



图形开发

用户指南

文档版本 00B01

发布日期 2013-08-31

版权所有 © 深圳市海思半导体有限公司 2013。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HISILICON、海思和其他海思商标均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受海思公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，海思公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

深圳市海思半导体有限公司

地址： 深圳市龙岗区坂田华为基地华为电气生产中心 邮编：518129

网址： <http://www.hisilicon.com>

客户服务电话： +86-755-28788858

客户服务传真： +86-755-28357515

客户服务邮箱： support@hisilicon.com



前 言

概述

本文为图形开发推荐了 2 个方案，分别从方案介绍、衍生方案、开发流程、应用场景及优点和限制介绍，为用户在进行图形开发时提供参考。

产品版本

与本文档相对应的产品版本如下。

产品名称	产品版本
Hi3535 芯片	V100



读者对象

本文档（本指南）主要适用于以下工程师：




- 技术支持工程师
- 软件开发工程师

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 危险	表示有高度潜在危险，如果不能避免，会导致人员死亡或严重伤害。
 警告	表示有中度或低度潜在危险，如果不能避免，可能导致人员轻微或中等伤害。



符号	说明
 注意	表示有潜在风险，如果忽视这些文本，可能导致设备损坏、数据丢失、设备性能降低或不可预知的结果。
 窍门	表示能帮助您解决某个问题或节省您的时间。
 说明	表示是正文的附加信息，是对正文的强调和补充。

修订记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

文档版本 00B01 (2013-08-31)

第 1 次临时版本发布。



目 录

前 言.....	i
1 图形层介绍.....	1
1.1 概述.....	1
1.2 图形层体系结构.....	1
1.2.1 Hi3535 图形层体系结构.....	1
2 图形开发推荐方案	3
2.1 概述.....	3
2.2 单图层实现用户界面方案	3
2.2.1 方案介绍	3
2.2.2 衍生方案	5
2.2.3 开发流程	5
2.2.4 应用场景	6
2.2.5 优点和限制	6
2.3 多个图层实现用户界面方案	7
2.3.1 方案介绍	7
2.3.2 开发流程	10
2.3.3 应用场景	10
2.3.4 优点和限制	10



插图目录

图 2-1 单图层方案的结构示意图.....	5
图 2-2 衍生方案的结构图.....	6
图 2-3 HD0 上显示 OSD、GUI 与鼠标以及 SD0 上显示 OSD 与鼠标的结构示意图	9
图 2-4 GUI 切换至 SD0 上的结构示意图	10



表格目录

表 1-1 Hi3535 FB 设备文件、图形层以及输出设备的对应关系 1

..... 2

..... 3



1 图形层介绍

1.1 概述

海思数字媒体处理平台提供一整套机制支持图形界面的开发，主要包括：

- 图形二维加速引擎（Two Dimensional Engine，简称 TDE），它利用硬件加速对图形图像进行处理。
- Hisilicon Framebuffer（以下简称 HiFB）用于管理叠加图形层，它不仅提供 Linux Framebuffer 的基本功能，还在 Linux Framebuffer 的基础上增加层间 colorkey、层间 Alpha 等扩展功能。



- TDE 相关使用方法请参见《TDE API 参考》
- HiFB 相关使用方法请参见《HiFB 开发指南》和《HiFB API 参考》

1.2 图形层体系结构

Hi3535 支持两路高清(HD0~HD1)、1 路标清（SD0）共 3 个独立的显示设备，同时支持 4 个图形层 G0~G3(其中 G3 为鼠标层)。



每个输出设备支持的接口类型和时序请参见《Hi3535H.264 编解码处理器用户指南》的“11.2 VDP 章节”。

各个图形层与各设备有一定的约束关系，如表 1-1 所示。

表1-1 Hi3531 FB 设备文件、图形层以及输出设备的对应关系

FB 设备文件	图形层	对应显示设备
/dev/fb0	G0	G0 只能在 HD0 设备上显示。
/dev/fb1	G1	G1 只能在 HD1 设备上显示。
/dev/fb2	G2	G2 只能在 SD0 设备上显示。



