intel

Intellec 系列 II 微型计算机开发系统 指南

科学技术文献出版社重庆分社

Intellec系列Ⅲ
微型计算机开发系统指南

黄焕运、郭秋卉、张效牛、杨尚农　译
科学技术文献出版社重庆分社　出版
重庆市市中区胜利路91号
新华书店重庆发行所　发行
科学技术文献出版社重庆分社印刷厂　印刷

开本：787×1092毫米1/16　印张：6.75；字数：18万
1985年10月第一版　1985年10月第一次印刷
科技新书目：109—250　印数：3400

书号：13176·146　定价：1.30元

前　言

本书是Intellec系列Ⅲ微机开发系统的一本基础读物，也是微机应用中软件开发方面的一本入门书。稍有微处理机及其应用的基础知识的读者都能读懂本书，而且并不要求读者了解任何特定的程序设计语言。本书不仅对初学者有用，甚至这方面有丰富经验的人浏览一下也会有所收获。

本书是一本使用系列Ⅲ系统的指导性书籍，尤其对该系统的8086执行环境的使用有指导作用。相应于系列Ⅱ的8085执行环境已在Daniel McCrackeh 著的《Intellec 微机开发系统指南》中论述。

本书以微机在建筑物空调系统的应用为例，引导读者从头到尾了解一个典型的软件开发过程。为使例子易于理解，我们只介绍软件开发工作，而假定同时进行着相应的硬件开发工作。实际上，所列举的某些软件开发课题，就是改变硬件设计造成的。

本书共分七章。第一章给出系列Ⅲ开发系统和应用例子的概貌，介绍了自顶向下设计法、逐步细化步骤、模块化编程、设计考虑，并说明了如何对每个模块选择合适的软件语言。第二章介绍系列Ⅲ操作系统的操作，讲解典型操作步骤。第三章介绍文本编辑程序CREDIT的使用，还说明应用例子中的主要控制算法的逐步深化过程。第四章叙述Pascal-86编程、结构化和模块化设计、参数传送、数据类型化和Pascal-86编译程序。第五章讨论PL/M-86编程，给出一个用在应用例子中的示范性的PL/M-86例程，并简要地介绍PL/M-86编译程序和8086/8087/8088宏汇编程序。第六章讨论利用实用程序产生可执行程序的有关问题。第七章介绍用DEBUG-86进行程序调试，并使用ICE-88仿真器进行硬件仿真。

书后给出了有关的参考书目，并列出了Intellec系列Ⅲ微机开发系统资料清单。

目　　录

第一章　软件开发过程……（1）

1. 产品软件的确定……（2）
2. 软件开发工具的选择……（3）
3. 语言……（4）
4. 模块化编程……（5）
5. 调试和在线仿真……（5）
6. 最终产品的使用……（6）

第二章　系列Ⅲ系统的操作……（7）

1. 系统加电……（7）
2. 显示目录表……（8）
3. 格式化磁盘……（10）

(1) 硬盘子系统用户……（10）

(2) 软盘用户……（11）

4. 文件名、路径名和文件属性……（12）
5. 文件的换名和删除……（15）
6. 复制文件到磁盘和外设……（15）
7. 命令和程序的执行……（18）
8. 系列Ⅲ操作系统小结……（21）

第三章　文本编辑……（22）

1. 文本文件的生成和文本的插入……（22）
2. 在文本文件内移动……（26）
3. 寻找老文本和代之以新文本……（27）
4. 宏命令和命令重复……（29）
5. 文本编辑会话的结束和后备文件的管理……（31）
6. 文本文件的显示和打印……（32）
7. 从文本到程序……（32）

第四章　用Pascal-86编程 …… (33)
1. 杂语Pascal翻译成 Pascal-86 …… (33)
2. Pascal-86数据类型 …… (37)
3. 模块化和信息掩蔽的另一观点 …… (38)
4. 把数据传送到其它模块——参数传送技术 …… (40)
5. 接口说明 …… (41)
6. 空调系统的测试程序 …… (43)
7. Pascal-86 编译程序 …… (47)
8. 小结 …… (58)
第五章　用其它语言编程 …… (59)
1. 为模块选择语言的另一观点 …… (59)
2. 用 PL/M-86 编程 …… (59)
3. 用 8086/8087/8088 汇编语言编程 …… (64)
4. 系列Ⅲ环境的编程 …… (71)
第六章　用实用程序准备可执行的程序 …… (72)
1. 准备程序模块库 …… (72)
2. 连接模块以形成可定位的程序 …… (73)
3. 程序的定位和运行 …… (74)
第七章　调试程序和执行程序 …… (78)
1. 用DEBUG-86进行符号调试 …… (78)
2. ICE-88在线仿真器的使用 …… (94)
3. 执行环境 …… (98)

※　※　※

参考书目 …… (99)
Intellec系列Ⅲ资料清单 …… (101)

第一章　软件开发过程

"硬件拥有计算潜力，但必须由软件控制才会有用"。

——Intel公司总经理Andrew S. Grove

Intel系列Ⅲ微机开发系统不止是由键盘、显示器、磁盘机和内有两台微处理器的机箱组成的装置，它是设计供iAPX86、88处理器系列或8080/8085处理器用的微机软件的有用工具。用户可选择适当的语言（PL/M、FORTRAN、Pascal、宏汇编语言）来写软件的各部分程序，然后分别调试这些程序，最后根据不同的使用要求以不同的方式把它们链接起来。这个最终得到的应用程序，能够在系列Ⅲ或是在由iAPX86/88或8080/8085系列构成的其它系统上运行。

Intel iAPX微处理器系列的体系结构非常适合于用高级语言按模块进行软件开发，Intellec系列Ⅲ微机开发系统充分利用了这种体系结构的优点，提供了既经济又实惠的编程环境，从而缩短了软件开发周期。

