MI FB API

Version 2.03

REVISION HISTORY

Revision No.	Description	Date
2.03	• Initial release	04/12/2018

TABLE OF CONTENTS

1. API 参考

1.1. 概述

MStarFB 的 API 分为以下几类：

文件操作类：

提供操作 MStarFB 的接口。通过调用这些接口，可以像操作文件一样操作叠加层。这些接口是 Linux 本身提供的标准接口，主要有 open、close、write、read、lseek 等。本文档不对这些标准接口进行描述。

显存映射类：

提供将物理显存映射到用户虚拟内存空间的接口。这些接口是 Linux 本身提供的标准接口，主要有 mmap、munmap 等。本文档不对这些标准接口进行描述。

显存控制和状态查询类：

允许设置像素格式和颜色深度等属性的接口。这些接口是 Linux 本身提供的标准接口，经常使用。本文档将对其进行简要描述。

层间效果控制和状态查询类：

MStarFB 可以管理多个图形叠加层，每层可以设置 Alpha 信息和显示区域等属性。相对于 Linux Framebuffer，这些是 MStarFB 的新增功能。本文档将重点描述该部分。该部分功能是通过扩展标准 linux Framebuffer ioctl 函数的命令参数实现的。

ioctl 函数：

MStarFB 的用户态接口以 ioctl 形式体现，其形式如下：

```
int ioctl(int fd,
          unsigned long cmd,
          ……
          );
```

该函数是 Linux 标准接口，具备可变参数特性。但在 MStarFB 中，实际只需要 3 个参数。因此，其语法形式等同于：

```
int ioctl (int fd,
           unsigned long cmd,
           CMD_DATA_TYPE *cmddata);
```

其中，CMD_DATA_TYPE 随参数 cmd 的变化而变化。这 3 个参数的详细描述如表 1-1 所示。

表 1-1 ioctl 函数的 3 个参数

参数名称	描述	输入/输出
Fd	Framebuffer 设备文件描述符，是调用 open 函数打开 Framebuffer 设备之后的返回值。	输入
Cmd	主要的 cmd（命令控制字）如下： • FBIOGET_VSCREENINFO：获取屏幕可变信息 • FBIOPUT_VSCREENINFO：设置屏幕可变信息 • FBIOGET_FSCREENINFO：获取屏幕固定信息 • FBIOPAN_DISPLAY：设置 PAN 显示 • FBIOGET_SCREEN_LOCATION：获取叠加层显示区域 • FBIOSET_SCREEN_LOCATION：设置叠加层显示区域 • FBIOGET_SHOW：获取叠加层显示状态 • FBIOSET_SHOW：设置叠加层显示状态 • FBIOGET_GLOBAL_ALPHA：获取叠加层 Alpha属性 • FBIOSET_GLOBAL_ALPHA：设置叠加层 Alpha属性 • FBIOGET_COLORKEY：获取叠加层的 Colorkey 属性 • FBIOSET_COLORKEY：设置叠加层的 Colorkey 属性 • FBIOGET_DISPLAYLAYER_ATTRIBUTES:获取叠加层属性 • FBIOSET_DISPLAYLAYER_ATTRIBUTES:设置叠加层属性 • FBIOGET_CURSOR_ATTRIBUTE:获取鼠标层属性 • FBIOSET_CURSOR_ATTRIBUTE:设置鼠标层属性	输入
cmddata	各 cmd 对应的数据类型分别是： • 获取或设置屏幕可变信息：`struct fb_var_screeninfo` *类型 • 获取屏幕固定信息：struct fb_fix_screeninfo *类型 • 设置 PAN 显示：`struct fb_var_screeninfo` *类型 • 获取或设置屏幕叠加层坐标原点：MI_FB_CursorAttr_t *类型 • 获取或设置叠加层显示状态：MI_BOOL *类型 • 获取或设置叠加层 Alpha：MI_FB_GlobalAlpha_t *类型 • 获取或设置叠加层Colorkey: MI_FB_ColorKey_t*类型 • 获取或设置叠加层属性:MI_FB_DisplayLayerAttr_t*类型 • 获取或设置鼠标图层属性: MI_FB_CursorAttr_t*	输入输出

2. 标准功能

2.1. FBIOGET_VSCREENINFO

- 功能

  获取屏幕的可变信息。

- 语法

```
int ioctl (int fd,FBIOGET_VSCREENINFO,struct fb_var_screeninfo *var);
```

- 描述

  使用此接口获取屏幕的可变信息，主要包括分辨率和像素格式。信息的详细描述请参见“3.1 struct fb_var_screeninfo ”。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOGET_VSCREENINFO	ioctl 号	输入
var	可变信息结构体指针	输出

- 返回值

  返回值：
  - 0 成功。
  - -1 失败。

- 依赖

  头文件：fb.h

- 相关接口

  FBIOPUT_VSCREENINFO

2.2. FBIOPUT_VSCREENINFO

- 功能

  设置 Framebuffer 的屏幕分辨率和像素格式等。

- 语法

```
int ioctl (int fd,FBIOPUT_VSCREENINFO, struct fb_var_screeninfo *var);
```

- 描述

  使用此接口设置屏幕分辨率、像素格式。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOPUT_VSCREENINFO	ioctl 号	输入
var	可变信息结构体指针	输入

