



# 射频性能测试 使用手册

版本： 2.25

版权 @ 2021

[www.bouffalolab.com](http://www.bouffalolab.com)

1 版本记录 . . . . .	7
2 概述 . . . . .	9
3 下载开发烧录软件工具包 . . . . .	11
4 烧写/下载测试固件 . . . . .	12
4.1 IOT 模组 (带 Flash) 烧写测试固件 . . . . .	12
4.2 SDIO 模组 (不带 Flash) 下载测试固件 . . . . .	15
4.2.1 模组带 UART 接口 . . . . .	15
4.2.2 模组不带 UART 接口 . . . . .	17
5 运行测试固件 . . . . .	20
6 频偏补偿设置 . . . . .	21
7 发送设置 . . . . .	23
7.1 Channel 和 Power 设置 . . . . .	23
7.2 发送数据包模式设置 . . . . .	25
7.2.1 11b 数据包发送 . . . . .	25
7.2.2 11g 数据包发送 . . . . .	25
7.2.3 11n 数据包发送 . . . . .	25
8 接收设置 . . . . .	26
9 BLE 测试 . . . . .	27
10 PDS/DTIM 设置 . . . . .	28
11 HBN 设置 . . . . .	29
12 CW 测试模式 . . . . .	30
12.1 WiFi CW 测试 . . . . .	30
13 发送功率温度补偿 . . . . .	31

14 RF 测试固件在量产中的使用说明	32
14.1 IOT 模组 (带 Flash)	32
14.1.1 生成烧写文件	32
自启动固件的工作流程	32
普通固件的工作流程	32
14.1.2 工厂烧录以及产测	33
自启动固件	33
普通固件	33
14.2 SDIO 模组 (不带 Flash)	34
14.2.1 设置产测模式	34
14.2.2 将固件保存到树莓派中	34
14.2.3 工厂产测	35
14.3 校准参数存储说明	35
15 使用 DTS 文件设置 RF 相关参数	36
15.1 DTS 文件介绍	36
15.2 设置功率温度补偿	37
15.2.1 概述	37
15.2.2 DTS 文件中温补参数说明	37
15.2.3 测试获得温补参数	38
15.2.4 示例	38
15.3 验证 RF 产测参数	38
15.4 验证功率校准数值	39
15.5 验证频偏校准数值	40
15.6 训练功率和频偏经验值	41
16 串口通信命令	42
16.1 Shakehand	42
16.2 TX on/off	42
16.3 TX modulation	42
16.3.1 2.4G 11n	42
16.3.2 2.4G 11g	43
16.3.3 2.4G 11b	43
16.4 2.4g channel	43
16.5 2.4g tx power	43
16.6 TX frame length	44
16.7 TX frequency	44
16.8 PDS	44
16.9 HBN	44
16.10 RX	44
16.11 Get MFG FW version	44

16.12 Get MFG FW building infomation . . . . .	45
16.13 Get current power level . . . . .	45
16.14 Get current channel . . . . .	45
16.15 Get current tx status . . . . .	45
16.16 Get tx frequency . . . . .	45
16.17 Get cap code . . . . .	45
16.18 Get MFG mode . . . . .	45
16.19 Set cap code . . . . .	46
16.20 Set MFG Test(CW) mode . . . . .	46
16.21 Write data to efuse . . . . .	46
16.21.1 Write data to efuse buffer . . . . .	46
16.21.2 Load data from efuse buffer . . . . .	46
16.21.3 Program data to efuse . . . . .	47
16.21.4 Read data from efuse . . . . .	47
16.22 Save calibration parameters to efuse . . . . .	47
16.22.1 Write cap code to efuse buffer . . . . .	47
16.22.2 Load cap code from efuse buffer . . . . .	48
16.22.3 Program cap code to efuse . . . . .	48
16.22.4 Read cap code from efuse . . . . .	48
16.22.5 Write power offset to efuse buffer . . . . .	48
16.22.6 Load power offset from efuse buffer . . . . .	48
16.22.7 Program power offset to efuse . . . . .	49
16.22.8 Read power offset from efuse . . . . .	49
16.22.9 Enable power offset in efuse . . . . .	49
16.22.10 Write mac address to efuse buffer . . . . .	49
16.22.11 Load mac address from efuse buffer . . . . .	49
16.22.12 Program mac address to efuse . . . . .	49
16.22.13 Read mac address from efuse buffer . . . . .	50
16.23 Save calibration parameters to flash . . . . .	50
16.24 Reset MFG FW . . . . .	50
16.25 Set tx duty . . . . .	51
16.26 Get tx duty . . . . .	51
16.27 BLE Test . . . . .	51
16.27.1 BLE TX Power . . . . .	51
16.27.2 BLE TX . . . . .	51
16.27.3 BLE RX . . . . .	51
16.27.4 BLE test stop . . . . .	52

## List of Figures

2.1 工具界面图 . . . . .	9
3.1 开发烧录软件工具包 . . . . .	11
4.1 IOT 模组评估套件 . . . . .	12
4.2 烧写界面 . . . . .	13
4.3 烧录成功界面 . . . . .	15
4.4 烧写成功界面 . . . . .	16
4.5 下载平台示意图 . . . . .	17
4.6 烧写成功界面 . . . . .	19
5.1 程序成功运行的 log . . . . .	20
6.1 更新补偿值 . . . . .	22
7.1 设置 Channel 和 Power 的参数 . . . . .	23
7.2 11b 数据包设置速率 . . . . .	25
7.3 11g 数据包设置速率 . . . . .	25
7.4 11n 数据包发送界面 . . . . .	25
8.1 接收数据包效果图 . . . . .	26
9.1 BLE Payload Type . . . . .	27
10.1 PDS 模式参数设置 . . . . .	28
11.1 HBN 模式参数设置 . . . . .	29
12.1 WiFi 测试模式参数设置 . . . . .	30
14.1 烧写界面 . . . . .	33
14.2 固件保存成功界面 . . . . .	34

15.1 温补参数说明 . . . . .	37
15.2 示例 . . . . .	38
15.3 使能 Power Offset 的 log 信息 . . . . .	39
15.4 勾选 Auto 后的 log 信息 . . . . .	40

表 1.1: 修改记录

版本	更新内容
V2.25	修改 mfg bin 下载方式
V2.24	修改 bin 文件的路径
V2.23	增加 SDIO 模组的下载和量产使用说明
V2.22	增加 DTS 文件的命令发送说明
V2.21	修改 MFG bin 文件的路径
V2.20	添加温补参数相关使用说明
V2.19	添加 RAM 版本 (SDIO 透传模组)MFG 固件的相关使用说明
V2.18	更新评估板硬件描述
V2.17	增加射频性能测试固件与应用固件的切换说明
V2.16	增加射频性能测试在生产中的使用说明
V2.15	增加 EFUSE 读写的相关描述
V2.14	增加温补功能的相关描述
V2.13	增加将产测校准参数写入 Flash 功能
V2.12	增加启用 efuse tx power offset 校准功能命令
V2.11	增加设置 BLE Tx Power 及 default 值功能
V2.10	增加 BLE 和 Auto(Default) TX power 功能
V2.9	将 MFG 功能集成到开发烧录工具中
V2.8	增加 TX Duty 选择, 去掉固件下载功能
V2.7	增加 CW 测试模式, 在该测试模式下, 只能设定 Power 和 Channel
V2.6	去掉 Ch14 的设定, 增加 Misc Cfg Get 功能获取 Channel, Power, Capcode

表 1.1: 修改记录

版本	更新内容
V2.5	增加 Shakehand (H) 命令, 增加读取 efuse power offset 和 Cap code 并设定功能
V2.4	增加 Reset 命令
V2.3	增加 Efuse Load 和 Save 指令, 用于保存校准参数到 Efuse 时候的校验
V2.2	去除 MFG 图形界面工具中数据包长, 发送频率, 制式参数等冗余设定, 简化用户使用步骤



RF 性能测试工具 (RF MFG) 是 Bouffalo Lab 提供的用于 RF 评估测试的工具, 包含测试工具和测试镜像 (MFG Firmware) 两部分。测试工具界面如下图所示。

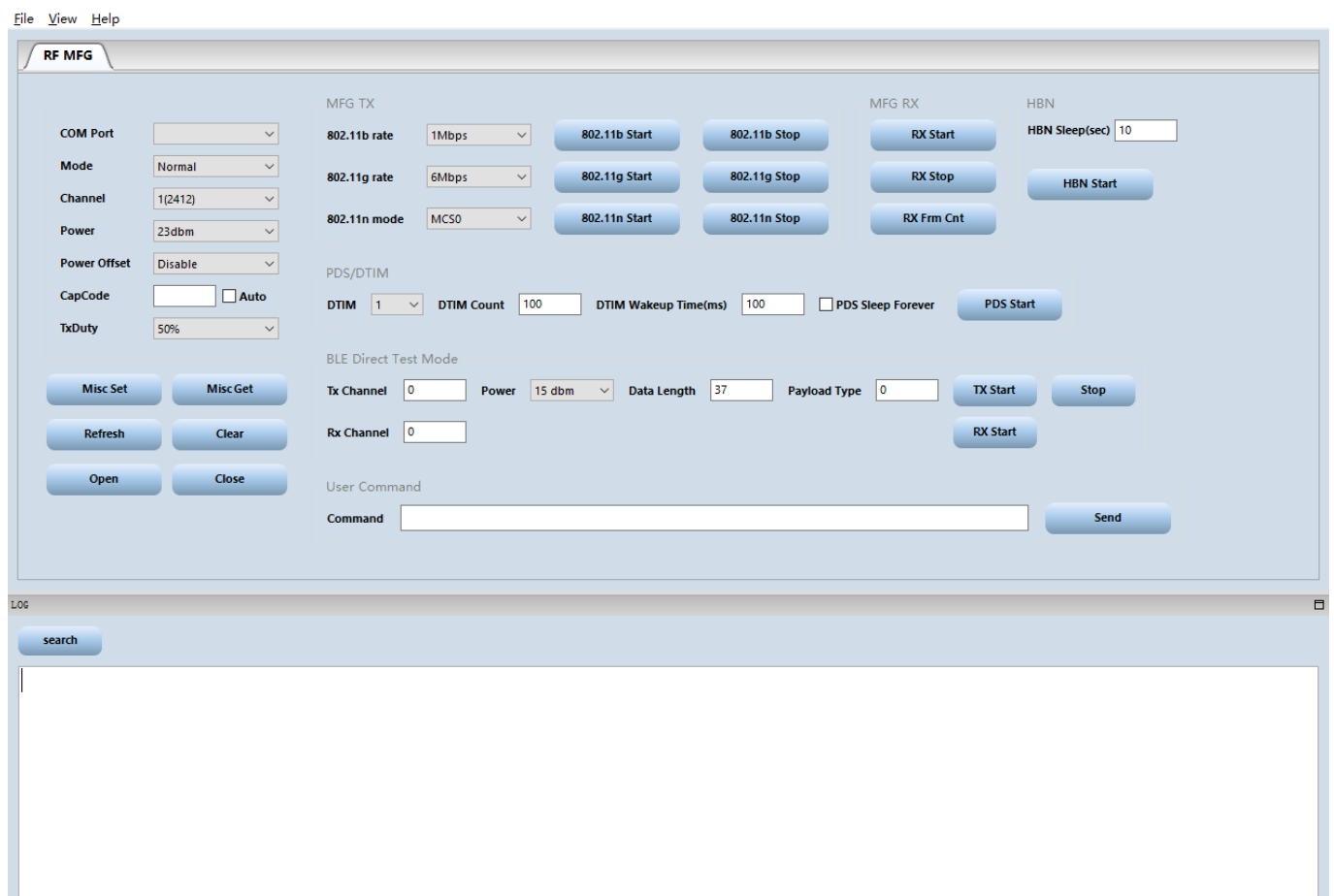


图 2.1: 工具界面图

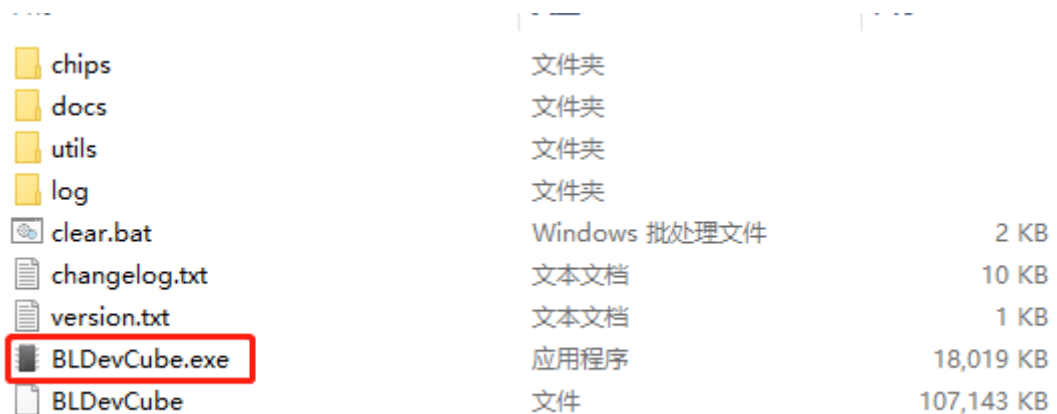
RF 性能测试工具 (RF MFG) 可以实现的功能包括:

- WiFi 数据包发送

- WiFi 数据包接收
- BLE 数据包发送 (具有 BLE 功能的芯片)
- BLE 数据包接收 (具有 BLE 功能的芯片)
- 芯片 PDS(Power down sleep) 测试
- 芯片 HBN(Hibernate) 测试
- RF 产测
- RF 校准验证
- 训练功率和频偏经验值

## 下载开发烧录软件工具包

如果用户没有开发烧录工具，可以通过 [Bouffalo Lab Dev Cube](#)，获取开发烧录软件工具包，在该工具包中，包含了 RF 测试固件，RF 测试固件烧写工具，RF 测试工具等。工具包解压后的效果如图所示。它是客户开发博流各种类型芯片使用到的工具集。



chips	文件夹	
docs	文件夹	
utils	文件夹	
log	文件夹	
clear.bat	Windows 批处理文件	2 KB
changelog.txt	文本文档	10 KB
version.txt	文本文档	1 KB
BLDevCube.exe	应用程序	18,019 KB
BLDevCube	文件	107,143 KB

图 3.1: 开发烧录软件工具包

## 4.1 IOT 模组 (带 Flash) 烧写测试固件

将芯片 Boot 引脚跳到高电平，使其从 UART 启动，即可通过 Flash 烧写工具，将 RF 的测试固件烧写到 Flash 中。测试固件成功烧录后，将 Boot 引脚跳到低电平，按下复位键就可以启动测试固件程序进行 RF 测试了。

在开发烧录软件工具包 (Bouffalo Lab Dev Cube) 中，包含了各个芯片的 RF 测试固件。下面以 BL602 的 IOT 模组评估套件烧写为例，介绍烧写过程。

BL602 的 IOT 模组评估套件 BL602\_IoT\_DVK-3S 如下图所示。

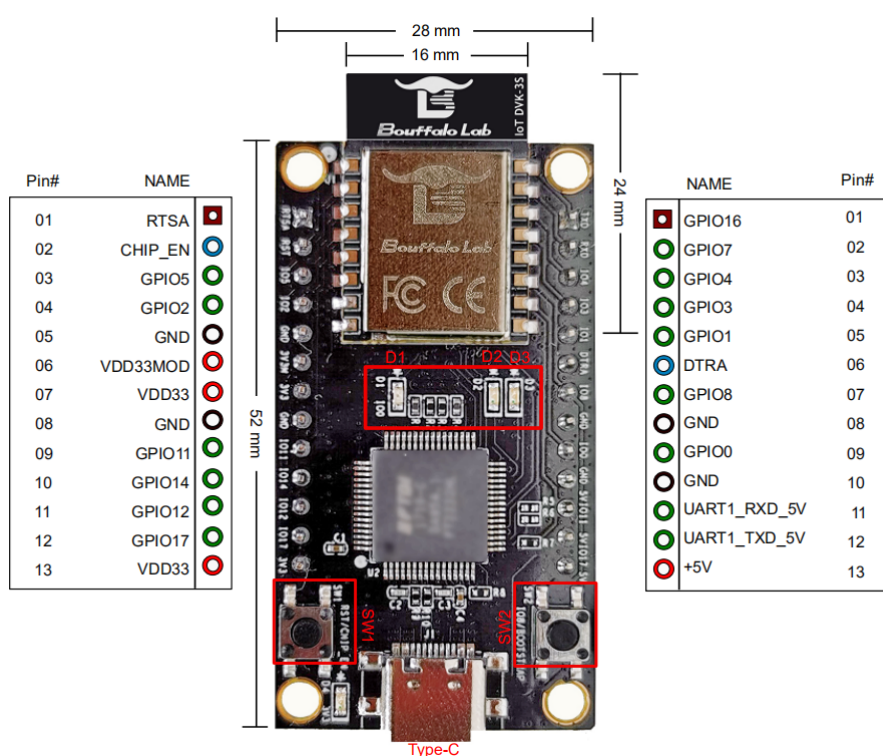


图 4.1: IOT 模组评估套件

套件由 IOT 模组和模组底板组成，模组底板使用 USB Type-C 接口供电同时带有一颗 FT 的 USB 转串口芯片，USB 转串口与模组的连接关系是：

- TXD: 与模组的 RXD 相连
- RXD: 与模组的 TXD 相连

当模组连接到 PC 后，会在 PC 的设备管理器出现两个 USB 转串口，并且这两个 COM 号是相邻的，与芯片的 UART 连接的是其中小号的串口。如果模组连接到 PC 后，没有自动安装驱动，请到<https://www.ftdichip.com/Drivers/VCP.htm>下载驱动自行安装。

连接模组后，首先运行 BLDevCube.exe, 在 Chip Type 中选择 BL602/604，进入如下烧写界面。

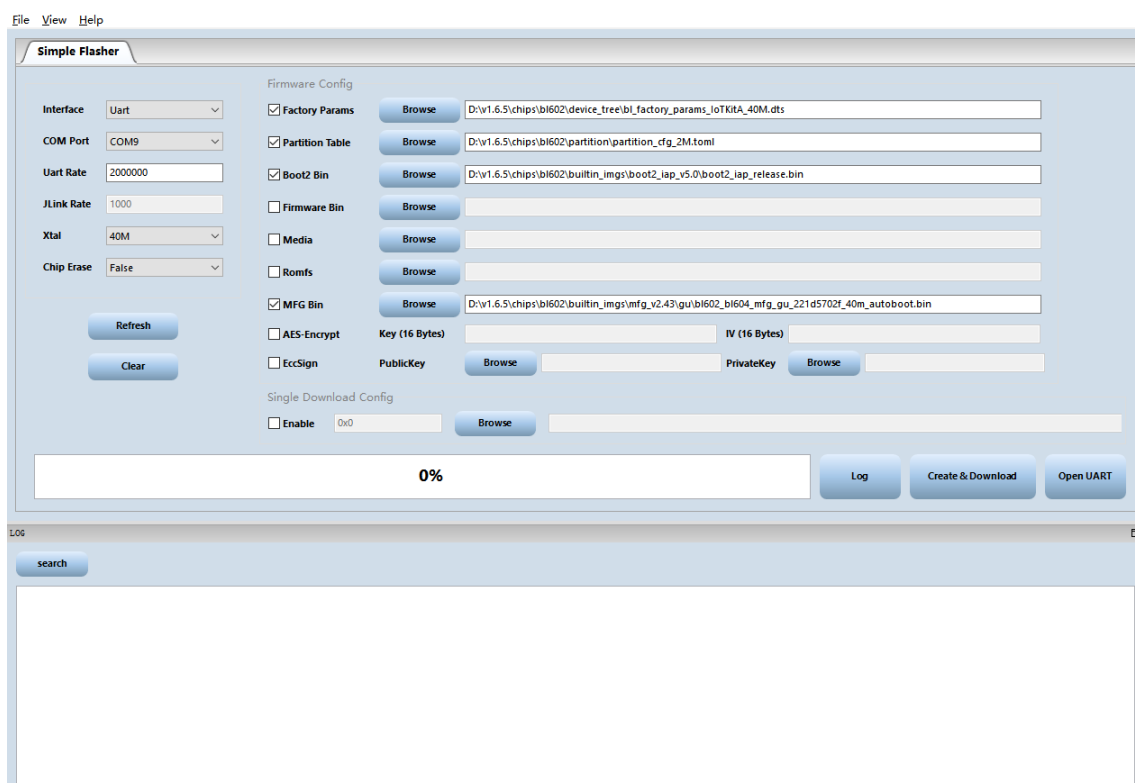


图 4.2: 烧写界面

在左侧通信接口设置中：

- **Interface:** 用于选择烧录的通信接口，这里选择 **uart** 进行烧写
- **COM Port:** 当选择 **UART** 进行烧写的时候这里选择与芯片连接的 **COM** 口号，可以点击 **Refresh** 按钮进行 **COM** 号的刷新
- **Uart Rate:** 当选择 **UART** 进行烧写的时候，填写波特率，可以填写 **2M** 即 **2000000**
- **Board:** 选择所使用的板子型号，这里选择 **IoTKitA**，当板子选定后，**Xtal** 和 **Chip/Flash** 会自动更新成与板子匹配的默认值，当然用户也是可以再次更改的
- **Xtal:** 用于选择板子所使用的晶振类型，对于评估板，这里选择 **40M**

- **Chip Erase:** 默认设置为 **False**，即下载时不擦除 Flash

其它项使用默认配置即可。

在右侧烧录镜像配置，分别选择：

- **分区表:** 使用烧写工具目录下的对应芯片型号 **partition** 目录下的分区表，本例中使用

`bl602/partition/partition_cfg_2M.toml`

- **Boot2:** 使用烧写工具目录下的对应芯片型号 **builtin\_imgs** 目录下的 **Boot2**，本例中使用

`bl602/builtin_imgs/boot2_iap_v5.0/boot2_iap_release.bin`

- **MFG Bin:** 在烧写工具目录下的对应芯片型号 **builtin\_imgs** 目录下的 **mfg** 文件夹中，选择对应的 **Flash** 版本固件。  
文件选择的是 `bl602/builtin_imgs/mfg_vx.xx/gu/mfg_gu_xxxxxxxx_40m_autoboot.bin`，其中 **x.xx** 和 **xxxxxxxx** 分别为版本号和 **commit** 号

---

注解：**MFG Bin** 需要选择烧录后自启动的固件 (带有 **autoboot** 后缀)，自启动固件有启动 **flag**，产测结束后工具发“ATSC”命令会清除产测固件 **flag**，重新启动时，**boot2** 检测到 **mfg** 固件的启动 **flag** 会优先启动 **mfg**，没有启动 **flag** 则会启动应用固件，这样设计的目的是避免产测结束后再次烧录程序，提高生产效率。

---

根据上述配置，设置好 **Dev Cube** 以后，将芯片配置成 **UART** 启动模式，即可开始烧写。

将芯片配置成 **UART** 启动模式方法如下：

- 按住模组上的按键 **SW2** 不松，同时按一下按键 **SW1**
- 松开 **SW1** 和 **SW2**

完成上述芯片启动设定后，点击 **Create&Download** 按钮，完成固件程序的烧录。烧录成功的示意如下。

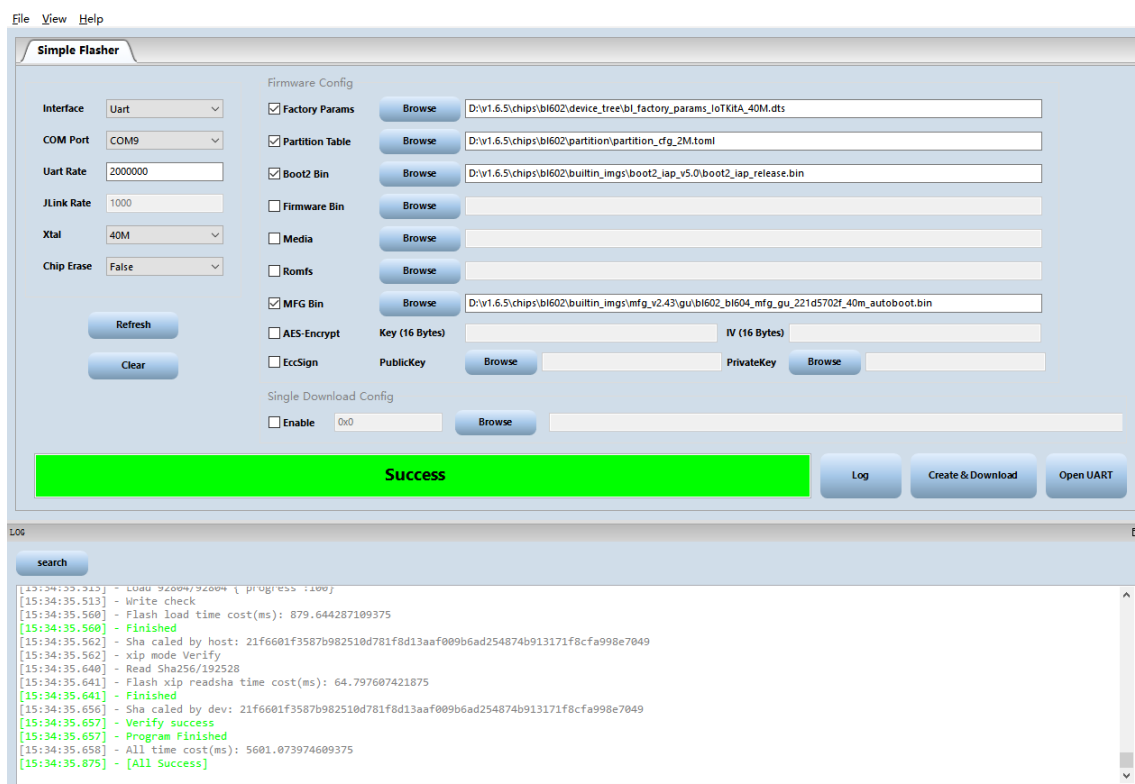


图 4.3: 烧录成功界面

## 4.2 SDIO 模组 (不带 Flash) 下载测试固件

### 4.2.1 模组带 UART 接口

如果模组带有 UART 接口, 可以使用 UART 口进行测试固件的下载。下载方式如下:

#### 1. 设置芯片启动方式

将芯片 Boot 引脚跳到高电平, 从 UART/SDIO 启动, 按下复位按键后, 芯片即可从 UART/SDIO 启动。

#### 2. 通过 Dev Cube 下载 MFG 固件

在 BLDevCube 界面, 通过 View->MCU 进入到 MCU 界面。

在左侧通信接口设置中:

- **Interface:** 用于选择下载的通信接口, 这里选择 **uart** 进行下载
- **COM Port:** 当选择 UART 进行下载的时候这里选择与芯片连接的 COM 口号, 可以点击 **Refresh** 按钮进行 COM 号的刷新
- **Uart Rate:** 当选择 UART 进行下载的时候, 填写波特率, 可以填写 500000
- **Xtal:** 用于选择板子所使用的晶振类型, 对于评估板, 这里选择 40M(SDIO/UART 下载无须关心)
- **Chip Erase:** 默认设置为 **False**, 即下载时不擦除 Flash(SDIO/UART 下载无须关心)

其它项使用默认配置即可。

在右侧烧录镜像配置，分别选择：

- **Boot Source:** 选择 UART/SDIO
- **Image Address:** 0x22008000
- **Image File:** 在烧写工具目录下的对应芯片型号 `builtin_imgs` 目录下的 `mfg` 文件夹中，选择对应的 **RAM** 版本固件。文件选择的是 `bl602/builtin_imgs/mfg_vx.xx/ram/mfg_ram_xxxxxxxx_40m.bin`，其中 `x.xx` 和 `xxxxxxxx` 分别为版本号 and `commit` 号
- **Device Tree:** 非必须下载，客户可先参考《使用 DTS 文件设置 RF 相关参数》章节，了解其功能后进行选择是否下载。如需下载，在烧写工具目录下的对应芯片型号 `device_tree` 目录下，选择对应晶振的配置文件即可。本例中选择的是 `bl602/device_tree/bl_factory_params_loTKitA_40M.dts`

完成上述设定后，点击 **Create&Download** 按钮，完成固件程序的下载。下载成功的示意如下。

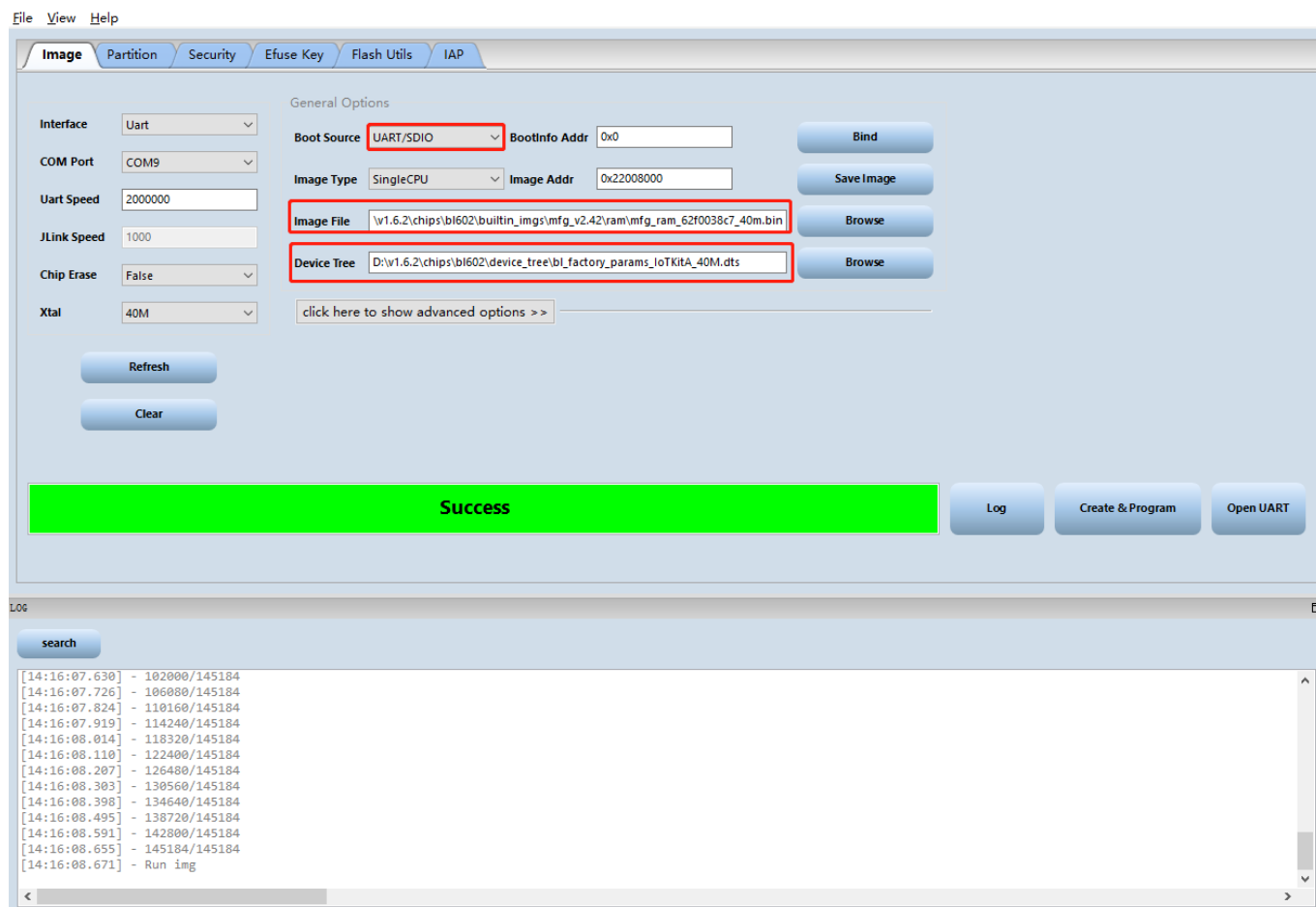


图 4.4: 烧写成功界面



### 4.2.2 模组不带 UART 接口

对于只有 SDIO 接口模组的 RF 性能测试评估，产测参数验证以及在不做产测情况下训练 RF 参数，我们提供了基于树莓派的下载平台。平台示意图如下：

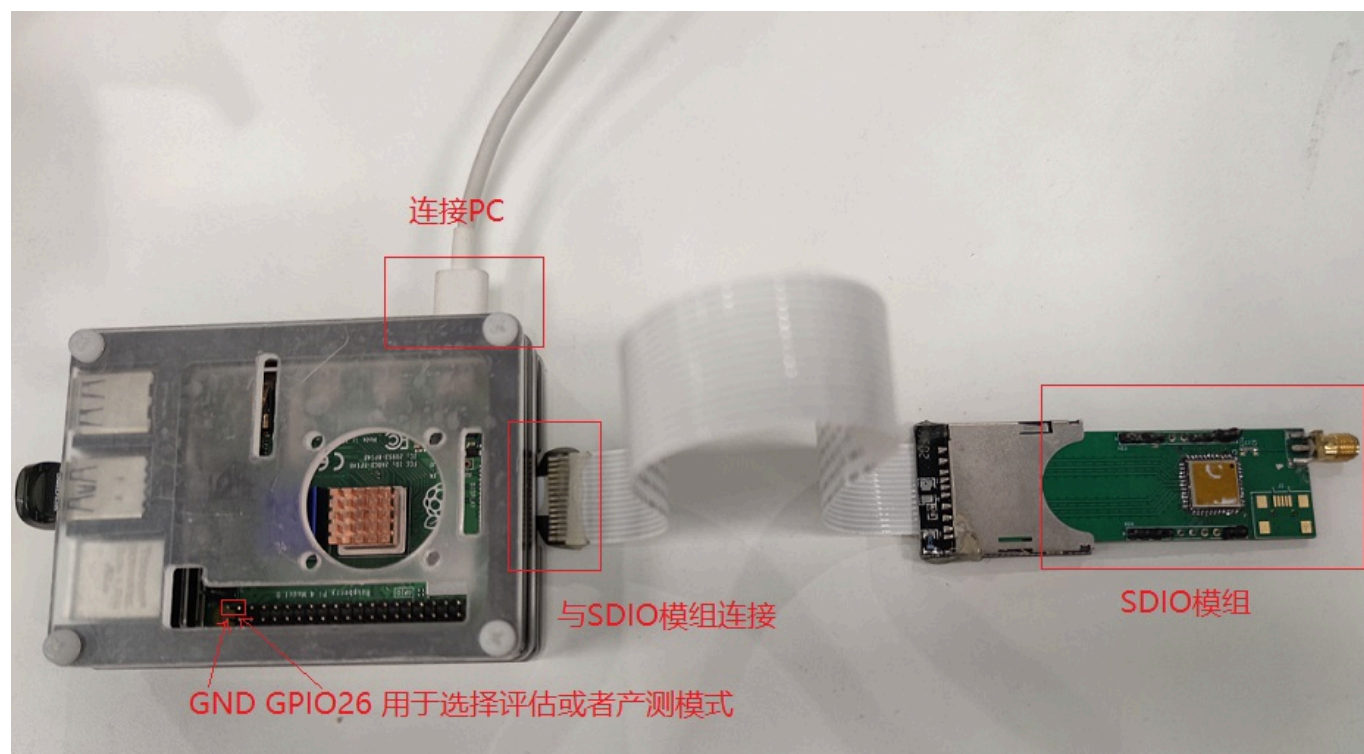


图 4.5: 下载平台示意图

下载步骤如下：

#### 1. 设置下载模式

下载模式分为评估和产测两种模式。评估模式时，将树莓派中 GPIO26 引脚悬空。产测模式时，将 GPIO26 引脚跳到低电平。当前下载选择评估模式。

#### 2. 将 SDIO 模组插入 SD card 接口

#### 3. 树莓派通过 USB 口连接 PC，此时树莓派会将自己枚举成 USB 转串口（树莓派启动时间大约 30s）

#### 4. 通过 Dev Cube 下载 MFG 固件

在 BLDevCube 界面，通过 View->MCU 进入到 MCU 界面。

在左侧通信接口设置中：

- **Interface:** 用于选择下载的通信接口，这里选择 Uart 进行下载
- **COM Port:** Dev Cube 会自动识别树莓派枚举出的串口，这里选择该串口，如果没有，可以点击 Refresh 按钮进行 COM 号的刷新

- **Uart Rate:** 当选择 **UART** 进行下载的时候, 填写波特率, 可以填写 **2000000**
- **Xtal:** 用于选择板子所使用的晶振类型, 对于评估板, 这里选择 **40M**(**SDIO/UART** 下载无须关心)
- **Chip Erase:** 默认设置为 **False**, 即下载时不擦除 **Flash**(**SDIO/UART** 下载无须关心)

其它项使用默认配置即可。

在右侧烧录镜像配置, 分别选择:

- **Boot Source:** 选择 **UART/SDIO**
- **Image Address:** **0x22008000**
- **Image File:** 在烧写工具目录下的对应芯片型号 **builtin\_imgs** 目录下的 **mfg** 文件夹中, 选择对应的 **RAM** 版本固件。文件选择的是 **bl602/builtin\_imgs/mfg\_vx.xx/ram/mfg\_ram\_xxxxxxxx\_40m.bin**, 其中 **x.xx** 和 **xxxxxxxx** 分别为版本号 and **commit** 号
- **Device Tree:** 非必须下载, 客户可先参考《使用 **DTS** 文件设置 **RF** 相关参数》章节, 了解其功能后进行选择是否下载。如需下载, 在烧写工具目录下的对应芯片型号 **device\_tree** 目录下, 选择对应晶振的配置文件即可。本例中选择的是 **chips/bl602/device\_tree/bl\_factory\_params\_loTKitA\_40M.dts**

