第29卷 第1期
2006年3月

电子器件
Chinese Journal Of Electron Devices

Vol.29 No.1
Mar.2006

Design and Realization of Secondary BOOTLOADER for TMS320DM642 DSP

HE Zheng-jun, ZHU Shan-an

(*Institute of Electric Engineering Zhejiang University, Hangzhou* 310027, *China*)

Abstract: If the size of an application programme for a system based on TMS320DM642 DSP is greater than 1 K bytes, senondary bootloader is needed to copy the application from FLASH to RAM once the power is on and the system is resetted. The senondary bootloader could be developed based on its own boot mechanism.. Three GPIO pins are used to divide the whole space of the FLASH to pages, so the address can be remapped.. The boot table which will be necessary for booting code could be generated automatically by hex conversion.

Keywords: TMS320DM642; boot table; secondary bootloader; address remapped; hex conversion utility

EEACC: 1265

TMS320DM642 DSP 二级引导程序的设计与实现

何正军，朱善安

（浙江大学 电气工程学院，杭州 310027）

摘 要：在基于 TMS320DM642 DSP 的应用系统中，当应用程序代码量超过 1 K byte 时，上电复位后，需要通过二级引导程序将存储在 FLASH 中的应用程序代码引导到 DSP 内部 RAM。根据二级引导程序的引导机制实现二级引导程序的开发。并使用三个通用 I/O 口，对整个 FLASH 空间进行分页，实现了地址的重映射。利用 16 进制转换工具自动生成可以用来帮助二级引导装载程序引导代码的引导表。

关键词：TMS320DM642；引导表；二级引导装载程序；地址重映射；16 进制转换工具

中图分类号：TN911　　**文献标识码**：A　　**文章编号**：1005-9490(2006)01-0260-04

DSP 芯片已成为人们日益关注并得到迅速地发展的一种集成电路，得到了越来越广泛的应用。TI 公司最新推出的 TMS320DM642 是一款高性能的数字多媒体处理器，具有 2 级存储器和高速缓冲器，以及超长指令字结构。

FLASH 则是基于 DSP 应用系统一个必不可少的基本配置，主要用来存放用户应用程序。当系统的应用程序完成调试，需要脱机运行时，烧写在 FLASH 中的应用程序需要被自动加载到 DSP 的 RAM 中高速地运行，这就是引导加载（bootloader）过程。正由于 DM642 引导机制的一些特点以及基于 DM642 的应用系统的复杂性，应用程序一般比较庞大，仅通过片上的自动引导机制（仅能引导 1K 代码[2]）并不能完成整个应用程序的移植，所以需要开发二级引导装载程序（secondary bootloader）。

1 应用系统采用的 ROOM BOOT

在 DSP 的应用系统中，采用得最普遍的引导的方式就是 ROM 引导。当 ROM 引导被确认为目标系统的引导方式时，一旦上电复位后，基于 C621X/C671X/C64X 系列的 DSP 的目标系统将通过 EDMA 把位于 CE1 地址空间开始的 1 K byte 的代码

收稿日期：2005-05-24

作者简介：何正军（1980-），男，工学硕士在读，从事 DSP 的嵌入式应用研究，robert. ap@sohu. com；

朱善安（1952-），男，工学博士，教授，博导，从事预测自适应控制理论与工业应用，PID 自整定理论与工业应用，PC 机的智能控制系统，信息传输与自动化系统等研究，zsa@zju. edu. cn

自动移植到内部从地址 0 开始的存储器(RAM)中。

DSP 应用程序的代码的大小不限于 1 K byte。基于 C621X/C671X/C64X 系列的 DSP 的目标系统应用程序代码通常远远超过 1 Kbyte，所以需要开发另外一个引导程序用来引导片上自动引导机制未能引导的代码，并且我们把这个引导程序称作二级引导装载程序(secondary bootloader)[1]。通常情况下，二级引导装载程序将被放置在只读存储器的起始地址处，一旦 DSP 上电复位，二级引导程序就会通过片上自动加载机制加载到 RAM 地址 0 处，且此时 CPU 复位，开始执行二级引导程序，当二级引导程序完成应用程序的引导任务后，CPU 跳转到 C_int00 处开始执行。C620X/C670X 系列的 DSP 能够通过 DAM 在上电复位后自动引导 64 K byte 的代码，而 C5000 系列的 DSP 本身片内就有固化了 BOOTLOADER 的掩模 ROM。DM642 上电复位时只能引导 1 K byte 大小的代码，但只要借助于二级引导程序，可使基于 DM642 的应用系统的引导将更具灵活性，同时也能节省对内部 ROM 掩模的需要，降低了电路设计的成本。

