

WS73V100

# 命令使用指南

文档版本 11

发布日期 2025-02-20

## 前言

### 概述

本文介绍 WS73V100 的指令格式及应用场景，为用户提供相应的指令格式和参数示例解释。




### 读者对象



本文档主要适用于以下工程师：

- 软件开发工程师
- 软件测试工程师
- 技术支持工程师

### 符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 <b>危险</b>	表示如不可避免则将会导致死亡或严重伤害的具有高等级风险的危害。
 <b>警告</b>	表示如不可避免则可能导致死亡或严重伤害的具有中等级风险的危害。
 <b>注意</b>	表示如不可避免则可能导致轻微或中度伤害的具有低等级风险的危害。

符号	说明
 须知	用于传递设备或环境安全警示信息。如不可避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

修改记录

文档版本	发布日期	修改说明
11	2025-02-20	<ul style="list-style-type: none"><li>更新“4.1.2 BLE AT 指令描述”章节内容。</li><li>更新“4.2.2 SLE AT 指令描述”章节内容。</li><li>更新“5.5 产测指令描述”章节内容。</li></ul>
10	2024-10-22	<ul style="list-style-type: none"><li>更新“2.1 通用指令一览表”章节内容。</li><li>新增“2.2.8 暂停模组”章节内容。</li><li>新增“2.2.9 恢复模组”章节内容。</li><li>更新“4.2.2 SLE AT 指令描述”章节内容。</li><li>新增“4.1.2.1.13 AT+BLESETADV DATFLT 设置 BLE 扫描过滤参数”章节内容。</li><li>新增“4.1.2.1.14 AT+BLECLNADV DATFLT 清除 BLE 扫描过滤参数”章节内容。</li></ul>
09	2024-08-05	<ul style="list-style-type: none"><li>更新“2.2.1 查询版本信息”章节内容。</li></ul>

文档版本	发布日期	修改说明
		<ul style="list-style-type: none"><li>新增“4.2.2.21 设置 SLE 连接传输单个数据包长度”章节内容。</li><li>新增“4.2.2.22 设置 SLE 连接 mcs 传输特性”章节内容。</li></ul>
08	2024-06-07	新增“2.2.7 查看当前芯片温度”章节内容。
07	2024-05-24	更新“2.2.1 查询版本信息”章节内容。
06	2024-04-26	更新“5.5 产测指令描述”章节内容。
05	2024-04-17	<ul style="list-style-type: none"><li>更新“4.2.2.15 建立 SLE 连接”章节内容。</li><li>更新“4.2.2.38 服务端向客户端通过 uuid 发送通知”章节内容。</li><li>更新“5.4.8 配置 Wi-Fi 的 RTS 模式”章节内容。</li></ul>
04	2024-04-09	更新“4.2 SLE”章节内容。
03	2024-02-06	更新“5.5 产测指令描述”章节内容。
02	2024-01-19	<ul style="list-style-type: none"><li>新增“4.2 SLE”章节内容。</li><li>更新“5.1 调测相关指令一览表”章节内容。</li><li>更新“5.2.15 配置单音功能”章节内容。</li></ul>
01	2023-12-11	<p>第一次正式版本发布。</p> <ul style="list-style-type: none"><li>更新“3.5.1 设置 ARP OFFLOAD 特性开关”、“3.5.2 设置 ARP OFFLOAD 功能发送 Free ARP 周期”和“3.5.3 设置 WoW 特性唤醒源”章节内容。</li><li>更新“3.5.11 设置 TWT SET_UP 会话时 TWT 相关参数”、“3.5.12 设置 TWT TEAR_DOWN 会话时 TWT 相关参数”章节内容。</li></ul>

文档版本	发布日期	修改说明
		<ul style="list-style-type: none"><li>更新“3.10.1 设置 SDP 特性使能开关”章节内容。</li><li>更新“3.10.4 设置 SDP 发送报文参数”章节内容。</li><li>更新“3.16.1 设置 SR 功能开关”章节内容。</li><li>更新“5.4.4 配置 Wi-Fi 固定速率参数”章节内容。</li></ul>
00B03	2023-12-01	更新“2.2 通用指令描述”小节中失败指令描述。
00B02	2023-11-23	<ul style="list-style-type: none"><li>更新“3.2 报文转发指令描述”章节内容。</li><li>更新“3.5.11 设置 TWT SET_UP 会话时 TWT 相关参数”章节内容。</li></ul>
00B01	2023-11-07	第一次临时版本发布。

## 目 录

前言 .....	i
<b>1 指令说明.....</b>	<b>1</b>
1.1 指令简介.....	1
1.2 注意事项.....	1
<b>2 通用指令介绍 .....</b>	<b>2</b>
2.1 通用指令一览表.....	2
2.2 通用指令描述 .....	3
2.2.1 查询版本信息 .....	3
2.2.2 设置国家码 .....	3
2.2.3 查询国家码 .....	3
2.2.4 设置管制域发送功率 .....	4
2.2.5 查看管制域发送功率 .....	4
2.2.6 设置是否根据关联 AP 更新国家码信息功能开关.....	4
2.2.7 查看当前芯片温度 .....	5
2.2.8 暂停模组.....	5
2.2.9 恢复模组.....	6
2.2.10 查询芯片与总线类型 .....	6
<b>3 Wi-Fi 特性指令介绍.....</b>	<b>7</b>
3.1 Wi-Fi 特性指令一览表 .....	8
3.2 报文转发指令描述 .....	11
3.2.1 设置软件重传次数 .....	11
3.2.2 设置数据帧分片门限 .....	11
3.2.3 查询数据帧分片门限 .....	11

3.2.4 设置报文强制 TID 队列号 .....	12
3.2.5 设置开启/关闭 A-MSDU 发送能力 .....	12
3.3 漫游相关指令描述 .....	13
3.3.1 设置触发漫游 RSSI 阈值 .....	13
3.3.2 设置发送 BSS Transition Query 报文参数 .....	13
3.4 黑白名单指令描述 .....	14
3.4.1 将 STA 添加到黑/白名单 (根据 MAC 地址) .....	14
3.4.2 将 STA 从黑/白名单中删除 (根据 MAC 地址) .....	14
3.4.3 设置黑/白名单的使能开关 .....	15
3.4.4 查看所有的黑/白名单列表 .....	15
3.5 低功耗指令描述 .....	16
3.5.1 设置 ARP OFFLOAD 特性开关 .....	16
3.5.2 设置 ARP OFFLOAD 功能发送 Free ARP 周期 .....	16
3.5.3 设置 WoW 特性唤醒源 .....	17
3.5.4 设置支持唤醒 WoW 特性的报文格式 .....	17
3.5.5 查看输出到日志的 WoW 唤醒原因 .....	18
3.5.6 设置 WoW 特性开关 .....	18
3.5.7 设置 PM 节能参数 .....	19
3.5.8 设置协议低功耗开关 .....	19
3.5.9 查询当前节能状态机状态 .....	20
3.5.10 设置 PS-POLL 能力 .....	20
3.5.11 设置 TWT SET_UP 会话时 TWT 相关参数 .....	20
3.5.12 设置 TWT TEAR_DOWN 会话时 TWT 相关参数 .....	21
3.5.13 设置 UAPSD 参数 .....	22
3.6 EDCA 指令描述 .....	22
3.6.1 配置 EDCA 参数 .....	22
3.6.2 查询 EDCA 参数 .....	23
3.7 APF 指令描述 .....	24
3.7.1 设置 APF 过滤规则 .....	24
3.7.2 查询 APF 过滤规则 .....	24
3.7.3 设置 APF 过滤开关 .....	24

3.7.4 设置暗屏开关 .....	25
3.8 Aware 指令描述 .....	25
3.8.1 设置 Wi-Fi Aware 固定发送功率 .....	25
3.8.2 设置 Wi-Fi Aware 为自动调整功率模式 .....	26
3.9 任意帧指令描述 .....	26
3.9.1 设置发送的任意帧内容 .....	26
3.10 Wi-Fi SDP 指令描述 .....	27
3.10.1 设置 SDP 特性使能开关 .....	27
3.10.2 设置 SDP 服务参数 .....	27
3.10.3 取消 SDP 服务 .....	28
3.10.4 设置 SDP 发送报文参数 .....	28
3.11 组播转单播指令描述 .....	29
3.11.1 设置组播转单播开关 .....	29
3.11.2 查看组播转单播用户表项 .....	29
3.11.3 设置组播转单播的黑名单 .....	30
3.11.4 设置组播转单播发送的 IGMP 报文参数 .....	30
3.12 自动调频指令描述 .....	31
3.12.1 设置自动调频使能开关 .....	31
3.12.2 设置自动调频的流量阈值 .....	31
3.12.3 设置自动调频频率档次 .....	32
3.12.4 查询自动调配频率档次 .....	32
3.13 Wi-Fi CSI 指令描述 .....	33
3.13.1 设置 Wi-Fi CSI 特性开关 .....	33
3.13.2 设置 Wi-Fi CSI 特性配置参数 .....	33
3.13.3 查询指定用户 CSI 参数配置 .....	34
3.13.4 设置存储 CSI 信息 Buffer 的个数和大小 .....	35
3.14 Wi-Fi CSA 指令描述 .....	35
3.14.1 设置 CSA 触发信道切换参数 .....	35
3.15 动态窄带指令描述 .....	36
3.15.1 设置 STA 动态窄带功能开关 .....	36
3.16 Wi-Fi 6 特性指令描述 .....	36



3.16.1 设置 SR 功能开关 .....	36
3.16.2 查询 SR 功能的结果信息 .....	37
3.16.3 设置 SR 冲突检测功能开关 .....	37
3.17 STA 信道评分指令描述 .....	38
<b>4 BLE&amp;SLE 模块 AT 指令 .....</b>	<b>39</b>
4.1 BLE .....	39
4.1.1 BLE AT 指令一览表 .....	39
4.1.1.1 gap 模块 AT 命令 .....	39
4.1.1.2 gatts 模块 AT 命令 .....	40
4.1.1.3 gattc 模块 AT 命令 .....	41
4.1.2 BLE AT 指令描述 .....	42
4.1.2.1 gap 模块 AT 命令 .....	42
4.1.2.1.1 AT+BLEENABLE 使能 ble 协议栈 .....	42
4.1.2.1.2 AT+BLEDISABLE 关闭 ble 协议栈 .....	43
4.1.2.1.3 AT+BLESETADDR 设置本地设备地址 .....	43
4.1.2.1.4 AT+BLEGETADDR 获取本地设备地址 .....	43
4.1.2.1.5 AT+BLESETNAME 设置本地设备名称 .....	44
4.1.2.1.6 AT+BLEGETNAME 获取本地设备名称 .....	44
4.1.2.1.7 AT+BLESETAPPEARANCE 设置本地设备外观 .....	44
4.1.2.1.8 AT+BLESETADVDATA 设置 BLE 广播数据 .....	45
4.1.2.1.9 AT+BLESETADVPAR 设置广播数据参数 .....	45
4.1.2.1.10 AT+BLESTARTADV 开始发送 BLE 广播 .....	46
4.1.2.1.11 AT+BLESTOPADV 停止发送 BLE 广播 .....	47
4.1.2.1.12 AT+BLESETSCANPAR 设置 BLE 扫描参数 .....	47
4.1.2.1.13 AT+BLESETADVDFLT 设置 BLE 扫描过滤参数 .....	48
4.1.2.1.14 AT+BLECLNADVDFLT 清除 BLE 扫描过滤参数 .....	48
4.1.2.1.15 AT+BLESTARTSCAN 启动 BLE 扫描 .....	49
4.1.2.1.16 AT+BLESTOPSCAN 停止 BLE 扫描 .....	49
4.1.2.1.17 AT+BLEPAIR 与对端设备发起配对 .....	49
4.1.2.1.18 AT+BLEGETPAIREDNUM 获取 BLE 设备配对设备数量 .....	50
4.1.2.1.19 AT+BLEGETPAIREDDEV 获取 BLE 设备配对设备 .....	50

4.1.2.1.20 AT+BLEGETPAIREDSTA 获取 BLE 设备配对状态.....	50
4.1.2.1.21 AT+BLEUNPAIR 取消配对.....	51
4.1.2.1.22 AT+BLEUNPAIRALL 取消所有配对.....	51
4.1.2.1.23 AT+BLECONNPARDUPD 更新连接参数.....	51
4.1.2.1.24 AT+BLECONN 与 BLE 设备连接.....	52
4.1.2.1.25 AT+BLEDISCONN 与 BLE 设备断开连接.....	53
4.1.2.1.26 AT+BLEGAPREGCBK 注册 BLE 回调函数.....	53
4.1.2.1.27 AT+BLESETPHY 设置 BLE PHY.....	53
4.1.2.2 gatts 模块 AT 命令.....	54
4.1.2.2.1 AT+GATTSREGSRV 创建一个 GATT server.....	54
4.1.2.2.2 AT+GATTSUNREG 删除 GATT server, 释放资源.....	55
4.1.2.2.3 AT+GATTSADDSERV 添加一个 GATT 服务.....	55
4.1.2.2.4 AT+GATTSSYNCAADDSERV 添加一个 GATT 服务 (同步).....	55
4.1.2.2.5 AT+GATTSADDCHAR 为 GATT 服务添加一个特征.....	56
4.1.2.2.6 AT+GATTSSYNCAADDSERV 为 GATT 服务添加一个特征 (同步).....	57
4.1.2.2.7 AT+GATTSADDDESCRIPTOR 为最新的特征添加一个描述符.....	58
4.1.2.2.8 AT+GATTSSYNCAADDSERV 为最新的特征添加一个描述符 (同步).....	59
4.1.2.2.9 AT+GATTSSTARTSERV 启动指定的 GATT 服务.....	59
4.1.2.2.10 AT+GATTSDELALLSERV 删除指定 server 上的所有服务.....	60
4.1.2.2.11 AT+GATTSSENDRESP 发送响应.....	60
4.1.2.2.12 AT+GATTSSENDNTFY 发送通知或指示.....	61
4.1.2.2.13 AT+GATTSSENDNTFYBYUUID 根据 uuid 发送通知或指示.....	61
4.1.2.2.14 AT+GATTSREGCBK 注册 GATT 服务端回调函数.....	62
4.1.2.2.15 AT+GATTSSETMTU 在连接之前设置 server rx mtu.....	62
4.1.2.3 gattc 模块 AT 命令.....	62
4.1.2.3.1 AT+GATTCREG 创建一个 GATT client.....	62
4.1.2.3.2 AT+GATTCUNREG 删除 GATT client, 释放资源.....	63
4.1.2.3.3 AT+GATTCFINDSERV 发现服务.....	63
4.1.2.3.4 AT+GATTCFINDCHAR 发现特征.....	63
4.1.2.3.5 AT+GATTCFINDDESCRIPTOR 发现描述符.....	64
4.1.2.3.6 AT+GATTCREADBYHDL 读取 by hdl.....	64

4.1.2.3.7 AT+GATTCREADBYUUID 读取 by_uuid .....	65
4.1.2.3.8 AT+GATTCWRITEREQ 写 by hdl req.....	65
4.1.2.3.9 AT+GATTCWRITECMD 写 by hdl cmd.....	66
4.1.2.3.10 AT+GATTCEXCHMTU 交换 MTU 请求 .....	66
4.1.2.3.11 AT+GATTCREGCBK 注册 GATT 客户端回调函数.....	66
4.2 SLE .....	67
4.2.1 SLE AT 指令一览表.....	67
4.2.2 SLE AT 指令描述 .....	69
4.2.2.1 SLE 使能.....	70
4.2.2.2 SLE 去使能.....	70
4.2.2.3 设置 SLE 广播参数.....	71
4.2.2.4 删除广播参数 .....	72
4.2.2.5 设置 SLE 广播数据.....	72
4.2.2.6 起 SLE 广播.....	72
4.2.2.7 停 SLE 广播.....	73
4.2.2.8 设置扫描参数 .....	73
4.2.2.9 使能扫描.....	74
4.2.2.10 关闭扫描.....	74
4.2.2.11 设置本端名称.....	74
4.2.2.12 获取本端名称 .....	75
4.2.2.13 设置本端地址 .....	75
4.2.2.14 获取本端地址 .....	75
4.2.2.15 建立 SLE 连接 .....	76
4.2.2.16 星闪逻辑链路更新参数.....	76
4.2.2.17 星闪读取远端 rssi.....	77
4.2.2.18 断开 SLE 连接.....	77
4.2.2.19 设置 SLE PHY.....	78
4.2.2.20 设置 SLE 默认连接参数.....	78
4.2.2.21 设置 SLE 连接传输单个数据包长度 .....	79
4.2.2.22 设置 SLE 连接 mcs 传输特性.....	80
4.2.2.23 进行加密配对 .....	80
4.2.2.24 移除加密配对 .....	81

4.2.2.25 获取配对设备数目 .....	81
4.2.2.26 获取配对设备 .....	81
4.2.2.27 获取设备配对状态 .....	82
4.2.2.28 获取绑定设备 .....	82
4.2.2.29 注册服务端 .....	82
4.2.2.30 去注册服务端 .....	83
4.2.2.31 添加服务 .....	83
4.2.2.32 添加服务同步 .....	84
4.2.2.33 添加属性 .....	84
4.2.2.34 添加属性同步 .....	85
4.2.2.35 添加属性描述符 .....	86
4.2.2.36 添加属性描述符同步 .....	87
4.2.2.37 服务端向客户端发送通知 .....	89
4.2.2.38 服务端向客户端通过 uuid 发送通知 .....	89
4.2.2.39 服务端发送响应 .....	90
4.2.2.40 服务端注册回调 .....	92
4.2.2.41 start service .....	92
4.2.2.42 注册 SSAPC 回调函数 .....	92
4.2.2.43 发现 service .....	93
4.2.2.44 通过句柄发现 service .....	94
4.2.2.45 客户端向服务端写入数据 .....	95
4.2.2.46 客户端向服务端发送写请求 .....	95
4.2.2.47 客户端发起信息交换 .....	96
4.2.2.48 设置服务端信息 .....	96
4.2.2.49 客户端通过 uuid 发送读请求 .....	97
4.2.2.50 客户端读取服务端属性数据 .....	97
4.2.2.51 SLE 断开所有连接 .....	98
4.2.2.52 SLE ssap 数传流控参数设置 .....	98
4.2.2.53 SLE ssap 客户端 write_cmd 数传开始指令 .....	99
4.2.2.54 SLE ssap 服务端 indicate&notify 数传开始指令 .....	100
4.2.2.55 SLE ssap 流控客户端清理接收包数量 .....	101
4.2.2.56 SLE ssap 流控服务端清理接收数据包数量 .....	101

4.2.2.57 SLE ssap 流控回调注册 .....	101
<b>5 调测相关指令介绍 .....</b>	<b>102</b>
5.1 调测相关指令一览表 .....	102
5.2 维测指令描述 .....	106
5.2.1 设置时延统计功能开关 .....	106
5.2.2 查询时延统计结果 .....	107
5.2.3 清除时延统计结果 .....	107
5.2.4 查询 vap 信道、频宽信息 .....	107
5.2.5 查询 Devcice 侧 Netbuf 内存分配情况 .....	108
5.2.6 查询连接用户的相关统计信息（包括连接失败原因） .....	108
5.2.7 查询驱动扫描结果 .....	109
5.2.8 设置连接用户的速率统计功能开关 .....	109
5.2.9 查询连接用户的统计信息 .....	109
5.2.10 查询当前 TID 队列信息 .....	110
5.2.11 设置多业务维测日志功能开关 .....	110
5.2.12 查询当前开启维测功能业务 .....	111
5.2.13 设置业务输出功率 .....	111
5.2.14 创建新的 vap 接口 .....	112
5.2.15 配置单音功能 .....	112
5.3 空口调试指令描述 .....	113
5.3.1 设置 Wi-Fi Monitor 模式 .....	113
5.3.2 设置 Sniffer 抓包文件大小 .....	114
5.3.3 查询空口实时情况，包括发包占比，空闲占比，干扰占比情况 .....	114
5.3.4 设置对外交互日志记录功能开关 .....	115
5.3.5 设置帧上报功能总开关 .....	115
5.3.6 设置 Beacon 帧上报开关 .....	115
5.3.7 设置 VIP 帧上报开关 .....	116
5.3.8 设置数据帧或管理帧上报开关 .....	116
5.3.9 设置描述符上报开关 .....	117
5.4 算法配置指令描述 .....	118
5.4.1 查询用户发送速率（根据 mac 地址） .....	118

5.4.2 配置 RTS 阈值.....	118
5.4.3 配置 Wi-Fi 发送速率模式.....	119
5.4.4 配置 Wi-Fi 固定速率参数.....	119
5.4.5 配置聚合自适应算法最大聚合个数.....	120
5.4.6 查询 Wi-Fi 干扰类型 .....	120
5.4.7 配置固定功率码.....	120
5.4.8 配置 Wi-Fi 的 RTS 模式.....	121
5.4.9 配置干扰场景优化模式.....	121
5.5 产测指令描述 .....	122
<b>6 新增 CCPRIV 命令方法 .....</b>	<b>153</b>

# 1 指令说明

## 1.1 指令简介

## 1.2 注意事项

## 1.1 指令简介

CCPRIV 命令是 WS73 自研的 Wi-Fi 命令集，可在应用层执行 CCPRIV 命令对 Wi-Fi 驱动进行调试，其整体功能类似于 iwpriv 工具。

## 1.2 注意事项

- 串口通信默认：波特率为 115200、8 个数据位、1 个停止位、无校验。
- 双引号表示字符串数据"string"，例如：echo "wlan0 get\_version" > /sys/ccsys/ccpriv。

## 2 通用指令介绍

### 2.1 通用指令一览表

### 2.2 通用指令描述

### 2.1 通用指令一览表

指令	描述
get_version	查询版本信息。
setcountry	设置国家码。
getcountry	查询国家码。
set_regdomain_pwr_p	设置管制域发送功率。
get_regdomain_pwr	查看管制域发送功率。
set_rd_by_ie_switch	设置是否根据关联 AP 更新国家码信息功能开关。
get_temp	查询当前芯片温度。
suspend_system	模组下电或者复位，驱动停止与业务软件和模组交互。
resume_system	模组上电，重新下载固件并且驱动恢复到暂停前状态。
get_chip_type	查询芯片与总线类型。



## 2.2 通用指令描述

### 2.2.1 查询版本信息

格式	echo "get_version" > /sys/ccsys/ccpriv
参数说明	-
示例	echo "get_version" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 2.2.2 设置国家码

格式	echo "\$vap_name setcountry \$value " > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等。</li><li>\$value: 需要设置的国家码。</li></ul>
示例	echo "wlan0 setcountry CN" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 2.2.3 查询国家码

格式	echo "\$vap_name getcountry" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等。
示例	echo "wlan0 getcountry" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li></ul>

	<ul style="list-style-type: none"><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 2.2.4 设置管制域发送功率

格式	echo "\$vap_name set_regdomain_pwr_p \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name：需要维测的 vap 名字，通常用 wlan0 等。</li><li>\$value：设置发送功率大小，可配范围 1~100，单位为 dB。</li></ul>
示例	echo "wlan0 set_regdomain_pwr_p 50" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	<ul style="list-style-type: none"><li>需要创建 wpa 前进行配置。</li><li>调整管制域发送功率（能突破管制域限制）。</li></ul>

## 2.2.5 查看管制域发送功率

格式	echo "\$vap_name get_regdomain_pwr" > /sys/ccsys/ccpriv
参数说明	\$vap_name：需要维测的 vap 名字，通常用 wlan0 等。
示例	echo "wlan0 get_regdomain_pwr" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 2.2.6 设置是否根据关联 AP 更新国家码信息功能开关

格式	echo "\$vap_name set_rd_by_ie_switch \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name：需要维测的 vap 名字，通常用 wlan0 等。</li></ul>

	<ul style="list-style-type: none"><li>• \$val: 设置是否根据关联 ap 更新国家码信息功能开关。</li><li>• 0: 关闭;</li><li>• 1: 开启。</li></ul>
示例	echo "wlan0 set_rd_by_ie_switch 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 2.2.7 查看当前芯片温度

格式	echo "get_temp" > /sys/ccsys/ccpriv
参数说明	NA
示例	echo "get_temp" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 2.2.8 暂停模组

格式	echo "suspend_system" > /sys/ccsys/ccpriv
参数说明	-
示例	echo "suspend_system" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• pm_suspend: success</li></ul>
注意	<ul style="list-style-type: none"><li>• 主控有 GPIO 控制模组, 会对模组下电; 无 GPIO 控制模组 (配置 -1), 会对模组进行复位。</li><li>• 有 WiFi 业务时, 会暂停所有 WiFi 业务, 命令下发后需要等待 3s 等待业务停止, WiFi 停止以后 wlan0 接口状态保持不变。</li><li>• 应用场景: 主控不下电、不休眠, 需要停止模组节省功耗或者热插</li></ul>

拔之前需要下发该命令。

### 2.2.9 恢复模组

格式	echo "resume_system" > /sys/ccsys/ccpriv
参数说明	-
示例	echo "resume_system" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>pm_resume: success</li></ul>
注意	<ul style="list-style-type: none"><li>模组上电重新下载固件，WiFi 业务恢复到 suspend_system 之前状态。</li><li>通道需要重新连接并重新下载固件、WiFi 恢复连接，该命令完成需要等待 3s，等待 DHCP 重新获取地址，整体时间一般需要 5s 左右。</li></ul>

### 2.2.10 查询芯片与总线类型

格式	echo "get_chip_type" > /sys/ccsys/ccpriv
参数说明	-
示例	echo "get_chip_type" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>WS73S, SDIO 总线: Chip WS73S, bus SDIO.</li><li>WS73U, USB 总线: Chip WS73U, bus USB.</li><li>WS73E, USB 总线: Chip WS73E, bus USB.</li><li>WS73E, UART 总线: Chip WS73E, bus UART.</li></ul>
注意	-

# 3

## Wi-Fi 特性指令介绍

- 3.1 Wi-Fi 特性指令一览表
- 3.2 报文转发指令描述
- 3.3 漫游相关指令描述
- 3.4 黑白名单指令描述
- 3.5 低功耗指令描述
- 3.6 EDCA 指令描述
- 3.7 APF 指令描述
- 3.8 Aware 指令描述
- 3.9 任意帧指令描述
- 3.10 Wi-Fi SDP 指令描述
- 3.11 组播转单播指令描述
- 3.12 自动调频指令描述
- 3.13 Wi-Fi CSI 指令描述
- 3.14 Wi-Fi CSA 指令描述
- 3.15 动态窄带指令描述
- 3.16 Wi-Fi 6 特性指令描述
- 3.17 STA 信道评分指令描述

