

คู่มือการใช้งาน

# ET-BASE GSM SIM900



บริษัท อีทีที จำกัด  
ETT CO., LTD.

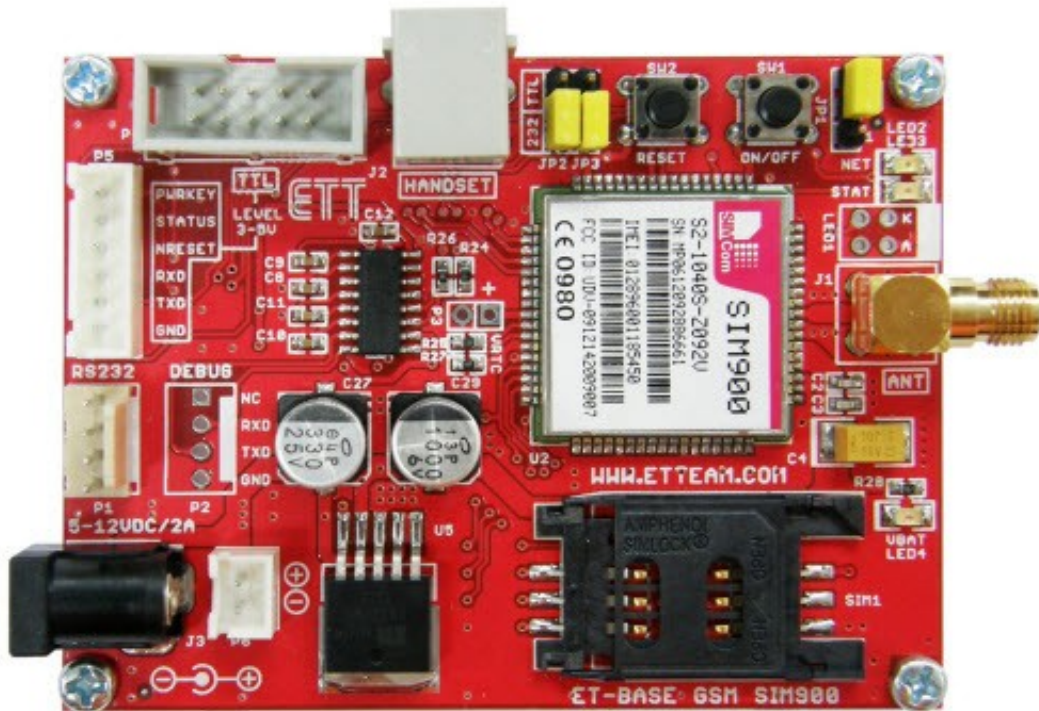


<http://www.etteam.com>

## CONTENTS

1. Specifications of Board ET-BASE GSM SIM900	3
2. Initial Specifications of Module SIM900	3
3. Compositions of Board ET-BASE GSM SIM900	4
4. How to Enable/Disable Operation of Module SIM900	6
5. How to Communicate with Module SIM900	7
6. Noteworthy Specifications of Signal	9
7. Example of using AT Command with Module SIM900	9
8. How to Test Operation of Board ET-BASE GSM SIM900	10
9. Example of Using Command about RS232	13
10. How to Setup and Check Configuration	15
11. How to Check Data of Module SIM900	16
12. How to Make, Receive, and Cancel a Call	17
13. How to Check Balance by USSD	18
14. How to Receive SMS	19
15. How to Send English SMS	20
16. Code of Thai SMS	21
17. Principles of Decoding Unicode	23
18. How to Send Thai SMS	26
19. How to Use SIM Command (SIM Application Toolkit: STK)	28
20. How to Read Data from Website by GPRS Connection (HTTP GET)	32
21. How to Connect Board ET-BASE GSM SIM900 with Board Microcontroller	34

# ET-BASE GSM SIM900



**ET-BASE GSM SIM900** is a kit to learn and develop wireless communication by using Module GSM/GPRS model SIM900 from **SIMCom** to be main device; this SIM900 is a small GSM/GPRS Module that supports GSM Frequency in the range of 850/900/1800/1900MHz. It communicates through Port RS232 by AT Command; it can be applied for various applications such as transmitting/receiving signal in the format of Voice, SMS, Data, FAX, including Protocol TCP/IP Communication. Normally, it provides the Circuit and Firmware internal Module SIM900 completely but it is not ready to use because user has to design the circuit of Peripheral device that is necessary to connect with some partial Pin of Module such as Circuit Power Supply, circuit for connecting with SIM Card, including Circuit Line Driver of RS232. So, ETT Team provides the intermediate board to connect between Module SIM900 and external device, it assists customer to learn and test the operation conveniently before modifying and adapting this Module for any application in the future. Although all connective circuits that are provided by ETT Team cannot support all resources internal Module, it can support the main capability of this Module efficiently.

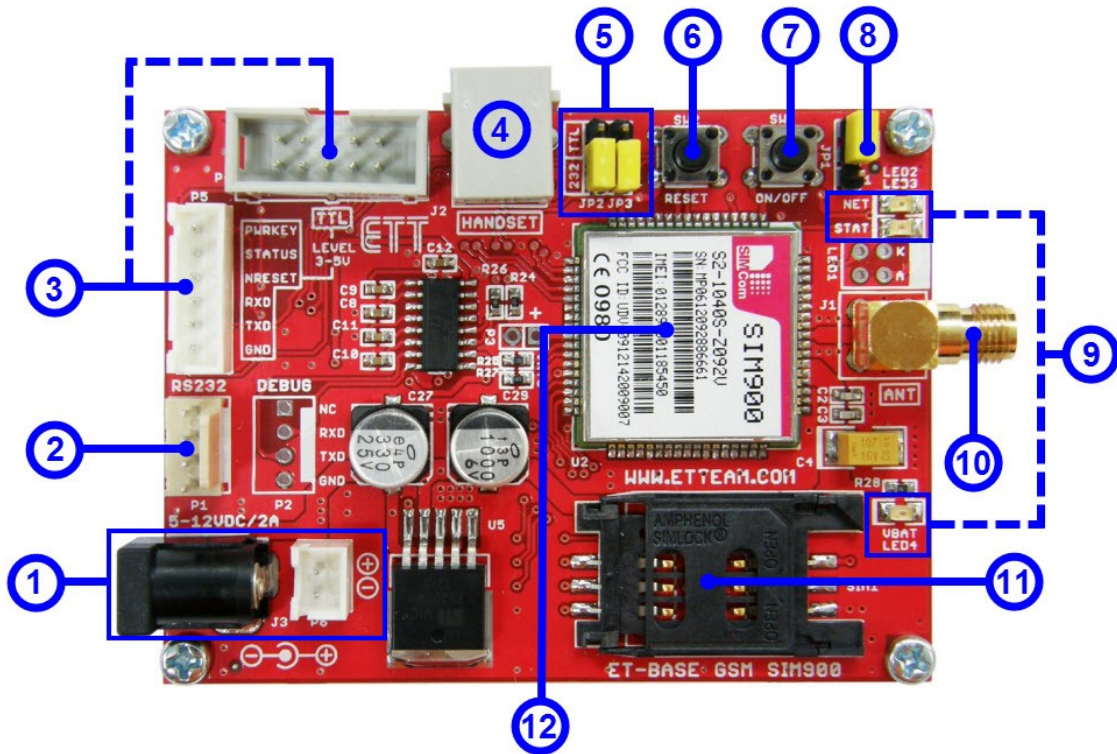
## 1. Specifications of Board ET-BASE GSM SIM900

- Has Push-Button Switch to Enable/Disable the operation of Module internal board
- Has Push-Button Switch to reset the operation of Module internal board
- Has Socket SIM to support SIM Card with Circuit ESD to protect SIM from damaged
- Has 2 of isolated Regulates that can be used with Adapter 5-12VDC; it can provide current for Module SIM900 and connective devices enough.
  - Has Circuit Regulate 4.2V/3A that can supply to Module SIM900 enough, it can be used with SIM of GSM900MHz 2-Watt without any problem
  - Has Circuit Regulate 2.8V/150mA to supply to the circuit that converts Signal Logic.
- Has Circuit Line Driver to convert Signal Logic from Module SIM900 into RS232 (1200bps-115200bps) for port communication that is used to command Module.
- Has circuit to convert Signal Logic TTL 3V-5V to directly connect with Microcontroller, without connecting through Circuit Line Driver RS232
- Has LED to display the ready status of board; status of Power Supply, status of Module, status of Network Connection, and status of Power-ON/Power-OFF of Module
- Has Connector to connect with Handset (microphone and speaker of home phone); it uses Connector Standard RJ11with Circuit Voice Filter. In this case, it can interface Handset of home phone with board through Connector RJ11 to make and receive a call conveniently.

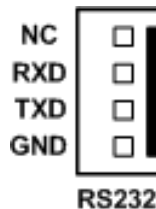
## 2. Initial specifications of Module SIM900

- Support Frequency in the range of GSM/GPRS 850/900/1800/1900MHz
- Support GPRS Multi-Slot Class10 and GPRS Mobile Station Class B
- Support standard AT Command (GSM 07.07/ 07.05 and additional commands from SIMCOM)
- Support SIM Applications Toolkit
- Run by the Frequency in the range of 3.2V to 4.8V
- Support external connection
  - Be compatible with SIM Card 1.8V and 3V
  - Has Circuit Analog Audio (MIC & Speaker)

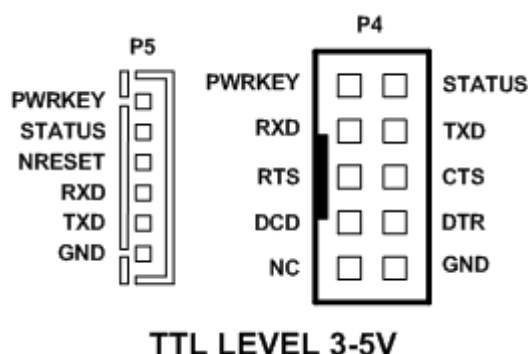
### 3. Compositions of Board ET-BASE GSM SIM900



- **No.1:** It is Connector Power Supply for board, there are 2 types. Firstly, it is DC JACK that is Anode-OUT(+) and Cathode-IN(-); secondly, It is Connector JST. The Power Supply that provides for board is 5-12VDC with at least 2A Current.
- **No.2:** It is Connector RS232 4PIN (ETT Standard); it interfaces with device that transmits/receives data by RS232 such as computer or Board Microcontroller, especially be connected through Circuit Line Driver RS232. It arranges signal pins as shown in the picture below;



- **No.3:** It is Connector TTL 3.5V to connect with Microcontroller directly, without interfacing through Circuit Line Driver RS232. It arranges signal pins as shown in the picture below;



