dBASE-II分析报告

张学平

计算机工程与应用编辑部

dBASE-II分析报告

广州军区后勤部电子计算机室　张学平

目　录

前言……1
第一章　系统概述　1
1.1　系统的结构……1
1.2　系统的功能……3
1.2.1　数据描述……3
1.2.2　传统的数据操作……3
1.2.3　特殊的数据操作——关系演算……4
1.2.4　其它功能……5
1.3 dBASE-II与操作系统的接口……5
第二章　变量和文件……6
2.1　变量及其内部表示……6
2.2　文件的种类、功用和构造……7
第三章　文件管理子系统……9
3.1　文件管理子系统的结构……9
3.2　文件基本操作模块的调用……10
3.3　文件的登录和调度……10
3.4　多重缓冲技术……11
3.5　文件管理信息块……12
第四章　半编译技术……13
4.1　中间代码处理模块……13
4.2　表达式的逆波兰代码生成模块……15
4.3　中间代码在不同情况下的生成……16
第五章　关系演算的实现……16
5.1　选择……16
5.2　投影……17
5.3　联结……18
第六章　B-树索引组织……19
6.1　索引逻辑结构……19
6.2　索引基本操作……20
6.2.1　查找……20
6.2.2　插入……20
6.2.3　删除……21
6.2.4　修改……22
6.2.5　其它……22
第七章　分类……23
7.1　新颖的外分类算法……23
7.2　分类模块……24
第八章　全屏幕操作……26
8.1　屏幕变量的格式化显示和登记……26
8.2　屏幕变量的编辑……26
8.3　全屏幕操作指南……27
第九章　控制语句结构……29
9.1　如果语句……29
9.2　情况语句……29
9.3　重复语句和过程语句……30
第十章　内存分配……31
10.1　内存分配的策略和实现……31
10.2　独立而可联系的两套工作区……31
10.3　动态分配区的使用……33
第十一章　分析中发现的问题和错误……33
附录……36
附录1　dBASE-II 命令一览表……36
附录2　新版本（V2.3B）所提供的新函数、新命令……37
附录3　信息号与信息对照表……38

前　言

1983年4月，我室派人去杭州参加了中国科学院计算研究所举办的“dBASE-II关系数据库推广应用第二期短训班”，复制了全套软件。从1983年5月起，笔者对dBASE-II进行了解剖、分析。先是用反汇编的方法得到了20000多条指令的源程序清单，然后对源程序进行通读、研究其设计思想和实现方法，并写出了约25万字的注释。最后，在《dBASE-II源程序清单与注释》的基础上，对dBASE-II作了进一步的消化、理解、写出了这本“分析报告”。

初稿承中山大学姚卿达副教授审阅。另外中山大学计算机系湛浩健、容慧二同志对初稿提了宝贵意见。在此谨致深切谢意。

由于本人水平不高，经验不足，加之资料匮乏，时间仓促，疏漏谬误在所难免，祈请读者批评指正。

作者　　1983年11月于广州

第一章　系统概述

dBASE-II是世界上最流行的微型机数据库之一，它适用环境广泛，语言简单灵活，功能相当强，被誉为“大众数据库”。dBASE—II是在CP/M 族多个操作系统支持下的关系数据库管理系统，用8080汇编语言编写，运行所需最小内存仅48K，这大大增强了它的通用性。

1.1 系统的结构

dBASE—II系统盘上的程序清单如下：

DBASE.COM	18K
DBASEMAI.OVR	7K
DBASEAPP.OVR	4K
DBASEBRO.OVR	2K
DBASEJOI.OVR	1K
DBASEMOD.OVR	3K
DBASEMSG.OVR	4K
DBASERPG.OVR	4K
DBASESRT.OVR	2K
DBASETTL.OVR	1K
DBASEUPD.OVR	1K
DBASEMSG.COM	8K
INSTALL.COM	10K

整个系统代码在单密度盘上占65 K。实为62336字节（约61K），反汇编并整理后的源程序清单共21130行（包括DW、DB、DS等伪指令）。

dBASE—II可粗略地认为由三大部分组成：

1．常驻内存的公用模块DBASE.COM，占用内存0100—447F；

2．主控模块DBASEMAI.OVR，占用内存4480—5FFF；

3．一组（九个）起址都为4480的复盖模块。

操作系统正常工作后，键入DBASE，则dBASE—II初启，先执行初始化部分，随即装入主控模块。主控模块从内存4480起把DBASE.COM的初始化部分复盖掉，未复盖部分即为常驻内存的公用模块（简称“常驻模块”）。

常驻模块可大致分为功用不同的四部分：

1．全屏幕编辑模块（0103—0AFC），其中0103—0160为外设参数区，内含执行了INSTALL.COM以后所确定的与外设（主要是显示终端）有关的参数。0 AA3—0AFC为该模块的

