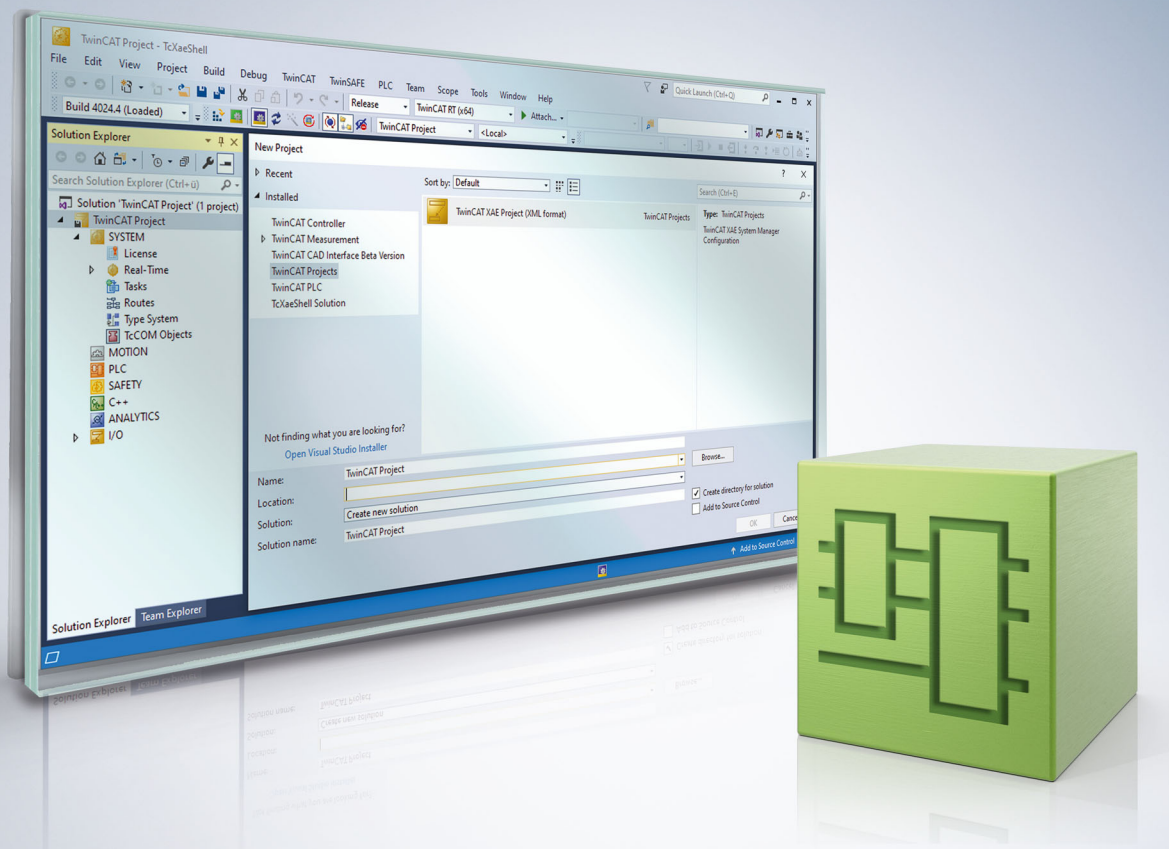
BECKHOFF New Automation Technology

手册 | ZH

TE1000

TwinCAT 3 | PLC Library: Tc2_Utilities



2025-08-05 | 版本： 2.17.0

目录

1 前言

1.1 文档说明

本说明仅适用于熟悉国家标准且经过培训的控制和自动化工程专家。
在安装和调试组件时，必须遵循文档和以下说明及解释。
操作人员应具备相关资质，并始终使用最新的生效文档。

相关负责人员必须确保所述产品的应用或使用符合所有安全要求，包括所有相关法律、法规、准则和标准。

免责声明

本文档经过精心准备。然而，所述产品正在不断开发中。
我们保留随时修改和更改本文档的权利，恕不另行通知。
不得依据本文档中的数据、图表和说明对已供货产品的修改提出赔偿。

商标

Beckhoff®、ATRO®、EtherCAT®、EtherCAT G®、EtherCAT G10®、EtherCAT P®、MX-System®、Safety over EtherCAT®、TC/BSD®、 TwinCAT®、TwinCAT/BSD®、TwinSAFE®、XFC®、XPlanar® 和 XTS®是 Beckhoff Automation GmbH
的注册商标并由其授权使用。本出版物中所使用的其它名称可能是商标名称，任何第三方出于其自身目的使用它们可能会侵犯商标所有者的权利。

EtherCAT®是注册商标和专利技术，由 Beckhoff Automation GmbH 授权使用。

版权所有

第三方商标

本文档可能使用了第三方商标。有关商标信息，可以访问：https://www.beckhoff.com/trademarks。

1.2 安全信息

安全规范

为了确保您的使用安全，请务必仔细阅读
并遵守本文档中每个产品的安全使用说明。

责任免除

所有组件在供货时都配有适合应用的特定硬件和软件配置。严禁未按文档所述修改硬件或软件配置，否则，德国倍福自动化有限公司对由此产生的后果不承担责任。

人员资格

本说明仅供熟悉适用国家标准的控制、自动化和驱动工程专家使用。

警示性词语

文档中使用的警示信号词分类如下。为避免人身伤害和财产损失，请阅读并遵守安全和警告注意事项。

人身伤害警告

⚠ 危险
存在死亡或重伤的高度风险。

⚠ 警告
存在死亡或重伤的中度风险。

⚠ 谨慎
存在可能导致中度或轻度伤害的低度风险。

财产或环境损害警告

注意
可能会损坏环境、设备或数据。

操作产品的信息

这些信息包括：
有关产品的操作、帮助或进一步信息的建议。

1.3 信息安全说明

Beckhoff Automation GmbH & Co.KG (简称 Beckhoff) 的产品，只要可以在线访问，都配备了安全功能，支持工厂、系统、机器和网络的安全运行。尽管配备了安全功能，但为了保护相应的工厂、系统、机器和网络免受网络威胁，必须建立、实施和不断更新整个操作安全概念。Beckhoff 所销售的产品只是整个安全概念的一部分。客户有责任防止第三方未经授权访问其设备、系统、机器和网络。它们只有在采取了适当的保护措施的情况下，方可与公司网络或互联网连接。

此外，还应遵守 Beckhoff 关于采取适当保护措施的建议。关于信息安全和工业安全的更多信息，请访问本公司网站 https://www.beckhoff.com/secguide。

Beckhoff 的产品和解决方案持续进行改进。这也适用于安全功能。鉴于持续进行改进，Beckhoff 明确建议始终保持产品的最新状态，并在产品更新可用后马上进行安装。使用过时的或不支持的产品版本可能会增加网络威胁的风险。

如需了解 Beckhoff 产品信息安全的信息，请订阅 https://www.beckhoff.com/secinfo 上的 RSS 源。

2 概述

PLC 库 Tc2_Utilities 包含用于调用 TwinCAT 系统函数和操作系统函数以及各种转换函数的功能块和函数。

操作系统函数

名称	描述
NT_Shutdown [▶ 125]	关闭操作系统（关闭）。
NT_AbortShutdown [▶ 119]	中断关闭过程。
NT_Reboot [▶ 121]	重新启动操作系统。
NT_GetTime [▶ 120]	读取当前本地 Windows 系统时间。
NT_SetLocalTime [▶ 122]	设置当前本地 Windows 系统时间。
NT_StartProcess [▶ 126]	从 PLC 启动 Windows 应用程序。
NT_SetTimeToRTCTime [▶ 123]	将本地 Windows 系统时间与 PC 的实时时钟同步。
FB_RegQueryValue [▶ 101]	从注册表中读取值。
FB_RegSetValue [▶ 104]	将值写入注册表。
FB_EnumFindFileEntry [▶ 51]	在目录中搜索与指定名称相似的文件或子目录。可以单独读取找到的任何条目。
FB_EnumFindFileList [▶ 52]	在目录中搜索与指定名称相似的文件或子目录。可以分组读出找到的条目。
FB_GetAdaptersInfo [▶ 66]	读取网络适配器信息。
FB_GetHostName [▶ 71]	读取本地 PC 的主机名称。
FB_GetHostAddrByName [▶ 69]	将主机名称转换为（IPv4）互联网协议网络地址。
FB_GetTimeZoneInformation [▶ 78]	读取操作系统的时区配置。
FB_SetTimeZoneInformation [▶ 109]	设置操作系统的时区配置。
FB_LocalSystemTime [▶ 90]	返回本地 Windows 系统时间和夏令时/冬令时信息。

PLC 功能

名称	描述
PLC_Reset [▶ 131]	重置 PLC。
PLC_Start [▶ 132]	启动 PLC。
PLC_Stop [▶ 132]	停止 PLC。
PLC_ReadSymInfo [▶ 127]	读取 PLC 的符号信息。
PLC_ReadSymInfoByName [▶ 128]	通过符号名称读取 PLC 变量的符号信息。
PLC_ReadSymInfoByNameEx [▶ 130]	通过符号名称读取 PLC 变量的符号信息。如果可用缓冲区大小不足，则会截断注释。
Profiler [▶ 133]	确定 PLC 代码的执行时间。
WritePersistentData [▶ 146]	从 PLC 程序中将持久性数据保存到数据存储设备中。
FB_WritePersistentData [▶ 117]	从 PLC 程序中将持久性数据保存到数据存储设备中（扩展版）。

校验和/CRC 函数

名称	描述
F_CheckSum16 [▶ 283]	计算 16 位校验和。
F_DATA_TO_CRC16_CCITT [▶ 285]	计算任何数据类型的 CRC16-CCITT（循环冗余校验）。
F_BYTE_TO_CRC16_CCITT [▶ 282]	计算单个数据字节的 CRC16-CCITT（循环冗余校验）。

系统函数

名称	描述
TC_Restart [▶ 143]	重启 TwinCAT 系统。
TC_Stop [▶ 144]	停止 TwinCAT 系统。
TC_Config [▶ 138]	将 TwinCAT 系统切换到 CONFIG 模式。
TC_CpuUsage [▶ 142]	确定 TwinCAT 系统的 CPU 使用率。
TC_SysLatency [▶ 145]	确定 TwinCAT 系统的当前和最大延迟时间。
GetRemotePCInfo [▶ 118]	通过已配置的远程 PC 读取路由器信息。
FB_GetLocalAmsNetId [▶ 75]	读取本地 TwinCAT PC 的 AmsNetId。
FB_GetRouterStatusInfo [▶ 76]	读取 TwinCAT 路由器状态信息。
FB_EnumRouteEntry [▶ 54]	读取路由器连接信息。
FB_AddRouteEntry [▶ 40]	添加新的路由器连接。
FB_RemoveRouteEntry [▶ 106]	删除现有的路由器连接。
ROUTETRANSPORT_TO_STRING [▶ 310]	将 AMS 消息路由器传输层 ID 转换为字符串。
FB_GetDeviceIdentification [▶ 26]	读取设备 ID。
FB_GetDeviceIdentificationEx [▶ 68]	读取设备 ID。（允许使用更长字符串包含硬件型号和序列号。）
FB_GetLicences [▶ 72]	读取有效和无效的 TwinCAT 授权。
FB_GetSystemId [▶ 77]	将 SystemID 读取为 GUID。
FB_GetVolumeId [▶ 79]	读取系统 ID 和批量系统 ID。

Scope View 函数

TwinCAT 3 不支持 TwinCAT Scope View 功能。

Scope Server 函数

名称	描述
FB_ScopeServerControl [▶ 107]	控制（启动/保存...）用于数据记录的 Scope Server。

ADS Monitor 函数

名称	描述
FB_AmsLogger [▶ 43]	从 PLC 启动/停止 AMS 记录器。

转换函数

名称	描述
DT_TO_SYSTEMTIME [▶ 148]	将 DATE_AND_TIME 转换为 Windows 系统时间结构。
DT_TO_FILETIME [▶ 261]	将 DATE_AND_TIME 转换为 Windows 文件时间。
SYSTEMTIME_TO_DT [▶ 159]	将 Windows 系统时间结构转换为 DATE_AND_TIME。
SYSTEMTIME_TO_FILETIME [▶ 264]	将 Windows 系统时间结构转换为文件时间。
SYSTEMTIME_TO_STRING [▶ 161]	将 Windows 系统时间结构转换为字符串。
STRING_TO_SYSTEMTIME [▶ 159]	将字符串转换为 Windows 系统时间结构。
FILETIME_TO_DT [▶ 263]	将 Windows 文件时间转换为 DATE_AND_TIME。
FILETIME_TO_SYSTEMTIME [▶ 264]	将 Windows 文件时间转换为系统时间结构。
DEC_TO_BCD [▶ 38]	将十进制数转换为 BCD 数。
BCD_TO_DEC [▶ 31]	将 BCD 数转换为十进制数。
DEG_TO_RAD [▶ 276]	将度角转换为弧度。
RAD_TO_DEG [▶ 309]	将弧度转换为度角。
TIME_TO_OTSTRUCT [▶ 162]	将 TIME 变量转换为具有解析毫秒、秒钟、分钟等的结构体。
OTSTRUCT_TO_TIME [▶ 158]	将具有解析的毫秒、秒钟、分钟等的结构转换为 TIME 变量。
F_SwapRealEx [▶ 291]	使用 REAL 变量的高位字和低位字替换。
BYTE_TO_LREALEX [▶ 270]	允许将 BYTE 类型显式转换为 LREAL 类型的正浮点数。
DWORD_TO_LREALEX [▶ 280]	允许将 DWORD 类型显式转换为 LREAL 类型的正浮点数。
UDINT_TO_LREALEX [▶ 314]	允许将 UDINT 类型显式转换为 LREAL 类型的正浮点数。
UINT_TO_LREALEX [▶ 315]	允许将 UINT 类型显式转换为 LREAL 类型的正浮点数。
ULINT_TO_ULARGE [▶ 316]	将 ULINT 类型的 64 位数字转换为 T_ULARGE_INTEGER 类型的 64 位数字。
USINT_TO_LREALEX [▶ 316]	允许将 USINT 类型显式转换为 LREAL 类型的正浮点数。
BYTEARR_TO_MAXSTRING [▶ 272]	将字节数组转换为字符串。
MAXSTRING_TO_BYTEARR [▶ 304]	将字符串转换为字节数组。
F_TranslateFileTimeBias [▶ 262]	将 UTC 时间转换为本地时间，以及将本地时间转换为 UTC 时间（按偏差）。
FB_TzSpecificLocalTimeToFileTime [▶ 29]	将连续的本地时间（文件时间格式）转换为 UTC 时间。
FB_TzSpecificLocalTimeToSystemTime [▶ 115]	将连续的本地时间（结构化系统时间格式）转换为 UTC 时间。
FB_FileTimeToTzSpecificLocalTime [▶ 27]	将 UTC 时间（文件时间格式）转换为本地时间。
FB_SystemTimeToTzSpecificLocalTime [▶ 112]	将 UTC 时间（结构化系统时间格式）转换为本地时间。
HEXASCNIBBLE_TO_BYTE [▶ 295]	将十六进制半字节字符的 ASCII 代码转换为十进制值。
HEXCHRNIBBLE_TO_BYTE [▶ 295]	将十六进制半字节字符转换为其十进制值。
GuidsEqualByVal [▶ 294]	比较 2 个 GUID 值

STRING 函数

名称	描述
LREAL_TO_FMTSTR [▶ 298]	将流点数转换为具有所需小数位数的字符串。
DWORD_TO_DECSTR [▶ 278]	将十进制数转换为十进制字符串。
DWORD_TO_HEXSTR [▶ 279]	将十进制数转换为十六进制字符串。
DWORD_TO_OCTSTR [▶ 281]	将十进制数转换为八进制字符串。
DWORD_TO_BINSTR [▶ 277]	将十进制数转换为二进制字符串。
LWORD_TO_DECSTR [▶ 301]	将十进制数转换为十进制字符串。
LWORD_TO_HEXSTR [▶ 302]	将十进制数转换为十六进制字符串。
LWORD_TO_OCTSTR [▶ 303]	将十进制数转换为八进制字符串。
LWORD_TO_BINSTR [▶ 300]	将十进制数转换为二进制字符串。
PVOID_TO_DECSTR [▶ 305]	将地址（指针）转换为十进制字符串。
PVOID_TO_HEXSTR [▶ 306]	将地址（指针）转换为十六进制字符串。
PVOID_TO_OCTSTR [▶ 308]	将地址（指针）转换为八进制字符串。
PVOID_TO_BINSTR [▶ 304]	将地址（指针）转换为二进制字符串。
PVOID_TO_STRING [▶ 309]	将地址（指针）转换为字符串。
STRING_TO_PVOID [▶ 313]	将字符串转换为地址（指针）。
LINT_TO_DECSTR [▶ 297]	将有符号十进制数（64 位）转换为十进制字符串。
DINT_TO_DECSTR [▶ 276]	将有符号十进制数（32 位）转换为十进制字符串。

名称	描述
F_FormatArgToStr [▶ 286]	将十进制数或浮点数转换并格式化为字符串。
BYTE_TO_BINSTR [▶ 269]	将字节类型的十进制数转换为二进制字符串。
BYTE_TO_DECSTR [▶ 269]	将十进制数转换为十进制字符串。
BYTE_TO_HEXSTR [▶ 270]	将十进制数转换为十六进制字符串。
BYTE_TO_OCTSTR [▶ 271]	将十进制数转换为八进制字符串。
WORD_TO_BINSTR [▶ 317]	将字类型的十进制数转换为二进制字符串。
WORD_TO_DECSTR [▶ 318]	将字类型的十进制数转换为十进制字符串。
WORD_TO_HEXSTR [▶ 319]	将字类型的十进制数转换为十六进制字符串。
WORD_TO_OCTSTR [▶ 320]	将字类型的十进制数转换为八进制字符串。
FB_FormatString [▶ 63]	最多转换 10 个参数（十进制数或流点数）并进行格式化。
FB_EnumStringNumbers [▶ 56]	在字符串中搜索数字。
F_ToUCase [▶ 292]	将字符串中的小写字母转换为大写字母。
F_ToLCase [▶ 291]	将字符串中的大写字母转换为小写字母。
F_LTrim [▶ 289]	移除字符串开头的空格。
F_RTrim [▶ 289]	移除字符串末尾的空格。
DATA_TO_HEXSTR [▶ 275]	将二进制数据转换为十六进制字符串。
HEXSTR_TO_DATA [▶ 296]	将十六进制字符串转换为二进制数据。
GUID_TO_STRING [▶ 294]	将结构化 GUID 变量转换为 GUID 字符串变量。
GUID_TO_REGSTRING [▶ 293]	将结构化 GUID 变量转换为注册表 GUID 字符串变量
REGSTRING_TO_GUID [▶ 310]	将注册表 GUID 字符串变量转换为结构化 GUID 变量。

扩展 STRING 函数

名称	描述
CHAR_TO_WCHAR [▶ 163]	将 STRING 数据类型的字符转换为 WSTRING 数据类型的字符（以空字符结尾）。
CONCAT2 [▶ 163]	连接 2 个 STRING 数据类型的字符串。
DELETE2 [▶ 165]	从 nPos 位置开始，移除字符串中的 nLen 字符。
F_StringIsASCII [▶ 166]	检查字符串是否只包含 ASCII 字符（0x000 至 0x7F），并返回 ASCII 字符数。
FIND2 [▶ 166]	在另一个字符串中查找一个可能出现多次的字符串。
FindAndDelete [▶ 167]	在另一个字符串中查找一个可能出现多次的字符串，并将其移除。
FindAndDeleteChar [▶ 168]	在一个字符串中查找一个可能出现多次的字符，并将其移除。
FindAndReplace [▶ 168]	在另一个字符串中查找一个可能出现多次的字符串，并使用另一个字符串替换它。
FindAndReplaceChar [▶ 169]	在一个字符串中查找一个可能出现多次的字符，并使用另一个字符替换它。
INSERT2 [▶ 174]	在 nPos 位置后，将一个字符串插入另一个字符串。
LEN2 [▶ 175]	返回字符串中的字符数。
REPLACE2 [▶ 175]	从 nPos 位置开始，使用另一个字符串替换一个字符串的 nLen 字符。
sLiteral_TO_UTF8 [▶ 176]	将 STRING 数据类型的任何字符串转换为 UTF-8 格式的字符串。
STRING_TO_UTF8 [▶ 177]	将 STRING 数据类型的变量的任何字符串转换为 UTF-8 格式的字符串。
STRING_TO_WSTRING2 [▶ 178]	将 STRING 数据类型的变量转换为 WSTRING 数据类型的变量。
STRNCPY [▶ 178]	复制 STRING 数据类型的变量的字符串，并检查是否完全复制该字符串
UTF8_TO_STRING [▶ 179]	将 UTF-8 格式的字符串转换为 STRING 数据类型的字符串。
UTF8_TO_WSTRING [▶ 180]	将 UTF-8 格式的字符串转换为 WSTRING 数据类型的字符串。
UTF8Len [▶ 181]	返回 UTF-8 字符串中的字符数。
WCHAR_TO_CHAR [▶ 182]	将 WSTRING 数据类型的变量转换为 STRING 数据类型的变量（以空字符结尾）
WCONCAT2 [▶ 182]	连接 2 个任何长度的 WSTRING 数据类型的字符串。
WLEN2 [▶ 183]	返回 数据类型为WSTRING 的 Unicode 字符串中的字符数。
wsLiteral_TO_UTF8 [▶ 184]	将 WSTRING 数据类型的字符串转换为 UTF-8 格式的字符串。
WSTRING_TO_STRING2 [▶ 185]	将 WSTRING 数据类型的变量转换为 STRING 数据类型的变量。

名称	描述
WSTRING_TO_UTF8 [▶ 185]	将 WSTRING 数据类型的变量的字符串转换为 UTF-8 格式的字符串。
WSTRNCPY [▶ 186]	复制 WSTRING 数据类型的变量的字符串，并检查是否完全复制该字符串。

64 位函数（无符号）

名称	描述
ULARGE_INTEGER [▶ 245]	初始化/设置一个 64 位数字。
UInt64Add64 [▶ 232]	将 2 个 64 位数字相加。
UInt64Add64Ex [▶ 233]	将 2 个 64 位数字相加（带溢出检查）。
UInt64Sub64 [▶ 243]	将 2 个 64 位数字相减。
UInt64Cmp64 [▶ 234]	比较 2 个 64 位数字。
UInt32x32To64 [▶ 231]	将 2 个 32 位数字相乘。结果是一个 64 位数字。
UInt64Mul64 [▶ 239]	将 2 个 64 位数字相乘。结果是一个 64 位数字。
UInt64Mul64Ex [▶ 240]	将 2 个 64 位数字相乘。结果是一个 64 位数字（带溢出检查）。
UInt64Div64 [▶ 235]	2 个 64 位数字的除法。
UInt64Div64Ex [▶ 236]	2 个 64 位数字的除法（带余数）。
Uint64Div16Ex [▶ 235]	将一个 64 位数字除以一个 16 位数字。结果是一个 64 位数字。
UInt64Mod64 [▶ 238]	2 个 64 位数字的模除。
UInt64And [▶ 233]	2 个 64 位数字的逐位 AND 运算。
UInt64Or [▶ 241]	2 个 64 位数字的逐位 OR 运算。
UInt64Not [▶ 240]	1 个 64 位数字的逐位 NOT 运算。
UInt64Xor [▶ 244]	2 个 64 位数字的逐位 XOR 运算。
UInt64Rol [▶ 241]	1 个 64 位数字的逐位左旋转。
UInt64Ror [▶ 242]	1 个 64 位数字的逐位右旋转。
UInt64Shl [▶ 242]	1 个 64 位数字的逐位左移。
UInt64Shr [▶ 243]	1 个 64 位数字的逐位右移。
UInt64Min [▶ 238]	最小函数
UInt64Max [▶ 237]	最大函数
UInt64Limit [▶ 237]	限值
UInt64isZero [▶ 236]	检查 64 位数字的值是否为零。
UINT64_TO_STRING [▶ 232]	将 64 位数字转换为字符串。
UINT64_TO_LREAL [▶ 231]	将 64 位数字转换为 LREAL。
STRING_TO_UINT64 [▶ 230]	将 STRING 转换为 64 位数字。
LREAL_TO_UINT64 [▶ 229]	将 LREAL 转换为 64 位数字。
LWORD_TO_ULARGE [▶ 230]	将 LWORD 类型的 64 位数字转换为 T_ULARGE_INTEGER 类型的 64 位数字。
ULARGE_TO_LWORD [▶ 246]	将 T_ULARGE_INTEGER 类型的 64 位数字转换为 LWORD 类型的 64 位数字。
ULARGE_TO_ULINT [▶ 245]	将 T_ULARGE_INTEGER 类型的 64 位数字转换为 ULINT 类型的 64 位数字。

64 位函数（有符号）

名称	描述
LARGE_INTEGER [▶ 226]	初始化/设置一个 64 位数字。
Int64Add64 [▶ 222]	将 2 个 64 位数字相加。
Int64Add64Ex [▶ 222]	将 2 个 64 位数字相加（带溢出检查）。
Int64Sub64 [▶ 226]	将 2 个 64 位数字相减。
Int64Cmp64 [▶ 223]	比较 2 个 64 位数字。
Int64Div64Ex [▶ 224]	2 个 64 位数字的除法（带余数）。
Int64Not [▶ 225]	1 个 64 位数字的逐位 NOT（取反） 运算。
Int64isZero [▶ 224]	检查 64 位数字的值是否为零。
Int64Negate [▶ 225]	对一个 64 位数字取反。
INT64_TO_LREAL [▶ 221]	将 64 位数字转换为 LREAL。
LREAL_TO_INT64 [▶ 228]	将 LREAL 转换为 64 位数字。

名称	描述
LARGE_TO_ULARGE [▶ 227]	将有符号 64 位数字转换为无符号 64 位数字。
ULARGE_TO_LARGE [▶ 229]	将无符号 64 位数字转换为有符号 64 位数字。
LARGE_TO_LINT [▶ 227]	将 LINT 类型的有符号 64 位数字转换为 T_LARGE_INTEGER 类型的有符号 64 位数字。
LINT_TO_LARGE [▶ 228]	将 T_LARGE_INTEGER 类型的有符号 64 位数字转换为 LINT 类型的有符号 64 位数字。

16 位定点数函数（有符号）

名称	描述
FIX16Add [▶ 216]	将 2 个定点数相加。
FIX16Align [▶ 217]	改变定点数的分辨率。
FIX16Sub [▶ 219]	将 2 个定点数相减。
FIX16Div [▶ 218]	将 2 个定点数相除。
FIX16Mul [▶ 218]	将 2 个定点数相乘。
LREAL_TO_FIX16 [▶ 220]	将 LREAL 转换为定点数。
WORD_TO_FIX16 [▶ 221]	将 WORD 转换为定点数。
FIX16_TO_LREAL [▶ 215]	将定点数转换为 LREAL。
FIX16_TO_WORD [▶ 216]	将定点数转换为 WORD。

字节顺序转换函数

名称	描述
HOST_TO_BE16 [▶ 187]	主机到网络转换（16 位数字）
HOST_TO_BE32 [▶ 188]	主机到网络转换（32 位数字）
HOST_TO_BE64 [▶ 188]	主机到网络转换（64 位数字，“传统”类型：T_ULARGE_INTEGER）
HOST_TO_BE64EX [▶ 189]	主机到网络转换（64 位数字，“本地”类型：LWORD）
HOST_TO_BE128 [▶ 189]	主机到网络转换（128 位数字，“传统”类型：T_UHUGE_INTEGER）
BE16_TO_HOST [▶ 190]	主机到网络转换（16 位数字）
BE32_TO_HOST [▶ 190]	网络到主机转换（32 位数字）
BE64_TO_HOST [▶ 191]	网络到主机转换（64 位数字，“传统”类型：T_ULARGE_INTEGER）
BE64_TO_HOSTEX [▶ 191]	网络到主机转换（64 位数字，“本地”类型：LWORD）
BE128_TO_HOST [▶ 192]	网络到主机转换（128 位数字，“传统”类型：T_UHUGE_INTEGER）

FLOAT 函数

名称	描述
BOOL_TO_FLOAT [▶ 193]	将 BOOL 类型的变量转换为 LREAL 类型的变量。
DINT_TO_FLOAT [▶ 193]	将 DINT 类型的变量转换为 FLOAT 类型的变量。
FLOAT_TO_BOOL [▶ 194]	将 FLOAT 类型的变量转换为 BOOL 类型的变量。
FLOAT_TO_DINT [▶ 194]	将 FLOAT 类型的变量转换为 DINT 类型的变量。
FLOAT_TO_INT [▶ 194]	将 FLOAT 类型的变量转换为 INT 类型的变量。
FLOAT_TO_SINT [▶ 195]	将 FLOAT 类型的变量转换为 SINT 类型的变量。
FLOAT_TO_STRING [▶ 195]	将 FLOAT 类型的变量转换为 STRING 类型的变量。
FLOAT_TO_TIME [▶ 196]	将 FLOAT 类型的变量转换为 TIME 类型的变量。
FLOAT_TO_UDINT [▶ 196]	将 FLOAT 类型的变量转换为 UDINT 类型的变量。
FLOAT_TO_UINT [▶ 197]	将 FLOAT 类型的变量转换为 UINT 类型的变量。
INT_TO_FLOAT [▶ 197]	将 INT 类型的变量转换为 FLOAT 类型的变量。
SINT_TO_FLOAT [▶ 198]	将 SINT 类型的变量转换为 FLOAT 类型的变量。
TIME_TO_FLOAT [▶ 198]	将 TIME 类型的变量转换为 FLOAT 类型的变量。
UDINT_TO_FLOAT [▶ 199]	将 UDINT 类型的变量转换为 FLOAT 类型的变量。
UINT_TO_FLOAT [▶ 199]	将 UINT 类型的变量转换为 FLOAT 类型的变量。
LrealIsFinite [▶ 200]	当 LREAL 类型的参数有一个有限值时，返回 TRUE。

