Cygnus 功耗管理概要设计

© 2022 Mooresilicon All rights reserved.

本文档版权归TCL摩星半导体所有，受相关法律法规的保护。未经书面许可不得复制传播。

修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 日期 | 作者 | 说明 |
| 0.1 | 2022-4-6 | 薛鹏宇 | 起草 |
|  |  |  |  |

目录

[1 引言 4](#_Toc100580782)

[1.1背景 4](#_Toc100580783)

[1.2 约束 4](#_Toc100580784)

[1.3术语 4](#_Toc100580785)

[1.4参考资料 4](#_Toc100580786)

[2总体设计 5](#_Toc100580787)

[2.1需求规定 5](#_Toc100580788)

[2.2架构设计 6](#_Toc100580789)

[1) 芯片电源域示意图 6](#_Toc100580790)

[2) 系统工作状态迁移图 7](#_Toc100580791)

[3) 软件整体架构 7](#_Toc100580792)

[3运行设计 9](#_Toc100580793)

[4.接口设计 10](#_Toc100580794)

[4.1应用和组件回调 10](#_Toc100580795)

[4.2 睡眠管理 10](#_Toc100580796)

[4.3 资源锁定管理 10](#_Toc100580797)

[4.4 电源管理 10](#_Toc100580798)

[4.5 频率管理 11](#_Toc100580799)

[5. 模块设计 11](#_Toc100580800)

[5.1 测试流程管理模块 11](#_Toc100580801)

[5.1.1模块逻辑框图 11](#_Toc100580802)

[5.1.2实现逻辑说明 12](#_Toc100580803)

[6.数据结构设计 12](#_Toc100580804)

[6.1数据结构一（从模块之间的接口中抽象而来） 12](#_Toc100580805)

[数据流一(*数据流是根据系统的运行控制中抽象而来的*) 12](#_Toc100580806)

[逻辑结构设计 12](#_Toc100580807)

[7. 异常处理设计 13](#_Toc100580808)

# 1 引言

## 1.1背景

Cygnus SDK平台的功耗管理设计的背景

1. 功耗管理是作为平台的组件提供给客户；
2. 功耗管理涉及的对象包括APP，组件，驱动，硬件板，芯片。其中硬件板，芯片部分通过platform下面的power驱动完成；
3. 蓝牙遥控器作为典型的对低功耗较高要求的应用，作为当前功耗管理模块设计质量重要的依据；

## 1.2 约束

1. SDK平台功耗管理的设计 面向当前的MS1008 BLE SOC芯片，需要freertos的支持。

## 1.3术语

* BLE：低功耗蓝牙

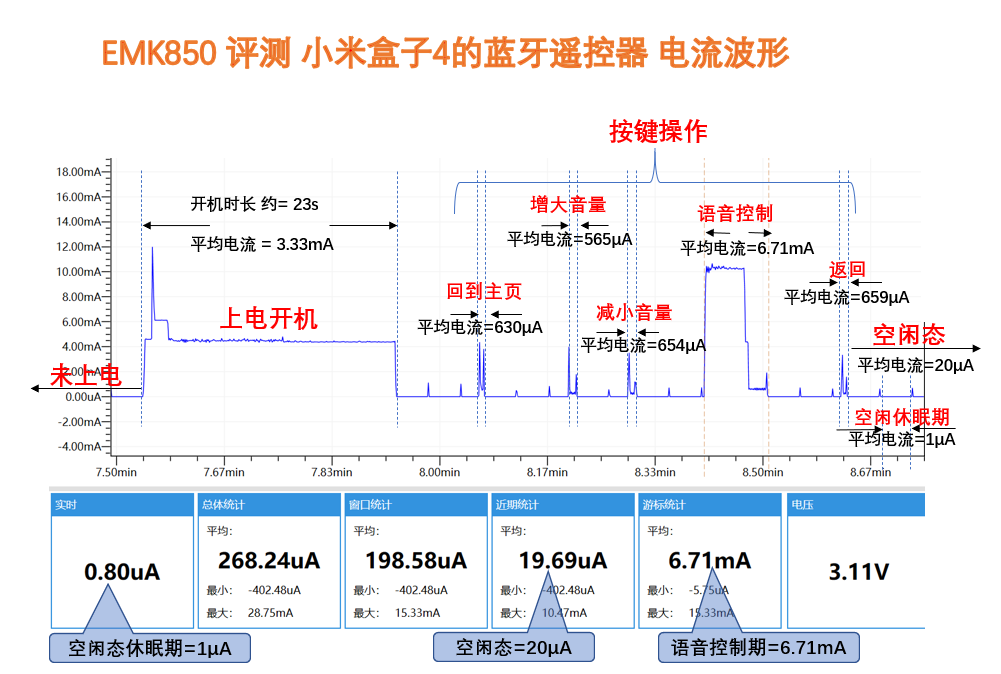
## 1.4参考资料

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 资料名称 | 版本 | 作者 |
| 1 | bt项目-低功耗详细设计 | V1.3 | 江澈 |
| 2 | 智能硬件所-TCL蓝牙遥控器技术规范V1.5-20210526A | V1.5 | 余力丛、林大煜 |

