



固高科技（深圳）有限公司

地 址：深圳市高新技术产业园南区深港产学研基地西座
二层 W211 室

电 话：0755-26970823 26970817 26970824

传 真：0755-26970821

电子邮件：support@gogoltech.com

网 址：<http://www.gogoltech.com.cn>

固高科技（香港）有限公司

地 址：香港九龙清水湾香港科技大学新翼楼 3639 室

电 话：(852) 2358-1033

传 真：(852) 2358-4931

电子邮件：info@gogoltech.com

网 址：<http://www.gogoltech.com/>

I0 扩展模块编程手册

务必将此手册交给用户

- 非常感谢您选购扩展模块
- 在您使用之前，请仔细阅读此手册，确保正确使用。
- 请将此手册妥善保存，以备随时查阅。


版权申明

固高科技有限公司
保留所有权力

固高科技有限公司（以下简称固高科技）保留在不事先通知的情况下，修改本手册中的产品和产品规格等文件的权力。

固高科技不承担由于使用本手册或本产品不当，所造成直接的、间接的、特殊的、附带的或相应产生的损失或责任。

固高科技具有本产品及其软件的专利权、版权和其它知识产权。未经授权，不得直接或者间接地复制、制造、加工、使用本产品及其相关部分。

 注意	运动中的机器有危险！使用者有责任在机器中设计有效的出错处理和安全保护机制，固高科技没有义务或责任对由此造成的附带的或相应产生的损失负责。
---	--

前言

感谢选用固高运动控制器和扩展模块

为回报客户，我们将以品质一流的运动控制器、完善的售后服务、高效的技术支持，帮助您建立自己的控制系统。

固高产品的更多信息

固高科技的网址是 <http://www.googoltech.com.cn>。在我们的网页上可以得到更多关于公司和产品的信息，包括：公司简介、产品介绍、技术支持、产品最新发布等等。

您也可以通过电话（0755－26970839）咨询关于公司和产品的更多信息。

技术支持和售后服务

您可以通过以下途径获得我们的技术支持和售后服务：

- ◆ 电子邮件： support@googoltech.com;
- ◆ 电话： (0755) 26970835
- ◆ 发函至： 深圳市高新技术产业园南区园深港产学研基地西座二楼 W211 室
固高科技（深圳）有限公司
邮编： 518057

编程手册的用途

用户通过阅读本手册，能够了解扩展模块的动态链库的使用，指令列表，指令说明，配置文件，使用方式说明等模块基本编程功能和特征。

编程手册的使用对象

本编程手册适用于，具有硬件基本知识，对控制有一定了解的工程人员。

目 录

版权申明	1
前言	0
	感谢选用固高运动控制器和扩展模块..... 0
	固高产品的更多信息..... 0
	技术支持和售后服务..... 0
	编程手册的用途..... 0
	编程手册的使用对象..... 0
第一章	扩展模块动态链接库的使用..... 2
1.1	DOS系统下函数库的使用..... 2
1.2	Windows系统下动态连接库的使用..... 3
	1.2.1 Visual C++中的使用..... 3
	1.2.2 Delphi中的使用..... 3
第二章	指令列表..... 5
第三章	指令说明..... 6
第四章	配置文件..... 12
	4.1 文件结构说明..... 12
	4.2 通过Demo生成配置文件..... 13
第五章	使用方式说明..... 15
第六章	使用Demo测试IO扩展模块..... 16

第一章 扩展模块动态链接库的使用

扩展模块提供 DOS 下的 C++ 语言函数库和 Windows 下的动态链接库。用户只要调用函数库中的指令，就可以实现运动控制器的各种功能。下面分别讲述 DOS、Windows 系统下函数库的使用方法。

1.1 DOS 系统下函数库的使用

固高 IO 扩展模块在 DOS 系统下的函数库和头文件存放在产品配套光盘的 lib/dos/gt 或 lib/dos/gts 文件夹下。gt 文件夹下的库适用于固高 GES、GT、GE、GEP 等运动控制器平台，gts 文件夹下的库适用于固高 GTS 系列运动控制平台，请根据您的控制器类型，选择适用的函数库及头文件。

