TMS320C6474 二次启动的实现

成丹
北京邮电大学信息与通信工程系，北京（100876）
E-mail:chengdan1985@gmail.com

摘　要：本文概要的介绍了三核 DSP TMS320C6474 的启动原理和启动过程，重点介绍了主 I2C 启动模式，以及基于这种方式实现 DSP 的二次启动，解决了启动代码过大导致无法加载以及启动速度慢的问题。并且利用二次启动实现了多核启动，使得 TMS320C6474 上电实现多核多任务的自动运行。在实际系统中，利用二次启动可丰富目标板的启动方式，实现目标系统工作模式切换等功能。

关键词：DSP；TMS320C6474；二次启动
中图分类号：TP274

1 引言

TMS320C6474 是 TI 推出的一款高性能的多核 DSP。它在单一的裸片上集成了三个 1GHz 的 TMS320C64X+™ 内核，实现了 3GHz 的原始 DSP 性能，能够执行多软件应用等高强度，高性能任务。

TMS320C6474 的软件开发是基于 TI 的 CCS(Code Composer Studio)集成开发环境(IDE)。CCS 扩展了基本的代码产生工具，集成了调试和实时分析功能[1]。在 CCS 下，用户利用仿真器和 JTAG 口使得 PC 与目标系统连接。用户在 CCS 下编写调试代码，编译链接生成可执行文件(.out 文件)，然后下载到目标系统的 DSP 上运行。当目标系统商品化，就要脱离 PC，如果按照通常的系统运行方式，程序在外部的非易失性存储设备内运行，将直接导致程序运行效率非常低，高速的 CPU 大部分时间耗在读写访问的等待时间上。传统的程序运行方式不能满足现在复杂实时性高的系统需用。目前普遍采用的运行方式是 boot 启动模式。程序存储在外部非易失性存储设备里，CPU 复位后首先把系统程序加载到内部 RAM 或外部高速 RAM，DRAM 中，然后系统程序在 RAM 中运行[2]。

2 TMS320C6474 启动原理

2.1 TMS320C6474 启动过程

TMS320C6474 从上电到程序开始运行的这段过程叫做启动过程(Boot),在这段时间里，固化在 TMS320C6474 ROM 中的一小段代码（Bootloader）自动运行，它会根据用户选择的启动方式的不同，先完成不同的初始化工作，之后从外部的存储器件中搬移启动数据或者接收启动数据到运行程序的 RAM 中，自动跳转到程序的入口地址，运行程序。TMS320C6474 支持多种启动方式，主 I2C 启动模式，从 I2C 启动模式，EMAC 启动模式，SRIO 启动模式。采用哪种启动方式由复位或上电采样 BOOTMODE[3:0]管脚来决定[3]。

与其他 DSP 的主要区别是 TMS320C6474 包含三个 TMS320C64X+™ 内核。在启动阶段，Core0 开始进行启动，例如从 EEPROM 中读取启动代码等等，而 Core1 和 Core2 处于复位状态。Core0 可通过向 EVTASRT 寄存器的 EVTPULSE4 位写入 1 使得 Core1 和 Core2 脱离复位状态，开始从各自的 L2 RAM 基址执行，Core0 则从启动代码的入口地址开始执行[4]。

2.2 启动数据的生成

用 TI 的编程工具 CCS 编译连接生成后缀为.out 可执行文件，此目标文件格式被称作通用目标文件格式（COFF）。COFF 按照模块化思想对程序进行管理，它的最小单位称为段（section）。段是占据一个连续空间的代码块或者数据块，与其他段一起在存储器映射图内。但各个段是分开的，各有特色。对于 C 语言文件，编译器生成的代码分配在.text 段中，全局变量和静态变量分配在.bss 段中，而局部变量或寄存器变量分配到.stack 段。