为了设计一个含有微处理器的产品，必须协调两项设计工作，即以微处理器为核心的硬件的设计和控制微处理器的软件的设计。硬件开发工作包括确定微处理器，存贮器和外围电路，以及专用I/O电路和处理器等之间的相互关系。软件开发工作包括用指令编写微处理机的程序。这些指令最后得存放在产品的存贮器里。当然，这些指令必须设计得能正确完成所要求的任务。

分别进行硬件和软件的开发是可行的。但实际上开发一个在样机上执行起来没有错误的软件要花很长的时间。为了得到一个完整的系统和节省时间，就必须早在样机能用来测试软件之前便开始软件的调试。

Intellec系列Ⅲ有在线仿真器(ICE)，它能够支援并行地进行软、硬件的开发，从而解决了上述问题。利用ICE-86或ICE-88仿真器，用户能仿真样机的各硬件部分，以便在一个与最终产品类似的、稳定的环境中进行软件测试。ICE-86或ICE-88仿真器还允许用户用系列Ⅲ系统的存贮器和其它资源来代替样机中还没有的存贮器和资源。这样，借助ICE-86或ICE-88仿真器，用户在设计的过程中，就可以把他的样机硬件纳入产品，软件和硬件亦可同时测试，因此加快了整个开发过程。

图1-1中的流程概括了从对最终产品仅有一个设想开始的软件开发过程。这样一个过程总是从设想开始，然后逐步深化，直到能确定实际的产品的环境、硬件和逻辑为止。

为使用户了解怎样使用Intellec系列Ⅲ及说明Inter的软件开发工具如何用于开发，我们提供了一个以8088微处理机为核心的iAPX88微系统的软件应用实例，这个应用是一个建筑物的空调系统，它使用太阳能集热器加热和致冷，当太阳能集热器不够用时，还具有其它辅助加热和致冷手段，如使用水、贮水罐、水—空气换热器及热力泵。

关于这个空调系统的设计决策将在后面叙述。例如，用模块方式设计系统软件，我们以后就能按需要增加更多加热和致冷手段，以及决定如何去控制水泵和阀门。现在要知道的是：软件的基本作用是要根据温度数据来选择一种加热和致冷的方法，同时去控制空调系统中泵和阀门的工作。图1-2是此应用的一个简化框图。

