


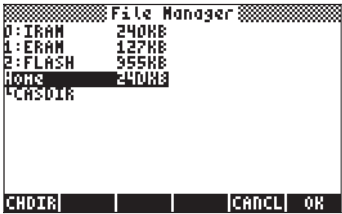
Chapter 26

管理内存

在第2章中，我们介绍了创建和管理变量和目录的基本概念和操作。在本章中，我们将讨论计算器内存的管理，包括内存分区和备份数据的技术。

存储器结构

计算器包含总共2.5 MB的内存，其中1 MB用于存储操作系统（系统内存），1.5 MB用于计算器操作和数据存储（用户的内存）。用户无权访问系统内存组件。要查看用户内存分区的方式，请使用FILES功能 ( FILES)。可能的结果如下所示：



除了与HOME目录对应的存储器外，此屏幕还指示存在三个存储器端口（请参阅本指南中的第2章）。可用的内存端口是：

- Port 0, labeled IRAM
- Port 1, labeled ERAM
- Port 2, labeled FLASH

端口0和HOME目录共享相同的内存区域，因此存储在HOME目录中的数据越多，例如，端口0存储可用的内存越少。端口0 / HOME目录存储区的内存总大小为241 KB。

端口1 (ERAM) 最多可包含128 KB的数据。端口1与端口0和HOME目录一起构成计算器内存的计算器RAM (随机存取存储器) 段。RAM存储器段需要来自计算器电池的连续电力供应才能运行。为避免丢失RAM存储器内容, 包括CR2032备用电池。请参阅本章末尾的其他详细信息。

端口2属于计算器的闪存ROM (只读存储器) 段, 不需要电源。因此, 取出计算器的电池不会影响计算器的Flash ROM段。端口2最多可存储1085 KB的数据。

第四个端口Port 3可用于SD闪存卡。一个例子如下所示。



只有插入SD卡时, 端口才会出现在文件管理器中。

HOME目录

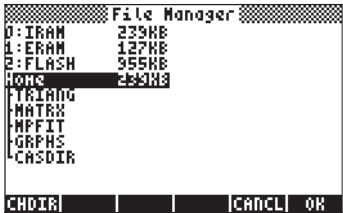
使用计算器时, 您可能会创建变量来存储中间结果和最终结果。某些计算器操作 (如图形和统计操作) 会创建自己的变量来存储数据。这些变量将存储在HOME目录或其中一个目录中。有关变量和目录操作的详细信息, 请参见第2章。

端口内存

与HOME目录不同, 端口0,1和2中的内存不能细分为目录, 只能包含备份对象或库对象。这些对象类型如下所述。

检查内存中的对象

要查看存储在存储器中的对象，可以使用FILES功能 (◀ FILES)。下面的屏幕显示了HOME目录，其中包含五个目录，即TRIANG, MATRX , MPFIT, GRPHS, and CASDIR.



通过在目录树中向下移动光标可以查看其他目录。 或者您可以向上移动光标以选择内存端口。 选择给定目录，子目录或端口后，按 **OK** 查看所选对象的内容。

访问端口内存的另一种方法是使用LIB菜单(▶ LIB ,与(2) 键相关联), 此操作将生成以下屏幕:



如果您的计算器中有任何库活动，它将显示在此屏幕中。 一个这样的库是上面屏幕中显示的**MPFIT** (演示) 库。 按相应的软菜单键(▶ FI) 将激活此库。 按端口软菜单键将打开该内存端口。 有关图书馆的更多信息如下。


备份对象

备份对象用于将主目录中的数据复制到内存端口。备份内存端口中的对象的目的是保留对象的内容以供将来使用。备份对象具有以下特征：

- 备份对象只能存在于端口内存中（即，您无法备份HOME目录中的对象，尽管您可以根据需要制作多个副本）
- 您无法修改备份对象的内容（但是，您可以将其复制回HOME目录中的目录，在那里进行修改，然后再次对其进行修改）
- 您可以将单个对象或整个目录存储为单个备份对象。但是，您无法从目录中的许多选定对象创建备份对象。

在端口存储器中创建备份对象时，计算器将根据对象中包含的二进制数据获取循环冗余校验（CRC）或校验和值。此值与备份对象一起存储，并由计算器用于监视备份对象的完整性。将备份对象还原到HOME目录时，计算器将重新计算CRC值并将其与原始值进行比较。如果发现差异，计算器会警告用户恢复的数据可能已损坏。