链接器生成的可执行 COFF 文件（后缀为.out），含有一些定位符号和文件头等信息，这些信息能够被仿真器识别，仿真器可以从 COFF 文件中提取有用的程序，并把提取的程序加载到 DSP 的 L2 SRAM。 但是，如果我们采用 I2C 等方式启动时，COFF 文件中的一些信息不能被识别，而且由于含有的无效信息较多，COFF 文件比较大，因此，我们首先应该对 COFF 文件进行提取和精简处理。这就需要用到 TI 提供的十六进制转换工具（Hex6x.exe）。

经过转化工具生成的启动代码主要由三部分构成，如图 1 所示。第一个 32 位数据是程序的入口地址；最后一个 32 位数据 0x00000000，它是启动代码的结束标志；中间部分是主体部分，由程序中的若干 section 构成。所有的 section 都有一个统一的结构，以 section n 为例，第一个 32 位数据是此 section 的大小，以字节为单位；第二个 32 位数据是此 section 在 L2 SRAM 中的存放首地址；剩下部分是此 section 的数据内容[5]。

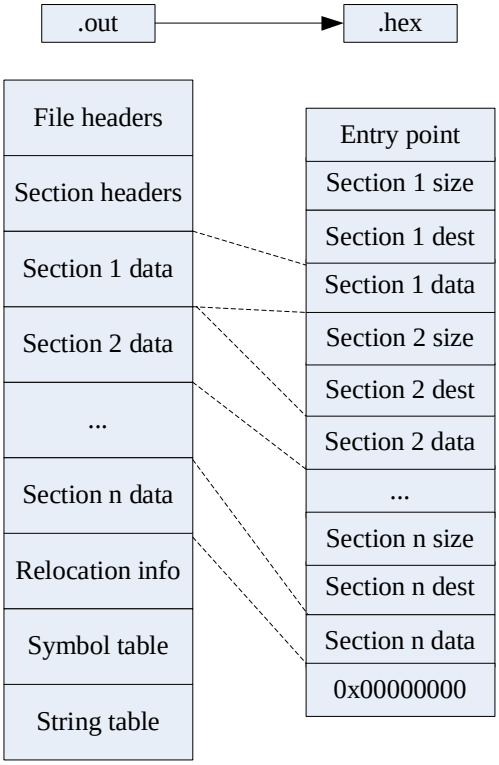

图 1 利用 hex6x.exe 工具转化.out 文件示意图[3]

3 TMS320C6474 主 I2C 方式的启动实现

当启动模式管脚 BOOTMODE[3:0]=0001b 时，TMS320C6474 的启动方式为 I2C 主启动方式。

用户首先要在从地址为 0x50 的 I2C EEPROM 中写入启动需要的数据块（block）。每个

数据块不能超过 128 个字节，每个数据块的前十个字节是数据块头部（Header）。当目标板上电启动后，DSP 计算出器件状态寄存器（DEVSTAT）中 CFGGP[2:0]位乘以 0x80，然后从这个地址开始读入数据块。头部结构如表 1：

表 1　I2C 数据块头部[3]

Offset(bytes)	0	2	4	6	8
Name	Block size	Checksum	Boot mode	Port	Sw　pll

数据块的内容包括两种，启动的参数表（I2C Boot Parameter Table）和启动数据块。启动的参数表分为配置表（Configuration Table）和启动表（Boot Table）。数据块则是配置和启动所需要的数据。启动参数表结构如表 2：

表 2　I2C 启动参数表[3]

Offset(bytes)	10	12	14	16	18
Name	Option	Dev addr low	Dev addr high	Broadcast add	Device ID
Offset(bytes)	20	22	24	26	28
Name	Core freq	I2C bus freq	Next addr low	Next add high	Addr delay

启动参数表中偏移头部 10 字节的位置是 Option 段，设置 Option 可以选择启动参数表的种类。如果用户想要在目标板运行程序之前，配置寄存器的值，可将 Option 置为 0010b，这个启动参数表为配置表。TMS320C6474 会根据配置表中偏移头部 12 字节的 Device address (low) 中的地址，从 EEPROM 相应的地址开始读取需要配置寄存器的地址以及相应的值，直至遇到连续三个全零的字（word）为结束。