### 3.1 Wi-Fi 特性指令一览表

特性	指令	描述
报文转发	set_soft_retry_num	设置软件重传次数。
	frag_threshold	设置数据帧分片门限。
	show_frag_threshold	查询数据帧分片门限。
	set_ac_mode	设置报文强制 TID 队列号。
	amsdu_tx_on	设置开启/关闭 A-MSDU 发送能力。
漫游	roam_cfg	设置触发漫游 RSSI 阈值。
	11v_tx_query	设置发送 BSS Transition Query 报文参数。
黑白名单	blacklist_add	将 STA 添加到黑/白名单（根据 MAC 地址）。
	blacklist_del	将 STA 从黑/白名单中删除（根据 MAC 地址）。
	blacklist_mode	设置黑/白名单的使能开关。
	blacklist_show	查看所有的黑/白名单列表。
低功耗	arp_offload_enable	设置 ARP OFFLOAD 特性开关。
	arp_offload_free_arp_interval	设置 ARP OFFLOAD 功能发送 Free ARP 周期。
	set_dhcpoffload_info	设置 DHCP OFFLOAD IP 地址。
	set_dhcpoffload_enable	设置 DHCP OFFLOAD 特性开关。
	wow_event	设置 WoW 特性唤醒源。
	wow_pattern	设置支持唤醒 WoW 特性的报文格式。
	wow_show_wakeup_reason	查看输出到日志的 WoW 唤醒原因。

特性	指令	描述
		因。
	wow_sleep	设置 WoW 特性开关。
	set_sta_pm	设置协议低功耗开关。
	set_psm_para	设置 PM 节能参数。
	get_sta_ps_stat	查询当前节能状态机状态（打印至 HSO）。
	set_ps_mode	设置 PS-POLL 能力。
	twi_setup_req	设置 TWT SET_UP 会话时 TWT 相关参数。
	twi_tearardown_req	设置 TWT TEAR_DOWN 会话时 TWT 相关参数。
	set_uapsd_para	设置 UAPSD 参数。
EDCA	set_edca_params	设置配置 EDCA 参数。
	get_edca_params	查询 EDCA 参数。
Wi-Fi APF	set_apf_list	设置 APF 过滤规则。
	get_apf_list	查询 APF 过滤规则。
	force_stop_apf	设置 APF 过滤开关。
	suspend_mode	设置暗屏开关。
Wi-Fi Aware	adjust_tx_power	设置 Wi-Fi Aware 固定发送功率。
	restore_tx_power	设置 Wi-Fi Aware 为自动调整功率模式。
任意帧	send_custom_pkt	设置发送的任意帧内容。
Wi-Fi SDP	sdp_enable	设置 SDP 特性使能开关。
	sdp_start_subscribe	设置 SDP 服务参数。
	sdp_cancle_subscribe	取消 SDP 服务。

特性	指令	描述
	sdp_send_data	设置 SDP 发送参数。
组播转单播	m2u_snoop_enable	设置组播转单播开关。
	m2u_snoop_list	查看组播转单播用户表项。
	m2u_snoop_deny_table	设置组播转单播的黑名单。
	m2u_snoop_send_igmp	设置组播转单播发送的 IGMP 报文参数。
自动调配	set_device_freq_enable	设置自动调频使能开关。
	set_device_flow_threshold	设置自动调频的流量阈值。
	set_device_freq_value	设置自动调频频率档次。
	get_device_freq_value	查询自动调配频率档次。
Wi-Fi CSI	csi_switch	设置 Wi-Fi CSI 特性开关。
	csi_set_config	设置 Wi-Fi CSI 特性配置参数。
	csi_get_config	查询指定用户 CSI 参数配置。
	csi_set_buffer	设置存储 CSI 信息 Buffer 的个数和大小。
Wi-Fi CSA	csa_ctrl	设置 CSA 触发信道切换参数。
动态窄带配置	set_sta_dnb_on	设置 STA 动态窄带功能开关。
Wi-Fi 11ax 特性配置	sr_en	设置 SR 功能开关。
	get_sr_info	查询 SR 功能的结果信息。
	collision_en	设置 SR 冲突检测功能开关。
STA 信道评分	channel_scoring	获取短时间内全信道的繁忙度。



## 3.2 报文转发指令描述

### 3.2.1 设置软件重传次数

格式	echo "\$vap_name set_soft_retry_num \$data_retry_times \$mgmt_retry_times" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$data_retry_times: 数据帧重传次数, 未限制范围, 默认 10。</li><li>\$mgmt_retry_times: 管理帧重传次数, 未限制范围, 默认 3。</li></ul>
示例	echo "wlan0 set_soft_retry_num 10 3" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.2.2 设置数据帧分片门限

格式	echo "\$vap_name frag_threshold \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$value: 数据帧分片门限, 配置范围 256~2346。</li></ul>
示例	echo "wlan0 frag_threshold 1000" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.2.3 查询数据帧分片门限

格式	echo "\$vap_name show_frag_threshold" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。

示例	echo "wlan0 show_frag_threshold" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.2.4 设置报文强制 TID 队列号

格式	echo "\$vap_name set_ac_mode \$mode" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$mode: <ul style="list-style-type: none"> <li>VO: VO 队列;</li> <li>VI: VI 队列;</li> <li>BE: BE 队列;</li> <li>BK: BK 队列;</li> <li>close: 恢复走默认队列。</li> </ul> </li> </ul>
示例	echo "wlan0 set_ac_mode VO" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.2.5 设置开启/关闭 A-MSDU 发送能力

格式	echo "\$vap_name amsdu_tx_on \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$value: <ul style="list-style-type: none"> <li>0: 关闭 A-MSDU 发送;</li> <li>1: 开启 A-MSDU 发送。</li> </ul> </li> </ul>
示例	echo "wlan0 amsdu_tx_on 1" > /sys/ccsys/ccpriv

响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	在连接 Wi-Fi 前下发该命令。

## 3.3 漫游相关指令描述

### 3.3.1 设置触发漫游 RSSI 阈值

格式	echo "\$vap_name roam_cfg \$enable \$trigger_rssi_2g \$trigger_rssi_5g" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$enable: 设置强信号漫游开关。 0: 关闭, 1: 开启。</li><li>\$trigger_rssi_2g: 设置 2G 频段漫游触发 RSSI 阈值。</li><li>\$trigger_rssi_5g: 设置 5G 频段漫游触发 RSSI 阈值。</li></ul>
示例	echo "wlan0 roam_cfg 1 -75 -80" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	设置 5G 频段漫游触发 RSSI 阈值默认不生效。

### 3.3.2 设置发送 BSS Transition Query 报文参数

格式	echo "\$vap_name 11v_tx_query \$src_mac \$channel1 \$dst_mac \$channel2" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$src_mac: 设置 Query 报文源 MAC 地址。</li><li>\$channel1: 设置源 AP 的工作信道。</li><li>\$dst_mac: 设置 Query 报文目的 MAC 地址。</li></ul>

	<ul style="list-style-type: none"><li>• \$channel2: 设置目的 AP 的工作信道。</li></ul>
示例	<code>echo "wlan0 11v_tx_query aa:bb:cc:dd:ee:ff 1 ff:ee:dd:cc:bb:aa 10" &gt; /sys/ccsys/ccpriv</code>
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 3.4 黑白名单指令描述

### 3.4.1 将 STA 添加到黑/白名单 (根据 MAC 地址)

格式	<code>echo "\$vap_name blacklist_add \$mac" &gt; /sys/ccsys/ccpriv</code>
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$mac: 设置需要添加 STA 的 MAC 地址。</li></ul>
示例	<code>echo "wlan0 blacklist_add aa:bb:cc:dd:ee:ff" &gt; /sys/ccsys/ccpriv</code>
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	需要先使能黑/白名单开关, 否则会提示 param error code[100]。

### 3.4.2 将 STA 从黑/白名单中删除 (根据 MAC 地址)

格式	<code>echo "\$vap_name blacklist_del \$mac" &gt; /sys/ccsys/ccpriv</code>
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$mac: 设置需要添加 STA 的 MAC 地址。</li></ul>
示例	<code>echo "wlan0 blacklist_del \$mac" &gt; /sys/ccsys/ccpriv</code>
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>

注意	-
----	---

### 3.4.3 设置黑/白名单的使能开关

格式	echo "\$vap_name blacklist_mode \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$val: 设置黑/白名单开关。 0: 关闭该功能; 1: 开启黑名单; 2: 开启白名单。</li></ul>
示例	echo "wlan0 blacklist_mode 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.4.4 查看所有的黑/白名单列表

格式	echo "\$vap_name blacklist_show" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。
示例	echo "wlan0 blacklist_show" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 3.5 低功耗指令描述

### 3.5.1 设置 ARP OFFLOAD 特性开关

格式	echo "\$vap_name arp_offload_enable \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$val: 特性开关。 1: 开启 0: 关闭。</li></ul>
示例	echo "wlan0 arp_offload_enable 0" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	ARP OFFLOAD 功能默认跟随 WoW 功能打开, 关闭需在 WoW 功能打开前下发。

### 3.5.2 设置 ARP OFFLOAD 功能发送 Free ARP 周期

格式	echo "\$vap_name arp_offload_free_arp_interval \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$val: 发送 Free ARP 周期, 默认值为 2, 对应 2 个 null data 帧发送一次。</li></ul>
示例	echo "wlan0 arp_offload_free_arp_interval 2" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	需要使能 WoW 之前下发该命令。

### 3.5.3 设置 WoW 特性唤醒源

格式	echo "\$vap_name wow_event \$val"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$val: 共 4 个 bit 位。 bit0 配置 magic 报文唤醒; bit1 配置特性 tcp 报文唤醒; bit2 配置特性 udp 报文唤醒; bit3 配置去关联唤醒。 各 bit 位置 0 表示不支持此唤醒源, 置 1 表示支持此唤醒源。</li></ul>
示例	echo "wlan0 wow_event 0xf"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	需要使能 WoW 之前下发该命令。

### 3.5.4 设置支持唤醒 WoW 特性的报文格式

格式	echo "\$vap_name wow_pattern \$type \$index \$pattern" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$type: add: 添加一条自定义报文格式规则; del: 删除一条自定义报文格式规则; clr: 清除所有自定义格式。</li><li>• \$index: 与 type 类型 "add" 、 "del" 配合使用, 指定操作下标, 范围为 0~3。</li><li>• \$pattern: 自定义报文格式, "0x"开头的十六进制报文内容。</li></ul>
示例	echo "wlan0 wow_pattern add 0 0xaabb" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li></ul>

	<ul style="list-style-type: none"><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	需要使能 WoW 之前下发该命令。

### 3.5.5 查看输出到日志的 WoW 唤醒原因

格式	echo "\$vap_name wow_show_wakeup_reason \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$val: 设置是否将 WoW 唤醒原因输出到日志。 0: 不输出; 1: 输出, 默认为 1。</li></ul>
示例	echo "wlan0 wow_show_wakeup_reason 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.5.6 设置 WoW 特性开关

格式	echo "\$vap_name wow_sleep \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$val: 1: 使能 WoW 特性; 0: 去使能 WoW 特性。</li></ul>
示例	echo "wlan0 wow_sleep 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	WS73 芯片中 WoW 特性默认配置为 AUTO 模式, 无需手动开关。



### 3.5.7 设置 PM 节能参数

格式	echo "\$vap_name set_psm_para \$pm_timer \$pm_counter \$beacon_timeout \$multicast_timeout \$sleep_time \$tbtt_offset" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$pm_timer: 节能定时器超时时间, 范围 10ms~100ms。</li><li>• \$pm_counter: 节能定时器次数阈值, 范围 1~15。</li><li>• \$beacon_timeout: beacon 超时时间, 范围 10ms~100ms。</li><li>• \$multicast_timeout: 进入低功耗收广播帧超时时间, 范围 10ms~100ms, 建议与 pm_timer 一致。</li><li>• \$sleep_time: 最大休眠时间, 范围 33ms~4000ms。</li><li>• \$tbtt_offset: TBTT 上报时间偏移量, 范围 10ms~100ms。</li></ul>
示例	echo "wlan0 set_psm_para 20 10 30 20 100 10" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.5.8 设置协议低功耗开关

格式	echo "\$vap_name set_sta_pm \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$val: 设置协议低功耗模式开关。 0: 不进入协议低功耗; 1: 进入协议低功耗。</li></ul>
示例	echo "wlan0 set_sta_pm 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.5.9 查询当前节能状态机状态

格式	echo "\$vap_name get_sta_ps_stat" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。
示例	echo "wlan0 get_sta_ps_stat" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.5.10 设置 PS-POLL 能力

格式	echo "\$vap_name set_ps_mode \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$value: 设置 PS 模式。 0: 不进入 POWER_SAVE; 1: 采用 FAST_PS。</li></ul>
示例	echo "wlan0 set_ps_mode 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.5.11 设置 TWT SET\_UP 会话时 TWT 相关参数

格式	echo "\$vap_name twt_setup_req \$mac_addr \$setup_cmd \$flow_type \$flow_ID \$trigger \$tw \$interval_exponent \$min_duration \$interval_mantissa \$duration_unit" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$mac_addr: 对端 AP MAC 地址。</li><li>\$setup_cmd:</li></ul>

	<p>0: Request TWT; 1: Suggest TWT。</p> <ul style="list-style-type: none"> <li>\$flow_type:                     <p>0: 发送 ps-poll/qos null 通知 AP 唤醒; 1: 直接唤醒。</p> </li> <li>\$flow_ID: 0~7, 标识同一个请求 STA 与响应 STA 之间的唯一的 TWT request。</li> <li>\$trigger:                     <p>0: TWT SP 不需要 trigger frame or TRS; 1: TWT SP 至少要有有一个 trigger frame or TRS。</p> </li> <li>\$twl: 无效, 默认配置为 0。</li> <li>\$interval_exponent: 0~31。</li> <li>\$min_duration:                     <p>duration_unit 0: 0~63TU; duration_unit 1: 0~255TU。</p> </li> <li>\$interval_mantissa: 0~65535。</li> <li>\$duration_unit:                     <ul style="list-style-type: none"> <li>0: 256μs;</li> <li>1: 1TU (1024μs)。</li> </ul> </li> </ul>
示例	echo "wlan0 twt_setup_req aa:bb:cc:dd:ee:ff 1 0 0 0 50 10 20 90 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	需要对接支持 TWT 的 AP 测试

### 3.5.12 设置 TWT TEAR\_DOWN 会话时 TWT 相关参数

格式	echo "\$vap_name twt_tearardown_req \$mac_addr \$flow_ID" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> </ul>

	<ul style="list-style-type: none"> <li>• \$mac_addr: STA 连接 AP 的 MAC 地址。</li> <li>• \$flow_ID: 需要删除的 TWT 会话对应的 ID。</li> </ul>
示例	echo "wlan0 twt_tearardown_req aa:bb:cc:dd:ee:ff 0" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.5.13 设置 UAPSD 参数

格式	echo "\$vap_name set_uapsd_para \$switch \$lenth \$BE \$BK \$VI \$VO" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>• \$switch: 设置 UAPSD 开关。 0: 关闭; 1: 开启。</li> <li>• \$lenth: 节能队列最大长度。</li> <li>• \$BE \$BK \$VI \$VO: 各优先级开关。</li> </ul>
示例	echo "wlan0 set_uapsd_para 1 4 1 1 1 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 3.6 EDCA 指令描述

### 3.6.1 配置 EDCA 参数

格式	echo "\$vap_name set_edca_params \$para \$prio \$val" > /sys/ccsys/ccpriv
----	---

参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$para: 配置 EDCA BSS 广播参数, 可配置 EDCA BSS 广播参数, 包括 AIFS、CwMin、CwMax、TXOP; 可配置 EDCA 参数 MIB 值和寄存器值, 包括 qAIFS、qCwMin、qCwMax、qTXOP。</li> <li>\$prio: 配置\$para 对应的优先级, 可配置 0/1/2/3, 表示 BE/BK/VI/VO 队列。</li> <li>\$val: AIFS 配置范围: 0~15; CwMin 配置范围: 0~10; CwMax 配置范围: 0~10; TXOP 配置范围: 0~65535。</li> </ul>
示例	echo "wlan0 set_edca_params cwmin 1 3" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.6.2 查询 EDCA 参数

格式	echo "\$vap_name get_edca_params \$para \$prio" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$para: 查询 EDCA BSS 广播参数、MIB 值和寄存器值, 包括 AIFS、CwMin、CwMax、TXOP、qAIFS、qCwMin、qCwMax、qTXOP。</li> <li>\$prio: 配置\$para 对应的优先级, 可配置 0/1/2/3, 表示 BE/BK/VI/VO 队列。</li> </ul>
示例	echo "wlan0 get_edca_params cwmin 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 3.7 APF 指令描述

### 3.7.1 设置 APF 过滤规则

格式	echo "\$vap_name set_apf_list"> /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。
示例	echo "wlan0 set_apf_list"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	Android 系统下发该命令才会生效。

### 3.7.2 查询 APF 过滤规则

格式	echo "\$vap_name get_apf_list"> /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。
示例	echo "wlan0 get_apf_list"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	Android 系统下发该命令才会生效。

### 3.7.3 设置 APF 过滤开关

格式	echo "\$vap_name force_stop_apf \$val"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$val:<ul style="list-style-type: none"><li>1: 停止 APF 过滤;</li></ul></li></ul>

	0: 恢复 APF 过滤（暗屏模式下才生效）。
示例	echo "wlan0 force_stop_apf 1"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	Android 系统下发该命令才会生效。

### 3.7.4 设置暗屏开关

格式	echo "\$vap_name suspend_mode \$val"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$val:<ul style="list-style-type: none"><li>1: 使能暗屏, 开始 APF 过滤;</li><li>0: 停止暗屏, 停止 APF 过滤。</li></ul></li></ul>
示例	echo "wlan0 suspend_mode 0"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	Android 系统下发该命令才会生效。

## 3.8 Aware 指令描述

### 3.8.1 设置 Wi-Fi Aware 固定发送功率

格式	echo "\$vap_name adjust_tx_power \$ch \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$ch: 设置信道编号。</li><li>\$value: 设置 Wi-Fi Aware 交互过程中报文的功率, 可配置范围-70dBm~+15dBm。</li></ul>

示例	echo "wlan0 adjust_tx_power 1 -50" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.8.2 设置 Wi-Fi Aware 为自动调整功率模式

格式	echo "\$vap_name restore_tx_power \$ch" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$ch: 设置信道编号。</li></ul>
示例	echo "wlan0 restore_tx_power 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 3.9 任意帧指令描述

### 3.9.1 设置发送的任意帧内容

格式	echo "\$vap_name send_custom_pkt \$data" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>\$data: 帧内容, 长度范围限制 10~490Byte, 对应字符 20~980Byte。</li></ul>
示例	echo "wlan0 send_custom_pkt 08010000D45D64A4CCD05ced93a10503D45D64A4CCD0eff5600fef f5600018f0a07" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>



注意	发送任意帧，要求 vap 已关联。
----	-------------------

## 3.10 Wi-Fi SDP 指令描述

### 3.10.1 设置 SDP 特性使能开关

格式	echo "\$vap_name sdp_enable \$mode" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字，通常用 wlan0。</li><li>\$mode: 设置 SDP 使能类型。 0: 去使能; 1: 使能，但不主动设置信道切换定时器; (需要切换到 6 信道接收协议报文) 2: 使能并且设置信道切换定时器。(sta 已关联 ap 的场景使用)</li></ul>
示例	echo "wlan0 sdp_enable 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 3.10.2 设置 SDP 服务参数

格式	echo "\$vap_name sdp_start_subscribe \$service_name \$local_handle" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字，通常用 wlan0。</li><li>\$service_name: 设置对端设备 SDP 服务号，默认使用。</li><li>\$local_handle: 设置本端 SDP 服务句柄，可配范围 1~255，不可为 0。</li></ul>
示例	echo "wlan0 sdp_start_subscribe 1E87E9A99321 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li></ul>

	<ul style="list-style-type: none"> <li>失败：INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.10.3 取消 SDP 服务

格式	echo "\$vap_name sdp_cancle_subscribe \$local_handle" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name：需要维测的 vap 名字，通常用 wlan0。</li> <li>\$local_handle：设置本端 SDP 服务句柄，可配范围 1~255，不可为 0。</li> </ul>
示例	echo "wlan0 sdp_cancle_subscribe 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.10.4 设置 SDP 发送报文参数

格式	echo "\$vap_name sdp_send_data \$peer_mac \$local_handle \$peer_handle \$len \$data" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name：需要维测的 vap 名字，通常用 wlan0。</li> <li>\$peer_mac：设置对端 MAC 地址。</li> <li>\$local_handle：设置本端句柄。</li> <li>\$peer_handle：设置对端句柄。</li> <li>\$len：设置报文数据长度，可配范围 2~38，要求偶数输入。</li> <li>\$data：设置报文数据，十六进制字符串，要求为 \$len 的 2 倍。</li> </ul>
示例	依次执行： echo "wlan0 sdp_enable 1" > /sys/ccsys/ccpriv echo "wlan0 sdp_start_subscribe 1E87E9A99321 5" > /sys/ccsys/ccpriv echo "wlan0 sdp_send_data 002233445566 5 9 10 112233445566aabbccdd" > /sys/ccsys/ccpriv

响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 3.11 组播转单播指令描述

### 3.11.1 设置组播转单播开关

格式	echo "\$vap_name m2u_snoop_enable \$val \$mod" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name：需要维测的 vap 名字，通常用 wlan0。</li><li>\$val：<ul style="list-style-type: none"><li>0：关闭组播转单播功能；</li><li>1：开启。</li></ul></li><li>\$mod：配置组播转单播的方式。<ul style="list-style-type: none"><li>0：不会转单播；</li><li>1：使用 Tunnel 进行转发；</li><li>2：直接转发，默认为 2。</li></ul></li></ul>
示例	echo "wlan1 m2u_snoop_enable 1 2" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	wlan1 表示 AP 模式下执行该命令。

### 3.11.2 查看组播转单播用户表项

格式	echo "\$vap_name m2u_snoop_list" > /sys/ccsys/ccpriv
参数说明	\$vap_name：需要维测的 vap 名字，通常用 wlan0。
示例	echo "wlan1 m2u_snoop_list" > /sys/ccsys/ccpriv

响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.11.3 设置组播转单播的黑名单

格式	echo "\$vap_name m2u_snoop_deny_table \$cmd \$ip_addr" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$cmd: 设置添加、删除、查看黑名单。 add_ipv4: 添加 IPV4 地址; add_ipv6: 添加 IPV6 地址; del_ipv4: 删除 IPV4 地址; del_ipv6: 删除 IPV6 地址; clear: 清空黑名单; list: 查询黑名单列表。</li> <li>\$ip_addr: 设置 IP 地址。</li> </ul>
示例	echo 'wlan1 m2u_snoop_deny_table add_ipv4 192.168.1.10' > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.11.4 设置组播转单播发送的 IGMP 报文参数

格式	echo "\$vap_name m2u_snoop_send_igmp \$val1 \$val2 \$val3 \$val4" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val1: 设置所发报文的 TID 队列号, 可配范围 0~3。</li> <li>\$val2: 设置所发报文数量, 可配范围 1~32。</li> </ul>

	<ul style="list-style-type: none"> <li>\$val3: 设置所发报文长度, 可配范围 60~1514。</li> <li>\$val4: 设置所发报文的 TA MAC 地址。</li> </ul>
示例	echo "wlan1 m2u_snoop_send_igmp 0 10 1000 aa:bb:cc:dd:ee:ff" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 3.12 自动调频指令描述

### 3.12.1 设置自动调频使能开关

格式	echo "\$vap_name set_device_freq_enable \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val: 设置自动调频使能开关。 0: 关闭自动调频; 1: 开启自动调频。</li> </ul>
示例	echo "wlan0 set_device_freq_enable 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.12.2 设置自动调频的流量阈值

格式	echo "\$vap_name set_device_flow_threshold \$val1 \$val2 \$val3 \$val4 \$val5 \$val6" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val1、\$val2、\$val3 : 对流量进行的门限设置, 有三档 (低频/中频/高频), 单位为 pps, 默认值分别为: 0、1100、2200。</li> </ul>

	<ul style="list-style-type: none"> <li>• \$val4: 定时器的周期时间, 可配范围 20~1000ms, 默认值: 200。</li> <li>• \$val5: 流量统计有包的情况下, 连续多少次才会进行降频处理, 默认值: 20。</li> <li>• \$val6: 流量统计无包的情况下, 连续多少次才会进行降频处理, 默认值: 4。</li> </ul>
示例	echo "wlan0 set_device_flow_threshold 0 1100 2200 200 20 4">/sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.12.3 设置自动调频频率档次

格式	echo "\$vap_name set_device_freq_value \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>• \$val: 设置自动调频频率档次。 0: 低频 (主频的 25%) ; 1: 中频 (主频的 50%) ; 2: 高频 (主频的 100%) 。</li> </ul>
示例	echo "wlan0 set_device_freq_value 0" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.12.4 查询自动调配频率档次

格式	echo "\$vap_name get_device_freq_value \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>• \$val: 查询某频率档次的具体数值。</li> </ul>

	0: 低频; 1: 中频; 2: 高频。
示例	echo "wlan0 get_device_freq_value 0" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 3.13 Wi-Fi CSI 指令描述

### 3.13.1 设置 Wi-Fi CSI 特性开关

格式	echo "\$vap_name csi_switch \$val"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val: <ul style="list-style-type: none"> <li>0: 关闭 Wi-Fi CSI 功能;</li> <li>1: 开启 Wi-Fi CSI 功能。</li> </ul> </li> </ul>
示例	echo "wlan0 csi_switch 1"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.13.2 设置 Wi-Fi CSI 特性配置参数

格式	echo "\$vap_name csi_set_config \$val1 \$val2 \$val3 \$val4 \$val5 \$val6 \$val7 \$val8 \$val9"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val1: 配置 USER ID, 配置范围 0~3。</li> </ul>

