User Manual for AT Commands on UART and SPI Interfaces of Fanstel's Nordic and Toshiba Modules

Table Of Contents

1. Introduction

An MCU can use AT Commands to communicate with a Bluetooth 5 module and to send data to a remote device over the Bluetooth 5 interface. Two hardware interfaces are supported:

- UART interface
- SPI interface

If AT commands are preloaded into a module, UART interface is supported. Hex codes for SPI interface can be downloaded from www.fanstel.com.

AT commands are supported for Fanstel BT Series modules using Nordic and Toshiba SoCs. Due to hardware limitation, some AT commands are not supported by some modules.

- BT680F Series Toshiba TC35680 modules (BT680F)
- BT832F Series Nordic nRF52832 modules (BT832, BT832F, BC832, BM832)
- BT832A Series Nordic nRF52810 modules (BT832A, BT832AF, BM832A)
- BT840F Series Nordic nRF52840 modules (BT840, BT840E, BT840F)

Starting with alpha release, AT command hex codes can be downloaded from Fanstel website.

For Nordic modules:

http://www.fanstel.com/download-document/

For Toshiba modules:

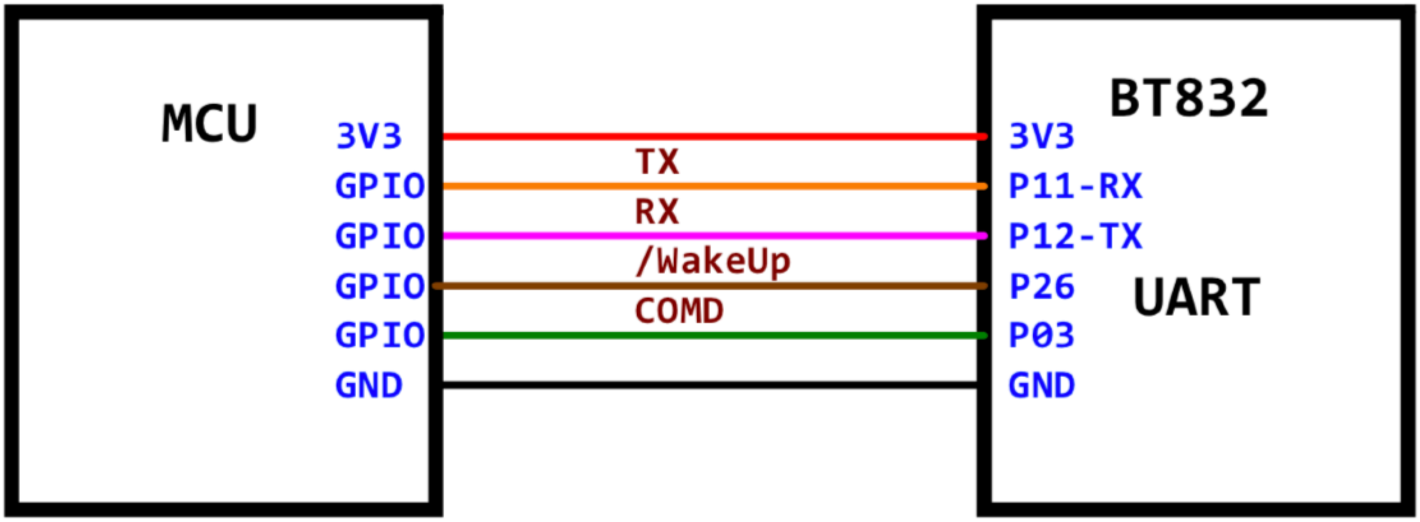
http://www.fanstel.com/download-bluefanmodule-document/

2. Hardware Interfaces

AT command codes for UART interface and SPI interface are different. In addition, connections between host MCU and module is different for SPI and UART interface.

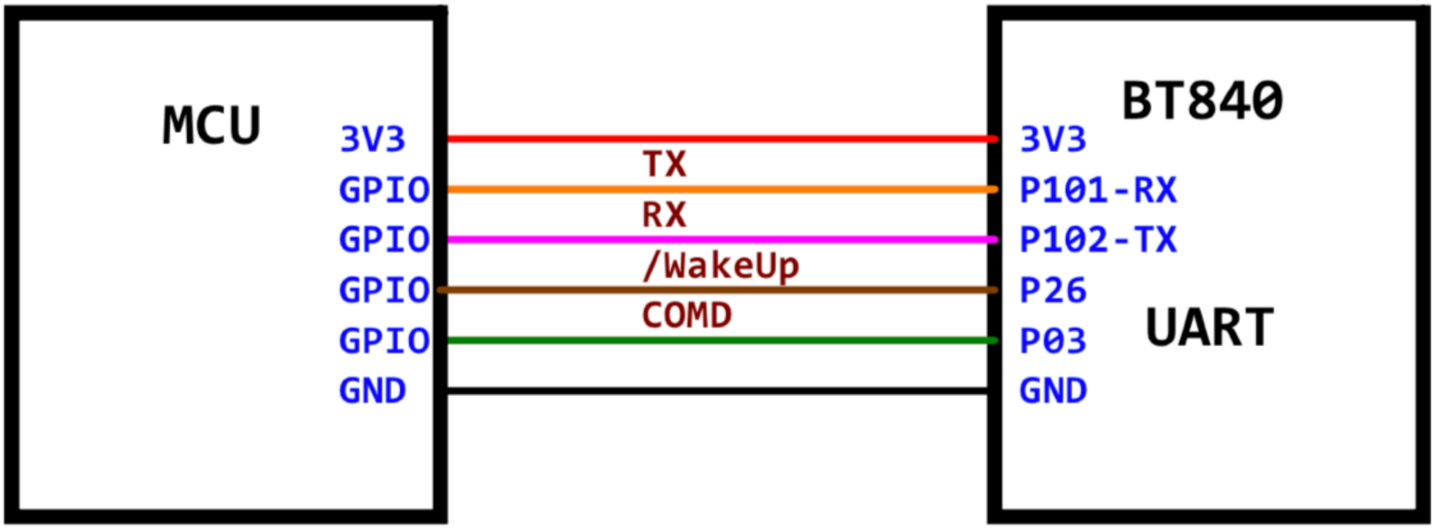
UART Interface for BT832F Series

UART interface connection between a host MCU and BT832F Series is shown below.