完成上述设定后, 点击 **Create&Download** 按钮, 完成固件程序的下载。下载成功的示意如下。

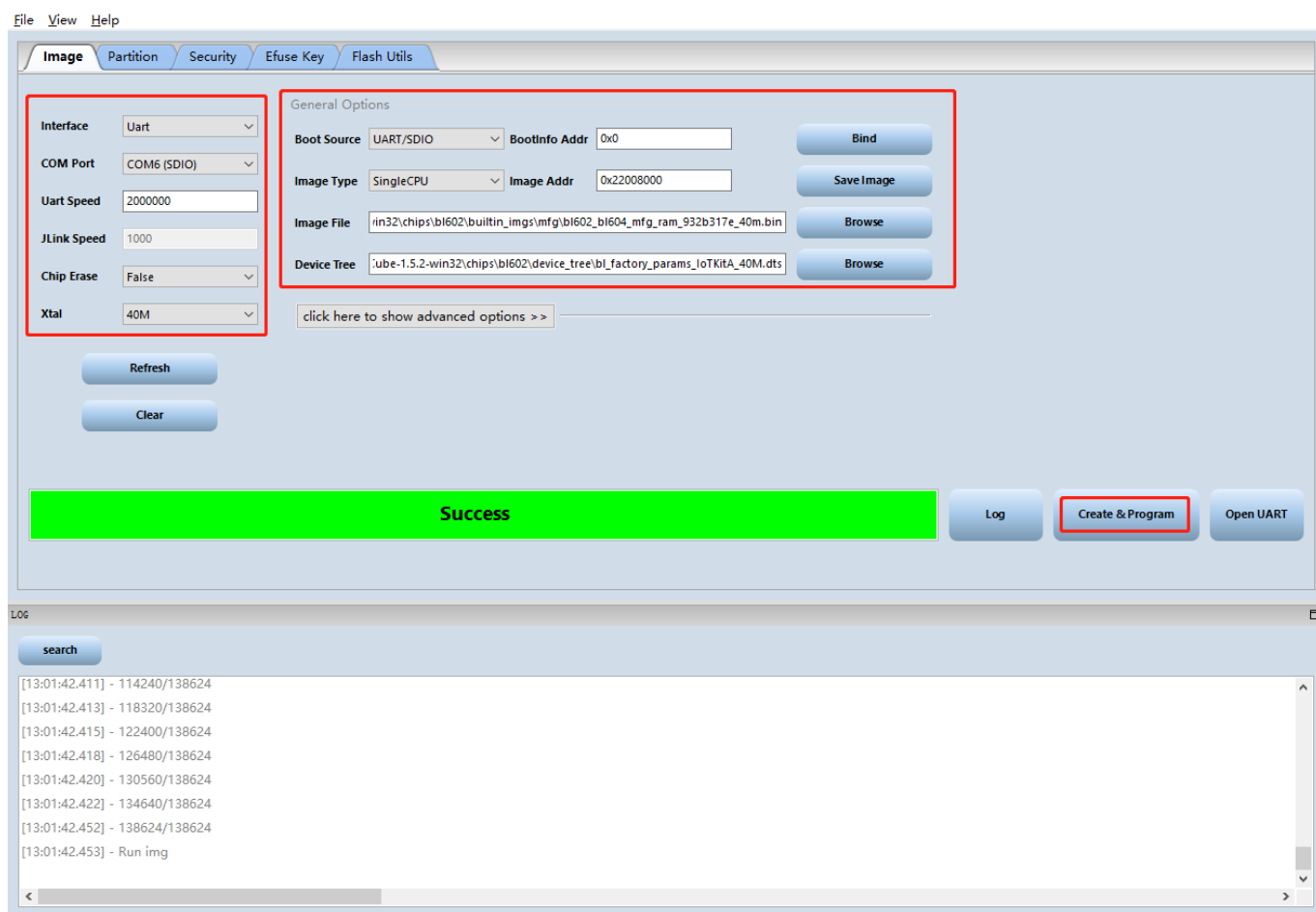


图 4.6: 烧写成功界面

注解: 如需将 SDIO 模组断电, 直接插拔 SDIO 模组即可。如果插拔重启不成功, 建议在 SDIO 模组插入的情况下将树莓派断电重启。

## 运行测试固件

对于 IOT 模组 (带 Flash), 完成测试固件的下载以后, 直接按下复位键 SW1, 芯片就可以运行 RF 测试固件了, 对于提供的模组评估套件, Boot 引脚会被串口程序自动控制, 无需额外的设定。对于 SDIO 模组 (不带 Flash), 固件下载完毕后, 无需任何设定, 固件会自动运行起来。

在 BLDevCube.exe 界面, 通过 View->RF MFG 进入到 RF MFG 测试界面。选择使用到的 COM 号, 点击 Open 按钮, 即可看到固件程序成功运行的 log, 示例如下。

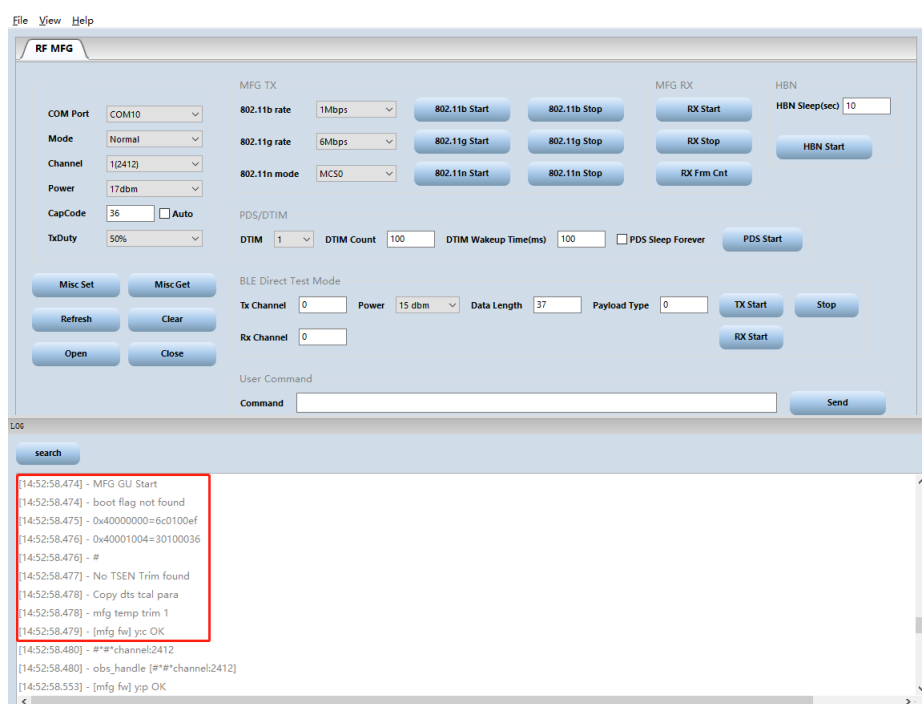


图 5.1: 程序成功运行的 log

上位机 UI 程序与测试固件通过 UART 通信, 使用的波特率是 115200, 数据位为 8 位, 没有奇偶校验。

针对晶体的负载电容，BL60X 系列芯片内部有电容补偿，不同的负载电容需求对应不同的电容补偿值，以下表格提供参考值。

备注：实际 PCB 走线也存在一定的寄生电容，所以最佳补偿值还是以实际测试结果为准。

表 6.1: BL606 对应的电容补偿值

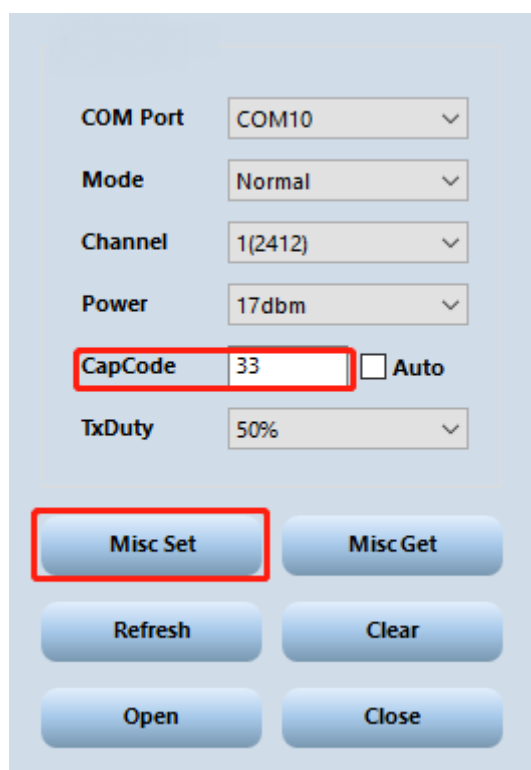
XTAL Loading Capacity (pF)	Capacity Code
8	40
10	58

表 6.2: BL602 对应的电容补偿值

XTAL Loading Capacity (pF)	Capacity Code
12	32~36
15	58~63

使用方法如下：

1. 在 Cap Code 中填写需要补偿的值。
2. 点击 Misc Set 按键更新补偿值。



COM Port: COM10

Mode: Normal

Channel: 1(2412)

Power: 17dbm

CapCode: 33 ☐ Auto

TxDuty: 50%

Misc Set Misc Get

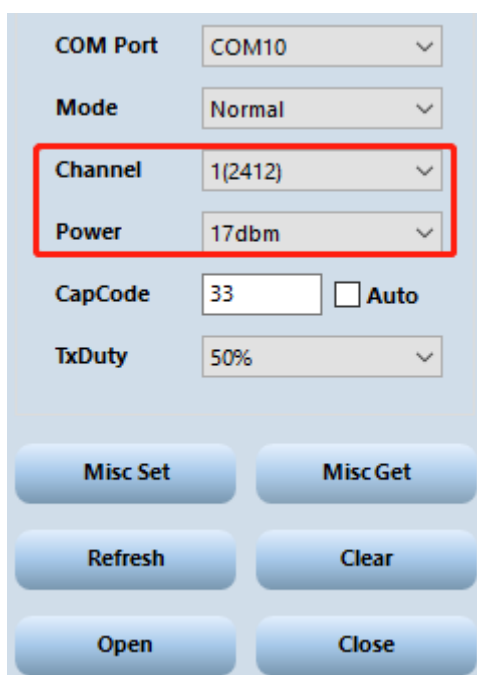
Refresh Clear

Open Close

图 6.1: 更新补偿值

## 7.1 Channel 和 Power 设置

通过 Channel 和 Power 下拉菜单框，可以设置数据包的发送通道和功率。Channel 可以选择 1-13，Power 可以选择 12-23dbm。



COM Port	COM10
Mode	Normal
Channel	1(2412)
Power	17dbm
CapCode	33 <input type="checkbox"/> Auto
TxDuty	50%

Misc Set   Misc Get

Refresh   Clear

Open   Close

图 7.1: 设置 Channel 和 Power 的参数

WiFi 不同的模式使用不同的调制方式，对信号质量 (EVM) 也有不同的要求，为了满足 WiFi 标准，针对不同制式推荐的最大功率如下表。

Mode	Rate	Maximum Power(dBm)
11n	MCS7	17
	MCS6	18
	MCS5	18
	MCS4	18
	MCS3	18
	MCS2	18
	MCS1	18
	MCS0	18
11g	54Mbps	18
	48Mbps	19
	36Mbps	20
	24Mbps	20
	18Mbps	20
	12Mbps	20
	9Mbps	20
	6Mbps	20
11b	11Mbps	18(BL606)/20(BL602)
	5.5Mbps	18(BL606)/20(BL602)
	2Mbps	18(BL606)/20(BL602)
	1Mbps	18(BL606)/20(BL602)

BL60X 系列芯片提供了功率校准机制,用户可在产品量产环节对各个 Channel 进行功率校准,将校准值写入芯片 Efuse,在应用程序启动后,根据写入的校准值纠正实际的 TX Power。

BL606/BL608 系列芯片针对功率偏差补偿预留了长度为 14 的数组空间 (Power\_Offset[14]), 每个元素为 4bit, MSB 为符号位, 允许的功率偏差范围为-4~3 (即-4dB~3dB), 超出该取值范围则校准失败。

BL602 相比 BL606/BL608, 在 efuse 容量上缩减了 50%, 预留给功率补偿的 efuse bit 数目有大幅减少, 因此 BL602 的功率补偿只能写入高中低 3 个信道的校准值, 其余信道的补偿采用插值的方法。

关于 BL60X 系列芯片的详细校准机制, 内容请参考《BL60X\_产测校准算法》。



## 7.2 发送数据包模式设置

### 7.2.1 11b 数据包发送

11b 数据包可以选择速率: 1Mbps, 2Mbps, 5.5Mbps, 11Mbps, 前导默认选择 Long preamble。设置完毕后, 就可以点击 802.11b Start 按钮进行发送, 在发送期间, log 区域会打印已经发送数据包的个数。如果想要停止发送, 点击 802.11b Stop 即可。



图 7.2: 11b 数据包设置速率

### 7.2.2 11g 数据包发送

11g 数据包可以选择速率: 6Mbps, 9Mbps, 12Mbps, 18Mbps, 24Mbps, 36Mbps, 48Mbps, 54Mbps, 设置完毕后, 就可以点击 802.11g Start 按钮进行发送, 在发送期间, log 区域会打印已经发送数据包的个数。如果想要停止发送, 点击 802.11g Stop 即可。

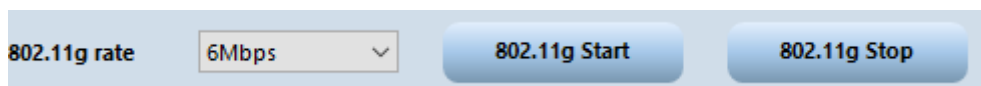


图 7.3: 11g 数据包设置速率

### 7.2.3 11n 数据包发送

11n 数据包可以选择速度模式 MCS0-MCS7, 默认带宽 20MHz, Long GI, 使用 HT-MF 模式。

---

注解: 目前 HT\_GF 模式不支持。

---

设置完毕后, 就可以点击 802.11n Start 按钮进行发送, 在发送期间, log 区域会打印已经发送数据包的个数。如果想要停止发送, 点击 802.11n Stop 即可。

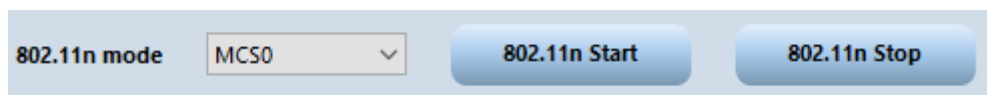


图 7.4: 11n 数据包发送界面

## 接收设置

接收设置较为简单，点击 **RX Start** 按钮后即可进入数据包接收模式，点击 **RX Frm Cnt** 按钮可以显示数据包接收个数以及 RSSI 的平均值，效果如下图。