每套库文件夹下都有七个文件，分别为：

extmdl.h	头文件
extmdlt.lib	微模式(Tiny)的函数库
extmdls.lib	小模式(Small)的函数库
extmdl.m.lib	中模式(Medium)的函数库
extmdlc.lib	紧凑模式(Compact)的函数库
extmdll.lib	大模式(Large)的函数库
extmdlh.lib	巨大模式(Huge)的函数库

该函数库是用 Borland C++ 3.1 编译生成的，用户可在 Borland C++ 3.1 或更高版本的开发环境下链接该函数库(用户 CD 中提供 Borland C++ 3.1 以及 Borland C++ 4.5 在各个编译模式下的静态库，请根据您的开发环境使用相应的库)。

下面以 Borland C++ 3.1 为例，介绍使用函数库开发应用程序的流程，如下：

1. 确定编译模式——

- 启动 Borland C++ 3.1；
- 选择“Project”菜单下“Open Project”，打开已有工程文件或建立新工程文件；
- 选择“Options”菜单下“Compiler”中的“Code generation...”菜单项；
- 在“Model”一栏中选择当前工程文件的编译模式；

2. 添加函数库——

- 选择相应编译模式的函数库和头文件，复制到用户当前的工程文件夹中；
- 选择“Window”菜单下的“Project”菜单项，切换到工程窗口；
- 选择“Project”菜单下的“Add Item...”菜单项；
- 在“Name”栏中输入“*.lib”后回车；
- 在“Files”栏中选择将要加入工程的函数库，然后点击“Add”按钮。

3. 添加源程序文件——

- 选择“File”菜单下的“New”菜单项，新建一个.CPP 或.C 文件；
- 在新建的.CPP 或.C 文件中加入函数库头文件的声明，例如：

```
#include "extmdl.h"; //声明头文件
```

- 保存该文件，然后将该源程序文件添加到当前工程当中。

1.2 Windows 系统下动态连接库的使用

在 Windows 系统下使用 IO 扩展模块，首先要安装驱动程序，驱动程序存放在产品配套光盘的 Driver 文件夹下。

扩展模块指令函数动态链接库存放在产品配套光盘的 lib\vc（或 lib\Delphi）文件夹下。动态链接库文件名为 ExtMdl.dll。

在 Windows 2000/XP 等系统下，用户可以使用任何能够支持动态链接库的开发工具来开发应用程序。下面分别以 Visual C++ 和 Delphi 为例讲解如何在这些开发工具中使用运动控制器的动态链接库。

1.2.1 Visual C++ 中的使用

1. 启动 Visual C++，新建一个工程；
2. 将产品配套光盘 Windows\VC 文件夹中的动态链接库、头文件和 lib 文件复制到工程文件夹中；
3. 选择“Project”菜单下的“Settings...”菜单项；
4. 切换到“Link”标签页，在“Object/library modules”栏中输入 lib 文件名（例如 ExtMdl.lib）；
5. 在应用程序文件中加入函数库头文件的声明，例如：
#include "ExtMdl.h"
6. 至此，用户就可以在 Visual C++ 中调用函数库中的任何函数，开始编写应用程序。

1.2.2 Delphi 中的使用

1. 启动 Delphi，新建一个工程；
2. 将产品配套光盘 lib\Delphi 文件夹中的动态链接库和函数声明文件复制到工程文件夹中；
3. 选择“Project”菜单下的“Add to Project...”菜单项；
4. 将函数声明文件添加到工程当中；
5. 在代码编辑窗口中，切换到用户的单元文件；
6. 选择“File”菜单下的“Use Unit...”菜单项，添加对函数声明文件的引用；
7. 至此，用户就可以在 Delphi 中调用函数库中的任何函数，开始编写应用程序。

另外，在 WinCE 系统下，用户也可以通过 EVC 开发 IO 扩展模块应用程序，相应的库在 lib\ce 文件夹下，动态链接库文件名为 ExtMdl.dll，使用方法与在 MSVC++ 6.0 中使用方法相仿。