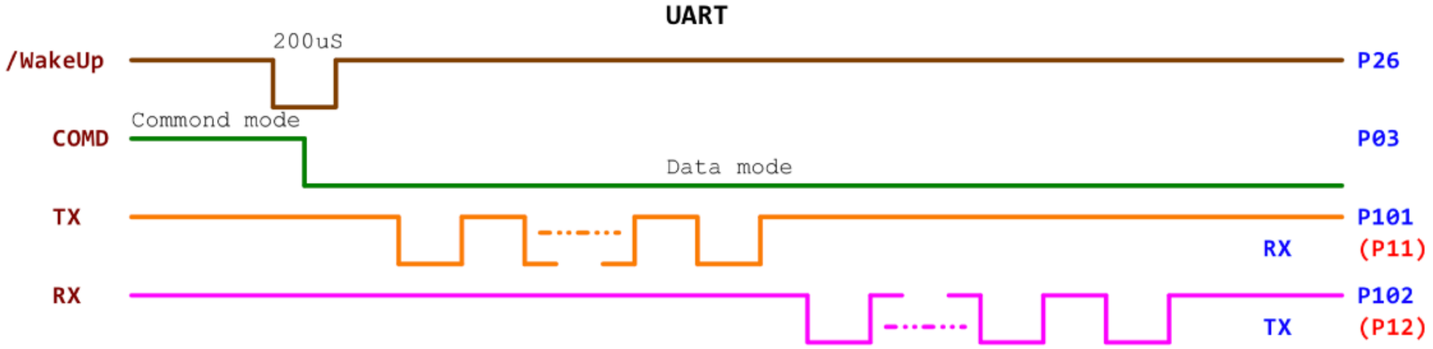
UART Interface for BT840F Series

UART interface connection between a host MCU and BT840F Series is shown below. nRF52840 port numbers are different from those of nRF52832/810. However, pin numbers on BT832F Series and BT840F Series modules are the same. Pin numbers on module evaluation board connectors are the same.



UART interface Timing Diagrams

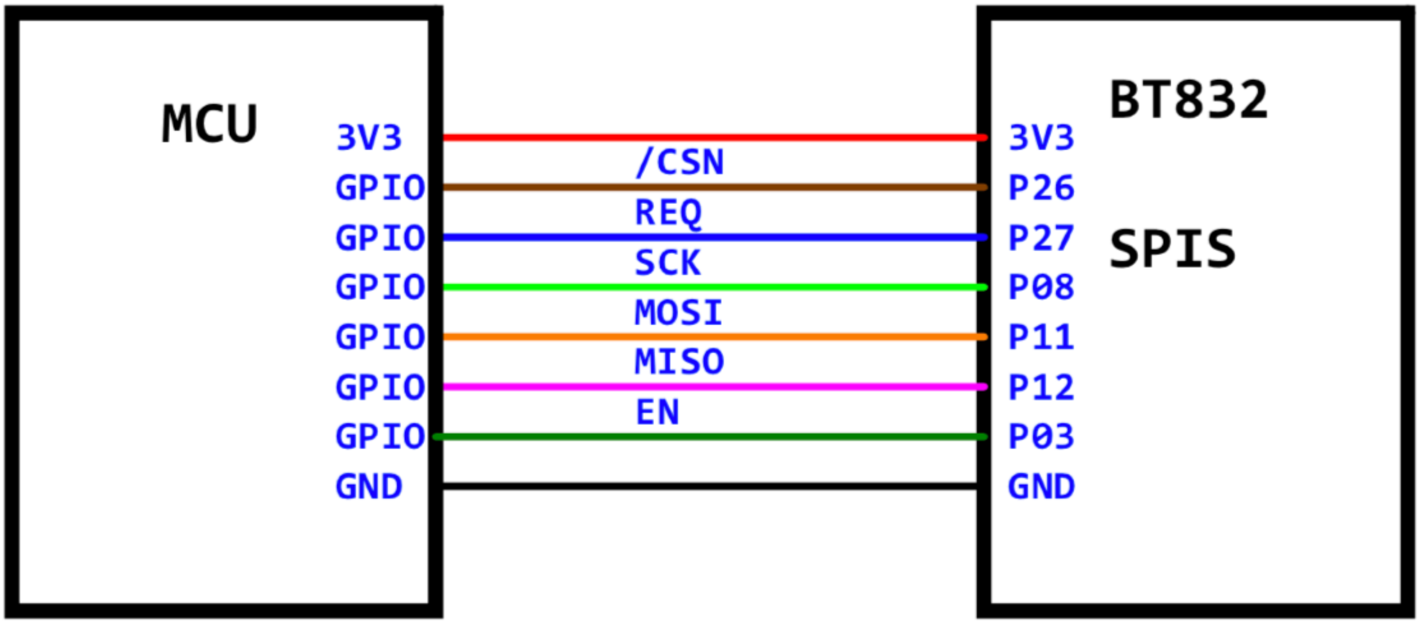
- Blue color port numbers are for both BT840F and BT832F mdules, for example, **P26**.
- If port number is different, BT840F port number is in blue, for example, **P101** and BT832F port number is in red, for example **P11**.
- Default baud rate for UART interface is 115200.
- CTS and RTS are not used.



- Module UART interface can be disable to save power by sending command “**AT+STOP**”. To wake up UART interface, pull the /WakeUp pin low for 200 uS or more.
- Set **COMD** pin high to communicate with MCU in module.
- Set **COMD** pin low to send data to a far end device (for example, a smartphone) transparently.

