2 OMAP3530的启动过程

有关OMAP3530的启动过程最详细的描述，在文档《OMAP35x Applications Processor Technical Reference Manual》（文档号SPRUF98）（这个文档会被经常用到，以后简称《OMAP35x TRM》）的25章有详细的介绍。

2.1 OMAP3530的7个用于设置启动的外部引脚

OMAP3530支持很多种方式的启动，并且可以像PC一样的设置启动顺序，它是通过7个引脚（boot0~boot6）电平来控制的。其中boot6用于选择时钟，当boot6接通过上拉电阻接高电平时，使用TPS659xx提供的方波作为系统的时钟源；如果boot6接低电平，就使用外部晶振作为时钟源。因为我的板子是有TPS65930的，因此boot6=0。boot0~boot5用于选择通过哪些方式启动，并且这些启动方式的顺序，具体参考表25-3（3363页）以及表25-4（3364页）。由于我的板子上只有Nand Flash和SD卡，因此，**使用boot5~0 = 0b001111这种方式，检测顺序是NAND FLASH > USB > UART3 > MMC1或boot5~0=0b101111，检测顺序是USB > UART3 > MMC1 > NAND FLASH**。值得说明的是，USB和UART3的方式是属于外设启动的方式，只有当这些接口接有可用于启动的设备时才会有效，相关内容可以参考《OMAP35x TRM》的“25.4.5 Peripheral Booting”，但是目前还用不到。

2.2 Nand Flash启动

Nand Flash属于Non-XIP（eXecute In Place）类的Flash，即不可直接在片内运行程序，因此需要由OMAP3530的内部固件——ROM Code把Nand Flash内的程序读取出来，然后再运行（“1.4 OMAP3530的内存映射”中提到的内部80KB的ROM就是用来存储这个ROM Code的）。完整的信息在《OMAP35x TRM》的3391页，25.4.7.4节。

2.2.1 Nand Flash的选型

ROM Code对Nand Flash是有要求的，《OMAP35x TRM》的表25-31（3391页）说明了ROM CODE访问Nand Flash的时序参数（GPMC的时钟频率会被自动设定为48MHz），设计时要选择符合要求的Nand Flash。不过，按照经验设计，一般选用**Micron的MT29C2G48MAKLCJA-6 IT（这个芯片其实是LPDRAM和Nand的二合一芯片，同时有多种容量可供选择，数据手册都可以参考这一个，我的板子情况是256MB Nand + 256MB LPDRAM）**，因此这两个表就可以不用太关心了。

2.2.2 关于美光的Nand+LPDRAM

美光的这个芯片上不会印有型号信息，只是有一个“FBGA Code”，去美光的网站 http://www.micron.com/support/fbga.html 输入这个代码可以查到芯片的型号，然后根据这个型号可以搜索到数据手册，但是要下载数据手册还得签署一个什么保密协议，通过审查后才会用电子邮件的方式通知下载，真够麻烦的！

2.2.3 Nand Flash的连接要求

（1）Nand Flash必须接在GPMC的CS0；

（2）GPMC_Wait0引脚要接在Nand Flash的BUSY输出引脚，即R/#B引脚；

2.2.4 Nand Flash启动过程

（1）ROM Code先将GPMC设定为8位，非同步模式，用于读取Nand Flash的ID信息，根

据读出的“Device ID”（Byte 1）与《OMAP35x TRM》表25-32（3392页）对照，确定容量（注意表中的单位是bit不是Byte）、数据线宽度、和页大小（单位是K Byte）。例如，我的板子上的Nand Flash的Device ID是0xBA，因此，查表得到数据线宽度是16位，页大小是2048KB；

（2）当Nand的厂家不同，可能查表得到的“页大小”参数是错误的，因此ROM Code会通过读出的第四字节（Byte 3）确认页大小并取得块大小（表25-33，3393页）。但是这个过程仅仅发生在Nand Flash容量大于等于2Gbit时，对于小于2Gbit的Nand Flash，当页为512Byte，块为32KB；当页为2KB时，块为128KB。我板子的请款是块大小：128KB，页大小：2KB；