名称	描述
LrealIsNaN [▶ 200]	当 LREAL 类型的参数有一个未定义值（NaN）时，返回 TRUE。
RealIsFinite [▶ 201]	当 REAL 类型的参数有一个有限值时，返回 TRUE。
RealIsNaN [▶ 201]	当 REAL 类型的参数有一个未定义值（NaN）时，返回 TRUE。

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

LCOMPLEX 函数

名称	描述
LcomplexIsNaN [▶ 202]	当 LCOMPLEX 类型的参数有一个未定义值（NaN）时，返回 TRUE。
LcomplexAbs [▶ 202]	返回传输的复数的绝对值。

P[TYPE]_TO_[TYPE] 转换函数

名称	描述
PBOOL_TO_BOOL [▶ 203]	返回 BOOL 指针变量的内容。
PBYTE_TO_BYTE [▶ 203]	返回 BYTE 指针变量的内容。
PDATE_TO_DATE [▶ 204]	返回 DATE 指针变量的内容。
PDINT_TO_DINT [▶ 204]	返回 DINT 指针变量的内容。
PDT_TO_TO_DT [▶ 205]	返回 DT 指针变量的内容。
PDWORD_TO_DWORD [▶ 205]	返回 DWORD 指针变量的内容。
PHUGE_TO_HUGE [▶ 206]	返回 T_HUGE_INTEGER 指针变量的内容。
PINT_TO_INT [▶ 206]	返回 INT 指针变量的内容。
PLARGE_TO_LARGE [▶ 207]	返回 T_LARGE_INTEGER 指针变量的内容。
PLINT_TO_LINT [▶ 207]	返回 LINT 指针变量的内容。
PLREAL_TO_TO_LREAL [▶ 208]	返回 LREAL 指针变量的内容。
PLWORD_TO_LWORD [▶ 208]	返回 LWORD 指针变量的内容。
PMAXSTRING_TO_MAXSTRING [▶ 208]	返回 T_MaxString 指针变量的内容。
PREAL_TO_REAL [▶ 209]	返回 REAL 指针变量的内容。
PSINT_TO_SINT [▶ 209]	返回 SINT 指针变量的内容。
PSTRING_TO_STRING [▶ 210]	返回 STRING 指针变量的内容。
PTIME_TO_TIME [▶ 210]	返回 TIME 指针变量的内容。
PTOD_TO_TOD [▶ 211]	返回 TOD 指针变量的内容。
PUDINT_TO_UDINT [▶ 211]	返回 UDINT 指针变量的内容。
PUHUGE_TO_UHUGE [▶ 212]	返回 T_UHUGE_INTEGER 指针变量的内容。
PUINT_TO_UINT [▶ 212]	返回 UINT 指针变量的内容。
PULARGE_TO_ULARGE [▶ 213]	返回 T_ULARGE_INTEGER 指针变量的内容。
PULINT_TO_ULINT [▶ 213]	返回 ULINT 指针变量的内容。
PUSINT_TO_USINT [▶ 214]	返回 USINT 指针变量的内容。
PWORD_TO_WORD [▶ 214]	返回 WORD 指针变量的内容。
PUINT64_TO_UINT64 [▶ 214]	返回 T_ULARGE_INTEGER 指针变量的内容。

T_Arg 辅助函数

名称	描述
F_ARGCMP [▶ 246]	比较 2 个 T_Arg 类型的变量
F_ARGCPY [▶ 247]	将 T_Arg 类型变量的值复制到另一个变量，并返回成功复制的数据字节数。
F_ARGISZERO [▶ 248]	如果其中一个 T_Arg 成员变量的值为零或未初始化，则返回 TRUE。
F_BIGTYPE [▶ 248]	返回有关 T_Arg 类型结构中的结构或数组变量的信息。
F_BOOL [▶ 249]	返回有关 T_Arg 类型结构中的 BOOL 变量的信息。
F_BYTE [▶ 249]	返回有关 T_Arg 类型结构中的 BYTE 变量的信息。

名称	描述
F_DINT [▶ 250]	返回有关 T_Arg 类型结构中的 DINT 变量的信息。
F_DWORD [▶ 250]	返回有关 T_Arg 类型结构中的 DWORD 变量的信息。
F_HUGE [▶ 251]	返回有关 T_Arg 类型结构中的 T_HUGE_INTEGER 变量的信息。
F_INT [▶ 251]	返回有关 T_Arg 类型结构中的 INT 变量的信息。
F_LARGE [▶ 252]	返回有关 T_Arg 类型结构中的 T_LARGE_INTEGER 变量的信息。
F_LINT [▶ 252]	返回有关 T_Arg 类型结构中的 LINT 变量的信息。
F_LREAL [▶ 253]	返回有关 T_Arg 类型结构中的 LREAL 变量的信息。
F_LWORD [▶ 253]	返回有关 T_Arg 类型结构中的 LWORD 变量的信息。
F_REAL [▶ 254]	返回有关 T_Arg 类型结构中的 REAL 变量的信息。
F_SINT [▶ 254]	返回有关 T_Arg 类型结构中的 SINT 变量的信息。
F_STRING [▶ 255]	返回有关 T_Arg 类型结构中的 T_MaxString 变量的信息。
F_UDINT [▶ 256]	返回有关 T_Arg 类型结构中的 UDINT 变量的信息。
F_UHUGE [▶ 256]	返回有关 T_Arg 类型结构中的 T_UHUGE_INTEGER 变量的信息。
F_UINT [▶ 257]	返回有关 T_Arg 类型结构中的 UINT 变量的信息。
F_ULARGE [▶ 257]	返回有关 T_Arg 类型结构中的 T_ULARGE_INTEGER 变量的信息。
F_ULINT [▶ 258]	返回有关 T_Arg 类型结构中的 ULINT 变量的信息。
F_USINT [▶ 258]	返回有关 T_Arg 类型结构中的 USINT 变量的信息。
F_WORD [▶ 259]	返回有关 T_Arg 类型结构中的 WORD 变量的信息。
F_PVOID [▶ 259]	返回有关 T_Arg 类型结构中的 PVOID 变量的信息。

CSV 格式函数

名称	描述
CSVFIELD_TO_STRING [▶ 273]	将带有 CSV 格式数据字段的字符串的值转换为 PLC 字符串变量。
STRING_TO_CSVFIELD [▶ 311]	将 PLC 字符串变量的值转换为带有 CSV 格式数据字段的字符串。
CSVFIELD_TO_ARG [▶ 272]	将带有 CSV 格式数据字段的字节缓冲区转换为随机 PLC 变量的值。
ARG_TO_CSVFIELD [▶ 266]	将随机 PLC 变量的值转换为带有 CSV 格式数据字段的字节缓冲区。
FB_CSVMemBufferReader [▶ 48]	将字节缓冲区中的 CSV 格式的数据集拆分成单个数据字段。
FB_CSVMemBufferWriter [▶ 50]	从各个单独的数据字段生成单个或多个数据集，并将其存储在字节缓冲区中

授权函数

名称	描述
FB_LicFileGetStorageInfo [▶ 86]	读取授权加密狗的存储信息和文件目录。
FB_LicFileCreate [▶ 84]	在授权加密狗上创建文件。
FB_LicFileDelete [▶ 85]	从授权加密狗上删除文件。
FB_LicFileRead [▶ 87]	将文件从授权加密狗读取到缓冲区。
FB_LicFileCopyToDongle [▶ 83]	将文件从硬盘复制到授权加密狗。
FB_LicFileCopyFromDongle [▶ 81]	将文件从授权加密狗复制到硬盘。
FB_CheckLicense [▶ 47]	确定给定授权 ID 的 TwinCAT 3 授权状态。
FB_GetDongleSystemId [▶ 68]	将 TwinCAT 3 授权加密狗的系统 ID 和批量 ID 读取为 GUID。
FB_GetLicenseDongle [▶ 71]	确定连接的授权加密狗的数量，并返回地址和状态。
FB_GetLicenses [▶ 72]	读取有效和无效的 TwinCAT 授权。
FB_GetLicensesEx [▶ 73]	确定所有 TwinCAT 3 授权和 OEM 授权的状态。

其他函数

名称	描述
FB_BasicPID [▶ 44]	简单 PID 控制器
F_GetVersionTcUtilities [▶ 265]	读取库的版本信息。
IsFinite [▶ 260]	按照 IEEE 验证浮点数的格式。
F_YearIsLeapYear [▶ 155]	确定某一年份是否为闰年。
F_GetMaxMonthDays [▶ 152]	确定一个月中的最大天数。
F_GetDOYOfYearMonthDay [▶ 151]	确定一年中的日期数。
F_GetMonthOfDOY [▶ 153]	根据一年中的天数确定月份。

名称	描述
F_GetDayOfWeek [▶ 151]	确定星期几。
F_GetWeekOfTheYear [▶ 154]	确定日历周。
F_GetDayOfMonthEx [▶ 150]	确定指定月份和年份中的第一个、第二个……星期几的日期。
F_GetWeekOfTheYear [▶ 154]	返回预定义日期的日历周数。
RTC [▶ 135]	“软件”–RTC（实时时钟）
RTC_EX [▶ 136]	“软件”–RTC（实时时钟）
RTC_EX2 [▶ 137]	“软件”–RTC（实时时钟）
FB_FileRingBuffer [▶ 59]	将数据集写入文件或从文件读取数据集（FIFO）。
FB_MemRingBuffer [▶ 96]	将数据集写入缓冲区变量或从缓冲区变量读取数据集（FIFO）。
FB_MemRingBufferEx [▶ 97]	将数据集写入缓冲区变量或从缓冲区变量读取数据集（FIFO）。
FB_StringRingBuffer [▶ 111]	将字符串写入缓冲区变量或从缓冲区变量读取字符串（FIFO）。
FB_MemStackBuffer [▶ 99]	将数据集写入缓冲区变量或从缓冲区变量读取数据集（LIFO）。
FB_MemBufferMerge [▶ 93]	将单个小数据段合并为一个较大的数据段。
FB_MemBufferSplit [▶ 95]	将内存区域（数据缓冲区）拆分成多个较小的数据段。
FB_HashTableCtrl [▶ 80]，F_CreateHashTableHnd [▶ 283]	简单哈希表。
FB_LinkedListCtrl [▶ 88]，F_CreateLinkedListHnd [▶ 284]	简单链表（双向链接）。
DCF77_TIME [▶ 32]	简单的 DCF77 解码器。
DCF77_TIME_EX [▶ 35]	DCF77 解码器，可对 2 个连续报文和时区信息进行可信度检查。

3 功能块

3.1 [已过时]

3.1.1 FB_GetDeviceIdentification

该块读取设备 ID。

过时的功能

对于较长的硬件型号和硬件序列号字符串，必须使用 FB_GetDeviceIdentificationEx [▶ 68] 块。

输入

```
VAR_INPUT
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId     : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetId	该参数可以用于指定要读取其设备 ID 的 TwinCAT 计算机的 AmsNetID（类型：T_AmsNetID）。可以为本地计算机指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy      : BOOL;
    bError     : BOOL;
    nErrorId   : UDINT;
    stDevIdent : ST_DeviceIdentification;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	如果在命令传输过程中出现错误，则在 bBusy 输出被重置后，将会设置该输出。
nErrorId	UDINT	在设置 bError 输出时，返回 ADS 错误编号 [▶ 376]。
stDevIdent	ST_DeviceIdentification	返回设备标识（类型：ST_DeviceIdentification [▶ 334]）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.1.2 FB_FileTimeToTzSpecificLocalTime

过时的功能

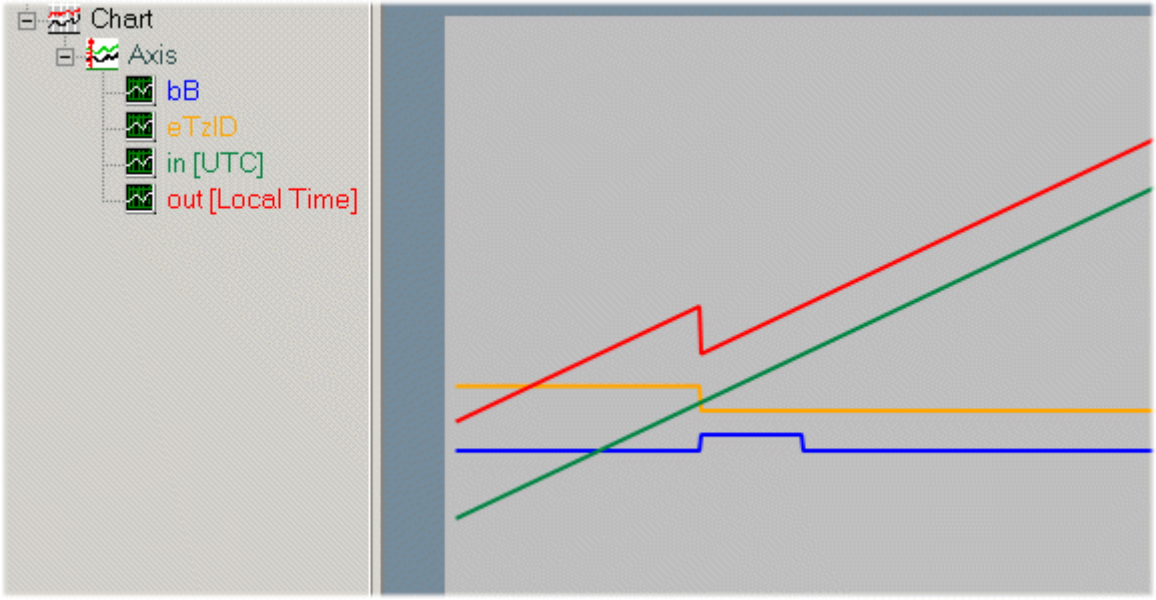
该功能块已过时。使用功能块 FB_FileTime64ToTzSpecificLocalTime [▶ 61] 代替。

该功能块将 UTC 时间（文件时间格式）转换为本地时间（文件时间格式），同时考虑指定的时区信息。功能块：FB_SystemTimeToTzSpecificLocalTime [▶ 112] 具有类似的功能，不同之处在于它可以转换为不同的时间格式（结构化系统时间格式）。

该功能块仅适用于转换**连续的** UTC 时间戳信息。该功能块使用时区信息来计算在本地时间中所需的时间步长（夏令时/冬令时转换）。不允许使用在 UTC 输入时间中的时间步长，否则会导致转换错误。原因：该功能块在内部存储最后转换的时间，因此，当本地时间发生变化时，它可以从 UTC 输入时间和存储值中检测出 B 时间（见下文）。

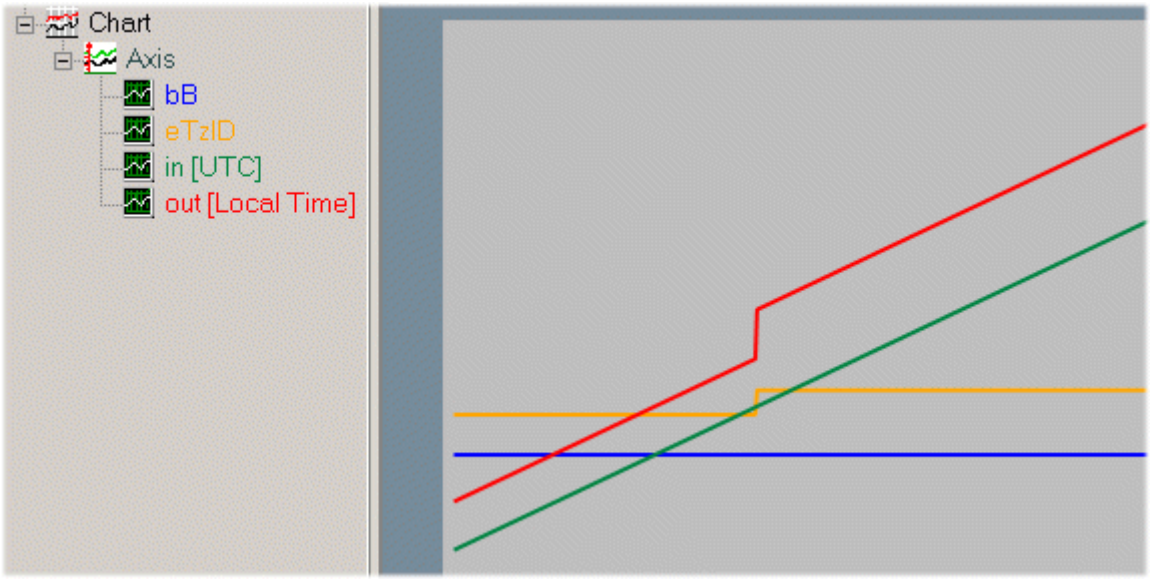
该功能块与A_Reset()相关联。如果调用该操作，则功能块输出和本地存储（最后转换）的时间将被重置为零。

1. 从夏令时到冬令时转换的图示（tzInfo = WEST_EUROPE_TZI）：



UTC 输入时间（绿色）是连续的。本地时间（红色）向后跳转。本地时间：**02h:59m:59s:999ms.**. 后面直接是：**02h:00m:00s:000ms.**. 2h 和 3h 之间的时间出现两次。例如，转换前的重复时间被称为 **02:05:00 CEST A**，转换后的时间被称为 **02:05:00 CET B**。输出变量 *bB* 表示它是第一次还是第二次*处理*。第二次*处理*时，输出变量 *bB*（蓝色）将被设置为 TRUE。在经过重复时间后，自动重置 *bB* 输出变量。时区 ID（橙色）从 *eTimeZoneID_Daylight*（夏令时）变为 *eTimeZoneID_Standard*（冬令时）。

2. 从冬令时到夏令时转换的图示（tzInfo = WEST_EUROPE_TZI）：



UTC 输入时间（绿色）是连续的。本地时间（绿色）向前跳转。本地时间：**2h:59m:59s:999ms..** 后面直接是：**3h:00m:00s:000ms..** 时区 ID（橙色）从 *eTimeZoneID_Standard*（冬令时）变为 *eTimeZoneID_Daylight*（夏令时）。

输入

```
VAR_INPUT
    in     : T_FILETIME;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
in	T_FILETIME	要转换的 UTC 时间（文件时间格式）（类型：T_FILETIME [▶ 344]）。
tzInfo	ST_TimeZoneInformation	带有操作系统的当前时区信息的结构变量（类型：ST_TimeZoneInformation [▶ 342]）。

输出

```
VAR_OUTPUT
    out   : T_FILETIME;
    eTzID : E_TimeZoneID := eTimeZoneID_Unknown;
    bB    : BOOL;
END_VAR
```

名称	类型	描述
out	T_FILETIME	转换后的本地时间（文件时间格式，类型：T_FILETIME [▶ 344]）。
eTzID	E_TimeZoneID	附加的夏令时/冬令时信息（类型：E_TimeZoneID [▶ 329]）。
bB	BOOL	TRUE => B 时间（例如：02:05:00 CET B），FALSE => 其他时间（例 如：02:05:00 CEST A）。当本地时间发生回跳时，该输出被置位，并在重复的本地时间通过后复位。

示例：

UTC 时间：DT#2011-09-02-09:01:31 被转换为本地时间。结果为：DT#2011-09-02-11:01:31。

```
PROGRAM MAIN
VAR
    in         : DT := DT#2011-09-02-09:01:31;(* UTC time *)
    out        : DT;(* Local time *)
    fbToLocal : FB_FileTimeToTzSpecificLocalTime;
END_VAR
```

```
fbToLocal( in := DT_TO_FILETIME( in ), tzInfo := WEST_EUROPE_TZI );
out := FILETIME_TO_DT( fbToLocal.out );
```

更多与时间和时区相关的功能及功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.1.3 FB_TzSpecificLocalTimeToFileTime

过时的功能

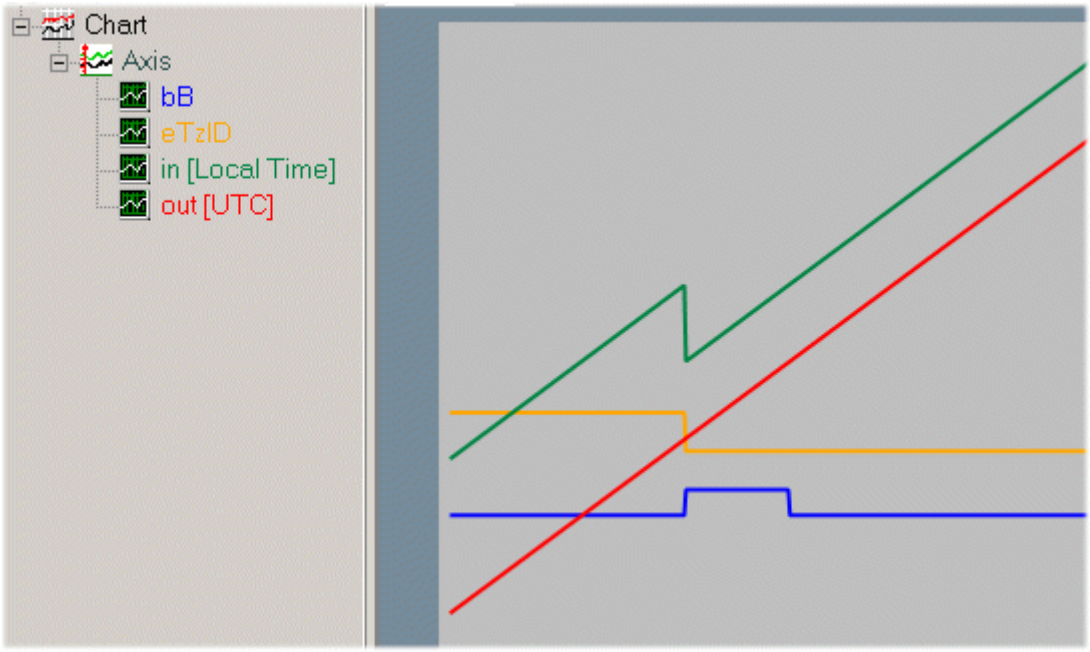
该功能块已过时。使用功能块 FB_TzSpecificLocalTimeToFileTime64 [▶ 113] 代替。

该功能块将本地时间（文件时间格式）转换为 UTC 时间（文件时间格式），同时考虑指定的时区信息。功能块：FB_TzSpecificLocalTimeToSystemTime [▶ 115] 具有类似的功能，不同之处在于它可以转换为不同的时间格式（结构化系统时间格式）。

该功能块仅适用于转换**连续的**本地时间戳信息。允许存在因夏令时/冬令时转换而引起的本地时间的阶跃变化，并且功能块可以进行正确检测。但是任意改变本地时间会导致转换错误。原因：在功能块内部存储最后转换的时间，以便在重置本地时间时能够识别夏令时/冬令时信息和 B 时间（见下文）。该功能块与A_Reset()相关联。如果调用该操作，则功能块输出和本地存储（最后转换）的时间将被重置为零。

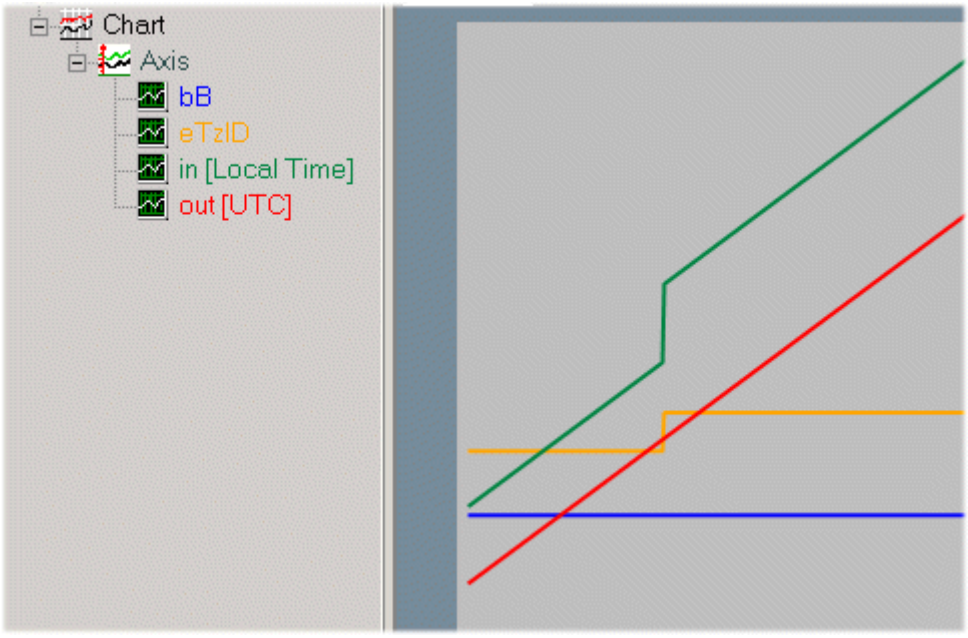
本地时间的阶跃变化有问题，因为它们必须转换为线性 UTC 时间。因此，建议使用（连续的）UTC 时间执行时间戳任务，并且仅在显示目的（例如在可视化中）时才将时间转换为相应的本地时间。

1. 从夏令时到冬令时转换的图示（tzInfo = WEST_EUROPE_TZI）：



本地时间（绿色）跳回。UTC 输出时间（红色）是连续的。本地时间：**02h:59m:59s:999ms..** 后面直接是：**02h:00m:00s:000ms..** 2h 和 3h 之间的时间出现两次。例如，转换前的重复时间被称为 **02:05:00 CEST A**，转换后的时间被称为 **02:05:00 CET B。**输出变量 *bB* 表示它是第一次还是第二次*处理*。第二次*处理*时，输出变量 *bB*（蓝色）将被设置为 TRUE。在经过重复时间后，自动重置 *bB* 输出变量。时区 ID（橙色）从 *eTimeZoneID_Daylight*（夏令时）变为 *eTimeZoneID_Standard*（冬令时）。

2. 从冬令时到夏令时转换的图示（tzInfo = WEST_EUROPE_TZI）：



本地时间（绿色）向前跳转。UTC 输出时间（红色）是连续的。本地时间：**2h:59m:59s:999ms..** 后面直接是：**3h:00m:00s:000ms..** 时区 ID（橙色）从 *eTimeZoneID_Standard*（冬令时）变为 *eTimeZoneID_Daylight*（夏令时）。

输入

```
VAR_INPUT
    in     : T_FILETIME;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
in	T_FILETIME	要转换的本地时间（文件时间格式）（类型：T_FILETIME [▶ 344]）。
tzInfo	ST_TimeZoneInformation	带有操作系统的当前时区信息的结构变量（类型：ST_TimeZoneInformation [▶ 342]）。

输出

```
VAR_OUTPUT
    out   : T_FILETIME;
    eTzID : E_TimeZoneID := eTimeZoneID_Unknown;
    bB    : BOOL;
END_VAR
```

名称	类型	描述
out	T_FILETIME	转换后的 UTC 时间（文件时间格式）（类型：T_FILETIME [▶ 344]）。
eTzID	E_TimeZoneID	附加的夏令时/冬令时信息（类型：E_TimeZoneID [▶ 329]）。
bB	BOOL	TRUE => B 时间（例如：02:05:00 CET B），FALSE => 其他时间（例如：02:05:00 CEST A）。当本地时间发生回跳时，该输出被置位，并在重复的本地时间通过后复位。

示例：

本地时间：DT#2011-09-02-11:01:31 被转换为 UTC 时间：DT#2011-09-02-09:01:31。

```
PROGRAM MAIN
VAR
    in      : DT := DT#2011-09-02-11:01:31;(* Local time *)
    out     : DT;(* UTC time *)
    fbToUTC : FB_TzSpecificLocalTimeToFileTime;
END_VAR
```

```
fbToUTC( in := DT_TO_FILETIME( in ), tzInfo := WEST_EUROPE_TZI );
out := FILETIME_TO_DT( fbToUTC.out );
```

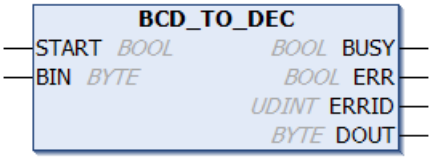
其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.2 BCD_TO_DEC



功能块“BCD_TO_DEC”用于将 二进制编码的十进制（BCD ）数字转换为十进制格式。检查要转换的 BCD 数值确认其有效性。

输入

```
VAR_INPUT
  START   : BOOL;
  BIN     : BYTE;
END_VAR
```

名称	类型	描述
START	BOOL	功能块由该输入端的上升沿触发激活。
BIN	BYTE	要转换的 BCD 数。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
    DOUT    : BYTE;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在转换过程开始时，该输出将被置位并保持置位状态，直至转换完成。一旦BUSY输出被复位，十进制数值即可在DOUT输出端获得。
ERR	BOOL	如果发生错误，则该变量将被设置为 TRUE。
ERRID	UDINT	错误代码
DOUT	BYTE	如果转换过程成功，则在该输出中可以获取十进制格式的转换变量。

错误代码：

错误代码	错误描述
0	无错误
0x000F	BCD 数字的低半字节中的值不可靠
0x00F0	BCD 数字的高半字节中的值不可靠

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.3 DCF77_TIME

该功能块由功能块 DCF77_TIME_EX [▶ 35] 取代。

“DCF77_TIME”功能块可以用于解码DCF-77 无线电时钟信号。当RUN 输入端出现上升沿时会启动解码过程，只要 RUN 输入保持有效状态，解码过程就会持续进行。在最坏的情况下，功能块最多需要 1 分钟完成自同步，另需 1 分钟解码下一分钟的时间数据。在此期间，系统会持续等待缺失的第 59 秒标志位。功能块内部正在对 DCF-77 信号进行采样。为了能够正确无误地对边沿进行采样，应在每个 PLC 周期中调用一次功能块。当循环时间 <= 25 ms 时，可以获得满意的结果。如果 DCF-77 信号缺失或异常，则 ERR 输出将被设置为 TRUE，并且在 ERRID 输出端生成对应错误代码。下一次接收正确信号时，ERR 和 ERRID 会自动复位。部分接收器设备输出的是反相 DCF-77 信号。在这种情况下，必须先对信号进行反相处理，然后才能传递到 DCF_PULSE 输入端。当操作无误时，CDT 输出端每分钟更新一次当前时间。因此，在第0秒时READY输出会在PLC的一个周期内被置为TRUE。此时，CDT 输出端的 DCF-77 时间有效，可供PLC 程序读取。只有在下一分钟的数据中未检测到错误时，READY输出才会被置位。传输的奇偶校验位可用于错误检测。如果接收条件较差，则无法保证 100% 无差错检测。即，如果有 2 个错误（反相）位，则功能块无法检测到错误，并且也会将 READY 输出设置为 TRUE。为了获确保时间信息的可靠性，必须采取额外的保障措施，例如，对连续几分钟的时间信息进行冗余分析。

在 DCF77_TIME 功能块中可实现对 2 个连续报文的简单可信度检查。通过全局布尔变量可以激活该功能，适用于 DCF77_TIME 功能块的所有实例。当激活可信度检查时，第一次同步将延长一分钟，最多不超过 3 分钟。

