1509, TFDCN-175-001, 001

NACHI

FD/CFD 控制装置用运行库模块
OpenNR-IF
操作说明书

第 1 版

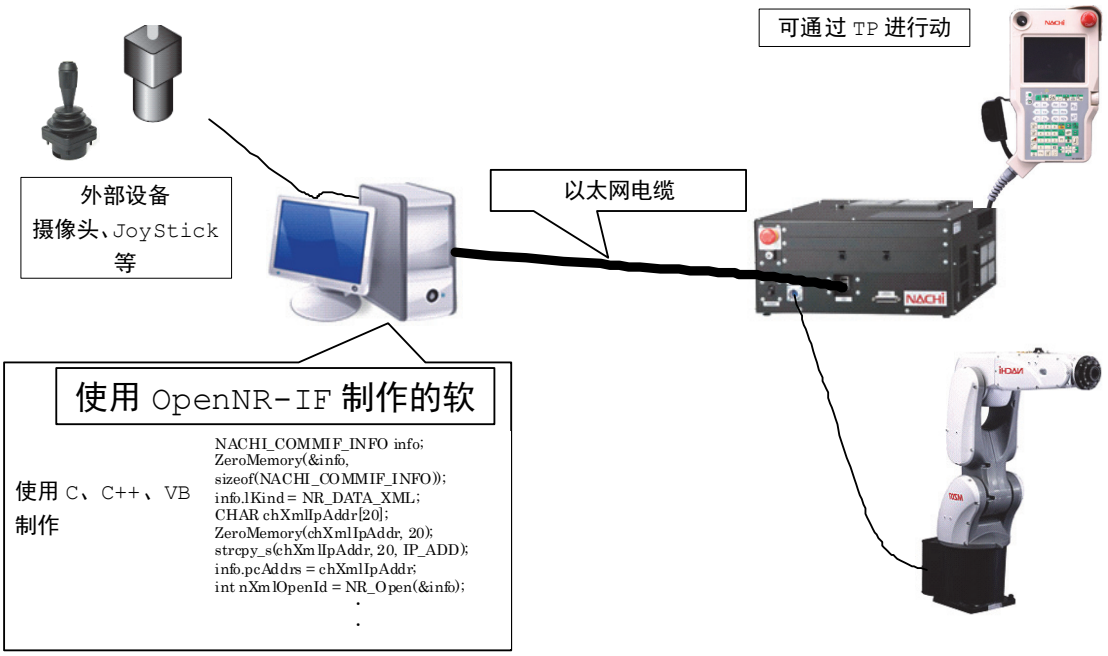
・使用机器人之前，请务必仔细阅读本操作说明书，遵守所有的安全相关事项并按照本文的指示进行操作。
・只限接受过相应培训的人员可以对本机器人进行相关处理（安装、操作、维护等）。
・使用本机器人时，请务必遵守各国的工业机器人相关法律及安全相关法律。
・务必将本说明书发给实际进行操作的人员。
・对本说明书的疑点及本机器人的售后服务，请咨询封底所述的本公司各服务中心。

株式会社 不二越

目　　录

1. OpenNR-IF 的概要

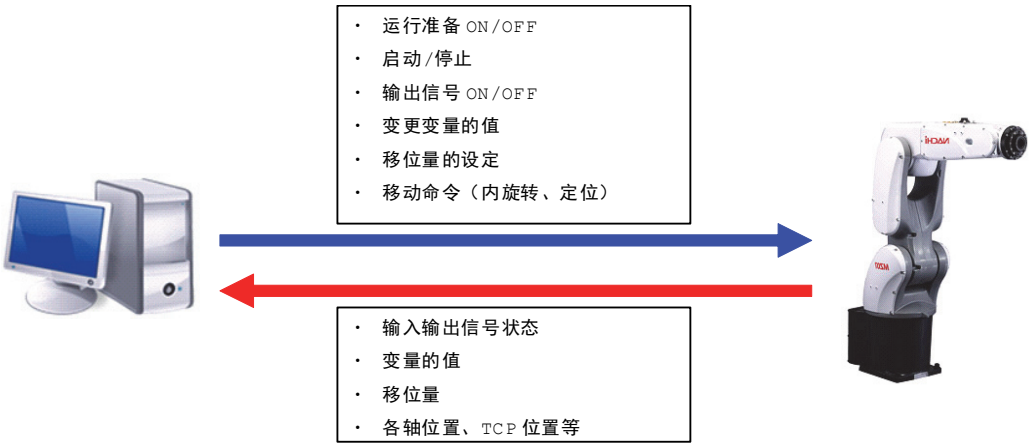
OpenNR-IF 是便于制作连接至 FD 控制装置及外部装置（摄像头等）的应用软件的运行库模块。
可搭载各类应用以满足各种需求，例如提高与视觉装置的通信（用户任务）以及制作独立画面（接口画面）这类既有功能的性能，通过电脑远程监控及实现可追溯性。



机器人操作＆监控

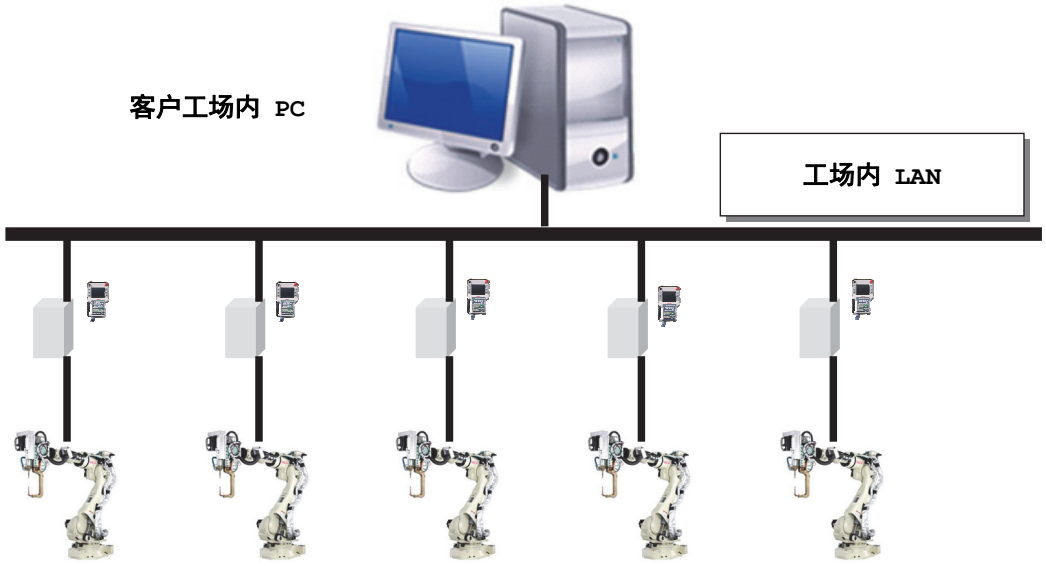
可使用标准的以太网端口进行机器人操作（包括再生）及各种监控。

- 可指示机器人的移动命令（绝对位置、相对位置）。
  也可进行内旋转及定位。
- 可获取机器人的状态（输入输出信号、变量、移位量等）。
- 可进行运转准备的 ON/OFF、启动/停止。
- 可控制输出信号的 ON/OFF 及变更各种变量参数。



远程监控、可追溯性

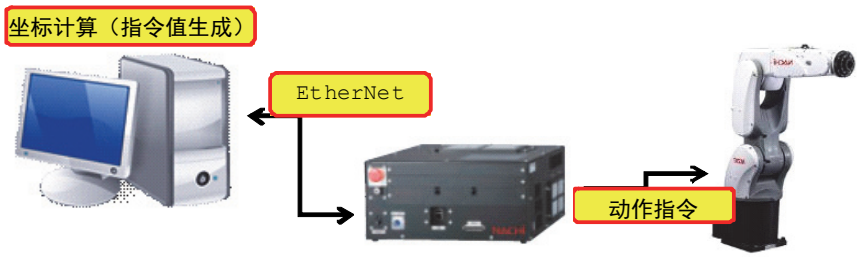
通过从机器人处收集/累积信息，不仅可监视运行状况及错误发生状况、焊接质量，还可监视机器人的电流及速度、寿命、剩余寿命的长期变化。



实时跟踪

通过追加增设的以太网基板，可在各插值周期中进行控制。（通过动作捕捉器的外部跟踪动作。）

・可对机器人位置（各轴角度、TCP 位置）进行直接插值。
・可获取输入信号状态与控制输出信号状态的 ON/OFF。



POINT

如果需要使用本功能，控制装置内必须配备扩展以太网基板（选购）。

1.1 运行环境

安装 OpenNR-IF 前，请确认所使用电脑是否满足以下条件。

表 1.1 OpenNR-IF 的运行环境

要素	规格
OS	Windows ® XP, Windows ® 7, Windows ® 8
CPU	主频 1.0GHz 以上的 CPU
内存	大于 1GB
硬盘容量	需要 70MB 以上的剩余容量
开发语言	Microsoft VisualC++ 2005 Microsoft VisualC++ 2010
其他	需检查许可证，应满足下述条件。 标准规格 ： 以太网 LAN 适配器 硬件加密锁规格 ： USB 接口

注 1 Windows 是美国 Microsoft Corporation 在美国及其他国家的注册商标。

1.2 支持控制器

支持 FD / CFD 控制装置的软件版本 V4.21 以后的版本。

1.3 许可证

许可证可选择许可证文件及 USB 硬件加密锁这 2 种形式。关于购买许可证相关事宜，请参照下述内容后，咨询本公司营业窗口。

通过连接许可证文件的许可证认证

请将所使用电脑的 MAC 地址发送到安装 CD 内的本公司联系地址。回复时将发行许可证文件。通过将许可证文件复制至与 OpenNR-IF.DLL 相同的文件夹下，即可认证、执行许可证文件。

许可证文件仅限在向本公司申请的电脑上使用。

MAC 地址的确认方法

MAC 地址可通过【Program】【Accessories】【Command Prompt】，输入"ipconfig/all"获取。Physical Address 即为其值。（例：00-10-C0-F6-B7-E9）

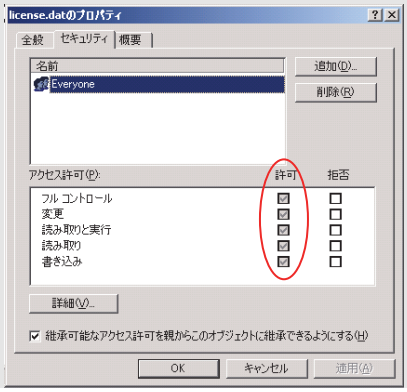
```
C:¥ipconfig /all
（中略）
 Physical Address . . . . . : 00-10-C0-F6-B7-E9
 Dhcp Enabled . . . . . : No
 IP Addres . . . . . : 192.168.1.1
 Subnet Mask . . . . . : 255.255.255.0
```

MAC 地址

POINT

已将许可证文件复制至正确位置，但仍发生许可证检查错误时，请确认以下内容。

1. 请确认文件名是否为"license.dat"。应特别注意扩展名。
2. 请确认文件夹中是否存在"dongle.dat"文件。如果没有"dongle.dat"文件，许可证将不会被认证。
3. 请确认许可证文件的安全属性设置画面中是否已全部勾选许可。



通过 USB 硬件加密锁的许可证认证

请将专用的硬件加密锁插入 USB 接口后进行使用。插入硬件加密锁后，可在安装有 OpenNR-IF 的任意电脑上使用。但在使用中不可拔下硬件加密锁。

硬件加密锁使用日本 SafeNet 株式会社（原（株）阿拉丁日本）的产品。请解压安装 CD 中的"HASP4_driver_cmdline.zip"或"HASP4_driver_setup.zip"后执行，安装驱动。

接通 PC 电源后，请确认硬件加密锁的 LED 已点亮。LED 未点亮时，请重新插入硬件加密锁。
如未正确插入，将无法进行许可证的认证。

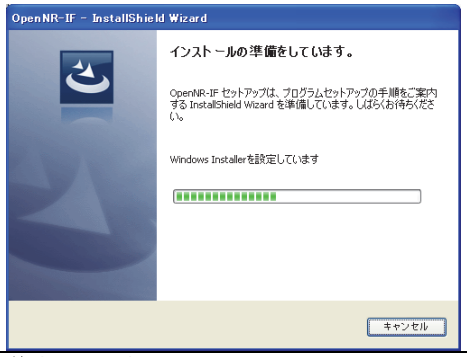
使用命令行的安装程序时，必须指定参数"-i"。
有关详细内容，在执行 hinstall.exe 时将显示对话框。

2． 设置

2.1 安装和版本升级

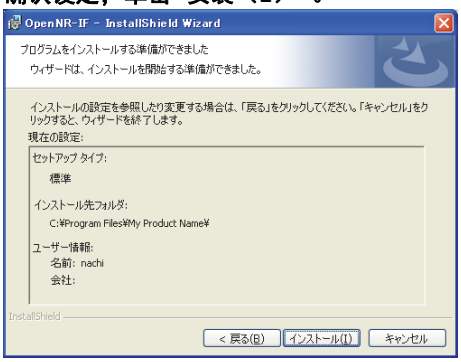
安装方法如下所示。
已安装有 OpenNR-IF 时，请先卸载原有的 OpenNR-IF 后，再安装新的 OpenNR-IF。
请事先准备好安装程序。安装程序由本公司提供，位于安装盘的 Setup 文件夹内。由以下数个文件组成。