- 返回值

  返回值：0 成功。
  返回值：-1 失败

- 依赖

  头文件：fb.h

※ 注意

  必须保证实际分辨率与偏移的和在虚拟分辨率范围内，否则系统会自动调整实际分辨率的大小让其在虚拟分辨率范围内。

- 相关接口

  FBIOGET_VSCREENINFO

2.3. FBIOGET_FSCREENINFO

- 功能

  获取 Framebuffer 的固定信息。

- 语法

```
int ioctl (int fd,FBIOGET_FSCREENINFO, struct fb_fix_screeninfo  *fix);
```

- 描述

  使用此接口获取 Framebuffer 固定信息，包括显存起始物理地址、显存大小和行间距等。信息的详细描述请参见“3.1 struct fb_fix_screeninfo ”。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOGET_FSCREENINFO	ioctl 号	输入
fix	固定信息结构体指针	输出

- 返回值

返回值 { 0 成功。 / -1 失败

- 依赖

头文件：fb.h

※ 注意

无。

- 相关接口

无。

2.4. FBIOPAN_DISPLAY

- 功能

设置从虚拟分辨率中的不同偏移处开始显示。

- 语法

```
int ioctl (int fd,FBIOPAN_DISPLAY, struct fb_var_screeninfo  *var);
```

- 描述

使用此接口设置从虚拟分辨率中的不同偏移处开始显示，实际的分辨率不变。如图2-1 所示：（xres_virtual, yres_virtual）是虚拟分辨率，（xres, yres）是实际显示的分辨率， (xoffset, yoffset) 是显示的偏移。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOPAN_DISPLAY	ioctl 号	输入
var	可变信息结构体指针	输入

- 返回值

返回值 { 0 成功。 / -1 失败

- 依赖

头文件：fb.h

※ 注意

必须保证实际分辨率与偏移的和在虚拟分辨率范围内，否则设置不成功。

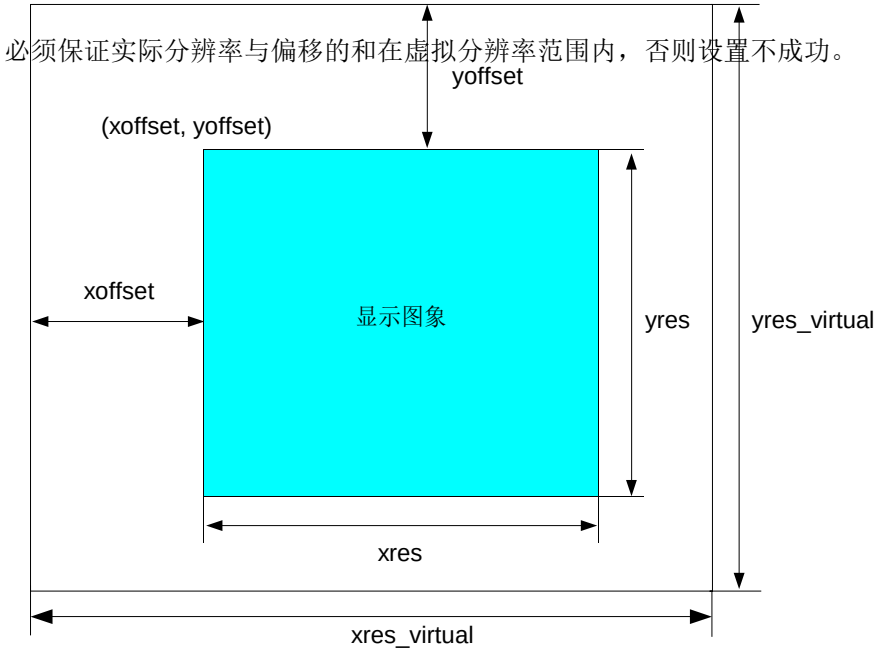

图 2-1 设置从虚拟分辨率中的不同偏移处开始显示

3. 扩展功能

3.1. FBIOGET_SCREEN_LOCATION

- 功能

  获取叠加层在屏幕上的显示区域

- 语法

```
int ioctl (int fd,FBIOGET_SCREEN_LOCATION,MI_FB_Rectangle_t*
pstRectangle);
```

- 描述

  使用此接口获取叠加层在屏幕上的显示区域

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOGET_SCREEN_LOCATION	ioctl 号	输入
pstRectangle	MI_FB_Rectangle_t结构体指针，包含(x,y)坐标和width,height。表示叠加层在屏幕上的显示区域	输出

- 返回值

  返回值：
  - 0 成功。
  - -1 失败

- 依赖

  头文件：mstarFb.h

- 相关接口

  FBIOSET_SCREEN_LOCATION

3.2. FBIOSET_SCREEN_LOCATION

- 功能

  设置叠加层在屏幕上的显示区域

- 语法

```
int ioctl (int fd,FBIOSET_SCREEN_LOCATION,MI_FB_Rectangle_t*
pstRectangle);
```

- 描述

  使用此接口设置叠加层在屏幕上的显示区域

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOSET_SCREEN_LOCATION	ioctl 号	输入
pstRectangle	MI_FB_Rectangle_t结构体指针，包含(x,y)坐标和width,height。表示叠加层在屏幕上的显示区域	输入

