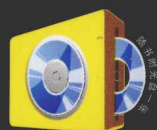


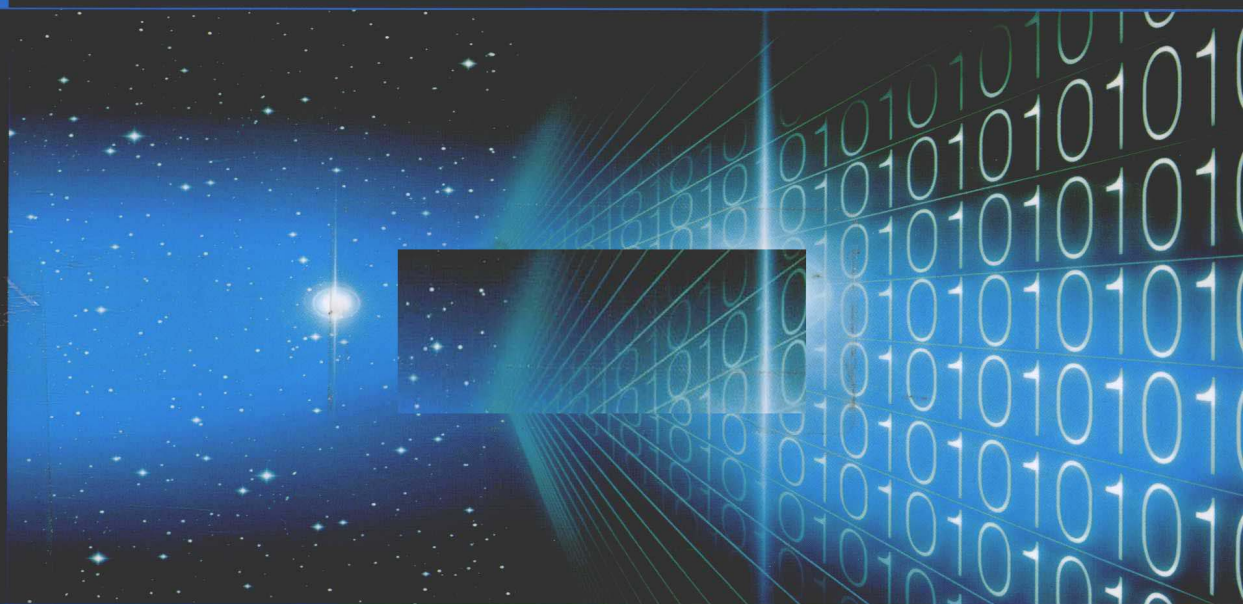
MATLAB
BIANYI CHENGXU
HE WAIBU JIEKOU



MATLAB

编译程序 和外部接口

董振海 编著



國防工業出版社

National Defense Industry Press

MATLAB 编译程序 和外部接口

董振海 编著

国防工业出版社

·北京·

前言

MATLAB 编译程序是一个不小的题目,且是一个不可缺少的工具,为什么就没有一本完整而系统的关于它的书呢?我想做,我想让还不十分了解编译程序的人们知道编译的全过程,知道它所包含的所有功能以及编译程序外延的事情,以便他们可以根据不同的 MATLAB 源程序设计个性化的编译过程。同时我也在关注着 MATLAB 与 VB、C/C++ 等混合编程的问题,即 MATLAB 的外部接口。将 MATLAB 编译程序与外部接口放在一本书中,作为姊妹篇,这恐怕是一种天然的渊源。因为编译程序本身具有连接外部程序的功能,利用编译程序就可以直接将 MATLAB 与 C/C++ 程序连接在一起,形成独立应用程序。

那么你知道为什么 MATLAB 与 C 或 C++ 有天然的接口吗?