	<ul style="list-style-type: none"> <li>• \$val2: 配置某个 USER 开启/关闭 Wi-Fi CSI 功能。</li> <li>• \$val3: 配置过滤 MAC 地址类型。 0: RA MAC; 1: TA MAC。</li> <li>• \$val4: 配置 MAC 地址。</li> <li>• \$val5: 配置帧类型过滤, 取值范围 0~7, 其中 bit0 管理帧, bit1 控制帧, bit2 数据帧。</li> <li>• \$val6: 配置子帧类型过滤开关。 0: 关闭; 1: 打开。</li> <li>• \$val7: 配置子帧类型过滤具体参数, 输入为 4 位二进制数 (如输入 1100, 表示过滤的子帧类型为 12)。</li> <li>• \$val8: 配置 PPDU Format 参数过滤, 取值范围 0~63。</li> <li>• \$val9: 配置 CSI 上报时间间隔, 单位 ms, 取值范围 0~4095。</li> </ul>
示例	echo "wlan0 csi_set_config 0 1 1 02:01:09:23:52:00 7 0 0 63 500" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	

### 3.13.3 查询指定用户 CSI 参数配置

格式	echo "\$vap_name csi_get_config \$val"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>• \$val: 设置用户 ID, 取值范围为 0~3。</li> </ul>
示例	echo "wlan0 csi_get_config 0"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-



### 3.13.4 设置存储 CSI 信息 Buffer 的个数和大小

格式	echo "\$vap_name csi_set_buffer \$val1 \$val2"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$val1: 设置 Buffer 个数, 可配范围 1~4。</li><li>• \$val2: 设置 Buffer 大小, 单位 KB, 可配范围 378~762。</li></ul>
示例	echo "wlan0 csi_set_buffer 1 400"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

## 3.14 Wi-Fi CSA 指令描述

### 3.14.1 设置 CSA 触发信道切换参数

格式	echo "\$vap_name csa_ctrl \$mode \$channel \$bandwidth \$count \$debug"> /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li><li>• \$mode: 设置信道切换模式。<ul style="list-style-type: none"><li>0: 允许 CSA 期间发包;</li><li>1: 禁止 CSA 期间发包。</li></ul></li><li>• \$channel: 设置目标信道, 可配范围 1~14。</li><li>• \$bandwidth: 设置目标频宽 (单位: Hz) 。<ul style="list-style-type: none"><li>0: 20M;</li><li>1: 40M Plus;</li><li>2: 40M Minus。</li></ul></li><li>• \$count: 设置信道切换计数, 表示\$count 个 beacon 周期后进行切换。</li></ul>

	<ul style="list-style-type: none"> <li>\$debug: 设置调试标记。</li> <li>0: 正常切信道。</li> </ul>
示例	echo "wlan0 csa_ctrl 0 1 0 5 0"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 3.15 动态窄带指令描述

### 3.15.1 设置 STA 动态窄带功能开关

格式	echo "\$vap_name set_sta_dnb_on \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val: 设置动态窄带功能开关。</li> <li>0: 关闭;</li> <li>1: 开启。</li> </ul>
示例	echo "wlan0 set_sta_dnb_on 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 3.16 Wi-Fi 6 特性指令描述

### 3.16.1 设置 SR 功能开关

格式	echo "\$vap_name sr_en \$val" > /sys/ccsys/ccpriv
----	---

参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val: 设置 SR 功能开关。 0: 关闭; 1: 开启。</li> </ul>
示例	echo "wlan0 sr_en 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	需连接 11AX 模式路由器。

### 3.16.2 查询 SR 功能的结果信息

格式	echo "\$vap_name get_sr_info \$val" > /sys/ccsys/ccpriv
参数说明	<p>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</p> <ul style="list-style-type: none"> <li>\$val: 查询 SR 的内容。 1: 查询 SR 的参数配置。 2: 查询 SR 的 pd_lv 和 tx_power。 4: 查询帧复用的数目。</li> </ul>
示例	echo "wlan0 get_sr_info 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 3.16.3 设置 SR 冲突检测功能开关

格式	echo "\$vap_name collision_en \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。</li> <li>\$val: 设置 SR 冲突检测功能开关。 0: 关闭;</li> </ul>

	1: 开启。
示例	echo "wlan0 collision_en 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	需连接 11AX 模式路由器。

### 3.17 STA 信道评分指令描述

格式	echo "\$vap_name channel_scoring" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0。
示例	echo "wlan0 channel_scoring" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	下发命令前 60s 之内, STA 至少触发过一次全信道扫描, 当前只支持 2.4GHz 信道。

# 4 BLE&SLE 模块 AT 指令

- 4.1 BLE
- 4.2 SLE

## 4.1 BLE

### 4.1.1 BLE AT 指令一览表

#### 4.1.1.1 gap 模块 AT 命令

指令	描述
AT+BLEENABLE	使能 BLE 协议栈
AT+BLEDISABLE	关闭 BLE 协议栈
AT+BLESETADDR=<参数>	设置本地设备地址
AT+BLEGETADDR	获取本地设备地址
AT+BLESETNAME=<参数>	设置本地设备名称
AT+BLEGETNAME	获取本地设备名称
AT+BLESETAPPEARANCE=<参数>	设置本地设备外观
AT+BLESETADVDATA=<参数>	设置 BLE 广播数据
AT+BLESETADVPAR=<参数>	设置 BLE 广播参数
AT+BLESTARTADV=<参数>	开始发送 BLE 广播

指令	描述
AT+BLESTOPADV=<参数>	停止发送 BLE 广播
AT+BLESETSCANPAR=<参数>	设置 BLE 扫描参数
AT+BLESETADVDFLT=<参数>	设置 BLE 扫描过滤参数
AT+BLECLNADVDFLT	清除 BLE 扫描过滤参数
AT+BLESTARTSCAN	启动 BLE 扫描
AT+BLESTOPSCAN	停止 BLE 扫描
AT+BLEPAIR=<参数>	与对端设备发起配对
AT+BLEGETPAIREDNUM	获取 BLE 设备配对设备数量
AT+BLEGETPAIREDDEV	获取 BLE 设备配对设备
AT+BLEGETPAIREDSTA=<参数>	获取 BLE 设备配对状态
AT+BLEUNPAIR=<参数>	取消配对
AT+BLEUNPAIRALL	取消所有配对
AT+BLECONNPARUPD=<参数>	连接参数更新
AT+BLECONN=<参数>	与 BLE 设备连接
AT+BLEDISCONN=<参数>	断开 BLE 设备连接
AT+BLEGAPREGCBK	注册 gap 回调函数

#### 4.1.1.2 gatts 模块 AT 命令

指令	描述
AT+GATTSREGSRV=<参数>	创建一个 GATT server
AT+GATTSUNREG=<参数>	删除 GATT server, 释放资源
AT+GATTSADDSERV=<参数>	添加一个 GATT 服务
AT+GATTSSYNCADDSERV=<参数>	添加一个 GATT 服务 (同步)

指令	描述
AT+GATTSADDCHAR=<参数>	为 GATT 服务添加一个特征
AT+GATTSSYNCAADDCHAR=<参数>	为 GATT 服务添加一个特征（同步）
AT+GATTSADDDESCR=<参数>	为最新的特征添加一个描述符
AT+GATTSSYNCAADDDESCR=<参数>	为最新的特征添加一个描述符（同步）
AT+GATTSSTARTSERV=<参数>	启动指定的 GATT 服务
AT+GATTSDELALLSERV=<参数>	删除指定 server 上的所有服务
AT+GATTSSENDRSP=<参数>	发送响应
AT+GATTSSNDNTFY=<参数>	发送通知或指示
AT+GATTSSNDNTFYBYUUID=<参数>	根据 uuid 发送通知或指示
AT+GATTSREGCBK	注册 GATT 服务端回调函数
AT+GATTSSETMTU=<参数>	在连接之前设置 server rx mtu

#### 4.1.1.3 gattc 模块 AT 命令

指令	描述
AT+GATTCREG=<参数>	创建一个 GATT client
AT+GATTCUNREG=<参数>	删除 GATT client, 释放资源
AT+GATTCFNDSERV=<参数>	发现所有服务(可 by uuid)
AT+GATTCFNDCHAR=<参数>	发现所有特征
AT+GATTCFNDDESCR=<参数>	发现所有描述符
AT+GATTCREADBYHDL=<参数>	读取 by hdl
AT+GATTCREADBYUUID=<参数>	读取 by_uuid
AT+GATTCWRITEREQ=<参数>	写 by hdl req
AT+GATTCWRITECMD=<参数>	写 by hdl cmd

指令	描述
AT+GATTCEXCHMTU=<参数>	交换 mtu 请求
AT+GATTCREGCBK	注册 GATT 客户端回调函数

## 4.1.2 BLE AT 指令描述

使用命令之前的配置：

- 加载平台和星闪驱动，komod 为驱动文件所在路径

```
insmod /komod/plat_soc.ko  
insmod /komod/ble_soc.ko
```

- 可执行程序增加执行权限，/bin 为可执行文件所在路径，如果和 sparklinkd 同时运行，需要放到不同目录。

```
cd /bin  
chmod +x bluetoothd  
chmod +x bluetoothctrl
```

- 进入视图

```
cd /bin  
bluetoothd &  
bluetoothctrl
```

### 4.1.2.1 gap 模块 AT 命令

#### 4.1.2.1.1 AT+BLEENABLE 使能 ble 协议栈

格式	AT+BLEENABLE
响应	打开 BLE 开关 OK
参数说明	-
示例	AT+BLEENABLE
注意	-



#### 4.1.2.1.2 AT+BLEDISABLE 关闭 ble 协议栈

格式	AT+BLEDISABLE
响应	关闭 BLE 开关 OK
参数说明	-
示例	AT+BLEDISABLE
注意	-

#### 4.1.2.1.3 AT+BLESETADDR 设置本地设备地址

格式	AT+BLESETADDR=<addr_type,addr>
响应	<ul style="list-style-type: none"><li>• 正确: OK</li><li>• 错误: ERROR</li></ul>
参数说明	<addr_type>: 蓝牙设备类型 <addr>: 蓝牙设备地址
示例	AT+BLESETADDR=0,0x112233445566
注意	设备类型取值范围为{0 (公共设备地址), 1 (随机设备地址), 2 (公共本端地址), 3 (随机静态本端地址)}, 设备地址为长度为 14 的字符串

#### 4.1.2.1.4 AT+BLEGETADDR 获取本地设备地址

格式	AT+BLEGETADDR
响应	<ul style="list-style-type: none"><li>• 正确: 本地设备地址</li><li>• 错误: ERROR</li></ul>
参数说明	-
示例	AT+BLEGETADDR
注意	-

#### 4.1.2.1.5 AT+BLESETNAME 设置本地设备名称

格式	AT+BLESETNAME=<len,name>
响应	<ul style="list-style-type: none"><li>• 正确：OK</li><li>• 错误：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>• &lt;len&gt;：本地设备名称长度。</li><li>• &lt;name&gt;：本地设备名称。</li></ul>
示例	AT+BLESETNAME=9,atcmdtest
注意	名称长度取值范围为 0~32，设备名称长度为 len-1 的字符串，名称最后默认存在'\0'

#### 4.1.2.1.6 AT+BLEGETNAME 获取本地设备名称

格式	AT+BLEGETNAME
响应	<ul style="list-style-type: none"><li>• 正确：本地设备名称</li><li>• 错误：ERROR</li></ul>
参数说明	-
示例	AT+BLEGETNAME
注意	-

#### 4.1.2.1.7 AT+BLESETAPPEARANCE 设置本地设备外观

格式	AT+BLESETAPPEARANCE=<appearance>
响应	<ul style="list-style-type: none"><li>• 正确：OK</li><li>• 错误：ERROR</li></ul>
参数说明	<appearance>：本地设备外观
示例	AT+BLESETAPPEARANCE=961
注意	参数值应为规定值，示例中 961 为键盘的外观值，可参考蓝牙规范设置。外观特征值应设置为由蓝牙 SIG 分配的 16 位数字之一，在

	《Core Specification Supplement, Part A, Data Types Specification》中进行了定义，也就是定义在核心规范补充部分中的数据类型规范中。
--	--

#### 4.1.2.1.8 AT+BLESETADVDATA 设置 BLE 广播数据

格式	AT+BLESETADVDATA=<adv_length,adv_data,scan_rsp_length,scan_rsp_data,adv_id>
响应	<ul style="list-style-type: none"> <li>正确：OK</li> <li>错误：ERROR</li> </ul>
参数说明	<p>&lt;adv_length&gt;：广播数据长度；</p> <p>&lt;adv_data&gt;：广播数据；</p> <p>&lt;scan_rsp_length&gt;：扫描返回数据长度；</p> <p>&lt;scan_rsp_data&gt;：扫描返回数据；</p> <p>&lt;adv_id&gt;：广播 id。</p>
示例	AT+BLESETADVDATA=6,0x112233445566,0,0,1
注意	广播数据长度单位为 Byte，所以广播数据应为长度两倍的字符串，扫描返回数据同理，广播 ID 取值范围为[1,255]

#### 4.1.2.1.9 AT+BLESETADVPAR 设置广播数据参数

格式	AT+BLESETADVPAR=<min_interval,max_interval,adv_type,own_addr,peer_addr_type,peer_addr,channel_map,adv_filter_policy,tx_power,duration,adv_id>
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	<p>&lt;min_interval&gt;：最小扫描间隔；取值范围[0x20, 0x4000],Time=N*0.625ms</p> <p>&lt;max_interval&gt;：最大扫描间隔；取值范围[0x20, 0x4000],Time=N*0.625ms</p> <p>&lt;adv_type&gt;：广播类型；</p> <p>&lt;own_addr&gt;：本端地址；</p>

	<p>&lt;peer_addr_type&gt;: 对端地址类型;</p> <p>&lt;peer_addr&gt;: 对端地址;</p> <p>&lt;channel_map&gt;: 信道; 取值范围为[0x01, 0x07]</p> <p>&lt;adv_filter_policy&gt;: 过滤策略;</p> <p>&lt;tx_power&gt;: 扫描功率;</p> <p>&lt;duration&gt;: 扫描周期; 仅取值为 0;</p> <p>&lt;adv_id&gt;: 广播 ID; 取值范围[1, 255]。</p>
示例	AT+BLESETADVPAR=48,48,0,0x112233445577,0,0x112233445566,7,0,1,0,1
注意	<ul style="list-style-type: none"><li>广播类型取值范围为 0~4:<ul style="list-style-type: none"><li>0: 可连接可扫描非定向广播(默认)</li><li>1: 可连接不可扫描高频定向广播</li><li>2: 不可连接可扫描非定向广播</li><li>3: 不可连接不可扫描非定向广播</li><li>4: 可连接不可扫描低频定向广播{</li></ul></li><li>过滤策略取值范围为 0~3:<ul style="list-style-type: none"><li>0: 处理所有设备的扫描和连接请求</li><li>1: 处理所有连接请求, 仅处理白名单的扫描请求</li><li>2: 处理所有扫描请求, 仅处理白名单的连接请求</li><li>3: 仅处理白名单中扫描请求和连接请求</li></ul></li><li>使用 BLEADDWHITELIST 指令添加白名单</li><li>扫描功率的范围为-127~+20, 单位为 dBm、</li></ul>

4.1.2.1.10 AT+BLESTARTADV 开始发送 BLE 广播

格式	AT+BLESTARTADV=<adv_id>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<adv_id>: 广播 id
示例	AT+BLESTARTADV=1

注意	-
----	---

#### 4.1.2.1.11 AT+BLESTOPADV 停止发送 BLE 广播

格式	AT+BLESTOPADV=<adv_id>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<adv_id>: 广播 id
示例	AT+BLESTOPADV=1
注意	-

#### 4.1.2.1.12 AT+BLESETSCANPAR 设置 BLE 扫描参数

格式	AT+BLESETSCANPAR=<scan_interval,scan_window,scan_type,scan_phy,scan_rsp_policy>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<p>&lt;scan_interval&gt;: 扫描间隔; 取值范围[0x04, 0x4000], Time=N×0.625ms</p> <p>&lt;scan_window&gt;: 扫描窗口; 取值范围[0x04, 0x4000], Time=N×0.625ms</p> <p>&lt;scan_type&gt;: 扫描类型; {0 (被动扫描), 1 (主动扫描)}</p> <p>&lt;scan_phy&gt;: 扫描 phy</p> <p>0x00: 无广播包</p> <p>0x01: 1M PHY</p> <p>0x02: 2M PHY</p> <p>0x03: 3M PHY</p> <p>&lt;scan_rsp_policy&gt;: 扫描过滤策略</p>
示例	AT+BLESETSCANPAR=0x48,0x48,0,1,0
注意	<ul style="list-style-type: none"><li>扫描过滤策略取值范围为 0~3:</li></ul>

	<p>0: 接收所有广播不接收目标地址不是本设备地址的定向广播</p> <p>1: 只接收白名单里设备的广播, 不接收目标地址不是本设备地址的定向广播</p> <p>2: 接收所有的非定向广播、地址是可解析私有地址的广播方发送的定向广播、发给该设备的定向广播</p> <p>3: 接收白名单中的所有非定向广播、地址是可解析私有地址的广播方发送的定向广播、发给该设备的定向广播</p> <ul style="list-style-type: none"><li>• 传统广播 phy 只有 1M, 扩展广播 phy 1 信道 1M 和 coded, 2 信道 1M、2M、coded。</li></ul>
--	---

4.1.2.1.13 AT+BLESETADVDATFLT 设置 BLE 扫描过滤参数

格式	AT+BLESETADVDATFLT=<scan_interval,scan_window,scan_type,scan_phy,scan_rsp_policy>
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: ERROR</li></ul>
参数说明	<p>&lt;filter_key_index&gt;: 过滤参数索引; 取值范围[0, 4]</p> <p>&lt;filter_key_len&gt;: 过滤参数数据长度; 取值范围[0, 128]</p> <p>&lt;filter_key&gt;: 过滤参数数据内容;</p> <p>&lt;filter_key_offset&gt;: 过滤数据在广播数据内的偏移位置;</p>
示例	AT+BLESETADVDATFLT=0,3,040506,0
注意	-

4.1.2.1.14 AT+BLECLNADVDATFLT 清除 BLE 扫描过滤参数

格式	AT+BLECLNADVDATFLT
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: ERROR</li></ul>
参数说明	-
示例	AT+BLECLNADVDATFLT

注意	-
----	---

#### 4.1.2.1.15 AT+BLESTARTSCAN 启动 BLE 扫描

格式	AT+BLESTARTSCAN
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+BLESTARTSCAN
注意	-

#### 4.1.2.1.16 AT+BLESTOPSCAN 停止 BLE 扫描

格式	AT+BLESTOPSCAN
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+BLESTOPSCAN
注意	-

#### 4.1.2.1.17 AT+BLEPAIR 与对端设备发起配对

格式	AT+BLEPAIR=<addr_type,addr>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<addr_type>：蓝牙设备类型 <addr>：蓝牙设备地址
示例	AT+BLEPAIR=0,0x112233445566

注意	设备类型取值范围为{0（公共设备地址）,1（随机设备地址）,2（公共本端地址）,3（随机静态本端地址）},设备地址为长度为 14 的字符串
----	---

#### 4.1.2.1.18 AT+BLEGETPAIREDNUM 获取 BLE 设备配对设备数量

格式	AT+BLEGETPAIREDNUM
响应	<ul style="list-style-type: none"><li>成功：配对设备数量</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+BLEGETPAIREDNUM
注意	-

#### 4.1.2.1.19 AT+BLEGETPAIREDDEV 获取 BLE 设备配对设备

格式	AT+BLEGETPAIREDDEV
响应	<ul style="list-style-type: none"><li>成功：配对设备地址</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+BLEGETPAIREDDEV
注意	-

#### 4.1.2.1.20 AT+BLEGETPAIREDSTA 获取 BLE 设备配对状态

格式	AT+BLEGETPAIREDSTA=<addr_type,addr>
响应	<ul style="list-style-type: none"><li>成功：BLE 设备配对状态</li><li>失败：ERROR</li></ul>
参数说明	<addr_type>：蓝牙设备类型 <addr>：蓝牙设备地址



示例	AT+BLEGETPAIREDSTA=0,0x112233445566
注意	设备类型取值范围为{0（公共设备地址）,1（随机设备地址）,2（公共本端地址）,3（随机静态本端地址）},设备地址为长度为 14 的字符串

#### 4.1.2.1.21 AT+BLEUNPAIR 取消配对

格式	AT+BLEUNPAIR=<addr_type,addr>
响应	<ul style="list-style-type: none"> <li>成功：断连</li> <li>失败：ERROR</li> </ul>
参数说明	<addr_type>：蓝牙设备类型 <addr>：蓝牙设备地址
示例	AT+BLEUNPAIR=0,0x112233445566
注意	设备类型取值范围为{0（公共设备地址）,1（随机设备地址）,2（公共本端地址）,3（随机静态本端地址）},设备地址为长度为 14 的字符串

#### 4.1.2.1.22 AT+BLEUNPAIRALL 取消所有配对

格式	AT+BLEUNPAIRALL
响应	<ul style="list-style-type: none"> <li>成功：断连</li> <li>失败：ERROR</li> </ul>
参数说明	-
示例	AT+BLEUNPAIRALL
注意	-

#### 4.1.2.1.23 AT+BLECONNPARDUPD 更新连接参数

格式	AT+BLECONNPARDUPD=<conn_handle,interval_min,interval_max,slave_latency,timeout_multiplier>
----	--

响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<p>&lt;conn_handle&gt;: 连接句柄;</p> <p>&lt;interval_min&gt;: 链路调度最小间隔, [0x06, 0x0C80], Time=N*1.25ms</p> <p>&lt;interval_max&gt;: 链路调度最大间隔, [0x06, 0x0C80], Time=N*1.25ms</p> <p>&lt;slave_latency&gt;: 延迟周期, 单位 slot(该值表示在设置值的周期内可以不回复, 为 0 时表示每包都需回复)</p> <p>&lt;timeout_multiplier&gt;: 超时断连间隔</p>
示例	AT+BLECONNPARDUPD=0, 0x48,0x48,0,500
注意	-

#### 4.1.2.1.24 AT+BLECONN 与 BLE 设备连接

格式	AT+BLECONN=<addr_type,addr>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<p>&lt;addr_type&gt;: 蓝牙设备类型</p> <p>&lt;addr&gt;: 蓝牙设备地址</p>
示例	AT+BLECONN=0,0x112233445566
注意	<p>设备类型取值范围为 0~3:</p> <p>0: 公共设备地址;</p> <p>1: 随机设备地址;</p> <p>2: 公共本端地址;</p> <p>3: 随机静态本端地址。</p> <p>设备地址为长度为 14 的字符串。</p>

#### 4.1.2.1.25 AT+BLEDISCONN 与 BLE 设备断开连接

格式	AT+BLEDISCONN=<addr_type,<addr>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<addr_type>: 蓝牙设备类型 <addr>: 蓝牙设备地址
示例	AT+BLEDISCONN=0,0x112233445566
注意	设备类型取值范围为{0 (公共设备地址),1 (随机设备地址),2 (公共本端地址),3 (随机静态本端地址)},设备地址为长度为 14 的字符串

#### 4.1.2.1.26 AT+BLEGAPREGCBK 注册 BLE 回调函数

格式	AT+BLEGAPREGCBK
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	-
示例	AT+BLEGAPREGCBK
注意	-

#### 4.1.2.1.27 AT+BLESETPHY 设置 BLE PHY

格式	AT+BLESETPHY=<conn_handle>,<all_phys>,<tx_phys>,<rx_phys>,<phy_options>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;conn_handle&gt;: 连接句柄</li><li>&lt;all_phys&gt;: 0: rx PHY 生效</li></ul>

	<p>1: tx PHY 生效</p> <p>3: tx/rx PHY 都生效</p> <ul style="list-style-type: none"> <li>• &lt;tx_phy&gt;: tx phy 值</li> </ul> <p>0: 1M PHY</p> <p>1: 2M PHY</p> <p>2: Code PHY</p> <ul style="list-style-type: none"> <li>• &lt;rx_phy&gt;: rx phy 值</li> </ul> <p>0: 1M PHY</p> <p>1: 2M PHY</p> <p>2: Code PHY</p> <ul style="list-style-type: none"> <li>• &lt;phy_options&gt;: PHY 选项</li> </ul> <p>0: 主机没有首选编码</p> <p>1: 使用 S=2 编码 PHY(1M/2=512k)</p> <p>2: 使用 S=8 编码 PHY(1M/8=128k)</p>
示例	AT+BLESETPHY=0,2,1,1,0
注意	-

#### 4.1.2.2 gatts 模块 AT 命令

##### 4.1.2.2.1 AT+GATTSREGRV 创建一个 GATT server

格式	AT+GATTSREGRV=<uuid>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<uuid>: 应用 uuid
示例	AT+GATTSREGRV=0x1122
注意	-

#### 4.1.2.2.2 AT+GATTSUNREG 删除 GATT server，释放资源

格式	AT+GATTSUNREG=<uuid>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<uuid>：应用 uuid
示例	AT+GATTSUNREG=0x1122
注意	-

#### 4.1.2.2.3 AT+GATTSADDSERV 添加一个 GATT 服务

格式	AT+GATTSADDSERV=<server_id,svc_uuid,is_primary_flag>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id <svc_uuid>：服务 uuid <is_primary_flag>：是否是首要服务
示例	AT+GATTSADDSERV=1,0x1812,1
注意	-

#### 4.1.2.2.4 AT+GATTSSYNCADDSERV 添加一个 GATT 服务（同步）

格式	AT+GATTSSYNCADDSERV=<server_id,svc_uuid,is_primary_flag>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id <svc_uuid>：服务 uuid <is_primary_flag>：是否是首要服务

示例	AT+GATTSSYNCADD SERV=1,0x1812,1
注意	-

#### 4.1.2.2.5 AT+GATTSADDCHAR 为 GATT 服务添加一个特征

格式	AT+GATTSADDCHAR=<server_id,service_handle,chara_uuid,permissions,property,value_len,value>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<p>&lt;server_id&gt;: 服务端 id</p> <p>&lt;service_handle&gt;: 服务句柄</p> <p>&lt;chara_uuid&gt;: 特征 uuid</p> <p>&lt;permissions&gt;: 权限</p> <p>0x01: 可读</p> <p>0x02: 可写</p> <p>0x04: 需要加密</p> <p>0x08: 需要认证</p> <p>0x10: 需要授权</p> <p>0x20: 需要 MITM 保护</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 0x08 值为 0x0B</p> <p>&lt;property&gt;: 特性</p> <p>0x01: 广播特征值</p> <p>0x02: 读特征值</p> <p>0x04: 写特征值并且不需要响应</p> <p>0x08: 写特征值</p> <p>0x10: 通知特征值</p> <p>0x20: 指示特征值</p> <p>0x40: 签名写特征值</p> <p>0x80: 在特征扩展特性描述符中定义了附加的特征特性</p>