当 Option 段中选择的是 0b001，启动参数表被设置为启动表。这个表中 Device address (low)中的地址就是 TMS320C6474 启动代码 EEPROM 中存放的起始地址。Bootloader 会根据启动表到从 EEPROM 相应的地址中读取这些数据[4]。

启动代码是用户工程的.out 文件，利用 hex6x.exe 工具所生成的。这些数据按照固定的格式写入 EEPROM 中。写入的初始地址为启动表 Device address (low)中的地址，按照 128byte 的数据块大小写入，包含 4 个字节的首部，以及 124 个字节的启动数据。当 TMS320C6474 读入参数表为启动表，则从 Device address (low)中的地址开始读入启动的数据，直到整个数据都放到程序运行的存储位置时，程序就开始自动运行。

综上，利用工具 hex6x.exe 将.out 文件转化成.hex 文件，将转化的文件按照数据块的格式写入 EEPROM 中，并在地址偏移量为 CFGGP[2:0]位乘以 0x80 的地址中写入启动参数表，参数表中的 Device address (low)中写入启动数据块的起始地址，整个流程见下图 2[3]：

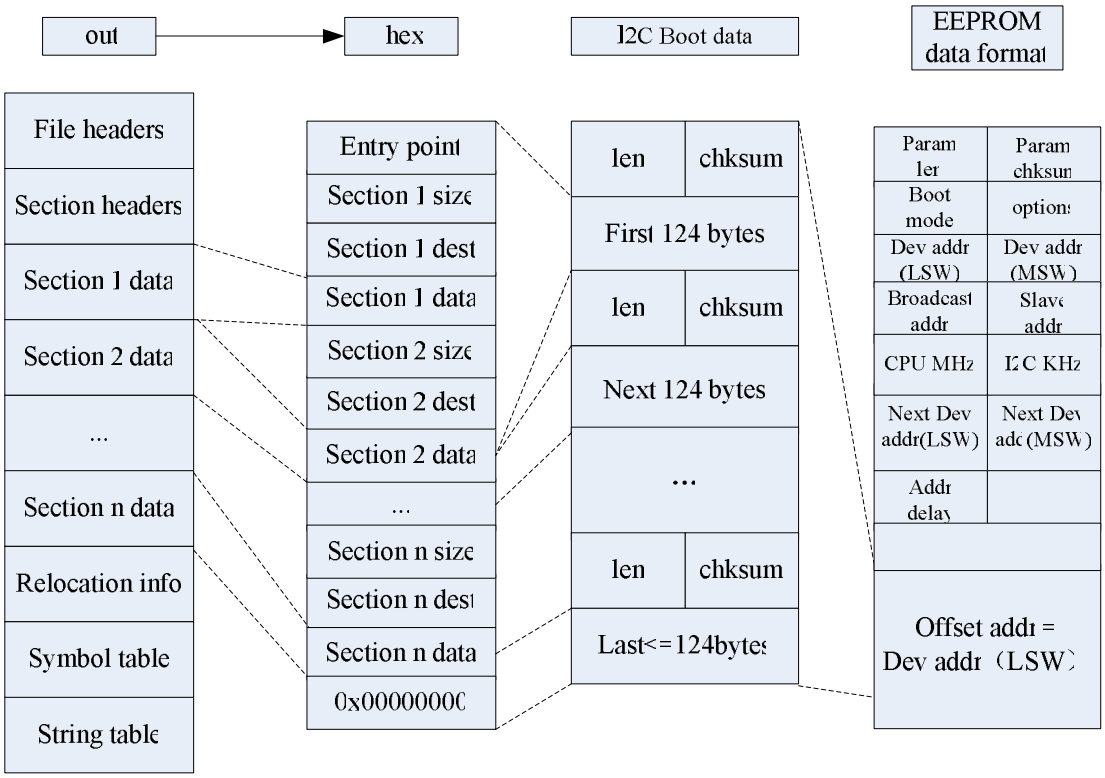

图 2　EEPROM 中存储数据示意图[3]