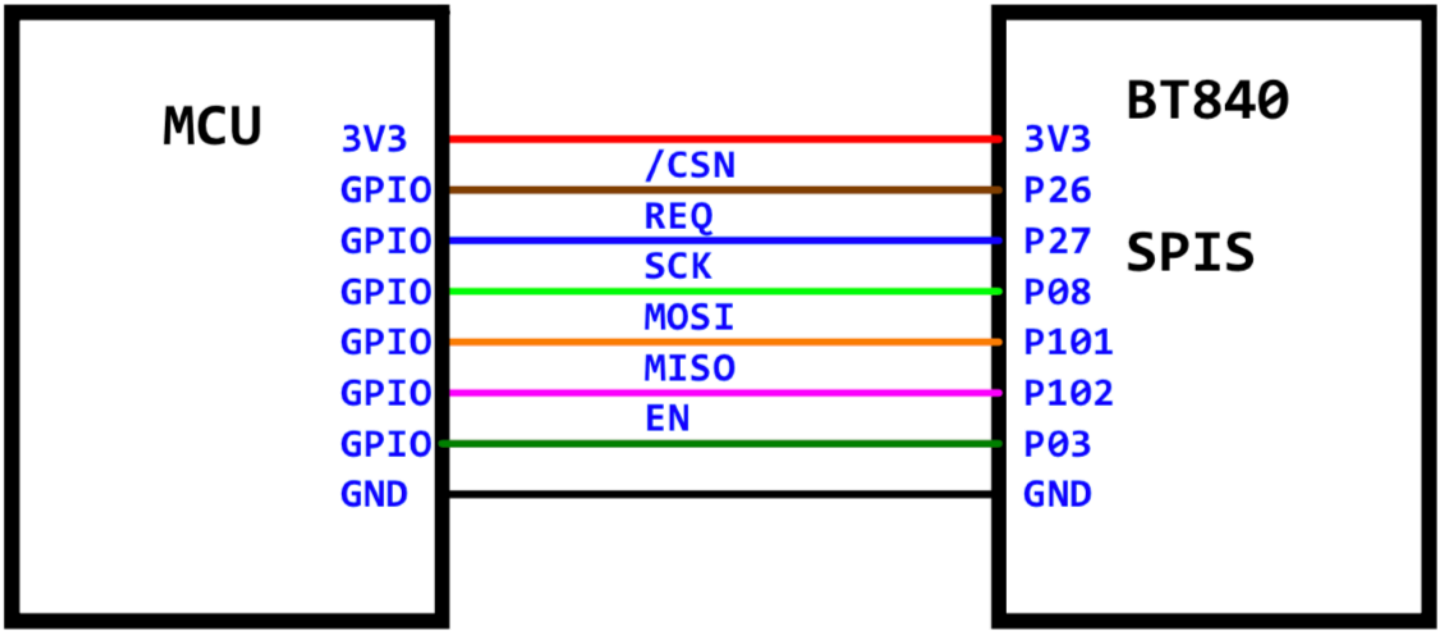
SPI Interface for BT832F Series

SPI interface connection between a host MCU and BT832F Series is shown below.



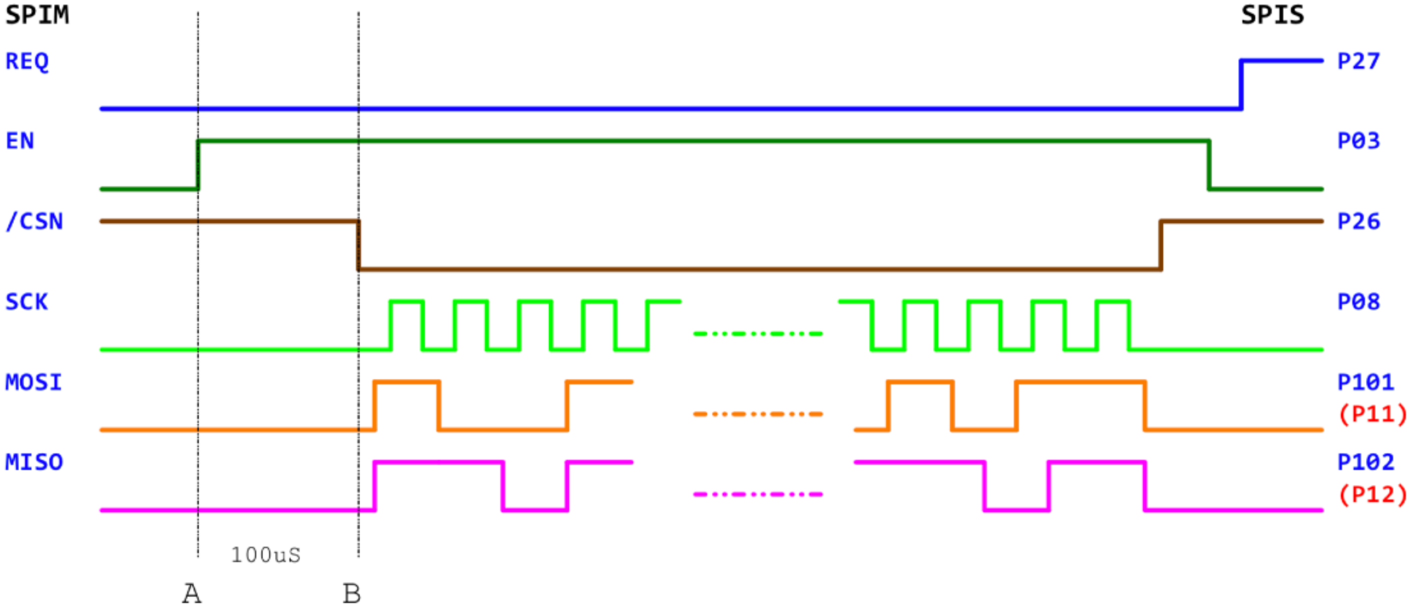
SPI Interface for BT840F Series

SPI interface connection between a host MCU and BT840F Series is shown below. nRF52840 port numbers are different from those of nRF52832/810. However, pin numbers on BT832F Series and BT840F Series modules are the same. Pin numbers on module evaluation board connectors are the same.



SPI Interface Timing Diagrams

- Bluetooth module is an SPI slave device and host MCU is the SPI master device.
- BT832F Series and BT840F Series can support up to 4 MHz clock rate.

