Melis Sunxi display2

移植使用调试说明书

1.0

2020.3.5

文档履历

版本号	日期	制/修订人	内容描述

目录

1. 概述

1.1. 编写目的

介绍 Melis 系统 sunxi display2 使用移植和调试方法。

1.2. 适用范围

Melis 系统 sunxi 平台 display2 模块。

1.3. 相关人员

系统整合人员，显示开发相关人员.

2. 相关配置

2.1. Config.mk

由于 melis 比较简单轻便不支持 dts 等复杂功能，显示驱动的配置是通过一个叫做 config.mk 的文件来配置的，后期不排除支持 kconfig，配置的作用是：

1. 选择用哪个 IC 平台。不同 IC 驱动有所不同，clk，irq，reg 地址有所不同。
2. 选择用哪个板子。不同板子硬件不一样，主要是 LCD 不一样。
3. 选择加载哪个屏驱动。不同板子屏不一样。

路径：source/ekernel/drivers/drv/source/disp2/config.mk

通过定义宏来配置，一个宏需要定义两次，一个是定义额外的编译 cflags，另外一个是 makefile 的宏。

比如下面的 config.mk 例子：

```
subdir-ccflags-y += -DCONFIG_ARCH_SUN8IW19=y -DCONFIG_LCD_SUPPORT_WTL096601G03=y \
        -DCONFIG_LCD_SUPPORT_ST7701S=y -DCONFIG_V459_PERF1=y

export CONFIG_LCD_SUPPORT_WTL096601G03 = y
export CONFIG_ARCH_SUN8IW19 = y
export CONFIG_LCD_SUPPORT_ST7701S = y
export CONFIG_V459_PERF1 = y
```

其中 CONFIG_V459_PERF1 这个宏的名字可以随意起，只要与 Makefile 前后统一即可。

其它几个宏，则是要看 disp2 中的 Makefile 和源码是否支持该 IC 是否支持该 LCD，是否有用到对应的宏，有则用，没有则需要增加，查看 2.2 和 2.3 小节就知道如何增加新平台支持。

2.2. 增加板级显示配置

路径：

Source/ekernel/drivers/drv/source/disp2/soc

按照指定规则增加一个源文件，并编译进去（Makefile 里面用 2.1 小节的宏来控制编译）：比如下面 Makefile 的 v459_perf1.c

```
1 obj-y += platform_resource.o disp_board_config.o

1 obj-$(CONFIG_ARCH_SUN8IW19) += sun8iw19.o
2 obj-$(CONFIG_V459_PERF1) += v459_perf1.o
3
```

源文件中定义 6 个全局变量，变量名字固定，不能变。

定义数据类型为 struct property_t 的三个全局数组变量，用于模拟”sys_config”

1. g_lcd0_config：第一个屏的配置变量。
2. g_lcd1_config：第二个屏的配置变量。
3. g_disp_config：显示配置变量。决定加载驱动的时候显示类型和显示分辨率等。
4. g_lcd0_config_len，g_lcd1_config_len 和 g_disp_config_len 是对应上面三个数组变量的长度，照搬即可。
5. hdmi，tv，eink 等等的配置暂时未支持。

struct property_t 数据类型，用于定义一个属性的信息：

1. 属性名字。对应其成员 name，字符串

2. 属性类型。对应其成员 type，看 enum proerty_type 的定义，共有整型，字符串，GPIO，专用 pin 和电源。

3. 属性值。根据上面的属性类型来选择 union 中成员来赋值。

举例：

整型：

```
struct property_t g_lcd0_config[] = {
        {
                .name = "lcd_used",
                .type = PROPERTY_INTGER,
                .v.value = 1,
        },
```

字符串：

```
{
        .name = "lcd_driver_name",
        .type = PROPERTY_STRING,
        .v.str = "st7701s",
},
```

电源：

```
{
        .name = "lcd_power",
        .type = PROPERTY_POWER,
        .v.power = {
                .power_name = "dldo1",
                .power_type = AXP2101_REGULATOR,
                .power_id = AXP2101_ID_DLDO1,
                .power_vol = 3300000,
        },
},
```

GPIO:

```
    {
            .name = "lcd_gpio_0",
            .type = PROPERTY_GPIO,
            .v.gpio_list = {
                    .gpio = GPIOD(9),
                    .mul_sel = GPIO_DIRECTION_OUTPUT,
                    .pull = 0,
                    .drv_level = 3,
                    .data = 1,
            },
    },
```