第二章 指令列表

指令	说明
GT_OpenExtMdl	打开扩展模块
GT_CloseExtMdl	关闭扩展模块
GT_SwitchtoCardNoExtMdl	切换当前扩展模块的卡号
GT_ResetExtMdl	复位扩展模块
GT_LoadExtConfig	加载扩展模块系统配置文件
GT_SetExtIoValue	设置数字量 IO 输出值
GT_GetExtIoValue	读取数字量 IO 输入值
GT_SetExtIoBit	按位设置数字量 IO 输出值
GT_GetExtIoBit	按位读取数字量 IO 输入值
GT_GetExtAdValue	读取 AD 输入的转换值
GT_GetExtAdVoltage	读取 AD 输入的电压值
GT_SetExtDaValue	设置 DA 输出的转换值
GT_SetExtDaVoltage	设置 DA 输出的电压值
GT_GetStsExtMdl	获取扩展模块的状态
GT_SetExtMdlMode	设置扩展模块的工作模式（仅 GTS）
GT_GetExtMdlMode	读取扩展模块的工作模式（仅 GTS）

第三章 指令说明

GT_OpenExtMdl

指令原型: short GT_OpenExtMdl(char *pDllName=NULL)

指令说明: 调用 GT_OpenExtMdl 指令打开扩展模块, 以获取对扩展模块的访问权。当扩展模块与运动控制器的运动控制功能同时使用时, 需要先打开运动控制功能, 再打开扩展模块功能。打开扩展模块功能时, 需要指定运动控制功能所使用的动态链接库的文件名称。

指令参数:

*pDllName: 需要指定的运动控制功能动态链接库的文件名称。

适用板卡: 含有扩展模块功能的运动控制器

指令示例: 请参考第五章相关部分或者用户光盘中的相关例程。

GT_CloseExtMdl

指令原型: GT_CloseExtMdl()

指令说明: 关闭扩展模块, 当不再使用扩展模块时, 调用该指令以释放扩展模块的访问权。

指令参数: 无。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

GT_SwitchtoCardNoExtMdl

指令原型: GT_SwitchtoCardNoExtMdl(short card)

指令说明: 该指令用于切换当前扩展模块相关的运动控制器卡号。

指令参数:

card: 将被设为当前运动控制器的卡号。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

GT_ResetExtMdl

指令原型: GT_ResetExtMdl()

指令说明: 该指令用于复位扩展模块, 调用该指令后, 系统会扫描所有连接的扩展模块, 所有连接的工作正常的模块都会被使能。

指令参数:

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

GT_LoadExtConfig

指令原型: GT_LoadExtConfig(char *pFileName)

指令说明: 加载扩展模块系统配置文件, 关于该文件的格式以及内容说明, 参照第四节。

指令参数:

*pFileName: 配置文件的文件名。

适用板卡：含有扩展模块功能的运动控制器

指令示例：建议通过 IO 扩展模块的 Demo 生成配置文件。

```
short rtn;  
rtn = GT_LoadExtConfig ("test.cfg");  
//加载配置文件，文件路径当前目录下的 test.cfg
```

GT_SetExtIoValue

指令原型：GT_SetExtIoValue(short mdl,unsigned short value)

指令说明：设置数字量 IO 扩展模块的输出值，按字(16 位)操作。

指令参数：

mdl：模块的编号，即参照配置文件，该数字量 IO 的模块编号。范围[0, 15]。

Value：输出到数字量 IO 上的 16 位值。

适用板卡：含有扩展模块功能的运动控制器

指令示例：

```
short rtn;  
unsigned short value;  
value = 0x55AA;  
rtn = GT_SetExtIoValue (0,value);  
//设置第一个模块的数字量输出为 0x55AA  
//第一个模块必须为数字量模块
```

GT_GetExtIoValue

指令原型：GT_GetExtIoValue(short mdl,unsigned short *pValue)

指令说明：读取数字量 IO 扩展模块的输入值，按字(16 位)操作。

指令参数：

mdl：模块的编号，即参照配置文件，该数字量 IO 的模块编号。

*pValue：读取的数字量 IO 上的 16 位值。

适用板卡：含有扩展模块功能的运动控制器

指令示例：

```
short rtn;  
unsigned short value;  
rtn = GT_GetExtIoValue (0,&value);  
//读取第一个模块的数字量输入值  
//第一个模块必须为数字量模块
```

GT_SetExtIoBit

指令原型：GT_SetExtIoBit(short mdl,short index,unsigned short value)

