Bluetooth® 4.0 Low Energy Modules (CC2640)

V4.01u（**Transparent Transmission**）

RF-BM-4077B1

SHENZHEN RFSTAR TECHNOLOGY CO.,LTD.
2017.4.24

Prefix

How to quickly develop new peripherals for smart phones with lower cost?

- About the application of Bluetooth Low Energy technology in smart mobile devices

-The birth of USB protocol has made peripherals of PC spring out. It is the same case with the latest opened Bluetooth Low Energy (BLE) technology. BLE technology makes it possible that electronic devices bridge smart phones. Compared with those wireless transmission technologies such as WiFi, Bluetooth 2.0, BLE technology can get lower energy consumption, faster connection, longer distance of communication range, and other advantages. So it brings wider development scopes for smart phone peripherals.

As a member of BT-SIG, Texas Instruments (TI) introduced CC2640 series of SOC low-power Bluetooth transceiver. With ARM ® Cortex ® -M3 inside, its most prominent features are rich periphery(including 31 IOs, UART, SPI, PWM, ADC, analog comparator, op-amp), super wide working voltage (1.8v – 3.8v), extreme low energy consumption (<0.1μA).

In order to facilitate the transplant and use of BLE application technology in every industry, RF-Star,the strategic partner of TI China in the field of wireless, has introduced low-power Bluetooth transparent transmission modules, **RF-BM-4077B1.**

Modules, working as a bridge between smart phones and peripherals, make it simple to develop host applications. In the bridge mode (with USART, users' products, embedded with the modules,can communicate with mobile devices (Bluetooth 4.0 supported) with ease. It realizes the super smart control and management.

RF-BM-4077B1 BLE module apply CC2640 from TI as core processor. The module runs at 2.4GHz ISM band, with GFSK (Gaussian Frequency Shift Keying) modulation scheme. It has 40 channels (channel space at 2MHz), among which there are 3 fixed radio channels and 37 data channels of adaptive automatic frequency hopping. Physical layers can be combined with classic Bluetooth RF to form a dual mode device. And 2 MHz's channel space can prevent interference from adjacent channels better. Besides, it has a wide output power adjustment (-21 dBm ~ 5 dBm) and a high gain receiving sensitivity of -97 dBm.The module is designed to quickly connect electronic products with smart mobile devices, and can be widely used in various electronic devices, such as instruments, logistics tracking, healthcare,smart home, sports metering, automotive electronics, toys, and etc. With Android 4.3 smart devices integrated with BLE technology, it will be a trend that BLE will be the standard

configuration of smart phones. And the market demand on smart phone peripherals will increase geometrically. With this module, users can integrate their existing solutions or products in the shortest development cycle, to occupy the market in the fastest speed, and to empower their company's growth with the strength of new technology.

Version History

Version num	File date	Update content
V4.00	2016/07/22	✓ First version; ✓ Support AT commands to adjust TX power,and broadcasting interval; ✓ Support AT commands customized broadcasting data,real time update,it won't save when power off; ✓ Support AT commands to set data delays（preparation time of user MCU serial port receiving)； ✓ Support the customization of equipment UDID,ll settings can be saved after power-off； ✓ If the packet is TTM:xxx，see it as AT commands，or see as transparent transmission dataIf the packet is TTM:,it will output "TTM:ERR\r\n\0"； ✓ To work with MCU,default baud rate is 9600，baud rate can be changed higher,saved when power off,lowest baud rate support 4800; ✓ Supporting the change of module name by APP, the change of UART baud rate and product UDID, the user-defining of broadcasting contents and cycles (all settings can be saved after power-off); ✓ Sending break line and overtime break line notifications from TX; ✓ Customize broadcasting data and add self-save when power off.
V4.01u	2016/8/25	✓ Add use APP to modify module connection interval,and success overtime notification. ✓ Test and optimize power consumption.
V4.01u	2017/4/24	✓ Add module picture

INDEX

Summery

The Bluetooth LE modules can work in bridge mode (transparent transmission mode) .
After being started, the module can broadcast automatically. Smart phone with specific application running will scan and pair with it. When connection is successful, the smart phone can monitor and control the module through Bluetooth protocol.

In bridge mode, user MCU can communicate with the mobile device bi-directionally through module's UART. Users can also manage and control certain communication parameters through specific serial port AT commands. The detailed meaning of the user data is defined by the up-application. Mobile devices can write the module through the APP. And the data written will be sent to the user MCU through serial ports. Then the module will transmit the data package from user MCU to the mobile devices automatically. Under the development in this mode, the user needs to undertake the work of code design for master CPU, and the code design of APP for smart mobile terminals.

Features：

- ✓ Easy to use, no need of any experience of Bluetooth protocol stack application;
- ✓ UART design for user interface, full-duplex bi-directional communication, and supporting the minimum baud rate of 4800 bps;
- ✓ Default connection interval of 20ms, which makes quick connection;
- ✓ Supporting module soft reset by AT command, and access to the MAC address;
- ✓ Supporting the adjustment of Bluetooth connection interval by AT command, and the control of different forwarding rates (dynamic power adjustment);
- ✓ Supporting the adjustment of the transmission power by AT command, the change of broadcasting interval, the customization of broadcasting data, the customization of equipment UDID, the setting of data delay (preparation time of user MCU serial port receiving), the change of the serial port baud rate, and the change of the module names (all settings can be saved after power-off);
- ✓ The length of the UART data packets can be any below or equal to the arbitrary length of 200 byte (large packet automatic distribution);
- ✓ High-speed transparent transmission rate maximum to 4 K/s and the stable rate to be 2.5 K/s to 2.8 K/s (IO5, IO6);
- ✓ Supporting the change of module name by APP, the change of UART baud rate and product UDID, the user-defining of broadcasting contents and cycles (all settings can be saved after power-off);
- ✓ Supporting the remote reset of module by APP, and the setting of transmission power;
- ✓ Supporting the adjustment of Bluetooth connection interval by APP but the setting cannot be saved after power-off (dynamic power adjustment);

- ✓ Supporting the connection status and the flexible configuration of broadcasting status prompt pin / general IO;
- ✓ Supporting battery power information reading and prompt, able to auto upload (notification of remaining battery power);
- ✓ Supporting the deep recovery modes and remote shutdown;
- ✓ Extremely low power in standby mode (current of 0.1μA from TI official data for CC2640 SoC), and the measured power consumption data is as follows：

Event	**Average current (integral computed*)**	**Average current (ammeter measured**)**	**Duration**	**Testing Conditions / Remarks**
Sleeping mode	—	0.1 uA	—	EN not connected
Broadcasting	—	60 uA	—	Broadcasting cycle is 200ms
Connection	—	70 uA	—	Connection cycle is 100ms
Module receiving data and send to APP	—	180uA	—	(20bytes,10 times/second) Connection cycle is 100ms
Module receiving APP data and send to MCU	—	160uA	—	(20bytes,10times/second) Connection cycle is 100ms

Notes:

* The official test method: Connect in series a 10Ω resistor in the circuit with power supply, and intercept voltage waveform with oscilloscope and perform integration.