- 返回值

返回值
- 0 成功。
- -1 失败

- 依赖

  头文件：mstarFb.h

- 举例

  无。

※ 注意

  无

- 相关接口

  FBIOGET_SCREEN_LOCATION

3.3. FBIOGET_SHOW

- 功能

  获取当前叠加层的显示状态。

- 语法

```
int ioctl (int fd,FBIOGET_SHOW,MI_BOOL *bShow);
```

- 描述

  使用此接口获取当前叠加层显示状态。

- 形参

参数名称	描述	输入/输出
Fd	Framebuffer 设备文件描述符	输入
FBIOGET_SHOW	ioctl 号	输入
bShow	指示当前叠加层的状态： • *bShow = MS_TRUE：当前叠加层处于显示状态 • *bShow = MS_FALSE：当前叠加层处于隐藏状态	输出

- 返回值

  返回值：
  - 0 成功。
  - -1 失败

- 依赖

  头文件：mstarFb.h

- 相关接口

  FBIOSET_SHOW

3.4. FBIOSET_SHOW

- 功能

  显示或隐藏该叠加层。

- 语法

```
int ioctl (int fd,FBIOSET_SHOW,MI_BOOL *bShow);
```

- 描述

  使用此接口设置叠加层显示状态：显示或隐藏。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOSET_SHOW	ioctl 号	输入
bShow	该叠加层的显示状态： • *bShow = MS_TRUE：显示当前叠加层 • *bShow = MS_FALSE：隐藏当前叠加层	输入

- 返回值

  返回值 { 0 成功。 / -1 失败

- 依赖

  头文件：mstarFb.h

- 相关接口

  FBIOGET_SHOW

3.5. FBIOGET_GLOBAL_ALPHA

- 功能

  获取叠加层 Alpha 设置。

- 语法

```
int ioctl (int fd,FBIOGET_GLOBAL_ALPHA,MI_FB_GlobalAlpha_t *pstAlpha);
```

- 描述

  使用此接口获取当前叠加层的 Alpha 设置。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOGET_GLOBAL_ALPHA	ioctl 号	输入
pstAlpha	MI_FB_GlobalAlpha_t 结构体指针	输出

- 返回值

返回值
- 0 成功。
- -1 失败

- 依赖

头文件：mstarFb.h

- 相关接口

FBIOSET_GLOBAL_ALPHA

3.6. FBIOSET_GLOBAL_ALPHA

- 功能

设置叠加层的 Alpha。

- 语法

```
int ioctl (int fd,FBIOSET_GLOBAL_ALPHA, MI_FB_GlobalAlpha_t *pstAlpha);
```

- 描述

使用此接口设置当前叠加层的 Alpha 功能。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOSET_GLOBAL_ALPHA	ioctl 号	输入
pstAlpha	MI_FB_GlobalAlpha_t 结构体指针	输入

- 返回值

返回值
- 0 成功。
- -1 失败

- 依赖

头文件：mstarFb.h

- 相关接口

FBIOGET_GLOBAL_ALPHA

3.7. FBIOGET_COLORKEY

- 功能

  获取叠加层的 colorkey。

- 语法

  ```
  int ioctl (int fd,FBIOGET_COLORKEY,MI_FB_ColorKey_t* pstColorKey);
  ```

- 描述

  使用此接口获取叠加层的 colorkey。

- 形参

参数名称	描述	输入/输出
Fd	Framebuffer 设备文件描述符	输入
FBIOGET_COLORKEY	ioctl 号	输入
pstColorKey	MI_FB_ColorKey_t 结构体指针	输出

- 返回值

  返回值：0 成功。
  返回值：-1 失败

- 依赖

  头文件：mstarFb.h

- 相关接口

  FBIOSET_COLORKEY

3.8. FBIOSET_COLORKEY

- 功能

  设置叠加层的 colorkey。

- 语法

  ```
  int ioctl (int fd,FBIOSET_COLORKEY,MI_FB_ColorKey_t* pstColorKey);
  ```

- 描述

  使用此接口设置当前叠加层的 colorkey 功能。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOSET_COLORKEY	ioctl 号	输入
pstColorKey	MI_FB_ColorKey_t 结构体指针	输入

- 返回值

返回值
- 0 成功。
- -1 失败

- 依赖

头文件：mstarFb.h

- 相关接口

FBIOGET_COLORKEY

3.9. FBIOSET_DISPLAYLAYER_ATTRIBUTES

- 功能

设置图层信息。

- 语法

```
int ioctl (int fd,FBIOSET_DISPLAYLAYER_ATTRIBUTES,
             MI_FB_DisplayLayerAttr_t* pstLayerInfo);
```

- 描述

此接口用于设置图层信息，包括图层在屏幕的显示区域、画布分辨率、显存分辨率、屏幕显示分辨率以及是否使能预乘。以上信息的更详细说明见MI_FB_DisplayLayerAttr_t的描述。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOSET_DISPLAYLAYER_ATTRIBUTES	ioctl 号	输入
pstLayerInfo	MI_FB_DisplayLayerAttr_t结构体指针	输出

- 返回值

返回值
- 0 成功。
- -1 失败

- 依赖

头文件：mstarFb.h

- 相关接口

FBIOSET_DISPLAYLAYER_ATTRIBUTES

3.10. FBIOGET_ DISPLAYLAYER_ATTRIBUTES

- 功能

获取图层信息。

- 语法

```
int ioctl (int fd,
           FBIOGET_DISPLAYLAYER_ATTRIBUTES,
           MI_FB_DisplayLayerAttr_t * pstLayerInfo);
```

