国防科技大学学报
JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第10卷第2期　1988年6月　Vol.10,No.2

互联系统的驱动器设计

杨　利　倪继坤
（计算机科学系）

摘　要　本文通过对一般驱动器的讨论，结合已在某互联系统中实现的驱动器设计，探讨根据物理机结构，开发驱动器与系统中其它各层软件间的并发性途径。

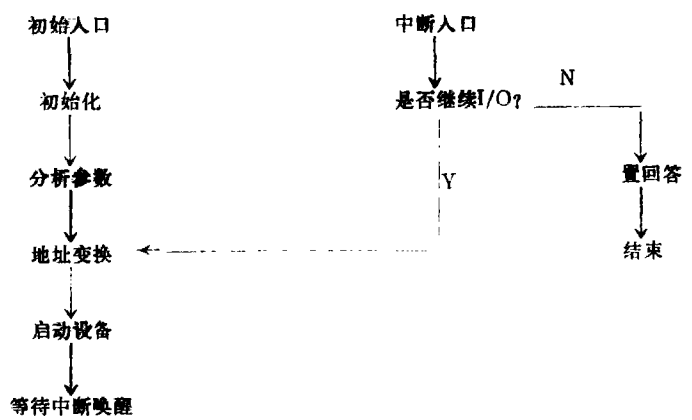
关键词　互联系统，并发，驱动器

1. 引　言

驱动器(Driver)是紧缚于裸机上的一层软件。在早期的计算机中，这部分功能是被编入统一的设备管理程序。按现代操作系统的分层概念，驱动器属于操作系统的内核部分。从与硬件的耦合程度来看，驱动器与机器结构紧密相关。因此，如何充分利用计算机本身的结构特点，是设计一个能高效运行的驱动器的关键。本文结合一个大型互联系统中的驱动器设计，提出若干开发并行驱动器的方法。

2. 一般驱动器的结构和特点

普通的I/O通道结构的计算机中的驱动器的基本逻辑流程如图a.



本文1987年3月3日收到

这种模式的驱动器完全是靠中断制导的，它的特点是：

- 程序的逻辑流程是线性的；
- 它的运行与操作系统的其它部分是串行的；
- 它的运行是完全被动的；
- 参数一次给出并固定不变；
- 不涉及系统的文件结构。

3. 互联系统概述

3.1 互联模式

我们所讨论的互联系统——又叫前端互联系统——具有下面的模式：

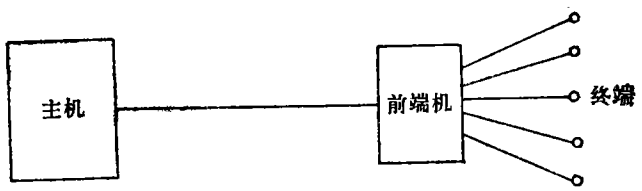


图 1

互联系统由主机（含海量存贮设备）和多个前端机（Front-end）所组成。前端机，作为主机与操作员、用户之间的媒介而存在，称为“前端站”（Front-end Station）。前端站从硬件上说，把主机和前端机互联起来；从软件上说，是把控制主机运行的O.S.与控制前端机运行的O.S.互联起来。前端站软件在前端机O.S.的控制下，专门负责前端机与主机的文件、命令传送。互联系统与网络的不同点在于它不追求通用性目标，没有网络寻址问题。但本质上，互联系统也解决了计算机间的通讯问题。