** Multi-meter test method: Connect multi-meter (set at μA or mA level) in series between the battery and the module to check the value displayed, with the test voltage of 3.3 V.

Above is the measured sampling data of module **RF-BM-4077B1** and for reference only.

If lower power consumption is expected, connection interval or broadcast cycle can be appropriately increased, as shown in the module parameter settings and the serial port AT demands in related chapters.

Schematic Diagram of Working Mode

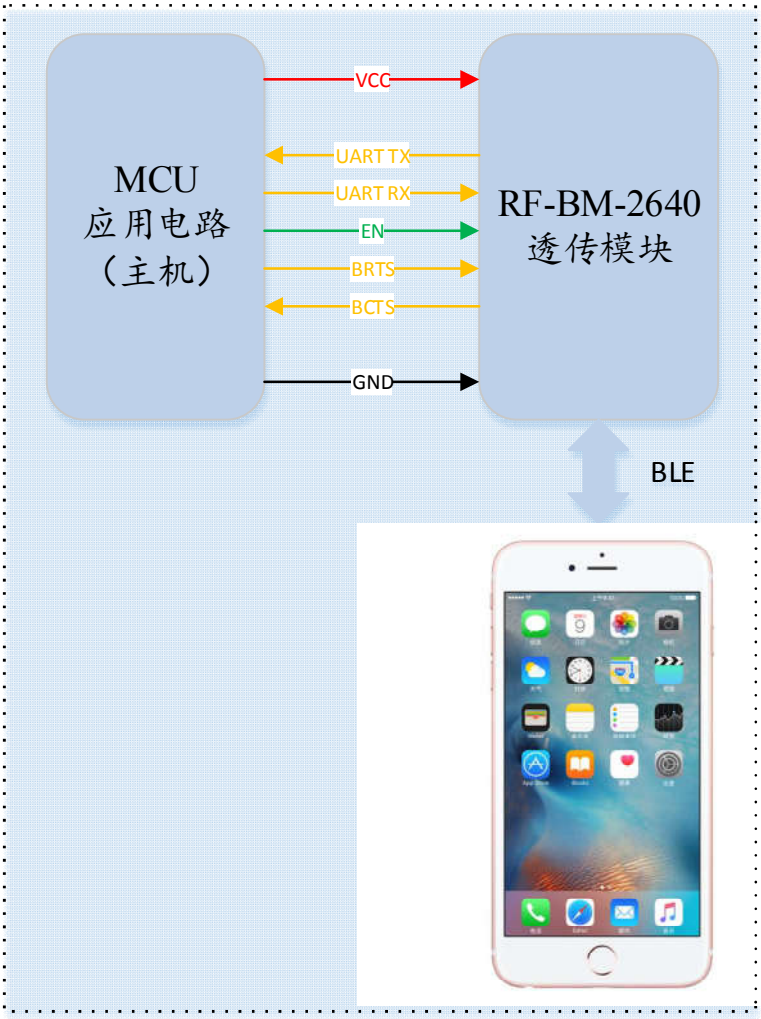


Figure1 Bridge mode Schematic Diagram

Note:

- In order to avoid the output level difference between user MCU's IO and module IO, which will result to high current, a small isolation resistor is suggested to be connected in series in the output signal line TX and BCTS.

