

# TK499 存储空间及 Bootloader 编写

## 1、 TK499 内存分配

TK499 内存地址从 0x0x70000000 起，总共 8MB；在系统启动时，固化在芯片内部的 ROM 及官方默认的 Bootloader 在启动时总共占用了最开头的 128KB，通常这 128KB 默认不使用，但是启动完进入主程序，用户仍然可以复用起来。官方默认的 bootloader 把余下的内存分为两大块，紧接着开头的 128KB 是用来存储代码，目前分配容量 2MB，剩下的(8MB-128KB-2MB)作为应用程序的运行内存。因为 Bootloader 用户可以自己编写，所以用户可以分配 1MB 作为存储代码，把剩下的 6.875MB(8MB-128KB-1MB)作为应用程序的运行内存也行。或者其它分配方式均可。

## 2、 TK499 启动中，FLASH 与内存的动作

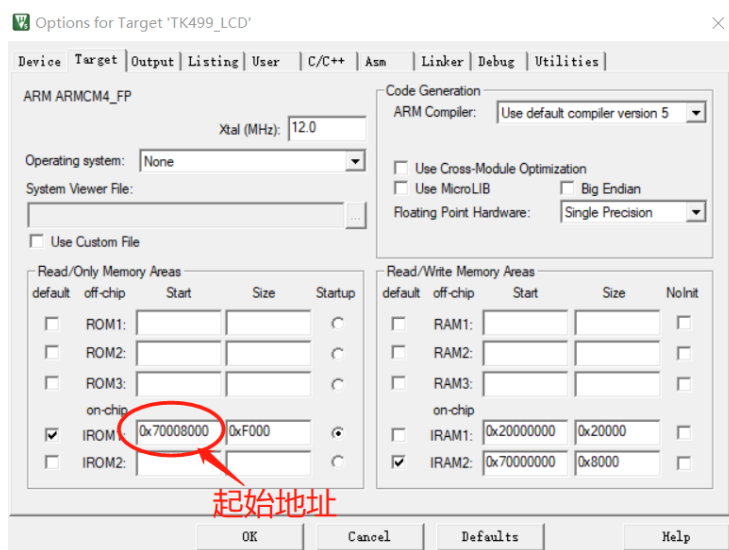
在芯片启动时，芯片内部固化的 ROM 负责启动 DMA，QSPI 模块，读取外部 FLASH（例如 W28Q128），如果有程序，则直接搬运到用户 bootloader 定义的内存地址上存放，搬完后就直接跳转到内存里运行。如果遇到 FLASH 中没有程序，那么芯片会再启动 USB 模块，同时进入了下载模式，用户可以用 USB 下载自己制作的 bootloader，也可以不制作 bootloader，直接下载应用程序（KEIL 设置 FLASH 地址为 0x70008000 及 TK499.H 里 #define T\_SDRAM\_BASE 0x70008000 即可）。

ROM 是从外部 SPI FLASH 的 0 址开始识别程序描述信息，然后按信息指示搬运相应长度代码到内存 0x70008000 的地址中运行。因为 ROM 是在芯片的金属层里制作的，所以速度很慢，48MHz 的时钟，如果你把很大的程序，例如 2MB，ROM 搬运就会很慢。为了更快的速度，于是就出现了 Bootloader 这个产物。首先让 ROM 搬一个十几 K 的 Bootloader 到内存里，然后再运行 Bootloader 到三四倍于 ROM 的速度，搬运大程序到内存里运行。所以如果你的程序不大，或者对启动速度没在太高的要求，可以不要 Bootloader。当然，bootloader 可以做出更丰富的功能，例如远程升级。