GPIO 之外的功能（比如 RGB，LVDS）的管脚设置与上面的区别只有 type 要设置成 PROPERTY_PIN。

至于 g_lcd0_config 有哪些可以配置的属性，属性应该取什么值则应该参考《Sunxi_display2 模块 LCD 调试文档》

如果想支持新的 LCD 屏，参考《Sunxi_display2 模块 LCD 调试文档.doc》，Melis 驱动目录结构与 Linux 一致。

2.3. 增加新 SOC 配置

路径：

Source/ekernel/drivers/drv/source/disp2/soc

按照指定规则增加一个源文件，并编译进去（Makefile 里面用 2.1 小节的宏来控制编译）：

```
obj-y += platform_resource.o disp_board_config.o

obj-$(CONFIG_ARCH_SUN8IW19) += sun8iw19.o
obj-$(CONFIG_V459_PERF1) += v459_perf1.o
```

在该源文件中定义几个全局变量，变量名字固定不能改变

1. g_irq_no。显示模块的 irq 号，顺序参考 linux 平台中 dts 配置的 disp 里面的顺序
2. g_reg_base。显示模块的基地址。顺序参考 linux 平台中 dts 配置的 disp 里面的顺序
3. g_clk_no。显示模块时钟信息。这里需要指定每个时钟的父时钟。
4. g_irq_no_len，g_reg_base_len 和 g_clk_no_len 是上面对应变量数组的长度，照搬即可。

2.4. 开发应用

Melis 的显示驱动照搬 linux 下 disp2 框架，并且 Melis 实现了类似 linux 的 ioctl 接口，所以接口不变，每个接口如何使用具体参考《sunxi display2 模块详细设计报告.doc》一文档。

使用实例：

```
#include <video/sunxi_display2.h>

rt_device_t dispfh = NULL;
unsigned long arg[6]
u32 screen_index = 0;
struct disp_layer_config config;

dispfh = rt_device_find(“disp);

arg[0] = screen_index;
arg[1] = (unsigned long)&config;
ret = dispfh->control(dispfh,  DISP_LAYER_SET_CONFIG,  (void *)arg);
```

2.5. 增加新显示设备

本次移植只将 lcd 部分移植到 melis，像 hdmi，tv 和 eink 并未移植，不过此次移植是按照 disp2 的框架

来移植所以代码是兼容的，只要做好部分接口的修改就能用了。

其它接口的驱动可以利用位于

ekernel/drivers/drv/source/disp2/disp/disp_sys_intf.h 定义的接口来对 pwm，gpio，pin，power，clk，delay, mutex，irq，”sys_config”等进行操作。

2.6. 源码位置

显示驱动：

source/ekernel/drivers/drv/source/disp2/

显示测试例子：

Source/ekernel/drivers/test/disp2

3. 调试测试方法

3.1. 测试

1. 打开使能 Source/ekernel/drivers/test/Makefile 中 disp2 文件夹的编译。
2. 编译烧写固件
3. 挂载 sd 卡，sd 卡根目录包含一个名字叫做 pic 文件夹的，里面放有显示测试锁需要的所有图片数据
4. cd 到 sd 卡所挂载目录，一般来说是/sdmmc
5. 执行测试命令。目前支持的测试命令有：

- disp_layer_alpha_test，用于测试 alpha blending
- disp_layer_scal_test，用于测试缩放
- disp_layer_format_test，用于测试显示支持格式

3.2. 查看显示信息

加载显示驱动后，进入到 melis shell 就能找到一个名字叫做 disp 的命令。

执行然后不带任何参数的话，则会打印出当前的显示信息，包括分辨率，fps 和图层信息，类似 linux 平台下的/sys/class/disp/disp/attr/sys 的打印。

```
screen 0:
de_rate 696000000 hz, ref_fps:60
mgr0: 1920x1080 fmt[yuv444] cs[0x101] range[limit] eotf[0x4] bits[8bits] err[370] force_sync[440] unblank direct_show[false]
dmabuf: cache[14] cache max[36] umap skip[7238] overflow[4708]
hdmi output mode(10) fps:60.6 1920x1080
err:8 skip:3510 irq:620319 vsync:619481 vsync_skip:1
BUF enable ch[0] lyr[0] z[0] prem[N] a[globl 255] fmt[ 1] fb[1920,1080; 960, 540; 960, 540] crop[ 0, 0,1920,1080] frame[ 5, 0,1915,1080]
addr[fb000000,fb1fa400,fb1fa400] flags[0x 0] trd[0,0]
BUF enable ch[1] lyr[0] z[1] prem[Y] a[globl 255] fmt[ 1] fb[1920,1080; 960, 540; 960, 540] crop[ 0, 0,1920,1080] frame[ 5, 0,1915,1080]
addr[f5800000,f59fa400,f59fa400] flags[0x 0] trd[0,0]
disp[0]all:5, sub:5, cur:5, free:3, skip:0
```