答案是:将 MATLAB 源程序编译成目标程序,分成两步。第一步编译 MATLAB 源程序成中间语言,即 C 或 C++ 代码;第二步将 C 或 C++ 代码编译成目标代码。

你知道服务器程序与客户程序不是网络概念,而是不同语言的程序连接时所扮演的“一仆一主”角色吗?你知道使用 MATLAB 的语法形式就能在 MATLAB 中直接操作 Java 的类和对象是为什么吗?你知道在 MATLAB 与 VB 的混合程序中是 MATLAB 程序控制 VB 程序,还是 VB 程序控制 MATLAB 程序呢?你知道在不打开 Excel 的情况下,用 MATLAB 可以建立工作簿、添加工作表、从工作表读写数据吗?这些答案尽在本书中。这些也正彰显出 MATLAB 外部接口的魅力。

MATLAB 这几年在中国的应用和普及日益扩展,它非常强的计算功能和适用于多学科、多领域的优势,是其他的编程语言所不能比拟的。将 MATLAB 作为计算引擎,再利用其他语言的长处写适当的子程序,或者使用已有的其他语言程序作为辅助,这岂不是扩大了 MATLAB 的能力!MATLAB 的外部接口就是给它增加了几只有力的“臂膀”。

这本书的特色可以概括为:内容全面,阐述细致,实例众多,理论与实际相结合。

本书中绝大部分的例子都经实实在在的编译、运行。编译、运行的过程和结果以及图形都尽量完整。对于实际编译和运行中,原始资料叙述不详和错误的地方,都加以说明和纠正。运行中出现的问题及改正的方法,也都有描述。力求让别人多从实例中更好、更快地学到东西。程序例子中用到的函数,其参数和用法都一一详细说明,对读懂程序非常有用。

这本书开门见山直接切入主题,没有专门的章节写 MATLAB 的基础内容。书中对个别用到的 MATLAB 编程知识,做了简单介绍,这是远远不够的。有关 MATLAB 编程的诸多问题,请参看我所编写的《精通 MATLAB 7 编程与数据库应用》。

受知识和经历的局限,书中难免有错误和不当之处,请读者朋友予以批评指正。

编者

2010 年 9 月

目 录

上篇 MATLAB 编译程序

引言	1
第 1 章 MATLAB 编译程序的有关命令、附注函数	3
1.1 编译程序的有关命令	3
1.1.1 mcc	3
1.1.2 buildmcr	8
1.1.3 mbuild	11
1.1.4 isdeployed	14
1.2 附注函数	14
1.2.1 % #external	14
1.2.2 % #function	14
1.3 mcc 命令选项的快速参考	15
第 2 章 MATLAB 编译程序生成目标程序	17
2.1 MATLAB 编译程序可以生成的目标程序	17
2.1.1 封装文件	17
2.1.2 独立应用程序	17
2.1.3 库	18
2.1.4 MATLAB 的 COM 和 Excel 创建程序	19
2.2 编译程序建立程序组件的基本过程	19
2.2.1 编译独立的应用程序	20
2.2.2 编译共享库	20
2.2.3 生成调用共享库的独立应用程序	20
2.2.4 在开发程序的机器上测试组件	21
2.2.5 部署程序组件到别的机器	21
2.3 MATLAB 编译程序的局限	22
2.3.1 编译 MATLAB 和工具箱的限制	22
2.3.2 对独立应用程序的限制	22
2.3.3 弥补 Callback 问题:丢失函数	23
第 3 章 安装与配置	25

