图片显示

1，常见图片格式介绍（了解）

1.1，常见图片格式

图片格式	文件后缀名	优点	缺点	说明
BMP	.bmp	一般不压缩、画质最好	随分辨率和颜色数增加，占用空间急剧增大	图像色深可选：1、4、8、16、24、32bit
JPEG（JPG）	.jpeg（.jpg）	压缩技术先进，支持多种压缩级别，压缩比高(有损压缩)、文件体积较小	画质有一定损失	在不影响人类可分辨的图片质量的前提下，尽可能的压缩文件大小
GIF	.gif	支持有/无损压缩、压缩比高、支持动画、支持背景透明、文件小	仅支持8位色深(256色)、画质差	文件小，利于网络传输

图片编解码相关资料路径：正点原子开发板资料盘(A盘)\6，软件资料\图片编解码

1.2，BMP

BMP全称：Bitmap(位图)，是windows中的标准图像文件格式
采用位映射存储方式，不做任何压缩，图像深度可选：1、4、8、16、24、32bit
BMP文件存储数据时，图像的扫描方式是按从左到右、从下到上的顺序

优点：不失真，画质好，缺点：占用空间大

BMP图像文件数据结构由四部分组成：位图文件头、位图信息头、颜色表、位图数据

详细介绍参考：《BMP图片文件详解.pdf》

1.3，JPEG（JPG）

JPEG全称：Joint Photographic Experts Group(联合图像专家小组)，是常用的图像文件格式，采用有损压缩格式，能够将图像压缩在很小的存储空间。
JPEG压缩技术先进，允许用不同压缩比对文件进行压缩，支持多种压缩级别，压缩比率通常在 10:1 到 40:1 之间，压缩比越大，品质就越低。
因此需要在图像质量和存储空间之间选择一个平衡点。

优点：可控的失真换来更小的存储空间，缺点：图像失真

1.4，GIF

GIF全称：Graphics Interchange Format (图像互换格式)，是常用的图像文件格式
GIF主要分为两个版本，即GIF 89a和GIF 87a
GIF 87a：是在1987年制定的版本
GIF 89a：是1989年制定的版本
GIF 文件格式采用了 LZW(Lempel-ZivWalch)压缩算法来存储图像数据，允许用户为图像设置背景透明属性，并支持动画功能

优点：支持有/无损压缩、压缩比高、支持动态效果，缺点：最多256色(8位色深)、画质差

2，图片显示实验（熟悉）

2.1，图片显示实验文件简介

PICTURE文件夹的文件	作用
bmp.c、bmp.h	实现对bmp文件的解码
tjpgd.c、tjpgd.h、tjpgdcnf.h	实现对jpeg/jpg文件的解码
gif.c、gif.h	实现对gif文件的解码
piclib.c、piclib.h	图片解码库代码

2.2，图片显示物理层接口结构体

```
/* 图片显示物理层接口 */
/* 在移植的时候,必须由用户自己实现这几个函数 */
typedef struct
{
    /* 读点函数 */
    uint32_t(*read_point)(uint16_t, uint16_t);
    /* 画点函数 */
    void(*draw_point)(uint16_t, uint16_t, uint32_t);
    /* 单色填充函数 */
    void(*fill)(uint16_t, uint16_t, uint16_t, uint16_t, uint32_t);
    /* 画水平线函数 */
    void(*draw_hline)(uint16_t, uint16_t, uint16_t, uint16_t);
    /* 颜色填充 */
    void(*fillcolor)(uint16_t, uint16_t, uint16_t, uint16_t, uint16_t *);
} pic_phy;
```

2.3，图片信息结构体

```
/* 图像信息 */
typedef struct
{
    uint16_t lcdwidth;          /* LCD的宽度 */
    uint16_t lcdheight;         /* LCD的高度 */
    uint32_t ImgWidth;          /* 图像的实际宽度和高度 */
    uint32_t ImgHeight;
    uint32_t Div_Fac;           /* 缩放系数 (扩大了8192倍的) */
    uint32_t S_Height;          /* 设定的高度和宽度 */
    uint32_t S_Width;
    uint32_t S_XOFF;            /* x轴和y轴的偏移量 */
    uint32_t S_YOFF;
    uint32_t staticx;           /* 当前显示到的 x y 坐标 */
    uint32_t staticy;
} _pic_info;
```

