



Tina

LED 快速配置使用手册 V1.0

文档履历

版本号	日期	制/修订人	制/修订记录
V1.0	2018/4/18		初始版本

目 录

1. 概述.....	4
1.1. 编写目的.....	4
1.2. 适用范围.....	4
2. LED 子系统介绍.....	5
2.1. Menuconfig 配置 LED-Support.....	5
2.2. Trigger 介绍.....	6
3. LED 快速配置.....	7
3.1. sys_config.fex 配置.....	7
3.2. 新增加一个 trigger.....	7
3.3. 应用空间控制 led 的方法.....	9
3.3.1. 设置 trigger.....	9
3.3.2. 设置延时.....	9
4. Declaration.....	10

1. 概述

1.1. 编写目的

介绍 Tina led 相关的快速配置和使用方法。

1.2. 适用范围

该文档适用于 Tina 下各个平台.

2. LED 子系统介绍

Tina 使用 linux 标准的 led 子系统，其代码路径为：

```
tina/lichee/linux3.4/drivers/leds/  
tina/lichee/linux3.10/drivers/leds/  
tina/lichee/linux4.4/drivers/leds/
```

主要包含三部分代码：led-core.c , ledtrigger-xxx.c , led-sunxi.c

其中 led-core.c 为 led 子系统的核心文件，

ledtrigger-xxx.c 为 trigger 相关的文件，

led-sunxi.c 跟平台相关的 led 驱动文件。

2.1. Menuconfig 配置 LED-Support

在命令行中进入 Tina 根目录，执行命令进入配置主界面：

```
source build/envsetup.sh (详见①)  
lunch (详见②)  
make kernel_menuconfig (详见③)
```

详注：

- ① 加载环境变量及 tina 提供的命令
- ② 输入编号，选择方案
- ③ 进入内核配置主界面(对一个 shell 而言，前两个命令只需要执行一次)

配置路径：

```
Device Drivers  
└─> LED_Support  
    └─> LED support for SUNXI  
        └─> trigger(需要用到 trigger 的时候才需要选择)
```

操作图示：

```
Device Drivers --->  
  Multifunction device drivers --->  
  [*] Voltage and Current Regulator Support --->  
  [*] Pulse-Width Modulation (PWM) Support --->  
  <*> Multimedia support --->  
      Graphics support --->  
  <*> Sound card support --->  
  [*] HID Devices --->  
  [*] USB support --->  
  <*> MMC/SD/SDIO card support --->  
  <> Sony MemoryStick card support (EXPERIMENTAL) --->  
  [*] LED Support --->  
  <*> Switch class support --->  
  [ ] Accessibility support --->  
  [*] Real Time Clock --->  
  [*] DMA Engine support --->  
  [ ] Auxiliary Display support --->  
  <> Userspace I/O drivers --->  
      Virtio drivers --->  
      Microsoft Hyper-V guest support --->  
  [*] Staging drivers --->  
      Common Clock Framework --->  
      Hardware Spinlock drivers --->  
  [ ] IOMMU Hardware Support --->  
      Remoteproc drivers (EXPERIMENTAL) --->  
      Rpmsg drivers (EXPERIMENTAL) --->  
  [ ] Virtualization drivers --->  
  [*] Generic Dynamic Voltage and Frequency Scaling (DVFS) --->  
  <> Gator driver for DS-5
```