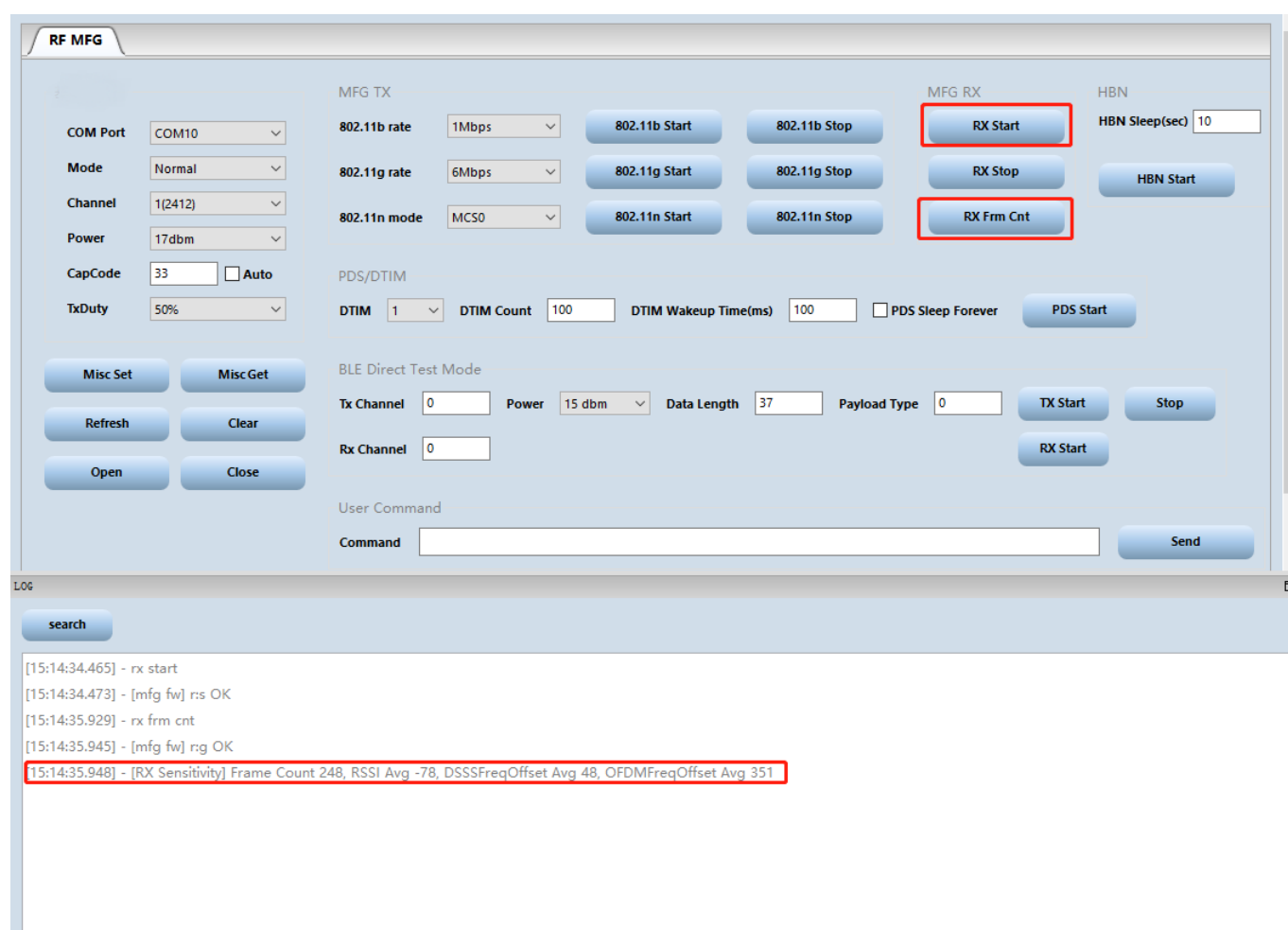


图 8.1: 接收数据包效果图

RF MFG 提供 BLE 的 TX 和 RX 测试。TX 测试可以设定测试的 Channel, Power, Data Length 和 Payload Type。设定完毕后点击 TX Start 按钮即可。RX 测试可以设定测试的 Channel, 然后点击 RX Start 按钮。测试可以使用 Stop 按钮停止。Payload Type 如下图所示。

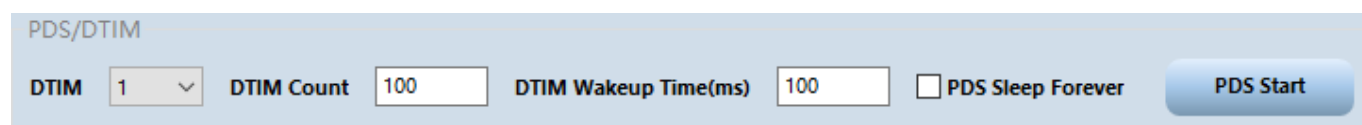
Value	Parameter Description
0x00	PRBS9 sequence '11111111100000111101...' (in transmission order) as described in [Vol 6] Part F, Section 4.1.5
0x01	Repeated '11110000' (in transmission order) sequence as described in [Vol 6] Part F, Section 4.1.5
0x02	Repeated '10101010' (in transmission order) sequence as described in [Vol 6] Part F, Section 4.1.5
0x03	PRBS15 sequence as described in [Vol 6] Part F, Section 4.1.5
0x04	Repeated '11111111' (in transmission order) sequence
0x05	Repeated '00000000' (in transmission order) sequence
0x06	Repeated '00001111' (in transmission order) sequence
0x07	Repeated '01010101' (in transmission order) sequence

图 9.1: BLE Payload Type

PDS 模式可设置芯片工作在较低功耗的同时可以唤醒 CPU 监听 WiFi AP 的 Beacon/DTIM 数据包，在检测到有数据需要接收时，启动 WiFi 的接收机进行数据的接收。PDS 模式可以设置的参数包括：

- DTIM, 设置间隔多少 Beacon 才含有 DTIM 信息，芯片会在该时刻唤醒，监听 Beacon
- DTIM Count, 设置接收多少个 DTIM（亦即睡眠次数）后，芯片进入正常的模式
- DTIM Wakeup Time, 设置芯片唤醒后，保持接收状态的时间，以便完整的接收到 Beacon

设置好上述三个参数后，点击 Start PDS 按钮即可设置芯片进入 PDS 模式。



PDS/DTIM

DTIM 1 ▼ DTIM Count 100 DTIM Wakeup Time(ms) 100 ☐ PDS Sleep Forever **PDS Start**

图 10.1: PDS 模式参数设置

HBN 测试可以让芯片进入 HBN 模式，在 HBN 模式下只有极少部分电路处在带电工作状态，其它电路的电源被关闭，功耗达到最低。芯片可以从 HBN 模式唤醒，唤醒后芯片会重新启动。芯片的 HBN 唤醒源支持 RTC 唤醒和 GPIO 唤醒，目前测试工具仅仅支持 RTC 定时唤醒，设置完 HBN 唤醒时间后，点击 **Start HBN** 按钮即可让芯片进入 HBN 模式，设定的时间到来后，芯片会重新启动。

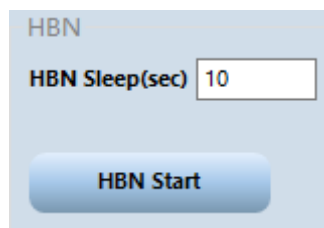


图 11.1: HBN 模式参数设置

MFG 支持 WiFi 和 BLE 的 CW(Continue Wave) 测试模式。

## 12.1 WiFi CW 测试

在界面 **Mode** 下拉菜单中选择 **Test(CW)** 模式，进入测试模式后，仅仅可以设定 **Power** 和 **Channel**，其它 WiFi 模式 (b/g/n) 相关的测试按钮是被禁止使用的。

The screenshot shows the WiFi CW test mode interface. The 'Mode' dropdown is set to 'Test(CW)' and is highlighted with a red box. The 'MFG TX' section shows 802.11b rate at 1Mbps, 802.11g rate at 6Mbps, and 802.11n mode at MCS0. The 'MFG RX' section has buttons for 802.11b Start/Stop, 802.11g Start/Stop, and 802.11n Start/Stop, all of which are disabled. The 'HBN' section has an 'HBN Start' button. The 'PDS/DTIM' section has a 'PDS Start' button. The 'BLE Direct Test Mode' section has 'TX Start', 'Stop', and 'RX Start' buttons. The 'User Command' section has a 'Send' button.

图 12.1: WiFi 测试模式参数设置

BL602/BL604 系列芯片，支持发送功率温度补偿，温度补偿相关设定请参考下文的《使用 DTS 文件设置 RF 相关参数》章节。

客户如果在生产环节需要对 RF 参数进行校准，也可使用 RF 测试固件完成产测。使用步骤如下：

## 14.1 IOT 模组 (带 Flash)

### 14.1.1 生成烧写文件

运行 BLDevCube.exe, 在 Chip Type 中选择对应的芯片型号 (比如 BL602/604), 进入烧写界面, 在该界面中勾选 MFG Bin, 并根据实际产品使用的晶振类型, 选择对应的 Flash 版本固件。固件位于烧写工具根目录下的芯片型号/bl602/builtin\_imgs/mfg\_vx.xx/gu 文件夹中。固件分为两类, 一类是普通的固件 (没有有 autoboot 后缀), 一类是烧录后自启动的固件 (带有 autoboot 后缀)。

#### 自启动固件的工作流程

自启动的 RF 测试固件在工厂烧录完整个 Flash 后, 会默认启动起来, 也就是烧录完成后默认进入产测模式, 在产测完成后, 芯片再次启动会进入用户应用程序固件。

#### 普通固件的工作流程

普通的 RF 测试固件在烧录完毕后, 启动后默认进入用户应用程序固件, 此时如需进入产测固件, 则需要应用程序调用 IOT SDK 中的 API 实现。

客户可以根据自己实际的产测场景和需求, 选择对应的产测固件。

以选择自启动固件为例, 烧录界面配置如下：



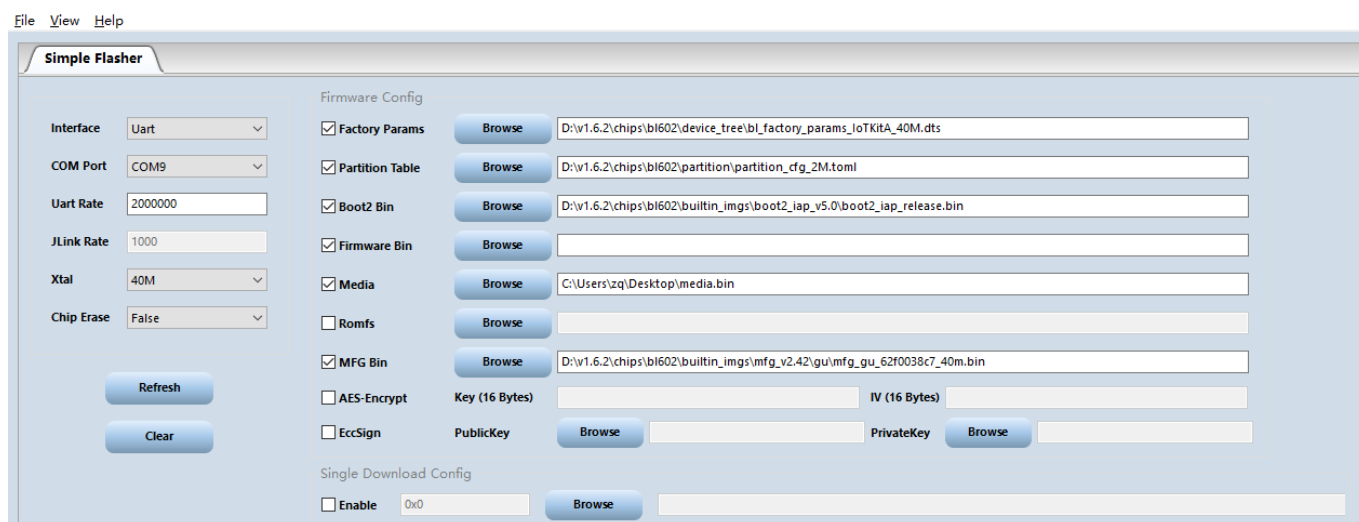


图 14.1: 烧写界面

完成上述配置后,点击 **Create&Download** 按钮,完成镜像文件的生成。生成的对应包含 MFG 的 bin 文件路径为:bl602/img\_create\_iot/whole\_flash\_data.bin 和 bl602/img\_create\_iot/whole\_img.pack。whole\_flash\_data.bin 文件为 flash 原始镜像,可以直接使用 flash 烧录器进行烧录,但是文件比较大。whole\_img.pack 为镜像压缩包,可以使用原厂自研的批量烧录工具烧写,烧录效率比较高。