Pin Name	Direction	Description
<b>PWRKEY</b>	<b>INPUT</b>	Control ON/OFF Module SIM900, it runs by Logic "1"
<b>STATUS</b>	<b>OUTPUT</b>	Report that status of Module SIM900 is active. If it is Logic "1", the Module is active; or, if it is Logic "0", the Module is inactive or it is the status of Power Down.
<b>NRESET</b>	<b>INPUT</b>	Reset operation of Module SIM900, it runs by Logic "1"
<b>RXD</b>	<b>INPUT</b>	Receive data
<b>TXD</b>	<b>OUTPUT</b>	Transmit data
<b>RTS</b>	<b>INPUT</b>	Request to send
<b>CTS</b>	<b>OUTPUT</b>	Clear to send
<b>DCD</b>	<b>OUTPUT</b>	Data carrier detect
<b>DTR</b>	<b>INPUT</b>	Data terminal ready
<b>NC</b>	-	Unused
<b>GND</b>		Ground

- **No.4:** It is Connector RJ11 to connect with Handset in order to use the Module SIM900 to make and receive a call. It can interface with general standard of Handset.
- **No.5:** It is Jumper to choose if user requires connecting Pin RXD, TXD of Module through Circuit Line Driver RS232. If user requires connecting pins through Connector RS232, it has to set Jumper JP2 and JP3 to the position of 232; or, if user requires connecting pins through Connector TTL P4, P5, it has to set Jumper JP2 and JP3 to the position of TTL.
- **No.6:** It is Push-Button Switch to reset the operation of Module.
- **No.7:** It is Push-Button Switch to Power-ON and Power-OFF the Module.
- **No.8:** It is Jumper to auto-enable the operation of Module SIM900 immediately after supplying power into board; it has to set Jumper to the position of **AT**. If user requires using Switch ON/OFF or Pin PWRKEY to control ON/OFF, it has to set this Jumper to the position of Pin 1-2.

- **No.9:** It is LED to display the operating status of board as described below;
  - ❖ **LED VBAT:** It displays status of external Power Supply that is connected with board. This LED is lit up when it supplies power into board completely.
  - ❖ **LED NET (NETLIGHT):** It displays status of Module while connecting with network. This LED is controlled by Signal NETLIGHT (PIN 52) of Module SIM900. It is Logic "1" when it is active. When the Module is ready to operate, this LED flashes with different speeds as described below;
    - **OFF:** Module is in the status of Power OFF (inactive).
    - **64mS ON/ 800mS OFF:** Module SIM900 cannot find out any network.
    - **64mS ON/ 3000mS OFF:** Module SIM99 found the network successfully.
    - **64mS ON/ 300mS OFF:** Module SIM900 is in the progress of connecting with network or other devices by GPRS.
  - ❖ **LED STAT (STATUS):** It shows whether the status of Module SIM900 is active. If LED is lit up, it means that the Module is active; or, if this LED is off, it means that this Module is inactive or, it is in the status of Power Down Mode.
- **No.10:** It is Connector Antennae GSM with the Frequency in the range of 850/900/1800/1900MHz.
- **No.11:** It is Socket to install SIM Card on Module.
- **No.12:** It is Module SIM900.

#### 4. How to Enable/Disable Operation of Module SIM900

Normally, Module SIM900 has several modes to enable and disable the operation of Module as described below;

- 4.1 **Switch ON/OFF (SW1):** It enables/disables the operation of Module SIM900 by pressing Push-Button Switch. It sets Logic status for Pin PWRKEY (PIN 1) of Module; it is Logic "0" when it presses the switch; or, it is Logic "1" when it releases the Switch. Remember, user has to continue pressing the Switch more than 1000mS (1 second) and it has an effect on the operation of Module. This Switch runs in the format of Toggle; if the Module is in the status of Power OFF and user presses and holds the Switch more than 1000mS (1 second), it commands the Module return to the status of Power ON that is ready to operate. Or, if the Module is in the status of Power ON and user presses and holds the Switch more than 1000mS (1 second) and then releases the Switch, it commands the Module to stop running and return to the status of Power OFF (stop running). Status of LED is listed in the Table below;

LED Status	Power-ON	Power-OFF
VBAT (GREEN)	ON	ON
NET (ORANGE)	FLASH	OFF
STAT (GREEN)	ON	OFF

**Table shows status of LED in various modes.**

**4.2 Control On/OFF by Pin PWRKEY:** It enables/disables the operation of this Module by external signal such as signal from Microcontroller; it is connected through Pin **PWRKEY** (Connector P4 or P5). It runs in the format of Toggle; if the Module is in the status of Power OFF and it sends Logic "1" more than 1000mS (1 second) and then it becomes Logic "0", it commands the Module to return the status of Power ON that is ready to run. Or, if the Module is in the status of Power ON and it sends Logic "1" more than 1000mS (1 second) and then it becomes Logic "0", it commands the Module to stop running and return to the status of Power OFF (stop running).

**4.3 Auto ON/OFF:** It auto enables the operation of Module SIM900 immediately after supplied power into Board ET-BASE GSM SIM900 completely; in this case, it has to set Jumper **JP1** to the position of **AT**.

**5. How to communicate with Module SIM900**

There are 2 types to communicate with Board ET-BASE GSM SIM900. Firstly, it is connected through RS232 Serial Port by Connector 4PIN that is arranged according to ETT Standard; it can be connected with standard Signal RS232 such as computer RS232 (Com Port) or ETT Board Microcontroller that is Connector RS232 4PIN instantly. Secondly, this Board ET-BASE GSM SIM900 provides Connector TTL 3-5V (P4 or P5) to interface with Board Microcontroller directly, without using any Circuit to convert the signal level to be RS232. Signals that are used to serially connect with Module SIM900 are listed as follows;

- **DCD(Data Carrier Detect)** of Module SIM900 is Output from SIM900; normally, it is interfaced with DCD Input of the device on the Host side.
- **TXD(Transmit Data)** of Module SIM900 is Output from SIM900; normally, it is interfaced with RXD(Receive Data) of the device on the Host side.
- **RXD(Receive Data)** of Module SIM900 is Input from SIM900; normally, it is interfaced with TXD(Transmit Data) of the device on the Host side.



- **DTR(Data Terminal Ready)** of Module SIM900 is Input from SIM900; normally, it is interfaced with DTR of the device on the Host side.
- **RTS(Request To Send)** of Module SIM900 is Input from SIM900; normally, it is interfaced with RTS of the device on the Host side.
- **CTS(Clear To Send)** of Module SIM900 is Output from SIM900; normally, it is interfaced with CTS of the device on the Host side.
- **RI(Ring Indicator)** of Module SIM900 is Output from SIM900; normally, it is interfaced with RI of the device on the Host side.
- **GND** of Module SIM900 must be interfaced with GND of the device on the Host side.

It illustrates how to interface Signals between ET-BASE GSM SIM900 and Microcontroller.

SIM900	Signal Direction	MCU
DCD	→	DCD
TXD	→	RXD
RXD	←	TXD
DTR	←	DTR
RTS	←	RTS
CTS	→	CTS
RI	→	RI
GND	—	GND

Table shows the full connection.

SIM900	Signal Direction	MCU
TXD	→	RXD
RXD	←	TXD
GND	—	GND

Table shows the connection by 3 Cables.

**NOTE:** In case of using 3 Cables for connection (RXD, TXD, GND), it has to setup the condition of Flow Control for Module SIM900 as *No flow control* by using the Command **"AT+IFC=0,0"**, or *XON/XOFF* by using the Command **"AT+IFC=1,1"**.

## 6. Noteworthy Specifications of Signal

- **RI(Ring Indicator)** is Output from Module SIM900; normally, it is High, but it becomes Active Low when there is an incoming telephone call according to these conditions;
  - When there is any Voice Calling, Signal RI is Active LOW and it still remains until it answers a calling (ATA) or it receives a command to cancel a calling (ATH) or it hangs up a telephone before answering.
  - When there is any Data calling, Signal RI is Active LOW and it still remains until it answers a calling (ATA) or it receives a command to cancel a calling (ATH).
  - When there is SMS, Signal RI is Active LOW for 120mS and it becomes HIGH automatically.
  
- **DTR(Data Terminal Ready)** is Input of Module SIM900. The Module runs when this Pin receives Logic LOW; but, if this Pin DTR receives Logic HIGH, the Module stops running and enters Sleep Mode automatically (if Enable Sleep Mode by Command "**AT+CSCLK=1**"). If user requires running the Module all the time, it has to control Pin DTR on the Module side to receive Logic LOW or it has to disable Sleep Mode by Command "**AT+CSCLK=0**" and then save this Configuration.

## 7. Example of using AT Command with Module SIM900

Module GSM/GPRS model SIM900 is designed and run as Modem, it is used to connect, command, and communicate with Module through RS232 Serial Port; it supports Baud Rate in the range of 1200-115200 bps and it uses AT Command. Its functions are similar to general Modem but it adds Option and other special commands, so it is suitable and compatible with capability of Module.

Please read further details of format and function of AT Command in order to connect and command Module SIM900 from the manual of AT Command (file document **SIM900\_AT Command Manual\_V1.06.pdf**) in CD-ROM. In this case, it only describes format and how to use commands briefly to be guideline for beginner to study and understand the command. The format of AT Command begins with ASCII Code of 2 characters that are "A" and "T"; in this case, it can be either capital letter or small letter because the meaning is the same. Next, it follows by Command Code and Option of command (if yes). Remember, every command always ends by Enter or 0DH(13); for example, the format of Command RESET is either "ATZ" or "atz". The command format is divided into 4 groups as follows;

Function	Format of Command	Description
Test command	<b>AT+&lt;x&gt;=?</b>	This command format is used to read format and parameter of command. If the command actually exists, the Module acknowledges the command by typing all existing parameters of command.
Read Parameter	<b>AT+&lt;x&gt;?</b>	This command format is used to read parameters that have been setup for the command. The Module acknowledges the command by typing current parameters.
Setup Configuration	<b>AT+&lt;x&gt;=&lt;...&gt;</b>	This command format is used to write or setup parameters for command such as setup Baud Rate.
Execute	<b>AT+&lt;x&gt;</b>	This command format is used to command the Module to execute according to the preferable command such as Command RESET (ATZ).

Table shows format of using AT Command (when <x> is Command Code).

