1 文档介绍

1.1 文档范围

本手册详细介绍了UC8288 WIOTA终端模块提供的AT指令集。

1.2 命令语法

1.2.1 命令格式

本手册中所有命令行必须以”AT”或”at”作为开头，以回车（）作为结尾。响应通常紧随命令之后，且通常以”<回车><换行><响应内容><回车><换行>”（<响应内容>）的形式出现。在命令介绍时，“<回车><换行>”（）通常被省略了。

1.2.2 命令类型

通常命令可以有如下表所示的四种类型中的一种或多种形式。

类型	格式	说明
测试命令	AT+<cmd>=?	用于查询设置命令或内部程序设置的参数及其取值范围
查询命令	AT+<cmd>?	用于返回参数的当前值
设置命令	AT+<cmd>=<...>	用于设置用户自定义的参数值
执行命令	AT+<cmd>	用于读取只读参数或不需要额外参数的情况

1.2.3 参数类型

命令参数虽然多种多样，但是都可以简单地归结为整数类型和字符串类型（包括不带双引号的字符串和带双引号的字符串）这两种基本的类型，如下表所示。

类型	示例
整数类型	123
字符串类型	abc
	"hellow ,world"

1.2.4 注意事项

- AT串口输入时不支持回删键(backspace)功能
- 本文档+ERROR指+CME ERROR或者+EXT ERROR

2 基础 AT命令详细说明

2.1 AT

&AT测试命令。

Command	Possible response(s)
AT	OK ERROR

2.2 AT+RST 重启

系统重启。

Command	Possible response(s)
+RST	OK ERROR

watchdog重启，执行RST返回OK后，1s后watchdog重启。

2.3 ATE 回显

AT指令回显功能。

Command	Possible response(s)
ATE<value>	OK ERROR

- <value>：默认AT回显关闭
- 0： 关闭回显
- 1： 打开回显

2.4 AT&L 查询AT列表

查询支持的AT列表。

Command	Possible response(s)
AT&L	OK ERROR

2.5 AT+UART UART0配置

UART0配置。

Command	Possible response(s)
AT+UART= <baudrate>, <databits>, <stopbits>, <parity>, <flow_control>	OK ERROR

- <baudrate>：波特率，最大支持的波特率921600.
- <databits>：有效数据长度

- <stopbits>：停止位
- <parity>：奇偶检验
- <flow_control>: 流控。不支持流控.

2.6 AT+YMODEM 进入Ymodem刷机模式

进入Ymodem串口刷机模式。

Command	Possible response(s)
AT+YMODEM	OK ERROR

2.7 系统上报

Command	Mean
+CHOOSEMODEM:D	等待2S输入'D'进入Ymodem下载模式
+SYSTEM:START	启动RT-THREAD系统

3 WITOA AT命令详细说明

3.1 AT+WIOTAVERSION 查询版本信息

查询当前wiota库的版本号、git 信息、编译生成库的时间。

Command	Possible response(s)
+WIOTAVERSION	+WIOTAVERSION?:<VERSION> +GITINFO:<GITINFO> +TIME:<maketime> +CCEVERSION:<cceversion> OK

- WIOTAVERSION：
  当前WIoTa库版本号
- GITINFO：
  当前库的git信息
- TIME：
  当前库的生成时间
- CCEVERSION:
  CCE版本号
- 举例：
  发送：
  AT+WIOTAVERSION?
  回显：
  +WIOTAVERSION:v0.10_iote
  +GITINFO:Fri Apr 15 14:20:26 2022
  +TIME:Apr 20 2022 11:42:23

+CCEVERSION:b5aac93
OK

3.2 AT+WIOTAINIT 初始化

初始化WIoTa终端的资源。

Command	Possible response(s)
+WIOTAINIT	OK ERROR

- 举例：
  发送：
  AT+WIOTAINIT
  回显：
  OK

3.3 AT+WIOTALPM 低功耗

低功耗设置

Command	Possible response(s)
+WIOTALPM=<mode>,<state>	OK ERROR

- <mode>：
- 0： sleep模式。外部串口唤醒后重新启动。该模式暂未支持。
- 1： Gating模式。WIoTa协议栈在空闲的时候进入Gating。
- <state>：
- 0： 关闭。
- 1： 打开。
- 举例：
  发送：
  AT+WIOTALPM=1,1
  回显：
  OK