3.1	系统要求	25
3.2	安装	26
3.2.1	MATLAB 编译程序的安装	26
3.2.2	安装 ANSI C 或 C++ 编译程序	26
3.3	配置	27
3.3.1	mbuild 应用程序简介	27
3.3.2	配置 ANSI C 或 C++ 编译程序	27
3.4	选项文件	29
3.4.1	寻找选项文件	29
3.4.2	修改选项文件	30
3.4.3	用 mbuild -setup 选择编译程序的提示	35
3.5	Windows 中编译程序的限制	36
第 4 章	编译过程	37
4.1	MATLAB 编译程序技术术语	37
4.1.1	MATLAB 组件运行时库	37
4.1.2	组件技术文件	37
4.2	编译过程	37
4.3	输入和输出文件	38
4.3.1	独立可执行程序	38
4.3.2	C 共享库	39
第 5 章	部署过程	40
5.1	概述	40
5.1.1	生成的代码部署于不同平台	40
5.1.2	抽取 CTF 档案而不执行组件	40
5.1.3	用户操纵编译路径	41
5.2	在部署的机器上安装 MCR	42
5.2.1	在 Windows 系统中安装 MCR	42
5.2.2	在 Linux 系统中安装 MCR	43
第 6 章	使用 mcc 所涉及的种种问题	45
6.1	编译程序的选项	45
6.2	宏与捆绑文件——简化编译选项	46
6.2.1	宏	46
6.2.2	捆绑文件	47
6.3	使用封装文件	48
6.3.1	主文件封装	49
6.3.2	C 库封装	51
6.3.3	C++ 库封装	54
6.3.4	三种封装文件的比较	58
6.3.5	COM 组件封装	58

6.4	使用% #external 附注函数	58
6.5	使用% #function 附注函数	64
6.6	脚本文件	65
6.6.1	转换脚本 M 文件为函数 M 文件	65
6.6.2	应用程序中包含的脚本文件	66
6.7	使用路径名	66
6.8	有关根目录和版本信息的命令	67
6.8.1	matlabroot 命令	67
6.8.2	ver 命令	67
第 7 章	独立应用程序	68
7.1	C/C++ 独立应用程序的目标程序	68
7.1.1	编译应用程序	68
7.1.2	测试应用程序	74
7.1.3	部署应用程序	75
7.1.4	运行应用程序	76
7.2	只用函数 M 文件写应用程序代码	77
7.3	混合 M 文件与 C 或 C++ 文件的独立应用程序——C/C++ 与 MATLAB 接口之 C/C++ 程序调用 MATLAB 程序	82
7.3.1	简单例子	82
7.3.2	稍复杂例子	89
7.3.3	小结	95
第 8 章	从函数 M 文件生成共享库——C/C++ 与 MATLAB 接口之 C++ 程序 调用 MATLAB 程序	96
8.1	C 共享库	96
8.1.1	C 共享库的封装	96
8.1.2	C 共享库例子	96
8.1.3	调用共享库	107
8.2	C++ 共享库	108
8.2.1	C++ 共享库的封装	108
8.2.2	C++ 共享库的例子	109
8.3	MATLAB 编译程序生成的接口函数	118
8.3.1	调用共享库程序的结构	132
8.3.2	库初始与结束函数	133
8.3.3	打印与错误处理函数	133
8.3.4	由 M 文件生成的函数	134
8.4	独立应用程序与共享库编译上的区别	135
第 9 章	COM 和 Excel 组件	137
9.1	COM 和 Excel 的 MATLAB 生成器	137
9.2	COM 对象的目标程序	137

9.3 Excel Plug-In 目标程序	143
第 10 章 错误和警告信息	145
10.1 编译时错误	145
10.2 警告信息	147
10.3 运行时错误	152
10.4 Depfun 错误(相关性分析错误)	153
10.5 问题解答	153
10.5.1 mbuild 有关的问题	154
10.5.2 MATLAB 编译程序有关的问题	155

下篇 MATLAB 外部接口

引言	156
第 11 章 输入和输出数据	157
11.1 使用 MAT 文件	157
11.1.1 输入数据到 MATLAB	157
11.1.2 从 MATLAB 输出数据	158
11.1.3 在不同平台之间交换数据	159
11.1.4 读/写 MAT 文件	160
11.1.5 写字符数据	160
11.2 读写 MAT 文件的例子	161
11.2.1 用 C 建立 MAT 文件	161
11.2.2 用 C 读 MAT 文件	166
11.2.3 用 Fortran 建立 MAT 文件	170
11.2.4 用 Fortran 读 MAT 文件	174
11.3 编译和连接处理 MAT 文件的程序	176
11.3.1 屏蔽浮点数异常	176
11.3.2 在 Windows 系统编译和连接程序	177
11.3.3 需要的第三方源文件	179
第 12 章 MATLAB 与通用 DLL 的接口	181
12.1 加载和卸载库	181
12.1.1 加载共享库——loadlibrary 函数	181
12.1.2 卸载库——unloadlibrary 函数	186
12.2 获取库的有关信息	188
12.2.1 libfunctions 函数	188
12.2.2 libfunctionsview 函数	189

