思 澈 科 技

SF32LB56x

低功耗场景测试报告

latest (2026-08-04)

RPT5602

www.sifli.com

目录

SF32LB56x 低功耗测试报告

欢迎使用 SF32LB56x 系列芯片低功耗测试报告！

芯片简介

关于芯片的介绍

测试环境

了解测试所需的硬件环境、软件配置和测试工具

BLE 低功耗测试

蓝牙低功耗场景下的详细测试方法和结果

BT 经典蓝牙测试

经典蓝牙应用场景的功耗测试数据

Coremark 性能测试

基准性能测试中的功耗表现分析

1. SF32LB56x 芯片简介

1.1. 芯片架构概述

SF32LB56x 是一颗基于 ArmCortex-M33STAR-MC1 处理器的大小核架构SoC芯片。

高性能大核

专门设计用于高性能运算场景：

- 图形处理
- 音频处理
- 神经网络运算

低功耗小核 (LCPU)

优化用于低功耗场景：

- 蓝牙协议栈
- 传感器数据采集
- 传感器算法的计算

1.2. 测试内容概览

本文档提供了 SF32LB56x 系列芯片在多种使用场景下的功耗测试方法与参考结果：

测试场景总览

类别	测试场景	说明
无线通信	BLE 低功耗蓝牙	广播、连接、数据传输等典型场景

类别	测试场景	说明
无线通信	BT 经典蓝牙	传统蓝牙协议的各种应用模式
性能测试	Coremark 基准测试	标准化 CPU 性能测试中的功耗表现
基础测试	While Loop 循环	基础循环操作的功耗基线
待机模式	关机模式	最低功耗状态下的功耗数据

1.3. 重要提醒

警告

测试结果仅供参考

本文档的测试结果与软硬件环境紧密相关，实际应用中可能存在差异。

1.3.1. 影响功耗的关键因素

芯片因素

- 芯片自身的个体差异
- 制造工艺的批次变化

硬件因素

- 外围元器件的选型与质量
- 芯片内置 Buck 的外接电感
- PCB 设计与布线

测试环境

- 测试设备的准确度与校准
- 电源的稳定性与纹波
- 环境温湿度条件

软件配置

- 固件版本与编译选项
- I/O 引脚的配置状态
- 时钟与电源管理设置

环境条件

- 工作温度范围
- 电磁干扰水平
- 供电电压波动

2. 测试环境配置指南

准确的功耗测试需要标准化的环境配置。本章节介绍了测试所需的各项环境要求。

2.1. 相关硬件

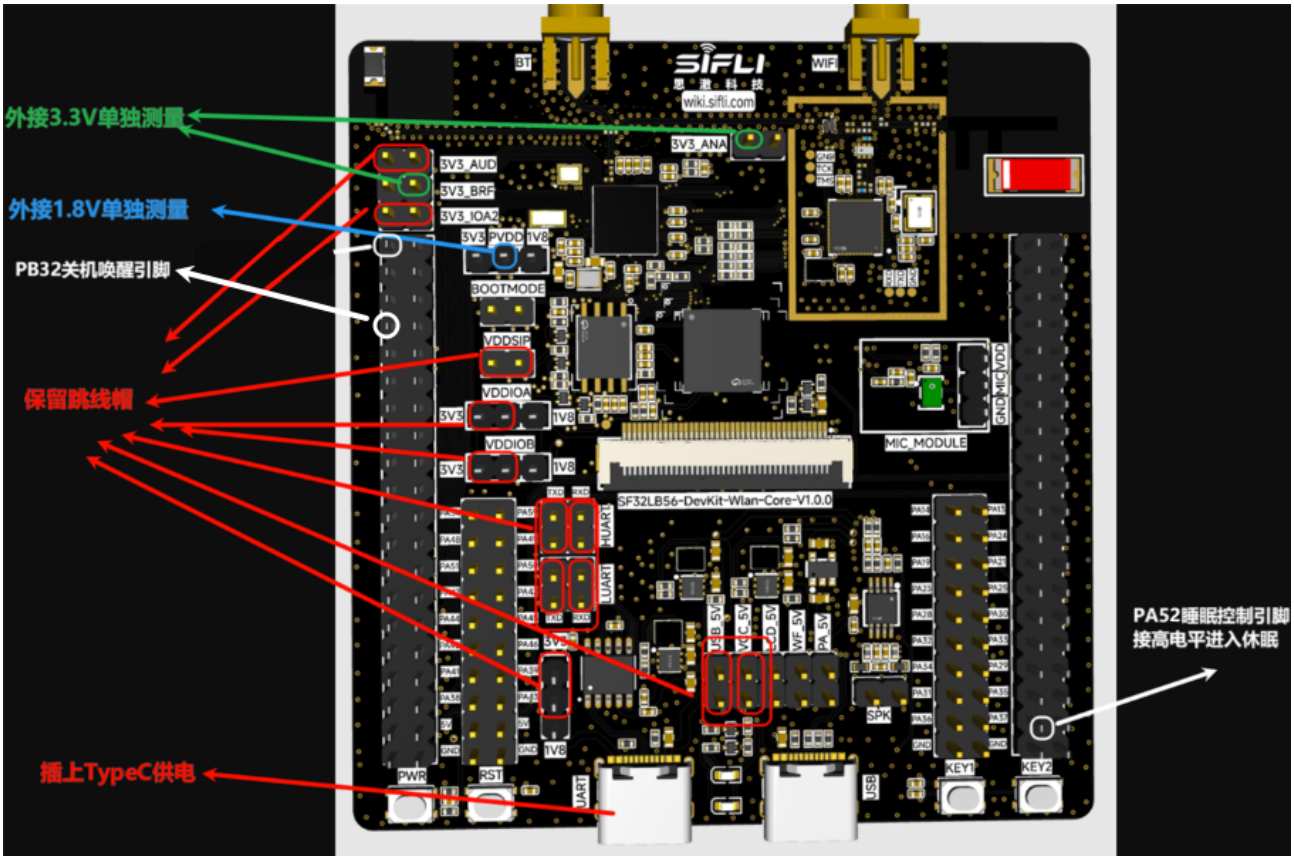
- SF32LB56V-DevKit-Wlan-Core使用方法参考SF32LB56V-DevKit-Wlan-Core开发板使用指南
- Agilent 66319D直流电源
- Keysight 34465A 数字电流计
- 串口板
- 安卓手机，安装 LightBlue App（com.punchthrough.lightblueexplorer_LightBlue_1.9.3.apk，可从GooglePlay 应用商店下载）或者安装 BLE调试助手 (从安卓自带的应用商店即可下载)。

2.2. 相关软件

- BLE 和 BT 功耗测试例程：example\pm\bt\src
- CoreMark 功耗测试例程：example\pm\coremark\src

2.3. 功耗测量方法

使用功耗测试仪器分别对 PVDD进行1.8V供电，3V3_ANA + 3V3_BRF进行3.3V供电。供电的针脚与跳线帽情况如图框出，两个唤醒引脚 PB32与PA52 可以暂时先插上杜邦线但不接确定电平以备后续测试使用。



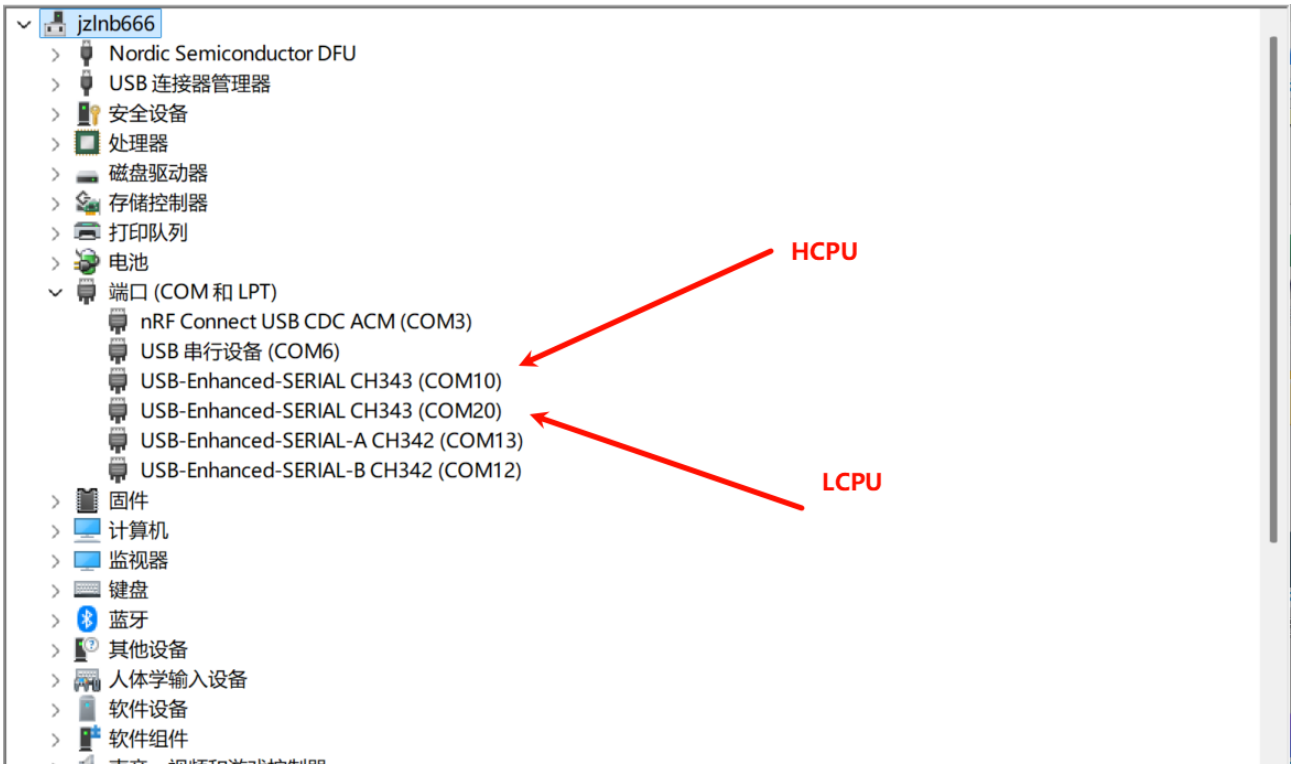
3. BLE 低功耗蓝牙测试

BLE (Bluetooth Low Energy) 是 SF32LB56x 的核心功能之一。本章节详细介绍各种 BLE 应用场景下的功耗测试方法和结果。

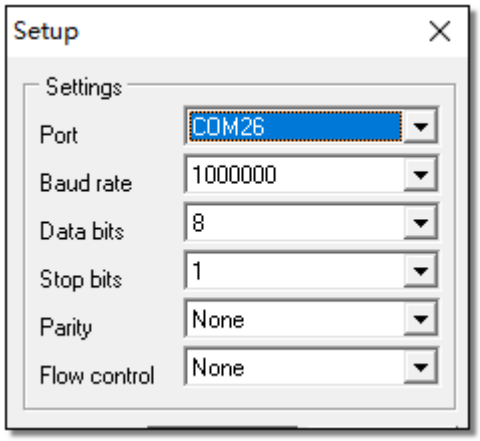
3.1. 测试说明

使用BLE例程可以测试典型的ADV和连接场景功耗，系统上电后例程自动开启ADV发送，通过HCPU的console发送命令修改ADV发送间隔和连接周期，发送的命令都要以回车换行符结尾。

PC 与调试板使用 USB Type-C 线连接后会枚举出两个串口，分别作为HCPU以及LCPU的console，如下图所示。



串口设置参见下图：波特率均设置为 1000000。



为便于控制测试条件，使用PA52作为HCPU的唤醒PIN。当唤醒PIN不为高电平时HCPU无法进人低功耗模式，此时可以通过console给HCPU发送命令执行指定任务，当唤醒PIN接高电平（即1.8V电压，下文如未特别说明，高电平均指1.8V电压）时，HCPU进人低功耗模式，此时HCPU无法响应console命令。

