MuYu

MY-BT50X Commands Guide

Version 1.0

Contact Us

Shenzhen MuYu Technology Co., Ltd
Email: info@muyusmart.com
Zipcode： 518100
Web: www.muyumodule.com

Revision History

Version	Date	Notes	Author
1.0	2024/01/01	First Release	Richard

Content

1. Introduction

This document provides a simple performance introduction and detailed explanation of AT commands of the MY-BT50X module, aiming to guide users to quickly become familiar with the development of the MY-BT50X module.

1.1 Overview

MY-BT50X module supports 2.4GHz BLE (Low Power Bluetooth) 5.2 protocol, with a transparent transmission rate of up to 6kBytes/s, supports master-slave integration, supports multiple connections, and supports up to 6 slave connections. Supported hardware interfaces include GPIO, UART, SPI, I2C, PWM, I2S, LED, ADC, etc. It has the best power consumption performance, RF performance, stability, versatility and reliability, and is suitable for various application scenarios and different power consumption requirements.

Support Profile:

GATT Server (slave)

GATT Client (master)

1.2 Default Setting:

Bluetooth Name	**MY-BT50X**
Broadcast Interval	**152ms**
Transmitter	**0DBm**
Serial Port Baud rate	**115200bps/8/N/1**
Service UUID	**FFF0**
Notify UUID	**FFF1**
Write or write without response UUID	**FFF2**
The default is slave mode. It starts broadcasting after powering on. If the slave device is connected using the **AT+CONN** command, it will automatically switch to the host mode.	

2. Command

2.1 AT command format

1. All command start with “AT”, end with <CR><LF>
2. <CR> stand for “carriage return”, corresponding hex is 0x0D

3. <LF> stands for “line feed”, corresponding hex is 0x0A
4. If command has parameter, parameter keep behind “=”
5. If command has multiple parameter, parameter must be separated by “,”
6. Module will always report command’s execution result using “**OK**” for success or “**ERROR**” for failure
7. In this document, << represents command input, >> represents command reply, and the content inside { } is optional.

AT+ Command {=Param1{, Param2{, Param3...}}} <CR><LF>

For example:

1. Read the version number

<< AT+VER

>> +VER=1.0.0,MY-BT502

>> OK

2. Change the illegal baud rate

<< AT+BAUD=1234

>>ERROR

2.2 General Command

2.2.1 UART Communication Test

Description: UART communication testing between HOST and Module
Format: AT
Response: OK
Eg: << AT >> OK

2.2.2 AT+CMDLIST

Description: AT command list, print and display all AT commands available in the current version

Format: AT+CMDLIST
Response: +CMDLIST=params
Eg: << *AT+CMDLIST* >> *+CMDLIST=* *{* *AT+NAME* *AT+VER* *...* *}* >> *OK*

2.2.3 AT+VER

Description: Read Firmware version
Format: AT+VER
Response: +VER=param

Eg: << *AT+VER* >> *+VER=1.0.0,MY-BT502* >> *OK*

2.2.4 AT+LEADDR

Description: Read module MAC address
Format: AT+LEADDR
Response: +LEADDR=param
Eg: << *AT+LEADDR* >> *+LEADDR=112233445566* >> *OK*

2.2.5 AT+LENAME

Description: Read, Setting Bluetooth Name Default name:**MY-BT502**
Format: AT+LENAME {=param1{, param2}}
param1: Device Name (Length1~25 Bytes ASCII) param2:(0~2) (0) Disable MAC address suffix (1) Device name + last 4 digits of MAC address (2) Device name + last 6 digits of MAC address

Response: +LENAME=param
Eg1. Read Bluetooth Name << *AT+LENAME* >> *+NAME=MY-BT502* >> *OK*
Eg2: Setting Bluetooth Name: AABBCCDDEEFF (Changes take effect immediately) << *AT+LENAME=AABBCCDDEEFF* >> *OK*
Eg3: Device name + last 4 digits of MAC address (Changes take effect immediately) << *AT+LENAME=AABBCCDDEEFF,1* >> *OK*
Eg4: Device name + last 6 digits of MAC address (Changes take effect immediately) << *AT+LENAME=AABBCCDDEEFF,2* >> *OK*