- 描述

用于获取图层信息，包括图层在屏幕的显示区域、显存分辨率、屏幕显示分辨率以及是否使能预乘。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOGET_DISPLAYLAYER_ATTRIBUTES	ioctl 号	输入
pstLayerInfo	图层信息结构体指针	输出

- 返回值

返回值：
- 0 成功。
- -1 失败

- 依赖

头文件：mstarFb.h

- 相关接口

FBIOGET_DISPLAYLAYER_ATTRIBUTES

4. 鼠标相关功能

4.1. FBIOGET_CURSOR_ATTRIBUTE

- 功能

  获取鼠标图层信息。

- 语法

```
int ioctl (int fd,
           FBIOGET_CURSOR_ATTRIBUTE,
            MI_FB_CursorAttr_t *pstCursorAttr)
```

- 描述

  获取鼠标图层信息，包括hotspot、位置、Alpha、Colorkey、是否可见。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOGET_CURSOR_ATTRIBUTE	ioctl 号	输入
pstCursorAttr	MI_FB_CursorAttr_t结构体指针	输出

- 返回值

  返回值：
  - 0 成功。
  - -1 失败

- 依赖

  头文件：mstarFb.h

- 相关接口

  FBIO_SET_CURSOR_ATTRIBUTE

4.2. FBIOSET_CURSOR_ATTRIBUTE

- 功能

  设置鼠标图层信息。

- 语法

```
int ioctl (int fd,
           FBIOSET_CURSOR_ATTRIBUTE,
           MI_FB_CursorAttr_t *pstCursorAttr)
```

- 描述

  设置鼠标图层信息，包括hotspot、位置、Alpha、Colorkey、是否可见。

- 形参

参数名称	描述	输入/输出
fd	Framebuffer 设备文件描述符	输入
FBIOSET_CURSOR_ATTRIBUTE	ioctl 号	输入
pstCursorAttr	MI_FB_CursorAttr_t结构体指针	输出

- 返回值

返回值：
- 0 成功。
- -1 失败

- 依赖

  头文件：mstarFb.h

- 相关接口

  FBIO_GET_CURSOR_ATTRIBUTE

5. 错误码

表 2-2 列出了当函数返回值小于 0 时有可能出现的所有错误码。这些错误码来自标准的 linux 错误码定义，详细内容请参见 linux 内核原码 errno_base.h。错误码可以通过打印 Linux 的标准错误码 errno 查看，或者用 strerror(errno)打印错误信息。

表 2-2 错误码

错误代码	宏定义	描述
1	EPERM	不支持该操作
12	ENOMEM	内存不够
14	EFAULT	传入参数指针地址无效
22	EINVAL	传入参数无效

6. FB 数据类型

6.1. 在标准中定义的数据类型

6.1.1 struct fb_bitfield

- 说明

  位域信息，用于设置像素格式。

- 定义

```
struct fb_bitfield
{
    __u32 offset;      /* beginning of bitfield */
    __u32 length;      /* length of bitfield */
    __u32 msb_right;   /* != 0: Most significant bit is right */
};
```

- 成员

成员名称	描述	支持情况
offset	颜色分量起始比特位。	支持。
length	颜色分量所占比特长度。	支持。
msb_right	右边的比特是否为最高有效位。	只支持该位为 0，即最左边的 bit 为最高有效位。

- 相关数据类型及接口

  无。

6.1.2 struct fb_var_screeninfo

- 说明

  可变的屏幕信息。

- 定义

```
struct fb_var_screeninfo
{
    __u32 xres;             /* visible resolution */
    __u32 yres;
    __u32 xres_virtual;     /* virtual resolution */
    __u32 yres_virtual;
    __u32 xoffset;          /* offset from virtual to visible */
    __u32 yoffset;          /* resolution */

    __u32 bits_per_pixel;   /* guess what */
    __u32 grayscale;        /* != 0 Graylevels instead of colors */

    struct fb_bitfield red;  /* bitfield in fb mem if true color,
    */ struct fb_bitfield green;  /* else only length is
    significant */ struct fb_bitfield blue;
    struct fb_bitfield transp; /* transparency */

    __u32 nonstd;           /* != 0 Non standard pixel format */

    __u32 activate;         /* see FB_ACTIVATE_* */

    __u32 height;           /* height of picture in mm */
    __u32 width;            /* width of picture in mm */

    __u32 accel_flags;      /* (OBSOLETE) see fb_info.flags */

    /* Timing: All values in pixclocks, except pixclock (of course) */
    __u32 pixclock;         /* pixel clock in ps (pico seconds) */
    __u32 left_margin;      /* time from sync to picture */
    __u32 right_margin;     /* time from picture to sync */
    __u32 upper_margin;     /* time from sync to picture */
    __u32 lower_margin;
    __u32 hsync_len;        /* length of horizontal sync */
    __u32 vsync_len;        /* length of vertical sync */
    __u32 sync;             /* see FB_SYNC_* */
    __u32 vmode;            /* see FB_VMODE_* */
    __u32 rotate;           /* angle we rotate counter clockwise */
    __u32 reserved[5];      /* Reserved for future compatibility */
};
```