FB 设备文件	图形层	对应显示设备
/dev/fb3	G3	G3 为鼠标层，它们总是处在显示设备叠加层的最高层。如 HD0 上有视频层、G0，则叠加顺序从下到上依次为：视频层，G0，G3。G3 可作为硬件鼠标层，也可作为软件鼠标层，由模块加载参数 <code>softcursor</code> 来决定。作为硬件鼠标层，它们的操作方法与其他图形层一样；而作为软件鼠标层，则应使用 HiFB 中实现的软件鼠标专用接口进行操作。

注：为了显示图形层，使用 Hi3535 芯片的用户必须先配置并启动输出设备（通过 VOU 模块的接口），最后通过 HiFB 模块接口操作图像层使之显示。



2 图形开发推荐方案

2.1 概述

在监控领域中，一般输出设备的图形用户界面内容包括：

- 后端 OSD：显示画面分割线、通道号、时间等信息，用以界定多画面显示布局。
- GUI 界面：包括各种菜单、进度条等元素，用户通过操作 GUI 界面进行设备配置。
- 鼠标：提供更方便易用的界面菜单操作方式。

以上 3 类图形内容可以通过 1 个图形层实现，也可以通过多个图形层实现。Hi35xx 芯片提供多个图形层，指导用户正确、合理、有效地利用这些图形层，以满足不同的输出界面应用场景。下面推荐几种方案供参考。

2.2 单图层实现用户界面方案

2.2.1 方案介绍

该方案总体思路是：每个设备都只使用 1 层图形层来完成本设备的后端 OSD、GUI 和鼠标的显示，鼠标也可以使用独立的鼠标层实现。

可具体描述为：每个输出设备使用一个图形层来完成本设备的后端 OSD、GUI；GUI 画在独立的缓存上，后端 OSD 直接画在 FB 显存中，再通过 TDE 进行 alpha 混合；鼠标可以使用单独的鼠标图形层，也可以跟 OSD、GUI 共用一个图层，共用图层的时候，可以画在 GUI 缓存上。

该方案使用了以下机制：

- 每个设备的后端 OSD 直接绘制在各自的 FB 显存中。
例如在每个图形层对应的 FB 显存中绘制分割布局、通道号或者时间。
- 每个设备一块 GUI 画布，GUI 变更时局部刷新。
每个设备使用一块独立的缓存绘制 GUI（称该块缓存为 GUI 画布），当 GUI 变更时仅需要进行局部刷新。
- GUI 画布整体搬移至相应图层的 FB 显存中

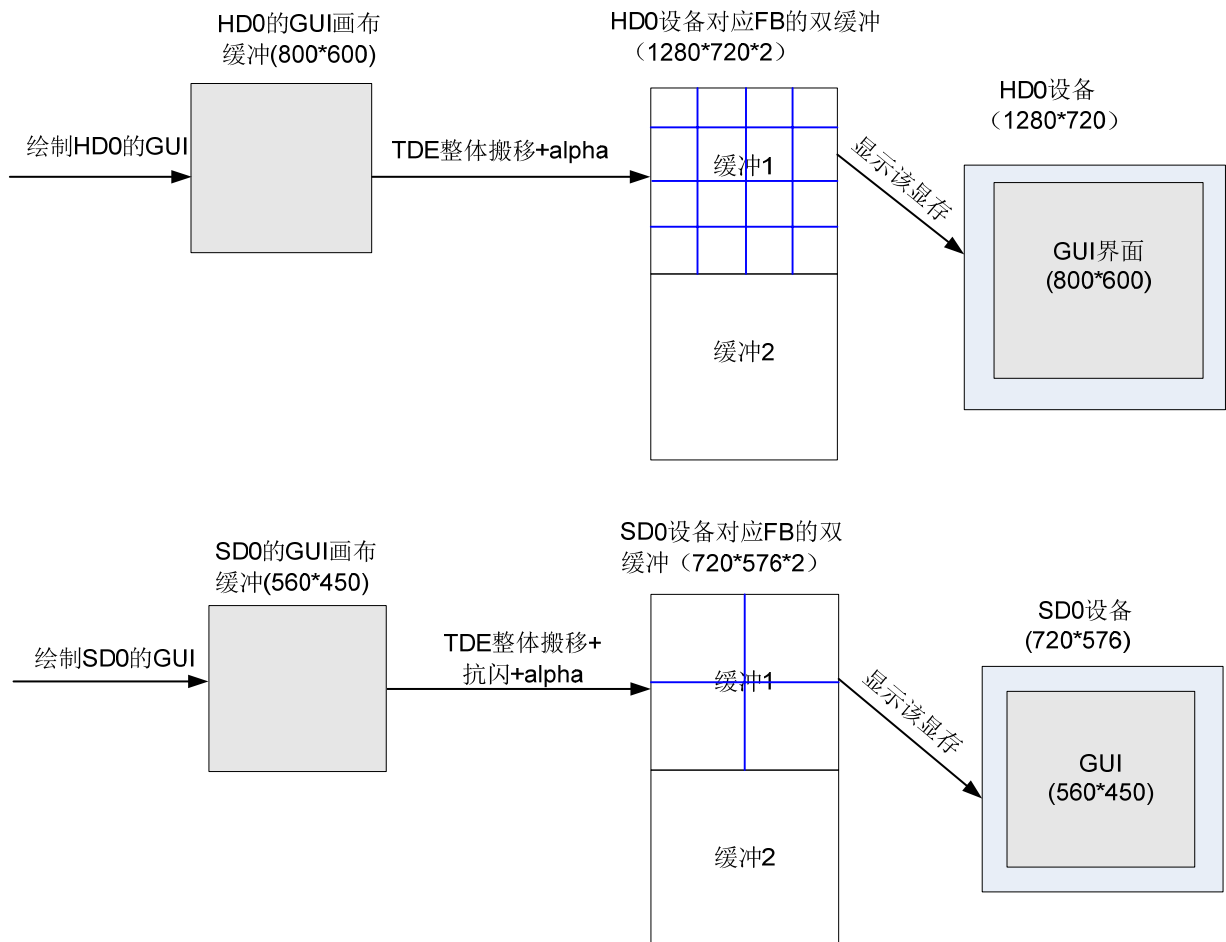
将绘制好的画布整体搬移到相应的 FB 缓冲中，在此过程中可利用 TDE 实现 GUI 和 OSD 的叠加透明效果。每次 GUI 或 OSD 有变动时，由于是对画布和 OSD 整体做叠加，故不需要针对局部信息计算 GUI 和 OSD 的叠加区域。