12.3	调用库函数——calllib 函数	190
12.4	传送参数	191
12.5	数据转换	193
12.5.1	原始数据类型	194
12.5.2	枚举类型	196
12.5.3	结构	197
12.5.4	建立引用	202
12.5.5	引用指针	206
第 13 章	MATLAB 与 C/C++ 和 Fortran 的接口——从 MATLAB 调用 C 和 Fortran 程序	208
13.1	MEX 文件	208
13.1.1	使用 MEX 文件	208
13.1.2	mx 和 mex 为前缀的子程序的区别	208
13.2	MATLAB 数据	209
13.2.1	C 语言程序中的 MATLAB 数组	209
13.2.2	数据存储	209
13.2.3	MATLAB 的数据类型	210
13.2.4	初试数据处理	214
13.3	生成 MEX 文件	223
13.3.1	mex 函数	223
13.3.2	mex 的选项文件	229
13.3.3	在 Windows 系统建立 MEX 文件的过程	231
第 14 章	用 C 语言写 MEX 文件	233
14.1	MEX 文件的结构	233
14.2	关于 mexFunction 函数	235
14.2.1	mexFunction 的定义	235
14.2.2	mexFunction 的一般结构	236
14.3	与 MEX 文件有关的内存管理	237
14.3.1	自动清除临时数组机制	237
14.3.2	永久数组	238
14.3.3	内存管理所涉及的主要函数	239
14.3.4	MEX 文件的执行、清除和锁定	251
14.4	建立 MEX 文件时常见的问题	252
14.4.1	MEX 文件本身的问题	252
14.4.2	内存管理的问题	253
14.5	C MEX 文件的例子	254
14.5.1	传送一个标量	254
14.5.2	传送字串	260
14.5.3	传送两个或多个输入或输出参数	264

14.5.4	传送结构数组和单元数组	265
14.5.5	处理复数数据	276
14.5.6	处理 8 位、16 位和 32 位数据	279
14.5.7	操作多维数值数组	280
14.5.8	处理稀疏矩阵	284
14.5.9	从 C MEX 文件调用 MATLAB 函数	290
第 15 章 MATLAB 与 C 和 Fortran 的接口——在 C 和 Fortran 程序中调用		
	MATLAB	293
15.1	MATLAB 引擎子程序库	293
15.2	调用引擎子程序的例子	297
15.2.1	从 C 应用程序调用 MATLAB(Windows 系统)	298
15.2.2	从 Fortran 应用程序调用 MATLAB	301
15.3	编译和连接引擎程序	304
15.3.1	屏蔽浮点异常	304
15.3.2	在 UNIX 系统编译和连接引擎程序	304
15.3.3	在 Windows 系统编译和连接引擎程序	305
15.3.4	需要的第三方文件	307
第 16 章 MATLAB 与 Java 的接口——从 MATLAB 调用 Java		309
16.1	概述	309
16.2	在 MATLAB 中使用 Java 类和方法	309
16.2.1	Java 类的源	309
16.2.2	定义新的 Java 类	310
16.2.3	Java 类的路径	310
16.2.4	使 Java 类变为可用于 MATLAB 的类	319
16.2.5	加载 Java 类	320
16.2.6	简化 Java 类名	321
16.2.7	寻找原来的方法库	322
16.3	建立和使用 Java 对象	322
16.3.1	构造和引用 Java 对象	323
16.3.2	连接 Java 对象	324
16.3.3	保存和加载 Java 对象	326
16.3.4	查看对象的公共数据字段	326
16.3.5	存取私有和公共数据	327
16.3.6	存取静态字段的数据	328
16.3.7	确定对象的类	328
16.4	调用 Java 对象的方法	330
16.4.1	使用 Java 和 MATLAB 的调用语法形式	330
16.4.2	调用 Java 类的静态方法	331
16.4.3	获取方法的有关信息	332