（3）上述过程的流程图在3395页图25-16，之后会读取ID2来确定更详细的Nand Flash参数，流程图在3396页图25-17，获得的参数在表25-34（3397页）；

（4）读取第一页和第二页的spare sectors来确定Nand Flash坏块，流程图在3398页图25-18；

（5）之后ROM Code从Nand Flash最开始读取启动镜像文件，当然对镜像的格式有特殊要求，将在“2.4 启动镜像文件的格式”说明。

2.3 SD卡启动

和Nand Flash类似，程序也不可以在SD卡上直接运行，必须读出来才能运行。ROM Code可以读取MMC/SD卡上的启动文件，但是对MMC/SD卡片上的分区、文件系统等有特殊要求。具体过程的详细描述在《OMAP35x TRM》的“25.4.7.6 MMC/SD Cards”。

2.3.1 ROM Code操作SD卡的参数

（1）初始化卡片的频率：400kHz；

（2）数据传送频率：20MHz；

（3）支持大容量（>2GB）的SDHC卡。

2.3.2 SD卡启动的过程

SD卡启动的流程图在《OMAP35x TRM》的图25-24（3405页），其中每一步的流程在后面都有详细介绍。下面简单总结下流程：

（1）ROM Code先通过I2C配置TPS659X0为MMC1配电，MMC1单元需要两个电压，一个是模块工作需要电源电压VDDS_VMMC1，另外一个是各信号线工作电平的参考电压VDDS_VSIM（这两个引脚如何与TPS65930连接，请看《Powering OMAP™3 With TPS65930/20: Design-In Guide》（文档号：SWCU059）的第8页的图2；如果使用TPS65950，请看《采用TPS65950 为OMAP™3 供电：应用设计指南》（文档号：ZHCU013）第9页的图2），ROM Code先将他们配置成3.0V。

（2）通过一系列命令初始化SD卡，流程图为图25-27（3408页）：

- 先发送CMD1，对于MMC卡，将返回应答，对于SD卡将超时；
- 如果上一步超时，发送CMD55和ACMD41；

- 如果上一步还是超时，将判断为“没有卡片插入”。

以上几部也回答了上一章留下的一个问题：**“在SD卡启动过程中，会不会通过TPS659xx来检查卡片是否插入呢？如果是这样，那使用小型的microSD卡槽（只有8个引脚）又该如何处理？”**SD卡启动过程中，并不使用CD引脚来检查卡片是否插入，而是通过图25-27的步骤，因此心使用小型的microSD不会有什么问题。

（3）之后是读取启动镜像文件，这里又可以使用两种模式来进行操作，其中“Raw模式”不使用，具体介绍在《OMAP35x TRM》3408页。对于“文件系统模式”，ROM Code对SD卡内的文件系统是有特殊要求的：首先，SD卡要有MBR（Master Boot Record Structure）。对于小于等于2GB的SD卡，直接格式化后通常没有MBR，0扇区直接就为Boot Sector，这样是不能用于启动的。其次，SD卡的一个主分区必须为FAT文件系统，通常把第一主分区格式化成FAT32就可以了。最后，**启动文件一定要放在上述FAT文件系统的根目录下，并且名字一定要为MLO**。

2.4 启动镜像文件的格式

OMAP3530要求启动镜像文件有文件头，文件头数据结构又分为多种，具体参考《OMAP35x TRM》的“25.4.8 Image Format”（3415页）。这里只用到最简单的一种（25.4.8.3 Image Format for GP Devices），即在镜像文件的最前面增加8个字节，前4个字节表示整个文件的大小（不包括文件头），个人猜测这个信息用来告诉ROM Code需要读取多少字节；后4个字节表示镜像文件要加载的地址，这里需要注意，这个地址需要放在OMAP3530的内部SRAM内，ROM Code把启动镜像文件加载到这个地址后会跳转到这个地址开始执行。

