LPC1100 系列微控制器

第五章 功率配置文件

用户手册 Rev.1.00

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4

网址：http://www.zlgmcu.com

销售与服务网络（一）

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4
邮编：510630
电话：(020)38730916　38730917　38730972　38730976　38730977
传真：(020)38730925
网址：www.zlgmcu.com

广州专卖店

地址：广州市天河区新赛格电子城 203-204 室
电话：(020)87578634　87569917
传真：(020)87578842

南京周立功

地址：南京市珠江路 280 号珠江大厦 1501 室
电话：(025) 68123901　68123902
传真：(025) 68123900

北京周立功

地址：北京市海淀区知春路 113 号银网中心 A 座 1207-1208 室（中发电子市场斜对面）
电话：(010)62536178　62536179　82628073
传真：(010)82614433

重庆周立功

地址：重庆市石桥铺科园一路二号大西洋国际大厦（赛格电子市场）1611 室
电话：(023)68796438　68796439
传真：(023)68796439

杭州周立功

地址：杭州市天目山路 217 号江南电子大厦 502 室
电话：(0571)89719480　89719481　89719482
89719483　89719484　89719485
传真：(0571)89719494

成都周立功

地址：成都市一环路南二段 1 号数码科技大厦 403 室
电话：(028)85439836　85437446
传真：(028)85437896

深圳周立功

地址：深圳市深南中路 2070 号电子科技大厦 C 座 4 楼 D 室
电话：(0755)83781788（5 线）
传真：(0755)83793285

武汉周立功

地址：武汉市洪山区广埠屯珞瑜路 158 号 12128 室（华中电脑数码市场）
电话：(027)87168497　87168297　87168397
传真：(027)87163755

上海周立功

地址：上海市北京东路 668 号科技京城东座 7E 室
电话：(021)53083452　53083453　53083496
传真：(021)53083491

西安办事处

地址：西安市长安北路 54 号太平洋大厦 1201 室
电话：(029)87881296　83063000　87881295
传真：(029)87880865

销售与服务网络（二）

广州致远电子有限公司

地址：广州市天河区车陂路黄洲工业区 3 栋 2 楼
邮编：510660
传真：(020)38601859
网址：www.embedtools.com （嵌入式系统事业部）
www.embedcontrol.com （工控网络事业部）
www.ecardsys.com （楼宇自动化事业部）

技术支持：

CAN-bus：

电话：(020)22644381 22644382 22644253
邮箱：can.support@embedcontrol.com

iCAN 及数据采集：

电话：(020)28872344 22644373
邮箱：ican@embedcontrol.com

MiniARM：

电话：(020)28872684 28267813
邮箱：miniarm.support@embedtools.com

以太网：

电话：(020)22644380 22644385
邮箱：ethernet.support@embedcontrol.com

无线通讯：

电话：(020) 22644386
邮箱：wireless@embedcontrol.com

串行通讯：

电话：(020)28267800 22644385
邮箱：serial@embedcontrol.com

编程器：

电话：(020)22644371
邮箱：programmer@embedtools.com

分析仪器：

电话：(020)22644375
邮箱：tools@embedtools.com

ARM 嵌入式系统：

电话：(020) 22644383 22644384
邮箱：NXPARM@zlgmcu.com

楼宇自动化：

电话：(020)22644376 22644389 28267806
邮箱：mjs.support@ecardsys.com
mifare.support@zlgmcu.com

销售：

电话：(020)22644249 22644399 22644372 22644261 28872524
28872342 28872349 28872569 28872573 38601786

维修：

电话：(020)22644245

目　录

第 5 章 LPC111x/102/202/302 功率配置文件……1
5.1 本章导读……1
5.2 特性……1
5.3 描述……1
5.4 定义……1
5.5 时钟程序……1
5.5.1 set_pll……1
5.6 功率程序……4
5.6.1 set_power……4