测试例程LCPU的主频为24MHz，HCPU与LCPU使用的低功耗模式为Standby模式。

HCPU 支持如下命令修改 BLE ADV 和连接周期，同时还可使用 btskey 命令操作菜单修改 BT 的配置。

- ble_config adv interval_in_ms: 其中interval_in_ms是毫秒为单位的 ADV 间隔； 例如：发送命令ble_config adv 100，可将 ADV 周期改为 100ms
- ble_config conn interval_in_ms: 其中interval_in_ms是毫秒为单位的连接周期； 例如：在连接之后发送 ble_config conn 100，可将连接周期改为 100ms

3.2. 例程编译与烧录

3.2.1. 编译

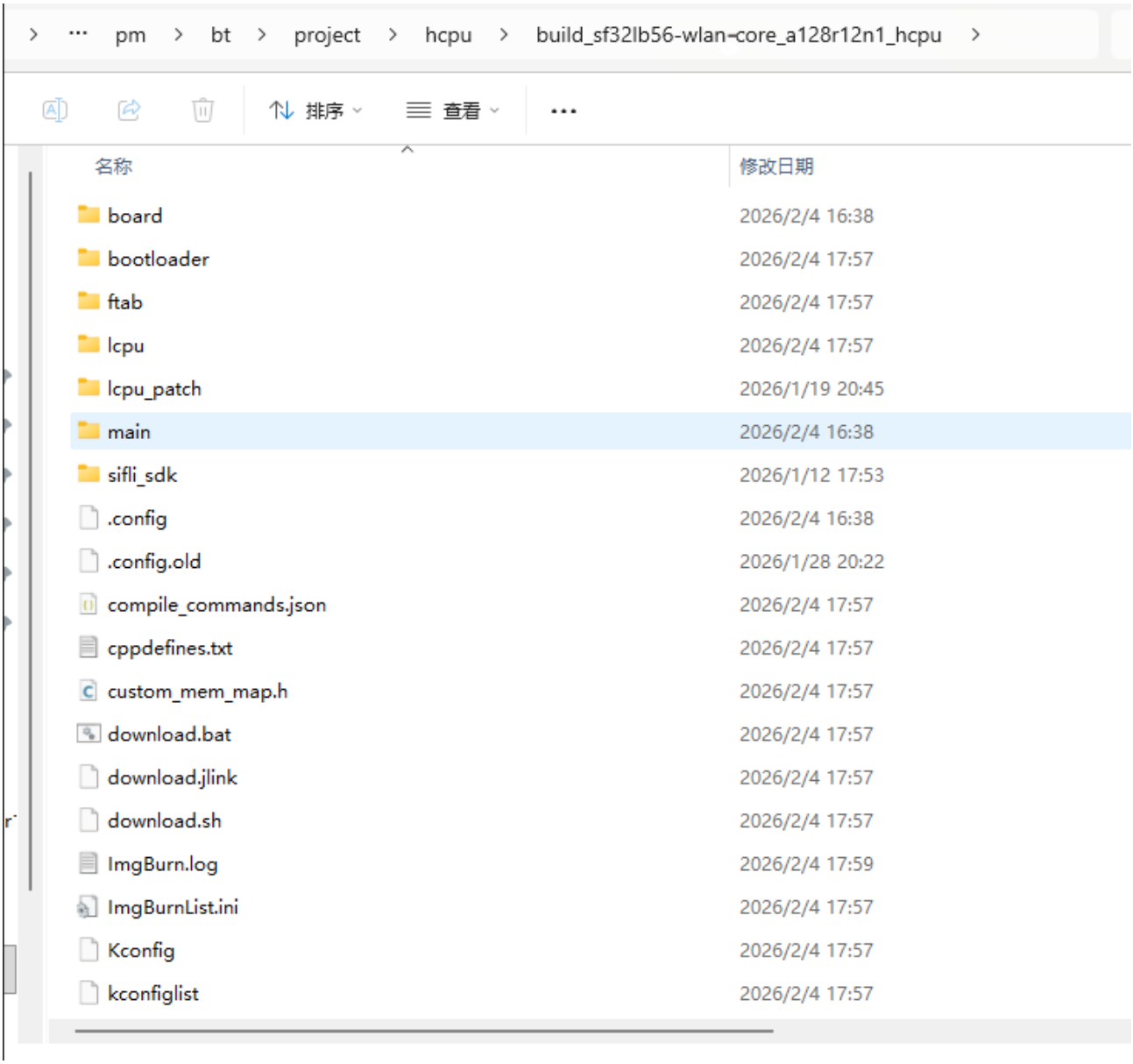
如果使用已编译好的 image 文件，可以直接跳到烧录部分进行烧录开始测试。 进入example\pm\bt\project\hcpu目录 如果使用的开发板型号为 `sf32lb56-wlan-core_a128r12n1`,就使用如下命令进行编译

```
scons --board=sf32lb56-wlan-core_a128r12n1 -j8
```

如果使用的开发板型号为 `sf32lb56-wlan-core_n16r12n1`,就使用如下命令进行编译

```
scons --board=sf32lb56-wlan-core_n16r12n1 -j8
```

以 `sf32lb56-wlan-core_a128r12n1` 为例，编译生成的 image文件保存在 build 目录下。



3.2.2. 烧录镜像

在命令行编译的目录下执行

```
build_sf32lb56-wlan_core_a128r12n1_hcpu\uart_download.bat
```

烧写 build 目录下编译生成的镜像文件。

3.2.3. 改变发射功率

工程默认配置的发射功率是0dbm，可以使用 `ble_tx_pwr_save x` 命令修改发射功率，x是需要修改的发射功率大小，例如改变发射功率为10dbm：`ble_tx_pwr_save 10` 。改完后会自动重启生效。

3.3. 设备初始化

第一次使用开发板板测试蓝牙需要在上电后执行以下初始化命令 (烧写文件系统镜像后蓝牙地址会丢失，也需要重新配置)，命令在 HCPU 的 console 中执行，唤醒 PIN 需接低电平避免 HPSYS 进入低功耗模式，否则 HCPU 无法处理命令

- nvds reset_all 1
- nvds update addr 6 <蓝牙地址>

例如 nvds update addr 6 1234567890C8，注意蓝牙地址分类需要保持一定的格式，建议 C8 不要改动，其他的用户可以自行改变，命令执行完成后按 Reset 键重启设备

3.4. ADV场景

- 注意：测试ADV场景时最好在屏蔽箱或者屏蔽房进行测试，否则有其他设备干扰会导致测试结果偏高

1. ◦ 打开串口调试工具，连接 HCPU 的 console 串口，连接测量设备与被测模块
2. ◦ 板子启动成功后出现如图下图的 log (此时唤醒引脚应保持悬空或者低电平，否则开机就会进入休眠)

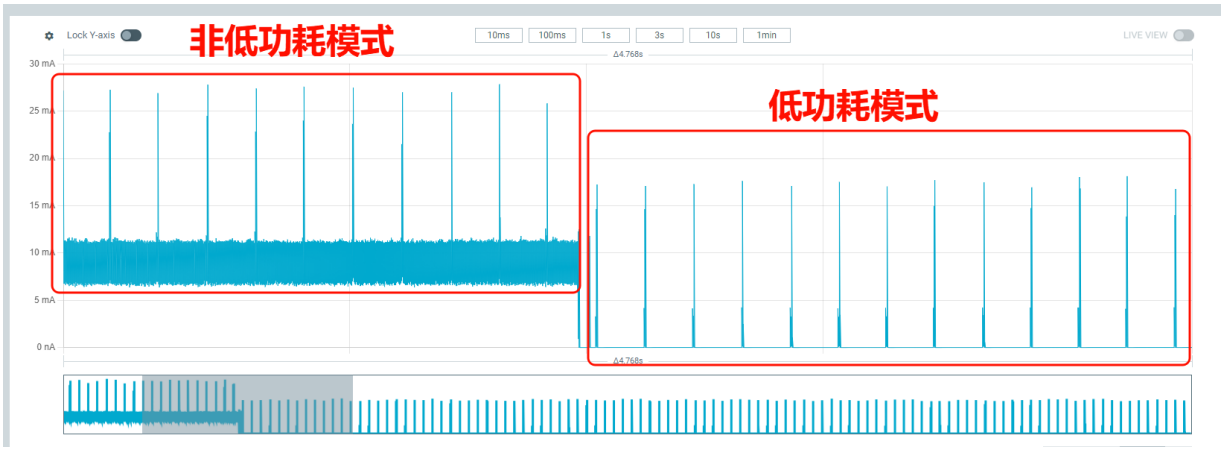
```
 \ | /
- SiFli Corporation
 / | \     build Jan 21 2021, v0.6.2
 2020 - 2021 Copyright by SiFli team
Patch hook install..
[1326] D/DS: data service subscribe (BLE_NV), service (0)
□[32m[1355] I/nvds: Wait nvds service.
□[0m[1379] D/DS: data service subscribe (ANCS), service (2010b6ac)
[1411] D/DS: datac subscribe dev open
□[32m[1464] I/ble_cm: read_bond_infor_from_flash: 7
□[0m□[32m[1502] I/sibles: enable BLE Core..
□[0m□[32m[1525] I/ble_app: status 0
□[0m□[32m[1548] I/nvds: nvds msg_id 3841
□[0m□[32m[1571] I/nvds: get_msg 0
□[0m□[32m[1590] I/nvds: Subscribed nvds service ret 0.
□[0m□[32m[1624] I/ancs_srv: ancs event handler 116

□[0m□[32m[1650] I/ble_cm: BLE_NVDS_AYSNC_FLUSH_TAG_CNF
□[0m□[32m[1678] I/ancs_srv: ancs event handler 112

□[0m□[32m[1704] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF
□[0m□[31m[1728] E/ble_cm: len check error! 210, 0
□[0m□[32m[1753] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF 0x10
□[0m□[32m[1780] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF 0x1
□[0m□[32m[1806] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF 0x17
□[0m□[32m[19753] I/sibles:
BLE ready!

□[0m□[32m[19777] I/ancs_srv: ancs event handler 0
```

1. ◦ 启动后默认的 ADV 周期为 200ms，由于 Inquiry Scan 和 Page Scan 也会自动打开，为测试 BLE 的功耗，需要使用 btskey 命令关闭 Scan，关闭命令参考经典蓝牙的Scan部分。例如，先发送 btskey s 命