Package Size and Pin Assignments

RF-BM-4077B1

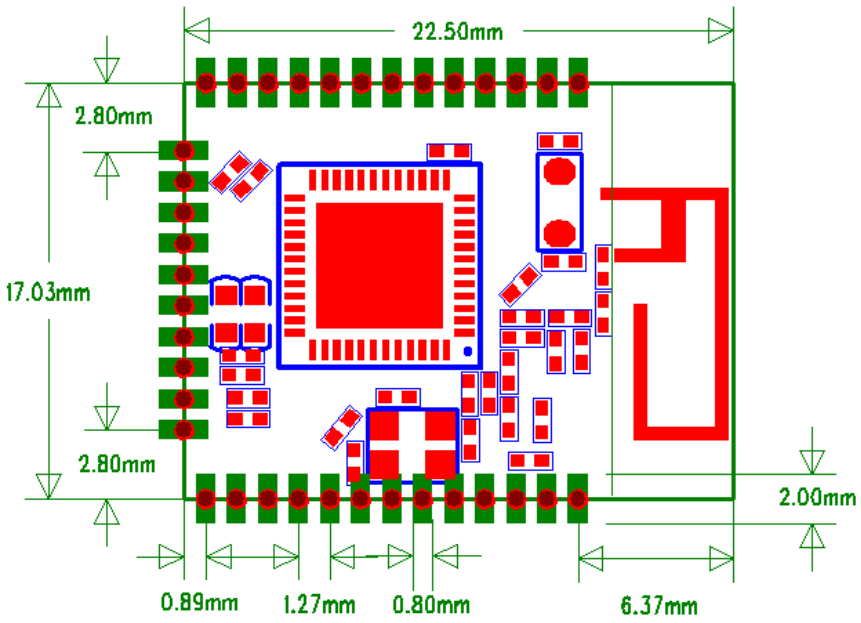


Figure 2 RF-BM-4044B2

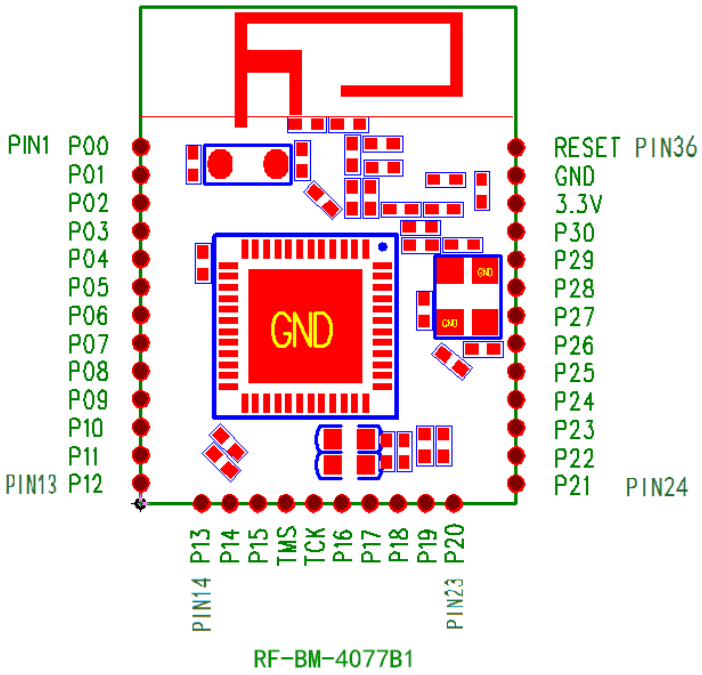


Figure 3 Pin Assignments

Pin No.	Module Pin Name	Chip Pin Name	I/O	Description
Pin1	RX	P00	I	Serial port RX
Pin2	TX	P01	O	Serial port TX
Pin3	BCTS	P02	O	Data input signal (for host wake-up, optional) - 0: Module has data to send, and the host will receive the data. 1: Module has no data to send, or data has been sent, and thevalue of the signal will be set at “1”.
Pin4	BRTS	P03	I	As the data sending requests (for module wake-up) - 0: Host has data to send, and module will wait for data transmission from the host so will not sleep 1. Host has no data to send, or data has been sent. So the value of the signal should be set at “1”.
Pin5	Broadcast instruction	P04	O	Broadcast instruction 0：Open broadcasting 1：Close broadcasting
Pin6	Link instruction	P05	O	Link instruction 0：BLE connected 1：BLE unconnected
Pin7	EN	P06	I	Module-enabled control line *(level trigger mode as default,*no internal pull-up*)* - Level trigger mode - Active when low level, without internal pull-up. 0: Module starting to broadcast, until connected to the mobile device 1: Entering sleep mode immediately, regardless of the current status (0.1μA）
Pin8	—	P07	I/O	
Pin9	—	P08	I/O	
Pin10	—	P09	I/O	
Pin11	—	P10	I/O	
Pin12	—	P11	I/O	
Pin13	—	P12	I/O	
Pin14	—	P13	I/O	
Pin15	—	P14	I/O	

Pin16	—	P15	I/O	
Pin17	TMS	—	I/O	Connect the TMS of XDS simulator
Pin18	TCK	—	I/O	Connect the TCK of XDS simulator
Pin19	—	P16	I/O	
Pin20	—	P17	I/O	
Pin21	—	P18	I/O	
Pin22	—	P19	I/O	
Pin23	—	P20	I/O	
Pin24	—	P21	I/O	
Pin25	—	P22	I/O	
Pin26	—	P23	I/O	
Pin27	—	P24	I/O	
Pin28	—	P25	I/O	
Pin29	—	P26	I/O	
Pin30	—	P27	I/O	
Pin31	—	P28	I/O	
Pin32	—	P29	I/O	
Pin33	—	P30	I/O	
Pin34	VCC	—	—	Power supply 1.8V~3.8V
Pin35	GND	—	—	Ground
Pin36	RST	—	—	Reset and effective at low level,no internal pull-up.

Protocol Description - UART Transparent Transmission (Bridge Mode)

The bridge mode means to set up two-way communication between user CPU and mobile devices by connecting the module with user CPU through serial port. Users can reset serial port baud rate and BLE connection interval, using the specified AT commands (see behind the section “Serial Port AT Command”). The module will have different data TX & RX capability, as per different serial port baud rates and BLE connection intervals. Considering the use of low-speed CPU, the default baud rate is set at 9600 bps. In the application where there is a large amount of data transmission, or there is high real-time demand, it is suggested to set the serial port baud rate at the high speed of 115200 bps. Settings can be saved and kept after power-off.

When the BLE connection interval is 20 ms and the serial port baud rate is at 115200 bps, the module has the highest transmit ability in theory (4k/s). Given the configuration in the level-enabled mode as an example, UART transparent transmission protocol will be detailed introduced as below.

The module can transmit through serial port maximum 200-byte packets at one time. According to the packet size, the packet will be sub-packed automatically and sent, with a maximum load of 20 bytes for each wireless sub-packet. Data packets from mobile devices to the module must be sub-packed by their own (into 1 – 20 bytes/packet) before sending. The module will forward them to the host serial RXD end in turn, when receiving the packets.

1. The serial port hardware protocol: 115200 bps, 8, no parity, 1 stop bit.
2. When EN is set at high level, the Bluetooth module is in full sleep mode. When EN is set low, the module will start broadcasting at the internal of 200 ms, until it pairs with mobile devices. When EN jumps from low to high, the module will enter into sleep mode immediately, regardless of current status.
3. After the module is connected, BRTS needs to be pulled low if the host (MCU) has data to send to the BLE module, and the data transmission can be started around 100 μs afterwards. BRTS should be raised high by the host after transmission finishes and make the module exit the serial RX mode. Pay attention to confirm that serial port data transmission has been completely finished before raising BRTS. Otherwise there will be data truncation.
4. When there is data upload request, the module will set BCTS low, until data transmission finishes. The transmission can start at least 500 us afterwards. And this delay can be configured through the AT command (see in section "serial AT command”). BCTS will be set high by the module when data transmission is finished.
5. If the host BRTS is being kept a low level, the Bluetooth module will always be in serial port RX mode and the power consumption will be high.
6. After the module is connected, a string of "TTM: OK \ r \ n \ 0" will be prompted from TX. The string could be used to determine if the normal transmit operation can be done. Of course the connection status prompt pin can be used instead. Also the connection can be confirmed by

sending a specific confirmation string to the module from mobile devices. When APP actively disconnect the module, there will be a prompt of string "TTM: DISCONNET \ r \ n \ 0" from TX. If the disconnection is abnormal, the string prompt will be "TTM: DISCONNET FOR a TIMEOUT \ r \ n \ 0".