- FB 双缓冲

为防止一块 FB 缓冲被边绘制边显示而导致绘制过程可见，推荐使用 FB 双缓冲机制或是 HiFB 实现的扩展模式中的 HIFB_LAYER_BUF_DOUBLE / HIFB_LAYER_BUF_DOUBLE_IMMEDIATE 机制。它们的原理都是为 FB 分配 2 块大小相同的缓冲作为显存交替绘制和显示。如 VO 正在显示缓冲 2，则本次绘制的对象为缓冲 1，然后对于 FB 标准模式可通过 FB 的 PAN_DISPLAY 或 FBIOFLIP_SURFACE 调用通知 VO 显示缓冲 1，而对于 FB 扩展模式可通过 FB 的 FBIO_REFRESH 调用通知 VO 显示缓冲 1。

方案的结构如图 2-1 所示。

图2-1 单图层方案的结构示意图



该方案在后端 OSD 或者 GUI 界面变动时，都需要重新绘制 FB 缓存：

- 本设备的后端 OSD 改变时，如 16 通道分割线切换到 9 通道分割线：先清空 FB 缓存，再绘制新的 OSD，再将 GUI 界面整体搬移到 FB 缓存中。



- GUI 界面每次变动时，都需要先清空 FB 缓存，再绘制 OSD，然后将新的 GUI 界面整体搬移到 FB 缓存中。

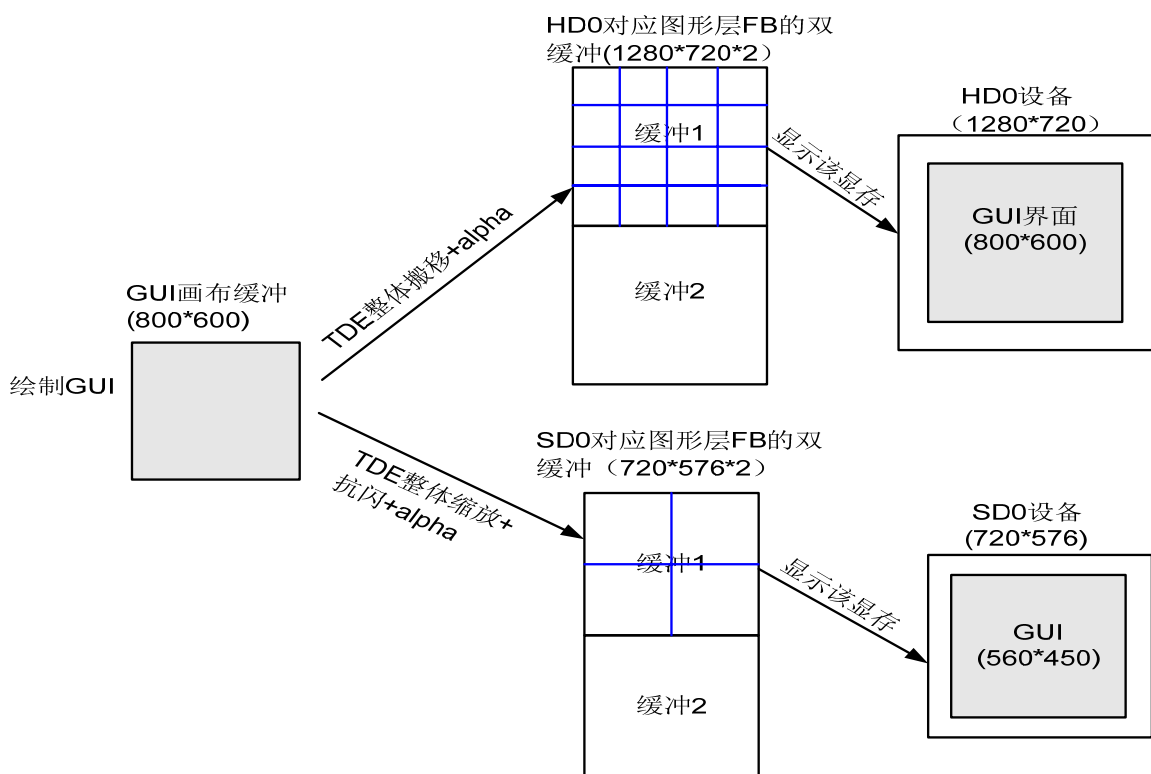
2.2.2 衍生方案

当 SD0 和 HD0 设备上想同时显示同样的 GUI 界面时，该方案可简化仅有一块 GUI 画布缓存：

- 画布大小与 HD0 的 GUI 层大小相同（800*600），用户可按照 HD0 的 GUI 规格（如 800*600）准备一套图片，每次 GUI 变更时仅局部绘制画布，而 SD0 的 GUI 则是将画布整体经过缩放、抗闪得到，其效果略差于 HD 上的 GUI。