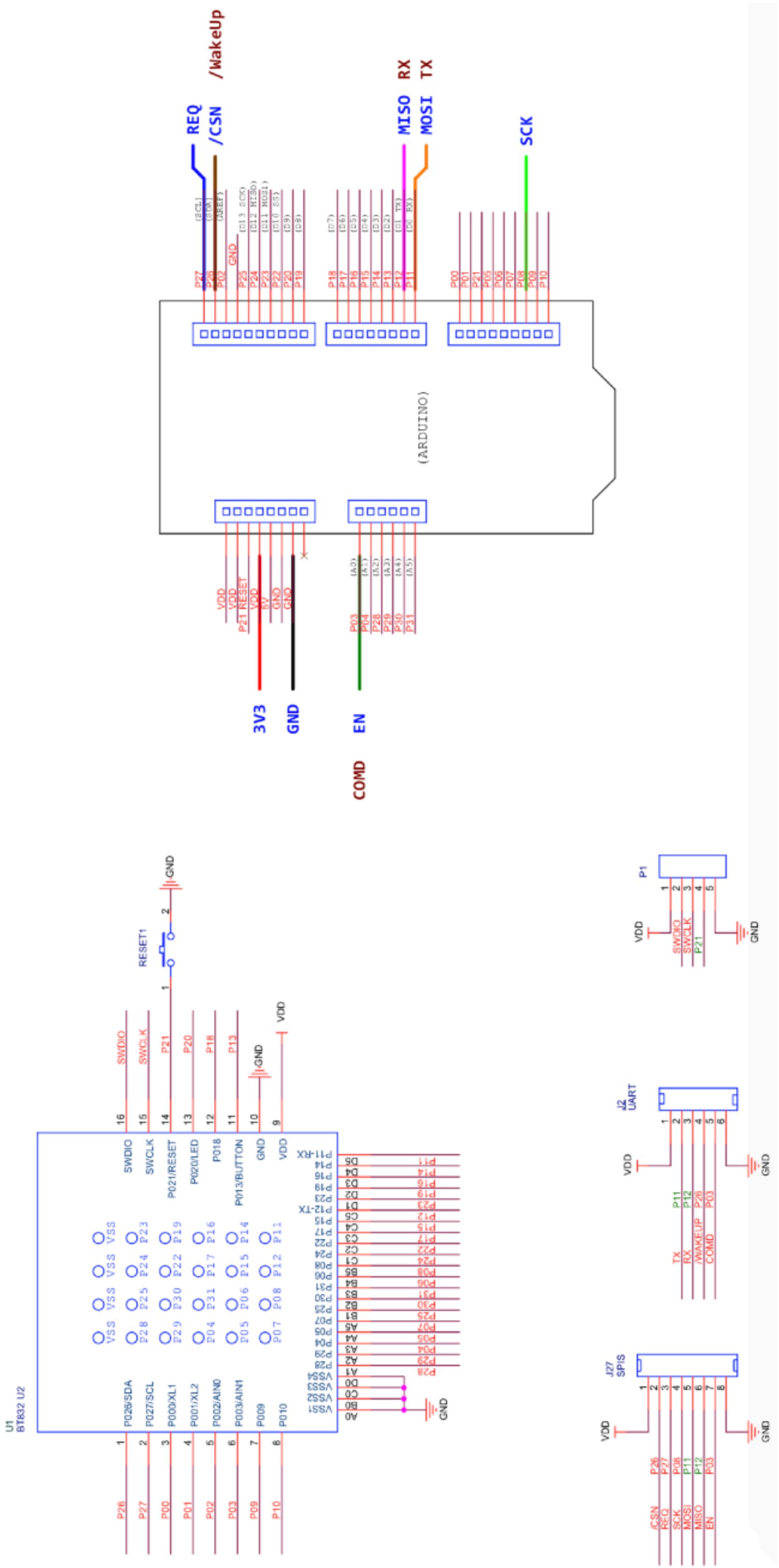


- To save power when not in use, host MCU sets **EN**able pin low to power down SPI interface.

- Before communicating with module, host sets **EN** pin high to wake-up.
- 100 uS or more delay between **EN** pin high and **/CSN** pin low. We use **/CSN** to indicate active low. It is the **CSN** pin in Nordic datasheets.
- Module set **REQ** pin high when ready to send data to host.
- **SCK**, **MOSI**, and **MISO** are SPI pins as defined in nRF52832 datasheets.
- Blue color port numbers are for both BT840F and BT832F modules, for example, **P26**.
- If port number is different, BT840F port number is in blue, for example, **P101** and BT832F port number is in red, for example **P11**.

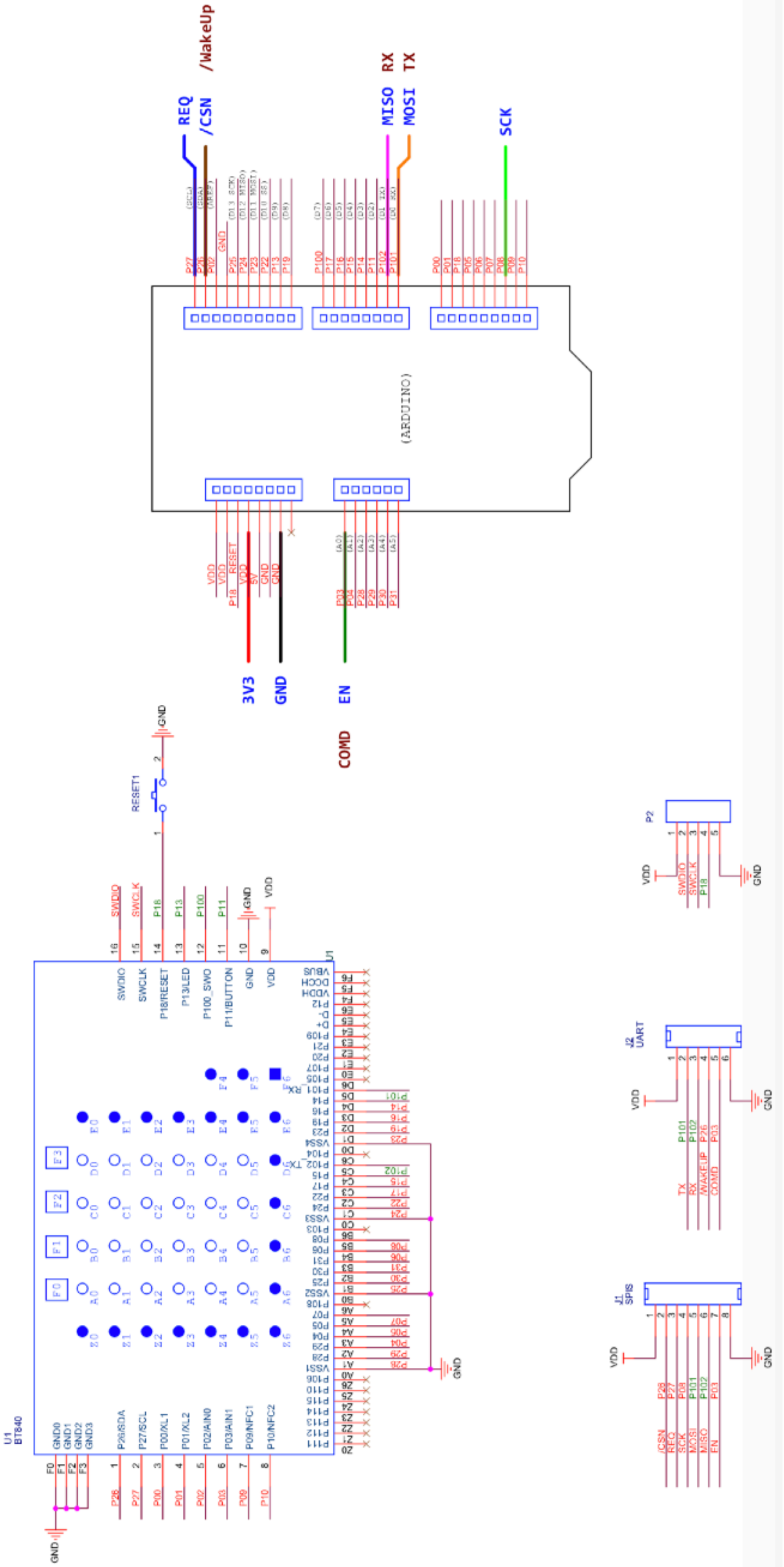
UART and SPI Pins on BT832F Evaluation Boards

UART and SPI pins can be accessed at Arduino type connector of EV-BT832 Series nRF52832 and nRF52810 modules. SPI pins in blue color (for example **MISO**) and UART pins in brown color (for example, **RX**).



UART and SPI Pins on BT840F Evaluation Boards

UART and SPI pins can be accessed at Arduino type connector of EV-BT840F Series nRF52840 modules. SPI pins in blue color (for example **MISO**) and UART pins in brown color (for example, **RX**).