7. The default Bluetooth connection interval is 20 ms. If low-speed TX mode is needed for saving power, connection interval must be adjusted by AT command (longest connection interval to be 2000 ms). In each interval maximum transmission is 80 bytes. Set the connection interval as V (unit: ms), and highest transmit rate per second as V (byte/s), then their relation is as follows:

   V = 80*1000/T (V is only relevant with T)

   If the Bluetooth connection interval of the module is 20 ms, and in each interval it can transmit maximum 80 bytes, the theoretical maximum transmission capacity (transmit rate) will be 80 * 50 = 4 k byte/s. Tests have shown that the packet loss is very little when transmit rate under 2 K/s. For safety's sake, it is suggested to do check-sum and re-transmission processing in the up-layer, no matter for high or low speed transmit applications.

8. Below is an example of the communication with 20 ms connection interval. Configuration can be set by your own. But the lower the transmit rate V0，the less packet leakage.

Communication Ref. Model	BLE Connect Interval	Highest theoretical transmit capacity V (byte/s) V = 80*1000/T	Serial Data Packet Length	Serial Port Transmit Interval TS(ms) When L<80，TS >= T When 80<L<160，TS >= T*2 When 160<L<200，TS >= T*3	Actual transmit rate V0 (byte/s) V0 = L*1000/TS	Remarks
1	20	4K	80	TS >= T, if TS=20ms	80*1000/20 = 4K	TS small, not recommended
2	20	4K	200	TS >= T*3, if TS=60ms	200*1000/60 =3.3K	
3	20	4K	200	TS >= T*3, if TS=70ms	200*1000/70 = 2.8K	
4	20	4K	80	TS >= T, if TS=35ms	80*1000/30 =2.6K	
5	20	4K	70	TS >= T, if TS=30ms	70*1000/30=2.3K	
6	20	4K	60	TS >= T, if TS=30ms	60*1000/30 =2K	
7	20	4K	40	TS >= T，if TS=30ms	40*1000/30 =1.3K	

8	20	4K	20	TS >= T，if TS=30ms	20*1000/3 0 = 666	

Table 5. Example of Communication at the Interval of 20ms

Note:

- Specific communication mode can be designed according to the practical application. Serial-packet length can be designed in between 80 and 200 bytes (large packet transmission) . As per the BLE protocol there is the following relations:

  When L<80，TS >= T；

  When 80<L<160，TS >= T*2；

  When 160<L<200，TS >= T*3；

- Transmission modes that comply with above-said conditions are generally safe in operation. However among them, when TS = T, T * TS = 2 or TS = T * 3, it is workable but not recommended, as the packet loss is relatively high and check-sum re-transmission mechanism must be added. In other words, when a serial data packet is as big as 80 byte < L < 200 byte, serial data can be sent to the module for one time, but certain time needs to be spared for data transmission. Otherwise there will be a rear-end data collision. For example, when the connection interval T = 20 ms, 3 = T * TS must be bigger than 60 ms, if the serial packet length L = 200. So setting TS = 70 ms is a logical choice.

9. The size of the serial data packets can be various and the length can be any value less than 200 bytes, as long as the above-said conditions are met. But in order to utilize the communications payload in highest efficiency, while to avoid communication running in full capacity, it is recommended to use serial data packets of 20,40, or 60 bytes in length, and interval between packets is made more than 20 ms.

Note:

- **Test show that in iOS, calling the writing function to Characteristic with the parameter CBCharacteristicWriteWithResponse (writing mode with response) will reduce partially the transmit efficiency, but the correctness of a single packet will be ensured. While with the parameter CBCharacteristicWriteWithoutResponse (writing mode without response), the transmit efficiency will be increased, but the correctness of data packet needs to be checked by APP in up layer.**

UART AT Command

Strings starting with "TTM" will be regarded as AT commands to be parsed and executed. and will return exactly the same from the serial port. Afterwards the execution result will be output (ie. TTM: OK \ r \ n \ "0" or "TTM: ERP \ r \ n \ 0", etc.) Serial data packets which do not start with “TTM”will be regarded as transparent transmission data.

- **Connection Interval Setting**

Input the following string to the serial port RX to set the BLE connection interval:

"TTM:CIT-Xms"

Where X = "20", "50", "100", "200", "300", "400", "500", "1000", "1500", or "2000" (ms). After the command is executed, the following confirmations will be got from serial port TX:

"TTM: TIMEOUT \ r \ n \ 0" (means timeout and the change failed);

"TTM: OK \ r \ n \ 0" (means the change is successful and the new connection interval is applied);

The success of connection interval setting depends on the constraints of connection intervals by mobile devices. The maximum connection intervals also vary in different version of iOS. Tests with iPhone 4s (iOS 5.1.1) show the fastest is 20ms and the slowest is 2s. On the other hand, due to the BLE protocol internal mechanism, execution efficiency of this command will be different with different connection intervals. In iOS5.1.1, it takes maximally around 100 s, changing from the current connection interval of 2000ms (max. 2000ms) to other connection intervals. While the execution will be fast when executing this AT command in other high-frequency connection intervals.

Note:

- The connection interval setting cannot be saved after power-off. And command of change is only effective when the connection is successful.

- **Module Rename**

Input the following string to the serial RX to rename the module (length of name should not exceed 16 bytes).

" TTM:REN-" + Name

Also confirmation of “TTM: OK \ r \ n \ 0" will be received from TX. And if the command format incorrect, the string as follows will be returned:

"TTM:ERP\r\n\0"

Test shows that device name can’t be changed immediately in iOS system,in Android system it can be changed immediately. Users can set the name on PC or mobile device.(See the “module parameter Settings "【service UUID: 0 xff90】”)The name can be saved after power-off.