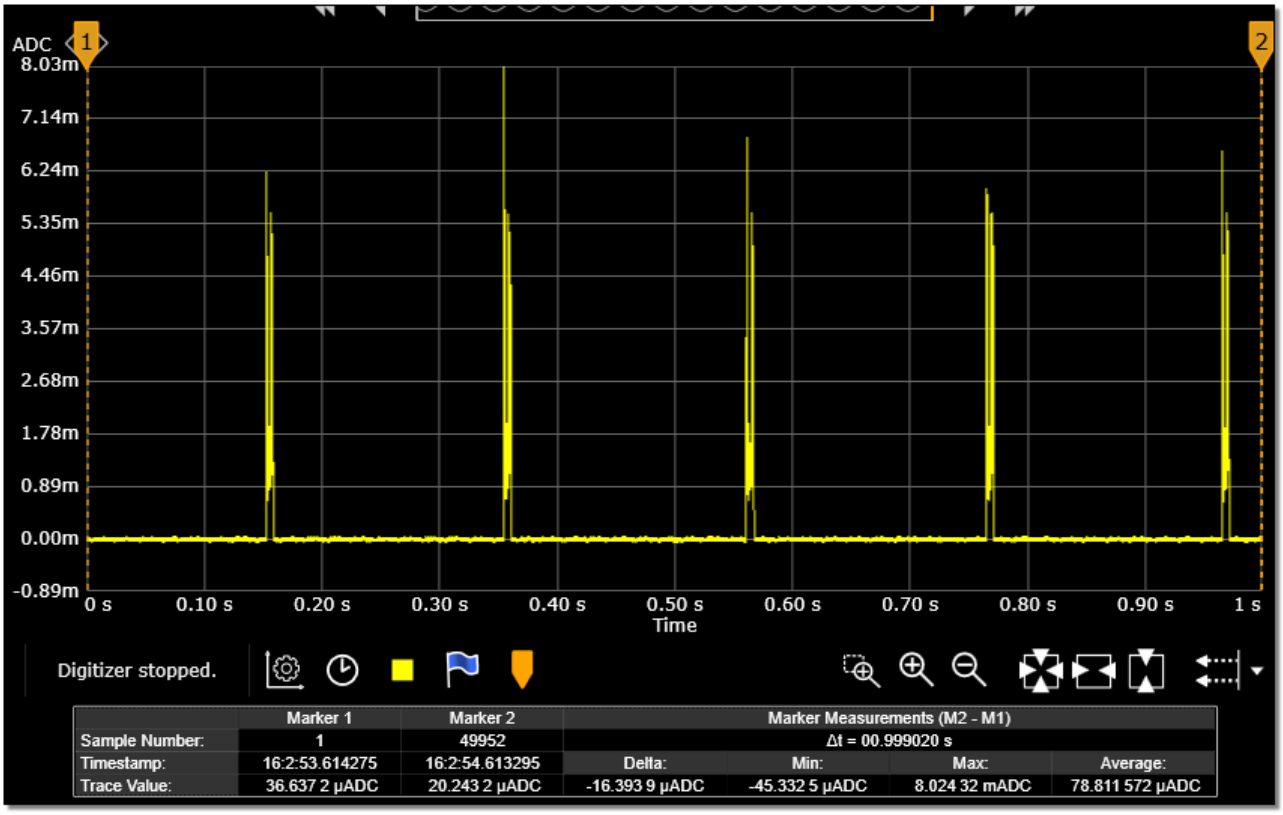
令，如果显示当前位于主菜单，就可以依次发送以下三个命令关闭 Page Scan 和 Inquiry Scan。发送完 btskey0，可以再发送 btskey 4 查询 Scan 的状态。

```
(a) btskey 1
(b) btskey 7
(c) btskey 0
```

1. ◦ 将唤醒 PIN 接高电平，系统进入低功耗模式，电流值如下图明显的下降，测量 200ms 间隔时的电流，低功耗模式下的电流波形见ADV=200ms 的电流波形图。

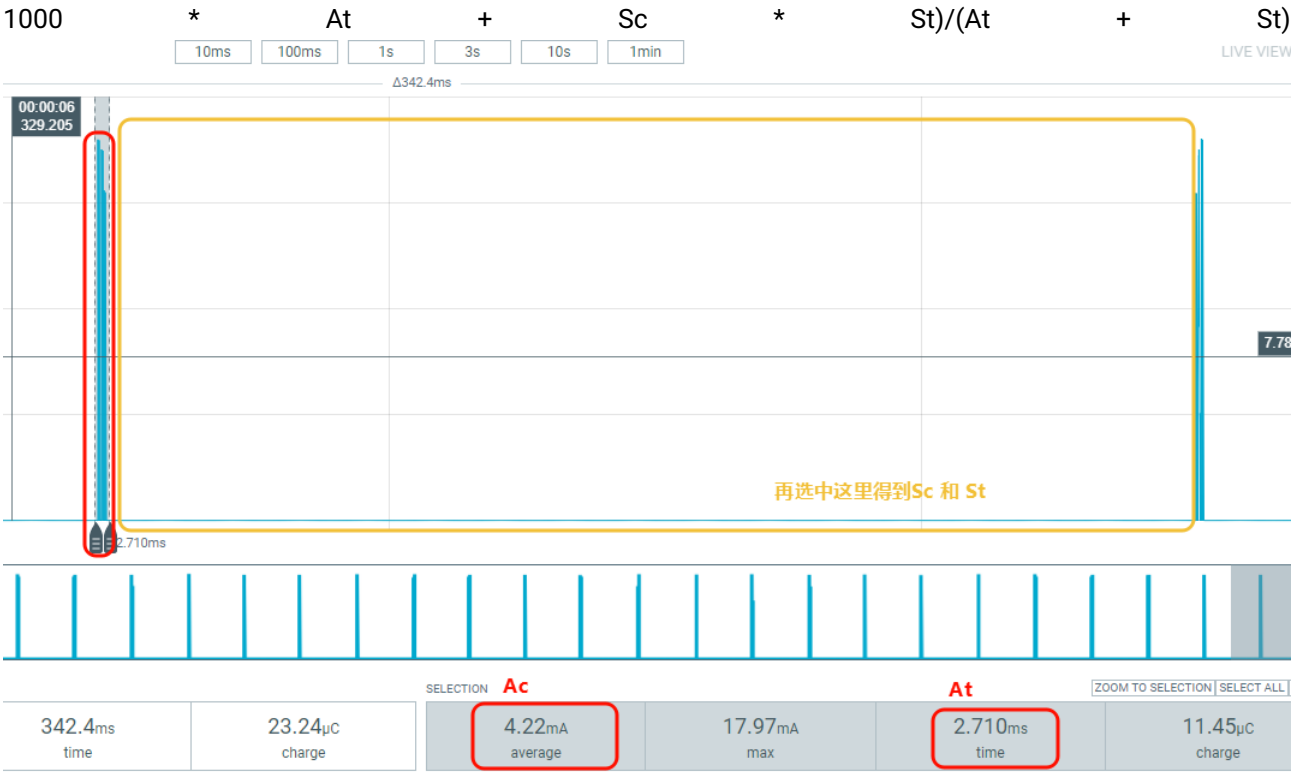


进入低功耗模式电流变化



ADV=200ms 的电流波形

挑选一个周期内的ADV的工作电流选中，得到一次工作的持续时间 At 与平均电流 AC，再得到当前周期内的休眠时间持续 St 与休眠平均电流 Sc，代入如下公式计算可得出平均电流消耗。 计算公式：平均电流 = (Ac *



选择周期

1. ◦ 将唤醒 PIN 接低电平，系统退出低功耗模式，在 console 里发送命令ble_config adv 500，将 ADV 间隔改为 500ms
2. ◦ 将唤醒 PIN 接高电平，系统再次进入低功耗模式，测量 500ms 间隔时的电流

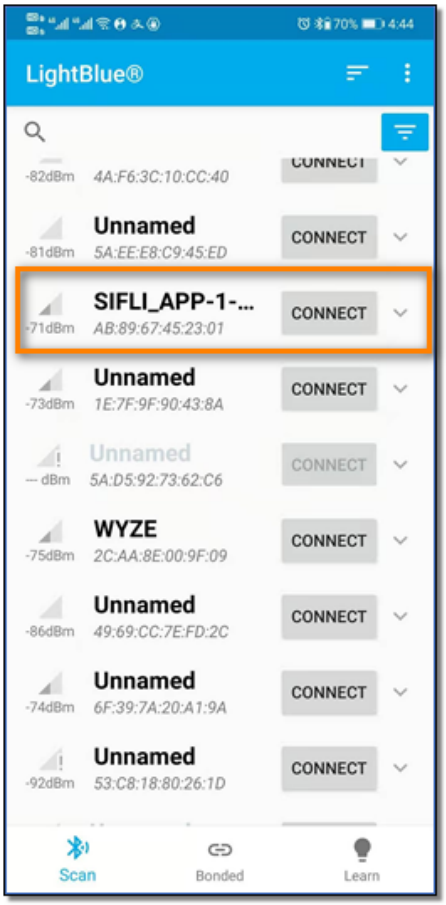
重复步骤5和6，可以测量ADV间隔为50ms、100ms、500ms和1000ms 时的电流。

3.5. 连接场景

1. ◦ 打开串口调试工具，连接 HCPU 的 console 串口，连接测量设备与被测模块
2. ◦ 唤醒 PIN 接低电平，开发板重新上电复位成功后出现如下图的 log。类似ADV章节量 ADV 电流，也需要关闭 BT Scan

```
 \ | /
- SiFli Corporation
 / | \     build Jan 21 2021, v0.6.2
 2020 - 2021 Copyright by SiFli team
Patch hook install..
[1326] D/DS: data service subscribe (BLE_NV), service (0)
□[32m[1355] I/nvds: Wait nvds service.
□[0m[1379] D/DS: data service subscribe (ANCS), service (2010b6ac)
[1411] D/DS: datac subscribe dev open
□[32m[1464] I/ble_cm: read_bond_infor_from_flash: 7
□[0m□[32m[1502] I/sibles: enable BLE Core..
□[0m□[32m[1525] I/ble_app: status 0
□[0m□[32m[1548] I/nvds: nvds msg_id 3841
□[0m□[32m[1571] I/nvds: get_msg 0
□[0m□[32m[1590] I/nvds: Subscribed nvds service ret 0.
□[0m□[32m[1624] I/ancs_srv: ancs event handler 116

□[0m□[32m[1650] I/ble_cm: BLE_NVDS_AYSNC_FLUSH_TAG_CNF
□[0m□[32m[1678] I/ancs_srv: ancs event handler 112

□[0m□[32m[1704] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF
□[0m□[31m[1728] E/ble_cm: len check error! 210, 0
□[0m□[32m[1753] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF 0x10
□[0m□[32m[1780] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF 0x1
□[0m□[32m[1806] I/ble_cm: BLE_NVDS_ASYNC_READ_CNF 0x17
□[0m□[32m[19753] I/sibles:
BLE ready!

□[0m□[32m[19777] I/ancs_srv: ancs event handler 0
```

