ATOLL SOLUTIONS

ATOLL AT1020-256P MODULE USER GUIDE

#229, Second Floor, 2A Main, 5th Cross, New Thippasandra, HAL 3rd Stage, Bangalore, India -560 075. Web: www.atollsolutions.comEmail: info@atollsolutions.com

CONTENTS

1 INTRODUCTION

AT1020 is a low power Bluetooth Low Energy Radio v4.2 specification compliant (Upgradable to 5.0) module and based on Qualcomm CSR1020 Bluetooth Low Energy SoC.

Atoll AT1020-256P is a special module for customers looking to build products with UART serial pass-through features. This module have inbuilt UART pass-through firmware (pipe-line) that help the customers to transfer data from HOST controller over BLE. This module allows customers to use Qualcomm 1020 solution without any firmware development on the BLE module.

2 FEATURES

Following are key features of AT1020-256P module:

- AT1020-256P supports two different baud rates for UART Pass-through (High Baud rate: 115200 & Low Baud rate: 9600).
- Auto Sleep mode feature.
- Configuration of TX power level from -60 dBm to +4dBm.
- Secured BLE using bonding feature.

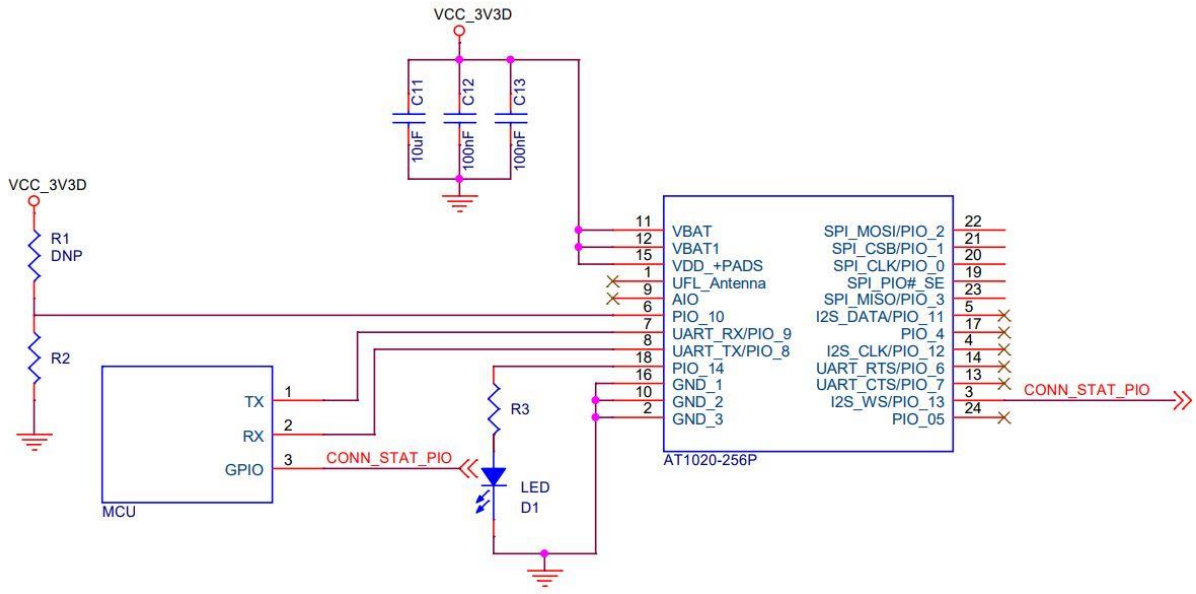
3 OVERVIEW

This module is part Atoll's AT1020 family of low power BLE modules. Following give overview of the module:

- UART pass-through code for the BLE data transfer.
- Based on Qualcomm CSR1020 Bluetooth Low Energy SoC.
- Low power Bluetooth Low Energy Radio v4.2 specification compliant (Upgradable to 5.0).
- -90.5 dBm Rx sensitivity.
- +4 dBm RF transmit power. No external power amplifier or Tx/Rx switch required.
- GAP, L2CAP, Security Manager, Generic Attribute Protocol, Attribute Profile, Bluetooth Low Energy technology profile support.
- Antenna: PCB Antenna.
- Input voltage 3.6 V to 1.8 V.
- Dimension: 18.91mm x 15.25mm x 2.7mm

4 HARDWARE CONFIGURATION

4.1 REFERENCE SCHEMATICS



4.2 CONNECTION STATUS LED

AT1020-256P module can drive a led on PIO14 (pin number18) of the module to indicate the BLE connection status.