3. AT Commands

.AT commands are text commands starting with the character sequence “AT” and terminated by a ‘\r’ character (ASCII code 0x0D). Commands are not case sensitive and have zero or more parameters.

- Each command line consists of a prefix, a body and a terminator.
- All command lines begin with the prefix AT (ASCII 065, 084) or at (ASCII 097, 116).
- Or, in hex format, all command lines begin with the prefix AT (HEX 0x41, 0x54) or at (HEX 0x61, 0x74).
- The body is a string of characters in the ASCII range 032-255. Control characters other than <CR> (carriage return; ASCII 013) and <BS> (back space; ASCII 008) in a command line are ignored.
- Or, in hex format, the body is a string of characters in the DEC range 032-255. Control characters other than <CR> (carriage return; HEX 0x0d) and <LF> (line feed; HEX 0x0a) in a command line are ignored.
- The terminator is <CR>.
- There is no distinction between upper-case and lower-case characters. A command line can have a maximum length of 80 characters. It is automatically discarded if the input is longer. Corrections are made
- The default baud rate is 115200, one stop bit and no parity

Command Mode

To set AT command mode,

- Pin 6 of BT680F (Px.xx of TC35680) is pulled high.
- Pin 6 of BT832F (P0.03 of nRF52832/810) is pulled high.
- Pin 6 of BT840F (P0.03 of nRF52840) is pulled high.

In AT command mode, the host processor communicates with the processor on module.

Data Mode

To set data mode,

- Pin 6 of BT680F (Px.xx of TC35680) is pulled low.
- Pin 6 of BT832 (P0.03 of nRF52832/810) is pulled low.
- Pin 6 of BT840F (P0.03 of nRF52840) is pulled low.

In data model, Bluetooth module provides transparent data transfer between the host processor and a remote device, for example, a smartphone.

AT Command List

When (Not for nrf52810) is listed below a command, this command is not supported for modules using nRF52810 (BT832A, BT832AF, BM832A).

Command	Response	Parameter	example
AT	OK or FAIL	none	AT/r/n
			OK/r/n
AT+RESET	OK or FAIL	none	AT+RESET/r/n
			OK/r/n
AT+VERSION?	+VERSION:<param>	Software version number	AT+VERSION?/r/n
			+VERSION140804 OK/
	OK		r/n
AT+RSSI?	+RSSI:-<param>	RSSI value	AT+RSSI?/r/n
	OK		+RSSI:-32 OK/r/n
	(Get connected rssi)		
AT+ADINTERVAL=<param>	OK or Fail	0040-9999 ms	AT+ADINTERVAL=0500/r/n
	(This command trigger re-advertise)		OK/r/n
AT+ADINTERVAL=?	+ADINTERVAL:<param>	0040-9999	AT+ADINTERVAL?/r/n
			+ADINTERVAL:0500/r/n
AT+ADSTART=<param>	OK or Fail	Length	AT+ADSTART=<0x04><0x0a><0x01><0x01><0xff>
		Timeout	OK/r/n
	Length always = 0x04	Type	
		Phy	
	Timeout 0x01-0xff	Channel	
	0xff = forever	Timeout 0xff = forever	
	Type=0x01 advertising		
	manufacturer, name, scan response		
	Type=0x02 advertising		
	manufacturer, name		
	Type=0x03 advertising		
	manufacturer, scan response		

	Type=0x04 advertising		
	Manufacturer		
	Type=0x05 advertising		
	name		
	Phy=0X01		
	BLE_GAP_PHY_1MBPS		
	Phy=0X02		
	BLE_GAP_PHY_2MBPS //TBD		
	Phy=0X03		
	BLE_GAP_PHY_CODED//TBD		
	Phy=0X04		
	BLE_GAP_PHY_AUTO		
	Channel = 0x37		
	Use only channel 37		
	Channel = 0x38		
	Use only channel 38		
	Channel = 39		
	Use only channel 39		
	Channel = all other values		
	Use channel 37,38,39		
	(This command trigger re-advertise)		
AT+ADSTART?	+ADSTART:<param>	Data	AT+ADSTART?/r/n
	Timeout 0x01-0xff		+ADSTART:FF1139/r/n
	0xff = forever		

	Type=0x31 advertising		
	manufacturer, name, scan response		
	Type=0x32 advertising		
	manufacturer, name		
	Type=0x33 advertising		
	manufacturer, scan response		
	Type=0x34 advertising		
	Manufacturer		
	Type=0x35 advertising		
	name		
	Phy=0X31		
	BLE_GAP_PHY_1MBPS		
	Phy=0X32		
	BLE_GAP_PHY_2MBPS //TBD		
	Phy=0X33		
	BLE_GAP_PHY_CODED//TBD		
	Phy=0X34		
	BLE_GAP_PHY_AUTO		
	Channel = 0x37		
	Use only channel 37		
	Channel = 0x38		
	Use only channel 38		
	Channel = 39		
	Use only channel 39		
	Channel = 0xff		
	Use all cannels		

AT+ADVMANU=<param>	OK or Fail (Manufacturer specific data)	Length,	AT+ADVMANU=<0x04><0x61><0x62><0x63><0x64>
		Data(Hex)	OK/r/n
AT+ADVMANU?	+ADVMANU:<param>	Data	AT+ADVMANU?/r/n
			+ADINTERVAL:abcd/r/n
AT+ADVRESP=<param>	OK or Fail (Scan response data)	Length,	AT+ADVRESP=<0x03><0x41><0x42><0x43>
		Data(Hex)	OK/r/n
AT+ADVRESP?	+ADVRESP:<param>	Data	AT+ADVRESP?/r/n
			+ADINTERVAL:ABCD/r/n
AT+SCANNER=<param>	OK or Fail	Enable	AT+SCANNER=11F/r/n
(Not for nrf52810)	Advertise data: first two bytes are RSSI.	Mode	OK/r/n
	1802010608ff59003031323334090941542035323834<CR><LF>	Channel	
	Scan response data		
	0bff34120102030405060708<CR><LF>		
	Enable = ‘0’		
	Disable scanner		
	Enable = ‘1’		
	Enable scanner		
	Enable = ‘2’		
	Enable scanner, advertise force off		
	Mode =’0’		
	Passive mode		
	Mode = ‘1’		
	Active mode		
	Channel = ‘7’		
	Use only channel 37		
	Channel = ‘8’		

	Use only channel 38		
	Channel = ‘9’		
	Use only channel 39		
	Channel = all other values		
	Use channel 37,38,39		
AT+SCANNER?	Mode =’0’	Data	AT+SCANNER?/r/n
(Not for nrf52810)	Passive mode		+SCANNER:138/r/n
	Mode = ‘1’		
	Active mode		
	Channel = ‘7’		
	Use only channel 37		
	Channel = ‘8’		
	Use only channel 38		
	Channel = ‘9’		
	Use only channel 39		
	Channel = all other values		
	Use channel 37,38,39		
AT+AUTOCON=<param>	OK or Fail	Length,	AT+AUTOCON=<0x05><0x00><0x01><0x02><0x03><0x01>
(Not for nrf52810)		ID(Hex)	OK/r/n
	Length always = 0x05	PHY	
	ID 4 bytes		
	If ID match the advertise manufacturer specific data first four bytes. Device will auto connect.		
	PHY = 0x01 1M ;PHY = 0x02 2M; PHY=0x03 coded PHY		
	(PHY for nrf52840 only)		
	After disconnect. If user want to re-connect need to send this command again.		

