QUINTIC

QS932x Bluetooth 4.0 Low Energy Module

User Manual

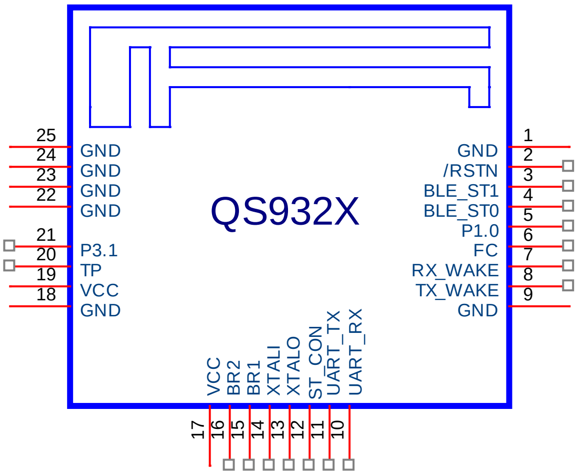
Version 0.2

VERSION HISTORY

Version	Comment
0.1	First draft
0.2	Add command(set ADV user data) and event(ADV user data)

1. IO Define

1.1 Pin out



Pin out

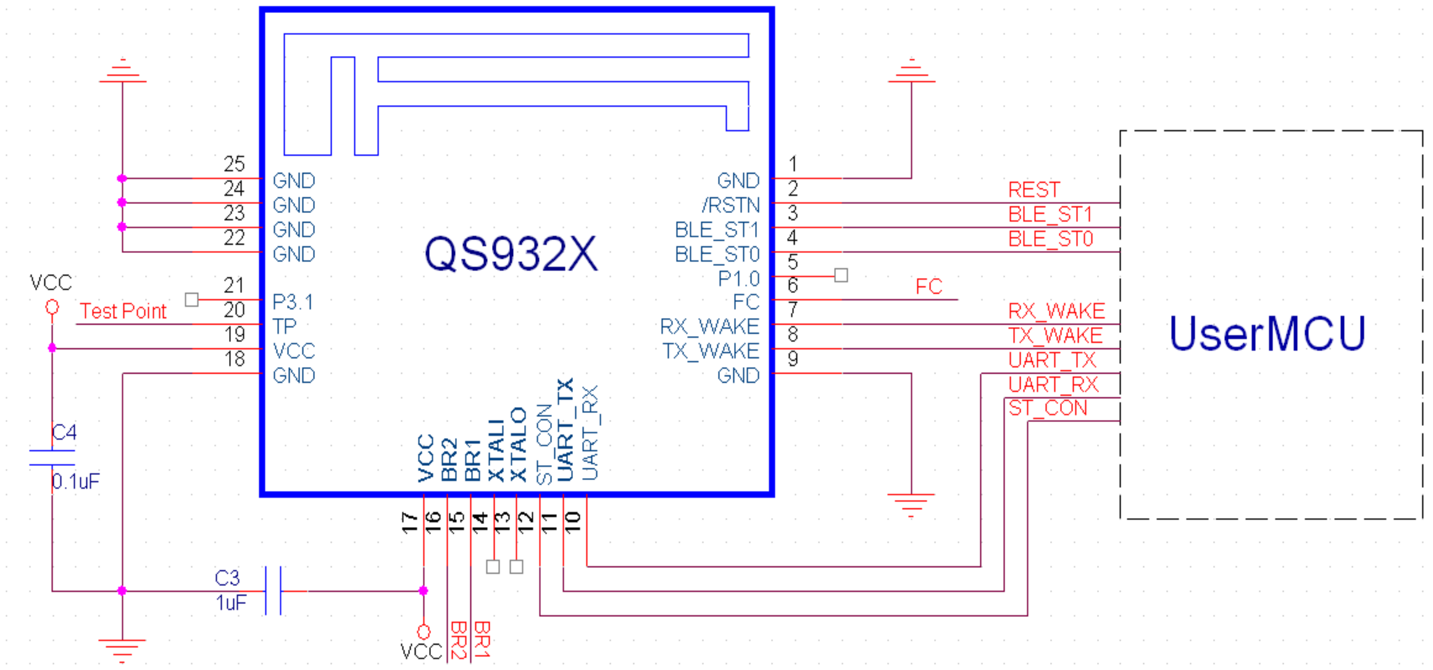
1.2 Pin Description

<table>
<tr><th>Pin NO</th><th>NAME</th><th>Director</th><th>Description</th></tr>
<tr><td><b>1</b></td><td>GND</td><td>Ground</td><td>Should be connected to ground plane on application PCB</td></tr>
<tr><td><b>2</b></td><td>/RSTN</td><td>Digital Input</td><td>Hardware reset, active low. （option connected）
If not used, leave it unconnected.</td></tr>
<tr><td><b>3</b></td><td>BLE_ST0</td><td>Output</td><td rowspan="2"><table><tr><th colspan="3">QS932X State indicate</th></tr><tr><td>ST1</td><td>ST0</td><td>State</td></tr><tr><td>0</td><td>0</td><td>Sleep</td></tr><tr><td>0</td><td>1</td><td>ADV</td></tr><tr><td>1</td><td>0</td><td>Con_busy</td></tr><tr><td>1</td><td>1</td><td>Con_idle</td></tr></table>
Hardware flow control : indicate QS932X state
Software flow control : unconnected</td></tr>
<tr><td><b>4</b></td><td>BLE_ST1</td><td>Output</td></tr>
<tr><td><b>5</b></td><td>P1.0</td><td>Digital in/out</td><td>No function. Unconnected.</td></tr>
<tr><td><b>6</b></td><td>FC</td><td>Input (internal pull-up)</td><td>Used to select the flow control mode.
Hardware flow control : Connect to VCC or unconnected
Software flow control : Connect to GND</td></tr>
<tr><td><b>7</b></td><td>RX_WAKEUP</td><td>Input</td><td>UART receive wakeup pin.
If don’t need QS932X work in low power mode, connect this pin to GND.</td></tr>
<tr><td><b>8</b></td><td>TX_ WAKEUP</td><td>output</td><td>UART transmit wakeup pin. (option connected)</td></tr>
</table>

<table>
<tr><td>9</td><td>GND</td><td>Ground</td><td>Should be connected to ground plane on application PCB</td></tr>
<tr><td>10</td><td>UART_RX</td><td>Input</td><td>UART receive data pin</td></tr>
<tr><td>11</td><td>UART_TX</td><td>output</td><td>UART transmit data pin</td></tr>
<tr><td>12</td><td>ST_CON</td><td>Input</td><td>QS932X state control pin, active falling edge.</td></tr>
