深入理解计算机系统总结报告

计算机与信息工程学院 2014级网络编程班 申春阳 20141105056

指导教师 朝力萌 讲师

摘要 本报告的主要内容是关于计算机体系结构与编译器和操作系统的交互，包括计算机体系结构，分别是总线、I/O设备、主存、处理器、存储器、虚拟存储器、指针等。本报告为读者描述计算机在工作当中实现的细节，帮助读者在大脑中构造一个层次型的计算机系统，从内存中的程序，通过编译系统，到流水线指令的执行，最后到应用程序的实现。

关键词 处理器；I/O设备；主存；编译；高速缓存；进程；程序语言；虚拟内存；动态存储分配；嵌入式

**Thorough understanding of Computer systems summary reports**

**SHEN Chunyang**

**Abstract: The main content of this report is a computer architecture and compiler and operating system interaction, including computer architecture, respectively is bus, I / O devices, main memory, the processor, memory, virtual memory, a pointer. This report describes the details of the implementation of the computer in the work, to help readers in the brain to construct a hierarchical computer system, from the memory of the data, through the compiler system, to the implementation of the assembly line, and finally to the implementation of the application.**

**Key words：Processor; I / O devices; memory; compile; cache; process; programming language; virtual memory; dynamic storage allocation; embedded**

0. 引言 报告讲到计算机系统是如何工作的，系统由硬件和软件系统组成的，他们共同工作来运行应用程序。计算机系统都有相似的硬件和软件组件，它们执行相似的功能。通过对计算机系统的一个学期的学习，让我们深入的了解了计算机系统，对日后的编程，会更加具有针对性。

1 计算机系统的组成部件

* 1. **系统的硬件组成**
     1. **总线**

总线是贯穿整个系统的一组电子管道，它携带信息字节并负责在各个部件传递。通常总线被设计成传送定长的字节块（字）。现在大多数机器字长有32个字节（32位），有的是8个字节（64位）。

* + 1. **I/O设备**

输入/输出（I/O）设备是系统与外部世界的联系通道。每一个I/O设备都通过一个控制器或者适配器与I/O总线相连接。控制器和适配器之间的区别主要在于它们的封装方式。控制器是置于I/O设备本身的或者系统的主印制电路板（通常的称为主版）上的芯片组，而适配器则是一块儿插在主板槽上的卡。它们的功能都是在I/O设备之间传递信息。

**1.1.3 主存**

主存是一个临时存储设备，在处理器执行程序的时候，用来存放程序和程序处理的数据。从物理上来说，主存是由一组动态随机存取存储器芯片组成的。从逻辑上讲，存储器是一个线性的字节数组，每一个都有其唯一的地址（数组索引），这些地址是从零开始的。一般来说，组成程序的每条机器指令都是由不同数量的字节构成的。

* + 1. **处理器**

处理器是中央处理单元的简称，是解释（或执行）存储在主存中指令的引擎。处理器的核心是一个字节长的存储设备（或寄存器），称为程序计数器（PC）。在任何时候，程序都指向主存中的某条机器语言指令（即含有该条指令的地址）。

从系统通电开始，直到系统断电，处理器一直在不断地执行程序计数器指向的指令，再更新程序计数器，使其指向下一条指令。处理器从程序计数器（PC）指向的存储器读取指令，解释指令中的位，执行该指令指示的简单操作，然后更新PC，使其指向下一条指令，更新过的指令指向的并不一是上一条执行过的指令所相邻的指令。

处理器看上去只是它的指令集结构的简单实现，但是实际上现代处理器使用了非常复杂的机制来加速程序的执行。

2 计算机是如何工作的

下面我们引用计算机程序经典中的经典，“hello，word”计算机初学者的第一个示范程序，它还可以用来确定该语言的编译器、程序开发环境，以及运行环境是否已经安装妥当。

#include<stdio.h>

int main()

{

printf("hello,word\n");

return 0;