1 **执行 setup.exe。**
持有安装 CD 时，插入 CD 后将会自动执行。
>> 启动安装程序。

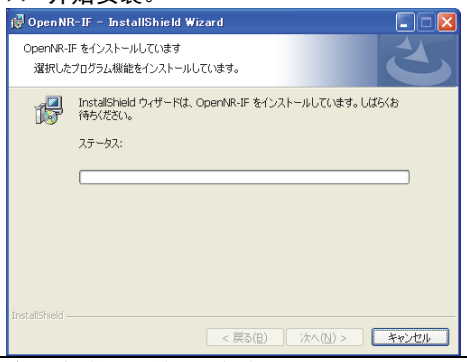


2 **单击"下一步（N）"。**

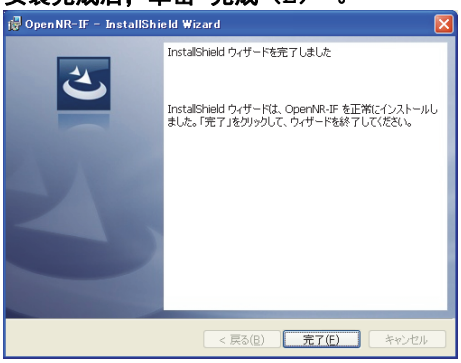
3 **确认设定，单击"安装（I）"。**



>> 开始安装。



6 **安装完成后，单击"完成（E）"。**



2.2 卸载

从"控制面板"的"添加/删除程序"中进行卸载。

2.3 方案的设定例

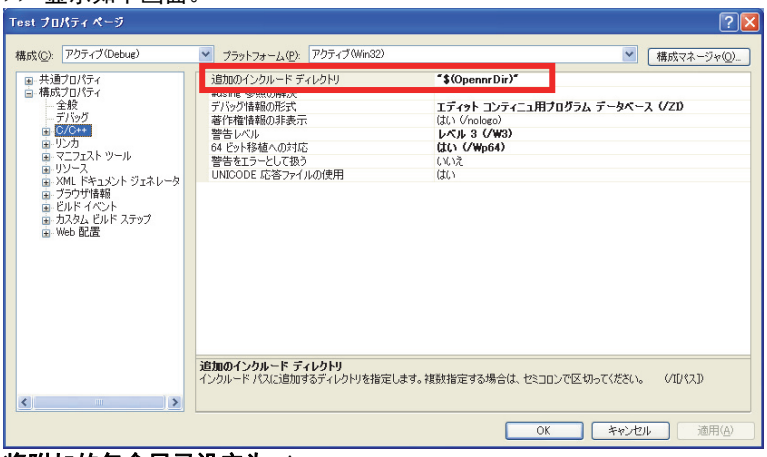
使用 Microsoft Visual Studio 2005 时的设定方法如下所示。

注 1） Windows 是美国 Microsoft Corporation 在美国及其他国家的注册商标。

1 **打开方案的属性。**

选择"配置属性:C/C++:整体"。

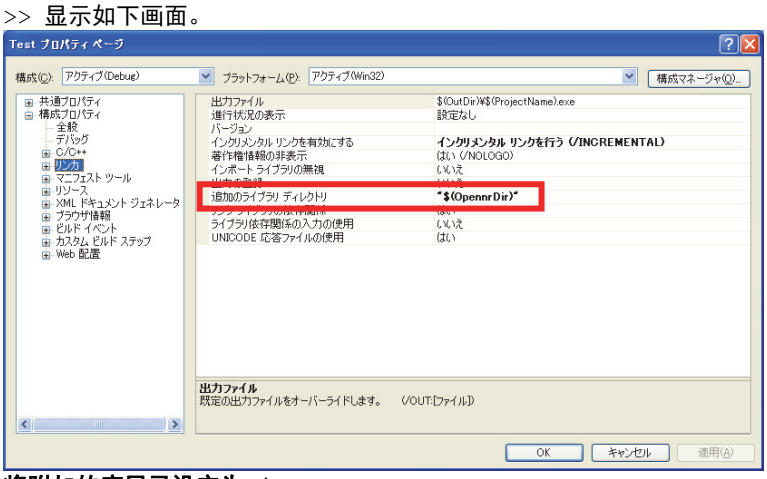
>> 显示如下画面。



将附加的包含目录设定为"$(OpennrDir)"。

2 **选择"配置属性:链接:整体"。**

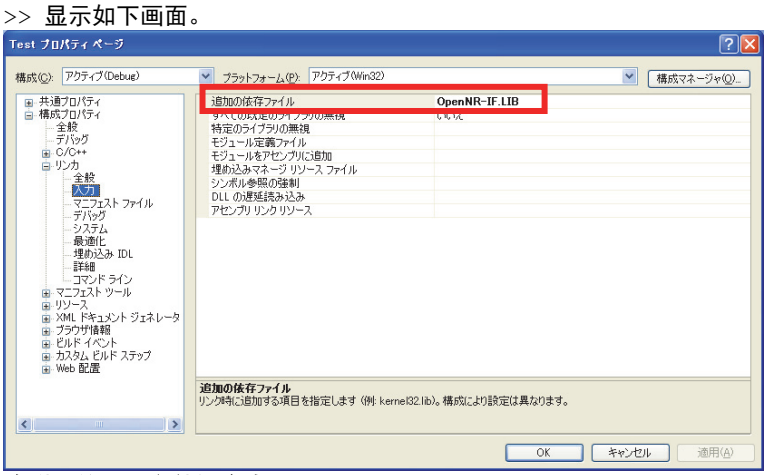
>> 显示如下画面。



将附加的库目录设定为"$(OpennrDir)"。

3 **选择"配置属性:链接:输入"。**

>> 显示如下画面。



将附加的从属文件设定为"OpenNR-IF.lib"。

3． 接口结构体

3.1 连接参数

指定连接目标、连接端口、连接种类等与连接相关的参数，作为 NR_Open 的参数。

NACHI_COMMIF_INFO

```
typedef      struct nachi_CommIfInfo
{
     char    *pcAddrs;
     long    lPortNo;
     long    lRetry;
     long    lSendTimeOut;
     long    lCommSide;
     long    lMode;
     long    lKind;
} NACHI_COMMIF_INFO;
```

成员

pcAddrs
　　连接目标地址
lPortNo
　　连接目标端口编号
lRetry
　　重试次数
lSendTimeOut
　　超时时间[ms]
lCommSide
　　指定连接目标
　　　　控制装置连接（NR_OBJECT_INTERNAL = 0）
　　　　外部连接 TCP（NR_OBJECT_EXTERNAL_TCP = 1）
　　　　外部连接 UDP（NR_OBJECT_EXTERNAL_UDP = 2）
lMode
　　指定连接模式
　　　　控制装置连接（0）
　　　　外部连接
　　　　　　服务器（NR_MODE_COMM_SERVER=0）
　　　　　　客户端（NR_MODE_COMM_CLIENT=1）
lKind
　　连接种类
　　　　实时跟踪通信（NR_DATA_REAL=0）
　　　　机器人操作＆监控通信（NR_DATA_XML=1）
　　　　外部设备连接（0 固定）

指定方法

	指定连接目标	指定连接模式	连接种类
实时 跟踪功能	NR_OBJECT_INTERNAL	0	NR_DATA_REAL
机器人操作＆ 监控功能	NR_OBJECT_INTERNAL	0	NR_DATA_XML
外部设备连接	NR_OBJECT_EXTERNAL_TCP NR_OBJECT_EXTERNAL_UDP	NR_MODE_COMM_SERVER NR_MODE_COMM_CLIENT	0

3.2 实时跟踪数据获取（标头）

是从控制装置获取实时跟踪数据的结构体（NR_GET_REAL_DATA_ALL）的标头。

NR_GET_CTRL_INFO

```
typedef struct NR_GetCtrlInfo{
     USHORT  ushEstopBit     : 1;
     USHORT  ushPlaybkBit    : 1;
     USHORT  ushConnectBit   : 1;
     USHORT  ushErrorBit     : 5;
     USHORT  ushMotorBit     : 1;
     USHORT  ushRsv          : 6;
     USHORT  ushProtcolBit   : 1;
} NR_GET_CTRL_INFO
```

成员

ushEstopBit
: 紧急停止状态(1:紧急停止中)

ushPlaybkBit
: 程序执行状态(0:启动中、1:停止中)

ushConnectBit
: 通信状态

ushErrorBit
: 错误代码（0:正常、4:动作指令值异常）

ushMotorBit
: 运转准备状态（1:运转准备投入）

ushRsv
: 预备

ushProtcolBit
: 协议类型（0:标准、1:扩展）

3.3 实时跟踪数据获取（主体部/标准模式）

是从控制装置获取实时跟踪数据的结构体（NR_GET_REAL_DATA_ALL）的主体部。
标准模式（标头的 ushProtcolBit=0）时，使用本结构体。

NR_GET_REAL_DATA_BODY_STD

```
typedef struct NR_GetRealDataBodyStd{
     float   fCurTcpPos[ NR_MAX_AXIS_STD ];
     float   fCurAngle[ NR_MAX_AXIS_STD ];
     float   fCurrent[ NR_MAX_AXIS_STD ];
     bool    bDigOut[ NR_MAX_DIGITAL_SIG ];
     bool    bDigIn[ NR_MAX_DIGITAL_SIG ];
} NR_GET_REAL_DATA_BODY_STD;
```

成员

fCurTcpPos
　　机器人的手臂前端位置：当前值[mm,deg]
fCurAngle
　　机器人的关节角度：指令值[deg·mm]
fCurrent
　　机器人的电机电流[Apeak]
bDigOut
　　来自控制装置的数字输出
bDigIn
　　发送至控制装置的数字输入

3.4 实时跟踪数据获取（主体部/扩展模式）

是从控制装置获取实时跟踪数据的结构体（NR_GET_REAL_DATA_ALL）的主体部。
扩展模式（标头的 ushProtcolBit=1）时，使用本结构体。

NR_GET_REAL_DATA_BODY_EXT

```
typedef struct NR_GetRealDataBodyExt{
     float   fCurTcpPos[ NR_MAX_AXIS ];
     float   fComTcpPos[ NR_MAX_AXIS ];
     float   fWorkCoord[ NR_MAX_XYZRPY ];
     float   fToolCoord[ NR_MAX_XYZRPY ];
     float   fCurAngle[ NR_MAX_AXIS ];
     float   fComAngle[ NR_MAX_AXIS ];
     float   fTorque[ NR_MAX_AXIS ];
     float   fCurrent[ NR_MAX_AXIS ];
     bool    bDigOut[ NR_MAX_DIGITAL_SIG ];
     bool    bDigIn[ NR_MAX_DIGITAL_SIG ];
     float   fAnaOut[ NR_MAX_ANALOG_CH ];
     float   fAnaIn[ NR_MAX_ANALOG_CH ];
} NR_GET_REAL_DATA_BODY_EXT;
```

成员

fCurTcpPos
　　机器人的手臂前端位置：当前值[mm,deg]
fComTcpPos
　　机器人的手臂前端位置：指令值[mm,deg]
fWorkCoord
　　机器人的手臂前端位置：作业坐标系[mm,deg]
fToolCoord
　　机器人的手臂前端位置：工具坐标[mm,deg]
fCurAngle
　　机器人的关节角度：当前值[deg,mm]
fComAngle
　　机器人的关节角度：指令值[deg,mm]
fTorque
　　机器人的关节扭矩[kgfm]
fCurrent
　　机器人的电机电流[Apeak]
bDigOut
　　来自控制装置的数字输出
bDigIn
　　发送至控制装置的数字输入
fAnaOut
　　来自控制装置的模拟输出[V]
fAnaIn
　　发送至控制装置的模拟输入[V]

3.5 实时跟踪数据获取

获取来自控制装置的实时跟踪数据时使用的数据结构体。
作为 NR_GetAll 的参数。

NR_GET_REAL_DATA_ALL

```
typedef struct NR_GetRealDataAll{
     NR_GET_CTRL_INFO                stCtrl;
     INT                             nTime;
     union {
             NR_GET_REAL_DATA_BODY_EXT       stExt;
             NR_GET_REAL_DATA_BODY_STD       stStd;
     } ustData;
} NR_GET_REAL_DATA_ALL;
```

成员

stCtrl
　　标头
nTime
　　发送时的时间（控制周期）
ustData(ustExt)
　　主体部：扩展(ushProtcolBit=1)
ustData(stStd)
　　主体部：标准(ushProtcolBit=0)

3.6 实时跟踪数据设定（标头）

是发送至控制装置的实时跟踪数据设定结构体（NR_SET_REAL_DATA_ALL）的标头。

NR_SET_CTRL_INFO

```
typedef struct NR_SetCtrlInfo{
     USHORT  ushEstopBit     : 1;
     USHORT  ushFinishBit    : 1;
     USHORT  ushOrderBit     : 1;
     USHORT  ushRsv          : 12;
     USHORT  ushProtcolBit   : 1;
} NR_SET_CTRL_INFO;
```

成员

ushEstopBit
　　紧急停止指令(1)
ushFinishBit
　　外部跟踪结束(1)
ushOrderBit
　　动作指令（0:手臂前端位置指令、1:关节角度指令）
ushRsv
　　预备
ushProtcolBit
　　协议类型（0:标准、1:扩展）

3.7 实时跟踪数据设定（主体部/标准模式）

是发送至控制装置的实时跟踪数据设定结构体（NR_SET_REAL_DATA_ALL）的主体部。
标准模式（标头的 ushProtcolBit=0）时，使用本结构体。

NR_SET_REAL_DATA_BODY_STD

```
typedef struct NR_SetRealDataBodyStd{
     float   fComTcpPos[ NR_MAX_AXIS_STD ];
     float   fWorkCoord[ NR_MAX_XYZRPY ];
     float   fToolCoord[ NR_MAX_XYZRPY ];
     float   fComAngle[ NR_MAX_AXIS_STD ];
     bool    bDigOut[ NR_MAX_DIGITAL_SIG ];
} NR_SET_REAL_DATA_BODY_STD;
```

成员

fComTcpPos
　　发送至机器人的手臂前端位置：指令值[mm,deg]
fWorkCoord
　　发送至机器人的手臂前端位置：作业坐标系[mm,deg]
fToolCoord
　　发送至机器人的手臂前端位置：工具坐标[mm,deg]
fComAngle
　　发送至机器人的关节角度：指令值[deg·mm]
bDigOut
　　发送至机器人的数字输出[V]

3.8 实时跟踪数据设定（主体部/扩展模式）

是发送至控制装置的实时跟踪数据设定结构体（NR_SET_REAL_DATA_ALL）的主体部。
扩展模式（标头的 ushProtcolBit=1）时，使用本结构体。

NR_SET_REAL_DATA_BODY_EXT

```
typedef struct NR_SetRealDataBodyExt{
     float   fComTcpPos[ NR_MAX_AXIS ];
     float   fWorkCoord[ NR_MAX_XYZRPY ];
     float   fToolCoord[ NR_MAX_XYZRPY ];
     float   fComAngle[ NR_MAX_AXIS ];
     bool    bDigOut[ NR_MAX_DIGITAL_SIG ];
     float   fAnaOut[ NR_MAX_ANALOG_CH ];
} NR_SET_REAL_DATA_BODY_EXT;
```