2.2.6 AT+BAUD

Description: Read, Setting Bluetooth UART Baud rate, default: 115200
Format: AT+BAUD {=param}
param: Baud rate, support: 1200/2400/4800/9600/14400/19200/38400/57600/115200/230400/460800/921600
Response: +BAUD=param

Eg1: Read << *AT+BAUD* >> *+BAUD=115200* >> *OK*
Eg2: Setting Baud rate: 9600 (Changes take effect immediately) << *AT+BAUD=9600* >> *OK*

2.2.7 AT+TPMODE

Description: Read / Setting Command and Throughput mode; Default: 1 (GATT Throughput Mode)
Format: AT+TPMODE {=param}
param:(0~1) (0) It is command mode in any state, including Bluetooth connection and Bluetooth disconnection. In command mode, any data received by UART will be analyzed for commands. In connection state, data can only be sent to the remote device through the AT+GATTSEND command. (1) GATT transparent transmission. When Bluetooth is disconnected, it is in command mode. You can use AT commands to change relevant parameters. When Bluetooth is connected, it is in GATT transparent transmission mode. AT commands are disabled. Any data received by UART will be sent to remote device
Response: +TPMODE=param
Eg1: Read Mode << *AT+TPMODE* >> *+TPMODE=1* >> *OK*
Eg2: Setting Command Mode (Changes take effect immediately) << *AT+TPMODE=0* >> *OK*

2.2.8 AT+LPM

Description: Read/Setting Low Energy Mode; Default: 0
Format: AT+LPM {=param}
param:(0~1) (0) Turn off low energy mode (1) Turn on low energy mode. After entering low-power mode, Bluetooth can broadcast and be connected. There are two ways to wake up. 1. The serial port sends the first packet of data to wake up. After waking up, the serial port starts to work. If no serial port data or APP data is received within 10 seconds, it will automatically Close the serial port and enter low power consumption again. 2. The APP sends data to wake up. Under low power consumption conditions, receiving APP data will wake up the serial port and output data. If no serial port or APP data is received within 10s, the serial port will automatically close and enter low power consumption mode. The time (10s) can be changed through the **AT+LPDLY** command
Response: +LPM=param
Eg1: Read Bluetooth status << *AT+LPM* >> *+LPM=0* >> *OK*
Eg2: Setting Low Energy Mode (Changes take effect immediately) << *AT+LPM=1* >> *OK*

2.2.9 AT+REBOOT

Description: Software reset, system restart
Format: AT+REBOOT

Response: OK
Eg: << *AT+REBOOT* >> *OK*

2.2.10 AT+RESTORE

Description: All Bluetooth parameters are restored to factory settings and the system is restarted.
Format: AT+RESTORE
Response: OK
Eg: << *AT+RESTORE* >> *OK*

2.2.11 AT+TXPOWER

Description: Read/ Setting Bluetooth module Transmitter Power, default: C (0DBm)
Format: AT+TXPOWER{=param}
Param (0~F)
Response: +TXPOWER=param

Eg1: Read << *AT+TXPOWER* >> *+TXPOWER=C* >> *OK*
Eg2: Setting 5DBm transmitter power (Changes take effect immediately) << *AT+TXPOWER=F* >> *OK*

2.2.12 AT+PIN

Description: Read/Setting Connection PIN Code
Format: AT+PIN{=param}
param: Pin code (6 Bytes ASCII)
Response: +PIN=param
Eg1: Read << *AT+PIN* >> *+PIN=000000* >> *OK*
Eg2: Setting PIN Code: 123456 (Changes take effect immediately) << *AT+PIN=123456* >> *OK*

2.2.13 AT+CLOSEAC

Description: Close AirCommand Mode (Only for AirCommand Mode)
Format: AT+CLOSEAC
Response: AirCommandClosed
Eg: << *AT+CLOSEAC* >> *AirCommandClosed*

2.2.14 AT+GPIOCFG

Description: Control the input function switch of two IOs, default (0,0)
Format: AT+GPIOCFG{=param1{,param2}}
param1:(0~1) (0) Disable command/ throughput mode switching function (1) Enable command/ throughput mode switching function param2:(0~1) (0) Disable Bluetooth disconnect function (1) Enable Bluetooth disconnect function
Response: +GPIOCFG=param
Eg1: Read IO Status << *AT+GPIOCFG* >> *+GPIOCFG=0,0* >> *OK*
Eg2: Enable command/transparent transmission mode switching function; disable Bluetooth disconnect function (Changes take effect immediately) >> *AT+GPIOCFG=1,0*