<tr><td>13</td><td>XTALO</td><td></td><td rowspan="2">Don’t connect.</td></tr>
<tr><td>14</td><td>XTALI</td><td></td></tr>
<tr><td>15</td><td>BR0</td><td>Input (internal pull-up)</td><td rowspan="2"><table>
<tr><th colspan="3">Baud rate pin</th></tr>
<tr><td>BR1</td><td>BR0</td><td>Baud rate</td></tr>
<tr><td>0</td><td>0</td><td>2400</td></tr>
<tr><td>0</td><td>1</td><td>9600</td></tr>
<tr><td>1</td><td>0</td><td>38400</td></tr>
<tr><td>1</td><td>1</td><td>115200</td></tr>
</table>
1: connect to VCC or unconnected
0: connect to GND</td></tr>
<tr><td>16</td><td>BR1</td><td>Input (internal pull-up)</td></tr>
<tr><td>17</td><td>VCC</td><td>Power</td><td>2.5-3.6V</td></tr>
<tr><td>18</td><td>GND</td><td>Ground</td><td>Should be connected to ground plane on application PCB</td></tr>
<tr><td>19</td><td>VCC</td><td>Power</td><td>2.5-3.6V</td></tr>
<tr><td>20</td><td>TP</td><td>Input</td><td>Test point .Used to DTM test.</td></tr>
<tr><td>21</td><td>P3.1</td><td>Digital in/out</td><td>No function ,keep unconnected</td></tr>
<tr><td>22</td><td>GND</td><td>Ground</td><td rowspan="4">Should be connected to ground plane on application PCB</td></tr>
<tr><td>23</td><td>GND</td><td>Ground</td></tr>
<tr><td>24</td><td>GND</td><td>Ground</td></tr>
<tr><td>25</td><td>GND</td><td>Ground</td></tr>
</table>

2. HWFC-Hardware Flow Control

2.1 Schematic Reference



Note:

BR2	BR1	BautRate
NC/1	NC/1	115200
NC/1	0	38400
0	NC/1	9600
0	0	2400

FC	FlowControl
NC/1	Hardware FC
0	Software FC

2.2 Command and Event

TYPE	ID	LEN(Byte)	PARAMS	Explanation
CMD 0XEA	Set Device Name (0x05)	0x01-0x10	Params[] = “QPPS”	Default:QPPS
	Peripheral update CONN parameter (0x08)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Default：Min=0x0018 (0x0008*1.25ms) Max=0x0028 (0x0010*1.25ms)
	Peripheral update Adv Interval (0x09)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Default: Interval Min=0x0030 (0x0030*0.625ms) Interval Max=0x0064 (0x0064*0.625ms)
	Read Address (0x0B)	0x00	No parameters	
	Set Tx Power (0x0C)	0x01	0=<param0<=11 power[param0]={-20,-18,-16,-14,-12,-10,-8,-6, -4,-2,0,2}	Default：param0 = 0x0A (0dbm)
	Set TX wakeup timer (0x0f)	0x01	0=<param0<=255 (ms)	Default：param0 = 0ms;

