S1D13716 API 编程指南

ECC SH
<table>
<thead>
<tr>
<th>日期</th>
<th>版本号</th>
<th>修改内容</th>
<th>修改人</th>
</tr>
</thead>
<tbody>
<tr>
<td>2004-07-07</td>
<td>V1.3</td>
<td>创建</td>
<td>ECC SH</td>
</tr>
</tbody>
</table>

修改履历
目录

1. 体系结构和编程方法 .............................................................................................................. 4
   1.1 体系结构.......................................................................................................................... 4
   1.2 编程方法.......................................................................................................................... 4
   1.3 一个实例............................................................................................................................ 5
2. API 函数详解 .......................................................................................................................... 7
   2.1 初始化................................................................................................................................ 7
   2.2 显示功能 ............................................................................................................................ 8
      2.2.1 得到窗体的地址(相对基本地址) ............................................................................... 8
      2.2.2 读写显存....................................................................................................................... 8
      2.2.4 特效显示 ..................................................................................................................... 8
      2.2.4 举例............................................................................................................................. 8
   2.3 画中画功能 ........................................................................................................................ 9
      2.3.1 开启 OVERLAY 窗口 ................................................................................................. 9
      2.3.2 设置 OVERLAY 窗口透明 ......................................................................................... 9
   2.4 CAMERA 功能 ................................................................................................................... 10
      2.4.1 预览............................................................................................................................. 10
   2.5 JPEG 功能 ........................................................................................................................ 11
      2.5.1 jpeg 压缩 ..................................................................................................................... 11
      2.5.2 jpeg 解压 ..................................................................................................................... 12
3. 关于函数重入 ......................................................................................................................... 13
S1D13716 是一款 EPSON 开发的高性能移动图像引擎 (MGE)，配以 EPSON 的专用 API，能充分发挥它的性能，且非常简单易用。使得开发人员迅速开发出高性能的应用程序，而不必花费大量的时间留连在纷杂的硬件设置中，从而把精力集中在系统的设计上，缩短开发周期。

1. 体系结构和编程方法

1.1 体系结构

用户的硬件平台总是千变万化，在 API 设计上我们充分考虑了这一点。出于一种灵活、简单且通用的考虑，我们采用了分层的体系结构：

这种体系在 HAL 实现上层应用和底层硬件的完全脱离。当您需要把它移植到您的硬件平台时，只需修改上图中的“配置”一项即可。至于“配置”的修改我们有专门的文档说明。实际上“配置”的修改相当简单，它是一个结构体，你要修改的其实是这个结构体内的一些参数，而硬件工程师会告诉您这些参数如何设置。

1.2 编程方法

实际上基于此 API 的编程非常容易，你只需做一件事情：调用 S1D13716 的两个初始化函数。
下面以实例来演示如何编制自己的应用程序:

### 1.3 一个实例

**功能：**
- 初始化控制器和液晶；
- 在 LCD 上显示一幅图；
- 打开 Main Window 窗口；
- 启动 CAMERA 预览；
- 设置预览尺寸把预览输出到 Main Window 窗口；
- 对 CAMERA 传送来的图像按指定的参数进行压缩。

```c
extern unsigned short car[];    //欲显示在主窗口上的图像
extern JPEGINFO JpegEInfo;   //JPEG 压缩的参数结构体
extern PREVIEWINFO ZoomInfo;  //预览信息结构体

void main(void)
{
    Boolean ret;
    UInt32 addr1,addr2;
    UInt32 memadrs;

    //第一步：初始化

    if (FALSE == halAcquireController())//
        return;
    if (FALSE == halInitController(fINIT_NORMAL))// 初始化 LCDC, LCD and CAMERA
        return;

    //第二步：调用 API 函数实现所需功能

    addr1=GetStartAddress(cl_MAIN_WINDOW);//得到主窗口的地址
    addr2=GetStartAddress(cl_OVERLAY_WINDOW);//得到 overlay 窗口地址
    halWriteDisplay8(addr2, 0xe0, 128*160);//把 overlay 写成红色
    clS tartOneFrameTransfer();//刷屏
    halWriteDisplay8(addr2+128*16, 0x01, 128*128);//设置透明区域
    halWriteData(sizeof(UInt16),addr1, car,128*128*2);//在主窗口上显示一幅图
```
clStartOneFrameTransfer(); // 帧屏
prvPreviewStart(); // 打开 CAMRA 预览
prvPreviewZoom(&ZoomInfo); // 调整预览尺寸
ret=jpegEncode(&JpegEInfo); // 进行 JPEG 压缩

这样一个复杂的功能，实现起来是不是非常简单！
2. API 函数详解

API 函数共分 5 大类，

- 初始化
- 显示功能
- 画中画功能
- CAMERA 功能
- JPEG 功能

下面就每一种 API 进行系统的全面的介绍。

2.1 初始化

在程序的基本结构中已经介绍了初始化函数，如果在 HAL 移植时没有改变原有函数形式。那么完成初始化只需调用函数。

- Boolean halAcquireController()
- Boolean halInitController( UInt32 Flags )