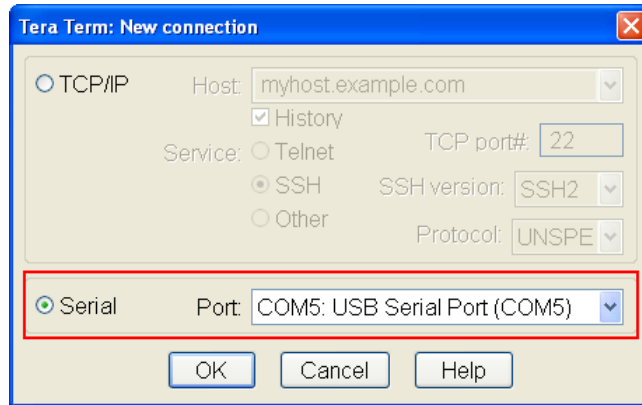
## 8. How to test operation of Board ET-BASE GSM SIM900

As mentioned above, it sends the Command Code in the format of AT Command to Module SIM900 through Serial Port. Normally, customer needs to write program to send Command Code into the Module by self, it depends on type of controller device that is used to control the operation of module; in this case, it may be computer PC or Microcontroller in any series that has RS232 Serial Port in order to connect and command Module SIM900. Moreover, it depends on ability or basic skill of user to choose any way and language to write program because it needs user with know-how to write program to command the device in order to transmit/receive data through RS232 Serial Port, this manual does not describe any detail of them.

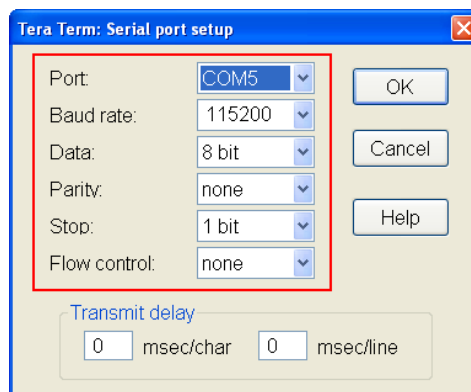
Initially, it is unnecessary to write program, but user can use ready-made program such as Program Serial Terminal of computer to test the operation; moreover, user can learn and understand the command format and operating result well. For example, if commanding the Module SIM900 to make a call to the telephone number 0811234567, user has to learn and understand the command format first; when user knows and understands how to use the Command "ATD0811234567;" to make a call, user can write the program to command the Module by self later. Next, user needs to exactly know how to write program to command the device to send Code "ATD0811234567;" through Serial Port; in this case, it suggest Program HyperTerminal from Windows to be a tool to initially test, this Program HyperTerminal is ready-made program that is provided with Windows Operating System. This program has several interesting parts because there are various capabilities to do; in this case, it only uses the part of function

Serial Terminal in Text Mode. After RUN program, any data that is received from RXD of Serial Port in the range of ASCII Code (20H.. FFH) will be converted to be character and displayed on the screen of program instantly. For other Data Code that is lower than 20H (00H-1FH), it is not displayed on the screen but it automatically assumes that it is command. For example, when it receives the Data Code as ODH, Program HyperTerminal assumes that it is the command to shift the Cursor position to the beginning of line; or, when it receives the data Code as 0AH, it shifts the Cursor position to the new line instead. On the contrary, when user presses any key, Program assumes that it is ASCII Code of character at the position that the key is pressed and it auto sends to Pin TXD of Serial Port. However, if using Windows7 or Windows8, there is no any Program HyperTerminal; so, this example illustrates how to use Program **Tera Term** instead as described below;

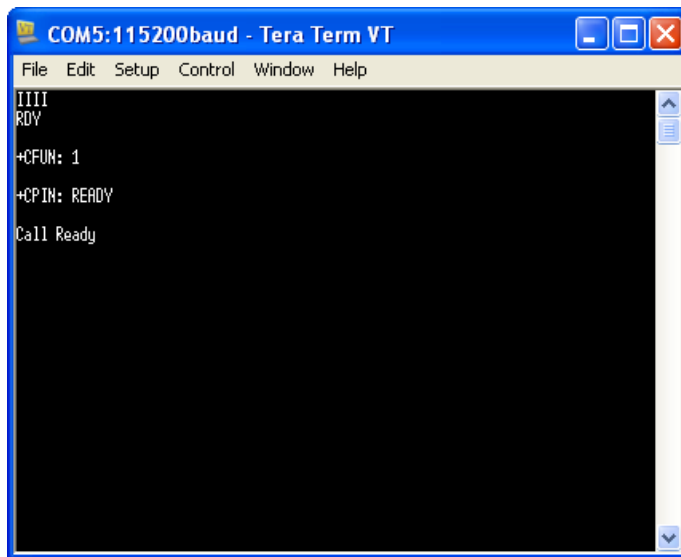
8.1 Install and open Program **Tera Term**, choose the connection as **Serial**, choose Port to interface with ET-BASE GSM SIM900, and then click **OK** as shown in the picture below;



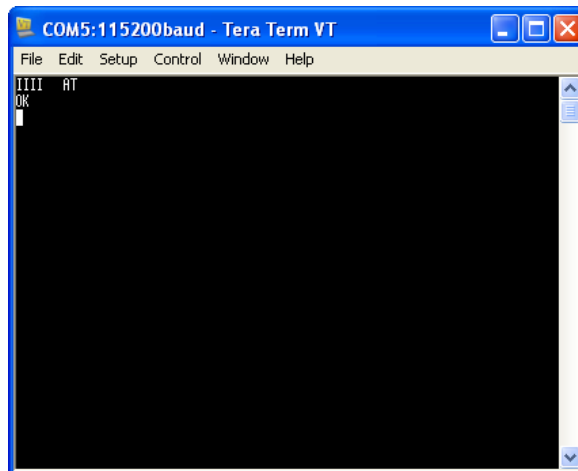
8.2 Click Menu **Setup-->Serial port...** to setup value of Port RS232. In this case, it has to choose Baud Rate according to the actual connection with Module. If setting Baud Rate of Module as **Auto-Baud Rate**, user can choose any Baud Rate that the Module supports such as 1200, 2400, 4800, 9600, 19200, 38400, 57600, or 115200; in this case, it chooses 115200. In the part of **Data**, it sets the value as **8 Bit, Parity = None, Stop bits = 1, Flow Control = None**; and finally, choose "OK" as shown in the example below;



8.3 After setup the connection successfully and everything is correct, user can connect Cable RS232 between Board and Comport of computer PC (it has to set Jumper JP2,JP3 to the position of RS232), and then supplies power into board completely. If everything is correct, user can see green LED VBAT on board is lit up. Next, it sets Power-ON Module by pressing and holding Switch ON/OFF for 1 second, user can see LED STAT (STATUS) is lit up and LED NET (NETLIGHT) starts flashing all the time, it means that the Module starts running. Moreover, it shows message on the screen of Program **Tera Term**, user needs to wait for a while until it shows the message "**Call Ready**", it means that the Module searched and found the Network successfully. Next, user can command the Module to execute by Commands as required.



If there is no any message on the screen of Program HyperTerminal, it may setup Baud Rate of Module SIM900 as **Auto-Baud Rate**, so the Module auto adjusts the Baud Rate. This Auto-Baud Rate is set by manufacturer. In this case, user types the character **AT** (only capital letter) and then presses **Enter**, the Module responds "OK" as shown in the picture below, and now the Module is ready to run.



If user would like to show the message "Call Ready" to the screen, it has to set fixed Baud Rate of board by Command **AT+IPR=115200** and then **press** Enter; this Command setup Baud Rate as 115200 bps. Next, user tries to close and re-open the Module, the message "Call Ready" appears on the screen, and the Module is now ready to run.

## 9. Example of using Command about RS232

In this case, it illustrates example of Command that is frequently used; for more information, please read from the file document **SIM900\_AT Command Manual\_V1.06.pdf**.

### 9.1 Example of using Command to setup Baud Rate

It uses Command **AT+IPR** to setup Baud Rate for the communication of Module. There are various ways to execute this command. Firstly, if user forgets what available Parameter of command or Baud Rate is, and including how to setup the value, it uses Command **AT+IPR=?** and follows by **Enter** to ask for more information. The Module responds **+IPR:** and types the all available Parameters of the Command that are 0, 300, 1200, ..., 115200 as shown in the example below (**the black message is command that is put in to ask for** and **the red message is the response that the Module answered**).

```
AT+IPR=?<Ent>
+IPR: (),(0,1200,2400,4800,9600,19200,38400,57600,115200)
OK
```

If user would like to know what the current Baud Rate is, it uses the Command **AT+IPR?** to read Parameter of this command. The Module responds **+IPR:** and follows by the current Parameter as shown in the example (0=Auto Baudrate).

```
AT+IPR?<Ent>
+IPR: 0
OK
```

Referred to the example above, the Parameter of the Command IPR is "0" that is Auto-Baud Rate, so the Module auto adjusts the Baud Rate when it initially starts first time. If user would like to set the fixed Baud Rate for the Module to keep running with this fixed Baud Rate all the time, it can do it by using the command format to setup. For example, if setting Baud Rate as 115200, it uses the command format as **AT+IPR=115200** and then follows by **Enter** as shown in the example below;

```
AT+IPR=115200<Ent>
```

```
OK
```

After changed the Baud Rate as 115200 completely, it can communicate with Module by this fixed Baud Rate all the time.

## 9.2 How to set Flow Control

It can setup Flow Control or the format of checking if the Module SIM900 is ready to communicate and transmit/receive data. This Flow Control is important because speed of processing data of each device is different. When it receives/transmits a lot of data continuously but the receiver is not ready to receive any data yet while the transmitter continues transmitting data, some data may lost and error happens. This SIM900 supports 2 types of Flow Control as mentioned below;

- **Software Flow Control (XON/XOFF Flow Control):** It uses Software to check if the Module SIM900 is ready. It uses Code *XOF(13H)* to stop transmitting data from the transmitter and it uses Code *XON(11H)* to notify or permit the transmitter to transmit the next data to the Module. This Flow Control type is suitable for connecting with the device that has no any Flow Control such as Microcontroller or device only uses 3 Cables (RXD, TXD and GND).
- **Hardware Flow Control (RTS/CTS Flow Control):** It uses Hardware to check if Module SIM900 is ready. It uses Active("LOW") Signal CTS to notify the transmitter to stop transmitting data when the Module is not ready to receive any data. On the contrary, it checks if the status of RTS is Active before transmitting data; if it is Active, it means that the receiver is not ready to receive any data and user needs to stop and wait until the status of RTS becomes "HIGH".

When Flow Control is set as **No Flow Control**, it uses Command **AT+IFC=0,0 (Default value)**.

When Flow Control is set as **XON/XOF**, it uses Command **AT+IFC=1,1**

When Flow Control is set as **RTS/CTS**, it uses Command **AT+IFC=2,2**

## 9.3 How to setup Data Format of RS232

User can setup Data Format as required; in this case, user can determine what format of transmitting data is, what size of data bit is, whether it checks Parity Bit, and how many Stop Bit is. Normally, it sets Data 8 Bit, None parity, 1 Stop Bit. However, user can change and edit this Data Format by Command **AT+ICF**.