3.4 AT+WIOTARATE 传输速率配置

设置最大速率模式和级别，三种模式：
第一种基本模式，是基本速率设置，有9档mcs速率级别（包括自动mcs），详见UC_MCS_LEVEL，默认为自动mcs，设置非自动mcs时同时关闭自动速率匹配功能

在第一种模式的基础上，在系统配置中dlul为1:2时，才能打开第二种模式，打开该模式能够提高该帧结构情况下两倍速率，默认第二种模式开启状态

在第一种模式的基础上，打开第三种模式，能够提升（8*(1 << group_number)）倍单终端的速率，但是会影响网络中其他终端的上行，建议在大数据量快速传输需求时使用，默认第三种模式关闭
备注：group_number为系统配置中的参数

Command	Possible response(s)
+WIOTARATE=<rate_mode> <rate_value>	OK ERROR

- <rate_mode>：枚举UC_DATA_RATE_MODE
- <rate_value>：当rate_mode为UC_RATE_NORMAL时，rate_value为UC_MCS_LEVEL
  当rate_mode为UC_RATE_MID时，rate_value为0或1，表示关闭或打开
  当rate_mode为UC_RATE_HIGH时，rate_value为0，表示关闭，rate_value为其他值，表示当实际发送数据量（byte）大于等于该值时才会真正开启该模式，常用建议设置rate_value为100

```
typedef enum {
  UC_RATE_NORMAL = 0,
  UC_RATE_MID,
  UC_RATE_HIGH,
}UC_DATA_RATE_MODE;

typedef enum {
  UC_MCS_LEVEL_0 = 0,
  UC_MCS_LEVEL_1,
  UC_MCS_LEVEL_2,
  UC_MCS_LEVEL_3,
  UC_MCS_LEVEL_4,
  UC_MCS_LEVEL_5,
  UC_MCS_LEVEL_6,
  UC_MCS_LEVEL_7,
  UC_MCS_AUTO = 8,
}UC_MCS_LEVEL;
```

BT_0.3时在不同symbol length和不同MCS时，对应每帧传输的应用数据量（byte），表中0表示不支持该MCS

symbol length	mcs0	mcs1	mcs2	mcs3	mcs4	mcs5	mcs6	mcs7
128	5	7	50	64	78	0	0	0
256	5	13	20	50	106	155	190	0
512	5	13	29	40	71	134	253	295
1024	5	13	29	61	106	218	449	617

初始化协议栈时默认打开自动速率匹配功能，调用该接口入参为0~7时，设置最大速率级别，同时关闭自动速率匹配功能，再次调用该接口入参为UC_MCS_AUTO（或者不是0~7）时，会打开自动速率匹配功能。
为了保证接入成功率，接入短消息暂只使用mcs0~3，由于其中需要携带user id，正常会再减去4个字节空间，实际给应用的数据量会比正常短消息少。
接入短消息的MCS还有其他限制（应用层可不关注），symbol length为128/256/512/1024时，接入短消息的MCS最高为1/2/3/3。

每帧时间长度（frameLen）的粗略计算公式：（单位微妙）

```
// dlGroupNum和ulGroupNum取值0,1,2,3，ulGroupNum即系统参数配置中的group_number
groupNum = (1 << dlGroupNum) + (1 << ulGroupNum);
symbolNum = 11 + 2 * (1 << pn_num) + 64 * groupNum;  // pn_num目前固定为1
frameLen = symbolNum * 4 * 128 * (1 << symbol_length);  // symbol_length取值为0,1,2,3
```

举例： 系统配置中group_num为0，dlul_ratio为0，symbol_length为1，则

```
groupNum = 1 + 1 = 2;
symbolNum = 15 + 128 = 143;
frameLen = 143 * 4 * 128 * 2 = 146432 us
```

在此帧结构配置情况下，如果选择MCS2，则应用数据速率为 8*20/0.146432 = 1093 bps
(计算上行数据速率时，一般不考虑第一个包即随机接入包)

- 注意
  一味提高速率，可能导致上行始终无法成功
- 举例：
  发送：
  AT+WIOTARATE=0,3
  回显：
  OK

3.5 AT+WIOTAPOW 发射功率配置

低功耗设置

Command	Possible response(s)
+WIOTAPOW=<mode>,<power>	OK ERROR

- <mode>：
- 0：设置当前发射功率。
- 1：设置最大发射功率。
- <power>：发射功率。范围-16 ~ 21db。tag0.09版本及之前由于代码限制，不支持负数解析，如at+wiotapow=0,-10，需要写成补码形式，即at+wiotapow=0,246。tag0.09版本之后，实际需要设置的功率加20则为输入值，例如想要设置功率-10，则 at+wiotapow=0,10，想要设置功率20，则 at+wiotapow=0,40
- 如果设置当前功率值为正常范围值，则设置成该功率，并且关闭自动功率模式；如果功率值为127（0x7F），则代表恢复自动功率模式
  - 举例：
    发送：
    AT+WIOTAPOW=0,40
    回显：
    OK