2.4，图片解码库接口函数

函数	作用
piclib_init()	画图初始化函数，初始化_pic_info和_pic_phy结构体变量
piclib_alpha_blend()	快速ALPHA BLENDING算法函数，用于实现半透明效果
piclib_ai_draw_init()	初始化智能画点函数，用于设置图片居中显示
piclib_is_element_ok()	判断一个像素是否应该显示出来，在图片缩放时用到
piclib_ai_load_picfile()	智能画图函数，实现bmp/jpg/jpeg和gif格式图片的显示
*piclib_mem_malloc()	动态分配内存函数
piclib_mem_free()	释放内存函数

2.5，图片显示实验演示

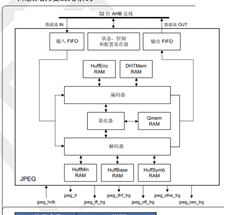
准备工作：需要准备一张TF卡，然后在TF卡根目录下新建一个PICTURE文件夹，并在里面存放bmp、jpg、jpeg或gif格式的图片文件

实验功能：在TFTLCD上循环显示支持的图片，通过KEY0和KEY1可以快速浏览下一张和上一张，KEY_UP按键用于暂停/继续播放，LED1用于指示当前是否处于暂停状态

3，硬件JPEG解码实验（熟悉）

3.1，STM32 硬件JPEG编解码器

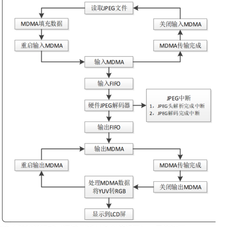
STM32H7的JPEG编解码器具有如下特点：
支持 JPEG 编码/解码
支持 RGB、YCbCr、YCMK 与 BW（灰度）色彩空间
单周期解码/编码一个像素
支持 JPEG 头数据编解码
多达 4 个可编程量化表
单周期动太最表编解码
完全可编程的哈弗曼表（AC 和 DC 各 2 个）
完全可编程的最小编码单元（MCU）
单周期动弗曼编码/解码



信号名称	说明
jpeg_hclk	JPEG 内核和寄存器接口时钟
jpeg_it	JPEG 全局中断
jpeg_ift_trg	用于 MDMA 的 JPEG 输入 FIFO 阈值信号
jpeg_ifnf_trg	用于 MDMA 的 JPEG 输入 FIFO 未满信号
jpeg_oft_trg	用于 MDMA 的 JPEG 输出 FIFO 阈值信号
jpeg_ofne_trg	用于 MDMA 的 JPEG 输出 FIFO 非空信号
jpeg_oec_trg	用于 MDMA 的 JPEG 转换结束信号

3.2，MDMA简介

硬件JPEG解码时FIFO数据的处理（读取/写入）有两种方式：1，中断方式 2，DMA 方式



MDMA 支持 4 种触发模式（通过 TRGM[1:0]位设置），具体如下：
1、缓冲传输模式（最多一次可以传输 128 字节）
2、块传输模式（最多一次可以传输 64K 字节）
3、重复块传输模式（每次传输后，可配置改变源/目标的起始地址）
4、链表模式（直到传个信道数据传输完毕，比较复杂，一般不用）
而 DMA1/DMA2 只支持单次传输和突发传输，且突发传输仅支持 4、 8 和 16 等个传输长度，本实验我们用 MDMA 的缓冲传输模式，因为 JPEG 的输入/输出 FIFO 每次都是读取/输出 32 字节，因此我们用缓冲模式（TRGM[1:0]=00），并设置单次触发传输长度为 32 字节（TLEN[6:0]=32-1），就可以很好的和 JPEG 的输入/输出 FIFO 匹配了