备份端口内存中的对象

将对象从用户存储器备份到其中一个内存端口的操作类似于将变量从一个子目录复制到另一个子目录的操作（请参阅第2章中的详细信息）。例如，您可以像使用普通计算器对象一样使用文件管理器 ( FILES) 复制和删除备份对象。此外，还有用于操作备份对象的特定命令，如下所述。


备份和恢复HOME

您可以在单个备份对象中备份当前HOME目录的内容。此对象将包含当前在HOME目录中定义的所有变量，键分配和闹钟。您还可以从先前存储在端口存储器中的备份对象还原HOME目录的内容。这些操作的说明如下。

备份HOME目录


要使用代数模式备份当前HOME目录，请输入命令：

```
ARCHIVE(:Port_Number: Backup_Name)
```

这里，Port_Number为0,1,2（或3，如果SD存储卡可用 - 见下文），Backup_Name是将存储HOME内容的备份对象的名称。使用按键序列 ::__。例如，要将HOME备份到端口1中的HOME1，请使用：



要以RPN模式备份HOME目录，请使用以下命令：

```
: Port_Number : Backup_Name  ARCHIVE
```

恢复HOME目录


要以代数模式恢复主目录，请使用以下命令：

```
RESTORE(: Port_Number : Backup_Name)
```

例如，要从备份对象HOME1恢复HOME目录，请使用：

```
RESTORE(: 1:HOME1)
```

在RPN模式下使用：


```
: Port_Number : Backup_Name  RESTORE
```





Note: 当您还原HOME目录备份时，会发生两件事：

- 备份目录将覆盖当前的HOME目录。因此，当前HOME目录中未备份的任何数据都将丢失。
- 计算器重新启动。历史或堆栈的内容丢失。

存储，删除和还原备份对象


要创建备份对象，请使用以下方法之一：

- 使用文件管理器 ( FILES) 将对象复制到端口。使用此方法，备份对象将与原始对象具有相同的名称。
- 使用STO命令将对象复制到端口。例如，在代数模式下，要将变量A备份到端口1中名为AA的备份对象，请使用按键序列：

   ::  /  ALPHA A ALPHA A ENTER

- 使用ARCHIVE命令创建HOME目录的备份（参见上文）。


要从端口删除备份对象：

- 使用文件管理器 ( FILES) 删除对象，就像删除HOME目录中的变量一样（参见第2章）。
- 使用PURGE命令如下：


在代数模式下，使用： PURGE(: Port_Number : Backup_Name)



在RPN模式下，使用： : Port_Number : Backup_Name PURGE

要还原备份对象：

- 使用文件管理器 ( FILES) 将备份对象从端口内存复制到HOME目录。
- 还原备份对象时，计算器通过计算其CRC值对还原的对象执行完整性检查。计算的和存储的CRC值之间的任何差异都会导致指示数据损坏的错误消息。

使用备份对象中的数据

虽然您无法直接修改备份对象的内容，但可以在计算器操作中使用这些内容。例如，您可以运行存储为备份对象的程序或使用备份对象中的数据来运行程序。要运行备份对象程序或使用备份对象中的数据，可以使用文件管理器( FILES) 将备份对象内容复制到屏幕。或者，您可以使用函数EVAL运行存储在备份对象中的程序，或使用函数RCL从备份对象恢复数据，如下所示：

- 在代数模式中：
 - 要评估备份对象，请输入：
EVAL(argument(s), : Port_Number : Backup_Name)
 - 要将备份对象调用到命令行，请输入：
RCL(: Port_Number : Backup_Name)
- In RPN mode:
 - 要评估备份对象，请输入：
Argument(s)  : Port_Number : Backup_Name EVAL
 - 要将备份对象调用到命令行，请输入::
Port_Number : Backup_Name  RCL

使用SD卡

计算器有一个存储卡端口，您可以在其中插入SD闪存卡以备份计算器对象，或从其他来源下载对象。计算器中的SD卡将显示为端口号3。.