3.6 AT+WIOTAFREQ 锁频

设置频点，iote和ap需要设置相同频点才能同步。在初始化系统之后，在系统启动之前调用，否则无法生效。

Command	Possible response(s)
+WIOTAFREQ=<freqpint>	OK ERROR
+WIOTAFREQ?	OK ERROR

- <freqpint>:
  频点idx，范围0~200，代表频点（470M+0.2*idx)。
  - 举例:
    发送:
    AT+WIOTAFREQ=115
    回显:
    OK
    发送:
    AT+WIOTAFREQ?
    回显:
    +WIOTAFREQ=115
    OK

3.7 AT+WIOTADCXO 设置频偏

设置终端频偏。在初始化系统之后，在系统启动之前调用，否则无法生效。

Command	Possible response(s)
+WIOTADCXO=<dcxo>	OK ERROR

- <dcxo>:
- 硬件的频偏参数，输入参数是16进制。有源晶体不能设置。
  - 举例:
    发送:
    AT+WIOTADCXO=20000
    回显:
    OK

3.8 AT+WIOTAUSERID 设置用户ID

设置终端userid。获取用户id，此id为终端唯一标识。在初始化系统之后，在系统启动之前调用，否则无法生效。
目前只支持4字节长度的user id.

Command	Possible response(s)
+WIOTAUSERID=<id0>	OK ERROR
+WIOTAUSERID?	+WIOTAUSERID:<id0> OK

- <id0>:
  获取用户id，此id为终端唯一标识。长度为4个字节。id是0-0xFFFFFFFF (16进制格式输入，不需要0x)
  - 举例：
    发送：
    AT+WIOTAUSERID=ae81c452
    回显：
    OK
    发送：
    AT+WIOTAUSERID?
    回显：
    +WIOTAUSERID=0xae81c452
    OK

3.9 AT+WIOTACONFIG 系统配置

设置系统配置。

Command	Possible response(s)
+WIOTACONFIG=<id_len>, <symbol>,<dlul>,<bt>, <group_num>,<ap_max_pow>, <spec_idx>,<systemid>, <subsystemid>	OK ERROR
+WIOTACONFIG?	+WIOTASYSTEMCONFIG:<id_len>,<symbol>,<dlul>, <bt>,<group_num>,<ap_max_pow>,<spec_idx>, <systemid>,<subsystemid> OK

- <id_len>: user id长度，取值0,1,2,3代表2,4,6,8字节
- <symbol>: 帧配置，取值0,1,2,3代表128,256,512,1024
- <dlul>: 帧配置，下上行比例，取值0,1代表1:1和1:2
- <bt>: 调制信号的滤波器带宽对应，BT越大，信号带宽越大，取值0,1代表1.2和0.3，BT_1.2的数据率比BT_0.3大
- <group_num>: 帧配置，取值0,1,2,3代表1,2,4,8个上行group数量
- <ap_max_pow>: ap最大功率，暂时0~30dbm，需要与AP侧配置一致，实际需要设置的功率加20则为输入值，更详细的解释参见3.5节AT+WIOTAPOW功率参数
- <spec_idx>: 使用的频段序号
- <systemid>: 系统id，每个id是0-0xFFFFFFFF (16进制格式输入，不需要0x)
- <subsystemid>: 子系统id， 每个id是0-0xFFFFFFFF (16进制格式输入，不需要0x)
  - 举例：
    发送：
    AT+WIOTACONFIG=1,1,0,1,0,20,3,11223344,21456981
    回显：
    OK

发送：
AT+WIOTACONFIG?
回显：
+WIOTASYSTEMCONFIG=1,3,0,1,0,0,3,0x11223344,0x21456981
OK

3.10 AT+WIOTARUN 启动/关闭WIoTa协议栈

启动wiota系统，进入空闲状态。
关闭wiota后，回收系统资源

Command	Possible response(s)
+WIOTARUN=<state>	OK ERROR

- <state>:
- 0： 关闭协议栈，回收WIoTa资源
- 1： 启动协议栈，进入空闲状态
- 举例：
  发送：
  AT+WIOTARUN=1
  回显：
  OK