图 2-1 kernel_menuconfig 主界面

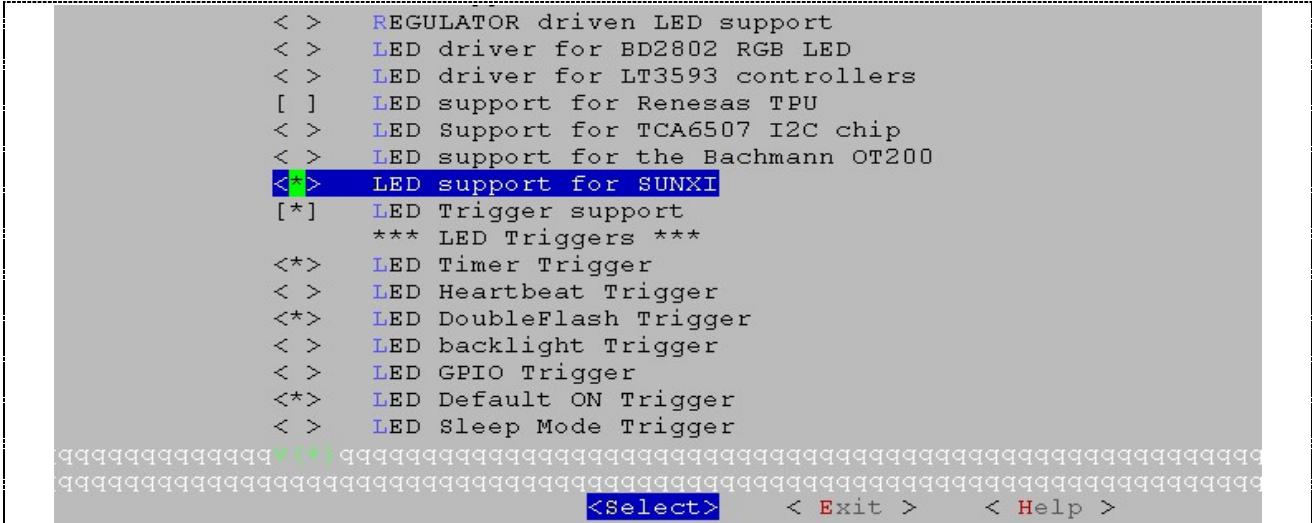


图 2-2 LED Support 界面

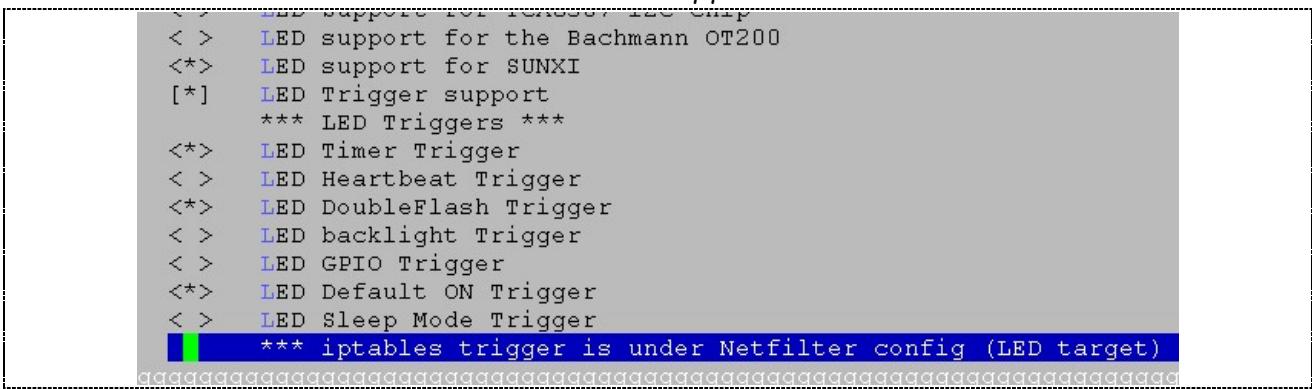


图 2-3 LED support for SUNXI 界面

2.2. Trigger 介绍

常用的 trigger 如下：Timer Trigger,Default_on Trigger, Heartbeat Trigger,IDE trigger

Timer Trigger: 闪烁触发器，使用场景最广泛。

当前内核默认的闪烁频率为 1HZ(delay_on = delay_off = 500ms)，用户可以通过修改 delay_on 和 delay_off 的值来修改当前 Led 闪烁的频率，具体修改方法见 Trigger 使用方法。

HeartBeat Trigger: 心跳灯触发器

顾名思义该触发器模拟的是心跳场景。在 linux kernel 中，提供一个可以修改的双曲线来模拟不同的心跳频率。

Default on: 使用该 trigger 的时候，LED 亮度不可以调节，只能是最大的亮度。

IDE Trigger: 硬盘指示灯，该 trigger 有透露出相关的接口给其它模块使用，当 IDE 或者存储介质有读写操作时，该指示灯会闪烁。

3. LED 快速配置

3.1. sys_config.fex 配置

1. 在 sys_config.fex 中添加 led_para 主键部分，将 led_used 修改成 1
2. Led* 修改成对应原理图上的 GPIO 管脚，功能设置为输出
示例：

```
[led_para]
compatible          = "allwinner,sunxi-leds"
led_used             = 1
led_num              = 2
led1                = port:PE17<1><default><default><default>
led1_trigger        = "none"
led2                = port:PE16<1><default><default><default>
led2_trigger        = "none"
```

ped_para 配置	配置说明
led_used	0:disable 1:enable
led_used	Led 数量
ledX	第 X 个 led
ledX_trigger	第 X 个 led 的 trigger
compatible	驱动名称，使用到 devices tree 的内核需要配置该值