AT+AUTOCON?	Autocon on = ‘1’,Autocon off = ‘1’	Data	AT+AUTOCON?/r/n
(Not for nrf52810)	Connected = ‘1’, Disconnected = ‘0’		+AUTOCON: 100F1234561/r/n
	ID 4 bytes AscII.		
	PHY =’1’ 1M PHY=’2’ 2M PHY=’3’ Coded PHY		
AT+DISCON=<param>	OK or fail (Disconnect from a connection)	ASCII	AT+DISCON=1/r/n
		1	OK/r/n
AT+FILTERA=<param>	OK or fail (Filter advertise manufacturer ID for scanner)	ASCII four bytes ID	AT+FILTERA=0059/r/n
(Not for nrf52810)			OK/r/n
AT+FILTERA?	+FILTERA:<param>	Data	AT+FILTERA?/r/n
(Not for nrf52810)			+FILTERA:0059/r/n
AT+FILTERR=<param>	OK or fail (Filter response manufacturer ID for scanner)	ASCII four bytes ID	AT+FILTERA=5a7d/r/n
(Not for nrf52810)			OK/r/n
AT+FILTERR?	+FILTERR:<param>	Data	AT+FILTERR?/r/n
(Not for nrf52810)			+FILTERA:5a7d/r/n
AT+MANUAID=<param>	OK or fail (advertise manufacturer ID)	ASCII four bytes ID	AT+MANUAID=12B4/r/n
	(only updated with a new ADSTART command)		OK/r/n
AT+MANUAID?	+MANUAID:<param>	Data	AT+MANUAID?/r/n
			+MANUAID:0059/r/n
AT+MANURID=<param>	OK or fail (scan response manufacturer ID)	ASCII four bytes ID	AT+MANUAIR=A3E5/r/n
	(only updated with a new ADSTART command)		OK/r/n
AT+MANURID?	+MANURID:<param>	Data	AT+MANURID?/r/n
			+MANURID:1234/r/n
AT+DCDC=<param>	OK or Fail	0:disable	AT+DCDC=1/r/n
		1:enable	OK/r/n
AT+DCDC?	Enabled or Disabled	Data	AT+DCDC?/r/n
			Enabled/r/n

AT+SAVE=<param>	OK or Fail Following parameters will save to flash and read back after power reset:	Param:1	AT+SAVE=1/r/n
	Name, TX power, DCDC,UART baud rate,		OK/r/n
	Advertise and response manufacturer ID,		
	Advertise interval, timeout, type, phy		
AT+STOP=<param>	OK or Fail	1: adv stop,scan stop,UART alive	AT+STOP=1/r/n
		2:adv stop ,scan stop, UART stop	OK/r/n
	Power consumption:		
	Tx Power:0dBm, DCDC enabled ,No advertise, No scanner, with 32k crystal		
	1: UART alive:400-500 uA		
	2: UART stop:1.5-2 uA		
	Note:		
	(When UART stop, Need to pull GPIO26 to low to wake up the UART)		
AT+NAME?	+NAME:<param> OK	Device name	AT+NAME?/r/n
			+NAME:EZPro OK/r/n
AT+NAME=<param>	OK or FAIL	Device name	AT+NAME=Fanstel/r/n Or AT_Name="Fanstel"/r/n
			OK/r/n
AT+UART?	+UART:<param>,<param2>,<param3	Baud rate, Stop bit,	AT+UART?/r/n
	> OK	Parity	+UART:115200,1,0
			OK/r/n
AT+UART=<parm>	+UART:<parm>	Baud rate	AT+UART=115200/r/n
			+UART:115200,1,0
			OK/r/n
			1200
			2400
			4800
			9600

			19200
			38400
			57600
			115200
			230400
			460800
			921600
			1000000
AT+ADDR?	+ADDR:<param> OK	Device MAC	AT+ADDR?/r/n
		address	+ADDR:abb5:cd:604ace OK/r/n
AT+RX?	+Name:<parm>	none	AT+RX?/r/n
	+UART:<parm>		+NAME:EZPro/r/n
	+ADDR:<parm>		+UART:115200,1,1/r/n
			+ADDR: abb5:cd:604ace/r/n
AT+DEFAULT	OK or FAIL	none	AT+DEFAULT/r/n
			OK/r/n
<nrf52832>	+RFPW:parm	+4~-8	AT+RFPW?/r/n
AT_RFPW?			+RFPW:-4 OK/r/n 0:+4
			1:+0 default
			2:-4
			3:-8
<nrf52832>	OK or FAIL	0-3	AT+RFPW= 1/r/n OK/r/n
AT_RFPW=<parm>			0:+4
			1:+0
			2:-4
			3:-8
<nrf52840>	+RFPW:parm	+8~-8	AT+RFPW?/r/n
AT_RFPW?			+RFPW:-4 OK/r/n 0:+8
			1:+4 default
			2:+0

			3:-4
			4:-8
<nrf52840>	OK or FAIL	0-4	AT+RFPW= 1/r/n OK/r/n
AT_RFPW=<parm>			0:+8 1:+4 2:+0 3:-4
			4:-8
AT	OK or FAIL	P00-P31	AT+PIO=005,0\r\n
+PIO=<param><param1>	Set GPIO as high or low	P100-115	
	Note that the pin must be configured as an output for this function to have any effect.	1=High , 0=low	OK/r/n
	GPIO:0,1,3,8,9,10,11,12,26,27,101,102 reverse		
AT	OK or FAIL	P00-P31	AT+PIR017?
+PIR<param>?	Function for reading the input level of a GPIO pin. Note that the pin must have input connected for the value returned from this function to be valid.	P100-115	
		1=High , 0=low	V=1/r/n
Param=000-031,100-115			
AT+PIS=<param><param1>	OK or FAIL	P00-P31	AT+PIS=005,1\r\n
	Set GPIO as input, output or back to default.	P100-115	
	GPIO: 0,1,3,8,9,10,11,12,26,27,101,102 reverse	0=input	OK/r/n
		1=output	
		2=back default	
	+CONNECTED		+CONNECTED/r/n
	(Device connected)		
	+DISCONNECTED		+DISCONNECTED/r/n
	(Device connected)		
	+SCAN TIMEOUT		+SCAN TIMEOUT/r/n
	(Scan timeout)		

SPI Master Code Example

The AT command also work in SPIS mode.