3.11 AT+WIOTASCANFREQ 扫频

在wiota启动后扫描频点信息，可扫一组频点和全扫，返回扫频结果，执行该命令后需要在窗口工具的发送区输入长度为dataLen（dataLen只能大于或等于输入的字符串长度，不能小于否则会获取字符串失败），个数为freqNum的字符串，并点击发送。

Command	Possible response(s)
+WIOTASCANFREQ =<timeout>, <dataLen>,<freqNum> ;	+WIOTASCAFREQ:(freq,rssi,snr,is_synced) OK > ERROR
+WIOTASCANFREQ	+WIOTASCAFREQ:(freq,rssi,snr,is_synced) OK ERROR

- <timeout>：扫描超时时间，单位ms。默认超时时间是2分钟。
- <dataLen>：发送字符串的总长度+\r\n，比如要扫描的频点为1,2,3,4,5这五个频点
  1）执行at命令AT+WIOTASCANFREQ=10000,11,5;
  2）当出现>时十秒钟内在串口工具的发送区内输入字符串1,2,3,4,5
  3）点击发送
  4）等待扫频结果返回，结果会通过串口打印出来
- <freqNum>: 频点个数
- freq：频点信息

- rssi：信号强度
- snr：信噪比
- is_synced：该频点是否能同步
    - 举例：
      发送：
      AT+WIOTASCANFREQ=60000,17,4
      >
      119,115,118,120

      回显：
      OK
      +WIOTASCANFREQ:
      115,-83,3,1
      120,-79,0,0
      119,-80,0,0
      118,-84,0,0
      OK

3.12 AT+WIOTARADIO 无线状态

只有在wiota同步成功后才能查询wiota无线状态信息，否则数据没有任何参考意义。

Command	Possible response(s)
+WIOTARADIO?	+WIOTARADIO=<temp>,<rssi>,<<ber>,<snr>, <cur_pow>,<max_pow>,<cur_mcs>,<max_mcs> OK ERROR

无线状态数据：

- temp: 当前芯片温度
- rssi: 信号强度
- ber: 误码率，暂不支持
- snr: 信噪比，范围 -25dB ~ 30dB
- cur_pow: 当前发射功率，范围 -16~21dBm
- max_pow: 最大发射功率，范围 -16~21dBm
- cur_mcs: 当前数据发送速率级别，范围 0~7
- max_mcs: 截止目前最大数据发送速率级别，范围 0~7
    - 举例：
      发送：
      AT+WIOTARADIO?
      回显：
      +WIOTARADIO=31,-22,0,20,-16,21,5,5
      OK

3.13 AT+WIOTACONNECT WIoTa连接ap

连接或断开与AP的同步。

Command	Possible response(s)
+WIOTACONNECT=<state>, <activetime>	OK ERROR

- <state>：
  - 0：断开连接，WIoTa进入空闲状态
  - 1：WIoTa连接ap，成功后进入同步状态
- <activetime>:
  - 连接保持时间，单位是秒（s）。默认设置与AP匹配，与帧长有关，具体计算参见API接口文档中的“设置终端连接时间”，最小参数值为1，当参数为0时，表示不修改参数，使用默认配置。(断开连接时填0)
  - 举例：
    发送：
    AT+WIOTACONNECT=1,0
    回显：
    OK
    发送：
    AT+WIOTACONNECT?
    回显：
    +WIOTACONNECT=1,0 (第一个参数1表示同步，0代表未同步，其他含义参考API文档中uc_wiota_get_state接口，第二个参数是当前activetime)
    OK

3.14 AT+WIOTASEND WIoTa发送数据

数据发送。

Command	Possible response(s)
+WIOTASEND=<timeout>,<len>	OK > ERROR
+WIOTASEND	> data OK ERROR

指定数据长度发送流程：

- <len>：数据的长度，最大310字节。
- <timeout>:发送超时时间，单位ms。取值范围0-65535. 0代表试用默认值（60s）。
- 发送失败返回"ERROR"， 发送数据成功返回“OK”。

无数据长度的数据发送流程：

- > :运行发送数据标志。一包数据最长为310字节。数据超过最长包310将被丢掉。如果应用层需要传超过310字节的数据，建议自己先分包。
- 在每读到一个字符之后等待10s，等待后续数据输入。

