前言

GPU即图形处理器，是现代显卡的核心。在没有GPU的时代，所有图形的绘制都是由CPU来完成的，CPU需要计算图形的边界、颜色等数据，并且负责将数据写入显存。简单的图形还没有什么问题，但随着计算机的发展（尤其是游戏的发展），需要显示的图形图像越来越复杂，CPU也就越来越力不从心。所以后来GPU应运而生，将CPU从繁重的图形计算任务中拯救了出来，大大加速了图形的显示速度。

而单片机这边也有类似的发展历程。在早期的单片机使用场景中，极少有图形显示的需求。即使有，也只是简单的12864之类的显示设备，运算量不大，单片机的CPU可以很好的处理。但是随着嵌入式图形的发展，单片机需要承担的图形计算和显示任务越来越多，嵌入式系统的显示分辨率和色彩也一路飙升。慢慢地，单片机的CPU对这些计算就开始力不从心了。所以，自STM32F429开始，一个类似GPU的外设开始加入到STM32的单片机中，ST称之为Chrom-ART Accelerator，也叫DMA2D（本文将使用此名称）。DMA2D可以在很多2D绘图的场合提供加速，完美嵌合了现代显卡中“GPU”的功能。

虽然这个“GPU”只能提供2D加速，而且功能非常简单，与PC中的GPU不可同日而语。但是它已经可以满足大多数嵌入式开发中的图形显示加速需求，只要用好了DMA2D，我们在单片机上也可以做出流畅、华丽的UI效果。

本文将从实例出发，介绍DMA2D在嵌入式图形开发中的可以发挥的作用。目的是使读者能简单、快速地对DAM2D的建立最基本的概念并且学会最基本的用法。为了防止内容过于晦涩和难懂，本文不会对DMA2D的高级功能和特性进行深入地刨析（如详细介绍DMA2D的架构、全部的寄存器等等）。如果需要更加详细、专业地学习DAM2D，可以在阅读完本文后参考《STM32H743中文编程手册》。

阅读本文之前需要对STM32中的TFT液晶控制器（LTDC）和基本的图形知识（如帧缓冲framebuffer、像素、颜色格式等概念）有一定的了解。

另，除了ST之外，其他不少厂商生产的MCU中也存在类似功能的外设（如NXP在RT系列中设计的的PxP），不过这些不在本文的讨论范围内，有兴趣的朋友可以自行了解。

准备工作

硬件准备

可以使用任何的，带有DMA2D外设的STM32开发板来验证本文中的例子，如STM32F429，STM32F746，STM32H750等MCU的开发板。本文中使用的开发板是ART-Pi 。ART-Pi是由RT-Thread官方出品的开发板，采用了主频高达480MHz的STM32H750XB+32MB SDRAM的强悍配置。而且板载了调试器（ST-Link V2.1），使用起来非常方便，特别适合各种技术方案的验证，用来作为本文的硬件演示平台再合适不过了。

显示屏可以是任意的彩色TFT显示屏，推荐使用16位或24位颜色的RGB接口显示屏。本文中使用的是一块 3.5‘’ 的TFT液晶显示屏，接口为RGB666，分辨率为320x240（QVGA）。在LTDC中，配置使用的颜色格式为RGB565

开发环境

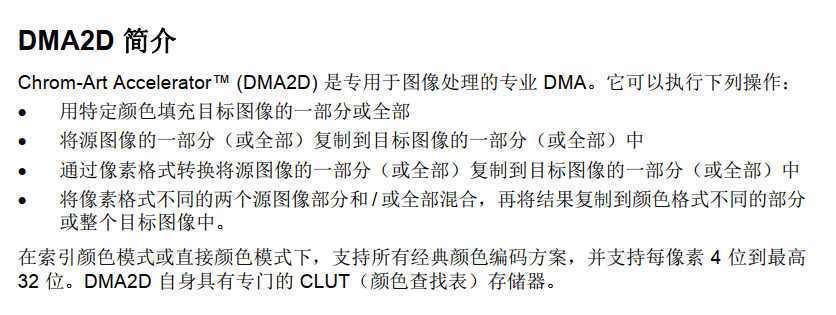
本文中介绍的内容和出现的代码可以在任何你喜欢的开发环境中使用，如RT-Thread Studio，MDK，IAR等。

开始本文的实验前你需要一个以framebuffer技术驱动LCD显示屏的基本工程。运行本文中所有的代码前都需要预先使能DMA2D。

使能DMA2D可以通过这个宏来实现（硬件初始化时使能一次即可）：

DMA2D的简介

我们先来看看ST是怎么描述DMA2D的：



**乍一看有点晦涩，但其实说白了就以下几个功能:**

1. 颜色填充（矩形区域）
2. 图像（内存）复制
3. 颜色格式转换（如YCbCr转RGB或RGB888转RGB565）
4. 透明度混合（Alpha Blend）

前两种都是针对内存的操作，后两个则是运算加速操作。其中，透明度混合、颜色格式转换可以和图像复制一起进行，这样就带来了较大的灵活性。

可以看到，ST对DMA2D的定位就像它的名字一样，是一个针对图像处理功能强化过的DMA。而在实际开发的过程中，我们会发现DMA2D的使用方式也非常类似传统的DMA控制器。在某些非图形处理场合，DMA2D甚至也可以代替传统的DMA来发挥作用。

需要注意的是，ST的不同产品线的DMA2D加速器是有微小区别的，比如STM32F4系列MCU的DMA2D就没有ARGB和AGBR颜色格式互转的功能，所以具体需要用到某个功能的时候，最好先查看编程手册看所需的功能是否被支持。

本文只介绍所有平台的DMA2D共有的功能