	<p>注：多个特征类型使用' '后的值，如读写特征为, 0x02 0x08 值为 0x0A</p> <p>&lt;value_len&gt;: 值长度</p> <p>&lt;value&gt;: 值</p>
示例	AT+GATTSSYNCADDCHAR=1,1,0x1234,0x01,0x02,4,01010003
注意	值是长度为值长度两倍的字符串，value_len 最大值没有规定，实际生效范围以对端扫描为准

#### 4.1.2.2.6 AT+GATTSSYNCADDCHAR 为 GATT 服务添加一个特征（同步）

格式	AT+GATTSSYNCADDCHAR=<server_id,service_handle,chara_uuid,permissions,propertise,value_len,value>
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	<p>&lt;server_id&gt;: 服务端 id</p> <p>&lt;service_handle&gt;: 服务句柄</p> <p>&lt;chara_uuid&gt;: 特征 uuid</p> <p>&lt;permissions&gt;: 权限</p> <p>0x01: 可读</p> <p>0x02: 可写</p> <p>0x04: 需要加密</p> <p>0x08: 需要认证</p> <p>0x10: 需要授权</p> <p>0x20: 需要 MITM 保护</p> <p>注：多个权限类型使用' '后的值，如可读可写需要授权为, 0x01 0x02 0x08 值为 0x0B</p> <p>&lt;propertise&gt;: 特性</p> <p>0x01: 广播特征值</p> <p>0x02: 读特征值</p> <p>0x04: 写特征值并且不需要响应</p> <p>0x08: 写特征值</p>

	<p>0x10: 通知特征值</p> <p>0x20: 指示特征值</p> <p>0x40: 签名写特征值</p> <p>0x80: 在特征扩展特性描述符中定义了附加的特征特性</p> <p>注: 多个特征类型使用' '后的值, 如读写特征为, 0x02 0x08 值为 0x0A</p> <p>&lt;value_len&gt;: 值长度</p> <p>&lt;value&gt;: 值</p>
示例	AT+GATTSSYNCADDCHAR=1,1,0x2a4a,0x01,0x02,4,01010003
注意	值是长度为值长度两倍的字符串

#### 4.1.2.2.7 AT+GATTSSADDDESCR 为最新的特征添加一个描述符

格式	AT+GATTSSADDDESCR=<server_id,service_handle,chara_uuid,permissions,value_len,value>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<p>&lt;server_id&gt;: 服务端 id</p> <p>&lt;service_handle&gt;: 服务句柄</p> <p>&lt;chara_uuid&gt;: 特征 uuid</p> <p>&lt;permissions&gt;: 权限</p> <p>0x01: 可读</p> <p>0x02: 可写</p> <p>0x04: 需要加密</p> <p>0x08: 需要认证</p> <p>0x10: 需要授权</p> <p>0x20: 需要 MITM 保护</p> <p>注: 多个特征类型使用' '后的值, 如可读可写需要授权为 0x01 0x02 0x08, 值为 0x0B</p> <p>&lt;value_len&gt;: 值长度</p> <p>&lt;value&gt;: 值</p>



示例	AT+GATTSADDDESCR=1,1,0x2902,0x03,2,0100
注意	值是长度为值长度两倍的字符串

4.1.2.2.8 AT+GATTSSYNCADDDESCR 为最新的特征添加一个描述符（同步）

格式	AT+GATTSSYNCADDDESCR=<server_id,service_handle,chara_uuid,permissions,value_len,value>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<p>&lt;server_id&gt;：服务端 id</p> <p>&lt;service_handle&gt;：服务句柄</p> <p>&lt;chara_uuid&gt;：特征 uuid</p> <p>&lt;permissions&gt;：权限</p> <p>0x01：可读</p> <p>0x02：可写</p> <p>0x04：需要加密</p> <p>0x08：需要认证</p> <p>0x10：需要授权</p> <p>0x20：需要 MITM 保护</p> <p>注：多个特征类型使用 后的值，如可读可写需要授权为 0x01 0x02 0x08，值为 0x0B</p> <p>&lt;value_len&gt;：值长度</p> <p>&lt;value&gt;：值</p>
示例	AT+GATTSSYNCADDDESCR=1,1,0x2902,0x03,2,0100
注意	值是长度为值长度两倍的字符串

4.1.2.2.9 AT+GATTSSSTARTSERV 启动指定的 GATT 服务

格式	AT+GATTSSSTARTSERV=<server_id,service_handle>
响应	<ul style="list-style-type: none"><li>成功：OK</li></ul>

	<ul style="list-style-type: none"><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id <service_handle>：服务句柄
示例	AT+GATTSSSTARTSERV=1,1
注意	-

#### 4.1.2.2.10 AT+GATTSDDELALLSERV 删除指定 server 上的所有服务

格式	AT+GATTSDDELALLSERV=<server_id>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id
示例	AT+GATTSDDELALLSERV=1
注意	-

#### 4.1.2.2.11 AT+GATTSSENDRSP 发送响应

格式	AT+GATTSSENDRSP=<server_id,conn_handle,request_id,status,offset,value_len,value>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id <conn_handle>：连接句柄 <request_id>：请求 id <status>：请求结果 <offset>：偏移 <value_len>：值长度 <value>：值
示例	AT+GATTSSENDRSP=1,0,req_id,0,0,2,0x4562

注意	值是长度为值长度两倍的字符串
----	----------------

#### 4.1.2.2.12 AT+GATTSSNDNTFY 发送通知或指示

格式	AT+GATTSSNDNTFY=<server_id,conn_handle,attr_handle,value_len,value>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id <conn_handle>：连接句柄 <attr_handle>：偏移 <value_len>：值长度 <value>：值
示例	AT+GATTSSNDNTFY=1,0,9,7,0x0000000000000014
注意	值是长度为值长度两倍的字符串

#### 4.1.2.2.13 AT+GATTSSNDNTFYBYUUID 根据 uuid 发送通知或指示

格式	AT+GATTSSNDNTFYBYUUID=<server_id,conn_handle,chara_uuid,start_handle,end_handle,value_len,value>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>：服务端 id <conn_handle>：连接句柄 <chara_uuid>：特征 uuid <start_handle>：起始句柄 <end_handle>：结束句柄 <value_len>：值长度 <value>：值
示例	AT+GATTSSNDNTFYBYUUID=1,0,0x2a4d,1,9,7,0x0000000000000014

注意	值是长度为值长度两倍的字符串
----	----------------

#### 4.1.2.2.14 AT+GATTSREGCBK 注册 GATT 服务端回调函数

格式	AT+GATTSREGCBK
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	-
示例	AT+GATTSREGCBK
注意	-

#### 4.1.2.2.15 AT+GATTSSETMTU 在连接之前设置 server rx mtu

格式	AT+GATTSSETMTU=<server_id,mtu_size>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<server_id>: 服务端 id <mtu_size>: mtu 大小
示例	AT+GATTSSETMTU=1,23
注意	-

### 4.1.2.3 gattc 模块 AT 命令

#### 4.1.2.3.1 AT+GATTCREG 创建一个 GATT client

格式	AT+GATTCREG=<uuid>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<uuid>: 客户端 uuid

示例	AT+GATTCREG=0x1212
注意	-

#### 4.1.2.3.2 AT+GATTCUNREG 删除 GATT client，释放资源

格式	AT+GATTCUNREG=<client_id>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<client_id>：客户端 id，AT+GATTCREG 命令注册返回
示例	AT+GATTCUNREG=1
注意	-

#### 4.1.2.3.3 AT+GATTCFNDSEVR 发现服务

格式	AT+GATTCFNDSEVR=<client_id,conn_id,uuid>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<client_id>：客户端 id <conn_id>：连接 id <uuid>：查找的服务的 uuid
示例	AT+GATTCFNDSEVR=1,1,0x1212
注意	-

#### 4.1.2.3.4 AT+GATTCFNDCHAR 发现特征

格式	AT+GATTCFNDCHAR=<client_id,conn_id,service_handle,uuid>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>

参数说明	<client_id>: 客户端 id; <conn_id>: 连接 id <service_handle>: 服务句柄 <uuid>: 查找的特征值的 uuid
示例	AT+GATTCFNDCHAR=1,1,0,0x1212
注意	-

#### 4.1.2.3.5 AT+GATTCFNDDESCR 发现描述符

格式	AT+GATTCFNDDESCR=<client_id,conn_id,handle>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<client_id>: 客户端 id <conn_id>: 连接 id <handle>: 服务句柄
示例	AT+GATTCFNDDESCR=1,1,0
注意	-

#### 4.1.2.3.6 AT+GATTREADBYHDL 读取 by hdl

格式	AT+GATTREADBYHDL=<client_id,conn_id,handle>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<client_id>: 客户端 id <conn_id>: 连接 id <handle>: 服务句柄
示例	AT+GATTREADBYHDL=1,1,0
注意	-

#### 4.1.2.3.7 AT+GATTCREADBYUUID 读取 by\_uuid

格式	AT+GATTCREADBYUUID=<client_id,conn_id,start_hdl,end_hdl,u uid>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<client_id>：客户端 id <conn_id>：连接 id <start_hdl>：起始句柄 <end_hdl>：结束句柄 <uuid>：想要读的 uuid
示例	AT+GATTCREADBYUUID=1,0,13,13,2a4d
注意	-

#### 4.1.2.3.8 AT+GATTCWRITEREQ 写 by\_hdl req

格式	AT+GATTCWRITEREQ=<client_id,conn_id,handle,data_len,data>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<client_id>：客户端 id <conn_id>：连接 id <handle>：句柄 <data_len>：数据长度 <data>：数据
示例	AT+GATTCWRITEREQ=1,0,13,1,0x11
注意	data_len 需小于 MTU

#### 4.1.2.3.9 AT+GATTWRITECMD 写 by hdl cmd

格式	AT+GATTWRITECMD=<client_id,conn_id,handle,data_len,data>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<client_id>: 客户端 id; <conn_id>: 连接 id <handle>: 句柄 <data_len>: 数据长度 <data>: 数据
示例	AT+GATTWRITECMD=1,0,13,1,0x11
注意	-

#### 4.1.2.3.10 AT+GATTCEXCHMTU 交换 MTU 请求

格式	AT+GATTCEXCHMTU=<server_id,conn_id,mtu_size>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<server_id>: 服务端 id <conn_id>: 连接 id <mtu_size>: client rx mtu 大小
示例	AT+GATTCEXCHMTU=1,0,100
注意	-

#### 4.1.2.3.11 AT+GATTCREGCBK 注册 GATT 客户端回调函数

格式	AT+GATTCREGCBK
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>



参数说明	-
示例	AT+GATTCREGCBCK
注意	-

## 4.2 SLE

### 4.2.1 SLE AT 指令一览表

指令	描述
AT+SLEENABLE	SLE 使能
AT+SLEDISABLE	SLE 去使能
AT+SLESETADVPAR	设置 SLE 广播参数
AT+SLERMVADV	删除广播参数
AT+SLESETADVDATA	设置指令
AT+SLESTARTADV	起 SLE 广播
AT+SLESTOPADV	停 SLE 广播
AT+SLESTARTSCAN	启动扫描
AT+SLESTOPSCAN	关闭扫描
AT+SLESETNAME	设置本端名称
AT+SLEGETNAME	获取本端名称
AT+SLESETADDR	设置本端地址
AT+SLEGETADDR	获取本端地址
AT+SLECONN	建立 SLE 连接
AT+SLEDISCONN	断开 SLE 连接
AT+SLESETPHY	设置 SLE PHY

指令	描述
AT+SLESETDEFAULTCONNP	设置 SLE 默认连接参数
AT+SLESETDATALEN	设置 SLE 连接传输单个数据包长度
AT+SLESETMCS	设置 SLE 连接 mcs 传输特性
AT+SLEPAIR	进行加密配对
AT+SLEUNPAIR	移除加密配对
AT+SLEGETPAIREDNUM	获取配对设备数目
AT+SLEGETPAIRDEV	获取配对设备
AT+SLEGETPAIRSTA	获取配对状态
AT+SLEGETBONDDEV	获取绑定设备状态
AT+SLECONNPARUPD	星闪逻辑链路更新参数
AT+SLEREADPEERRSSI	读取对端 rssi
AT+SSAPSADDSRV	注册服务端
AT+SSAPSDELALLSRV	去注册服务端
AT+SSAPSADDSERV	添加服务
AT+SSAPSSYNCAADDSERV	添加服务同步
AT+SSAPSADDPROPERTY	添加属性
AT+SSAPSSYNCAADDPROPERTY	添加属性同步
AT+SSAPSADDDDESCR	添加属性描述符
AT+SSAPSSYNCAADDDDESCR	添加属性描述符同步
AT+SSAPSSTARTSERVICE	start service
AT+SSAPSSNDNTFY	服务端向客户端发送通知
AT+SSAPSNTFYBYUUID	服务端向客户端通过 uuid 发送通知

指令	描述
AT+SSAPSSNDRESP	服务端向客户端发送响应
AT+SSAPSREGCBK	服务端注册回调函数
AT+SSAPCREGCBK	注册 SSAPC 回调函数
AT+SSAPCFNDSTRU	发现 service
AT+SSAPCFNDSTRUB YHDL	通过句柄发现 service
AT+SSAPCWRITECMD	客户端向服务端写入数据
AT+SSAPCWRITEREQ	客户端向服务端发送写请求
AT+SSAPCEXCHINFO	客户端发起信息交换
AT+SSAPSSETINFO	设置服务端信息
AT+SSAPCREADBYUU ID	客户端通过 uuid 发送读请求
AT+SSAPCREADREQ	客户端读取服务端属性数据
AT+SLESETSCANPAR	设置扫描参数
AT+SLEDISCONNALL	SLE 断开所有连接
AT+SSAPPERFORMAN CECFG	SLE ssap 数传流控参数设置
AT+SSAPPERFORMAN CEWRITE	SLE ssap 客户端 write_cmd 数传开始指令
AT+SSAPPERFORMAN CENOTIFY	SLE ssap 服务端 indicate&notify 数传开始指令
AT+SSAPCCLEANREC VCOUNT	SLE ssap 流控客户端清理接收包数量
AT+SSAPSCLEANREC VCOUNT	SLE ssap 流控服务端清理接收数据包数量
AT+SSAPQOSRECVCB KREG	SLE ssap 流控回调注册

## 4.2.2 SLE AT 指令描述

使用命令之前的配置：

- 加载平台和星闪驱动，komod 为驱动文件所在路径

```
insmod /komod/plat_soc.ko  
insmod /komod/sle_soc.ko
```

- 可执行程序增加执行权限，/bin 为可执行文件所在路径，如果和 bluetoothd 同时运行，需要放到不同目录

```
cd /bin  
chmod +x sparklinkd  
chmod +x sparklinkctrl
```

- 进入视图

```
cd /bin  
sparklinkd&  
sparklinkctrl
```

#### 4.2.2.1 SLE 使能

设置指令	AT+SLEENABLE
响应	<ul style="list-style-type: none"><li>• 成功：OK</li><li>• 失败：ERROR</li></ul>
参数说明	-
示例	AT+SLEENABLE
注意	-

#### 4.2.2.2 SLE 去使能

设置指令	AT+SLEDISABLE
响应	<ul style="list-style-type: none"><li>• 成功：OK</li><li>• 失败：ERROR</li></ul>
参数说明	-
示例	AT+SLEDISABLE
注意	-

#### 4.2.2.3 设置 SLE 广播参数

设置指令	AT+SLESETADVPAR=<announce_handle>,<announce_mode>,<announce_interval_min>,<announce_interval_max>,<own_addr_type>,<own_addr_addr>,<peer_addr_type>,<peer_addr_addr>,[tx_power]
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;announce_handle&gt;：设备公开句柄，取值范围[0, 0x10]</li><li>&lt;announce_mode&gt;：设备公开类型<ul style="list-style-type: none"><li>0：不可连接不可扫描</li><li>1：可连接不可扫描</li><li>2：不可连接可扫描</li><li>3：可连接可扫描</li></ul></li><li>&lt;announce_interval_min&gt;：最小设备公开周期，0x000020~0xfffff，单位 125μs</li><li>&lt;announce_interval_max&gt;：最大设备公开周期，0x000020~0xfffff，单位 125μs</li><li>&lt;own_addr_type&gt;：SLE 本端地址类型<ul style="list-style-type: none"><li>0：公有地址，</li><li>6：随机地址</li></ul></li><li>&lt;own_addr_addr&gt;：SLE 本端设备地址</li><li>&lt;peer_addr_type&gt;：SLE 对端设备地址类型<ul style="list-style-type: none"><li>0：公有地址，</li><li>6：随机地址</li></ul></li><li>&lt;peer_addr_addr&gt;：SLE 对端设备地址</li><li>[tx_power]：功率，取值范围：-127~+20，或 127(缺省参数，默认值为 127)</li></ul>
示例	AT+SLESETADVPAR=1,3,200,200,0,000000000000,0,000000000000
注意	此命令需在 SLE 使能 AT+SLEENABLE 后下发。

#### 4.2.2.4 删除广播参数

设置指令	AT+SLERMVADV=<adv_enable>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<adv_handle>: adv handle, 取值范围[0, 0x10]
示例	AT+SLERMVADV=1
注意	此命令需在广播未启动状态下执行。

#### 4.2.2.5 设置 SLE 广播数据

设置指令	AT+SLESETADVDATA=<adv_handle>,<announce_data_len>,<seek_rsp_data_len>,<announce_data>,<seek_rsp_data>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;adv_handle&gt;: 广播 handle, 取值范围[0, 0x10]</li><li>&lt;announce_data_len&gt;: 设备公开数据长度。</li><li>&lt;seek_rsp_data_len&gt;: 扫描响应数据长度。</li><li>&lt;announce_data&gt;: 设备公开数据 (hex 类型字符串, 最大长度 521 个字符)。</li><li>&lt;seek_rsp_data&gt;: 扫描响应数据 (hex 类型字符串, 最大长度 521 个字符)。</li></ul>
示例	AT+SLESETADVDATA=1,10,4,aabbccddeeff11223344,11224455
注意	此命令需在 SLE 使能 AT+SLEENABLE 后下发。

#### 4.2.2.6 起 SLE 广播

设置指令	AT+SLESTARTADV=<adv_enable>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>

参数说明	<adv_handle>: adv handle, 取值范围[0, 0x10]
示例	AT+SLESTARTADV=1
注意	此命令需在 SLE 使能 AT+SLEENABLE 后下发。

#### 4.2.2.7 停 SLE 广播

设置指令	AT+SLESTOPADV=<adv_handle>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<adv_handle>: adv handle
示例	AT+SLESTOPADV=1
注意	此命令需在 SLE 起广播 AT+SLESTARTADV=1 后下发。

#### 4.2.2.8 设置扫描参数

设置指令	AT+SLESETSCANPAR=<scan_type>,<scan_interval>,<scan_window>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;scan_type&gt;: 扫描类型 0: 被动扫描 1: 主动扫描</li> <li>&lt;scan_interval&gt;: 扫描间隔, 取值范围[0x14, 0xFFFF], 单位 125μs</li> <li>&lt;scan_window&gt;: 扫描窗口, 取值范围[0x14, 0xFFFF], 单位 125μs</li> </ul>
示例	AT+SLESETSCANPAR=0,0x48,0x48
注意	此命令需在 SLE 起扫描 AT+SLESTARTSCAN 前下发

#### 4.2.2.9 使能扫描

设置指令	AT+SLESTARTSCAN
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+SLESTARTSCAN
注意	-

#### 4.2.2.10 关闭扫描

设置指令	AT+SLESTOPSCAN
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+SLESTOPSCAN
注意	-

#### 4.2.2.11 设置本端名称

设置指令	AT+SLESETNAME
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;len&gt;: name 长度。</li><li>&lt;name&gt;: 名字。</li></ul>
示例	AT+SLESETNAME=7,SDKTEST
注意	-



#### 4.2.2.12 获取本端名称

设置指令	AT+SLEGETNAME
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	-
示例	AT+SLEGETNAME
注意	-

#### 4.2.2.13 设置本端地址

设置指令	AT+SLESETADDR
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<addr_type>: 地址类型, 当前仅支持联盟分配地址标识-0 <addr>: 地址
示例	AT+SLESETADDR=0,0x0000000000001
注意	-

#### 4.2.2.14 获取本端地址

设置指令	AT+SLEGETADDR
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	-
示例	AT+SLEGETADDR
注意	-

#### 4.2.2.15 建立 SLE 连接

设置指令	AT+SLECONN=<sle_addr_type>,<sle_addr>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul> 连接成功后会打印[connected]字样以及对端设备地址与 handle 值。
参数说明	<ul style="list-style-type: none"><li>&lt;sle_addr_type&gt;：SLE 设备地址类型。 取值：<ul style="list-style-type: none"><li>0：公有地址。</li><li>6：随机地址。</li></ul></li><li>&lt;sle_addr&gt;：SLE 设备地址。</li></ul>
示例	AT+SLECONN=0,0x00000000000000
注意	-

#### 4.2.2.16 星闪逻辑链路更新参数

设置指令	AT+SLECONNPARUPD=<conn_id>,<interval_min>,<interval_max>,<max_latency>,<supervision_timeout>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;conn_id&gt;：连接 ID（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;interval_min&gt;：链路调度最小间隔，取值范围[0x0020, 0x32000]，单位 125μs</li><li>&lt;interval_max&gt;：链路调度最大间隔，取值范围[0x0020, 0x32000]，单位 125μs</li><li>&lt;max_latency&gt;：延迟周期，单位 slot（该值表示在设置值的周期内可以不回复，为 0 时则表示每包都需回复）</li><li>&lt;supervision_timeout&gt;：超时时间，单位 10ms</li></ul>
示例	AT+SLECONNPARUPD=0,20,20,0,500

注意	-
----	---

#### 4.2.2.17 星闪读取远端 rssi

该命令的作用：读取当前 SLE ACB 链路的 RSSI 信号强度，RSSI（Received Signal Strength Indicator）是接收信号的强度指示。

设置指令	AT+SLEREADPEERRSSI=<conn_id>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<conn_id>：连接 ID。
示例	AT+SLEREADPEERRSSI=0
注意	-

#### 4.2.2.18 断开 SLE 连接

设置指令	AT+SLEDISCONN=<sle_addr_type>,<sle_addr>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul> <p>连接成功后会打印[disconnected]字样以及对端设备地址与 handle 值。</p>
参数说明	<ul style="list-style-type: none"><li>&lt;sle_addr_type&gt;：SLE 设备地址类型。 取值：<ul style="list-style-type: none"><li>0：公有地址。</li><li>6：随机地址。</li></ul></li><li>&lt;sle_addr&gt;：SLE 设备地址。</li></ul>
示例	AT+SLEDISCONN=0,000000000000
注意	-

#### 4.2.2.19 设置 SLE PHY

默认 1M 4M 暂时不支持

该命令作用：某些场景下需要高的传输速率，此时就通过设置 tx\_phy、rx\_phy 参数为 2M。如果两端都支持 2M,才能设置成功。

设置指令	AT+SLESETPHY=<conn_id>,<tx_phy>,<rx_phy>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;conn_id&gt;：连接 id（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;tx_phy&gt;：tx phy 值 0：1M PHY 1：2M PHY 2：4M PHY</li><li>&lt;rx_phy&gt;：tx phy 值 0：1M PHY 1：2M PHY 2：4M PHY</li></ul>
示例	AT+SLESETPHY=0,1,1
注意	-

#### 4.2.2.20 设置 SLE 默认连接参数

设置指令	AT+ SLESETDEFAULTCONNP =<enable_filter_policy>,<initiate_phys>,<gt_negotiate>,<scan_interval>,<scan_window>,<max_interval>,<min_interval>,<timeout>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;enable_filter_policy&gt;：是否打开链路过滤</li><li>&lt;initiate_phys&gt;：链路扫描带宽</li></ul>

	<p>1: 1M PHY</p> <p>2: 2M PHY</p> <ul style="list-style-type: none"> <li>• &lt;gt_negotiate&gt;: 链路建立时是否进行 G 和 T 交互</li> <li>• &lt;scan_interval&gt;: 扫描对端设备的 interval 最小值: 0x14 最大值: 无上限</li> <li>• &lt;scan_window&gt;: 扫描对端设备的 windows 最小值: 0x14 最大值: 传入的&lt;scan_interval&gt;</li> <li>• &lt;max_interval&gt;: 链路最大调度 interval 最大值: 32000</li> <li>• &lt;min_interval&gt;: 链路最小调度 interval 最小值: 10</li> <li>• &lt;timeout&gt;: 链路超时时间(N×10ms) 最小值: 10 最大值: 3200</li> </ul>
示例	AT+SLESETDEFAULTCONNP=0,1,0x1,0x20,0x20,0x64,0x64,0x1FC
注意	-

#### 4.2.2.21 设置 SLE 连接传输单个数据包长度

设置指令	AT+SLESETDATALEN =<conn_id>,<tx_octets>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;conn_id&gt;: SLE 连接 id (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>• &lt;tx_octets&gt;: 传输单个数据包最大长度 WS73E 最大值: 1524 WS73US 最大值: 254</li> </ul>

示例	AT+SLESETDATALEN=0,251
注意	传输单个数据包最大长度的最终值为主机、从机中的较小值

#### 4.2.2.22 设置 SLE 连接 mcs 传输特性

设置指令	AT+SLESETMCS=<conn_id>,<mcs>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;conn_id&gt;: SLE 连接 id (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>&lt;mcs&gt;: conn_id 对应的连接的传输方式</li> </ul>
示例	AT+SLESETMCS=0,14
注意	-

#### 4.2.2.23 进行加密配对

设置指令	AT+SLEPAIR=<sle_addr_type>,<sle_addr>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;sle_addr_type&gt;: SLE 设备地址类型 取值: <ul style="list-style-type: none"> <li>0: 公有地址</li> <li>6: 随机地址</li> </ul> </li> <li>&lt;sle_addr&gt;: SLE 设备地址</li> </ul>
示例	AT+SLEPAIR=0,000000000000
注意	需在 SLE 建立连接以后, 和对端启动加密配对