- 成员

成员名称	描述	支持情况
xres	可见屏幕宽度（像素数）。	支持。
yres	可见屏幕高度（像素数）。	支持。
xres_virtual	虚拟屏幕宽度（显存中图像宽 度），当该值小于 xres 时会修改xres，使xres值与该值相等。	支持。
yres_virtual	虚拟屏幕高度（显存中图像高 度），当该值小于 yres 时会修改yres，使 yres 值与该值相等。结合 xres_virtual，可以用来快速水平或垂直平移图像。	支持。
xoffset	在 x 方向上的偏移像素数。	支持，默认为 0。
yoffset	在 y 方向上的偏移像素数。	支持，默认为 0。
bits_per_pixel	每个像素所占的比特数。	支持。
grayscale	灰度级。	不支持，缺省值为 0，表示彩色。
red	颜色分量中红色的位域信息。	支持。
green	颜色分量中绿色的位域信息。	支持。
blue	颜色分量中蓝色的位域信息。	支持。
transp	颜色分量中 alpha 分量的位域信息。	支持。
nonstd	是否为标准像素格式。	不支持，缺省值为 0，表示支持标准象素格式。

成员名称	描述	支持情况
activate	设置生效的时刻。	不支持，缺省值为 FB_ACTIVATE_NOW，表示设置立刻生效。
height	屏幕高，单位为 mm。	不支持，缺省值为–1。
width	屏幕宽，单位为 mm。	不支持，缺省值为–1。
accel_flags	加速标志。	不支持，缺省值为–1。
pixclock	显示一个点需要的时间，单位为ns。	不支持，缺省值为–1。
left_margin	分别是左消隐信号、右消隐信号、水平同步时长，这三个值之和等于水平回扫时间，单位为点时钟。	不支持，缺省值为64。
right_margin		
hsync_len		
upper_margin	分别是上消隐信号、下消隐信号、垂直同步时长，这三个值之和等于垂直回扫时间，单位为点时钟。	不支持，缺省值为–1。
lower_margin		
vsync_len		
sync	同步信号方式。	不支持，缺省值为–1。
vmode	扫描模式。	不支持，缺省值为–1。
rotate	顺时针旋转的角度。	不支持，缺省值为 0，表示无旋转。

※ 注意事项

高清设备图形层默认的分辨率为 1280x720；标清设备图形层默认的分辨率为 720x576，鼠标层默认的分辨率为 128x128。像素格式为 ARGB1555。

➢ 相关数据类型及接口

struct fb_bitfield
FBIOGET_VSCREENINFO
FBIOPUT_VSCREENINFO

6.1.3 struct fb_fix_screeninfo

➢ 说明

固定的屏幕信息。

➢ 定义

```
struct fb_fix_screeninfo
{
   char id[16];  /* identification string eg "TT Builtin" */
   unsigned long smem_start; /* Start of frame buffer mem
   (physical
                                address) */
   __u32 smem_len;           /* Length of frame buffer mem */
   __u32 type;               /* see FB_TYPE_* */
   __u32 type_aux;           /* Interleave for interleaved Planes */
   __u32 visual;             /* see FB_VISUAL_* */
   __u16 xpanstep;           /* zero if no hardware panning */
   __u16 ypanstep;           /* zero if no hardware panning */
   __u16 ywrapstep;          /* zero if no hardware ywrap */
   __u32 line_length;        /* length of a line in bytes */
   unsigned long mmio_start; /* Start of Memory Mapped I/O
   (physical
                                address) */
   __u32 mmio_len;           /* Length of Memory Mapped I/O */
   __u32 accel; /* Indicate to driver which specific chip/card we have
   */
   __u16 reserved[3];        /* Reserved for future compatibility */
};
```

➢ 成员

成员名称	描述	支持情况
id	设备驱动名称。	支持。
smem_start	显存起始物理地址。	支持。
smem_len	显存大小。	支持。
type	显卡类型。	固定为 FB_TYPE_PACKED_PIXELS，表示像素值紧密排列。
type_aux	附加类型。	不支持，在 FB_TYPE_PACKED_PIXELS 显卡类型下无含义。
visual	色彩模式。	不支持，默认为 FB_VISUAL_TRUECOLOR，真彩色。
xpanstep	支持水平方向上的 PAN 显示： 0：不支持。 非0：支持，此时该值用于表示在水平方向上每步进的像素值。	固定为 1。

成员名称	描述	支持情况
ypanstep	支持垂直方向上的 PAN 显示： 0：不支持。 非0：支持，此时该值用于表示在垂直方向上每步进的像素值。	固定为 1。
ywrapstep	该方式类似于 ypanstep，不同之处在于：当其显示到底部时，能回到显存的开始处进行显示。	不支持，默认为 0。
line_length	每行字节数。	支持。
mmio_start	显存映射 I/O 首地址。	不支持，默认为 0。
mmio_len	显存映射 I/O 长度。	不支持，默认为 0。
accel	显示所支持的硬件加速设备。	不支持，默认为 FB_ACCEL_NONE，无加速设备。
reserved	保留。	不支持，缺省值为 0。