```
GLOBAL_DCF77_SEQUENCE_CHECK : BOOL := FALSE;
(* TRUE = Enable plausibility check (two telegrams are checked), FALSE = Disable check *)
```

在接收过程中发生的错误由功能块记录。ERRCNT 输出是一个错误计数器。该计数器表示自上次正确接收信号以来发生的错误数。下一次正确接收信号时，计数器将被重置。

时间代码

在每分钟内，以BCD格式编码的年、月、日、星期几、时和分的信息会通过秒脉冲的调制进行传输。传输的信息总是描述接下来的一分钟。每秒传输一个秒标记。持续时间为 0.1 s 的秒标记表示二进制0，而持续时间为 0.2 s 的秒标记表示二进制1。使用 3 个校验位扩展信息。在第 59 秒时，接收器可以利用这个间隔进行同步。

通过一个全局变量可以配置短脉冲信号和长脉冲信号的长度。如果信号不佳，则脉冲宽度会变小。接收器规格通常会规定 2 个逻辑信号的最小和最大脉冲信息，当场强度较高时，预期值较高，当场强度较低或受到干扰时，预期值较低。如果逻辑零的脉冲宽度过大，则在发送器附近（场强度非常大的地方）也可能会出现问题。因此，根据接收器规格设置了一个固定限值来区分零和一。**检查所用接收器的规格，并对脉冲长度进行相应配置。**

```
GLOBAL_DCF77_PULSE_SPLIT : TIME := T#140ms;(* 0 == pulse < 140ms, 1 == pulse > 140 *)
```

例如：在 Atmel T4227（时间代码接收器）的规格中提供了以下脉冲长度：
100 ms 脉冲（零）：最小值：70 ms，典型值：95 ms，最大值：**130 ms**
200 ms 脉冲（一）：最小值 **170 ms**，典型值 195 ms，最大值 235 ms
对于该 IC，150 ms 的限值为最佳值 = 130 + ((170 ms - 130 ms) / 2)。

提示：

如果配置的脉冲长度限值过小，则会将短脉冲检测为长脉冲。反之，如果配置的限值过大，则会将长脉冲检测为短脉冲。如果校验和正确，则接收器无法检测到这些错误。在第一种情况下，接收器可能会提供未来的时间，而在第二种情况下，时间可能是过去的时间。

输入

```
VAR_INPUT
    DCF_PULSE : BOOL;
    RUN       : BOOL;
END_VAR
```

名称	类型	描述
DCF_PULSE	BOOL	DCF-77 信号。
RUN	BOOL	该输入的上升沿会初始化功能块，并开始对 DCF-77 信号进行解码。如果该输入被重置，则解码过程将停止。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
    ERRCNT  : UDINT;
    READY   : BOOL;
    CDT     : DATE_AND_TIME;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在转换过程开始时，该输出将被置位并保持置位状态，直至转换完成。一旦BUSY输出被复位，十进制数值即可在DOUT输出端获得。
ERR	BOOL	如果发生错误，则该变量将被设置为 TRUE。
ERRID	UDINT	错误代码
ERRCNT	UDINT	自上次正确接收信号以来发生的错误数。
READY	BOOL	如果已设置该输出，则 CDT 输出处的数据有效。
CDT	DATE_AND_TIME	以 DATE_AND_TIME 格式表示的 DCF-77 时间

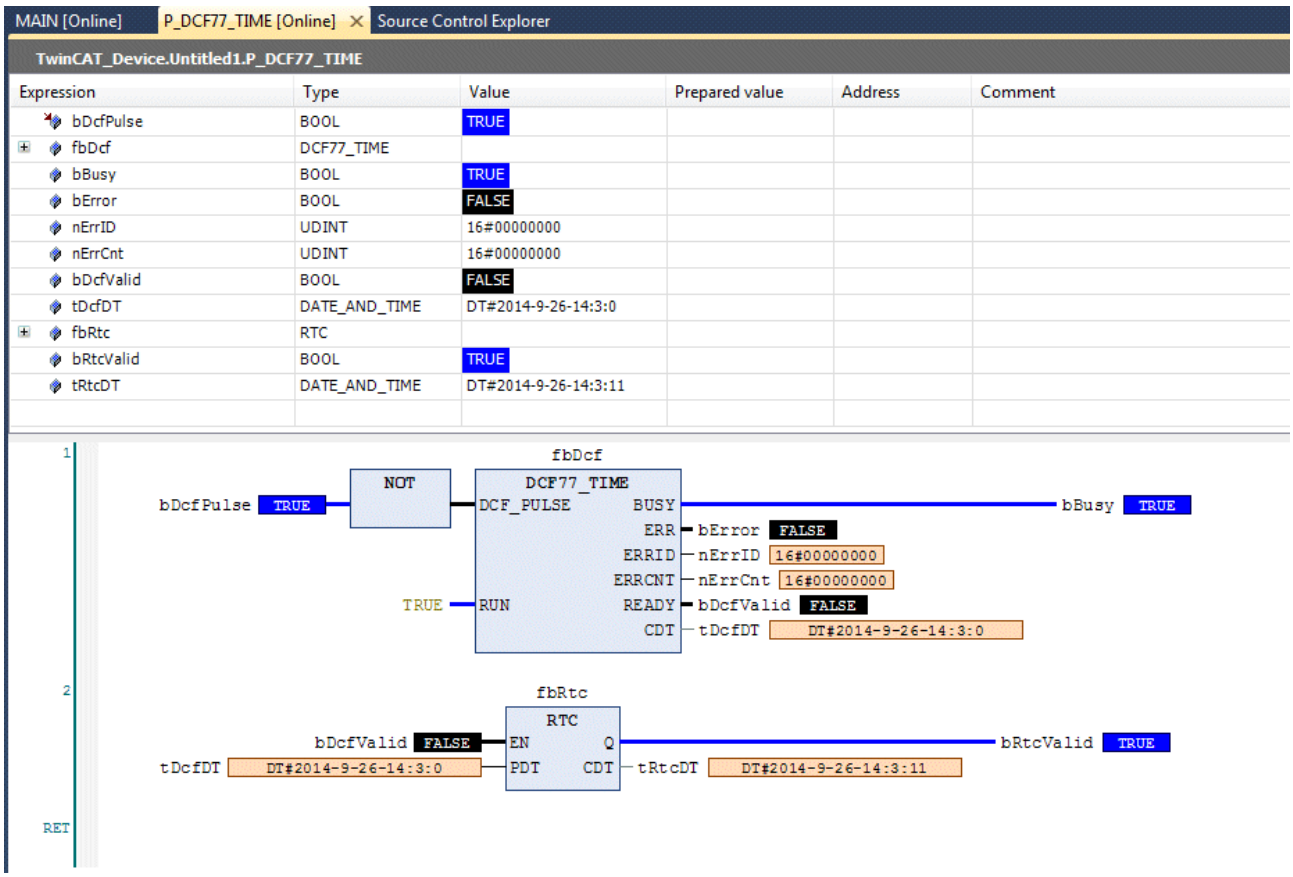
错误代码	错误描述
0	无错误
0x100	超时错误。可能是因为没有检测到 DCF-77 信号。
0x200	奇偶校验错误。在接收的数据中检测到错误位。
0x300	接收错误数据。由于奇偶校验仅可检测到一个错误位，因此要再次检查接收数据的有效性（例如，如果月 = 13，则会生成该错误代码）。
0x400	最后一个解码周期过长。当接收效果不佳时（接收到的秒脉冲不足），可能会发生该错误。
0x500	最后一个解码周期过短。当接收效果不佳时（接收到额外的边沿信号），可能会发生该错误。

示例：

在示例应用程序中，如果接收的数据无差错，则 TwinCAT 软件时钟（RTC）将与无线电时间同步。

```
PROGRAM P_DCF77_TIME
VAR
    bDcfPulse    : BOOL;
    fbDcf        : DCF77_TIME;
    bBusy        : BOOL;
    bError       : BOOL;
    nErrID       : UDINT;
    nErrCnt      : UDINT;
    bDcfValid    : BOOL;
    tDcfDT       : DT;
    fbRtc        : RTC;
    bRtcValid    : BOOL;
    tRtcDT       : DT;
END_VAR
```

在线视图：

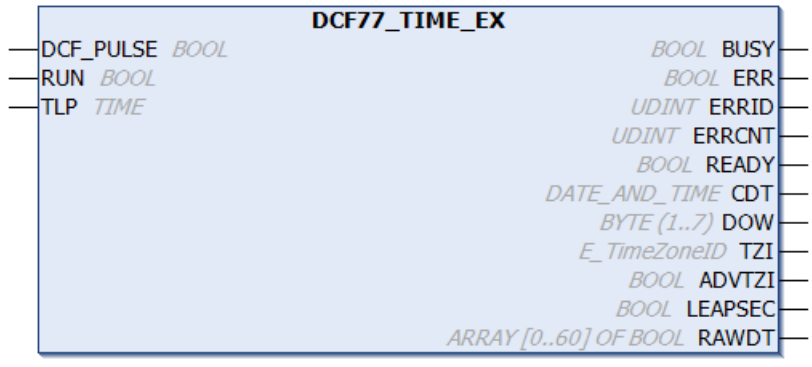


另请参见 DCF77_TIME_EX [▶ 35] 功能块的相关描述。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.4 DCF77_TIME_EX



功能块“DCF77_TIME_EX”用于对 DCF-77 无线电时钟信号进行解码。与“DCF77_TIME [▶ 32]”功能块相比，该块可按照标准检查 2 个连续报文的可信度。

当RUN 输入端出现上升沿时会启动解码过程，只要 RUN 输入保持有效状态，解码过程就会持续进行。在最坏的情况下，功能块的同步最多需要 1 分钟，并且需要另外 2 分钟来解码下一分钟的数据。在此期间，系统会持续等待缺失的第 59 秒标志位。功能块内部正在对 DCF-77 信号进行采样。为了能够正确无误地对边沿进行采样，应在每个 PLC 周期中调用一次功能块。当循环时间 <= 25 ms 时，可以获得满意的结果。如果 DCF-77 信号缺失或异常，则 ERR 输出将被设置为 TRUE，并且在 ERRID 输出端生成对应错误代码。下一次接收正确信号时，ERR 和 ERRID 会自动复位。一些接收器可提供反相 DCF-77 信号。在这种情况下，必须先对信号进行反相处理，然后才能将其传递到 DCF_PULSE 输入。当操作无误时，在 CDT 输出处每分钟更新一次当前时间。

因此，在第0秒时，READY输出会在PLC的一个周期内被置为TRUE。此时，CDT 输出端的 DCF-77 时间有效，可以由 PLC 程序进行评估。只有在能够正确无误地接收下一分钟的数据时，READY 输出才会被置位。传输的奇偶校验位可用于故障检测，并且会检查最后 2 个报文的可信度。如果接收条件较差，则无法保证 100% 无差错检测。即，如果在 2 个报文中有 2 个错误（反相）位，则功能块无法检测到错误，并且还会将 READY 输出设置为 TRUE。由于进行了可信度检查，因此相应位发生几何畸变从而导致无法检测到此类错误的概率非常小。

在接收过程中发生的错误由功能块记录。ERRCNT 输出是一个错误计数器。该计数器表示自上次正确接收信号以来发生的错误数。下一次正确接收信号时，计数器将被重置。

时间代码

在每分钟内，以BCD格式编码的年、月、日、星期几、时和分的信息会通过秒脉冲的调制进行传输。传输的信息总是描述随后的一分钟。每秒传输一个秒标记。持续时间为 0.1 s 的秒标记表示二进制0，而持续时间为 0.2 s 的秒标记表示二进制1。使用 3 个校验位扩展信息。在第 59 秒时，接收器可以利用这个间隔进行同步。

输入

```
VAR_INPUT
    DCF_PULSE  : BOOL;
    RUN        : BOOL;
    TLP        : TIME := T#140ms;
END_VAR
```

名称	类型	描述
DCF_PULSE	BOOL	DCF-77 信号。
RUN	BOOL	该输入的上升沿会初始化功能块，并开始对 DCF-77 信号进行解码。如果该输入被重置，则解码过程将停止。
TLP	TIME	根据接收器规格，通过该输入设置了一个固定限值来区分0和1。如果信号不佳，则脉冲宽度会变小。

TLP（传输线脉冲） 规范

接收器规格通常包含有关 2 个逻辑信号的最小和最大脉冲的信息，当场强度较高时，预期值较高，当场强度较低或受到干扰时，预期值较低。如果逻辑零的脉冲宽度过大，则在发送器附近（场强度非常大的地方）也可能会出现问题。**检查所用接收器的规格，并对脉冲长度进行相应配置。**

例如：在 Atmel T4227（时间代码接收器）的规格中提供了以下脉冲长度：
100 ms 脉冲（零）：最小值：70 ms，典型值：95 ms，最大值：**130 ms**
200 ms 脉冲（一）：最小值 **170 ms**，典型值 195 ms，最大值 235 ms
对于该 IC，150 ms 的限值为最佳值 = 130 + ((170 ms - 130 ms) / 2)。

如果配置的脉冲长度限值过小，则会将短脉冲检测为长脉冲。反之，如果配置的限值过大，则会将长脉冲检测为短脉冲。如果校验和正确，则接收器无法检测到这些错误。在第一种情况下，接收器可能会提供未来的时间，而在第二种情况下，时间可能是过去的时间。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
    ERRCNT  : UDINT;
    READY   : BOOL;
    CDT     : DATE_AND_TIME;
    DOW     : BYTE(1..7);(* ISO 8601 day of week: 1 = Monday.. 7 = Sunday *)
    TZI     : E_TimeZoneID;(* time zone information *)
    ADVTZI  : BOOL;(* MEZ->MESZ or MESZ->MEZ time change notification *)
    LEAPSEC : BOOL;(* TRUE = Leap second *)
    RAWDT   : ARRAY[0..60] OF BOOL;(* Raw decoded data bits *)
END_VAR
```

名称	类型	描述
BUSY	BOOL	在激活功能块时，将会设置该输出。
ERR	BOOL	在解码期间发生错误时，将会设置该输出。

名称	类型	描述
ERRID	UDINT	如果已设置 ERR 输出，则该参数会提供错误编号。
ERRCNT	UDINT	自上次正确接收信号以来发生的错误数。
READY	BOOL	如果已设置该输出，则 CDT 输出处的数据有效。
CDT	DATE_AND_TIME	以 DATE_AND_TIME 格式表示的 DCF-77 时间。
DOW	BYTE	根据 ISO 8601 规定的星期几：1 = 星期一…… 7 = 星期日
TZI	E_TimeZoneID	时区信息（夏令时/冬令时）
ADVTZI	BOOL	夏令时/冬令时转换通知，例如，CET -> CEST 或r CEST -> CET。在该小时结束时会发生 CEST/CET 之间的转换（见报文示例）。
LEAPSEC	BOOL	闰秒通知。在该小时结束时增加一个闰秒（见报文示例）。
RAWDT	ARRAY[0..60] OF BOOL	最后解码的（原始）位信息。请确保仅检查时间信息的奇偶校验位。不会分析天气数据的奇偶校验位！

错误代码	错误描述
0	无错误
0x100	超时错误。可能是因为没有检测到 DCF-77 信号。
0x200	奇偶校验错误。在接收的数据中检测到错误位。
0x300	接收错误数据。由于奇偶校验仅可检测到一个错误位，因此要再次检查接收数据的有效性（例如，如果月 = 13，则会生成该错误代码）。
0x400	最后一个解码周期过长。当接收效果不佳时（接收到的秒脉冲不足），可能会发生该错误。
0x500	最后一个解码周期过短。当接收效果不佳时（接收到额外的边沿信号），可能会发生该错误。

报文示例：

```
CEST -> CET (daylight-saving time -> standard time)

'0 01110100100111 011001 00011011 0100001 011001 111 00001 000100000': Sunday, 26.10.08 02:58:00, TZI = eTimeZoneID_Daylight, ADVTZI = TRUE
'0 11110001101110 011001 10011010 0100001 011001 111 00001 000100000': Sunday, 26.10.08 02:59:00, TZI = eTimeZoneID_Daylight, ADVTZI = TRUE
'0 01000001001110 010101 00000000 0100001 011001 111 00001 000100000': Sunday, 26.10.08 02:00:00, TZI = eTimeZoneID_Standard, ADVTZI = TRUE
'0 01111110100000 000101 10000001 0100001 011001 111 00001 000100000': Sunday, 26.10.08 02:01:00, TZI = eTimeZoneID_Standard, ADVTZI = FALSE


CET -> CEST (standard time -> daylight-saving time)

'0 01000110111010 010101 00011011 1000001 000011 111 11000 000100000': Sunday, 30.03.08 01:58:00, TZI = eTimeZoneID_Standard, ADVTZI = TRUE
'0 01000010100111 010101 10011010 1000001 000011 111 11000 000100000': Sunday, 30.03.08 01:59:00, TZI = eTimeZoneID_Standard, ADVTZI = TRUE
'0 10000111100011 011001 00000000 1100000 000011 111 11000 000100000': Sunday, 30.03.08 03:00:00, TZI = eTimeZoneID_Daylight, ADVTZI = TRUE
'0 01010000010110 001001 10000001 1100000 000011 111 11000 000100000': Sunday, 30.03.08 03:01:00, TZI = eTimeZoneID_Daylight, ADVTZI = FALSE


Leap second

'0 10110000100001 000111 10011010 0000000 100000 001 10000 100100001':   Thursday, 01.01.09 00:59:00, TZI = eTimeZoneID_Standard, LEAPSEC = TRUE
'0 11010010111000 000111 00000000 1000001 100000 001 10000 100100010': Thursday, 01.01.09 01:00:00, TZI = eTimeZoneID_Standard, LEAPSEC = TRUE
'0 01000110011101 000101 10000001 1000001 100000 001 10000 100100001':   Thursday, 01.01.09 01:01:00, TZI = eTimeZoneID_Standard, LEAPSEC = FALSE


: LEAPSEC bit;
: CET/CEST-Information;
: ADVTZI bit;
```

示例：

在示例应用程序中，如果接收的数据无差错，则 TwinCAT 软件时钟（RTC）将与无线电时间同步。

```
PROGRAM MAIN
VAR
    bDcfPulse AT%I* : BOOL;
    fbDcf           : DCF77_TIME_EX;
    bBusy           : BOOL;
    bError          : BOOL;
    nErrID          : UDINT;
    nErrCnt         : UDINT;
    bDcfValid       : BOOL;
    tDcfDt          : DT;
    nDow            : BYTE(1..7);
    eTzi            : E_TimeZoneID;(* time zone information *)
```

```
    bAdvTzi         : BOOL;(* MEZ->MESZ or MESZ->MEZ time change notification *)
    bLeapSec        : BOOL;(* TRUE = Leap second *)
    arRawDt         : ARRAY[0..60] OF BOOL;
    fbRtc           : RTC;
    bRtcValid       : BOOL;
    tRtcDt          : DT;
END_VAR
```

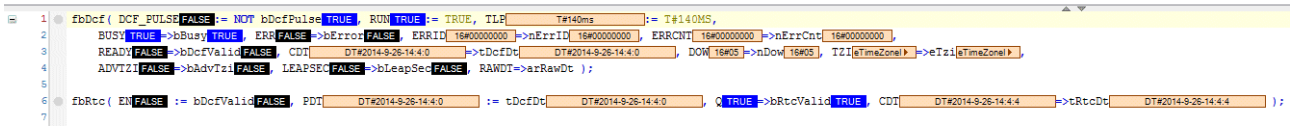
```
fbDcf( DCF_PULSE:= NOT bDcfPulse, RUN:= TRUE, TLP:= T#140MS,
       BUSY=>bBusy, ERR=>bError, ERRID=>nErrID, ERRCNT=>nErrCnt,
       READY=>bDcfValid, CDT=>tDcfDt, DOW=>nDow, TZI=>eTzi,
       ADVTZI=>bAdvTzi, LEAPSEC=>bLeapSec, RAWDT=>arRawDt );

fbRtc( EN := bDcfValid, PDT := tDcfDt, Q=>bRtcValid, CDT=>tRtcDt );
```

在线视图：

MAIN [Online] | P_DCF77_TIME [Online] | Source Control Explorer

TwinCAT_Device.Untitled1.MAIN

Expression	Type	Value	Prepared value	Address	Comment
bDcfPulse	BOOL	TRUE		%I*	
fbDcf	DCF77_TIME_EX				
bBusy	BOOL	TRUE			
bError	BOOL	FALSE			
nErrID	UDINT	16#00000000			
nErrCnt	UDINT	16#00000000			
bDcfValid	BOOL	FALSE			
tDcfDt	DATE_AND_TIME	DT#2014-9-26-14:4:0			
nDow	BYTE (BYTE#1..7)	16#05			
eTzi	E_TIMEZONEID	eTimeZoneID_Daylight			time zone information
bAdvTzi	BOOL	FALSE			MEZ->MESZ or MESZ-..time change notifi...
bLeapSec	BOOL	FALSE			TRUE = Leap second
arRawDt	ARRAY [0..60] OF B...				
fbRtc	RTC				
bRtcValid	BOOL	TRUE			
tRtcDt	DATE_AND_TIME	DT#2014-9-26-14:4:4			

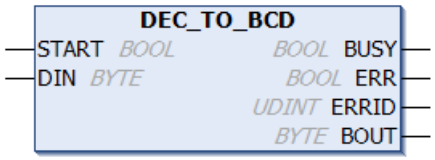


另请参见 DCF77_TIME [▶ 32] 功能块的相关描述。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.5 DEC_TO_BCD



功能块“DEC_TO_BCD”允许将十进制数转换为 BCD 格式。待转换的数值会进行有效性检查。

输入

```
VAR_INPUT
    START   : BOOL;
    DIN     : BYTE;
END_VAR
```

名称	类型	描述
START	BOOL	功能块由该输入端的上升沿触发激活。
DIN	字节	需要转换的十进制数

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
    BOUT    : BYTE;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在转换过程开始时，该输出将被置位并保持置位状态，直至转换完成。在 BUSY 输出被复位后，在 BOUT 输出中可以获取 BCD 值。
ERR	BOOL	如果发生错误，则该变量将被设置为 TRUE。
ERRID	UDINT	错误代码
BOUT	BYTE	如果该过程成功，则在该输出中可以获取 BCD 格式的转换变量。

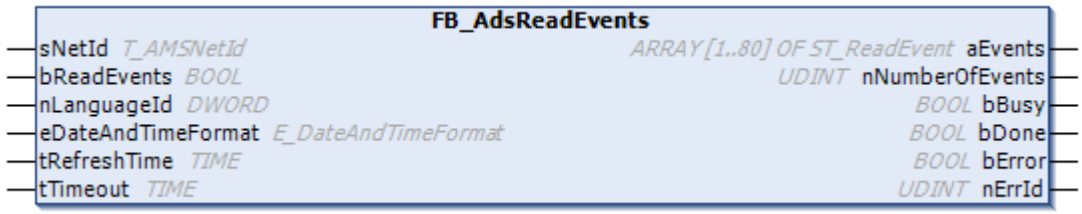
错误代码：

错误代码	错误描述
0	无错误
0x00FF	待转换的变量具有一个不允许的十进制值

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.6 FB_AdsReadEvents



该功能块通过 ADS 查询 EventLogger 的活动消息，并将这些消息以数组 `aEvents` 的形式提供。如要在可视化组件事件表中显示消息，则必须在其属性消息数据数组中输入数组 `aEvents`。

文本长度小于或等于 255 个字符的消息可以在输出端完整地输出。文本长度大于 255 个字符且小于或等于 1023 个字符的消息，将以截断文本的形式输出。文本长度超过 1023 个字符的消息将无法输出，功能块将返回错误消息。

输入

```
VAR_INPUT
    sNetId              : T_AMSNetId;
    bReadEvents         : BOOL;
    nLanguageId         : DWORD;
    eDateAndTimeFormat  : E_DateAndTimeFormat;
    tRefreshTime        : TIME
    tTimeout            : TIME
END_VAR
```

名称	类型	描述
sNetId	T_AMSNetId	设备的 AmsNetId，该设备中可以查询 EventLogger 消息。如果要在本地读取消息，可以指定一个空字符串。

名称	类型	描述
bReadEvents	BOOL	该输入可以用于启用消息读取。当取消发布时，也会重置错误输出（bError 和 nErrId）。
nLanguageId	DWORD	（语言 ID）定义要查询的消息文本的翻译。
eDateAndTimeFormat	E_DateAndTimeFormat	定义时间戳的格式。提供以下选项： • de_De – 德国符号：dd.MM.yyyy hh:mm:ss（24 小时制） • en_GB – 英国符号：dd/MM/yyyy hh:mm:ss（12 小时制） • en_US – 美国符号：MM/dd/yyyy hh:mm:ss（12 小时制）
tRefreshTime	TIME	定义重复消息查询的时间间隔。
tTimeout	TIME	定义触发超时错误的时间间隔。

输出

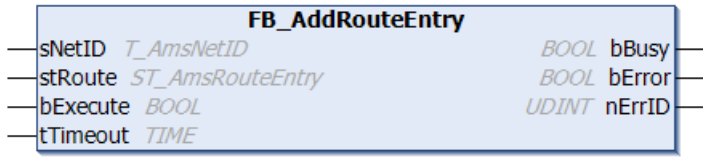
```
VAR_OUTPUT
    aEvents         : ARRAY[1..80] OF ST_ReadEvent;
    nNumberOfEvents : UDINT;
    bBusy           : BOOL;
    bDone           : BOOL;
    bError          : BOOL
    nErrorId        : UDINT;
END_VAR
```

名称	类型	描述
aEvents	ARRAY[1..80] OF ST_ReadEvent	功能块使用该数组提供读取消息。该数组最多可以存储 80 条消息。（请参见 ST_ReadEvent [▶ 339]）
nNumberOfEvents	UDINT	指明当前在数组 aEvents 中已存储多少条消息。
bBusy	BOOL	指明功能块当前是否繁忙。
bDone	BOOL	若功能块当前不处于忙碌状态，但已至少执行过一次操作，则返回 TRUE。
bError	BOOL	指明是否已发生错误。
nErrorId	UDINT	指明错误编号。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.7 FB_AddRouteEntry



该功能块可以用于将一个新的 AMS 路由器连接（远程路由）添加到 TwinCAT 系统中。

AMS 路由器连接列表

AMS 路由器连接的 2 个通信对端都有一个 AMS 路由器连接列表。这些列表包含 AMS 路由器连接。只有双方均已将对方输入各自的连接列表时，AMS 路由器连接才能正常运行。
在使用功能块时，仅会扩展通信对端的列表。

输入

```
VAR_INPUT
    sNetID   : T_AmsNetID;
    stRoute  : ST_AmsRouteEntry;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要将新连接添加到 AMS 路由器连接列表中的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
stRoute	ST_AmsRouteEntry [▶ 332]	带有新连接参数的结构体。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

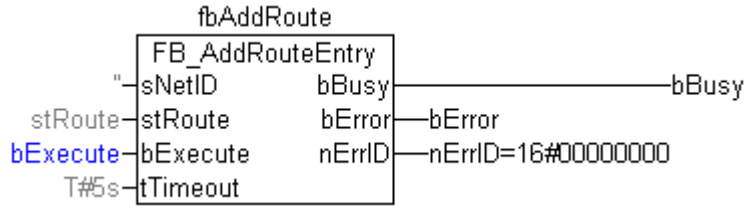
名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

示例：

在本地 TwinCAT 系统上要添加一个新的 AMS 路由器连接，连接名称：“TEST”，TwinCAT 网络地址：“172.16.6.111.1.1”，IP 地址：“172.16.6.111”，传输路由：“TCP/IP”。

```
PROGRAM P_TEST3
VAR
    fbAddRoute  : FB_AddRouteEntry;
    bExecute    : BOOL;
    bBusy       : BOOL;
    bError      : BOOL;
    nErrID      : UDINT;
    stRoute     : ST_AmsRouteEntry := ( sName := 'TEST',
                                        sNetID := '172.16.6.111.1.1',
                                        sAddress := '172.16.6.111',
                                        eTransport := eRouteTransport_TCP_IP );
END_VAR
```

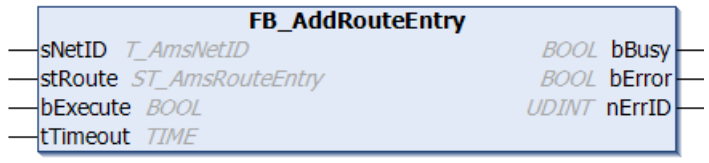
所需的连接参数已在声明部分初始化。如果在 bExecute 变量上检测到上升沿，则添加新的连接。



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.8 FB_AddRouteEntryEx



该功能块可以用于将一个新的 AMS 路由器连接（远程路由）添加到 TwinCAT 系统中。

该功能块通过虚拟 AmsNetId 支持 AmsNAT 功能。这样可以创建具有相同 AmsNetId 的 2 个或多个控制器的路由。

AMS 路由器连接列表

AMS 路由器连接的 2 个通信对端都有一个 AMS 路由器连接列表。这些列表包含 AMS 路由器连接。如果 2 个通信对端均已将对方输入其连接列表，则 AMS 路由器连接可以正常运行。
在使用功能块时，仅会扩展一个通信对端的列表。

输入