第5章 LPC111x/102/202/302 功率配置文件

5.1 本章导读

功率配置文件只适用于 LPC111x/102/202/302 器件。

5.2 特性

- 包括基于 ROM 的应用服务；
- 功率管理服务；
- 时钟服务。

5.3 描述

本章描述了一些可供调用的程序，应用中可在片上 ROM 的程序里放入对这些程序的调用，以方便功率管理和时钟设置操作。

5.4 定义

在需要使用功率配置文件时，须定义以下的几个参数：

```
typedef struct _PWRD {
    void (*set_pll)(unsigned int cmd[], unsigned int resp[]);
    void (*set_power)(unsigned int cmd[], unsigned int resp[]);
} PWRD;
typedef struct _ROM {
    const PWRD * pWRD;
} ROM;
ROM ** rom = (ROM **) 0x1FFF1FF8;
unsigned int command[4], result[2];
```

5.5 时钟程序

5.5.1 set_pll

该程序根据程序调用的参数设置系统的 PLL。若简单地分频系统 PLL 的输入时钟就可获得期望的时钟，则 set_pll 程序会旁路 PLL 以降低系统功耗。

需要注意的是，在调用这个程序之前，必须先选择 PLL 时钟源（IRC/系统振荡器）（见表“系统 PLL 时钟源选择寄存器位描述”），系统 PLL 的输入时钟必须设置为主时钟（见表“主时钟选择寄存器位描述”），且系统/AHB 时钟分频器的分频值必须设为 1（见表“系统 AHB 时钟分频器位描述”）。

set_pll 尝试找一个匹配调用参数的 PLL 设置。一旦找到一个反馈分频器分频值（SYSPLLCTRL，M）、后置分频器分频值（SYSPLLCTRL，P）和系统/AHB 时钟分频器分频值（SYSAHBCLKDIV）的组合，set_pll 会用已选择的值，并将系统 PLL 时钟输出作为主时钟（如果有必要的话）。

程序返回一个结果码，该结果码指示系统 PLL 是否设置成功。若设置成功，则返回 PLL_CMD_SUCCESS；若没有设置成功，则返回码会标识出是什么运行错误。同时返回当前系统的频率值。应用中可利用这个信息来调整器件中的其它时钟（如 SSP、UART 和 WDT 时钟，以及/或时钟输出）。

表 5.1 set_pll 程序

程序	set_pll
输入	**Param0**：系统 PLL 输入频率（单位 KHz） **Param1**：期望的系统时钟（单位 KHz） **Param2**：模式（CPU_FREQ_EQU、CPU_FREQ_LTE、CPU_FREQ_GTE、CPU_FREQ_APPROX） **Param3**：系统 PLL 锁定超时
结果	**Result0**：PLL_CMD_SUCCESS|PLL_INVALID_FREQ|PLL_INVALID_MODE|PLL_FREQ_NOT_FOUND|PLL_NOT_LOCKED **Result1**：系统时钟（单位 KHz）

当调用 set_pll 功率程序时，需定义以下内容：

```
/* set_pll mode options */
#define     CPU_FREQ_EQU            0
#define     CPU_FREQ_LTE            1
#define     CPU_FREQ_GTE            2
#define     CPU_FREQ_APPROX         3
/* set_pll result0 options */
#define     PLL_CMD_SUCCESS         0
#define     PLL_INVALID_FREQ        1
#define     PLL_INVALID_MODE        2
#define     PLL_FREQ_NOT_FOUND      3
#define     PLL_NOT_LOCKED          4
```

1. 系统 PLL 输入频率和期望的系统时钟

set_pll 要找一个系统 PLL 时钟不超过 50MHz 的设置。当期望的系统时钟和系统 PLL 输入频率之比为整数时，问题很容易解决，不过若是其它情况也可以解决问题。

系统 PLL 输入时钟的频率（Param0）必须在 10000KHz～25000KHz（即 10MHz～25MHz，包括 25MHz）之间。期望的系统时钟（Param1）必须在 1KHz～50000KHz（包括 50000KHz）之间。如果这两个条件中的任一个不满足，set_pll 会返回 PLL_INVALID_FREQ，且由于 PLL 设置没有变化，因此在 Result1 中返回 Param0。

2. 模式

set_pll 的首要任务是找到一个设置，该设置能准确产生 Param1 指定频率的系统时钟。如果找不到精确的匹配，应该用输入参数模式（Param2）说明实际系统时钟是小于或等于、大于或等于或约等于期望系统时钟（Param1）指定的值。