1. 产品软件的确定

在确定产品的硬件时，用户也要确定其软件的目的。例如，这个应用软件的目的是要收

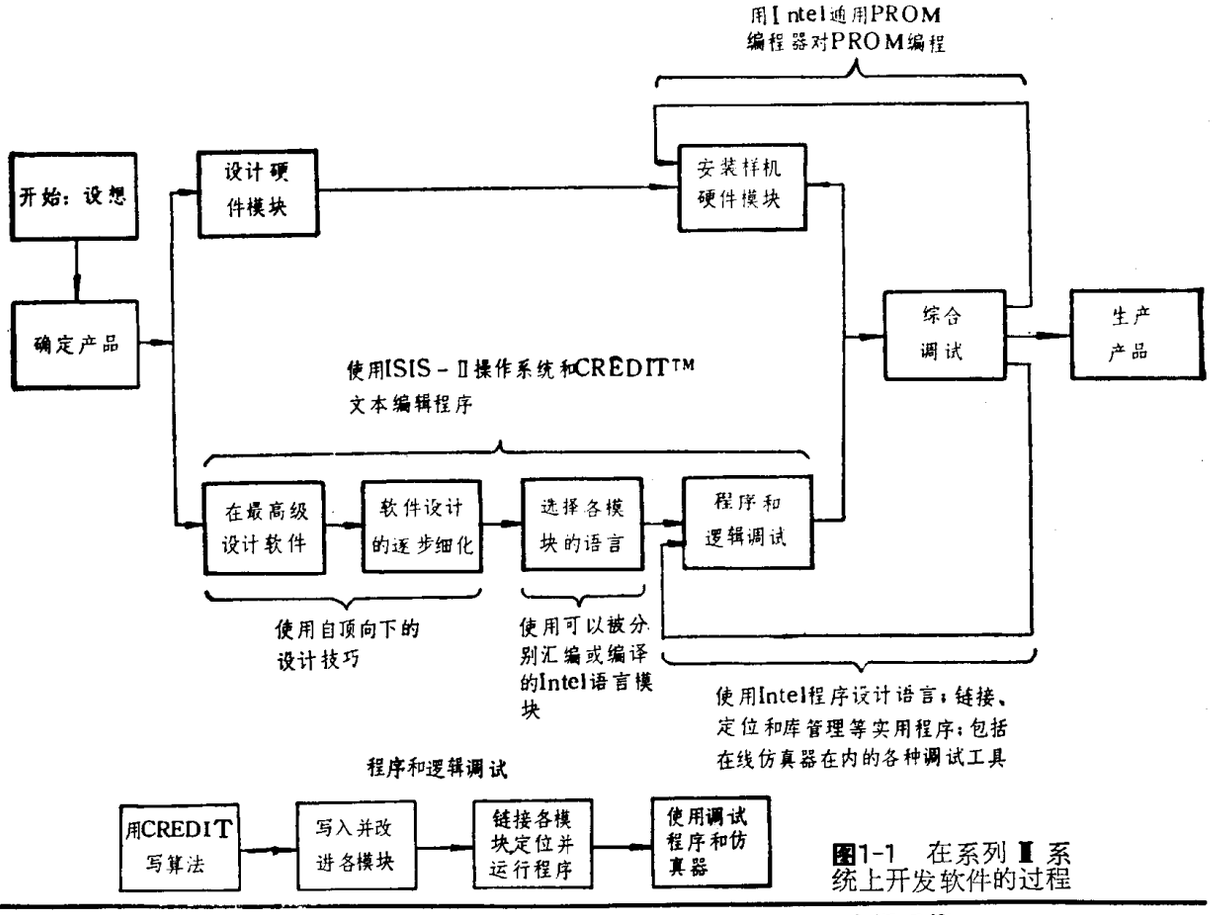


图1-1 在系列Ⅱ系统上开发软件的过程

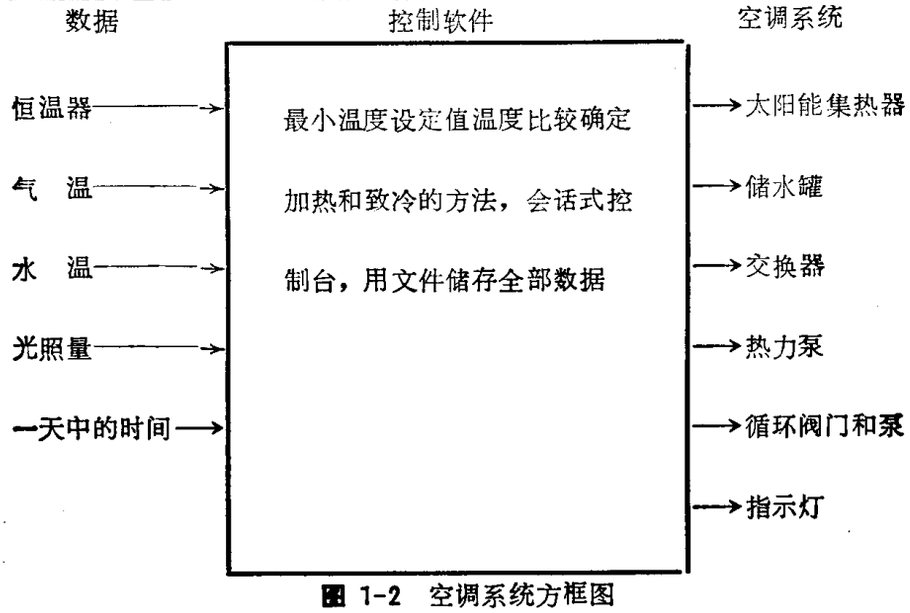


图 1-2 空调系统方框图

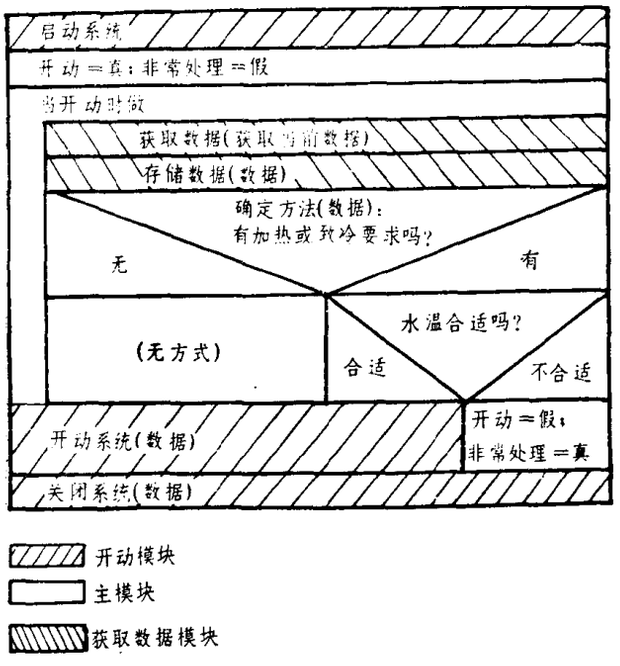
集和存贮相应的温度数据，根据这些数据决定加热和致冷的方法，在空调系统中执行之，并维持空调系统的运转。每一个任务可设计成一段程序，称为模块。将任务模块化后，用户可以改变任一任务的细节，而不会影响其它任务的细节。

要记住：软件有着改变和添加产品性能的能力。整个产品似乎是由硬件和逻辑电路组成的，然而，用户可能要对各个部件重新组合，达到增加功能或改变逻辑流程的目的。软件若

不模块化且易于维护，就不能解决这些问题。因此，用户在确定软件的整个目的时，应考虑到产品的进一步发展。在使软件模块化时，应做到能很容易地替换一些模块，而无需去重写那些有效的模块。

对于本空调系统，我们确定软件由一组模块组成，这些模块执行获取和存贮数据的操作，根据这些数据来决定加热或致冷的方式，最后决定怎样开动空调系统中的有关硬件。

现在我们还不作更详细的说明，因为详细说明反而有碍于逐步深化。重要的是了解这些动作的顺序：首先，必须用软件来启动空调系统。系统启动后，软件得反复做几件事（除非系统停止工作）：(1)读各种温度数据，(2)存贮这些数据以供参考，(3)确定加热和致冷方式，(4)开动空调系统加热或致冷，并维持这个系统（例如，维持增温）。



加热方式	数据
集热器至交换器	内部温度
罐至交换器	恒温器设定值
集热器至热力泵	集热器水温
罐至热力泵	罐的水温
加热过的罐至热力泵	加热后罐的水温
无方式（无加热要求）	

图1-3 空调控制软件的Nassi-Schneiderman图

一个好的软件设计方法是，把总任务分成若干能解决的任务。这些任务必须出现的次序就确定了软件的结构。如果结构很容易理解，那么它就越容易实现和维护。

通常用图1-3所示的Nassi-Schneiderman图表示软件的结构块。使用你发现有益的任何图，但要撇开许多细节，以便不致局限于用某一种方法来解决问题。这里，我们撇开了关于数据类型、输入/输出、实际的致冷或加热方式等许多细节。最重要的是：设计的每个模块应该是独立的，也就是说，一个模块不必知道其它模块中的细节，特别是那些将来可能要改变的细节。

2．软件开发工具的选择

在目前市场上极为缺乏软件开发工具的情况下，所买到的软件开发系统往往不合乎要求。有些系统要求用户将整个程序放在系统所允许的有限空间内，而且系统不能提供建立程序库并将它链接至不同程序的手段。