HallInitController 按指定的初始化条件（Flags），初始化 13716 和液晶屏幕，camera。
2.2 显示功能

图像显示 API 分为两个部分，一是窗口信息，另一个是读写显存。
13716 内部有 112K RAM，这些 RAM 可由用户配置，自由度很大非常灵活。
RAM 具体配置一般由 HAL 来完成。这里只介绍得到这些配置信息的 API。

2.2.1 得到窗体的地址(相对基本地址)

有时候你需要知道这个地址，例如在设置 CAMERA 的预览输出到主窗体或 Overlay 窗体
不同位置时，要告诉 CAMERA 输出地址。

```c
UInt32 GetStartAddress(WindowDef window);
```

window 可以是主窗体 cl_MAIN_WINDOW, 或是 overlay 窗体 cl_OVERLAY_WINDOW。

2.2.2 读写显存

要想在屏幕上显示图像，得到图像就要读写显存，这些函数是：
```c
void halReadData( int AccessSize, UInt32 Offset, void* pData, UInt32 nBytes )
void halWriteData( int AccessSize, UInt32 Offset, const void* pData, UInt32 nBytes )
```
第一个参数 AccessSize 指定访问按 8 位、16 位还是 32 位进行
第二个参数 Offset 指定要读写的显存地址(相对基本地址)。

2.2.4 特效显示

13716 还支持一些特效显示，如旋转、镜像和画中画。画中画在后面有详细介绍。

a) 旋转 void SetSwivelViewMode(WindowDef window, int mode)

```c
window：要设置显示模式的窗口，主窗口 cl_MAIN_WINDOW 或 PIP 窗口
cl_PIP_WINDOW。
mode：旋转角度 (0, 90, 180, 270 期它值做 0 处理)。
```

b) 左右镜像 void SetWinMirror(WindowDef window, Boolean Enable)

```c
window：镜像的窗口，主窗口 cl_MAIN_WINDOW 或 PIP 窗口 cl_PIP_WINDOW。
enable：指定是对窗口否镜像。
```

2.2.4 举例

car 是一幅 16 位 RGB 汽车图案，把她显示在主窗口上。

```c
addr=GetStartAddress(cl_MAIN_WINDOW);//得到主窗口的地址
halWriteData( sizeof(UInt16), addr, car, 128*160*2 );//在主窗口上显示一幅图
```
2.3 画中画功能

13716 提供了强大的画中画功能，画中画的大小可以配置，可以以 Main Image 方式显示在 Overlay 窗体上。在屏幕上 Overlay 窗口和 MAIN 窗口是交叠显示的，Overlay 覆盖显示在 Main 之上，被 Overlay 遮盖的部分是看不见的，但 OVERLAY 透明功能可以使 Main 内容显示出来。下面是效果图：

相关函数是:

2.3.1 开启 OVERLAY 窗口

a) void SetOverlayEnable(Boolean Enable)

作用是开启 OVERLAY 窗口。
Enable:  FALSE 关闭 OVERLAY 窗口
         TRUE 打开 OVERLAY 窗口

2.3.2 设置 OVERLAY 窗口透明

要使 OVERLAY 透明，只要使要透明的区域像素值为 1 就可以了。Overlay 是 16bpp 时是 0x0001，8bpp 时是 0x01。
2.4 CAMERA 功能

13716 提供了 camera 接口。Camera 初始化在 hal 中完成。对于从 CAMERA 过来的数据，13716 有很强的处理功能，它可以对它截取（trim）、缩放（scale）、jpeg 压缩、选择输出地址等。如果把图像输出到 main 窗口，就可以使用强大的 overlay 功能，作出各种特殊应用，例如像框（overlay 透明）等，再利用 13716 的显存压缩功能就可以保存这些特效照片。

2.4.1 预览

a) void prvPreviewStart() 开始预览  
b) void prvPreviewStop() 结束预览  
c) void prvPreviewZoom(PREVIEWINFO * pPreviewInfo) 设置预览的区域（截取）和缩放。

截取示意图

scale 示意图

typedef struct tagPREVIEWINFO
{
(UInt16 XStart;
(UInt16 YStart;
(UInt16 Width;
(UInt16 Height;
(UInt8 ScaleRate;
}PREVIEWINFO;

这个函数完成从 camera 图像中截取一个矩形区域，然后对它进行缩放，最后把处理后的图像输出到 main window 指定的显存地址上。然后你可以用 SetSwivelViewMode 等图像显示函数作一些想要的处理，例如旋转 180 度，镜像。
2.5 JPEG 功能

13716 提供硬件 jpeg 的压缩，数据源可以是显存里的任何区域和 camera 过来的图像。还可以对数据源进行截取和缩放压缩。结合 camera 和 overlay 功能可以让你非常容易地制作出精美的相片和图片。