- 每次更新画布后，对于 HD 设备，由于画布大小与 GUI 界面大小相同，故利用 TDE 做整体搬移操作即可；对于 SD0 设备，需要利用 TDE 对画布整体进行缩放至和 SD0 绑定的图形层对应的 FB 显存中，同时进行抗闪烁处理（因 SD0 是隔行设备）。

该衍生方案的结构如图 2-2 所示。

图2-2 衍生方案的结构图



2.2.3 开发流程

方案 1 的开发流程

以 HD0 和 SD0 设备上的 GUI 和 OSD 为例：HD0 设备上 16 画面等分分割线，SD0 设备上 4 画面等分分割线，且 HD0 和 SD0 同时显示同样的 GUI。



若此时 GUI 界面有变化，则该方案的实现过程为：

1. 清空 HD0 和 SD0 对应图形层的 FB 的空闲缓冲（假设为缓冲 1，缓冲 2 正在被 VO 显示）。
2. 在 HD0 对应图形层的 FB 缓冲 1 中绘制 16 通道分割线。
3. 在 SD0 对应图形层的 FB 缓冲 1 中绘制 4 通道分割线。
4. 局部更新画布。
5. 用 TDE 将画布整体搬移到 HD0 对应图形层的 FB 缓冲 1 的合适位置，此过程可以做 alpha 透明度叠加以实现 GUI 半透明效果。
6. 用 TDE 将画布整体缩放到 SD0 对应图形层的 FB 缓冲 1 的合适位置，此过程可以做抗闪、alpha 透明度叠加（以实现 GUI 半透明效果）。
7. 通过 FB 接口调用 PAN_DISPLAY 通知 HD0 显示和本设备绑定图形层已准备好的 FB 的缓冲 1。
8. 通过 FB 接口调用 PAN_DISPLAY 通知 SD0 显示本设备绑定图形层已准备好的 FB 的缓冲 1。