Intellec系列Ⅲ系统提供了将各部分程序链接在一起的能力，也提供了建立程序库并将它链接至不同程序的手段。模块概念是此系统所固有的。只要用户建立一个部分程序，它就是一个模块；这个模块能引用存于程序库中其它模块的过程、函数和变量。

Intellec系列Ⅲ系统支持几种语言，其中包括高级语言PL/M-86、Pascal-86和Fortran-86。对本应用而言，我们不必确定只使用一种编程语言。可是我们要决定是否使用系统中已有的模块，并利用此决定进行应用设计，或者以后改变决定，建立其它的模块来取代现有的模块。

例如，我们已有一个用PL/M-86编写的模块，这个模块执行将热电偶电压转换成摄氏温度的例程。我们暂定当前使用此例程，这样，由于用不着将这个例程编进主程序，因而就节省

了时间。我们还决定以后不用这个例程，并用我们自己的模块来取代它。直到主程序完成以后，我们再确定链接哪一个模块。

如果选择的工具合适，用户就能够节省研制时间，也能暂时不管一些特定的细节，等到以后要确定这些细节时再作出决定。推迟这些决定，软件开发工作就避免了因严格的设计决定而造成的极为杂乱的情况，而且也能使开发工作灵活地适应新的变化。

合适的工具是：(1) 有相应的高级语言可供选用；(2) 具有管理程序库的方法；(3) 有链接程序，使用户能链接已完成（或未完成）的各种模块，这些模块是为不同应用以不同语言编写的；(4) 有定位程序，将用户程序在存贮器中定位，而且能让用户规定程序地址；(5) 有符号调试程序，使用户易于对模块和部分程序进行调试；(6) 有在线仿真器，使在实际部件提供使用之前，能对其进行仿真。Intellec 系列Ⅲ微机开发系统提供了上述所有的工具。

3. 语 言

使用自顶向下的方法设计一个系统时，为了揭示逻辑原理和避开细节问题，用户在对方案深化时，每一步都要按可允许的最高级语言进行思考和用它来表达概念。

如何判断你是否在按"高级"语言进行思考呢？假如你采用的语言能确定软件的总体结构的话，那么这个语言对有关应用是"高级"的。低级语言用于详尽地描述程序的各个部份。借助高级语言，用户能够在最高级获得系统控制结构的清晰轮廓，也能发现控制结构中存在的造成程序中有细小差错的某些逻辑错误。记住这些优点后，你就会有意识地采用最高级语言——英语来编制你的程序。当用英语确定了软件的总体结构后，你可用一种与英语相似的高级编程语言——Pascal来更详细地表达程序。

在第三章中，我们逐步地对温控算法进行了深化，用的是一种称为杂语 (Pidgin)Pascal 的语言，它实际上是一种简洁的说明性英语语句的语言。由于控制结构很容易从英语翻译成 Pascal，所以我们决定将Pascal-86用于温控主模块（这不是最终的决定）。直到对杂语Pascal算法进行逻辑检查之后，我们才开始将英语翻译成Pascal-86。

Pascal是一种类似人们思维支配结构的语言。人的思维通常并不按照GO TO转移进行思考，而是将一项作业看成是一组任务，当某些事件是真时才执行这些任务 (DO WHILE)，或一直执行到某事件做完为止 (DO UNTIL)。假如 (IF) 某事件是真或假，则 (THEN) 做一件事，否则 (ELSE) 做另一件事。当问题有几个不同情况 (CASE) 时，依据情况解决各个问题。有时，我们可能需要从故障中解脱出来 (GO TO到一个紧急处理例行程序)，而且在设计算法时，我们要很仔细地考虑这些异常情况。

值得指出的是：我们要把系统的控制结构视为结构，而不要看作是一些个别的转移语句。Pascal是一种能表述控制结构的语言；PL/M是另一种这样的语言，有些新版的 Fortran也能相应地结构化。

另一个问题现在已很明显，即用户应该选择一个适合相应算法的最好语言。例如，开动空调系统的模块必须操纵泵和阀门，完成选定的加热或致冷操作。这个模块从用 Pascal语言编写的决策主模块得到数据，并用一个字中的某些位作为控制信号，传送给实际与空调系统硬件接口的过程。

我们也可用汇编语言或PL/M语言编写这个开动模块，因为这两种语言容易对一个字节

或一个字中的位进行处理，并根据这些位组合格式相应地进行动作。我们以后再作出最终的决定，同时，在空调系统样机硬件一切就绪之后，才用一个测试性的开动模块。此模块用Pascal语言编写，它仅在控制台上显示些相应的测试信息。

这个开动模块的最后版本很可能不用Pascal语言编写，因为Pascal语言不能提供位处理操作。可以猜测，这个模块也不用Fortran语言编写，因为虽然Fortran表达复杂的算术公式的能力比PL/M语言强，但空调系统中没有用到复杂的算术公式。

如果应用中需要最有效地利用处理机的时间和存贮器，就要用汇编语言编写开动模块的最终版本。由于PL/M语言容易学，而且所写的程序易读和易于维护，所以用PL/M语言编程能够节省开发和维护的时间。PL/M语言的唯一缺点是它不能有效地利用处理机的时间和存贮器，也不能进行十进制和实数运算*。因为本应用中不需要用十进制和实数运算，速度和存贮容量也没有严格限制，因此并不影响我们把PL/M语言用于这种应用。这样，从减少开发时间和维护费用出发，用PL/M语言编程的优点超过了汇编语言。

4. 模块化编程

Intellec系列Ⅲ系统有链接和定位的工具来支持模块化编程，并有库实用程序来维护例程库。虽然我们所作出的决策不相互牵扯在一起，但它们也是很有用的。现在我们能决定将哪些类型的程序划到相应模块里。要做到这一点，必须遵循下述两条设计准则：(1) 写一个模块的程序时，可基本不了解其它模块的程序；(2) 重新装配和取代任何模块不影响其它模块。空调系统中各模块的设计决策是彼此独立的，因而这些决策不相互牵扯在一起。