- **Baud Rate Setting**

Input the following string to the serial port RX (parameter after "-" being the new baud rate):

"TTM:BPS-115200"

Afterwards confirmation string of "TTM: OK \ r \ n \ 0" will be received from TX. If the value set is not in the options, or the command format is incorrect, the string as follows will be returned:

"TTM:ERP\r\n\0"

Test shows that in iOS5 the baud rate cannot be changed, but can be changed immediately in iOS6 and above versions. Users can set through PC, or through the BLE APP interface of mobile devices. (See the "module parameter Settings "【service UUID: 0 xff90】")

- **Acquiring Physical Address MAC**

Input the following string to a serial port RX:

"TTM:MAC-?"

And the following string will be received from TX:

" TTM:MAC-xxxxxxxxxxxx\r\n\0"

"xxxxxxxxxxxx" after "-" is the Bluetooth address in 6 bytes.

- **Module Reset**

Input the following string to serial port RX will force the module to be soft-reset once:

"TTM:RST-SYSTEMRESET"

- **Broadcast Cycle Setting**

Input the following string to serial port RX, to set the broadcast cycle of the module, T = X * 100 ms

"TTM:ADP-(X)"

Where X = "2", "5", "10", "15", "20", "25", "30", "40" or "50". Confirmation string of "TTM: OK \ r \ n \ 0" will be received from TX. If the command format incorrect, the following string will be returned:

"TTM: ERP\r\n\0"

Broadcast cycle setting can be saved after power-off. After the module is rebooted, the module will broadcast as per the new broadcast cycle.

- **Additional Customized Contents of Broadcast**

Input the following string to the serial port RX to customize broadcast contents

"TTM:ADD-"+ Data

Where "Data"is the additional data ready to be broadcast (0 < Length < = 16 bytes). The confirmation string of "TTM: OK \ r \ n \ 0" will be received from TX. If the command format incorrect, the following string will be returned :

"TTM:ERP\r\n\0"

It takes immediate effect when command is executed. Certain customized contents can be broadcast in this way. And the data can be saved after power-off. If setting all 16 bit data as 0, customized broadcast data will not be used. Instead, the default broadcast contents are applied.

➢ **Defining Product Identification Code**

Input the following string to the serial port RX to define product identification code:

"TTM:PID-"+ Data

where “Data” is for a two-byte product identification code (ranging from 0x0000 range to 0xFFFF and length =2). The confirmation string of "TTM: OK \ r \ n \ 0" will be received from TX. If the command format incorrect, the following string will be returned:

"TTM:ERP\r\n\0"

This identification code can be saved after power-off. It will show in the broadcasting, and can be used to filter devices or to determine if it is a specific product.

➢ **Transmission Power Setting**

Input the following string to the serial port RX to set the corresponding transmission power (in dBm).

"TTM:TPL-(X)"

Where X = "+ 4", "0", "6", or "- 23". The confirmation string of “TTM: OK \ r \ n \ 0" will be received from TX. And the module will immediately communicate with the new transmission power. If the command format incorrect, the following string will be returned:

"TTM:ERP\r\n\0"

Note: this setting cannot be saved when power-off.

Broadcast Data Setting

➢ **Default broadcast data**:

When the module EN pin is set low, or into the TEST mode (plugged after set low), the module will broadcast at an interval of 200 ms. In the domain of the broadcast data GAP_ADTYPE_MANUFACTURER_SPECIFIC (iOS officially defined programming macro), the following contents are included (default of 3 bytes):

{

0x00,0x00, Customizing equipment type code, default setting 00 00, and can be set by the AT command;

0x00, Percentage of module power suppl (2.0 v = 0%);

}

➢ **Custom broadcast data:**

If you use the AT command custom the broadcast content, a maximum length of 16 bytes (Blue font part), in the broadcast data GAP_ADTYPE_MANUFACTURER_SPECIFIC domain will contain the following content, length is 2 + n bytes:

{

0x00,0x00, Custom coding equipment type, the default is 00 00, can be set by the AT command;

Data [n], Custom broadcast data, n < = 16;

}

Note: The broadcast data can be modified by the AT command and saved after power-off. After power on again, last-time customized broadcast data will be used. If customized broadcast data is set all 0 (16 byte), the customized broadcast will not be used but the system default broadcast contents. To avoid the too long broadcast data to cause extra power consumption, you can also set the customized broadcast data to be any value of 1 byte.

System Reset and Recovery

There are two methods of module reset:

1. Use the AT command to reset module (see the section UART AT Command);
2. Use an APP to reset module from remote, through the service channel interface (see the section BLE Protocol Descriptions (APP Interface) - Module Parameter Settings ").

◆ **System parameters, system parameters that will be restored in the deep recovery include:**

A. Serial port baud rate, recovering to 9600 bps.
B. Device name, recovering to "TAv22u - XXXXXXXX" and X is the last four bytes;
C. Broadcast cycle, recovering to 2 (200 ms);
D. Production ID, recovering to 0 x00, 0 x00;
E. Customized broadcasting length, recovering to 0;
F. Customized broadcast data, recovering to all 0 (meaning to use default broadcast data i/o customized broadcast data);
G. EN mode recovering to 0, and level-enabled mode in use by default;

iOS APP Programming Reference

The module is always to broadcast as slave, waiting for Smart phone to scan and connect as master. The scanning and connection is usually completed by APP. Due to the particularity of BLE protocol, there is no need to scan and connect Bluetooth LE devices in the system settings of the Smart phone. Smart devices are responsible for BLE connection, communication, disconnection, and etc. And usually it is implemented by the APP.