16.4.4	影响 MATLAB 命令的 Java 方法	336
16.4.5	MATLAB 怎样处理未定义的方法	337
16.4.6	MATLAB 怎样处理 Java 异常	337
16.5	在 MATLAB 中使用 Java 数组	338
16.5.1	MATLAB 怎样表示 Java 数组	338
16.5.2	在 MATLAB 中建立对象的数组	341
16.5.3	存取 Java 数组的元素	343
16.5.4	给 Java 数组赋值	345
16.5.5	连接 Java 数组	348
16.5.6	对 Java 数组建立新的引用	349
16.5.7	建立 Java 数组的复制	350
16.6	传送数据到 Java 方法	352
16.6.1	MATLAB 参数数据的转换	352
16.6.2	传送内建数据类型	353
16.6.3	传送字符串参数	354
16.6.4	传送 Java 对象	354
16.6.5	另外的数据转换问题	357
16.6.6	传送数据到重载方法	357
16.7	处理从 Java 方法返回的数据	358
16.7.1	Java 返回数据的转换	359
16.7.2	内建数据类型	359
16.7.3	Java 对象	359
16.7.4	转换对象为 MATLAB 数据类型	360
16.8	程序设计的例子	362
16.8.1	读 URL——URLdemo	362
16.8.2	查找 IP 地址——resolveip 函数	364
16.8.3	经串行端口通信——serialexample 程序	366
16.8.4	建立和使用电话簿——phonebook 函数	369
第 17 章	MATLAB 与 VB 和 Excel 的接口——COM 和 DDE(仅支持 Windows)	382
17.1	概述 MATLAB COM	382
17.1.1	概念和术语	382
17.1.2	MATLAB 支持的客户程序与服务器程序配置	383
17.1.3	注册控件和服务器程序	386
17.2	建立和操作 COM 控件和服务器程序的 MATLAB 函数	389
17.2.1	建立 ActiveX 控件	389
17.2.2	建立 DLL 组件和 EXE 组件的对象——actxserver 函数	398
17.2.3	获取对象的接口	400
17.2.4	COM 对象的方法或函数	404

17.2.5	对象的属性	410
17.2.6	控件和服务程序的事件	417
17.2.7	编写事件处理程序(或响应函数)	423
17.2.8	保存和加载 COM 控件的对象——save 和 load 函数	426
17.2.9	释放 COM 对象和接口——release 和 delete 函数	427
17.2.10	获取有关 COM 对象的信息	429
17.2.11	MATLAB 与 Excel 的接口——MATLAB 作为自动操作客户程序的 例子	430
17.3	自动操作服务器程序	432
17.3.1	建立自动操作的服务器程序	433
17.3.2	VB 与 MATLAB 的接口实例——连接到已存在的 MATLAB 服务器 程序	434
17.3.3	MATLAB 服务器程序函数	435
17.4	MATLAB 与 VB 和 Excel 接口综述	449
17.4.1	MATLAB 与 VB 接口	449
17.4.2	MATLAB 与 Excel 接口	450
17.5	动态数据交换	451