### 14.1.2 工厂烧录以及产测

工厂拿到开发人员的 whole\_flash\_data.bin 或 whole\_img.pack,使用工厂批量烧写工具,完成所有镜像的烧写,具体烧写方法,请参考工厂批量烧写文档。

#### 自启动固件

如果烧录的 RF 测试固件是自启动的,完成烧写后,芯片启动会进入 RF 测试固件,此时可以对接测试仪器 (比如极致汇仪的仪器) 完成产测,产测完成后,芯片再次启动,会进入用户应用程序,产测失败,芯片再次启动还是进入 RF 测试固件。

#### 普通固件

如果烧录的 RF 产测镜像是普通固件,芯片启动后进入用户程序,需要用户程序切换到 RF 测试固件。默认的 SDK 程序中,自带切换到 RF 产测固件功能,用户可以通过串口,使用默认的 2M 波特率发送“mfg\r\n”命令,即可进入到 RF 产测固件。RF 产测固件使用的串口通信波特率是 115200,用户可以使用 115200 的波特率,发送握手命令“H\r\n”与产测固件通信,检查是否切换成功。如果需要退出 RF 产测程序,可以使用 115200 的波特率,发送命令“Reset\r\n”,即可重启芯片,进入到用户固件。

## 14.2 SDIO 模组 (不带 Flash)

对于 SDIO 接口的模组，我们提供了基于树莓派的的产测方案，树莓派一端通过 SDIO 接口连接 SDIO 模组，另外一端通过 USB 连接装有产测软件的 PC。树莓派在检测到 SDIO 模组插入后，会先自动下载产测固件，然后完成 UART 产测命令和数据的转发。

### 14.2.1 设置产测模式

将树莓派中 GPIO26 引脚跳到低电平。

### 14.2.2 将固件保存到树莓派中

按照评估的下载步骤，设置好产测固件和对应的配置参数。下载完成后，点击下载界面中 **Save Image** 按钮，会将固件保存到树莓派中。保存成功界面如下：

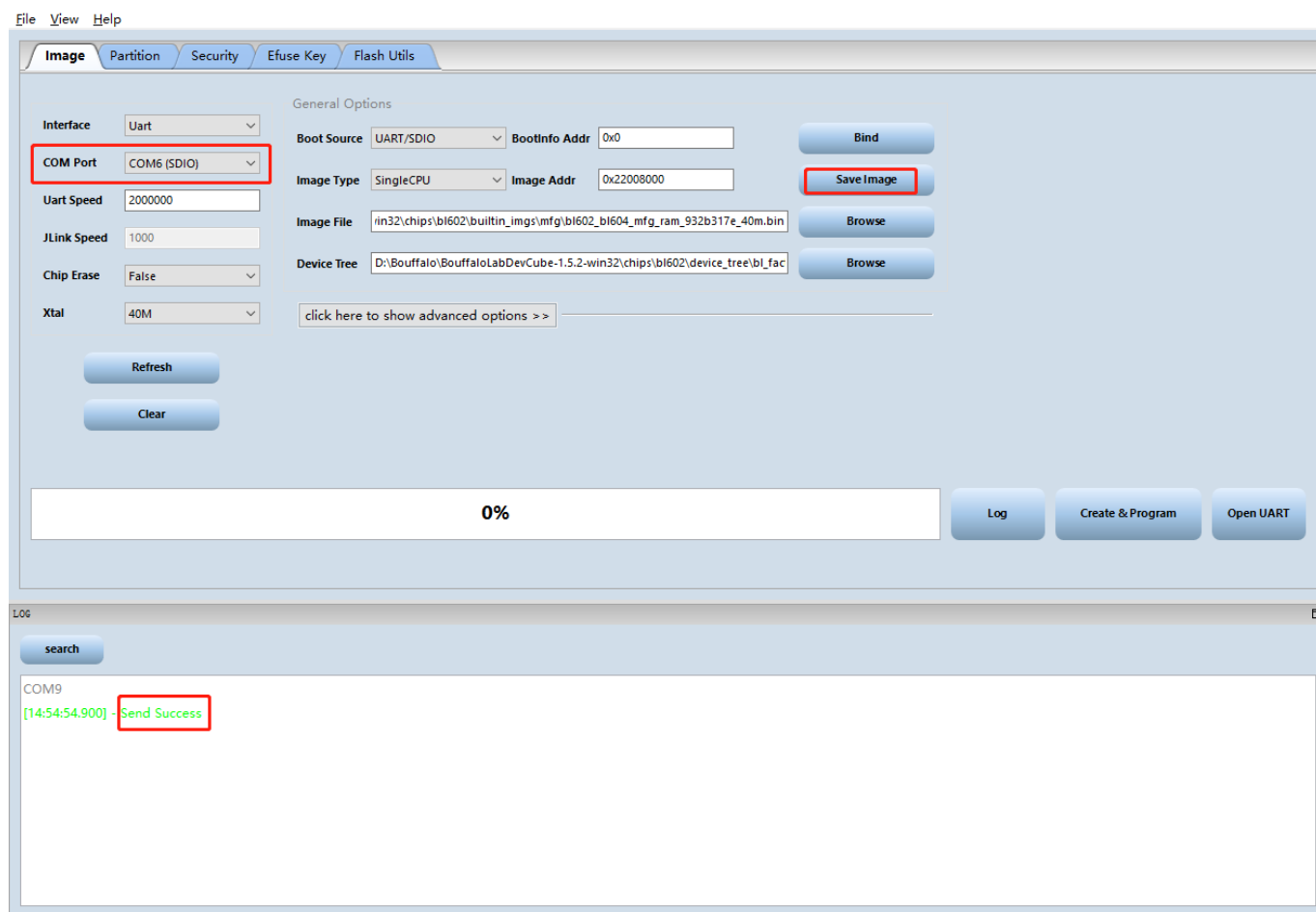


图 14.2: 固件保存成功界面

### 14.2.3 工厂产测

将保存了产测固件的树莓派交给工厂，树莓派一端通过 SD Card 接口连接 SDIO 模组，一端通过 USB 接口连接产测软件，在产测软件端呈现的是串口。当有 SDIO 模组接入时，树莓派会自动检测到模块插入，自动下载产测固件，然后完成产测软件和 SDIO 模组之间的数据和命令的转发。

## 14.3 校准参数存储说明

产测参数主要包括频偏参数，功率校准参数以及用户设定的 MAC 地址等，这些参数默认存储在芯片的 **Efuse** 中，由于 **efuse** 具有写入后无法改写的特点，故产测校准参数写入到 **efuse** 是有次数限制的，不同系列芯片，最大写入次数不同。

对于 IOT 产品 (带 Flash)，为了提供更高的灵活性，产测固件支持将产测的频偏参数，功率校准参数，用户 MAC 地址等参数，写入到 **Flash** 中。

若需要将产测参数写入到 **Flash** 中，用户在烧写产测固件时，在分区表中增加 “**rf\_para**” 分区。产测软件启动后，如果在分区表中找到 “**rf\_para**” 分区，就会把 RF 的相关参数写在 **flash** 中，而不会写在 **efuse** 中。

对于 SDIO 透传产品 (没有 Flash)，则只能将产测参数存储在 **Efuse** 中。

## 使用 DTS 文件设置 RF 相关参数

MFG 固件还具备以下两个功能：

1. 验证 RF 产测参数
2. 训练功率和频偏经验值

这些功能可以通过修改 DTS 文件中的相关参数实现。

### 15.1 DTS 文件介绍

DTS 文件是用于客户在不修改代码情况下，向固件传递参数的方法。

DTS 文件根据晶振大小选择，以 BL602/BL604 为例，DTS 文件路径为：chips/tg7100c/device\_tree/chip\_factory\_params\_loTKitA\_40M.dts

DTS 文件中的配置项包括外设配置，WiFi/BLE 配置，RF 参数配置等，本文主要介绍 RF 参数配置。

RF 可配置的参数包括：

- 温补参数：具体包括温补的使能和关闭，温补参数的调整，客户可以根据自身的补偿需求进行设定
- 功率参数：可以设定 WiFi 在 b/g/n 三种模式以及对应速度下的发射功率，同时也可以设定 BLE 的发送功率
- 功率校准模式及数值：功率校准模式可以设定从 Efuse 加载功率校准数值或者从 DTS 文件中加载功率校准数值
- 频偏校准模式及数值：频偏校准模式可以设定从 Efuse 加载频偏校准数值或者从 DTS 文件中加载频偏校准数值

设置频偏校准模式的参数为 `xtal_mode`，其模式可以设置如下：

- `xtal_mode = "M"`: 只从 Efuse 中加载频偏校准数值
- `xtal_mode = "F"`: 只从 DTS 中加载频偏校准数值
- `xtal_mode = "MF"`: 优先从 Efuse 中加载频偏校准数值，如果加载失败再从 DTS 中加载频偏校准数值
- `xtal_mode = "FM"`: 优先从 DTS 中加载频偏校准数值，如果加载失败再从 Efuse 中加载频偏校准数值

设置功率校准模式的参数为 `pwr_mode`，其模式可以设置如下：

- `pwr_mode = "B"`: 只从 `Efuse` 中加载功率校准数值
- `pwr_mode = "F"`: 只从 `DTS` 中加载功率校准数值
- `pwr_mode = "BF"`: 优先从 `Efuse` 中加载功率校准数值，如果加载失败再从 `DTS` 中加载功率校准数值
- `pwr_mode = "FB"`: 优先从 `DTS` 中加载功率校准数值，如果加载失败再从 `Efuse` 中加载功率校准数值

具体参数使用方法和注意事项可参考 `DTS` 文件中各个参数的注释。

## 15.2 设置功率温度补偿

### 15.2.1 概述

BL602 采取低于室温和高于室温两段，并随信道变化的温补机制，在  $-40^{\circ}\text{C}$  到  $125^{\circ}\text{C}$  环境温度下控制功率变化  $\leq \pm 1.5\text{dB}$ 。

### 15.2.2 DTS 文件中温补参数说明

```
rf_temp {
    en_tcal = <0>;
    linear_or_follow = <1>;
    Tchannels      = <2412 2427 2442 2457 2472>;
    Tchannel_os    = <180 168 163 160 157>;
    Tchannel_os_low = <199 186 170 165 160>;
    Troom_os       = <255>;
    //negative value is NOT supported. So we use '256' for 0, '255' for -1, '257' for 1, '511' for 256
};
```

图 15.1: 温补参数说明

- `en_tcal`: 1= 打开温补机制，0= 关闭温补机制（默认）。
- `Tchannels`: 温补参数对应的信道。
- `Tchannel_os`: 环境温度  $\geq$  室温，每  $0.5\text{dB}$  功率变化对应的环境温度变化  $\times 10$ 。
- `Tchannel_os_low`: 环境温度  $<$  室温，每  $0.5\text{dB}$  功率变化对应的环境温度变化  $\times 10$ 。
- `linear_or_follow`: 其它信道补偿速度，1= 信道间插值（默认），0= 前一个信道 in `Tchannels`。
- `Troom_os`: 室温下，内部温度 `sensor` 偏移值，默认是 0（256）。

### 15.2.3 测试获得温补参数

Step1: 将 BL602 置于温箱中，关闭温补机制（en\_tcal=0），使用 MFG 进行 Wi-Fi Tx 测试，选取 50% Duty Cycle，以目标功率持续 Tx。

Step2: 环境温度 TMid=20°C，稳定 5 分钟后，记录 Tchannels 里每个信道上的 Tx 功率大小 Pwr\_Mid[Channel]。

Step3: 环境温度 TLow=-40°C，稳定 5 分钟后，记录 Tchannels 里每个信道上的 Tx 功率大小 Pwr\_Low[Channel]。

Step4: 环境温度 THigh=120°C，稳定 5 分钟后，记录 Tchannels 里每个信道上的 Tx 功率大小 Pwr\_High[Channel]。

Step5: 计算各信道补偿速度， $Tchannel\_os\_low[Channel] = (T_{Mid} - T_{Low}) / |Pwr\_Low[Channel] - Pwr\_Mid[Channel]| / 2 * 10$   
 $Tchannel\_os[Channel] = (T_{High} - T_{Mid}) / |Pwr\_Mid[Channel] - Pwr\_High[Channel]| / 2 * 10$

Step6: 增加 BL602 样本（建议样本总数 ≥3），重复 Step1~5，对 Tchannel\_os\_low[Channel]、Tchannel\_os[Channel] 取平均值。

Step7: 更新 DTS 温补参数 Tchannel\_os 和 Tchannel\_os\_low，打开温补机制（en\_tcal=1），在温箱中验证功率变化。

