基于 TMS320C6X 的软件动态链接技术

陈延太　樊美琦

中电天奥有限公司第十研究所航空事业部　四川成都　610036

摘　要：介绍了软件动态链接及 ELF 文件格式，提出了一种基于 TMS320C6X 系列 DSP 的软件动态链接技术。该技术解决了可重构 DSP 系统中波形软件的动态加载及卸载的问题。使用该技术可实现平台软件与波形软件的隔离，提升软件重构效率，提高系统稳定性。

关键词：动态链接；TMS320C6X；重构

Software Dynamic Link Technology Based on TMS320C6X

Yantai Chen, Meiqi Fan

Aviation Business Department of the 10th Research Institute of China Electronics Tianao Co., Ltd, Chengdu, Sichuan, 610036

Abstract: This paper introduces the concepts of software dynamic linking and the ELF file format and proposes a software dynamic linking technique based on the TMS320C6X series DSP. This technique addresses the issue of dynamic loading and unloading of waveform software in reconfigurable DSP systems. By utilizing this technique, it is possible to achieve isolation between platform software and waveform software, thereby improving software reconfiguration efficiency and enhancing system stability.

Keywords: dynamic linking；TMS320C6X；reconstruction

一、问题提出

软件无线电（SDR）的关键思想是构造一个具有开放性、标准化、模块化的通用硬件平台，各种功能，如工作频段、调制解调类型、数据格式、加密模式、通信协议等，用软件来完成，并使宽带 A/D 和 D/A 转换器尽可能靠近天线，以研制出具有高度灵活性、开放性的新一代无线通信系统。可以说这种平台是可用软件控制和再定义的平台，选用不同软件模块就可以实现不同的功能，而且软件可以升级更新。

为了减少波形软件与硬件平台的耦合，提高其移植性，在软件无线电中往往会引入 SCA，SCA 主要作用在于提供标准的波形创建、释放及控制，并对波形软件提供通信中间件。

DSP+FPGA 的硬件架构在软件无线电中常常被选作标准的硬件平台[1][2]，一个完整的无线电软件如图 1 所示。

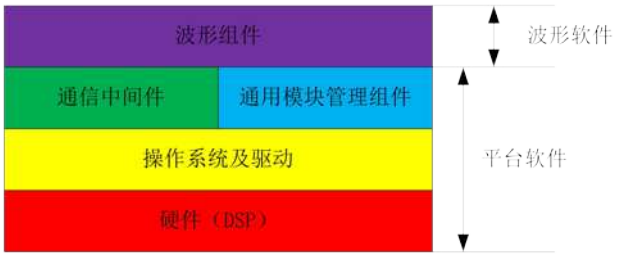

图 1 无线电软件结构框图

其中通用模块管理组件提供模块 BIT、波形加载卸载、软件在线升级等功能。

由于分工关系，波形软件与平台软件的开发者往往隶属不同部门，甚至属于不同公司，通常平台软件以静态库的形式提供波形软件开发者，由后者最终生成可执行文件。

由此带来以下问题：

(1)波形软件在编译时依赖平台软件提供的静态库，波形与平台在物理上未完全解耦，平台的更新同样会导致波形软件的升级；

(2)波形的切换必须通过 DSP 的复位后启动不同分区的代码来实现，这种方式不仅提高了硬件电路的复杂性，并且波形切换的时间长，同时 DSP 的复位也可能降低系统总线的可靠性（如 RIO 总线/1394 总线）；

(3)对于一些多核处理器，如果在多个核部署了不同的波形软件，其中一个波形软件的切换会导致整个处理器复位，从而导致其他波形软件的重新加载，这会大大增加系统设计的复杂性。

上述种种问题可通过动态加载卸载来实现[3]。首先，平台软件不再提供静态库，转而提供符号表文件（类似于 DLL），波形软件利用符号表文件完成编译（编译后的文件中包含部分未重定位的符号，不可独立运行）。其次，平台软件在处理器上电完成初始化后应对自身状态做一个快照，在对波形软件进行卸载时，平台软件对快照后由波形软件创建的资源（包括操作系统和通信中间件创建的资源）全部删

ISSN: 2661-3719(Print); 2661-3727(Online)

