

NEXCOM

EtherCAT Master

CiA402 servo control API

用户手册

Manual Rev.: V1.4

Revision Date: February 24th, 2016

Revise note:

Ver	Description
V0.1	2013/12/16: First beta version release
V1.0	2014/8/19: Enhance API descriptions. Add torque control APIs
V1.1	2015/5/15: Typo fix
V1.2	2015/7/13: Add description of moving direction to Jog and JogA
V1.3	2015/8/14: Add Ch2.4.11 NEC_CoE402SetTargetVelocity() API Add Ch2.4.12 NEC_CoE402GetActualVelocity() API
V1.4	2016/2/24: Add Ch2.7.4 NEC_CoE402ClearHomeStartBit() API Add Ch2.7.5 NEC_CoE402CheckHomeStatus() API

目录

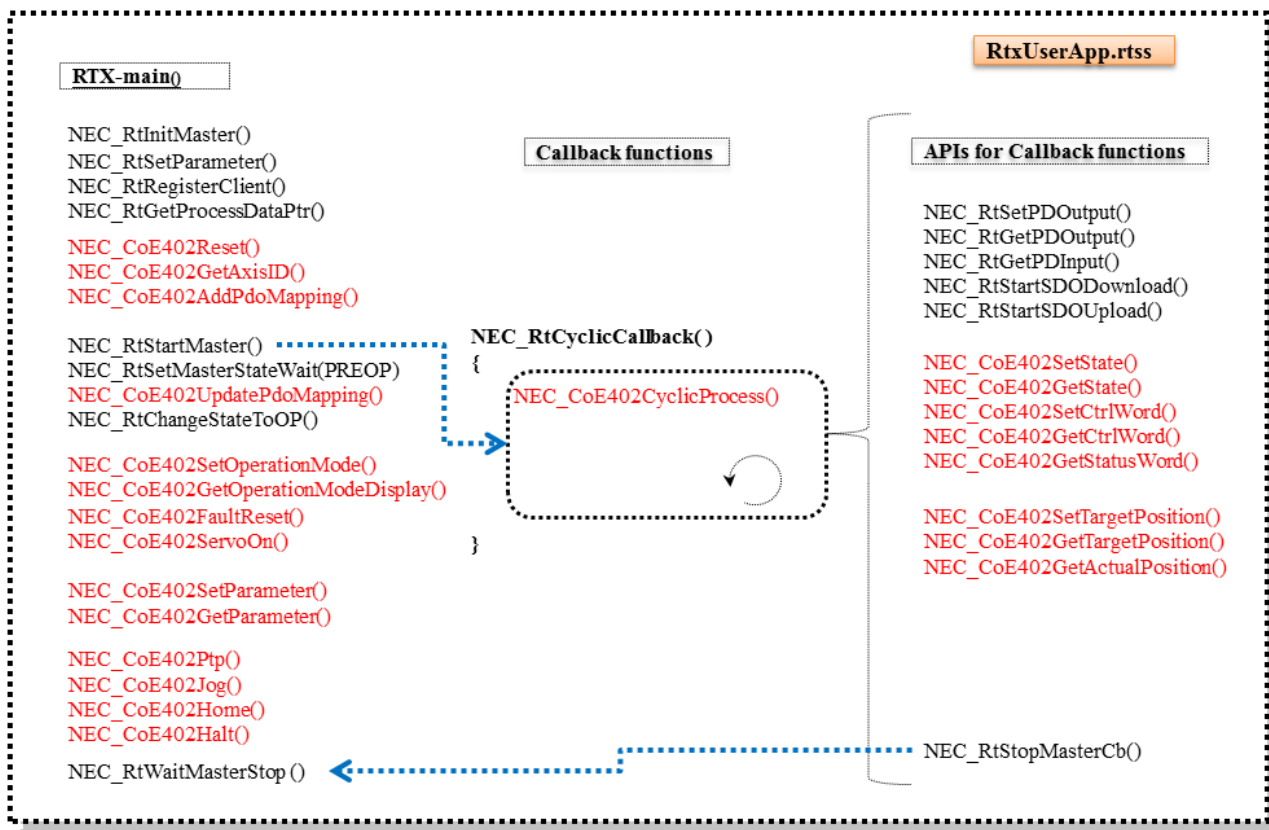
NEXCOM	1
Revise note:	2
目录	i
1. CoE CiA402 操作说明	1
1.1. 流程图	1
1.2. 基本规格	1
1.3. 注意事项	1
2. CoE CiA402 函式库	2
2.1. API 总览	2
2.2. 初始化相关函式	5
2.2.1. NEC_CoE402Reset	5
2.2.2. NEC_CoE402Close	6
2.2.3. NEC_CoE402GetAxisId	7
2.2.4. NEC_CoE402GetDefaultMapInfo	8
2.2.5. NEC_CoE402GetAxisIdEx	10
2.2.6. NEC_CoE402ResetPdoMapping	12
2.2.7. NEC_CoE402AddPdoMapping	13
2.2.8. NEC_CoE402UpdatePdoMapping	14
2.3. CiA402 状态控制函式	15
2.3.1. NEC_CoE402CyclicProcess	15
2.3.2. NEC_CoE402SetCtrlWord	17
2.3.3. NEC_CoE402GetCtrlWord	17
2.3.4. NEC_CoE402GetStatusWord	19
2.3.5. NEC_CoE402ChangeState	20
2.3.6. NEC_CoE402SetState	21

2.3.7.	NEC_CoE402GetState.....	22
2.3.8.	NEC_CoE402FaultReset.....	23
2.3.9.	NEC_CoE402ServoOn	24
2.4.	CiA402 驱动器基本操作函数	25
2.4.1.	NEC_CoE402SetOperationMode.....	25
2.4.2.	NEC_CoE402GetOperationModeDisplay	26
2.4.3.	NEC_CoE402SetParameter.....	27
2.4.4.	NEC_CoE402GetParameter	28
2.4.5.	NEC_CoE402GetActualPosition.....	29
2.4.6.	NEC_CoE402SetTargetPosition	30
2.4.7.	NEC_CoE402GetTargetPosition.....	30
2.4.8.	NEC_CoE402SetQuickStopDec	31
2.4.9.	NEC_CoE402SetSoftPosLimit.....	32
2.4.10.	NEC_CoE402SetMaxVelLimit.....	33
2.4.11.	NEC_CoE402SetTargetVelocity	34
2.4.12.	NEC_CoE402GetActualVelocity.....	35
2.5.	点对点运动 (Profile Position mode)	36
2.5.1.	NEC_CoE402Ptp	36
2.5.2.	NEC_CoE402PtpV	36
2.5.3.	NEC_CoE402PtpA	36
2.5.4.	NEC_CoE402WaitTargetReached	38
2.6.	速度运动(Profile Velocity mode)	40
2.6.1.	NEC_CoE402Jog.....	40
2.6.2.	NEC_CoE402JogA	40
2.6.3.	NEC_CoE402Halt	42
2.7.	归零运动 (Homing mode)	43
2.7.1.	NEC_CoE402Home	43

- 2.7.2. NEC_CoE402HomeEx..... 43
- 2.7.3. NEC_CoE402WaitHomeFinished 45
- 2.7.4. NEC_CoE402ClearHomeStartBit..... 46
- 2.7.5. NEC_CoE402CheckHomeStatus 47
- 2.8. 扭力控制 (Torque Control)..... 48
 - 2.8.1. NEC_CoE402SetTargetTorque..... 48
 - 2.8.2. NEC_CoE402GetTargetTorque 48
 - 2.8.3. NEC_CoE402GetActualTorque 49
 - 2.8.4. NEC_CoE402SetTorqueProfile..... 50

