ISSN1000-9825 Journal of Software 软件学报 2000,11(4):520~525

驱动程序生成器的设计与实现*

王颖 金涛 孙玉芳

(中国科学院软件研究所 北京 100080)

E-mail: yfsun@sonata.iscas.ac.cn

摘要 开发SCSI(small computer system interface)驱动程序自动生成器的目的是为了高效地开发各种驱动程序.该生成器可以自动完成SCSI子系统的代码维护,自动生成驱动程序绝大部分的模块,并能自动打包,提供主盘、从盘两种运行方式下的驱动程序.该文介绍了自动生成器的设计思想、系统控制流程及数据流程、系统模块及层次结构、接口和运行处理以及生成器的实现,包括模块管理、自动打包及模板生成.

关键词 自动生成器,代码维护,模板生成.

中图法分类号 TP319

自80年代初以来,SCSI(small computer system interface,小型计算机系统接口)[1]产品作为高效的存取手段被广泛地用于各种计算机.COSIX作为国产的类UNIX操作系统[2]最初的版本所支持的适配卡种类少、效率低,并且已基本过时.国外的外设厂家倾心于自身的外设产品,只需提供自身设备的驱动程序;操作系统厂家只需做驱动程序与核心程序的集成工作,他们都不必考虑驱动程序的生成问题.而对于国产操作系统,为了适应多变的驱动器硬件,就必须向用户提供高效的驱动程序,借助于自动生成器生成各种驱动程序,则是最理想的途径.当然,由于驱动器及适配器的多变及硬件依赖性,开发这种自动生成器难度极大.

本课题组经过近两年的艰苦努力圆满地完成了这一任务,经过实际评测和使用得知,自动生成的COSIX V1.1 SCSI驱动程序接近于UNIX SVR 4.0系统的驱动程序性能,在I/O操作的效率和占用核心空间大小上与传统驱动程序相当.利用这一生成器可大幅度减小SCSI设备驱动程序开发的复杂性,缩短研制周期,并可得到正确并且可信度很高的代码.本文第1节讲述驱动程序生成器的设计.第2节讲述系统的实现.第3节小结.

1 系统设计

1.1 系统构思

SCSI驱动程序生成器的设计与实现必然牵涉到SCSI驱动程序本身的结构设计.为了便于生成器的开发,我们首先为SCSI驱动子系统设计了清晰的层次结构,并使其数据结构和控制流程能灵活地适应不同的适配卡.

我们考察了许多厂家提供的SCSI子系统,深切地了解到不但各个厂家提供的SCSI系统结构不同,就是同一厂家的SCSI系统也不尽相同,而且系统内部结构不清晰.进一步的研究表明,虽然不同厂商的SCSI适配卡因硬件不同,软硬件的接口差异很大,但它们在较高层逻辑上有近似的操作步骤,毕竟它们遵循相同的SCSI规范.为此,在进行生成器总体结构设计时,将SCSI驱动程序在结构上进行划分,把不同适配卡的共性提炼出来,构成逻辑上相对独立的公共模块,而将不同适配卡的差异封装在各自不同的模块内,高层通过相同的函数接口引用.具体来说,我们把生成器分成以下三大部分:

(1) SCSI驱动程序公共模块

---

* 本文研究得到国家"九五"重点科技攻关项目基金(No.96-737)资助.作者王颖,女,1972年生,硕士生,主要研究领域为操作系统.金涛,1973年生,硕士生,主要研究领域为操作系统.孙玉芳,1947年生,研究员,博士生导师,主要研究领域为操作系统,中文信息处理,人型数据库,网络应用系统.

本文通讯联系人:孙玉芳,北京 100080,中国科学院软件研究所

本文1999-01-11收到原稿,1999-04-28收到修改稿

① PCI(外设控制接口)[3]设备支持函数库.PCI 总线是微机的主流总线结构.COSIX 以往的驱动程序编写方法不能适应 PCI 的灵活性及其特定的规则,因此,提供支持不同 PCI 设备的底层函数库为将来的系统核心扩展打下了基础.本模块作为 SCSI 驱动程序的可选模块,用以支持 PCI 总线上的 SCSI 适配卡.