	Set ADV user data
	(0x10)
		0x03-0x0d	param0 = x (length of data+1)
			param1 = AD type (recommend 0xff)
			param2~paramn: data
				0xff(AD type) :Manufacturer Specific Data
				0x16 (AD type): Service Data
				For example : 0x06 0xff 0x11 0x22 0x33 0x44 0x55

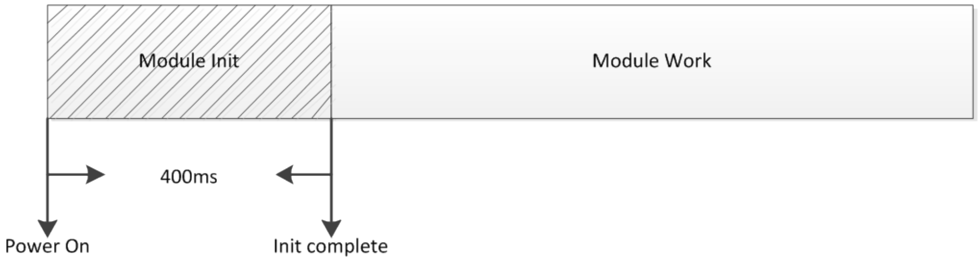
EVENT 0XED	Device Name (0x05)	0x01-0x10	Params[] = “QPPS”	Set successful: return the new name Set fail: return the last name
	CONN parameter (0x08)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Update successful: return the new connect parameter Update fail :return the last connect parameter
	Adv Interval (0x09)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Update successful: return the new adv parameter Update fail :return the last adv parameter

	Address (0x0B)	0x06	Params[ ] = {0x08,0x7c,0xbe,xx,xx,xx}	Return the device address.
	Tx Power (0x0C)	0x01	0=<param0<=11 power[param0]={-20,-18,-16,-14,-12,-10,-8,-6,-4,-2,0,2}	Update successful: return the new power value Update fail :return the last power value
	TX wakeup timer (0x0f)	0x01	0=<param0<=255 (ms)	Update successful: return the new wakeup timer Update fail :return the last wakeup timer

	ADV user data
	(0x10)
		0x03-0x0d	param0 = x (length of data+1)
			param1 = AD type (recommend 0xff)
			param2~paramn: data
				Update successful: return the adv user data
				Update fail :return the last adv user data

2.3 Power on reset

User MCU must delay at least **400ms after power on the QS932X. During the time, QS932X is booting and initializing BLE protocol stack.**



2.4 State control

The PIN ST_CON (pin12) is used to change QS932X state. It active at falling edge and need keeping low 5 ms .

The PIN BLE_ST1 (PIN3) and PIN BLE_ST0 (PIN4) are used to indicate the QS932X state.

BLE_ST1	BLE_ST0	QS932X State
0	0	Sleep
0	1	Advertise
1	0	Connected and idle
1	1	Connected and busy

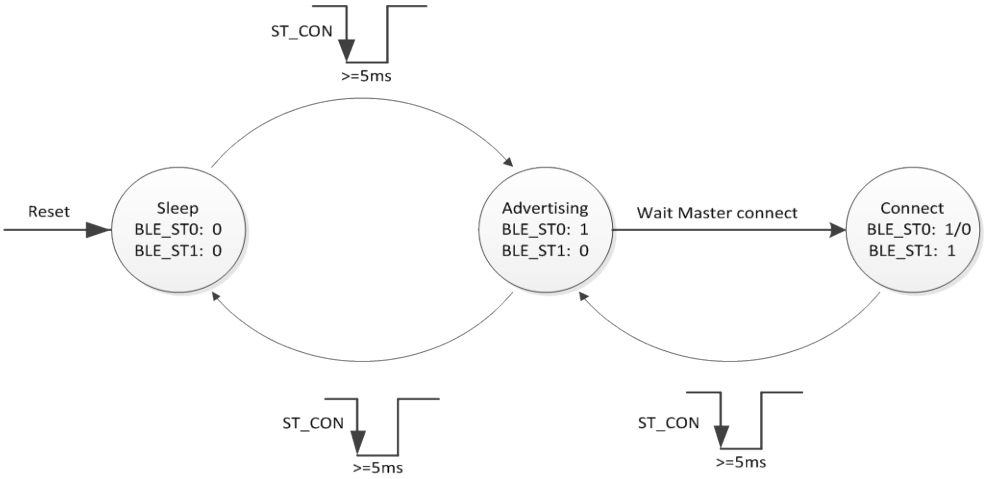
Note:

Sleep---BLE protocol stack is not working.

Advertise--- BLE protocol stack is advertizing, the QS932X can be discovered by BLE master.