```
VAR_INPUT
    sNetID   : T_AmsNetID;
    stRoute  : ST_AmsRouteEntry;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要将新连接添加到 AMS 路由器连接列表中的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
stRoute	ST_AmsRouteEntry [▶ 332]	带有新连接参数的结构体。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

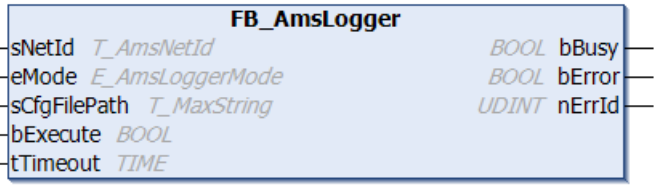
```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= v3.3.41.0

3.9 FB_AmsLogger



“TwinCAT AMS 记录器”是“TwinCAT ADS Monitor”的一个组件（..\TwinCAT\AdsMonitor\Logger**TcAmsLog.exe**）。记录器可将 AMS/ADS 命令记录在数据载体上。这些记录可以在后续进行显示和分析，例如，在使用“TwinCAT AMS ADS Viewer”进行故障排除期间。
FB_AmsLogger 功能块可用于从 PLC 程序启动或停止日志记录。FB_AmsLogger 功能块仅可与现有/正在运行的 TcAmsLog.exe 实例通信。换句话说，TcAmsLog.exe 必须已经启动，例如，通过开始菜单手动启动或通过 NT_StartProcesss [▶ 126] 功能块启动。

输入

```
VAR_INPUT
    sNetId       : T_AmsNetId := '';
    eMode        : E_AmsLoggerMode := AMSLOGGER_RUN;
    sCfgFilePath : T_MaxString := '';
    bExecute     : BOOL;
    tTimeout     : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetID	在这里可以指定应该修改“TwinCAT AMS 记录器”状态的 TwinCAT 计算机的网络地址。可以为本地计算机上的记录器指定一个空字符串。
eMode	E_AmsLoggerMode [▶ 323]	要设置的“TwinCAT AMS 记录器”的新状态（启动/停止记录）。
sCfgFilePath	T_MaxString	“TwinCAT AMS 记录器”配置文件的（可选）路径。目前尚未实施，为未来的应用保留（输入一个空字符串）。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

示例：

当 PLC 程序启动时，在本地系统上将会启动 TcAmsLog.exe 实例。当 bRecord 变量被设置为 TRUE 时，将会开始记录 AMS/ADS 命令。当变量被重置为 FALSE 时，将会停止记录。

声明部分：

```
PROGRAM MAIN
VAR
    bRecord        : BOOL := TRUE;(* TRUE => start recording, FALSE => stop recording *)
    fbStartProcess : NT_StartProcess := ( NETID := '', PATHSTR: = 'c:
\TwinCAT\AdsMonitor\Logger\TcAmsLog.exe',
                                          DIRNAME:= 'c:
\TwinCAT\AdsMonitor\Logger', COMNDLINE := '', TMOUT := DEFAULT_ADS_TIMEOUT );
```

```
    fbAmsLogger    : FB_AmsLogger := ( sNetID := '', eMode := AMSLOGGER_STOP, sCfgFilePath := '', tT
imeout := DEFAULT_ADS_TIMEOUT );
    state          : BYTE;
    bBusy          : BOOL;
    bError         : BOOL;
    nErrID         : UDINT;
    eCurrMode      : E_AmsLoggerMode := AMSLOGGER_STOP;(* Current mode/state *)
    eNewMode       : E_AmsLoggerMode := AMSLOGGER_STOP;(* New mode/state *)
    timer          : TON := ( PT := T#5s );
END_VAR
```

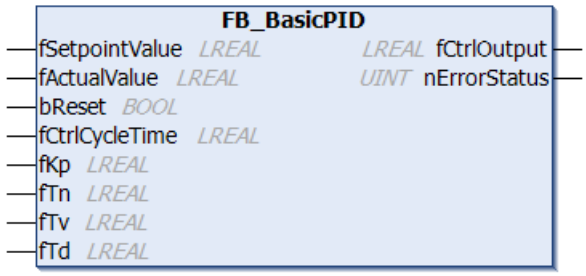
实现：

```
CASE state OF
0:(* Start instance of TcAmsLogger.exe *)
    fbStartProcess( START := FALSE );
    fbStartProcess( START:= TRUE );
    state := 1;

1:(* Wait until command execution started *)
    fbStartProcess( START := FALSE, BUSY=>bBusy, ERR=>bError, ERRID=>nErrID );
    IF NOT bBusy THEN
        IF NOT bError THEN(* Success *)
            state := 2;
        ELSE(* Error *)
            state := 100;
        END_IF
    END_IF

2:(*Wait until instance started or new AMS logger mode/state set *)
    timer( IN := TRUE );
    IF timer.Q THEN
        timer( IN := FALSE );
        state := 3;
    END_IF

3:(* Change TcAmsLog.exe mode/state *)
    eNewMode := SEL( bRecord, AMSLOGGER_STOP, AMSLOGGER_RUN);
    IF ( eNewMode <> eCurrMode ) THEN
        fbAmsLogger( bExecute := FALSE );
        fbAmsLogger( eMode:= eNewMode, bExecute := TRUE );
        state := 4;
    END_IF

4:(* Wait until command execution started *)
    fbAmsLogger( bExecute := FALSE, bBusy=>bBusy, bError=>bError, nErrID=>nErrID );
    IF NOT bBusy THEN
        IF NOT bError THEN(* Success *)
            eCurrMode := eNewMode;
            state := 2;
        ELSE(* Error *)
            state := 100;
        END_IF
    END_IF

100:(* Error state *)
    ;
END_CASE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.10 FB_BasicPID

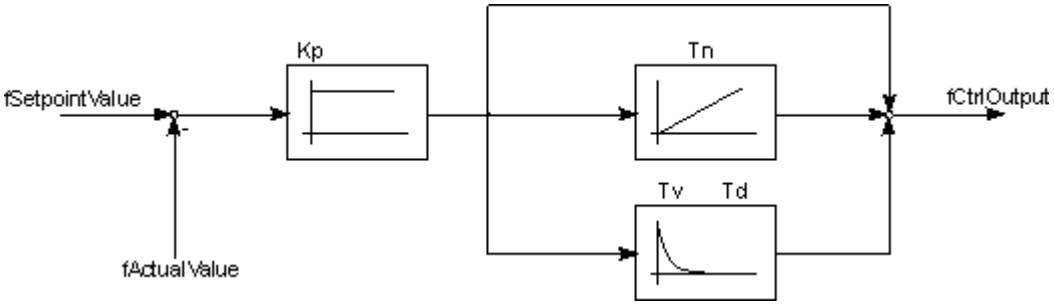


该功能块是一个简单的离散化 PID 元件。

传递函数：

$$G(s) = K_p(1 + \frac{1}{T_n s} + \frac{T_v s}{1 + T_d s})$$

功能图：



输入

```
VAR_INPUT
    fSetpointValue : LREAL; (* setpoint value *)
    fActualValue   : LREAL; (* actual value *)
    bReset         : BOOL;
    fCtrlCycleTime : LREAL; (* controller cycle time in seconds [s] *)
    fKp            : LREAL; (* proportional gain Kp (P) *)
    fTn            : LREAL; (* integral gain Tn (I) [s] *)
    fTv            : LREAL; (* derivative gain Tv (D-T1) [s] *)
    fTd            : LREAL; (* derivative damping time Td (D-T1) [s] *)
END_VAR
```

名称	类型	描述
fSetpointValue	LREAL	控制变量的设定点
fActualValue	LREAL	控制变量的实际值
bReset	BOOL	该输入为 TRUE 时会重置内部状态变量和控制器输出。
fCtrlCycleTime	LREAL	调用功能块和处理控制回路的循环时间 [s]。 如果在每个 PLC 周期中都调用该功能块，则**必须**在这里指定 PLC 任务的循环时间，否则应指定 PLC 循环时间的相应倍数。
fKp	LREAL	控制器放大/控制器系数
fTn	LREAL	积分作用时间 [s]
fTv	LREAL	微分作用时间 [s]
fTd	LREAL	阻尼时间 [s]

输出

```
VAR_OUTPUT
    fCtrlOutput  : LREAL;
    nErrorStatus : UINT
END_VAR
```

名称	类型	描述
fCtrlOutput	LREAL	PID 元件的输出
nErrorStatus	UINT	在发生错误时，指明错误编号（nErrorStatus <> 0）。

错误代码：

值	常量	错误描述
0	nERR_NOERROR	无错误
1	nERR_INVALIDPARAM	参数无效
2	nERR_INVALIDCYCLETIME	循环时间无效。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.11 FB_CalcHashValue

该功能块计算哈希值。

为此，可以使用方法 start()、update() 和 finish()。

这些方法可将输入数据的附加与哈希值的实际计算过程解耦，还允许通过多个步骤逐个添加输入数据。如果不需要多次添加输入数据，建议使用函数 F_GenerateHashValue() [▶ 288] 代替功能块。

方法 start()

该方法使用指定的哈希模式对哈希计算进行初始化。

```
METHOD start : BOOL
VAR_INPUT
    hashMode : E_HashMode;
END_VAR
```

名称	描述
hashMode	在这里指定一种哈希模式，例如 SHA 512。请参见 E_HashMode [▶ 325]。

方法 update()

该方法可以调用一次或多次。每次调用都会添加用于哈希计算的输入数据。

```
METHOD update : BOOL
VAR_INPUT
    pData : PVOID;
    nData : UDINT;
END_VAR
```

名称	描述
pData	在这里指定输入数据的地址。
nData	在这里指定输入数据的大小，以字节为单位。

方法 finish()

该方法执行哈希计算，并输出计算出的哈希值。

```
VAR_INPUT
    pHash : PVOID;
    nHash : UDINT;
END_VAR
```

名称	描述
pHash	在这里指定要存储哈希值的缓冲区的地址。
nHash	在这里指定哈希值的缓冲区的大小。大小取决于哈希模式，另请参见 E_HashMode [▶ 325]。

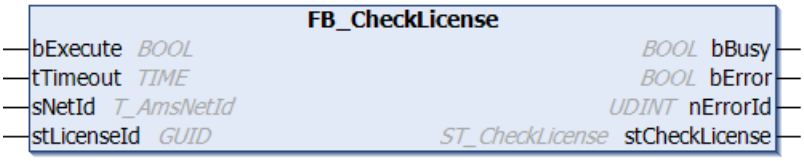
要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.29	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.51.0

3.12 FB_CheckLicense

使用 OEM 授权时，请确保启动项目已加密！

请记住，通过二进制代码的 FB_CheckLicense 查询的授权码很容易被找到，并可通过十六进制编辑器轻松地进行控制。因此，请确保加密启动项目 (最安全)，或尽可能地在源代码中隐藏查询到的授权码。



功能块 FB_CheckLicense 用于确定给定许可证 ID 的 TwinCAT 3 许可证状态。

输入

```
VAR_INPUT
    bExecute     : BOOL;
    tTimeout     : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId       : T_AmsNetId;
    stLicenseId  : GUID;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。
stLicenseId	GUID [▶ 331]	授权 ID

输出

```
VAR_OUTPUT
    bBusy          : BOOL;
    bError         : BOOL;
    nErrId         : UDINT
    stCheckLicense : ST_CheckLicense;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stCheckLicense	ST_CheckLicense [▶ 333]	带有授权数据的结构体

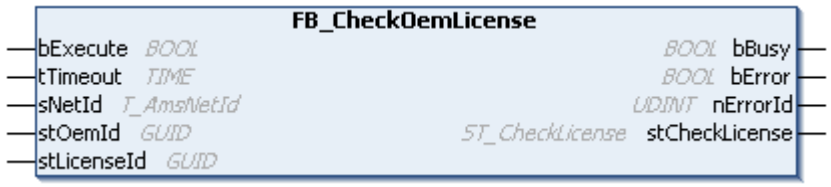
要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.24.0

3.13 FB_CheckOemLicense

如果您使用 OEM 授权，请确保对您的启动项目进行加密！

请记住，通过 FB_CheckOemLicense 功能块在二进制代码中查询的授权 ID很容易被找到，而且您可以使用十六进制编辑器进行操作（需稍作努力）。因此，请务必对您的启动项目进行加密（最安全），或者至少尽可能地隐藏在源代码中查询的授权 ID。



功能块 FB_CheckOemLicense 可确定给定授权 ID 和给定 OEM ID 的 TwinCAT 3 授权状态。

输入

```
VAR_INPUT
    bExecute     : BOOL;
    tTimeout     : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId       : T_AmsNetId;
    stOemId      : GUID;
    stLicenseId  : GUID;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。
stOemId	GIUD	OEM ID
stLicenseId	GUID [▶ 331]	授权 ID

输出

```
VAR_OUTPUT
    bBusy          : BOOL;
    bError         : BOOL;
    nErrId         : UDINT
    stCheckLicense : ST_CheckLicense;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stCheckLicense	ST_CheckLicense [▶ 333]	带有授权数据的结构体

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

3.14 FB_CSVMemBufferReader

该功能块可以用于将存储在外部缓冲区中的数据集分解/解释为各个数据字段。例如，借助文件访问功能块可以首先从文件中读取缓冲区数据。该功能块可读取第一个或下一个数据字段，并且在 *getValue*输出处以字符串的形式返回其值，或在 *pValue*/*cbValue* 输出处以地址/字节值的形式返回其值。

缓冲区中的数据必须采用特定的格式，以确保功能块能够正确地解析这些数据。CRLF 数据集分隔符（CR = 回车，LF = 换行）可用于分隔数据集。最后一个数据集必须以 CRLF 结束。各个数据字段必须使用数据字段分隔符进行分隔。默认的数据字段分隔符为分号。通过 PLC 全局变量 **DEFAULT_CSV_FIELD_SEP** 可以将分隔符从分号配置为逗号。

输入

```
VAR_INPUT
    eCmd     : E_EnumCmdType := eEnumCmd_First;
    pBuffer  : POINTER TO BYTE;
    cbBuffer : UDINT;
END_VAR
```

名称	类型	描述
eCmd	E_EnumCmdType [▶ 325]	缓冲区组件的控制参数。eEnumCmd_First 可读取第一个数据字段，eEnumCmd_Next 可读取下一个数据字段。不使用其他参数值。
pBuffer	BYTE	源缓冲区变量的地址（指针）。使用 ADR 运算符可以确定地址。该缓冲区包含要读取的数据集/数据字段数据。
cbBuffer	UDINT	在源缓冲区中要解释的数据的字节大小（数据集/数据字段数据）。缓冲区大小可能远大于要解释的数据量。请输入要解释的数据的实际长度。

输出

```
VAR_OUTPUT
    bOk      : BOOL;
    getValue : T_MaxString := '';
    pValue   : POINTER TO BYTE := 0;
    cbValue  : UDINT := 0;
    bCRLF    : BOOL := FALSE;
    cbRead   : UDINT := 0;
END_VAR
```

名称	类型	描述
bOk	BOOL	TRUE = 成功，FALSE = 错误数据/错误输入参数，或者已到达数据末尾且无法读取其他数据字段。
getValue	T_MaxString	最后读取的数据字段为字符串。对于不带控制字符和二进制数据的数据字段，此输出会将整个数据字段作为一个以空字符（null）结尾的字符串返回。不过，带有控制字符或二进制数据的数据字段可能会在该输出处返回不完整的字符串。在这种情况下，输出 *pValue/cbValue* 可用于访问最后读取的数据字段。
pValue	BYTE	数据字段中第一个数据字节的地址（指针）。请注意，空的无效数据字段并非以空字符结尾（对于 PLC 字符串通常如此），因此没有数据。在这种情况下，地址为空。
cbValue	UDINT	数据字段长度，以字节为单位。请注意，空的无效数据字段并非以空字符结尾（对于 PLC 字符串通常如此），因此没有数据。在这种情况下，长度也为空。使用全局参数 `cMaxCSVFieldValueSize` 可以指定最大大小。
bCRLF	BOOL	如果在最后读取的命令期间到达数据集的末尾，则会设置该输出。最后读取的数据字段属于上一个数据集。下一个数据字段属于一个新的数据集。
cbRead	UDINT	成功读取/解析的数据字节数。该数字可能大于 *cbValue* 输出处的数据字段长度。*cbRead* 输出的长度包括解析的数据字段/数据集分隔符。

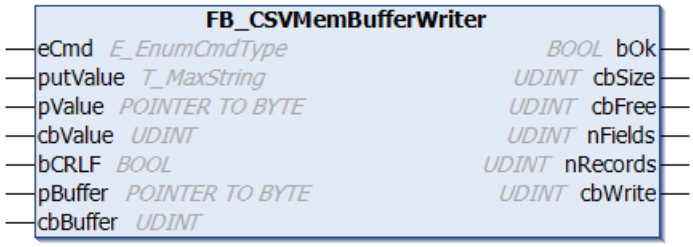
示例：

请参见：示例：写入/读取 CSV 文件。 [▶ 369]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.15 FB_CSVMemBufferWriter



该功能块可以用于从各个数据字段生成 CSV 格式的数据集，并储存在外部缓冲区中。然后可以将缓冲区的内容写入文件，例如，借助文件访问功能块。通过 *putValue 变量*（字符串）或 *pValue* 和 *cbValue* 变量可以将新的数据字段传输到功能块。具体使用哪种方式，这取决于您是想要向数据集中写入不带控制字符（字符串）的数据字段，还是想要写入带有控制字符或二进制数据的数据字段。该功能块可以在缓冲区中生成多个数据集，直到达到最大可用缓冲区大小。如果在写入数据字段的过程中将 bCRLF 变量设置为 TRUE，则会将数据集的末尾（当前数据集中的最后一个数据字段）自动添加到数据字段中。该功能块可自动添加数据字段分隔符。默认的数据字段分隔符为分号。通过 PLC 全局变量 **DEFAULT_CSV_FIELD_SEP** 可以将分隔符从分号配置为逗号。

输入

```
VAR_INPUT
    eCmd      : E_EnumCmdType := eEnumCmd_First;
    putValue  : T_MaxString := '';
    pValue    : POINTER TO BYTE := 0;
    cbValue   : UDINT := 0;
    bCRLF     : BOOL := FALSE;
    pBuffer   : POINTER TO BYTE;
    cbBuffer  : UDINT;
END_VAR
```

名称	类型	描述
eCmd	E_EnumCmdType [▶ 325]	eEnumCmd_First 可将第一个数据字段添加到缓冲区，eEnumCmd_Next 可添加下一个数据字段。不使用其他参数值。
putValue	T_MaxString	作为字符串形式的新数据字段。如果使用可选参数 *pValue* 和 *cbValue* 代替该输入，则该输入必须为空字符串。
pValue	BYTE	可选：包含新数据字段的外部字节缓冲区的地址。例如，与 cbValue 参数一起使用时，该参数可以用于将带有控制字符或二进制数据的数据字段写入数据集。数据字段中的控制字符或二进制数据可能会在非预期的位置截断 *putValue* 字符串，因此会以字节缓冲区的形式传输。如果不使用，该输入必须为空。
cbValue	UDINT	可选：外部字节缓冲区中的数据字段数据的长度。如果不使用，该输入必须为空。使用全局参数 cMaxCSVFieldValueSize 可以指定最大大小。
bCRLF	BOOL	如果设置该输入，则新的数据字段将以 CRLF 数据集分隔符结尾。后续数据字段属于新的数据集。
pBuffer	BYTE	目标缓冲区变量的地址（指针）。使用 ADR 运算符可以确定地址。在该缓冲区中，功能块可生成 CSV 格式的数据集。
cbBuffer	UDINT	目标缓冲区变量的最大可用大小（以字节为单位）。使用 SIZEOF 运算符可以确定大小

输出

```
VAR_OUTPUT
    bOk      : BOOL;
    cbSize   : UDINT;
    cbFree   : UDINT;
    nFields  : UDINT;
    nRecords : UDINT;
    cbWrite  : UDINT;
END_VAR
```

名称	类型	描述
bOk	BOOL	TRUE = 成功，FALSE = 缓冲区溢出或错误输入参数。
cbSize	UDINT	当前缓冲区填充状态（在缓冲区中创建的数据字节数）。
cbFree	UDINT	缓冲区中的空闲数据字节数。
nFields	UDINT	写入的数据字段的数量。
nRecords	UDINT	写入的数据集的数量。
cbWrite	UDINT	最后写入的数据字节数（最后一个数据字段的长度 + 任何数据集或数据字段分隔符）。

示例：

请参见：示例：写入/读取 CSV 文件。 [▶ 369]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.16 FB_EnumFindFileEntry

该功能块可在目录中搜索具有与指定名称相似的名称的文件或子目录。可以单独读取找到的任何条目。另请参见 FB_EnumFindFileList 功能块的相关描述。输入参数 *eCmd* 可用于浏览条目列表。例如，*eCmd* 输入可确定读取第一个输入还是下一个输入。

重要说明：

只有在前一次搜索已全部完成的情况下，才能开始新的搜索。功能块实例可能需要多次激活（通过 bExecute 输入的上升沿）才能完成搜索。只有在达到 bEOE =TRUE 或使用 ECMD = eEnumCmd_Abort 提前终止搜索的情况下，搜索才算已全部完成。

对于 TwinCAT 系统，如果 PLC 应用程序已经找到所寻找的文件或目录，则搜索可能尚未完成。

如果没有读取所有条目（即没有达到 *bEOE=TRUE*），则随后必须使用输入参数 *eCmd = eEnumCmd_Abort* 调用功能块。为了完成搜索并释放所有内部资源（文件句柄），此操作必不可少。如果达到 *bEOE=TRUE* 或发生错误，则将在内部自动执行 *eEnumCmd_Abort*。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID;
    sPathName : T_MaxString;
    eCmd      : E_EnumCmdType := eEnumCmd_First;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要执行目录搜索的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sPathName	T_MaxString	有效的目录名称或目录，带有字符串形式的文件名。字符串可以包含 (* 和 ?) 作为通配符。如果路径以通配符、点或目录名称结尾，则用户必须有对该路径及其子目录的访问权。
eCmd	E_EnumCmdType [▶ 325]	eEnumCmd_First 可将第一个数据字段添加到缓冲区，eEnumCmd_Next 可添加下一个数据字段。不使用其他参数值。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy      : BOOL;
    bError     : BOOL;
    nErrId     : UDINT;
    bEOE       : BOOL;
    stFindFile : ST_FindFileEntry;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
bEOE	BOOL	枚举结束。在第一次尝试读取不存在的条目时，该输出被设置为 TRUE。这意味着，只要 bEOE = FALSE 且 bError = FALSE，读取的条目就是有效的。
stFindFile	ST_FindFileEntry [▶ 337]	如果成功，则该结构变量可返回有关找到的文件的信息。

示例：

请参见：示例：文件搜索（FB_EnumFindFileEntry，FB_EnumFindFileList）。 [▶ 355]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.17 FB_EnumFindFileList

该功能块可在目录中搜索具有与指定名称相似的名称的文件或子目录。可以单独读取找到的任何条目。另请参见 FB_EnumFindFileEntry 功能块的相关描述。输入参数 *eCmd* 可用于浏览条目列表。例如，*eCmd* 输入可确定读取第一个输入还是下一个输入。

重要说明：

只有在前一次搜索已全部完成的情况下，才能开始新的搜索。功能块实例可能需要多次激活（通过 bExecute 输入的上升沿）才能完成搜索。只有在达到 bEOE =TRUE 或使用 ECMD = eEnumCmd_Abort 提前终止搜索的情况下，搜索才算已全部完成。

对于 TwinCAT 系统，如果 PLC 应用程序已经找到所寻找的文件或目录，则搜索可能尚未完成。

如果没有读取所有条目（即没有达到 *bEOE=TRUE*），则随后必须使用输入参数 *eCmd = eEnumCmd_Abort* 调用功能块。为了完成搜索并释放所有内部资源（文件句柄），此操作必不可少。如果达到 *bEOE=TRUE* 或发生错误，则将在内部自动执行 *eEnumCmd_Abort*。

输入

```
VAR_INPUT
    sNetID      : T_AmsNetID;
    sPathName   : T_MaxString;
    eCmd        : E_EnumCmdType := eEnumCmd_First;
    pFindList   : POINTER TO ST_FindFileEntry;
    cbFindList  : UDINT;
    bExecute    : BOOL;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要执行目录搜索的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sPathName	T_MaxString	有效的目录名称或目录，带有字符串形式的文件名。字符串可以包含 (* 和 ?) 作为通配符。如果路径以通配符、点或目录名称结尾，则用户必须有对该路径及其子目录的访问权。
eCmd	E_EnumCmdType [▶ 325]	eEnumCmd_First 可将第一个数据字段添加到缓冲区，eEnumCmd_Next 可添加下一个数据字段。不使用其他参数值。
pFindList	ST_FindFileEntry [▶ 337]	数组变量的地址（指针变量）
cbFindList	ST_FindFileEntry [▶ 337]	数组变量的字节大小
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    bEOE        : BOOL;
    nFindFiles  : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
bEOE	BOOL	枚举结束。在第一次尝试读取不存在的条目时，该输出被设置为 TRUE。这意味着，只要 bEOE = FALSE 且 bError = FALSE，读取的条目就是有效的。
nFindFiles	UDINT	缓冲区中的有效条目的数量。

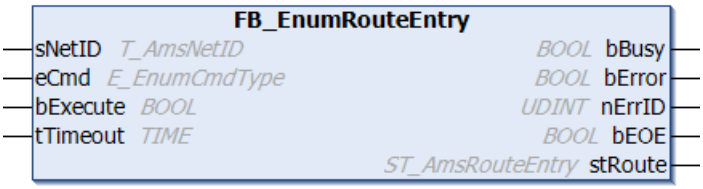
示例：

请参见：示例：文件搜索（FB_EnumFindFileEntry，FB_EnumFindFileList）。 [▶ 355]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.18 FB_EnumRouteEntry



该功能块可用于通过 AMS 路由器连接（远程路由）向其他 TwinCAT 系统传输信息。如果使用多个连接，则必须重复调用该功能块。每次调用只能处理一个条目。输入参数 *eCmd* 可用于浏览条目列表。例如，*eCmd* 输入可确定读取第一个输入还是下一个输入。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID;
    eCmd      : E_EnumCmdType := eEnumCmd_First;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要执行目录搜索的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
eCmd	E_EnumCmdType [▶ 325]	枚举块的命令参数。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    bEOE      : BOOL;
    stRoute   : ST_AmsRouteEntry;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
bEOE	BOOL	枚举结束。在第一次尝试读取不存在的条目时，该输出被设置为 TRUE。这意味着，只要 bEOE = FALSE 且 bError = FALSE，读取的条目就是有效的。

名称	类型	描述
stRoute	ST_AmsRouteEntry [▶ 332]	包含最后读取的连接参数的结构体。

示例：

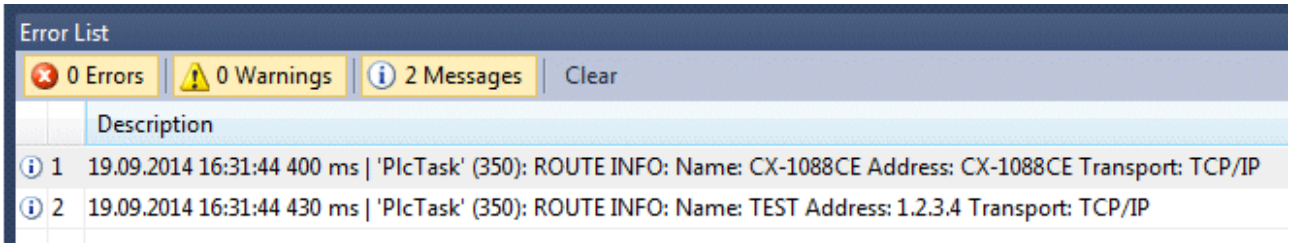
在本地 TwinCAT 系统上将读取配置的 AMS 路由器连接，并将其作为消息写入 TwinCAT 系统管理器记录器输出。

```
PROGRAM P_EnumRouteEntries
VAR
    fbEnum : FB_EnumRouteEntry := ( sNetID := '', tTimeout := T#5s );
    bEnum  : BOOL := TRUE;
    nState : BYTE := 0;
    sInfo  : T_MaxString;
END_VAR
```

bEnum 变量的上升沿会触发读取连接信息。

```
CASE nState OF
    0:
    IF bEnum THEN (* flag set ? *)
        bEnum := FALSE; (* reset flag *)
        fbEnum.eCmd := eEnumCmd_First; (* enum first entry *)
        nState := 1;
    END_IF

    1: (* enum one entry *)
    fbEnum( bExecute := FALSE );
    fbEnum( bExecute := TRUE );
    nState := 2;

    2: (* wait until function block not busy *)
    fbEnum( bExecute := FALSE );
    IF NOT fbEnum.bBusy THEN
        IF NOT fbEnum.bError THEN
            IF NOT fbEnum.bEOE THEN
                sInfo := CONCAT( 'Name: ', fbEnum.stRoute.sName );
                sInfo := CONCAT( sInfo, ' Address: ' );
                sInfo := CONCAT( sInfo, fbEnum.stRoute.sAddress );
                sInfo := CONCAT( sInfo, ' Transport: ' );
                sInfo := CONCAT( sInfo, ROUTETRANSPORT_TO_STRING( fbEnum.stRoute.eTransport ) );
                ADSLOGSTR( ADSLOG_MSGTYPE_HINT OR ADSLOG_MSGTYPE_LOG, 'ROUTE INFO: %s', sInfo );
                    fbEnum.eCmd := eEnumCmd_Next; (* enum next entry *)
                nState := 1;
            ELSE (* no more route entries *)
                nState := 0;
            END_IF
        ELSE (* log error *)
            ADSLOGSTR( ADSLOG_MSGTYPE_ERROR OR ADSLOG_MSGTYPE_LOG, 'FB_EnumRouteEntry error:
%s', DWORD_TO_HEXSTR( fbEnum.nErrID, 0, FALSE ) );
            nState := 0;
        END_IF
    END_IF
END_CASE
```