It uses Command **AT+ICF=3,3** to setup Data Format as 8 Bit Data, None Parity, 1 Stop Bit.

#### 9.4 How to Enable/Disable Echo

Echo is the return of a transmitted command to its source when user typed commands on Program Terminal to notify user to know what the typed and transmitted command is (normally, it is Default value). If user would like to disable this function, it uses Command **ATE0&W** and follows by **Enter**; when user types the command, user does not see it on Program Terminal because it only displays the response from Module SIM900. If user would like to enable this function, it only uses the Command **ATE1&W** and follows by **Enter**.

### 10. How to Setup and Check Configuration

Normally, there are several ways to setup Configuration of Module SIM900 such as conditions of communicating with Module; moreover, user can change and edit values as required such as Baud rate or format of Handshakes for communication. Remember, it is necessary to setup Configuration of the Module to suit user's needs. Normally, these conditions always have a fixed value after resetting or Power ON; the Module sets conditions for its own when it starts running by the values that are setup and saved in Configuration. However, user can change and edit this Configuration as required. There are 2 ways to setup conditions of the Module as described below;

- **Permanently Setup:** It saves conditions of Module according to the setting format in the permanent memory internal Module by Command **AT&W**. After Module starts running or after reset Module in each time, the conditions of the Module always are set according to the saved values in Configuration.
- **Temporarily Setup:** It uses AT Commands to setup condition for running Module but it does not save any Configuration by the Command **AT&W**; so, the operation of Module also changes and varies according to the current command. When reset the Module or Power ON, the specification of Module also return to the old one instantly.

It can use AT Command to check and save Configurations into Module SIM900 as follows;

- It uses Command **AT&V** to command the Module to show the current Configuration.
- It uses Command **AT&F** to reset all Configurations to be standard value.
- It uses Command **AT&W**; it saves the current Configurations that are setup by user.

#### Suggested Configuration

- **AT+CMGF=1** (SMS Message = Text Mode)
- **ATE=1** (Echo Mode ON)
- **AT+CSCLK=0** (Disable Sleep Mode)



## 11. How to check data of Module SIM900

### 11.1 How to check signal quality

It uses the Command AT+CSQ to check signal quality; this command is used to check the strength of signal. The module responds the numeric value 0..31 or 99. If it is in the range of 2..30, it means that the signal is good; or, if it is 31, it means that the signal is excellent; or, if it is 99, it means that it cannot check any signal as shown in the example below;

```
AT+CSQ<Ent>
```

```
+CSQ: 16,0
```

```
OK
```

### 11.2 How to check Product ID

```
ATI<Ent>
```

```
SIM900 R11.0
```

```
OK
```

### 11.3 How to check Manufacturer ID

```
AT+GMI<Ent>
```

```
SIMCOM_Ltd
```

```
OK
```

### 11.4 How to check model code

```
AT+GMM<Ent>
```

```
SIMCOM_SIM900
```

```
OK
```

### 11.5 How to check Firmware Version

```
AT+GMR<Ent>
```

```
Revision:1137B10SIM900M64_ST_PZ
```

```
OK
```

### 11.6 How to check Serial Number(IMEI) of Module

```
AT+GSN<Ent>  
012896001185450  
  
OK
```

### 11.7 How to check SIM Network Code of facilitator

```
AT+COPS?<Ent>  
+COPS: 0,0,"TH GSM"  
  
OK
```

## 12. How to make, receive, and cancel a call

- It uses the Command **ATD** to make a call; the format of command is to follow by the destination number
- It uses the Command **ATDL** to make a call; it uses the telephone number that user has recently made a call.
- It uses the Command **ATA** to receive an incoming telephone call. When there is an incoming call, user can hear ringtone from speaker (receiver) of handset; in this case, user can use the Command "**ATA**" to receive a call and speak instantly through Handset or microphone (transmitter) of home phone.
- It uses the Command **ATH** to hang up a telephone or cancel a call.

This is an example of making a call by VOICE, it has to end the command by semicolon sign (;) and **Enter (0x0D)**; for example, when it makes a call to the telephone number 0894469xxx, it is

```
ATD0894469xxx;<Ent>  
OK
```

If it makes a call but it does not answer the telephone or the line is busy, the Module reports by the message "**BUSY**" as shown in the example below;

```
ATD0894469xxx;<Ent>  
OK  
BUSY
```

This is an example of checking balance of 1-2-CALL. If it is general cell phone, it only types "\*121#" and call; but, if it is Module SIM900, it has to use the Command **ATD** and follows by the sign instead as shown in the example below;

```
ATD*121#<Ent>
```

```
OK
```

```
+CUSD: 0,"The balance of 0870681xxx is 111.62 B. & valid until 03/05/13  
Pay59B.Get3G/EDGE 70MB within7days exceed up to main pro.Press*500*70#",64
```

This is an example of receiving a call. When there is an incoming telephone call in case of using Module SIM900, it shows the message "**RING**" and user can hear ringtone from speaker (receiver) of handset. It uses the Command **ATA** in order to receive a call, or it uses the Command **ATH** to hang up or cancel or reject an incoming call as shown in the example below;

```
RING
```

```
ATA<Ent>
```

```
OK
```

It uses the Command **AT+CLIP=1** and follows by **Enter** in order to show an incoming telephone number, the Module will show the incoming telephone number as shown in the example below;

```
RING
```

```
+CLIP: "0894469xxx",129,"",",",0
```

### 13. How to check balance by USSD

It also uses the Command **AT+CUSD** and follows by **USSD (Unstructure Supplementary Service Data)** to check balance. The example below shows how to check balance in case of 1-2-CALL.

```
AT+CUSD=1,"*121#"<Ent>
```

```
OK
```

```
+CUSD: 0,"The balance of 0870681xxx is 111.62 B. & valid until 03/05/13  
Pay59B.Get3G/EDGE 70MB within7days exceed up to main pro.Press*500*70#",64
```

#### 14. How to receive SMS

Normally, Module SIM900 can set 2 operation modes for message or SMS that are PDU Mode and Text Mode. Firstly, it is **PDU Mode**; it receives and displays the operation of command in the format of Binary Code. Secondly, it is **Text Mode**; it receives and displays the operation of command in the format of text. In this case, Text Mode is easier than PDU Mode because it is easier to compile and understand. This illustrates an example of Text Mode.

- It uses the Command **AT+CMGF=1** to set the data format as Text Mode; when it sends SMS to Module, it shows message to notify user.  
For example, if the message as **+CMTI: "SM",3** appears, it means that there is an incoming message and it is saved in the third order of the memory.
- It uses the Command **AT+CMGR** to read message. For example, if user requires reading the message in the third order, it has to use the Command as **AT+CMGR=3**.
- It uses the Command **AT+CMGL="ALL"** to show all messages that are saved in the memory; in this case, use can choose type of message as required such as new message or all messages.
- It uses the Command **AT+CMGD** to delete message from the memory. For example, if user requires deleting the message in the third order, it has to use the Command as **AT+CMGD=3**.
- It uses the Command **AT+CMGDA="DEL ALL"** to delete all messages from the memory.

This is an example of receiving SMS by sending the message **"Hello 12345"** to Module SIM900B. When it received the message completely, it shows the message **text+CMTI: "SM",n**; in this case, **n** means the order of message.

```
+CMTI: "SM",3
AT+CMGR=3<Ent>
+CMGR: "REC UNREAD","+66894469xxx",,"07/11/19,13:29:25+28"
Hello 12345
OK
```

If user re-reads the old message repeatedly, the status of message becomes **"REC READ"** instead to notify user to know that this message has been read completely as shown in the example below;

```
AT+CMGR=3<Ent>
+CMGR: "REC READ","+66894469xxx",,"07/11/19,13:29:25+28"
Hello 12345
OK
```

## 15. How to send English SMS

Before sending SMS, user has to setup format of message to be Text Mode first by the Command **AT+CMGF=1**; set Parameter of SMS by the Command **AT+CSMP=17,167,0,0**; and choose the set of character for sending by the Command **AT+CSCS="GSM"** as shown in the example (in this case, user can check if all 3 values are correct by the Command **AT+CMGF?**, **AT+CSMP?** and **AT+CSCS?**; if yes, user does not set any new value).

```
AT+CMGF=1<Ent>
OK
AT+CSMP=17,167,0,0<Ent>
OK
AT+CSCS="GSM"<Ent>
OK
```

It uses the Command **AT+CMGS** to send SMS. If it is Text Mode, the format of command is **AT+CMGS="+ telephone number of receiver"** and the telephone number of receiver always replaces the zero by Country Code. In case of Thailand, the Country Code of Thailand is "66". If user requires sending SMS to the telephone number in Thailand such as 089-4469xxx, it has to set the telephone number of the receiver as 6689-4469xxx instead; so, the code of person who receives the message is "+66894469xxx". When the Module SIM900 receives the Command **AT+CMGS** completely, it acknowledges and responds the sign ">" to user; next, user can type any preferable message to send to the Module instantly. Finally, it has to end the message by pressing Key **Ctrl+Z (0x1A)**. For example, if sending SMS **"Hello Test SMS"** to the telephone number 0894469xxx, it will be

```
AT+CMGS="+66894469xxx"<Ent>
> Hello Test SMS<Ctrl+Z>
+CMGS: 6
OK
```

If typing the Command **AT+CMGS="+66894469xxx"** and it responds **ERROR** to user, it means that the message that user has typed is error or user does not set the format of message as Text Mode yet. In this case, it has to check the operation by Command **AT+CMGF?**; if it responds by **+CMGF: 0**, it means that user does not set the format of message as Text Mode yet; so, it has to use the Command **AT+CMGF=1** and follows by **Enter** to setup the format of message as Text Mode.

**16. Code of Thai SMS**

In case of Thai MSM, it cannot be displayed by general Program Terminal because of different the system of character. Program Terminal only uses normal ASCII Code that is 1 Byte but Thai Code that is used with cell phone uses special code called "Unicode", it is particularly provided for Thai SMS, 1 Character consists of 2 Byte Data. The Unicode of Thai Language is in the range of 0E00H... 0E7FH while the Unicode of English Language uses 2 Byte as same as Thai and it is in the range of 0000H...007FH. Normally, if it is only English message, all characters that are used in SMS is ASCII Code; it uses 1 Byte Code and it omits 00H that is the first byte in Unicode. For example, code of "A" that is **0041H** will be **41H** instead.