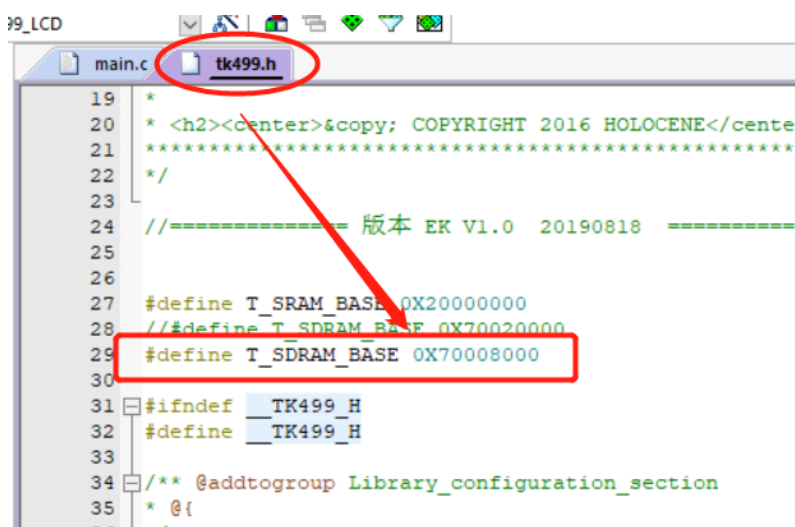
## 3、 Bootloader 制作的关键信息

制作 bootloader，主要了解程序头的信息描述及关键地址。主要涉及三个地址，①程序存在 SPI FLASH 哪个地址；②程序要搬动到内存哪个地址运行；③ bootloader 及应用程序运行的内存地址。

首先，ROM 执行的搬运工作是从 SPI FLASH 的 0 地址开始。ROM 运行时，首先通过 QSPI 读取 SPI FLASH 的 0 地址的前 24 个字节；其中第一个 32 位（4 个字节）解释出程序存放在 RAM 中的起始地址，与 KEIL 设置中一致，第二个 32 位是存放起始地址的反码，第三个 32 位是程序的长度，第四个是长度的反码；第五个 32 位是常数 0x12346789，第六个是常数 0x12346789 的反码；ROM 解释出这些信息后，进行正反码校对，如果确认匹配，说明有程序，如果发现不匹配，说明没有程序，那就得进入下载模式了。当检测到是匹配的情况下，ROM 会把跟据解释出来的长度及起始地址，从（0 地址+1KB）的偏移地址搬运指定长度的程序到 SDRAM 中指定的起始地址，搬完跳转运行。目前样例的 bootloader 起始地址是 0X70008000，如下图：



与程序中，TK499.H 中地址一样：



然后，在 bootloader 中，定义你的应用程序在 SPI FLASH 中的起始地址，及程序要运行时，搬到内存的地址；

样例 bootloader 定义用户应用程序描述信息在 SPI FLASH 的 0x10000 地址，同样是偏移 1KB 是程序主体，详见 SDIO.C 的第 110 行注释；

最后，定义用户应用程序在 SDRAM 中存放地址




Bootloader 到此就编写完成，记得以后你的应用程序要按照你的 Bootloader 定义的

SDRAM 地起始地址来设置。KEIL 设置必须与你 Bootloader 预先设置的一样才能运行。

#### 4、不用 bootloader 直接写入主程序的办法

依赖 bootloader 的程序，先烧 bootloader 才能烧主程序，某些场合下还是有点麻烦，不过主程序无 bootloader 也是可以运行。其实 bootloader 就是一个主程序，只不过这是一个专用于下载程序的主程序而矣。Bootloader 的存在，只是在于芯片启动时更快搬运程序到内部 SDRAM 中运行，仅在启动时快一丢丢，并且 bootloader 也占 64K 空间。无需依赖 bootloader，直接上主程序的算法基本与有 bootloader 的一样。首先程序在 FLASH 上存储是从 0 地址开始的，其次就是这几行代码有区别：



```
160 bin_size -= 512;
161 }
162 while(bin_size > 511);
163 if(bin_size)
164 {
165     f_read(&fsrc, Buf, bin_size, &br);
166     QFLASH_FunProgram(addr, (unsigned char *) (Buf), bin_size);
167 }
168 //===== 写入程序头 =====//
169 *(u32*) (Buf) = 0x70008000 ;//程序存放在RAM中的起始地址，与KEIL设置中一致
170 *(u32*) (Buf+4) = ~0x70008000 ;
171 *(u32*) (Buf+8) = fsrc.obj.objsize;//程序长度
172 *(u32*) (Buf+12) = ~(fsrc.obj.objsize);
173 *(u32*) (Buf+16) = 0x12346789 ;
174 *(u32*) (Buf+20) = ~0x12346789 ;
175 QSPI_PageXbytesProgram_Standard_NoCheck(QSPI_FLASH_SAVE_APP_ADD, (unsigned char
176
177 f_close(&fsrc);//升级完成，关闭文件
178 f_mount (NULL, "0:", 1);//注销文件系统
179
180 printf(" Update OK! \r\n");
181 return 1;
```