在 TwinCAT 系统管理器记录器输出中记录消息：



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.19 FB_EnumStringNumbers

该功能块可以用于在 REPEAT 或 WHILE 循环中的字符串中搜数字。字符串可能包含多个数字。在功能块输出处，找到的任何数字都会作为子字符串输出。该函数从当前位置搜索第一个可被解析为数字的字符。如果找到的字符不能被解析为数字，则会中止搜索。*eCmd* 参数可确定搜索第一个数字还是下一个数字。*eType* 参数可确定搜索字符串中的数字的格式。

输入

```
VAR_INPUT
    sSearch    : T_MaxString;
    eCmd       : E_EnumCmdType := eEnumCmd_First;
    eType      : E_NumGroupTypes := eNumGroup_Float;
END_VAR
```

名称	类型	描述
sSearch	T_MaxString	要搜索数字的搜索字符串
eCmd	E_EnumCmdType [▶ 325]	枚举块的控制命令
eType	E_NumGroupTypes [▶ 327]	所搜索数字的数字格式。该参数可确定要忽略哪些字符，以及要将哪些字符解释为数字：

值	含义
eNumGroup_Float	数字“0”至“9”、“+”、“-”（符号）以及“e”或“E”（指数字符）可被解释为有效数字。
eNumGroup_Unsigned	数字“0”至“9”可被解释为有效数字。“+”、“-”、“e”、“E”字符可被忽略为数字。
eNumGroup_Signed	数字“0”至“9”、“+”、“-”（符号）可被解释为有效数字。“e”、“E”字符可被忽略。

如果字符串包含采用指数表示法的数字，则必须设置 eType = *eNumGroup_Float*（默认值）。

输出

```
VAR_OUTPUT
    sNumber    : T_MaxString;
    nPos       : INT;
    bEOS       : BOOL;
END_VAR
```

名称	类型	描述
sNumber	T_MaxString	最后以字符串形式的找到的数字
nPos	INT	该变量始终返回最后一个成功识别且格式正确的数字字符之后的位置索引。即，此时功能块将在下一次调用时搜索新的数字字符。如果已遍历完 *sSearch 字符串*的全部内容，则 *nPos* 为零。字符串中的首个字符位置编号为1（采用非零基位置编号）。
bEOS	BOOL	如果找到一个新的数字且尚未到达字符串的末尾，则该变量为 FALSE。在这种情况下，*sNumber* 会以字符串形式返回一个有效数字。如果没有找到其他数字，则该变量为 TRUE。在这种情况下，必须中止任何进一步的搜索（*sNumber* 没有返回有效值）。

示例：

在下面的示例中，将搜索 *sNumber* 变量中的有效数字。任何找到的子字符串都会存储在数组变量 *arrNums* 中。

```
TYPE ST_ScanRes :
STRUCT
    sNumber   : T_MaxString;
    nPos      : INT;
    sRemain   : T_MaxString;
END_STRUCT
END_TYPE
```

```
PROGRAM MAIN
VAR
    sSearch    : T_MaxString := 'Some numbers in string: +-12e-34, -56, +78';
    fbEnum     : FB_EnumStringNumbers := ( eType := eNumGroup_Float (* eNumGroup_Signed, eNumGroup_U
nsigned *) );
    arrNums    : ARRAY[1..MAX_SCAN_NUMS] OF ST_ScanRes;
    idx        : INT;
    length     : INT;
    bEnum      : BOOL := TRUE;
END_VAR
VAR CONSTANT
    MAX_SCAN_NUMS : INT := 10;
END_VAR
```

```
IF bEnum THEN
    bEnum := FALSE;

    MEMSET( ADR( arrNums ), 0, SIZEOF( arrNums ) );
    idx := 0;
    length := LEN( sSearch );

    fbEnum( sSearch := sSearch, eCmd := eEnumCmd_First );
    WHILE NOT fbEnum.bEOS DO
        IF idx < MAX_SCAN_NUMS THEN
            idx := idx + 1;

            arrNums[idx].sNumber:= fbEnum.sNumber;
            arrNums[idx].nPos := fbEnum.nPos;
            IF fbEnum.nPos <> 0 THEN
                arrNums[idx].sRemain:= RIGHT( sSearch, length - fbEnum.nPos + 1 );
            END_IF

        END_IF
        fbEnum( eCmd := eEnumCmd_Next );
    END_WHILE
END_IF
```

找到的字符串：

- '-12e-34'
- '-56'
- '+78'

eType 参数 *eNumGroup_Signed* 返回以下结果：

- '-12'
- '-34'
- '-56'
- '+78'

eType 参数 *eNumGroup_Unsigned* 返回以下结果：

- '12'
- '34'
- '56'
- '78'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.20 FB_FileProperties

该功能块可输出所选文件的文件属性。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID;
    sPathName : T_MaxString;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要执行目录搜索的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sPathName	T_MaxString	有效的目录名称或目录，带有字符串形式的文件名。字符串可以包含 (* 和 ?) 作为通配符。如果路径以通配符、点或目录名称结尾，则用户必须有对该路径及其子目录的访问权。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
ePath	E_OpenPath	该输入可以用于选择目标设备上的 TwinCAT 系统路径，以打开文件。

输出

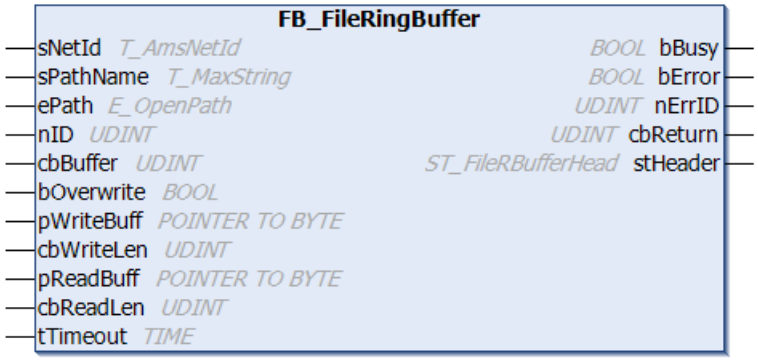
```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrId      : UDINT;
    stProperties : ST_FindFileEntry;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stProperties	ST_FindFileEntry [▶ 337]	如果成功，则该结构变量可返回有关找到的文件的信息。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.21 FB_FileRingBuffer



功能块 FB_FileRingBuffer 允许将不同长度的数据集写入环形缓冲区文件，或者将先前已写入的数据集从环形缓冲区文件中移除。根据 FIFO 原则（先进先出）读出已写入的数据集，与先前将它们写入环形缓冲区文件的顺序相同。这意味着最早写入的数据集（即最旧的数据集）会最先被读取。数据集的打开、关闭、写入和读取操作均通过调用方法/函数来控制。该功能块包含以下核心功能：

- **A_Open**（打开一个现有的环形缓冲区文件，以添加或生成新的数据集。如果文件已经打开，则不会返回错误。）
- **A_Close**（关闭一个打开的环形缓冲区文件。如果文件已经关闭，则不会返回错误。）
- **A_Create**（打开一个新的环形缓冲区文件。如果文件已经存在，则会将其覆盖。如果文件已经打开，则不会返回错误）
- **A_AddTail**（将新的数据集写入环形缓冲区文件。）
- **A_GetHead**（从环形缓冲区文件中读取最早的数据集，但不会将其移除 – 文件指针不会移动到下一个数据集。）
- **A_RemoveHead**（从环形缓冲区文件中读取并移除最旧的数据集 – 将文件指针移动到下一个数据集。）
- **A_Reset**（从打开的环形缓冲区文件中删除所有数据集。只有文件指针和数据集的数量会被重置；现有的物理文件大小不会改变，不过，最旧的数据集将被新的数据集覆盖。

在打开已经存在的环形缓冲区文件时，首先会读取文件头。在先前已无差错关闭的环形缓冲区文件中，文件头状态（Header.status.Bit 0）中的第 0 位必须为零。如果不是这样的话，则假定事先没有正确关闭文件，损坏的文件将被一个新的空文件替换，同时 Header.status.Bit 1 将被设置为 1（文件损坏）。在关闭文件时，Header.status.Bit 0 会被设置为 0，文件中的完整文件头会进行更新。

输入

```
VAR_INPUT
    sNetId      : T_AmsNetId := '';
    sPathName   : T_MaxString := 'c:\Temp\data.dat';
    ePath       : E_OpenPath := PATH_GENERIC;
    nID         : UDINT := 0;
    cbBuffer    : UDINT := 16#100000;
    bOverwrite  : BOOL := FALSE;
    pWriteBuff  : POINTER TO BYTE;
    cbWriteLen  : UDINT;
    pReadBuff   : POINTER TO BYTE;
    cbReadLen   : UDINT;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要执行目录搜索的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sPathName	T_MaxString	包含要打开的缓冲区文件的路径和文件名（类型：T_MaxString）。**注**：路径只能指向本地计算机的文件系统！这意味着在这里不能使用网络路径！
ePath	E_OpenPath	该输入可以用于选择目标设备上的 TwinCAT 系统路径，以打开文件。

名称	类型	描述
nID	UDINT	用户定义的 32 位值。在打开一个新文件时，该值会保存在文件中，并且可以用于检查缓冲区文件的版本等。
cbBuffer	UDINT	要打开的缓冲区文件的最大大小，以字节为单位。在创建文件时会将该参数保存在文件头中，在再次打开相同文件时会对其进行检查。您只能重新打开已经使用相同的最大缓冲区大小创建的文件。换句话说，您不能使用较小的缓冲区大小创建文件，用数据集填满，然后再使用较大的缓冲区大小打开它。如果对最大缓冲区大小检查失败，则会自动创建并打开一个具有新的缓冲区大小的新文件。在文件头状态中也设置了第 1 位（文件损坏）。
bOverwrite	BOOL	在达到最大文件缓冲区大小时的写入行为。如果该输入为 TRUE，则在已达到最大文件缓冲区大小的情况下，最早的条目会被覆盖（条目会被删除，直到空闲缓冲区大小足以存储新的条目为止）。如果该输入为 FALSE，则在达到最大文件缓冲区大小时，缓冲区溢出将被报告为错误。
pWriteBuff	BYTE	要写入的数值数据所对应的 PLC 变量地址或缓冲区变量地址。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便可以从中获取 *cbWriteLen* 数据字节。
cbWriteLen	UDINT	要写入的数据字节数（对于字符串变量，这包括末尾的空字符）。
pReadBuff	BYTE	PLC 变量或缓冲区变量的地址，已读取的值数据将被复制到该地址中。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便其可以容纳 cbReadLen 数据字节。缓冲区变量的大小（以字节为单位）必须大于或等于要读取的数据集的大小。
cbReadLen	UDINT	要读取的值数据字节数。如果缓冲区大小过小，则不会复制任何数据，功能块会报告缓冲区下溢错误，并在 *cbReturn* 输出处返回要读取的下一个数据集所需的缓冲区大小。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrId    : UDINT;
    cbReturn  : UDINT;
    stHeader  : ST_FileRBufferHead;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
cbReturn	UDINT	成功读取的值数据字节数。如果发生读取缓冲区下溢错误，则该输出将提供必要的读取缓冲区大小（以字节为单位）。
stHeader	ST_FileRBufferHead [▶ 336]	环形缓冲区文件头/状态。

示例：

请参见：示例：文件环 FiFo（FB_FileRingBuffer）。 [▶ 357]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

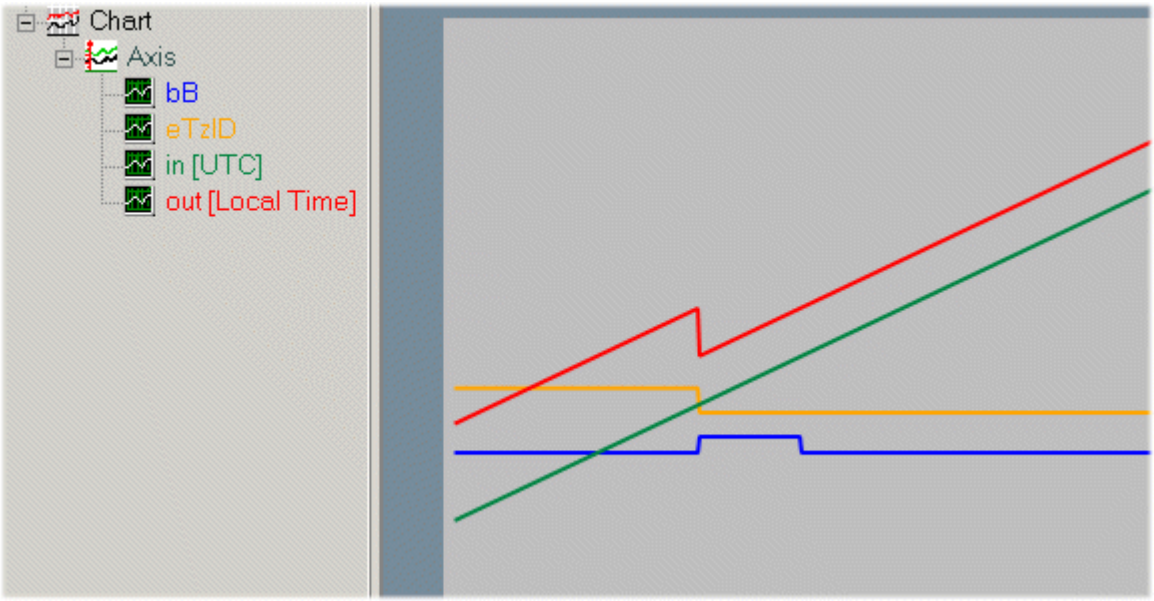
3.22 FB_FileTime64ToTzSpecificLocalTime

该功能块将 UTC 时间（文件时间格式）转换为本地时间（文件时间格式），同时考虑指定的时区信息。功能块：FB_SystemTimeToTzSpecificLocalTime [▶ 112] 具有类似的功能，但使用不同的时间格式（结构化系统时间格式）。

该功能块仅适用于转换**连续的** UTC 时间戳信息。该功能块使用时区信息来计算在本地时间中所需的时间步长（夏令时/冬令时转换）。不允许使用在 UTC 输入时间中的时间步长，否则会导致转换错误。原因：该功能块在内部存储最后转换的时间，因此，当本地时间发生变化时，它可以从 UTC 输入时间和存储值中检测出 B 时间（见下文）。

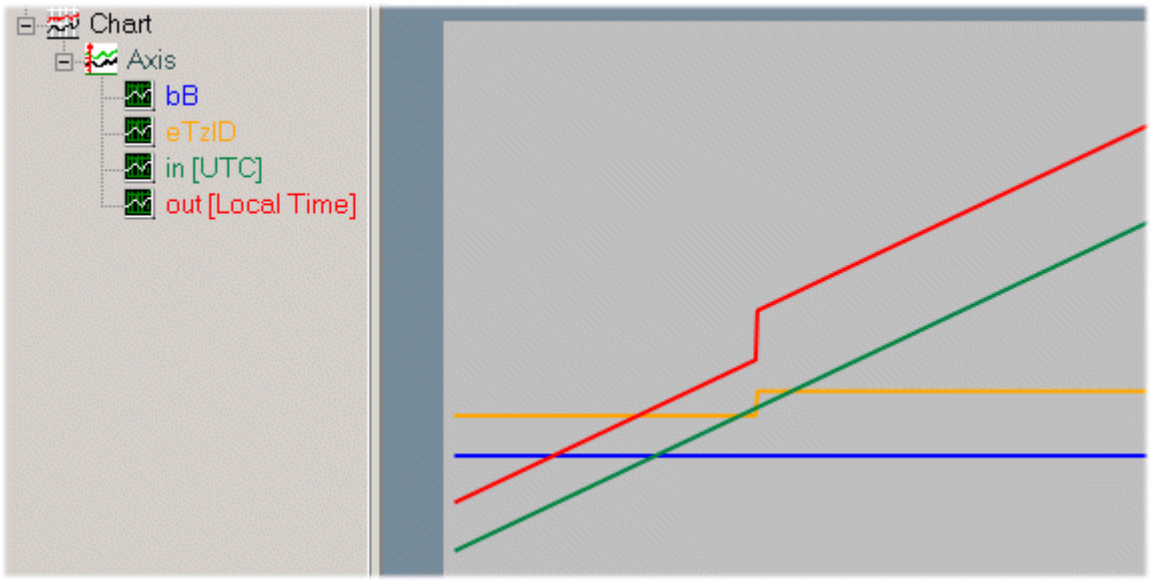
该功能块与A_Reset()相关联。如果调用该操作，则功能块输出和本地存储（最后转换）的时间将被重置为零。

1. 从夏令时到冬令时转换的图示（tzInfo = WEST_EUROPE_TZI）：



UTC 输入时间（绿色）是连续的。本地时间（红色）向后跳转。本地时间：**02h:59m:59s:999ms..** 后面直接是：**02h:00m:00s:000ms..** 2h 和 3h 之间的时间出现两次。例如，时间转换前的重复时间被称为 **02:05:00 CEST A**，转换后的时间被称为 **02:05:00 CET B**。输出变量 *bB* 指示它是第一次还是第二次*经过*。第二次*经过*时，输出变量 *bB*（蓝色）将被设置为 TRUE。在经过重复时间后，自动重置 *bB* 输出变量。时区 ID（橙色）从 *eTimeZoneID_Daylight*（夏令时）变为 *eTimeZoneID_Standard*（冬令时）。

2. 从冬令时到夏令时转换期间的时间行为图示（tzInfo = WEST_EUROPE_TZI）：



UTC 输入时间（绿色）是连续的。本地时间（绿色）向前跳转。本地时间：**2h:59m:59s:999ms..** 后面直接是：**3h:00m:00s:000ms..** 时区 ID（橙色）从 *eTimeZoneID_Standard*（冬令时）变为 *eTimeZoneID_Daylight*（夏令时）。

输入

```
VAR_INPUT
    in     : T_FILETIME64;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
in	T_FILETIME64 [▶ 344]	要转换的 UTC 时间（文件时间格式）。
tzInfo	ST_TimeZoneInformation [▶ 342]	带有操作系统的当前时区信息的结构变量。

输出

```
VAR_OUTPUT
    out   : T_FILETIME64;
    eTzID : E_TimeZoneID := eTimeZoneID_Unknown;
    bB    : BOOL;
END_VAR
```

名称	类型	描述
out	T_FILETIME64 [▶ 344]	转换后的本地时间（文件时间格式）
eTzID	E_TimeZoneID [▶ 329]	附加的夏令时/冬令时信息。
bB	BOOL	TRUE => B 时间（例如：**02:05:00 CET B**），FALSE => 其他时间（例如：**02:05:00 CEST A**）。当本地时间发生回跳时，该输出被置位，并在重复的本地时间通过后复位。

示例：

UTC 时间：DT#2011-09-02-09:01:31 被转换为本地时间。结果为：DT#2011-09-02-11:01:31。

```
PROGRAM MAIN
VAR
    in        : DT := DT#2011-09-02-09:01:31;(* UTC time *)
    out       : DT;(* Local time *)
    fbToLocal : FB_FileTime64ToTzSpecificLocalTime;
END_VAR
```

```
fbToLocal( in := DT_TO_FILETIME64( in ), tzInfo := WEST_EUROPE_TZI );
out := FILETIME64_TO_DT( fbToLocal.out );
```

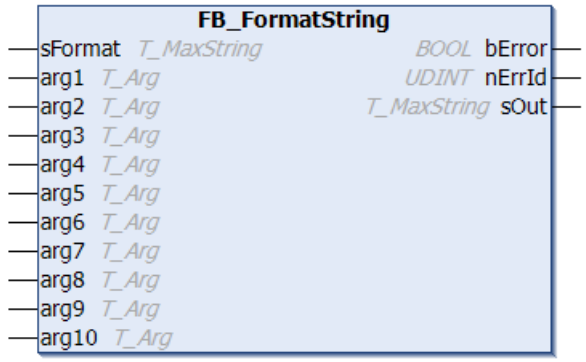
其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime64 [▶ 113]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTime64Bias [▶ 154]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

3.23 FB_FormatString



该功能块可以用于将最多 10 个参数（类似于 fprintf）转换为字符串，并按照格式规格 [▶ 373]对它们进行格式化。在相同的 PLC 周期内进行格式化。这意味着在调用 FB 后，输出字符串立即可用。

输入

```
VAR_INPUT
    sFormat    : T_MaxString;
    arg1       : T_Arg;
    arg2       : T_Arg;
    arg3       : T_Arg;
    arg4       : T_Arg;
    arg5       : T_Arg;
    arg6       : T_Arg;
    arg7       : T_Arg;
    arg8       : T_Arg;
    arg9       : T_Arg;
    arg10      : T_Arg;
END_VAR
```

名称	类型	描述
sFormat	T_MaxString	字符串格式规范（例如，“%+20.5f”或“Measure X: %+.10d, Y: %+.10d”）。
arg1 至 arg10	T_Arg [▶ 343]	要格式化的参数。以下辅助函数可以用于将不同类型的 PLC 变量转换为所需的数据类型 T_Arg [▶ 343]：F_BYTE [▶ 249]、F_WORD [▶ 259]、F_DWORD [▶ 250]、F_LWORD [▶ 253]、F_SINT [▶ 254]、F_INT [▶ 251]、F_DINT [▶ 250]、F_LINT [▶ 252]、F_USINT [▶ 258]、F_UINT [▶ 257]、F_UDINT [▶ 256]、F_ULINT [▶ 258]、F_STRING [▶ 255]、F_REAL [▶ 254]、F_LREAL [▶ 253]。

输出

```
VAR_OUTPUT
    bError : BOOL;
    nErrId : UDINT;
    sOut   : T_MaxString;
END_VAR
```

名称	类型	描述
bError	BOOL	如果在格式化期间出现错误，则为 TRUE。
nErrId	UDINT	如果设置了 bError 输出，则返回格式错误代码 [▶ 375]。
sOut	T_MaxString	如果成功，则该输出将返回格式化的输出字符串

示例：

```
PROGRAM MAIN
VAR
    fbFormat    : FB_FormatString;
    iY          : DINT;
    iX          : DINT;
    bError      : BOOL;
    nErrID      : UDINT;
    sOut        : T_MaxString;
END_VAR
```

```
iX := iX + 1;
iY := iY + 1;
fbFormat( sFormat := 'Measure X: %+.10d, Y: %
+.10d', arg1 := F_DINT( iX ), arg2 := F_DINT( iY ), sOut => sOut, bError => bError, nErrID => nErrID
 );
```

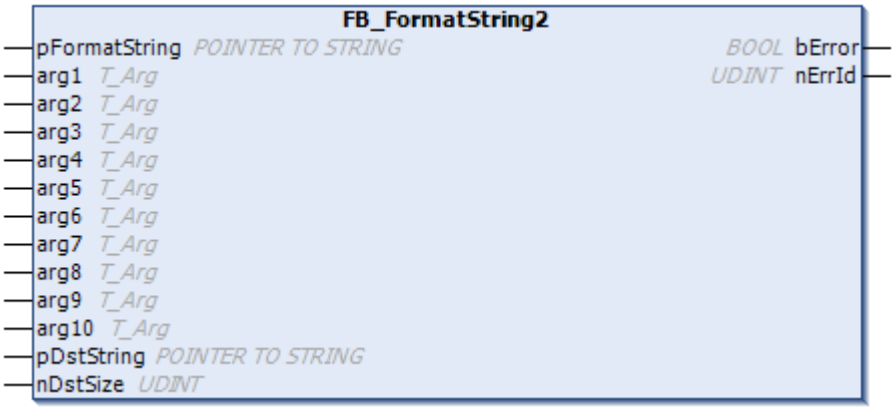
结果：

sOut = 'Measure X: +0000000130, Y: +0000000130'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.24 FB_FormatString2



该功能块可以用于将最多 10 个参数（类似于 fprintf）转换为字符串，并按照格式规格 [▶ 373]对它们进行格式化（例如，“%+20.5f”或“Measure X: %+.10d, Y: %+.10d”）。在相同的 PLC 周期内进行格式化。这意味着在调用功能块后，输出字符串立即可用。

与功能块 FB_FormatString [▶ 63] 相反，格式字符串和输出字符串的大小不限于 255 个字符。不过，每个参数最多只能表示 250 个字符。如果超出字符数或输出字符串对于格式化的字符串来说过小，则该功能块会返回错误。

输入

```
VAR_INPUT
    pFormatString : POINTER TO STRING;
    arg1          : T_Arg;
    arg2          : T_Arg;
    arg3          : T_Arg;
    arg4          : T_Arg;
    arg5          : T_Arg;
    arg6          : T_Arg;
    arg7          : T_Arg;
    arg8          : T_Arg;
    arg9          : T_Arg;
    arg10         : T_Arg;
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
END_VAR
```

名称	类型	描述
pFormatString	STRING	指向格式规范字符串的指针。每次调用功能块时必须分配地址。运算符 ADR() 可以用于赋值。
arg1 至 arg10	T_Arg [▶ 343]	要格式化的参数。以下辅助函数可以用于将不同类型的 PLC 变量转换为所需的数据类型 T_Arg [▶ 343]：F_BYTE [▶ 249]、F_WORD [▶ 259]、F_DWORD [▶ 250]、F_LWORD [▶ 253]、F_SINT [▶ 254]、F_INT [▶ 251]、F_DINT [▶ 250]、F_LINT [▶ 252]、F_USINT [▶ 258]、F_UINT [▶ 257]、F_UDINT [▶ 256]、F_ULINT [▶ 258]、F_STRING [▶ 255]、F_REAL [▶ 254]、F_LREAL [▶ 253]。
pDstString	STRING	指向生成的 STRING 变量的指针。如果成功，则在这里将会写入格式化的字符串。每次调用功能块时必须分配地址。运算符 ADR() 可以用于赋值。
nDstSize	UDINT	结果 STRING 变量的大小，以字节为单位。运算符 SIZEOF() 可以用于赋值。

输出

```
VAR_OUTPUT
    bError : BOOL;
    nErrId : UDINT;
END_VAR
```

名称	类型	描述
bError	BOOL	如果在格式化期间出现错误，则为 TRUE。
nErrId	UDINT	如果设置了 bError 输出，则返回格式错误代码 [▶ 375]。

示例

```
PROGRAM MAIN
VAR
    fbFormat    : FB_FormatString2;
    sFormat     : STRING := 'Measure X: %+.10d, Y: %+.10d';
    iY          : DINT;
    iX          : DINT;
    bError      : BOOL;
    nErrID      : UDINT;
    sOut        : STRING(600);
END_VAR
```

```
iX := iX + 1;
iY := iY + 1;
fbFormat( pFormatString := ADR(sFormat), arg1 := F_DINT( iX ), arg2 := F_DINT( iY ), pDstString :=
ADR(sOut), nDstSize := SIZEOF(sOut), bError => bError, nErrID => nErrID );
```

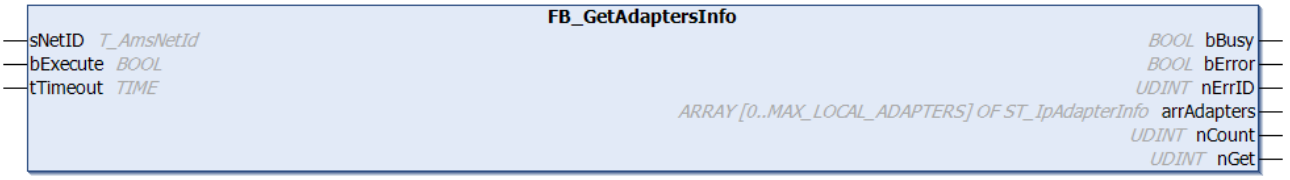
结果：

sOut = 'Measure X: +0000000130, Y: +0000000130'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

3.25 FB_GetAdaptersInfo



该功能块可以用于读取 TwinCAT PC 的适配器信息。目前，可以读取的适配器信息最大数量限值为 MAX_LOCAL_ADAPTERS + 1（默认值 = 6）。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要读取其适配器信息的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

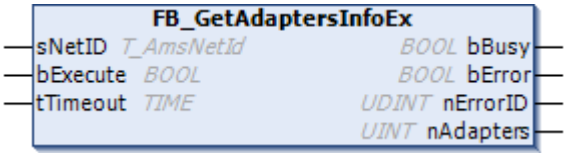
```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrID      : UDINT;
    arrAdapters : ARRAY[0..MAX_LOCAL_ADAPTERS] OF ST_IpAdapterInfo;
    nCount      : UDINT;
    nGet        : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
arrAdapters	ARRAY OF ST_IpAdapterInfo [▶ 337]	包含最近读取的适配器信息的数组变量。每个数组元素会返回一个适配器的信息。
nCount	UDINT	找到的本地适配器的最大数量。
nGet	UDINT	*arrAdapters* 输出变量中的有效条目数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64）	Tc2_Utilities（系统）