② SCSI 目标设备模块.此模块包含若干子模块,不同子模块针对磁盘、磁带等不同类的设备.以磁盘目标设备模块为例,它屏蔽了不同容量、不同规格的 SCSI 硬盘的差异,向高层提供逻辑上完整的连续的磁盘空间.

③ SDI 接口及适配卡公共程序模块.COSIX SDI(SCSI 驱动程序接口)是 UNIX SVR4.0[4]SDI 的超集,在其基础上进行了扩充和完善.适配卡公共程序模块作为不同适配卡驱动程序的抽象和公共部分独立成一个模块,供 SCSI 适配卡驱动程序调用.

(2) SCSI 适配卡接口函数族

此函数族的地位类似于 SDI,只不过它不是针对目标设备,而是针对适配卡设备驱动设计的.对于不同 SCSI 适配卡,此函数族接口应是最优集合.也就是说,此函数族参与维护的逻辑数据结构数目最小,函数功能最为简单.这部分与不同的硬件结构紧密相关,必须手工干预,以实现其基本功能,并使整个 SCSI 驱动模块效率最高.

(3) SCSI 驱动程序生成及维护工具

在具备了上述(1)和(2)的基础上,生成器应能够:① 自动搜索目录结构,寻找适配器接口模块,并能最大限度地检测模块内部的合法性;② 自动生成 SCSI 驱动程序及系统核心,以便系统程序员分析调试用;③ 根据系统程序员需求,生成不同的打包文件,方便地生成系统安装盘;④ 交互地完成与系统有关的配置,有良好的用户界面,较强的容错性能.

由于不同的适配卡硬件差异很大,同时受到将来硬件升级扩展的约束,SCSI 适配卡接口函数族的最优化设计很难 100%地得以实现,但设计时将考虑各方面的要求,以实现某种平衡.

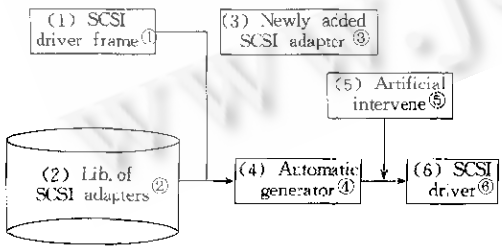
在此基础上,形成了生成器的驱动程序框架,并进一步提出了生成器的设计构想,如图 1 所示.

在图 1 中,SCSI 驱动程序框架(1)内包含 SCSI 驱动程序公共模块.而已有的 SCSI 不同型号适配器有关函数库(2)中主要包含 SCSI 适配卡接口函数,这部分函数与各种不同的硬件结构紧密相关,但却是一个个独立的模块.对于(2)中没有而新增的 SCSI 适配器,只需提供相关的适配器函数(3).在必要的人工干预(5)下通过自动生成软件(4)最终生成所需的 SCSI 驱动程序(6),当然,生成的 SCSI 驱动程序还需经过测试打包等过程才能最终完成.

在图 1 中,生成器由 Makefile,Shell 命令和应用程序组成,程序员通过一种交互界面而控制有关驱动程序生成绝大部分操作.系统中每一工具是交互界面下的子工具,同时又是一个能独立完成固定功能的模块.

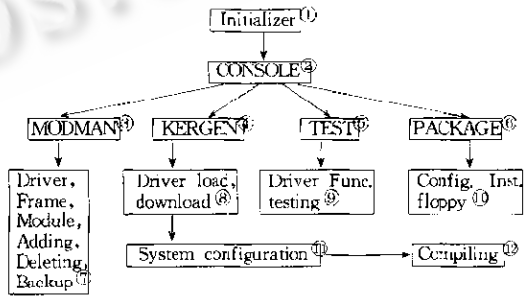
1.2 系统控制流程

生成器系统的控制流程如图 2 所示,图中的这些子模块完成生成器的各项功能.



①SCSI驱动程序框架,②已有的SCSI不同型号适配器函数库,③新增的SCSI适配器函数,④自动生成器软件,⑤人工干预,⑥SCSI驱动程序.