指令说明：设置数字量 IO 扩展模块的输出值，按位操作。

指令参数：

mdl：模块的编号，即参照配置文件，该数字量 IO 的模块编号，范围[0, 15]。

Index：所要操作的位，取值范围[0,15]。

Value：所要输出的 IO 的值，取值范围[0,1]。

适用板卡：含有扩展模块功能的运动控制器

指令示例：

```
short rtn;
unsigned short value;
value = 1;
rtn = GT_SetExtIoBit (0,0,value);
//设置第一个模块的第一个通道的数字量输出值为 1
//第一个模块必须为数字量模块
```

GT_GetExtIoBit

指令原型: GT_GetExtIoBit(short mdl,short index,unsigned short *pValue)

指令说明: 读取数字量 IO 扩展模块的输入值, 按位操作。

指令参数:

mdl: 模块的编号, 即参照配置文件, 该数字量 IO 的模块编号, 范围[0, 15]。

Index: 所要操作的位, 取值范围[0,15]。

*pValue: 所读取的输入 IO 相应位的值。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

```
short rtn;
unsigned short value;
rtn = GT_GetExtIoBit (0,0,&value);
//读取第一个模块的第一个通道的数字量输入值
//第一个模块必须为数字量模块
```

GT_GetExtAdValue

指令原型: GT_GetExtAdValue(short mdl,short chn,unsigned short *pValue)

指令说明: 读取 AD 扩展模块输入的转换值。

指令参数:

mdl: 模块的编号, 即参照配置文件, 该 AD 扩展模块的模块编号, 范围[0, 15]。

Chn: 需要读取的 AD 扩展模块中的通道号。

*pValue: 读取的 AD 扩展模块转换位的值。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

```
short rtn;
unsigned short value;
rtn = GT_GetExtAdValue (0,0,&value);
//读取第 0 号模块的第一个通道输入值
//电压值为: vol = volMin+(value/65535)*(volMax-volMin)
//其中 volMin、volMax 代表该通道设置的最小电压值和最大电压值
```

GT_GetExtAdVoltage

指令原型: GT_GetExtAdVoltage(short mdl,short chn,double *pValue)

指令说明: 读取 AD 扩展模块输入的电压值。

指令参数:

mdl: 模块的编号, 即参照配置文件, 该 AD 扩展模块的模块编号, 范围[0, 15]。

Chn: 需要读取的 AD 扩展模块中的通道号。

***pValue:** 读取的 AD 扩展模块输入的电压值。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

```
short rtn;
```

```
double vol;
```

```
rtn = GT_GetExtAdVoltage (0,0,&vol);
```

```
//读取第 0 号模块的第一个通道电压输入值
```

GT_SetExtDaValue

指令原型: `GT_SetExtDaValue(short mdl,short chn, unsigned short value)`

指令说明: 设置 DA 扩展模块输出电压的转换值。

指令参数:

mdl: 模块的编号, 即参照配置文件, 该 DA 扩展模块的模块编号, 范围[0, 15]。

Chn: 需要设置的 DA 扩展模块中的通道号, 范围[0,模块可用通道数-1]。

Value: 设置的 DA 扩展模块输出值。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

```
short rtn;
```

```
rtn = GT_SetExtDaValue (0,0,20000);
```

```
//第 0 号模块的第一个通道输出值为 20000,
```

```
//电压值为: vol = volMin+(value/65535)*(volMax-volMin)
```

```
//其中 volMin、volMax 代表该通道设置的最小电压值和最大电压值
```

GT_SetExtDaVoltage

指令原型: `GT_SetExtDaVoltage(short mdl,short chn,double value)`

指令说明: 设置 DA 扩展模块输出电压的电压值。

指令参数:

mdl: 模块的编号, 即参照配置文件, 该 DA 扩展模块的模块编号, 范围[0, 15]。

Chn: 需要设置的 DA 扩展模块中的通道号。

Value: 设置的 DA 扩展模块输出电压值, 范围[0,模块可用通道数-1]。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

```
short rtn;
```

```
rtn = GT_SetExtDaVoltage(0,0,2.0);
```

```
//第 0 号模块的第一个通道输出电压 2V
```