插入和取出SD卡

SD插槽位于计算器的底部边缘，位于数字键的正下方。SD卡必须朝下插入。大多数卡片上都有一个标签，通常被认为是卡片的顶部。如果您在键盘朝上的情况下握住HP 50g，则插入HP 50g时，SD卡的这一面应朝下或远离您。卡片将在大部分长度内无阻力地进入插槽，然后需要稍微用力才能完全插入。完全插入的卡几乎与外壳齐平，只留下卡的顶部边缘。

要取出SD卡，请关闭HP 50g，轻轻按压卡的外露边缘并推入。卡应从插槽中弹出一小段距离，以便现在可以轻松地从计算器中取出。

格式化SD卡

大多数SD卡已经过格式化，但可能会使用与HP 50g不兼容的文件系统进行格式化。HP 50g仅适用于FAT16或FAT32格式的卡。


您可以从PC或计算器格式化SD卡。如果您从计算器（使用下面描述的方法）执行此操作，请确保您的计算器有新的或相当新的电池。

Note: 格式化SD卡会删除当前所有数据。



1. 将SD卡插入卡槽（如上一节所述）。
2. 按住 ON 键，然后按 F4 键。松开 F4 键，然后松开 ON 键。系统菜单显示有多个选项。
3. 按0进入FORMAT。格式化过程开始。
4. 格式化完成后，HP 50g将显示该消息"FORMAT FINISHED. PRESS ANY KEY TO EXIT". 要退出系统菜单，请按住 ON 键，按下并释放 F3 键，然后松开 ON 键。

SD卡现在可以使用了。它将以FAT32格式格式化。

替代方法

插入SD卡时，在文件管理器中出现另一个菜单项。选择此选项会重新格式化卡，该过程也会删除卡上的每个对象。

访问SD卡上的对象

从SD卡访问对象类似于对象位于端口0,1或2中时。但是，使用LIB功能 ( LIB). 时，端口3不会出现在菜单中。 只能使用Filer或文件管理器 ( FILES). 管理SD文件。 启动Filer时，如果插入了SD卡，树视图将如下所示：






Filer中支持SD卡上的长文件名，但显示为8.3个字符，如DOS中所示，即显示的名称最多包含8个字符，后缀中包含3个字符。 除非是PC对象或未知类型的对象，否则将显示每个对象的类型。（在这些情况下，其类型列为String。）

除了使用文件管理器操作之外，您还可以使用STO和RCL函数在SD卡上存储对象并从中调用对象，如下所示。 您还可以使用PURGE命令擦除SD卡中的备份对象。 长名称可以与这些命令一起使用（即STO，RCL和PURGE）。

将对象存储在SD卡上






要存储对象，请使用函数STO，如下所示：


- In algebraic mode: 输入对象，按STO，使用端口3键入存储对象的名称
Enter object, press , type the name of the stored object using port 3 (e.g., `3:VAR1`), press .
- In RPN mode: 输入对象，使用端口3键入存储对象的名称
Enter object, type the name of the stored object using port 3 (e.g., `3:VAR1`), press .

请注意，如果要存储在SD卡上的对象名称超过8个字符，则一旦存储在卡上，它将在Filer的端口3中以8.3 DOS格式显示。

从SD卡中调用对象

要将SD卡中的对象调用到屏幕上，请使用功能RCL，如下所示：








- In algebraic mode: 按RCL，使用端口3键入存储对象的名称
Press   , type the name of the stored object using port 3 (e.g., #3=VAR1), press  .
- In RPN mode: 使用端口3键入存储对象的名称
Type the name of the stored object using port 3 (e.g., #3=VAR1), press   .

使用RCL命令，可以通过在命令中指定路径来调用变量，例如，在RPN模式下：#3: {path}  RCL. 与DOS驱动器一样，路径是一系列目录名，它们一起指定目录树中变量的位置。但是，通过指定路径无法调用存储在备份对象中的某些变量。在这种情况下，必须调用完整备份对象（例如，目录），然后在屏幕中访问各个变量。

Note: 请注意，对于具有长文件名的对象，可以在发出RCL命令时指定对象的全名或其截断的8.3名称。

评估SD卡上的对象






要评估SD卡上的对象，请插入卡，然后：

1. 按  :: . 这会在编辑行上放置一个双冒号，光标在冒号之间闪烁。这是HP 50g处理存储在其各个端口中的项目的方式。端口3是SD卡端口。
2. 按     [name of the object]  . 这将在堆栈上放置要计算的对象的名称和路径。
3. 要评估对象，请按  .