1. ◦ 在手机上打开如下图的 LightBlue 软件，在 Scan 列表找到名为 SIFLI_APP 的设备，点击 CONNECT 连接设备

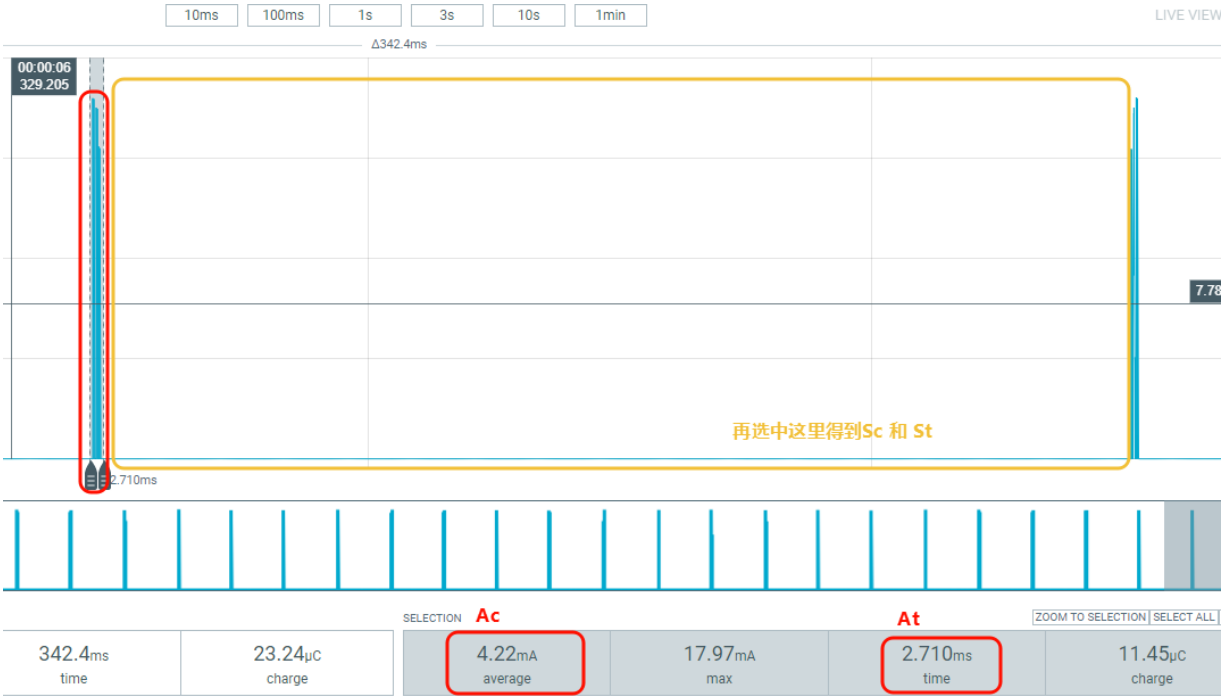


1. ◦ 连接成功后 BLE 进入连接状态，默认的连接周期为 15ms

2. ◦ 将唤醒 PIN 接低电平，系统退出低功耗模式，在 console 里发送命令ble_config conn 50，将连接周期改为 50ms，确认 console 出现如下图打印，其中Updated connection interval：40表示 1.25ms 单位的连接间隔，40×1.25=50ms，如果未出现打印，则表示参数更新失败，需要再次发送命令，同时观察电流波形确认间隔是否更新成功

```
msh >
msh >ble_config conn 50
□□□\0Q\0X鏖□a□□ □□\0(\0(\0\0\0?□\0\0□□□□\0R\0晒?b□□□\0□□ msh >
msh >□[32m[2017749] I/sibles: GAP event 3584
□[0m[2017772] D/BLE_GAP: GAPC event status, op: 9, ret 0
□[32m[2017798] I/ancs_srv: ancs event handler 63

□[0m□[32m[2017827] I/sibles: GAP event 3601
□[0m□[32m[2017850] I/ble_app: Updated connection interval :40
□[0m□[32m[2017880] I/ancs_srv: ancs event handler 62

□[0m□[32m[2017908] I/ble_cm: BLE_GAP_UPDATE_CONN_PARAM_IND
□[0m□[32m[2017937] I/ble_cm: connection_manager_event_process 0x49
□[0m□[32m[2017970] I/ancs_srv: ancs event handler 170
```

1. ◦ 将唤醒 PIN 接高电平，系统进入低功耗模式，与 ADV 电流类似，挑选一个周期内的工作电流选中，得到一次工作的持续时间 At 与平均电流 AC，再得到当前周期内的休眠时间持续 St 与休眠平均电流 Sc，代入如下公式计算可得出平均电流消耗。 计算公式：平均电流 = (Ac * 1000 * At + Sc * St)/(At + St)



选择周期

重复步骤 5 和 6，可以测量连接周期为50ms、100ms、200ms、500ms和1000ms时的电流。

3.6. 测量结果汇总

表格中 ADV 与 Connection 的电流值为该模式的增量电流，standby 为睡眠电流，实际的平均电流等于增量电流与睡眠电流之和

3.6.1. 测试结果表 - 3.8V转换数据

模式	间隔 (ms)	电源电压：3.8V TXpower 0dBm (uA)	电源电压：3.8V TXpower 4dBm (uA)	电源电压：3.8V TXpower 10dBm (uA)
ADV	50	172.9	184.8	360.8
	100	83.6	102.8	186.1
	200	42.7	52.3	94.9
	500	17.7	21.2	39.2
	1000	8.9	10.6	19.5
Connection	50	122.4	130.7	147.9
	100	60.8	64.8	74.6
	200	30.4	32.4	37.3
	500	12.2	13	15.0
	1000	6.1	6.5	7.6
Standby	-	2		

1. ◦ 电源：1.8V对PVDD供电，3.3V对3V3_ANA + 3V3_BRF供电。
2. ◦ 以上电源电压3.8V的功耗是依据1.8V及3.3V两路供电的测试数据按照效率计算的结果（计算公式：I3.8v = I1.8v * 1.8 / 90% / 3.8 + I3.3v）。

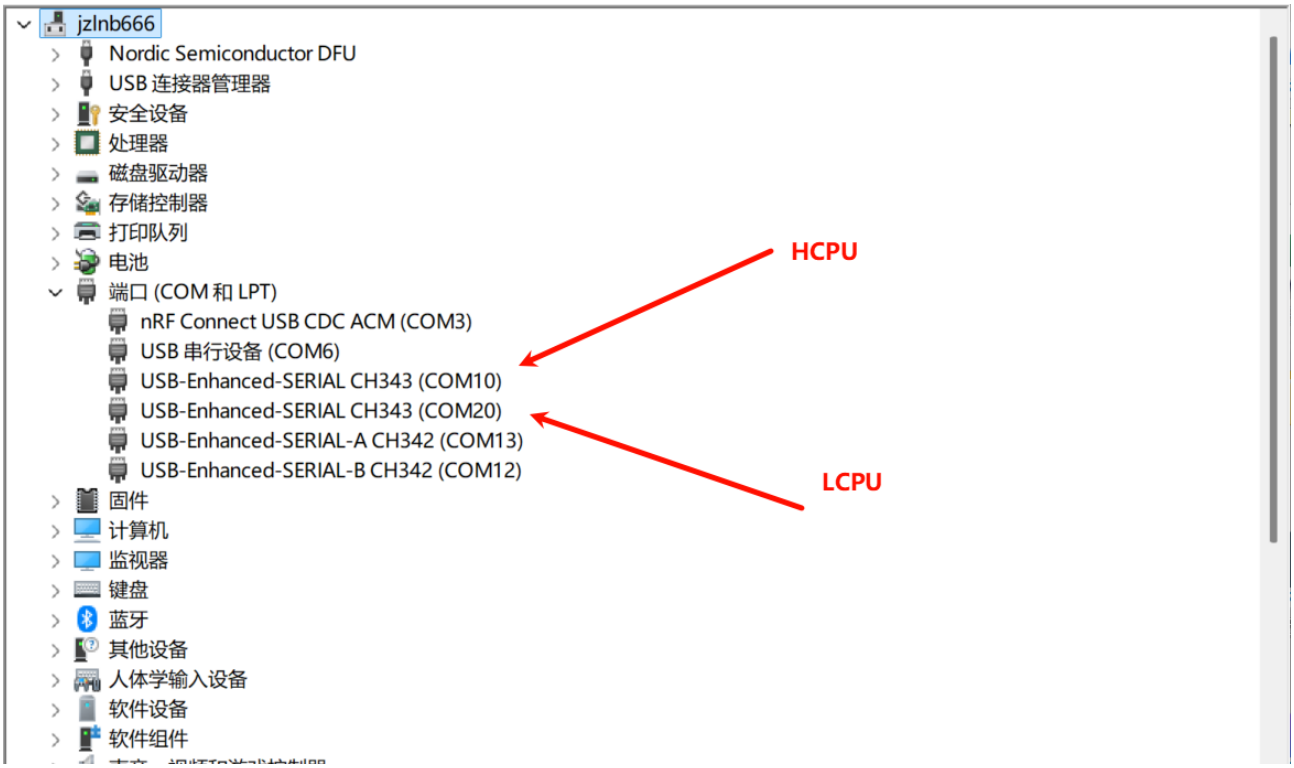
4. 经典蓝牙功耗测试

经典蓝牙 (Bluetooth Classic) 提供了比 BLE 更高的数据传输速率，适用于音频传输、数据同步等应用场景。本章节详细介绍经典蓝牙在各种工作模式下的功耗特性。

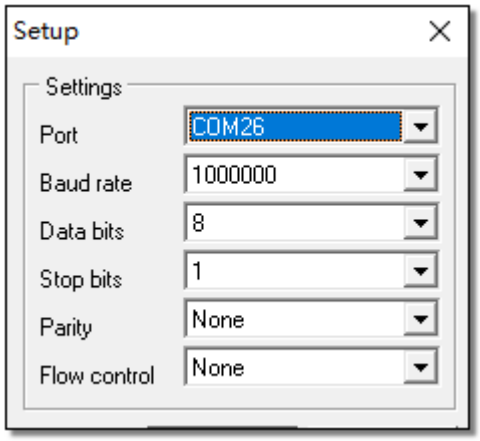
4.1. 测试说明

使用 BT 例程可以测试 Scan 和 Sniff 模式的功耗（发射功率0dBm），系统上电后测试例程自动开启 Scan 和 ADV，使用手机连接蓝牙设备，在 HCPU 的 console 里可以发送命令修改配置，发送的命令都需以回车换行结尾。

PC 与调试板使用 USB Type-C 线连接后会枚举出两个串口，分别作为HCPU以及LCPU的console，如下图所示。



串口设置参见下图，波特率均设置为 1000000。



为便于控制测试条件，使用PA52作为HCPU的唤醒PIN。当唤醒PIN不为高电平时HCPU无法进人低功耗模式，此时可以通过console给HCPU发送命令执行指定任务，当唤醒PIN接高电平（即1.8V电压，下文如未特别说明，高电平均指1.8V电压）时，HCPU进人低功耗模式，此时HCPU无法响应console命令。

测试例程LCPU的主频为24MHz，HCPU与LCPU使用的低功耗模式为Standby模式。 HCPU使用btskey命令操作菜单修改配置。

4.2. 例程编译与烧录

4.2.1. 编译

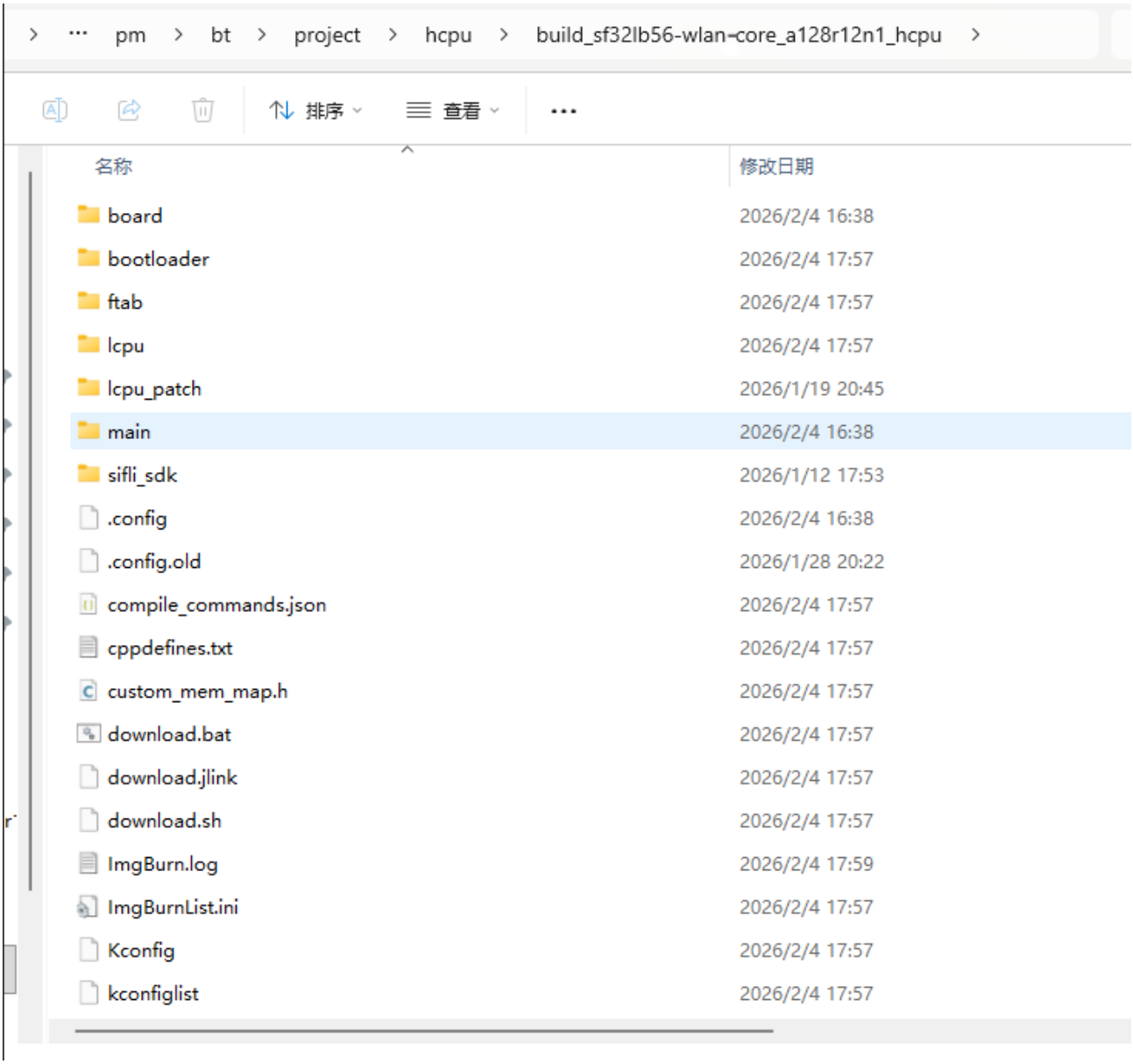
如果使用已编译好的 image 文件，可以直接跳到烧录部分进行烧录开始测试。 进入example\pm\coremark\project\hcpu目录 如果使用的开发板型号为 `sf32lb56-wlan-core_a128r12n1`,就使用如下命令进行编译

```
scons --board=sf32lb56-wlan-core_a128r12n1 -j8
```