我们的应用场合是在基于 DM642 的硬件平台上进行图像的采集与压缩处理，网络传输，以及本地视频回放，应用程序的代码远远超过 1 K byte，所以需要开发二级引导装载程序。

2　系统 DM642 与 FLASH 硬件连接

本应用系统采用外部 ROM BOOT 的方式引导应用程序。DM642 与 FLASH 的硬件连接见图 1。FLASH 采用的是 AMD 公司的 Am29LV640MT。它支持 4M×16 的 byte 操作或 8M×8 的 byte 操作，由引脚 BYTE# 决定。当 BYTE# 置高时，进行的是 16 bit 的字操作；当 BYTE# 置低时，进行的是 8 bit 的 byte 操作，它可寻址 8 M byte，此时，DQ15/A-1 用来作为地址线的最低位，再加上原来 A0－A21 的 22 根地址线，就可以寻址 8 Mbyte 地址空间[2]。

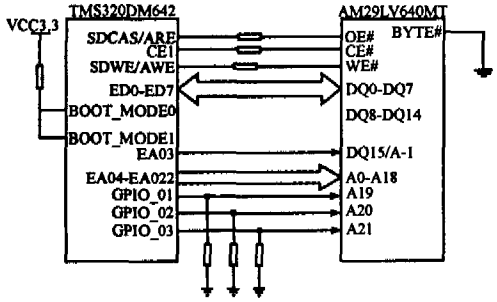

图 1　DM642 与 FLASH 的硬件连接图

本应用系统采用 byte 操作模式，因此将引脚 BYTE# 置低，DM642 可以对 FLASH 寻址 8 M byte。但是，外部存储器接口(EMIF)仅有 20 根地址线，所以一个 CE1 空间最大的寻址能力就是 1 M byte。为解决能对 FLASH 寻址 8 Mbyte 的能力，采用了对 FLASH 进行分页技术，将整个 FLASH 空间分成 8 页，每页 1 M byte，共 8 M byte。也就是将 FLASH 的高三位地址线与 DM642 的三个通用 I/O(GPIO)引脚连接，组合通用 I/O 的高低输出，激活相应的 8 个页面，当 GPIO_01 、GPIO_02 、GPIO_03 三个引脚输出值为 0、0、0 时，激活第 0 页；为 1、0、0 时，则激活第 1 页，依次类推。同时三个通用 I/O 口通过下拉电阻将其拉低，以保证在 DSP 刚启动未对通用 I/O 口初始化时，片上自动引导机制从 FLASH 的代码引导是从第 0 页开始的。

因此，所有 8 个页面均映射到 DM642 的相同的地址空间 0x90000000 ～ 0x900FFFFF（由于 FLASH 通过 CE1 # 选通，而 CE1 的地址空间从 0x90000000 开始[3]），对 FLASH 进行操作时，通过 EMIF 输入给 FLASH 的地址信号，只是 FLASH 中的每一页的页内相对地址 0x00000～0xFFFFF，这样就实现了地址的重映射。当二级引导程序在引导应用程序时，每当对 FLASH 的操作到达页的边界，而且需要对下一页继续操作时，就需要重新组合 GPIO 的三个引脚的输出，用来激活下一页。

3　引导表

二级引导程序的引导需要借助于引导表。引导表包含了引导结束后程序的入口地址，每个数据块的大小，目标地址和数据块的数据以及引导结束判断标志。引导表的形式如表 1 所示。