Fig. 1 Generator architecture
图1 生成器结构

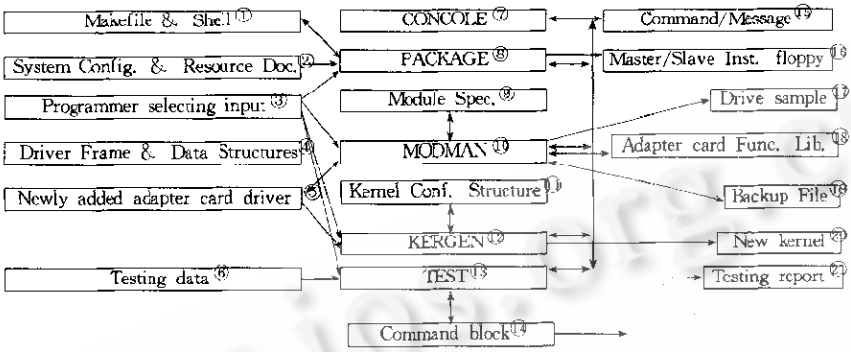


①初始化,②生成器控制台,③模块管理,④核心生成,⑤测试,⑥打包工具,⑦驱动程序、框架、模块、加入、删除、备份等,⑧驱动程序上、下载,⑨驱动程序功能测试,⑩配置生成安装盘,⑪系统配置,⑫编译.

Fig. 2 System flow
图2 系统流程

1.3 数据流程

在驱动程序框架的基础上,根据程序员对生成器的操作,生成器完成相应的功能.因此,生成器的数据源主要来自驱动程序框架、与核心生成相关的各种配置以及程序员的输入选择,经过生成器相关模块,产生输出,如图3所示.



①Makefile及Shell程序,②系统配置及资源说明文件,③程序员选择输入,④驱动程序框架及数据结构,⑤新编适配卡驱动程序,⑥测试数据,⑦交互控制台,⑧打包工具模块,⑨模块说明结构,⑩模块管理模块,⑪核心配置结构,⑫核心生成模块,⑬测试模块,⑭命令块,⑮命令/消息,⑯主、从安装盘,⑰驱动程序样本,⑱适配卡程序库,⑲备份文件,⑳新核心,㉑测试报告.

Fig. 3　Data flow
图3　数据流程

1.4 系统体系结构

1.4.1 系统组成

生成器的组成元素主要包括交互控制台(CONSOLE)、模块管理(MODMAN)、核心生成(KERGEN)、测试(TEST)、打包工具(PACKAGE)、驱动程序框架(SDF)、核心相关信息(KRI)及核心(KERNEL).下面作一概述.

交互控制台:为程序员提供控制驱动程序生成的交互界面.程序员可以通过菜单选项实现具体的操作与配置;同时,控制台向程序员显示各个子功能模块的执行情况和出错信息.本模块将通过信号和管道与各子功能进程通信,从而控制子功能模块的执行.

模块管理:对整个SCSI驱动程序模块进行管理.程序员可以通过此模块增加、删除、拷贝各SCSI模块,对新增加的适配卡程序模块提供样本指导.

核心生成及调试:对当前系统核心SCSI部分进行配置.程序员可以通过此模块增加、删除、核心各SCSI模块,并配置所需的参数,将错误及时反馈给控制台,帮助程序员方便地找到出错点.本模块将通过信号和管道与父进程通信,从而返回模块执行的情况.

测试:向程序员提供测试SCSI驱动程序的界面,显示测试例程执行的情况.由于不能保证程序员提供驱动程序的正确性,测试有可能造成系统崩溃的严重后果.本模块下属几组测试例程涵盖了基本SCSI设备测试的各个方面.本模块将通过信号和管道与父进程通信,从而返回模块执行的情况.

打包工具:按程序员的要求进行驱动程序打包的工作.本模块使用的makefile,shell脚本与原来的UNIX系统的兼容性,详见文献[5,6].由于SCSI设备既要求作为系统盘(即操作系统核心安装在上面的基本存储设备),又可以作为辅助的存储盘,所以要分别对主盘(前者)和从盘(后者)作不同的处理和打包.