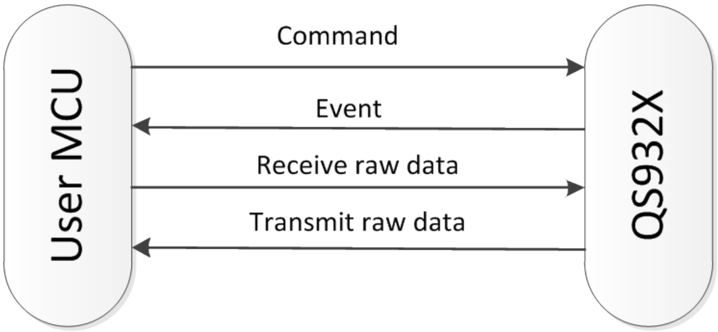
Connected and idle --- The QS932X has connected with BLE master and user MCU can send one frame data (0~120Byte) to the QS932X through UART port.

Connect and busy --- The QS932X has connected with BLE master but UART bus is busy. So user MCU can't send the data to QS932X.

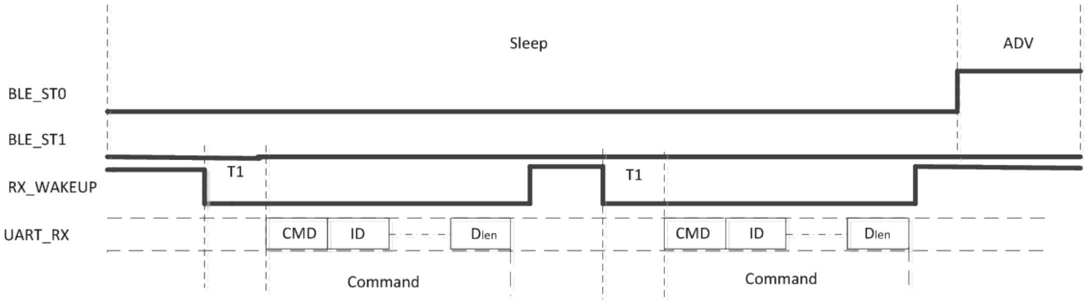


State machine

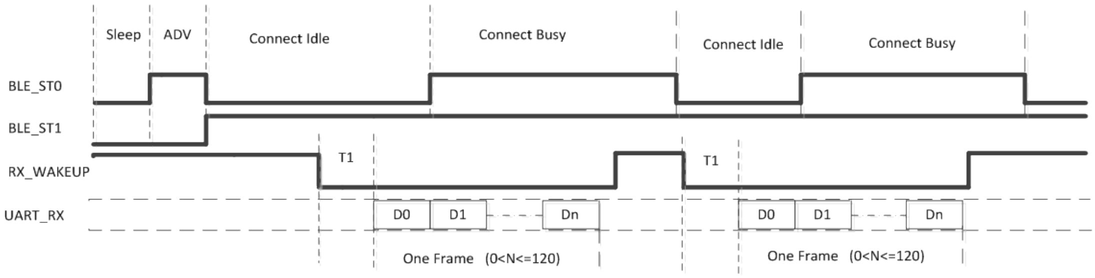
2.5 Communication



- **Command**



- **Receive raw data**



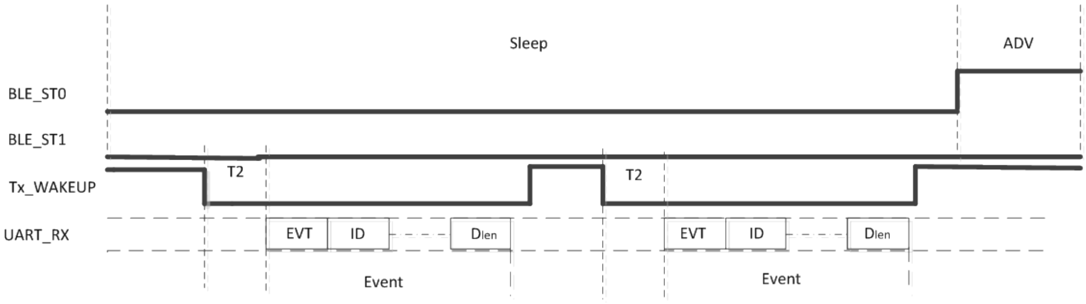
Note:

Command: All the command should be transmitted at sleep state. For the detail, please see 2.2

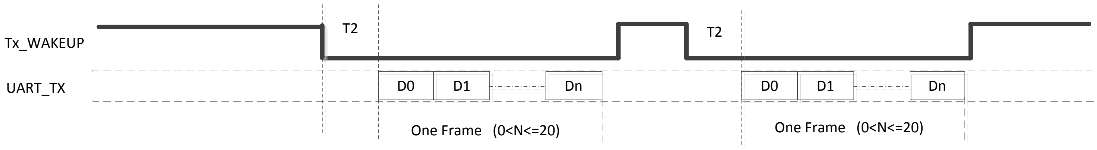
T1: Wakeup timer>=5ms

One Frame: 0~120 Byte can be send once.