工作参数区，称为系统参数区1。

2．中间代码的生成和处理模块（0AFD—1C96），其中1C06—1C96为该模块的工作参数区，称为系统参数区2。

3．变量和文件管理模块（1C97—3059），其中2F50—3059为该模块的工作参数区，称为系统参数区3。

4．底层公用子程序（305A—4436），其中42E4—4436为该模块的工作参数区和整个系统的参数区，称为系统参数区4。

此外，4437—4472是几个“补丁”程序段和dBASE—II重新启动程序。

主控模块是系统的核心，其简化的流程如图1—1所示。

流程图中所说的“复盖命令”是指其执行模块在复盖模块中的那些命令，如表1—1所列。除此之外，主控模块中包含大约50多条命令的

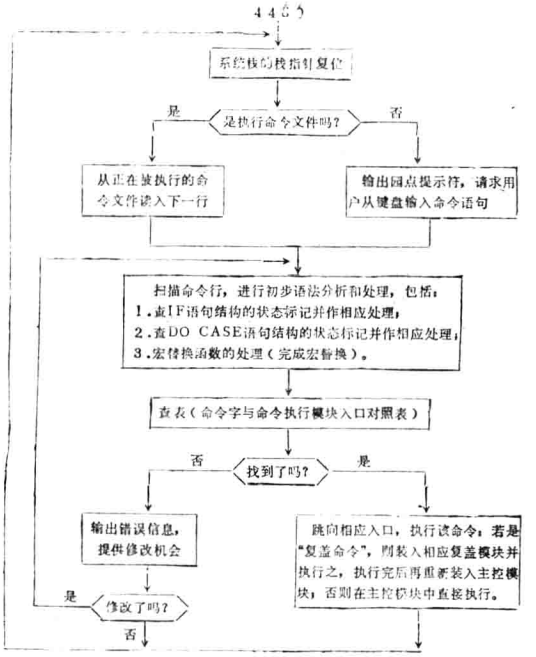

图1—1 主控模块简化流程图

表1—1“复盖命令”一览表

命　　令	复　盖　模　块	命令执行模块地址
APPEND	DBASEAPP. OVR	4492
CHANGE	DBASEAPP.OVR	4802
CREATE	DBASEAPP.OVR	48 F E
EDIT	DBASEAPP.OVR	4 B F B
INSERT	DBASEAPP.OVR	4DD7
INDEX	DBASEAPP.OVR	4E8D
COPY	DBASEMSC.OVR	4492
SUM	DBASEMSC.OVR	4985
QUIT TO〈COM-filename[,List]〉	DBASEMSC.OVR	4 B 05
DISPLAY FILES	DBASEMSC.OVR	4 C 14
DISPLAY MEMORY	DBASEMSC.OVR	4 E 41
PACK	DBASEMSC.OVR	4 E F 0
MODIFY STRUCTURE	DBASEMOD.OVR	449B
MODIFY COMMAND	DBASEMOD.OVR	47 B 0
BROWSE	DBASEBRO.OVR	4480
JOIN	DBASEJOI.OVR	4480
REPORT	DBASERPG.OVR	4480
SORT	DBASESRT.OVR	4480
TOTAL	DBASETTL.OVR	4480
UPDATE	DBASEUPD.OVR	4480

执行模块。所有这些命令，构成了dBASE—II数据库语言的命令系统（详见附录1）。

从主控流程图可以看出，主控模块实质上是一个“解释器”。所以，从编译方法角度来说，dBASE—II数据库语言是一种解释型语言。该语言设有“寄宿”功能，是一个“五脏俱全”的自含语言。

DBASEMSG.COM是一个信息文件，内有

各种提示信息、错误报警信息共123条。其组织方式是：每条信息占用64字节，两条正好占一个磁盘记录。若要输出第n条信息，则调用子程序3ADC，读入第n/2号磁盘记录块，并从该块的开头或正中起输出字符串，此即指定的信息。信息号与信息对照表见附录3。

1.2 系统的功能

1.2.1 数据描述

dBASE—II没有提供专门的数据描述语言，它的数据描述（即数据定义）主要是通过命令CREATE的执行而实现的。

在CREATE命令的执行过程中，首先要求给出“文件名”，这个文件名实际上就是“关系名”；其次，要求列出所有的“字段”，这些字段等价于“属性”。对于每个字段，都要求给出名字、类型、长度等。dBASE—II规定了三种类型：字符型、数字型、逻辑型。在描述一个数字型字段时，还要求给出小数位数。系统每接收一个字段的所有描述，便建立一个相应的字段描述块。描述块长为16个字节，其格式如下：

11B		1B	1B	2B	1B
字段名	00	类型符	长度	字段值存贮地址	小数位数

所有字段描述块拼在一起，便构成一个表，称为字段描述表。因为它表示出dBASE—II所处理的基本数据结构，所以又可称为结构表，简称结构（STRUCTURE），这实际上是记录的“型”。在dBASE—II中，记录的型就叫结构，它等价于“关系框架”；而记录的值则直称为记录，它等价于“元组”。

刚建立的结构中的字段值存贮地址是不确定的，只有在“记录缓存区”的地址确定后，它们才能随之确定。

属于数据描述范畴的“关键字指定”问题，在dBASE-II中是由SORT、INDEX、TOTAL、UPDATE等命令中的ON（key）短语来描述的。

除了定义一个新结构的CREATE命令以外。dBASE—II还提供了以下与结构有关的命令：

DISPLAY STRUCTURE　　查看结构
LIST STRUCTURE　　查看结构
COPY TO (newfile) STRUCTURE　　复制结构
MODIFY STRUCTURE　　修改结构
COPY TO (stru-file) STRUCTURE EXTENDED　　保存结构
CREATE (newfile) FROM (stru-file)　　还原结构

后两条命令配合使用，可先把某一文件的结构作为数据，以记录的形式保存在“结构文件”中，当需要产生一个同样的结构时，可从“结构文件”中读入数据而由系统自动产生之（而不带FROM短语的CREATE命令只能由用户从控制台输入描述内容才产生一个结构）。这些命令的提供，使应用程序员在数据描述的问题上，可以获得灵活多样的解决办法。

1.2.2 传统的数据操作

dBASE—II支持传统的数据操作，主要由以下命令来实现：

1. DISPLAY或LIST

查看记录。可以查看记录的所有字段，亦可只查看指名的部分字段。

2. APPEND

追加记录。所追加的记录可以来自终端控制台，亦可来自另一数据文件。

3. INSERT

插入记录。可以指定插入位置为某一记录之前或之后。

4. DELETE

删除记录。对不准备用的记录作删除标记。

5. RECALL

恢复记录。去掉已删除记录的删除标记。

6. PACK

剔除记录。“压缩”整个文件，使已删除

此为试读，需要完整PDF请访问：www.ertongbook.com

的记录从文件中永久地除去（再不可恢复）。

7．EDIT

对指定的记录进行修改。

8．COPY

复制记录到另一文件，或者选取部分字段组成新的记录再复制到另一文件。

9．REPLACE

对所选记录集的指名字段赋以指定表达式的值。

10．CHANGE

对所选记录集的指名字段进行修改。

11．UPDATE

用一个文件中的部分字段值更新另一个文件中的同名字段。更新的方式有两种：取代和增加。

12．BROWSE

是查询和修改数据的最有效的命令之一。如果把数据文件看作一个二维表的话，则BROWSE相当于把整个屏幕作为一个活动窗口在二维表上移动，并随时查看和修改窗口内的数据。屏幕上每次显示19行。一般情况下，每个记录占一行，一行内的字段数取决于字段的长度：在屏幕宽度以内，能容纳几个连续字段就容纳几个（字段间有空格相隔）。当一个字段的长度大于屏幕宽度时，则对字段换行显示，需占几行就占几行。总之要保证所显示字段的完整，以便于查看和修改。

13．SORT

以任一非逻辑型字段为关键字，对数据文件进行分类（可指定升序或降序）。

14．INDEX

以任一字段名或由字段名组成的表达式为关键字，生成数据库的索引机构，以便进行快速检索。对同一个文件，系统允许生成多达7套不同的索引机构，用不同的索引名加以区分。应用程序员可使用任一套索引机构进行检索，若要换用另一套，只须执行SET INDEX TO (index-name) 即可。关键字表达式可为任意表达式（但不能是逻辑型），一般用形为“字段1＋字段2＋字段3＋…”的表达式更有实用价值。关键字表达式的长度和关键字值的长度都可达100个字符。

此外，dBASE—II还提供了“统计算子”（或称“聚合操作”）：COUNT（计数）、SUM（求和）、TOTAL（小计、合计）。

1.2.3 特殊的数据操作——关系演算

关系数据库的创始人 E.F.Codd 在1981年获得图灵奖时的讲演中说过，一个数据库管理系统要想被称为关系数据库系统至少应具有三种操作：选择、投影、联结。dBASE—II支持这三种操作，并在某些场合允许组合使用这些操作。