Fast Blinking: Status LED toggle in every 1 sec interval. This indicate the device is advertising for the connection.

Slow Blinking: Status LED toggle in every 5sec interval. This indicate the device is in connected state.

LED OFF State: The device is in sleep mode/stop advertisement.

4.3 CONNECTION STATUS GPIO

CONN_STAT_GPIO is used to know the BLE is connected or not. If BLE is connected, it drive active high signal. Using this pin external MCU can check current BLE connection status. PIO13 (pin number 3) is used for CONN-STAT-GPIO

PIN STATE	BLE STATUS	UART MODE
LOW	NOT CONNECTED	COMMAND MODE
HIGH	CONNECTED	DATA MODE

4.4 UART PIN CONFIGURATION

The UART pin configuration for the BLE is given by

UART PIN	PIO	PIN NO.
UART-TX	PIO8	7
UART-RX	PIO9	8

4.5 UART BAUD RATE CONFIGURATION.

AT1020-256P module have one GPIO for UART baud rate select PIO10 (pi number 6). The device configure the baud rate depends on the digital level of this PIO10. If the PIO10 is Low then baud rate is 9600 and if the PIO10 is high baud rate is 115200. By default the PIO10 is internally pull down. Default baud rate is 9600.

PIO10	Baudrate
0	9600
1	115200

5 DEFAULT CONFIGURATION

The AT1020-256P having the following default. Some of these configuration can be modified using set of AT commands.

Sl.No.	Configuration Parameter	Default Value
1	Device Name	AT1020-P
2	UART Baud rate	9600
3	UART Parity Mode	None
4	UART Word Length	8 bit
5	UART-Flow Control	Disabled
6	Sleep Mode	Disabled
7	TX Power	+4dBm
8	Advert Interval	60 ms
11	Advert Timeout(sleep mode disabled)	0 (Always)
12	Advert Timeout(sleep mode enabled)	60 ms
13	Connection Interval	30 ms

6 CONFIGURATION COMMANDS

We provide a set of AT commands for modifying the configurations of pass-through firmware. These command support when the device is in the disconnected state only. Before using these command please make sure that the device is disconnected from client.

NOTE: These configurations are one time process. Once you configured it will store the configurations in the NVM of BLE module except AT+SLEEPREQ and AT+UNBOND.

6.1 AT

This command is used to test the UART interface in command mode.

Syntax: AT<CR>

It will return "OK" response for successful command.

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT\r");
Write_to_ble(Request);
```

6.2 AT+RESET

This command is used to reset/reboot (soft reset) the device. When we configure any parameter you need to send this command to save the changes.

Syntax: AT+RESET<CR>

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+RESET\r");
Write_to_ble(Request);
```

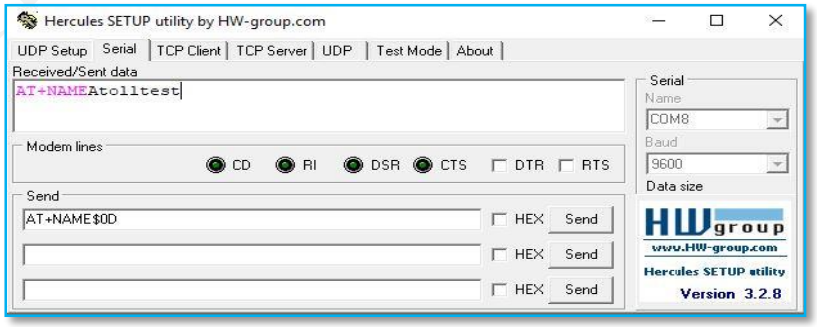
6.3 AT+NAME

This command is used to read or change the device name of the BLE device. The device name should be less than 20 bytes. Default device name is "AT1020-P"

Get Device Name

Syntax: AT+NAME<CR>

It will return the device name of BLE device.