GT_GetStsExtMdl

指令原型: `GT_GetStsExtMdl(short mdl,short chn,unsigned short *pStatus)`

指令说明: 读取扩展模块的状态, 当该模块通信工作正常时, 读回的值为 0, 否则为 1。

指令参数:

mdl: 模块的编号, 即参照配置文件, 相应扩展模块的模块编号, 范围[0, 15]。

Chn: 当模块为 AD 模块或者 DA 模块时, 需要制定的通道号。当为数字量输入/输出模块时, 该参数无效。

*pStatus: 读取的状态值, 0: 工作正常; 1: 有故障。

适用板卡: 含有扩展模块功能的运动控制器

指令示例:

```
short rtn;
unsigned short sts;
rtn = GT_GetStsExtMdl (0,0,&sts);
//读取第一个模块的第一个通道的在线状态
```

GT_SetExtMdlMode

指令原型: GT_SetExtMdlMode(short mode);

指令说明: 设置当前扩展模块的工作模式。

扩展模块有两种工作模式, 直接控制模式和内部控制模式, 默认为直接控制模式。

如果用户希望通过运动控制程序 (运动控制程序是固高 GTS 系列运动控制器的一个功能, 详情请参考固高 GTS 运动控制器的相关资料), 则需要将扩展模块的工作方式设置为内部工作模式。

其他情况下, 请使用直接控制模式。

注意: 该指令必须用于 GT_LoadExtConfig 指令之后。

指令参数:

mode: 模式, 0—直接模式, 1—内部控制模块。

适用板卡: 仅适用于 GTS 系列运动控制器

指令示例: 使用内部控制模式的示例代码如下, 具体请参考用户光盘中的相关例程。

```
short rtn;

//打开扩展模块;
rtn = GT_OpenExtMdl("gts.dll");
if (rtn)
{
    strTemp.Format(_T("GT_OpenExtMdl error,error code = %d"),rtn);
    MessageBox(strTemp);
    return;
}

//配置模块
rtn = GT_LoadExtConfig("extmdl_test.cfg");
if (rtn)
{
    strTemp.Format(_T("GT_LoadExtMdlConfig error,error code = %d"),rtn);
    MessageBox(strTemp);
    return;
}

//设置模块的工作方式为内部模式
rtn = GT_SetExtMdlMode(1);
if (rtn)
```

```
{  
    strTemp.Format(_T("GT_SetExtMdlMode error,error code = %d"),rtn);  
    MessageBox(strTemp);  
    return;  
}
```

GT_GetExtMdlMode

指令原型: GT_GetExtMdlMode(short *pMode);

指令说明: 读取当前扩展模块的工作模式。

指令参数:

pMode: 模式, 0—直接模式, 1—内部控制模块。

适用板卡: 仅适用于 GTS 系列运动控制器

指令示例:

第四章 配置文件

4.1 文件结构说明

扩展模块包括数字量 IO 扩展模块、AD 扩展模块、DA 扩展模块等，所以在使用扩展模块之前，要根据具体选用的硬件类型以及硬件的地址设置来配置扩展模块的软件部分，可以通过指令 GT_LoadExtConfig 将配置文件导入软件模块内部。配置文件的编写如下：

```
[module0]
type=6
address=0
adChannels=2
adMaxVoltage=5
adMinVoltage=0

daChannels=4
daMaxVoltage=5
daMinVoltage=0
[module1]
type=3
address=4
.....
```

配置文件中主要包括以下一些部分：

1. [module x]：配置的模块的编号， x 的取值范围[0,15]。所要配置的该模块的编号。
2. Type：配置的模块的类型，取值范围[1,6]，具体的意义如下：
 - 1：数字量输入模块
 - 2：数字量输出模块
 - 3：数字量输入/输出模块
 - 4：模拟量输入模块
 - 5：模拟量输出模块
 - 6：模拟量输入/输出模块
3. Address：该模块的首地址，与扩展模块硬件上的拨码开关对应，取值范围[0,15]。
4. adChannels：当模块类型为模拟量输入模块或者模拟量输入/输出模块时，模拟量输入的通道数。
5. adMaxVoltage：当模块类型为模拟量输入模块或者模拟量输入/输出模块时，AD 量程的最大电压值。
6. adMinVoltage：当模块类型为模拟量输入模块或者模拟量输入/输出模块时，AD 量程的最小电压值。
7. daChannels：当模块类型为模拟量输出模块或者模拟量输入/输出模块时，模拟量输出的通道数。
8. daMaxVoltage：当模块类型为模拟量输出模块或者模拟量输入/输出模块时，DA 所能输出电压的最大值。