成员

fComTcpPos
　　发送至机器人的手臂前端位置：指令值[mm,deg]
fWorkCoord
　　发送至机器人的手臂前端位置：作业坐标系[mm,deg]
fToolCoord
　　发送至机器人的手臂前端位置：工具坐标[mm,deg]
fComAngle
　　发送至机器人的关节角度：指令值[deg·mm]
bDigOut
　　发送至机器人的数字输出
fAnaOut
　　发送至机器人的模拟输出[V]

3.9 实时跟踪数据设定

向控制装置设定实时数据时所使用的数据结构体。
作为 NR_SetAll 的参数。

NR_SET_REAL_DATA_ALL

```
typedef struct NR_SetRealDataAll{
     NR_SET_CTRL_INFO        stCtrl;
     INT                     nTime;
     union {
             NR_SET_REAL_DATA_BODY_EXT       stExt;
             NR_SET_REAL_DATA_BODY_STD       stStd;
     } ustData;
} NR_SET_REAL_DATA_ALL;
```

成员

stCtrl
　　标头
nTime
　　发送时的时间（控制周期）的回显
ustData(stExt)
　　主体部：扩展(ushProtcolBit=1)
ustData(stStd)
　　主体部：标准(ushProtcolBit=0)

3.10 错误通知数据

设定控制装置中发生异常时的错误信息的结构体。
作为使用 NR_SetNotify 设定的回调方法的参数。

NR_NOTIFICATION

```
typedef struct NR_Notification {
     int             nErrCode;
     int             nUnitNo;
     int             nMechNo;
     int             nAxisNo;
     LPTSTR          csProg;
     int             nStepNo;
} NR_NOTIFICATION;
```

成员

nErrCode
　　错误代码
nUnitNo
　　单元编号
nMechNo
　　机构编号
nAxisNo
　　轴编号
csProg
　　程序名称
nStepNo
　　STEP 编号

3.11 移位量数据

控制装置的保存移位量数据的结构体。

NR_SHIFT

```
typedef struct NR_Shift {
     float           fX;
     float           fY;
     float           fZ;
     float           fRoll;
     float           fPitch;
     float           fYaw;
} NR_SHIFT;
```

成员

fX
　　X 坐标[mm]
fY
　　Y 坐标[mm]
fZ
　　Z 坐标[mm]
fRoll
　　翻滚角度[deg]
fPitch
　　俯仰角度[deg]
fYaw
　　偏航角度[deg]

3.12 姿势（POSE）数据

控制装置的保存姿势量数据的结构体。

NR_POSE

```
typedef struct NR_Pose{
     float           fX;
     float           fY;
     float           fZ;
     float           fRoll;
     float           fPitch;
     float           fYaw;
} NR_POSE;
```

成员

fX
　　X 坐标[mm]
fY
　　Y 坐标[mm]
fZ
　　Z 坐标[mm]
fRoll
　　翻滚角度[deg]
fPitch
　　俯仰角度[deg]
fYaw
　　偏航角度[deg]

3.13 码垛寄存器

控制装置的保存码垛寄存器的结构体。

NR_PalletReg

```
typedef struct NR_PalletReg {
     int             PalletID;
     int             Execute;
     int             Kind;
     int             Layer;
     int             Work;
} NR_PALLETREG;
```

成员

PalletID
　　托板编号
Execute
　　0:停止中、1:执行中
Kind
　　0:码垛、1:卸垛
Layer
　　当前的层数
Work
　　当前的个数/层数

3.14 码垛WORK

控制装置的保存码垛 WORK 的结构体。

NR_PalletWork

```
typedef struct NR_PalletWork {
     float           PalletID;
     float           Length;
     float           Width;
     float           Height;
     float           dL;
     float           dW;
     float           Xa;
     float           Ya;
} NR_PALLETWORK;
```

成员

PalletID
     托板 ID
Length
     工件尺寸 L [mm]
Width
     工件尺寸 W [mm]
Height
     工件尺寸 H [mm]
dL
     工件尺寸 dXp [mm]
dW
     工件尺寸 dYp [mm]
Xa
     靠近距离 Xa [mm]
Ya
     靠近距离 Ya [mm]

3.15 码垛Layer

控制装置的保存码垛 Layer 的结构体。

NR_PalletLayer

```
typedef struct NR_PalletLayer {
     float           LayerNum;
     float           TotalHeight;
     float           LayerType;
     struct {
             float   LayerNo;
             float   Height;
     } Layer[50];
} NR_PALLETLAYER;
```

成员

LayerNum
　　层数
TotalHeight
　　堆积高度[mm]
LayerType
　　堆积形状（0：纵向堆积、1：交互、2：自定义）
LayerNo
　　平面图案编号
Height
　　高度修正 [mm]

3.16 码垛Plene

控制装置的保存码垛 Plene 的结构体。

NR_PalletPlene

```
typedef struct NR_PalletPlene {
     float           Type;
     float           WorkNum[2];
     float           Shift[2];
     struct {
             float   PosX;
             float   PosY;
             float   PosZ;
             float   ThetaZ;
             float   Approach;
     } Work[99];
} NR_PALLETPLENE;
```

成员

Type
　　堆积方式（0：列、1：互锁、2：针轮、3：自定义）
WorkNum
　　工件个数
Shift
　　平行移动（X,Y） [mm]
PosX
　　X [mm]
PosY
　　Y [mm]
PosZ
　　Z [mm]
ThetaZ
　　θz [deg]
Approach
　　靠近方向

4． 接口定义值

4.1 接口返回值

NR_E_NORMAL

接口正常结束。

```
#define      NR_E_NORMAL                    0
```

NR_E_ALREADY

已打开（关闭）时，再次调出打开（关闭）时，进行冗余处理。

```
#define      NR_E_ALREADY                   1001
```

NR_E_NONE_SEND

设定实时跟踪数据时，已经发送完成，但未进行数据设定。

```
#define      NR_E_NONE_SEND                 1002
```

NR_E_EXIT

实时跟踪数据的接收发送已停止。

```
#define      NR_E_EXIT                      1003
```

NR_E_PARAM

接口的调出参数错误。

```
#define      NR_E_PARAM                     -1001
```

NR_E_INVALID

调出接口时，处于某种错误环境中。

```
#define      NR_E_PARAM                     -1002
```

NR_E_NONE_SUPPORT

接口不支持。(连接参数不一致。)

```
#define      NR_E_NONE_SUPPORT              -1003
```

NR_E_EXEC_FAIL

调出接口时，无法执行。

```
#define      NR_E_EXEC_FAIL                  -1004
```

NR_E_EXEC_SEQ

接口的调出顺序有误。

```
#define      NR_E_SEQ                        -1005
```

NR_E_LICENSE

无许可证。

```
#define      NR_E_LICENSE                    -1006
```

NR_E_XML_SCHEMA

XML 架构文件不存在。

```
#define      NR_E_XML_SCHEMA                 -1007
```

NR_E_SEND

发生发送错误。

```
#define      NR_E_SEND                       -2001
```

NR_E_SEND_TIME_OUT

检测到发送超时。

```
#define      NR_E_SEND_TIME_OUT              -2002
```

NR_E_RECV

发生接收错误。

```
#define      NR_E_RECV                       -2003
```

NR_E_RECV_RESPONSE

超过发送接收重试次数。

```
#define      NR_E_RECV_RESPONSE              -2004
```

NR_E_RETRY_OVER

检测到接收超时。

```
#define      NR_E_RETRY_OVER         -2005
```

NR_E_UDATE_FATAL

数据设定失败。

```
#define      NR_E_UDATE_FATAL              -3001
```

NR_E_UDATE_READONLY

无法设定数据。

```
#define      NR_E_UDATE_READONLY           -3002
```

NR_E_UDATE_MISMATCH

操作模式不同，无法设定数据。

```
#define      NR_E_UDATE_MISMATCH           -3003
```

NR_E_UDATE_MISSING

数据设定参数有异。

```
#define      NR_E_UDATE_MISSING            -3004
```

NR_E_UDATE_UNITINVALID

数据设定参数（单元编号）有误。

```
#define      NR_E_UDATE_UNITINVALID        -3005
```

NR_E_CTRL_FATAL

操作请求失败。

```
#define      NR_E_CTRL_FATAL               -4001
```

NR_E_CTRL_FORMAT

操作请求参数有误。

```
#define      NR_E_CTRL_FORMAT              -4002
```

NR_E_CTRL_LIMIT

操作请求参数范围异常。

```
#define      NR_E_CTRL_LIMIT               -4003
```

NR_E_CTRL_EXECFAIL

操作请求无法执行。

```
#define      NR_E_CTRL_EXECFAIL            -4004
```

4.2 接口设定参数

NR_OBJECT_INTERNAL

NR_Open 中，NACHI_COMMIF_INFO 的成员 lCommSide 中设定的参数。
进行与控制装置的数据通信。

```
#define      NR_OBJECT_INTERNAL              0
```

NR_OBJECT_EXTERNAL_TCP

NR_Open 中，NACHI_COMMIF_INFO 的成员 lCommSide 中设定的参数。
进行与外部设备的 TCP 数据通信。

```
#define      NR_OBJECT_EXTERNAL_TCP          1
```

NR_OBJECT_EXTERNAL_UDP

NR_Open 中，NACHI_COMMIF_INFO 的成员 lCommSide 中设定的参数。
进行与外部设备的 UDP 数据通信。

```
#define      NR_OBJECT_EXTERNAL_UDP          2
```

NR_KIND_COMM_SERVER

NR_Open 中，NACHI_COMMIF_INFO 的成员 lMode 中设定的参数。
在与外部设备的数据通信中，作为服务器使用。

```
#define      NR_KIND_COMM_SERVER             0
```

NR_KIND_COMM_CLIENT

NR_Open 中，NACHI_COMMIF_INFO 的成员 lMode 中设定的参数。
在与外部设备的数据通信中，作为客户端使用。

```
#define      NR_KIND_COMM_CLIENT             1
```

NR_DATA_REAL

NR_Open 中，NACHI_COMMIF_INFO 的成员 lKind 中设定的参数。
进行与控制装置的实时跟踪通信。

```
#define      NR_DATA_REAL                    0
```

NR_DATA_XML

NR_Open 中，NACHI_COMMIF_INFO 的成员 lKind 中设定的参数。
进行与控制装置的机器人操作＆监控通信。

```
#define      NR_DATA_XML                     1
```

NR_ACCESS_WAIT

NR_SetAll、NR_GetAll 的参数。
Get 时：等待至接收时。
Set 时：等待至发送时。

```
#define      NR_ACCESS_WAIT                0
```

NR_ACCESS_NO_WAIT

NR_SetAll、NR_GetAll 的参数。
Get：获取最新获取数据。
Set：缓冲并获取。

```
#define      NR_ACCESS_NO_WAIT             1
```

NR_STANBY_OFF_REQ

NR_CtrlRun 的参数。
发行控制装置的运转准备 OFF 请求。

```
#define      NR_STANBY_OFF_REQ             0
```

NR_STANBY_ON_REQ

NR_CtrlRun 的参数。
发行控制装置的运转准备 ON 请求。

```
#define      NR_STANBY_ON_REQ              1
```

NR_STOP_REQ

NR_CtrlRun 的参数。
发行控制装置的停止请求。

```
#define      NR_STOP_REQ                   0
```

NR_RUN_REQ

NR_CtrlRun 的参数。
发行控制装置的启动请求。

```
#define      NR_RUN_REQ                    1
```

4.3 控制装置参数

NR_MAX_AXIS_STD

实时跟踪的标准轴数。

```
#define      NR_MAX_AXIS_STD                8
```

NR_MAX_AXIS

实时跟踪的扩展轴数。

```
#define      NR_MAX_AXIS                    7
```

NR_MAX_XYZRPY

实时跟踪的姿势定义。

```
#define      NR_MAX_XYZRPY                  6
```

NR_MAX_DIGITAL_SIG

实时跟踪的数字输入输出数

```
#define      NR_MAX_DIGITAL_SIG             64
```

NR_MAX_ANALOG_CH

实时跟踪的模拟输入输出数

```
#define      NR_MAX_ANALOG_CH               16
```

4.4 各枚举值

NR_GET_REAL_DATA_EXT_EXT_ID

```
typedef      enum NR_GetRealDataExtId
{
     eNR_GET_STATUS_EXT_ID = 0,
     eNR_GET_TIME_EXT_ID,
     eNR_GET_CUR_TCP_POS_EXT_ID,
     eNR_GET_COM_TCP_POS_EXT_ID,
     eNR_GET_WORK_COORD_EXT_ID,
     eNR_GET_TOOL_COORD_EXT_ID,
     eNR_GET_CUR_ANGLE_EXT_ID,
     eNR_GET_COM_ANGLE_EXT_ID,
     eNR_GET_TORQUE_EXT_ID,
     eNR_GET_MORTOR_AMP_EXT_ID,
     eNR_GET_DIG_OUT_EXT_ID,
     eNR_GET_DIG_IN_EXT_ID,
     eNR_GET_ANA_OUT_EXT_ID,
     eNR_GET_ANA_IN_EXT_ID,
     eNR_GET_REAL_MAX_EXT_ID
} NR_GET_REAL_DATA_EXT_EXT_ID;
```

NR_SET_REAL_DATA_EXT_EXT_ID

```
typedef      enum NR_SetRealDataExtId
{
     eNR_SET_STATUS_EXT_ID = 0,
     eNR_SET_TIME_EXT_ID,
     eNR_SET_COM_TCP_POS_EXT_ID,
     eNR_SET_WORK_COORD_EXT_ID,
     eNR_SET_TOOL_COORD_EXT_ID,
     eNR_SET_COM_ANGLE_EXT_ID,
     eNR_SET_DIG_OUT_EXT_ID,
     eNR_SET_ANA_OUT_EXT_ID,
     eNR_SET_REAL_MAX_EXT_ID
} NR_SET_REAL_DATA_EXT_EXT_ID;
```