3.26 FB_GetAdaptersInfoEx



该功能块可以用于读取 TwinCAT PC 的适配器信息。读取的适配器信息的最大数量限值为 64。

在调用成功后，读取过程结束，使用 Get() 方法可以复制读取的适配器信息。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要读取其适配器信息的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrorID    : UDINT;
    nAdapters   : UINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
nAdapters	UDINT	找到的本地适配器的数量。已读取它们的适配器信息，使用 Get() 方法可以进行复制。

Get() 方法

在成功调用该功能块后，就会完成读取操作，使用 Get() 方法可以复制读取的适配器信息。

可以将所有适配器的信息一起复制。同样，也可以逐个单独复制每个适配器的信息。

在调用时会指定一个本地数组（类型：ST_IpAdapterInfo [▶ 337]）。因此，每个数组元素会返回一个适配器的信息。

```
METHOD Get : BOOL
VAR_INPUT
    pAdapters         : POINTER TO ST_IpAdapterInfo; // pointer to array of adapter info (variable ar
ray length)
    nAdaptersSize     : UDINT; // size in bytes of array of adapter info
END_VAR
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.48.0

3.27 FB_GetDeviceIdentificationEx

该功能块读取设备 ID。与 FB_GetDeviceIdentification [▶ 26] 相比，允许使用更长的硬件型号和硬件序列号的字符串。

输入

```
VAR_INPUT
    bExecute   : BOOL;
    tTimeout   : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId     : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	在这里可以指定一个字符串，其中包含要读取其设备信息的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy      : BOOL;
    bError     : BOOL;
    nErrorID   : UDINT;
    stDevIdent : ST_DeviceIdentificationEx;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stDevIdent	ST_DeviceIdentificationEx [▶ 334]	返回设备 ID

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.28 FB_GetDongleSystemID

功能块 FB_GetDongleSystemID 将 TwinCAT 3 授权加密狗的系统 ID 和批量 ID 读取为 GUID。

输入

```
VAR_INPUT
    bExecute  : BOOL;
    tTimeout  : TIME;
    sNetId    : T_AmsNetId;
    stAmsAddr : AMSADDR;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。
stAmsAddr	AMSADDR	授权加密狗的网络地址（AmsNetId 和端口）

输出

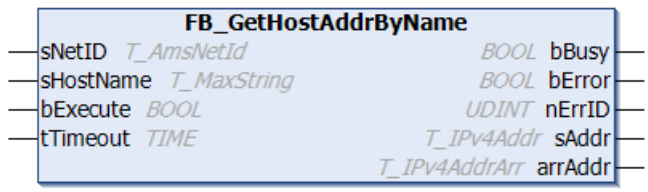
```
VAR_OUTPUT
    bBusy      : BOOL;
    bError     : BOOL;
    nErrorId   : UDINT;
    stSystemID : GUID;
    stVolumeID : GUID;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stSystemID	GUID [▶ 331]	授权加密狗的系统 ID
stVolumeID	GUID [▶ 331]	授权加密狗的批量 ID

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.24.0

3.29 FB_GetHostAddrByName



该功能块可以用于读取指定主机名称的（IPv4）互联网协议网络地址。该地址以字符串和字节数组的形式返回。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetId;
    sHostName : T_MaxString := '';
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定要执行命令的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sHostName	T_MaxString	主机名（以字符串形式表示）。例如：“DataServer1”。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy    : BOOL;
    bError   : BOOL;
    nErrID   : UDINT;
    sAddr    : T_IPv4Addr := '';
    arrAddr  : T_IPv4AddrArr :=[ 0, 0, 0, 0];
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
sAddr	T_Ipv4AddrArr	IPv4互联网协议网络地址（以字符串形式表示）。例如：“172.16.7.199”
arrAddr	T_Ipv4AddrArr	互联网协议网络地址（以字节数组形式表示）

示例：

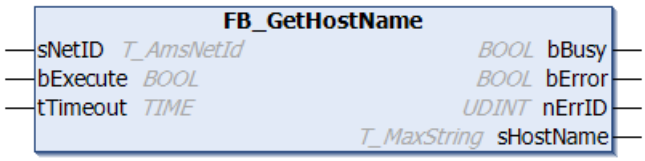
```
PROGRAM MAIN
VAR
    fbGet    : FB_GetHostAddrByName;
    bGet     : BOOL := TRUE;
    bError   : BOOL;
    nErrID   : UDINT;
    sIPv4    : T_IPv4Addr;(* Result: '87.106.8.100' *)
    arrIPv4  : T_IPv4AddrArr;
    state    : BYTE;
END_VAR
```

```
CASE state OF
0:
    IF bGet THEN
        bGet := FALSE;
        sIPv4 := '';
        fbGet( bExecute:= FALSE );
        fbGet( bExecute:= TRUE, sHostName := 'www.beckhoff.com' );
        state := 1;
    END_IF
1:
    fbGet( bExecute:= FALSE, bError=>bError, nErrID=>nErrID, sAddr=>sIPv4, arrAddr=>arrIPv4 );
    IF NOT fbGet.bBusy THEN
        state := 0;
    END_IF
END_CASE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7）	Tc2_Utilities（系统）

3.30 FB_GetHostName



该功能块可以用于读取 TwinCAT PC 的主机名称。

输入

```
VAR_INPUT
    sNetID   : T_AmsNetId;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	此处可指定一个字符串，其中包含待读取主机名的TwinCAT计算机的网络地址。对于本地计算机，可以指定一个空字符串。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy     : BOOL;
    bError    : BOOL;
    nErrID    : UDINT;
    sHostName : T_MaxString;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
sHostName	T_MaxString	主机名（字符串形式）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.31 FB_GetLicenseDongles

该功能块可确定连接的授权加密狗的数量，并返回地址和状态。

输入

```
VAR_INPUT
    bExecute  : BOOL;
    tTimeout  : TIME;
    sNetId    : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrorId : UDINT;
    nLicenseDeviceDongles : UDINT;
    aLicenseDeviceDongles : ARRAY[1..nMaxLicenseDevices] OF ST_LicenseDongle;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
nLicenseDeviceDongles	UDINT	授权加密狗的数量
aLicenseDeviceDongles	ARRAY OF ST_LicenseDongle [▶ 338]	连接的授权加密狗的识别数据

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.24.0

3.32 FB_GetLicenses

该功能块可读取有效和无效的 TwinCAT 授权。

输入

```
VAR_INPUT
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId    : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。

名称	类型	描述
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy           : BOOL;
    bError          : BOOL;
    nErrorId        : UDINT;
    nValidLicenses  : UDINT;
    aValidLicenses  : ARRAY [1..nMaxLicenses] OF ST_TcOnlineLicenseInfoData;
    nInvalidLicenses : UDINT;
    aInvalidLicenses : ARRAY [1..nMaxLicenses] OF ST_TcOnlineLicenseInfoData;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
nValidLicenses	UDINT	返回有效的 TwinCAT 授权的数量。
aValidLicenses	ARRAY OF ST_TcOnlineLicenseInfodata [▶ 341]	返回有效的 TwinCAT 授权的列表。
nInvalidLicenses	UDINT	返回无效的 TwinCAT 授权的数量。
aInvalidLicenses	ARRAY OF ST_TcOnlineLicenseInfodata [▶ 341]	返回无效的 TwinCAT 授权的列表。

默认情况下，授权列表条目的最大数量为 50。通过 nMaxLicenses 在库的参数列表中可以更改该限值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4018	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）v3.3.9.0 或更高

3.33 FB_GetLicensesEx

功能块 FB_GetLicensesEx 可确定所有 TwinCAT 3 授权和 OEM 授权的状态。

输入

```
VAR_INPUT
    bExecute  : BOOL;
    tTimeout  : TIME;
    sNetId    : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy             : BOOL;
    bError            : BOOL;
    nErrorId          : UDINT;
    nValidLicenses    : UDINT
    aValidLicenses    : ARRAY[1..nMaxLicenses} OF ST_TcOnlineLicenseInfoDataEx
    nPendingLicenses  : UDINT
    aPendingLicenses  : ARRAY[1..nMaxLicenses} OF ST_TcOnlineLicenseInfoDataEx
    nDemoLicenses     : UDINT
    aDemoLicenses     : ARRAY[1..nMaxLicenses} OF ST_TcOnlineLicenseInfoDataEx
    nOemLicenses      : UDINT
    aOemLicenses      : ARRAY[1..nMaxLicenses} OF ST_TcOnlineLicenseInfoDataEx
    nFailedLicenses   : UDINT
    aFailedLicenses   : ARRAY[1..nMaxLicenses} OF ST_TcOnlineLicenseInfoDataEx
    nInvalidLicenses  : UDINT
    aInvalidLicenses  : ARRAY[1..nMaxLicenses} OF ST_TcOnlineLicenseInfoDataEx
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
nValidLicenses	UDINT	返回有效授权的数量。
aValidLicenses	ARRAY OF ST_TcOnlineLicenseInfoDataEx [▶ 340]	返回有效授权的列表。
nPendingLicenses	UDINT	开放授权的数量
aPendingLicenses	ARRAY OF ST_TcOnlineLicenseInfoDataEx [▶ 340]	待处理授权的信息
nDemoLicenses	UDINT	有效的展示授权的数量
aDemoLicensese	ARRAY OF ST_TcOnlineLicenseInfoDataEx [▶ 340]	关于有效的演示授权的信息
nOemLicenses	UDINT	有效的 OEM 授权的数量
aOemLicenses	ARRAY OF ST_TcOnlineLicenseInfoDataEx [▶ 340]	关于有效的 OEM 授权的信息
nFailedLicenses	UDINT	失效授权的数量
aFailedLicenses	ARRAY OF ST_TcOnlineLicenseInfoDataEx [▶ 340]	关于有效授权的信息
nInvalidLicenses	UDINT	返回无效的 TwinCAT 授权的数量。

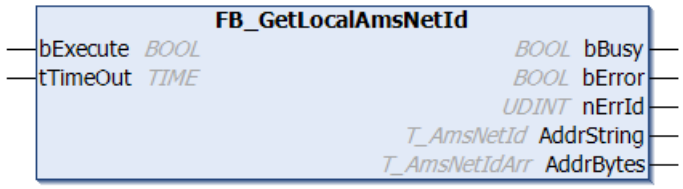
名称	类型	描述
aInvalidLicenses	ARRAY OF ST_TcOnlineLicenseInfo data [▶ 341]	返回无效的 TwinCAT 授权的列表。

默认情况下，授权列表条目的最大数量为 50。通过 nMaxLicenses 在库的参数列表中可以更改该限值。

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.24.0

3.34 FB_GetLocalAmsNetId



该功能块可以用于读取本地 TwinCAT PC 的网络地址（AmsNetID）。

输入

```
VAR_INPUT
    bExecute  :BOOL;
    tTimeOut  :TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy       :BOOL;
    bError      :BOOL;
    nErrId      :UDINT;
    AddrString  :T_AmsNetId;
    AddrBytes   :T_AmsNetIdArr;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
AddrString	T_AmsNetID	本地PC的AmsNetID（以字符串形式表示）
AddrBytes	T_AmsNetIDArr	本地PC的AmsNetID（以字节数组形式表示）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.35 FB_GetOemOfLicenseId

功能块 FB_GetOemLicenseId 可返回给定授权 ID 的 OEM ID。

输入

```
VAR_INPUT
    bExecute      : BOOL;
    tTimeout      : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId        : T_AmsNetId;
    stLicenseId   : GUID;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其授权状态的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。
stLicenseId	GUID [▶ 331]	授权 ID

输出

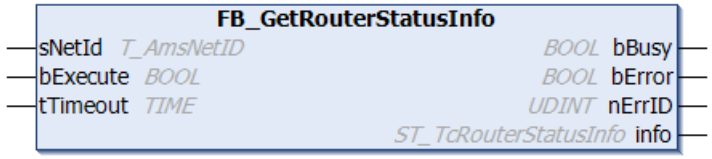
```
VAR_OUTPUT
    bBusy         : BOOL;
    bError        : BOOL;
    nErrId        : UDINT
    stOemId       : GUID;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stOemId	GUID [▶ 331]	OEM ID

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.48.0

3.36 FB_GetRouterStatusInfo



功能块 FB_GetRouterStatusInfo 可以用于从 PLC 中读取 TwinCAT 路由器的状态信息（可用内存、注册端口号等）。

输入

```
VAR_INPUT
    sNetID   : T_AmsNetId;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要读取其 TwinCAT 路由器信息的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy       : BOOL;
    bError      : BOOL;
    nErrID      : UDINT;
    info        : ST_TcRouterStatusInfo;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
info	ST_TcRouterStatusInfo [▶ 341]	带有 TwinCAT 路由器状态信息的结构变量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.37 FB_GetSystemId

该功能块可读取 GUID 类型的系统 ID（参见‘关于TwinCAT...’） TwinCAT 图标上的“关于 TwinCAT...”）。

输入

```
VAR_INPUT
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
    sNetId    : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	TwinCAT 计算机的 AmsNetId（AMS 网络标识符），要读取的系统 ID。对于本地计算机，可以指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy      : BOOL;
    bError     : BOOL;
    nErrorId   : UDINT;
    stSystemId : GUID;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stSystemId	GUID [▶ 331]	返回系统 ID。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.38 FB_GetTimeZoneInformation

该功能块可以用于读取操作系统的时区设置。

输入

```
VAR_INPUT
    sNetID   : T_AmsNetID;
    bExecute : BOOL;
    tTimeout : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	要读取其时区设置的 TwinCAT 计算机的 AmsNetId（AmsNetId）。对于本地计算机，可以指定一个空字符串。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
    nErrID : UDINT;
    tzID   : E_TimeZoneID;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。

名称	类型	描述
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
tzID	E_TimeZoneID [▶ 329]	附加的夏令时/冬令时信息（不一定存在）
tzInfo	ST_TimeZoneInformation [▶ 342]	如果成功，则该结构变量将返回操作系统的当前时区信息。

示例：

请参见 FB_SetTimeZoneInformation [▶ 109] 功能块的相关描述。

更多与时间和时区相关的功能及功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeToTzSpecificLocalTime [▶ 27]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7）	Tc2_Utilities（系统）

3.39 FB_GetVolumeId

功能块 FB_GetVolumeId 将系统 ID 和卷的系统读取为 GUID。

输入

```
VAR_INPUT
    bExecute  : BOOL;
    tTimeout  : TIME;
    sNetId    : T_AmsNetId;
END_VAR
```

名称	类型	描述
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。
sNetId	T_AmsNetID	要读取其系统 ID 的 TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。

输出

```
VAR_OUTPUT
    bBusy       :   BOOL;
    bError      :   BOOL;
    nErrorId    :   UDINT;
    stVolumeId  :   GUID;
    stSystemId  :   GUID;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
stVolumeId	GUID [▶ 331]	返回卷系统 ID
stSystemId	GUID [▶ 331]	返回系统 ID

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4018	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities

3.40 FB_HashTableCtrl

哈希表可以用于在较大数量的数据元素中快速查找单个数据元素。数据对象必须具有唯一的键。通过键可以在表格中明确识别并快速找到数据对象。

功能块 FB_HashTableCtrl 可以用于在 PLC 项目中实现一个简单哈希表。哈希表采用带链表（分离链表法）的散列法。

在运行时不能更改数据元素的最大数量，必须提前指定。通过操作调用来控制数据元素的添加/删除/查找。该功能块包含以下核心功能：

- **A_Add**向表中添加新的数据元素（键/值）。如果一个具有相同键的元素已经存在，则会将其覆盖！）
- **A_GetFirst**（读取第一个表数据元素。如果成功，则 *getValue* 将提供相关值。）
- **A_GetNext**（读取下一个表数据元素。地址：*putPosPtr* 必须指向上一个数据元素！）
- **A_Lookup**（查找与键匹配的数据元素。如果成功，则 *getValue* 将提供相关值。）
- **A_Remove**（移除与键匹配的数据元素。）
- **A_RemoveAll**（移除所有数据元素）
- **A_RemoveFirst**（移除第一个数据元素）
- **A_reset**（删除所有数据元素并重置表。）
- **A_GetIndexAtPosPtr**（返回地址为：*putPosPtr* 的数据元素的数组索引。成功后，*getValue* 将返回基于空值的数组索引。不使用 *putValue* 值。请注意，*getValue* 值将返回数据元素索引，而不是数据元素值！）

输入

```
VAR_INPUT
    key       : DWORD := 0;
    putValue  : PVOID := 0;
    putPosPtr : POINTER TO T_HashTableEntry := 0;
END_VAR
```

名称	类型	描述
key	DWORD	键（无符号 32 位数字或指针）。通过该键可以在表格中快速识别并找到数据对象。
putValue	PVOID	值/数据元素（输入参数，32/64 位，无符号数字或指针）。
putPosPtr	T_HashTableEntry [▶ 345]	数据元素的地址（输入参数）

输入/输出

```
VAR_IN_OUT
    hTable  : T_HHASHTABLE;
END_VAR
```

名称	类型	描述
hTable	T_HHASHTABLE [▶ 345]	哈希表句柄。在使用句柄之前，必须使用函数：F_CreateHashTableHnd [▶ 283] 对其进行一次初始化。必须为每个哈希表创建句柄变量的相应实例并对其进行初始化。

输出

```
VAR_OUTPUT
    bOk         : BOOL := FALSE;
    getValue    : PVOID := 0;
    getPosPtr   : POINTER TO T_HashTableEntry := 0;
END_VAR
```

名称	类型	描述
bOk	BOOL	如果在表中添加/移除或找到一个新的数据元素，则返回 TRUE。如果未找到搜索的数据元素、表为空或发生溢出（没有更多空闲数据元素），则返回 FALSE。
getValue	PVOID	与键/数据元素匹配的值（输出参数，32/64 位，无符号数字或指针）。
getPosPtr	T_HashTableEntry [▶ 345]	数据元素的地址（输出参数）。

示例：

请参见：示例：哈希表（FB_HashTableCtrl）。 [▶ 361]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.41 FB_LicFileCopyFromDongle

该功能块将文件从授权狗复制到硬盘。如果文件比缓冲区大（cbCopyLen），则自动将文件复制步骤拆分成多次读写操作，直到整个文件被复制为止。只有这样，bBusy 才切换到 FALSE。

输入

```
VAR_INPUT
    sNetIdSrc          : T_AmsNetId;
    nPortSrc           : UINT;
    sNetIdDest         : T_AmsNetId;
    sFileNameSrc       : STRING;
    sFilePathNameDest  : T_MaxString;
    pCopyBuff          : PVOID;
    cbCopyLen          : UDINT;
    bExecute           : BOOL;
    dwPassCode         : DWORD;
    tTimeout           : TIME    := DEFAULT_ADS_TIMEOUT
END_VAR
```

名称	类型	描述
sNetIdSrc	T_AmsNetId	授权狗的 AmsNetId（AMS 网络 ID）。 • USB 加密狗：TwinCAT 计算机的 AmsNetId。对于本地计算机，可以指定一个空字符串。 • EL6070：EtherCAT 主站的 AmsNetId（请参见 EL6070 的 InfoData 中的 AdsAddr.netId）
nPortSrc	UINT	授权加密狗的 AMS 端口 • USB：ESB 设备的 ADS 端口（请参见 USB 加密狗的 ESB 设备选项卡上的 ADS 端口；默认为 16#7100） • EL6070：EtherCAT 端子模块的 ADS 端口（请参见 EL6070 的 InfoData 中的 AdsAddr.port）
sNetIdDest	T_AmsNetId	TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。
sFileNameSrc	STRING	授权加密狗上的文件的名称
sFilePathNameScr	T_MaxString	硬盘上的文件的路径名称
pCopyBuff	PVOID	用于写入的缓冲区地址
cbCopyLen	UDINT	用于写入的字节数
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
dwPassCode	DWORD	文件访问的密码
tTimeout	TIME	在执行命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    bBusy       :   BOOL;
    bError      :   BOOL;
    nErrorId    :   UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.26.0

3.42 FB_LicFileCopyToDongle

该功能块将文件从硬盘复制到授权加密狗。如果文件比缓冲区大（cbCopyLen），则自动将文件复制步骤拆分成多次读写操作，直到整个文件被复制为止。只有这样，bBusy 才切换到 FALSE。

输入

```
VAR_INPUT
    sNetIdSrc          : T_AmsNetId;
    sNetIdDest         : T_AmsNetId;
    nPortDest          : UINT;
    sFilePathNameSrc   : T_MaxString;
    sFileNameDest      : STRING;
    pCopyBuff          : PVOID;
    cbCopyLen          : UDINT;
    bExecute           : BOOL;
    dwPassCode         : DWORD;
    tTimeout           : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetIdSrc	T_AmsNetId	TwinCAT 计算机的 AmsNetId（AMS 网络标识符）。对于本地计算机，可以指定一个空字符串。
sNetIdDest	T_AmsNetId	授权加密狗的 AmsNetId（AMS 网络 ID） • USB 加密狗：TwinCAT 计算机的 AmsNetId。对于本地计算机，可以指定一个空字符串 • EL6070：EtherCAT 主站的 AmsNetId（请参见 EL6070 的 InfoData 中的 AdsAddr.netId）
nPortDest	UINT	授权加密狗的 AMS 端口 • USB：ESB 设备的 ADS 端口（请参见 USB 加密狗的 ESB 设备选项卡上的 ADS 端口；默认为 16#7100） • EL6070：EtherCAT 端子模块的 ADS 端口（请参见 EL6070 的 InfoData 中的 AdsAddr.port）
sFilePathNameScr	T_MaxString	硬盘上的文件的路径名称
sFileNameDest	STRING	授权加密狗上的文件的名称
pCopyBuff	PVOID	用于写入的缓冲区地址
cbCopyLen	UDINT	用于写入的字节数
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
dwPassCode	DWORD	文件访问的密码
tTimeout	TIME	在执行命令时不得超过的超时时间。

输出

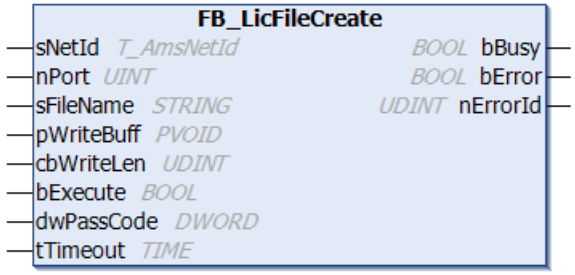
```
VAR_OUTPUT
    bBusy       :   BOOL;
    bError      :   BOOL;
    nErrorId    :   UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.26.0

3.43 FB_LicFileCreate



功能块可在授权加密狗上创建文件。bExecute 的上升沿会将缓冲区中的数据（pWriteBuff 和 cbWriteLen）直接写入加密狗的新文件中。

输入

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nPort       : UINT;
    sFileName   : STRING;
    pWriteBuff  : PVOID;
    cbWriteLen  : UDINT;
    bExecute    : BOOL;
    dwPassCode  : DWORD;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetId	授权加密狗的 AmsNetId（AMS 网络 ID） • USB 加密狗：TwinCAT 计算机的 AmsNetId。对于本地计算机，可以指定一个空字符串。 • EL6070：EtherCAT 主站的 AmsNetId（请参见 EL6070 的 InfoData 中的 AdsAddr.netId）
nPort	UINT	授权加密狗的 AMS 端口 • USB：ESB 设备的 ADS 端口（请参见 USB 加密狗的 ESB 设备选项卡上的 ADS 端口；默认为 16#7100） • EL6070：EtherCAT 端子模块的 ADS 端口（请参见 EL6070 的 InfoData 中的 AdsAddr.port）
sFileName	STRING	要创建的文件的名称
pWriteBuff	PVOID	用于写入的缓冲区地址
cbWriteLen	UDINT	用于写入的字节数
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
dwPassCode	DWORD	文件访问的密码
tTimeout	TIME	在执行命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    bBusy       :   BOOL;
    bError      :   BOOL;
    nErrorId    :   UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.26.0

3.44 FB_LicFileDelete

功能块可从授权加密狗上删除文件。文件名和文件长度归零，要删除的文件数据字节在加密狗中被释放，但不覆盖。

输入

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nPort       : UINT;
    sFileName   : STRING;
    bExecute    : BOOL;
    dwPassCode  : DWORD;
    tTimeout    : TIME      := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetId	授权加密狗的 AmsNetId（AMS 网络 ID） • USB 加密狗：TwinCAT 计算机的 AmsNetId。对于本地计算机，可以指定一个空字符串。 • EL6070：EtherCAT 主站的 AmsNetId（请参见 EL6070 的 InfoData 中的 AdsAddr.netId）
nPort	UINT	授权加密狗的 AMS 端口 • USB：ESB 设备的 ADS 端口（请参见 USB 加密狗的 ESB 设备选项卡上的 ADS 端口；默认为 16#7100） • EL6070：EtherCAT 端子模块的 ADS 端口（请参见 EL6070 的 InfoData 中的 AdsAddr.port）
sFileName	STRING	要删除的文件的名称
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
dwPassCode	DWORD	文件访问的密码
tTimeout	TIME	在执行命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    bBusy       :   BOOL;
    bError      :   BOOL;
    nErrorId    :   UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.26.0

3.45 FB_LicFileGetStorageInfo

该功能块读取授权加密狗的存储信息和文件目录。

存储信息包括数据载体的管理数据（例如，容量、可用字节数、文件数目……）和一个含各文件条目的数组（文件的名称、大小、属性……）。

输入

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nPort       : UINT;
    bExecute    : BOOL;
    dwPassCode  : DWORD;
    tTimeout    : TIME       := DEFAULT_ADS_TIMEOUT
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetId	授权加密狗的 AmsNetId（AMS 网络 ID） • USB 加密狗：TwinCAT 计算机的 AmsNetId。对于本地计算机，可以指定一个空字符串。 • EL6070：EtherCAT 主站的 AmsNetId（请参见 EL6070 的 InfoData 中的 AdsAddr.netId）
nPort	UINT	授权加密狗的 AMS 端口 • USB：ESB 设备的 ADS 端口（请参见 USB 加密狗的 ESB 设备选项卡上的 ADS 端口；默认为 16#7100） • EL6070：EtherCAT 端子模块的 ADS 端口（请参见 EL6070 的 InfoData 中的 AdsAddr.port）
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
dwPassCode	DWORD	文件访问的密码（仅适用于受特殊保护的文件）
tTimeout	TIME	在执行命令时不得超过的超时时间。

输入/输出

```
VAR_IN_OUT
    stStorageInfo : ST_LicStorageInfo;
END_VAR
```

名称	类型	描述
stStorageInfo	ST_LicStorageInfo	授权加密狗的存储信息

输出

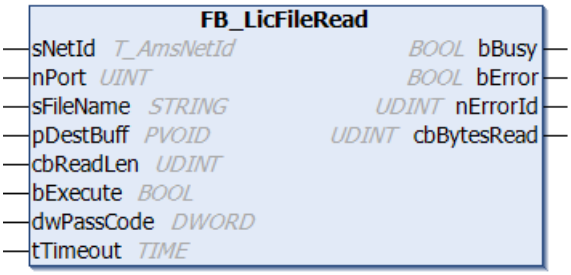
```
VAR_OUTPUT
    bBusy        : BOOL;
    bError       : BOOL;
    nErrorId     : UDINT;
    nFileEntries : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
nFileEntries	UDINT	授权加密狗上的文件的数量

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.26.0

3.46 FB_LicFileRead



该功能块可通过 bExecute 的上升沿将授权加密狗上的文件读取至所提供的缓冲区（pDestBuff 和 cbReadLen）。缓冲区必须足够大，否则只读取文件的第一部分。

输入

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    nPort       : UINT;
    sFileName   : STRING;
    pDestBuff   : PVOID;
    cbReadLen   : UDINT
    bExecute    : BOOL;
    dwPassCode  : DWORD;
    tTimeout    : TIME       := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetId	授权加密狗的 AmsNetId（AMS 网络 ID） • USB 加密狗：TwinCAT 计算机的 AmsNetId。对于本地计算机，可以指定一个空字符串。

名称	类型	描述
		• EL6070：EtherCAT 主站的 AmsNetId（请参见 EL6070 的 InfoData 中的 AdsAddr.netId）
nPort	UINT	授权加密狗的 AMS 端口 • USB：ESB 设备的 ADS 端口（请参见 USB 加密狗的 ESB 设备选项卡上的 ADS 端口；默认为 16#7100） • EL6070：EtherCAT 端子模块的 ADS 端口（请参见 EL6070 的 InfoData 中的 AdsAddr.port）
sFileName	STRING	要读取的文件的名称
pDestBuff	PVOID	用于读取的缓冲区地址
cbReadLen	UDINT	用于读取的字节数
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
dwPassCode	DWORD	文件访问的密码
tTimeout	TIME	在执行命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    bBusy        : BOOL;
    bError       : BOOL;
    nErrorId     : UDINT;
    cbBytesread  : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrorId	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。
cbBytesRead	UDINT	

要求

开发环境	目标平台	待集成的 PLC 库
TwinCAT v3.1.4022	PC 或 CX（x64、x86）	Tc2_Utilties >= 3.3.26.0

3.47 FB_LinkedListCtrl

功能块 FB_LinkedListCtrl 可以用于在PLC中构建链表。创建一个双向链表。链表允许存储值（被称为节点）。可以从后向前或从前向后遍历列表。可以快速添加或删除节点。

在运行时无法更改最大节点数；必须在编译前指定。类型的数组：*T_LinkedListEntry* 可用作“节点池”。通过调用来实现节点的添加/移除/查找操作。该功能块包含以下核心功能：

- **A_AddHeadValue**（在链表头部插入新的节点（节点值为 *putValue*）。允许重复值。如果成功，则 *getPosPtr* 将返回新节点地址，而 *getValue* 将返回新节点的值。）
- **A_AddTailValue**在链表尾部追加新节点 （节点值为*putValue*）。允许重复值。如果成功，则 *getPosPtr* 将返回新节点地址，而 *getValue* 将返回新节点的值。）
- **A_FindNext**（从指定位置 *putPosPtr*起，向后查找下一个值与*putValue* 匹配的节点 如果成功，则 *getPosPtr* 将返回地址，而 *getValue* 将返回节点的值。）

- **A_FindPrev**从指定位置 *putPosPtr*）起，向前查找上一个值与 *putValue* 匹配的节点（ putPosPtr）起。如果成功，则 *getPosPtr* 将返回地址，而 *getValue* 将返回节点的值。）
- **A_GetNext**（（从指定位置 *putPosPtr*）起，向后遍历至下一个节点。地址：*putPosPtr* 必须指向上一个节点！ *putValue* 值无效）。
- **A_GetPrev**（从指定位置（ *putPosPtr*）起，沿与 *A_GetNext* 相反的方向遍历至上一个节点。地址：*putPosPtr* 必须指向上一个节点！ *putValue* 值无效）。
- **A_GetHead**（读取起始节点。如果成功，则 *getPosPtr* 将返回节点的地址，而 *getValue* 将返回相关的值。 *putValue* 和 *putPosPtr* 值在此操作中未被使用。）
- **A_GetTail**（读取最终节点。如果成功，则 *getPosPtr* 将返回节点的地址，而 *getValue* 将返回相关的值。 *putValue* 和 *putPosPtr* 值在此操作中未被使用。）
- **A_RemoveHeadValue**（从链表头部移除一个节点。如果成功，则 *getPosPtr* 将返回地址，而 *getValue* 将返回节点的值。 *putValue* 和 *putPosPtr* 值在此操作中未被使用。）
- **A_RemoveTailValue**（从链表尾部移除一个节点。如果成功，则 *getPosPtr* 将返回地址，而 *getValue* 将返回节点的值。 *putValue* 和 *putPosPtr* 值在此操作中未被使用。）
- **A_RemoveValueAtPosPtr**（搜索并移除地址为 *putPosPtr* 的节点。如果成功，则 *getPosPtr* 将返回地址，而 *getValue* 将返回节点的值。 *putValue* 值无效）。
- **A_GetIndexAtPosPtr**（返回地址为 *putPosPtr* 的节点的数组索引（来自“节点池”）。成功后，*getValue* 将返回基于空值的数组索引。不使用 *putValue* 值。请注意，getValue 值是一个节点索引，而不是节点的值！）
- **A_SetValueAtPosPtr**（使用地址为 *putPosPtr* 的 *putValue* 更新/设置节点的值。如果成功，则 *getPosPtr* 将返回地址，而 *getValue* 将返回节点的值。）
- **A_Reset**（删除链表中的所有元素并重置链表状态。）

输入