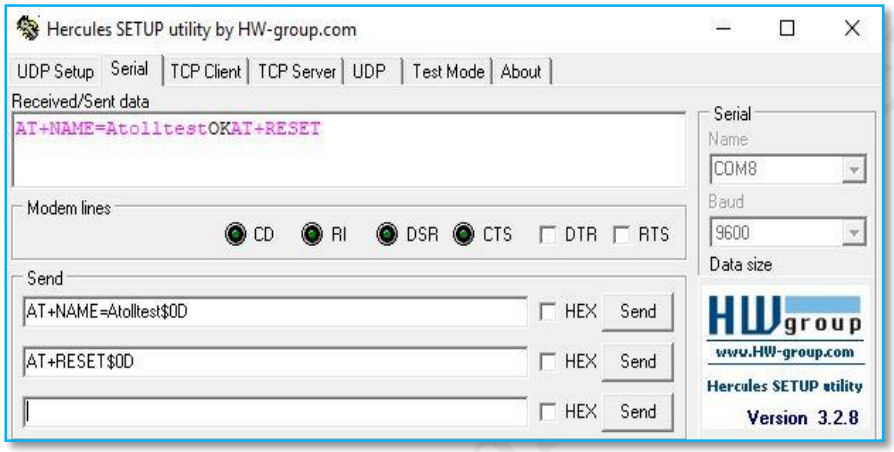
Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+NAME\r");
Write_to_ble(Request);
```

Set Device Name

Syntax: AT+NAME=<Device name><CR>

It will return "OK" response for successful command and "ERROR" for invalid command. After getting the OK response, need to send the reset (AT+RESET) command to feel the changes.



Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+NAME=%s\r",device_name);
Write_to_ble(Request);
```

6.4 AT+SLEEPMODE

This command is used to enable or disable auto sleep mode. In active mode the device always advertising at time advert interval of 60 ms.

In sleep mode, device advertising for 60 sec and it check the BLE inactivity (in not connected state) and goes to the sleep mode. UART Rx pin is used to wake up the device from the sleep mode.

Get Sleep Mode

Syntax: AT+SLEEPMODE<CR>

It will return the sleep mode, 0 -> Disabled and 1-> Enabled

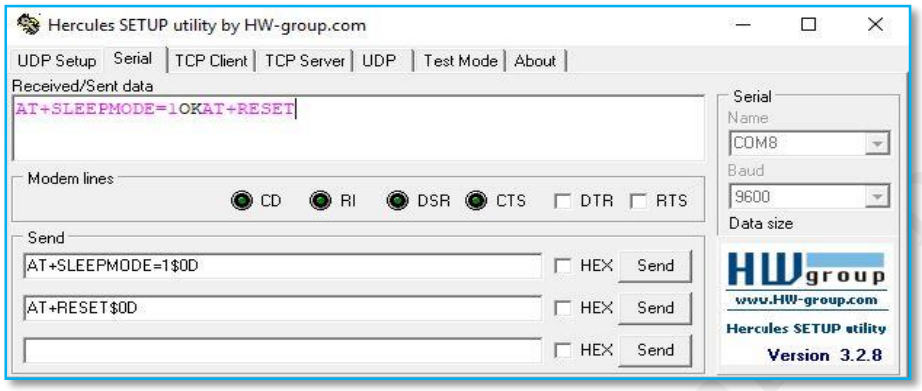
Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+SLEEPMODE\r");
Write_to_ble(Request);
```

Set Sleep Mode

Syntax: AT+SLEEPMODE=<0 or 1><CR>

It will return “OK” response for successful command and “ERROR” for invalid command. After getting the OK response, need to send the reset (AT+RESET) command to feel the changes.



Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+SLEEPMODE=%d\r",sleep_mode);
Write_to_ble(Request);
```

6.5 AT+SLEEPREQ

This command is used to issue a sleep request (dormant) from the external MCU. UART Rx pin is used to wake up the device from the sleep mode.