选择被“分解”为二种操作：范围选择和条件选择。在命令中使用 ALL、RECORDn、NEXTn三个短语之一，就实现了范围选择，其含义依次为：“全部记录”、“第n个记录”，“从当前记录开始的n个记录”。在命令中使用FOR exp或WHILEexp短语，就实现了条件选择。两者的含义都是“当条件为真时记录入选，当条件为假时记录落选”；两者的区别是：前者当条件为假时再查下一个记录，后者当条件为假时退出再选择。范围选择和条件选择可独立使用，但也常常同时使用在同一命令行上，这使选择操作更加灵活。

投影的表达方式是：在某些命令行的适当位置使用字段名序列（有时还可扩充为表达式序列）。字段名序列一般作为 FIELDS短语的成分被使用，但亦有例外。

联结直接使用命令JOIN，它的执行是由一个专用的复盖模块来完成的。

“组合式”的关系演算大都体现在选择和投影的组合上，主要见诸于以下命令：

CHANGE〔scope〔FIELD〈field〔,list〕〉,〔FOR〈exp〉〕

COPY〔scope〕TO〈filename〉〔FIELDS〈field〔,list〕〉〕〔FOR〈exp〉〕

DISPLAY〔scope〕〔〈field〔,list〕〉〕〔FOR〈exp〉〕

REPLACE〔scope〕〈field〉WITH

〈exp〉〔,〈field〉WITH〈exp〉…〕
〔FOR〈exp〉〕

REPORT〔scope〕〔TO PRINT〕〔FOR〈exp〉〕（由用户从控制台指定字段名序列）

SUM〔scope〕〈field〔,list〕〉〔TO〈var〔, list〕〉〕〔FOR〈exp〉〕

TOTAL〔scope〕TO〈filename〉〔ON〈key〉〕〔FIELDS〈field〔,list〕〉〕〔FOR〈exp〉〕

其中,scope 即代表某一个范围选择短语。在多数情况下，条件选择短语 FOR exp 都能用 WHILE exp来代替。此外，JOIN 命令可选用 FIELDS 短语，即可在同一个命令的执行中直接对投影的结果进行联结，这使应用程序员在某些场合可用它进行询问的优化。

这些“组合”，增强了dBASE—II的关系处理能力，提高了使用的灵活性。

1.2.4 其它功能

作为独立的程序设计语言（自含语言），dBASE—II除了提供有关数据库“询问”功能的命令之外，其它各种命令亦应有尽有，主要表现在以下5个方面：

1．为了适应结构程序设计的需要，提供了可嵌套的控制语句结构：IF—ELSE—ENDIF语句、DO WHILE—ENDDO语句、DO CASE—ENDCASE语句。另外，还提供了DO〈filename〉命令，允许把编制好的命令文件作为应用程序的独立功能模块，就象 PASCAL语言中的“无参过程”一样使用。详见第九章。

2．为了使程序编制中能够使用中间变量和满足其它需要，系统允许使用内存变量。详见第二章。

3．参照COBOL语言和BASIC语言中人们已经熟悉的输入方式和输出格式，提供了灵活多样的交互输入命令和格式化输出（显示和打印）命令。

4．可利用SET命令改变某些系统参数（共25个），使系统的运行更符合用户要求。

5．低级接口：新版本（V2.3B）所提供的新命令中有3个可用于低级接口：POKE命令可从指定的内存单元开始存放机器代码（用十进制数表示）；SET CALL命令可设置一个待调用的子程序入口；而CALL命令则可执行对此子程序的调用。这种低级接口为系统的简易扩充提供了一条可行的途径。上述 3 条命令的详细说明见附录2。

1.3 dBASE—II与操作系统的接口

dBASE—II是在 CP/M1.4、CP/M2.2、MP/M II、CDOS或CROMIX等操作系统支持下工作的，它与操作系统的接口有以下三种情况：

1．所有的磁盘文件输入、输出操作都通过操作系统提供的调用入口0005进行，调用时的功能编号按需取用。

2．打印机输出亦通过操作系统提供的调用入口0005进行，调用时的功能编号 = 05。

3．控制台输入、输出不通过CALL0005来实现，为了满足dBASE—II特殊的控制要求和灵活多样的输入、输出方式，系统自备了一组I/O子程序，其中控制台输入、输出子程序直接以操作系统BIOS模块的转移矢量表中的三个入口为调用入口，这三个入口是：测试控制台状态入口、控制台输入入口、控制台输出入口，它们在系统启动时通过初始化子程序39C1来获取。获取的方法是：先从内存0001—0002字节取热启动入口地址（一般情况下，它等于转移矢量表中的第二条跳转指令的地址），然后以它为基址顺次加3，则分别得到三个入口地址。

第二章　变量和文件

2.1 变量及其内部表示

dBASE-II使用两种变量：字段变量（简称字段）和内存变量。前者是用户所定义的结构中的任一数据项，对应于关系中的某一属性；而后者则是系统运行时的“零散”数据项，与关系中的属性没有对应关系。

内存变量也同字段一样，有自己的描述块。二种描述块统称为变量描述块，其格式如下：

11B	1B	1B	2B	1
变量名 00	类型符	长度	变量值存贮地址	小数位数

变量名规定以字母开头，后面可以是字母、数字或冒号。变量名的长度规定不得超过10个字符。子程序2A93就是用作变量名的合法性检查的。

dBASE—II对所使用的变量规定了三种类型：字符型、数字型、逻辑型。字段的三种类型分别用类型符43、4E、4C（即字符C、N、L）来标识，而内存变量的三种类型则用字段的相应类型符的最高位置1、即用C3、CE、CC来标识。这样，当需要区别字段和内存变量时，查类型符的最高位即可；而不需要区别时，则屏蔽类型符的最高位后统一处理。

变量值的长度规定不得越过254个字符。逻辑型字段值的长度总是01；逻辑型内存变量描述块的“长度”字节不用，填入00。变量的小数位数只对数字型变量有意义，非数字型变量的小数位数永远是00。

若干个字段描述块拼在一起并以回车符0D作为结束符，便组成一个字段描述表，称为“结构”。系统运行时，一个结构对应一个记录缓存区，二者在内存中的位置都因所执行的命令不同而异。结构中各字段值的存贮地址只有在记录缓存区的位置确定之后才能确定，子程序2530就是根据记录缓存区的起址而计算和填写它们的。在记录缓存区中，各字段的值都是以ASCII码的形式存贮的，这和外部文件中的记录存贮格式是一致的。

内存变量描述表固定在内存6000—6400，即可容纳64个内存变量描述块的区域。随后的1536字节（6401—6A00），固定为内存变量值的存贮区。内存变量描述表的结束符0D固定在6400单元；内存变量经过多次建立和删除以后，表中的内存变量描述块便不再一一拼接而以“散落”的形式存在了。这是它和字段描述表的两点不同之处。