#### 4.2.2.24 移除加密配对

设置指令	AT+SLEUNPAIR=<sle_addr_type>,<sle_addr>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;sle_addr_type&gt;：SLE 设备地址类型。 取值：<ul style="list-style-type: none"><li>0：公有地址。</li><li>6：随机地址。</li></ul></li><li>&lt;sle_addr&gt;：SLE 设备地址。</li></ul>
示例	AT+SLEUNPAIR=0,000000000000
注意	-

#### 4.2.2.25 获取配对设备数目

设置指令	AT+SLEGETPAIREDNUM
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	-
示例	AT+SLEGETPAIREDNUM
注意	-

#### 4.2.2.26 获取配对设备

设置指令	AT+SLEGETPAIRDEV
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	-

示例	AT+SLEGETPAIRDEV
注意	--

#### 4.2.2.27 获取设备配对状态

设置指令	AT+SLEGETPAIRSTA=<sle_addr_type>,<sle_addr>
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt; sle_addr_type &gt;: SLE 设备地址类型 0: 公有地址, 6: 随机地址</li><li>&lt;sle_addr&gt;: SLE 设备地址</li></ul>
示例	AT+SLEUNPAIR=0,000000000000
注意	-

#### 4.2.2.28 获取绑定设备

设置指令	AT+SLEGETBONDDEV
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	-
示例	AT+SLEGETBONDDEV
注意	-

#### 4.2.2.29 注册服务端

设置指令	AT+SSAPSADDSRV=<uuid>
响应	<ul style="list-style-type: none"><li>成功: OK</li></ul>



	<ul style="list-style-type: none"> <li>失败：ERROR</li> </ul>
参数说明	-
示例	AT+SSAPSADDSRV=0x1234
注意	-

#### 4.2.2.30 去注册服务端

设置指令	AT+SSAPSDELALLSRV
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	-
示例	AT+SSAPSDELALLSRV
注意	-

#### 4.2.2.31 添加服务

注册成“次要服务”无主要影响。现在都是主要服务，次要服务的概念弱化，因此通常情况下注册成主要服务。

设置指令	AT+SSAPSADDSERV=<uuid>,<is_primary>
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;uuid&gt;：SSAP 服务 UUID。</li> <li>&lt;is_primary&gt;：是否是首要服务。</li> </ul>
示例	AT+SSAPSADDSERV=0x2222,1
注意	功能使用 AT+SSAPSSYNCADDSERV 替代，推荐使用 AT+SSAPSSYNCADDSERV 添加服务。

#### 4.2.2.32 添加服务同步

设置指令	AT+SSAPSSYNCADDSERV=<uuid>,<is_primary>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;uuid&gt;: SSAP 服务 UUID。</li> <li>&lt;is_primary&gt;: 是否是首要服务。</li> </ul>
示例	AT+SSAPSSYNCADDSERV=0x2222,1
注意	-

#### 4.2.2.33 添加属性

设置指令	AT+SSAPSADDPROPERTY=<service_handle>,<uuid>,<permissions>,<operate_indication>,<value_len>,<value>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;service_handle&gt;: 服务的 handle。</li> <li>&lt;uuid&gt;: SSAP 特征 UUID。</li> <li>&lt;permissions&gt;: 特征权限。  0x01: 可读  0x02: 可写  0x04: 需要加密  0x08: 需要认证  0x10: 需要授权  注: 多个权限类型使用" "后的值, 如可读可写需要授权为,  0x01 0x02 0x08 值为 0x0B</li> <li>&lt;operate_indication&gt;: 操作指示。  0x01: 数据值可被读取  0x02: 数据值可被写入, 写入后无反馈  0x04: 数据值可被写入, 写入后产生反馈给客户端</li> </ul>

	<p>0x08: 数据值通过通知方式传递给客户端</p> <p>0x10: 数据值通过指示方式传递给客户端</p> <p>0x20: 数据值可携带在广播中</p> <p>0x100: 数据值说明描述符可被写入</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 值为 0x03</p> <ul style="list-style-type: none"> <li>• &lt;value_len&gt;: 响应的数据长度。</li> <li>• &lt;value&gt;: 响应的数据。</li> </ul>
示例	AT+SSAPSADDPROPERTY=1,0x2323,5,5,2,0x1234
注意	功能使用 AT+SSAPSSYNCADDPROPERTY 替代, 推荐使用 AT+SSAPSSYNCADDPROPERTY 添加属性

#### 4.2.2.34 添加属性同步

设置指令	AT+SSAPSSYNCADDPROPERTY=<service_handle>,<uuid>,<permissions>,<operate_indication>,<value_len>,<value>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;service_handle&gt;: 服务的 handle, AT+SSAPSSYNCADDSERV 返回的 handle。</li> <li>• &lt;uuid&gt;: SSAP 特征 UUID。</li> <li>• &lt;permissions&gt;: 特征权限。 <ul style="list-style-type: none"> <li>0x01: 可读。</li> <li>0x02: 可写。</li> <li>0x04: 需要加密。</li> <li>0x08: 需要认证。</li> <li>0x10: 需要授权。</li> </ul> </li> </ul> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 0x08 值为 0x0B</p> <ul style="list-style-type: none"> <li>• &lt;operate_indication&gt;: 操作指示。 <ul style="list-style-type: none"> <li>0x01: 数据值可被读取。</li> </ul> </li> </ul>

	<p>0x02: 数据值可被写入, 写入后无反应。</p> <p>0x04: 数据值可被写入, 写入后产生反馈给客户端。</p> <p>0x08: 数据值通过通知的方式传递给客户端。</p> <p>0x10: 数据值通过指示的方式传递给客户端。</p> <p>0x20: 数据值可携带在广播中。</p> <p>0x100: 数据值说明描述符可被写入。</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 值为 0x03</p> <ul style="list-style-type: none"> <li>• &lt;value_len&gt;: 响应的数据长度。</li> <li>• &lt;value&gt;: 响应的数据。</li> </ul>
示例	AT+SSAPSSYNCADDPROPERTY=1,0x2323,5,5,2,0x1234
注意	WS73 1.10.111 及之后版本, 服务 handle 从 16 开始, 命令可使用 AT+SSAPSSYNCADDPROPERTY=16,0x2323,5,5,2,0x1234

#### 4.2.2.35 添加属性描述符

设置指令	AT+SSAPSADDDDESCR=<service_handle>,<property_handle>,<uuid>,<permissions>,<operate_indication>,<type>,<value_len>,<value>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;service_handle&gt;: 服务 handle。</li> <li>• &lt;property_handle&gt;: 属性 handle。</li> <li>• &lt;uuid&gt;: SSAP 描述符 UUID。</li> <li>• &lt;permissions&gt;: 特征权限。</li> </ul> <p>0x01: 可读</p> <p>0x02: 可写</p> <p>0x04: 需要加密</p> <p>0x08: 需要认证</p> <p>0x10: 需要授权</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为,</p>

	<p>0x01 0x02 0x08 值为 0x0B</p> <ul style="list-style-type: none"> <li>• &lt;operate_indication&gt;: 操作指示。</li> </ul> <p>0x01: 数据值可被读取</p> <p>0x02: 数据值可被写入, 写入后无反馈</p> <p>0x04: 数据值可被写入, 写入后产生反馈给客户端</p> <p>0x08: 数据值通过通知方式传递给客户端</p> <p>0x10: 数据值通过指示方式传递给客户端</p> <p>0x20: 数据值可携带在广播中</p> <p>0x100: 数据值说明描述符可被写入</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 值为 0x03</p> <ul style="list-style-type: none"> <li>• &lt;type&gt;: 描述符类型。</li> <li>• &lt;value_len&gt;: 数据长度。</li> <li>• &lt;value&gt;: 数据。</li> </ul>
示例	AT+SSAPSADDDESCRIPTOR=1,2,0x3333,5,5,2,2,0x0200
注意	功能使用 AT+SSAPSSYNCAADDDESCRIPTOR 替代, 推荐使用 AT+SSAPSSYNCAADDDESCRIPTOR 添加属性描述符。

#### 4.2.2.36 添加属性描述符同步

设置指令	AT+SSAPSSYNCAADDDESCRIPTOR=<service_handle>,<property_handle>,<uuid>,<permissions>,<operate_indication>,<type>,<value_len>,<value>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;service_handle&gt;: 服务 handle, AT+SSAPSSYNCAADDSERV 返回的 handle</li> <li>• &lt;property_handle&gt;: 属性 handle, AT+SSAPSSYNCAADDPROPERTY 返回的 handle</li> <li>• &lt;uuid&gt;: SSAP 描述符 UUID</li> <li>• &lt;permissions&gt;: 特征权限</li> </ul>

	<p>0x01: 可读</p> <p>0x02: 可写</p> <p>0x04: 需要加密</p> <p>0x08: 需要认证</p> <p>0x10: 需要授权</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 0x08 值为 0x0B</p> <ul style="list-style-type: none"> <li>• &lt;operate_indication&gt;: 操作指示</li> </ul> <p>0x01: 数据值可被读取</p> <p>0x02: 数据值可被写入, 写入后无反应</p> <p>0x04: 数据值可被写入, 写入后产生反馈给客户端</p> <p>0x08: 数据值通过通知的方式传递给客户端</p> <p>0x10: 数据值通过指示的方式传递给客户端</p> <p>0x20: 数据值可携带在广播中</p> <p>0x100: 数据值说明描述符可被写入</p> <p>注: 多个权限类型使用' '后的值, 如可读可写需要授权为, 0x01 0x02 值为 0x03</p> <ul style="list-style-type: none"> <li>• &lt;type&gt;: 描述符类型。</li> </ul> <p>0: 特征值。</p> <p>1: 属性说明描述符。</p> <p>2: 客户端配置描述符</p> <p>3: 服务端配置描述符</p> <p>4: 格式描述符</p> <p>5: 服务管理保留描述符, 0x05~0x1F</p> <p>0xFF: 厂商自定义描述符</p> <ul style="list-style-type: none"> <li>• &lt;value_len&gt;: 数据长度</li> <li>• &lt;value&gt;: 数据</li> </ul>
示例	AT+SSAPSSYNCADDDESCR=1,2,0x3333,5,5,2,2,0x0200
注意	WS73 1.10.111 及之后版本, 服务 handle 从 16 开始, 命令可以使用 AT+SSAPSSYNCADDDESCR=16,17,0x3333,5,5,2,2,0x0200

#### 4.2.2.37 服务端向客户端发送通知

设置指令	AT+SSAPSSNDNTFY=<conn_id>,<handle>,<type>,<value_len>,<value>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;conn_id&gt;：连接 ID（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;handle&gt;：属性 handle，AT+SSAPSSYNCADDPROPERTY 返回的 handle</li><li>&lt;type&gt;：SSAP 特征类型<ul style="list-style-type: none"><li>0：特征值</li><li>1：属性说明描述符</li><li>2：客户端配置描述符</li><li>3：服务端配置描述符</li><li>4：格式描述符</li><li>5：服务管理保留描述符，0x05~0x1F</li><li>0xFF：厂商自定义描述符</li></ul></li><li>&lt;value_len&gt;：数据长度</li><li>&lt;value&gt;：数据</li></ul>
示例	AT+SSAPSSNDNTFY=0,2,0,2,0x0200
注意	WS73 1.10.111 及之后版本，服务 handle 从 16 开始，命令可以使用 AT+SSAPSSNDNTFY=0,17,0,2,0x0200

#### 4.2.2.38 服务端向客户端通过 uuid 发送通知

设置指令	AT+SSAPSNTFYBYUUID=<conn_id>,<uuid>,<start_hdl>,<end_hdl>,<type>,<value_len>,<value>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>

参数说明	<ul style="list-style-type: none"> <li>• &lt;conn_id&gt;: 连接 ID (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>• &lt;uuid&gt;: 属性 uuid</li> <li>• &lt;start_hdl&gt;: 开始句柄</li> <li>• &lt;end_hdl&gt;: 结束句柄</li> <li>• &lt;type&gt;: SSAP 特征类型                         <ul style="list-style-type: none"> <li>0: 特征值</li> <li>1: 属性说明描述符</li> <li>2: 客户端配置描述符</li> <li>3: 服务端配置描述符</li> <li>4: 格式描述符</li> <li>5: 服务管理保留描述符, 0x05~0x1F</li> <li>0xFF: 厂商自定义描述符</li> </ul> </li> <li>• &lt;value_len&gt;: 数据长度</li> <li>• &lt;value&gt;: 数据</li> </ul>
示例	AT+SSAPSNTRYBYUUID=0,0x1234,0,0xFFFF,0,2,0x0200
注意	-

#### 4.2.2.39 服务端发送响应

设置指令	AT+SSAPSSNDRESP=<conn_id>,<request_id>,<status>,<value_len>,<value>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;conn_id&gt;: 连接 ID (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>• &lt;request_id&gt;: 请求 id</li> <li>• &lt;status&gt;: 发送响应原因</li> <li>• &lt;value_len&gt;: 数据长度</li> <li>• &lt;value&gt;: 数据</li> </ul>



示例	AT+SSAPSSNDRESP=0,0,1,2,0x0200
注意	-

发送响应原因说明:

```
typedef enum {
    ERRCODE_SSAP_INVALID_PDU = ERRCODE_SLE_SSAP_BASE + 0x01,          /*
    服务端接收的 PDU 无效*/
    ERRCODE_SSAP_PDU_NOT_SUPPORT = ERRCODE_SLE_SSAP_BASE + 0x02,
    /*服务端不支持处理接收的PDU*/
    ERRCODE_SSAP_UNKNOW = ERRCODE_SLE_SSAP_BASE + 0x03,
    /*服务端执行请求时发生未知错误*/
    ERRCODE_SSAP_INVALID_HANDLE = ERRCODE_SLE_SSAP_BASE + 0x04,
    /*请求中的句柄无效*/
    ERRCODE_SSAP_INSUFFICIENT_RESOURCES = ERRCODE_SLE_SSAP_BASE + 0x05,
    /*服务端没有足够资源完成请求*/
    ERRCODE_SSAP_PROHIBIT_READING = ERRCODE_SLE_SSAP_BASE + 0x06,
    /*服务端禁止客户端读取值*/
    ERRCODE_SSAP_PROHIBIT_WRITE = ERRCODE_SLE_SSAP_BASE + 0x07,
    /*服务端禁止客户端写入值*/
    ERRCODE_SSAP_CLIENT_NOT_AUTHENTICATED = ERRCODE_SLE_SSAP_BASE + 0x08,
    /*客户端未认证*/
    ERRCODE_SSAP_CLIENT_NOT_AUTHORIZATION = ERRCODE_SLE_SSAP_BASE + 0x09,
    /*客户端未被授权*/
    ERRCODE_SSAP_BEARER_NOT_ENCRYPTED = ERRCODE_SLE_SSAP_BASE + 0x0A,
    /*传输 PDU 的承载未加密*/
    ERRCODE_SSAP_ENTRIES_NOT_FOUND = ERRCODE_SLE_SSAP_BASE + 0x0B,
    /*服务端未找到对应条目*/
    ERRCODE_SSAP_DATA_NOT_FOUND = ERRCODE_SLE_SSAP_BASE + 0x0C,
    /*服务端未找到对应类型数据*/
    ERRCODE_SSAP_INCORRECT_DATA_TYPE = ERRCODE_SLE_SSAP_BASE + 0x0D,
    /*客户端发送写入数据类型不符的错误*/
    ERRCODE_SSAP_INCORRECT_DATA_VALUE = ERRCODE_SLE_SSAP_BASE + 0x0E,
    /*客户端发送写入值不符的错误*/
}
```

```

ERRCODE_SSAP_VALUE_OUT_OF_RANGE = ERRCODE_SLE_SSAP_BASE + 0x0F,
/*客户端写入的值超出范围*/

ERRCODE_SSAP_UPPERLAYER_APPLICATION_ERROR_MIN =
ERRCODE_SLE_SSAP_BASE + 0xAF,          /*预留给上层协议定义应用错误*/

ERRCODE_SSAP_UPPERLAYER_APPLICATION_ERROR_MAX =
ERRCODE_SLE_SSAP_BASE + 0xFF,          /*预留给上层协议定义应用错误*/

} errcode_sle_ssap_t;
    
```

#### 4.2.2.40 服务端注册回调

设置指令	AT+SSAPSREGCBK
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	-
示例	AT+SSAPSREGCBK
注意	-

#### 4.2.2.41 start service

设置指令	AT+SSAPSSTARTSERV=<service_handle>
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: ERROR</li> </ul>
参数说明	<service_handle>: 服务 handle
示例	AT+SSAPSSTARTSERV=1
注意	-

#### 4.2.2.42 注册 SSAPC 回调函数

设置指令	AT+SSAPCREGCBK
响应	<ul style="list-style-type: none"> <li>成功: OK</li> </ul>

	<ul style="list-style-type: none"> <li>失败：ERROR</li> </ul>
参数说明	-
示例	AT+SSAPCREGCBK
注意	-

#### 4.2.2.43 发现 service

设置指令	AT+SSAPCFNDSTRU=<client_id>,<conn_id>,<type>,<uuid>,<start_hdl>,<end_hdl>
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;client_id&gt;：客户端 id</li> <li>&lt;conn_id&gt;：连接 id（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li> <li>&lt;type&gt;：查找类型，取值如下： <ul style="list-style-type: none"> <li>0：服务结构</li> <li>1：首要服务</li> <li>3：属性</li> </ul> </li> <li>&lt;uuid&gt;：查找的 uuid，如果输入 0，代表不过滤 UUID，使用 type、start_hdl、end_hdl 去查找</li> <li>&lt;start_hdl&gt;：查找的服务起始句柄</li> <li>&lt;end_hdl&gt;：查找的服务结束句柄</li> </ul>
示例	AT+SSAPCFNDSTRU=0,0,1,0x1234,0,0xff
注意	-

查找类型字段说明：

```
typedef enum {
    SSAP_FIND_TYPE_SERVICE_STRUCTURE = 0x00,    /*服务结构*/
    SSAP_FIND_TYPE_PRIMARY_SERVICE    = 0x01,    /*首要服务*/
    SSAP_FIND_TYPE_REFERENCE_SERVICE = 0x02,     /*引用服务*/
}
```

```
SSAP_FIND_TYPE_PROPERTY      = 0x03,      /*属性*/
SSAP_FIND_TYPE_METHOD        = 0x04,      /*方法*/
SSAP_FIND_TYPE_EVENT         = 0x05,      /*事件*/
} ssap_find_type_t;
```

4.2.2.44 通过句柄发现 service

设置指令	AT+SSAPCFNDSTRUBYHDL=<client_id>,<conn_id>,<type>,<start_hdl>,<end_hdl>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;client_id&gt;：客户端 id</li><li>&lt;conn_id&gt;：连接 id（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;type&gt;：查找类型，取值如下： 0：服务结构 1：首要服务 3：属性</li><li>&lt;start_hdl&gt;：查找的服务起始句柄</li><li>&lt;end_hdl&gt;：查找的服务结束句柄</li></ul>
示例	AT+SSAPCFNDSTRUBYHDL=0,0,1,0,0xff
注意	-

查找类型字段说明：

```
typedef enum {
SSAP_FIND_TYPE_SERVICE_STRUCTURE = 0x00,      /*服务结构*/
SSAP_FIND_TYPE_PRIMARY_SERVICE   = 0x01,      /*首要服务*/
SSAP_FIND_TYPE_REFERENCE_SERVICE = 0x02,      /*引用服务*/
SSAP_FIND_TYPE_PROPERTY          = 0x03,      /*属性*/
SSAP_FIND_TYPE_METHOD            = 0x04,      /*方法*/
SSAP_FIND_TYPE_EVENT            = 0x05,      /*事件*/
}
```

```
} ssap_find_type_t;
```

#### 4.2.2.45 客户端向服务端写入数据

设置指令	AT+SSAPCWRITECMD=<client_id>,<conn_id>,<handle>,<type>,<len>,<write_data>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;client_id&gt;：客户端 id</li><li>&lt;conn_id&gt;：连接 id（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;handle&gt;：服务端注册服务的属性的 handle，AT+SSAPSSYNCADDPROPERTY 返回的 handle</li><li>&lt;type&gt;：客户端类型，取值：0/1/3 0：特征值 1：属性说明描述符 3：服务端配置描述符</li><li>&lt;len&gt;：写入数据长度</li><li>&lt;write_data&gt;：写入数据段</li></ul>
示例	AT+SSAPCWRITECMD=0,0,2,0,2,0x8899
注意	-

#### 4.2.2.46 客户端向服务端发送写请求

设置指令	AT+SSAPCWRITEREQ=<client_id>,<conn_id>,<handle>,<type>,<len>,<write_data>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;client_id&gt;：客户端 id</li><li>&lt;conn_id&gt;：连接 id（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;handle&gt;：服务端注册的服务的属性的 handle，</li></ul>

	<p>AT+SSAPSSYNCADDPROPERTY 返回的 handle</p> <ul style="list-style-type: none"> <li>• &lt;type&gt;: 客户端类型, 取值: 0/1/3 0: 特征值 1: 属性说明描述符 3: 服务端配置描述符</li> <li>• &lt;len&gt;: 写入数据长度</li> <li>• &lt;write_data&gt;: 写入数据段</li> </ul>
示例	AT+SSAPCWRITEREQ=0,0,2,0,2,0x8899
注意	-

#### 4.2.2.47 客户端发起信息交换

设置指令	AT+SSAPCEXCHINFO=<client_id>,<conn_id>,<mtu_size>,<version>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;client_id&gt;: 客户端 id</li> <li>• &lt;conn_id&gt;: 连接 id (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>• &lt;mtu_size&gt;: ssap 通道 mtu 最大值: 1500</li> <li>• &lt;version&gt;: 版本号</li> </ul>
示例	AT+SSAPCEXCHINFO=0,0,251,1
注意	-

#### 4.2.2.48 设置服务端信息

设置指令	AT+SSAPSSSETINFO=<mtu_size>,<version>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> </ul>

	<ul style="list-style-type: none"> <li>失败：ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;mtu_size&gt;: ssap 通道 mtu 最大值：1500</li> <li>&lt;version&gt;: 版本号</li> </ul>
示例	AT+SSAPCEXCHINFO=251,1
注意	-

#### 4.2.2.49 客户端通过 uuid 发送读请求

设置指令	AT+SSAPCREADBYUUID=<client_id>,<conn_id>,<uuid>,<type>,<start_hdl>,<end_hdl>
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;client_id&gt;: 客户端 id</li> <li>&lt;conn_id&gt;: 连接 id (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>&lt;handle&gt;: 服务端注册服务的属性的 handle, AT+SSAPSSYNCADDPROPERTY 返回的 handle</li> <li>&lt;type&gt;: 客户端类型, 取值: 0/1/3 0: 特征值 1: 属性说明描述符 3: 服务端配置描述符</li> <li>&lt;start_hdl&gt;: 开始 handle</li> <li>&lt;end_hdl&gt;: 结束 handle</li> </ul>
示例	AT+SSAPCREADBYUUID=0,0,0x1234,0,0,0xFFFF
注意	-

#### 4.2.2.50 客户端读取服务端属性数据

设置指令	AT+SSAPCREADREQ=<client_id>,<conn_id>,<handle>,<type>
------	---

响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	<ul style="list-style-type: none"><li>&lt;client_id&gt;：客户端 id（预留参数）</li><li>&lt;conn_id&gt;：连接 id（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li><li>&lt;handle&gt;：服务端注册服务属性的 handle，AT+SSAPSSYNCADDPROPERTY 返回的 handle</li><li>&lt;type&gt;：客户端类型，取值：0/1/3 0：特征值 1：属性说明描述符 3：服务端配置描述符</li></ul>
示例	AT+SSAPCREADREQ=0,0,2,0
注意	读数据时的 handle 需与写入数据时的 handle 一致

#### 4.2.2.51 SLE 断开所有连接

设置指令	AT+SLEDISCONNALL
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>
参数说明	
示例	AT+SLEDISCONNALL
注意	-

#### 4.2.2.52 SLE ssap 数传流控参数设置

设置指令	AT+SSAPPERFORMANCECFG=<send_id>,<total_num>,<send_delay>,<send_interval>
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：ERROR</li></ul>



参数说明	<ul style="list-style-type: none"> <li>• &lt;send_id&gt;: 每个链路数传的 id, 最多支持 10 个链路同时进行数传测试</li> <li>• &lt;total_num&gt;: 每次数传一共发送数据包数量 (0~65535)</li> <li>• &lt;send_delay&gt;: 数传开始之前延迟时间, 单位 ms(0~65535)</li> <li>• &lt;send_interval&gt;: 两个数据包传输之间延迟时间, 单位 sle slot, 一个 sle slot = 0.125ms(0~65535)</li> </ul>
示例	AT+SSAPPERFORMANCECFG=0,10000,100,160
注意	读数据时的 handle 需与写入数据时的 handle 一致

#### 4.2.2.53 SLE ssap 客户端 write\_cmd 数传开始指令

设置指令	AT+SSAPPERFORMANCEWRITE=<send_id>,<client_id>,<conn_id>,<handle>,<type>,<len>,<write_data>
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>• &lt;send_id&gt;: 每个链路数传的 id, 最多支持 10 个链路同时进行数传测试</li> <li>• &lt;client_id&gt;: 客户端 id</li> <li>• &lt;conn_id&gt;: 连接 id (AT+SLECONN 指令执行成功以后, 返回的回调里面打印)</li> <li>• &lt;handle&gt;: 服务端注册服务的属性的 handle, AT+SSAPSSYNCADDPROPERTY 返回的 handle</li> <li>• &lt;type&gt;: 客户端类型, 取值: 0/1/3 0: 特征值 1: 属性说明描述符 3: 服务端配置描述符</li> <li>• &lt;len&gt;: 写入数据长度, 最小值为 0, 最大值为: MTU 值-5(MTU 默认值为 1500)</li> <li>• &lt;write_data&gt;: 写入数据内容</li> </ul>
示例	AT+SSAPPERFORMANCEWRITE=0,0,0,2,0,4,01020304