Regarding BLE programming in iOS, the key point is the **reading, writing and enabling notify switch** to Characteristic (or called channel) to. To read and write to the channel can realize the direct control on the direct-drive mode functions of the module and no extra CPU is needed. Typical functions that are involved are as follows:

```
/*!
* @method writeValue:forCharacteristic:withResponse:
* @param data The value to write.
* @param characteristic The characteristic on which to perform the write operation.
* @param type The type of write to be executed.
* @discussion Write the value of a characteristic.
* The passed data is copied and can be disposed of after the call finishes.
* The relevant delegate callback will then be invoked with the status of the request.
* @see peripheral:didWriteValueForCharacteristic:error:
*/
- (void)writeValue:(NSData *)data forCharacteristic:(CBCharacteristic *)characteristic
type:(CBCharacteristicWriteType)type;
```

Note: to write to a characteristic

```
NSData *d = [[NSData alloc] initWithBytes:&data length:mdata.length];
[p writeValue:d
forCharacteristic:c
type:CBCharacteristicWriteWithoutResponse];
```

```
/*!
* @method readValueForCharacteristic:
* @param characteristic The characteristic for which the value needs to be read.
* @discussion Fetch the value of a characteristic.
* The relevant delegate callback will then be invoked with the status of the request.
* @see peripheral:didUpdateValueForCharacteristic:error:
*/
- (void)readValueForCharacteristic:(CBCharacteristic *)characteristic;
```

Note: to read a characteristic

```
[p readValueForCharacteristic:c];
```

```
/*!
* @method setNotifyValue:forCharacteristic:
```

```
* @param notifyValue The value to set the client configuration descriptor to.
* @param characteristic The characteristic containing the client configuration.
* @discussion Ask to start/stop receiving notifications for a characteristic.
* The relevant delegate callback will then be invoked with the status of the request.
* @see peripheral:didUpdateNotificationStateForCharacteristic:error:
*/
- (void)setNotifyValue:(BOOL)notifyValue forCharacteristic:(CBCharacteristic *)characteristic;
```

Note: to open a characteristic notify enable switch.

```
[self setNotifyValue:YES forCharacteristic:c];//open notify enable switch.
[self setNotifyValue:NO forCharacteristic:c]; //close notify enable switch.
/*
* @method didUpdateValueForCharacteristic
* @param peripheral Pheripheral that got updated
* @param characteristic Characteristic that got updated
* @error error Error message if something went wrong
* @discussion didUpdateValueForCharacteristic is called when CoreBluetooth has updated a
* characteristic for a peripheral. All reads and notifications come here to be processed.
*
*/
- (void)peripheral:(CBPeripheral *)peripheral didUpdateValueForCharacteristic:(CBCharacteristic *)
characteristic error:(NSError *)error
```

Note: after each reading operation, this callback function will be performed. The application layer save the data that is read in this function.

About the details of scanning, connecting, and other communication operations, please refer to the test App source code (BLE Transmit Module v1.29) for transparent transmission in IOS, by RF-Star Technology, in which it realizes, for FFE9 and FFE4, the operations of data transmit from BLE to serial port and from serial port to BLE characteristics (notify and write). Other controls on direct-drive functions are similar, all by reading or writing to certain characteristic. The only difference is the characteristic UUID and the bytes of reading and writing operations.

BLE Protocol Description (APP Interface)

➢ **Bluetooth data channel 【service UUID: 0xFFE5 】**

Characteristic UUID	Operation	Bytes	Default Value	Remarks
FFE9 (handle:0x0013)	Write	20	NO	Written data will output from serial port TX

Directions: Bluetooth input data will be transmitted to serial port TX output. APP write to this channel through BLE API, and the data will be output from serial port TX. For operation details, please see the section “Protocol Description - UART Transparent Transmission (Bridge Mode)”.

➢ **Serial data channel【service UUID:0xFFE0】**

Characteristic UUID	Operation	Bytes	Default Value	Remarks
FFE4	Notify	20	NO	Notification will be generated from the input data of serial RX input and sent to smart devices

Directions: Serial input data will be transmitted to BLE output. If notify EN switch of FFE4 is opened, a notify event will be generated in this channel when the host CPU transmit legal data to module RX through serial port. App can directly process and use it in the callback function. For operation details, please see the section “Protocol Description - UART Transparent Transmission (Bridge Mode)”.

➢ **Battery Power Report【Service UUID：0x180F】**

Characteristic UUID	Operation	Bytes	Default Value	Remarks
2A19	Read/ Notify	1	Percentage of power remaining	Reading the current battery power remaining percentage, or automatically creating notification

Directions: The channel for battery power reading or notifying.

The APP reads 2A19 through the BLE API interface, to obtain the percentage of the current power supply to the module. If the notify-enabled function of the channel is switched on , every time the batter power information is read, a notify event will be created in this channel together with the power remaining percentage (maximum of 100% (3 v), and minimum of 0% (2 v)). The APP can directly process and use the data in the callback function

- **Module parameter settings【Service UUID：0xFF90】**

Characteristic UUID	Operation	Can Be Saved	Bytes	Default Value	Remarks
FF91	Read/ Write	Yes	16	Tv401u-x xxxxxxx (ASCII string with terminator)	Device name, XXXXXXXX for the last four bytes of the physical address
FF92	Read/ Write	No	1	2	Bluetooth connection interval: 0: 20ms / 1: 50ms / **2: 100ms** / 3: 200ms / 4: 300ms / 5: 400ms / 6: 500ms / 7: 1000ms / 8: 1500ms/9: 2000ms
FF93	Read/ Write	Yes	1	1	Set the baud rate of serial ports: 0: 4800 bps / **1: 9600 bps** / 2: 19200 bps / 3: 38400 bps / 4: 57600 bps / 5: 115200 bps
FF94	Write	-	1	no	Channel to control remote reset and recovery: - Remote reset control, by writing 0x55 to reset the module - Remote deep recovery control, by writing 0x36 to deep-recover the module (all back to factory settings) and reset
FF95	Read/ Write	Yes	1	0	Set the broadcast cycle: 0: 200 ms / 1: 500 ms / 2: 1000 ms / 3: 1500 ms / 4: 2000 ms / 5: 2500 ms / 6:

					3000 ms / 7: 4000 ms / 8: 5000 ms
FF96	Read/ Write	Yes	2	0x0000	Set product identification code
FF97	Read/ Write	No	1	1	Set the transmission power: 0: +5 dBm / **1: 4 dBm** / 2: 3 dBm / 3: 2 dBm / 4: 1dBm / 5: 0 dBm / 6:-3 dBm / 7: -6dBm / 8:-9dBm / 9:-12dBm / 10:-15dBm /11:-18dBm / 12:-21dBm
FF98	Read/ Write	Yes	16	Default broadcast content.	Set customized broadcast data Customizing broadcast data: 0 < n <= 16 See the section “Broadcast Data Setting”

Directions: Module information configuration channel.
FF91 is the channel for setting device names. Reading and writing to this channel can obtain and set the module name. The length of the name set must meet the condition: 0 < L<17. **And the name is suggested to end with the terminator (' \ 0 ').** The default name is "Tvvvvu - XXXXXXXX \ 0" (16 byte), where vvvv is the current firmware version number and XXXXXXXX is the last four bytes of the MAC address.