除。

二、ELF 文件及链接

ELF，即Executable and Linking Format，译为“可执行可链接格式”，具有这种格式的文件称为ELF文件[4]。分别从链接和运行的角度，可以ELF文件的组成部分做如图2的划分。

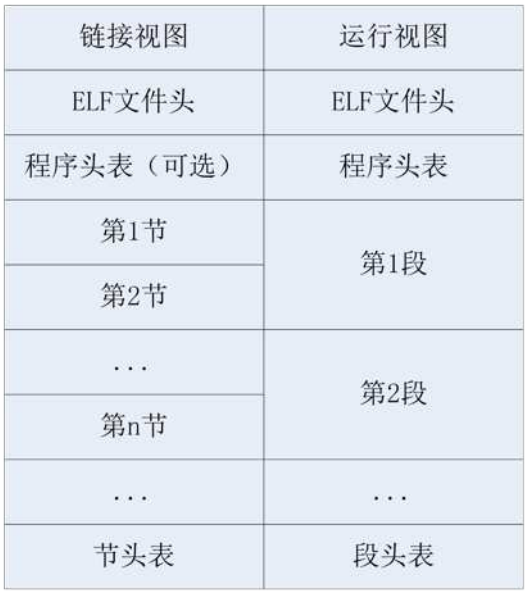

图 2 ELF 文件格式

ELF 文件主要分为以下三种类型：

(1)可重定位文件(relocatable file)，用于与其它目标文件进行链接以构建可执行文件或动态链接库。可重定位文件就是常说的目标文件，由源文件编译而成，但还没有链接成可执行文件。在 UNIX 系统下，一般有扩展名”.o”。之所以称其为“可重定位”，是因为在这些文件中，如果引用到其它目标文件或库文件中定义的符号（变量或者函数）的话，只是给出一个名字，这里还并不知道这个符号在哪里，其具体的地址是什么。需要在链接的过程中，把对这些外部符号的引用重新定位到其真正定义的位置上，所以称目标文件为“可重定位”或者“待重定位”的。

(2)可执行文件(executable file)，经过链接的，可以执行的程序文件。

(3)共享目标文件(shared object file)，即动态链接库文件。它在以下两种情况下被使用：第一，在链接过程中与其它动态链接库或可重定位文件一起构建新的目标文件；第二，在可执行文件被加载的过程中，被动态链接到新的进程中，成为运行代码的一部分。

三、波形软件动态链接的实现

要实现波形软件的动态加载，首先需要实现平台软件与波形软件在编译链接时的隔离，二者独立生成目标文件。一般而言，平台软件将编译生成可执行文件，而波形软件将生成共享目标文件，同时二者需要一个符号表文件包含平台软件对波形软件提供的所有符号。

3.1 符号库文件的生成

共享目标文件编译链接时存在未定义符号，因此需要链接一个符号表文件，以此来保证其符号完整性。符号表文件的源代码可分两部分，一部分为平台软件中的实现部分，一部分为符号表软件中的实现部分。

```
#define __SYMBOL(sym) {#sym, (Ptrval) &sym},
const DlSymbol    dlSymbol[] = {
        __SYMBOL(func1),
        __SYMBOL(func2),
//              __SYMBOL(func3),
//              {0}//terminated by NULL
};
```

在平台软件中，符号表包含全局符号及其编译地址。

```
#define __SYMBOL(sym) void sym(void){;}
        __SYMBOL(func1),
        __SYMBOL(func2),
//              __SYMBOL(func3),
```

在符号表软件中，仅仅是一组和平台软件中同名的空函数，供波形软件编译链接使用。

3.2 编译选项的设定[5]

对于平台软件而言，没有特殊的编译选项。对于符号表文件而言，有一个特殊的编译选项：

--dynamic=exe;

此编译软件可将文件中的所有全局符号动态导出。

对于波形软件而言，有以下特殊编译选项：

Memory data = far

此编译选项将所有全局或静态变量强制编译在.far 或.fardata 段，而不是将试图将一些符号编译在.bss 或.neardata 段。因为.bss 或.neardata 段使用 DP 指针间接寻址，而.far 或.fardata 是 32 位直接寻址。在波形软件启用之前，DP 指针的基址已赋值给平台软件的.bss 段首地址。如果波形软件也编译出.bss 段，将导致这部分数据寻址错误。