※ 注意事项

无。

➢ 相关数据类型及接口

FBIOGET_FSCREENINFO

6.2. 扩展的数据类型

6.2.1 MI_FB_ColorFmt_e

➢ 说明

MStarFB 支持的像素格式集合。

➢ 定义

```
typedef enum
{
   E_MI_FB_COLOR_FMT_RGB565 = 1,
   E_MI_FB_COLOR_FMT_ARGB4444       =2,
   E_MI_FB_COLOR_FMT_ARGB8888       =5,
   E_MI_FB_COLOR_FMT_ARGB1555       =6,
   E_MI_FB_COLOR_FMT_YUV422         =9,
  /// Invalid color format.
  E_MI_FB_COLOR_FMT_INVALID = 12,
}MI_FB_ColorFmt_e;
```

➢ 成员

成员名称	描述
E_MI_FB_COLOR_FMT_RGB565	RGB565 格式。
E_MI_FB_COLOR_FMT_ARGB4444	ARGB444 格式。
E_MI_FB_COLOR_FMT_ARGB8888	ARGB8888
E_MI_FB_COLOR_FMT_ARGB1555	ARGB1555
E_MI_FB_COLOR_FMT_YUV422	YUV422
E_MI_FB_COLOR_FMT_INVALID	无效像素格式

※ 注意事项

无。

➢ 相关数据类型及接口

MI_FB_DisplayLayerAttr_t
MI_FB_CursorAttr_t

6.2.2 MI_FB_DisplayLayerAttrMaskbit_e

➢ 说明

标识MI_FB_DisplayLayerAttr_t结构体内哪些成员有更新

- 定义

```
typedef enum
{
 E_MI_FB_DISPLAYLAYER_ATTR_MASK_DISP_POS = 0x1,
   E_MI_FB_DISPLAYLAYER_ATTR_MASK_DISP_SIZE = 0x2,
   E_MI_FB_DISPLAYLAYER_ATTR_MASK_BUFFER_SIZE = 0x4,
   E_MI_FB_DISPLAYLAYER_ATTR_MASK_SCREEN_SIZE = 0x8,
   E_MI_FB_DISPLAYLAYER_ATTR_MASK_PREMUL = 0x10,
   E_MI_FB_DISPLAYLAYER_ATTR_MASK_COLOR_FMB = 0x20,
 E_MI_FB_DISPLAYLAYER_ATTR_MASK_OUTPUT_COLORSPACE = 0x40,
 E_MI_FB_DISPLAYLAYER_ATTR_MASK_DST_DISP = 0x80,
} MI_FB_DisplayLayerAttrMaskbit_e;
```

- 成员

成员名称	描述
E_MI_FB_DISPLAYLAYER_ATTR_MASK_DISP_POS	叠加层位置发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_DISP_SIZE	叠加层在屏幕上的显示区域的宽度或高度发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_BUFFER_SIZE	显示分辨率发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_SCREEN_SIZE	屏幕分辨率发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_PREMUL	是否预乘法属性发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_COLOR_FMB	FB的Color format属性发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_OUTPUT_COLORSPACE	叠加层输出的Color Space发生变化
E_MI_FB_DISPLAYLAYER_ATTR_MASK_DST_DISP	叠加层输出到display chain目标模组发生变化

- 成员

无。

- 相关数据类型及接口

MI_FB_DisplayLayerAttr_t

6.2.3 MI_FB_GlobalAlpha_t

- 说明

Alpha 信息结构体

➢ 定义

```
typedef struct MI_FB_GlobalAlpha_s
{
   MI_BOOL bAlphaEnable;     /* alpha enable flag */
   MI_BOOL bAlphaChannel;    /* alpha channel enable flag
   */

   MI_U8 u8Alpha0;           /* alpha0 value */
   MI_U8 u8Alpha1;           /* alpha1 value */
   MI_U8 u8GlobalAlpha;      /* global alpha value
   */ MI_U8 u8Reserved;
} MI_FB_GlobalAlpha_t;
```

➢ 成员

成员名称	描述
bAlphaEnable	Alpha 叠加使能，默认为 1。
bAlphaChannel	Alpha 通道使能，默认为 0。
u8Alpha0	Alpha0 值，范围 0～255，默认为 255。在 RGB1:5:5:5 格式下，当最高位为 0 时，选择该值作为 Alpha 叠加的 Alpha 值。
u8Alpha1	Alpha1 值，范围 0～255，默认为 255。在 RGB1:5:5:5 格式下，当最高位为 1 时，选择该值作为 Alpha 叠加的 Alpha 值。
u8GlobalAlpha	全局 Alpha 值，范围为 0～255，默认为 255。在 Alpha 通道使能时起作用。
u8Reserved	保留

※ 注意事项

只有在 Alpha 叠加使能的情况下才进行 Alpha 叠加，否则处于上层的叠加层将覆盖下层的叠加层。

叠加 Alpha 值的计算公式有以下几种情况：

当Alpha 通道使能时，全局 Alpha 参与叠加。

对于不支持全局 Alpha 和像素 Alpha 叠加的芯片，叠加 Alpha 值的计算公式如下所示：$\alpha = u8GlobalAlpha$

对于支持全局 Alpha 和像素 Alpha 叠加的芯片，叠加 Alpha 值的计算公式如下所示：$\alpha = u8GlobalAlpha * \alpha_{pixel}$