如果使用的开发板型号为 `sf32lb56-wlan-core_n16r12n1`,就使用如下命令进行编译

```
scons --board=sf32lb56-wlan-core_n16r12n1 -j8
```

以 `sf32lb56-wlan-core_a128r12n1` 为例，编译生成的 image文件保存在 build 目录下。



4.2.2. 烧录镜像

在命令行编译的目录下执行

```
build_sf32lb56-wlan-core_a128r12n1_hcpu\uart_download.bat
```

烧写 build 目录下编译生成的镜像文件。

4.2.3. 改变发射功率

工程默认配置的发射功率是0dbm，可以使用 `ble_tx_pwr_save x` 命令修改发射功率，x是需要修改的发射功率大小，例如改变发射功率为10dbm：`ble_tx_pwr_save 10` 。改完后会自动重启生效。

4.3. 设备初始化

第一次使用开发板测试蓝牙需要在上电后执行以下初始化命令，命令在 HCPU的 console中执行，唤醒 PIN需接低电平避免 HPSYS进入低功耗模式，否则 HCPU无法处理命令

1. nvds reset_all 1

2. nvds update addr 6 <蓝牙地址>

例如 nvds update addr 6 1234567890C8，注意蓝牙地址分类需要保持一定的格式，建议 C8不要改动，其他的用户可以自行改变，命令执行完成后按 Reset键重启设备

4.4. Scan

1. ◦ 打开串口调试工具，连接 HCPU的 console串口，连接测量设备与被测模块

2. ◦ 唤醒 PIN接低电平，重新上电复位开发板，启动成功后出现如下图的 log

```
 \ | /
- Si
[20:18:56.320]收←◆Fli Corporation
 / | \     build on Oct 28 2022, 2.0.4 build cda680
 2020 - 2022 Copyright by SiFli team
mount /dev sucess

[20:18:57.747]收←◆[1] I/drv.rtc main: PSCLR=0x80000100 DivAI=128 DivAF=0 B=
[1] I/drv.rtc main: Init RTC, wake = 0

[134] I/drv.audprc main: init 00 ADC_PATH_CFG0 0x924

[134] I/drv.audprc main: HAL_AUDPRC_Init res 0

[136] I/drv.audcodec main: HAL_AUDCODEC_Init res 0

[140] I/audio main: Device audcodec opened[handle = 20031e64]

call par CFG1(35bb)
fc 9, xtal 2000,
[20:18:57.803]收←◆pll 2044
call par CFG1(35bb)
fc 9, xtal 2000, pll 2046
mbox_stack:20020000,1024
proc_stack:20020400,4096
Register mpi3 to mtd device with base addr 0x64300000
disk_init:0
init
mount fs on mpi3 to root success
ble db init: sector_size 4096 size 16384
[689] I/nvds main: read sleep time 4500
[694] D/DS main: data service subscribe (BT_EN), service (0)
[698] I/mw.sys ds_proc: turn on lcpu
msh />[40645] I/NO_TAG ds_proc: BT enable LCPU 0.

[20:18:57.831]收←◆[42814] I/NO_TAG ds_proc: BT enabled result 0

[20:18:58.562]收←◆
##########################################################
##                                                      ##
##           BTS2 Demo Main Menu                        ##
##    1. Generic Command                                ##
##    2. SPP Client                                     ##
##    3. SPP Server                                     ##
##    4. HFP HF                                         ##
##    6. A2DP Sink                                      ##
##    p. AVRCP                                          ##
##    s. Show Menu                                      ##
##    q. Exit                                           ##
##                                                      ##
##########################################################

ble db init: sector_size 4096 size 16384
[66439] I/nvds main: read sleep time 4500
[66925] I/ble_cm main: read_bond_infor_from_flash: 0
[66926] I/sibles main: enable BLE Core..

[20:18:59.155]收←◆<< Local device addreess: CDAB:78:563412
>> SPP enabled
>> Handfree enable success
>> AVCTP enabled
>> Headset enable success
<< Local LMP version: 12
<< Local LMP subversion: 0
<< Local controller manufacturer name id: 2636
<< Local device name changed!
<< Class of device has been changed!
<< Write scan enable success
<< Local device name: SifliDemo-12-34-56-78-ab-cd
```

3. ◦ 启动后 ADV、Inquiry Scan和 Page Scan都会打开，为测试 Scan的功耗，需先关闭 ADV，关闭命令为

```
diss adv_stop
```

1. ◦ 开机后默认处于 BTS主菜单，通过发送 btskey命令可以配置打开或关闭 Scan。发送btskey s命令显示当前菜单，再按菜单提示发送命令进入子菜单。比如，在主菜单下，依次发送如下三个命令可以打开 Page Scan并关闭 Inquiry Scan

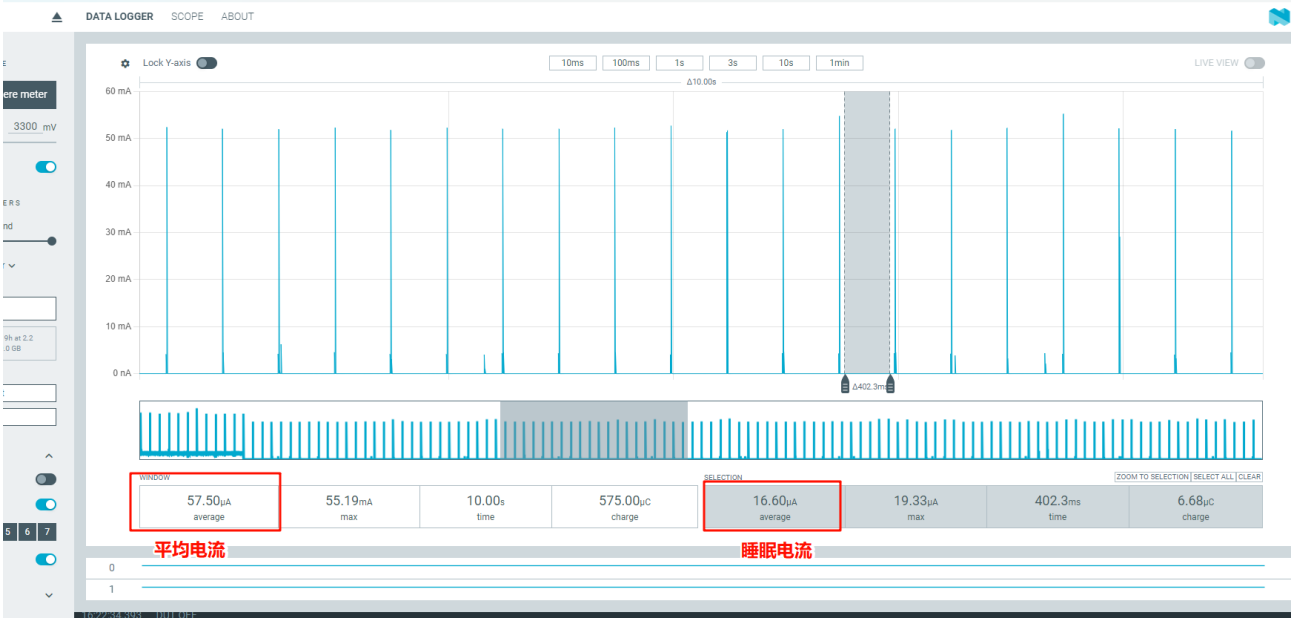
```
(a) btskey 1
(b) btskey 7
(c) btskey 2
```

```
[21:29:38.944]发→◇btskey s
□
[21:29:38.971]收←◆
##########################################################
##                                                      ##
##           BTS2 Demo Main Menu                        ##
##    1. Generic Command                                ##
##    2. SPP Client                                     ##
##    3. SPP Server                                     ##
##    4. HFP HF                                         ##
##    6. A2DP Sink                                      ##
##    p. AVRCP                                          ##
##    s. Show Menu                                      ##
##    q. Exit                                           ##
##                                                      ##
##########################################################

btskey s
s

msh />
msh />
[21:29:47.349]发→◇btskey 1
□
[21:29:47.368]收←◆
##########################################################
##                                                      ##
##           Generic Command Menu                       ##
##    1. Inquiry start                                  ##
##    2. Inquiry cancel                                 ##
##    3. Select device from Inquiry list                ##
##    4. Sc menu                                        ##
##    5. Local device info                              ##
##    6. Get remote device info                         ##
##    7. Scan mode                                      ##
##    8. Link menu                                      ##
##    9. Service Browse                                 ##
##    a. Set IO Settings                                ##
##    s. Show Menu                                      ##
##    r. Return to last menu                            ##
##                                                      ##
##########################################################

btskey 1
1

msh />
msh />
[21:30:05.706]发→◇btskey 7
□
[21:30:05.725]收←◆
##########################################################
##                                                      ##
##           Scan Mode Select Menu                      ##
##    0. No scans enabled                               ##
##    1. Inquiry scan enabled and Page scan disabled    ##
##    2. Page scan enabled and Inquiry scan disabled    ##
##    3. Inquiry scan and Page scan enabled             ##
##    4. Read scan mode                                 ##
##    s. Show Menu                                      ##
##    r. Return to last menu                            ##
##                                                      ##
##########################################################
```

1. ◦ 发送 BTS命令配置设备只发送 Inquiry Scan或者 Page Scan，唤醒 PIN接高电平，系统进入低功耗模式。

挑选一个周期内的scan工作电流选中，得到一次工作的持续时间 At 与平均电流 AC，再得到当前周期内的休眠时间持续 St 与休眠平均电流 Sc，代入如下公式计算可得出Inquiry Scan或者 Page Scan平均电流消耗。 计算

公式：平均电流 = (Ac * 1000 * At + Sc * St)/(At + St)



选择周期

测试程序的 Page Scan周期为 1.28秒，Inquiry Scan周期为 2.56秒，所以Inquiry Scan的增量电流为 Page Scan 的一半。

1. ◦ 唤醒 PIN 接低电平，发送 BTS 命令配置设备同时发送 Inquiry Scan 和 Page Scan，再把唤醒 PIN 接高电平，系统进入低功耗模式，测量 1分钟的平均电流，记为 both scan 的平均电流 C1，同时再测量两个峰之间的底电流，记为睡眠电流 C2，both Scan的增量电流即为 C=C1-C2。