测试例程库:本模块与测试模块的接口将按有利于程序移植的方式编制.也就是说,程序员按此接口编制的测试例程可以方便地加入测试模块中.

1.4.2 系统层次结构

生成器各元素的层次结构如图4所示.程序员通过控制台使用生成器的各个子模块,操纵SDF和KRI,得到核心并完成生成器相应的功能.而各个功能模块向控制台返回执行情况和出错信息.

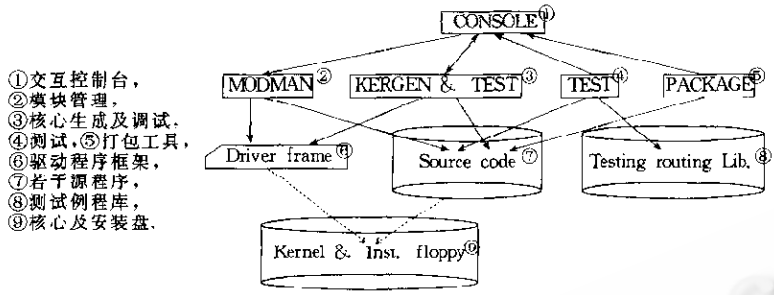


Fig. 4 System hierarchy
图4 系统层次结构

1.5 接口设计

1.5.1 用户接口

生成器将提供用户交互式界面,以便使用其各项功能.各项操作由菜单来引导,其菜单结构如图5所示.

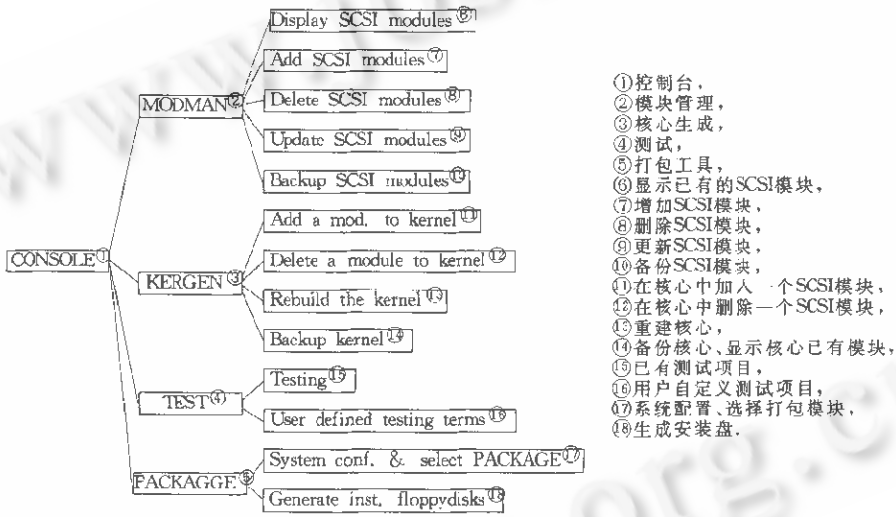


Fig. 5 Menu structure
图5 菜单结构

1.5.2 外部接口

生成器将调用大量操作系统功能,驱动程序的外部接口严格遵照COSIX不同版本支持软件的接口定义.

1.5.3 内部接口

生成器各子功能模块虽然相互影响,但是它们之间没有直接的接口.它们与控制台通过信号(signal)及管道(pipe)传递控制与信息.各子功能模块既可在控制台之下作后台运行,又可独立地以命令行形式启动运行.

1.6 运行设计

1.6.1 系统初始化

生成器初始化时先检查自身的合法性,看是否丢失了必要的文件;然后搜索目录检验各个SCSI模块,建立内部数据结构;最后搜索核心的配置情况,供子功能模块使用.

1.6.2 运行控制

生成器可以运行在两种模式下:(1)以交互界面的形式供程序员使用;(2)子功能模块独立运行,此形式下子功能模块从命令行中获得所需参数.