当Alpha 通道不使能时，叠加 Alpha 值等于像素 Alpha 值，即：$\alpha = \alpha_{pixel}$

➢ 相关数据类型及接口

FBIOGET_GLOBAL_ALPHA
FBIOSET_GLOBAL_ALPHA

6.2.4 MI_FB_ColorKey_t

➢ 说明

Colorkey信息 结构体，用于 colorkey 的属性设置。

➢ 定义

```
typedef struct MI_FB_ColorKey_s
{
 MI_BOOL bKeyEnable;
   MI_U8 u8Red;
   MI_U8 u8Green;
   MI_U8 u8Blue;
} MI_FB_ColorKey_t
```

➢ 成员

成员	描述
bKeyEnable	Colorkey 是否使能标识。 TRUE：使能； FALSE：不使能。
u8Red	Colorkey的R分量数值
u8Green	Colorkey的G分量数值
u8Blue	Colorkey的B分量数值

※ 注意事项

无。

➢ 相关数据类型及接口

FBIOGET_COLORKEY

FBIOSET_COLORKEY

6.2.5 MI_FB_Rectangle_t

➢ 说明

MI_FB_Rectangle_t结构体，用于描述叠加层的显示区域

➢ 定义

```
typedef struct MI_FB_Rectangle_s
{
        MI_U32 u32Xpos;
        MI_U32 u32Ypos;
        MI_U32 u32Width;
        MI_U32 u32Height;
}MI_FB_Rectangle_t;
```

➢ 成员

成员	描述
u32Xpos	左上角x坐标
u32Ypos	左上角y坐标
u32Width	宽度
u32Height	高度

※ 注意事项

无。

➢ 相关数据类型及接口

FBIOGET_SCREEN_LOCATION
FBIOSET_SCREEN_LOCATION

6.2.6 MI_FB_DisplayLayerAttr_t

➢ 说明

MI_FB_DisplayLayerAttr_t结构体，描述叠加层的属性信息。

➢ 定义

```
typedef struct MI_FB_DisplayLayerAttr_s
{
    MI_U32 u32Xpos;       /**the x pos of orign point in screen*/
    MI_U32 u32YPos;      /**the y pos of orign point in screen*/
    MI_U32 u32dstWidth; /**display buffer dest with in screen*/
    MI_U32 u32dstHeight; /**display buffer dest hight in screen*/
    MI_U32 u32DisplayWidth;    /**the width of display buf in fb */
    MI_U32 u32DisplayHeight;    /**the height of display buf in fb. */
    MI_U32 u32ScreenWidth;    /**the width of screen */
```

```
    MI_U32 u32ScreenHeight; /** the height of screen */
    MI_BOOL bPreMul;    /**the data drawed in buffer whether is premultiply alpha or not*/
    MI_FB_ColorFmt_e eFbColorFmt; /**the color format of framebuffer*/
    MI_FB_OutputColorSpace_e    eFbOutputColorSpace;    /**output color space*/
    MI_FB_DstDisplayplane_e    eFbDestDisplayPlane;    /**destination displayplane*/
    MI_U32 u32SetAttrMask; /** display attribute modify mask*/
}MI_FB_DisplayLayerAttr_t;
```

- 成员

成员	描述
u32Xpos	图层在屏幕上的原点横坐标。
u32YPos	图层在屏幕上的原点纵坐标。
u32dstWidth	图层在屏幕上的显示宽度
u32dstHeight	图层在屏幕上的显示高度
u32DisplayWidth	显存分辨率的宽。
u32DisplayHeight	显存分辨率的高。
u32ScreenWidth	屏幕显示分辩率的宽。
u32ScreenHeight	屏幕显示分辩率的高。
bPreMul	FB 中的数据是否为预乘数据。
eFbColorFmt	FB的像素格式。
u32SetAttrMask	设置图层信息时参数修改掩码位。
eFbOutputColorSpace	设置叠加层输出的Color space
eFbDestDisplayPlane	设置叠加层输出到display chain的目的模组。默认为E_MI_FB_DST_OP0。

- 相关数据类型及接口

  MI_FB_DisplayLayerAttrMaskbit_e

  MI_FB_ColorFmt_e

  MI_FB_DstDisplayplane_e

  MI_FB_OutputColorSpace_e

6.2.7 MI_FB_OutputColorSpace_e

- 说明

MStarFB叠加层输出的Color Space。

- 定义

```
typedef enum
{
  E_MI_FB_OUTPUT_RGB = 0,
  E_MI_FB_OUTPUT_YUV = 1
}MI_FB_OutputColorSpace_e
```

- 成员

成员	描述
E_MI_FB_OUTPUT_RGB	RGB ColorSpace
E_MI_FB_OUTPUT_YUV	YUV ColorSpace

※ 注意事项

无。

- 相关数据类型及接口

MI_FB_DisplayLayerAttr_t

6.2.8 MI_FB_DstDisplayplane_e

- 说明

MStarFB叠加层输出到display chain的目的模组。

- 定义

```
typedef enum
{
   E_MI_FB_DST_IP0 = 0,
   E_MI_FB_DST_MIXER2VE = 1,
   E_MI_FB_DST_OP0 = 2,
   E_MI_FB_DST_VOP = 3,
   E_MI_FB_DST_IP1 = 4,
   E_MI_FB_DST_IP_MAIN = 5,
   E_MI_FB_DST_IP_SUB = 6,
   E_MI_FB_DST_MIXER2OP = 7,
   E_MI_FB_DST_VOP_SUB = 8,
```

```
    E_MI_FB_DST_FRC = 9,
    E_MI_FB_DST_VE = 10,
    E_MI_FB_DST_OP1 = 11,
    E_MI_FB_DST_MIXER2OP1 = 12,
    E_MI_FB_DST_DIP = 13,
    E_MI_FB_DST_GOPScaling = 14,
    E_MI_FB_DST_BYPASS = 15,
    E_MI_FB_DST_OP_DUAL_RATE = 16,
}MI_FB_DstDisplayplane_e
```