0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	000A	000B	000C	000D	000E	000F
0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	001A	001B	001C	001D	001E	001F
	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
0020	0021	0022	0023	0024	0025	0026	0027	0028	0029	002A	002B	002C	002D	002E	002F
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0030	0031	0032	0033	0034	0035	0036	0037	0038	0039	003A	003B	003C	003D	003E	003F
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0040	0041	0042	0043	0044	0045	0046	0047	0048	0049	004A	004B	004C	004D	004E	004F
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
0050	0051	0052	0053	0054	0055	0056	0057	0058	0059	005A	005B	005C	005D	005E	005F
`	a	B	c	d	e	f	g	h	i	j	k	l	m	n	o
0060	0061	0062	0063	0064	0065	0066	0067	0068	0069	006A	006B	006C	006D	006E	006F
p	q	R	s	t	u	v	w	x	y	z	{		}	~	
0070	0071	0072	0073	0074	0075	0076	0077	0078	0079	007A	007B	007C	007D	007E	007F

Table shows Unicode for English Language.

	!	“	#	\$	%	&	‘	(	)	*	+	,	-	.	/
20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
`	a	B	c	d	e	f	g	h	i	j	k	l	m	n	o
60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
p	q	R	s	t	u	v	w	x	y	z	{		}	~	
70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F

Table shows ASCII Code for English Language.

	ก	ข	ฃ	ค	ค	ฅ	ง	จ	ฉ	ช	ช	ฌ	ญ	ฎ	ฏ
0E00	0E01	0E02	0E03	0E04	0E05	0E06	0E07	0E08	0E09	0E0A	0E0B	0E0C	0E0D	0E0E	0E0F
ฐ	ฑ	ฒ	ณ	ด	ต	ถ	ท	ธ	น	บ	ป	ผ	ฝ	พ	ฟ
0E10	0E11	0E12	0E13	0E14	0E15	0E16	0E17	0E18	0E19	0E1A	0E1B	0E1C	0E1D	0E1E	0E1F
ภ	ม	ย	ร	ฤ	ล	ฬ	ว	ศ	ษ	ส	ห	ฬ	อ	ฮ	ฯ
0E20	0E21	0E22	0E23	0E24	0E25	0E26	0E27	0E28	0E29	0E2A	0E2B	0E2C	0E2D	0E2E	0E2F
ะ	ั	า	ำ	ิ	ี	ึ	ุ	ู	ุ	ุ					฿
0E30	0E31	0E32	0E33	0E34	0E35	0E36	0E37	0E38	0E39	0E3A	0E3B	0E3C	0E3D	0E3E	0E3F
เ	แ	เ	เ	เ	า	า	เ	เ	เ	เ	+	เ	เ	เ	☉
0E40	0E41	0E42	0E43	0E44	0E45	0E46	0E47	0E48	0E49	0E4A	0E4B	0E4C	0E4D	0E4E	0E4F
อ	อ	๒	๓	๔	๕	๖	๗	๘	๙	๑	๒				
0E50	0E51	0E52	0E53	0E54	0E55	0E56	0E57	0E58	0E59	0E5A	0E5B	0E5C	0E5D	0E5E	0E5F
0E60	0E61	0E62	0E63	0E64	0E65	0E66	0E67	0E68	0E69	0E6A	0E6B	0E6C	0E6D	0E6E	0E6F
0E70	0E71	0E72	0E73	0E74	0E75	0E76	0E77	0E78	0E79	0E7A	0E7B	0E7C	0E7D	0E7E	0E7F

Table shows Unicode for Thai Language.

## 17. Principles of decoding Unicode

The structure of Unicode always consists of 2 Byte Code; the first byte notifies Table to know what language of Unicode is. If it is Unicode of English Language, the first byte is 00H and the second byte is character code that accords with ASCII Code. If it is Thai Language, the first byte is 0EH and the second byte is character code. Referred to the demonstration of receiving SMS, it is all ASCII Code if it only sends SMS by English Language; it means that there is 1 Byte Data for each a character. If it sends SMS by both Thai and English characters together, English character is encoded by Unicode.

In summary, if sending SMS by Thai characters, it always uses Unicode; but, if it is English, it can be both Unicode and ASCII Code. If it is Unicode, it uses 2 Byte character code as same as Thai character; its value is in the range of 0000H...007FH and Code 00H is the first Byte Data. If SMS only has English character, the character code for SMS is ASCII Code; it uses 1 Byte Code and it omits 00H that is the first byte in Unicode. For example, if it is "A", it will be **41H** instead of **0041H**. If SMS has both Thai and English characters, characters are encoded by Unicode as same as Thai character.

So, user has to consider this issue when decoding any character. If user found the character code in the range of 20H-7FH, it means that it is ASCII Code and it can be displayed instantly; or, if user found 00H, it means that it is English Unicode and its character code will be in the next Byte Data; or, if user found 0EH, it means that it is Thai Unicode and its character code will be in the next data byte as well.

For example, it sends SMS as "สวัสดี Jack" to the Module SIM900, the Module received the message successfully and stored the message in the first order; if it uses Program Hyper Terminal or other Terminal that displays the operating result as ASCII, it will reports the operating result as shown in the picture below;

```
+CMTI: "SM",1
```

When it displays the received data in the format of HEX String, user found that the amount of the received data is much than the received data is displayed through the screen of Program Hyper Terminal because Program Terminal only displays the received data in the part of ASCII Code (20H...FFH); other code below 20H (00H-1FH) is assumed the Command. For example, if it is **0DH,0AH**, it is not displayed but it assumes that it is the Command to shift the Cursor position to the beginning of line and it starts the new line. In this case, it only describes the received data in the format of HEX String instead; for example, when it receives ASCII Code of character "A", it displays the result as "41" instead. It displays the result of HEX String on the left side and it displays the result of ASCII Code on the right side in order to compare; in this case, user can understand the format better. Referred to the message **+CMTI: "SM",1** on the screen of Program Hyper Terminal, if it is displayed in the format of HEX String, the result is;



```
0D 0A      ..
2B 43 4D 54 49 3A 20 22 53 4D 22 2C 31 0D 0A  +CMTI: "SM",1..
```

Referred to the message **+CMTI: "SM",1**, it means that there is an incoming message and it is stored in the first order of memory; user can read this message by Command **AT+CMGR=1** as shown in the example below;

```
41 54 2B 43 4D 47 52 3D 31 0D      AT+CMGR=1.
```

When it received the Command **AT+CMGR=1**, the Module SIM900 displays the message in the first order as shown in the format below;

```
+CMGR: "REC UNREAD","+66811234567",,"07/11/22,10:21:37+28"
<...ข้อความที่รับได้...>
```

- +CMGR:** It responds to the command for reading message.
- "REC UNREAD":** It is status of message. If it is *REC UNREAD*, it means it never reads the message; but, if it is *REC READ*, it means it has read the message.
- "+66811234567":** It is telephone number of transmitter; in this case, it is Country Code of Thailand that is 0811234567.
- "07/11/22,10:21:37+28":** It is date that received the message.

Referred to the example, if it displays the data that received from the Module in the format of HEX String, the result of reading the message is shown below;

```
0D 0A      ..
2B 43 4D 47 52 3A 20 22 52 45 43 20 55 4E 52 45  +CMGR: "REC UNRE
41 44 22 2C 22 2B 36 36 38 31 31 32 33 34 35 36  AD","+6681123456
37 22 2C 2C 22 30 37 2F 31 31 2F 32 32 2C 31 30  7",,"07/11/22,10
3A 32 31 3A 33 37 2B 32 38 22 0D 0A      :21:37+28"..
0E 2A 0E 27 0E 31 0E 2A 0E 14 0E 35 00 20 00 4A  .*'.1.*...5..J
00 61 00 63 00 6B 0D 0A      .a.c.k..
0D 0A      ..
4F 4B 0D 0A      OK..
```

When user considers the code in the part that is message, all code of message is Unicode; the first character is **0E 2A** and the last character is **00 6B**. When it decodes the message, it will be

- 0EH 2AH = It is Unicode of Thai character ส
- 0EH 27H = It is Unicode of Thai character ๓
- 0EH 31H = It is Unicode of Thai character ๗
- 0EH 2AH = It is Unicode of Thai character ส
- 0EH 14H = It is Unicode of Thai character ด

0EH 35H = It is Unicode of Thai character **ก**  
 00H 20H = It is Unicode of English character **Space**  
 00H 4AH = It is Unicode of English character **J**  
 00H 61H = It is Unicode of English character **a**  
 00H 63H = It is Unicode of English character **C**  
 00H 6BH = It is Unicode of English character **k**

When it only sends on English message such as "Hello Jack" to the Module SIM900, the Module SIM900 received the message successfully and saves the message in the second order. If it uses Program Hyper Terminal or other Terminal to display the result in the format of ASCII, it reports the result as shown below;

```
+CMTI: "SM",2
```

When it displays the result in the format of HEX String, the result will be

```
0D 0A ..
2B 43 4D 54 49 3A 20 22 53 4D 22 2C 32 0D 0A +CMTI: "SM",2..
```

Referred to the message **+CMTI: "SM",2**, it means that there is an incoming message and it is stored in the second order of the message. User can read this message by Command **AT+CMGR=2** as shown in the example below;

```
41 54 2B 43 4D 47 52 3D 32 0D AT+CMGR=2.
```

Referred to the example, if it displays the data that received from the Module in the format of HEX String, the result of reading the message is shown below

```
0D 0A ..
2B 43 4D 47 52 3A 20 22 52 45 43 20 55 4E 52 45 +CMGR: "REC UNRE
41 44 22 2C 22 2B 36 36 38 31 31 32 33 34 35 36 AD", "+6681123456
37 22 2C 2C 22 30 37 2F 31 31 2F 32 32 2C 31 31 7", "07/11/22,11
3A 33 34 3A 30 36 2B 32 38 22 0D 0A :34:06+28"..
48 65 6C 6C 6F 20 4A 61 63 6B 0D 0A Hello Jack..
0D 0A ..
4F 4B 0D 0A OK..
```

In this case, Code of message in SMS is general ASCII Code; each character uses 1 Byte Data as follows;

48H = It is ASCII Code of **H**  
 65H = It is ASCII Code of **e**

6CH = It is ASCII Code of **I**  
 6FH = It is ASCII Code of **O**  
 20H = It is ASCII Code of **Space**  
 4AH = It is ASCII Code of **J**  
 61H = It is ASCII Code of **a**  
 63H = It is ASCII Code of **C**  
 6BH = It is ASCII Code of **k**

**18. How to send Thai SMS**

When sending SMS in this mode, it is important to setup number of SMS Service Center of each facilitator correctly as follows;

**TRUE = +66891009120**  
**DTAC = +66816110400**  
**AIS = +66818110888**                      **1-2-CALL = +66818310808**

It uses Command **AT+CSCA?** and follows by **Enter** to check number of SMS Service Center. If the number is correct, it is unnecessary to setup new number. If no, it has to setup new number correctly by Command **AT+CSCA** as shown in the example below;

Now, it illustrates an example of **1-2-CALL = +66818310808** and it sends the message as "กขค" to the telephone number **+66811234567**. Before sending any data, it has to convert values into Unicode first. Please refer to the table.