JPEG 库函数可以对 CAMERA 图形进行多帧压缩。

2.5.1 jpeg 压缩

jpeg 压缩很简单，只要在一个结构体中指定压缩的基本信息，然后调用一个函数即可完成。解压和压缩使用的是同一个结构体。

a) Boolean jpegEncode( JENCODEINFO* pJEncodeInfo )
这个结构体结构为
typedef struct tagJPEGINFO
{
  U1nt8* m_pBuf;   // 存储 jpeg 压缩或要解压的数据
  U1nt32 m_BufSize;  // m_pBuf 的大小，字节单位
  U1nt32 m_Delay;   // 多帧压缩时每帧之间的间隔
  U1nt32 m_EncSize;  // 压缩后的 jpeg 文件大小，或解压时设置 jpeg 文件大小
  char* m_pszFile;   // 如果你用文件系统，可以用它存储 jpeg 文件
  Boolean m_fFastCpu;  // 一般设成 TRUE。
  U1nt32 m_FrameCount;  // jpeg 压缩的帧数。
  U1nt16 m_Height;   // 压缩时设置压缩区域的高度
  U1nt16 m_HtScaled;  // 解压后返回图像的高度
  U1nt8 m_HScaleRate;  // 设置数据源水平方向（camera/memory）将要被缩小的倍数
  Boolean m_fIndScale;  // 设置水平和垂直的比例是否相同
  Boolean m_fInfLoop;  // 此值设为 TRUE 则，多帧压缩时，在 buffer 中下
  // 一幅覆盖上一幅，否则按一定格式顺序存放。
  Boolean m_fMotionJPEG; // 设置是否压制成 motion jpeg 格式
  Boolean m_fMulFrm;  // 如果你用文件系统，设置是否是存储多帧 jpeg 文件
  Boolean m_fPause;   // 保留变量，调试时使用
  U1nt16 m_RWidth;   // 如果设值这个值大于零，
  // 这个参数设置截取的实际需要压缩图像的宽度。
  U1nt16 m_RHeight;  // 如果设值这个值大于零，
  // 这个参数设置截取的实际需要压缩图像的高度。
  Boolean m_fScale;   // 是否进行缩放
  U1nt8 m_ScaleMode;  // 设置缩放的模式
  Boolean m_fSOE;   // 压缩是是否进行 overlay 的缩放
  U1nt8 m_VScaleRate;  // 设置数据源垂直方向（camera/memory）将要被缩小的倍数
  U1nt16 m_XStart;  // 13716 有 trim 截取功能，能够从 camera 或
  // memory 过来的图像中截取一块。
  // 这个参数设置裁剪的起始 X 坐标。
### 2.5.2 jpeg 解压

同 jpeg 压缩一样，解压也很简单，也是填充结构体，然后调用解压函数即可。

a) Boolean jpegDecode(JDECODEINFO *pJdecode) 单帧和多帧 jpeg 解压函数
   结构体请参照压缩函数。

b) Boolean jpegMDecode(JDECODEINFO *pJdecode) Motion jpeg 解压函数
   结构体请参照压缩函数。

<table>
<thead>
<tr>
<th>错误代码</th>
<th>描述</th>
</tr>
</thead>
<tbody>
<tr>
<td>JPEG_ERR_NONE</td>
<td>没有错误。</td>
</tr>
<tr>
<td>JPEG_ERR_LINEBUF_OVERFLOW</td>
<td>Line buffer 溢出。</td>
</tr>
<tr>
<td>JPEG_ERR_ENCSIZELIMIT</td>
<td>超过大小限制。</td>
</tr>
<tr>
<td></td>
<td>Fifo 没有数据。</td>
</tr>
<tr>
<td>JPEG_ERR_JPEG_FULL</td>
<td>Fifo 满了。</td>
</tr>
<tr>
<td>JPEG_ERR_NULL_BUFFER</td>
<td>存放压缩数据的内存没有分配，是 NULL。</td>
</tr>
<tr>
<td>JPEG_ERR_ENCODE_OVERFLOW</td>
<td>存放压缩数据的内存已用尽。</td>
</tr>
<tr>
<td>JPEG_ERR_NO_ENDOFIMAGE</td>
<td>压缩完成后或要解压的文件，文件结尾处不是 0xffd9 (JPEG 图像结束标记)</td>
</tr>
<tr>
<td>JPEG_ERR_NO_JPEG_DATA</td>
<td>压缩完成后没有得到压缩尺寸。</td>
</tr>
<tr>
<td>JPEG_ERR_NO_STARTOFIMAGE</td>
<td>压缩完成后或要解压的文件没有 ffd8 标记 (JPEG 图像标记)</td>
</tr>
<tr>
<td>JPEG_ERR_ZERO_BUFFER_SIZE</td>
<td>分配的内存尺寸为零。</td>
</tr>
<tr>
<td>JPEG_ERR_UNKNOWN_ERROR</td>
<td>未知错误。</td>
</tr>
</tbody>
</table>
3. 关于函数重入

目前，本函数提供的库函数没有实现函数可重入，但是提供了函数入口：

```c
void halSetProcessExclusivity( Boolean fExclusive )
```

如果用户可能会因为中断处理和任务切换，调用本函数库中函数，引起函数重入并导致显示不正常的现象，请参照具体硬件手册和提供的源代码，实现此函数进行必要的函数重入保护或互斥管理。函数的位置在 hal 中，具体目录是：

```
S1D13716LIB\SOURCECODEOFHAL\SOURCECODE\HAL.C
```