- **Event**



- **Transmit raw data**



Note:

T2 : TX wakeup timer , default value is 0ms. The timer value can be set by Command ---- “ Set TX wakeup timer ”.

One Frame: 0~20 Byte can be received once.

2.6 Speed Test

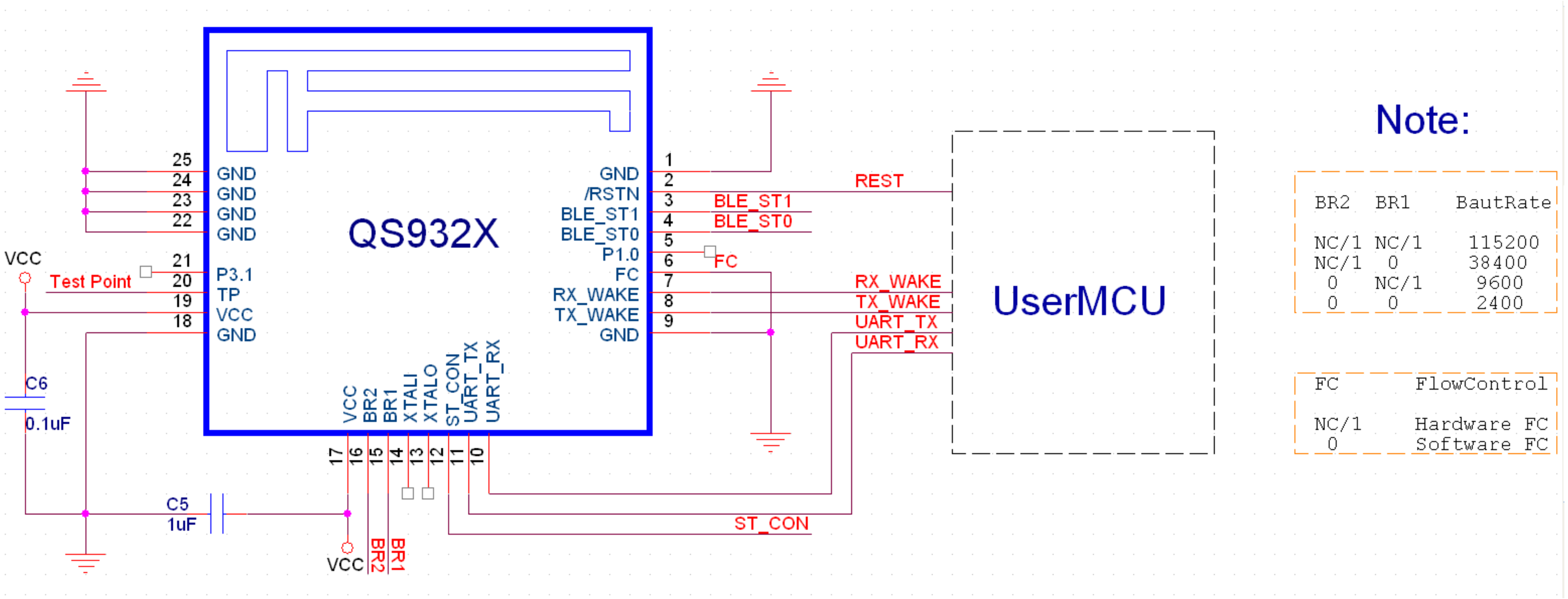
Operating system	Baud Rate	Connect interval(ms)	SPEED	Direction
IOS	115200	18.75	6.4KBps	QS932X to phone
IOS	38400	18.75	3.8KBps	QS932X to phone
IOS	9600	18.75	0.96KBps	QS932X to phone
IOS	2400	18.75	0.24KBps	QS932X to phone
Android	115200	7.5	6.1KBps	QS932X to phone
Android	38400	7.5	3.8KBps	QS932X to phone
Android	9600	7.5	0.96KBps	QS932X to phone
Android	2400	7.5	0.24KBps	QS932X to phone