有了变量描述表，变量的查找就成了简单的查表。子程序26E3就是最基本的变量查找子程序，它被字段查找子程序284D和内存变量查找子程序29CE所调用，其流程图示于图2—1。

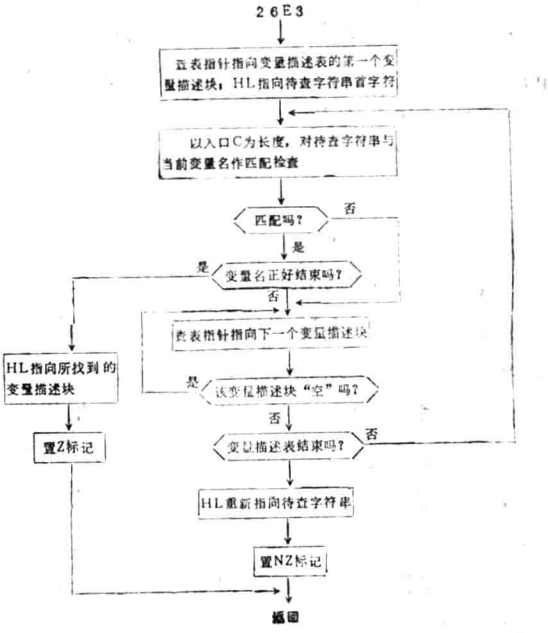

图2—1 从变量描述表查找变量子程序

内存变量通过命令ACCEPT、INPUT、COUNT、STORE、SUM、WAIT被使用。这些

命令的共同特点是都有 TO (var)短语。内存变量名取TO后面的合法标识符，长度、类型、小数位数由命令本身的功能所决定，而值的存贮地址则由内存变量存贮分配子程序29F2分配确定。分配的策略是：从存贮区起址6401开始扫描，遇空白段即查其长度是否够用，够则取之，不够则继续扫描、检查，只要待存贮值的末字节不超过6A00都可取用。

内存变量值的存贮形式与字段不同，其存贮地址的前一字节记载着本内存变量的存贮长度（包括这一字节本身），以便于存贮管理。字符型内存变量存贮地址第一字节为字符串长度，其后才是字符串本身，所以字符型内存变量的存贮长度＝字符串长度＋2；数字型内存变量的值是用BCD码形式存贮的，系统所使用的BCD数为6字节，所以其存贮长度固定为7，但它在描述块中的长度却为10，因为dBASE—II的BCD数的精度是10位有效数字、逻辑型内存变量值的存贮长度固定为2，值本身为一个字节——00或FF。

内存变量的使用比较自由，系统对它的处理思想是：有则用之，无则产生之。这分两步来实现：第一步，CALL2A51，查找或产生内存变量，亦即确定内存变量的描述块地址，确认或填写内存变量名；第二步，CALL2941，填写内存变量描述块中的长度、类型、小数位数、值的存贮地址。其中填写值的存贮地址是在内存变量存贮分配（CALL29F2）之后才进行的。

内存变量的删除使用命令 RELEASE。若要删除全部内存变量，则调用内存变量总清子程序28DB来完成；否则，通过CALL2A31来删除指名的内存变量。被删除的内存变量的描述块16字节全部清零，它的值所占的存贮单元亦全部清零。

为了弥补同时使用的内存变量不得超过64个的缺点，系统提供了SAVE命令和RESTORE命令，允许把某一时刻的全部内存变量转往外存，需要的时候再以复盖的方式调入内存。

在系统内部，对一个变量的引用一般都取其描述块地址，这个地址是通过变量查找而获取的。另外，有些命令的执行需要使用临时性的中间变量（如：@命令、ACCEPT 命令、INPUT 命令、SORT命令等），这些中间变量没有变量名，但为了满足处理上的规整和统一，仍被分配16字节的描述块（往往是固定分配），对这些中间变量的引用直接使用其描述块地址。

2.2 文件的种类、功用和构造

数据库管理系统是在文件的基础上发展起来的一门数据管理技术。通过文件管理来实现数据管理亦是dBASE—II的基本手段。dBASE—II使用7种不同的文件，各自都有固定的扩展名。7种文件可分为三类：主文件、索引文件、杂文件。

2.2.1 数据库主文件

主文件扩展名为DBF，是数据库最基本的文件。一个主文件对应一个“关系”，主文件名就是关系名，这个“关系”的所有“元组”都以记录的形式存贮于主文件中。任何数据操作都要访问主文件。

前面各章节中提到的文件都是指数据库主文件。

主文件由三部分组成：头信息、结构表、记录集。

头信息包括8个字节，其内容如下：

1B	2B	3B	2B
数据库标识号	文件内记录总数	上次更新日期	记录长度

其中，数据库标识号是为了区别由其它 DBMS 所管理的主文件而设置的。dBASE—II的标识号等于02。其它内容的含义是显而易见的。

结构表占用513字节，即可容纳32个字段描述块外加一个结束符的空间。不满32个字段的结构表仍占用513字节，结束符紧跟最后一个字段描述块。

头信息和结构表共521（0209H）字

节，在文件管理中它们是被作为一个整体进行读、写的，两个专用的读、写子程序是2D23和2E74。系统运行时，内存为头信息和结构表备有专用的存贮区，详见第十章。

记录集是文件中所有记录的集合，它们按写入时的次序排在结构表的后面。每个记录的第一个字节是“删除”标记，后面才是各字段的值。所以，记录长度＝各字段长度之和＋1。当一个记录被删除后，系统即对该记录的首字节填一星号；当不想删除而恢复该记录时，则填一空格。系统约定记录在主文件中的位置序号为记录号，所以，主文件最后一个记录的记录号总是等于文件内记录总数。记录亦有专门的读、写子程序：读——2BCC，写——24E8。

2.2.2 数据库索引文件

索引文件扩展名为NDX，是INDEX命令执行后的结果文件。索引文件和主文件一起组成完整的索引机构，是数据库快速检索的组织基础。

索引文件是由大小都等于512字节的若干个“块（索引块）所组成的。其中第00块是索引头，后面各块被组成B-树结构。对索引文件的读、写是以块为单位进行的。详见第六章。

2.2.3 数据库杂文件

主文件和索引文件之外的另5种文件统称为杂文件，它们是：

1．命令文件：扩展名为CMD，是用dBASE-II数据库语言书写的源程序文件。

2．报告格式文件：扩展名为FRM，用来保存REPORT命令首次执行时用户所回答的内容。这样，当再次执行REPORT命令时，系统自动按用户首次回答的内容进行所希望的处理，勿须用户再从键盘上回答。

3．内存变量文件：扩展名为MEM，是SAVE命令执行后的结果文件，用来保存某一时刻的全部内存变量。内存变量文件由二部分组成：内存变量描述表和内存变量的值，它实际上是内存6000——6A00的原样复制。