1. **SMS Service Center of 1-2-CALL; telephone number +66818310808;** When referred to the table, the value that will be sent is **AT+CSCA**.

**+ = 002B**  
**6 = 0036**  
**6 = 0036**  
**8 = 0038**  
**1 = 0031**  
**8 = 0038**  
**3 = 0033**  
**1 = 0031**  
**0 = 0030**  
**8 = 0038**  
**0 = 0030**  
**8 = 0038**

2. The destination number **+66811234567** to receive message (please change the number according to the actual telephone number of receiver). When referred to the table, the command that will be sent is **AT+CMGS**.

```

+ = 002B
6 = 0036
6 = 0036
8 = 0038
1 = 0031
1 = 0031
2 = 0032
3 = 0033
4 = 0034
5 = 0035
6 = 0036
7 = 0037

```

3. Thai message that will be sent is “กขค”. When referred to the table, the command that will be sent is

```

ก = 0E01
ข = 0E02
ค = 0E04

```

When all values are complete, user can send the command as shown in the example.

```

AT+CMGF=1<Ent>
OK
AT+CSCS="UCS2"<Ent>
OK
AT+CSCA="002B00360036003800310038003300310030003800300038",145<Ent>
OK
AT+CSMP=17,167,0,25<Ent>
OK
AT+CMGS="002B00360036003800310031003200330034003500360037"<Ent>
>0E010E020E04<Ctrl+Z> ;ข้อความภาษาไทยที่จะส่ง
+GMGS: 57
OK

```

**19. How to use SIM Command (SIM Application Toolkit: STK)**

SIM Application Tool Kit is special command of Module SIM900 that is used to command SIM through special command, especially in a series of SIM supports user's Application. Normally, person who requires developing Applications for SIM needs to know and understand how to develop SIM well; please read restrictions and information from Technical Reference of "GSM11.14". For SIM Service Provider or facilitator in Thailand, it now creates and contains Application internal some kinds of SIM, especially pre-paid SIM (Mobile Top Up Service); facilitators in Thailand such as AIS or DTAC or TRUE supports and provides this service for customers.

In this case, it does not describe how to develop Application for SIM, but it describes how to communicate and enter command in the created Application of SIM instead. For example, it is Pre-paid SIM of DTAC or SIM Happy online that has already developed Application; in this case, it is unnecessary to know much detail of Application on SIM, customer only knows and understands the restrictions and how to choose menu and enter data correctly. The example below illustrates how to refill or top up;

1. Enter Menu Happy Refill
2. Choose Language to Top Up
3. Choose Menu Top Up
4. Identify telephone number to Top Up
5. Choose price or amount of money to refill
6. Enter password
7. Confirm

After completed all 7 steps above successfully, Program Application on SIM is active; it commands the mobile phone to send SMS to the Server to refill money or top up to the identified telephone number. As mentioned above, customer has no chance of knowing and understanding all process of sending SMS to ask for top up; or, what telephone number it sends to is; or, how to edit, modify, include or encode data. Customer never knows these processes; so, it is a secret process and safer because it is not worry about fake SMS from anyone else who tries to send to Server.

**Example of Balance Check for True Money of SIM True**

Now, it illustrates an example of Application on SIM because user can learn to understand the process better; in this case, it illustrates an example of using Application on Pre-Paid SIM from True that has already downloaded Application of True Money into SIM successfully. First of all, user should try to use the mobile phone to execute first; next, try to command Module SIM900 do the same thing because user can see and understand the operation clearly. When using mobile phone, the process is described as follows;

1. Choose Menu True Money, there are 6 sub-menus as follows;
  - Check balance
  - Transfer
  - Refill
  - Change Password

- Register
  - About
2. If user requires checking balance, it has to choose the first submenu that is "Check Balance".
  3. When Program Application of mobile phone acknowledges that any key is pressed, it commands Application in SIM to perform according to the chosen submenu. Next, the message as password appears on the screen of mobile phone, user needs to enter 4-digit password correctly according to the registration.
  4. When entered the password successfully, Program Application on the mobile phone converts the password and sends to Program Application on SIM; next, the Application on SIM converts and encodes the password to be SMS and then sends to Server of Service Provider.
  5. User receives SMS to notify the balance as shown in the example.

Referred to the process above, it is easier if using mobile phone to perform because there is Program Application that supports all operation completely; moreover, there is keyboard and screen to display data conveniently. If using Module SIM900 to perform the process, it cannot perform directly because there is no any readymade Program Application that supports SIM unlike mobile phone. However, user can use Commands of Module SIM900 to communicate with SIM; for example, access Application in SIM, see lists of Menu Application on SIM, or command module to perform according to the chosen Submenu.

### Example of Balance Check of "True Money" by using SIM900

First of all, it has to setup values for Module SIM900 completely, (please read more information from the Document [AN\\_SIM900\\_STK\\_UGD\\_V1.00.pdf](#))

AT*PSSTKI=1<Ent>	; เปิดการใช้งาน STK function
OK	
AT+CMGF=1<Ent>	; แสดงในรูปแบบ TEXT mode
OK	
AT+CMEE=2<Ent>	; แสดงการรายงานของคำสั่ง
OK	
AT+CSCS="UCS2"<Ent>	; รูปแบบตัวอักษรแบบ UCS2
OK	

- Close Module SIM900 and re-open; user can see message as shown in the example, it means that there are 7 menus of STK.

```
*PSSTK: "SETUP MENU",1,4,"Menu",0,0,1,0,0,7
```

- Use Command to start the operation of Menu

```
AT*PSSTK="SETUP MENU",1,1<Ent>
```

OK

```
*PSSTK: "END SESSION"
```

- Use Command to show what lists of Menu is

```
AT*PSSTK="GET ITEM LIST",7<Ent>
```

- Next, the Module reports all 7 menus as shown in the example.

```
*PSSTK: "GET ITEM LIST",1,1,4,"True Money",0,0,0
```

```
*PSSTK: "GET ITEM LIST",2,2,4,"True Product",0,0,0
```

```
*PSSTK: "GET ITEM LIST",3,3,4,"True Payment",0,0,0
```

```
*PSSTK: "GET ITEM LIST",4,4,4,"Other Service",0,0,0
```

```
*PSSTK: "GET ITEM LIST",5,5,4,"True Transfer",0,0,0
```

```
*PSSTK: "GET ITEM LIST",6,6,4,"Fun & Smart",0,0,0
```

```
*PSSTK: "GET ITEM LIST",7,7,4,"Setting2Play",0,0,0
```

OK

- Use Command to choose Menu True Money. Referred to the example; it chooses the first Menu, it shows 6 sub-menus in Menu True Money.

```
AT*PSSTK="MENU SELECTION",1<Ent>
```

OK

```
*PSSTK: "SELECT ITEM",0,0,"",0,0,1,0,0,6
```

- Use Command to show what sub-menu in Menu True is

```
AT*PSSTK="GET ITEM LIST",6<Ent>
*PSSTK: "GET ITEM LIST",1,1,4,"Check Balance",0,0,0

*PSSTK: "GET ITEM LIST",2,2,4,"Transfer",0,0,0

*PSSTK: "GET ITEM LIST",3,3,4,"Refill",0,0,0

*PSSTK: "GET ITEM LIST",4,4,4,"Change Password",0,0,0

*PSSTK: "GET ITEM LIST",5,5,4,"Register",0,0,0

*PSSTK: "GET ITEM LIST",6,6,4,"About",0,0,0

OK
```

- Use Command to choose Menu Check Balance as shown in the example. Next, it shows warning message to force customer to enter password according to the registration with True Money.

```
AT*PSSTK="SELECT ITEM",1,1,0,0<Ent>
OK

*PSSTK: "GET INPUT",1,0,4,1,4,"password:",0,0,255,"",1,4,0
```

- Use Command to enter password(1234) as shown in the example below;

```
AT*PSSTK="GET INPUT",1,4,"1234",0,0
OK

*PSSTK: "NOTIFICATION",1,19,0,255,"",0,0
```

- Enter Command as shown in the example, please wait for a while until user found new message in the inbox. Next, user can use the Command **AT+CMGR** to read the message.



```
AT*PSSTK="NOTIFICATION",1,0
```

```
OK
```

```
*PSSTK: "END SESSION"
```

```
+CMTI: "SM",7
```

```
; มีข้อความใหม่เข้ามา
```

```
มา
```

## 20. How to read data from website by GPRS Connection (HTTP GET)

When connecting GPRS, user needs to know the value of APN that is used to connect with internet network of each facilitator as follows;

```
AIS = internet
TRUE = internet
DTAC = www.dtac.co.th
```

This example illustrates how to pull data from web site [www.etteam.com](http://www.etteam.com) by sending the commands below (please read more information from the Document [SIM900\\_IP\\_Application\\_Note\\_V1.03.pdf](#))

```
AT+SAPBR=3,1,"Contype","GPRS"<Ent>
```

```
; เริ่มเปิดการใช้งาน GPRS
```

```
OK
```

```
AT+SAPBR=3,1,"APN","internet"<Ent>
```

```
OK
```

```
AT+SAPBR=1,1<Ent>
```

```
OK
```

```
AT+SAPBR=2,1<Ent>
```

```
+SAPBR: 1,1,"10.179.72.166"
```

```
OK
```

```
AT+HTTPINIT<Ent>
```

```
; เริ่มการใช้งาน HTTP
```

```
OK
```

```
AT+HTTTPARA="CID",1<Ent>
```

```
OK
```

```
AT+HTTTPARA="URL","www.etteam.com"<Ent>
```

```
; เว็บไซต์ที่ต้องการดึงข้อมูล
```

```

OK
AT+HTTPACTION=0<Ent>

OK

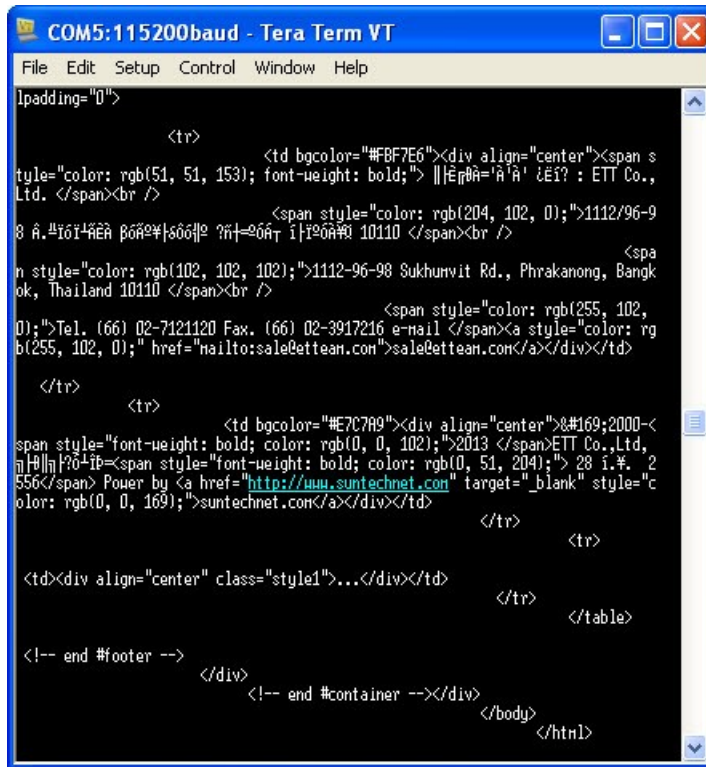
+HTTPACTION:0,200,58509

AT+HTTPREAD<Ent>
; เริ่มการอ่านข้อมูล HTTP

+HTTPREAD:58509