我们根据这些准则对系统进行设计。系统由下述模块组成：获取和存贮数据的获取数据模块（GetData模块），完成实际的系统操作（如开、闭泵和阀门等）的开动模块（Operation模块），完成高级决策的主模块（Main模块）。

到现在为止，唯一相互有牵扯的决策是在模块间传输数据。我们想使传输的数据量减到最少，或者搞一个较容易遵守的数据传输标准。在空调系统中，我们只需要传送一个数据记录的引用信息给其它模块；其它模块则必须知道用这个信息去做什么。我们能用的这种数据传输技术有如下两种：传送引用信息和传送数值。这一问题将在第四章加以详细讨论。

利用Intellec系列Ⅲ系统的功能，我们可对同一模块写出不同的版本，再对各个版本进行调试，然后决定哪一版本与其它模块相链接。我们也可在样机硬件与在线仿真器（ICE-86或ICE-88）一起调试后，再确定这些模块的实际存贮单元（在最终产品中的存贮单元）。在一些应用中，用户不必考虑实际存贮单元，而由Intellec系列Ⅲ的定位程序解决这个问题。

5. 调试和在线仿真

在软件开发期间，用户随时都能利用DEBUG-86调试一个已编译好的模块。有时，用户要将该模块改为独立的模块。例如，主模块需要从GetData模块得到相应的数据，而GetData模块还未编写好。我们可以很快地写一个过程，利用它从控制台上通过对话方式得到数据，

*译注：指PL/M-80而言，PL/M-86有相应运算能力。

然后我们把这个暂时的数据采集模块链接到主模块中，并利用DEBUG-86来调试主模块。

当模块写好并编译后，可以利用ICE-86或ICE-88以对话方式对模块进行测试。这两种仿真器能仿真最终产品中的处理机。如果最终产品中有8086处理器，就使用ICE-86仿真器；如果产品中有8088处理器，则应用ICE-88仿真器（记住：iAPX86、88应用软件能在8086也能在8088上运行）。

例如，数据模块要从最终产品中的8088处理器的端口中读数据，而在有最终产品的样机以前，ICE-88仿真器能仿真这些端口。同时，用户也可在有任何样机硬件之前，利用ICE-88开始软件调试工作。在样机部件完成后，用户就能使用已完成的部件，同时通过仿真器从系列Ⅲ系统借用象存贮器等这样的资源。

利用在线仿真器，用户就能在自己的环境中，或是在一个仿真最终产品环境的稳定环境中，对其产品加以控制、查询、修正，直至完全调试好产品。符号调试是Intellec微机开发系统和在线仿真器的主要特点之一。符号调试允许用户使用程序中的符号和行号来调试程序，而无需把符号转换成实际的存贮器地址。

6. 最终产品的使用

Intel公司还提供了一些工具帮助用户将最终产品组合成一个整体。如果产品的程序要写入ROM中，可使用Intel的通用PROM写入器（UPP）和它的通用PROM映象程序（UPM）来作成存放程序的PROM芯片，然后把这个片子插入SDK-86或SDK-88（8086或8088处理器系统设计套件），或插入iSBC（单板机）系统中。

用户程序既能在特定的应用环境中，也可在其它的系统中运行。例如，用户首先把程序送入SDK-86、SDK-88或iSBC 86/12A（有一个86/12A板的单板机系统）上的RAM中，然后用OH86实用程序（见《8086开发系统iAPX86/88系列实用程序用户指南》）把程序转换成16进制目标格式，最后使用适当的工具（在线仿真器ICE-86、SDK-C86软件和电缆接口、或iSBC957接口和执行软件包）将它装入执行板里。

用户也能使用系列Ⅲ开发系统作为最终软件产品的环境。系列Ⅲ开发系统有一个8086处理器（称"8086方面"或8086执行环境），能运行iAPX86、88应用程序。只要用户调试完了他的软件，就可随时把它放在任何系列Ⅲ系统中运行。第六章介绍用以使Pascal-86程序在系列Ⅲ环境中运行的运行时间库；用户要在其它执行环境中运行这些程序时，应自行编制这些库程序。第六章还介绍用户怎样能将模块链接起来并将它们在系列Ⅲ中定位，使之准备好在系列Ⅲ系统上执行。总之，无论用户为其最终产品选择什么样的环境，Intel都能提供合适的硬件和软件工具来开发、调试和制作出你的最终产品。Intel环境适合于最终的软件产品，它还可以根据软件的进一步发展加以改进提高。

第二章　系列Ⅲ系统的操作

"最终会证明：使用一个标准化的操作系统所得到的好处，同使用标准化的微机硬件一样。这样能极大地降低开发和编程的费用，同时对将来的产品有一个向上兼容的接口"。

——Intel公司总经理Andrew S. Grove

如第一章所述，Intellec系列Ⅲ微机开发系统有软件开发所需的全部工具。为了使用这些工具，用户必须学会操作这个系统，也就是要会使用系统的命令和实用程序。

为指出对系统编程和使用系统之间的差别，用一台多用户的大型计算机系统进行类比更好理解。一个运行着的系统通常同时支持这两项活动。在这样一个大型系统中，一个或多个程序员可能正在设计程序，以便在系统上运行，一个或多个用户（也可能是程序员）可能只是使用那些为系统已开发好的程序。显然，一个只使用系统的用户，并不需要知道有关系统的详细情况；反之，一个编程序的程序员则需要知道这些详细情况，以便为系统编程序。

系列Ⅲ系统有着大型系统的大部分功能，但它一次只允许一个人使用。这个用户能使用这个系统和它的程序，或为该系统编制程序。

这一章主要讲如何使用系列Ⅲ系统，其内容对使用系统和为它编制程序的人来讲都是重要的，并且对新用户来讲既实用，又不繁琐。为使其容易阅读，在讲述系统命令和实用程序时，避开了复杂的细节。要了解系统命令的详细情况，请参见《Intellec系列Ⅲ微机开发系统控制台操作说明书》。