4．标准数据文件：扩展名为TXT，是dBASE-II和其它高级语言交换数据的“接口”文件。这里，“标准”的含义是：记录从文件头部开始存放，每个记录用回车换行结束，相当于一般高级语言的顺序文件。另外，记录中的数字型数据项（字段）一律用ASCII码形式。当使用命令COPY TO 〈filename〉 SDF时，将产生一个具有上述格式的数据文件，可供其它高级语言（或dBASE—II本身）使用。若其它高级语言程序产生了同样格式的TXT文件，则可使用APPEND FROM 〈filename〉 SDF 命令将其数据传到dBASE—II 主文件上（转换成DBF文件所要求的格式）。当TXT文件上的记录采用“数据项用逗号隔开、且字符串用单引号作为定界符括起来”的“通用”形式时，可用DELIMITED代替SDF。若不用单引号作定界符，可在COPY命令行上的DELIMITED后面加WITH短语予以指明。

另外，当需要“交互”输出到磁盘上时（即使用SET ALTERNATE TO 〈filename〉命令后），所指定的磁盘文件也以TXT为扩展名。

5．格式化I/O文件：扩展名为FMT，由一系列@命令语句所组成的、专供READ命令所使用的文件。其中，每一命令行上可省略命令字@。FMT文件用SETFOPM ATTO 〈filename〉命令指定和打开，在此后的READ命令执行时被按“行”读入和解释执行。

数据库杂文件中，除了内存变量文件以外，都是“行结构”文件，对其读、写都是以“行”为单位进行的。

第三章　文件管理子系统

3.1 文件管理子系统的结构

为了按数据库的需要有效地使用各种文件，dBASE-11的编制者在操作系统之上又设计了一套完整的文件管理子系统。子系统由两大部分组成：数据库文件管理模块和数据库文件基本操作模块。其中，管理模块是通过对基本操作模块的调用而实现其功能的，而基本操作模块则建立在操作系统的文件管理系统之上，即通过对操作系统的文件管理系统的调用来实现其功能。

文件管理子系统的上下层次调用关系如图3—1所示。

图3－1

数据库文件管理模块所包括的子程序大都散布在内存 1C97-3059 之间，其中直接调用基本操作模块 （CALL3B85） 的有下列子程序：

2277——从索引文件读索引头；
2313——关闭所有打开的索引文件；
237A——从索引文件读一索引块；
23AC——向索引文件写一索引块；
244A——向索引文件写索引头；
24EB——关闭文件使之变为后方文件；
251A——建立一个新文件；
256D——从杂文件读一行；
2584——向杂文件写一行；
26A6——向非索引文件写文件结束符 CTRL-Z；
26C1——关闭文件但仍保持“活动”状态；
26D5——删除一个后方文件同时注销其它后方文件；
2B82——打开一个文件；
5BCC——从主文件读一个记录；
2COF——从文件读指定长度的数据（用于内存变量文件和主文件）；
2C47——更改文件名；
2C89——按记录号置文件读/写指针；
2D16——文件读/写指针“归零”（即指向文件头部）；
2D23——读主文件的头信息和结构表；
2DE4——数据库复位（RESET）
2E39——向文件写指定长度的数据（用于内存变量文件和主文件）；
2E48——向主文件写一个记录；
2E74——向主文件写头信息和结构表；
2EB5——取非索引文件读/写指针；
2EF2——置非索引文件读/写指针；
另外，还有2个直接调用基本操作模块的子程序；
393E——从文件读一行到命令行缓冲区；
3979——把命令行缓冲区内容作为一行写到文件上。

数据库文件基本操作模块位于3B85-4409，它包括14个基本操作，并有约定的操作号，如下所示：

00——建立文件；
01——打开文件；
02——删除文件；
03——按长度读；
04——按长度写；
05——入口参数D＝0时，文件读/写指针“归零”；
入口参数D≠0时，关闭文件但仍保持“活动”状态；
06——写文件结束符CTRL-Z；
07——关闭文件并将之变为后方文件；
08——读一行；

此为试读，需要完整PDF请访问：www.ertongbook.com

09——写一行；

0A——取文件读/写指针；

0B——置文件读/写指针；

0C——dBASE-11复位（RESET）；

0D——更改文件名；

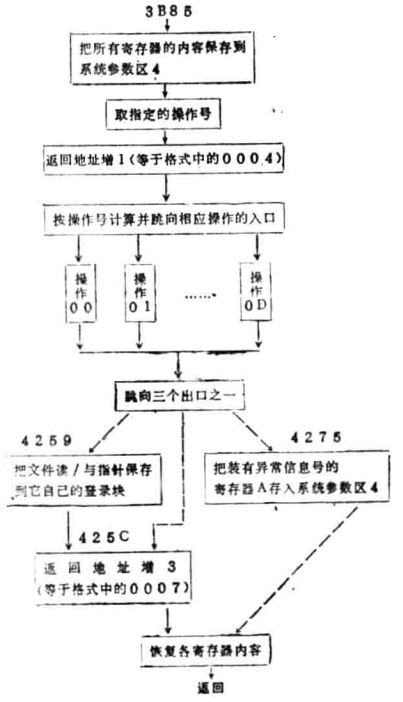

图3—2 数据库文件基本操作模块流程图

3.2 文件基本操作模块的调用

数据库文件基本操作模块有一个总入口和三个出口（二个正常出口、一个异常出口）它要求如下的调用格式：

```
0000   CALL    3B85
0003   DB      n     ；操作号
0004   JP      nn    ；异常跳转
0007   ……
       ⋮
```

文件基本操作模块的工作流程如图3-2所示。从流程图可知，正常出口和异常出口返回不同的地址。另外，异常出口都带一个出口参数（A＝异常信息号），而正常出口仅当“打开文件”操作成功后才带一个出口参数（A＝所分配的文件号）。

3.3 文件的登录和调度

dBASE-11允许用户同时打开多至16个文件，为此，在系统参数区4记置了数据库文件登录表（4351——43D0）。登录表内含16个文件登录块，每块8字节，形式如下：

1B	2B	2B	1B	2B
文件状态	块内读/写起点	块号	操作标志字	FCB地址

系统初启，文件状态字节都为00，表示“空闲登录块”；当一个文件被打开后，文件状态＝FF，表示“活动文件”；当一个活动文件被关闭后，则文件状态＝01，表示“后方文件”；如果把系统尚未访问的文件或已注销的后方文件称为“局外文件”的话，则三者的转换关系如图3 3所示。