注意	-
----	---

#### 4.2.2.54 SLE ssap 服务端 indicate&notify 数传开始指令

设置指令	AT+SSAPPERFORMANCENOTIFY=<send_id>,<server_id>,<conn_id>,<handle>,<type>,<value_len>,<value>
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：ERROR</li> </ul>
参数说明	<ul style="list-style-type: none"> <li>&lt;send_id&gt;：每个链路数传的 id，最多支持 10 个链路同时进行数传测试。</li> <li>&lt;server_id&gt;：使用 AT+SSAPSADDSRV 指令进行服务端注册返回的 id 值</li> <li>&lt;conn_id&gt;：连接 ID。（AT+SLECONN 指令执行成功以后，返回的回调里面打印）</li> <li>&lt;handle&gt;：服务端注册服务的属性的 handle，AT+SSAPSSYNCADDPROPERTY 返回的 handle。</li> <li>&lt;type&gt;：SSAP 特征类型 <ul style="list-style-type: none"> <li>0：特征值</li> <li>1：属性说明描述符</li> <li>2：客户端配置描述符</li> <li>3：服务端配置描述符</li> <li>4：格式描述符</li> <li>5：服务管理保留描述符，0x05~0x1F</li> <li>0xFF：厂商自定义描述符</li> </ul> </li> <li>&lt;value_len&gt;：数据长度，最小值为 0，最大值为：MTU 值-5(MTU 默认值为 1500)</li> <li>&lt;value&gt;：数据内容</li> </ul>
示例	AT+SSAPPERFORMANCENOTIFY=0,1,0,17,0,5,0102030405
注意	WS73 1.10.111 及之后版本，服务 handle 从 16 开始，命令可以使用 AT+SSAPSSNDNTFY=0,17,0,2,0x0200

#### 4.2.2.55 SLE ssap 流控客户端清理接收包数量

设置指令	AT+SSAPCCLEANRECVCOUNT
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	
示例	AT+SSAPCCLEANRECVCOUNT
注意	-

#### 4.2.2.56 SLE ssap 流控服务端清理接收数据包数量

设置指令	AT+SSAPSCLEANRECVCOUNT
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	
示例	AT+SSAPSCLEANRECVCOUNT
注意	-

#### 4.2.2.57 SLE ssap 流控回调注册

设置指令	AT+SSAPQOSRECVCBKREG
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: ERROR</li></ul>
参数说明	
示例	AT+SSAPQOSRECVCBKREG
注意	-

# 5 调测相关指令介绍

- 5.1 调测相关指令一览表
- 5.2 维测指令描述
- 5.3 空口调试指令描述
- 5.4 算法配置指令描述
- 5.5 产测指令描述

## 5.1 调测相关指令一览表

特性	指令	描述
维测	latency_stat	设置时延统计功能开关。
	report_latency_stat	查询时延统计结果。
	clear_latency_stat	清除时延统计结果。
	get_channel	查询 vap 信道、频宽信息。
	memoryinfo	查询 Device 侧 Netbuf 内存分配情况。
	get_user_conn_record	查询连接用户的相关统计信息（包括连接失败原因）。
	get_scan_result	查询驱动扫描结果。
	enable_user_rate_info	设置连接用户的速率统计功能开关。
	get_connection_info	查询连接用户的统计信息。

特性	指令	描述
	get_tid_info	查询当前 TID 队列信息。
	service_control_set	设置多业务维测日志功能开关。
	service_control_get	查询当前开启维测功能业务。
	set_cal_rpwr	设置业务输出功率。
	create	创建新的 vap 接口。
	cal_tone	配置单音
空口调试	set_monitor	设置 Wi-Fi Monitor 模式。
	sniffer_save_file	设置 Sniffer 抓包文件大小。
	get_vap_sniffer_info	查询空口实时情况，包括发包占比、空闲占比、干扰占比情况。
	enable_external_record	设置对外交互日志记录功能开关。
	global_frame_switch	设置帧上报功能总开关。
	beacon_frame_switch	设置 Beacon 帧上报开关。
	vip_frame_switch	设置 VIP 帧上报开关。
	80211_frame_switch	设置数据帧或管理帧上报开关。
算法配置	dscr_switch	设置描述符上报开关。
	get_tx_params	查询用户发送速率（根据 mac 地址）。
	rts_threshold	配置 RTS 阈值。
	set_udata_fix_rate	配置 Wi-Fi 固定速率。
	alg_intrf_mode	配置干扰场景优化模式。
	set_udata_rate_mode	配置 Wi-Fi 发送速率模式。
	alg_cfg aggr_max_aggr_num	配置聚合自适应算法最大聚合个数。
	alg_cfg intf_type_enquiry	查询 Wi-Fi 干扰类型。

特性	指令	描述
产测	alg_cfg tpc_fix_tx_pwr	配置固定功率码。
	alg_cfg rts_mode	配置 Wi-Fi 的 RTS 模式。
	al_tx	设置常发参数。
	set_al_tx_ratio	设置常发退避时间参数。
	al_rx	设置常收参数。
	al_rx_info	查询常收收包数统计。
	freq	设置发送频率。
	mode	设置协议模式。
	reginfo	读取当前 vap 下寄存器的值。
	regwrite	设置当前 vap 下寄存器的值。
	set_mfg_mode	配置 WS73 是否进入产测模式。
	get_mfg_status	查询单板工作模式。
	get_al_rx_rssi	查询单板 RSSI 信息。
	set_rssi_offset	配置某一信道 RSSI 偏移量。
	get_rssi_offset	查询某一信道 RSSI 偏移量。
	set_curve_factor	配置功率放大系数。
	get_curve_factor	获取功率放大系数。
	set_tar_power	配置发送目标功率值。
	cali_power	配置发送实际功率值。
	set_curve_param	配置高功率曲线参数。
	set_low_curve_param	配置低功率曲线参数。
	get_curve_param	查询高功率曲线参数。
	get_low_curve_param	查询低功率曲线参数。
	set_xo_trim_coarse	配置频偏粗调校正码值。

特性	指令	描述
	set_xo_trim_fine	配置频偏细调校正码值。
	get_cmu_xo_trim	查询频率校正码值。
	get_temp	查询芯片当前温度。
	check_gpio	查询 GPIO 焊接导通性。
	efuse_write_mfg_flag	配置 eFuse 产测标志位。
	set_efuse_rssi_offset	配置 eFuse RSSI 校准偏移值。
	get_efuse_rssi_offset	查询 eFuse RSSI 校准偏移值。
	efuse_write_power_info	配置 eFuse 功率校准参数。
	efuse_read_power_info	查询 eFuse 功率校准参数。
	efuse_write_cmu_xo_trim	配置 eFuse 频率校正码值。
	efuse_read_cmu_xo_trim	查询 eFuse 频偏校正码值。
	efuse_write_temp	配置 eFuse 产测温度。
	efuse_read_temp	查询 eFuse 产测温度。
	efuse_status	查询 eFuse 产测校准参数（包含 RSSI 校准、功率校准、频率校准、温度等信息）。
	efuse_remain	查询 eFuse 获取剩余组数。
	efuse_write_vid_pid	配置芯片 VID、PID 信息。
	get_dieid	查询芯片 CHIP ID、DIE ID 信息。
	set_jtag_enable	配置 eFuse JTAG 标志位。
	get_jtag_enable	查询 eFuse JTAG 标志位。
	set_ssi_spi_mask	配置 eFuse SSI SPI 标志位。
	get_ssi_spi_mask	查询 eFuse SSI SPI 标志位。
	set_efuse_mac	配置 eFuse MAC 地址。

特性	指令	描述
	efuse_read_mac	查询 eFuse MAC 地址。

## 5.2 维测指令描述

### 5.2.1 设置时延统计功能开关

格式	echo "\$vap_name latency_stat \$enable \$direct \$mode \$trace_cnt" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$enable: 设置时延统计功能开关 0: 关闭 1: 开启</li><li>• \$direct: 设置时延统计方向 0: TX; 1: RX; 2: TX&amp;RX</li><li>• \$mode: 设置时延数据统计模式 0: Trace 模式, 以报文为粒度, 统计每条报文在各个节点间的耗时数据, 与报文方向当统计报文的数量达到 report_cnt 时停止数据统计 1: Stats 模式, 以节点为粒度统计各个流程阶段中报文耗时的分布</li><li>• \$trace_cnt: TRACE 模式时统计报文个数: 1~200</li></ul>
示例	echo "wlan0 latency_stat 1 2 1 50" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-



### 5.2.2 查询时延统计结果

格式	echo "\$vap_name report_latency_stat" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等。
示例	echo "wlan0 report_latency_stat" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.3 清除时延统计结果

格式	echo "\$vap_name clear_latency_stat" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等。
示例	echo "wlan0 clear_latency_stat" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.4 查询 vap 信道、频宽信息

格式	echo "\$vap_name get_channel"> /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等。
示例	echo "wlan0 get_channel"> /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.5 查询 Device 侧 Netbuf 内存分配情况

格式	echo "\$vap_name memoryinfo \$val \$mode" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val: 字符串 "dfx_switch", 表示当 Device 侧 Netbuf 不够会自动打印 device 侧 netbuf 内存池信息</li><li>• \$mode: 打印模式 0: 打印一次内存信息 1: 循环打印内存信息</li></ul>
示例	echo "wlan0 memoryinfo dfx_switch 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.6 查询连接用户的相关统计信息（包括连接失败原因）

格式	echo "\$vap_name get_user_conn_record \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val: 查询对应的统计信息 enable: 开启 disable: 关闭 stat: 连接信息统计 fail: 连接失败原因统计 offline: 掉线原因统计</li></ul>
示例	echo "wlan0 get_user_conn_record enable" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.7 查询驱动扫描结果

格式	echo "\$vap_name get_scan_result" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等
示例	echo "wlan0 get_scan_result" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.8 设置连接用户的速率统计功能开关

格式	echo "\$vap_name enable_user_rate_info \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>\$val: 设置速率统计功能开关 0: 关闭 1: 开启</li></ul>
示例	echo "wlan0 enable_user_rate_info 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.9 查询连接用户的统计信息

格式	echo "\$vap_name get_connection_info" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等
示例	echo "wlan0 get_connection_info" > /sys/ccsys/ccpriv

响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.10 查询当前 TID 队列信息

格式	echo "\$vap_name get_tid_info" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等
示例	echo "wlan0 get_tid_info" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功：OK</li><li>失败：INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.11 设置多业务维测日志功能开关

格式	echo "\$vap_name service_control_set \$mask \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>\$mask: 设置维测日志总配置项, 默认为 0xFFFFFFFF</li><li>\$val: 设置某项维测日志开关<ul style="list-style-type: none"><li>bit0: 发送管理帧维测日志开关</li><li>bit1: 接收管理帧维测日志开关</li><li>bit2: 发送数据帧维测日志开关</li><li>bit3: 接收数据帧维测日志开关</li><li>bit4: 扫描业务维测日志开关</li><li>bit5: 维测日志中滤除 PROBE 报文</li><li>bit6: 报文丢包日志维测开关</li><li>bit7: Wi-Fi Aware 业务日志维测开关</li></ul></li></ul>
示例	echo "wlan0 service_control_set 0xffffffff 0x00000001" >

	/sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.12 查询当前开启维测功能业务

格式	echo "\$vap_name service_control_get" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等
示例	echo "wlan0 service_control_get" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>成功: OK</li><li>失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.2.13 设置业务输出功率

格式	echo "\$vap_name set_cal_rpwr \$protocol \$rate \$offset" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>\$protocol: 设置业务的协议模式 0: 11b 协议 1: 11g 协议 2: 11n/11ax 20M 协议 3: 11n40M 协议</li><li>\$rate: 设置速率索引 11b: 0~3 表示 1, 2, 5.5, 11Mbps; 4 表示全速率修改 11g: 0-7 表示 6, 9, 12, 18, 24, 36, 48, 54Mbps; 8 表示全速率修改 11n/11ax 20M: 0-9 表示 mcs0~mcs9; 10 表示全速率修改</li></ul>

	<p>11n 40M: 0~9 表示 mcs0~mcs9, 10 表示 mcs32; 11 表示全速率修改</p> <ul style="list-style-type: none"> <li>\$offset: 设置功率偏移-100~+40, 单位 0.1dB</li> </ul>
示例	echo "wlan0 set_cal_rpwr 2 6 20" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	- 11n 20M/40M 实际支持到 mcs7。

### 5.2.14 创建新的 vap 接口

格式	echo "Featureid0 create \$vap_name \$mode" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan1 等</li> <li>\$mode: 设置创建 vap 模式, 其中可配置为 ap/sta</li> </ul>
示例	echo "Featureid0 create wlan1 ap" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 5.2.15 配置单音功能

格式	echo "\$vap_name cal_tone \$sw \$freq" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li> <li>\$sw: 设置单音开关 <ul style="list-style-type: none"> <li>0: 关闭</li> <li>1: 开启</li> </ul> </li> <li>\$freq: 设置单音频率 <ul style="list-style-type: none"> <li>范围: (-40000, 40000), 单位: KHz, 表示基于中心频率的偏移频率</li> </ul> </li> </ul>

示例	echo "wlan0 cal_tone 1 5000" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功: OK</li> <li>失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	- 单音功能的命令在常发后使用

## 5.3 空口调试指令描述

### 5.3.1 设置 Wi-Fi Monitor 模式

格式	echo "\$vap_name set_monitor \$switch \$val1 \$val2 \$val3 \$val4" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li> <li>\$switch: 设置 Wi-Fi Monitor 功能开关 <ul style="list-style-type: none"> <li>0: 关闭</li> <li>1: 开启</li> <li>2: 串口输出混杂信息</li> </ul> </li> <li>\$val1: 设置广播数据帧开关 <ul style="list-style-type: none"> <li>0: 关闭</li> <li>1: 开启</li> </ul> </li> <li>\$val2: 设置单播数据帧开关 <ul style="list-style-type: none"> <li>0: 关闭</li> <li>1: 开启</li> </ul> </li> <li>\$val3: 设置广播管理帧开关 <ul style="list-style-type: none"> <li>0: 关闭</li> <li>1: 开启</li> </ul> </li> <li>\$val4: 设置单播管理帧开关 <ul style="list-style-type: none"> <li>0: 关闭</li> <li>1: 开启</li> </ul> </li> </ul>
示例	echo "wlan0 set_monitor 2 1 1 1 1" > /sys/ccsys/ccpriv

响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 5.3.2 设置 Sniffer 抓包文件大小

格式	echo "\$vap_name sniffer_save_file \$switch \$file_num \$file_size">/sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li> <li>\$switch: 设置 Sniffer 抓包功能开关 0: 关闭 1: 开启</li> <li>\$file_num: 设置创建抓包文件数量, 取值范围为 1~15, 单位个</li> <li>\$file_size: 设置创建抓包文件大小, 取值范围为 1~50, 单位 M</li> </ul>
示例	echo "wlan0 sniffer_save_file 1 2 5" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	\$file_num * \$file_size 的取值范围为 1~pcap_file_len_max, 其中 pcap_file_len_max 可在 ws73_cfg.ini 配置文件中配置

### 5.3.3 查询空口实时情况, 包括发包占比, 空闲占比, 干扰占比情况

格式	echo "\$vap_name get_vap_sniffer_info" > /sys/ccsys/ccpriv
参数说明	\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等
示例	echo "wlan0 get_vap_sniffer_info" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-



### 5.3.4 设置对外交互日志记录功能开关

格式	echo "\$vap_name enable_external_record \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val: 设置外交交互日志记录功能开关 0: 关闭 1: 开启</li></ul>
示例	echo "wlan0 enable_external_record 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.3.5 设置帧上报功能总开关

格式	echo "\$vap_name global_frame_switch \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val: 设置帧上报功能开关 0: 关闭 1: 开启</li></ul>
示例	echo "wlan0 global_frame_switch 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.3.6 设置 Beacon 帧上报开关

格式	echo "\$vap_name beacon_frame_switch \$val" > /sys/ccsys/ccpriv
----	---

参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val: 设置 Beacon 帧上报功能开关 0: 关闭 1: 开启</li></ul>
示例	echo "wlan0 beacon_frame_switch 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.3.7 设置 VIP 帧上报开关

格式	echo "\$vap_name vip_frame_switch \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val: 设置 VIP 帧上报功能开关 0: 关闭 1: 开启</li></ul>
示例	echo "wlan0 vip_frame_switch 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"><li>• 成功: OK</li><li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>
注意	-

### 5.3.8 设置数据帧或管理帧上报开关

格式	echo "\$vap_name 80211_frame_switch \$val1 \$val2 \$val3 \$val4 \$val5 \$val6" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>• \$val1: 设置帧上报类型 0: 上报单播帧</li></ul>

	<p>1: 上报组播帧</p> <ul style="list-style-type: none"> <li>• \$val2: 设置帧上报方向</li> <li>0: TX 方向</li> <li>1: RX 方向</li> <li>• \$val3: 设置帧上报子帧类型</li> <li>0: 管理帧</li> <li>1: 数据帧</li> <li>• \$val4: 设置帧上报帧内容开关</li> <li>0: 不上报帧内容</li> <li>1: 上报帧内容</li> <li>• \$val5: 设置帧上报 CB 域开关</li> <li>0: 不上报</li> <li>1: 上报</li> <li>• \$val6: 设置帧上报描述符开关</li> <li>0: 不上报</li> <li>1: 上报</li> </ul>
示例	echo "wlan0 80211_frame_switch 0 1 1 1 1 1" > /sys/ccsys/ccpriv
响应	<ul style="list-style-type: none"> <li>• 成功: OK</li> <li>• 失败: INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

### 5.3.9 设置描述符上报开关

格式	echo "\$vap_name dscr_switch \$val" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>• \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li> <li>• \$val: 设置描述符上报开关</li> <li>0: 关闭</li> <li>1: 开启</li> </ul>
示例	echo "wlan0 dscr_switch 1" > /sys/ccsys/ccpriv

响应	<ul style="list-style-type: none"> <li>成功：OK</li> <li>失败：INPUT_ERROR or CMD_NOT_FOUND</li> </ul>
注意	-

## 5.4 算法配置指令描述

### 5.4.1 查询用户发送速率（根据 mac 地址）

格式	echo "\$vap_name get_tx_params \$mac_addr" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name：需要维测的 vap 名字，通常用 wlan0 等</li> <li>\$mac_addr：需要查询的用户的 mac 地址，为 xx:xx:xx:xx:xx:xx 形式</li> </ul>
示例	echo "wlan0 get_tx_params 12:34:56:78:9a:bc" > /sys/ccsys/ccpriv
响应	串口输出对应的 user_id 和 tx_rate
注意	-

### 5.4.2 配置 RTS 阈值

格式	echo "\$vap_name rts_threshold \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name：需要维测的 vap 名字，通常用 wlan0 等</li> <li>\$value：配置 RTS 周期的阈值，范围：0~2147483647</li> </ul>
示例	echo "wlan0 rts_threshold 20" > /sys/ccsys/ccpriv
响应	串口打印 RTS 的阈值。
注意	<ul style="list-style-type: none"> <li>设置阈值输入非法字符时，阈值会设置为 0</li> <li>只会改变阈值，RTS 的开关能否被阈值控制需要看 RTS 的模式，改为 MIB 模式可以受阈值控制开关</li> </ul> <p>修改模式为 MIB，命令为配置 WI-FI 的 RTS 模式：</p>

```
echo "wlan0 alg_cfg rts_mode mib" > /sys/ccsys/ccpriv
```

### 5.4.3 配置 Wi-Fi 发送速率模式

格式	echo "\$vap_name set_udata_rate_mode \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>\$value: 配置模式 0: auto (自动速率) 1: fixed (固定速率)</li></ul>
示例	echo "wlan0 set_udata_rate_mode [0 1]" > /sys/ccsys/ccpriv
响应	-
注意	<p>第一次改固定速率的时候, 没有默认值, 需要配合配置 Wi-Fi 固定速率参数使用:</p> <pre>echo "\$vap_name set_udata_fix_rate \$protocol_bw \$rate_index" &gt; /sys/ccsys/ccpriv</pre> <p>后续改固定速率会以上一次配置的固定速率参数进行发送</p>

### 5.4.4 配置 Wi-Fi 固定速率参数

格式	echo "\$vap_name set_udata_fix_rate \$protocol_bw \$rate_index" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"><li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li><li>\$protocol_bw: 输入协议带宽的字符串 目前支持输入格式有以下列表, 对应协议和带宽: 11g、11b、11n20M、11n40M、11ax20M、11axer20M、11axer106tone</li><li>\$rate_index: 配置各个协议的速率, 具体字符串如下: 11b: 1M、2M、5.5M、11M 11g: 6M、9M、12M、18M、24M、36M、48M、54M 11n: mcs0、mcs1、mcs2、mcs3、mcs4、mcs5、mcs6、mcs7 11ax/11axer: mcs0、mcs1、mcs2、mcs3、mcs4、mcs5、mcs6、mcs7、mcs8、mcs9</li></ul>

示例	echo "wlan0 set_uctdata_fix_rate 11b 1M" > /sys/ccsys/ccpriv
响应	串口打印协议和带宽。
注意	该命令只是配置固定速率的参数，不会改变发送速率模式，需要配置 Wi-Fi 发送速率模式为固定速率：echo "\$vap_name set_uctdata_rate_mode 1" > /sys/ccsys/ccpriv 后才能生效

#### 5.4.5 配置聚合自适应算法最大聚合个数

格式	echo "\$vap_name alg_cfg aggr_max_aggr_num \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name：需要维测的 vap 名字，通常用 wlan0 等</li> <li>\$value：最大聚合个数。取值范围 1~16</li> </ul>
示例	echo "wlan0 alg_cfg aggr_max_aggr_num 2" > /sys/ccsys/ccpriv
响应	-
注意	原默认值为 16。

#### 5.4.6 查询 Wi-Fi 干扰类型

格式	echo "\$vap_name alg_cfg intf_type_enquiry 1" > /sys/ccsys/ccpriv
参数说明	\$vap_name：需要维测的 vap 名字，通常用 wlan0 等
示例	echo "wlan0 alg_cfg intf_type_enquiry 1" > /sys/ccsys/ccpriv
响应	HSO 及串口输出同频、邻叠频干扰类型及状态机状态
注意	-

#### 5.4.7 配置固定功率码

格式	echo "\$vap_name alg_cfg tpc_fix_tx_pwr \$pow_code" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name：需要维测的 vap 名字，通常用 wlan0 等</li> </ul>

	<ul style="list-style-type: none"> <li>\$pow_code: 功率码, 取值 0~255, 255 表示恢复自动功率</li> </ul>
示例	echo "wlan0 alg_cfg tpc_fix_tx_pwr 7" > /sys/ccsys/ccpriv
响应	-
注意	-

### 5.4.8 配置 Wi-Fi 的 RTS 模式

格式	echo "\$vap_name alg_cfg rts_mode \$arg" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li> <li>\$arg: 配置 RTS 的模式: auto、enable、disable、rank0off、threshold            auto: 速率等级 0 动态调整            enable: 所有速率等级强制开启 RTS            disable: 所有速率等级关闭 RTS            rank0off: 速率等级 0 关闭 RTS            threshold: 所有速率等级的 RTS 开关由 MIB 值 dot11RTSThreshold 来决策</li> </ul>
示例	echo "wlan0 alg_cfg rts_mode enable" > /sys/ccsys/ccpriv
响应	-
注意	如果最后一个参数配成 mib, 可以用配置 RTS 阈值: echo "\$vap_name rts_threshold \$value" > /sys/ccsys/ccpriv 进行改变 dot11RTSThreshold 值

### 5.4.9 配置干扰场景优化模式

格式	echo "\$vap_name alg_intrf_mode \$intrf_mode \$value" > /sys/ccsys/ccpriv
参数说明	<ul style="list-style-type: none"> <li>\$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等</li> <li>\$intrf_mode: 干扰模式, 字符串列表如下:            11b_switch、cca_switch、edca_switch、11n_switch、</li> </ul>

	<div>no_11b_switch、long_range_intrf_switch</div> <div><ul style="list-style-type: none"><li>\$value:</li></ul><div>0: 关闭</div><div>1: 开启</div></div>
示例	<div>echo "wlan0 alg_intrf_mode 11b_switch 1" &gt; /sys/ccsys/ccpriv</div>
响应	<div>-</div>
注意	<div>-</div>

5.5 产测指令描述

说明

以“echo”开头的命令，在“~#”目录或“~#”目录的子目录下执行；以“bpcmd”开头的命令，在“sparklinkctrl>”命令行下执行。

- 上电后初次进入“sparklinkctrl>”命令行配置：“

```
insmod /komod/bp_test.ko
chmod a+x /bin/sparklinkctrl
chmod a+x /bin/bp_channel
/bin/bp_channel &
/bin/sparklinkctrl
```

主控不同、对应 ko 文件和工具文件存放路径不同，则需要将“/komod/”和“/bin/”替换为对应的路径。

- 在“sparklinkctrl>”命令行下，输入“exit”即可退出“sparklinkctrl>”命令行。
- 退出后，若需再次进入“sparklinkctrl>”命令行，只需输入“/bin/sparklinkctrl”，勿重复运行 bp\_channel。

序号	测试命令	命令说明
1	初始化 Wi-Fi（可选）	<div>命令格式</div> <div>ifconfig wlan0 up</div>
2	检测单板启动	<div>检测单板启动完成有两种方式。</div> <div><ul style="list-style-type: none"><li>监控单板 UART 输出特定 log：“WiFi Init OK”</li><li>主动下发单板状态查询命令：</li></ul></div> <div>echo "wlan0 get_mfg_status" &gt; /sys/ccsys/ccpriv</div>



序号	测试命令	命令说明
		输出: "WiFi OK"
3	常发命令	<p><b>关闭常发</b></p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 al_tx 0" &gt; /sys/ccsys/ccpriv</pre> <p><b>关闭速率自适应</b></p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 alg_cfg rate_mode &lt;mode&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <p>&lt;mode&gt;: auto: 自动速率; fixed (默认): 固定速率</p> <p><b>设置协议模式</b></p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 mode &lt;mode&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <p>&lt;mode&gt;: 协议模式: 11b、11g2g20、11n2g20、11n2g40、11ax2g20</p> <p><b>设置频点</b></p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 freq &lt;freq&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <p>&lt;freq&gt;: 频点, 信道号, 2.4G 为 1~14。中国区不支持 14 信道。</p> <p>只有 11b 有信道 14。</p> <p><b>设置单播数据帧的固定速率参数</b></p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 alg_cfg fix_rate bcast_mgmt &lt;rank&gt; &lt;协议模式和带宽&gt; &lt;流数&gt; &lt;速率&gt; &lt;GI&gt; &lt;HE_LTF&gt; &lt;HE_DCM&gt; &lt;前导码格式&gt; &lt;编码方式&gt; &lt;发送次数&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <ul style="list-style-type: none"> <li>- &lt;rank&gt;: rank0、rank1、rank2、rank3, 默认使用 rank0</li> </ul>