>> *OK*
Eg3: Disable the command/transparent transmission mode switching function and enable the Bluetooth disconnection function. (Changes take effect immediately) >> *AT+GPIOCFG=0,1* >> *OK*

2.2.15 AT+DISC

Description: Disconnect Bluetooth (Only be used in command mode)
Format: AT+DISC{=param}
Response: OK
param: Without parameters, disconnect all connections; with parameters, disconnect the specified channel.
Eg1: Disconnect all connections << *AT+DISC* >> *OK* **Eg2:** Disconnect devices of Channel 0 << *AT+DISC=0* >> *OK*

2.3 GATT

2.3.1 AT+GATTSTATE

Description: Check Bluetooth connection State
Format: AT+GATTSTATE
Response: +GATTSTATE=param1,param2
param1(0~8): Connection Channel param2(1~3):1. Not connected, 2 Connecting, 3 Connected
Eg: Check Bluetooth connection State << *AT+GATTSTATE* >> *+GATTSTATE=0,3* >> *OK*

2.3.2 AT+GATTINFO

Description: Read Bluetooth connection State
Format: AT+GATTINFO
Response: +GATTINFO=param1,param2,param3,param4,param5
param1(0~5)： Connection Channel param2(1~3)： 1. Not connected, 2 Connecting, 3 Connected param3(0~1)： Connection master-slave mode, 0 master mode, 1 slave mode; 255 disconnect

param4：MAC address of the connected remote device param5：Connection MTU size
Eg: Read Bluetooth connection State << *AT+GATTSTATE* >> *+GETINFO=* >> *{* >> *+GETINFO=0,1,255,000000000000,23* >> *}* >> *OK*

2.3.3 AT+IBEACON

Description: Read/ Setting iBeacon broadcasting function, default: 1
Format: AT+IBEACON {=param}
param:(0~2) (0) Turn off iBeacon (1) Turn on iBeacon, set the iBeacon content through the command **AT+ADVDATA**
Response: +IBEACON=param
Eg: Read << *AT+IBEACON* >> *+IBEACON=1* >> *OK*
Eg2: turn off iBeacon broadcast (Changes take effect immediately)

<< *AT+IBEACON=0* >> *OK*

2.3.4 AT+ADVDATA

Description: Read, Setting iBeacon broadcast content
Format: AT+ADVDATA{=param}
param: iBeacon data (2~56 Bytes ASCII)
Response: +ADVDATA=param
Eg1: Read << *AT+ADVDATA* >> *+ADVDATA=4C0002155B198FF269A011EE8C990242AC12000200000000B5* >> *OK*
Eg2: Setting iBeacon broadcast data: 0x4C 0x00 0x02 0x15 0x5B 0x19 0x8F 0xF2 0x69 0xA0 0x11 0xEE 0x8C 0x99 0x02 0x42 0xAC 0x12 0x00 0x02 0x00 0x00 0x00 0x00 0xB6(Changes take effect immediately) << *AT+ADVDATA=4C0002155B198FF269A011EE8C990242AC12000200000000B6* >> *OK*

2.3.5 AT+ADBLE

Description: Broadcast Bluetooth MAC address, default: 1
Format: AT+ADBLE{=param}

param:(0~1) (0) Turn off broadcast MAC address function. (1) Turn on broadcast MAC address function
Response: +ADBLE=param
Eg1: Read Broadcast Bluetooth MAC address << *AT+ADBLE* >> *+ADBLE=1* >> *OK*
Eg2: Turn off the broadcasting of MAC address (Changes take effect immediately) << *AT+ADBLE=0* >> *OK*

2.3.6 AT+ADVIN

Description: Read/Setting Broadcast interval, default: 152s
Format: AT+ADVIN {=param}
param:(25~10000), unit: ms
Response: +ADVIN=param
Eg1: Read << *AT+ADVIN* >> *+ADVIN=152*

>> *OK*
Eg2: Setting broadcast interval is 1000s (Changes take effect immediately) << *AT+ADVIN=1000* >> *OK*

2.3.7 AT+GATTSEND

Description: Send data to remote device (Only on Command mode using)
Format: AT+GATTSEND =param1,param2,param3
param1: Connection channel number (0~5) Param2: Data length (1~182) Param3: Data（1~182Bytes）
Response: OK
Eg: Send the data “0123456789” to remote device << *AT+GATTSEND=0,10,0123456789* >> *OK*