}

**2.1 程序被其他程序翻译成不同的格式**

hello程序的生命周期是从一个高级C语言程序开始的，因为这种形式能够被人们读懂。在系统上运行hello.c程序，每一条C语句都必须被其他程序转化为一系列低级机器语言指令。然后这些指令按照一种称为可执行目标程序的格式打包好，在这里，编译器驱动程序读取源文件hello.c，并把它翻译成一个可执行的目标文件hello，并且以二进制磁盘文件的形式存放起来。

编译的过程可分为四个阶段完成，执行着四个阶段的程序分别是预处理器、编译器、汇编器和链接器一起构成了编译系统。

* 预处理阶段。预处理器（cpp）根据以字符#开头的命令，修改原始的C程序。例如：hello.c中第一行的#include<stdio.h>命令告诉预处理器读取系统头文件stdio.h的内容，并把它直接插入到程序文本中。结果就得到了另一个C程序，通常是以.i作为文件扩展名。
* 编译阶段。编译器（ccl）将文本文件hello.i翻译成文本文件hello.s，它还包含一个汇编语言程序。汇编语言程序中每一条语句都以一种标准的文本格式确切地描述了一条低级机器语言指令。汇编语言为不同的高级语言的不同编译器提供了通用的输出语言。
* 汇编阶段。接下来，汇编器（as）将hello.s翻译成机器语言指令，把这些指令打包成一种叫做可重定位目标目标程序的格式，并将结果保存在目标文件hello.o中。hello.o文件是一个二进制文件，它的字节编码是机器语言指令而不是字符。我们如果打开hello.o文件，看到的将是一堆乱码。
* 链接阶段。Hello程序调用了printf函数，它是为每个编译器都会提供的标准C库中的一个函数。printf函数存在于一个名为printf.o的单独的预编译好了的目标文件中，而这个文件必须以某种方式合并到hello.o程序中。链接器（ld）就负责处理这个合并。结果就得到hello文件，它是一个可执行的目标文件，可以被加载到内存中，由系统执行。

**2.2 处理器读并解释存储在存储器中的指令**

此刻，hello.c源程序已经被编译系统翻译成可执行的目标文件hello，并存放在磁盘上。外壳是一个命令行解释器，它输出一个提示符，等待输入一个命令行，然后执行这个命令。初始时，外壳程序执行它的指令，等待我们输入一个指令。当我们在键盘上输入字符串“./hello”后，外壳程序将字符逐一读到寄存器，再把它存放到存储器中，当我们在键盘上敲回车键的时候，外壳程序就知道我们已经结束了命令的输入。然后外壳执行一系列指令来加载可执行的hello文件，将hello目标文件中的代码和数据从磁盘复制到主存。

一旦目标文件hello中的代码被加载到主存，处理器就开始执行hello程序的main程序中的机器语言指令。这些指令将“hello word\n”字符串中的字节从主存复制到寄存器文件，再从寄存器文件中复制到显示设备，最终显示在屏幕上。

**2.3 高速缓存至关重要**

hello程序的机器指令最初始存放在磁盘上的，当程序加载的时候，它们被复制到主存上；当处理器运行程序时，指令又从主存复制到处理器。相似地，数据串“hello word\n”初始时在磁盘上，然后复制到主存，最后从主存上复制到显示设备。这些大量的复制就是开销，减缓了程序“真正”的工作，因此，系统设计者的一个主要目的就是使这些复制操作尽可能快的完成。

根据机械原理，较大的存储设备要比较小的存储设备运行的慢，而快速设备的造价远高于同类的低速设备。加快处理器的运行速度比加快主存的运行速度要容易和便宜的多。针对这种处理器与主存之间的差异，体统设计者采用了更小、更快的存储设备，即高速缓存存储器（简称高速缓存），作为暂时的集结区域，用来存放处理器近期可能会需求的信息。