### 15.2.4 示例

样本	信道 (MHz)	设置功率 (dBm)	温箱温度 (°C)	实际功率 (dBm)	20° 到 -40° 功率变化 (dB)	20° 到 120° 功率变化 (dB)	20° 到 -40° 0.5dB 功率变化的温度间隔 (°C)	20° 到 120° 0.5dB 功率变化的温度间隔 (°C)	20° 到 -40° 建议选用的补偿间隔	20° 到 120° 建议选用的补偿间隔	平均值 Tchannel_os_low	平均值 Tchannel_os
1	2412	12	-40	15.54494444	1.498078027	-2.946283425	20.02565919	16.97053297	200	169	169	169
			20	14.04686642								
			120	11.10058299								
	2427	12	-40		0	0	#DIV/0!	#DIV/0!	196	162	196	162
			20									
			120									
	2442	12	-40	16.25948829	1.549513344	-3.191956097	19.36091749	15.66437585	193	156	193	156
			20	14.70997495								
			120	11.51801885								
	2457	12	-40		0	0	#DIV/0!	#DIV/0!	162	137	162	137
			20									
			120									
	2472	12	-40	17.24917177	2.284675432	-4.205990523	13.130968	11.88780615	131	118	131	118
			20	14.96449634								
			120	10.75850582								
2	2412	12	-40	15.54494444	1.498078027	-2.946283425	20.02565919	16.97053297	200	169		
			20	14.04686642								
			120	11.10058299								
	2427	12	-40		0	0	#DIV/0!	#DIV/0!	196	162		
			20									
			120									
	2442	12	-40	16.25948829	1.549513344	-3.191956097	19.36091749	15.66437585	193	156		
			20	14.70997495								
			120	11.51801885								
	2457	12	-40		0	0	#DIV/0!	#DIV/0!	162	137		
			20									
			120									
	2472	12	-40	17.24917177	2.284675432	-4.205990523	13.130968	11.88780615	131	118		
			20	14.96449634								
			120	10.75850582								

图 15.2: 示例

## 15.3 验证 RF 产测参数

对于做 RF 产测的客户，如果需要在非信令模式下验证产测数值是否符合预期，可以通过修改 DTS 文件中的配置实现。

## 15.4 验证功率校准数值

1. 准备一块做完产测的模组。
2. 修改 DTS 文件中的 `pwr_mode` 为“B”(如果确认 Efuse 中有功率校准数值, 模式设置为“BF”亦可), 即只从 Efuse 中加载功率校准数值。
3. 按照上述烧写或者下载固件的步骤, 完成 MFG 固件的烧写或者下载。
4. MFG 固件运行后, 通过 View->RF MFG 进入到 RF MFG 测试界面, 将基本配置项中的 Power Offset 设置为 Enable, 此时 MFG 固件就会按照 `pwr_mode` 设定的模式, 从 Efuse 中加载功率校准数值。
5. 设定发送功率, 使用仪器测量实际输出的功率。

使能 Power Offset 以后, MFG 固件输出相关 log 信息如下:

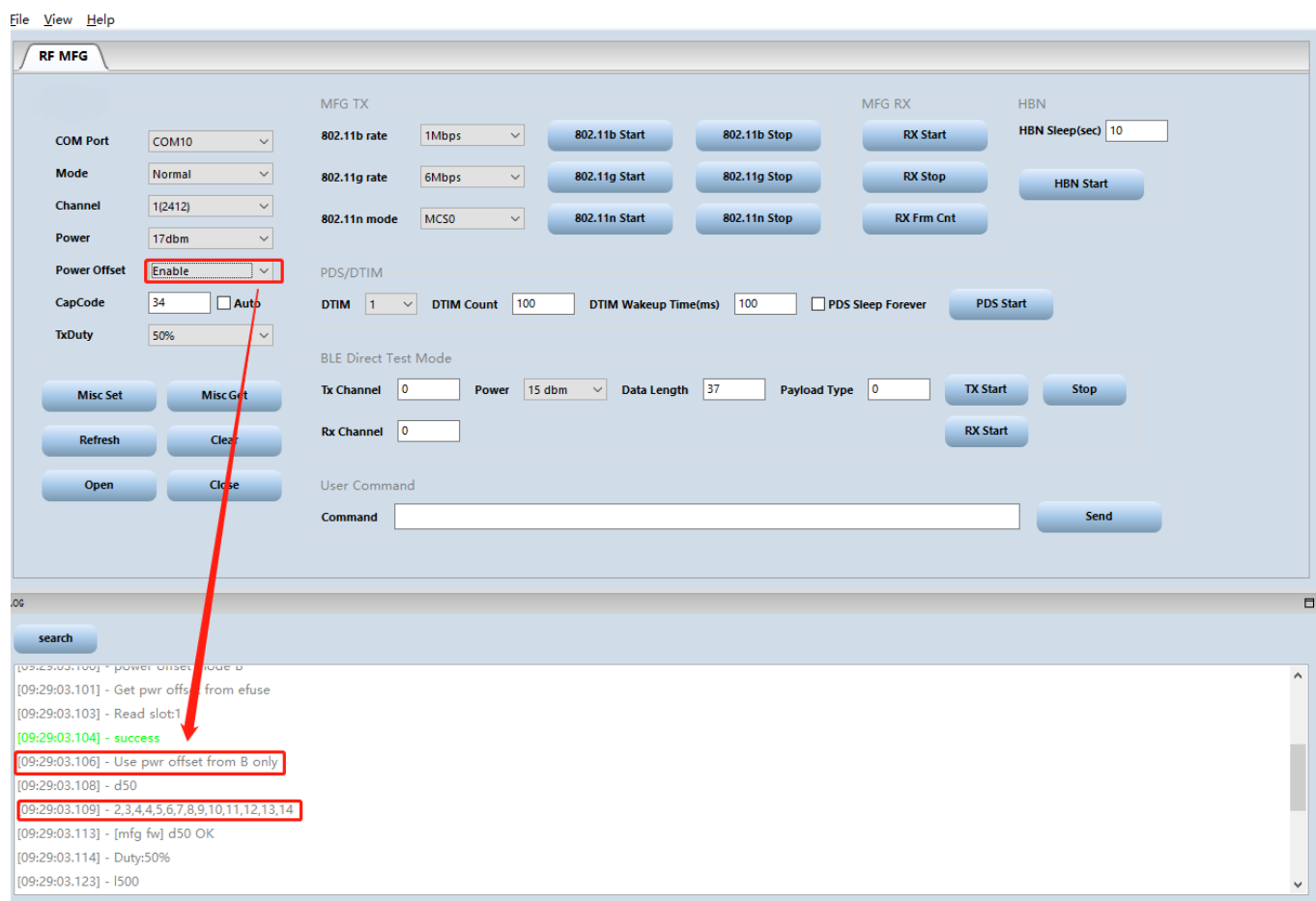


图 15.3: 使能 Power Offset 的 log 信息

注解: 使能 Power Offset 功率补偿的命令是 `V1[r\n]` 或者 `V-1[r\n]`。如果需要去掉 Power Offset 功率补偿, 对应的命令是 `V0[r\n]`, 此时固件会将 Power Offset 清零。

## 15.5 验证频偏校准数值

1. 准备一块做完产测的模组。
2. 修改 DTS 文件中的 `xtal_mode` 为“M”(如果确认 Efuse 中有频偏校准数值, 模式设置为“MF”亦可), 即只从 Efuse 中加载频偏校准数值。
3. 按照上述烧写或者下载固件的步骤, 完成 MFG 固件的烧写或者下载。
4. MFG 固件运行后, 通过 View->RF MFG 进入到 RF MFG 测试界面, 将基本配置项中 Cap Code 的 Auto 选项勾选, 此时 MFG 固件就会按照 `xtal_mode` 设定的模式, 从 Efuse 中加载频偏校准数值。
5. 设定发送数据包, 使用仪器测量实际输出的频率。

勾选 Cap Code 的 Auto 选项以后, MFG 固件输出相关 log 信息如下:

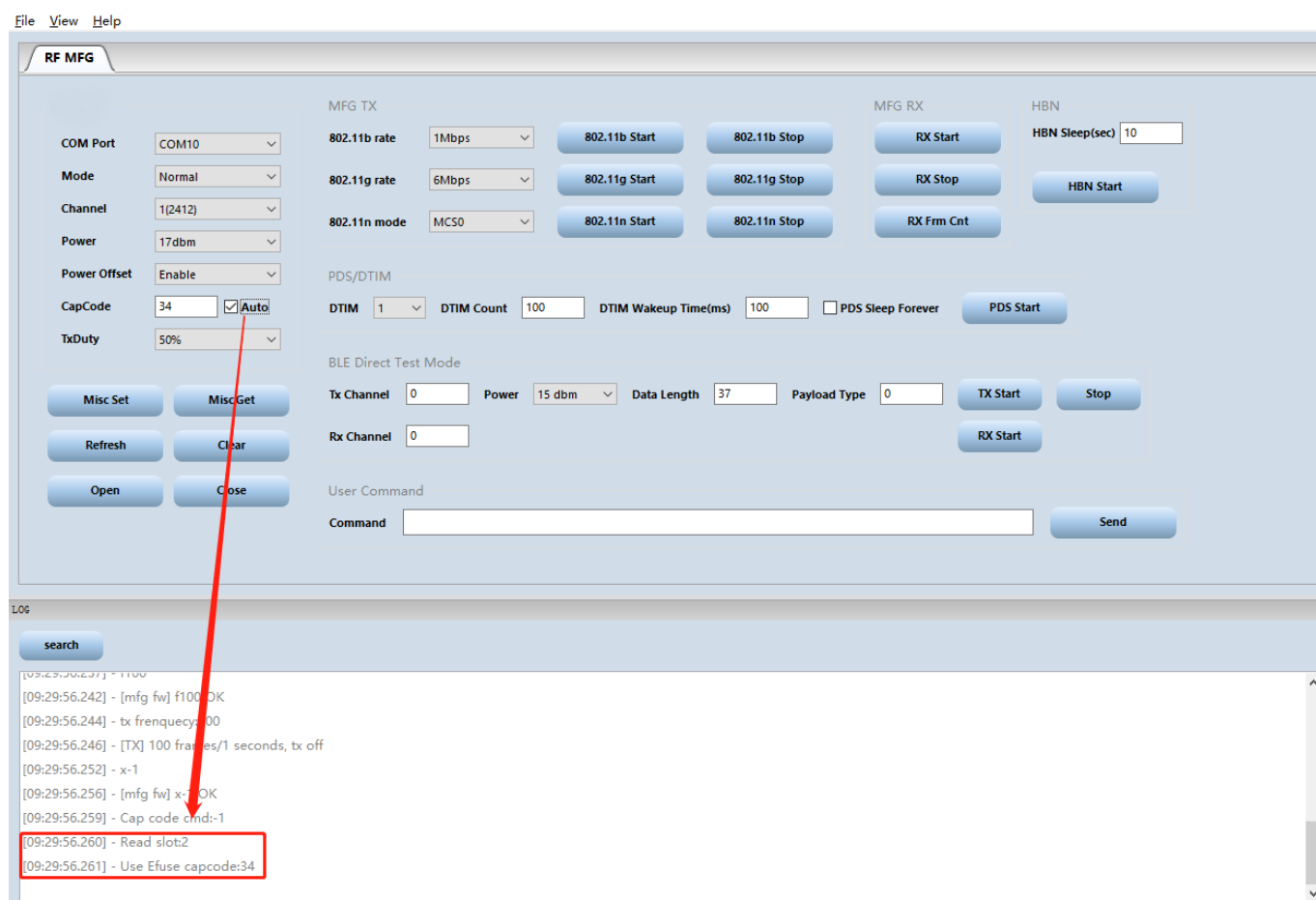


图 15.4: 勾选 Auto 后的 log 信息

注解: 此时发送给固件的命令是 `X-1[\r\n]`, 表示从 DTS 文件中加载功率参数, 并按照 DTS 中定义规则, 设置频偏。



## 15.6 训练功率和频偏经验值

对于不需要做 RF 功率和频偏校准的用户，MFG 提供了设置经验值的方法。用户可抽检一批产品，通过调整 `pwr_table_11b` / `pwr_table_11g` / `pwr_table_11n` / `pwr_table_ble` 四个表项的功率数值，从而找出 WiFi/BLE 平均发射功率值，以此这些 Golden 值作为默认的发射功率。频偏参数同样可以通过调整 `xtal` 来实现。

训练功率经验值方法如下：

1. 准备一块模组。
2. 按照上述烧写或者下载固件的步骤，完成 MFG 固件的烧写或者下载。
3. MFG 固件运行后，通过 View->RF MFG 进入到 RF MFG 测试界面，将基本配置项中 Power 选项选择 Auto，此时 MFG 固件就会从 DTS 文件中加载 `pwr_table_11b` / `pwr_table_11g` / `pwr_table_11n` / `pwr_table_ble` 参数。
4. 设定发送数据包，使用仪器测量实际输出的功率。
5. 如果功率符合预期，则完成测试，此时 DTS 文件中的功率数值就是要寻找的功率值。
6. 如果功率不符合预期，则根据情况修改 DTS 文件中的 `pwr_table_11b` / `pwr_table_11g` / `pwr_table_11n` / `pwr_table_ble` 参数，重复步骤 2-6。

---

注解：此时发送给固件的命令是 `P-1[\r\n]`，表示从 DTS 文件中加载功率参数，并按照 DTS 中定义规则，设置功率。由于功率是按照 b/g/n 三种模式及速率进行设定，所以在发送 P-1 命令前，要先发送设置 Channel，模式，速度等命令，再发送 P-1 命令。