FF92 is the channel to set the connection interval. The interval of connection between mobile devices and the module can be set by writing to this channel. Thus the device power consumption and the data throughput can be controlled in a flexible way. In order to raise the connection speed, the setting of connection interval will not be saved. It will always work at the default value (20ms) after power on. After writing to this channel，you will the confirmation from TX:

"TTM:TIMEOUT\r\n\0" means modify overtime,failed;

"TTM:OK\r\n\0" mean modify successful,working in new connection interval;

Test shows that it takes around 30s to wait when the connection interval is changed from 500 ms to another by iPhone 4S (iOS 5.1.1). But it will be very quick if the connection interval is changed from a high frequency one (ie. 20ms), resulting from BLE protocols.

FF93 is the channel to set the baud rate of module serial ports. Baud rate of the module’s universal serial ports can be set by reading and writing to the channel. The new baud rate will take effect in two seconds after setting and can be saved after power-off. The default factory setting is 1 (9600

bps).

FF94 is the channel to control the remote reset and recovery.Various controlling functions can be realized by writing different values to the channel.

1. **Write 0x55 to this channel will soft-reset the module.**
2. **Writing 0x36 to the channel will deep-recover the module. All system settings will be recovered to the factory defaults and the module will be reset afterwards.**

FF95 is the channel to set the broadcast cycle of the module. Broadcast cycle can be set by reading and writing to this channel.The setting can be saved after power-off. And the default factory setting is 0 (200 ms).

FF96 is the channel to set the product identification code of the module, by reading and writing to the channel. The APP can filter and connect to specific product type through this code. The setting can be saved after power-off. And the default factory setting is 0 x0000.

FF97 is the channel to set the transmission power of the module, by writing to this channel.The setting **cannot be saved** after power-off. And the default factory setting is 5 (0 dBm).

FF98 is the channel to set the broadcast contents of the module. Broadcast data can be customized by writing to this channel. The setting can be saved after power-off. When the data is all 0 (16 byte), it is regarded that default broadcast data is used, instead of customized data. (see the section "Broadcast Data Setting").

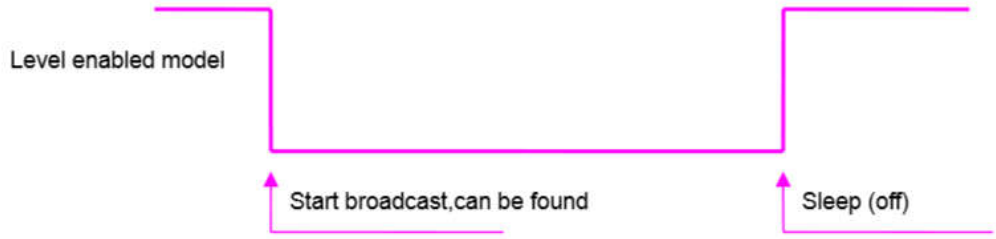


Figure. 9 Level Enabled Model

In level-EN mode, broadcasting (so can be found and connected) has the following features:

1. If EN pin is enabled (set low), the module will keep broadcasting,until it is connected or EN is set high.
2. Regularly disconnected or timeout disconnected, as long as EN is set low,the module will always keep broadcasting, until it is connected again.

In level-EN mode when P04 works as signal prompt pin (prompt of Bluetooth connection status by default) and is connected to output low level;When P05 works as link prompt pin,when Bluetooth connected output low level, if Bluetooth is disconnected (either timeout or active disconnecting) and

not re-connected, output high level.

When Bluetooth is disconnected for timeout, it will output the square wave of 2Hz and last for 2 minutes. During this period, it will keep broadcasting and cannot be shut down, until the module re-connects with master device.
Broadcasting status & IO6 prompt ways in different EN modes are summarized as follows:

In level-EN mode,the notifications of P04 and P05：

Enabled but Not Connected		Connected		Actively Disconnected		Timeout Disconnected	
P04 Prompt Way	P05 Prompt Way	P04 Prompt Way	P05 Prompt Way	P04 Prompt Way	P05 Prompt Way	P04 Prompt Way	P05 Prompt Way
Broadcasting in low level	Unconnected in high level	Unbroadcasting in high level	Connected in low level	Broadcasting in low level	Unconnected in high level	Broadcasting in low level	Unbroadcasting in high level

Device information【Service UUID：0x180A】

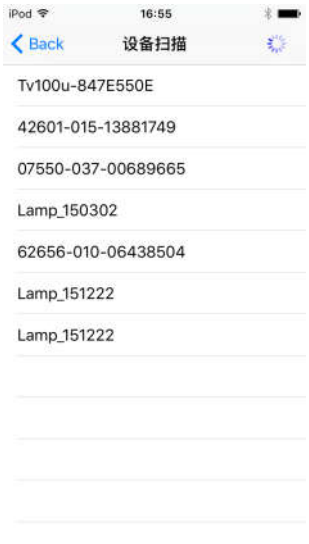
Characteristic UUID	Operation	Bytes	Default Value	Remarks
2A23	Read	8	xxxxxx0000xxxxxx (Hex)	System ID, where xxxxxxxxxxxx is the physical address of module chip, with low byte in front
2A26	Read	5	V4.01u (ASCII)	Software version number of module

Directions: Module information reading channel.
2A23 is the channel to obtain the module information. Reading this channel can get the module ID, in the format of xxxxxx0000xxxxxx, where xx part is the physical address MAC of the module chip in six bytes (low byte in the front).
2A26 is the channel to read the software version number of the module. Reading this channel will get the module software version, in the format of Vx.xx where x.xx stands for the firmware version number.