Syntax: AT+SLEEPREQ<CR>

It will return “OK” response for successful command and “ERROR” for invalid command.

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+SLEEPREQ\r");
Write_to_ble(Request);
```

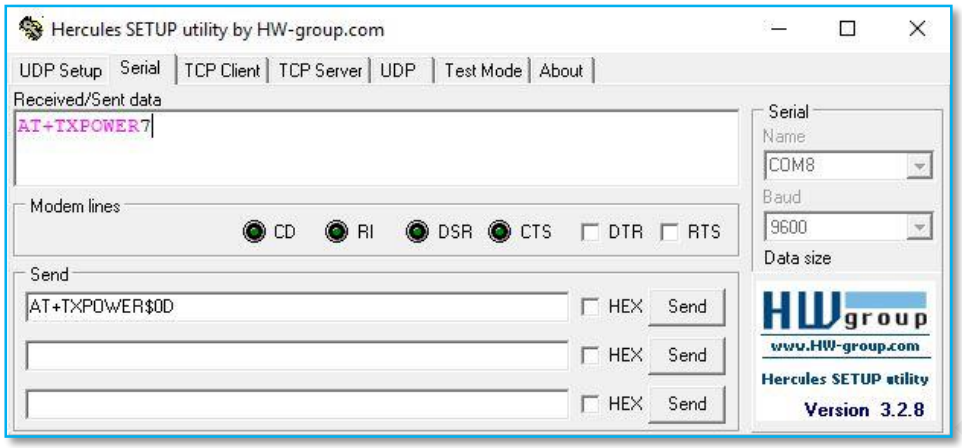
6.6 AT+TXPOWER

This command is used to change the transmission power of the device. AT1020-256P support the TX power from -11dBm t0 +4dBm represented in 0 to 15.

Get the TX power level

Syntax: AT+TXPOWER<CR>

It will return the current transmission power (0-15).



Example MCU code snippet:

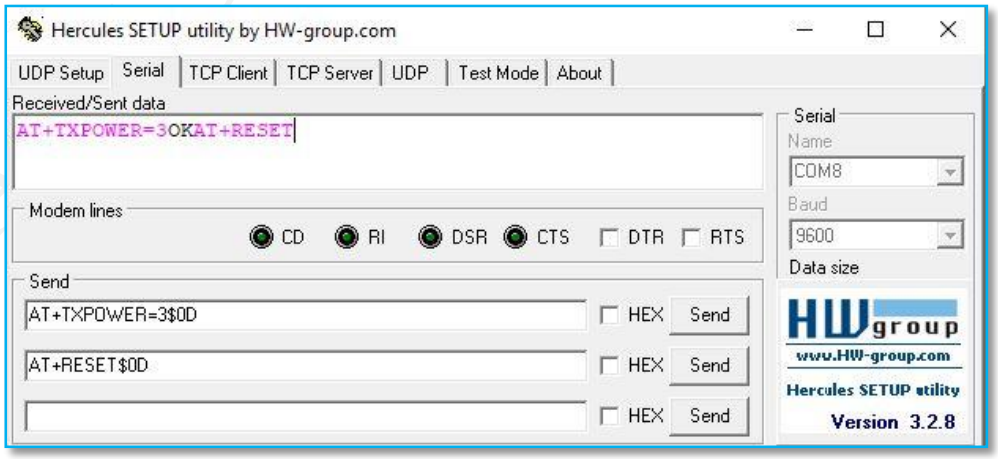
```
sprintf((char*)Request,(const char*)"AT+TXPOWER\r");
Write_to_ble(Request);
```

Set the TX power level

Syntax: AT+TXPOWER =<tx_power_level><CR>

TX Power	tx_power_level		TX Power	tx_power_level
-60 dBm	0		-3 dBm	8
-30 dBm	1		-2 dBm	9
-22 dBm	2		-1 dBm	10
-16 dBm	3		0 dBm	11
-13 dBm	4		+1 dBm	12
-10 dBm	5		+2 dBm	13
-6 dBm	6		+3 dBm	14
-4 dBm	7		+4 dBm	15

It will return “OK” response for successful command and “ERROR” for invalid command. After getting the OK response, need to send the reset (AT+RESET) command to save the changes.



Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+TXPOWER=%d\r",TxPower);
Write_to_ble(Request);
```

6.7 AT+BONDSTAT

This command is used to check the bond status of the device.