2.7 Reference code

```
Void main (void)
{
    gpio_write_pin ( MODULE_VCC_EN,HIGH);   //Power on the Module.
    Delay_ms(400);                          // wait 400ms for module power on init.
    uart_rx_enable();                       //enable uart receive.( RX interrupt enable )

    //before TX command to module, should wakeup it, and delay 5ms
    gpio_write_pin(RX_WAKEUP , LOW );
    Delay_ms(5);

    //transmit command--- Set TX wakeup timer
    uart_tx( &CMD_ Set TX wakeup timer[0], sizeof(CMD_ Set TX wakeup timer));
    //transmit command--- updata adv param
    uart_tx( &CMD_ updata_adv_param[0], sizeof(updata_adv_param));

                        |
                        |
                        |
                        |

    // after command send, stop wakeup
    gpio_write_pin(RX_WAKEUP , HIGH );




    //enable advertising
    gpio_write_pin(ST_CON,GPIO_LOW);
    delay_ms(5);
    gpio_write_pin(ST_CON,GPIO_HIGH);

    while(1)
    {
        // read module's state
        if(gpio_read_pin(BLE_ST0) == GPIO_LOW) ble_state&= (~(1<<0));
        else   ble_state|=   (1<<0);
        if(gpio_read_pin(BLE_ST1) == GPIO_LOW) ble_state&= (~(1<<1));
        else   ble_state|=   (1<<1);


        switch(ble_state)
        {
            case SLEEP:
            case ADVERITSE:
            case CON_BUSY:
            break;

            Case CON_IDLE:        //in this state, user can send data to module.
            {
                // before send data, wake up module.
                gpio_write_pin(RX_WAKEUP , LOW );
                Delay_ms(5);
                //send data to module
                uart_tx( &user_data [0], sizeof(user_data));    //len_max = 120byte
                // stop wakeup
                gpio_write_pin(RX_WAKEUP , HIGH );
            }break;
    }

}
```

3. SWFC-Software Flow Control.

3.1 Schematic Reference



BR2	BR1	BautRate
NC/1	NC/1	115200
NC/1	0	38400
0	NC/1	9600
0	0	2400

FC	FlowControl
NC/1	Hardware FC
0	Software FC

3.2 Command and Event

TYPE	ID	LEN(Byte)	PARAMS	Explanation

CMD
0XEA
	Advertising
	(0x01)
		0x01	param1=0 stop advertise
			param1=1 start advertise
				This command’ s response is the QS932X state.

	Connect (0x04)	0x01	param1=0 disconnect command	This command’ s response is the QS932X state.
	Set Device Name (0x05)	0x01-0x10	Params[] = “QPPS”	Default: QPPS
	Peripheral update CONN parameter (0x08)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Default： Min=0x0018 (0x0008*1.25ms) Max=0x0028 (0x0010*1.25ms)
	Peripheral update Adv Interval (0x09)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Default: Interval Min=0x0030 (0x0030*0.625ms) Interval Max=0x0064 (0x0064*0.625ms)
	Read Address (0x0B)	0x00	No parameters	
	Set TX Power (0x0C)	0x01	0=<param0<=11 power[param1]={-20,-18,-16,-14,-12,-10,-8,-6, -4,-2,0,2}	Default： param0 = 0x0A (0dbm)
	Read Module state (0x0D)	0x00	No parameters	This command’ s response is the QS932X state.
	Read RSSI Value (0x0E)	0x00	No parameters	This command’ s response is the QS932X TX power
	Set TX wakeup timer	0x01	0=<param0<=255 (ms)	Default： param0 = 0ms;

	(0x0f)			

	Set ADV user data
	(0x10)
		0x03-0x0d	param0 = x (length of data+1)
			param1 = AD type (recommend 0xff)
			param2~paramn: data
				0xff(AD type) :Manufacturer Specific Data
				0x16 (AD type): Service Data
				For example : 0x06 0xff 0x11 0x22 0x33 0x44 0x55

EVENT 0XED	Device Name (0x05)	0x01-0x10	Param[] = "QPPS"	Set successful: return the new name Set fail: return the last name
	CONN parameter (0x08)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Update successful: return the new connect parameter Update fail :return the last connect parameter
	Adv Interval (0x09)	0x04	Interval Min =param0+(param1<<8) Interval Max =param2+(param3<<8)	Update successful: return the new adv parameter Update fail :return the last adv parameter
	Address (0x0B)	0x06	param [6] = {0x08,0x7c,0xbe,xx,xx,xx}	Return the device address.
	TX Power (0x0C)	0x01	0=<param1<=11 power[param1]={-20,-18,-16,-14,-12,-10,-8,-6, -4,-2,0,2}	Update successful: return the new power value Update fail :return the last power value

	Module state
	(0x0D)
		0x01	Param0=0x00 sleep
			param0=0x01 adv
			param0=0x02 connect full
			param0=0x03 connect empty
				There are 3 reasons for sending the state.
				1. Receive start/stop adv command.
				2. Receive stop connect command
				3. Module's state have changed.