```

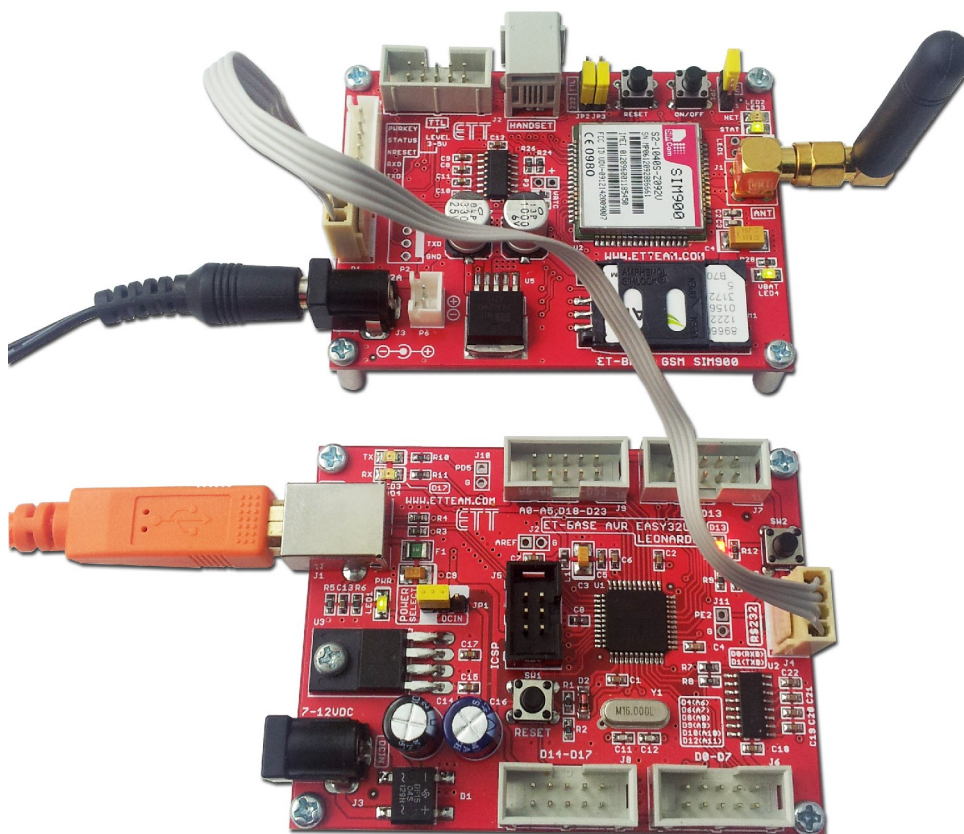
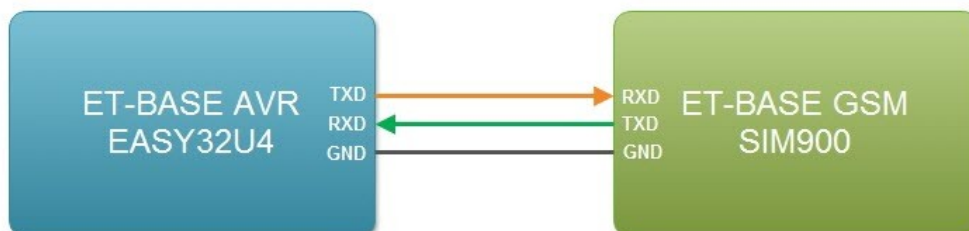
Next, it displays data of web site [www.etteam.com](http://www.etteam.com) as shown in the example below. If user requires terminating HTTP Connection, it has to use Command **AT+HTTPTERM** and Command **AT+SAPBR=0,1** to close GPRS Connection, respectively.



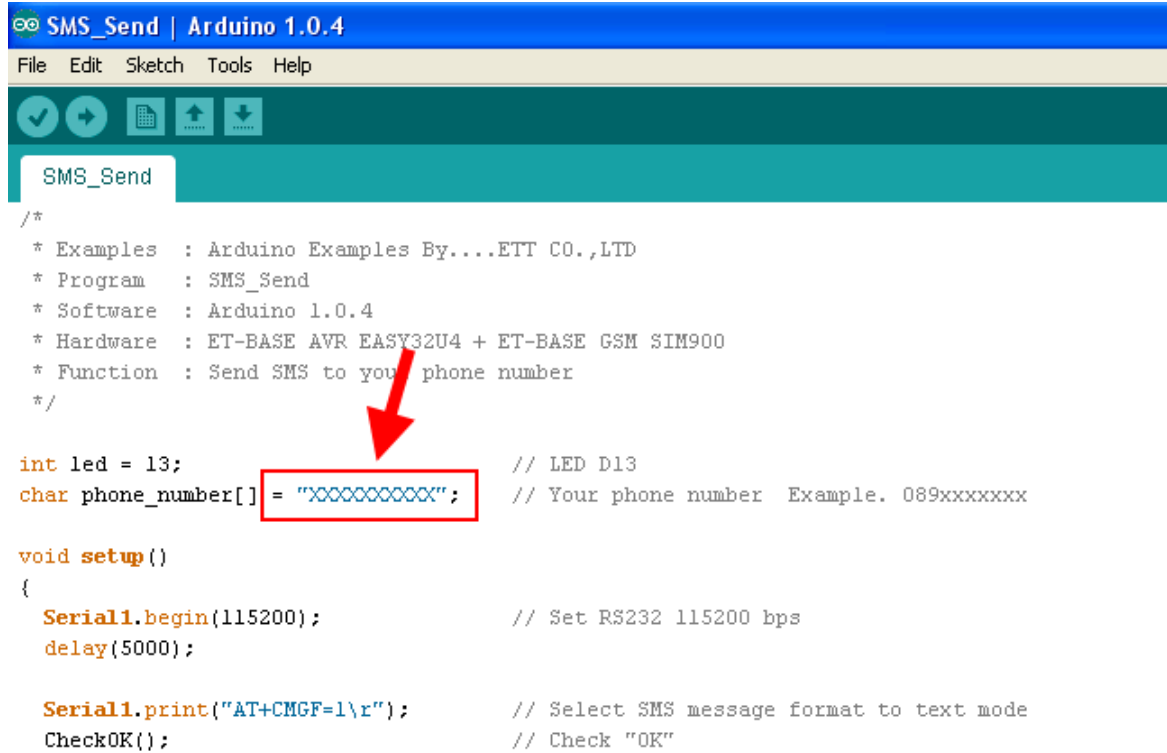
**NOTE:** While connecting GPRS, it charges customers for connection; so, customer needs to study and understand service fee of each facilitator well. Moreover, it should disable the connection when it is unused.

## 21. How to connect Board ET-BASE GSM SIM900 with Board Microcontroller

Now, it illustrates an example of connecting **Board ET-BASE GSM SIM900** with **Board ET-AVR EASY32U4** through Port RS232. Diagram and Circuit Connection is shown in the picture below;