--import_undef = on

此编译选项将导入外部未定义符号，假设符号表文件名为 symbol.out，同时应增加-lsymbol.out 编译选项，这样就能导入符号表文件中的外部符号。

--inline_plt = on

此编译选项将强制消除.plt 与.got 段的生成。因为.plt 与.got 的运行机制涉及到延迟绑定，其符号重定位的时机在程序运行时，增加了程序运行的不确定性，这在某些潜入式系统中是不可接受的，因此利用该编译选项去掉应用文件中.plt 与.got 段的生成，这同时也简化了加载程序的设计。

--dynamic=lib;

此编译选项将应用程序编译为共享目标文件。选用该编译选项后，应用将不生成.cinit 段，这样就带来一个隐患。假定程序中有如下全局变量：

char gucDataBuffer[0x1000000] = {1,2};

此时，将有 0x1000000 个字节编译在.fardata 段，造成需要烧写的文件过大。解决方法有二：一个是通过编码规范强制大数组不得设定默认值，二是在程序烧写时增加压缩环节，同时程序解压时增加解压环节，此处与本文主旨无关，因此不展开论述。

3.3 符号重定位的实现

如前所述，本文波形软件有独特的编译选项，经过编译后文件中会生成四个特殊段：.dynamic/.dynstr/.dynsym/.rela.dyn[6]，其中.dynamic 表明需要动态链接的一些特殊段，通过它可以搜索到后面三个段；.dynstr 为动态链接涉及到的字符串表；.dynsym 为动态链接涉及到的符号表；.rela.dyn 为动态链接重定位表项。除去.rela.dyn 中所列重定位地址，共享目标文件的其余加载代码已经重定位，因此，在加载共享目标文件后根据加载文件的符号表和上述 4 个特殊段即可完成代码的重定位。

```
typedef struct {
Elf32_Addr r_offset;
Elf32_Word r_info;
Elf32_Sword r_addend;
} Elf32_Rela;
```

图 3 .rela.dyn 重定位表项结构

其中，r_offset 表示重定位发生的位置，r_info 低 8 位表示重定位类型，其余高位表示重定位符号在.dynsym 中的序号。r_addend 表示重定位域的值需要增加的数。

图 4 重定位实现流程

重定位类型分 R_C6000_ABS_L16 和 R_C6000_ABS_H16，其汇编指令如图 5 所示[7]。

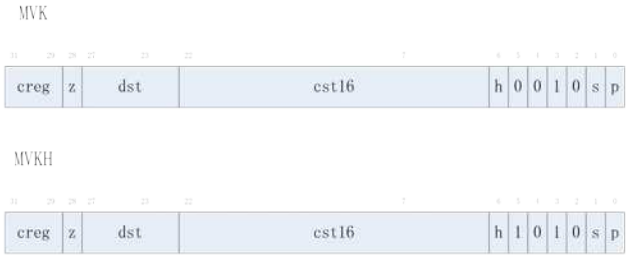

图 5 重定位涉及的汇编指令

对于 R_C6000_ABS_L16 类型的重定位，其重定位公式为

*r_offset = (*r_offset & ~MASK(16, 7)) | (((dlSymbol[n].val+r_addend) & ~MASK(15, 0)) << 7);

(1)

对于 R_C6000_ABS_H16 类型的重定位，其重定位公式为

*r_offset = (*r_offset & ~MASK(16, 7)) |

(((dlSymbol[n].val+r_addend) & ~MASK(31, 15)) >> 16 << 7);

(2)

其中，MASK(b, a)表示从 bit a 到 bit b 的掩码，dlSymbol 为平台符号表，n 表示该符号在平台符号表中的下标。

四、波形软件动态删除的实现

要实现波形软件的动态删除，首先应删除干净由波形软件创建的操作系统对象，其次应删除波形软件在通信中间件中创建的资源，最后应对波形软件在平台中注册的回调进行注销。特别需要提出的是，平台使用的堆与波形软件使用的堆应进行隔离，在删除波形软件时也应对其创建的堆进行回收。