**2.4 存储设备形成层次结构**

在处理器和一个又大又慢的设备（例如主存）之间插入一个更小更快的存储设备（例如高速缓存）的想法已经成为了一个普遍观念。实际上，每一个计算机系统中的存储设备都被组织成了一个存储器层次结构，存储器层次结构的主要是一层上的存储作为低一层存储器的高速缓存。因此，寄存器文件就是主存的高速缓存，而主存又是磁盘的高速缓存。在某些具有分布式文件系统的网络系统中，本地磁盘就是存储在其他系统中磁盘上的数据的高速缓存。

3 计算机的程序语言（机器语言、汇编语言、高级语言）

用机器语言编写程序，编程人员要首先熟记所用计算机的全部指令代码和代码的涵义。手编程序时，程序员得自己处理每条指令和每一数据的存储分配和输入输出，还得记住编程过程中每步所使用的工作单元处在何种状态。这是一件十分繁琐的工作，编写程序花费的时间往往是实际运行时间的几十倍或几百倍。而且，编出的程序全是些0和1的指令代码。

缺点：直观性差，还容易出错。除了计算机生产厂家的专业人员外，绝大多数程序员已经不再去学习机器语言了。

为了克服机器语言难读、难编、难记和易出错的缺点，人们就用与代码指令实际含义相近的英文缩写词、字母和数字等符号来取代指令代码（如用ADD表示运算符号“+”的机器代码），于是就产生了汇编语言。汇编语言是一种用助记符表示的仍然面向机器的计算机语言。汇编语言亦称符号语言。汇编语言比用机器语言的二进制代码编程要方便些，在一定程度上简化了编程过程。汇编语言的特点是用符号代替了机器指令代码。而且助记符与指令代码一一对应，基本保留了机器语言的灵活性。汇编语言像机器指令一样，是硬件操作的控制信息，因而仍然是面向机器的语言，使用起来还是比较繁琐费时，通用性也差。汇编语言是低级语言。但是，汇编语言用来编制系统软件和过程控制软件，其目标程序占用内存空间少，运行速度快,有着高级语言不可替代的用途。

随着计算机事业的发展，促使人们去寻求一些与人类自然语言相接近且能为计算机所接受的语意确定、规则明确、自然直观和通用易学的计算机语言。这种与自然语言相近并为计算机所接受和执行的计算机语言称高级语言。高级语言是面向用户的语言，但计算机并不能直接地接受和执行用高级语言编写的源程序，源程序在输入计算机时，通过“翻译程序”翻译成机器语言形式的目标程序，计算机才能识别和执行。这种“翻译”通常有两种方式，即编译方式和解释方式，这里我们主要讲编译方式。编译方式是：事先编好一个称为编译程序的机器语言程序，作为系统软件存放在计算机内，当用户由高级语言编写的源程序输入计算机后，编译程序便把源程序整个地翻译成用机器语言表示的与之等价的目标程序，然后计算机再执行该目标程序，以完成源程序要处理的运算并取得结果。

高级语言与自然语言（英语）更接近，而与硬件功能相分离（彻底脱离了具体的指令系统），便于广大用户掌握和使用。高级语言的通用性强，兼容性好，便于移植。

4 计算机的存储器（虚拟内存技术）

在计算机系统中，CPU执行指令，存储器系统为CPU存放指令和数据。CPU寄存器存放着最常用的数据，靠近CPU的小的、快速的高速缓存存储器作为一部份存储在相对慢速的主存储器（简称主存）中的数据和指令的缓冲区域。主存暂时存放存储在容量较大的、慢速磁盘上的数据，而这些磁盘常常又作为存储在通过网络网络连接的其他机器的磁盘或磁带上的数据的缓冲区域。

磁盘是广为应用的保存大量数据的存储设备，存储数据的数量级可达到几百到几千千兆字节，而基于RAM的存储器只能有几百或几千兆字节。从磁盘上读信息的时间为毫秒级，从DRAM读比从磁盘读快10万倍，从SRAM读比从磁盘读快100万倍。磁盘是由盘面构成的，每个盘面有两面或者称为表面，表面覆盖着磁性记录材料。盘面中央有一个可以旋转的主轴，它使得盘面以固定的旋转速率旋转，通常是5400~15000转每分钟。磁盘通常包含一个或多个这样的盘面，并封装在一个密封的容器内。整个装置通常称为磁盘驱动器，简称磁盘。

