**);**

**}**

**}**

// end region modem connect to 3g

Serial**.**println**(**"\nWiFi connected. IP address: " **+** WiFi**.**localIP**().**toString**());**

Serial**.**println**(**"Connecting to GET: " **+** String**(**host**)** **+** "/" **+** getMethod**);**

**if** **(!**client**.**connect**(**host**,** port**))** **{**

Serial**.**println**(**"Connection failed."**);**

client**.**stop**();**

WiFi**.**mode**(**WIFI\_OFF**);**

**return** ""**;**

**}**

client**.**println**(**"GET /" **+** getMethod **+** " HTTP/1.1"**);**

client**.**println**(**"Host: " **+** String**(**host**)** **+** " \n\n"**);**

int currLoop **=** 0**;**

**while** **(!**client**.**available**()** **&&** currLoop**++** **<** 3000**)**

**{**

delay**(**10**);**

**}**

**if** **(**client**.**available**()** **>** 0**)**

**{**

**while(**client**.**available**()** **>** 0**)**

**{**

response **=** client**.**readStringUntil**(**'\n'**);**

**}**

Serial**.**println**(**response**);**

**}**

**else**

**{**

Serial**.**println**(**"client.available() timed out "**);**

client**.**stop**();**

WiFi**.**mode**(**WIFI\_OFF**);**

**return** ""**;**

**}**

Serial**.**println**(**"Closing connection."**);**

client**.**stop**();**

WiFi**.**mode**(**WIFI\_OFF**);**

**return** response**;**

**}**