2.4 Host Command

2.4.1 AT+SCAN

Description: Scan as a host to obtain the broadcast information of the slave

Format: AT+SCAN=param1{,param2{,param3}}
param1(0~2) (0) Stop scanning (1) (1) Scan the surrounding BLE devices to obtain MAC type, MAC, rssi, name len, name and other information. Use the default time scan of 10 seconds to automatically end the scan. By default, only 7 pieces of information of different BLE devices can be stored and printed. (2) param2:(100~10000, unit: ms), scanning time param3: Scan for devices with specified Bluetooth names
Response: +SCAN=param1,param2,param3,param4,param5,param6 Param(1): param1: serial number, Param2: MAC address type, param3: MAC address, param4: RSSI signal value, param5: length of device name, param6: device name Param(2): Param1: MAC address type, param2: MAC address, param3: RSSI signal value, param4: broadcast type, param5: broadcast length, param6: broadcast content
Eg1: Scan device name using default time << *AT+SCAN=1*
Eg2: Scan the device name and set the scan time to 1000ms. << *AT+SCAN=1,1000*
Eg3: Scan the device name, and set the scan time to 1000ms, and only scan the device named MY-BT502 << *AT+SCAN=1,1000,MY-BT502*
Eg4: Scan and print all broadcast information << *AT+SCAN=2*

2.4.2 AT+UUID

Description: When connected as a host, it will look for the UUID of the slave communication. Before establishing a Bluetooth connection, the slave communication UUID needs to be registered in the host. The default is FFF0, FFF1, FFF2
Format: AT+UUID{=param1,param2,param3}
param1: service uuid param2: notify uuid Param3: write uuid
Response: +GUUID=param1,param2,param3
Eg1: Check the info << *AT+UUID* >> *+UUID=FFF0,FFF1,FFF2* >> *OK*
Eg2: The slave UUID is service FFF0, notify FFF1, write FFF2, and the registration instructions are as follows (Changes take effect immediately) << *AT+UUID=FFF0,FFF1,FFF2* >> *OK*

2.4.3 AT+CONN

Description: Connecting slave devices as a master
Format: AT+CONN=param1param2

param1: Device MAC address param2: Device MAC address type
Response: OK
Eg1: The master initiates the connection (specifies the MAC address and address type) << *AT+CONN=1122334455660* >> *OK*

2.4.4 AT+AUTOCFG

Description: Turn on or off the host mode automatic connection function. The default value is 0. After setting it to 1, the module will record the information of the host's last connection to the device, and will automatically reconnect when the host is restarted or disconnected abnormally.
Format: AT+AUTOCFG{=param}
Param (0~1): (0) Disable the master mode auto-connect function (1) Enable the master mode auto-connect function
Response: OK
Eg1: Read host mode automatic connection function << *AT+AUTOCFG* >> *+AUTOCFG=0* >> *OK*
Eg2: Turn on host mode automatic connection function (Changes take effect immediately)

<<	*AT+AUTOCFG=1*
>>	*OK*

3. Detailed explanation of air commands

MY-BT50X supports entering AT command mode through air data. After entering air command mode, you can send commands to change related parameters.

Usage steps:

1. Use APP to establish a connection with the Bluetooth module

2. APP sends **$OpenMysAtEngine$** to the Bluetooth module. If APP can receive the data **$OK,Opened$** fed back by the module, it successfully enters the air command mode

3. The command format of air commands and UART is the same. Please refer to 2.1~2.4.4

4. After changing the parameters, you can send **AT+CLOSEAC** command to close the air command mode. If it is closed successfully, you will receive a reply **AirCommandClosed** or disconnect the Bluetooth connection to automatically exit the air command mode.

Note: In order to prevent errors in transparent data transmission, if the first data packet received by the module after Bluetooth connection is not **$OpenMysAtEngine$**, the module will no longer enter the air command mode unless it is disconnected and reconnected.

4. Notes

1. **Throughput mode:** any data received by UART will be sent to the remote device through the GATT protocol.
2. **Command mode:** any data received by UART in any state will be parsed and processed, and will not be sent directly to the remote device.
3. **Air command mode:** any data sent by the remote device will be parsed and processed by the command, and will not be sent out through UART