4.5. Sniff Mode

1. ◦ 参考Scan章节的步骤复位开发板并关闭 ADV

2. ◦ 手机连接开发板对应的蓝牙设备，点击后弹出配对窗口，点确认后配对成功，蓝牙设备进入已配对设备列表，并显示已连接状态,如下图：

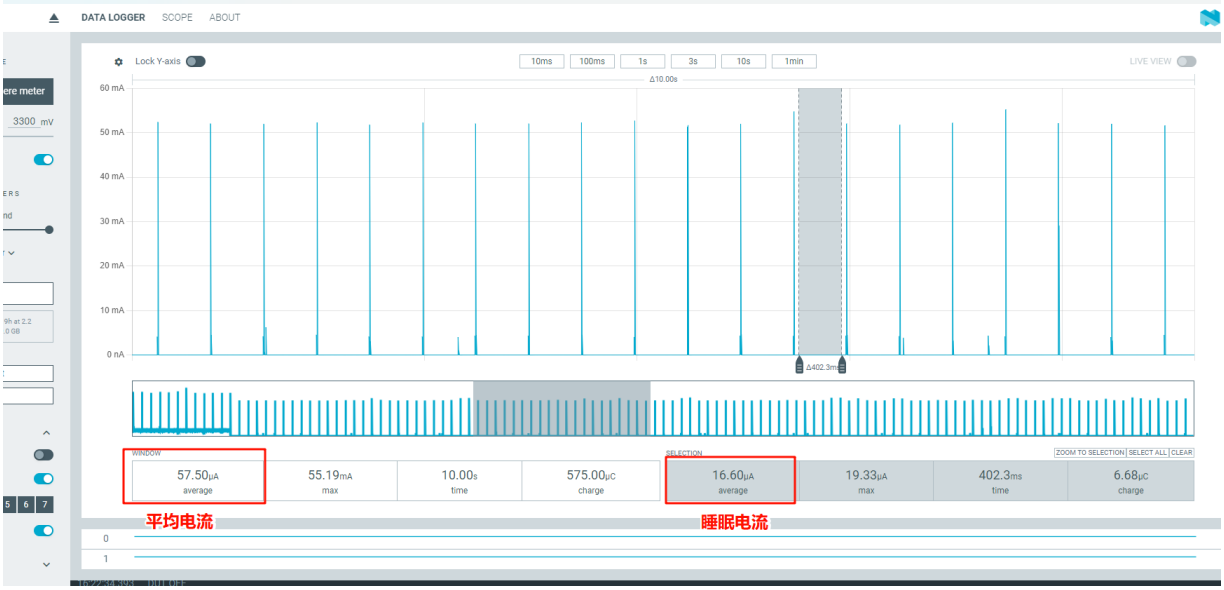
1. ◦ 连接完成后，等待一会儿，console 窗口会打印 »Sniff mode，表示设备已进入 Sniff 模式

2. ◦ 发送 btskey 命令关闭 Scan，具体步骤如下：

```
(a) btskey s: 查看当前菜单
(b) 如果是 HFP HP Menu，则发送
(c) btskey r: 返回上一级菜单 BTS2 Demo Main Menu，再顺序发送以下命令
(d) btskey 1: 选择 Generic Command
```

```
(e) btskey 7: 选择 Scan mode
(f) btskey 0: 关闭 scan
```

1. ◦ 与 Scan 电流的测量方法类似，挑选一个周期内的sniff工作电流选中，得到一次工作的持续时间 At 与平均电流 AC，再得到当前周期内的休眠时间持续 St 与休眠平均电流 Sc，代入如下公式计算可得出sniff平均电流消耗。 计算公式：平均电流 = (Ac * 1000 * At + Sc * St)/(At + St)



选择周期

1. ◦ Sniff 模式的间隔与 attemp 次数与手机有关，本次测试中使用 一加ace 6，attemp 次数为4。

4.6. 测量结果汇总

表中的 Scan和Snif电流为增量电流，平均电流需加上 Standby的底电流，计算方法参照备注中的计算示例。

4.6.1. BT 功耗

4.6.1.1. BT Sniff Mode 测试结果

模式	时间周期 (s)	电源电压：3.8V TXpower 0 dBm (uA)	电源电压：3.8V TXpower 4 dBm (uA)	电源电压：3.8V TXpower 10 dBm (uA)	单位
BT Sniff Mode	50	214.9	222.8	253.6	uA
	100	107	111.4	211.6	
	200	53.4	55.7	63	

模式	时间周期 (s)	电源电压：3.8V TXpower 0 dBm (uA)	电源电压：3.8V TXpower 4 dBm (uA)	电源电压：3.8V TXpower 10 dBm (uA)	单位
	500	21.3	22.3	25.1	
	1000	10.7	11.1	12.6	
Standby	-	2			uA

4.6.1.2. Scan 测试结果

场景	电源电压 3.8V 电流 (uA)	单位
Inquiry enable, Page disable	14.9	uA
Page enable, Inquiry disable	29.7	
Both scans enable	45.6	
Standby	2	uA

1. Inquiry Scan 每 2.56s 接收 16.87ms，Page Scan 每 1.28s 接收 16.87ms。
2. ◦ 电源：1.8V对PVDD供电，3.3V对3V3_ANA + 3V3_BRF供电。
3. ◦ 以上电源电压3.8V的功耗是依据1.8V及3.3V两路供电的测试数据按照效率计算的结果（计算公式：I3.8v = I1.8v * 1.8 / 90% / 3.8 + I3.3v）。
4. ◦ 计算示例： Standby mode bt 500ms sniff @TXpower10dBm: =23.1+0=23.1uA

5. 处理器性能与功耗测试

本章节专注于 SF32LB56x 处理器在不同计算负载下的功耗特性分析，包括标准化基准测试和基础循环测试。

5.1. 测试目标

通过标准化的处理器测试，评估芯片在不同计算强度下的功耗表现：

- Coremark 基准测试：业界标准的 CPU 性能基准测试
- While Loop 测试：基础循环操作的功耗基线测试
- 关机模式测试：最低功耗状态的功耗测量

5.2. 测试说明

CoreMark 例程测试以下场景下的功耗:

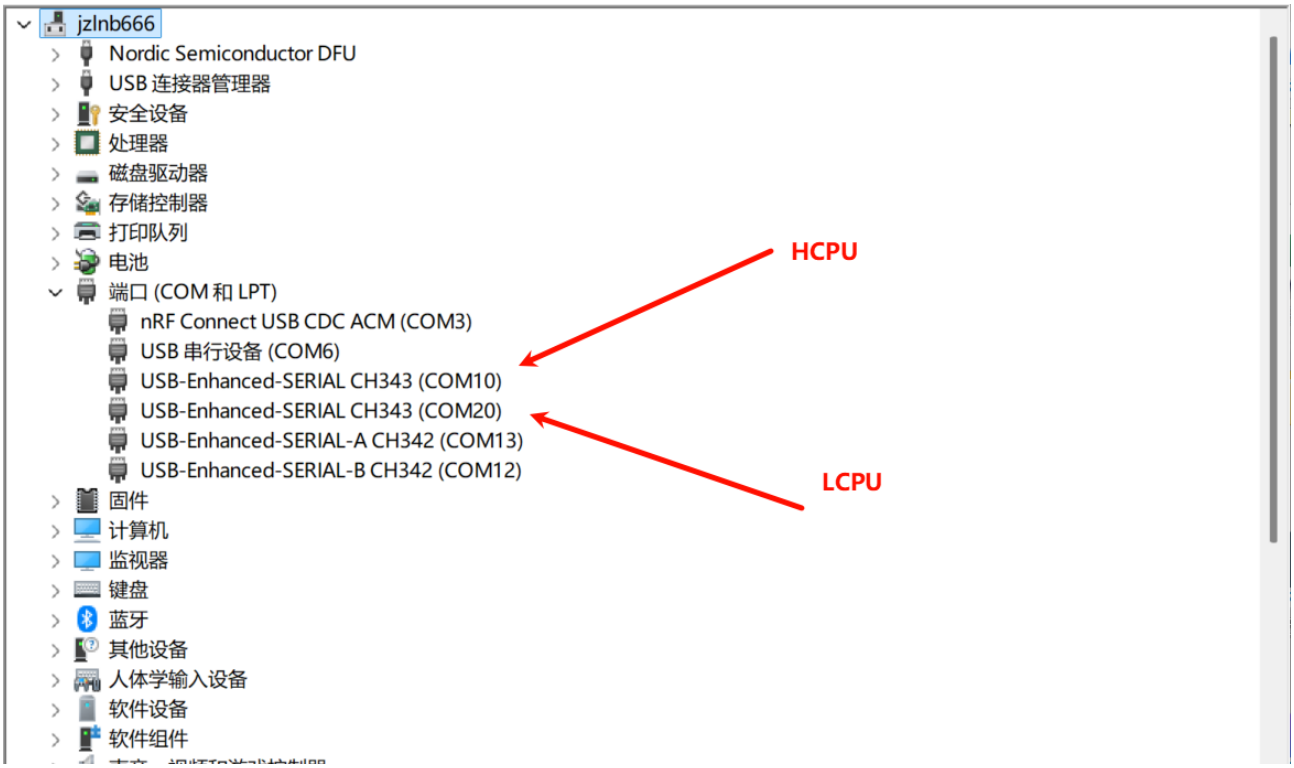
- 一个核执行 CoreMark 基准测试程序
- 一个核执行一段时间的 while 循环，循环中执行 nop 指令
- 系统关机，可由 RTC 定时唤醒
- 系统关机，由唤醒引脚唤醒

为便于控制测试条件，使用PA52作为HCPU的唤醒PIN。当唤醒PIN不为高电平时HCPU无法进人低功耗模式，此时可以通过console给HCPU发送命令执行指定任务，当唤醒PIN接高电平（即1.8V电压，下文如未特别说明，高电平均指1.8V电压）时，HCPU进人低功耗模式，此时HCPU无法响应console命令。

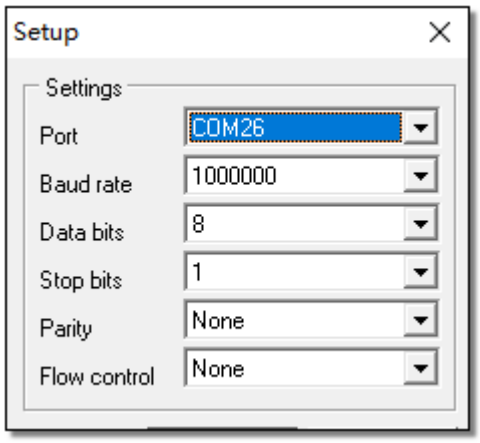
LCPU始终不进人低功耗模式，如果启动了LCPU，当不执行任务时LCPU处于WFI状态，可以响应来自console的命令，未启动LCPU时，则认为LCPU处于halt状态，无法处理console命令。

需要注意发送的命令都要以回车换行符结尾。

PC 与调试板使用 USB Type-C 线连接后会枚举出两个串口，分别作为HCPU以及LCPU的console，如下图所示。



串口设置参见下图，波特率均设置为 1000000。



非关机场景下，当一个核在执行指定任务时，另外一个核需进人如下状态以保证测量值的准确

- HCPU 执行任务时，LCPU 处于wfi 状态 (hcpu的console端发送 lcpu on 命令)
- LCPU执行任务时，HCPU进人Standby低功耗模式 (唤醒pin接高电平)

HCPU 可使用以下命令启动相应任务：

- run_coremark freq_in_mhz：修改主频并执行 CoreMark，freq_in_mhz是以 MHz 为单位的频率 例如：run_coremark 48，以 48MHz 主频执行 CoreMark
- run_while_loop freq_in_mhz：修改主频并执行一段时间 while loop，freq_in_mhz是以 MHz 为单位的频率 例如：run_while_loop 48，以 48MHz 主频执行 while loop