1. 系统加电

每一台微型机都有一个称为操作系统的东西，操作系统有时也称为管理程序或监控程序。它通常是用户要使用的第一个软件，即系统加电后控制着计算机并准备响应用户打入的第一条命令的软件。如，用户打一条命令要计算机列出磁盘上的文件，或者执行一个具体的程序。

给Intellec系列Ⅲ微机开发系统加电后，下面的信息出现在显示屏上：

SERIES II MONITOR, V*x.y*

此信息表明监控程序已经启动，并且正在运行（这里*x*和*y*表示版本号）。监控程序是一种软件，它让用户能直接控制系列Ⅲ硬件。虽然监控程序可用于调试和其它一些操作，但其主要作用是完成一个快速的动作：按RESET键后，它从0号驱动器的系统盘上把操作系统装入计算机。

为了装入或"引导"系统，需要有一个系统盘，即含有操作系统文件的磁盘。Intel提供了一个标有"ISIS-Ⅱ Operating System"（ISIS-Ⅱ操作系统）的软盘，它就是用户系统盘。用户首先要执行的是格式化操作，以生成一个系统盘的复制品。

用户先将"ISIS-Ⅱ Operating System"盘插入0号软盘驱动器，然后按RESET键（更详细的说明见《Intellec系列Ⅲ微机开发系统控制台操作说明书》），显示屏上将显示下面的信息：

ISIS-Ⅰ，V*x.y*

Intellec系列Ⅲ微机开发系统的操作系统称为ISIS-Ⅱ〔"ISIS"即Intel系统实现管理程序的缩写〕，*x*和*y*表示版本号。ISIS-Ⅱ实际上是一个运行在老的Intellec系统（Intellec微机开发系统和Intellec系列Ⅱ微机开发系统）上的ISIS操作系统版本。

ISIS-Ⅱ管理着系统配备的各种软件工具。用户能用ISIS-Ⅱ运行一些实用程序，如运行CREDIT（文本编辑程序）或LINK86，前者用于写程序，后者用于链接各程序模块成为一个最终的程序。用户还可使用ISIS-Ⅱ来运行Pascal-86、PL/M-86编译程序，以及用户自已编制的程序。另外，用户还能使用ISIS-Ⅱ命令复制程序和控制连到系统上的设备。

大多数计算机操作都以文件作为对象。文件是信息的集合。每个文件都有个名字，称为文件名。在介绍了最一般的操作系统命令DIR后，我们将详细地说明文件和文件名。

DIR命令用于显示某一指定盘上文件名的目录。另一些命令则执行文件管理操作，例如，使用COPY命令复制文件或把文件复制品送给打印机打印出来；使用RENAME命令改变一个文件的文件名。这一章仅介绍这些命令中的一部分。

2. 显示目录表

要介绍的第一个命令是最容易使用的命令——DIR命令。当系统启动投入运行时，在显示屏的左边能看到提示符（-），它告诉用户：系统已做好接收下一条命令的准备。现在，用户可通过按字母"DIR"打入DIR命令，然后按RETURN键执行DIR命令。下面的例子中，凡是用斜体字表示的部分都是用户输入的，其余的是系统显示的内容，符号〈cr〉表示要按RETURN键（"cr"代表"回车"键，在大多数打字机上都有）。

```
-DIR〈cr〉

DIRECTORY OF :F0:970003.06
NAME    .EXT BLKS LENGTH ATTR NAME .EXT BLKS LENGTH ATTR
SYSTEM  .LIB  24    2849    WS   FPAL .LIB  74    9125    W
PLM80   .LIB  45    5615    W
                            143
1317/4004 BLOCKS USED
```

执行DIR命令时，系统显示一个称为目录表的文件名表。这些文件名是存放在0号驱动器磁盘上的文件的文件名。若DIR命令中未指定其它的驱动器号，则显示0号驱动器上盘的目录，这点下段中还要讲到。0号驱动器通常放置系统盘（存有系统文件的盘），所以执行DIR命令，便得到了0号驱动器上盘的部分文件的目录表（不是全部的，因为有些文件是不可见的）。

用DIR命令指定驱动器号后，用户便能显示其它驱动器上的盘或硬盘子系统的目录表。例如，要看1号驱动器上盘的目录，把"CREDLT ISIS-Ⅱ CRT-Based Text Editor"（ISIS-Ⅱ CRT为基础的文本编辑程序CREDIT）盘插入1号驱动器，然后打入下面的命令：

```
-DIR 1〈cr〉
DIRECTORY OF :F1:970049.02
NAME .EXT BLKS LENGTH ATTR    NAME    .EXT BLKS LENGTH ATTR
CREDIT      156         19470 CREDLT .HLP 25    2985    W
```

```
ADDS .MAC 3 171 W                MICROB .MAC 3 158 W
VT52  .MAC 3 163 W               VT100   .MAC 3 190 W
1510T .MAC 3 165 W               1510T   .MAC 3 170 W
LEAR .MAC 2 123
                    201
310/4004 BLOCKS USED
```

《Intellec系列Ⅲ微机开发系统控制台操作说明书》详细说明了硬盘和软盘的典型配置情况以及相应的驱动器号。

目录表的每一行中，给出了文件的名字，有些文件名后还有用三个字符表示的扩展名。这是一个任选的标识符，用于说明文件的类型。有时，在某些文件与特定的程序一起使用的情况下，便要求有扩展名。在本章第四节中将对一些扩展名加以说明。

在每个文件名项的后面，有三个分别标为"BLKS"、"LENGTH"和"ATTR"的列。"BLKS"这一列告诉用户，文件占用了多少"块"存贮空间，这里每一"块"为128个字节。"LENGTH"这一列给出了文件实际占用的字节数。在目录表底部的一行信息给出了该磁盘可用总块数中已用掉的块数。用户可用块数来描述文件的大小和确定一个盘是否能放一个已知长度（按块计）的文件。