表 1　引导表

程序入口地址
数据块 1 大小
数据块 1 目的地址
数据块 1 数据
⋮
数据块 N 大小
数据块 N 目的地址
数据块 N 数据
0x00000000

4　二级引导程序的编写

当完成了项目的编译和连接后，就需要开发二级引导程序了。因为在引导阶段，C 运行环境未得到初始化，同时代码移植工作的高效与快速也是非常必要的，所以二级引导程序需要用汇编语言来开

发。开发二级引导程序需要完成以下几项工作：

① 配置 EMIF，用来访问外部存储器；② 将 ROM 中被初始化后的块移植到运行地址处；③ 调用_c_int00（ ）。

二级引导程序首先对 EMIF 进行配置，然后进行代码的移植，将 FLASH 中第 0 页 1 K byte 之后的应用程序代码引导到运行地址处。在引导应用程序代码的过程中，最先获得的是移植完成后的程序入口地址，并存放在寄存器中；接着按照每个数据块的格式，依次获得数据块的 byte 数和目标地址，并根据这两个信息完成数据块中数据的移植工作。这样依次重复，完成所有数据块的移植。在进行移植的过程中，需要不断地判断 FLASH 地址是否到达页内的末地址，当到达页内末尾时，就需要通过组合 GPIO 输出激活下一页。当读取到需要被引导的数据块 byte 数为 0 时，则表示引导结束，调用_c_int00（ ）。如图 2 所示。

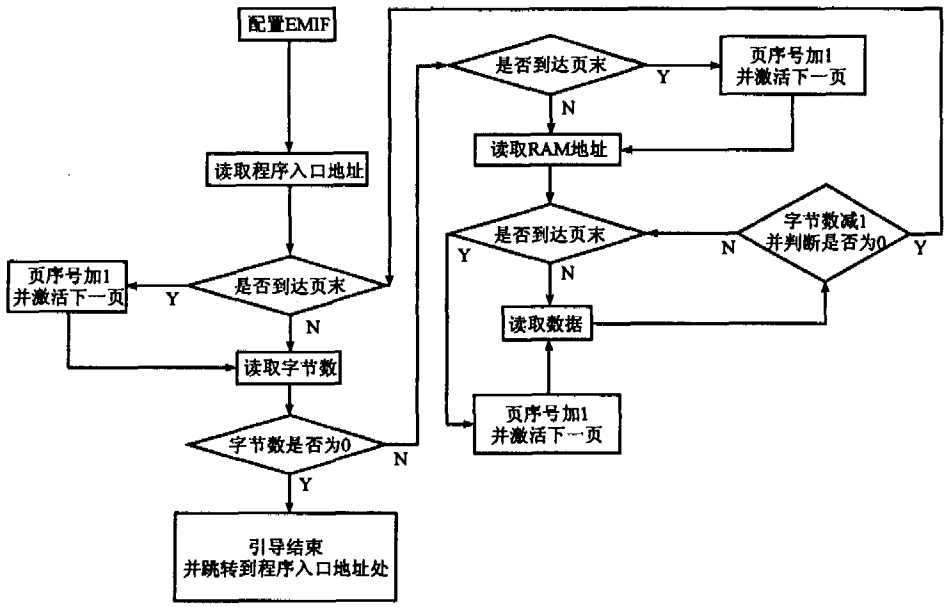

图 2 二级引导程序工作流程图

此外，我们还需要考虑另外一个问题，因为每个数据块开头在 FLASH 中的存放是 32 bit 字对齐的，而我们的系统中又是对 FLASH 进行 byte 操作的，当对当前数据块的移植完成后，指向的 byte 如果不是字的开始处，需要直接跳转到下一个字，同时这个字也应该是下一个数据快的第一个字。引导程序中应有相应的处理程序，如图 3 所示（图中的 a_3 为二级引导程序中指向 FLASH 被操作 byte 的指针变量，并且数据在 FLASH 中是按小端格式进行存储的）。