9. daMinVoltage: 当模块类型为模拟量输出模块或者模拟量输入/输出模块时, DA 所能输出电压的最小值。

10. defaultoutput: 当模块为数字量输出模块时, 输出的默认值。

说明:

根据模块的类型, 设置相应的配置信息。如数字量输入、数字量输出和数字量输入/输出类型的模块, 则只用指定 type 值和模块首地址、默认输出值, 不需要指定其他的信息; 模拟量输入需要指定与模拟量输入相关的信息; 模拟量输出需要指定与模拟量输出相关的信息; 模拟量输入/输出则需要指定所有的信息。

配置文件应该与 ExtMdl.dll 放在相同的文件夹下使用。

4.2 通过 Demo 生成配置文件

IO 扩展模块的 Demo 存放于用户光盘的 Demo 文件夹下。

下面介绍通过 Demo 程序的配置模块生成配置文件。

第一步: 双击启动 Demo, 点击‘视图’—‘模块配置’菜单项, 弹出模块配置对话框, 如下图所示。



第二步: 根据用户搭建系统的配置情况, 完成模块的配置。

第三步: 生成配置文件。点击配置模块的菜单项‘文件’—‘写入到文件’, 在弹出的文件保存对话框中输入配置文件名, 保存配置文件。

例如, 用户当前系统中用到 2 个模块, 第一个为数字量模块 (编码地址为 0), 第二个为 2AD/2AD 的模拟量模块 (编码地址为 4, 输入输出范围都是 $\pm 10V$), 则配置过程如下:

- 1、启动模块配置对话框。
- 2、模块下拉对话框选择 0, 起始地址输入 0, 类型选择 DI/DO 模块。
- 3、模块下拉对话框选择 1, 起始地址输入 4, 类型选择 AD/DA 模块, 根据该模拟量模块编码开关的状态完成参数设置。这样就完成了两个模块的配置。
- 4、点击配置模块的菜单‘文件’—‘写入到文件’将配置信息保存到文件。

第五章 使用方式说明

在 Windows（不包括 DOS，Dos 下直接调用 GT_OpenExtMdl 和 GT_Open 即可）下开发 IO 扩展模块的应用程序时，需要根据用户对 IO 扩展模块的使用情况分以下两种情况对待。

1. 独立使用扩展模块

所谓独立使用扩展模块，是仅使用运动控制器的扩展模块，不使用运动控制功能，则需要先打开扩展模块，进行操作，操作完毕之后，关闭扩展模块。示例代码如下：

```
GT_OpenExtMdl();
GT_LoadExtConfig("ExtMdl.cfg");
.....
GT_CloseExtMdl();
```

2. 与运动控制功能同时使用

当与运动控制器的运动控制功能同时使用时，需要先打开运动控制器的运动控制功能，再打开扩展模块，并且还要根据所使用运动控制功能的动态链接库设置 GT_OpenExtMdl 的参数；关闭相关功能时，需要先关闭扩展模块，再关闭运动控制器的运动控制功能。示例代码如下：

```
GT_Open();                // 打开运动控制器
GT_OpenExtMdl("gep.dll"); // 使用 GE 点位运动控制功能
GT_LoadExtConfig("ExtMdl.cfg");
.....
GT_CloseExtMdl();
GT_Close();
```