## TKM32F499 官方 Bootloader 说明

如果没有什么特殊需要，也可以直接用好钜润官方的 Bootloader.官方的 Bootloader 运行在 192M 下，只支持 USB 下载模式，支持下载 BIN 程序、TXT、JPG、GIF、XBF 及 HJR 等几种格式的文件。

芯片第一次使用时，什么程序也没有，只有固化在芯片内部的 ROM 可执行。芯片启动时，首先运行 ROM。ROM 就自动执行芯片初始化，启动相关时钟，启动 QSPI、DMA、UART 及 USB，所以在芯片第一次运行时，已经支持 USB 下载模式。然后 ROM 会检测外部 SPI FLASH(例如 W25Q128)的 0 地址程序描述信息，如果发现没有没有相关信息，那直接跳到下载模式。Bootloader 会把程序下载到外部 FLASH 中的 0X10000+0X400 处，然后下载完程序，会把程序运行在 SDRAM 的地址及程序长度等信息写入 0X10000 处。Bootloader 默认预留 2MB 空间用作存储程序，所以 SPI FLASH 中 0X210000 之后的地址用作存储数据。如果你把图片字库数据下载进去，Bootloader 会把相关数据顺序存放于 0X210000 之后的地址。如果你要获取你的数据首地址，可以用 get\_file\_address\_NOR\_FLASH 函数获取.在上述下载程序、数据等，有相关重要的信息都会通过串口 1 打印出来，可以用 460800 波特率去监听。

**注意：**由于目前版本 Bootloader 不知道你外挂了什么容量的 FLASH 上去，所以是否下载满了单片机不清楚，需要用户自己停止丢文件进去。你也可以用串口来监控。或者你先全选所有要丢进去的文件，在电脑上显示不超容量再丢进去。另外，如果丢多了，或者丢重复了，可以丢这个文件“DEL56789.txt”进去，把数据清空。或者你重新下载一次 Bootloader，也会删除用户数据。

所有单次丢进去的文件，不能超过 7.8MB，因为这个弹出的 U 盘是用单片机内存虚拟的。总共有 8MB 的内存，ROM 及 Bootloader 占用了 128K，文件系统也用去一点，所最

大单次文件只能到达 7.8MB，如果你的文件总大小超过 7.8MB，请分多次丢入。本 Bootloader 存放文件功能最大只支持 16MB，再大得自己研究一下，要识别文件大于 16MB 后，把地址扩展到 4 字节才行，本官方 Bootloader 就不搞那么复杂。

官方 Bootloader 要用的文件都在“TKM32F499 评估板”资料包里，其中“DEL56789.txt”在“汉英字库，数码管字体.rar”中，get\_file\_address\_NOR\_FLASH 函数应用样例之一在 16 号例程中“16、TK499\_LCD\_TK035F5589\_FLASH 字库图片\_自动 get 地址.rar”。另外，关于 FLASH 读写，相关函数都在 qspi\_fun.c 及 QSPI\_REG.c 中，一共两套函数库，其中 qspi\_fun.c 主要是 DMA 写法，QSPI\_REG.c 则是侧重于轮询法。

运行 Bootloader，仅需芯片最小系统即可。主要包括 12M 晶振、USB 接口、SPI FLASH（容量建议 W25Q16 以上），下载按键（PA1 与 PA13，按键共阳）单电源 3.3V 供电。