Syntax: AT+BONDSTAT<CR>

It will return the “BONDED” if device is bonded and return “NOT_BONDED” if device not bonded.

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+BONDSTAT\r");
Write_to_ble(Request);
```

6.8 AT+UNBOND

This command is used to remove the bond information from the BLE module.

Syntax: AT+UNBOND<CR>

It will return “OK” response for successful command and “ERROR” for invalid command.

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+UNBOND\r");
Write_to_ble(Request);
```

6.9 AT+DEFAULT

This command is used to set the default state

Syntax: AT+DEFAULT<CR>

It will return “OK” response for successful command and “ERROR” for invalid command.

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+DEFAULT\r");
Write_to_ble(Request);
```

6.10 AT+VERSION

This command is used to check the pass-through firmware revision.

Syntax: AT+VERSION<CR>

It will return the firmware version. Ex. “AT1020-256P V0.3”

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"AT+VERSION\r");
Write_to_ble(Request);
```

6.11 +++

This command is used to disconnect the device in connected state. This command should send to BLE on connected state. In BLE connected state, AT commands won't work.

Syntax: +++

Example MCU code snippet:

```
sprintf((char*)Request,(const char*)"+++");
Write_to_ble(Request);
```

7 SERVICE CHARACTERISTICS

We are using four BLE services in the pass-through firmware.

1. GAP Profile Service.
2. GATT Profile Service.
3. Battery Profile Service.
4. Pass-through Service (Custom Service).

7.1 Pass-through service characteristics

We implemented a custom BLE service for the pass-through data transfer. It has two property 1. write_cmd and 2. Notify. The “write_cmd” property is used to send the data from client to server and “Notify” is used to send the data from server to client. The service characteristics is given below

```
/*! \brief Serial Service UUID */
#define UUID_SERIAL_SERVICE 0x00005500D10211E19B2300025B00A5A5
/*! \brief Serial Data Transfer UUID */
#define UUID_SERIAL_DATA_TRANSFER 0x00005501D10211E19B2300025B00A5A5

/* Primary service declaration of Serial service */
primary_service {
    uuid:   UUID_SERIAL_SERVICE,
    name:   "SERIAL_SERVICE",

    /* Serial Data Transfer characteristic */
    characteristic  {
        uuid:   UUID_SERIAL_DATA_TRANSFER,
        name:   "SERIAL_DATA_TRANSFER",

        /* Dynamic value maintained by the application */
        flags:  [FLAG_IRQ],

      properties: [write_cmd, notify],
      size_value: 120,

         client_config {
            flags : [FLAG_IRQ],
            name : "SERIAL_DATA_C_CFG"
         }

    }
}
```