Next, open Example Program **SMS\_Send** as shown in the picture. This example illustrates how to send message to the number of receiver; it needs to edit the message **XXXXXXXXXX** to be the telephone number that user needs to command ET-BASE GSM SIM900 to send message. Next, it should choose **Verify** and **Upload** the program that has been edited completely, please wait for a while until it shows the message **"Hello from ET-BASE GSM SIM900"** on the side of receiver's number.



```
SMS_Send | Arduino 1.0.4
File Edit Sketch Tools Help

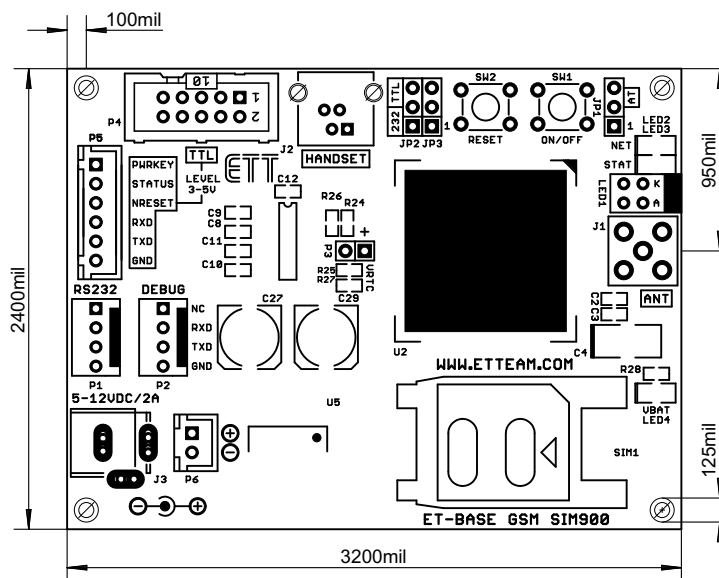
SMS_Send

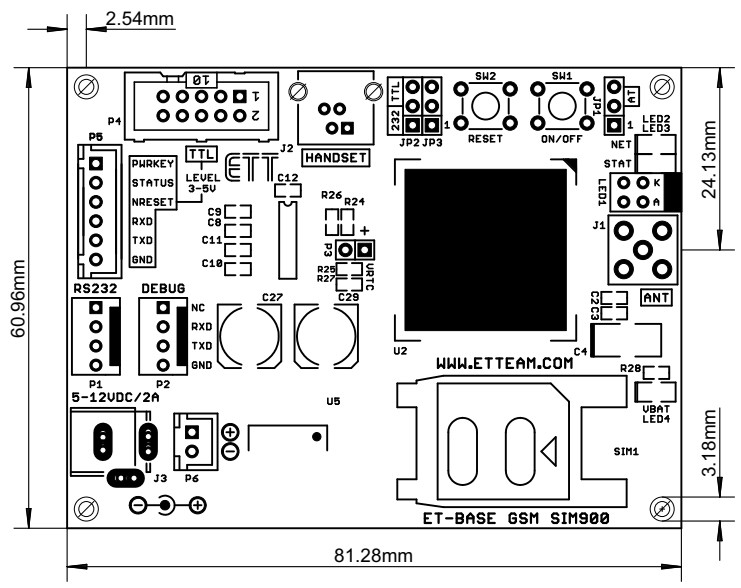
/*
 * Examples : Arduino Examples By...ETT CO.,LTD
 * Program : SMS_Send
 * Software : Arduino 1.0.4
 * Hardware : ET-BASE AVR EASY32U4 + ET-BASE GSM SIM900
 * Function : Send SMS to your phone number
 */

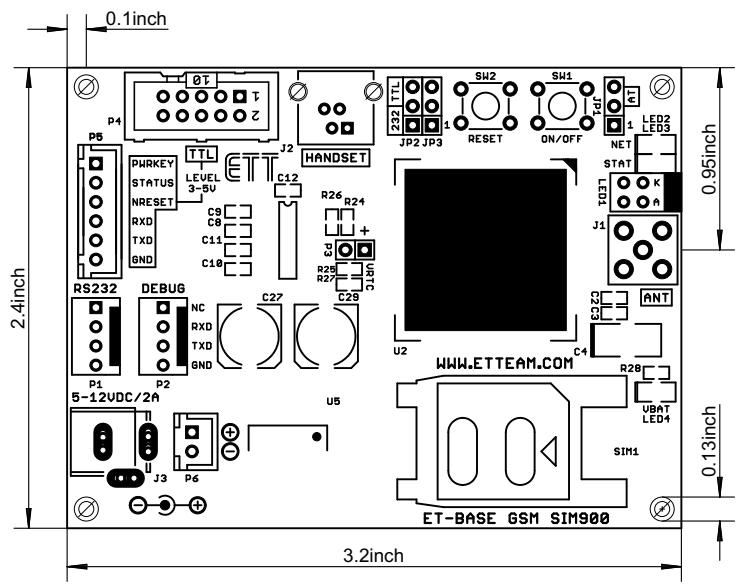
int led = 13; // LED D13
char phone_number[] = "XXXXXXXXXX"; // Your phone number Example. 089xxxxxxx

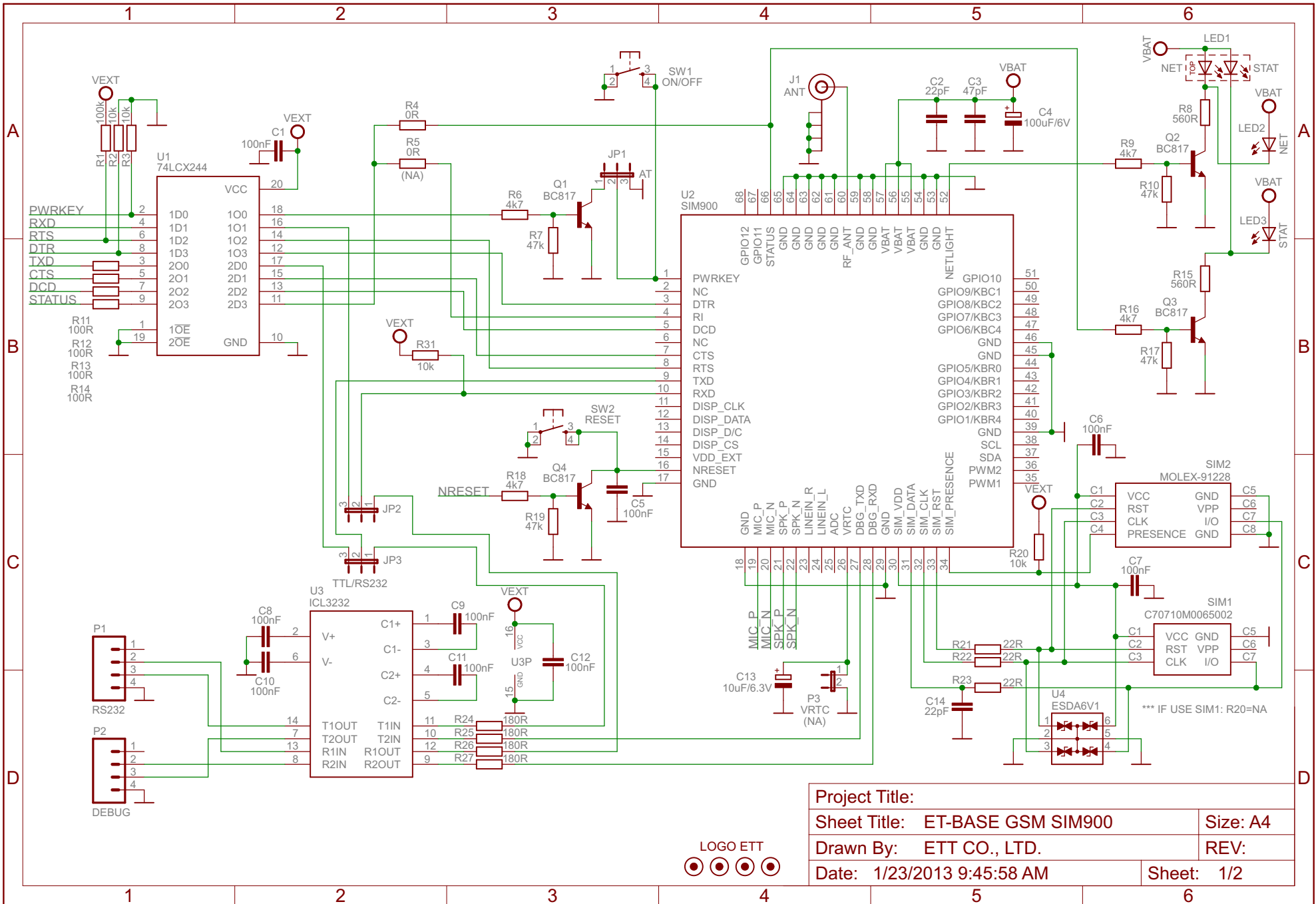
void setup()
{
  Serial1.begin(115200); // Set RS232 115200 bps
  delay(5000);

  Serial1.print("AT+CMGF=1\r"); // Select SMS message format to text mode
  CheckOK(); // Check "OK"
```





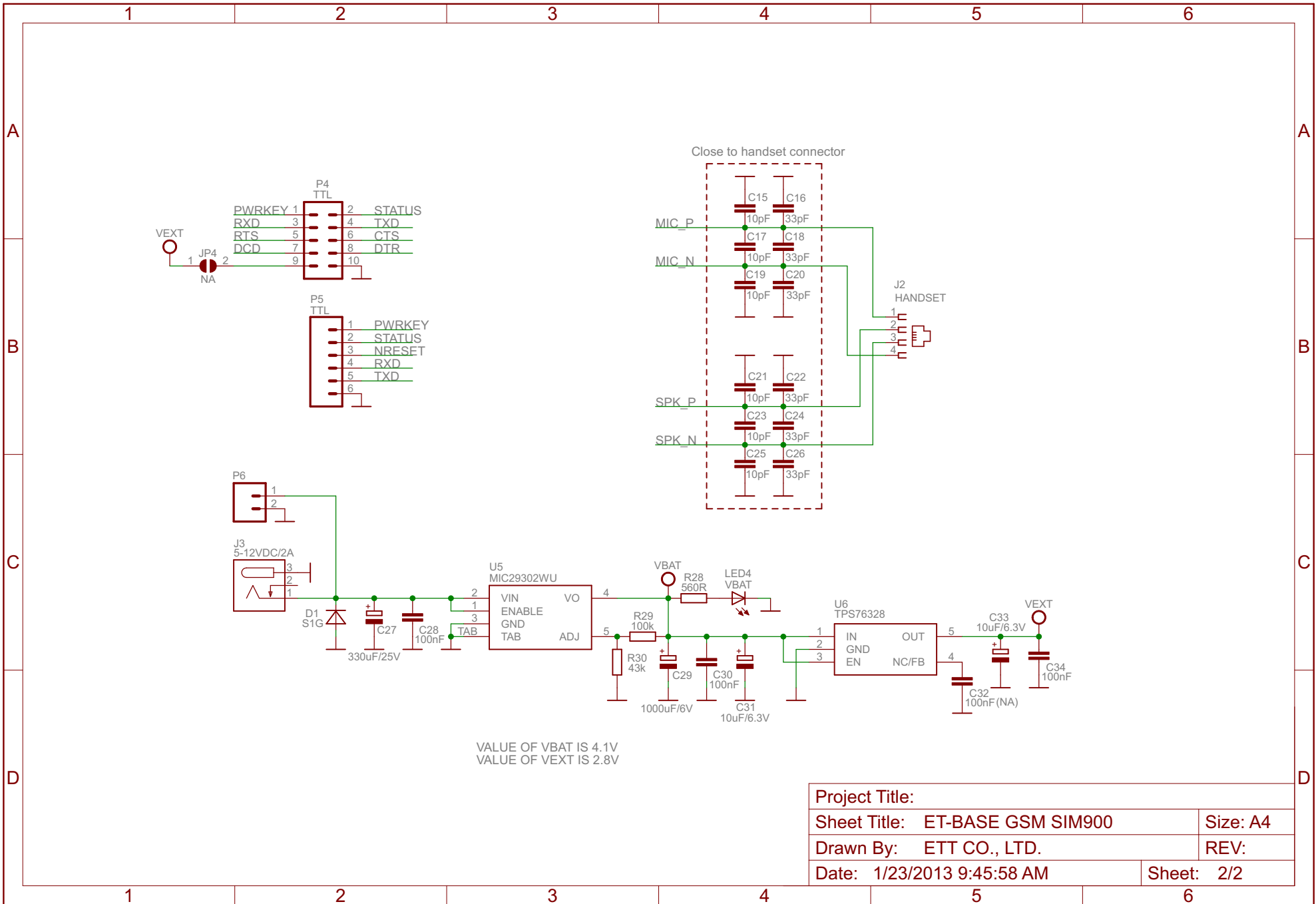




Project Title:		Size: A4	
Sheet Title: ET-BASE GSM SIM900		REV:	
Drawn By: ETT CO., LTD.		Date: 1/23/2013 9:45:58 AM	
Date: 1/23/2013 9:45:58 AM		Sheet: 1/2	







Project Title:	
Sheet Title: ET-BASE GSM SIM900	Size: A4
Drawn By: ETT CO., LTD.	REV:
Date: 1/23/2013 9:45:58 AM	Sheet: 2/2