国际标准化组织提出的开放系统互联的参考模型（OSI模型）被广泛地用来设计和分析网络的体系结构。按照网络分层观点，我们建立了一个前端互联系统的四层模型（图 2）：

前端站软件（以后简称站软件）的实现与分层模型的对应关系如下（图 3）。

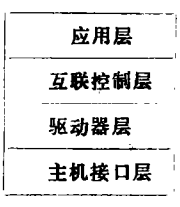


图 2

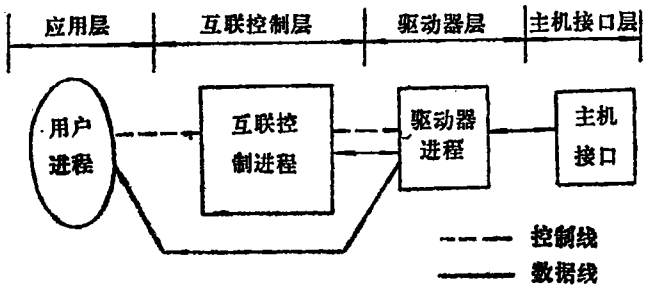


图 3

3.2 前端机的系统结构特点

在互联系统中所采用的前端机之一是一台具有多个“输入/输出处理机”（简称“I/O处理机”）的大型计算机。这种结构有如下特点：

• 每个IOP_i($i=1,2,\cdots,n$)都是一台小型处理机（同构的或类同构的），并自带内存，有自己的指令系统，可独立运行程序；

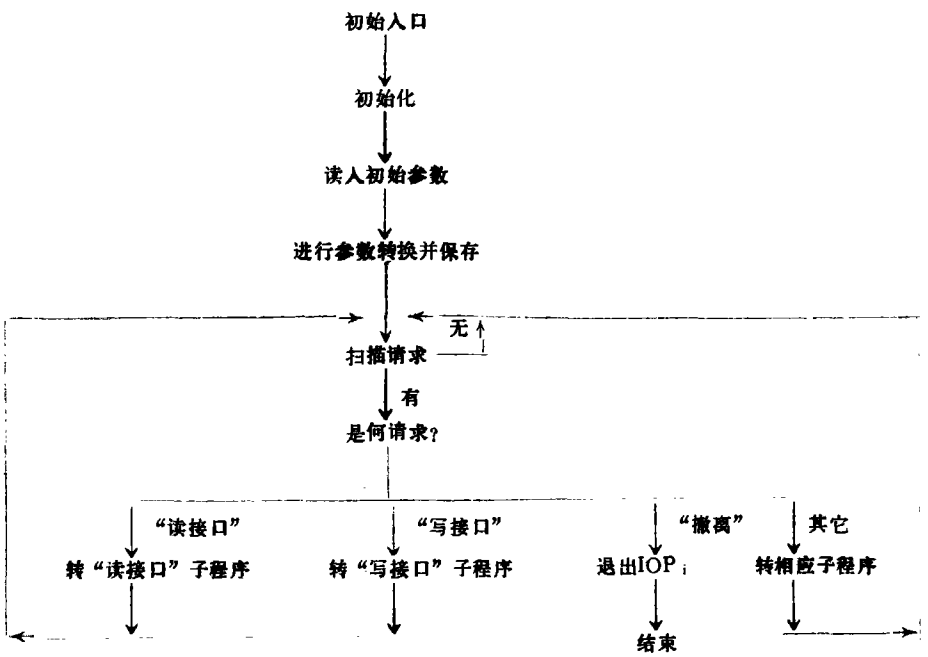
• 各IOP_i可直接访问全部中央内存；

• 由各IOP_i来实施对外部设备的管理；

• 主机CPU不直接处理各I/O请求和I/O中断。在这种两级处理机结构的支持下，操作系统被从物理上分成了两大部分：一部分运行于主机CPU上；另一部分则分布于各个IOP_i中。两部分协调工作，共同完成对整个系统的管理。特别要指出的是，驱动器全部运行在IOP_i中。因此，驱动器可与操作系统中的其它各部分并发执行。在站软件的组成中，驱动器进程运行在IOP上，而用户进程和互联控制进程运行于中央处理机上。

4. 互联系统驱动器的设计特点

互联系统中前端站驱动器的主要任务是管理主机接口硬件，以完成主机与前端机间的数据交换。我们充分考虑了前端机的特点，把驱动器设计成了一个常驻运行、主动式并具有一定程度“智能”的软件。它的逻辑流程如图b.



4.1 工作流程

首先由系统操作员在前端机控制台拍入相应命令，操作系统将先加载互联控制进程。互联控制进程在完成自身初始化工作之后，再通知系统加载驱动器进程到某一个空

闲的IOP中。一旦驱动器进程投入运行，它就将常驻在这个IOP中，直至站软件系统撤离。站驱动器的主体循环部分利用一些系统提供的和自定义的通讯手段，按约定的程式与运行在中央内存的互联控制进程进行协调通讯，捕捉I/O请求。

要求驱动器完成的I/O功能请求一般分为两类：

1．源于用户进程的请求。例如接收来自主机数据段的请求，发送数据段至主机的请求等。互联控制进程首先在主循环时捕捉到这些请求，然后通知站驱动器；

2．源于互联控制进程的请求。例如，传递用于同主机保持通讯的控制报文、命令报文等。另外，一旦通讯过程中出现异常，需要互联控制进程请求站驱动器给用户进程转达出错信息。