针对 CPU_FREQ_EQU 模式的调用只有在 PLL 输出准确的 Param1 要求的频率的情况下才会成功。

若没有超过要求的频率（如总电流消耗和/或功率预算等原因），则可针对 CPU_FREQ_LTE 模式来调用程序。

CPU_FREQ_GTE 模式适合于只需最小级别 CPU 处理能力的应用。

CPU_FREQ_APPROX 产生尽可能接近期望值的系统时钟（可能大于或小于期望的值）。

若选择了非法模式，set_pll 会返回 PLL_INVALID_MODE。若期望的系统时钟超过了该程序所支持的范围，set_pll 会返回 PLL_FREQ_NOT_FOUND。在这种情况下，当前的 PLL 设置

不会改变，并且在 Result1 中返回 Param0。

3. 系统 PLL 锁定超时

若选择了有效的配置，则系统 PLL 在 100μs 内即可锁定。若 Param3 为 0，则 set_pll 会一直等待直到 PLL 锁定。若 Param3 为非零值，则该数值就是返回 PLL_NOT_LOCKED 之前，set_PLL 程序去检查 PLL 锁定成功事件的次数。在这种情况下，PLL 设置不会变化，并且在 Result1 中返回 Param0。

提示：将 Param3 的值设定为系统 PLL 频率 10000 分频的值[Hz]，可以有充分的 PLL 锁定事件轮询周期。

4. 代码示例

以下示例对上面讨论的 set_pll 的一些特性给出了说明。

（1）无效的频率（超过器件所能支持的最高时钟频率）

```
command[0] = 12000;
command[1] = 60000;
command[2] = CPU_FREQ_EQU;
command[3] = 0;
(*rom)->pWRD->set_pll(command, result);
```

以上代码指定了一个 12MHz 的 PLL 输入时钟和一个要求精确达到的 60MHz 的系统时钟，并指定要一直等 PLL 锁定。由于期望的系统时钟 60MHz 超过了最大值 50MHz。因此 set_pll 在 result[0]中返回 PLL_INVALID_FREQ，在 result[1]中返回 12000，且不会改变 PLL 设置。

（2）无效的频率选择（受系统时钟分频器的约束）

```
command[0] = 12000;
command[1] = 40;
command[2] = CPU_FREQ_LTE;
command[3] = 0;
(*rom)->pWRD->set_pll(command, result);
```

以上代码指定 PLL 输入时钟为 12MHz、系统时钟不大于 40KHz、等待 PLL 锁定时无超时。由于系统时钟的最大分频值为 255，而在 40KHz 下运行时系统需要 300 分频，因此 set_pll 在 result[0]中返回 PLL_INVALID_FREQ，在 result[1]中返回 12000，且不改变 PLL 设置。

（3）不能找到精确值（PLL）

```
command[0] = 12000;
command[1] = 25000;
command[2] = CPU_FREQ_EQU;
command[3] = 0;
(*rom)->pWRD->set_pll(command, result);
```

以上代码指定了 PLL 输入时钟须为 12MHz、系统时钟须为精确的 25MHz、并且一直等待 PLL 锁定。由于在约束中没有预先给出有效的 PLL 设置，因此 set_pll 在 result[0]中返回 PLL_FREQ_NOT_FOUND，在 result[1]中返回 12000，且不会改变 PLL 设置。

（4）系统时钟小于或等于期望值

```
command[0] = 12000;
command[1] = 25000;
command[2] = CPU_FREQ_LTE;
```

```
command[3] = 0;
(*rom)->pWRD->set_pll(command, result);
```

以上代码指定了 12MHz 的 PLL 输入时钟、不大于 25MHz 的系统时钟和一直等待 PLL 锁定。set_pll 在 result[0]中返回 PLL_CMD_SUCCESS，在 result[1]中返回 24000。新系统时钟为 24MHz。

（5）系统时钟大于或等于期望值

```
command[0] = 12000;
command[1] = 25000;
command[2] = CPU_FREQ_GTE;
command[3] = 0;
(*rom)->pWRD->set_pll(command, result);
```