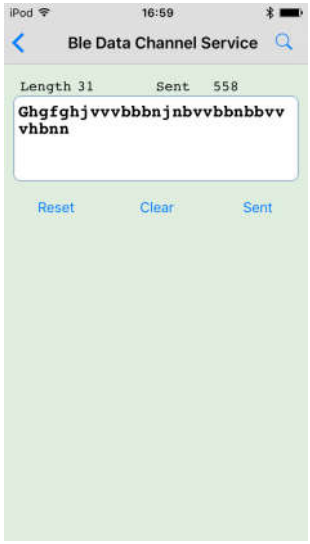
Transmission test by APP

Module test tool (APP) for the IOS platform can be downloaded in AppStore. Running AppStore in iPhone 5S, iPhone 6.... and searching CubicBLE, the APP can be found,downloaded and installed. (source code is available if needed).

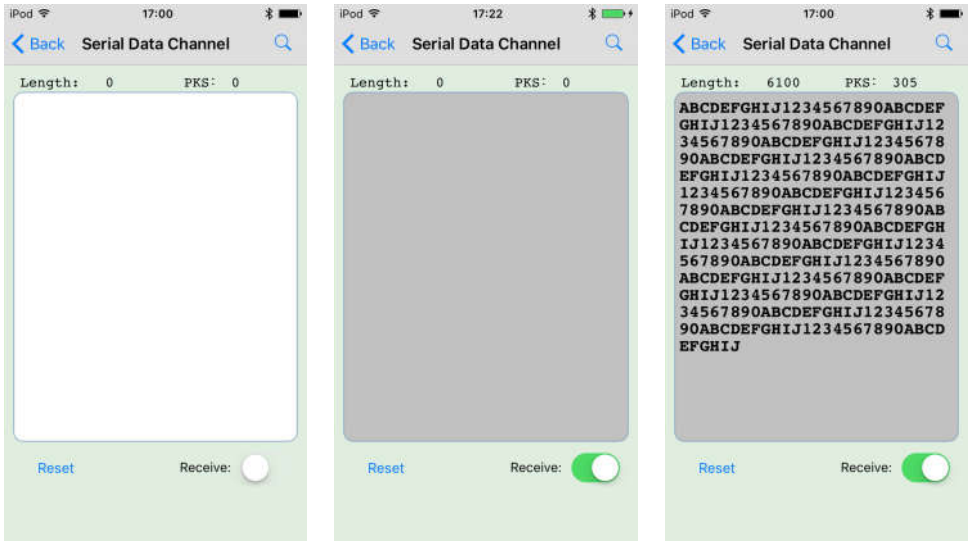
When the APP is installed and run, open Bluetooth,select the blue button on the up right corner,there will be automatic scanning and the devices scanned and found will be shown in a list (probably will be a prompt to ask for Bluetooth enabling). Clicking the name of a device in the list, the APP will try to connect it. And when the connection is successful,the APP will jump to main UI of control.



At this time if the serial port of the module is ready (which means the main CPU or the serial terminal is connected),work can start to perform manual and automatic transceiving test. Select Bluetooth data channel,input the data and then click”Sent”,the data will be sent to the module.



Selectserial data channel,click"Receive",then receive the data from module.



Notes: if a serial terminal is used for test, the data from the terminal must be sent to the mobile phone, **and BRTS must be set low**, in case the module will enter into sleep mode.Regarding IOS programming, as per the Bluetooth protocol, Mobile device and send data by writing to the corresponding service (UUID) of channel B (transmission). Transmission of module data to mobile device is done in the way of notification. So the notification of the corresponding service (UUID) of Channel A (receive) needs to be enabled, after the APP is started. Then the data packet will be sent automatically from module's serial port to the mobile device.

Master reference code (transparent transmission)

Logical relation: The two IOs – BCTS and BRTS – are used for notifying and controlling of transceiving.These two IOs are normally set high, and triggered when setting low. If the module needs to send data, set BCTS low to inform MCU to receive. If MCU needs to send data, set BRTS low to inform the module to receive. Schematic codes are as follows:

```
void main(void)
{
  EN = 0 ;                                  //Enable,start broadcast

  while(!BLEMoudleAck("TTM:OK\r\n\0"));     //Waiting for Phone scan,Connecting

                                            //Waiting for the connection is
                                            successful,Also can add wait time limit.

                                            //Also can judge connection prompt
                                            signal
                                            level


 BRTS = 0;                                  //Low BRTS, Notice CC2640 module is
ready to receive

  halMcuWaitMs(2);                          //Delay for 2ms
  UARTWrite( HAL_UART_PORT_0, "TTM:CIT-100ms", 14);
                                            //modify connect interval,from the serial port
                                            to be confirmed.
 halMcuWaitMs(5);                           //Delay for 5ms to ensure the data has been
                                            issued
  BRTS = 1;                                 //RTS high and sent
       while(!BLEMoudleAck("TTM:OK\r\n\0"));//Waiting to set sucess,also can add wait
                                            time limit.

  while(1){                                 //Loop transceiver test.
   while(1){
    if(BCTS == 0){                          //Testing,if BCTS low is ready to receive.
      while(BCTS==0);                       //Waiting to be sent,but also timed wait.
```

```
        if(UARTRead(uartBuffer) == SUCCESS)        //A serial port to read data.
        {... ...}                                  //Use data.
      }
      BRTS = 0;                                    //RTSlow,and notice CC2640to receive.
      halMcuWaitMs(2);                                 //Delay for 2ms
      send_TX("1234567890");                       //Send any data.（Within 200byte）
        halMcuWaitMs(5);                                 //Delay for 5ms to ensure the data has
                                                                                  been issued
      BRTS = 1;                                    //RTS high,sent
       halMcuWaitMs(20);                             //Delay again to send next packet,and delay
                                                                        depanding on the packet size.
      }
    }
}}
```

Contact us

深圳市信驰达科技有限公司

SHENZHEN RF STAR TECHNOLOGY CO.,LTD.

Tel：0755-8632 9829（Sales） 0755-36953756（FAE） Web: www.szrfstar.com

Fax：0755-8632 9413 E-mail：sales@szrfstar.com

地址：深圳市宝安区宝源路互联网产业基地 A 区 8 栋 2 楼

Add：2F,Block8,Dist.A,Internet Industry Base,Baoyuan Road ,Baoan Dist,Shenzhen