----结束

2.2.4 应用场景

应用场景如下：

- 每个设备上有各自的后端 OSD（如 HD0 为 16 画面分割布局，HD1 为 8 画面分割布局，SD0 为 4 画面分割布局）。
- 2 或多个输出设备上同时有 GUI 界面（相同或者不同）。

2.2.5 优点和限制

该方案具有以下优点：

- 可同时在多个设备上显示 GUI 界面。
- GUI 画布可局部刷新，节省总线带宽和 TDE 性能。
- 可实现 GUI 和 OSD 的叠加透明效果，且用户控制流程简单。每次 GUI 或 OSD 有变动时，由于是对画布和 OSD 整体做叠加，故不需要针对局部信息计算 GUI 和 OSD 的叠加区域。
- 对于衍生方案，用户仅需要一套 GUI 界面的图片，就可适应不同分辨率设备的 GUI 需求，节省 Flash 空间。

该方案具有以下约束：

- 对于衍生方案：标清设备上的 GUI 是画布缩放得到的，故效果略差于高清设备上的 GUI。



2.3 多个图层实现用户界面方案

2.3.1 方案介绍

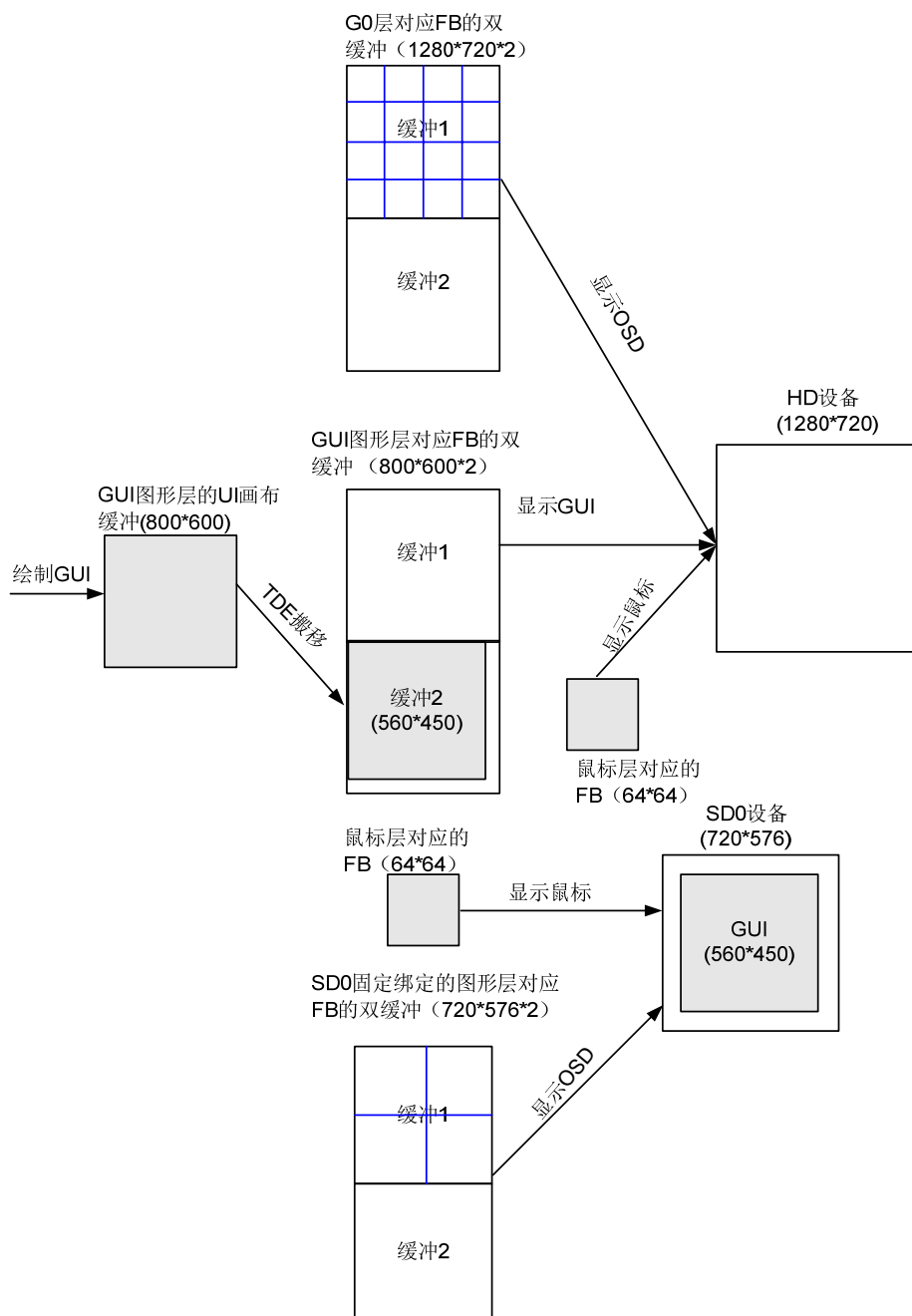
若芯片支持独立的图形层专门用来实现 GUI 界面，则称该层为 GUI 图形层。本方案利用 3 个图形层分别实现 OSD、GUI 界面、鼠标。