以上代码指定了 12MHz 的 PLL 输入时钟、最小 25MHz 的系统时钟和一直等待 PLL 锁定。set_pll 在 result[0]中返回 PLL_CMD_SUCCESS，在 result[1]中返回 36000。新系统时钟为 36MHz。

（6）系统时钟接近于期望值

```
command[0] = 12000;
command[1] = 16500;
command[2] = CPU_FREQ_APPROX;
command[3] = 0;
(*rom)->pWRD->set_pll(command, result);
```

以上代码指定了一个 12MHz 的 PLL 输入时钟、一个接近 16.5MHz 的系统时钟和一直等待 PLL 锁定。set_pll 在 result[0]中返回 PLL_CMD_SUCCESS，在 result[1]中返回 16000。系统时钟为 16MHz。

5.6 功率程序

5.6.1 set_power

该程序根据要调用的参数配置器件的内部功率控制设置。目的是在保证芯片当前应用性能的前提下，将有效功耗减少到最合适的值。

set_power 返回结果码，该结果码反映功率控制是否更改成功。

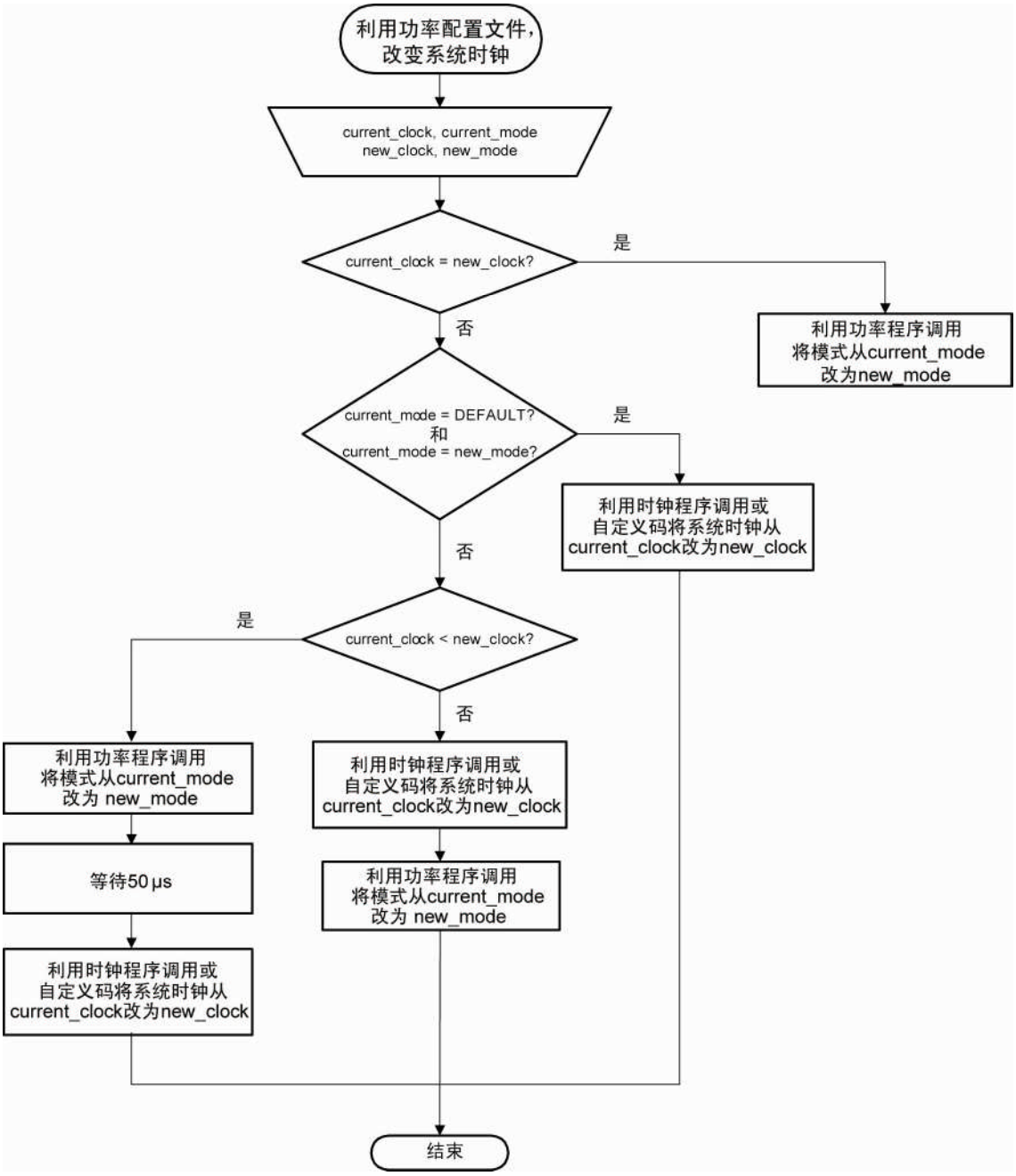

图 5.1　功率文件的用法

表 5.2　set_power 程序

程序	set_power
输入	**Param0**：新的系统时钟（单位 MHz） **Param1**：模式（PWR_DEFAULT、PWR_CPU_PERFORMANCE、PWR_EFFICIENCY、PWR_LOW_CURRENT） **Param2**：当前系统时钟（单位 MHz）
结果	**Result0**：PWR_CMD_SUCCESS|PWR_INVALID_FREQ|PWR_INVALID_MODE

调用 set_power 程序需定义以下内容：

```
/* set_power mode options */
#define     PWR_DEFAULT                 0
#define     PWR_CPU_PERFORMANCE         1
#define     PWR_EFFICIENCY              2
#define     PWR_LOW_CURRENT             3
/* set_power result0 options */
#define     PWR_CMD_SUCCESS             0
#define     PWR_INVALID_FREQ            1