Source codes for SPI Master MCU can be downloaded from:
http://www.fanstel.com/download-document/

Modification can be required your master MCU. The following are example codes for SPI Master.

Please download the firmware for SPIS mode if you need.

```
    .frequency    = NRF_DRV_SPI_FREQ_4M,
    .mode         = NRF_DRV_SPI_MODE_0,
    .bit_order    = NRF_DRV_SPI_BIT_ORDER_MSB_FIRST,
/< SCK active high, sample on leading edge of clock.
```

Connection:
SPIS CS : GPIO0.26 chip enable
SPIS MISO: GPIO 1.02 (GPIO 12 for 52832/52810) master input slave output
SPIS MOSI: GPIO 1.01 (GPIO 11 for 52832/52810) master output slave input
SPIS SCK: GPIO 0.08 clock
SPIS REQ: GPIO 0.27 if there is data pending. SPIS REQ will into the high state.
SPIS EN: GPIO 0.03 high enable SPIS, low disable SPIS

commands to the radio will begin with <length>C....

data to the radio will begin with <length>D....

radio responses to commands will begin with <length>C...

radio responses to data will begin with <length>D...

when the radio is sending data from some BLE device after the IRQ the data will begin with <length>B....

when the radio is sending scan data after the IRQ the data will begin with <length>S....

when the radio is sending link status after the IRQ the data will begin with <length>L

....

The commands for SPIS are a little different from UART mode.

The data format is total length + type + data

At UART mode: AT<0x0d><0x0a>

At SPIS mode : <0x05><0x43><0x41><0x54><0x0d><0x0a>

The response at SPIS mode is for pervious command.
When the SPIS REQ in high state.

Please send <0x07>CAT+?<0x0d><0x0a> to get the data.

or <0x07>DAT+?<0x0d><0x0a> to get the data.

Example for command type ‘C’.

AT

Host send:

```
05 43 41 54 0D 0A     |.CAT..
```

Slave response at next command

```
05 43 4F 4B 0D 0A       |.COK..
```

AT+VERSION?

Host send:

```
0E 43 41 54 2B 56 45 52|.CAT+VER
53 49 4F 4E 3F 0D 0A   |SION?..
```

Slave response at next command

```
15 43 2B 56 45 52 53 49|.C+VERSI
4F 4E 3A 31 38 30 37 30|ON:18070
33 20 4F 4B 0D 0A       |3 OK..
```

AT+NAME?

Host send:

```
0B 43 41 54 2B 4E 41 4D|.CAT+NAM
45 3F 0D 0A             |E?..
```

Slave response at next command

```
13 43 2B 4E 41 4D 45 3A|.C+NAME:
46 61 6E 73 74 65 6C 20|Fanstel
4F 4B 0D 0A             |OK..
```

Example for command type 'D'.

Host send string Test but no connection:

```
07 44 54 65 73 74 0D 0A|.DTest..
```

Slave response at next command

```
0F 44 4E 6F 63 6F 6E 6E|.DNoconn
65 63 74 69 6F 6E 0D 0A|ection..
```

Host send:

```
07 44 41 54 2B 3F 0D 0A|.DAT+?..
```

Slave response at next command

```
08 44 4E 4F 49 52 51 0D|.DNOIRQ.
0A                      |.
```

Host send string Test with BLE connection:

```
07 44 54 65 73 74 0D 0A|.DTest..
```

Slave response at next command

```
07 44 53 45 4E 44 0D 0A|.DSEND..Host send:
```

Host send:

```
07 44 41 54 2B 3F 0D 0A|.DAT+?..
```

Slave response at next command

```
08 44 4E 4F 49 52 51 0D|.DNOIRQ.
0A                     |.
```

Example for command type ‘L’.

IRQ high, Remote BLE connected. Slave response at next command

```
0D 4C 2B 43 4F 4E 4E 45|.L+CONNE
43 54 45 44 0D 0A      |CTED..
```

IRQ high, Remote BLE disconnected. Slave response at next command

```
10 4C 2B 44 49 53 43 4F|.L+DISCO
4E 4E 45 43 54 45 44 0D|NNECTED.
0A                     |.
```

Example for command type ‘B’.

IRQ high, Remote BLE send text “123456”.

```
09 42 31 32 33 34 35 36|.B123456
0D 0A                  |..
```

IRQ high, Remote BLE send text “abcd”.

```
07 42 61 62 63 64 0D 0A|.Babcd..
```

Example for command type ‘S’.

IRQ high, BLE scanning on.

```
19 53 30 62 66 66 33 34|.S0bff34
31 32 30 31 30 32 30 33|12010203
30 34 30 35 30 36 30 37|04050607
30 38                  |08

1D 53 32 63 30 32 30 31|.S2c0201
30 36 30 38 66 66 35 39|0608ff59
30 30 31 39 31 61 31 62|00191a1b
31 63 31 64 30 39      |1c1d09.
```

IRQ high, Remote BLE send text “abcd”.

```
07 42 61 62 63 64 0D 0A|.Babcd..
```

Timing

Following are the example codes to control the SPIS.

```
int main(void)
{

    nrf_drv_spi_config_t spi_config = NRF_DRV_SPI_DEFAULT_CONFIG;
    spi_config.ss_pin   = 26;
    spi_config.miso_pin = NRF_GPIO_PIN_MAP(1, 2);
    spi_config.mosi_pin = NRF_GPIO_PIN_MAP(1, 1);
    spi_config.sck_pin  = 8;
    APP_ERROR_CHECK(nrf_drv_spi_init(&spi, &spi_config, spi_event_handler, NULL));

    NRF_LOG_INFO("SPI example started.");

        nrf_gpio_cfg_input(SPI_REQ,BUTTON_PULL);
        SPI_Send("AT\r\n",4);
        SPI_Send("AT+SCANNER=21F\r\n",16);

    while (1)
    {
                        if(nrf_gpio_pin_read(SPI_REQ) == 0){

                        }
                        else{
                                SPI_Send("AT+?\r\n",6);



                        }


                }
/**
 * @brief SPI user event handler.
 * @param event
 */
void spi_event_handler(nrf_drv_spi_evt_t const * p_event,
                       void *                    p_context)
{
    spi_xfer_done = true;
    NRF_LOG_INFO("Transfer completed.");
```

```
        NRF_LOG_HEXDUMP_INFO(m_tx_buf, strlen((const char *)m_tx_buf));
   if (m_rx_buf[0] != 0)
   {
      NRF_LOG_INFO(" Received:%d",strlen((const char *)m_rx_buf));
   }
}

void SPI_Send(uint8_t *send_buf,uint8_t length){
                m_length = length+1;
   memset(m_rx_buf, 0, m_length);
   spi_xfer_done = false;
                for(int i=0;i < sizeof(m_tx_buf) ;i++)
                        m_tx_buf[i] = 0;
                m_tx_buf[0] = m_length;
                for(int i=1,j = 0;j < length ;i++,j++)
                        m_tx_buf[i] = send_buf[j];
    APP_ERROR_CHECK(nrf_drv_spi_transfer(&spi, m_tx_buf, m_length, m_rx_buf,
30));

      while (!spi_xfer_done)
      {
         __WFE();
      }
      NRF_LOG_FLUSH();

      bsp_board_led_invert(BSP_BOARD_LED_0);
      nrf_delay_ms(200);
}
```