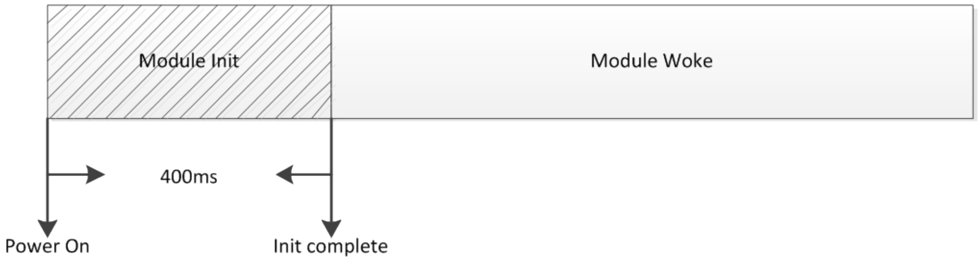
	RSSI Value (0x0E)	0x01		
	TX wake up timer (0x0f)	0x01	0=<param0<=255 (ms)	Update successful: return the new wakeup timer Update fail :return the last wakeup timer

	ADV user data
	(0x10)
		0x03-0x0d	param0 = x (length of data+1)
			param1 = AD type (recommend 0xff)
			param2~paramn: data
				Update successful: return the adv user data
				Update fail :return the last adv user data

DATA_TX **0xEB**	Data (0x01)	0x01-0x78	Param[0]~Param[119]	User MCU send data to QS932X. The length of frame should less than 121 byte.
DATA_RX **0xEC**	Data (0x01)	0x01-0x14	Param[0]~Param[19]	QS932X data send to user MCU. And one frame length will be less than 21 byte.

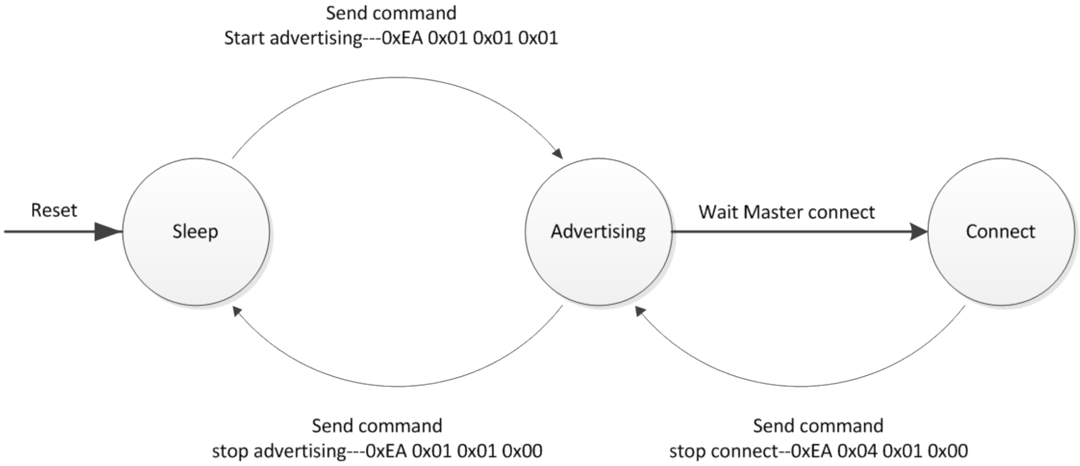
3.3 Power on reset

User MCU must delay at least **400ms after power on the QS932X. During the time, QS932X is booting and initializing BLE protocol stack.**



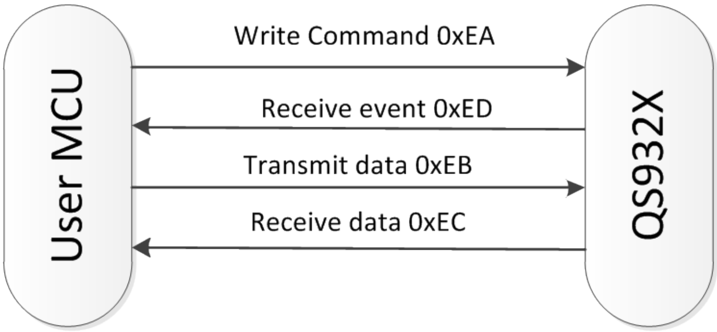
3.4 State control

User MCU can use command--- [**Advertising]** and command--- [**connect]** to control QS932X state. QS932X will return event--- [**Module state]** to indicate current state.

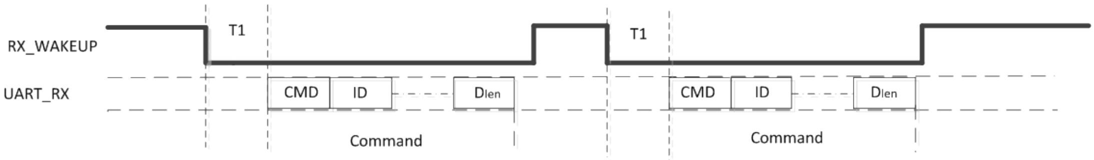

Send command
Start advertising---0xEA 0x01 0x01 0x01
Reset
Sleep
Advertising
Wait Master connect
Connect
Send command
stop advertising---0xEA 0x01 0x01 0x00
Send command
stop connect--0xEA 0x04 0x01 0x00