2.5 ROM Code提供的内存映射

（1）ROM Code会把自己映射到0x14000的地址处，具体的说明在《OMAP35x TRM》3369页的“25.4.2.1 ROM Memory Map”。在0x14000处存放的是中断向量，中断向量的排列顺序和传统的ARM体系结构一样，**只不过他们不是从0x0开始的**。

（2）ROM Code会把片内64KB的SRAM分割成好几个区域，具体见《OMAP35x TRM》3370页的“25.4.2.2 RAM Memory Map”。

- 0x4020 0000~0x4020 EFFF：为镜像文件存放的空间，启动镜像文件应该被加载到这个地址，**启动镜像文件的文件头的后4个字节所指明的地址应该在此区域内**。
- 0x4020 F000~0x4020 FCAF：通用的堆栈空间，启动程序中的汇编程序部分应该把堆栈设定到此，这样才能运行C语言的程序。（当然，这个不是必须的，也可以先把外部的LPDRAM初始化，然后再把堆栈堆栈放进去。）
- 0x4020 FCB0~0x4020 FFAF：保留空间。
- 0x4020 FFB0~0x4020 FFC7：ROM Code的Tracing Data，具体内容可以查看《OMAP35x TRM》

3422 页“25.4.9 Tracing”。

- 0x4020 FFC8~0x4020 FFFF：RAW 内的中断向量，要其中注意并没有 Reset 中断向量。

2.6 ROM Code 提供的中断机制

（1）中断向量是从 0x14000 开始的，依次为“Reset”、“Undef”、“SWI”、“Prefetch Abort”、“Data Abort”、“保留”、“IRQ”、“FIQ”。例如：当 IRQ 中断出现，程序会**自动**跳转至 0x14018（而标准的 ARM 体系结构则是自动跳至 0x18）。

（2）当“Reset”中断出现，直接执行 ROM Code；当其他中断出现，会直接跳至内部 SRAM 的中断向量处（《OMAP35x TRM》3369 页表 25-7），内部 SRAM 中断向量从 0x4020 FFC8 开始，依次为“Undef”、“SWI”、“Prefetch Abort”、“Data Abort”、“保留”、“IRQ”、“FIQ”。例如：当中断为 IRQ 时，在 0x14018 中会存有一条“PC=0x4020FFDC”的指令，让程序跳转至 0x4020FFDC。

（3）ROM Code 会**自动**在上述 SRAM 中断向量中写入相同的指令，功能是**把“SRAM 中断向量地址+0x1C”处存放的 32 位的无符号数（其实是个指针）放入 PC，以实现跳转**。例如，在 IRQ 的 SRAM 中断向量（0x4020FFDC）会存有这样一条指令：“PC=[0x4020FFF8]”（注意中括号，0x4020FFF8=0x4020FFDC+0x1C）。因此，**只要在[0x4020 FFF8]中存放我们自己的 IRQ 中断处理函数的指针（函数名），程序就能跳转至我们自己的函数去执行了**。其他中断的响应过程同理。

2.7 疑问

（1）中断向量是从 0x14000 开始的（低中断向量），这种情况是不是仅仅是在 Nand Flash 或 SD 卡启动时才适用呢？提出这个问题的原因有二：其一，如果使用 XIP 的 Nor Flash，该 Flash 要接在 GPMC_0，其范围可能覆盖掉 0x14000；其二，在《OMAP35x TRM》的 1049 页“Note”中所描述的中断过程是完全符合 ARM 体系结构的，即 IRQ 中断向量是 0x18，这里和“25 Initialization”这一章描述的（即上面的总结）是矛盾的（IRQ 中断向量是 0x14018），但是通过实验发现，25 章描述的是正确的，因此猜测可能是启动方式不同造成的。但是我的实验板上没有 Nor Flash，因此无法做实验来验证。