"ATTR"这一列可以是空白，也可以包含"W"、"S"或两者均有。这一列给出了每个文件的属性，即决定文件使用的某些性质。如果一个文件有"W"属性，定便是写保护的，也就是说不能对该文件写入新内容，也不能删除（或重写）这个文件。如果一个文件有"S"属性，它便是一个系统文件。系统文件出现在大部分系统盘上。用户能使用ATTRIB命令和文件属性来保护自己的文件，以免受不适当的删除或写入操作的影响，也能用它们指定某些文件为系统文件(在下段说明)。

有些文件不能用一般的DIR命令来显示。这些文件是不可见的，也就是说它们有I属性。如果要看这些文件，应使用一种特殊形式的DIR命令：

-DIR I⟨cr⟩

DIRECTORY OF :F0: 970003.06

NAME	.EXT	BLKS	LENGTH	ATTR	NAME	.EXT	BLKS	LENGTH	ATTR
ISIS	.DLR	26	3200	IF	ISIS	.MAP	5	512	IF
ISIS	.TO	24	2944	IF	ISIS	.LAB	54	6784	IF
ISIS	.BIN	94	11740	SIF	ISIS	.CLI	20	2407	SIF
ATTRIB		40	4909	WSI	COPY		69	8489	WSI
DELETE		39	4824	WSI	DIR		55	6815	WSI
EDIT		58	7240	WSI	FIXMAP		52	6498	WSI
HDCOPY		48	5994	WSI	HEXOBJ		34	4133	WSI
LDISK		63	7894	WSI	FORMAT		62	7794	WSI
LIB		82	10227	WSI	LINK		105	13074	WSI
LINK	.OVL	37	4578	WSI	LOCATE		120	15021	WSI
OBJHEX		28	3337	WSI	RENAME		20	2346	WSI
SUBMIT		39	4821	WSI	SYSTEM	.LIB	24	2849	WS
FPAL	.LIB	74	9125	W	PLM80	.LIB	45	5615	W

1317

1317/4004 BLOCKS USED

上述那个"I"称为开关，它用来显示那些具有不可见属性的文件。在一个DIR命令中规定了"I"开关后，该DIR命令显示所有的文件，包括有"I"属性的文件在内。在执行一般DIR命令时不显示的文件，现在连同它们的属性（包括"I"属性）一起被显示出来了。"F"属性留给使磁盘格式化的系统文件使用，这些文件称为格式文件，用户绝对不能改变它们的属性。

用户可以练习一下DIR命令的使用，同时检查一下随机所带磁盘的目录表，这只要把软盘逐个地插入1号驱动器，并打入"DIR I"即可。

注　意

如果用户有一个硬盘子系统，而只有一个软盘驱动器，则使用下面方式的DIR命令：

-DIR P ⟨cr⟩

LOAD SOURCE DISK, THEN TYPE⟨cr⟩

（装入源盘，然后打⟨cr⟩）

这时拿出系统盘，插入要显示目录的盘，然后按RETURN键。在列出目录表后，会出现下面的信息：

LOAD SYSTEM DISK, THEN TYPE⟨cr⟩

（装入系统盘，然后打⟨cr⟩）

这时又要将系统盘放回软盘驱动器。为检查其它盘的目录表，对每一个盘都要重复这些步骤。

3. 格式化磁盘

在这一节中，我们介绍将Intel公司提供的文件副本存放到硬盘或软盘上的一般方法。本书后面的例子中，假设某些程序驻留在0和1号硬盘驱动器上，或0和1号软盘驱动器上。用户可以按自己的想法将文件分存于各个盘上，但为使下面的例子有典型性，用户必须按这里给出的方法去做。如果懂得了本章所讲的路径名的使用，也能正确地使用COPY命令，用户便能按自己选定的任意分配方案把文件复制到盘上，并且只要代之以自己指定的路径名，仍能使用书中的例子。

如果用硬盘子系统，用户必须按照《Intellec系列Ⅲ微机开发系统控制台操作说明书》中所述的步骤去配置磁盘和使它们做好使用准备。如果用软盘驱动器，应该参考上述手册关于软盘的保护和插入等说明部分。这一节讲述怎样将一个硬盘片或一张软盘格式化成系统盘，将其它的硬盘或软盘格式化成非系统盘（用于存贮用户文件或其它程序）。

（1）硬盘子系统用户

应按照《Intellec系列Ⅲ微机开发系统控制台操作说明书》的步骤安装硬盘子系统及给硬盘加电。当硬盘子系统准备好后，把"ISIS-Ⅱ Operating System"软盘插入计算机的软盘驱动器中，然后按RESET键。在显示出提示符(-)后（它出现在ISIS-Ⅱ信息之后），打入下面的命令：

-:F4:FORMAT :F0:SYSTEM .HDK S FROM 4 ⟨cr⟩

这个命令将0号驱动器的硬盘格式化成系统盘，它的名字叫做"SYSTEM .HDK"。当

然，用户可以使用其它的名字，但在句号的左边的字符不得多于6个，而右边不得多于3个。这个操作完成后，显示屏上重新显示提示符(—)，再使用下面的命令把系统盘上的文件传送到0号驱动器的硬盘上：

*-:F4:COPY :F4: *.* TO :F0: ⟨cr⟩*

这个命令将把软盘驱动器（4号驱动器）里"ISIS-Ⅱ Operating System"盘的文件全部复制到0号驱动器的硬盘。这个操作完成后，可从4号驱动器把"ISIS-Ⅱ Operating System"盘取出，换上别的软盘。除了"ISIS-Ⅱ Operating System"盘外，用户还应该复制"CREDIT ISIS-Ⅱ CRT-Based Text Editor"盘、"Resident 8086/8087/8088 Macro Assembler"（常驻的8086/8087/8088宏汇编程序）盘和"Resident 8086/8088 Utilities and Linkage Libraries"（常驻的8086/8088实用程序和链接程序库）盘的文件。对每个软盘，执行命令