1. CoE CiA402 操作说明

1.1. 流程图



1.2. 基本规格

1. 支持最高 32 轴
2. 支持标准 CoE402 EtherCAT 伺服驱动器

1.3. 注意事项

使用本函数库前，下列几点须注意：

1. ControlWord(0x6040)须设定为 RxPDO, Offset 不限制，但建议设定为 0
2. StatusWord(0x6041)须设定为 TxPDO, Offset 不限制，但建议设定为 0
3. TargetPosition(0x607A) 建议设定为 RxPDO, Offset 建议设定为 2
4. Position actual value(0x6064) 建议设定为 TxPDO, Offset 建议设定为 2

2. CoE CiA402 函式库

2.1. API 总览

下表列出 CiA402 函式库 API 的列表，API 定义于 NexCoEMotion.h 头文件之中。

T:字段代表该函式可被呼叫的位置

C:只能在 Callback 函式中呼叫

X:不能再 Callback 函式中呼叫

B:没有限制

(T: Type → C: Callback only, X:Not for callback, B:Both)

函数名称	说明	T
初始化相关函式		
NEC_CoE402Reset	重置 CiA402 函式库	X
NEC_CoE402Close	关闭 CiA402 函式库	X
NEC_CoE402GetAxisId	注册并取得 CiA 伺服轴的标识符 (单模块单轴)	X
NEC_CoE402GetDefaultMapInfo	初始化 MapInfo 数据结构	X
NEC_CoE402GetAxisIdEx	注册并取得 CiA 伺服轴的标识符 (单模块多轴)	X
NEC_CoE402ResetPdoMapping	重置 PDO mapping	X
NEC_CoE402AddPdoMapping	新增 PDO mapping 设定	X
NEC_CoE402UpdatePdoMapping	更新 PDO mappingg 设定	X
CiA402 状态控制函式		
NEC_CoE402CyclicProcess	周期状态控制函式	C
NEC_CoE402SetCtrlWord	设定 CiA402 ControlWord (0x6040)	B
NEC_CoE402GetCtrlWord	读取 CiA402 ControlWord (0x6040)	B
NEC_CoE402GetStatusWord	读取 CiA402 StatusWord (0x6041)	B
NEC_CoE402ChangeState	改变控制轴的状态	X
NEC_CoE402SetState	设定状态	B
NEC_CoE402GetState	读取驱动器状态	B
NEC_CoE402FaultReset	清除/重置伺服驱动器错误	X
NEC_CoE402ServoOn	设定驱动器激磁/解激磁	X
CiA402 驱动器基本操作函式		
NEC_CoE402SetOperationMode	设定驱动器运动模式 (0x6060)	X
NEC_CoE402GetOperationModeDisplay	读取实际驱动器运动模式 (0x6061)	X
NEC_CoE402SetParameter	设定 CiA402 驱动器对象参数	X
NEC_CoE402GetParameter	读取 CiA402 驱动器对象参数	X
NEC_CoE402GetActualPosition	读取驱动器马达实际位置 (0x6064)	*B

NEC_CoE402SetTargetPosition	设定目标位置 (0x607A, PP, CSP)	*B
NEC_CoE402GetTargetPosition	读取目标位置 (0x607A, PP, CSP)	*B
NEC_CoE402SetQuickStopDec	设定 Quick Stop 减速率 (0x6085)	X
NEC_CoE402SetSoftPosLimit	设定软件位置极限 (0x607D)	X
NEC_CoE402SetMaxVelLimit	设定最大速度限制 (0x607F)	X
点对点运动 (Profile Position mode)		
NEC_CoE402Ptp	启动单轴点对点运动	X
NEC_CoE402PtpV	启动单轴点对点运动, 带速度参数	X
NEC_CoE402PtpA	启动单轴点对点运动, 带速度、加速度参数	X
NEC_CoE402WaitTargetReached	等待目标位置到达	X
速度运动 (Profile Velocity mode)		
NEC_CoE402Jog	启动单轴指定速度运转	X
NEC_CoE402JogA	启动单轴指定速度运转, 带加速度参数	X
NEC_CoE402Halt	停止/暂停运动	X
归零运动 (Homing mode)		
NEC_CoE402Home	启动单轴原点归零运动 (Homing)	X
NEC_CoE402HomeEx	启动单轴原点归零运动 (Homing) 带参数设定	X
NEC_CoE402WaitHomeFinished	等待归零运动结束	X
NEC_CoE402ClearHomeStartBit	清除原点归零运动 (Homing) 启动位	B
NEC_CoE402CheckHomeStatus	读取原点归零运动 (Homing) 状态	B
扭力控制 (Torque Control)		
NEC_CoE402SetTargetTorque	设定目标扭力 (0x6071, PT, CST)	*B
NEC_CoE402GetTargetTorque	读取目标扭力 (0x6071, PT, CST)	*B
NEC_CoE402GetActualTorque	读取实际扭力值 (0x6077)	*B
NEC_CoE402SetTorqueProfile	设定目标扭力与变化率 (0x6071, 0x6087, PT)	X

(*B 表示需将该信息设定为 PDO mapping, 才能在 Callback 中呼叫, 否则回传错误码)

API 所使用的 C/C++ 数据类型定义于 nex_type.h 中, 说明如下表:

型别	C/C++ 原型	说明	大小 byte	范围
BOOL_T	Int	布尔型别	4	0:False, 1:True
U8_T	unsigned char	无号整数	1	0 ~ 255
U16_T	unsigned short	无号整数	2	0 ~ 65535
U32_T	unsigned int	无号整数	4	0 ~ 4294967295
U64_T	unsigned __int64	无号整数	8	0 ~ 18446744073709551615

I8_T	char	有号整数	1	-128 ~ 127
I16_T	short	有号整数	2	-32768 ~ 32767
I32_T	int	有号整数	4	-2147483648 ~ 2147483647
I64_T	__int64	有号整数	8	-9223372036854775808 ~ 9223372036854775807
F32_T	float	浮点数	4	IEEE-754, 有效小数后 7 位
F64_T	double	双精浮点数	8	IEEE-754, 有效小数后 15 位
RTN_ERR	int	错误代码	4	-2147483648 ~ 2147483647

2.2. 初始化相关函数

2.2.1. NEC_CoE402Reset

重置 CiA402 函数库

C/C++语法:

```
RTN_ERR NEC_CoE402Reset();
```

参数:

<无参数>

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

使用其他 CoE402 函数库前，呼叫此函数进行函数库内部初始化工作。

注意! 禁止于 Callback 函数中调用此函数

参阅:

```
NEC_CoE402Close ();
```

2.2.2. NEC_CoE402Close

关闭 CiA402 函式库

C/C++语法:

```
RTN_ERR NEC_CoE402Close ();
```

参数:

<无参数>

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

应用程序结束前，调用此函数释放 CiA402 函式库内部资源。

注意! 禁止于 Callback 函式中调用此函数

参阅:

```
NEC_CoE402Reset();
```

2.2.3. NEC_CoE402GetAxisId

注册并取得 CiA 伺服轴的标识符

C/C++语法:

```
RTN_ERR NEC_CoE402GetAxisId( U16_T MasterId, U16_T SlaveAddr, CANAxis_T  
*pAxis );
```

参数:

U16_T MasterId: 指定目标 EC-Master 代号, 单一 EC-Master 请设为 0

U16_T SlaveAddr: 指定目标 EC-Slave 代号, 依网络配线顺序, 从 0 开始依序增号

CANAxis_T *pAxis: 回传所对应的 CiA 伺服轴的控制标识符(Identification)。用于稍后其他函式。

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

初始化 CiA402 函式库后, 使用本函式注册 CiA402 模块并取得 CiA 伺服轴的控制代号。取得之轴代号用于尔后的其他 CiA402 函式库的呼叫。

此 API 适用于只有单一 CiA402 伺服轴的 Slave 装置, 若单一 Slave 装置具有多轴 (*)需使用 NEC_CoE402GetAxisIdEx()进行注册的动作。

注意! 禁止于 Callback 函式中调用此函数

(*)单一 Slave 装置具有多轴 例如 NEXCOM AXE-5904 (4 axes CoE step/servo interface module)

参阅:

NEC_CoE402CyclicProcess();NEC_CoE402GetAxisIdEx()

2.2.4. NEC_CoE402GetDefaultMapInfo

初始化 MapInfo 数据结构

C/C++语法:

```
RTN_ERR NEC_CoE402GetDefaultMapInfo( U16_T TypeOfSlave, U16_T SlaveAddr,  
EcCiA402AxisMapInfo_T *pMapInfo );
```

参数:

U16_T TypeOfSlave: Slave 型态

EC_CIA402_SERVO_DRIVE (0):标准 CiA402 单轴模块 (单模块单轴)

EC_MULTIPLE_CIA402_SLAVE (1):标准 CiA402 多轴模块 (单模块多轴)

U16_T SlaveAddr: 指定目标 EC-Slave 代号, 依网络配线顺序, 从 0 开始依序增号
EcCiA402AxisMapInfo_T *pMapInfo: CiA 轴 PDO 映像数据, 此数据结构的内容将被初始化如下述:

```
typedef struct  
{  
    U16_T TypeOfSlave;  
    U16_T SlaveAddr;  
    U16_T SlaveSlotNum;  
    U16_T Reserved;          // Reserved, set zero.  
    U16_T CoeObjectOffset; // ex. 0x0800  
    U16_T pdoMapOffset;    // ex. 0x0010  
    U16_T MinRxPdoIndex;   // ex. 0x1600  
    U16_T MaxRxPdoIndex;   // ex. 0x1603  
    U16_T MinTxPdoIndex;   // ex. 0x1A00  
    U16_T MaxTxPdoIndex;   // ex. 0x1A03  
}EcCiA402AxisMapInfo_T;
```

U16_T TypeOfSlave: Slave 型态,同上述并初始化为上述值。

U16_T SlaveAddr: 目标 EC-Slave 代号, 同上述并初始化为上述值。

U16_T SlaveSlotNum: 单模块中之轴顺序,由零递增, 初始化为 0

U16_T Reserved: 保留, 初始化为 0

U16_T CoeObjectOffset: 轴对映的 CoE object 偏移量, 初始化为 0x0800

U16_T pdoMapOffset; 轴对映的 PDO map object 偏移量, 初始化为 0x0010

U16_T MinRxPdoIndex: 第一轴的最小 RxPDO 号码, 初始化为 0x1600

U16_T MaxRxPdoIndex: 第一轴的最大 RxPDO 号码, 初始化为 0x1603

U16_T MinTxPdoIndex: 第一轴的最小 TxPDO 号码, 初始化为 0x1A00

U16_T MaxTxPdoIndex: 第一轴的最大 TxPDO 号码, 初始化为 0x1A03

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

在呼叫 *NEC_CoE402GetAxisIdEx()* 注册 CiA402 伺服轴时必须输入该轴所对应的 PDO 映像数据” EcCiA402AxisMapInfo_T”，可利用本 API 先将” EcCiA402AxisMapInfo_T” 进行基本初始化后，再将此数据结构参数带入 *NEC_CoE402GetAxisIdEx()*

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402GetAxisIdEx()

2.2.5. NEC_CoE402GetAxisIdEx

注册并取得 CiA 伺服轴的标识符(单模块多轴)

C/C++语法:

```
RTN_ERR NEC_CoE402GetAxisIdEx( U16_T MasterId, EcCiA402AxisMapInfo_T *pInfo,  
CANAxis_T *pAxis );
```

参数:

U16_T MasterId: 指定目标 EC-Master 代号, 单一 EC-Master 请设为 0

EcCiA402AxisMapInfo_T *pMapInfo: CiA 轴 PDO 映像数据

```
typedef struct  
{  
    U16_T TypeOfSlave;  
    U16_T SlaveAddr;  
    U16_T SlaveSlotNum;  
    U16_T Reserved;           // Reserved, set zero.  
    U16_T CoeObjectOffset;    // ex. 0x0800  
    U16_T pdoMapOffset;       // ex. 0x0010  
    U16_T MinRxPdoIndex;      // ex. 0x1600  
    U16_T MaxRxPdoIndex;      // ex. 0x1603  
    U16_T MinTxPdoIndex;      // ex. 0x1A00  
    U16_T MaxTxPdoIndex;      // ex. 0x1A03  
}EcCiA402AxisMapInfo_T;
```

U16_T TypeOfSlave: Slave 型态

EC_CIA402_SERVO_DRIVE (0):标准 CiA402 单轴模块 (单模块单轴)

EC_MULTIPLE_CIA402_SLAVE (1):标准 CiA402 多轴模块 (单模块多轴)

U16_T SlaveAddr: 目标 EC-Slave 代号, 由 0 开始依接线递增

U16_T SlaveSlotNum: 单模块中之轴顺序,由零递增, 初始化为 0

U16_T Reserved: 保留, 初始化为 0

U16_T CoeObjectOffset: 轴对映的 CoE object 偏移量, 初始化为 0x0800

U16_T pdoMapOffset; 轴对映的 PDO map object 偏移量, 初始化为 0x0010

U16_T MinRxPdoIndex: 第一轴的最小 RxPDO 号码, 初始化为 0x1600

U16_T MaxRxPdoIndex: 第一轴的最大 RxPDO 号码, 初始化为 0x1603

U16_T MinTxPdoIndex: 第一轴的最小 TxPDO 号码, 初始化为 0x1A00

U16_T MaxTxPdoIndex: 第一轴的最大 TxPDO 号码, 初始化为 0x1A03

CANAxis_T *pAxis: 回传所对应的 CiA 伺服轴的控制标识符(Identification)。用于稍后其他函式。

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

初始化 CiA402 函式库后，使用本函式注册 CiA402 模块并取得 CiA 伺服轴的控制代号。取得之轴代号用于尔后的其他 CiA402 函式库的呼叫。

此 API 对于单一 CiA402 伺服轴的 Slave 装置和单一 Slave 装置具有多轴(*)皆适用

注意! 禁止于 Callback 函式中调用此函数

(*)单一 Slave 装置具有多轴 例如 NEXCOM AXE-5904 (4 axes CoE step/servo interface module)

参阅:

NEC_CoE402CyclicProcess(); NEC_CoE402GetAxisId()

2.2.6. NEC_CoE402ResetPdoMapping

重置 PDO mapping 设定

C/C++语法:

```
RTN_ERR NEC_CoE402ResetPdoMapping( CANAxis_T Axis );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得。

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

重置函数库内的 PDO mapping 设定，呼叫 NEC_CoE402Reset()时会自动呼叫此函数。当 CiA402 Slave 的状态再进入“OP”状态之前。使用者必须正确的设定 PDO mapping 方式。其设定方式有下列 2 种方式:

1. 使用 NEC_CoE402AddPdoMapping()将 PDO 的 Mapping 方式输入置函数库
2. 使用 NEC_CoE402UpdatePdoMapping(), 直接由 CiA402 模块端读取 PDO mapping 的设定。

建议采用方法二叫简洁不易发生人为错误。

注意! 禁止于 Callback 函数中调用此函数

参阅:

NEC_CoE402AddPdoMapping()

2.2.7. NEC_CoE402AddPdoMapping

新增 PDO mapping 设定

C/C++语法:

```
RTN_ERR NEC_CoE402AddPdoMapping( CANAxis_T Axis, U16_T Type, U16_T  
CANIndex, U8_T CANSubIndex, U8_T OffsetOfByte, U8_T LenOfByte );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T Type, 指定新增的项目为 **TYPE_RXPDO (0)**或 **TYPE_TXPDO (1)**

U16_T CANIndex: CAN 对象 Index, 请参 CoE402 伺服马达使用说明

U8_T CANSubIndex: CAN 对象 sub-index, 请参 CoE402 伺服马达使用说明

U8_T OffsetOfByte: 该对象在 Slave PDO Mapping 的位置偏移量, byte 为单位

U8_T LenOfByte: 该对象的长度大小, byte 为单位

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

使用 *NEC_CoE402SetParameter()* / *NEC_CoE402GetParameter()* Ec-Master 会根据 PDO mapping 规则自动判断以 SDO 或 PDO 的方式存取之。透过下列两种方式让 EC-Master 取得 CiA402 slave 模块的 PDO mapping 规则。

方法一:

使用本函数手动设定 EC-Slave 装置的 PDO mapping, 在使用前先呼叫 *NEC_CoE402ResetPdoMapping()*清除 PDO mapping 的设定。上述方式是用户建立该装置的 PDO mapping 规则, 基本上必须依照 ENI 的内容设定, 若设定不一致可能出现异常。

方法二:

另外也可用 *NEC_CoE402UpdatePdoMapping()*函数自动从 EC-Slave 上更新 PDO mapping 数据。

以上两种方式请择一使用。切勿混用。

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402ResetPdoMapping(); *NEC_CoE402UpdatePdoMapping()*

2.2.8. NEC_CoE402UpdatePdoMapping

更新 PDO mapping 设定

C/C++语法:

```
RTN_ERR NEC_CoE402UpdatePdoMapping( CANAxis_T Axis );
```

参数:

CANAxis_T Axis, 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

使用 *NEC_CoE402UpdatePdoMapping()*函数自动从 CiA402 的 EC-Slave 上更新 PDO mapping 数据。注意在函数必须在 EC-Slave 状态为“PREOP”，“SAFEOP”或 OP 状态下方能呼叫此函数。

注意! 禁止于 Callback 函数中调用此函数

参阅:

NEC_CoE402AddPdoMapping();

2.3. CiA402 状态控制函式

2.3.1. NEC_CoE402CyclicProcess

周期状态控制函式

C/C++语法:

```
RTN_ERR NEC_CoE402CyclicProcess();
```

参数:

<无参数>

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

标准 CiA402 slave 模块可透过控制 ControlWord(0x6040) 和 StatusWord(0x6041) 来切换该模块的状态。

使用者除了可以自行控制 ControlWord 来切换 CiA402 的状态，亦可透过本函数来切换状态。

透过 NEC_CoE402GetAxisId()将 CoE 从站模块注册到系统中，所有已注册的模块的状态将由 NEC_CoE402CyclicProcess()管控。

本函数需周期性的执行，一般在 Callback 中呼叫执行本函数。本函数用于控制所有 CiA402 EC-Slaves 的状态，内部实作状态切换程序，用户可直接下达欲切换的 CiA402 状态，无须自行处理状态切换。切换状态相关函数请参考 NEC_CoE402ChangeState(), NEC_CoE402SetState()和 NEC_CoE402ServoOn()

注意，本函数会控制 ControlWord(0x6060) 有关状态切换相关的(Bit0~3)。若用户使用本函数后又自行控制 ControlWord(0x6040)必须避免修改 bit0~3 的内容，否则可能造成竞争状态(Race condition)。

Bit NO.	Function
0	Switch on
1	Enable voltage
2	Quick Stop

3	Enable operation
---	------------------

(CiA402 ControlWord bit0~3 定义)

注意! ControlWord(0x6040)必须被设定为 RxPDO mapping, StatusWord (0x6041)必须被设定为 TxPDO mapping。

参阅:

NEC_CoE402ChangeState(); NEC_CoE402SetState(); NEC_CoE402GetState();
NEC_CoE402ServoOn()

2.3.2. NEC_CoE402SetCtrlWord

2.3.3. NEC_CoE402GetCtrlWord

设定/读取 CiA402 ControlWord (0x6040)

C/C++语法:

```
RTN_ERR NEC_CoE402SetCtrlWord( CANAxis_T Axis, U16_T CtrlInBit );
```

```
RTN_ERR NEC_CoE402GetCtrlWord( CANAxis_T Axis, U16_T *CtrlInBit );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T CtrlInBit: 设定 CiA402 ControlWord, Object index = 0x6040

U16_T *CtrlInBit: 读取 CiA402 ControlWord, Object index = 0x6040

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数用于控制 CiA402 ControlWord，对象序号 0x6040，本函数可用于 Callback 函数之中。注意! ControlWord(0x6040)必须被设定为 RxPDO mapping。ControlWord 于 CiA402 中的定义如下:

Bit NO.	Function
0	Switch on
1	Enable voltage
2	Quick Stop
3	Enable operation
4 ~ 6	<Depend on operation mode> (*)
7	Fault reset
8	Halt
9	<Depend on operation mode> (*)
10	Reserved
11~15	<Vendor specific> (*)

(*)请参与 CiA402 装置使用手册

若采用 NEC_CoE402CyclicProcess()来控制 CiA402 状态切换，该函数会控制 ControlWord(0x6040)的 Bit0~3。因此应避免 Bit0~3 的修改。若要同时使用，可先读取 ControlWord 修改欲设定的 bit 后再写入。

参阅:

NEC_CoE402CyclicProcess()

2.3.4. NEC_CoE402GetStatusWord

读取 CiA402 StatusWord (0x6041)

C/C++语法:

```
RTN_ERR NEC_CoE402GetStatusWord( CANAxis_T Axis, U16_T *StatusInBit );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T *StatusInBit: 读取 CiA402 StatusWord, Object index = 0x6041

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数用于读取 CiA402 StatusWord(对象序号 0x6041)，本函数可用于 Callback 函数之中。注意! StatusWord (0x6041)必须被设定为 TxPDO mapping。StatusWord 于 CiA402 中的定义如下:

Bit NO.	Function	Bit NO.	Function
0	Ready to switch on	8	<Vendor specific> (*)
1	switch on	9	Remote
2	Operatiion enable	10	<Depend on operation mode> (*)
3	Fault	11	Internal limit active
4	Voltage enable	12	<Depend on operation mode> (*)
5	Quick stop	13	
6	Swithec on disabled	14	<Vendor specific> (*)
7	Warning	15	

参阅:

2.3.5. NEC_CoE402ChangeState

改变控制轴的状态

C/C++语法:

```
RTN_ERR NEC_CoE402ChangeState( CANAxis_T Axis, U16_T TargetState, I32_T
TimeoutMs );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T TargetState: 欲切换之 CiA402 状态。

#define	CiA402 state	Note
COE_STA_DISABLE(0)	Switch on disable	Servo OFF
COE_STA_READY_TO_SWITCH_ON(1)	Ready to switch on	
COE_STA_SWITCH_ON (2)	Switched on	
COE_STA_OPERATION_ENABLE (3)	Operation enable	Servo ON
COE_STA_QUICK_STOP_ACTIVE (4)	Quick stop active	

I32_T TimeoutMs: 切换状态逾时等待时间，单位 millisecond。当 TimeoutMs 设定为 0 时，等同于呼叫 NEC_CoE402SetState() 函式。

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于 EcErrors.h 头文件中。

用法:

设定马达状态，当函数成功返回表示已成功切换状态。若需在 Callback 之中切换马达状态必须使用 NEC_CoE402SetState()

本函式必须配合 NEC_CoE402CyclicProcess() 一起使用。

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402CyclicProcess(); NEC_CoE402SetState();

2.3.6. NEC_CoE402SetState

设定驱动器状态

C/C++语法:

```
RTN_ERR NEC_CoE402SetState( CANAxis_T Axis, U16_T State );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T State: 欲切换之 CiA402 状态。

#define	CiA402 state	Note
COE_STA_DISABLE(0)	Switch on disable	
COE_STA_READY_TO_SWITCH_ON(1)	Ready to switch on	
COE_STA_SWITCH_ON (2)	Switched on	
COE_STA_OPERATION_ENABLE (3)	Operation enable	Servo ON
COE_STA_QUICK_STOP_ACTIVE (4)	Quick stop active	

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，用于设定欲切换之马达状态，此函数设定完后立即返回不会等待马达状态改变，用户必须使用 NEC_CoE402GetState()确定马达的状态。注意! ControlWord(0x6040)必须被设定为 RxPDO mapping。

本函式必须配合 NEC_CoE402CyclicProcess()一起使用。

参阅:

NEC_CoE402CyclicProcess(); NEC_CoE402ChangeState();

2.3.7. NEC_CoE402GetState

读取驱动器状态

C/C++语法:

```
RTN_ERR NEC_CoE402GetState( CANAxis_T Axis, U16_T *State );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T *State: 回传马达状态。

#define	CiA402 state	Note
COE_STA_DISABLE(0)	Switch on disable	
COE_STA_READY_TO_SWITCH_ON(1)	Ready to switch on	
COE_STA_SWITCH_ON (2)	Switched on	
COE_STA_OPERATION_ENABLE (3)	Operation enable	Servo ON
COE_STA_QUICK_STOP_ACTIVE (4)	Quick stop active	
COE_STA_FAULT(5)	Fault	Error
COE_STA_FAULT_REACTION_ACTIVE (6)	Fault reaction active	Error

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，读取马达目前实际状态。

参阅:

NEC_CoE402SetState();

2.3.8. NEC_CoE402FaultReset

清除/重置伺服驱动器错误

C/C++语法:

```
RTN_ERR NEC_CoE402FaultReset( CANAxis_T Axis, I32_T TimeoutMs );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T TimeoutMs: 逾时等待时间, 单位 millisecond。当 TimeoutMs 设定为 0 或小于 0 时等待时间设定为 5000ms

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

当驱动器状态进入 **Fault** 状态时, 先确认驱动器的错误状态并施以排除后, 使用此函数解除 **Fault** 状态。

注意! 禁止于 Callback 函式中调用此函数

参阅:

2.3.9. NEC_CoE402ServoOn

设定驱动器激磁/解激磁

C/C++语法:

```
RTN_ERR NEC_CoE402ServoOn( CANAxis_T Axis, U16_T OnOff );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T OnOff:

0: Servo off (Switch on disable)

1 Servo ON (Operation enable)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

设定驱动器激磁(ServoOn)与解激磁(ServoOff)

ServoOn 相当于 CiA 状态: Operation Enable

ServoOff 相当于 CiA 状态: Switch on disable

本函数必须配合 NEC_CoE402CyclicProcess()一起使用。

注意! 禁止于 Callback 函数中调用此函数

参阅:

NEC_CoE402CyclicProcess(); NEC_CoE402SetState(); NEC_CoE402ChangeState();

2.4. CiA402 驱动器基本操作函式

2.4.1. NEC_CoE402SetOperationMode

设定驱动器运动模式(0x6060)

C/C++语法:

```
RTN_ERR NEC_CoE402SetOperationMode( CANAxis_T Axis, I8_T MotionMode,
I32_T CheckTimeoutMs );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I8_T MotionMode: 设定运动模式。其支持的模式请参考驱动器使用手册

Define	CiA402 运动模式
CiA402_OP_MODE_PROFILE_POSITION (1)	Profile Position mode
CiA402_OP_MODE_PROFILE_VELOCITY (3)	Profile Velocity mode
CiA402_OP_MODE_TORQUE_PROFILE (4)	Torque Profile mode
CiA402_OP_MODE_HOMING (6)	Homing mode
CiA402_OP_MODE_INTERPOLATED_POSITION (7)	Interpolated Position mode
CiA402_OP_MODE_CYCLIC_POSITION (8)	Cyclic Sync Position mode
CiA402_OP_MODE_CYCLIC_VELOCITY (9)	Cyclic Sync Velocity mode
CiA402_OP_MODE_CYCLIC_TORQUE (10)	Cyclic Sync Torque mode

I32_T CheckTimeoutMs: 逾时等待时间，单位 millisecond。

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

设定驱动器运动模式(Mode of operation)，对象序号 0x6060。函数成功返回代表已成功切换驱动器的运动模式。

注意! 禁止于 Callback 函式中调用此函数

参阅:

2.4.2. NEC_CoE402GetOperationModeDisplay

读取实际驱动器运动模式(0x6061)

C/C++语法:

```
RTN_ERR NEC_CoE402GetOperationModeDisplay( CANAxis_T Axis, I8_T
*MotionMode );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I8_T *MotionMode: 回传目前驱动器的运动状态

Define	CiA402 运动模式
CiA402_OP_MODE_PROFILE_POSITION (1)	Profile Position mode
CiA402_OP_MODE_PROFILE_VELOCITY (3)	Profile Velocity mode
CiA402_OP_MODE_TORQUE_PROFILE (4)	Torque Profile mode
CiA402_OP_MODE_HOMING (6)	Homing mode
CiA402_OP_MODE_INTERPOLATED_POSITION (7)	Interpolated Position mode
CiA402_OP_MODE_CYCLIC_POSITION (8)	Cyclic Sync Position mode
CiA402_OP_MODE_CYCLIC_VELOCITY (9)	Cyclic Sync Velocity mode
CiA402_OP_MODE_CYCLIC_TORQUE (10)	Cyclic Sync Torque mode

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

读回驱动器运动模式(Mode of operation display)，对象序号 0x6061。

注意! 禁止于 Callback 函式中调用此函数

参阅:

2.4.3. NEC_CoE402SetParameter

设定 CiA402 驱动器对象参数

C/C++语法:

```
RTN_ERR NEC_CoE402SetParameter( CANAxis_T Axis, U16_T Index, U8_T SubIndex,  
U8_T LenOfByte, I32_T Value);
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T Index: CiA 对象参数序号, 请参阅驱动器手册

U8_T SubIndex: CiA 对象参数子序号, 请参阅驱动器手册

U8_T LenOfByte: 物件的大小(1~4 bytes), 单位 byte

I32_T Value: 物件参数值

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

设定 CiA402 驱动器参数。使用本函数存取 CiA402 模块参数, 本函数会自动根据该模块目前 PDO mapping 的方式自动采用 PDO 或者 SDO 的方式存取。若对象参数为 PDO mapping 参数, 本指令以 PDO 存取。若非 PDO mapping 参数则采用 SDO 方式存取。使用本函数必须前必须在程序初始化的过程中呼叫 NEC_CoE402UpdatePdoMapping()。

驱动器的相关参数请参阅驱动器使用手册。

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402GetParameter (); NEC_CoE402UpdatePdoMapping()