```
VAR_INPUT
    putValue   : PVOID := 0;
    putPosPtr  : POINTER TO T_LinkedListEntry := 0;
END_VAR
```

名称	类型	描述
putValue	PVOID	值/数据元素（输入参数，32/64 位，无符号数字或指针）。
putPosPtr	T_LinkedListEntry [▶ 347]	节点元素的地址（输入参数）。

输入/输出

```
VAR_IN_OUT
    hList  : T_HLINKEDLIST;
END_VAR
```

名称	类型	描述
hList	T_HLINKEDLIST [▶ 346]	链表句柄。在使用句柄之前，必须使用函数 F_CreateLinkedListHnd [▶ 284] 对其进行一次初始化。必须为每个链表创建句柄变量的相关实例并对其进行初始化。

输出

```
VAR_OUTPUT
    bOk       : BOOL := FALSE;
    getValue  : PVOID := 0;
    getPosPtr : POINTER TO T_LinkedListEntry := 0;
END_VAR
```

名称	类型	描述
bOk	BOOL	最后调用的操作的结果。如果可以在列表中添加、移除或找到一个新的节点元素，则返回 TRUE。如果不能找到搜索的节点元素、列表为空或已溢出（没有更多空闲节点元素），则返回 FALSE。
getValue	PVOID	值/数据元素（输出参数，32/64 位，无符号数字或指针）。
getPosPtr	T_LinkedListEntry [▶ 347]	节点元素的地址（输出参数）

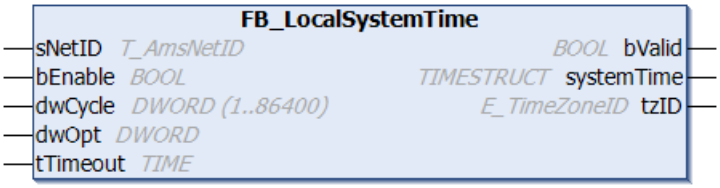
示例：

请参见：示例：链表（FB_LinkedListCtrl）。 [▶ 364]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.48 FB_LocalSystemTime



在一些应用程序中，通过 SNTP 时间服务器或无线电时钟可以同步本地 Windows 系统时间。在许多情况下，在 PLC 中必须使用本地 Windows 系统时间（例如，以时间戳日志消息的形式发送到 HMI）。在任务栏中可显示本地 Windows 系统时间。对于此类应用程序，FB_LocalSystemTime 功能块可能十分有用。

该功能块在内部结合了以下功能块的功能：RTC_EX2 [▶ 137]、NT_GetTime [▶ 120]、FB_GetTimeZoneInformation [▶ 78] 和 NT_SetTimeToRTCTime [▶ 123]。例如，RTC_EX2 功能块可以用于为日志输出生成时间戳。不过，该功能块的缺点是其时间与本地 Windows 系统时间不同步，必须通过 NT_GetTime 功能块周期性地重新同步（请参见文档中的 RTC 功能块示例）。在该功能块中已实现内部时间的循环同步（*systemTime* 输出）。通过 *dwCycle* 输入可以对循环时间进行配置。该功能块还可提供时区信息（夏令时/冬令时）。

必须周期性地调用 FB_LocalSystemTime 功能块（例如，每秒或在每个 PLC 周期中）。这对于计算同步之间的时间必不可少。

抖动！

借助非周期性服务（ADS 功能块）读取本地 Windows 系统时间。由于系统特性，无法指定/估计 ADS 命令的运行时。命令运行时中的差异可能会导致 systemTime 输出的时间抖动，具体取决于操作系统、同步间隔以及 PLC 周期时间。因此，功能块提供的时间仅在有限程度上适用于更精确的测量任务。不过，其精度足以满足楼宇自动化等领域的应用需求。

夏令时与标准时间之间的切换

在从夏令时切换到标准时间以及从标准时间切换到夏令时之时，不能调用该功能块。为了避免复杂的计算，我们已选择以下实现方法（在示例中解释）。

在我们的示例中，功能块每 60 秒将自身时间与本地 Windows 系统时间（灰色）进行一次同步。
PLC 应用程序需要并读取功能块中的时间（例如，每 30 秒一次，蓝色）。在我们的示例中，夏令时与标准时间之间的切换检测存在15秒的延迟。对于大多数应用程序来说，这种行为应该没有问题。

从标准时间切换到夏令时

- ...
- 30-03-2008-01:58:10，tzID = 标准时间
- 30-03-2008-01:58:15，内部同步后
- 30-03-2008-01:58:40，tzID = 标准时间
- 30-03-2008-01:59:10，tzID = 标准时间
- 30-03-2008-01:59:15，内部同步后
- 30-03-2008-01:59:40，tzID = 标准时间
- 30-03-2008-02:00:00，操作系统将时间从凌晨 2 点改为凌晨 3 点
- 30-03-2008-02:00:10，tzID = 标准时间（仍然如此！）
- 30-03-2008-03:00:15，内部同步后，随后的 tzID = 夏令时
- 30-03-2008-03:00:40，tzID = 夏令时
- 30-03-2008-03:01:10，tzID = 夏令时
- 30-03-2008-03:01:15，内部同步后
- 30-03-2008-03:01:40，tzID = 夏令时
- ...

从夏令时切换到标准时间

- ...
- 26-10-2008-02:58:10，tzID = 夏令时
- 26-10-2008-02:58:15，内部同步后
- 26-10-2008-02:58:40，tzID = 夏令时
- 26-10-2008-02:59:10，tzID = 夏令时
- 26-10-2008-02:59:15，内部同步后
- 26-10-2008-02:59:40，tzID = 夏令时
- 26-10-2008-03:00:00，操作系统将时间从凌晨 3 点改为凌晨 2 点
- 26-10-2008-03:00:10，tzID = 夏令时（仍然如此！）
- 26-10-2008-02:00:15，内部同步后，随后的 tzID = 标准时间
- 26-10-2008-02:00:40，tzID = 标准时间
- 26-10-2008-02:01:10，tzID = 标准时间
- 26-10-2008-02:01:15，内部同步后
- 26-10-2008-02:01:40，tzID = 标准时间
- ...

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID := '';
    bEnable   : BOOL;
    dwCycle   : DWORD(1..86400) := 5;
    dwOpt     : DWORD := 1;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	此处可以指定一个包含TwinCAT计算机网络地址的字符串。对于本地计算机，可以指定一个空字符串。

名称	类型	描述
bEnable	BOOL	该输入的上升沿会触发内部时间与本地 Windows 系统时间的立即同步。输出 *bValid* 仍被设置为 FALSE，直到同步完成。第一个上升沿可激活周期性同步功能。随后的周期性同步将自动执行。在大多数情况下，应用程序只需将该输入设置为 TRUE 一次。
dwCycle	DWORD	功能块重新同步自身时间的循环时间（以秒为单位）。在 *bEnable* 输入的第一个上升沿后可激活周期性同步功能。默认值：每 5 秒同步一次。
dwOpt	DWORD	其他选项参数。目前可用的参数如下：
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bValid     : BOOL;
    systemTime : TIMESTRUCT;
    tzID       : E_TimeZoneID := eTimeZoneID_Invalid;
END_VAR
```

名称	类型	描述
bValid	BOOL	如果该输出为 FALSE，则 *systemTime* 输出处的时间无效。如果为 TRUE，则时间有效（即至少与本地 Windows 时间同步一次）。
systemTime	TIMESTRUCT [▶ 347]	本地 Windows 系统时间。
tzID	E_TimeZoneID [▶ 329]	时区信息（夏令时、冬令时）

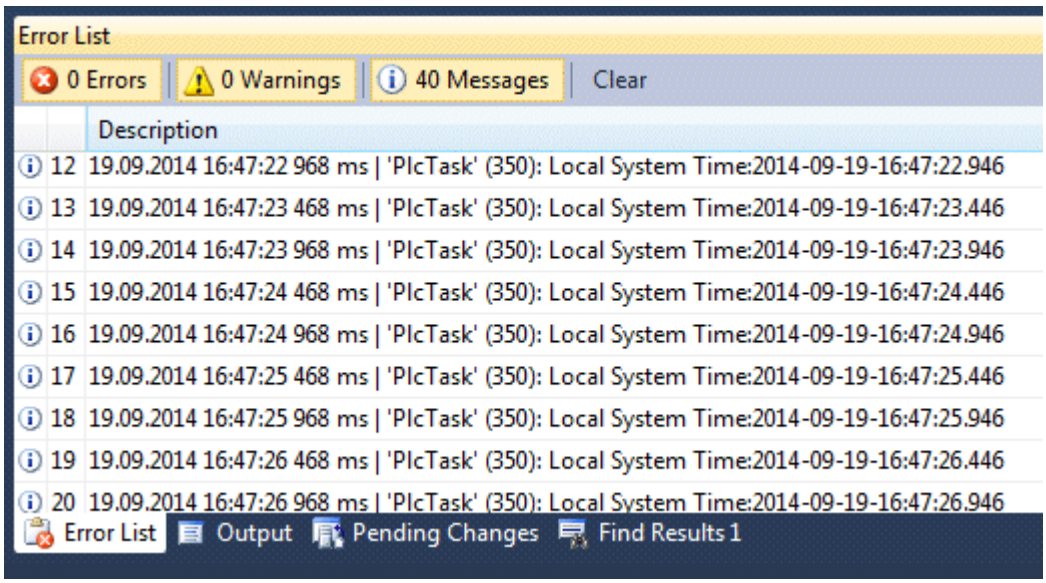
示例：

在示例中，在程序启动时激活 FB_LocalSystemTime 功能块（*bEnable* 输入的上升沿）。一旦时间同步完成（*bValid* = TRUE），PLC 将每 500 ms 向 TwinCAT“错误列表”窗口写入一条消息。每秒执行一次内部同步。

```
PROGRAM MAIN
VAR
    fbTime   : FB_LocalSystemTime := ( bEnable := TRUE, dwCycle := 1 );
    logTimer : TON := ( IN := TRUE, PT := T#500ms );
END_VAR
```

```
fbTime();

logTimer( IN := fbTime.bValid );
IF logTimer.Q THEN
    logTimer( IN := FALSE ); logTimer( IN := fbTime.bValid );
    ADSLOGSTR( ADSLOG_MSGTYPE_HINT OR ADSLOG_MSGTYPE_LOG, 'Local System Time:
%s', SYSTEMTIME_TO_STRING(fbTime.systemTime));
END_IF
```

您可以在 TwinCAT“错误列表”窗口中看到写入的消息。



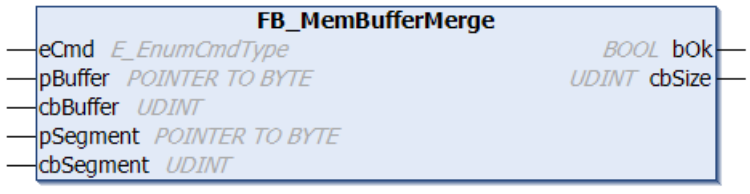
其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.49 FB_MemBufferMerge



该功能块将多个独立的小数据段合并成一个较大的数据段。目标缓冲区必须作为输入参数传递给功能块。如果要添加的数据段超过了剩余的空闲缓冲区大小，则不会添加其他数据字节。

输入

```
VAR_INPUT
    eCmd       : E_EnumCmdType := eEnumCmd_First;
    pBuffer    : POINTER TO BYTE;
    cbBuffer   : UDINT;
```

```
    pSegment  : POINTER TO BYTE := 0;
    cbSegment : UDINT := 0;
END_VAR
```

名称	类型	描述
eCmd	E_EnumCmdType [▶ 325]	枚举模块的控制参数。eEnumCmd_First 可添加第一个数据段，eEnumCmd_Next 可添加下一个数据段。没有使用其他参数。
pBuffer	BYTE	目标缓冲区变量的地址（指针）。使用 ADR 运算符可以确定地址。
cbBuffer	UDINT	目标缓冲区变量的最大可用大小（以字节为单位）。使用 SIZEOF 运算符可以确定大小。
pSegment	BYTE	要添加的下一个数据段的地址（指针）（可选，可能为零）。使用 ADR 运算符也可以确定地址。
cbSegment	UDINT	要添加的下一个数据段的大小（可选，可能为零）。使用 SIZEOF 运算符也可以确定大小。

输出

```
VAR_OUTPUT
    bOk    : BOOL;
    cbSize : UDINT;
END_VAR
```

名称	类型	描述
bOk	BOOL	TRUE = 成功，FALSE = 缓冲区溢出或错误输入参数。
cbSize	UDINT	当前缓冲区填充状态（缓冲区中的数据字节数）。

示例：

在下面的示例中，在合并较小的数据段（出于测试目的）后，将大数据段转换为十六进制字符串。

```
PROGRAM MAIN
VAR
    bMerge    : BOOL := TRUE;
    fbMerge   : FB_MemBufferMerge;
    buffer    : ARRAY[0..25] OF BYTE;
    seg1      : ARRAY[0..5] OF BYTE := [0,1,2,3,4,5];
    seg2      : ARRAY[0..3] OF BYTE := [6,7,8,9];
    seg3      : ARRAY[0..9] OF BYTE := [10,11,12,13,14,15,16,17,18,19];
    sHex      : T_MaxString;
END_VAR
```

```
IF bMerge THEN
    bMerge := FALSE;

    fbMerge( eCmd := eEnumCmd_First, pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pSegment :=
 ADR(seg1), cbSegment:= SIZEOF(seg1) );
    fbMerge( eCmd := eEnumCmd_Next, pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pSegment :=
ADR(seg2), cbSegment:= SIZEOF(seg2 ) );
    fbMerge( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pSegment := ADR(seg3), cbSegment:=
SIZEOF(seg3) );
    fbMerge( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pSegment := 0, cbSegment:= 0 );
(* merge zero length segment *)
    fbMerge( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pSegment := ADR(seg3), cbSegment:=
SIZEOF(seg3) );
    IF NOT fbMerge.bOk THEN
        ;(* TODO: Error handler *)
    END_IF

    sHex := DATA_TO_HEXSTR( pData := ADR(buffer), cbData := fbMerge.cbSize, FALSE );
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.50 FB_MemBufferSplit

该功能块可根据需要将内存区域（数据缓冲区）拆分成多个较小具有特定最大长度的数据段。如果最后一个数据段的长度小于要求的长度，则功能块会返回一个较小的部分数据段。

输入

```
VAR_INPUT
    eCmd     : E_EnumCmdType := eEnumCmd_First;
    pBuffer  : POINTER TO BYTE;
    cbBuffer : UDINT;
    cbSize   : UDINT;
END_VAR
```

名称	类型	描述
eCmd	E_EnumCmdType [▶ 325]	功能块的控制参数。eEnumCmd_First 可返回第一个数据段，eEnumCmd_Next 可返回下一个数据段。没有使用其他参数。
pBuffer	BYTE	要划分的数据缓冲区的地址（指针）。使用 ADR 运算符可以确定地址。
cbBuffer	UDINT	要划分的数据缓冲区的长度。使用 SIZEOF 运算符可以确定长度。
cbSize	UDINT	要划分的数据缓冲区的最大数据段大小。

输出

```
VAR_OUTPUT
    bOk       : BOOL;
    pSegment  : POINTER TO BYTE;
    cbSegment : UDINT;
    bEOS      : BOOL;
END_VAR
```

名称	类型	描述
bOk	BOOL	TRUE = 成功，FALSE = 错误、参数值无效或没有其他数据段可用。
pSegment	BYTE	下一个数据段的地址（指针）。
cbSegment	UDINT	下一个数据段的长度（字节）。
bEOS	BOOL	数据段的末尾。TRUE = 最后一个数据段。FALSE = 后面还有其他数据段。

示例：

在下面的示例中，*缓冲区*变量被划分为 5 字节长度数据段。为了进行测试，将返回的数据段转换为十六进制字符串。

```
PROGRAM MAIN
VAR
    bSplit   : BOOL := TRUE;
    buffer   : ARRAY[1..30] OF BYTE := [16#A,1,2,3,4,5,6,7,8,9,16#B,1,2,3,4,5,6,7,8,9,16#C,1,2,3,4,5
,6,7,8,9];
    fbSplit  : FB_MemBufferSplit;
    sHex     : T_MaxString;
END_VAR
```

```
IF bSplit THEN
    bSplit := FALSE;
    fbSplit.eCmd := eEnumCmd_First;

    REPEAT
        fbSplit( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), cbSize := 5 );
        IF fbSplit.bOk THEN
            sHex := DATA_TO_HEXSTR( pData := fbSplit.pSegment, cbData := fbSplit.cbSegment, FALSE );
            fbSplit.eCmd := eEnumCmd_Next;
```

```
        END_IF
    UNTIL NOT fbSplit.bOk
    END_REPEAT
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.51 FB_MemRingBuffer

功能块 FB_MemRingBuffer 可以用于将不同长度的数据集写入环形缓冲区，或从环形缓冲区中读取先前写入的数据集。根据 FIFO 原则读出已写入的数据集，与先前将它们写入环形缓冲区的顺序相同。这意味着最早写入的数据集（即最旧的数据集）会最先被读取。缓冲区内存通过 *pBuffer*/*cbBuffer* 输入变量提供给功能块使用。通过操作调用控制数据集写入/读取。该功能块包含以下核心功能：

- **A_AddTail**（将新的数据集写入环形缓冲区。）
- **A_GetHead**（读取环形缓冲区文件中最旧的数据集，但不会将其移除。）
- **A_RemoveHead**（从环形缓冲区中读取并移除最旧的数据集。）
- **A_Reset**（删除环形缓冲区中的所有数据集。）

输入

```
VAR_INPUT
    pWrite  : POINTER TO BYTE;
    cbWrite : UDINT;
    pRead   : POINTER TO BYTE;
    cbRead  : UDINT;
    pBuffer : POINTER TO BYTE;
    cbBuffer: UDINT;
END_VAR
```

名称	类型	描述
pWrite	ARRAY OF BYTE	要写入的数值数据所对应的 PLC 变量地址或缓冲区变量地址。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便可以从中获取 *cbWrite* 数据字节。
cbWrite	UDINT	要写入的数据字节数（对于字符串变量，这包括末尾的空字符）。
pRead	BYTE	PLC 变量或缓冲区变量的地址，已读取的值数据将被复制到该地址中。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便其可以接受 cbRead 数据字节。缓冲区变量的大小（以字节为单位）必须大于或等于要读取的数据集的大小。
cbRead	UDINT	要读取的值数据字节数。如果缓冲区大小过小，则不会复制数据。功能块会报告缓冲区下溢错误（bOk = FALSE），并在 *cbReturn* 输出处返回要读取的下一个数据集所需的缓冲区大小。
pBuffer	ARRAY OF BYTE	功能块用作缓冲存储器的 PLC 变量（例如 ARRAY[....] OF BYTES）的地址。使用 ADR 运算符可以确定地址。
cbBuffer	UDINT	要用作缓冲存储器的 PLC 变量的最大字节大小。使用 SIZEOF 运算符可以确定大小。

输出

```
VAR_OUTPUT
    bOk      : BOOL;
    nCount   : UDINT;
    cbSize   : UDINT;
    cbReturn : UDINT;
END_VAR
```

名称	类型	描述
bOk	BOOL	如果成功添加或移除一个新的数据集，则返回 TRUE。如果出现缓冲区溢出或缓冲区中没有更多可用条目，则返回 FALSE。
nCount	UDINT	返回当前缓冲数据集的数量。
cbSize	UDINT	返回缓冲区中当前分配的数据字节数。分配的数据字节数总是大于实际写入值数据的数量。每个数据集都补充了附加信息，以便以后可以找到它。
cbReturn	UDINT	成功读取的值数据字节数。如果发生读取缓冲区下溢错误，则该输出将提供必要的读取缓冲区大小（以字节为单位）。在这种情况下，*cbRead* 长度过小。

示例：

请参见：示例：内存环 FiFo（FB_MemRingBuffer）。 [▶ 359]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.52 FB_MemRingBufferEx

功能块 FB_MemRingBufferEx 可以用于将不同长度的数据集写入环形缓冲区，或从环形缓冲区中读取先前写入的数据集。根据 FIFO 原则读出已写入的数据集，与先前将它们写入环形缓冲区的顺序相同。这意味着最早写入的数据集（即最旧的数据集）会最先被读取。缓冲区内存通过 *pBuffer*/*cbBuffer* 输入变量提供给功能块使用。通过操作调用控制数据集写入/读取。

该功能块的功能与 FB_MemRingBuffer [▶ 96] 功能块类似。在读取数据集期间，FB_MemRingBuffer 会将数据复制到外部缓冲区变量中。FB_MemRingBufferEx 只提供数据集的引用（地址指针/长度）。然后，应用程序必须自行复制数据，以便进行进一步处理。

该功能块包含以下核心功能：

- **A_AddTail**（将新的数据集写入环形缓冲区。）
- **A_GetHead**（返回一个引用：环形缓冲区中最旧的数据集的地址指针/长度，但不会将其移除。）
- **A_FreeHead**（从环形缓冲区中读取并移除最旧的数据集。返回的地址指针/长度为零！为新的数据集释放空闲内存段。）
- **A_Reset**（删除环形缓冲区中的所有数据集。）
- **A_GetFreeSize**（返回缓冲区中最大空闲内存段的字节大小）

输入

```
VAR_INPUT
    pWrite   : POINTER TO BYTE;
    cbWrite  : UDINT;
    pBuffer  : POINTER TO BYTE;
    cbBuffer : UDINT;
END_VAR
```

名称	类型	描述
pWrite	ARRAY OF BYTE	要写入的数值数据所对应的 PLC 变量地址或缓冲区变量地址。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便可以从中获取 *cbWrite* 数据字节。
cbWrite	UDINT	要写入的数据字节数（对于字符串变量，这包括末尾的空字符）。
pBuffer	ARRAY OF BYTE	功能块用作缓冲存储器的 PLC 变量（例如 ARRAY[....] OF BYTES）的地址。使用 ADR 运算符可以确定地址。
cbBuffer	UDINT	要用作缓冲存储器的 PLC 变量的最大字节大小。使用 SIZEOF 运算符可以确定大小。

输出

```
VAR_OUTPUT
    bOk    : BOOL;
    pRead  : POINTER TO BYTE;
    cbRead : UDINT;
    nCount : UDINT;
    cbSize : UDINT;
    cbFree : UDINT;
END_VAR
```

名称	类型	描述
bOk	BOOL	如果成功添加或移除一个新的数据集，则返回 TRUE。如果出现缓冲区溢出或缓冲区中没有更多可用条目，则返回 FALSE。
pRead	BYTE	该变量在操作后返回对环形缓冲区中最旧的数据集的引用（地址指针）：如果成功（bOk=TRUE），则调用 *A_GetHead*。如果环形缓冲区中没有更多可用数据集，则返回零。
cbRead	UDINT	该变量在操作后返回环形缓冲区中最旧数据集的长度：如果成功（bOk=TRUE），则调用 *A_GetHead*。如果环形缓冲区中没有更多可用数据集，则返回零。
nCount	UDINT	返回当前缓冲数据集的数量。
cbSize	UDINT	返回缓冲区中当前分配的数据字节数。分配的数据字节数总是大于实际写入值数据的数量。每个数据集都补充了附加信息，以便以后可以找到它。
cbFree	UDINT	在操作后返回缓冲区中最大空闲内存段的字节大小：调用 *A_GetFreeSize*。数据集必须使用缓冲存储器中的连续地址，因为功能块会返回数据集的引用。这会自动导致缓冲区末端的分段。如果新的数据集大于缓冲区末端的空闲数据段，则无法使用该内存。

示例：

请参见：示例：内存环 FiFo（FB_MemRingBufferEx） [▶ 360]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.53 FB_MemStackBuffer

功能块 FB_MemStackBuffer 可以用于将不同长度的数据集写入缓冲区，或从缓冲区中读取先前写入的数据集。根据 LIFO 原则（后进 – 先出）读取数据集，即与将它们写入缓冲区的顺序相反。换句话说，最先读取最新的条目。通过 *pBuffer* 和 *cbBuffer* 输入变量，缓冲存储器可用于功能块。通过操作调用控制数据集写入/读取。该功能块包含以下核心功能：

- **A_Push()**：将新的数据集写入缓冲区；
- **A_Top()**：从缓冲区中读取最后添加/最新的数据集，但不会将其移除；
- **A_Pop()**：从缓冲区中读取并移除最后添加/最新的数据集；
- **A_Reset()**：删除缓冲区中的所有数据集；

输入

```
VAR_INPUT
    pWrite  : POINTER TO BYTE;
    cbWrite : UDINT;
    pRead   : POINTER TO BYTE;
    cbRead  : UDINT;
    pBuffer : POINTER TO BYTE;
    cbBuffer: UDINT;
END_VAR
```

名称	类型	描述
pWrite	ARRAY OF BYTE	要写入的数值数据所对应的 PLC 变量地址或缓冲区变量地址。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便可以从中获取 *cbWrite* 数据字节。
cbWrite	UDINT	要写入的数据字节数（对于字符串变量，这包括末尾的空字符）。
pRead	BYTE	PLC 变量或缓冲区变量的地址，已读取的值数据将被复制到该地址中。使用 ADR 运算符可以确定地址。程序员自己负责确定缓冲区变量的大小，以便其可以接受 cbRead 数据字节。缓冲区变量的大小（以字节为单位）必须大于或等于要读取的数据集的大小。
cbRead	UDINT	要读取的值数据字节数。如果缓冲区大小过小，则不会复制数据。功能块会报告缓冲区下溢错误（bOk = FALSE），并在 *cbReturn* 输出处返回要读取的下一个数据集所需的缓冲区大小。
pBuffer	ARRAY OF BYTE	功能块用作缓冲存储器的 PLC 变量（例如 ARRAY[....] OF BYTES）的地址。使用 ADR 运算符可以确定地址。
cbBuffer	UDINT	要用作缓冲存储器的 PLC 变量的最大字节大小。使用 SIZEOF 运算符可以确定大小。

输出

```
VAR_OUTPUT
    bOk      : BOOL;
    nCount   : UDINT;
    cbSize   : UDINT;
    cbReturn : UDINT;
END_VAR
```

名称	类型	描述
bOk	BOOL	如果成功添加或移除一个新的数据集，则返回 TRUE。如果出现缓冲区溢出或缓冲区中没有更多可用条目，则返回 FALSE。
nCount	UDINT	返回当前缓冲数据集的数量。
cbSize	UDINT	返回缓冲区中当前分配的数据字节数。分配的数据字节数总是大于实际写入值数据的数量。每个数据集都补充了附加信息，以便以后可以找到它。
cbReturn	UDINT	成功读取的值数据字节数。如果发生读取缓冲区下溢错误，则该输出将提供必要的读取缓冲区大小（以字节为单位）。在这种情况下，*cbRead* 长度过小。

示例：

下面的示例说明了功能块的简单应用。要缓冲的不同长度的字符串。*bReset* 的上升沿会清除缓冲区。如果 *bAdd* 为 TRUE，则会将 10 个新字符串写入缓冲区。如果 *bRemove* 为 TRUE，则会从缓冲区中移除最后写入的字符串。*bGet* 的上升沿会导致读取最后写入的字符串，但不会将其移除。