State machine

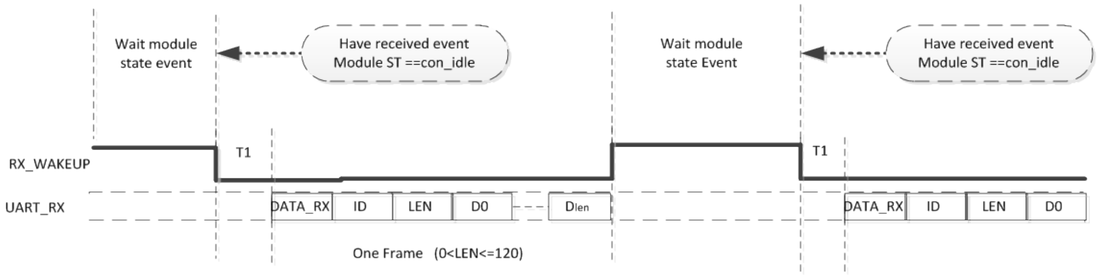
3.5 Communication



- **Command**



- **DATA_RX**

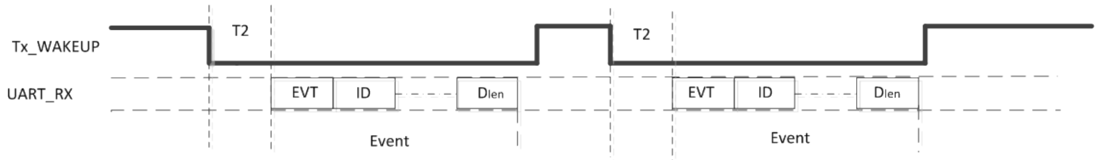


Note:

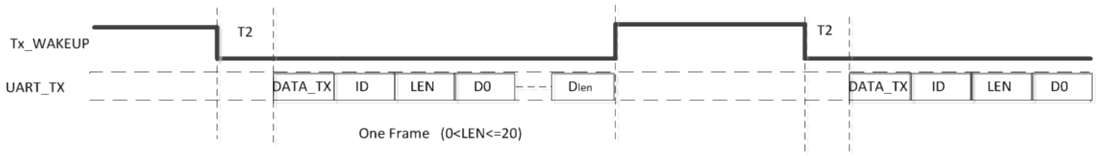
T1: Wakeup timer>=5ms

One Frame: 0~120 Byte can be send once.

- **Event**



- **DATA_TX**



3.6 Speed Test

Operating system	Baud Rate	Connect interval(ms)	SPEED	Direction
IOS	115200	18.75	6.4KBps	QS932X to phone
IOS	38400	18.75	3KBps	QS932X to phone
IOS	9600	18.75	0.9KBps	QS932X to phone
IOS	2400	18.75	0.2KBps	QS932X to phone
Android	115200	7.5	6.1KBps	QS932X to phone
Android	38400	7.5	3.8KBps	QS932X to phone
Android	9600	7.5	0.96KBps	QS932X to phone
Android	2400	7.5	0.24KBps	QS932X to phone

3.7 Reference Code

```
Void main (void)
{
    Uint8_t module_state = 0;

    gpio_write_pin ( MODULE_VCC_EN,HIGH);   //Power on the Module.
    Delay_ms(400);                          // wait 400ms for module power on init.
    uart_rx_enable();                       //enable uart receive.( RX interrupt enable )
```

```
//before TX command to module, should wakeup it, and delay 5ms
gpio_write_pin(RX_WAKEUP , LOW );
Delay_ms(5);

//(1)transmit command--- Set TX wakeup timer
uart_tx( &CMD_ Set TX wakeup timer[0], sizeof(CMD_ Set TX wakeup timer));
//(2)transmit command--- updata adv param
uart_tx( &CMD_ updata_adv_param[0], sizeof(CMD_ updata_adv_param));


//start advertising
uart_tx( &CMD_ Advertising[0], sizeof(CMD_ Advertising));
// after command send, stop wakeup
gpio_write_pin(RX_WAKEUP , HIGH );

while(1)
{
        switch(ble_state)
        {
                case SLEEP:
                case ADVERITSE:
                case CON_BUSY:
                break;

                Case CON_IDLE:        //in this state, user can send data to module.
                {
                        //note:
                        //before send data,module_state's value should be set CON_BUSY manually
                        module_state = CON_BUSY;

                        // before send data, wake up module.
                        gpio_write_pin(RX_WAKEUP , LOW );
                        Delay_ms(5);
                        //send data to module
                        uart_tx( &user_data [0], sizeof(user_data));    //len_max = 120byte
                        // stop wakeup
                        gpio_write_pin(RX_WAKEUP , HIGH );
                }break;
        }

}

Void uart_rx_process(void)
{
        //receive event
        If(event_type == 0xED)
        {
                Switch(event_id)
                {
                        //receive the module state
                        Case (Module state):
                        {
                                module_state = module_st_value;

                        }break;

                }
        }
}
```