4.2 驱动器进程与互联控制进程的通讯

常规驱动器都是由中断进入的。这也就是说，当需调用驱动器时，系统根据“中断向量”找到对应的驱动器入口，切换当前的运行环境。毫无疑问，这是一笔开销。

站驱动器直接受互联控制进程的控制。所有功能请求都是由互联控制进程转送给站驱动器。除初始功能外，其它所有I/O功能请求均不经过O.S.，它们两者的通讯区常驻在中央内存。I/O请求及参数的传送是利用“传值”(Call-by-Vaule)方式，完全靠通讯区进行，与O.S.无关。驱动器将及时把I/O的完成状态返回到通讯区中。

4.3 驱动器进程与用户进程的通讯

站驱动器与用户进程之间没有直接的控制路径（图3）。用户进程的I/O请求参数表是通过“传指针”(Call-by Reference)进入互联控制进程空间，再经过驱动器与互联控制进程的通讯区进入驱动器空间。驱动器通过引用“指针”对该参数表访问，从中获得用户进程的数据区地址、数据长度及其它控制信息。驱动器将根据这些参数进行I/O操作。

4.4 驱动器进程的并行设计

充分开发并行因素应是驱动器进程的设计原则。在设计站驱动器时，可供开发的三级并行是：

- 驱动器与O.S.的并行
- 驱动器与互联控制进程及用户进程的并行
- 驱动器内部的并行

4.4.1 驱动器与O.S.的并行

这是前端机O.S.本身提供的能力。由于驱动器与CPU O.S.运行在不同的处理机上，使这两者的并行成为可能。

4.4.2 驱动器与互联控制进程及用户进程的并行

这里仅结合驱动器进程对“写接口”功能的实现，说明这种并行设计。

所谓“写接口”功能意指将前端机的一个信息包通过主机接口硬件发往主机去的一次I/O操作。它的完成在站软件内部要历经三个过程（图4）。

- 过程1：用户进程采集磁带等介质上的数据至中央内存；
- 过程2：站驱动器将这批数据复制到IOP内存；
- 过程3：站驱动器将IOP内存中的数据组装后发往主机（通过主机接口硬件）。

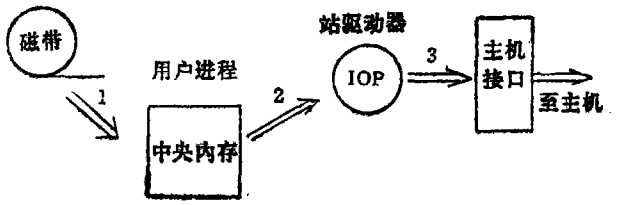


图 4

过程1和过程2在常规处理中是串行完成的。为了使这两个过程能并行进行，在用户进程的空间内设立一个循环缓冲区（Circular Buffer），用户进程关于该缓冲区持有如下一张表：

表 1

IN指针	OUT指针
数据长度	

其中，（对“写接口”功能请求而言）“IN指针”指向缓冲区中的自由空间的起始地址，它是由用户进程自身使用并修改的；“OUT指针”指向缓冲区内可用数据的起始地址，由站驱动器使用并修改；“数据长度”指（“写接口”功能请求）要完成的传送长度（受限于通讯规程）。表1通过“传指针”(Call-by-Reference)方式被站驱动器引用。由于站驱动器和用户进程共享I/O参数表和循环缓冲区，而这两个进程又运行在不同的处理机上，因而使得用户进程采集数据的过程（过程1）与站驱动器读取数据的过程（过程2）可以并行进行。限于篇幅，这里略去循环缓冲区的管理算法。

4.4.3 站驱动器内部的并行

从图4可以看出：过程2与过程3在一个程序内只能串行。如若串行，将会损失接口硬件50%以上的传输率。前端机有多个IOP部件，因此，我们将站驱动器分成两部分：主驱动器和辅驱动器，构成主—辅（Master/Slave）驱动器结构。主驱动器与辅驱动器分别运行在两个不同的IOP中。它们之间的通讯协调和实现方法，在此就不详述了。特别需要指出的是，主—辅驱动器结构对互联控制进程及用户进程是透明的。