每个设备都只使用 1 层图形层来实现本设备的 OSD，此图形层即为和设备固定绑定的图形层。GUI 使用 GUI 图形层实现。当 GUI 界面移到某个设备上时，如移到 HD0 设备上，HD0 同时绑定两个图像层（设备固定绑定的图形层和 GUI 层），且利用图形层间 alpha 叠加可实现 GUI 半透明。

当 GUI 界面从 HD0 移到 HD1 上时，先关闭 GUI 图形层，解绑定 GUI 图形层与 HD0，再通过设置绑定关系将 GUI 图形层绑定到 HD1 上，最后通过 HiFB 接口开启 GUI 层即可。在动态绑定的过程中，不需要禁止 VO 设备。鼠标利用鼠标层实现，鼠标层大小为鼠标图片大小。鼠标移动时，通过设置鼠标层对应的 FB 的显示位置实现。鼠标可以动态的在支持绑定多图形层的显示设备间移动，也需要用户先关闭鼠标层，解绑定之前的绑定关系，再通过设置绑定关系将鼠标层绑定到新的设备上，最后开启鼠标层。



图2-3 HD0 上显示 OSD、GUI 与鼠标以及 SD0 上显示 OSD 与鼠标的结构示意图

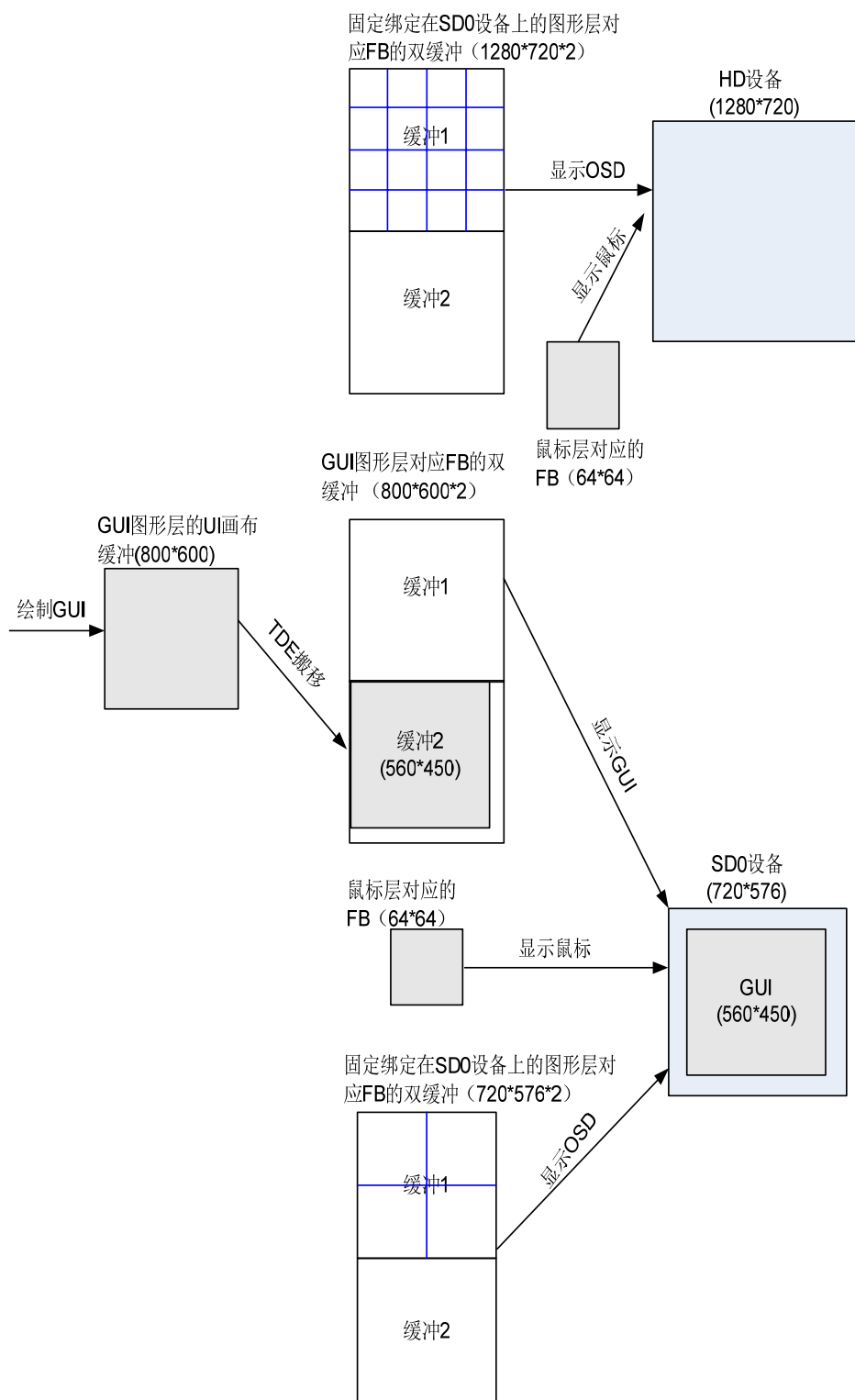


由于切换 GUI 图层以及鼠标层时不需要关闭显示设备，所以不会导致短暂黑屏现象。在此推荐此方案以实现 GUI 的动态切换。

GUI 和鼠标切换至 SD0 上的结构如图 2-4 所示。



图2-4 GUI 切换至 SD0 上的结构示意图





2.3.2 开发流程

以 HD 和 SD 设备上的 GUI 和 OSD 为例：HD0 设备上 16 画面等分分割线，SD0 设备上 4 画面等分分割线，GUI 在 HD0 和 SD0 间切换。

方案 2 的开发流程