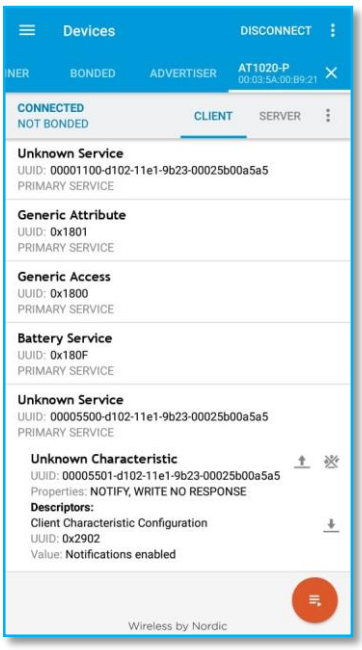
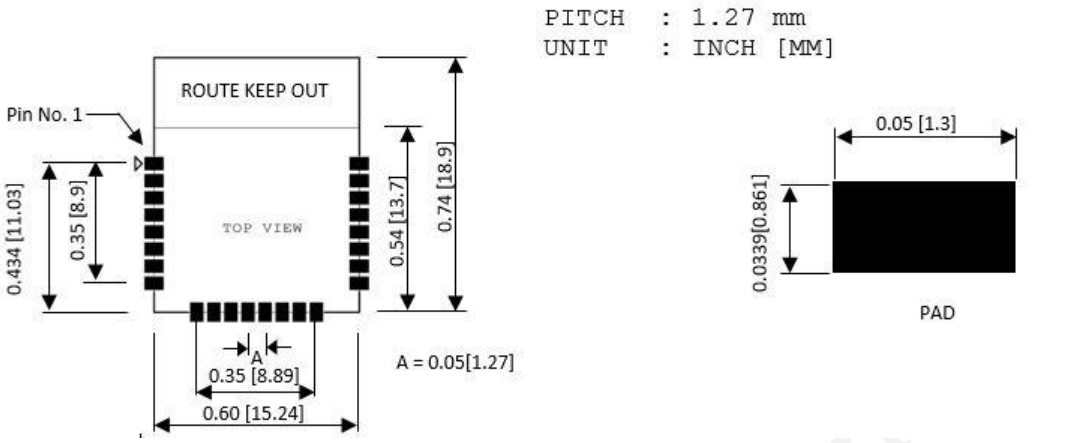


Fig: NRF Connect App screen, listing all available BLE services after making the connection

8 RECOMMENDED FOOTPRINT INFORMATION



Pin Configuration

Pin	Pin Name
1	ANTENNA PIN
2	GND_3
3	I2S_WS/PIO_13
4	SCL/PIO_12
5	SDA/PIO_11
6	PIO_10
7	UART_RX/PIO_9
8	UART_TX/PIO_8

Pin	Pin Name
9	AIO
10	GND_2
11	VBAT
12	VBAT1
13	RESERVED
14	RESERVED
15	VDD_+PADS
16	GND_1

Pin	Pin Name
17	RESERVED
18	PIO_14
19	SPI_PIO#_SE
20	DEBUG_SPI_CLK/PIO_0
21	DEBUG_SPI_CSB/PIO_1
22	DEBUG_SPI_MOSI/PIO_2
23	DEBUG_SPI_MISO/PIO_3
24	RESERVED

9 STANDARD TERMS AND CONDITIONS

Customer Support:

Atoll Solutions offers customers an easy "out of box" experience by providing, user manuals and other electro mechanical documentation to get our boards up and running. We also provide further electronic (email, wiki and discussion forum) support for evaluation of our products.
Customer product development support is not part of standard support offer from Atoll Solutions. If customers are interested, Atoll can offer product development services around Atoll solutions.

Usage Restriction:

Atoll products are excellent starting point for customer's applications development. But, selection and usage of Atoll Solutions products for a particular application is responsibility of customers. In order to minimize risks associated with customer applications, the customer must use adequate design and operating safeguards to minimize inherent or procedural hazards.

Atoll Solutions products are not intended for use in life support systems and appliances, nuclear systems or systems where malfunction can reasonably be expected to result in personal injury, death or severe property or environmental damage. Any use of products by the customer for such purposes are at the customer's own risk.
Off the shelf products from Atoll Solutions are commercial temperature grade. If customers are looking for Industrial or Extended temperature products, please order them specifically.

- Atoll Solutions will use parts made by various manufacturers in performing the repair. This can be different from the components used in the original products.
- The repaired products shall be warranted subjected to the original warranty only (If the original warranty period left was three months, the repaired product warranty will be only for three months)
- Customers shall agree that an independent third party assigned by Atoll Solutions may repair the products covered under this limited warranty.

RMA (Return Merchandise Authorization)

- Customer shall enclose the completed "Atoll Solutions RMA Service Form" with the returned packages.
- Customers shall provide all the relevant information of the defect in the "Atoll Solutions RMA Service Form". This will reduce delay in defect identification and repair.
- Customers shall take responsibility to ensure that the packages of defective Products are durable enough to be resistant against further damage and deterioration during shipment.
- In case of damages occurred during the transportation, the repair is treated as "Out of Warranty