序号	测试命令	命令说明
		<ul style="list-style-type: none"> <li>- &lt;协议模式和带宽取值&gt;: 11g、11b、11n20M、11n40M、11ax20M</li> <li>- &lt;流数&gt;: 1ss</li> <li>- &lt;速率&gt;: <ul style="list-style-type: none"> <li>11b: 1M、2M、5.5M、11M</li> <li>11g: 6M、9M、12M、18M、24M、36M、48M、54M</li> <li>11n: mcs0~mcs7</li> <li>11ax: mcs0~mcs9</li> </ul> </li> <li>- &lt;GI&gt;: (默认使用 long, gi 不同, 速率不同) <ul style="list-style-type: none"> <li>11b/11g: long</li> <li>11n: long、short</li> <li>11ax: long、short、mid</li> </ul> </li> <li>- &lt;HE_LTF&gt;: <ul style="list-style-type: none"> <li>11ax: 2x、4x (默认使用 4x)</li> <li>other: 0</li> </ul> </li> <li>- &lt;HE_DCM&gt;: (默认使用 dcm_off) <ul style="list-style-type: none"> <li>11ax: dcm_on、dcm_off</li> <li>Other: dcm_off</li> </ul> </li> <li>- &lt;前导码格式&gt;: <ul style="list-style-type: none"> <li>short_preamble (默认)、long_preamble</li> </ul> <p>特殊: 11b 协议模式发送速率 1M 需要设置前导码格式为 long_preamble</p> </li> <li>- &lt;编码方式&gt;: bcc (默认)、ldpc</li> <li>- &lt;发送次数&gt;: 参数范围 0~7 <ul style="list-style-type: none"> <li>1: 表示发送失败, 硬重传 0 次;</li> <li>2: 表示发送失败, 硬重传 1 次;</li> <li>3: 表示发送失败, 硬重传 2 次。</li> </ul> </li> </ul> <ul style="list-style-type: none"> <li>• 示例 <pre>echo "wlan0 alg_cfg fix_rate bcast_mgmt rank0 11b 1ss 1M"</pre> </li> </ul>

序号	测试命令	命令说明
		<pre>long 0 dcm_off short_preamble bcc 3" &gt; /sys/ccsys/ccpriv</pre> <p><b>设置常发</b></p> <ul style="list-style-type: none"><li>命令格式</li></ul> <pre>echo "wlan0 al_tx &lt;flag&gt; [payload] [len] [tpc_code]" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"><li>参数说明<ul style="list-style-type: none"><li>&lt;flag&gt;:<ul style="list-style-type: none"><li>0: 关闭常发</li><li>1: 打开 RF 测试广播报文</li></ul></li><li>[payload]:<ul style="list-style-type: none"><li>0: 全 0</li><li>1: 全 1 (默认)</li><li>2: 全 1010</li><li>3: 随机值</li></ul></li><li>[len]: payload 的长度; 不大于 1500。</li><li>[tpc_code] (可选参数): 0~146; tpc code 越小, 发送功率越大。</li></ul><p>以版本功率为 23dbm 为例:</p><p>TPC code, 于配置功率的对应关系:</p><p>11g、11n、11ax 的档位计算为: (23dBm-配置功率) × 2。</p><p>11b 的档位计算为: (23dBm-配置功率) × 2 + 74。</p><p>例如 20dBm, 应配置 value 为 6, 11b 为 80, 13dBm 应配置 value 为 20, 11b 为 94。</p></li></ul>
4	常收命令	<p><b>设置常收</b></p> <ul style="list-style-type: none"><li>命令格式</li></ul> <pre>echo "wlan0 al_rx &lt;flag&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"><li>参数说明<ul style="list-style-type: none"><li>&lt;flag&gt;:<ul style="list-style-type: none"><li>0: 关闭</li></ul></li></ul></li></ul>

序号	测试命令	命令说明
		1: 打开
5	关闭常发	关闭常发, 命令格式: echo "wlan0 al_tx 0" > /sys/ccsys/ccpriv
6	关闭常收	关闭常收, 命令格式: echo "wlan0 al_rx 0" > /sys/ccsys/ccpriv
7	常收查询	<ul style="list-style-type: none"> <li>命令格式 echo "wlan0 al_rx_info &lt;flag&gt; &lt;val&gt;" &gt; /sys/ccsys/ccpriv</li> <li>响应 al_rx_info::rx succ num[mpdu,ampdu]:[45713,975] fail num:47959 rssi:-69</li> <li>参数说明 <ul style="list-style-type: none"> <li>&lt;flag&gt;: <ul style="list-style-type: none"> <li>0: 清除</li> </ul> </li> <li>&lt;val&gt;: <ul style="list-style-type: none"> <li>0: 所有数据;</li> <li>1: 总帧数</li> <li>2: self fcs correct</li> <li>3: other fcs correct</li> <li>4: fcs error.</li> </ul> </li> </ul> </li> <li>示例 echo "wlan0 al_rx_info 0 0" &gt; /sys/ccsys/ccpriv al_rx_info::rx succ num[mpdu,ampdu]:[45713,975] fail num:47959 rssi:-69 说明: 接收成功 45713 个 MPDU 报文、975 个 AMPDU 报文, 接收失败 47959 个报文。</li> <li>注意事项 在设置常收命令后执行查询, 在 Host 侧查看打印结果。</li> </ul>
8	设置粗调校正码值	<ul style="list-style-type: none"> <li>命令格式 echo "wlan0 set_xo_trim_coarse &lt;value&gt;" &gt; /sys/ccsys/ccpriv</li> <li>响应</li> </ul>

序号	测试命令	命令说明
		<p>OK 或 ERROR</p> <ul style="list-style-type: none"><li>参数说明: value 范围: 0~15</li><li>示例 echo "wlan0 set_xo_trim_coarse 12" &gt; /sys/ccsys/ccpriv 表示设置粗调校正码值为 12。</li></ul>
9	设置细调校正码值	<ul style="list-style-type: none"><li>命令格式 echo "wlan0 set_xo_trim_fine &lt;value&gt;" &gt; /sys/ccsys/ccpriv</li><li>响应 OK 或 ERROR</li><li>参数说明 value 范围: 0~127</li><li>示例 echo "wlan0 set_xo_trim_fine 60" &gt; /sys/ccsys/ccpriv 表示设置细调校正码值为 60。</li></ul>
10	校正码值回读	<ul style="list-style-type: none"><li>命令格式: echo "wlan0 get_cmu_xo_trim " &gt; /sys/ccsys/ccpriv</li><li>响应 OK 或 ERROR</li><li>示例 echo "wlan0 get_cmu_xo_trim " &gt; /sys/ccsys/ccpriv OK xo_trim_coarse = 8 , xo_trim_fine = 64 表示获取当前粗调与细调校正码值。</li></ul>
11	设定功率放大系数	<ul style="list-style-type: none"><li>命令格式 echo "wlan0 set_curve_factor &lt;flag&gt; [value1] [value2] [value3]" &gt; /sys/ccsys/ccpriv</li><li>响应 OK 或 ERROR</li><li>参数说明 &lt;flag&gt;:</li></ul>

序号	测试命令	命令说明
		<p>0: 使用 ini 默认值, 后面参数不解析。</p> <p>1: 设置高功率放大系数。</p> <p>2: 设置低功率放大系数。</p> <ul style="list-style-type: none"> <li>示例</li> </ul> <pre>echo "wlan0 set_curve_factor 1 25 11 0" &gt; /sys/ccsys/ccpriv</pre> <p>OK</p> <p>说明: 25 11 为 rf 给出的经验值, &lt;value3&gt;为设置的放大系数。</p>
12	设定目标功率	<ul style="list-style-type: none"> <li>命令格式:</li> </ul> <pre>echo "wlan0 set_tar_power &lt;value1&gt; &lt;value2&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>OK 或 ERROR</p> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <ul style="list-style-type: none"> <li>- &lt;value&gt;: 目标功率值, 单位 0.1dBm。</li> </ul> <p>value1: 高功率 (大于或等于 15dBm) 测试设置的目标发射功率。</p> <p>value2: 低功率 (小于 15dBm) 测试设置的目标发射功率。</p> <ul style="list-style-type: none"> <li>- 参数范围: 100~230</li> </ul> <ul style="list-style-type: none"> <li>示例:</li> </ul> <pre>echo "wlan0 set_tar_power 200 130" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>说明: 目标功率严格按照从大到小排列。</li> </ul>
13	下发实际功率给驱动	<ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 cali_power &lt;value1&gt; &lt;value2&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>OK 或 ERROR</p> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <ul style="list-style-type: none"> <li>- &lt;value&gt;: 实际功率值, 单位 0.1dBm。</li> </ul> <p>value1: 高功率 (大于或等于 15dBm) 测试设</p>

序号	测试命令	命令说明
		<p>置的目标发射功率。</p> <p>value2: 低功率 (小于 15dBm) 测试设置的目标发射功率。</p> <ul style="list-style-type: none"> <li>- 参数范围: 0~300</li> <li>• 示例:</li> </ul> <pre>echo "wlan0 cali_power 200 130" &gt; /sys/ccsys/ccpriv</pre> <p>使用说明: 此命令不可单独下发, 需要特定校准流程, 参考《WS73V100 模组产线工装 用户指南》2.2.2 中步骤 6。</p>
14	获取当前放大系数	<ul style="list-style-type: none"> <li>• 命令格式</li> </ul> <pre>echo "wlan0 get_curve_factor" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>• 响应</li> </ul> <p>OK 或 ERROR</p> <ul style="list-style-type: none"> <li>• 示例:</li> </ul> <pre>echo "wlan0 get_curve_factor" &gt; /sys/ccsys/ccpriv OK 26 12 13 0 0 25</pre>
15	获取高低功率补偿值	<p>获取高功率补偿值:</p> <ul style="list-style-type: none"> <li>• 命令格式</li> </ul> <pre>echo "wlan0 get_curve_param" &gt; /sys/ccsys/ccpriv OK 574 -31 58 335 72 15 426 6 55</pre> <ul style="list-style-type: none"> <li>• 响应</li> </ul> <p>OK 或 ERROR</p> <p>获取低功率补偿值:</p> <ul style="list-style-type: none"> <li>• 命令格式</li> </ul> <pre>echo "wlan0 get_low_curve_param" &gt; /sys/ccsys/ccpriv OK 258 120 -18 95 353 -57</pre>

序号	测试命令	命令说明
		<pre>62 384 -61</pre> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>OK 或 ERROR</p>
16	设置功率补偿参数	<p>设置高功率补偿参数：</p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 set_curve_param &lt;flag&gt; &lt;value1&gt; ... &lt;value8&gt; &lt;value9&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>OK 或 ERROR</p> <ul style="list-style-type: none"> <li>参数说明</li> </ul> <ul style="list-style-type: none"> <li>- &lt;flag&gt;:</li> </ul> <p>0: 使用 ini 默认值。</p> <p>1: 使用后面的 9 个参数。</p> <p>&lt;value&gt;: 3 条曲线每个曲线 3 个参数。</p> <p>参数范围: -32768 ~ 32767</p> <p>设置低功率补偿参数：</p> <ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 set_low_curve_param &lt;flag&gt; &lt;value1&gt; &lt;value2&gt; ... &lt;value9&gt;" &gt; /sys/ccsys/ccpriv</pre> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>OK 或 ERROR</p> <ul style="list-style-type: none"> <li>示例</li> </ul> <pre>echo "wlan0 set_curve_param 1 574 -31 58 335 72 15 426 6 55"&gt; /sys/ccsys/ccpriv echo "wlan0 set_curve_param 1 258 120 -18 95 353 -57 62 384 -61"&gt; /sys/ccsys/ccpriv</pre>
17	获取 rssi	<ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 al_rx_info 0 0" &gt; /sys/ccsys/ccpriv al_rx_info::rx succ num[mpdu,ampdu]:[24,0] fail num:10 rssi:-47</pre>
18	获取驱动上报的 RSSI 值, 计算	<ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>echo "wlan0 set_rssi_offset &lt;channel&gt; &lt;value&gt;" &gt; /sys/ccsys/ccpriv</pre>



序号	测试命令	命令说明
	RSSI 偏差值	<ul style="list-style-type: none"> <li>响应 OK 或 ERROR</li> <li>参数说明 <ul style="list-style-type: none"> <li>&lt;channel&gt;: 信道号 2.4G, 参数范围 1~14。</li> <li>&lt;value&gt;: 最大范围±15, 单位 1dB, 同信道多次校准时表示相对目标功率调整的累加值。</li> </ul> </li> <li>示例: echo "wlan0 set_rssi_offset 3 1" &gt; /sys/ccsys/ccpriv</li> </ul>
19	获取 RSSI 功率偏移数据	<ul style="list-style-type: none"> <li>命令格式 echo "wlan0 get_rssi_offset &lt;channel&gt;" &gt; /sys/ccsys/ccpriv</li> <li>响应 OK 或 ERROR</li> <li>参数说明 <ul style="list-style-type: none"> <li>&lt;channel&gt;: 信道号 2.4G, 参数范围 1~14。</li> </ul> </li> <li>注意: channel 需要和当前常发命令配置的参数一致。</li> <li>示例: echo "wlan0 get_rssi_offset 3" &gt; /sys/ccsys/ccpriv</li> </ul>
20	Wi-Fi MAC 值写入 eFuse	<ul style="list-style-type: none"> <li>Wi-Fi MAC 地址写入 eFuse 的命令格式: echo "set_efuse_mac 00,0C,18,EF,FF,ED" &gt; /sys/ccsys/ccpriv</li> <li>读取单板 eFuse MAC 的命令: echo "efuse_read_mac" &gt; /sys/ccsys/ccpriv</li> <li>读取单板 (Wi-Fi) MAC 的命令 (重启后生效): busybox ifconfig wlan0   grep HWaddr</li> </ul>
21	eFuse 写入频偏校正码值	<ul style="list-style-type: none"> <li>频偏校正码值写入 eFuse 命令格式 echo "wlan0 efuse_write_cmu_xo_trim" &gt; /sys/ccsys/ccpriv 响应: OK 或 ERROR</li> <li>读取 eFuse 频偏校正码值命令格式 echo "wlan0 efuse_read_cmu_xo_trim" &gt; /sys/ccsys/ccpriv 响应: OK 或 ERROR</li> </ul>

序号	测试命令	命令说明
22	功率校准参数写入 eFuse	<ul style="list-style-type: none"> <li>功率校准参数写入 eFuse 命令格式  <pre>echo "wlan0 efuse_write_power_info" &gt; /sys/ccsys/ccpriv</pre>                     响应: OK 或 ERROR                 </li> <li>读取 eFuse 功率校准参数命令格式  <pre>echo "wlan0 efuse_read_power_info" &gt; /sys/ccsys/ccpriv</pre> </li> </ul>
23	RSSI 校准参数写入 eFuse	<ul style="list-style-type: none"> <li>RSSI 校准参数写入 eFuse 命令格式  <pre>echo "wlan0 set_efuse_rssi_offset &lt;values1&gt; &lt;values2&gt; &lt;value3&gt;" &gt; /sys/ccsys/ccpriv</pre>                     响应: OK 或 ERROR                 </li> <li>参数说明                      &lt;value&gt;: 最大范围<math>\pm 15</math>, 单位 1dB, 同信道多次校准时表示相对目标功率调整的累加值。                      value1: rssi 校准完成后 3 信道得到的偏移值。                      value2: rssi 校准完成后 7 信道得到的偏移值。                      value3: rssi 校准完成后 11 信道得到的偏移值。                 </li> <li>读取 eFuse RSSI 校准参数命令格式  <pre>echo "wlan0 get_efuse_rssi_offset" &gt; /sys/ccsys/ccpriv</pre>                     响应: OK 或 ERROR                 </li> </ul>
24	写产测标志位	<ul style="list-style-type: none"> <li>命令格式  <pre>echo "efuse_write_mfg_flag" &gt; /sys/ccsys/ccpriv</pre> </li> <li>响应: OK 或 ERROR</li> <li>使用说明: 一个 dut 测试完成后, 执行此命令, 表示进行了一次产测并在 eFuse 中记录</li> </ul>
25	查询产测 eFuse 所有数据	<ul style="list-style-type: none"> <li>查询当前生效所有的校准数据:  <pre>echo "wlan0 efuse_status" &gt; /sys/ccsys/ccpriv</pre> </li> <li>返回实例:  <pre>OK Group left count: 2 Power Calibration Param: Curve_factor,    0    0 11b_constant_offset:    0    0 ofdm_20M_constant_offset:    0    0 ofdm_40M_constant_offset:    0    0 Temp:    0</pre> </li> </ul>

序号	测试命令	命令说明
		Freq Calibration Param: 0 0 Rssi Calibration Param: 0 0 0 Mac Addr: 00:00:00:00:00:00
26	复位单板	<ul style="list-style-type: none"> <li>复位单板命令格式： reboot</li> <li>说明：直接使用系统的重启命令。</li> </ul>
27	切换到产测/业务模式	<ul style="list-style-type: none"> <li>使能产测模式的命令格式： echo "wlan0 set_mfg_mode 0 1" &gt; /sys/ccsys/ccpriv</li> <li>参数说明 0：退出产测模式，进入业务模式； 1：进入产测模式。 说明：调用点在产测程序在启动后，建议先下发使能产测模式命令。单板重启后，自动恢复缺省状态0。</li> </ul>
28	查询 SDK 版本	查询 device 版本命令格式： echo "get_version" > /sys/ccsys/ccpriv 示例： SDK_Version: 1.10.T7 FirmWare_Version:1.10.T7 Chip_Rom_Version:1.10.T7 SDK_Patch_Version:1.0.0 OK.
29	获取芯片 DIEID	<ul style="list-style-type: none"> <li>命令格式 echo "get_dieid"&gt; /sys/ccsys/ccpriv</li> <li>说明 获取芯片 CHIP ID 以及 DIE ID 信息。</li> <li>示例： OK CHIP_ID: 0x00 DIE_ID: 0x00 0000 0000 0000 0000 0000 0000 0000 0000 0000</li> </ul>

序号	测试命令	命令说明
		00
30	读取芯片当前温度 产测温度写入 eFuse	<ul style="list-style-type: none"> <li>读取芯片当前温度的命令格式：  <pre>echo "get_temp" &gt; /sys/ccsys/ccpriv</pre>           输出示例：(成功输出//返回摄氏度)  <pre>OK temperature 56 OK</pre> </li> <li>产测温度写入 eFuse 的命令格式：  <pre>echo "wlan0 efuse_write_temp 56" &gt; /sys/ccsys/ccpriv</pre> </li> <li>从 eFuse 读出产测温度的命令格式：  <pre>echo "wlan0 efuse_read_temp" &gt; /sys/ccsys/ccpriv</pre>           输出示例  <pre>OK 9</pre> </li> <li>说明            输出为温度转换后的温度档位信息，数据输出为温度档位，数据范围为 0~15。温度涵盖范围-40°C ~+120°C，每 10°C为 1 档，共 16 档。计算方法：            输出温度档位值= (写入 eFuse 温度+40) /10。         </li> </ul>
31	环回读取 GPIO 状态，获取 IO 焊接的导通性 (选做)	<ul style="list-style-type: none"> <li>功能说明：            为检测模组焊接电气导通性，需要进行 GPIO 测试。            通过工装基台底座，使用线缆将所有 IO 引出到测试底板，在底板上进行测试。一般进行成对 IO 测试。  <b>连焊测试：</b>成对 IO 不连接，其中一个 GPIO1 设置为输出 A 状态，与之匹配对接的 GPIO2 设置为输入状态，通过读取 GPIO2 的状态判断是否连焊，如果 GPIO1 与 GPIO2 状态相同，则为连焊，不同则不存在连焊。  <b>虚焊测试：</b>成对 IO 相互连接，其中一个 GPIO1 设置为输出 A 状态，与之匹配对接的 GPIO2 设置为输入状态，通过读取 GPIO2 的状态判断是否虚焊，如果 GPIO1 与 GPIO2 状态不相同，则为虚         </li> </ul>

序号	测试命令	命令说明
		<p>焊，相同则不存在虚焊。</p> <ul style="list-style-type: none"><li>命令格式： <code>echo "check_gpio &lt;checkType&gt; &lt;gpio1_id&gt; &lt;gpio2_id&gt;" &gt; /sys/ccsys/ccpriv</code></li><li>参数说明 checkType: 1: 连焊 2: 虚焊 gpio_id: 0~e (十六进制数)</li><li>响应: OK 或 ERROR</li><li>示例 <code>echo "check_gpio 2 0 3" &gt; /sys/ccsys/ccpriv</code> 表示检查 GPIO_0 和 GPIO_3 是否存在虚焊。</li></ul>
32	WiFi 单音指令	<p><b>设置单音</b></p> <ul style="list-style-type: none"><li>命令格式 <code>echo "\$vap_name cal_tone \$sw \$freq" &gt; /sys/ccsys/ccpriv</code></li><li>参数说明 \$vap_name: 需要维测的 vap 名字, 通常用 wlan0 等。 \$sw: 设置单音开关。 0: 关闭; 1: 开启; \$freq: 设置单音频率 范围: (-40000, 40000), 单位: KHz, 表示基于中心频率的偏移频率</li><li>示例 <code>echo "wlan0 cal_tone 1 5000" &gt; /sys/ccsys/ccpriv</code></li><li>响应 成功: OK 失败: INPUT_ERROR or CMD_NOT_FOUND</li></ul>

序号	测试命令	命令说明
		<ul style="list-style-type: none"> <li>注释</li> </ul> <p>单音功能的命令在 WIFI 常发后使用。</p> <ul style="list-style-type: none"> <li>使用说明</li> </ul> <p>商用版本才支持单音功能，产测版本不支持。</p>
<p>以下为 BLE/SLE 产测命令，以 “bpcmd” 开头的命令，需在 “sparklinkctrl&gt;” 命令行下执行。初次进入 “sparklinkctrl&gt;” 命令行配置如下：</p> <pre>insmod /komod/bp_test.ko chmod a+x /bin/sparklinkctrl chmod a+x /bin/bp_channel /bin/bp_channel &amp; /bin/sparklinkctrl</pre> <p>注意：主控不同、对应 ko 文件和工具文件存放路径不同，则需要将 “/komod/” 和 “/bin/” 替换为实际对应的路径。</p>		
33	开启 BLE/SLE	<ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>bpcmd^dev_enable</pre> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>CMD Send OK</p> <ul style="list-style-type: none"> <li>功能说明</li> </ul> <p>启动 BLE/SLE 的 device，开始接收命令。</p> <p>注意：执行本条命令成功后，才能执行后续的 BLE/SLE 测试和校准命令。</p>
34	关闭 BLE/SLE	<ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>bpcmd^dev_disable</pre> <ul style="list-style-type: none"> <li>功能说明</li> </ul> <p>关闭 BLE/SLE，停止接收命令。</p>
35	BLE/SLE 产测初始化	<ul style="list-style-type: none"> <li>命令格式</li> </ul> <pre>bpcmd^bp_init</pre> <ul style="list-style-type: none"> <li>功能说明</li> </ul> <p>设置频偏校准、功率校准的初始值。</p> <ul style="list-style-type: none"> <li>响应</li> </ul> <p>CMD Send OK</p> <p>04 0E 05 01 BD FD val1 val2</p>

序号	测试命令	命令说明
		<ul style="list-style-type: none"><li>- val1: 命令执行状态 00: OK 其他: FAIL</li><li>- val2: 命令号</li></ul> <ul style="list-style-type: none"><li>• 示例 bpcmd^bp_init</li></ul>
36	BLE 常发	<ul style="list-style-type: none"><li>• 命令格式 bpcmd^ble_tx &lt;channel&gt; &lt;data_len&gt; &lt;payload_type&gt; &lt;phy&gt;</li><li>• 参数说明<ul style="list-style-type: none"><li>- channel: 0x00~0x27, 对应 BLE 的 40 个 channel。</li><li>- data_len: 0~0xFF, 表示发送测试包的长度, 单位: Byte。</li><li>- payload_type: 0~7, 表示发送测试包携带的内容。 00: PRBS9 01: '111110000' 02: '10101010' 03: PRBS15 04: '111111111' 05: '00000000' 06: '00001111' 07: '01010101'</li></ul></li><li>- phy: 范围 0x01~0x04, 表示发送测试包使用的物理调试链路。 01: LE 1MPhy 02: LE 2MPhy 03: LE CodedPhy (S=8) 04: LE CodedPhy (S=2)</li></ul>

序号	测试命令	命令说明
		<ul style="list-style-type: none"><li>• 响应 CMD Send OK 04 0E 04 01 34 20 val1 - val1: 命令执行状态, 00: OK 其他: FAIL</li><li>• 示例 bpcmd^ble_tx 00 ff 00 01 表示在 channel0, 1M Phy 发送长度 255, 包内容为 PRBS9 序列的包。</li></ul>
37	BLE 常收	<ul style="list-style-type: none"><li>• 命令格式 bpcmd^ble_rx &lt;channel&gt; &lt;phy&gt; &lt;modulation&gt;</li><li>• 参数说明 - channel: 0x00~0x27, 对应 BLE 的 40 个 channel。 - phy: 0x01~0x03, 监听的物理调试链路。 01: LE 1MPhy 02: LE 2MPhy 03: LE CodedPhy - modulation: 00: standard 01: stable (不支持)</li><li>• 响应 CMD Send OK 04 0E 04 01 33 20 val1 - val1: 命令执行状态 00: OK 其他: FAIL</li><li>• 示例 bpcmd^ble_rx 00 01 00 表示在 channel0, 1MPhy 的物理调制链路, 用标</li></ul>



序号	测试命令	命令说明
		准调试方式来监听测试包。
38	结束 BLE 常发/常收	<ul style="list-style-type: none"> <li>命令格式 bpcmd^ble_trx_end</li> <li>响应 CMD Send OK 04 0E 06 01 1F 20 val1 val2 val3 <ul style="list-style-type: none"> <li>val1: 命令执行状态 00: OK 其他: FAIL</li> <li>val2、val3: 正确收到的包数（见下文示例）</li> </ul> </li> <li>示例 bpcmd^ble_trx_end 串口打印: CMD Send OK 04 0E 06 01 1F 20 00 E8 03 说明：串口上传数据后两字节为正确接收包的数量，十六进制小端字节序表示。即收到包数为 0x03E8 个，十进制表示为收到 1000 个包。</li> </ul>
39	SLE 常发	<ul style="list-style-type: none"> <li>命令格式 bpcmd^sle_tx &lt;channel&gt; &lt;power&gt; &lt;data_len&gt; &lt;payload_type&gt; &lt;phy&gt; &lt;format&gt; &lt;rate&gt; &lt;pilot_ratio&gt; &lt;polar&gt; &lt;interval&gt;</li> <li>参数说明 <ul style="list-style-type: none"> <li>channel: 0x00~0x4E, 对应经典蓝牙 0~78。</li> <li>power: 发送功率档位, 最高 2 个档位(06, 07) 只在 GFSK 调制信号时有效。 00: -6dBm 01: -2dBm 02: 2dBm 03: 6dBm 04: 10dBm 05: 14dBm</li> </ul> </li> </ul>