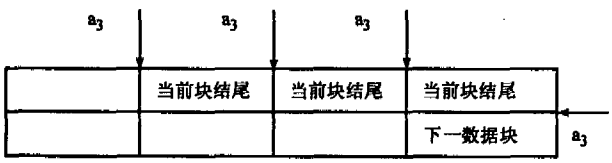

图 3 数据块引导结束指针操作示意图

5 16 进制转换生成引导获得. hex 文件

可以使用 16 进制转换工具自动生成引导表，但在使用该工具之前，需要将二级引导程序加入到项目中，并且重新编译，同时改变用户定义的连接命令文件，更新后的连接命令文件需要包含如下的数据块定义：

```
-l EXAMP. cmd
SECTIONS{
. boot_load :>BOOT_RAM }
```

引导表将被放置在 FLASH 1K 之后的开始地址处，故在二级引导程序中需要定义引导表的起始地址，如下：

```
SND_COPY_TABLE  . equ  0x90000400
```

表 2 16 进制转换工具 boot 相关选项说明

选项	描述
-boot	将所有的块转化为引导表的形式
-bootorg value	设置引导表的源地址
-bootsection sectname value	定义 sectname 二级引导程序的程序名；而 value 为放置二级引导程序的地址
-e value	设置引导完成后开始执行的地址。被设置的值可以是地址也可以是一个全局符号。

建立引导表的步骤：

第一步：连接文件。引导表中的每一个块对应于 COFF 文件中的初始化块。必须连接到应用程序，这样二级引导程序就能从引导表中读取到应用程序，完成引导的任务，

第二步：定义引导表中的块。使用-boot 选项将使引导表包括所有的块，

第三步：设置引导表在 ROM 中的地址。使用-bootorg 选项设置引导表的在 ROM 中的起始地址，

第四步：设置引导装载特定程序。设置入口地址，如果－e 选项不使用，则入口地址将被默认为 COFF 目标文件中的入口地址，

第五步：描述引导程序。使用-bootsection 选项来设置一个 COFF 块用来包含引导程序，并使引导程序不包含在引导表中。同时，这个选项也设置了引导程序在 ROM 中的放置位置。

第六步：描述系统存储器的配置[4]。

根据以上步骤，给出一个使用 16 进制转换工具建立引导表的实例：

建立 CMD 文件，设置 hex6x 的程序参数，以 EXAMP. cmd 为例。

```
EXAMP. out  / * Input file * /
-a  /* Select ASCII hex file as the output file * /
-memwidth 8  /* Generate output for an 8-bit FLASH device * /
-boot  /* Create a boot table * /
-bootorg 0x9000400  / * Place the boot table at 0x90004000 * /
-bootsection . boot_load 0x90000000 / * Place the boot loader at the beginning * /
-map EXAMP. map  /* Generate a map file * /
ROMS
{ FLASH: org = 0x90000000, len = 0x80000, romwidth = 8,files ={EXAMP. hex} }
```

注意在 ROM 大括号中的定义了文件的长度，0x80000 为 512k bytes，如果文件较大，长度也应改大。

① 建立. bat 文件，命令如下：

```
hex6x EXAMP. cmd
```

② 运行 bat 文件，执行文件转换操作，生成两个文件，EXAMP. hex 和 EXAMP. map，其中 EXAMP. hex 就是要烧写到 FLASH 中的文件，可以通过 FLASH 烧写程序将该文件烧写到 FLASH 中。

6　结束语

本文介绍了基于 TMS320DM642 DSP 的应用系统中二级引导程序的开发，为采用此款 DSP 的应用系统引导 FLASH 中的代码提供了灵活性，为开发基于像 DM642 片上自动引导机制引导的代码数量有限，而应用程序又比较庞大的产品提供了一项必须具备的技术。

参考文献：

[1] Creating a Second-Level Bootloader for FLASH Bootloading on TMS320C6000 Platform With Code Composer Studio [S]. Texas Instruments，2004.

[2] Data sheet for Am29LV640MT[S]. AMD 2004.

[3] TMS320DM642 Video/Imaging Fixed-Point Digital Signal Processor[S]. Texas Instruments，2004.

[4] TMS320C6000 Assembly Language Tools User's Guide[S]. Texas Instruments，2004.

[5] 赵训成，基于 TMS320C6200 系列 DSP 芯片的应用与开发[ ]