---

训练频偏经验值方法如下：

1. 准备一块模组。
2. 修改 DTS 文件中的 `xtal_mode` 为“F”(如果确认 Efuse 中无频偏校准数值，模式设置为“MF”亦可)，即只从 DTS 中加载频偏校准数值。
3. 按照上述烧写或者下载固件的步骤，完成 MFG 固件的烧写或者下载。
4. MFG 固件运行后，通过 View->RF MFG 进入到 RF MFG 测试界面，将基本配置项中 Cap Code 的 Auto 选项勾选，此时 MFG 固件就会按照 `xtal_mode` 设定的模式，只从 DTS 中加载频偏校准数值。
5. 设定发送数据包，使用仪器测量实际输出的频率。
6. 如果频率符合预期，则完成测试，此时 DTS 文件中的频偏校准数值就是要寻找的值。
7. 如果频率不符合预期，则根据情况修改 DTS 文件中的 `xtal_mode` 参数，重复步骤 2-6。

---

注解：同上，此时发送给固件的命令是 `X-1[\r\n]`，表示从 DTS 文件中加载功率参数，并按照 DTS 中定义规则，设置频偏。

---

所有命令均是字符串类型。

## 16.1 Shakehand

- 命令: `H`
- 返回: `mfg`

主机工具应该先发送“`H\r\n`”去检测 MFG 固件是否已经运行起来，如果 MFG 固件已经运行起来，它收到“`H\r\n`”命令后会以“`mfg\r\n`”应答。如果 MFG 没有在运行：

1. 主机用正常固件使用的波特率（默认 9600）发送“`mfg\r\n`”命令，让正常固件切换到 MFG 固件。
2. 主机使用 115200 的波特率发送“`H\r\n`”并检查能否收到“`mfg\r\n`”。
3. 如果主机收不到“`mfg\r\n`”，重复步骤 1。
4. 主机收到“`mfg\r\n`”后可以进行正常的测试。

## 16.2 TX on/off

on:`t1`

off:`t0`

## 16.3 TX modulation

### 16.3.1 2.4G 11n

mcs idx = 0 - 7

1. short GI + HT-GF + HT20:`msg2[mcs idx]`
2. short GI + HT-MF + HT20:`msm2[mcs idx]`

3. long GI + HT-GF + HT20:mlg2[mcs idx]
4. long GI + HT-MF + HT20:mlm2[mcs idx]
5. short GI + HT-GF + HT40:msg4[mcs idx]
6. short GI + HT-MF + HT40:msm4[mcs idx]
7. long GI + HT-GF + HT40:mlg4[mcs idx]
8. long GI + HT-MF + HT40:mlm4[mcs idx]

---

注解：BL602 不支持 HT40。

---

### 16.3.2 2.4G 11g

rate idx = 0 - 7, 0:6Mbps 1:9Mbps 2:12Mbps 3:18Mbps 4:24Mbps 5:36Mbps 6:48Mbps 7:54Mbps

- 命令: g[rate idx]

### 16.3.3 2.4G 11b

rate idx = 0 - 3, 0:1Mbps 1:2Mbps 2:5.5Mbps 3:11Mbps

1. Long Preamble:B[rate idx]
2. short Preamble:b[rate idx]

## 16.4 2.4g channel

channel idx = 1 - 13

- 命令: c[channel idx]

## 16.5 2.4g tx power

power dbm = 12 - 23dbm

- 命令: p[power dbm]

---

注解：p-1 表示从 DTS 文件中加载功率参数，并按照 DTS 中定义规则，设置功率，详见《使用 DTS 文件设置 RF 相关参数》章节。

---

## 16.6 TX frame length

- 命令: `l[length]`

## 16.7 TX frequency

max value=1000

- 命令: `f[freq]`

## 16.8 PDS

enter into pds mode

1. sleep forever `sa`
2. rtc wakeup mode and dtim mode dtim:1 - 9 dtim count `s:[dtim] [dtim count]`
3. wakeup keep time keep ms: Unit is microsecond `a:w[keep ms]`

## 16.9 HBN

enter into hbn mode

1. rtc wake up mode `hr[second]`
2. gpio wake up mode `TODO`

## 16.10 RX

1. start rx `r:s`
2. get rx information `r:g`
  - 返回: `[RX Sensitivity] Frame Count [frame count], RSSI Avg [anverage of RSSI], DSSSFreqOffset Avg [anverage of DSSS Frequency Offset], OFDMFreqOffset Avg [anverage of OFDM Frequency Offset]`

## 16.11 Get MFG FW version

- 命令: `y:v`
- 返回: `###version:[version]`

## 16.12 Get MFG FW building infomation

- 命令: `y:d`
- 返回: `***date:[building date] time:[building time]`

## 16.13 Get current power level

- 命令: `y:p`
- 返回: `***power:[power level dbm]`

## 16.14 Get current channel

- 命令: `y:c`
- 返回: `***channel:[channel freq]`

## 16.15 Get current tx status

- 命令: `y:t`
- 返回: `***tx:[0 or 1]`

## 16.16 Get tx frequency

- 命令: `y:f`
- 返回: `***freq:[tx frequency]`

## 16.17 Get cap code

- 命令: `y:x`
- 返回: `***capcode:[capcode value]`

## 16.18 Get MFG mode

- 命令: `y:M`
- 返回: `***mfgmode:[MFG mode]`

## 16.19 Set cap code

- 命令: `X[cap code]`

注解: X-1 表示从 DTS 文件中加载功率参数, 并按照 DTS 中定义规则, 设置频偏, 详见《使用 DTS 文件设置 RF 相关参数》章节。

## 16.20 Set MFG Test(CW) mode

0 for normal mode, 1 for CW test mode

- 命令: `M[MFG mode]`

## 16.21 Write data to efuse

注意, 由于 Efuse 具有写入后不可修改的特点, 所以在对 efuse 进行读写的时候, 要确保芯片正确的收到了主机发出的数据, 为此, 主机要按照如下流程进行设定:

1. 主机使用 WEA 命令将要写入的数据发给 MFG 的 FW, 此时 FW 只是将数据暂存, 并没有写入 Efuse。
2. 主机使用 LEA 命令从 Efuse 暂存区读取设定的参数, 判断 FW 是否正确接收, 如果没有正确接收, 重复步骤 1。
3. 主机判断设定的参数正确后, 使用 SEA 命令, 将参数真正的写入 Efuse。
4. 主机使用 REA 命令, 从 Efuse 中读取设定的参数, 校验正确则可认为 Efuse 写入成功。

### 16.21.1 Write data to efuse buffer

- 命令: `WEA[address in hex string]=[value in hex string]`

示例:

写入向 0x04 地址写 0x80000008

`WEA0x00000004=0x80000008`

### 16.21.2 Load data from efuse buffer

- 命令: `LEA[address in hex string]`
- 返回: `Read efuse [address in hex string]=[value in hex string]`

示例:

读取 0x04 地址处的数据

`LEA0x00000004`

返回

Read efuse 0x00000004=0x80000008

### 16.21.3 Program data to efuse

- 命令: `SEA`
- 返回: `Save efuse OK`

### 16.21.4 Read data from efuse

- 命令: `REA[address in hex string]`
- 返回: `Read efuse [address in hex string]=[value in hex string]`

示例:

读取 0x04 地址处的数据

`LEA0x00000004`

返回

Read efuse 0x00000004=0x80000008

## 16.22 Save calibration parameters to efuse

注意, 由于 **Efuse** 具有写入后不可修改的特点, 所以在使用 **Efuse** 进行参数设定的时候, 要确保芯片正确的收到了主机发出的参数, 为此, 主机要按照如下流程进行设定:

1. 主机使用 **WEx** 命令将要写入的数据发给 **MFG** 的 **FW**, 此时 **FW** 只是将数据暂存, 并没有写入 **Efuse**。
2. 主机使用 **LEx** 命令从 **Efuse** 暂存区读取设定的参数, 判断 **FW** 是否正确接收, 如果没有正确接收, 重复步骤 1。
3. 主机判断设定的参数正确后, 使用 **SEx** 命令, 将参数真正的写入 **Efuse**。
4. 主机使用 **REx** 命令, 从 **Efuse** 中读取设定的参数, 校验正确则可认为 **Efuse** 写入成功。

### 16.22.1 Write cap code to efuse buffer

- 命令: `WEX[cap code]`

### 16.22.2 Load cap code from efuse buffer

- 命令: `LEX`
- 返回: `Cap code2:[cap code]`

### 16.22.3 Program cap code to efuse

- 命令: `SEX`

### 16.22.4 Read cap code from efuse

- 命令: `REX`
- 返回: `Cap code2:[cap code]`

### 16.22.5 Write power offset to efuse buffer

- 命令: `WEP[Channel 1 power offset],[Channel 2 power offset]...[Channel 12 power offset],[Channel 14 power offset]`

示例:

写入 Channel 1-14 的功率偏移-1,2,3,3,3,2,1,0,-1,-2,-3,-4,1,3

WEP-1,2,3,3,3,2,1,0,-1,-2,-3,-4,1,3

---

注解: 如果功率校准采用的是线性插值方法, 比如只做 1,7,13 通道的校准, 但是使用 `WEP` 命令的时候仍然需要传递 14 个通道的数值, 其它通道的偏移值可以写 0. 同样的道理, 如果只做某两个通道的校准, 也是需要传递 14 个通道的数值, 不关心的通道功率偏移值可以设置为 0.

---

### 16.22.6 Load power offset from efuse buffer

- 命令: `LEP`
- 返回: `Power offset:[Channel 1 power offset],[Channel 2 power offset]...[Channel 12 power offset],[Channel 14 power offset]`



### 16.22.7 Program power offset to efuse

- 命令: SEP

### 16.22.8 Read power offset from efuse

- 命令: REP
- 返回: Power offset: [Channel 1 power offset], [Channel 2 power offset]... [Channel 12 power offset], [Channel 14 power offset]

### 16.22.9 Enable power offset in efuse

- 命令: V

MFG 固件默认不会使能 TX Power Offset 的校准功能, 如果需要验证校准的准确性, 需要发送 V 命令使能 TX Power Offset 校准功能, 收到该命令后 MFG 固件启用校准功能, 并会通过 log 打印使用的 efuse 校准值。

### 16.22.10 Write mac address to efuse buffer

- 命令: WEM[MAC0 hex string]:[MAC1 hex string]:[MAC2 hex string]:[MAC3 hex string]:[MAC4 hex string]:[MAC5 hex string]

示例:

写入 MAC 地址: 18:B9:05:60:0E:74,

WEM18:B9:05:60:0E:74

### 16.22.11 Load mac address from efuse buffer

- 命令: LEM
- 返回: MAC: [MAC0 hex string]:[MAC1 hex string]:[MAC2 hex string]:[MAC3 hex string]:[MAC4 hex string]:[MAC5 hex string]

示例:

返回 MAC:18:B9:05:60:0E:74

### 16.22.12 Program mac address to efuse

- 命令: SEM

### 16.22.13 Read mac address from efuse buffer

- 命令: `REM`
- 返回: `MAC:[MAC0 hex string]:[MAC1 hex string]:[MAC2 hex string]:[MAC3 hex string]:[MAC4 hex string]:[MAC5 hex string]`

示例:

返回 `MAC:18:B9:05:60:0E:74`

## 16.23 Save calibration parameters to flash

由于 `efuse` 具有写入后无法改写的特点, 故产测校准参数写入到 `efuse` 是有次数限制的, 不同系列芯片, 最大写入次数不同。为了提供更高的灵活性, 产测固件支持将产测的频偏参数, 功率校准参数, 用户 `MAC` 地址等参数, 写入到 `Flash`。如需要将产测参数写入到 `Flash`, 用户需要在烧写产测固件时, 在分区表中增加 “`rf_para`” 这个分区, 产测软件启动后, 如果在分区表中找到了 “`rf_para`” 分区, 就会把 `RF` 的相关参数写在 `flash` 中, 而不会写在 `efuse` 中。校准参数的写入命令和流程同《`Save calibration parameters to efuse`》

`flash` 中 `RF` 参数结构体如下:

```
typedef struct rf_para_flash_tag{
    uint32_t magic;           //"RFPA"
    uint8_t capcode_valid;    //0x5A
    uint8_t capcode;
    uint8_t poweroffset_valid; //0x5A
    int8_t poweroffset[3];
    uint8_t mac_valid;        //0x5A
    uint8_t mac[6];
    uint8_t rsvd[3];
    uint32_t crc32;
}rf_para_flash_t;
```

## 16.24 Reset MFG FW

- 命令: `Reset`

---

注解: It's only design for MFG firmware running from flash.

---

## 16.25 Set tx duty

- 命令: `d[duty]`

---

注解: Duty value is between 0-100

---

## 16.26 Get tx duty

- 命令: `y:i`
- 返回: `###duty:[tx duty]`

## 16.27 BLE Test

### 16.27.1 BLE TX Power

- 命令: `EP[power]`

Power 参数是 16 进制字符串。

举例: EP11

Set Tx Power 17 dbm.

### 16.27.2 BLE TX

- 命令: `ET[channel][data length][payload type]`

所有的参数都是 16 进制字符串。

举例: ET261600

To transmit le test data on RF Channel38, with data Length 22 and with PRBS9 packet payload type and data len 22.

### 16.27.3 BLE RX

- 命令: `ER[channel]`

channel 参数是 16 进制字符串。

举例: ER26

To receive le test data on RF Channel38.

#### 16.27.4 BLE test stop

- 命令: `EE`