上篇 MATLAB 编译程序

引言

有人问我：

编译程序(有的人称为编译器)有什么用？

我立即回答：

能将编程语言写成的源程序编译成可以执行的目标程序。

这我知道。除此之外它还能做什么？

我明白了，他想问的不是“编译”的笼统概念，而是想知道有关 MATLAB 编译程序的诸多事情。

于是，我告诉他：

MATLAB 编译程序有很多选项，用不同的选项主要可以做到：

- (1) 生成独立应用程序，或别的软件组件(如共享库、COM 对象等)。
 - (2) 可选择不同的中间语言的编译程序(如 C 或 C++ 编译程序)。
 - (3) 可以输出追踪和查错信息，便于改正源程序的错误。
 - (4) 可以只生成中间语言(C 或 C++)程序，或者连接需要的库文件做成完整的独立应用程序。
 - (5) 输出整个的编译信息，用于查找编译过程的问题。
 - (6) 选择最终的输出，避免不必要的资源耗费。
- 等等。

当然，这仅仅是编译过程(或编译程序本身)所要完成的工作。还有编译程序外延的一些事情：

- (1) 如何将目标程序部署在没有安装 MATLAB 的机器上，需要准备哪些文件。
- (2) 怎样安装 MATLAB 编译程序，怎样安装 C 或 C++ 编译程序。

MATLAB 编译程序版本 4(R14)适用于 MATLAB 7.0，MATLAB 编译程序版本 4 兼容以前的版本。以前的版本能够编译的 M 文件，这个版本也可以编译。

编译程序版本 4 生成的 API(应用编程接口)与以前的版本生成的不同。如果要开发软件组件，需要调整为新的 API。

MATLAB 编译程序版本 4 能够将 M 文件作为输入，生成独立应用程序或软件组件。生成的这些应用程序和软件组件只能用于特定的平台。MATLAB 编译程序能够生成如下类型的应用程序或软件组件：

1) 独立的应用程序

独立的应用程序运行时不需要 MATLAB, 即使 MATLAB 没有安装在用户的系统上, 它们也能够运行。

2) C 和 C++ 共享库(在 Microsoft Windows 中叫动态链接库(DLL))

即使 MATLAB 没有安装在用户的系统上, 它们也能被使用。

3) Excel add-ins

需要 MATLAB Builder for Excel。

4) COM objects

需要 MATLAB Builder for COM。

MATLAB 编译程序支持所有的 MATLAB 语言的功能, 包括对象。

MATLAB 解释程序与编译程序的区别: MATLAB 解释程序是 MATLAB 的一个应用程序, 它接受 MATLAB 命令, 执行 M 和 MEX 文件。当你使用 MATLAB, 在命令窗口写入命令或运行函数时, 就在使用 MATLAB 的解释程序。它解释命令, 执行命令赋予的功能, 有时将结果输出在命令窗口。

MATLAB 编译程序是一个单独的产品, 它接受 M 文件作为输入, 且生成独立的应用程序或软件组件。调用 MATLAB 编译程序, 用 `mcc` 命令。

第 1 章 MATLAB 编译程序的有关命令、附注函数

1.1 编译程序的有关命令

此一节有四个命令: `mcc`, `buildmcr`, `mbuild`, `isdeployed`。

1.1.1 `mcc`

`mcc` 调用 MATLAB 编译程序。

1. 语法形式

```
mcc[-options] mfile1 [mfile2 ... mfileN] [C/C++file1 ... C/C++fileN]
```

说明:

`mcc` 是调用 MATLAB 编译程序的命令,能够从 MATLAB 命令提示符、DOS 命令行或 UNIX 命令行发出这个命令。

`mcc` 为 MATLAB 环境之外准备部署的 M 文件。用 C 或 C++ 生成封装文件,并选择性地建立独立二进制文件。写结果文件到当前目录(此为默认选择)。

如果在命令行指定了多于一个的 M 文件,编译程序为每个 M 文件生成一个 C 或 C++ 函数。如果指定了 C 或目标文件,它们和生成的 C 文件一起被传送给 `mbuild`。

2. 参数说明

<code>-options</code>	选项,皆以“-”为先导,可选参数。注意,同一个字母的大小写是不同的选项。
<code>mfileN</code>	被编译的 M 文件,1 到 N 个。
<code>C/C++fileN</code>	被编译的 C/C++ 文件,1 到 N 个。这些文件与编译生成的 C/C++ 文件一起被传送给 <code>mbuild</code> 。