当用户的启动程序过大，无法装载在 EEPROM 中，或者对启动速度要求高的时候，或者想要更灵活的启动目标板，只使用 I2C 主启动模式是无法满足用户需求的，这时我们使用 TMS320C6474 的二次启动。

4 基于自启动的二次启动及多核启动

4.1 二次启动方式

二次启动就是用户根据需求，编写一个二次加载的程序 Secondary Bootloader。这段程序类似于 DSP 中固化的 Bootloader 的作用，就是搬移最终运行的启动代码，之后运行最终程序。但是 Secondary Bootloader 这段程序的运行是由一次启动加载的。整个过程就是，DSP 上电运行用户编写的 Secondary Bootloader 这段程序，这段程序搬移真正的用户代码，搬移之后跳转运行用户代码，开始执行。

例如用户最终要在目标板运行的工程是 User.pjt，用户编写的 Secondary Bootloader 的工程是 sec_boot.pjt，用户利用二次启动实现对 SPI flash 中程序的加载。整个流程如图 3：

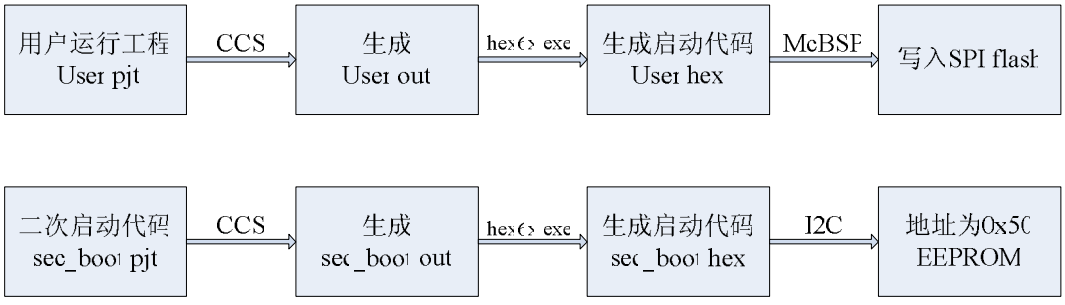

图 3 二次启动流程

目标板的启动模式选择 I2C 主启动模式。上电后目标板运行的流程入图 4：

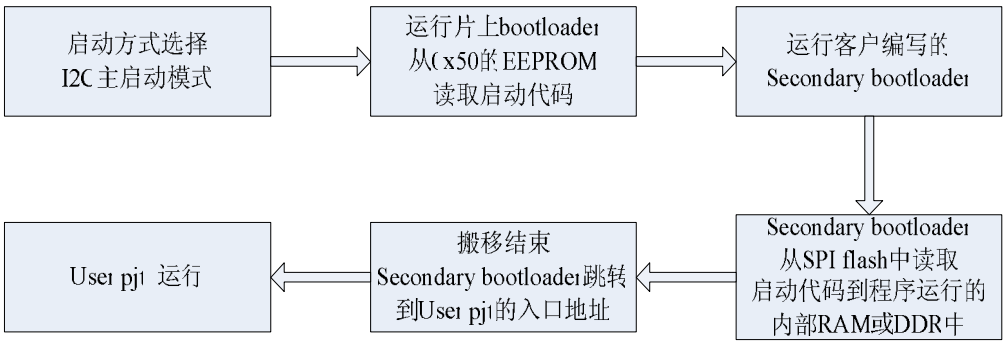

图 4 目标板上电运行流程图

根据二次启动的思想，用户可进行灵活的变换。只需要更改 Secondary Bootloader 的功能，用户可实现多种启动方式。例如，可以从其他存储器件中读启动数据，也可以等待主器件发送启动数据，只要最终实现启动代码完整的搬移到运行的 RAM 上，然后跳转的入口地址运行程序即可。