HCPU还支持以下命令：

- lcpu on:启动LCPU，启动成功后LCPU可以接收console命令执行指定任务
- shutdown[wakeup_time_in_sec]:关机，wakeup_time_in_sec为可选参数，单位为秒，表示关机后多久自动开机，如果不带参数，则关机后只能被唤醒引脚唤醒

注：由于编译采用 Ofast 优化等级，跑分结果仅作参考，并不能达到 Omax 时的最高分。

5.3. 例程编译与烧录

5.3.1. 编译

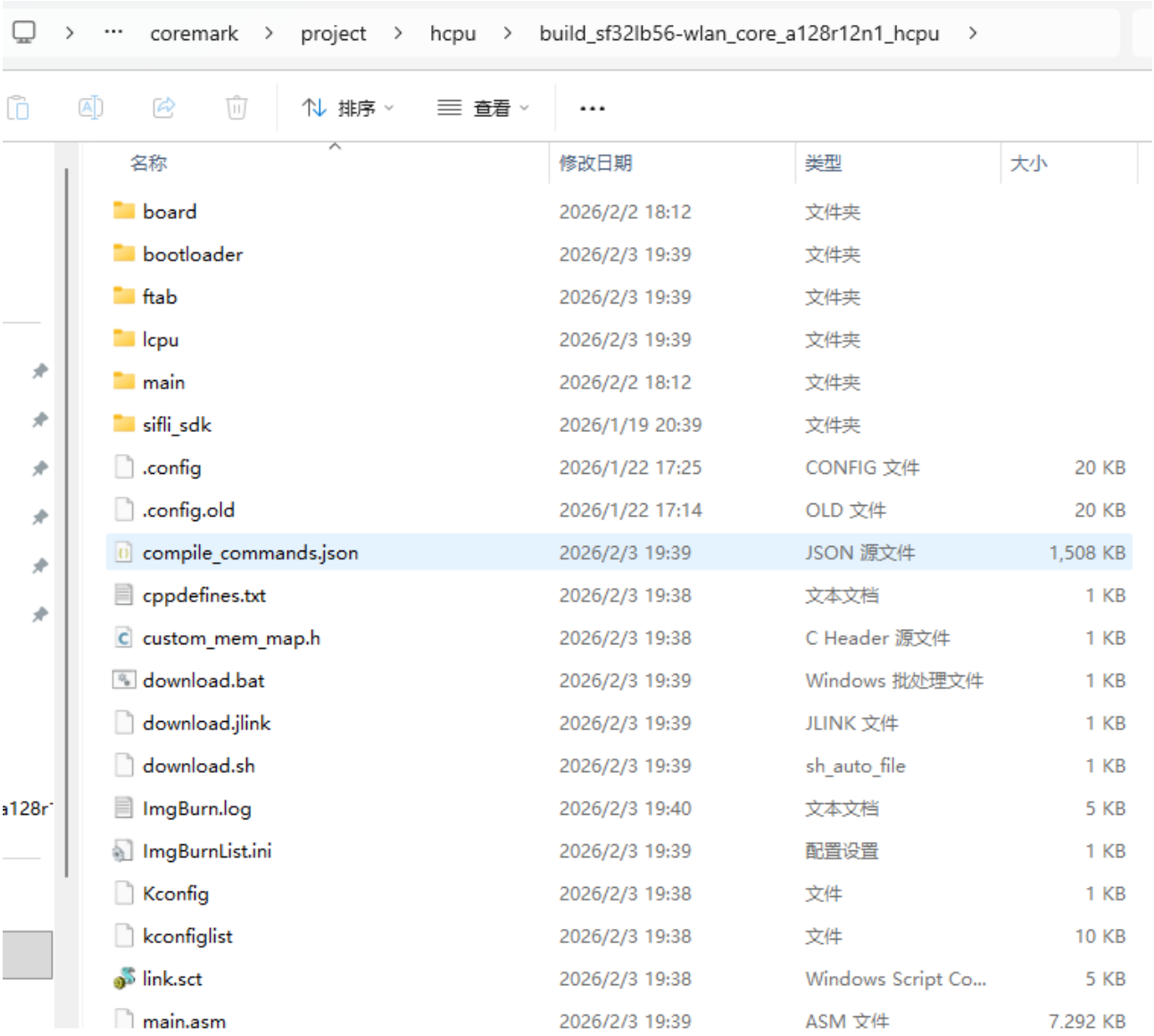
如果使用已编译好的 image 文件，可以直接跳到烧录部分进行烧录开始测试。 进入example\pm\coremark\project\hcpu目录 如果使用的开发板型号为 `sf32lb56-wlan-core_a128r12n1`,就使用如下命令进行编译

```
scons --board=sf32lb56-wlan-core_a128r12n1 -j8
```

如果使用的开发板型号为 `sf32lb56-wlan-core_n16r12n1`,就使用如下命令进行编译

```
scons --board=sf32lb56-wlan-core_n16r12n1 -j8
```

以 `sf32lb56-wlan-core_a128r12n1` 为例，编译生成的 image文件保存在 build 目录下。



5.3.2. 烧录镜像

在命令行编译的目录下执行

```
build_sf32lb56-wlan-core_a128r12n1_hcpu\uart_download.bat
```

烧写 build 目录下编译生成的镜像文件。

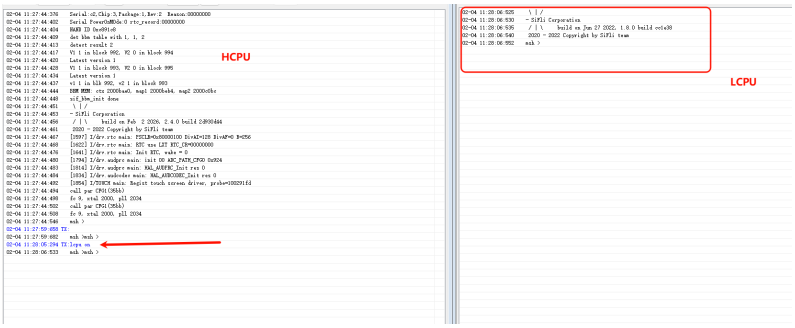
5.4. CoreMark

5.4.1. HPSYS

1. ◦ 打开串口调试工具，连接 HCPU 与 LCPU 的 console 串口，连接测量设备与被测模块

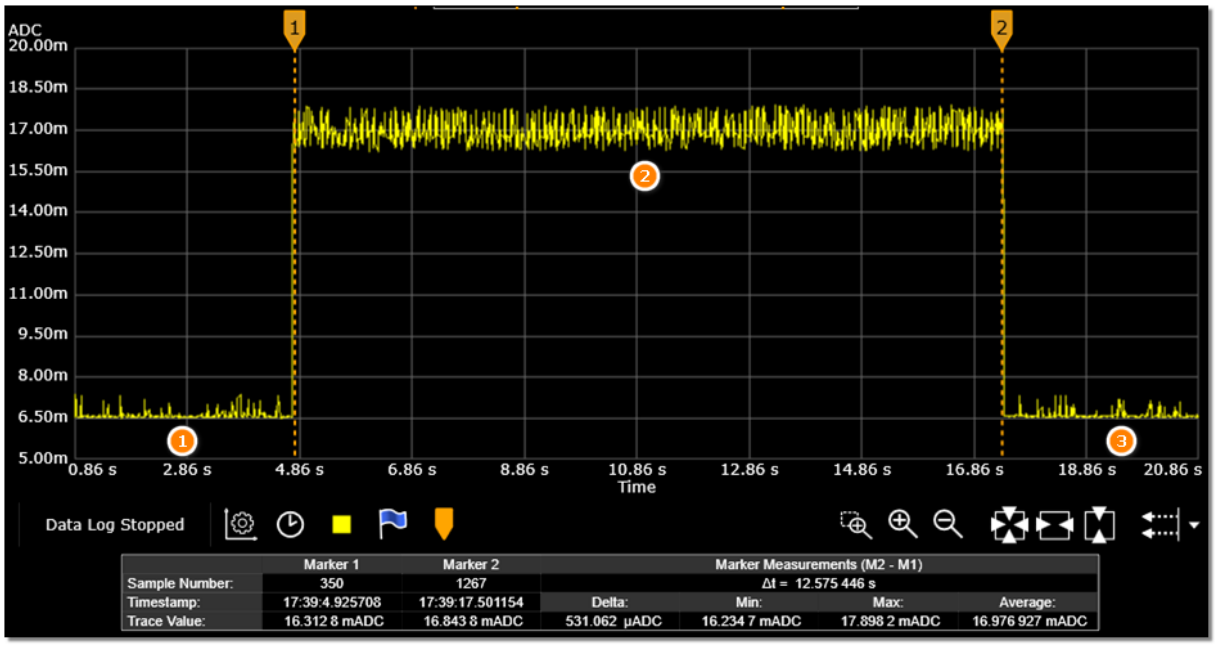
2. ◦ 复位，启动成功后在 HCPU 的 console 中出现如下图的 log

1. ◦ 发送命令 lcpu on，指示LCPU进入 wfi 模式，LCPU启动后会显示如下图所示的 log



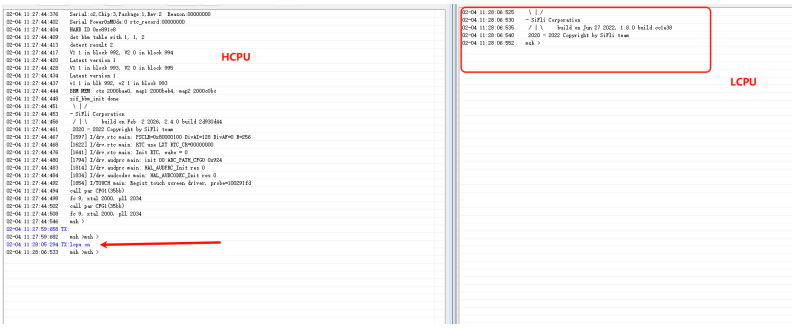
1. ◦ 发送命令run_coremark 240得到平均电流C1，发送run_coremark 192得到平均电流C2，得到240MHz和192MHz的增量电流 C=(C1-C2)/(240-192)

2. ◦ 如下图所示，阶段 1 是 HCPU 跑在 192MHz 主频时 WFI 模式下的电流波形，开始执行 CoreMark后进入阶段 2，电流上升并保持至测试结束，阶段 3 为回到 WFI 模式的电流波形



5.4.2. LPSYS

1. ◦ 打开串口调试工具，同时连接HCPU和LCPU的 console串口，连接测量设备与被测模块
2. ◦ HCPU的串口窗口发送命令 lcpu on，指示LCPU进入 wfi 模式，LCPU启动后会显示如下图所示的 log



1. ◦ 将唤醒PIN接高电平使HPSYS进人低功耗模式
2. ◦ 在LCPU的console中发送命令run_coremark 48，指示LCPU在48MHz频率下执行CoreMark基准测试，测量此时的平均电流C1
3. ◦ 发送命令run_coremark 24，指示LCPU在24MHz下执行CoreMark 基准测试，测量此时的平均电流C2，由此得到每MHz的电流增量为C=(C1-C2)/(48-24)

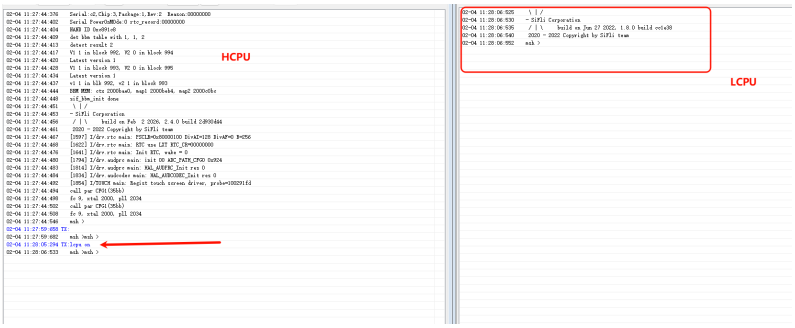
5.5. While Loop

5.5.1. HPSYS

1. ◦ 打开串口调试工具，连接 HCPU 与 LCPU 的 console 串口，连接测量设备与被测模块
2. ◦ 复位，启动成功后在 HCPU 的 console 中出现如下图的 log