NR_GET_REAL_DATA_STD_STD_ID

```
typedef      enum NR_GetRealDataStdId
{
     eNR_GET_STATUS_STD_ID = 0,
     eNR_GET_TIME_STD_ID,
     eNR_GET_CUR_TCP_POS_STD_ID,
     eNR_GET_CUR_ANGLE_STD_ID,
     eNR_GET_MORTOR_AMP_STD_ID,
     eNR_GET_DIG_OUT_STD_ID,
     eNR_GET_DIG_IN_STD_ID,
     eNR_GET_REAL_MAX_STD_ID
} NR_GET_REAL_DATA_STD_STD_ID;
```

NR_SET_REAL_DATA_STD_STD_ID

```
typedef      enum NR_SetRealDataStdId
{
     eNR_SET_STATUS_STD_ID = 0,
     eNR_SET_TIME_STD_ID,
     eNR_SET_COM_TCP_POS_STD_ID,
     eNR_SET_WORK_COORD_STD_ID,
     eNR_SET_TOOL_COORD_STD_ID,
     eNR_SET_COM_ANGLE_STD_ID,
     eNR_SET_DIG_OUT_STD_ID,
     eNR_SET_REAL_MAX_STD_ID
} NR_SET_REAL_DATA_STD_STD_ID;
```

5． 接口（环境/控制）

5.1 连接

根据连接参数 pInfo 的内容，开始连接（通信）。
结束连接（通信）时，请务必进行切断（NR_Close）。

NR_Open

int NR_Open(NACHI_COMMIF_INFO *pInfo)

参数

pInfo[in]
　　连接参数

返回值

成功：通信目标 ID(大于 1 时正常）
失败：负值（※参照接口返回值定义）

5.2 切断

结束连接（通信）。

NR_Close

int NR_Close(int nOpenId)

参数

nOpenId[in]
　　通信目标 ID
　　　　指定为 0 时，切断所有的连接。

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

5.3 实时跟踪数据获取

获取控制装置的实时跟踪数据。

NR_GetAll

```
int NR_GetAll (int nOpenId, NR_GET_REAL_DATA_ALL *pData,
                      long lAccessMode = NR_ACCESS_NO_WAIT)
```

参数

nOpenId[in]
　　通信目标 ID
pData
　　数据存储缓冲区
lAccessMode
　　数据访问模式
　　0（NR_ACCESS_WAIT）　　：等待至接收时
　　1（NR_ACCESS_NO_WAIT）　：获取最新数据

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

5.4 实时跟踪数据设定

设定控制装置的实时跟踪数据。

NR_SetAll

```
int NR_SetAll (int  nOpenId, NR_SET_REAL_DATA_ALL *pData,
                      long lAccessMode = NR_ACCESS_NO_WAIT)
```

参数

nOpenId[in]
　　通信目标 ID
pData [in]
　　数据设定缓冲区
lAccessMode[in]
　　数据访问模式
　　0（NR_ACCESS_WAIT）　　：等待至接收时
　　1（NR_ACCESS_NO_WAIT）　：获取最新数据

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

5.5 错误通知方法的设定

设定控制装置中当发生异常时，进行错误通知时的处理（回调）方法。
通过该设定，当控制装置中发生异常时，可获得错误信息。
错误通知时，当希望将通过接口获取的信息保存至文件时使用。

NR_SetNotify

```
int NR_SetNotify (int  nOpenId, Notify notify )
```

参数

nOpenId[in]
　　通信目标 ID
notify[in]
　　处理（回调）方法指针

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

5.6 外部设备数据发送

向外部设备发送数据。

NR_Send

int NR_Send (int nOpenId , char* sendBuf, int nBufSize)

参数

nOpenId[in]
　　通信目标 ID
sendBuf[in]
　　发送数据
nBufSize[in]
　　发送数据大小

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

5.7 外部设备数据接收

从外部设备接收数据。

NR_Recv

int NR_Recv(int nOpenId , char* recvBuf, int nBufSize, int nTimeOut)

参数

nOpenId[in]
　　通信目标 ID
recvBuf [in]
　　接收数据缓冲区
nBufSize [in]
　　接收数据缓冲区大小
nTimeOut [in]
　　超时时间
　　在指定的时间内，未接收到数据时，发生错误。

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6． 接口（固定输入输出）

6.1 固定输入信号状态的获取请求

批量获取控制装置的固定输入信号状态。

NR_AcsFixedIOInputSignal

```
int NR_AcsFixedIOInputSignal(int nOpenId, BOOL value[], int nSubId, int nCount,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01)
```

参数

nOpenId[in]
　　通信目标 ID
Value[out]
　　固定输入信号状态（48 点）的存储缓冲区（1：TRUE、0：FALSE）
nSubId[in]
　　先头固定输入信号编号（1～48）
nCount[in]
　　信号点数（1～48）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.2 运转准备投入（固定输入信号）状态的获取请求

获取控制装置的运转准备投入（固定输入信号）状态。

NR_AcsFixedIOMotorsOn

```
int NR_AcsFixedIOMotorsOn(int nOpenId, BOOL* value,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　运转准备投入按钮为 ON 时 TRUE，运转准备投入按钮为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.3 G-STOP（固定输入信号）状态的获取请求

获取控制装置的 G-STOP（固定输入信号）状态。

NR_AcsFixedIOGStop1

```
int NR_AcsFixedIOGStop1(int nOpenId, BOOL* value,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　G-STOP 输入时 FALSE、G-STOP 解除时 TRUE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.4 启动1（固定输入信号）状态的获取请求

获取控制装置的启动 1（固定输入信号）状态。

NR_AcsFixedIOStart1

```
int NR_AcsFixedIOStart1(int nOpenId, BOOL* value,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　启动按钮为 ON 时 TRUE、启动按钮为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.5 启动2（固定输入信号）状态的获取请求

获取控制装置的启动 2（固定输入信号）状态。

NR_AcsFixedIOStart2

```
int NR_AcsFixedIOStart2(int nOpenId, BOOL* value,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　启动 2 按钮为 ON 时 TRUE、启动 2 按钮为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.6 启动3（固定输入信号）状态的获取请求

获取控制装置的启动 3（固定输入信号）状态。

NR_AcsFixedIOStart3

```
int NR_AcsFixedIOStart3(int nOpenId, BOOL* value,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　启动 3 按钮为 ON 时 TRUE、启动 3 按钮为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.7 启动4（固定输入信号）状态的获取请求

获取控制装置的启动 4（固定输入信号）状态。

NR_AcsFixedIOStart4

```
int NR_AcsFixedIOStart4(int nOpenId, BOOL* value,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　启动 4 按钮为 ON 时 TRUE、启动 4 按钮为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.8 停止（固定输入信号）状态的获取请求

获取控制装置的停止（固定输入信号）状态。

NR_AcsFixedIOStop

```
int NR_AcsFixedIOStop(int nOpenId, BOOL* value,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　所有单元停止中时 TRUE、某个单元启动中时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.9 再生模式（固定输入信号）状态的获取请求

获取控制装置的再生模式（固定输入信号）状态。

NR_AcsFixedIOPlayBack

```
int NR_AcsFixedIOPlayBack(int nOpenId, BOOL* value,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　再生模式时 TRUE、其他模式时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.10 罩面开关（固定输入信号）状态的获取请求

获取控制装置的罩面开关（固定输入信号）状态。

NR_AcsFixedIOMatSwitch

```
int NR_AcsFixedIOMatSwitch(int nOpenId, BOOL* value,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　罩面开关接通时 TRUE、罩面开关切断时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.11 高速示教（固定输入信号）状态的获取请求

获取控制装置的高速示教（固定输入信号）状态。

NR_AcsFixedIOHighSpeedTeach

```
int NR_AcsFixedIOHighSpeedTeach(int nOpenId, BOOL* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　高速示教模式时 TRUE、其他模式时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.12 P1正常（固定输入信号）状态的获取请求

获取控制装置的 P1 正常（固定输入信号）状态。

NR_AcsFixedIOP1Correct

```
int NR_AcsFixedIOP1Correct (int nOpenId, BOOL* value,
                            int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　P1 正常时 TRUE、P1 异常时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.13 外部紧急停止输入（固定输入信号）状态的获取请求

获取控制装置的外部紧急停止输入（固定输入信号）状态。

NR_AcsFixedIOExeStop

```
int NR_AcsFixedIOExeStop(int nOpenId, BOOL* value,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　外部紧急停止输入时 FALSE、外部紧急停止解除时 TRUE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.14 紧急停止输入（固定输入信号）状态的获取请求

获取控制装置的紧急停止输入（固定输入信号）状态。

NR_AcsFixedIOEStop

```
int NR_AcsFixedIOEStop(int nOpenId, BOOL* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　紧急停止输入时 FALSE、紧急停止解除时 TRUE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.15 安全插头（固定输入信号）状态的获取请求

获取控制装置的安全插头（固定输入信号）状态。

NR_AcsFixedIOSafetyPlug

```
int NR_AcsFixedIOSafetyPlug(int nOpenId, BOOL* value,
                            int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　安全插头为 ON 时 TRUE、安全插头为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.16 运转准备投入确认（固定输入信号）状态的获取请求

获取控制装置的运转准备投入确认（固定输入信号）状态。

NR_AcsFixedIOConfirmMotorsOn

```
int NR_AcsFixedIOConfirmMotorsOn(int nOpenId, BOOL* value,
                                 int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　运转准备投入确认为 ON 时 TRUE、运转准备投入确认为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.17 TP紧急停止（固定输入信号）状态的获取请求

获取控制装置的 TP 紧急停止（固定输入信号）状态。

NR_AcsFixedIOTPEStop

```
int NR_AcsFixedIOTPEStop(int nOpenId, BOOL* value,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　TP 紧急停止输入时 FALSE、TP 紧急停止解除时 TRUE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.18 示教模式（固定输入信号）状态的获取请求

获取控制装置的示教模式（固定输入信号）状态。

NR_AcsFixedIOTeach

```
int NR_AcsFixedIOTeach(int nOpenId, BOOL* value,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　示教模式时 TRUE、其他模式时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.19 TP启用SW（固定输入信号）状态的获取请求

获取控制装置的 TP 启用 SW（固定输入信号）状态。

NR_AcsFixedIOTPEnableSW

```
int NR_AcsFixedIOTPEnableSW(int nOpenId, BOOL* value,
                            int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　TP 启用 SW ON 时 TRUE、TP 启用 SW OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.20 CRON（固定输入信号）状态的获取请求

获取控制装置的 CRON（固定输入信号）状态。

NR_AcsFixedIOCRON

```
int NR_AcsFixedIOCRON(int nOpenId, BOOL* value,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　CRON 为 ON 时 TRUE、非 CRON 为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.21 伺服ON（固定输入信号）状态的获取请求

获取控制装置的伺服 ON（固定输入信号）状态。

NR_AcsFixedIOServoOn

```
int NR_AcsFixedIOServoOn(int nOpenId, BOOL* value,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　伺服 ON 时 TRUE、伺服 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.22 伺服启用（固定输入信号）状态的获取请求

获取控制装置的伺服启用（固定输入信号）状态。

NR_AcsFixedIOServoEnable

```
int NR_AcsFixedIOServoEnable(int nOpenId, BOOL* value,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　伺服启用为 ON 时 TRUE、伺服启用为 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.23 电磁开关ON（固定输入信号）状态的获取请求

获取控制装置的电磁开关 ON（固定输入信号）状态。

NR_AcsFixedIOMagentON

```
int NR_AcsFixedIOMagentON(int nOpenId, BOOL* value,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　电磁开关 ON 时 TRUE、电磁开关 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.24 熔敷检测（固定输入信号）状态的获取请求

获取控制装置的熔敷检测（固定输入信号）状态。

NR_AcsFixedIOWeldDetection

```
int NR_AcsFixedIOWeldDetection(int nOpenId, BOOL* value,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　检测到熔敷时 TRUE、未检测到熔敷时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.25 不一致检测（固定输入信号）状态的获取请求

获取控制装置的不一致检测（固定输入信号）状态。

NR_AcsFixedIOInConsistency

```
int NR_AcsFixedIOInConsistency(int nOpenId, BOOL* value,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　检测到不一致时 TRUE、未检测到不一致时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.26 固定输出信号状态的获取请求

批量获取控制装置的固定输出信号状态。

NR_AcsFixedIOOutputSignal

```
int NR_AcsFixedIOOutputSignal(int nOpenId, , BOOL value[], int nSubId, int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
Value[out]
　　固定输出信号状态（24 点）的存储缓冲区（1：TRUE、0：FALSE）
nSubId[in]
　　先头固定输出信号编号（1～24）
nCount[in]
　　信号点数（1～24）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.27 运转准备投入（固定输出信号）状态的获取请求

获取控制装置的运转准备投入（固定输入信号）状态。

NR_AcsFixedIOMotorsOnLAMP

```
int NR_AcsFixedIOMotorsOnLAMP(int nOpenId, BOOL* value,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　运转准备投入时 TRUE、运转准备切断时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.28 运转准备ON请求（固定输出信号）状态的获取请求

获取控制装置的运转准备 ON 请求（固定输入信号）状态。

NR_AcsFixedIOMotorsOnRequest

```
int NR_AcsFixedIOMotorsOnRequest(int nOpenId, BOOL* value,
                                 int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　运转准备 ON 请求投入时 TRUE、运转准备 ON 请求切断时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.29 启动指示灯1（固定输出信号）状态的获取请求

获取控制装置的启动指示灯 1（固定输入信号）状态。

NR_AcsFixedIOStartDisplay1

```
int NR_AcsFixedIOStartDisplay1(int nOpenId, BOOL* value,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　单元 1 正在启动中时 TRUE、停止中时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.30 启动指示灯2（固定输出信号）状态的获取请求

获取控制装置的启动指示灯 2（固定输入信号）状态。

NR_AcsFixedIOStartDisplay2

```
int NR_AcsFixedIOStartDisplay2(int nOpenId, BOOL* value,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　单元 2 正在启动中时 TRUE、停止中时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.31 启动指示灯3（固定输出信号）状态的获取请求

获取控制装置的启动指示灯 3（固定输入信号）状态。

NR_AcsFixedIOStartDisplay3

```
int NR_AcsFixedIOStartDisplay3(int nOpenId, BOOL* value,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　单元 3 正在启动中时 TRUE、停止中时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.32 启动指示灯4（固定输出信号）状态的获取请求