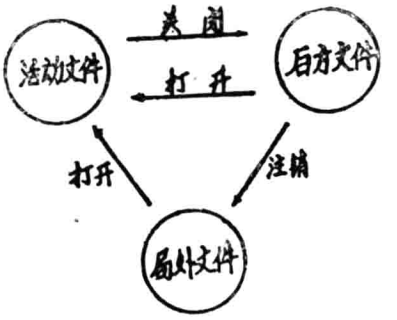

图 3—3

当对一个指名的文件进行“打开”操作时，首先查其是不是后方文件：若是，则仍使用原来的登录块，并把文件状态由01改为FF；若不是，则要申请一个文件登录块。申请文件登录块的原则是：先找空闲登录块。有则取之，无则再找后方文件登录块。找到并取用了一个后方文件登录块后，原后方文件自动注销；找不到则说明已经打开16个文件了。不管文件怎样获取文件登录块，都以登录块在表中的位置号作为分配给自己的文件号。此后，对该文件的读、写等操作就以此文件号作为访问的标识了。被打开文件的文件状态总是FF。

“块内读/写起点”和“块号”这四个字节共同构成了文件读/写指针。这里，“块”是

指逻辑块，其大小与I/O缓冲区一致，等于512字节。块内读/写起点用相对地址来表示，当数据在缓冲区内、外之间传送前，要以缓冲区首址为基址把相对地址化为绝对地址。每当系统通过CALL3B85进行了一次读、写之后，都要把当时的文件读/写指针保存在这四个字节，以备下次读、写使用。

“操作标志字”目前只使用了第6位：1表示对文件施行过“写”操作，0表示没有。当关闭该文件时，要据此而决定要不要“通知”操作系统执行关闭。

“FCB地址”即数据库文件控制块地址，是由系统固定分配和预先填好的。FCB的意义和格式与操作系统FCB一致，它是在文件“打开”操作中从005C——007F复制而得的。

为了使读、写等基本操作能统一地调用基本操作模块中的公用子程序（读/写子程序、缓冲区管理子程序等），在文件登录表的上方（4346——4350）还设置了个11一字节的“前线文件登录块”，其格式如下：

2B	2B	2B	1B	2B	2B
调度指针	块内读/写起点	块号	操作标志字	FCB地址	I/O 缓冲区地址

在进行读、写等操作时，先要把指定文件号的活动文件调往“前线”，即，把活动文件登录块的后7个字节复制到前线文件登录块的中间7个字节，这由子程序4087来完成；读、写等操作成功后，则把这7个字节的前5个字节（即已发生变化的文件读/写指针和操作标志字）又复制到文件登录块的原来位置，这由子程序424C来完成。在这来回的复制过程中，都以前线文件登录块的前2个字节作为“调度指针”，以指示被调度信息的来源或去处。调度指针总是指向调往前线的原活动文件登录块的第2字节。前线文件登录块的最后2字节是当前缓冲区地址，它的填写是在盘缓冲区管理子程序3FA4的执行中进行的（详见下节）。

3.4 多重缓冲技术

为了有效地支持数据库文件管理子系统，dBASE-II采用了多重缓冲技术。它设有一个缓冲池，内含8个缓冲区，每个512字节。为了控制这些缓冲区的使用，系统参数区4为每个缓冲区提供了一个控制块，8个控制块拼成一张缓冲区控制表（43D1-4409），用FF作为表结束标记。缓冲区控制块的结构如下：

1B	2B	2B	1B	1B
文件号	块号	缓冲区地址	“占用”标记	“待写”标记

缓冲区控制块第一字节是使用该缓冲区的文件号，兼作缓冲区忙/闲标记：该字节为00表示“闲”，否则表示“忙”。

“块号”指缓冲区中的内容所对应的文件逻辑块号。

“缓冲区地址”是系统固定分配并预先填好的。

“占用”标记表示该文件的某一块对该缓冲区的“占用”（01）和“释放”（00）。这种占用是“一次性”的，即某次读/写占用某个缓冲区后，下一次同块号读/写前若又有其它文件读/写过，则不能保证仍占用同一个缓冲区。

“待写”标记用以表示缓冲区内容有无变化：读操作时，肯定无变化；写操作时，只有向缓冲区传送数据后才有变化。若有变化，则“待写”标记=01，指示在释放该缓冲区前要把512字节写到磁盘上；无变化则“待写”标记=00。

当系统要读/写一块信息时，先查缓冲池中有无此块，有则不须启动磁盘传输了，直接对缓冲区取/送信息即可；若无此块，才申请新的缓冲区，也就是说，在读/写信息传送之前，要确认或申请一个缓冲区为当前缓冲区（即前线文件I/O缓冲区），以保证文件读/写指针所指定的块与当前缓冲区内容相符。确认或申请缓冲区的大致过程是：先检查缓冲池中有无被同一个文件的同一块所占用的缓冲区，有则确认它，无则申请一个。申请的原则是：“有空则用，无空则挤”。在有空闲缓冲区的情况下，一个文件不管是否已占用缓冲区，只

要是不同的块，就可以再申请新的缓冲区，所以，在最有利的情况下，一个文件可占用8个缓冲区。在无空闲缓冲区的情况下，则要“强占”一个非空闲缓冲区，即把第二遍查表首先遇到的缓冲区占为己有。若其“待写”标记=01则要先把缓冲区内容写往磁盘后再占用。确认或申请缓冲区是调用子程序3FA4来完成的，图3-4是它的工作流程图，这就是文件管理子系统的缓冲区管理子程序。

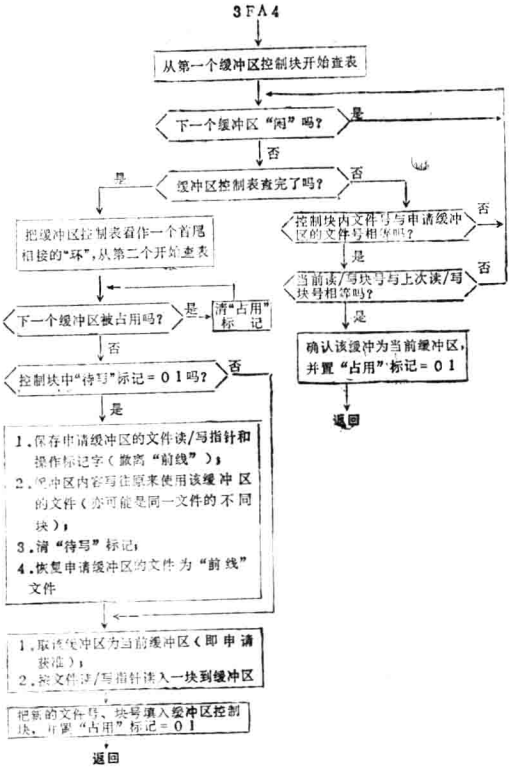

图3—4 确认或申请缓冲区子程序

读、写基本操作调用同一个子程序——读/写子程序。为了适应“按长度读/写”和“按行读/写”两种操作的调用，读/写子程序的返回条件设置了两种：一是“读/写剩余长度=0”，二是“读/写遇到了行结束符”。相应地，被读/写子程序所调用的复制子程序42F7也能判断和分析处理两种情况。读/写子程序3E44的流程示于图3-5。