# 2总体设计

## 2.1需求规定

1. 竞品功耗分析（蓝牙遥控器 deep sleep典型应用）



数据分析：

1. 开机初始状态（开机后的广播发送，和电视连接）
2. 空闲休眠期（遥控器没有和电视发生数据连接）：电流1uA。此时CPU，flash, RAM全部掉电
3. 空闲态（遥控器处于待机状态，没有按键，连接保持）：平均电流20uA。每隔4秒左右和电视有一次数据连接，用于维持遥控器和电视之间的蓝牙连接
4. 操作状态（处理遥控器按键或者语音操作）：键盘操作，发送语音等

假设每天按键500次（6分钟），语音6分钟，每天16小时待机，除去操作之外，剩下7.8小时是电视开机时（蓝牙连接状态）遥控器的待机时间。

|  |  |  |  |
| --- | --- | --- | --- |
| 状态 | 平均电流 | 时长 | 一天总耗电0.907 mAh |
| 电视关机 待机 | 1uA | 16小时 | 0.016 mAh |
| 电视开机 待机 | 20uA | 7.8小时 | 0.156 mAh |
| 按键 | 650uA | 6分钟 | 0.065 mAh |
| 语音 | 6.7mA | 6分钟 | 0.670 mAh |

遥控器 （7号电池，电量 650 mAh），约可以用700天

考虑到目前TCL电视支持待机被蓝牙遥控器语音唤醒，所以事实上 即使电视关机，和电视的蓝牙连接也是一直存在。这样的遥控器的电池使用时间修正如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 状态 | 平均电流 | 时长 | 一天总耗电0.907 mAh |
| 电视关机 待机 | 1uA | 0小时 | 0.0 mAh |
| 电视开机 待机 | 20uA | 23.8小时 | 0.476 mAh |
| 按键 | 650uA | 6分钟 | 0.065 mAh |
| 语音 | 6.7mA | 6分钟 | 0.670 mAh |

遥控器 （7号电池，电量 650 mAh），约可以用530天

1. 总体目标。
2. 系统支持 正常工作，浅睡眠，深睡眠 三种工作模式。对于蓝牙遥控器这个特定应用场景，每次响应按键或者ble定时器操作的时候处于正常工作模式，事件处理完进入idle task， 如果此时蓝牙协议栈允许进入睡眠，则进入深睡眠模式。
3. 正常工作模式下，通过优化系统性能，降低CPU工作频率 来优化功耗。对于蓝牙遥控器这个特定应用场景，按键电流不超过 XXmAh，语音电流不超过xxmAh
4. 浅睡眠工作模式通过freertos的 tickles idle来管理。在没有事件需要处理的情况下，进入idle task 完成浅睡眠模式设置，系统时钟停止 (WFI)。
5. 深睡眠情况下，除了always on模块和 retention ram之外，其他模块全部掉电。对于蓝牙遥控器这个特定应用场景，深睡眠电流不超过2uA。深睡眠情况下，遥控器和电视的蓝牙连接保持，通过定时唤醒遥控器和电视保持连接。

## 2.2架构设计

## 1) 芯片电源域示意图

* power down （DS1）:

上图中绿色区域模块不掉电，CPU、外设(除GPIO)、RAM(SYS\_RAM、RetentionRAM)等均掉电。

* DeepSleep （DS2）:

上图中绿色和黄色区域模块不掉电，包括GPIO、RTC、RetentionRAM、UART2(低功耗串口)。 其它外设、CPU、SYS\_RAM等均掉电。

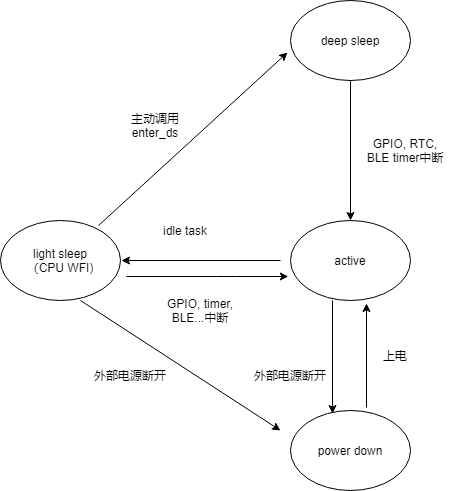
* Sleep（DS4）:

上图中绿色、黄色、深黄色等模块均不掉电， 相比DeepSleep模式支持SYS\_RAM不掉电。 其它外色、CPU等均掉电。

* Active模式:

包括上图中绿色、黄色、深黄色、紫色区域等所有模块均不掉电。

## 2) 系统工作状态迁移图



注意：

系统工作状态迁移图中的light sleep和电源域的Sleep模式（DS4）不是对应的。DS4是RAM 不掉电，CPU掉电，暂时没有使用场景。工作模式的light sleep是CPU进入WFI，不需要使用的外设掉电。

系统工作状态迁移图中的deep sleep和电源域的deep sleep模式（DS2）是对应的

## 软件整体架构

功耗管理包含下图浅红色的5个子模块：

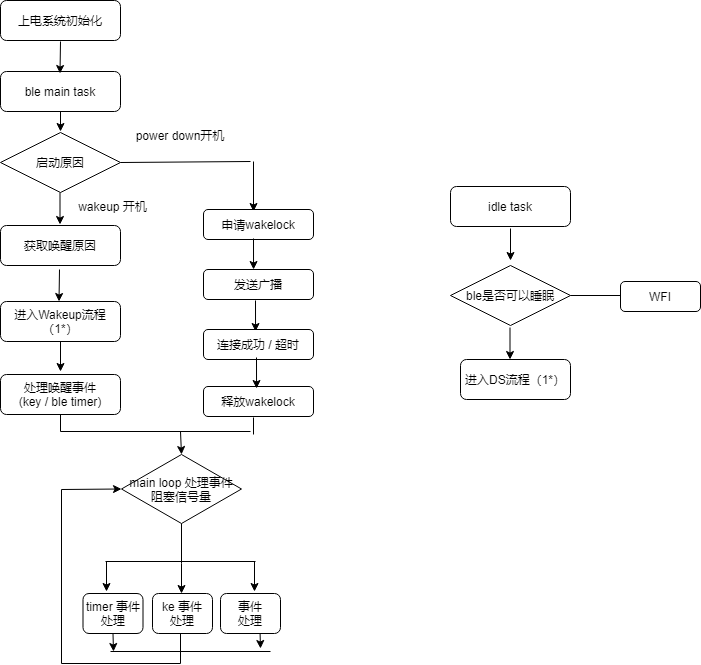


功耗管理各个子模块的功能介绍：

|  |  |  |
| --- | --- | --- |
| 序号 | 模块名 | 主要功能 |
| 1 | 回调管理模块 | 各个应用和组件如果需要在进入睡眠（浅睡眠或者深睡眠）前后需要做一些特殊的处理，需要调用该模块注册回调。在系统进入或者退出睡眠的时候，由底层驱动触发，功耗管理模块检查当前的注册情况进行处理。 |
| 2 | 睡眠管理 | 1. 浅睡眠模式管理。Idle task调用Freertos vApplicationIdleHook 进行浅睡眠处理。关闭不需要的模块，进入WFI。 2. 深睡眠管理。由应用或者idle task主动调用 enter deepsleep接口。保存必要的参数到retention ran。Flash, ram, CPU断电处理。 3. 进入睡眠前，需要通过 资源锁定管理模块 确认当前是否有应用/组件正在工作（比如 在进行语音传输 按键还没有释放）。在深睡眠中，蓝牙需要间隔唤醒和电视保持通信连接，睡眠前需要从蓝牙模块获取本次进入睡眠的时间，设置蓝牙唤醒timer。 4. 唤醒之后，需要判断当次唤醒的source 并做响应的系统恢复处理。蓝牙的恢复由蓝牙模块提供。 |
| 3 | 资源锁定管理 | 1. 应用或者组件可以通过 wakelock接口获取需要保持的资源。这些资源包括CPU最高/最低工作频率，AHP/APB 最高/最低工作频率， 保持工作状态不进入睡眠等等。 2. 查询获取当前wakelock的情况。 |
| 4 | 电源管理（电池电量管理） | 1. 获取当前电池电量。给 ble battery profile提供支撑。 2. 低电量告警。每次在深休眠唤醒后，前面5次按键每一次按完后检测电池电压，平均值低于 2.2V则为低电压。(3V以上 100% ，2V 及以下为0% ，精度为0.01V（1% ），在蓝牙工作模式下电压波动范围不超过 5%)   -- 通过 ble battery profile发 报警指令“ctrl+alt+b ”给电视，在电视端告警  -- 通过遥控器本身的指示灯告警? |
| 5 | 工作频率管理 | 设置CPU运行的频率，降低功耗 |