4. Evaluation Board

Communicating with a PC

A quick and easy way to evaluate BT832 is to use a PC as the host processor. Connect the evaluation board to a PC with an USB cable. Then,

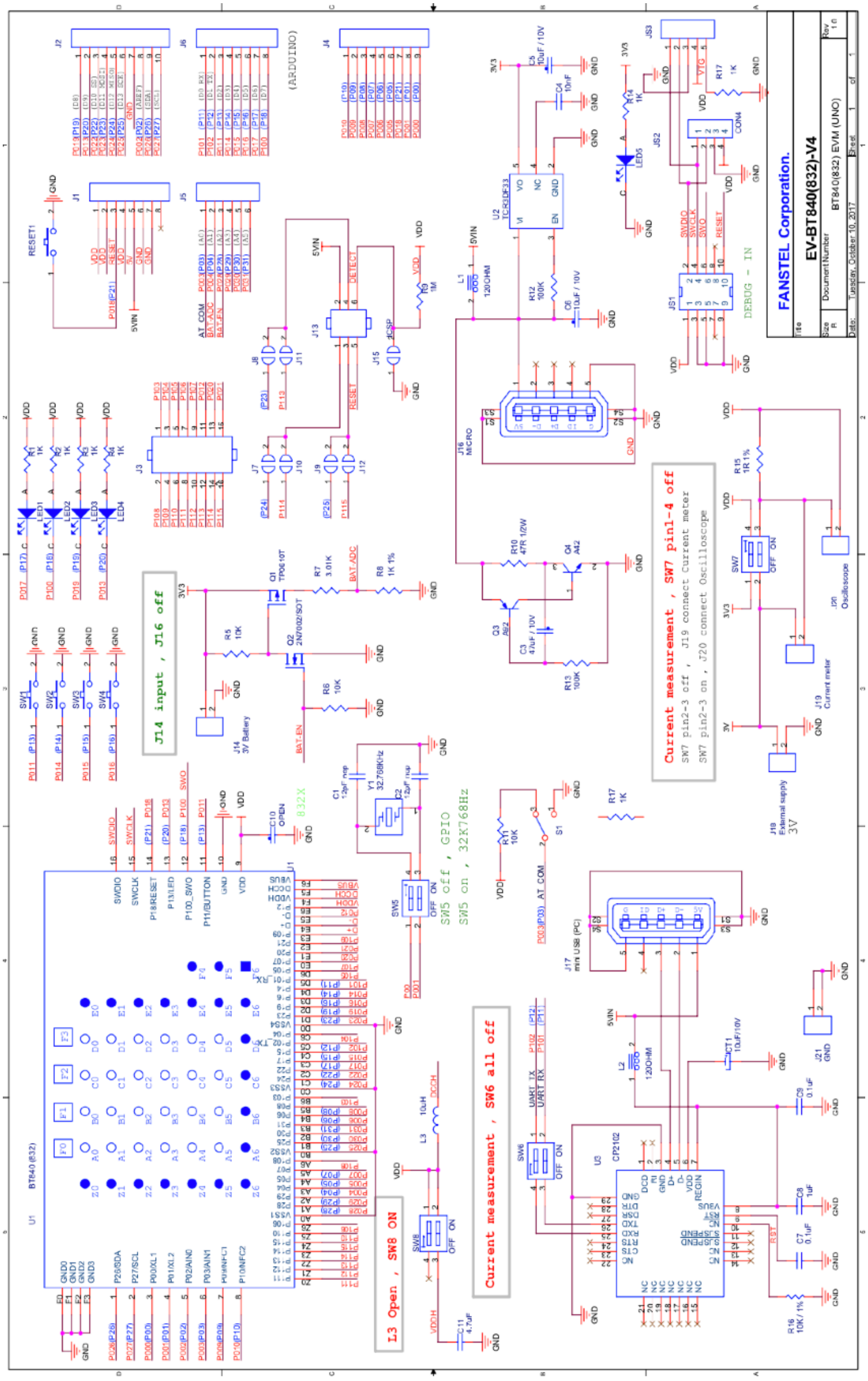
- Set S1, Bluetooth module is set to command mode. PC will communicate with module.
- Set switch S1 to the other position, Bluetooth module is set to data mode. PC will communicate with a remote device through Bluetooth wireless connection.

Docklight is a testing, analysis and simulation tool for serial communication protocols (RS232, RS485/422 and others). It allows you to monitor the communication between two serial devices or to test the serial communication of a single device. Docklight significantly increases productivity in a broad range of industries, including automation and control, communications, automotive, equipment manufacturers, and embedded / consumer products. Docklight is easy to use and runs on almost any standard PC using Windows 10, Windows 8, Windows 7, Windows Vista or Windows XP operating system.

Docklight software can be downloaded from the following:

http://www.docklight.de/download_en.htm

BT832F and BT840F Evaluation Board Schematics



Revision History

- July 2018, Ver. 0.1: Draft copy

Contact Us

United States:

Fanstel Corp.
7466 E. Monte Cristo Ave. Scottsdale AZ 85260
Tel. 1 480-948-4928
Fax. 1-480-948-5459
Email: info@fanstel.com
Website: www.fanstel.com

Taiwan:

Fanstel Corp.
10F-10, 79 Xintai Wu Road
Xizhu, New Taipei City, Taiwan 22101
泛世公司
臺灣省新北市汐止區新臺五路79號10樓之10, 22101
Tel. 886-2-2698-9328
Fax. 886-2-2698-4813
Email: info@fanstel.com
Website: www.fanstel.com

China:

Fanstel Technologies Corp.
11 Jiale Street
Ping-Dih, Long-Gang, Shen Zhen, GD 518117
泛世康科技(深圳)有限公司
廣東省深圳市龍崗區坪地鎮佳樂街11號
Tel. 86-755-8409-0928
Fax. 86-755-8409-0973
QQ. 3076221086
Email: info@fanstel.com
Website: www.fanstel.com