3. 命令选项

1) -a

加一个文件到 CTF 档案。如:

```
-a filename
```

将指定的文件 `filename` 直接加到 CTF 档案,允许多个 `-a` 选项。编译程序在 MATLAB 路径寻找这些文件,因此,指定完全的路径名是可以的。这些文件不被传送到 `mbuild`,因此可包含数据文件。

2) -b

生成 Excel 兼容的公式函数(Formula Function)。

生成一个 VB 文件(.bas),文件包含 Microsoft Excel 公式函数与编译程序生成的 COM 对象的接口。当引入到工作簿 VB 代码时,代码允许这个 MATLAB 函数作为单元格公式

函数。要求 MATLAB 的 Excel 生成器。

3)-B 指定捆绑文件

用指定捆绑文件 filename 的内容代替 mcc 命令行的文件本身,形式如下:

```
-B filename[: <a1> , <a2> , ..., <an> ]
```

例如: `mcc -B 'ccom:addin,addin,1.0' main.m`

捆绑文件 filename 应当只含有 mcc 命令行选项和相应的参数,以及别的文件名。这个文件可以包含另外的-B 选项。捆绑文件能够包含接受名字和版本号的编译程序选项的替换参数。(参见表 5.2。)

4)-c 只生成 C 代码

与宏选项一起使用时,生成 C 代码但不调用 mbuild,即不生成独立应用程序。这个选项等价于放在 mcc 命令行末尾的-T codegen。

5)-d 输出指定目录

输出到一个指定目录,用

```
-d directory
```

将来自编译程序的输出文件放到-d 选项指定的目录。

6)-f 指定选项文件

由指定选项文件代替默认选项文件,形如:

```
-f filename
```

调用 mbuild 的时候,指定 filename 作为选项文件。这个选项允许对 MATLAB 编译程序的不同需求使用不同的 ANSI(美国国家标准学会)编译程序。选项的指定文件直接传送给 mbuild。

注:建议使用 mbuild-setup,预先设置选项文件,这样可以不用-f。

7)-g 生成追踪信息

包含对封装的查错符号信息。

8)-G 仅用于查错

选项使 mbuild 用适当的 C/C++ 编译程序的查错选项调用 C/C++ 编译程序。如果想用查错程序对独立应用程序查错,应当指定-G。

9)-I 加目录到路径

每个-I 选项加一个目录到当前搜索路径的末尾,例如:

```
-I <directory1> -I <directory2>
```

将建立搜索路径,以便为寻找 M 文件首先搜索 directory1,再搜索 directory2。在独立编译中 MATLAB 路径不可用时,这个选项很重要。

10)-l 生成函数库

建立函数库的宏。这个选项对命令行的每个 M 文件生成一个库封装函数,并调用 C 编译程序建立一个共享库。库名是命令行的第一个 M 文件名。这个宏等价于

```
-W lib: <string> -T link:lib
```

11)-m 生成独立应用程序

也是一个宏,产生独立应用程序。它等价于

```
-W main -T link:exe
```

12)-M 直接传送

指定编译时选项,用

`-M string`

它直接传送 `string` 到 `mbuild`。这为指定编译时选项提供了有用的方法。例如:

`-M "-Dmacro = value"`

注:如果出现多个 `-M` 选项,只使用最右边的一个。

13)-N 清除路径

有效清除所有目录的路径,除下列核心目录外(这个列表有时会有变化):

`<matlabroot> /toolbox/matlab`

`<matlabroot> /toolbox/local`

`<matlabroot> /toolbox/compiler`

编译时出现在 MATLAB 路径中的上述目录的所有子目录也在不清除之列。在命令行使用选项 `-N`,也允许替换原始路径的目录,而保持被包含目录的相对顺序,当然,也包括出现在原始路径中被包含目录的所有子目录。