本文以 TI 自带的 SYS/BIOS 操作系统为例，其操作系统内核对象包括 Clock/Mailbox/Semaphore/Event/Task/Swi/Hwi[8]等。图 6 演示了内核对象的快照代码，图 7 演示了内核对象的删除代码。

```
#define SYS_OBJ_TAIL_GET(objName, objGroup, objHandle, objHandleNext) do{ \
                    objHandle = (UINT32)objName##_Object_first(); \
                                if(objHandle != NULL)\
                                {\
                                  while((objHandleNext =
(UINT32)objName##_Object_next((objName##_Handle)objHandle)) != NULL)\
                                  {\
                                    objHandle = objHandleNext; \
                                  }\
                                }\
                    objGroup.##objName##Obj = objHandle;\
                                }while(0)
```

图 6 操作系统内核对象快照

```
#define SYS_OBJ_RESTORE(objName, objGroup, objHandle, objHandleNext) do{ \
                    objHandle = (UINT32)objName##_Object_first();\
                                objHandleNext = NULL;\
                                while(objHandle != NULL)\
                                {\
                                if(objGroup.##objName##Obj == NULL)\
                                  {\
                                    objHandleNext = objHandle;\
                                      break;\
                                  }\
                                  objHandleNext =
(UINT32)objName##_Object_next((objName##_Handle)objHandle); \
                                if(objHandle == objGroup.##objName##Obj)\
                                  {\
                                    break;\
                                  }\
                                objHandle = objHandleNext;\
                                }\
                                while(objHandleNext != NULL)\
                                {\
                                  objHandle =
(UINT32)objName##_Object_next((objName##_Handle)objHandleNext); \
                                objName##_delete((objName##_Handle
*)&objHandleNext); \
                                    objHandleNext = objHandle;\
                                  }\
                                }while(0)
```

图 7 操作系统内核对象删除

在 SYS/BIOS 中操作系统中，默认堆的指针由变量 xdc_runtime_Memory_defaultHeapInstance__C 保存，当加载波形软件时，此值应设为用户堆，当卸载波形软件时，此值应设为系统堆。

五、结束语

本文提出了一种基于 TMS320C6X 系列 DSP 的软件动态链接技术，使用该技术可快速实现软件无线电中的功能重构。在本技术具体应用的项目中，一个典型的波形重构时间可由原来的 2s 提升至 200ms。本技术依赖于特定的处理器及特定的操作系统，在可移植性上尚存在一定缺陷，这个问题尚需进行进一步的研究加以解决。

参考文献：

[1]赵谦，邓豹，刘婷婷. 嵌入式可重构信号处理计算机设计技术［J］． 计算机工程与设计，2020，41(11):3289 － 3293.

[2]刘可． 通用信号处理模块功能线程动态重构技术［J］． 无线电工程，2014，44(11):77 － 80.

[3] 邓伟， 许扬婧. 一种基于 TI TMS320 DSP 的软件动态链接 技术[J]. 电子设计工程，2012，20（11）:167-169.

[4] 何先波，唐宁九，吕方，等． ELF 文件格式及应用［J］. 计算机应用研究，2001，18(11) : 144 － 145，150.

[5] Texas Instrument． TMS320C6000 optimizing compiler user′s guide [EB/ OL]. [2012-06-XX]. http：//www.ti.com/

[6]陈宇，廖湘科，李慰. 静态链接动态库的ELF文件软件设计[J]. 微计算机信息，2008，24（3）:162-164

[7]Texas Instrument．TMS320C64x/C64x+DSP CPU and instruction set reference guide [EB/ OL].[2010-07-10]. http：//www.ti.com/

[8]Texas Instrument．TI－ＲTOS kernel (SYS /BIOS) user's guide［EB/ OL］. [2020-06-XX]. http：//www.ti.com/

作者简介：

陈延太，1982.06，男，汉族，籍贯四川遂宁，硕士研究生，中级工程师，研究方向为软件无线电与航空电子

樊美琦，1996.07，女，汉族，籍贯四川简阳，硕士研究生，助理工程师，研究方向为机载通信技术