- 默认发送超时时间为60s
    - 举例：
      发送：
      AT+WIOTASEND=5000,12
      >
      123456789012
      回显：
      SEND SUCC
      OK

3.15 AT+WIOTATRANS 数据透传模式

进入数据透传模式，在发送任意长度（不超过310字节）数据后不会立即退出。

Command	Possible response(s)
+WIOTATRANS=<timeout>, <endflag>	Enter transmission mode > Leave transmission mode OK
+WIOTATRANS	Enter transmission mode > Leave transmission mode OK

指定结束标记的数据发送流程：

- <timeout>: 发送超时时间，单位ms。取值范围0-65535，0代表使用默认值（60s）。
- <endflag>: 指定退出数据透传模式标记，标记可为任意可见字符，长度最少为1个字节，最长不超过8个字节。
- Enter transmission mode >: 进入透传模式
- Leave transmission mode: 退出透传模式
- 发送失败返回"SEND FAIL"， 发送数据成功返回“SEND SUCC”。

不指定结束标记的数据发送流程：

- 采用默认结束标记"$$$$"
- 默认发送超时时间为60s
    - 举例：
      发送：
      AT+WIOTATRANS=5000,AA
      Enter transmission mode >
      123456789012
      回显：
      SEND SUCC
      发送：
      123456789012AA
      回显：
      SEND SUCC
      Leave transmission mode
      OK

注: 与普通AT命令类似，"\r\n"作为发送标记，发送内容不包含"\r\n"，同时发送内容也不包含结束标记，只有实际消息的内容。

3.16 +WIOTARECV WIoTa数据上报

接收数据上报。

Command	Possible response(s)
无	+WIOTARECV=<type>,<len>,<data>

- <type>：上报数据类型
  - 0： 短消息
  - 1：广播消息
  - 2：OTA消息
  - 3：扫频结果
  - 4：同步异常
- <len>：上报的数据长度
- <data>：数据长度不为0时，上报的数据
  - 举例：
    回显：
    +WIOTARECV,0,12,123456789012
    OK
    回显：
    +WIOTARECV,0,12,1234567890 （有时候是这样，可能是AP发送的数据是带了\r\n的，所以显示不出来，实际传输的还是12个字符）
    OK

3.17 +WIOTALOG WIoTa log设置

WIoTa log设置。

Command	Possible response(s)
+WIOTALOG=<type>	OK

- <type>：LOG类型
  - 0：关rv uart log，即协议栈串口LOG
  - 1：开rv uart log
  - 2：rv uart log使用uart0，如果从uart1切换到uart0，会把uart0的波特率改为460800，此时AT的波特率也是用该值
  - 3：rv uart log使用uart1，如果从uart0切换到uart1，会把uart0的波特率恢复为115200
  - 4：关rv spi log，即协议栈SPI LOG，需要通过另外的TRACE工具抓取
  - 5：开rv spi log
- 注意：默认状态下，rv uart log使用uart1，波特率460800，AT使用uart0，波特率115200，在rv uart log的串口切换后，需要特别注意串口工具使用的波特率是否对应，如果AT的波特率不对时，发送at cmd会直接导致at挂住！
  - 举例：
    发送：
    AT+WIOTALOG=2
    回显：
    B （由于此时AT的波特率已经切换成460800，但是工具还是原来的115200波特率接收，所以字符都是乱码，此时OK显示成了B，此时需要将工具波特率改为460800，才能正常显示rv uart log和使用AT命令）
    发送：

AT+WIOTALOG=3 （在上一步基础上，先把工具波特率改为460800，然后再发该AT）
回显：
鴢x黿繡<□\0€x\0 （由于此时AT的波特率已经切换成115200，但是工具还是原来的460800波特率接收，所以字符都是乱码，此时OK显示成了乱码，乱码有可能不同，此时需要将工具波特率改为115200，才能正常使用AT命令）

3.18 AT+WIOTASTATS WIoTa统计信息

WIoTa统计信息

Command	Possible response(s)
+WIOTASTATS=<mode>,<type>	+WIOTASTATS=type,data OK > ERROR
+WIOTASTATS?	+WIOTASTATS=0,rach_fail,active_fail,ul_succ,dl_fail,dl_succ,bc_fail,bc_succ OK