- screen

  显示通道

- de_rate

  de 的时钟频率

- ref_fps

  输出设备的参考刷新率

- lcd output backlight(97) fps:61.4 1280x 800

  屏输出 | 背光值（97）|屏刷新率:61.4hz|分辨率为：800x1280

- err

  de 缺数的次数

- skip

  de 跳帧的次数

- irq

  中断的次数

- Vsync

  已发送的 vsync 消息个数

- hdmi output mode(10) fps:60.6 1920x1080

  HDMI 输出 | 模式为（10：1080P@60HZ） | 刷新率为：60.6hz | 分辨率为：1920x1080

- Hdmi 参数

  mgr0: 1920x1080 fmt[yuv444] cs[0x101] range[limit] eotf[0x4] bits[8bits] err[1] force_sync[1] unblank direct_show[false]

  mgr0: DE0 | 分辨率 | tcon 输出像素格式（YUV/RGB） | 颜色空间（BT709）| 颜色范围 | 光电强度 | 色 深 | err 更新寄存器是否在安全范围 | 强制刷新次数 | mgr 是否清掉数据 | 直传

- 图层信息说明：

BUF	图层类型，BUF/COLOR，一般为 BUF，即图层是带 BUFFER 的。COLOR 意思是显示一个纯色的画面，不带 BUFFER（dim layer）。
enable	显示处于 enable 状态
ch[0]	该图层处于 blending 通道 0
lyr[0]	该图层处于当前 blending 通道中的图层 0
z[0]	图层 z 序，越小越在底部，可能会被 z 序大的图层覆盖住
prem[Y]	是否预乘格式，Y 是，N 否
a	alpha 参数， globl/pixel/; alpha 值
fmt	图层格式，值 64 以下为 RGB 格式；以上为 YUV 格式，常见的 72 为 YV12, 76 为 NV12
fb	图层 buffer 的 size，width,height，三个分量
crop	图像 buffer 中的裁减区域， [x,y,w,h]
frame	图层在屏幕上的显示区域，[x,y,w,h]

addr	三个分量的地址，如果用 iommu 方式，那就是映射的设备虚拟地址，否则是物理地址
flags	一般为 0， 3D SS 时 0x4, 3D TB 时为 0x1, 3D FP 时为 0x2
trd	是否 3D 输出，3D 输出的类型（HDMI FP 输出时为 1）
err	de 缺数的次数，de 缺数可能会出现屏幕抖动，花屏的问题。de 缺数一般为带宽不足引起
skip	表示 de 跳帧的次数，跳帧会出现卡顿问题。跳帧是指本次中断响应较慢，de 模块判断在本次中断已经接近或者超过了消隐区，将放弃本次更新图像的机会，选择继续显示原有的图像
irq	表示该通路上垂直消隐区中断执行的次数，一直增长表示该通道上的 timing controller 正在运行当 中
vsync	表示显示模块往用户空间中发送的 vsync 消息的数目，一直增长表示正在不断地发送中
fbd	Afbc 功能是否使能 Y 是 N 否

3.3. 其它调试方法

同样是 disp 命令，如果带选项和参数的话：

选项	参数	解释	举例
-c	Screen_id, color 模式	显示 colorbar。共有 8 种模式，0 到 8	disp -c 0 1 disp -c 0 8
-b	Screen_id, 背光值	调整 lcd 背光，背光值范围时 0 到 255	disp -b 0 255
-d	Screen_id, 文件路径	抓 DE 回写到文件	disp -d 0 /sdmmc/xx.bmp
-s	Screen_id， 显 示 类型，显示分辨率	切换显示（类型或者分辨率）	disp -s 0 0 0 关闭显示 disp -s 0 1 4 打开 LCD 显示

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 acknowledgement 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.tates nor implies warranty of any kind, including fitness for any particular application.