虚拟内存别称虚拟存储器。电脑中所运行的程序均需经由内存执行，若执行的程序占用内存很大或很多，则会导致内存消耗殆尽。为解决该问题，Windows中运用了虚拟内存技术，即匀出一部分硬盘空间来充当内存使用。当内存耗尽时，电脑就会自动调用硬盘来充当内存，以缓解内存的紧张。若计算机运行程序或操作所需的随机存储器(RAM)不足时，则 Windows 会用虚拟存储器进行补偿。它将计算机的RAM和硬盘上的临时空间组合。当RAM运行速率缓慢时，它便将数据从RAM移动到称为“分页文件”（指操作系统反映构建并使用虚拟内存的硬盘空间大小而创建的文件）的空间中。将数据移入分页文件可释放RAM，以便完成工作。 一般而言，计算机的RAM容量越大，程序运行得越快。若计算机的速率由于RAM可用空间匮乏而减缓，则可尝试通过增加虚拟内存来进行补偿。但是，计算机从RAM读取数据的速率要比从硬盘读取数据的速率快，因而扩增RAM容量（可加内存条）是最佳选择。

5 指针（动态存储分配）

**5.1 指针**

如果想要去图书馆借一本《java从入门到精通》，这件事情我想不会难倒读者，你会进入图书馆，先找到理科类阅览室，进入阅览室，用电子查询器输入书名，显示书的编号是JSJL-2234O，按照编号就可以从相应的书架上找到该书了，而且是准确无误的你想要的那一本书。

同样在计算机系统里面，利用书的编号同样对应着一项技术，叫做指针。指针利用地址，直接指向存放在电脑存储器另外一个地方的值。

#include <iostream>

int main()

{

int a,\*p;

a=100;

p=&a;

printf("a=%x &a=%x &p=%x p=%x \*p=%x\n",a,&a,&p,p,\*p);

return 0;

}

上述程序就是输出变量a所在内存中的地址。

**5.2 动态存储分配**

在程序代码中，编译器为变量分配存储空间时，根据变量的类型知道所需内存的大小，从而系统为它们适当地分配存储空间，这是静态存储分配；有些时候只有在程序运行的时候，才能知道变量所需的内存空间大小，系统运行时会根据要求进行分配，这就是动态存储分配。

当程序需要一个动态分配的变量，此时必须向系统申请取得堆中的一块所需的内存空间大小，用来储存变量。当不再使用变量时，要释放它所占用的内存空间大小，为其他变量的使用，再次分配空间大小，这样重复利用有限内存资源。

6 嵌入式系统

嵌入式系统是一种“完全嵌入受控器件内部，为特定应用而设计的专用计算机系统”嵌入式系统为控制、监视或辅助设备、机器或用于工厂运作的设备。它的出现最初是基于单片机，随着对实时性要求的提高，软件规模不断壮大，嵌入式系统满足应用系统的功能、可靠性、成本、体积等要求，拥有系统内核小、专用性强、系统精简等特征。

在我们的日常生活，随着科学技术的发展，嵌入式系统的应用随处可见，智能洗衣机逐渐替代了双桶洗衣机，省电、省水、使上下翻滚的衣服由中心向四周扩散，洗净均度高达99.3%，全面呵护衣物。在学习阶段，我们接触了单片机——Arduino ,将编写好的代码传送在单片机上，可以控制发光二极管的亮度，发光时间，还可以做温湿度感应器，舵机等等。

[参考文献]

**BRYANT R E，HALLAREN D R O [计算机科学丛书：深入理解计算机系统 （原书第二版）] 机械工业出版社，2011**

**Github 账号 shenchunyang**