获取控制装置的启动指示灯 4（固定输入信号）状态。

NR_AcsFixedIOStartDisplay4

```
int NR_AcsFixedIOStartDisplay4(int nOpenId, BOOL* value,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　单元 4 正在启动中时 TRUE、停止中时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.33 暂停指示灯（固定输出信号）状态的获取请求

获取控制装置的暂停指示灯（固定输入信号）状态。

NR_AcsFixedIOStopDisplay

```
int NR_AcsFixedIOStopDisplay(int nOpenId, BOOL* value,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　暂停指示灯 ON 时 TRUE、暂停指示灯 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.34 TP启用释放（固定输出信号）状态的获取请求

获取控制装置的 TP 启用释放（固定输入信号）状态。

NR_AcsFixedIOTPEnableRelease

```
int NR_AcsFixedIOTPEnableRelease(int nOpenId, BOOL* value,
                                 int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　TP 启用 SW OFF 时 TRUE、TP 启用 SW ON 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.35 运转准备ON许可（固定输出信号）状态的获取请求

获取控制装置的运转准备 ON 许可（固定输入信号）状态。

NR_AcsFixedIOMotorsOnEnable

```
int NR_AcsFixedIOMotorsOnEnable(int nOpenId, BOOL* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　运转准备 ON 许可时 TRUE、运转准备 ON 禁止时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.36 电磁开关ON许可（固定输出信号）状态的获取请求

获取控制装置的电磁开关 ON 许可（固定输入信号）状态。

NR_AcsFixedIOMagnetOnEnable

```
int NR_AcsFixedIOMagnetOnEnable(int nOpenId, BOOL* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　电磁开关 ON 许可时 TRUE、电磁开关 ON 禁止时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.37 内部/外部选择（固定输出信号）状态的获取请求

获取控制装置的内部/外部选择（固定输入信号）状态。

NR_AcsFixedIOInternalExternal

```
int NR_AcsFixedIOInternalExternal(int nOpenId, BOOL* value,
                                  int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　外部模式时 TRUE、内部模式时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.38 WPS紧急停止控制（固定输出信号）状态的获取请求

获取控制装置的 WPS 紧急停止控制（固定输入信号）状态。

NR_AcsFixedIOWPSEStop

```
int NR_AcsFixedIOWPSEStop(int nOpenId, BOOL* value,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　WPS 紧急停止控制 OFF 时 TRUE、WPS 紧急停止控制 ON 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.39 CPU异常输出（固定输出信号）状态的获取请求

获取控制装置的 CPU 异常输出（固定输入信号）状态。

NR_AcsFixedIOCPUFailure

```
int NR_AcsFixedIOCPUFailure(int nOpenId, BOOL* value,
                            int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　CPU 异常时 TRUE、CPU 无异常时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.40 TP模式选择（固定输出信号）状态的获取请求

获取控制装置的 TP 模式选择（固定输入信号）状态。

NR_AcsFixedIOTPMode

```
int NR_AcsFixedIOTPMode(int nOpenId, BOOL* value,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　TP 选择器 ON 时 TRUE、TP 选择器 OFF 时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

6.41 外部准备ON请求（固定输出信号）状态的获取请求

获取控制装置的 TP 模式选择（固定输入信号）状态。

NR_AcsFixedIOEXTMotorsOn

```
int NR_AcsFixedIOEXTMotorsOn(int nOpenId, BOOL* value,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　外部准备 ON 请求投入时 TRUE、外部准备 ON 请求切断时 FALSE 的存储场所
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

7． 接口（通用输入输出）

7.1 通用输入信号的获取请求

获取控制装置的通用输入信号。

NR_AcsGeneralInputSignal

```
int NR_AcsGeneralInputSignal(int nOpenId, BOOL value[], int nSubId, int nCount,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　通用输入信号状态存储缓冲区（1：TRUE、0：FALSE）
nSubId[in]
　　先头输入信号编号（1～2048）
nCount[in]
　　信号点数（1～2048）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

7.2 通用输入信号（BYTE值）的获取请求

获取控制装置的通用输入信号（BYTE）。
按 8bit 进行整数化（nSubId 的位置数据为 LSB、之后第 8bit 为 MSB）。

NR_AcsGeneralInputSignalB

```
int NR_AcsGeneralInputSignalB(int nOpenId, int[] value, int nSubId, int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　将通用输入信号按每 8bit 进行整数化的值的存储缓冲区
nSubId[in]
　　先头输入信号编号（1～2048）
nCount[in]
　　信号（BYTE）数（1～256）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

7.3 通用输出信号的获取/设定请求

获取或设定控制装置的通用输出信号。

NR_AcsGeneralOutputSignal

```
int NR_AcsGeneralOutputSignal(int nOpenId, BOOL value[], BOOL bUpdate, int nSubId,
int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　通用输出信号状态存储缓冲区（1：TRUE、0：FALSE）
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　先头输出信号编号（1～2048）
nCount[in]
　　信号点数（1～2048）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

7.4 通用输出信号（BYTE值）的获取/设定请求

获取或设定控制装置的通用输出信号（BYTE）。
按 8bit 进行整数化（nSubId 的数据为 LSB、第 8bit 为 MSB）。

NR_AcsGeneralOutputSignalB

```
int NR_AcsGeneralOutputSignalB(int nOpenId, int value[], BOOL bUpdate, int nSubId,
int nCount,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　将通用输出信号按每 8bit 进行整数化的值的存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　先头输出信号编号（1～2048）
nCount[in]
　　信号（BYTE）数（1～256）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

8． 接口（信号名称）

8.1 输入信号名称的获取

获取控制装置的输入信号名称。
指定超过从 value 中获取的输入信号字符串长度的数组。

NR_AcsStrInputSignalName

```
int NR_AcsStrInputSignalName(int nOpenId, LPWSTR value[],
                             size_t nBufSize, int nSubId, int nCount,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输入信号名称存储缓冲区（字符串的数组）
nBufSize[in]
　　各输入信号名称的大小
nSubId [in]
　　先头输入信号编号（1～2048）
nCount [in]
　　信号点数（1～2048）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

8.2 输出信号名称的获取

获取控制装置的对象单元的输出信号名称。
指定超过从 value 中获取的输入信号字符串长度的数组。

NR_AcsStrOutputSignalName

```
int NR_AcsStrOutputSignalName(int nOpenId, LPWSTR value[],
                              size_t nBufSize, int nSubId, int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输出信号名称存储缓冲区（字符串的数组）
nBufSize[in]
　　各输出信号名称的大小
nSubId [in]
　　先头输出信号编号（1～2048）
nCount [in]
　　信号点数（1～2048）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9. 接口（变量）

9.1 全局整数变量值的获取/设定请求

获取或设定控制装置的全局整数变量值。

NR_AcsGlobalInt

```
int NR_AcsGlobalInt(int nOpenId, int value[], BOOL bUpdate, int nSubId, int nCount,
                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　全局整数变量存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　变量编号（1～200）
nCount [in]
　　变量数（1～200）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9.2 全局实数变量值的获取/设定请求

获取或设定控制装置的全局实数变量值。

NR_AcsGlobalFloat

```
int NR_AcsGlobalFloat(int nOpenId, int value[], BOOL bUpdate, int nSubId, int nCount,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　全局实数变量存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　变量编号（1～200）
nCount [in]
　　变量数（1～200）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9.3 全局字符串变量值的获取/设定请求

获取或设定控制装置的全局字符串变量值。

NR_AcsGlobalString

```
int NR_AcsGlobalString(int nOpenId, LPWSTR value[],size_t szBufSize,
                       BOOL bUpdate, int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　全局字符串变量存储缓冲区（字符串的数组）
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
szBufSize[in]
　　各全局字符串变量的字符串大小
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　变量编号（1～50）
nCount [in]
　　变量数（1～50）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9.4 局部整数变量值的获取/设定请求

获取或设定控制装置的指定单元的局部整数变量值。

NR_AcsLocalInt

```
int NR_AcsLocalInt(int nOpenId, int value[],
                                BOOL bUpdate, int nSubId, int nCount, int nUnitId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　局部整数变量存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　变量编号（1～200）
nCount [in]
　　变量数（1～200）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9.5 局部实数变量值的获取/设定请求

获取或设定控制装置的指定单元的局部实数变量值。

NR_AcsLocalFloat

```
int NR_AcsLocalFloat(int nOpenId, float value[],
                                BOOL bUpdate, int nSubId, int nCount, int nUnitId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　局部实数变量存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　变量编号（1～200）
nCount [in]
　　变量数（1～200）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9.6 局部字符串变量值的获取/设定请求

获取或设定控制装置的指定单元的局部字符串变量值。

NR_AcsLocalString

```
int NR_AcsLocalString(int nOpenId, LPWSTR value[],size_t szBufSize,
                              BOOL bUpdate, int nSubId, int nCount, int nUnitId = 1,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　局部字符串变量存储缓冲区（字符串的数组）
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
szBufSize[in]
　　各局部字符串变量的字符串大小
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　变量编号（1～50）
nCount [in]
　　变量数（1～50）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

9.7 移位寄存器值的获取/设定请求

通过结构体（NR_Shift）获取或设定控制装置的移位寄存器值。

NR_AcsShift

```
int NR_AcsShift(int nOpenId, NR_SHIFT value[],
                                BOOL bUpdate, int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　移位寄存器存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　先头寄存器编号（1～9）
nCount [in]
　　寄存器数（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10. 接口（系统信息）

10.1 系统版本的获取请求

获取控制装置的系统版本。

NR_AcsVersion

int NR_AcsVersion(int nOpenId, LPWSTR value,size_t szBufSize)

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　系统版本存储缓冲区
szBufSize[in]
　　系统版本的字符串大小

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.2 单元名称的获取请求

获取控制装置的指定单元的单元名称。

NR_AcsUnitName

int NR_AcsUnitName(int nOpenId, LPWSTR value[],size_t szBufSize, int nSubId, int nCount)

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　单元名称存储缓冲区（字符串的数组）
szBufSize[in]
　　各单元名称的字符串大小
nSubId[in]
　　先头单元 ID（1～9）
nCount [in]
　　单元数（1～9）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.3 单元轴数的获取请求

获取控制装置的指定单元的单元轴数。

NR_AcsUnitAxisCnt

int NR_AcsUnitAxisCnt(int nOpenId, int value[], int nSubId, int nCount)

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　单元轴数存储缓冲区
nSubId[in]
　　先头单元 ID（1～9）
nCount [in]
　　单元数（1～9）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.4 当前单元编号的获取/设定请求

获取或设定控制装置的当前单元编号。

NR_AcsCurrentUnitNo

```
int NR_AcsCurrentUnitNo(int nOpenId, int* value, BOOL bUpdate,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　当前单元编号的存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.5 当前机构编号的获取/设定请求

获取或设定控制装置的指定单元的当前机构编号。

NR_AcsCurrentMechaNo

```
int NR_AcsCurrentMechaNo(int nOpenId, int* value, BOOL bUpdate,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　当前机构编号的存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.6 远程操作许可状态的获取请求

获取控制装置的远程操作（连接 FDonDesk 视图模式）许可状态。

NR_AcsRemoteMode

```
int NR_AcsRemoteMode(int nOpenId, BOOL* value,
                                  int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　远程模式状态存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.7 程序编号（包含堆栈）的获取请求

获取控制装置的指定单元的程序编号（包含堆栈）。

NR_AcsPrgNo

```
int NR_AcsPrgNo(int nOpenId, int value[], int nCount, int nUnitId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　程序编号的存储缓冲区
nCount [in]
　　堆栈数（1～9）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.8 STEP编号（包含堆栈）的获取请求

获取控制装置的指定单元的 STEP 编号（包含堆栈）。

NR_AcsStepNo

```
int NR_AcsStepNo(int  nOpenId, int value[], int nCount, int nUnitId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　STEP 编号的存储缓冲区
nCount [in]
　　堆栈数（1～9）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.9 用户等级的获取请求

获取控制装置的用户等级。

NR_AcsUserLevel

```
int NR_AcsUserLevel(int nOpenId, int* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　用户等级存储缓冲区（
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.10 错误编号的获取请求

获取控制装置的指定单元的错误编号。

NR_AcsErrInfo

```
int NR_AcsErrInfo(int nOpenId, LPTSTR value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　错误编号存储缓冲区
nSubId[in]
　　先头单元 ID（1～9）
nCount [in]
　　单元数（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.11 CPU负荷的获取请求

获取控制装置的 CPU 负荷。

NR_AcsCPULoad

```
Int NR_AcsCPULoad(int nOpenId, float* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　CPU 负荷存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.12 3.3V电压值的获取请求

获取控制装置的 3.3V 电压值。

NR_AcsCPUVoltage3

```
Int NR_AcsCPUVoltage3(int nOpenId, float* value,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　3.3V 电压值存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.13 5V电压值的获取请求

获取控制装置的 5V 电压值。

NR_AcsCPUVoltage5

```
Int NR_AcsCPUVoltage5(int nOpenId, float* value,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　5V 电压值存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.14 CPU温度传感器1的获取请求

获取控制装置的 CPU 温度传感器 1。

NR_AcsCPUTempe1

```
Int NR_AcsCPUTempe1(int nOpenId, float* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　CPU 温度传感器 1 存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.15 CPU温度传感器2的获取请求

获取控制装置的 CPU 温度传感器 2。

NR_AcsCPUTempe2

```
Int NR_AcsCPUTempe2(int nOpenId, float* value,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　CPU 温度传感器 2 存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

10.16 伺服通信异常计数器获取请求

获取控制装置的伺服通信异常计数器。

NR_AcsServoCommErr

```
int NR_AcsServoCommErr(int nOpenId, int* value[],
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　伺服通信异常计数器存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11. 接口（监视器信息）

11.1 各轴编码器值的获取请求

获取控制装置的指定机构的各轴编码器值[pulse]。

NR_AcsAxisEncode

```
int NR_AcsAxisEncode(int nOpenId, int value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴编码器值存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.2 各轴电流值的获取请求

获取控制装置的指定机构的各轴电流值[Apeak]。

NR_AcsAxisAmpValue

```
int NR_AcsAxisAmpValue(int nOpenId, float value[], int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴电流值存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.3 各轴电流指令值的获取请求

获取控制装置的指定机构的各轴电流指令值[Apeak]。

NR_AcsAxisOrderAmpValue

```
int NR_AcsAxisOrderAmpValue(int nOpenId, float value[], int nSubId, int nCount,
                            int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴电流指令值存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.4 各轴速度的获取请求

获取控制装置的指定机构的各轴速度[rpm]。

NR_AcsAxisSpeed