不论以何种形式使用,执行的结果是相同的.子功能模块自身能够分辨运行于何种模式下,并据此决定处理

结果的不同方法.生成器的各子功能模块是互斥的,即在一个子功能模块执行时,不能启动另一个子功能模块运行,从而保证有关 SCSI 的各个模块的一致性,也保证运行的安全性.

1.7 系统出错处理

系统出错将有 3 种情况:

• 不可恢复的错误,如死机.这类错误不是生成器的错误,往往是由新加入的 SCSI 模块的错误导致的,由于生成器对这部分不具有控制能力,所以不能向程序员提供出错信息及出错原因.

• 严重错误,如编译错、文件丢失等.对于这类错误,系统生成器一方面向程序员提供出错原因,另一方面尽可能地从错误中恢复,但仍需程序员参与处理.

• 警告信息.系统将不再中止正常的运行,而只是向程序员发出一些消息.

2 SCSI 驱动程序生成器功能实现

生成器是本着方便程序员编写 SCSI 驱动程序、提高劳动生产率的目的而设计的.它包括模块管理、自动打包、自动生成适配卡模板及编程指南等文档,提交给程序员的将是一个方便的编程环境.下面挑其重点加以阐述.

2.1 SCSI 驱动程序的模块管理

SCSI 驱动程序自己不但分成若干部分,它和其他模块的相互依赖关系也较为复杂.生成器的模块管理是严格遵循 COSIX 各版本 SCSI 驱动程序结构设计的,在此基础上,将这些文件分类.基本模块类别包括:① SDI 界面模块(MOD_SCSI),只有一个模块;② SCSI 设备驱动程序模块(MOD_STD),包含若干独立的模块;③ SCSI 适配卡驱动程序模块(MOD_SAD),包含若干独立的模块;④ SCSI 驱动程序支持模块(MOD_SUPPORT),是与 SCSI 驱动程序相独立的全局性支持模块,如 PCI 模块;⑤ SCSI 设备应用程序模块(MOD_COMMAND),如格式化 SCSI 硬盘等应用程序;⑥ SCSI 驱动程序安装模块(MOD_INSTALL),自动地打包、安装模块;⑦ SCSI 主盘安装模块(MOD_PROTO);⑧ SCSI 驱动程序公共前导文件模块(MOD_SYS);⑨ SCSI 驱动程序的其他 Makefile 文件(MOD_MAKEFILE).

SCSI 驱动程序中几乎所有的文件都包含在上述模块类中.生成器将按模块类别对模块及其不同版本进行管理,版本管理的粒度是以模块为单位进行的.程序员较为关心的是前 4 类模块,其他模块只是为了支持驱动程序正常的编译才提供的.生成器已为每个模块定义了一个模块描述文件"module",以便于模块内的文件管理.文件位于 COSIX 各种版本 SCSI 驱动程序根目录的相应模块子目录中.

生成器模块管理功能包括从代码库中抽出指定模块的指定版本、加入指定模块的指定版本到库中、抽出完整的 SCSI 驱动程序代码指定版本、加入完整的 SCSI 驱动程序代码指定版本到代码库中的功能、列表显示某一模块版本信息、列表显示完整的驱动程序代码版本信息、备份某一版本源码等.

程序员一般不需要了解 *.module 文件,因为生成器为生成的新的适配卡驱动程序自动生成 *.module 文件.若需要加入一个新文件到某一模块中,则需要按语法格式加入新文件.加入的新文件既可以是普通文件,也可以是目录名,其路径前缀都是相对于 SCSI 各版本驱动程序根目录的.

2.2 SCSI 适配卡驱动程序模板生成

生成器最主要的功能是自动生成 SCSI 适配卡驱动程序模板,建立一个程序框架,而后由程序员将与硬件有关的部分填入相应的函数中.