序号	测试命令	命令说明
		<p>06: 16dBm</p> <p>07: 20dBm</p> <p>说明</p> <p>默认配置下:</p> <p>针对产测版本:</p> <p>GFSK 调制的 0 到 7 档对应功率为: -6、-2、2、6、10、14、16、20dBm;</p> <p>PSK 调制的 0 到 7 档对应功率为: -12、-8、-4、0、4、8、10、14dBm。</p> <p>针对商用版本:</p> <p>GFSK 调制的 0 到 7 档对应功率为: -6、-2、2、6、10、14、16、20dBm;</p> <p>PSK 调制的 0 到 7 档对应功率为: -6、-2、2、6、10、14、14、14dBm。</p> <p>如需修改功率配置, 请参考《WS73V100 单板配置文件 说明书》。</p> <ul style="list-style-type: none"> <li>- data_len: 0x0000~0xFF00, 包长度, 十六进制小端字节序表示, 表示范围: 0~255, 单位: Byte。</li> <li>- payload_type: 包类型</li> </ul> <p>00: PRBS9</p> <p>01: '11110000'</p> <p>02: '10101010'</p> <p>03: PRBS15</p> <p>04: '11111111'</p> <p>05: '00000000'</p> <p>06: '00001111'</p> <p>07: '01010101'</p> <ul style="list-style-type: none"> <li>- phy: 表示发送测试包使用的物理链路</li> </ul> <p>00: 1M PHY</p> <p>01: 2M PHY</p> <p>04: 4M PHY</p>

序号	测试命令	命令说明
		<ul style="list-style-type: none"> <li>- format: 帧格式 00: GFSK 02: 短帧 (short frame, rate 为 QPSK 和 8PSK 时, format 选择短帧)</li> <li>- rate: 调制方式 00: GFSK 02: QPSK 03: 8PSK</li> <li>- pilot_ratio: 导频密度 00: no 01: 1:1 02: 4:1 03: 16:1</li> <li>- polar: 编码方式, 针对 format 取值含义不同 short frame(format=0x02): 00: 1 (no polar) 02: 3/4</li> <li>- interval: 0x0400~0xFFFF 单位: 125μs, 表示两个 packet 之间的发送时间间隔, 十六进制小端字节序表示, 大小范围: 4~65535。根据帧长度选择适当的参数值, 建议填 3200, 即 50×125=6250μs。interval 时间要确保数据能发送出去。</li> </ul> <p>说明: polar 与 pilot_ratio 参数同时配置为 0 或同时不为 0。ws73 射频测试 PSK 调制支持 MCS6 (QPSK, polar = 3/4)、MCS8 (QPSK, polar = 1)、MCS10 (8PSK,polar =3/4)、MCS12 (8PSK,polar =1) 四个速率。</p> <ul style="list-style-type: none"> <li>• 响应 CMD Send OK A2 02 00 04 00 02 FC 01 val1</li> </ul>

序号	测试命令	命令说明
		<ul style="list-style-type: none"> <li>- val1: 命令执行状态 00: OK, 其他: FAIL</li> <li>• 示例 bpcmd^sle_tx 00 07 ff00 00 00 00 00 00 00 3200 表示在 0 号信道, 功率档位为 7, 发送包长度为 255 字节 (说明: 参数中的 ff00 为发送数据长度, 小端字节序表示, 即 0x00ff, 转换成十进制表示为 255), 发送包类型 PRBS9, 1M Phy, GFSK 的 format 和调制方式, 无导频无编码, 发包间隔 6250μs (说明: 参数中的 3200, 小端字节序表示, 即 0x0032, 转换成十进制表示为 50, 即 50 个 slot, sle 的 1 个 slot 为 125μs, 因此时间间隔为 50×125=6250μs)。</li> </ul>
40	SLE 常收	<ul style="list-style-type: none"> <li>• 命令格式 bpcmd^sle_rx &lt;channel&gt; &lt;phy&gt; &lt;format&gt; &lt;pilot_ratio&gt; &lt;interval&gt;</li> <li>• 参数说明 <ul style="list-style-type: none"> <li>- channel: 0x00 ~ 0x4E, 对应 SLE 的 79 个信道, 频点: (2402+1×channel) MHz</li> <li>- phy: 表示发送测试包使用的物理链路 00: 1M PHY 01: 2M PHY 04: 4M PHY</li> <li>- format: 帧格式 00: GFSK 02: 短帧 (short frame)</li> <li>- pilot_ratio: 导频密度 00: no 01: 1:1 02: 4:1</li> </ul> </li> </ul>

序号	测试命令	命令说明
		<p>03: 16:1</p> <ul style="list-style-type: none"> <li>- interval: 0x0400~0xFFFF, 单位: 125μs, 表示两个 packet 之间的发送时间间隔, 十六进制小端字节序表示, 大小范围: 4~65535。和 TX 端的帧间隔设置为一致。根据帧长度选择适当的参数值, 建议填 3200, 即 <math>50 \times 125 = 6250\mu s</math>。</li> </ul> <ul style="list-style-type: none"> <li>• 响应           <p>CMD Send OK</p> <p>A2 02 00 04 00 01 FC 01 val1</p> <p>val1: 命令执行状态</p> <ul style="list-style-type: none"> <li>- 00: OK</li> <li>- 其他: FAIL</li> </ul> </li> <li>• 示例           <pre>bpcmd^sle_rx 00 00 00 00 3200</pre> <p>表示在 0 号信道接收 1M PHY、GFSK 帧格式的包, 包间隔 6250μs (说明: 参数中的 3200, 小端字节序表示, 即 0x0032, 转换成十进制表示为 50, 即 50 个 slot, sle 的 1 个 slot 为 125μs, 因此时间间隔为 <math>50 \times 125 = 6250\mu s</math>)。</p> </li> </ul>
41	结束 SLE 常发/常收	<ul style="list-style-type: none"> <li>• 命令格式           <pre>bpcmd^sle_trx_end</pre> </li> <li>• 响应           <pre>bpcmd^sle_trx_end</pre> <p>CMD Send OK</p> <p>A2 02 00 06 00 03 FC 01 val1 val2 val3</p> <ul style="list-style-type: none"> <li>- val1: 命令执行状态,</li> <li>00: OK</li> <li>其他: FAIL</li> <li>- val2、val3: 正确收到/发出的包数</li> </ul> </li> <li>• 示例           <pre>bpcmd^sle_trx_end</pre> <p>串口打印:</p> </li> </ul>

序号	测试命令	命令说明
		<p>CMD Send OK</p> <p>A2 02 00 06 00 02 FC 01 00 e8 03</p> <p>说明：执行成功，正确接收/发出 0x03e8 个包（1000 个）。</p> <p>注：结束接收测试时，串口上传数据后两字节为正确接收包的数量，十六进制小端字节序表示。</p>
42	频偏校准粗调	<ul style="list-style-type: none"><li>命令格式 bpcmd^xotrim_corse &lt;value&gt;</li><li>参数说明 value: 0 ~ 15, 频偏粗调寄存器值。</li><li>响应 CMD Send OK 04 0E 05 01 BD FD val1 val2<ul style="list-style-type: none"><li>val1: 命令执行状态 00: OK 其他: FAIL</li><li>val2: 命令号</li></ul></li><li>示例: bpcmd^xotrim_corse 8 设置粗调频偏寄存器值为 8。</li></ul>
43	频偏校准细调	<ul style="list-style-type: none"><li>命令格式 bpcmd^xotrim_fine &lt;value&gt;</li><li>参数说明 value: 0 ~ 127, 频偏细调寄存器值。</li><li>响应 CMD Send OK 04 0E 05 01 BD FD val1 val2<ul style="list-style-type: none"><li>val1: 命令执行状态 00: OK 其他: FAIL</li><li>val2: 命令号</li></ul></li></ul>

序号	测试命令	命令说明
		<ul style="list-style-type: none"><li>• 示例: bpcmd^xotrim_fine 63 设置粗调频偏寄存器值为 63。</li></ul>
44	读取芯片当前温度	<ul style="list-style-type: none"><li>• 命令格式 bpcmd^read_temp</li><li>• 响应 CMD Send OK 04 0E 07 01 BD FD val1 val2 val3 val4<ul style="list-style-type: none"><li>- val1: 命令执行状态 00: OK 其他: FAIL</li><li>- val2: 命令号</li><li>- val3、val4: 温度值, 16 进制小端字节序表示</li></ul></li><li>• 示例: bpcmd^read_temp 串口打印: CMD Send OK 04 0E 07 01 BD FD 00 04 1B 00 温度为: 001B, 十进制表示为 27°C。</li></ul>
45	频偏校准值写入 eFuse	<ul style="list-style-type: none"><li>• 命令格式 bpcmd^xotrim_wr_efuse</li><li>• 响应 CMD Send OK 04 0E 05 01 BD FD val1 val2<ul style="list-style-type: none"><li>- val1: 命令执行状态 00: OK 其他: FAIL</li><li>- val2: 命令号</li></ul></li><li>• 示例: bpcmd^xotrim_wr_efuse</li></ul>
46	读频偏校准 eFuse	<ul style="list-style-type: none"><li>• 命令格式</li></ul>

序号	测试命令	命令说明
	值	<p>bpcmd^xotrim_rd_efuse</p> <ul style="list-style-type: none"> <li>• 响应</li> </ul> <p>CMD Send OK 04 0E 07 01 BD FD val1 val2 val3 val4</p> <ul style="list-style-type: none"> <li>- val1: 命令执行状态, 00: OK 其他: FAIL</li> <li>- val2: 命令号</li> <li>- val3: 粗调值, 范围: 00~0F</li> <li>- val4: 细调值, 范围: 00~FF</li> </ul> <ul style="list-style-type: none"> <li>• 示例:</li> </ul> <p>bpcmd^xotrim_rd_efuse</p> <p>串口打印: CMD Send OK 04 0E 07 01 BD FD 00 06 08 3F 粗调值为: 8, 细调值为: 63。</p>
47	产测温度写入 eFuse	<ul style="list-style-type: none"> <li>• 命令格式</li> </ul> <p>bpcmd^temp_wr_efuse &lt;temp&gt;</p> <ul style="list-style-type: none"> <li>• 参数说明</li> </ul> <p>temp: 温度值, 单位: °C, 十进制有符号数表示。</p> <ul style="list-style-type: none"> <li>• 响应</li> </ul> <p>CMD Send OK 04 0E 05 01 BD FD val1 val2</p> <ul style="list-style-type: none"> <li>- val1: 命令执行状态 00: OK 其他: FAIL</li> <li>- val2: 命令号</li> </ul> <ul style="list-style-type: none"> <li>• 示例:</li> </ul> <p>bpcmd^temp_wr_efuse 27</p> <p>将温度 27°C写进 eFuse。</p>
48	从 eFuse 读产测温	<ul style="list-style-type: none"> <li>• 命令格式</li> </ul>



序号	测试命令	命令说明
	度	<p>bpcmd^temp_rd_efuse</p> <ul style="list-style-type: none"> <li>• 响应</li> </ul> <p>CMD Send OK</p> <p>04 0E 06 01 BD FD val1 val2 val3</p> <ul style="list-style-type: none"> <li>- val1: 命令执行状态。</li> </ul> <p>00: OK</p> <p>03: 未写过 efuse, efuse 没读取到值</p> <p>其他: FAIL</p> <ul style="list-style-type: none"> <li>- val2: 命令号。</li> <li>- val3: 温度等级。</li> </ul> <ul style="list-style-type: none"> <li>• 示例:</li> </ul> <p>bpcmd^temp_rd_efuse</p> <p>串口打印:</p> <p>CMD Send OK</p> <p>04 0E 06 01 BD FD 00 08 06</p> <p>温度等级: 6。</p>
49	设置功率曲线常数项放大系数	<ul style="list-style-type: none"> <li>• 命令格式</li> </ul> <p>bpcmd^pwr Cali_set_curve &lt;flag&gt; &lt;value&gt;</p> <ul style="list-style-type: none"> <li>• 参数说明</li> </ul> <ul style="list-style-type: none"> <li>- flag: 放大系数设置选择标志。</li> </ul> <p>0: 采用默认值。</p> <p>1: 将常数项放大系数设置为命令中的设置值。</p> <ul style="list-style-type: none"> <li>- value: 放大系数设置值, 0~15, flag=1 时有效。</li> </ul> <p>注意: BLE/SLE 产线功率校准采用默认常数项放大系数, 本条命令两个参数输入无实际意义, 请输入 0 和 0。</p> <ul style="list-style-type: none"> <li>• 响应</li> </ul> <p>CMD Send OK</p> <p>04 0E 05 01 BD FD val1 val2</p> <ul style="list-style-type: none"> <li>- val1: 命令执行状态。</li> </ul> <p>00: OK</p>

序号	测试命令	命令说明
		<p>其他：FAIL</p> <ul style="list-style-type: none"><li>- val2：命令号。</li></ul> <p>• 示例：</p> <pre>bpcmd^pwr cali_set_curve 0 0</pre> <p>将功率校准常数项系数设为默认值。</p>
50	下发实测功率	<ul style="list-style-type: none"><li>• 命令格式 <pre>bpcmd^pwr cali_set_pwr &lt;target_pwr&gt; &lt;msr_pwr&gt;</pre></li><li>• 参数说明<ul style="list-style-type: none"><li>- target_pwr：目标功率，单位 0.1dBm。</li><li>- msr_pwr：实测功率，单位 0.1dBm。</li></ul></li><li>• 响应 CMD Send OK 04 0E 05 01 BD FD val1 val2<ul style="list-style-type: none"><li>- val1：命令执行状态。 00：OK 其他：FAIL</li><li>- val2：命令号。</li></ul></li><li>• 示例： <pre>bpcmd^pwr cali_set_pwr 200 180</pre><p>下发目标功率 20dBm，实测功率 18dBm。</p></li></ul>
51	读取功率校准补偿值	<ul style="list-style-type: none"><li>• 命令格式 <pre>bpcmd^pwr cali_rd_comp</pre></li><li>• 响应 CMD Send OK 04 0E 08 01 BD FD val1 val2 val3 val4 val5<ul style="list-style-type: none"><li>- val1：命令执行状态。 00：OK 其他：FAIL</li><li>- val2：命令号。</li><li>- val3：放大系数，范围：00~0F。</li><li>- val4、val5：功率校准补偿值，最高 Bit 为符号</li></ul></li></ul>

序号	测试命令	命令说明
		<p>位。</p> <ul style="list-style-type: none"><li>• 示例： bpcmd^temp_rd_efuse 串口打印： CMD Send OK 04 0E 08 01 BD FD 00 0B 0A 5D 00 放大系数：2<sup>10</sup>=1024；补偿值：93 (0x005D)</li></ul>
52	应用功率校准补偿值	<ul style="list-style-type: none"><li>• 命令格式 bpcmd^pwrcali_set_comp</li><li>• 响应 CMD Send OK 04 0E 05 01 BD FD val1 val2<ul style="list-style-type: none"><li>- val1: 命令执行状态。 00: OK 其他: FAIL</li><li>- val2: 命令号。</li></ul></li><li>• 示例： bpcmd^pwrcali_set_comp 将结果下发，进行功率校准流程。</li></ul>
53	将功率校准补偿值写入 eFuse	<ul style="list-style-type: none"><li>• 命令格式 bpcmd^pwrcali_wr_efuse</li><li>• 响应 CMD Send OK 04 0E 05 01 BD FD val1 val2<ul style="list-style-type: none"><li>- val1: 命令执行状态。 00: OK 其他: FAIL</li><li>- val2: 命令号。</li></ul></li><li>• 示例： bpcmd^pwrcali_wr_efuse 软件驱动将保存的校准值写入对应 eFuse 位。</li></ul>

序号	测试命令	命令说明
54	从 eFuse 读功率校准补偿值	<ul style="list-style-type: none"><li>命令格式 bpcmd^pwr Cali_rd_efuse</li><li>响应 CMD Send OK 04 0E 08 01 BD FD val1 val2 val3 val4 val5<ul style="list-style-type: none"><li>val1: 命令执行状态。 00: OK 03: 未写过 efuse, efuse 没读取到值 其他: FAIL</li><li>val2: 命令号。</li><li>val3: 放大系数, 范围: 00 ~ 0F</li><li>val4, val5: 功率校准补偿值, 最高 Bit 为符号位。</li></ul></li><li>示例: bpcmd^pwr Cali_rd_efuse 串口打印: CMD Send OK 04 0E 08 01 BD FD 00 0E 0A 5D 00 方法系数: <math>2^{10}=1024</math>; 补偿值: 93 (0x005D)</li></ul>
55	设置 BT 单音	<ul style="list-style-type: none"><li>命令格式 bpcmd^bt_lo &lt;freq&gt; &lt;mode&gt;</li><li>参数说明<ul style="list-style-type: none"><li>freq: 频点, 取值范围: 0~78, 表示 (2402+freq) Mhz。</li><li>mode: 模式和开关, 0: 发数字 LE 1M 调制; 1: 发 LO 单音; 255: 停止。</li></ul></li><li>响应 CMD Send OK 04 0E 04 01 B7 FD val1</li></ul>

序号	测试命令	命令说明
		<ul style="list-style-type: none"><li>- val1: 命令执行状态。 00: OK 其他: FAIL。</li><li>• 示例: bpcmd^bt_lo 0 1 串口打印: CMD Send OK 04 0E 04 01 B7 FD 00 在 2402MHz 频点发送 LO 单音。</li></ul>
56	退出 sparklinkctrl 命令行	<ul style="list-style-type: none"><li>• 命令格式 exit</li><li>• 命令说明 在 sparklinkctrl 命令行输入, 作用是退出 sparklinkctrl 测试命令行。</li></ul>
57	写入 SLE MAC 地址	<ul style="list-style-type: none"><li>• 命令格式 echo "set_sle_mac xx,xx,xx,xx,xx,xx" &gt; /sys/ccsys/ccpriv</li><li>• 参数说明<ul style="list-style-type: none"><li>- xx,xx,xx,xx,xx,xx: 48 位 16 进制表示的 MAC 地址, 以 "," 分隔。</li></ul></li><li>• 响应 OK 或 ERROR。</li></ul>
58	获取 SLE MAC 地址	<ul style="list-style-type: none"><li>• 命令格式 echo "efuse_read_sle_mac" &gt; /sys/ccsys/ccpriv</li><li>• 命令说明 读取 SLE 的 MAC 地址, 写入后回读。</li><li>• 响应 (以下为示例, 实际 MAC 地址为写入的 MAC) Mac Addr: 11:22:33:44:55:66 OK.</li></ul>
59	获取 BLE MAC 地址	<ul style="list-style-type: none"><li>• 命令格式 echo "get_ble_mac" &gt; /sys/ccsys/ccpriv</li></ul>

2025-02-20

## 6 新增 CCPRIV 命令方法

在 Wi-Fi 系统初始化 proc 系统时，会在 “/sys” 目录下创建 “ccsys/ccpriv” 节点，用户可以输入 “echo xxxxxx > /sys/ccsys/ccpriv”，将命令字符串写入 “/sys/ccsys/ccpriv” 节点中，Wi-Fi 系统会解析命令字符串中的命令字及输入参数，执行对应命令。

当前 CCPRIV 命令框架已成型，且相关代码已开源，客户可针对需求进行新增接口的二次开发，步骤如下：

### 步骤 1 注册命令字及注册相关参数解析函数

命令字及解析函数注册方式可参考 “wal\_linux\_ccpriv.c” 文件的 “g\_ast\_ccpriv\_cmd” 数组，以查询版本信息命令为例，命令字为 “get\_version”，对应参数解析函数为 “uapi\_ccpriv\_get\_version”。

```
OAL_STATIC OAL_CONST wal_ccpriv_cmd_entry_stru g_ast_ccpriv_cmd[] = {  
    /* 设置管制域最大发送功率(可以突破管制域的限制), ccpriv "Featureid0 set_regdomain_pwr_priv 20",单位dBm */  
    {"set_regdomain_pwr_p", uapi_ccpriv_set_regdomain_pwr_priv},  
    {"get_regdomain_pwr", uapi_ccpriv_get_regdomain_pwr},  
    {"adjust_tx_power", uapi_ccpriv_adjust_tx_power},  
    {"restore_tx_power", uapi_ccpriv_restore_tx_power},  
    {"send_custom_pkt", uapi_ccpriv_send_pkt}, /* 发送任意报文命令为:"wlan0 send_custom_pkt data" */  
    {"get_version", uapi_ccpriv_get_version},  
    {"set_soft_retry_num", uapi_ccpriv_set_soft_retry_num}, /* sh ccpriv.sh 'wlan0 set_soft_retry_num 3 5' */  
}
```

### 步骤 2 参数解析函数开发

参数解析流程是通过 “wal\_get\_cmd\_one\_arg” 函数逐一解析获取，在参数解析流程中同时需加上参数合法性校验。

```
/* 获取第一个参数 ch */
ret = wal_get_cmd_one_arg(pc_param, args, OAL_SIZEOF(args), &off_set);
if (ret != OAL_SUCC) {
    oam_warning_log1(0, OAM_SF_ANY, "{uapi_ccpriv_set_extend_ie::parse arg failed [%d]}", ret);
    return ret;
}
pc_param += off_set;
adjust_tx_power->ch = (osal_u8)oal_atoi((const osal_s8 *)args);

/* 获取第二个参数 power */
ret = wal_get_cmd_one_arg(pc_param, args, OAL_SIZEOF(args), &off_set);
if (ret != OAL_SUCC) {
    oam_warning_log1(0, OAM_SF_ANY, "{uapi_ccpriv_set_extend_ie::parse arg failed [%d]}", ret);
    return ret;
}
pc_param = pc_param + off_set;
adjust_tx_power->power = (osal_s8)oal_atoi((const osal_s8 *)args);

ret = wal_sync_post2hmac_no_rsp(wal_util_get_vap_id(net_dev), WLAN_MSG_W2H_CFG_ADJUST_TX_POWER,
    (osal_u8 *)&tx_power, OAL_SIZEOF(tx_power));
if (osal_unlikely(ret != HI_SUCCESS)) {
    oam_warning_log1(0, OAM_SF_ANY, "{uapi_ccpriv_adjust_tx_power::return err code [%d]}", ret);
    return (osal_u32)ret;
}
```

参数解析

向驱动hmac层  
抛消息

### 说明

部分命令需要在设备处于 down 状态执行（比如设置国家码、工作信道等参数），对此类命令进行开发时，在解析参数前需要增加相关限制。

### 步骤 3 注册命令消息，并通过事件形式抛到 hmac 层处理

命令字需要注册对应的事件 ID，相关事件 ID 注册在“wlan\_msg.h”的“wlan\_msg\_w2h\_enum”中，枚举值依次增加。事件消息的填写和调用模式是固定的，可以参考现有机制，只需将事件 ID 修改为对应新增的事件 ID 即可。

```
WLAN_MSG_H2D_C_CFG_SDP_INIT, /* wifi aware, sdp, nan */
WLAN_MSG_H2D_C_CFG_SDP_ADD_PEER_MAC,
WLAN_MSG_H2D_C_CFG_SDP_DW_PRD_TIME_CFG,
WLAN_MSG_H2D_C_CFG_ADJUST_TX_POWER,
WLAN_MSG_H2D_C_CFG_RESTORE_TX_POWER,
WLAN_MSG_H2D_C_CFG_AMPDU_TX_ON,
WLAN_MSG_H2D_C_CFG_CLEAR_WOW_OFFLOAD_INFO, /* 清空wow offload info */
WLAN_MSG_H2D_C_CFG_WOW_PARAM_SYNC, /* WOW启动，同步offload的参数等信息 */
WLAN_MSG_H2D_C_CFG_MAC_ANT_SEL_TX_DSQR, /* mac层tx方向描述符方式配置天线选择参数 */
WLAN_MSG_H2D_C_CFG_APF_EXEC, /* 设置/查询APF过滤规则，强制停止APF过滤 */
WLAN_MSG_H2D_C_CFG_SLP_ENABLE,
WLAN_MSG_H2D_C_CFG_SLP_RM_START,
```

### 步骤 4 注册 hmac 处理函数

注册事件 ID 后，需要在 hmac 层注册对应事件 ID 的处理函数，相应函数钩子会在“wal\_config.c”文件的“wal\_cfg\_init”函数中注册，“hmac\_config\_xxx”函数中就是针对相关命令字的处理流程。



```
#ifdef _PRE_WLAN_CFGID_DEBUG
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_RSSI_LIMIT_CFG, hmac_config_rssi_limit);
#endif
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_QUERY_PSST, hmac_config_query_psst);
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_GET_DIEID, hmac_config_get_dieid);
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_SET_DBM, hmac_config_set_dbm);
#ifdef _PRE_WLAN_FEATURE_DAQ
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_CHIP_TEST_SET_DIAG_PARAM_PHY, hmac_config_chip_test_set_diag_param_phy);
#endif
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_ADJUST_TX_POWER, hmac_config_adjust_tx_power);
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_RESTORE_TX_POWER, hmac_config_restore_tx_power);
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_GET_TSF, hmac_config_get_tsf);
#ifdef _PRE_WLAN_DFR_STAT
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_CHECK_INFO, hmac_get_check_info);
#endif
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_SET_RF_LIMIT_POWER, hmac_config_set_rf_limit_power);
#ifdef _PRE_WLAN_FIT_BASED_REALTIME_CALI
    frw_msg_hook_register(WLAN_MSG_W2H_CFG_DYN_CALI_CFG, hmac_config_set_dyn_cali_param);
#endif
```

步骤 5 CCPRIV 有部分命令需要 Device 侧负责处理，此时通过注册 DMAC 侧消息并抛到 Device 侧进行处理，Device 侧消息处理逻辑闭源。

----结束