4.2 基于二次启动的多核启动

基于二次启动，用户可以实现多核启动。只要 Secondary Bootloader 搬移 Core 0 的启动数据后，将 Core1 和 Core2 运行的代码搬移到他们各自的 L2 RAM 上，并通过 Core 0 向 EVTASRT 寄存器的 EVTPULSE4 位写入 1 的方式，使得 Core1 和 Core2 开始从各自的 L2 RAM 基址开始执行[3]。与一般的启动代码不同的是，搬移的 Core1 和 Core2 的启动代码对入口地址有特别的要求，即程序的入口地址要是 L2 RAM 的基址，否则程序无法运行，所以要在用户工程中利用汇编语言或其他方式使程序的入口地址变为 L2 RAM 的基址，再将程序的.out 生成启动代码。

4.3 二次启动在实际系统的应用

在作者参与的实际项目中，设计的板卡上主要的数据处理芯片为 TMS320C7474 以及 FPGA（Field－Programmable Gate Array，现场可编程门阵列）。板卡的启动数据来源于 PC 端。板卡启动后，DSP 启动数据由 PC 通过 PCI 口将数据发送给 FPGA，再由 FPGA 通过千兆网口发给 DSP 进行启动。

DSP 的启动方式选为 I2C 主启动方式，系统上电后，DSP 从 EEPROM 中读取二级启动的代码运行，这段代码主要是使能千兆网口，配置寄存器以及开启中断，同时 FPGA 开始通过 PCI 从 PC 端接收启动数据并同时以以太网帧的形式传送给 DSP。每帧数据长为 1000 字节。前 4 个字节为启动数据要存放的 RAM 地址，之后两个字节为本帧中有效数据的长度，之后为启动数据。DSP 收到启动帧后，按照头部信息将帧中的启动数据放入 RAM 中，然后发回一个确认帧，FPGA 收到确认后继续发第二帧。以这样的顺序直到所有启动数据发送完毕，DSP 跳转指针开始运行程序。

当我们想切换板卡的功能时，只需给 DSP 复位信号，DSP 重新进行启动， PC 端发出板卡另一种功能的的启动代码，这样就可实现功能的转化。

5. 结束

本文介绍了 TMS320C6474 的 I2C 主启动方式，并利用这种启动方式实现二次启动和多核启动。用户可根据自己产品的需求来选择合适的启动方式。只要实现将最终运行的代码搬移到程序运行的内存中，然后跳转执行就可以了。TMS320C6474 的单核或多核启动都可基于这种方式实现。

参考文献

[1] 李芳慧，王飞，何佩琨．TMS320C6000 系列 DSPs 原理与应用（第二版）［M］．北京：电子工业出版社，2003．
[2] 季昱，林俊超，余本喜．嵌入式应用系统开发典型实例系列［M］．北京：中国电力出版社，2005．
[3] TEXAS INSTRUMENTS．TMS320C6474 DSP Bootloader User's Guide［Z］．Texas, TI, Oct. 2008．
[4] TEXAS INSTRUMENTS．TMS320C6474 Multicore Digital Signal Processor［Z］．Texas, TI, Oct. 2008．
[5] 吴海洲, 刘恒甫，黄克武．基于 TMS320C6455 的 DSP 加载模式研究［J］．北京：电子测量技术核心期刊，2008,31(6)：155-161．

Secondary Bootloader for TMS320C6474

Cheng Dan
Department of Communication and Information System, Beijing University of Posts and Telecommunications, Beijing (100876)

Abstract

This paper introduces multi-core DSP TMS320C6474's bootloader and boot sequence, as well as the master I2C boot mode. To solve the problems that the boot code is too large and the boot speed is too slow, secondary bootloader is presented. And based on this new method, the multi-core DSP can run multi-task automatically after reset. In the actual system, using the secondary bootloader can also boot the target board through different ways and achieve the function of switching among different operation modes.

Keywords: DSP;TMS320C6474;Secondary Bootloader

作者简介：

成丹，女，1985 年生，北京邮电大学硕士研究生，主要研究方向是嵌入式系统。