```
int NR_AcsAxisSpeed(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴速度存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.5 各轴速度指令的获取请求

获取控制装置的指定机构的各轴速度指令。

NR_AcsAxisOrderSpeed

```
int NR_AcsAxisOrderSpeed(int nOpenId, float value[], int nSubId, int nCount,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴速度指令存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.6 各轴角度的获取请求

获取控制装置的指定机构的各轴角度[deg·mm]。

NR_AcsAxisTheta

```
int NR_AcsAxisTheta (int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴角度存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.7 各轴角度指令的获取请求

获取控制装置的指定机构的各轴角度指令[deg·mm]。

NR_AcsAxisThetaOrder

```
int NR_AcsAxisThetaOrder(int nOpenId, float value[], int nSubId, int nCount,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴角度指令存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.8 各轴扭矩的获取请求

获取控制装置的指定机构的各轴扭矩[kgfm]。

NR_AcsAxisTorque

```
int NR_AcsAxisTorque(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　各轴扭矩存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.9 工具前端位置的获取请求

获取控制装置的指定机构的工具前端位置[mm,deg]。

NR_AcsToolTipPos

```
int NR_AcsToolTipPos(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　工具前端位置存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.10 工具前端位置指令的获取请求

获取控制装置的指定机构的工具前端位置指令[mm,deg]。

NR_AcsOrderToolTipPos

```
int NR_AcsOrderToolTipPos(int nOpenId, float value[], int nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　工具前端位置指令存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.11 工具前端速度的获取请求

获取控制装置的指定机构的工具前端速度[rpm]。

NR_AcsTcpSpeed

```
int NR_AcsToolTipPos(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　工具前端速度存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.12 工具前端速度指令的获取请求

获取控制装置的指定机构的工具前端速度指令[rpm]。

NR_AcsOrderTcpSpeed

```
int NR_AcsOrderToolTipPos(int nOpenId, float value[], nt nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　工具前端速度指令存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.13 工具前端速度比例模拟输出的获取请求

获取控制装置的指定机构的工具前端速度比例模拟输出[V]。

NR_AcsProportionTcpSpeedAnalog

```
int NR_AcsProportionTcpSpeedAnalog(int nOpenId, float value[], int nSubId, int nCount,
                                   int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　工具前端速度比例模拟输出存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.14 工具前端速度比例数字输出的获取请求

获取控制装置的指定机构的工具前端速度比例数字输出。

NR_AcsProportionTcpSpeedDigital

```
int NR_AcsProportionTcpSpeedDigital(int nOpenId, float value[], int nSubId, int nCount,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　工具前端速度比例数字输出存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.15 各轴不平衡扭矩的获取请求

获取控制装置的指定机构的各轴不平衡扭矩[kgfm]。

NR_AcsTorqueImbalance

```
int NR_AcsTorqueImbalance(int nOpenId, float value[], int nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　各轴不平衡扭矩存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.16 各轴最大扭矩的获取请求

获取控制装置的指定机构的各轴最大扭矩[kgfm]。

NR_AcsTorqueAxisMax

```
int NR_AcsTorqueAxisMax(int nOpenId, int value[], int nSubId, int nCount,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各轴最大扭矩存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.17 当前的在线移位量的获取请求

获取控制装置的指定单元的当前在线移位量。

NR_AcsShiftOnlineCurrent

```
int NR_AcsShiftOnlineCurrent (int nOpenId, NR_SHIFT value[], int nUnitId = 1,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　当前在线移位量存储缓冲区
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.18 当前的机器人移位量的获取/设定请求

获取或设定控制装置的指定单元的当前机器人移位量。

NR_AcsShiftRobotCurrent

```
int NR_AcsShiftRobotCurrent(int nOpenId, NR_SHIFT value[], BOOL bUpdate, int nUnitId
= 1,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　当前机器人移位量存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.19 当前的整体移位量的获取请求

获取控制装置的指定单元的当前整体移位量。

NR_AcsShiftAllCurrent

```
int NR_AcsShiftAllCurrent (int nOpenId, NR_SHIFT value[], int nUnit = 1,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　当前的整体移位量存储缓冲区
bPush[in]
　　数据传输模式（Push：TRUE、Pull：FALSE）
　　指定为 Push 时，获取一次当前的状态后，在各数据确认周期中，每当变化达到数据阈值时获取。
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.20 各机构伺服ON/OFF状态的获取请求

获取控制装置的指定机构的各机构伺服 ON/OFF 状态。

NR_AcsServomotorOnOff

```
int NR_AcsServomotorOnOff(int nOpenId, int value[], int nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　各机构伺服 ON/OFF 状态存储缓冲区
nSubId[in]
　　先头机构编号（1～9）
nCount [in]
　　机构数（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.21 耗电量的获取请求

获取控制装置的耗电量[kWh]。

NR_AcsElePowerCost

```
int NR_AcsElePowerCost (int nOpenId, float value[],
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　耗电量存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.22 伺服状态获取请求

获取控制装置的指定机构的伺服状态。

NR_AcsServomotorStatus

```
int NR_AcsServomotorStatus(int nOpenId, int value[], int nSubId, int nCount,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　伺服状态存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.23 伺服错误状态获取请求

获取控制装置的指定机构的伺服错误状态。

NR_AcsServomotorErrStatus

```
int NR_AcsServomotorErrStatus(int nOpenId, int value[], int nSubId, int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　伺服错误状态存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.24 编码器传输异常计数器获取请求

获取控制装置的指定机构的编码器传输异常计数器。

NR_AcsEncErrCntTrans

```
int NR_AcsEncErrCntTrans(int nOpenId, int value[], int nSubId, int nCount,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　编码器传输异常计数器存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.25 编码器跳位计数器获取请求

获取控制装置的指定机构的编码器跳位计数器。

NR_AcsEncErrCntBitJmp

```
int NR_AcsEncErrCntBitJmp(int nOpenId, int value[], int nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　编码器跳位计数器存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.26 编码器错误计数器获取请求

获取控制装置的指定机构的编码器错误计数器。

NR_AcsEncErrCntStatus

```
int NR_AcsEncErrCntStatus(int nOpenId, int value[], int nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　编码器错误计数器存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.27 编码器任意ID获取请求

获取控制装置的指定机构的编码器任意 ID。

NR_AcsEncErrCntStatus

```
int NR_AcsEncErrCntStatus(int nOpenId, int value[], int nSubId, int nCount,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　编码器任意 ID 存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.28 剩余寿命的获取请求

获取控制装置的指定机构的剩余寿命[Hr]。

NR_AcsOverHaul

```
int NR_AcsOverHaul(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　剩余寿命存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.29 寿命的获取请求

获取控制装置的指定机构的寿命[Hr]。

NR_AcsLifeSpan

```
int NR_AcsLifeSpan(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　寿命存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.30 机器人温度预测的获取请求

获取控制装置的指定机构的温度预测。

NR_AcsRobotTempe

```
int NR_AcsRobotTempe(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　温度预测存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.31 气压检查基准扭矩的获取请求

获取控制装置的指定机构的气压检查基准扭矩[kgfm]。

NR_AcsGasPrsBase

```
int NR_AcsGasPrsBase(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　气压检查基准扭矩存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

11.32 气压检查测量扭矩的获取请求

获取控制装置的指定机构的气压检查测量扭矩[kgfm]。

NR_AcsGasPrsMeasure

```
int NR_ AcsGasPrsMeasure(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　气压检查测量扭矩存储缓冲区
nSubId[in]
　　机构编号（1～9）
nCount [in]
　　轴数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

12. 接口（模拟信息）

12.1 模拟输入值的获取请求

获取控制装置的模拟输入值[V]。

NR_AcsInputAnalog

```
int NR_AcsInputAnalog(int nOpenId, float value[], int nSubId, int nCount,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　模拟输入值存储缓冲区
nSubId[in]
　　先头模拟输入编号（1～8）
nCount [in]
　　模拟输入数（1～8）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

12.2 模拟输出值的获取请求

获取控制装置的模拟输出值[V]。

NR_AcsOutputAnalog

```
int NR_AcsOutputAnalog(int nOpenId, float value[], int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　模拟输入值存储缓冲区
nSubId[in]
　　先头模拟输出编号（1～4）
nCount [in]
　　模拟输出数（1～4）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

13. 接口（数字信息）

13.1 数字输入值的获取请求

获取控制装置的数字输入值。

NR_AcsInputDigital

```
int NR_AcsInputDigital(int nOpenId, float value[], int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　数字输入值存储缓冲区
nSubId[in]
　　先头数字输入编号（1～9）
nCount [in]
　　数字输入数（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

13.2 数字输出值的获取请求

获取控制装置的数字输出值。

NR_AcsOutputDigital

```
int NR_AcsOutputDigital(int nOpenId, float value[], int nSubId, int nCount,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　数字输出值存储缓冲区
nSubId[in]
　　先头数字输出编号（1～9）
nCount [in]
　　数字输出数（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14． 接口（接缝信息）

14.1 接缝焊枪编号的获取请求

获取控制装置的指定电焊机的接缝焊枪编号。

NR_AcsSeamWeldNo

```
int NR_AcsSeamWeldNo(int nOpenId, int value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　接缝焊枪编号存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.2 接缝移动侧机构编号的获取请求

获取控制装置的指定电焊机的接缝移动侧机构编号。

NR_AcsSeamMechaNoMoveSide

```
int NR_AcsSeamMechaNoMoveSide(int nOpenId, int value[], int nSubId, int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　接缝移动侧机构编号存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.3 接缝固定侧机构编号的获取请求

获取控制装置的指定电焊机的接缝固定侧机构编号。

NR_AcsSeamMechaNoStationarySide

```
int NR_AcsSeamMechaNoStationarySide(int nOpenId, int value[], int nSubId, int nCount,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　接缝固定侧机构编号存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.4 接缝移动侧旋转数的获取请求

获取控制装置的指定电焊机的接缝移动侧旋转数[rpm]。

NR_AcsSeamRotationMoveSide

```
int NR_AcsSeamRotationMoveSide(int nOpenId, float value[], int nSubId, int nCount,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　接缝移动侧旋转数存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.5 接缝固定侧旋转数的获取请求

获取控制装置的指定电焊机的接缝固定侧旋转数[rpm]。

NR_AcsSeamRotationStationarySide

```
int NR_AcsSeamRotationStationarySide(int nOpenId, float value[], int nSubId, int
nCount,
                                     int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　接缝固定侧旋转数存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.6 接缝焊接距离的获取请求

获取控制装置的指定电焊机的接缝焊接距离[mm]。

NR_AcsSeamWeldDistance

```
int NR_AcsSeamWeldDistance(int nOpenId, float value[], int nSubId, int nCount,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　接缝焊接距离存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.7 接缝焊接时间的获取请求

获取控制装置的指定电焊机的接缝焊接时间[sec]。

NR_AcsSeamWeldTime

```
int NR_AcsSeamWeldTime(int nOpenId, float value[], int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　接缝焊接时间存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.8 接缝通电距离的获取请求

获取控制装置的指定电焊机的接缝通电距离[mm]。

NR_AcsSeamEnergizationDistance

```
int NR_AcsSeamEnergizationDistance(int nOpenId, float value[], int nSubId, int nCount,
                                   int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　接缝通电距离存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

14.9 接缝通电时间的获取请求

获取控制装置的指定电焊机的接缝通电时间[sec]。

NR_AcsSeamEnergizationTime

```
int NR_AcsSeamEnergizationTime(int nOpenId, float value[], int nSubId, int nCount,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　接缝通电时间存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

15. 接口（点焊信息）

15.1 伺服焊枪指令加压力的获取请求

获取控制装置的指定电焊机的伺服焊枪指令加压力[KN]。

NR_AcsPressureOrderServoGun

```
int NR_AcsPressureOrderServoGun(int nOpenId, float value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　伺服焊枪指令加压力存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

15.2 伺服焊枪实际加压力的获取请求

获取控制装置的指定电焊机的伺服焊枪实际加压力[KN]。

NR_AcsPressureRealServoGun

```
int NR_AcsPressureRealServoGun(int nOpenId, float value[], int nSubId, int nCount,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　伺服焊枪实际加压力存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

15.3 伺服焊枪磨损量的获取请求

获取控制装置的指定电焊机的伺服焊枪磨损量[mm]。

NR_AcsFrictionServoGun

```
int NR_AcsFrictionServoGun(int nOpenId, float value[], int nSubId, int nCount,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　伺服焊枪磨损量存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

15.4 伺服焊枪移动磨损量的获取请求

获取控制装置的指定电焊机的伺服焊枪移动磨损量[mm]。

NR_AcsFrictionServoGunMoveSide

```
int NR_AcsFrictionServoGunMoveSide(int nOpenId, float value[], int nSubId, int nCount,
                                   int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　伺服焊枪移动磨损量存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

15.5 伺服焊枪固定磨损量的获取请求

获取控制装置的指定电焊机的伺服焊枪固定磨损量[mm]。

NR_AcsFrictionServoGunStationarySide

```
int NR_AcsFrictionServoGunStationarySide(int nOpenId, float value[], int nSubId, int nCount,
                                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　伺服焊枪固定磨损量存储缓冲区
nSubId[in]
　　电焊机编号（1～6）
nCount [in]
　　电焊机数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16. 接口（PLC 信息）

16.1 PLC物理输入信号状态的获取请求

获取控制装置的 PLC 物理输入信号状态。

NR_AcsInputSignalPhysicsPlc

```
int NR_AcsInputSignalPhysicsPlc(int nOpenId, BOOL value[], int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　PLC 物理输入信号存储缓冲区
nSubId[in]
　　物理输入信号编号（1～192）
nCount [in]
　　物理输入信号数（1～192）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.2 PLC物理输出信号状态的获取请求

获取控制装置的 PLC 物理输出信号状态。

NR_AcsOutputSignalPhysicsPlc

```
int NR_AcsOutputSignalPhysicsPlc(int nOpenId, BOOL value[], int nSubId, int nCount,
                                 int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　PLC 物理输出信号存储缓冲区
nSubId[in]
　　物理输出信号编号（1～192）
nCount [in]
　　物理输出信号数（1～192）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.3 PLC BOOL变量值的获取/设定请求