14)-o 指定可执行程序

指定最后的可执行程序的名字和位置(仅限独立应用程序),以这样的形式:

`-o outputfile`

指定编译程序的最后可执行输出文件的名字。有些情况下,要对指定的名字加扩展名(例如,对 Windows 独立应用程序要加 `.exe`)。

15)-p 加目录到路径

加目录到编译路径,按照在 MATLAB 路径中寻找的顺序加到适当的位置。

`-p directory`

这里的 `directory` 是被加的目录。如果 `directory` 不是绝对路径,就认为它是在当前工作目录下。加这些目录的规则是:

(1) 如果 `-p` 后面的目录是在原始 MATLAB 路径中,那么这个目录以及它的所有出现在原始路径中的子目录被加到编译路径,并且按照在 MATLAB 路径中寻找的顺序。

(2) 如果 `-p` 后面的目录不在原始 MATLAB 路径中,这个目录不被加到编译路径(但可以使用 `-I` 选项加它)。

如果 `-I` 选项能够被用来加一个目录(要求 `-N` 选项),且这个目录已经在 MATLAB 路径中,它将被加到编译路径(按照在 MATLAB 路径中寻找的顺序),就像使用 `-p` 选项一样。否则,这个目录被加到编译路径的头上,就像正常使用 `-I` 选项一样。

16)-R 运行时

提供 MCR 运行时选项,形如

`-R option`

提供下列运行时选项,见表 1.1。

表 1.1 运行时选项

选项	描 述
<code>-nojvm</code>	不使用 Java Virtual Machine (JVM)
<code>-nojit</code>	不使用 MATLAB JIT(二进制代码的生成用来加速 M 文件执行)

注:-R 选项仅仅用于独立应用程序。为了在别的 MATLAB 编译中代替 MCR 选项,使用 `clInitializeApplication` 和 `mclTerminateApplication` 函数(更详细的信息参看 8.1.3 节)。

下面是有效使用的-R 选项:

```
mcc -m -R "-nojvm -nojit" -v foo.m
mcc -m -R "-nojvm" -v -R "-nojit" foo.m
mcc -m -R -nojvm -R -nojit foo.m
mcc -m -R -nojvm -v foo.m
mcc -m -R -nojvm -R -nojit foo.m
```

注:nojvm 或 nojit 前面的横线必不可少。

而这个例子是非法的:

```
mcc -m -R -nojvm -nojit foo.m
```

17)-T 指定目标程序级

指定输出的目标程序级和类型,用如下形式:

```
-T target
```

定义输出类型。有效的 target 值见表 1.2。

表 1.2 有效的 target 值

target 值	描 述
codegen	生成 C/C++ 封装文件,默认值为 codegen
compile:exe	具有 codegen 的功能,再加上编译 C/C++ 文件成为适合于连接到独立可执行程序的目标形式
compile:lib	具有 codegen 的功能,再加上编译 C/C++ 文件成为适合于连接到共享库/ DLL 的目标形式
link:exe	与 compile:exe 一样,还能连接目标文件到独立可执行程序
link:lib	与 compile:lib 一样,还能连接目标文件到共享库/DLL
注:exe 使用 mbuild 生成可执行程序,lib 使用 mbuild 生成共享库。	

18)-v(Verbose)编译信息

显示编译阶段的信息,包括:

- (1)编译程序的版本号。
- (2)被处理的源文件名。
- (3)被建立的输出文件名。
- (4)mbuild 的调用。

-v 传送它的选项给 mbuild,并显示有关 mbuild 的信息。

19)-w 警告信息

显示警告信息,用语法形式

```
-w option[:<msg>]
```

控制警告信息的显示。表 1.3 列出了有效的警告选项。

表 1.3 有效的警告选项

句 法 形 式	描 述
-w list	形成一个表,表中是使用了 nable, disable 和 error 选项的警告信息字符串(<string>)。参考“第 10 章错误和警告信息”