➢ 成员

成员	描述
E_MI_FB_DST_IP0	IP0 path
E_MI_FB_DST_MIXER2VE	Mixer to VE path
E_MI_FB_DST_OP0	OP path
E_MI_FB_DST_VOP	VOP path
E_MI_FB_DST_IP1	IP1 path
E_MI_FB_DST_IP_MAIN	IP man path
E_MI_FB_DST_IP_SUB	IP sub path
E_MI_FB_DST_MIXER2OP	Mixer to OP Path
E_MI_FB_DST_VOP_SUB	VOP path
E_MI_FB_DST_FRC	FRC path
E_MI_FB_DST_VE	DIRECT TO VE path
E_MI_FB_DST_OP1	OP1 path
E_MI_FB_DST_MIXER2OP1	MIXER2OP1
E_MI_FB_DST_DIP	DIP path
E_MI_FB_DST_GOPScaling	GS path
E_MI_FB_DST_BYPASS	4K2K BYPASS path
E_MI_FB_DST_OP_DUAL_RATE	op with double frequency

※ 注意事项

无。

➢ 相关数据类型及接口

MI_FB_DisplayLayerAttr_t

6.2.9 MI_FB_CursorAttrMaskbit_e

- 说明

  描述MI_FB_CursorAttr_t结构体哪些属性发生变化

- 定义

```
typedef enum
{
    E_MI_FB_CURSOR_ATTR_MASK_ICON = 0x1,
    E_MI_FB_CURSOR_ATTR_MASK_POS = 0x2,
    E_MI_FB_CURSOR_ATTR_MASK_ALPHA = 0x4,
    E_MI_FB_CURSOR_ATTR_SHOW = 0x8,
    E_MI_FB_CURSOR_ATTR_HIDE = 0x10,
    E_MI_FB_CURSOR_ATTR_MASK = 0x1F
} MI_FB_CursorAttrMaskbit_e;
```

- 成员

成员	描述
E_MI_FB_CURSOR_ATTR_MASK_ICON	鼠标图标发生变化
E_MI_FB_CURSOR_ATTR_MASK_POS	鼠标位置发生变化
E_MI_FB_CURSOR_ATTR_MASK_ALPHA	Alpha发生变化
E_MI_FB_CURSOR_ATTR_SHOW	显示鼠标
E_MI_FB_CURSOR_ATTR_MASK_HIDE	隐藏鼠标
E_MI_FB_CURSOR_ATTR_MASK	用于标识鼠标所有可设置属性

- 相关数据类型及接口

  MI_FB_CursorAttr_t

6.2.10 MI_FB_CursorImage_t

- 说明

  MI_FB_CursorImage_t结构体, 描述鼠标图标数据信息。

- 定义

```
typedef struct MI_FB_CursorImage_s
{
MI_U32 u32Width; /**width, unit pixel*/
MI_U32 u32Height; /**Height, unit pixel*/
MI_U32 u32Pitch; /**Pitch, unit pixel*/
MI_FB_ColorFmt_e eColorFmt; /**Color format*/
const char* data;
}MI_FB_CursorImage_t;
```

➢ 成员

成员	描述
u32Width	图标宽度
u32Height	图标高度
u32Pitch	pitch
eColorFmt	图标数据像素格式
data	图标数据

※ 注意事项

图标的宽度和高度的最大值为128。

➢ 相关数据类型及接口

MI_FB_CursorAttr_t
MI_FB_ColorFmt_e

6.2.11 MI_FB_CursorAttr_t

➢ 说明

MI_FB_CursorAttr_t结构体, 描述鼠标图层信息

➢ 定义

```
typedef struct MI_FB_CursorAttr_s
{
    MI_U32 u32XPos;
    MI_U32 u32YPos;
    MI_U32 u32HotSpotX;
    MI_U32 u32HotSpotY;
    MI_FB_GlobalAlpha_t stAlpha;
    MI_FB_ColorKey_t stColorKey;
    MI_BOOL bShown;
  MI_FB_CursorImage_t stCursorImageInfo;
    MS_U16 u16CursorAttrMask;
}MI_FB_CursorAttr_t;
```

- 成员

成员	描述
u32XPos	鼠标位置横坐标
u32YPos	鼠标位置纵坐标
u32HotSpotX	鼠标热点横坐标
u32HotSpotY	鼠标热点纵坐标
stAlpha	MI_FB_GlobalAlpha_t结构体,鼠标图层alpha信息
stColorKey	MI_FB_ColorKey_t结构体，鼠标图层colorkey信息
bShown	鼠标是否可见 TRUE表示可见，FALSE表示不可见
MI_FB_CursorImage_t	图标数据信息。
u16CursorAttrMask	设置鼠标图层参数修改掩码位

- 相关数据类型及接口

  MI_FB_CursorAttrMaskbit_e

  MI_FB_CursorImage_t