2.4.4. NEC_CoE402GetParameter

读取 CiA402 驱动器对象参数

C/C++语法:

```
RTN_ERR NEC_CoE402GetParameter( CANAxis_T Axis, U16_T Index, U8_T SubIndex,  
U8_T LenOfByte, void *pRetVal );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T Index: CiA 对象参数序号, 请参阅驱动器手册

U8_T SubIndex: CiA 对象参数子序号, 请参阅驱动器手册

U8_T LenOfByte: 物件的大小(1~4 bytes), 单位 byte

void *pRetVal: 回传对象参数值, 给入指针的数据值的大小必需符合 LenOfByte 以避免指标存取错误.

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

读取 CiA402 驱动器的参数, 驱动器的相关参数请参阅驱动器使用手册。使用本函数存取 CiA402 模块参数, 本函数会自动根据该模块目前 PDO mapping 的方式自动采用 PDO 或者 SDO 的方式存取。若对象参数为 PDO mapping 参数, 本指令以 PDO 存取。若非 PDO mapping 参数则采用 SDO 方式存取。使用本函数必须前必须在程序初始化的过程中呼叫 NEC_CoE402UpdatePdoMapping()。

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402SetParameter (); NEC_CoE402UpdatePdoMapping();

2.4.5. NEC_CoE402GetActualPosition

读取驱动器马达实际位置(0x6064)

C/C++语法:

```
RTN_ERR NEC_CoE402GetActualPosition( CANAxis_T Axis, I32_T *Position );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T *Position: 回传马达实际位置(对象序号:0x6040)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，必须将对象 0x6064 设定为 TxPDO mapping 对象。

参阅:

2.4.6. NEC_CoE402SetTargetPosition

2.4.7. NEC_CoE402GetTargetPosition

设定/读取目标位置(0x607A,PP,CSP)

C/C++语法:

```
RTN_ERR NEC_CoE402SetTargetPosition( CANAxis_T Axis, I32_T TargetPos );
```

```
RTN_ERR NEC_CoE402GetTargetPosition( CANAxis_T Axis, I32_T *TargetPos );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T TargetPos: 设定目标位置(CoE: 0x607A)

I32_T *TargetPos: 回传目标位置(CoE: 0x607A)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，必须将对象 0x607A 设定为 RxPDO mapping 对象。TargetPositon 被用于 Profile Position mode(PP)和 Cyclic Sync Position mode(CSP)运动模式，可用 NEC_CoE402SetOperationMode()切换运动模式

参阅:

NEC_CoE402SetOperationMode()

2.4.8. NEC_CoE402SetQuickStopDec

设定 Quick Stop 减速率(0x6085)

C/C++语法:

```
RTN_ERR NEC_CoE402SetQuickStopDec( CANAxis_T Axis, U32_T QuickStopDec );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U32_T QuickStopDec: 设定 Quick Stop 减速率(0x6085)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

此函数等同呼叫 NEC_CoE402SetParameter(Axis, 0x6085, 0, 4, (I32_T) QuickStopDec);

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402SetParameter ();

2.4.9. NEC_CoE402SetSoftPosLimit

设定软件位置极限(0x607D)

C/C++语法:

```
RTN_ERR NEC_CoE402SetSoftPosLimit( CANAxis_T Axis, I32_T MinPositionLimit,  
I32_T MaxPositionLimit);
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T MinPositionLimit: 负软件极限位置 (0x607D : 01)

I32_T MaxPositionLimit: 正软件极限位置 (0x607D : 02)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

若要解除软件正负极限的功能将正负极限同时设定为 0。

此函数等同呼叫 NEC_CoE402SetParameter(Axis, 0x607D, 1, 4, MinPositionLimit)

设定负极限 and NEC_CoE402SetParameter(Axis, 0x607D, 2, 4, MaxPositionLimit)

设定正极限.

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402SetParameter ();

2.4.10. NEC_CoE402SetMaxVelLimit

设定最大速度限制(0x607F)

C/C++语法:

```
RTN_ERR NEC_CoE402SetMaxVelLimit( CANAxis_T Axis, U32_T MaxVelocityLimit );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U32_T MaxVelocityLimit: 设定最大速度限制(0x607F)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

在 Profile Position mode (P2P), Profile Velocity mode(Jog) 等模式中设定最大速度限制。此函数等同呼叫 NEC_CoE402SetParameter(Axis, 0x607F, 0, 4, (I32_T) MaxVelocityLimit);

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402SetParameter ();

2.4.11. NEC_CoE402SetTargetVelocity

设定目标速度(0x60FF,PV,CSV)

C/C++语法:

```
RTN_ERR NEC_CoE402SetTargetVelocity ( CANAxis_T Axis, I32_T TargetVel );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T TargetVel: 设定目标速度(CoE: 0x60FF)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，必须将对象 0x60FF 设定为 RxPDO mapping 对象。TargetVelocity 被用于 Profile Velocity mode(PV)和 Cyclic Sync Velocity mode(CSV) 运动模式，可用 NEC_CoE402SetOperationMode()切换运动模式

参阅:

NEC_CoE402SetOperationMode()

2.4.12. NEC_CoE402GetActualVelocity

读取当前速度(0x606C,PV,CSV)

C/C++语法:

```
RTN_ERR NEC_CoE402GetActualVelocity( CANAxis_T Axis, I32_T *ActualVel);
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T *ActualVel: 读取当前速度(CoE: 0x606C)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，必须将对象 0x606C 设定为 TxPDO mapping 对象。ActualVelocity 被用于 Profile Velocity mode(PV)和 Cyclic Sync Velocity mode(CSV)运动模式，可用 NEC_CoE402SetOperationMode()切换运动模式

参阅:

NEC_CoE402SetOperationMode()

2.5. 点对点运动 (Profile Position mode)

2.5.1. NEC_CoE402Ptp

2.5.2. NEC_CoE402PtpV

2.5.3. NEC_CoE402PtpA

启动单轴点对点运动(Profile Position mode)