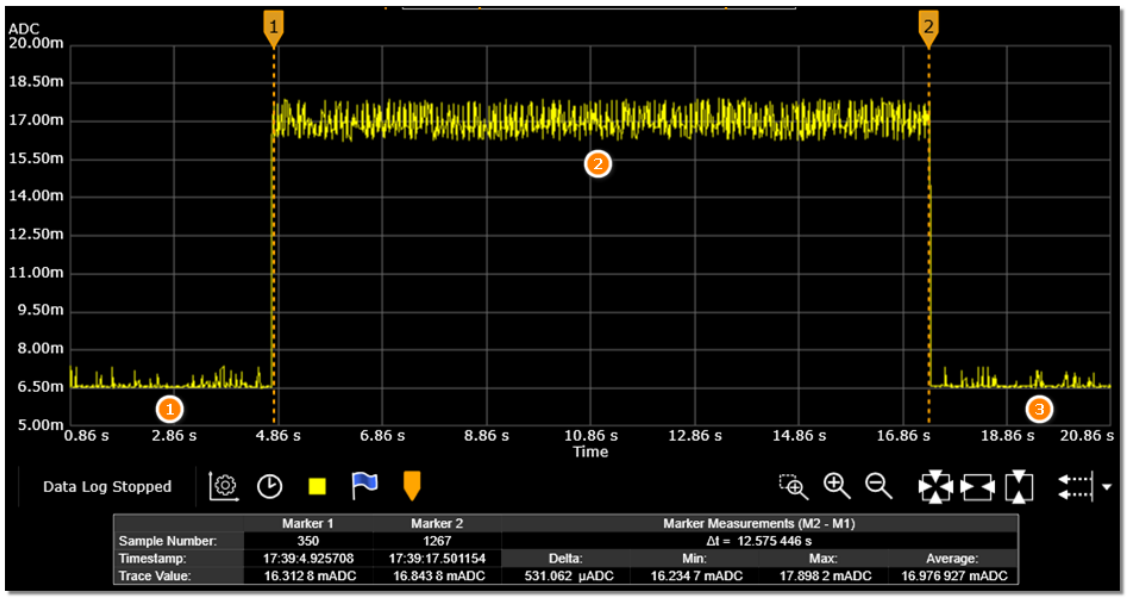
```
\0Serial:c2,Chip:1,Package:0,Rev:0
 \ | /
- SiFli Corporation
 / | \     build Jun  7 2021, v0.9.1.1-290-g6c1494aad
 2020 - 2021 Copyright by SiFli team
msh >[134] I/TOUCH main: Regist touch screen driver, probe=10038041
```

1. ◦ 发送命令 lcpu on，指示LCPU进入 wfi 模式，LCPU启动后会显示如下图所示的 log



1. ◦ 发送命令run_while_loop 240，指示HCPU在240MHz频率下执行whileloop测试，测量此时的平均电流C1，如HCPU While Loop 测试电流图所示，阶段1是HCPU跑在240MHz主频时WFI模式下的电流波形，开始执行whileloop后进入阶段2，电流上升并保持至测试结束，阶段3为回到WFI模式的电流波形。console 中显示HCPU 的频率和whileloop的持续时间，如HCPU While Loop Log 图所示

2. ◦ 发送命令run_while_loop 192，指示HCPU在192MHz下执行while loop测试，测量此时的平均电流C2，由此得到每MHz的电流增量为C=(C1-C2)/(240-192)



HCPU While Loop 测试电流图

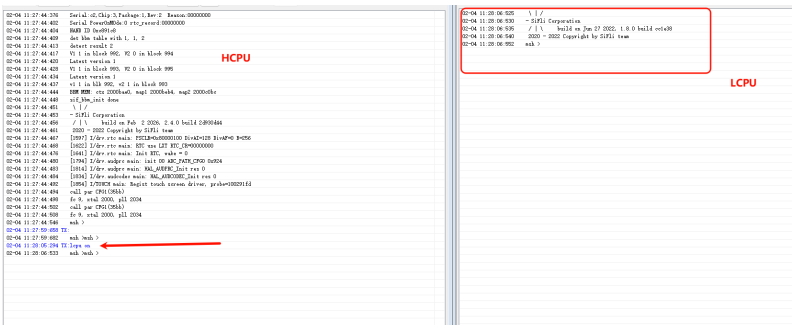
```
\OSerial:c2,Chip:1,Package:b,Rev:0
 \ | /
- SiFli Corporation
 / | \     build Jun 16 2021, v0.9.3-3-gf1419402d
 2020 - 2021 Copyright by SiFli team
msh >[783] I/TOUCH main: Regist touch screen driver, probe=1003948d

msh >run_while_loop 240
Current HCPU freq: 240000000
New HCPU freq: 240000000
Start
Done. Run for 5.38 seconds
msh >
```

HCPU While Loop Log 图

5.5.2. LPSYS

1. ◦ 打开串口调试工具，同时连接HCPU和LCPU的 console串口，连接测量设备与被测模块

2. ◦ HCPU的串口窗口发送命令 lcpu on，指示LCPU进入 wfi 模式，LCPU启动后会显示如下图所示的 log



1. ◦ 将唤醒PIN接高电平使HPSYS进人低功耗模式

2. ◦ 在LCPU的console中发送命令run_while_loop 48，指示LCPU在48MHz频率下执行while loop测试，测量此时的平均电流C1

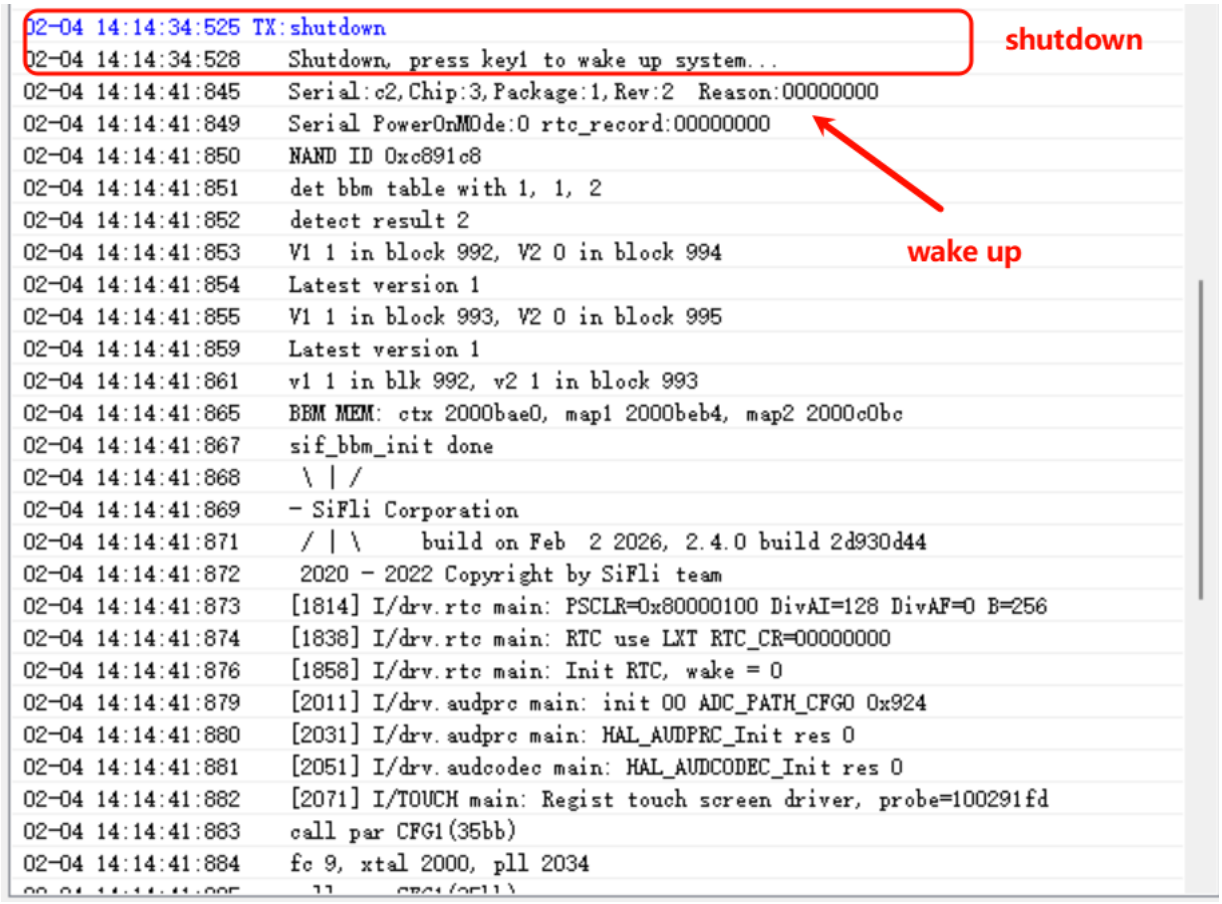
3. ◦ 发送命令run_while_loop 24，指示LCPU在24MHz下执行while loop测试，测量此时的平均电流C2，由此得到每MHz的电流增量为C=(C1-C2)/(48-24)

5.6. 关机

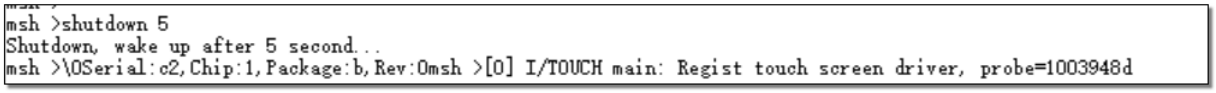
1. ◦ 打开串口调试工具，连接 HCPU 的 console 串口

2. ◦ 复位，启动成功后在 HCPU 的 console 发送命令shutdown，指示系统关机，关机后只能被唤醒引脚唤醒，由于关机电流比较小，建议使用万用表测量

3. ◦ 将唤醒引脚接高电平后再悬空唤醒系统，console 中再次出现开机阶段的 log，但不会出现版本号的 log



4. ◦ 发送命令shutdown 5，如下图指示系统关机并在 5 秒后会自动开机，为了便于测量，可以将等待时间延长



5.7. 测量结果汇总

5.7.1. 处理器功耗

由于PVDD支持3.3V或者1.8V供电，所以本次测量分别测试了PVDD在 3.3V 和 1.8V 供电情况下的数据。

5.7.1.1. CoreMark 测试结果

CPU	频率	电源电压 3.3V 电流(uA)	增量 (uA/MHz)
HCPU	240MHz	9550	30.62

CPU	频率	电源电压 3.3V 电流(uA)	增量 (uA/MHz)
	192MHz	8080	
LCPU	48MHz	936	17.41
	24MHz	518	

CPU	频率	电源电压 1.8V 电流(uA)	增量 (uA/MHz)
HCPU	240MHz	16740	56.04
	192MHz	14050	
LCPU	48MHz	1540	26.2
	24MHz	911	

5.7.1.2. WhileLoop 测试结果

CPU	频率	电源电压 3.3V 电流(uA)	增量 (uA/MHz)
HCPU	240MHz	7910	23.75
	192MHz	6770	
LCPU	48MHz	721	13.54
	24MHz	396	

CPU	频率	电源电压 1.8V 电流(uA)	增量 (uA/MHz)
HCPU	240MHz	13740	42.91
	192MHz	11680	
LCPU	48MHz	1140	17.62
	24MHz	717	

5.7.1.3. 关机功耗

测试项	电源电压 1.8V 电流(uA)
唤醒引脚唤醒(发送 shutdown)	0.30
RTC唤醒(发送 shutdown 5)	0.31