获取或设定控制装置的 PLC BOOL 变量值。

NR_AcsBoolValuePlc

```
int NR_AcsBoolValuePlc(int nOpenId, BOOL value[], BOOL bUpdate, int nSubId, int
nCount,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　BOOL 变量值存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　BOOL 变量编号（1～2000）
nCount [in]
　　BOOL 变量数（1～2000）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.4 PLC实数变量值的获取/设定请求

获取或设定控制装置的 PLC 实数变量值。

NR_AcsRealValuePlc

```
int NR_AcsRealValuePlc(int nOpenId, float value[], BOOL bUpdate, int nSubId, int
nCount,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　实数变量值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　实数变量编号（1～100）
nCount [in]
　　实数变量数（1～100）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.5 PLC整数变量值的获取/设定请求

获取或设定控制装置的 PLC 整数变量值。

NR_AcsDintValuePlc

```
int NR_AcsDintValuePlc(int nOpenId, int value[], BOOL bUpdate, int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　整数变量值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　整数变量编号（1～500）
nCount [in]
　　整数变量数（1～500）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.6 PLC短整数变量值的获取/设定请求

获取或设定控制装置的 PLC 短整数变量值。

NR_AcsSintValuePlc

```
int NR_AcsSintValuePlc(int nOpenId, int value[], BOOL bUpdate, int nSubId, int nCount,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　短整数变量值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　短整数变量编号（1～100）
nCount [in]
　　短整数变量数（1～100）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.7 PLC计时器变量值的获取/设定请求

获取或设定控制装置的 PLC 计时器变量值。

NR_AcsTimerValuePlc

```
int NR_AcsTimerValuePlc(int nOpenId, int value[], BOOL bUpdate, int nSubId, int
nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　计时器变量值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　计时器变量编号（1～500）
nCount [in]
　　计时器变量数（1～500）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

16.8 PLC 字符串变量值的获取/设定请求

获取或设定控制装置的 PLC 字符串变量值。

NR_AcsStringValuePlc

```
int NR_AcsStringValuePlc(int nOpenId, LPWSTR value[], size_t nBufSize,
                         BOOL bUpdate, int nSubId, int nCount,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　字符串变量值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　字符串变量编号（1～10）
nCount [in]
　　字符串变量数（1～10）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

17． 接口（码垛信息）

17.1 码垛名称的获取/设定请求

获取或设定控制装置的码垛名称。

NR_AcsPalletizingStringValue

```
int NR_AcsPalletizingStringValue(int nOpenId, LPWSTR value[],size_t szBufSize,
                                 BOOL bUpdate, int nSubId, int nCount,
                                 int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　码垛名称存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
szBufSize[in]
　　存储缓冲区大小
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　托板编号（1～100）
nCount [in]
　　托板数（1～100）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

17.2 码垛计数器值的获取/设定请求

获取或设定控制装置的码垛计数器值。

NR_AcsPalletizingCounter

```
int NR_AcsPalletizingCounter(int nOpenId, int* value, BOOL bUpdate, int nSubId,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　码垛计数器值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　码垛样式编号（1～255）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

17.3 码垛寄存器值的获取/设定请求

通过结构体（NR_PALLETREG）获取或设定控制装置的码垛寄存器值。

NR_AcsPalletizingReg

```
int NR_AcsPalletizingReg(int nOpenId, NR_PALLETREG value[], size_t szBufSize,
                         BOOL bUpdate, int nSubId,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　码垛寄存器值存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
szBufSize[in]
　　存储缓冲区大小
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　码垛寄存器编号（1～32）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

17.4 码垛WORK的获取/设定请求

通过结构体（NR_PALLETREG）获取控制装置的码垛 WORK。

NR_AcsPalletizingWork

```
int NR_AcsPalletizingWork(int nOpenId, NR_PALLETWORK* value,
                          BOOL bUpdate, int nSubId,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　码垛 WORK 存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　码垛样式编号（1～255）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

17.5 码垛Layer的获取/设定请求

通过结构体（NR_PALLETLAYER）获取控制装置的码垛 Layer。

NR_AcsPalletizingLayer

```
int NR_AcsPalletizingLayer(int nOpenId, NR_PALLETLAYER* value,
                           BOOL bUpdate, int nSubId,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　码垛 Layer 存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　码垛样式编号（1～255）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

17.6 码垛Plene的获取请求

通过结构体（NR_PALLETPLENE）获取控制装置的码垛 Plene。

NR_AcsPalletizingPlene

```
int NR_AcsPalletizingPlene(int nOpenId, NR_PALLETPLENE* value,
                           BOOL bUpdate, int nSubId,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　码垛 Plene 存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nSubId[in]
　　码垛样式编号（1～255）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18． 接口（示教信息）

18.1 记录速度的获取/设定请求

获取或设定控制装置的指定单元的记录速度。

NR_AcsRecordSpeed

```
int NR_AcsRecordSpeed(int nOpenId, int value[], BOOL bUpdate, int nUnitId = 1,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　记录速度存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18.2 记录工具编号的获取/设定请求

获取或设定控制装置的指定单元的记录工具编号。

NR_AcsRecordToolNo

```
int NR_AcsRecordToolNo(int nOpenId, int value[], BOOL bUpdate, int nUnitId = 1,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　记录工具编号存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18.3 记录精度编号的获取/设定请求

获取或设定控制装置的指定单元的记录精度编号。

NR_AcsRecordAccuracyNo

```
int NR_AcsRecordAccuracyNo(int nOpenId, int value[], BOOL bUpdate, int nUnitId = 1,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　记录精度编号存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18.4 记录平滑度编号的获取/设定请求

获取或设定控制装置的指定单元的记录平滑度编号。

NR_AcsRecordSmoothNo

```
int NR_AcsRecordSmoothNo(int nOpenId, int value[], BOOL bUpdate, int nUnitId = 1,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　记录平滑度编号存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18.5 记录加速度编号的获取/设定请求

获取或设定控制装置的指定单元的记录加速度编号。

NR_AcsAccelerationNo

```
int NR_AcsAccelerationNo(int nOpenId, int value[], BOOL bUpdate, int nUnitId = 1,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　记录加速度编号存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18.6 记录插值类型的获取/设定请求

获取或设定控制装置的指定单元的记录插值类型。

NR_AcsInterpolationKind

```
int NR_AcsInterpolationKind(int nOpenId, int value[], BOOL bUpdate, int nUnitId =
1,
                            int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　记录插值类型存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

18.7 姿势变量值的获取/设定请求

获取或设定控制装置的指定单元的姿势变量值。

NR_AcsPoseValue

```
int NR_AcsPoseValue(int nOpenId, NR_Pose value[],
                                BOOL bUpdate, int nId, int nSubId, int nUnitId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　姿势变量值存储缓冲区
　　　　bUpdate=TRUE 时　　：输入域
　　　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nId[in]
　　姿势文件编号
nSubId[in]
　　姿势编号
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

19. 接口（手动操作信息）

19.1 手动坐标系登录的获取请求

获取控制装置的手动坐标系登录。

NR_AcsEntryManualCoordinateType

```
int NR_AcsEntryManualCoordinateType(int  nOpenId, LPWSTR value[], size_t szBufSize,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　手动坐标系登录存储缓冲区
szBufSize[in]
　　存储缓冲区大小
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

19.2 手动坐标系的获取/设定请求

获取或设定控制装置的当前机构的手动坐标系。

NR_AcsManualCoordinateType

```
int NR_AcsManualCoordinateType(int nOpenId, int* value, BOOL bUpdate,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　手动坐标系存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

19.3 手动速度的获取/设定请求

获取或设定控制装置的手动速度。

NR_AcsManualSpeed

```
int NR_AcsManualSpeed(int nOpenId, int* value, BOOL bUpdate,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　手动速度存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

20． 接口（检查操作信息）

20.1 检查速度的获取/设定请求

获取或设定控制装置的检查速度。

NR_AcsCheckSpeed

```
int NR_AcsCheckSpeed(int nOpenId, int* value, BOOL bUpdate,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　检查速度存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

20.2 检查模式的获取/设定请求

获取或设定控制装置的检查模式（单独/连续）。

NR_AcsCheckMode

```
int NR_AcsCheckMode(int nOpenId, BOOL* value, BOOL bUpdate,
                                    int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　检查模式存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

20.3 STEP进展率的获取请求

获取控制装置的指定单元的 STEP 进展率[1/100]。

NR_AcsCheckOperationProgress

```
int NR_AcsCheckOperationProgress(int nOpenId, float* value, int nUnitId = 1,
                                 int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　STEP 进展状态存储缓冲区
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

21． 接口（再生信息）

21.1 运转模式的获取/设定请求

获取或设定控制装置的运转模式。

NR_AcsOperationModePlayback

```
int NR_AcsOperationModePlayback(int nOpenId, int* value, BOOL bUpdate,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　运转模式存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

21.2 输入输出信号等待状态的获取/设定请求

获取或设定控制装置的指定单元的输入输出信号等待状态。

NR_AcsWaittingStatusSignal

```
int NR_AcsWaittingStatusSignal(int nOpenId, BOOL* value, BOOL bUpdate, int nUnitId
= 1,
                               int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[in/out]
　　输入输出信号等待状态存储缓冲区
　　bUpdate=TRUE 时　　：输入域
　　bUpdate=FALSE 时　　：输出域
bUpdate[in]
　　设定时使用 TRUE、获取时使用 FALSE
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

21.3 程序调用层数的获取请求

获取控制装置的指定单元的程序调用层数（嵌套数）。

NR_AcsStackDepthPrg

```
int NR_AcsStackDepthPrg(int nOpenId, int* value, int nUnitId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　程序调用状态存储缓冲区
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

21.4 速度倍率的获取请求

获取控制装置的指定单元的速度倍率。

NR_AcsOverRideSpeed

```
int NR_AcsOverRideSpeed(int nOpenId, float* value, int nUnitIId = 1,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　速度倍率存储缓冲区
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

21.5 低速再生状态的获取请求

获取控制装置的低速再生状态。

NR_AcsStatusSlowPlayback

```
int NR_AcsStatusSlowPlayback(int nOpenId, int* value,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　低速再生状态存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

21.6 节能状态的获取请求

获取控制装置的节能状态。

NR_AcsStatusSavingEnergy

```
int NR_AcsStatusSavingEnergy(int nOpenId, int* value,
                             int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　节能状态存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

22． 接口（力传感器信息）

22.1 力传感器的获取请求

获取控制装置的力传感器输入[N,Nm]。

NR_AcsForceCtrl

```
int NR_AcsForceCtrl(int  nOpenId, float* value, int nSubId, int nCount,
                                int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　力传感器输入存储缓冲区
nSubId[in]
　　力传感器输入编号（1～6）
nCount [in]
　　力传感器输入数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

22.2 力控制移位量的获取请求

获取控制装置的力控制移位量[mm,rad]。

NR_AcsForceShift

```
int NR_AcsForceShift(int  nOpenId, float* value, int nSubId, int nCount,
                              int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　力传感器输入存储缓冲区
nSubId[in]
　　力控制移位量（1～6）
nCount [in]
　　力控制移位量数（1～6）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

23． 接口（输送带信息）

23.1 输送带寄存器值的获取请求

获取控制装置的输送带寄存器值。

NR_AcsConveyerRegister

```
int NR_AcsConveyerRegister(int nOpenId, float* value, int nSubId, int nCount,
                           int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输送带寄存器值存储缓冲区
nSubId[in]
　　输送带编号（1～2）
nCount [in]
　　输送带数（1～2）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

23.2 输送带寄存器值的获取请求

获取控制装置的输送带模式。

NR_AcsConveyerMode

```
int NR_AcsConveyerMode(int nOpenId, float* value,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输送带模式存储缓冲区
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

23.3 输送带速度的获取请求

获取控制装置的输送带速度[mm/sec]。

NR_AcsConveyerSpeed

```
int NR_AcsConveyerSpeed(int nOpenId, float* value, int nSubId, int nCount,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输送带速度存储缓冲区
nSubId[in]
　　输送带编号（1～2）
nCount [in]
　　输送带数（1～2）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

23.4 输送带脉冲的获取请求

获取控制装置的输送带脉冲。

NR_AcsConveyerPulse

```
int NR_AcsConveyerPulse(int nOpenId, float* value, int nSubId, int nCount,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输送带脉冲存储缓冲区
nSubId[in]
　　输送带编号（1～2）
nCount [in]
　　输送带数（1～2）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

24． 接口（操作）

24.1 运转准备ON/OFF请求

控制控制装置的运转准备 ON/OFF。
启动选择必须设定为外部模式。

NR_CtrlMotor

int NR_CtrlMotor(int nOpenId, long lCtrlSW)

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　ON(1)/OFF(0)

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

24.2 启动/停止请求

控制控制装置的程序启动/停止。
启动选择必须设定为外部模式。

NR_CtrlRun

int NR_CtrlRun(int nOpenId, long lCtrlSW, long lUnitId = 1)

参数

nOpenId[in]
　　通信目标 ID
value[in]
　　启动(1)/停止(0)
lUnitId[in]
　　单元 ID（1～9）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

24.3 程序选择请求

选择控制装置的当前单元的程序。

NR_CtrlProgram

int NR_CtrlProgram(int nOpenId, int nProgNo)

参数

nOpenId[in]
　　通信目标 ID
nProgNo[in]
　　程序编号

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

24.4 STEP选择请求

选择控制装置的当前单元的 STEP。

NR_CtrlStep

int NR_CtrlStep(int nOpenId, int nStepNo)

参数

nOpenId[in]
　　通信目标 ID
nStepNo[in]
　　STEP 编号

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

24.5 JOG操作请求

进行控制装置的当前机构的 JOG 操作。
仅在用户应用模式下动作。

NR_CtrlJog

```
int NR_CtrlJog(int nOpenId, NR_POSE* pose, int nUnitId = 1);
```

参数

nOpenId[in]
　　通信目标 ID
pose[in]
　　JOG 操作动作速度（1-250[mm/sec]）
nUnitId[in]
　　单元 ID（1～9）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

备注

JOG 操作是机器人在手动坐标系中所选择的坐标系中进行动作。
此外，机器人的动作速度在手动速度中进行限制。

■相关接口
19.2 手动坐标系的获取/设定请求
19.3 手动速度的获取/设定请求

24.6 绝对位置（工具前端位置指定）动作请求

控制装置的当前单元动作至指定的工具前端位置。
仅在用户应用模式下动作。

NR_CtrlMoveX