C/C++语法:

```
RTN_ERR NEC_CoE402Ptp( CANAxis_T Axis, U32_T Option, I32_T TargetPos );  
RTN_ERR NEC_CoE402PtpV( CANAxis_T Axis, U32_T Option, I32_T TargetPos, U32_T  
MaxVel );  
RTN_ERR NEC_CoE402PtpA( CANAxis_T Axis, U32_T Option, I32_T TargetPos, U32_T  
MaxVel, U32_T Acc, U32_T Dec );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U32_T Option: 功能选项，采 bit 形式。两个功能以上，以 OR 方式输入

OPT_ABS (0x00000000): TargetPos 为绝对位置坐标

OPT_REL(0x00000040): TargetPos 为相对位置坐标

OPT_WMC (0x00010000): 等待 PTP 运动位置到达

OPT_IMV(0x10000000): 忽略 MaxVel 输入参数

OPT_IAC (0x20000000):忽略 Acc 输入参数

OPT_IDC (0x40000000): 忽略 Dec 输入参数

I32_T TargetPos: 绝对或相对目标位置(CoE: Target Position 0x607A), 由 Option 设定决定 TargetPos 代表绝对或相对位置。

U32_T MaxVel: 最大速度 (CoE: Profile Velocity 0x6081)

U32_T Acc: 加速率 (CoE: Profile Acceleration 0x6083)

U32_T Dec: 减速率 (CoE: Profile Deceleration 0x6084)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函式使用“Profile Position mode”来完成点对点运动，因此须确认 CiA402 模块是否支持 Profile Position (PP)模式。呼叫本函式时会检查并自动切换 Operation Mode 是否为 PP 模式。而 Ptp, PtpV 和 PtpA 的差别在于输入参数不同：

*NEC_CoE402Ptp()*只设定目标位置 TargetPos (CoE: Target Position 0x607A) 并启动点对点运动，最大速度 MaxVel (CoE: Profile Velocity 0x6081)及加减速速度 Acc, Dec (CoE: Profile Acceleration 0x6083, Profile Deceleration 0x6084) 基本上依照 CiA402 模块目前的设定值。

NEC_CoE402PtpV() 设定目标位置 TargetPos (CoE: Target Position 0x607A) 和最大速度 MaxVel (CoE: Profile Velocity 0x6081)并启动点对点运动，而加减速速度 Acc, Dec (CoE: Profile Acceleration 0x6083, Profile Deceleration 0x6084)是依照 CiA402 模块目前的设定值。

*NEC_CoE402PtpA()*设定所有相关参数包含目标位置 TargetPos (CoE: Target Position 0x607A)，最大速度 MaxVel (CoE: Profile Velocity 0x6081)，加速度 Acc (CoE: Profile Acceleration 0x6083) 和减速度 Dec (CoE: Profile Deceleration 0x6084)并启动点对点运动。

Option “OPT_REL (0x00000040)” 决定 Target Position 参数为相对位置或绝对位置

当 Option:OPT_WMC (Wait motion complete) 至能(enable)，此函数会等待点对点运动完成后或错误发生时返回，其效果等同于呼叫 *NEC_CoE402WaitTargetReached()*

OPT_IMV(0x10000000), OPT_IAC (0x20000000)和 OPT_IDC (0x40000000)是适用于 *NEC_CoE402PtpA()*，表示忽略输入参数使用模块内部设定值。

呼叫 *NEC_CoE402Halt()* 可以暂停或从启 PTP 运动。

注意! 禁止于 Callback 函式中调用此函数

参阅：

NEC_CoE402WaitTargetReached();NEC_CoE402Halt()

2.5.4. NEC_CoE402WaitTargetReached

等待目标位置到达

C/C++语法:

```
RTN_ERR NEC_CoE402WaitTargetReached( CANAxis_T Axis );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

此函数为同步函数，通常配合 NEC_CoE402Ptp(), NEC_CoE402PtpV(), NEC_CoE402PtpA(), NEC_CoE402Jog() 和 NEC_CoE402JogA() 函式使用。

若在 PTP 后呼叫，呼叫后直到 PTP 运动结束后才返回。

若在 Jog 后呼叫，呼叫后直到到达 Target Velocity 速度后才返回。

若返回错误代码表示运动失败或者暂停。

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402Ptp(); NEC_CoE402PtpV();

NEC_CoE402PtpA(); NEC_CoE402Jog(); NEC_CoE402JogA();

2.6. 速度运动(Profile Velocity mode)

2.6.1. NEC_CoE402Jog

2.6.2. NEC_CoE402JogA

启动单轴指定速度运转(Profile velocity mode)

C/C++语法:

```
RTN_ERR NEC_CoE402Jog( CANAxis_T Axis, U32_T Option, I32_T MaxVel );  
RTN_ERR NEC_CoE402JogA( CANAxis_T Axis, U32_T Option, I32_T MaxVel, I32_T Acc,  
I32_T Dec );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U32_T Option: 功能选项, 采 bit 形式。两个功能以上, 以 OR 方式输入

OPT_WMC (0x00010000): 等待到达所设定的最大速度后函数返回

OPT_IAC (0x20000000): 忽略 Acc 输入参数

OPT_IDC (0x40000000): 忽略 Dec 输入参数

I32_T MaxVel: 目标速度 (CoE: Target Velocity 0x60FF)

数值正负决定运动方向。

U32_T Acc: 加速度 (CoE: Profile Acceleration 0x6083)

U32_T Dec: 减速度 (CoE: Profile Deceleration 0x6084)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

本函数使用“Profile Velocity mode”来完成 Jog 运动, 因此须确认 CiA402 模块是否支持 Profile Velocity(PV)模式。呼叫本函数时会检查并自动切换 Operation Mode 是否为 PV 模式。而 Jog, 和 JogA 的差别在于输入参数不同:

NEC_CoE402Jog()只设定目标速度 “MaxVel” (CoE: Target Velocity 0x60FF) 并启动 Jog 运动, 加减速速度(CoE: Profile Acceleration and Deceleration: 0x6083, 0x6084) 基本上是依照 CiA402 模块目前的设定值。

NEC_CoE402JogA() 设定所有相关参数包含目标速度"MaxVel" (CoE: Target Velocity 0x60FF), 加速度 "Acc" (CoE: Profile Acceleration 0x6083) 和减速度"Dec" (CoE: Profile Deceleration 0x6084)并启动 Jog 运动。

当 Option:OPT_WMC (Wait motion complete) 至能(enable), 此函数会等待 Jog 运动到达设定的速度后或错误发生时返回, 其效果等同于呼叫 *NEC_CoE402WaitTargetReached()*

OPT_IAC (0x20000000)和 OPT_IDC (0x40000000)是适用于 *NEC_CoE402JogA ()*, 表示忽略输入参数使用模块内部设定值。

呼叫 *NEC_CoE402Halt()* 可以暂停或从启 Jog 运动。

注意! 禁止于 Callback 函式中调用此函数

参阅:

NEC_CoE402WaitTargetReached(); NEC_CoE402Halt()

2.6.3. NEC_CoE402Halt

停止/暂停运动

C/C++语法:

```
RTN_ERR NEC_CoE402Halt( CANAxis_T Axis, I32_T OnOff );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I32_T OnOff: 0:解除, 1:停止或暂停

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

本含式主要控制 Controlword bit8:Halt (CoE: 0x6040), 用于停止或暂停目前的运动, 停止的动作行为通常搭配 Halt option code (CoE:0x605D)参数的设定。详细的动作规格需参阅该模块说明书。

NEC_CoE402Halt()指令适用于所有操作模式(mode of operation), 当参数“OnOff”设定为 1 时表示停止或暂停运动, 若要重启运动必须先解除 Halt 状态, 须将参数“OnOff”设定为 0。

注意! 禁止于 Callback 函数中调用此函数

参阅:

NEC_CoE402SetCtrlWord(); NEC_CoE402GetCtrlWord()

2.7. 归零运动 (Homing mode)

2.7.1. NEC_CoE402Home

2.7.2. NEC_CoE402HomeEx

启动单轴原点归零运动(Homing)