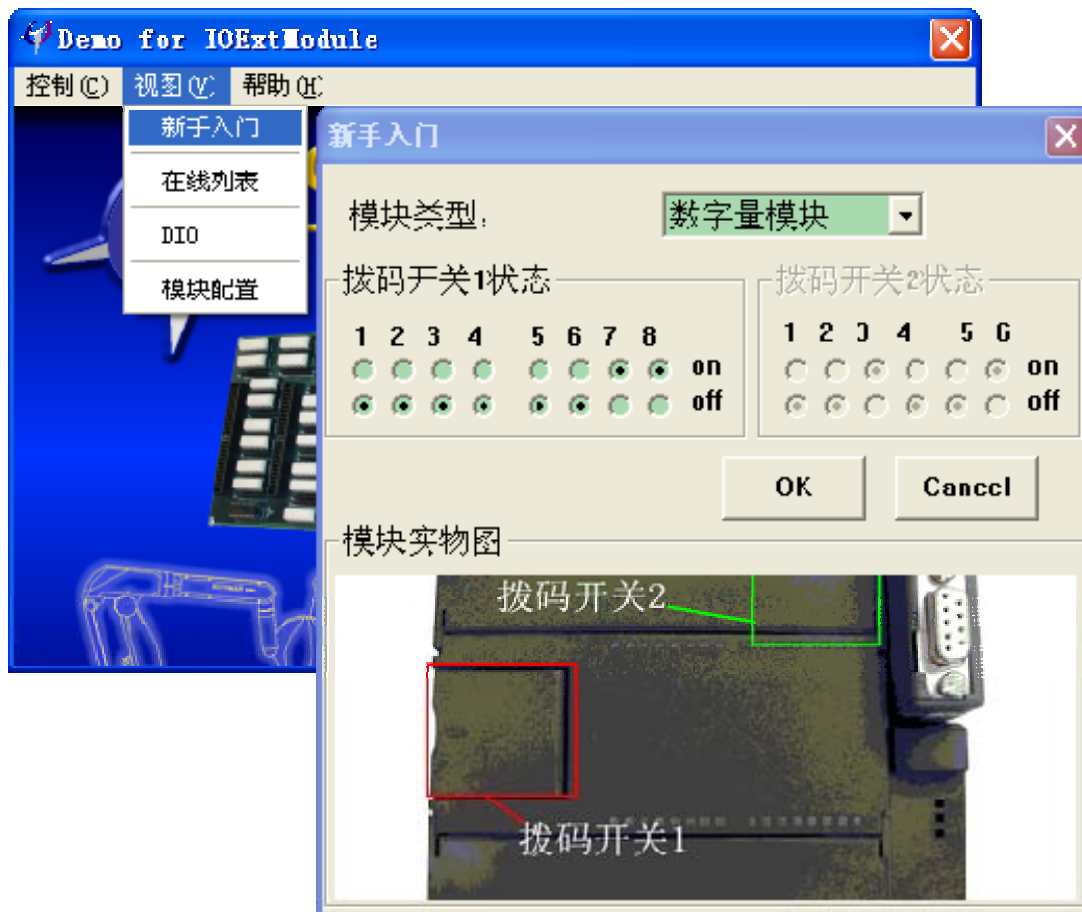
请参考例程，相关例程存放在用户光盘的 example 目录下。

第六章 使用 Demo 测试 IO 扩展模块

IO 扩展模块的 Demo 存放于用户光盘的 Demo 文件夹下。

双击启动 Demo，提示‘打开运动控制器的扩展模块失败’，请检查控制器及控制器的驱动是否正确安装。

点击菜单‘视图’—‘新手入门’，弹出如下图所示模块对话框。



根据测试模块的配置，选择模块类型、编码开关的状态等，点击‘OK’按钮。

用户可以在弹出的模块测试对话框上对该模块进行测试。

点击菜单‘视图’—‘在线列表’，弹出的对话框会显示所有在线的模块及通道，绿色代表通道在线，点击‘刷新’按钮刷新。

Demo 的其他功能的使用请参见 Demo 的帮助文档（按下‘F1’或者点击菜单项‘帮助’—‘使用帮助’）。

固高科技（深圳）有限公司

地 址：深圳市高新技术产业园南区深港产学研基地
西座二层 W211 室

电 话：0755-26970823 26970819 26970824

传 真：0755-26970821

电子邮件：support@gogoltech.com

网 址：<http://www.gogoltech.com.cn/>

固高科技（香港）有限公司

地 址：香港九龙清水湾香港科技大学新翼楼 3639 室

电 话：(852) 2358-1033

传 真：(852) 2358-4931

电子邮件：info@gogoltech.com

网 址：<http://www.gogoltech.com/>