MDMA 的详细介绍请参考《STM32H7xx 参考手册_V7（英文版）.pdf》第 14 章

3.3，YUV 转 RGB

常见的3种色彩模型

色彩模型	说明
RGB	发光体混色模型，RGB表示红、绿、蓝三个分量，RGB应用广泛：电子显示屏、投影仪、数码相机等，通常还会加一个透明度分量，即ARGB
CMYK	反光色彩模型，CMYK表示青色、洋红色、黄色、黑色，CMYK模式基于油墨的光吸收、反射特性，吸收白光中特定波长的光而反射其余的光的颜色，从而显示不同的颜色，主要应用：印刷
YUV	YUV色彩编码模型，Y表示亮度，U和V表示色度，色度：色调和饱和度，YUV编码可以降低图片数据的内存占用，提高数据处理效率，但是不能直接用于显示，还需要将其转换为RGB数据，才能够正常显示图像。主要应用：图片处理、压缩等

YUV主流的三种采样方式

YUV采样方式	说明
YCbCr(4:4:4（YUV444）	每个 RGB 像素占 3 个 YCbCr 字节
YCbCr(4:2:2（YUV422）	每个 RGB 像素占 2 个 YCbCr 字节
YCbCr(4:2:0（YUV420）	每个 RGB 像素占 1.5 个 YCbCr 字节

YUV 转 RGB 操作

STM32F7要实现YUV到RGB的转换，必须使用纯软件的方式来实现，比较耗时间。
而STM32H7可以通过DMA2D很方便的实现YUV到RGB的硬件转换

用DMA2D实现YUV转RGB

3.4，JPEG寄存器

3.5，硬件JPEG解码JPG/JPEG的简要步骤

1）初始化硬件 JPEG 内核
2）初始化硬件 JPEG 解码
3）配置硬件 JPEG 输入/输出 MDMA
4）编写相关中断服务函数，启动 MDMA
5）处理 JPEG 数据输出数据，执行 YUV→RGB 转换，并送 LCD 显示

3.6，JPEGCODEC 驱动代码

函数	作用
jpeg_in_dma_init()	JPEG硬件解码输入MDMA配置
jpeg_out_dma_init()	JPEG硬件解码输出MDMA配置
MDMA_IRQHandler()	MDMA中断服务函数
JPEG_IRQHandler()	JPEG解码中断服务函数
jpeg_core_init()	初始化硬件 JPEG内核
jpeg_core_destroy()	关闭硬件 JPEG内核,并释放内存
jpeg_decode_init()	初始化硬件 JPEG解码器
jpeg_in_dma_start()	启动 jpeg in mdma, 开始解码 JPEG
jpeg_out_dma_start()	启动 jpeg out mdma, 开始输出YUV数据
jpeg_dma_stop()	停止 JPEG MDMA解码过程

函数	作用
jpeg_in_dma_resume()	恢复MDMA IN过程
jpeg_out_dma_resume()	恢复MDMA OUT过程
jpeg_get_info()	获取图像信息
jpeg_get_quality()	得到JPEG图像质量
jpeg_dma2d_yuv2rgb_conversion()	将YUV数据转换成RGB数据

3.7，hjpgd.c和hjpgd.h驱动代码

函数	作用
jpeg_dma_in_callback()	JPEG输入数据流回调函数
jpeg_dma_out_callback()	JPEG输出数据流(YCBCR)回调函数
jpeg_endofcovert_callback()	JPEG整个文件解码完成回调函数
jpeg_hdrover_callback()	JPEG header解析成功回调函数
hjpgd_decode()	JPEG硬件解码图片

准备工作：需要准备一张TF卡，然后在TF卡根目录下新建一个PICTURE文件夹，并在里面存放bmp、jpg、jpeg或gif格式的图片文件

3.8，硬件JPEG解码实验演示

实验功能：在TFTLCD上循环显示支持的图片，通过KEY0和KEY1可以快速浏览下一张和上一张，KEY_UP按键用于暂停/继续播放，LED1用于指示当前是否处于暂停状态

当图片分辨率小于等于LCD分辨率时，JPEG解码速度变快很多

4，课堂总结（了解）