在设计前端站驱动器时，我们几乎把全部并发因素开发贻尽，加上编码方面的考虑，为整个站软件系统的性能提供了可观的余量。

4.5 站驱动器的特点及性能分析

1) 驱动器的逻辑结构是非线性的；

2) 采用主动式工作方式。由于把站驱动器设计成常驻运行的，它的初始化工作可在系统调用时一次完成，以后的I/O请求调用均可节省进管和初始化这两部分时间开销。一旦有I/O请求，站驱动器还可自动寻找和产生所需的参数；

3) 与用户进程共享参数表格，便于用户进程动态扩展内存，也利于并行设计；

4) 根据I/O功能请求码和数据负载情况，主驱动器可自动决定它与辅驱动器的协调，这方面对其它进程是透明的；

5) 各级别的充分并行。

我们从对图4中三个过程的处理上分析站驱动器的性能。

在分析开始之前，先做一些说明和假设。前端机与主机的一次I/O传输即是前端机互联控制层与主机的对应层之间的一次通话。由于通话的信息量较大，反应到驱动器层，可能要进行若干次通话。也即驱动器要将一次I/O传输的信息进行分割，分若干次启动主机接口传送，每一次的传送长度记作τ。假定：

1) 设图4中的过程1、过程2及过程3都以τ为单位，且传送单位长度数据的时间最长为O(t)

2) 设前端机与主机的一次I/O传输的信息长度为$n\times\tau$

对于一次I/O传输，图4中的三个过程按串行执行和并行执行时的时间图分别由图5和图6给出。

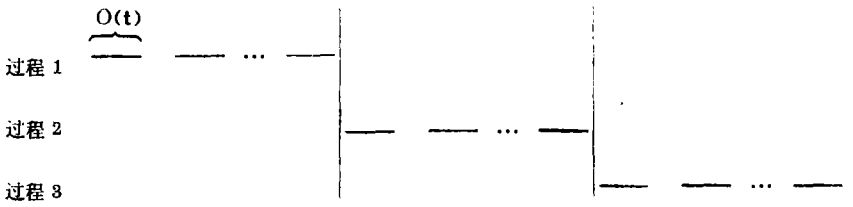


图5

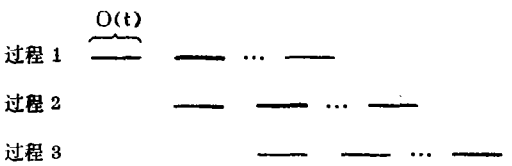


图 6

图5的时间开销为：

$$T_1=3\times n\times O(t)$$

图6的时间开销为：

$$T_2=4\times O(t)+(n-2)\times O(t)$$
$$=(n+2)\times O(t)$$

并且，$T_1/T_2\longrightarrow 3$，当$n\to+\infty$.

由上面的分析，并行驱动器使前端机与主机的一次I/O传输率较传统的驱动器提高了近两倍，而且，一次I/O的数据越长，效率提高就越明显，见图7。

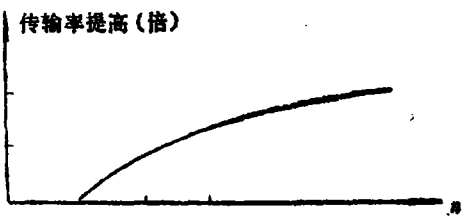


图 7

5. 结　语

本文试图说明，驱动器由于其与硬件的紧耦合和它处在操作系统内核这一事实，在设计时要根据物理机的结构特点，充分开发各种并行因素，极力使其代码空间尽量地小，运行时间（指有效运行时间）尽量地短，效率尽量地高。综合这几点，驱动器的设计就不只是程序设计问题。必须按照系统环境、应用环境，进行反复分析、评价。本文

给出的是一个已在实际运行的并证明是高效的互联系统驱动器的设计例子。

国防科技大学计算机系陈立杰教授在本课题研究和本文成文过程中提出了许多指导性意见并审阅了全文，在此致谢。

参 考 文 献

[1] 苏东庄，计算机的系统结构，国防工业出版社

[2] [荷兰] A.S. 格南鲍姆，分级式计算机的构成，科学出版社

Design of Driver in an Interconnection System

Yang Li Ni Jikun

Abstract

This paper is intended as a research on developing the concurrency between a driver and other layers of software in terms of the structure of physical machines. After a discussion of general driver, the driver implemented in an interconnection system is given.

KEY WORDS Interconnection system, Concurrency, Driver