C/C++语法:

```
RTN_ERR NEC_CoE402Home( CANAxis_T Axis, U32_T Option );
```

```
RTN_ERR NEC_CoE402HomeEx( CANAxis_T Axis, U32_T Option, I8_T Method, I32_T  
Offset, U32_T MaxVel, U32_T ZeroVel, U32_T Acc );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U32_T Option: 功能选项, 采 bit 形式。两个功能以上, 以 OR 方式输入

OPT_WMC (0x00010000): 等待 Home 运动完成

OPT_MTD (0x08000000): 忽略 Method 输入参数

OPT_IMV (0x10000000): 忽略 MaxVel 输入参数

OPT_IAC (0x20000000): 忽略 Acc 输入参数

OPT_IZV (0x40000000): 忽略 ZeroVel 输入参数

OPT_IOF (0x80000000): 忽略 Offset 输入参数

I8_T Method: 归零模式(CoE: 0x6098), 请参与 CiA402 驱动器手册

I32_T Offset: 原点位置偏移量(CoE: 0x607C)

U32_T MaxVel: 归零运动最大速度(CoE: 0x6099:1)

U32_T ZeroVel: 靠近寻找原点速度(CoE: 0x6099:2)

U32_T Acc: 归零运动加速率(CoE: 0x609A)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0), 反之函数调用失败回传错误代码, 错误代码定义于EcErrors.h头文件中。

用法:

本函式使用 CoE “Homing mode”来完成归零运动(Homing), 因此须确认 CiA402 模块是否支持 Homing(HM)模式。呼叫本函式时会检查并自动切换 Operation Mode 是否为 HM 模式。而 Home,和 HomeEx 的差别在于输入参数不同:

NEC_CoE402Home() 单纯用来启动归零运动(Homing), 相关的 Homing 参数如归零模式(CoE: 0x6098), 原点位置偏移量(CoE: 0x607C)等基本上是依照 CiA402 模块目前的设定值。可使用 *NEC_CoE402SetParameter()*单独设定相关参数值。

NEC_CoE402HomeEx() 设定所有相关参数包含归零模式 “Method “ (CoE: 0x6098), 原点位置偏移量 “Offset “ (CoE: 0x607C), 归零运动最大速度 “MaxVel” (CoE: 0x6099:1), 靠近寻找原点速度 “ZeroVel” (CoE: 0x6099:2)和归零运动加速率 “Acc” (CoE: 0x609A)并启动 Homing 运动。

当 Option:OPT_WMC (Wait motion complete) 至能(enable), 此函数会等待 Homing 运动到完成后或错误发生时返回, 其效果等同于呼叫 *NEC_CoE402WaitHomeFinished()*

Option: OPT_MTD (0x08000000), OPT_IMV (0x10000000), OPT_IAC (0x20000000), OPT_IZV (0x40000000), OPT_IOF (0x80000000)只适用于 *NEC_CoE402HomeEx()*, 表示忽略输入参数使用模块内部设定值。

呼叫 *NEC_CoE402Halt()* 可以暂停或从启 Homing 运动。

注意!

禁止于 Callback 函式中调用此函数

除了引用“ OPT_WMC”选项外, 此函数返回时, 不会清除归零运动启动位。我们建议用户在归零运动程序完成后, 调用 *NEC_CoE402ClearHomeStartBit* 函数清除归零运动启动位。

参阅:

NEC_CoE402WaitHomeFinished(); *NEC_CoE402Halt()*;
NEC_CoE402ClearHomeStartBit();

2.7.3. NEC_CoE402WaitHomeFinished

等待单轴原点归零运动(Homing)结束

C/C++语法:

```
RTN_ERR NEC_CoE402WaitHomeFinished( CANAxis_T Axis );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

此函数为同步函数，呼叫后直到归零运动结束后才返回。若返回错误代码表示归零运动失败或者暂停。

注意! 禁止于 Callback 函数中调用此函数

参阅:

```
NEC_CoE402Home();NEC_CoE402HomeEx();
```

2.7.4. NEC_CoE402ClearHomeStartBit

清除单轴原点归零运动(Homing)启动位

C/C++语法:

```
RTN_ERR NEC_CoE402ClearHomeStartBit ( CANAxis_T Axis );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

此函数用于清除原点归零运动启动位。当马达正处在归零运动种态下时，调用此函数，马达的行为会依据驱动器的设计，立即停止或继续执行归零运动。

参阅:

```
NEC_CoE402Home();NEC_CoE402HomeEx();
```

2.7.5. NEC_CoE402CheckHomeStatus

检查当前归零运动状态

C/C++语法:

```
RTN_ERR NEC_CoE402CheckHomeStatus( CANAxis_T Axis, U16_T *pStatus );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

U16_T *pStatus: 传回当前归零运动状态

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

此函数用以取得当前归零运动状态。用户可在执行归零运动后，调用此函数已得知当前归零运动状态。此函数可能回传的状态如下所列：

#define HOMING_IN_PROGRESS	(0)
#define HOMING_TARGET_NOT_REACH	(1)
#define HOMING_COMPLETE	(2)
#define HOMING_INTERRUPTED	(3)
#define HOMING_ERR_VEL_ZERO	(4)
#define HOMING_ERR_VEL_NON_ZERO	(5)

除了归零运动仍处在“HOMING_IN_PROGRESS”之外，此函数在返回前，会一并清除归零运动启动位。

参阅:

NEC_CoE402Home(); NEC_CoE402HomeEx();

2.8. 扭力控制 (Torque Control)

2.8.1. NEC_CoE402SetTargetTorque

2.8.2. NEC_CoE402GetTargetTorque

设定/读取目标扭力 (CoE: 0x6071, PT, CST)

C/C++语法:

```
RTN_ERR NEC_CoE402SetTargetTorque( CANAxis_T Axis, I16_T TargetTorque );  
RTN_ERR NEC_CoE402GetTargetTorque( CANAxis_T Axis, I16_T *TargetTorque );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I16_T TargetTorque: 设定目标扭力(CoE: 0x6071)

I16_T * TargetTorque: 回传目标扭力(CoE: 0x6071)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，但必须将对象 0x6071 设定为 RxPDO mapping 对象。

TargetTorque 通常被用于 Profile Torque mode(PT)和 Cyclic Sync Torque mode(CST) 运动模式，可用 NEC_CoE402SetOperationMode()切换运动模式。

Torque 单位视驱动器容量而定，通常一刻度为 0.1%，详细请参阅相关手册。

参阅:

NEC_CoE402SetOperationMode();NEC_CoE402GetActualTorque()

2.8.3. NEC_CoE402GetActualTorque

读取实际扭力 (CoE: 0x6077, PT, CST)

C/C++语法:

```
RTN_ERR NEC_CoE402GetActualTorque( CANAxis_T Axis, I16_T  
*TorqueActualValue );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I16_T *TorqueActualValue: 回传实际扭力 (CoE: 0x6077)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函数可于 Callback 函式中使用，但必须将对象 0x6077 设定为 TxPDO mapping 对象。

Torque 单位视驱动器容量而定，通常一刻度为 0.1%，详细请参阅相关手册。

参阅:

NEC_CoE402SetTargetTorque(); NEC_CoE402GetTargetTorque();

2.8.4. NEC_CoE402SetTorqueProfile

设定目标扭力与变化率(0x6071, 0x6087)

C/C++语法:

```
RTN_ERR NEC_CoE402SetTorqueProfile( CANAxis_T Axis, I16_T TargetTorque, U32_T TorqueSlope );
```

参数:

CANAxis_T Axis: 指定控制轴代号。轴代号由 NEC_CoE402GetAxisId()取得

I16_T TargetTorque: 设定目标扭力 (CoE: 0x6071)

U32_T TorqueSlope: 设定扭力变化率 (CoE: 0x6087)

回传值:

回传错误代码。

调用函数成功回传“ECERR_SUCCESS” (0)，反之函数调用失败回传错误代码，错误代码定义于EcErrors.h头文件中。

用法:

本函式使用“Profile Torque mode”来完成 Torque 控制，因此须确认 CiA402 模块是否支持 Profile Torque (PT)模式。呼叫本函式时会检查并自动切换 Operation Mode 是否为 PT 模式。

本函式会同时设定目标扭力 “TargetTorque” (CoE: 0x6071)与扭力变化率 “TorqueSlope” (CoE: 0x6087)，并自动判断相关的 CoE 对象使用 PDO 或 SDO 存取。

Torque 单位视驱动器容量而定，通常一刻度为 0.1%，详细请参阅相关手册。

注意! 禁止于 Callback 函式中调用此函数

参阅:

```
NEC_CoE402SetOperationMode(); NEC_CoE402SetTargetTorque();  
NEC_CoE402GetActualTorque()
```