如果不想预先设置 trigger 的话，可以将 led*_trigger = 设置为”none“ 如果需要预先设定 trigger 只需要将 led*_trigger 设置为需要设置的 trigger 名称即可。

例如：需要设置 led1 默认的 trigger 为 timer

```
led1_trigger = "timer"
```

编译打包，烧录固件之后进入到串口控制台

可以看到在 sysfs 中已经有生成 led* 的设备节点，如 图 3-1 操作演示 。

```
root@TinaLinux:/# cd /sys/class/leds/
root@TinaLinux:/sys/class/leds# ls
led1  led2  led3
```

图 3-1 操作演示

当正确识别到节点后，进入到其中一个 led 设备中，这里以 led1 为例，可以看到在 led1 节点下，有如下的相关属性，其中比较重要的就是 brightness、trigger、max_brightness。其中 brightness 需要硬件支持，针对这种简单的 gpio-led 来说，只有一个亮度就是 max_brightness. 所以我们在设置 brightness 的时候，设置 0~255 之间的任何整数值都能将 led 点亮。

示例：

```
echo 128 > /sys/class/leds/led1/brightness
```

3.2. 新增加一个 trigger

由于内核中预制的 trigger 有限，无法适用于所有的场景，当用户希望更深层定制自己的 trigger 的时候（如 led 的双闪），这时候就需要添加自己 trigger。添加一个 trigger 需要经过如下步骤：

1. 注册 trigger

使用 led_trigger_register() 注册一个名字为 “doubleflash“ 的 trigger，用户需要实现 activate 和 deactivate。如果当前的 trigger 很简单甚至连 deactivate 都无需实现，如 default-on trigger 就没有实现 deactivate。

```
static struct led_trigger doubleflash_led_trigger = {
    .name      = "doubleflash",
    .activate = doubleflash_trig_activate,
```

```
.deactivate = doubleflash_trig_deactivate,  
};  
  
static int __init doubleflash_trig_init(void)  
{  
    return led_trigger_register(&doubleflash_led_trigger);  
}
```

2. 实现 activate 和 deactivate

注册完成后，我们需要实现该 trigger 的 activate 和 deactivate 函数指针。

当设置某个 led 的 trigger 为”doubleflash”时，最终执行的是 doubleflash_led_trigger -> activate()
所以 activate()中实现的正式该 trigger 的主体功能 double flash(双闪)的功能。

3.3. 应用空间控制 led 的方法

3.3.1. 设置 trigger

设置 trigger 的方法主要有以下两种：

1. sys_config.fex 中预设值
2. 在应用空间根据需要设置。

应用空间设置 trigger 的方法如下，如需设置 led1 的 trigger 为 timer。

```
echo timer > /sys/class/leds/led1/trigger
```

此时 led1 会按照默认的 1HZ 的频率闪烁。

3.3.2. 设置延时

如果需要控制 led 的闪烁频率，则需要通过设置 on-off 的延时来控制。在 3.3.1 中设置 led1 trigger 为 timer，如果想更改闪烁的频率或者更改亮灭的占空比，可以通过修改 timer-trigger 提供的 delay_on 和 delay_off 这两个属性来实现，默认情况下，timer trigger 的闪烁频率为 1HZ。

如需要设置 led 灯点亮的延时为 200ms，熄灭的延时为 300ms，此时只需要进行如下设置即可。

```
echo 200 > /sys/class/leds/led1/delay_on  
echo 300 > /sys/class/leds/led1/delay_off
```

4. Declaration

This document is the original work and copyrighted property of Allwinner Technology (“Allwinner”). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgment to the copyright owner.

The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.