-COPY :F4:.* TO :F0: ⟨cr⟩*

就把该软盘上的全部文件复制到0号驱动器的硬盘。如果用户对所有随机带的软盘都执行一次上述操作，则在0号驱动器的硬盘中便存贮了这些盘的全部文件。假如用户要有选择地复制一些文件到0号驱动器上，请参阅本章中第6节"复制文件"。

用户也能把文件复制到硬盘子系统的1号驱动器上。为此，首先必须打入下面的命令，格式化1号驱动器硬盘：

-FORMAT :F1: PROG86.HDK ⟨cr⟩

考虑到要用1号驱动器的硬盘存贮iAPX86,88应用程序，故在这里选用"PROG86.HDK"作为它的名字。如果你要用Pascal-86，则要把"Pascal-86 Compiler and Run-Time Libraries"（Pascal-86编译程序和运行时间库）盘插入4号软盘驱动器，并打入下面的命令。这样便将全部文件复制到1号驱动器的硬盘。

*-COPY :F4: *.* TO :F1: ⟨cr⟩*

用同样的方法可进行PL/M-86软盘和其它盘的复制。硬盘有足够大的空间供用户使用。

(2) 软盘用户

在这一节中，我们假设用户至少有两个倍密度的软盘驱动器。如果只有单密度的驱动器，在试下面的例子时，有可能出现盘的容量不够。软盘的有关情况，请参阅《Intellec系列Ⅲ微机开发系统控制台操作说明书》。

系统加电后，把"ISIS-Ⅱ Operating System"盘插入0号驱动器，然后按RESET键。操作系统引导进来后，显示出ISIS-Ⅱ信息，后跟提示符（-）。这时把一张空白盘插入1号驱动器，打入下面的FORMAT命令：

-FORMAT :F1: SYSTEM .FLX S⟨cr⟩

这个命令生成一个新的系统盘，并自动地将0号驱动器中有"S"属性的文件全部复制过来。此操作完成后，可对新系统盘进行检查，这只要把0号驱动器中的"ISIS-Ⅱ Operating System"盘取出，把这个新系统盘插入，然后按RESET键重新启动系统就能实现。

使用0号驱动器的新系统盘时，把"CREDIT ISIS-Ⅱ CRT-Based Text Editor"盘插入1号驱动器，并打入下面的命令。

-COPY :F1: CREDIT TO :F0: ⟨cr⟩

则把CREDIT程序复制到0号驱动器中新的系统盘上。从1号驱动器取出CREDIT盘，插入"Resident 8086/8087/8088 Macro Assembler"盘，打入下面的命令：

-COPY :F1: RUN. TO :F0:⟨cr⟩*

:F1: RUN COPIED TO :F0: RUN

:F1: RUN. OV0 COPIED TO :F0: RUN. OV0

此命令将RUN和RUN.OV0这两个文件从1号驱动器一次复制到0号驱动器。这两个文件是使用后面要讲到的特殊RUN命令所需要的。

取出1号驱动器中的软盘，换上"Resident 8086/8088 Utilities and Linkage Libraries"盘，同时打入下面的命令：

-COPY :F1: LIB86.86 TO :F0: ⟨cr⟩

:F1: LIB86.86 COPIED TO :F0: LIB86.86

-COPY :F1: LOC86.86 TO :F0:⟨cr⟩

:F1: LOC86.86 COPIED TO :F0: LOC86.86

-COPY :F1: LINK86.86 TO :F0:⟨cr⟩

:F1: LINK86.86 COPIED TO :F0: LINK86.86

至此我们就有了一张供本书中例子使用的系统盘了。此外，用户若用Pascal-86编译程序，还需要另一个盘来存贮Pascal-86编译程序、运行时间库和书中例子使用的实例程序。如要把一个空白盘格式化成非系统盘，则先把一个空白盘插入1号驱动器，然后打入下面的命令：

-IDISK :F1: PASC86.FLX⟨cr⟩

NON-SYSTEM DISK

IDISK命令能把一个盘格式化成系统盘或非系统盘，但它只能复制ISIS-Ⅱ格式文件(即带"F"属性的文件)，而不能复制其它文件。

有了1号驱动器中的空盘和0号驱动器中的系统盘，用户可把"Pascal-86"盘放入2号驱动器（如果有2号驱动器的话)。下面实例假设用户只有0号和1号驱动器。为了从"Pascal 86"盘复制文件，首先打入下面的命令（注意：不要忘记"P"!)：

*-COPY *.* TO :F1: P⟨cr⟩*

LOAD SOURCE DISK, THEN TYPE⟨CR⟩

这时系统暂时停下来（因为在命令的末尾指定了"P")，等着你把"源盘"插入0号驱动器。如果你用的是Pascal-86，这时"源盘"便是"Pascal-86"盘（如果用户只使用PL/M-86，"源盘"就是PL/M-86盘)。把这个软盘插入0号驱动器，然后按RETURN键，显示屏会出现：

LOAD OUTPUT DISK, THEN TYPE⟨CR⟩

(装入输出盘，然后打⟨cr⟩)

因为准备接收Pascal-86文件的盘已经在1号驱动器上，所以只需按RETURN键就行了。系统每次返回到0号驱动器复制其它文件时，都要显示上面的信息(以及"COPIED"信息)。用户所要做的事只是连续按RETURN键，直到整个复制工作全部做完为止。

用户可以重复这些步骤，在另一个空盘上复制PL/M-86编译程序。在这些格式化和复制工作完成以后，用户便有了供本书例子使用的相应的磁盘。

4. 文件名、路径名和文件属性

大多数操作系统命令都是用来处理文件。文件是信息的集合，它包括文本、数字数据、程序指令以及所有这些信息的组合。用户通过文件名来引用文件。文件名遵守一定的命名约