关闭文件前，要检查缓冲区控制表，凡是被该文件使用的“待写”标记=01的缓冲区内容都要写往磁盘。

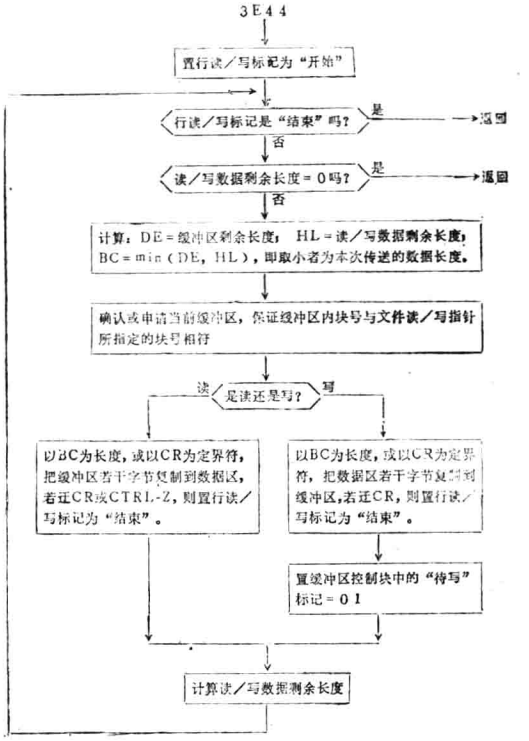

图3—5读/写子程序

3.5 数据库文件管理信息块

为了有效地对所使用的文件进行管理，系统对有关参数和流动信息实行集中存放的办法，形成文件管理信息块。信息块位于系统参数区3，占用45字节，其结构和内容如表3-1所示。

表3—1 文件管理信息块

偏移量	地址	初值	内容
00	2FD3—2FE1	全00	主文件名
0F	2FE2	00	EOF标记
10	2FE3—2FE4	0000	记录计数器/当前记录号
12	2FE5—2FE6	0000	记录长度
14	2FE7—2FE8	0000	当前记录总数
16	2FE9	00	头信息和结构表“重写”标记
17	2FEA	00	主文件号
18	2FEB	00	主索引文件号
19	2FEC—2FF3	全00	索引文件登记表
21	2FF4—2FF5	0000	索引路径栈的栈指针
23	2FF6—2FF7	7C7E	索引区起址
25	2FF8—2FF9	7BA3	索引头起址
27	2FFA—2FFB	814E	索引路径栈的栈底
29	2FFC—2FFD	6ACF	结构描述表存贮区起址
2B	2FFE—2FFF	6CD0	记录缓存区起址

当一个数据库主文件被打开以后，主文件名被复制到信息块的第00——0E字节，所分配的文件号则填入第17H字节。这一信息传递工作是由子程序2716来完成的。当主文件的结构表连同头信息被读入内存后，主文件内记录总数被复制到14～15H字节，记录长度被复制到12～13H字节。当一组索引文件（最多7个）被打开后，所分配的文件号依次填入第19H字节开始的索引文件登记表中，其中第一个被认为是主索引文件号并复制到第18H字节。

在文件管理信息块之外，还设有即席文件号（2F9C）和即席文件名（2F9D-2FAB）。所谓即席文件，是指当前正调用文件基本操作模块而进行某项操作的文件。除了索引文件的专门操作（如读/写索引头、读/写索引块等）之外，其它各种文件操作都统一地以即席文件号和即席文件名作为入口参数。例如：建立文件、打开文件、删除文件、文件改名之前，都CALL2795对给定的文件名进行合法性检查并把合法者复制到即席文件名位置；打开文件之后，所分配的文件号便填入即席文件号字节，如果需要的话，再转入其它地方保存；读、写文件都以即席文件号作为文件基本操作模块内部的前线文件号。

当一个主文件被使用（USE）后，主文件名和主文件号占据即席位置。如果要使用其它文件而占用即席文件名或即席文件号时，主文件只是暂时退让，一旦其它文件的操作结束后，系统自动通过CALL273E恢复主文件为即席文件。对用户来说，系统总是自动维持最近一次使用的主文件为即席文件，直到执行下一个USE命令为止。

索引文件的专门操作不涉及即席文件，它们直接以信息块中的主索引文件号或各个索引文件号作为入口参数。

系统参数区3中共有三块结构、大小都一样的文件管理信息块，这是为了实现两套数据库文件的交叉并用而设置的，详见第十章。

第四章　半编译技术

为了提高某些命令的执行效率，同时为了系统本身编码时的易读性、可靠性和结构化，dBASE-II采用了半编译技术。所谓“半编译”即不是真正的编译，只是把命令所涉及的“运算”转换为一种符合一定规则的中间代码，然后对中间代码进行解释执行。在这里，“运算”的概念是广义的，除了通常的表达式求值所进行的各种“运算”之外，dBASE-II还使用了更多的运算，如赋值、传送、变量输出、变量制表输出……等。另外，所提供的17个函数都各自对应一个“运算”。在采用中间代码的命令执行模块中，中间代码可在语法分析过程中生成，亦可在语法分析之后生成，这取决于具体的功能需求和编码的方便。中间代码一次生成后，可在一个命令执行期间反复解释执行（如果需要的话），这无疑提高了系统的效率。

4.1 中间代码处理模块

对中间代码的解释执行是由中间代码处理模块（OAFD—1474）来完成的。该模块包括49个运算子程序，每个子程序有一个约定的编号，称为运算号。各运算子程序的运算号、入口地址、所对应的运算符、功能说明等，如表4-1所列。

在各个运算中，凡是数字量的四则运算以及单目减、取整等运算，都必须施于BCD数。在常驻模块的底层（305A—351B）备有BCD数的加、减、乘、除、单目减、取整运算的子程序和BCD数对ASCII码、二进制数的转换子程序。