```
int NR_CtrlMoveX(int nOpenId, NR_POSE* pose,
                 int nType = 0, int nUnitId = 1, int nConf = 0,
                 float fExtPos[] = NULL, int nExtPosSize = 0)
```

参数

nOpenId[in]
　　通信目标 ID
pose[in]
　　工具前端位置
nType[in]
　　0:通过、1:定位、2:END
nUnitId[in]
　　单元 ID（1～9）
nConf[in]
　　形态指定
fExtPos[in]
　　追加轴角度/距离
nExtPosSize[in]
　　追加轴大小

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

备注

机器人的动作（插值及速度、工具、精度）使用记录状态。
通过事先在接口（示教信息）中进行设定，可变更机器人的动作。

■记录状态：

■相关接口
18.1 记录速度的获取/设定请求
18.2 记录工具编号的获取/设定请求
18.3 记录精度编号的获取/设定请求
18.4 记录平滑度编号的获取/设定请求
18.5 记录加速度编号的获取/设定请求
18.6 记录插值类型的获取/设定请求

24.7 绝对位置（各轴角度指定）动作请求

控制装置的当前单元动作至指定的各轴角度。
仅在用户应用模式下动作。

NR_CtrlMoveJ

```
int NR_CtrlMoveJ(int nOpenId, float fAngle[], int nAngleSize, int nType = 0, int nUnitId = 1)
```

参数

nOpenId[in]
　　通信目标 ID
fAngle[in]
　　各轴角度
nAngleSize[in]
　　各轴角度大小
nType[in]
　　0:通过、1:定位、2:END
nUnitId[in]
　　单元 ID（1～9）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

备注

机器人的动作（插值及速度、工具、精度）使用记录状态。
通过事先在接口（示教信息）中进行设定，可变更机器人的动作。

■记录状态：

■相关接口
18.1 记录速度的获取/设定请求
18.2 记录工具编号的获取/设定请求
18.3 记录精度编号的获取/设定请求
18.4 记录平滑度编号的获取/设定请求
18.5 记录加速度编号的获取/设定请求
18.6 记录插值类型的获取/设定请求

24.8 相对位置（机器人坐标移位）动作请求

控制装置的当前单元从当前位置开始，在机器人坐标系中进行移位动作。
仅在用户应用模式下动作。

NR_CtrlMoveXR

```
int NR_CtrlMoveXR(int nOpenId, NR_POSE* pose,
                  int nType = 0, int nUnitId = 1, int nConf = 0,
                  float fExtPos[] = NULL, int nExtPosSize = 0)
```

参数

nOpenId[in]
　　通信目标 ID
pose[in]
　　机器人坐标系中的移位量
nType[in]
　　0:通过、1:定位、2:END
nUnitId[in]
　　单元 ID（1～9）
nConf[in]
　　形态指定
fExtPos[in]
　　追加轴角度
nExtPosSize[in]
　　追加轴大小

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

备注

机器人的动作（插值及速度、工具、精度）使用记录状态。
通过事先在接口（示教信息）中进行设定，可变更机器人的动作。

■记录状态：

■相关接口

18.1 记录速度的获取/设定请求
18.2 记录工具编号的获取/设定请求
18.3 记录精度编号的获取/设定请求
18.4 记录平滑度编号的获取/设定请求
18.5 记录加速度编号的获取/设定请求
18.6 记录插值类型的获取/设定请求

24.9 相对位置（工具坐标移位）动作请求

控制装置的当前单元从当前位置开始，在工具坐标系中进行移位动作。
仅在用户应用模式下动作。

NR_CtrlMoveXT

```
int NR_CtrlMoveXT(int nOpenId, NR_POSE* pose,
                  int nType = 0, int nUnitId = 1, int nConf = 0,
                  float fExtPos[] = NULL, int nExtPosSize = 0)
```

参数

nOpenId[in]
　　通信目标 ID
pose[in]
　　工具坐标系中的移位量
nConf[in]
　　形态指定
nType[in]
　　0:通过、1:定位、2:END
nUnitId[in]
　　单元 ID（1～9）
fExtPos[in]
　　追加轴角度
nExtPosSize[in]
　　追加轴大小

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

备注

机器人的动作（插值及速度、工具、精度）使用记录状态。
通过事先在接口（示教信息）中进行设定，可变更机器人的动作。

■记录状态：

■相关接口

18.1 记录速度的获取/设定请求
18.2 记录工具编号的获取/设定请求
18.3 记录精度编号的获取/设定请求
18.4 记录平滑度编号的获取/设定请求
18.5 记录加速度编号的获取/设定请求
18.6 记录插值类型的获取/设定请求

24.10 相对位置（各轴角度移位）动作请求

控制装置的当前单元从当前位置开始，以各轴角度进行移位动作。
仅在用户应用模式下动作。

NR_CtrlMoveJA

```
int NR_CtrlMoveJA(int nOpenId, float* fAngle, int nAngleSize, int nType = 0, int nUnitId = 1)
```

参数

nOpenId[in]
　　通信目标 ID
fAngle[in]
　　工具坐标系中的移位量
fAngle[in]
　　各轴角度
nAngleSize[in]
　　各轴角度大小
nType[in]
　　0:通过、1:定位、2:END
nUnitId[in]
　　单元 ID（1～9）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

备注

机器人的动作（插值及速度、工具、精度）使用记录状态。
通过事先在接口（示教信息）中进行设定，可变更机器人的动作。

■记录状态：

■相关接口
18.1 记录速度的获取/设定请求
18.2 记录工具编号的获取/设定请求
18.3 记录精度编号的获取/设定请求
18.4 记录平滑度编号的获取/设定请求
18.5 记录加速度编号的获取/设定请求
18.6 记录插值类型的获取/设定请求

25． 接口（程序诊断）

25.1 运转开始时间的获取

获取控制装置的指定程序的运转开始时间。

NR_AcsProgStartTime

```
int NR_AcsProgStartTime(int nOpenId, LPWSTR value, size_t szBufSize,
                        int nSubId, int nUnitId = 1,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　运转开始时间存储缓冲区
szBufSize[in]
　　运转开始时间的字符串大小
nSubId[in]
　　程序编号(0～9999)
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.2 循环时间的获取

获取控制装置的指定程序的循环时间。

NR_AcsProgStartTime

```
int NR_AcsProgStartTime(int nOpenId, LPWSTR value, size_t szBufSize,
                        int nSubId, int nUnitId = 1,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　循环时间存储缓冲区
szBufSize[in]
　　循环时间的字符串大小
nSubId[in]
　　程序编号(0～9999)
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.3 循环数的获取

获取控制装置的指定程序的循环数。

NR_AcsProgCycleCnt

```
int NR_AcsProgCycleCnt(int nOpenId, int* value,
                       int nSubId, int nUnitId = 1,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　循环数存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.4 输入信号等待时间的获取

获取控制装置的指定程序的输入信号等待时间。

NR_AcsProgIWaitTime

```
int NR_AcsProgIWaitTime(int nOpenId, LPWSTR value, size_t szBufSize,
                        int nSubId, int nUnitId = 1,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　输入信号等待时间存储缓冲区
szBufSize[in]
　　输入信号等待时间的字符串大小
nSubId[in]
　　程序编号(0～9999)
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.5 计时器信号等待时间的获取

获取控制装置的指定程序的计时器等待时间。

NR_AcsProgDelayTime

```
int NR_AcsProgDelayTime(int nOpenId, LPWSTR value, size_t szBufSize,
                        int nSubId, int nUnitId = 1,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　计时器等待时间存储缓冲区
szBufSize[in]
　　计时器等待时间的字符串大小
nSubId[in]
　　程序编号(0～9999)
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.6 点焊时间的获取

获取控制装置的指定程序的点焊时间。

NR_AcsProgWeldTime

```
int NR_AcsProgWeldTime(int nOpenId, LPWSTR value, size_t szBufSize,
                       int nSubId, int nUnitId = 1,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　点焊时间存储缓冲区
szBufSize[in]
　　点焊时间的字符串大小
nSubId[in]
　　程序编号(0～9999)
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.7 点焊数的获取

获取控制装置的指定程序的点焊数。

NR_AcsProgWeldCnt

```
int NR_AcsProgWeldCnt(int nOpenId, int value[],
                      int nSubId, int nCount, int nUnitId = 1,
                      int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　点焊数存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　电焊机数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.8 平均速度的获取

获取控制装置的指定程序的平均速度[rpm]。

NR_AcsProgAveSpeed

```
int NR_AcsProgAveSpeed(int nOpenId, float value[],
                       int nSubId, int nCount, int nUnitId = 1,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　平均速度存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.9 平均扭矩的获取

获取控制装置的指定程序的平均扭矩[Kgfm]。

NR_AcsProgAveTorque

```
int NR_AcsProgAveTorque(int nOpenId, float value[],
                        int nSubId, int nCount, int nUnitId = 1,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　平均扭矩存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.10 平均电流的获取

获取控制装置的指定程序的平均电流[Apeak]。

NR_AcsProgAveCurrent

```
int NR_AcsProgAveCurrent (int nOpenId, float value[],
                          int nSubId, int nCount, int nUnitId = 1,
                          int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　平均电流存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.11 最大速度的获取

获取控制装置的指定程序的最大速度[rpm]。

NR_AcsProgMaxSpeed

```
int NR_AcsProgMaxSpeed(int nOpenId, float value[],
                       int nSubId, int nCount, int nUnitId = 1,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　最大速度存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.12 最大扭矩的获取

获取控制装置的指定程序的最大扭矩[Kgfm]。

NR_AcsProgMaxTorque

```
int NR_AcsProgMaxTorque(int nOpenId, float value[],
                        int nSubId, int nCount, int nUnitId = 1,
                        int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　最大扭矩存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.13 最大电流的获取

获取控制装置的指定程序的最大电流[Apeak]。

NR_AcsProgMaxCurrent

```
int NR_AcsProgMaxCurrent(int nOpenId, float value[],
                         int nSubId, int nCount, int nUnitId = 1,
                         int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　最大电流存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

25.14 寿命的获取

获取控制装置的指定程序的寿命[Hr]。

NR_AcsProgLifeSpan

```
int NR_AcsProgLifeSpan(int nOpenId, float value[],
                       int nSubId, int nCount, int nUnitId = 1,
                       int nPriority = NR_PULL_MODE, float fThreshold = 0.01f)
```

参数

nOpenId[in]
　　通信目标 ID
value[out]
　　寿命存储缓冲区
nSubId[in]
　　程序编号(0～9999)
nSubId[in]
　　轴数（1～6）
nUnitId[in]
　　单元 ID（1～9）
nPriority[in]
　　数据传输模式（Pull 模式：-1、Push 模式：0～6）
　　Pull 模式　　：获取当前的状态
　　Push 模式　　：获取当前的状态后，在各指定周期内，每当变化达到数据阈值时获取
　　（0：5ms、1：10ms、2：50ms、3：100ms、4：200ms、5：500ms、6：1s）
fThreshold[in]
　　数据传输模式为 Push 时的数据阈值（变化量）

返回值

成功：大于 0（NR_E_NORMAL）
失败：负值（※参照接口返回值定义）

26. 故障排除

OpenNR-IF 的故障现象及处理方法如下所述。

表 26.1 故障排除

No.	故障现象	措施
1	连接控制装置时发生 **NR_E_LICENSE**。	使用许可证文件的情形： （1） 请将许可证文件“license.dat”复制至 OpenNR-IF.DLL 所在的同一文件夹下，或生成的执行文件所在的同一文件夹下。 （2） “license.dat”文件与所使用的 PC 不一致。 许可证文件只能用于申请了 MAC 地址的 PC。 请确认申请的 MAC 地址是否正确。
		使用 USB 硬件加密锁的情形： （1） 请确认 USB 硬件加密锁是否已经连接。 （2） 仅可用于申请了 MAC 地址的 PC。 请确认申请的 MAC 地址是否正确。
2	连接控制装置时发生 **NR_E_NONE_SEND**。	（1） 请确认是否已经与控制装置连接。 （2） 请确认控制装置的 IP 地址。
3	执行时发生错误。 “由于本应用程序的配置不正确，应用程序启动失败。”	请安装 Microsoft Visual C++ 2010 的运行库。

NACHI
株式会社 不二越

http://www.nachi-fujikoshi.co.jp/

总公司　东京都港区东新桥1-9-2　汐留住友大厦17层　邮编 105-0021
Tel +81-3-5568-5245 Fax +81-3-5568-5236

中国

那智不二越（上海）贸易有限公司
上海市普陀区丹巴路98弄7号　龙裕财富中心11层　邮编 200062
Tel 021-6915-2200　Fax 021-6915-5427

重庆分公司
重庆市江北区红鼎国际名苑C座17-18, 17-19　邮编 400020
Tel 023-8816-1967　Fax 023-8816-1968

沈阳分公司
辽宁省沈阳市沈河区悦宾街1号方圆大厦304室　邮编 110000
Tel 024-3120-2252　Fax 024-2250-5316

北京分公司
北京市朝阳区朝外大街乙12号 昆泰国际大厦 0-1110室　邮编 100020
Tel 010-5879-0181　Fax 010-5879-0182

长春事务所
长春市绿园区普阳街1688号长融大厦B座707室　邮编 130061
Tel 0431-8507-8700　Fax 0431-8507-8701

广州事务所
广州市番禺区东环路431号港信城B座505室　邮编 510120
Tel 020-2293-9503　Fax 020-2293-9503

那智不二越（江苏）精密机械有限公司
江苏省张家港市经济技术开发区(南区)南园路39号　邮编 215618
Tel 0512-3500-7616　Fax 0512-3500-7615

上海不二越精密轴承有限公司
上海市嘉定区马陆镇丰茂路258号易通工业园　邮编 201801
Tel 021-6915-6200　Fax 021-6915-6202

耐锯（上海）精密刀具有限公司
上海市嘉定区马陆镇丰茂路258号易通工业园　邮编 201801
Tel 021-6915-5899　Fax 021-6915-5898

东莞建越精密轴承有限公司
东莞市洪梅镇凼涌村
Tel 0769-8843-1300　Fax 0769-8843-1330

机器人事业部
富山市不二越本町1-1-1, JAPAN 邮编930-8511
Tel +81-76-423-5137
Fax +81-76-493-5252