例如:modman-g test "The Test SCSI Card" cosix,将在 cosix 目录下建立完整的 test 模块,并且将模块加入 cosix/scsi.in/ID/Makefile.cosix/scsi.in/io/Makefile 中,新的 test 模块将可顺利地编译到整个 SCSI 驱动程序包中.除此以外,生成器还修改 cosix 目录下有关从盘、主盘打包的相关文件,使得程序员几乎不需要维护这些文件就可以生成包含 test 适配卡驱动程序的从盘、主盘的软盘安装盘.若需要增加 test 模块内的文件,程序员可以编辑 *.module 文件,将要加入的文件项附在程序尾部,若加入的文件是一个目录,则一定要保证依赖此目录的其他文件项在此目录项之后.生成器对 test.c,test.h 中的某些数据结构和函数名称作了一些限制,这些限制也

是程序自动生成的必要代价.新的 test.c 是可以正确编译并纳入核心的适配卡驱动程序框架,但有待于程序员填写有关 SCSI 适配卡硬件的代码才能成为有实际意义的 SCSI 适配卡驱动程序.

程序员也可通过键入命令 modman-d modname directory 将适配卡驱动程序模块从 SCSI 根目录中删去,生成工具会自动调整 Makefile,Shell 程序,使之恢复到加入适配卡驱动程序之前的状态.

2.3 SCSI 驱动程序的自动打包

COSIX 各版本 SCSI 驱动程序自动打包功能建立在 Makefile 基础上.程序员只需简单地键入 Makefile 的相应入口,就可以生成从盘、主盘的软盘安装盘.

3 小 结

COSIX 两种版本(COSIX V1.x 和 COSIX V2.x)的 SCSI 系统及驱动程序生成器经过一年多的调研、设计、编码、测试和实际应用,在 1998 年 1 月顺利通过了国家相关攻关项目课题组的验收.本项目突破了一系列技术难关,SCSI 驱动程序生成器具有鲜明的特色.投入实际应用以来的情况表明,该系统不仅提高了开发效率,而且大大提高了代码的正确率及可信度.这说明本项目实现了设计目标,扩展了 COSIX 的应用领域.

参考文献

1 Ou-Yang Xing-hua. Computer system interface —SCSI. Beijing: Electronic Industry Press, 1995
(欧阳兴华.计算机系统接口——SCSI.北京:电子工业出版社,1995)

2 Liu Ri-sheng, Sun Yu-fang. The analyst report of the UNIX operating system. Computer Research and Development, 1982,19(9,10):1~115
(刘日升,孙玉芳.UNIX 操作系统分析报告.计算机研究与发展,1982,19(9,10):1~115)

3 PCI Special Interest Group. PCI LOCAL BUS SPECIFICATION Rev 2.1, USA Portland, 1995

4 AT&T UNIX system operation. UNIX System V/386 Release 4 Programmer Guide: SCSI Driver Interface. Englewood Cliffs, NJ: Prentice Hall, Inc., 1990
(史树民,董湘端译.UNIX 系统 V/386 第 4 版程序员指南:SCSI 驱动程序界面.北京:电子工业出版社,1992)

5 Li Zhi-chen, Wang Ying. COSIX V1.x SCSI-like hard disk driver low level design specifications. Beijing, Open System & Chinese Information Processing Center, Institute of Software, the Chinese Academy of Sciences, 1997
(李志臣,王颖.Cosix V1.x SCSI 类硬盘驱动程序详细设计说明书.北京:中国科学院软件研究所开放系统与中文信息处理中心,1997)

6 Jin Tao, He Jun. COSIX V2.x SCSI-like hard disk driver low level design specifications. Beijing, Open System & Chinese Information Processing Center, Institute of Software, the Chinese Academy of Sciences, 1997
(金涛,何军.Cosix V2.x SCSI 类硬盘驱动程序详细说明书.北京:中国科学院软件研究所开放系统与中文信息处理中心,1997)

Design and Implementation of a Driver Generator

WANG Ying JIN Tao SUN Yu-fang

(Institute of Software The Chinese Academy of Sciences Beijing 100080)

Abstract An SCSI (small computer system interface) driver automatic generator has been developed for developing SCSI drivers efficiently. The generator can automatically maintain SCSI subsystem code, generate most driver modules, package, and provide drivers under the master and slave running modes. In this paper, the authors present the automatic generator's design method, system control flow, data flow, system modules and hierarchy, interface and running processing. The paper describes in brief the system implementation, including the module management, automatic packaging, and model generation.

Key words Automatic generator, code maintainable, model generation.