中间代码处理模块的处理方法是：构造了一个运算分量栈和一个运算分量存贮区，栈中存放暂时不能参与运算的分量的存贮地址和中

表 4-1 “运 算” 一 览 表

运算号	入口地址	运算符	功能说明	运算号	入口地址	运算符	功能说明
00	1052		数值0入栈	1A	11DD	函数LEN	测栈顶字符串分量的长度
01	1066	单目减	把栈顶分量置成它的相反数	1B	11EC	函数*	查当前记录是否删除
02	1074	/	除	1C	11FF	函数EOF	查当前使用的主文件是否结束
03	108B	*	乘	1D	1212	函数@	测栈顶二分量(字符串)一个在另一个之中的位置
04	1097	-	减(数字值)	1E	1248	—	字符串相减
05	10A3	+	加(数字值)	1F	129A	函数!	栈顶字符串分量中的小写字母全部转换为大写
06	10AF		数字值“相等”比较	20	12BA	函数CHR	取与栈顶数字分量等价的ASCII码字符(包括控制字符)
07	10B8	()或"	数字值“不等”比较	21	12D0	函数 DATE	取格式为 MM/DD/YY 的日期，结果以字符串分量入栈
08	10C1	>	数字值“大于”比较	22	130A	函数 TRIM	消去栈顶字符串分量后面的空格
09	10CD	<	数字值“小于”比较	23	1331	函数FILE	以栈顶字符串分量为名查文件是否存在
0A	10D6	>=	数字值“不小于”比较	24	136A	函数 TEST	测试表达式结果值的类型、长度
0B	10DF	<=	数字值“不大于”比较	25	13A4	函数 PEEK	取栈顶数字分量作为16位二进制数时的低8位值
0C	10EB	+	字符串相加	26	0BB1		变量入栈
0D	1111	$	字符串“匹配”测试	27	0B9B		常量入栈（包括函数TYPE）
0E	1144	=	字符串“相等”比较	28	0E5E		传送（运算号后第一个分量的值传送给第二个分量）
0F	114D	()或"	字符串“不等”比较	29	0C39		输出栈顶分量的值
10	1156	>	字符串“大于”比较	2A	0E8A		栈顶分量赋值给运算号后面的内存变量
11	1162	<	字符串“小于”比较	2B	0F04		输出运算号后面的分量值
12	116B	>=	字符串“不小于”比较	2C	0F82		栈顶分量赋值给运算号后面的字段变量
13	1174	<=	字符串“不大于”比较	2D	0C95		栈顶分量按运算号后面4个制表参数输出其值
14	1180	AND	逻辑与	2E	0FE9	函数STR	三目运算：以栈顶第一个分量为小数位数，第二个分量为总长度，把第三个分转换为玉ASCII码字符串
15	1191	OR	逻辑或				
16	11A2	NOT	逻辑非	2F	100A	函数 $	三目运算：以栈顶第一个分量为子串长度，第二个分量为子串起点，从第三个分量（字符串）中取子字符串
17	11B7	函数 INT	对栈顶数字值分量取整				
18	11C3	函数RLC	取当前记录号	30	0B85		①返回（A=栈顶逻辑值） ②完成字符串制表输出（2D）后“超长”部分的换行输出（在指定输出域下方）
19	11CF	函数VAL	栈顶字符串分量转换为数字值				

间运算结果的存贮地址，存贮区则用来存放它们的值。对已生成的中间代码从左到右进行扫描，每遇到“入栈”运算号时，就把运算号后面的运算分量（指其存贮地址，下同）入栈；每遇到n目运算符时，就弹出栈顶的n个分量进行运算符所指定的运算，然后把中间结果入栈……这样一直处理下去，直到整个中间代码段扫描完为止。这实际上就是逆波兰表达式的通用处理方法之一。但是dBASE-II 的“运算”概念的扩充，使得有些运算不完全符合上述处理方法。有几个运算不从栈顶取运算分量而取自运算号之后(如21、28、2B)、或取自别处（如18、1B、1C、21）；有几个运算的运算分量既有取自栈顶的、又有取自运算号之后的（如2A、2C、2D）；至于运算的中间结果，亦有不需入栈的（如28、29、2A、2B、2C、

2D）。这些不同的处理方法虽然看起来有点儿“零乱”、“不规整”，但却适应了被扩充的运算的特殊要求而并不影响它纳入中间代码处理模块统一处理。

中间代码处理模块的流程图示于图4-1。

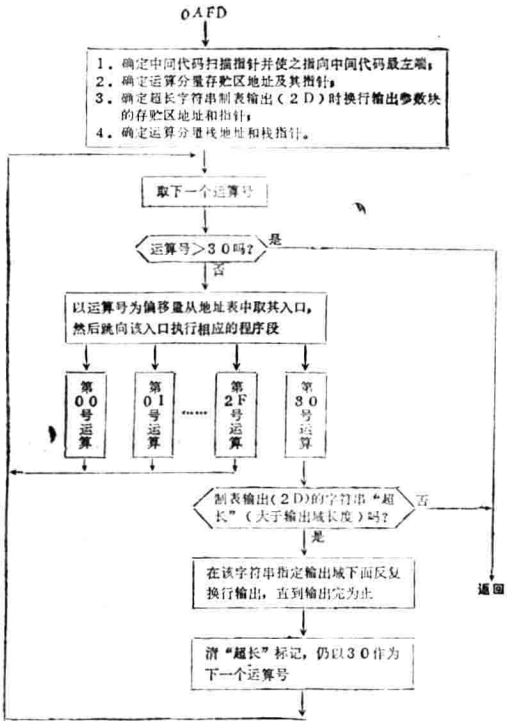

图4-1 中间代码处理模块流程图

4.2 表达式的逆波兰代码生成模块

表达式是任何计算机高级语言的重要成分。与通常的“表达式”相比，dBASE-II表达式”的概念有所扩充。通过对表达式的逆波兰代码生成模块的分析，可以给出dBASE-II表达式的BNF形式定义如下：

表达式::=关系表达式〔逻辑运算符关系表达式〕

逻辑运算符::=NOT|AND|OR

关系表达式::=算术表达式〔关系运算符算术表达式〕

关系运算符::=$|=|≠|>|<|≥|≤

算术表达式::=项〔加减运算符　项〕

加减运算符::=+|-

项::=因式〔乘除运算符　因式〕

乘除运算符::=*|/

因式::=数字型常量|〔单目运算符〕初级量

单目运算符::=+|-

初级量::=函数|字符型常量|字符型变量|数字型变量|逻辑型变量|逻辑型常量|（表达式）

dBASE-II所特有的字符串相加、字符串相减、字符匹配测试等运算，都被含入其中了。

dBASE-II表达式的运算符优先级如下：

1．单目+-

2．*/

3．+-

4．=<>　><　>=　<=　$

5．NOT

6．AND

7．OR

表达式的逆波兰代码生成模块（1475—1C96）采用递归子程序法对表达式进行扫描处理而生成逆波兰形式的中间代码。表达式转换子程序149A由多层嵌套调用的子程序组成归其中的“括号处理子程序”对149A进行递，调用，既把括号内的内容又作为表达式进行处理。

逆波兰形式的中间代码由一串运算号及分量存贮地址组成，最后总是30。其中，分量都占2字节，全部是分量值为存贮地址（包括间接地址），而运算号占用1字节。如：

X+Y*Z-0.5

转换后形为

26	X	26	Y	26	Z	03	05	27	0.5	04	30	C	B	30

又如：P>0.2　AND Q<=lim

转换后形为

26	P	26	0.2	08	26	Q	26	lim	0B	14	30	00	00	30

逆波兰代码的尾部有2个30，第一个就是“结束码”。两个30之间的2个字节表示表达式

此为试读，需要完整PDF请访问：www.ertongbook.com