# 3运行设计

典型流程的运行过程：



流程说明：

1. 按键的处理流程需要单独展开。按键的处理流程中，需要判断 是否组合键（比如触发配对的组合，触发语音传输的组合）， 是否特殊键（比如电源键 只用红外发送），当前是否有蓝牙连接（有蓝牙连接则选择蓝牙发送，否在选择红外发送）。按键操作流程的设计 请参考 参考资料2。
2. 以上流程的“进入wakeup流程”和“进入DS流程”包含了应用/组件注册的回调函数处理流程。在进入睡眠前需要处理回调。进入DS/wakeup流程 请参考 参考资料1。
3. 以上流程 获取唤醒原因之后的“处理按键”和 “处理ble timer”包含了蓝牙协议栈必要的变量恢复以及蓝牙基带必要的寄存器恢复，才能保证能正常 发送蓝牙按键或者蓝牙连接。
4. 以上流程对时间的要求：

* 从唤醒到ble timer的处理完成，在100ms之内 （TBD）
* 从唤醒到按键的处理完成，在200ms之内（TBD）

# 4.接口设计

## 4.1应用和组件回调

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| typedef void (\*app\_sleep \_cb)(void) | 应用/组件回调函数定义 |
| int32\_t pm\_registerbefore\_sleepaction(app\_sleep \_cb func) | 应用/组件注册浅睡眠回调 |
| int32\_t pm\_deregisterbefore\_sleepaction(app\_sleep \_cb func) | 应用/组件取消浅睡眠回调 |
| int32\_t pm\_registerafter\_dsleepaction(app\_sleep \_cb func) | 应用/组件注册深睡眠回调 |
| int32\_t pm\_deregisterafter\_dsleepaction(app\_sleep \_cb func) | 应用/组件取消深睡眠回调 |

## 睡眠管理

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| void vApplicationIdleHook (void) | 浅睡眠处理函数，由freertos调用 |
| void pm\_sys\_dsleep\_init (void) | Power down开机之后deep sleep初始化 |
| void pm\_sys\_enter\_dsleep (void) | 进入深睡眠状态，由app调用 |
| void pm\_sys\_exit\_dsleep (void) | 退出深睡眠，启动之后判断睡眠开机之后调用 |
| int32\_t pm\_get\_pre\_mode (void) | 启动之后判断开机类型（上电还是唤醒） |
| uint32\_t pm\_sys\_check\_if\_enter\_sleep (void) | 启动之后判断唤醒类型（gpio/RTC/ble timer） |

## 资源锁定管理

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| lockhandle (void) pm\_lock\_create(int32\_t locktype) | 应用/组件 根据需要申请的资源类型，创建资源锁 |
| int32\_t pm\_lock\_acquire(lockhandle) | 应用/组件获取创建的资源锁，锁定CPU频率，不进入睡眠等 |
| int32\_t pm\_lock\_release(lockhandle) | 应用/组件释放创建的资源锁 |
| int32\_t pm\_lock\_delete(lockhandle) | 应用/组件删除创建的资源锁 |

## 电源管理

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| int32\_t pm\_get\_batterylevel (void) | 应用/组件 获取当前的电池电量(百分比) |
| int32\_t pm\_low\_battery (lockhandle) | 低电量处理（闪led灯，发送告警信息） |

## 频率管理

|  |  |
| --- | --- |
| 函数原型定义 | 说明 |
| int32\_t pm\_set\_frequency (freqconfig) | 应用/组件设置当前的CPU/AHB等频率 |
| freqconfig pm\_get\_ frequency (void) | 应用/组件获取当前的CPU/AHB等频率 |

# 5. 模块设计

## 测试流程管理模块

### 5.1.1模块逻辑框图

### 5.1.2实现逻辑说明

# 6.数据结构设计

## 6.1数据结构一（从模块之间的接口中抽象而来）

### 数据流一(*数据流是根据系统的运行控制中抽象而来的*)

（*描述如下内容：*

*1、数据处理：说明关键数据在传送时引起状态变化的数据处理情况，说明对数据进行处理的模块；*

*2、数据状态：说明对每种数据状态的数据存储形式、表现方式*）



### 逻辑结构设计

（*说明系统全局、模块接口所采用的数据结构*）

（*说明对系统数据物理存储的模式和方法*）

# 7. 异常处理设计

|  |  |
| --- | --- |
| 异常类型 | 处理方案 |
|  |  |
| 。 |  |
| 。 |  |
|  |  |
|  | 。 |