- <mode>：UC_STATS_MODE，0: 读数据， 1: 重置数据
- <type>: UC_STATS_TYPE，需要获取的数据类型

```
typedef enum {
    UC_STATS_READ = 0,
    UC_STATS_WRITE,
}UC_STATS_MODE;

typedef enum {
    UC_STATS_TYPE_ALL = 0,
    UC_STATS_RACH_FAIL,
    UC_STATS_ACTIVE_FAIL,
    UC_STATS_UL_SUCC,
    UC_STATS_DL_FAIL,
    UC_STATS_DL_SUCC,
    UC_STATS_BC_FAIL,
    UC_STATS_BC_SUCC,
    UC_STATS_UL_SM_SUCC,
    UC_STATS_UL_SM_TOTAL,
    UC_STATS_TYPE_MAX,
}UC_STATS_TYPE;
```

- 举例：
- AT+WIOTASTATS=0,0 和 AT+WIOTASTATS? 的返回数据一样
- AT+WIOTASTATS=0,4，返回+WIOTASTATS=4,(下行失败次数)
- AT+WIOTASTATS=1,4，重置下行失败次数

3.19 AT+WIOTACRC WIoTa校验设置

CRC校验设置。

Command	Possible response(s)
+WIOTACRC=<crc_limit>	OK > ERROR
+WIOTACRC?	+WIOTACRC=1 OK

- <crc_limit>：0: 关闭CRC校验功能， 大于1: 表示数据长度大于等于crc_limit时，才打开CRC校验功能，所以crc_limit设置为1，则可表示任意长度的数据均加CRC
- 举例：
  发送：
  AT+WIOTACRC=100
  回显：
  OK

3.20 AT+WIOTAOSC 有源晶体设置

设置是否有源晶体版本的硬件，此项设置与DCXO设置互斥，如果设置了有源晶体，就不能再设置DCXO

Command	Possible response(s)
+WIOTAOSC=<mode>	OK > ERROR
+WIOTAOSC?	+WIOTAOSC=1 OK

- <mode>：0: 设置非有源晶体， 1: 设置有源晶体
- 举例：
  发送：
  AT+WIOTAOSC=1
  回显：
  OK

3.21 AT+WIOTALIGHT 指示灯设置

开关指示灯，在二次开发版本中，可关闭指示灯，即停止协议栈对相应GPIO（2/3/7/16/17）的操作，避免冲突

Command	Possible response(s)
+WIOTALIGHT=<mode>	OK > ERROR

- <mode>：0: 关闭指示灯， 1: 打开指示灯

- 举例：
  发送：
  AT+WIOTALIGHT=1
  回显：
  OK

3.22 AT+WIOTASAVESTATIC 保存用户静态数据

在IOTE非RUN状态下执行用户静态数据保存。

Command	Possible response(s)
+WIOTSAVESTATIC	OK

- 举例：
  发送：
  AT+WIOTASAVESTATIC
  回显：
  OK

4 WIOTA 测试 AT

[16:16:16.485]发→◇at+wiotainit
□
[16:16:16.520]收←◆OK

[16:16:26.203]发→◇at+wiotafreq=115
□
[16:16:26.216]收←◆OK

[16:16:35.942]发→◇at+wiotauserid=63c8b54c
□
[16:16:35.959]收←◆OK

[16:16:42.244]发→◇at+wiotafreq?
□
[16:16:42.247]收←◆+WIOTAFREQ=115
OK

[16:16:44.580]发→◇at+wiotauserid?
□
[16:16:44.586]收←◆+WIOTAUSERID=0x63c8b54c
OK

[16:17:22.244]发→◇at+wiotaconfig=1,1,0,1,0,20,3,11223344,21456981
□
[16:17:22.261]收←◆OK

[16:17:25.763]发→◇at+wiotarun=1
□
[16:17:25.797]收←◆OK

[16:17:27.164]发→◇at+wiotaconnect=1,0
□
[16:17:27.173]收←◆OK

[16:17:29.516]发→◇at+wiotaconnect?
□
[16:17:29.528]收←◆+WIOTACONNECT=1,3
OK

[16:19:50.836]发→◇at+wiotasend=5000,22
12345012345678901234
□
[16:19:50.847]收←◆>
[16:19:51.182]收←◆SEND SUCC
OK

[16:20:28.045]收←◆+WIOTARECV,0,12,1234567890

[16:20:38.327]发→◇at+wiotaconnect=0,0
□
[16:20:38.349]收←◆OK

[16:20:39.404]发→◇at+wiotarun=0
□
[16:20:40.044]收←◆OK