#define     PWR_INVALID_MODE            2
```

1. 新的系统时钟

新的系统时钟是指：成功执行时钟程序调用或用户提供的相似代码之后，微控制器所运行的时钟频率。该操作数必须为 1MHz～50MHz（包括 50MHz）范围内的一个整数。若提供的值超出该范围，set_power 返回 PWR_INVALID_FREQ，并且不会改变功率控制系统。

2. 模式

模式参数（Param1）指定 4 个可用功率控制模式中的一个。如果没有按规定范围选择，set_power 会返回 PWR_INVALID_MODE，且不会改变功率控制系统。

PWR_DEFAULT 使器件保持为类似于复位状态的基本功率控制设置。

PWR_CPU_PERFORMANCE 配置微控制器，以便其能为应用提供更多的处理能力。CPU 性能比默认选项下的性能提高 30%。

PWR_EFFICIENCY 设置是用于在有效电流和 CPU 执行代码与处理数据能力之间找到一个平衡。在该模式下，器件可以比默认模式更好地兼顾提高 CPU 性能和降低有效电流这两个方面。

PWR_LOW_CURRENT 是着眼于降低功耗而非 CPU 性能而设计。

3. 当前系统时钟

当前系统时钟是指调用 set_power 时微控制器运行的时钟频率。该参数是 1MHz～50MHz（包括 50MHz）之间的一个整数

4. 程序示例

以下示例阐明了以上讨论的 set_power 的一些特性。

（1）无效频率（超过器件所能支持的最大时钟频率）

```
command[0] = 55;
command[1] = PWR_CPU_PERFORMANCE;
command[2] = 12;
(*rom)->pWRD->set_power(command, result);
```

上面的程序用于将在 12MHz 下运行的系统切换至 55MHz 的系统时钟，且要求有最大的 CPU 处理功率。由于规定的 55MHz 时钟高于最大值 50MHz，因此 set_power 在 result[0]中返回 PWR_INVALID_FREQ，且不会对现有的功率控制内容作任何修改。

（2）有效的功率控制

```
command[0] = 24;
command[1] = PWR_CPU_EFFICIENCY;
command[2] = 12;
(*rom)->pWRD->set_power(command, result);
```

上面代码是指将一个 12MHz 系统时钟下运行的应用切换至 24MHz 下，且指定要兼顾功耗与性能。set_power 在配置完微控制器的内部功率特性后，在 result[0]中返回 PWR_CMD_SUCCESS。