以 GUI 界面从 HD0 切换到 SD0 上为例，配置步骤如下：

1. 局部更新 GUI 图形层对应的画布。
2. 关闭 GUI 图形层。
3. 设置 GUI 图形层绑定到 SD 设备。
4. 通过 HiFB 接口设置 GUI 图形层属性并开启 GUI 图形层：用 TDE 将画布整体缩放到 GUI 图形层缓冲 2 的合适位置。注意：由于 SD0 上的 GUI 界面大小小于画布大小，故需要将画布缩放到和 SD0 的 GUI 大小一样，即改变 GUI 图形层对应的 FB 的显示分辨率 560 x 450，以适应 SD0 上的 GUI 界面大小）。

----结束

2.3.3 应用场景

应用场景如下：

- 每个输出设备有各自的后端 OSD（如 HD0 为 16 画面分割布局，HD1 为 8 画面分割布局，SD0 为 4 画面分割布局）。
- 某一时刻仅有一个设备显示 GUI，且要求 GUI 能在支持多图形层的显示设备上切换（动态或静态地切换）。
- 若 HiFB 最多支持 n 个鼠标层，则某一时刻有 n 个设备上显示鼠标，且要求鼠标能在支持多图形层的显示设备上动态切换。

2.3.4 优点和限制

该方案具有以下优点：

- GUI 界面和后端 OSD 使用不同的图层，其中一个变化时不需要更新另一个图层，用户控制简单。
- 鼠标可以在支持多图形层的显示设备间动态切换，切换时不需要关闭输出设备。
- 支持 GUI 在多个显示设备间动态切换，用户感受增强。
- 用户仅需要一套 GUI 界面的图片，可适应不同分辨率设备的 GUI 需求。
- 一块 GUI 画布，可局部刷新。

该方案具有以下限制：

由于 GUI 和 OSD 使用了 2 个图层，相对于方案 1 占用了更多的系统带宽。