请注意，对于具有长文件名的对象，在评估SD卡上的对象时，可以指定对象的全名或其截断的8.3名称。


从SD卡清除对象

要将SD卡中的对象清除到屏幕上，请使用PURGE功能，如下所示：



- In algebraic mode:
Press  , 使用端口3键入存储对象的名称(e.g., #3:VAR1), press .
- In RPN mode:
使用端口3键入存储对象的名称(e.g., #3:V R1), press  .

请注意，对于具有长文件名的对象，可以在发出PURGE命令时指定对象的全名或其截断的8.3名称。

清除SD卡上的所有对象（通过重新格式化）

您可以通过重新格式化来清除SD卡中的所有对象。插入SD卡时， 在文件管理器中显示另一个菜单项。选择此选项将重新格式化整个卡，该过程也会删除卡上的每个对象。

指定SD卡上的目录

您可以存储，调用，评估和清除SD卡上目录中的对象。 请注意，要使用SD卡根级别的对象，请使用  键。但是当处理子目录中的对象时，必须使用  键括起包含目录路径的名称。

例如，假设您要将名为**PROG1**的对象存储到SD卡上名为**PROGS**的目录中。如果此对象仍在堆栈的第一级，请按：

← :: 3 → → " ALPHA ALPHA P R O G S → ÷ P R O G / ENTER STO▶

这会将先前在堆栈上的对象存储到SD卡上，并将名为**PROGS**的目录存储到名为**PROG1**的对象中。 注意：如果**PROGS**不存在，将自动创建目录。

您可以指定任意数量的嵌套子目录。例如，要引用第三级子目录中的对象，您的语法将是：

:3:"DIR1/DIR2/DIR3/NAME"

请注意，按 α \rightarrow \div 会产生正斜杠字符。

使用库

库是用户创建的二进制语言程序，可以加载到计算器中，并可在**HOME**目录的任何子目录中使用。此外，计算器附带两个库，它们共同提供了方程式库的所有功能。

库可以作为常规变量下载到计算器中，然后安装并附加到HOME目录。


安装和附加库

要安装库，请在堆栈中列出库内容（使用 \rightarrow 变量软菜单键或函数**RCL**）并将其存储到端口0或1。例如，要将库变量安装到端口中，请使用：

- 在代数模式中: **STO**(Library_variable, port_number)
- 在RPN模式下: Library_variable ENTER port_number STO

在内存中安装库内容后，需要将库附加到**HOME**目录。这可以通过重新启动计算器（关闭和重新打开计算器）或同时按下 ON F3 来完成。此时库应该可供使用。要查看库激活菜单，请使用**LIB**菜单 (\rightarrow LIB)。 库名称将在此菜单中列出。

库编号

如果使用LIB菜单( **LIB**) 并按下与端口0,1或2对应的软菜单键, 您将看到软菜单键标签中列出的库号。每个库都有一个与之关联的三位或四位数字。(例如, 组成公式库的两个库位于端口2中, 编号为226和227.) 这些编号由库创建者分配, 用于删除库。

删除库

要从端口删除库, 请使用:

- 在代数模式中: `PURGE(:port_number: lib_number)`
- 在RPN模式下: `: port_number : lib_number PURGE`

其中lib_number是上述库号。

WARNING: 端口2中的库226和227构成了公式库。您可以像删除用户创建的库一样删除这些库。但是, 如果您考虑删除这些库, 但将来有可能需要使用公式库, 则应在使用HP 48/50计算器连接套件将它们复制到PC之后再删除它们。计算机。然后, 当您需要使用公式库时, 您将能够在以后重新安装库。

创建库

库可以用汇编语言, 系统RPL语言编写, 也可以使用库创建库(如LBMKR)编写。后一个程序可在线获得(参见例如<http://www.hpccalc.org>)。使用汇编语言或系统RPL语言编写计算器的详细信息超出了本文档的范围。邀请用户在线查找有关该主题的其他信息。

备用电池

计算器中包含CR2032备用电池, 可在更换主电池时为易失性存储器提供电源备份。建议您每5年更换一次电池

屏幕消息将指示何时需要更换此电池。 下图显示了备用电池在计算器背面顶部隔室中的位置。