声明部分：

```
PROGRAM MAIN
VAR
    buffer   : ARRAY[0..1000] OF BYTE;
    fbStack  : FB_MemStackBuffer;
    bReset   : BOOL := TRUE;
    bAdd     : BOOL := TRUE;
    bGet     : BOOL := TRUE;
    bRemove  : BOOL := TRUE;
    putEntry : ARRAY[0..9] OF STRING(20) := ['Str_1', 'Str_2', 'Str_3', 'Str_4', 'Str_5', 'Str_6', '
Str_7', 'Str_8', 'Str_9', 'Str_10'];
    getEntry : STRING;
    i        : UDINT;
END_VAR
```

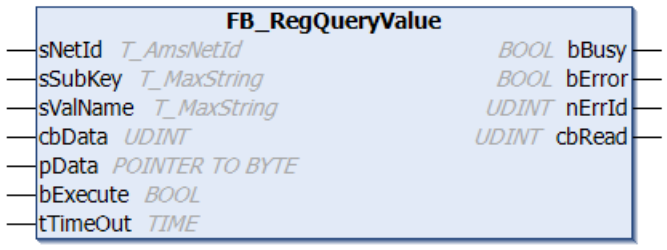
程序代码：

```
IF bReset THEN(* Clear buffer *)
    bReset := FALSE;
    fbStack.A_Reset( pBuffer := ADR( buffer ), cbBuffer := SIZEOF( buffer ) );
END_IF

IF bAdd THEN(* Add entries *)
    bAdd := FALSE;
    FOR i:= 0 TO 9 BY 1 DO
        fbStack.A_Push( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pWrite := ADR(putEntry[i
]), cbWrite := LEN(putEntry[i]) + 1 );
        IF fbStack.bOk THEN(* Success *)
            ;
        ELSE(* Buffer overflow *)
            ;
        END_IF
    END_FOR
END_IF

IF bGet THEN(* Peek newest entry *)
    bGet := FALSE;
    fbStack.A_Top( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pRead := ADR(getEntry), cbRea
d := SIZEOF(getEntry) );
    IF fbStack.bOk THEN(* Success *)
        ;
    ELSE(* Buffer is empty *)
        ;
    END_IF
END_IF

IF bRemove THEN(* Remove newest entry *)
    bRemove := FALSE;
    fbStack.A_Pop( pBuffer := ADR(buffer), cbBuffer := SIZEOF(buffer), pRead := ADR(getEntry), cbRea
d := SIZEOF(getEntry) );
    IF fbStack.bOk THEN(* Success *)
        ;
    ELSE(* Buffer is empty *)
        ;
    END_IF
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.54 FB_RegQueryValue



系统注册表是一个分层结构的树。树中的节点被称为键。每个键可能包含子键和数据值。功能块“FB_RegQueryValue”可以用于从带有预定义句柄 **HKEY_LOCAL_MACHINE** 的分支中读取各个系统注册表值。如果成功，则会将 *cbData* 数据字节复制到地址为 *pData* 的缓冲区中。该功能块可以用于读取任何值类型（例如 REG_DWORD、REG_SZ）或字节长度不限的二进制数据（REG_BINARY）。

注释：

sSubKey 和 *sValueName* 字符串不能为空！

64 位操作系统的 HKEY_LOCAL_MACHINE\SOFTWARE

在 64 位 Windows 操作系统中，32 位应用程序的所有注册表项都不会存储在 HKEY_LOCAL_MACHINE\SOFTWARE\ 下，而是存储在 HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\ 下。
当选择 SOFTWARE 文件夹下的一个注册表项时，FB_RegQueryValue 和 FB_RegSetValue 功能块会在 WOW6432Node 文件夹下自动工作，与任何 32 位应用程序一样。由操作系统自动执行重定向。

输入

```
VAR_INPUT
    sNetId    : T_AmsNetId;
    sSubKey   : T_MaxString;
    sValName  : T_MaxString;
    cbData    : UDINT;
    pData     : POINTER TO BYTE;
    bExecute  : BOOL;
    tTimeOut  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetID	在这里可以指定一个字符串，其中包含要读取其系统注册表的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sSubKey	T_MaxString	包含子键名称的字符串
sValName	T_MaxString	包含值名称的字符串
cbData	UDINT	要读取的值数据字节数。
pData	BYTE	数据缓冲区/变量的地址，值数据将被复制到该地址中。使用 ADR 运算符可以确定地址。程序员负责确定数据缓冲区的大小，以便其可以容纳 cbData 数据字节。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy  : BOOL;
    bError : BOOL;
```

```
    nErrId : UDINT;
    cbRead : UDINT;
END_VAR
```

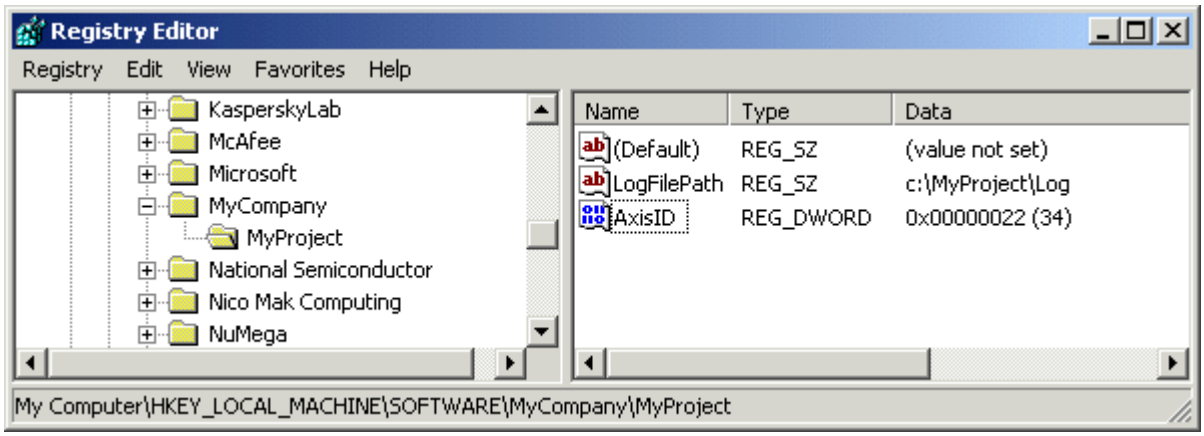
名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 bError 输出时，返回 ADS 错误编号 [▶ 376]或命令特定的错误代码（表）。

错误代码	错误描述
0x00	无错误
0x01/	无法打开/找到名称为 **sSubKey** 的键。
0x02/	无法打开/找到名称为 **sValName** 的键值。

名称	类型	描述
cbRead	UDINT	成功读取的值数据字节数。

示例：

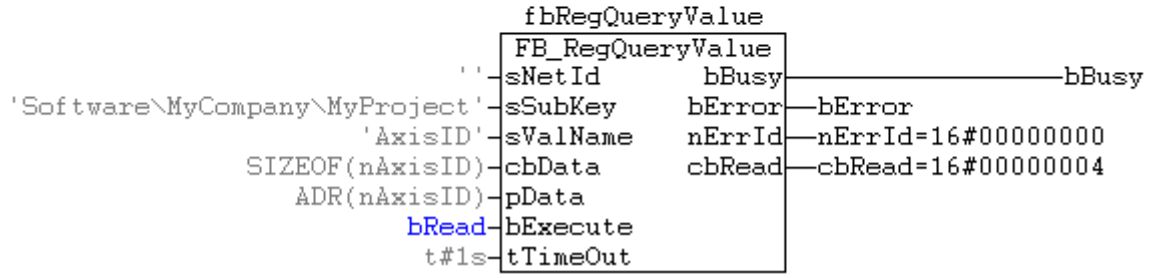
将从系统注册表中读取值 *AxisID* 和 *LogFilePath*。




```
PROGRAM MAIN
VAR
    fbRegQueryValue   : FB_RegQueryValue;
    bRead             : BOOL;
    bBusy             : BOOL;
    bError            : BOOL;
    nErrId            : UDINT;
    cbRead            : UDINT;
    sValData          : STRING;
    nAxisID           : DWORD;
END_VAR
```

读取 REG_DWORD-Value：

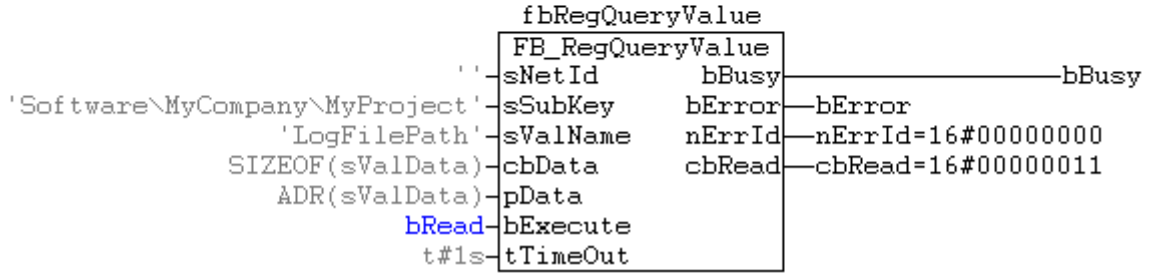
```
fbRegQueryValue
   bRead = TRUE
   bBusy = FALSE
   bError = FALSE
   nErrId = 16#00000000
   cbRead = 16#00000004
   nAxisID = 16#00000022
```



在 PLC 变量 *nAxisId* 中读取注册表的值 0x22。

读取 REG_SZ-Value：

```
fbRegQueryValue
   bRead = TRUE
   bBusy = FALSE
   bError = FALSE
   nErrId = 16#00000000
   cbRead = 16#00000011
   sValData = 'c:\MyProject\Log'
```



在 PLC 变量 *sValData* 中读取注册表的字符串“c:\MyProject\Log”。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.55 FB_RegSetValue

系统注册表是一个分层结构的树。树中的节点被称为键。每个键可能包含子键和数据值。

功能块“FB_RegSetValue”可以用于在带有预定义句柄 **HKEY_LOCAL_MACHINE** 的分支中写入或生成各个键值或新的键名称和值（子键+值）。可以将任意数量的值类型（例如 REG_DWORD、REG_SZ）或最多 500 字节的二进制数据（REG_BINARY）写入系统注册表。如果键值尚不存在，则会自动生成。

注释：

sSubKey 和 *sValueName* 字符串不能为空！

64 位操作系统的 HKEY_LOCAL_MACHINE\SOFTWARE

在 64 位 Windows 操作系统中，32 位应用程序的所有注册表项都不会存储在 HKEY_LOCAL_MACHINE\SOFTWARE\ 下，而是存储在 HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\ 下。
当选择 SOFTWARE 文件夹下的一个注册表项时，FB_RegQueryValue 和 FB_RegSetValue 功能块会在 WOW6432Node 文件夹下自动工作，与任何 32 位应用程序一样。由操作系统自动执行重定向。

输入

```
VAR_INPUT
    sNetId   : T_AmsNetId;
    sSubKey  : T_MaxString;
    sValName : T_MaxString;
    eValType : E_RegValueType;
    cbData   : UDINT;
    pData    : POINTER TO BYTE;
    bExecute : BOOL;
    tTimeOut : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetId	T_AmsNetID	在这里可以指定一个字符串，其中包含要读取其系统注册表的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sSubKey	T_MaxString	包含子键名称的字符串
sValName	T_MaxString	包含值名称的字符串
eValType	E_RegValueType [▶ 328]	要写入的注册表数据的数据类型格式，例如 REG_DWORD 或 REG_SZ。
cbData	UDINT	要读取的值数据字节数。（对于字符串变量，这包括最后的空值）。
pData	BYTE	包含值数据的数据缓冲区/PLC 变量的地址。使用 ADR 运算符可以确定地址。程序员负责确定数据缓冲区的大小，以便可以从其中读取 *cbData* 数据字节。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy    : BOOL;
    bError   : BOOL;
```

```
    nErrId   : UDINT;
    cbWrite  : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrId	UDINT	在设置 bError 输出时，返回 ADS 错误编号 [▶ 376]或命令特定的错误代码（表）。

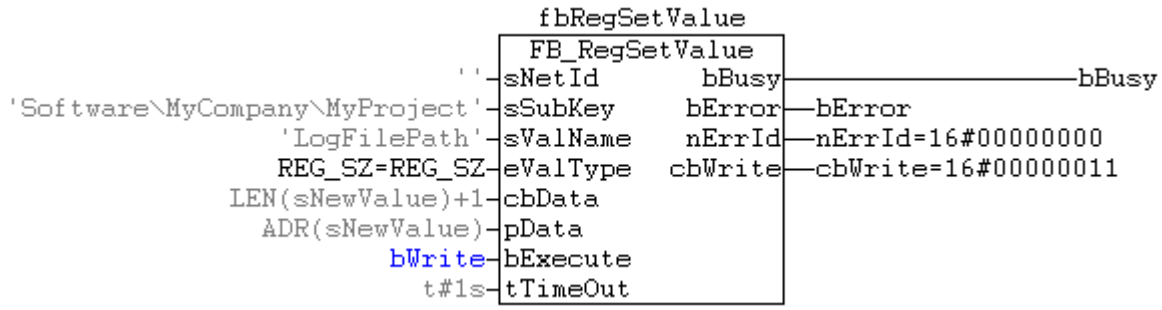
错误代码	错误描述
0x00	无错误
0x01/	无法打开/找到名称为 **sSubKey** 的键。
0x02/	无法打开/找到名称为 **sValName** 的键值。

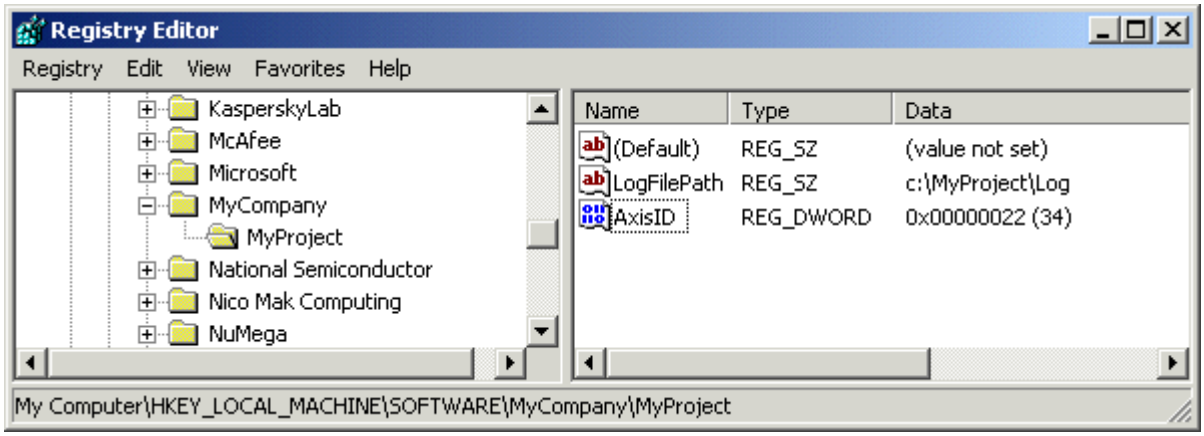
名称	类型	描述
cbWrite	UDINT	成功写入的值数据字节数。

示例：

在带有预定义句柄 *HKEY_LOCAL_MACHINE* 的分支中，要创建并设置一个子键 *“SOFTWARE\MyCompany\MyProject”*，键名称为 *“LogFileName”*，类型为 *REG_SZ* 值为 *“c:\MyProject\Log”*。

```
PROGRAM MAIN
VAR
    fbRegSetValue   : FB_RegSetValue;
    bBusy             : BOOL;
    bError          : BOOL;
    nErrId          : UDINT;
    cbWrite         : UDINT;
    bWrite          : BOOL;
    sNewValue       : STRING := 'c:\MyProject\Log';
END_VAR
```

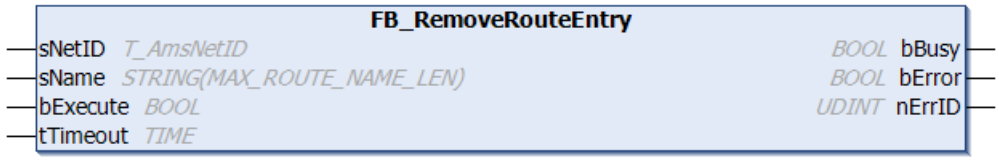




要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.56 FB_RemoveRouteEntry



该功能块可以用于从 AMS 路由器连接（远程路由）列表中删除与 TwinCAT 系统的现有连接。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetId;
    sName     : String (MAX_ROUTE_NAME_LEN);
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定要删除 AMS 路由器连接的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
sName	STRING	要删除的连接的名称。字符串最大长度由常量限定（默认值：31 个字符）。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

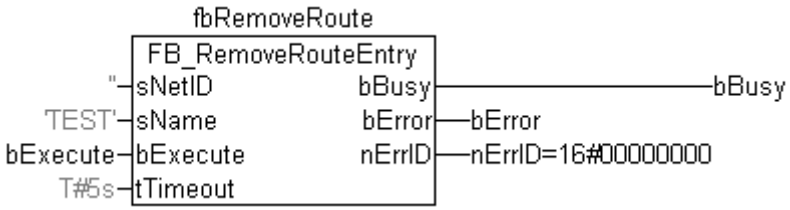
```
VAR_OUTPUT
    bBusy    : BOOL;
    bError   : BOOL;
    nErrID   : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

示例：

从本地 TwinCAT 系统的 AMS 路由器连接列表中删除名称为“TEST”的连接。如果在 bExecute 变量上检测到上升沿，则删除连接。

```
PROGRAM P_TEST2
VAR
    fbRemoveRoute  : FB_RemoveRouteEntry;
    bExecute       : BOOL;
    bBusy          : BOOL;
    bError         : BOOL;
    nErrID         : UDINT;
END_VAR
```



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.57 FB_ScopeServerControl

功能块 FB_ScopeServerControl 使 PLC 能够收集数据，以便随后使用 TwinCAT Scope 2 显示。

输入

```
VAR_INPUT
    sNetId      : T_AmsNetId;
    eReqState   : E_ScopeServerState := SCOPE_SERVER_IDLE;
    sConfigFile : STRING;
    sSaveFile   : STRING;
    tTimeout    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定包含 TwinCAT 目标系统的网络地址的字符串。对于本地计算机，可以指定一个空字符串。
eReqState	E_ScopeServerState [▶ 329]	请求的范围服务器状态
sConfigFile	STRING	带有配置文件名称的完整路径（例如：*C:\TwinCAT\TwinCATScope2\First.sv2*）。
sSaveFile	STRING	带有数据文件名称的完整路径（例如：*C:\TwinCAT\TwinCATScope2\First.svd*）。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

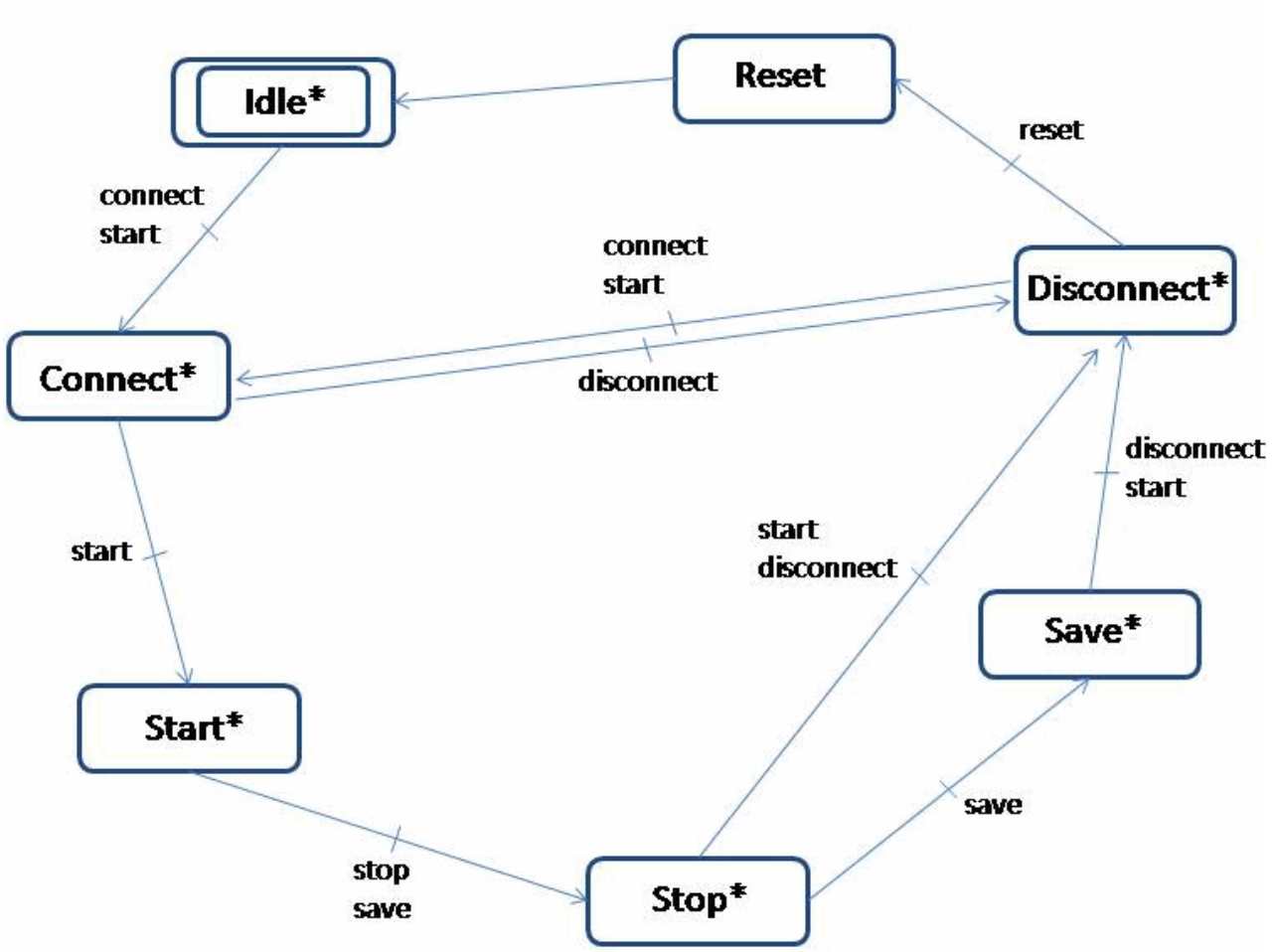
```
VAR_OUTPUT
    bBusy    : BOOL;
    bDone    : BOOL;
    bError   : BOOL;
    nErrorId : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bDone	BOOL	如果已激活请求的状态，则进行设置。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。

名称	类型	描述
nErrorId	UDINT	如果已设置错误输出 bError，则显示错误编号。通常，这可以是 ADS 错误编号 [▶ 376]或该库的特定错误代码 [▶ 376]。

配置文件（*.sv2）只允许使用一个目标系统。

状态图：



*) 始终可以重置状态更改

该状态图显示了 eReqState 的可能转换。如果请求其他状态更改，则将设置 bError。

示例：

声明部分：

```
FUNCTION_BLOCK FB_ScopeServerSample
VAR_INPUT
    bExternalTriggerEvent: BOOL := FALSE;
END_VAR
VAR_OUTPUT
END_VAR
VAR
    fbScopeServerControl: FB_ScopeServerControl;
    eReqState: E_ScopeServerState := SCOPE_SERVER_IDLE;
    bBusy: BOOL := FALSE;
    bDone: BOOL := FALSE;
    bError: BOOL := FALSE;
    nErrorId: UDINT := 0;
    fbTimer: TON;
```

```
    bTriggerTimer: BOOL := FALSE;
    nState: UDINT := 0;
END_VAR
```

FB_ScopeServerSample 的实现

```
CASE nState OF
0:
    eReqState := SCOPE_SERVER_START;
    nState := 10;
10:
    IF fbScopeServerControl.bDone AND bExternalTriggerEvent
    THEN
        bTriggerTimer := TRUE;
        nState := 20;
    END_IF
20:
    IF fbTimer.Q THEN
        eReqState := SCOPE_SERVER_SAVE;
        bTriggerTimer := FALSE;
        nState := 30;
    END_IF
30:
    IF fbScopeServerControl.bDone THEN
        eReqState := SCOPE_SERVER_DISCONNECT;
    END_IF
END_CASE
fbTimer(IN:=bTriggerTimer, PT:=t#10s);
fbScopeServerControl( sNetId:= '',
                      eReqState:= eReqState,
                      sConfigFile:= 'C:\twinCat\scope\test.sv2',
                      sSaveFile:= 'C:\twinCat\scope\test.svd',
                      tTimeout:= t#5s,
                      bBusy=>bBusy,
                      bDone=>bDone,
                      bError=>bError,
                      nErrorId=>nErrorId);
```

该示例演示了如何使用 Scope Server 执行长期记录。

为此，可加载现有配置（Test.sv2）。在该示例中已存储 Test.sv2，以便在环形缓冲区中运行。只有在由 FB_ScopeServerControl 触发后，才会完成数据记录。如果发生内部触发事件（这可能是错误事件），则会启动定时器，并在 10 秒后将数据保存在 Test.svd 中。这样，数据文件包含了触发事件之前和之后的信息。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.58 FB_SetTimeZoneInformation

该功能块可以用于修改或设置操作系统的时区设置。

时间设置

在设置新的时区设置后，操作系统可能会更改某些时间设置。

通过功能块 NT_SetLocalTime [▶ 122] 可以重置时间。

输入

```
VAR_INPUT
    sNetID    : T_AmsNetID;
    tzInfo    : ST_TimeZoneInformation;
    bExecute  : BOOL;
    tTimeout  : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
sNetID	T_AmsNetID	在这里可以指定一个字符串，其中包含要更改其时区配置的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
tzInfo	ST_TimeZoneInformation [▶ 342]	具有要设置的新时区设置的结构。
bExecute	BOOL	功能块由该输入的正沿（上升沿）启用。
tTimeout	TIME	规定执行 ADS 命令时不得超过的超时长度。

输出

```
VAR_OUTPUT
    bBusy        : BOOL;
    bError       : BOOL;
    nErrID       : UDINT;
END_VAR
```

名称	类型	描述
bBusy	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
bError	BOOL	信号如果在命令传输过程中出现错误，则在 *bBusy* 输出被复位后，此输出信号将被置位。
nErrID	UDINT	在设置 *bError* 输出时，返回 ADS 错误编号 [▶ 376]。

示例：

在本地 TwinCAT 系统上，要设置时区：“西欧标准时间”。例如，在 PLC 库中已经声明了一个带有合适的参数值的常量：**WEST_EUROPE_TZI**。要配置其他时区，必须为功能块的 *tzInfo* 输入分配相应的值（请参见 ST_TimeZoneInformation [▶ 342] 结构的相关描述）。

```
VAR_GLOBAL CONSTANT
...
    (* West Europa Standard Time Zone settings *)
    WEST_EUROPE_TZI : ST_TimeZoneInformation := (bias:=-60,
                      standardName:='W. Europe Standard Time',
                      standardDate:=(wYear:=0,wMonth:=10,wDayOfWeek:=0,wDay:=5,wHour:=3),
                      standardBias:=0,
                      daylightName:='W. Europe Daylight Time',
                      daylightDate:=(wYear:=0,wMonth:=3,wDayOfWeek:=0,wDay:=5,wHour:=2),
                      daylightBias:=-60);
...
END_VAR
```

声明部分：

```
>PROGRAM MAIN
VAR
    fbGet    : FB_GetTimeZoneInformation;
    fbSet    : FB_SetTimeZoneInformation;
    tzi_get  : ST_TimeZoneInformation;
    tzID     : E_TimeZoneID;
    bGet     : BOOL := TRUE;
    bSet     : BOOL := FALSE;
END_VAR
```

bSet 变量出现上升沿时，将触发所需时区设置。为了进行验证，通过检测 *bGet* 变量的上升沿来读取当前设置。

```
IF bGet THEN
    bGet := FALSE;
    fbGet(bExecute := TRUE);
ELSE
    fbGet(bExecute := FALSE, tzInfo => tzi_get, tzID => tzID );
END_IF

IF bSet THEN
    bSet := FALSE;
    fbSet( bExecute := TRUE, tzInfo := WEST_EUROPE_TZI );
ELSE
    fbSet( bExecute := FALSE );
END_IF
```

其他时间和时区函数和功能块：

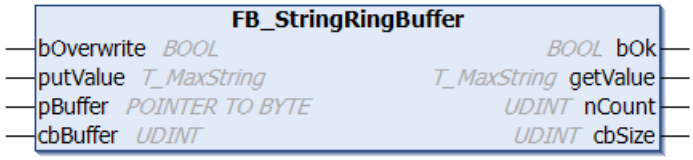
- FB_TzSpecificLocalTimeToSystemTime [▶ 115]

- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7），不适用于 TwinCAT/BSD	Tc2_Utilities（系统）

3.59 FB_StringRingBuffer



功能块 FB_StringRingBuffer 可以用于将字符串变量写入环形缓冲区，或从环形缓冲区中读取先前写入的字符串变量。根据 FIFO 原则读出已写入的字符串，与先前将它们写入环形缓冲区的顺序相同。这意味着最早写入的数据集（即最旧的数据集）会最先被读取。缓冲区内存通过 *pBuffer*/*cbBuffer* 输入变量提供给功能块使用。字符串的写入/读取通过调用动作指令进行控制。该功能块包含以下核心功能：

- **A_AddTail**（将新的字符串写入环形缓冲区。）
- **A_GetHead**（读取环形缓冲区中最旧的字符串，但不会将其移除。）
- **A_RemoveHead**（从环形缓冲区中读取并移除最旧的字符串。）
- **A_Reset**（删除环形缓冲区中的所有字符串。）

输入

```
VAR_INPUT
    bOverwrite : BOOL;
    putValue   : T_MaxString := '';
    pBuffer    : POINTER TO BYTE;
    cbBuffer   : UDINT;
END_VAR
```

名称	类型	描述
bOverwrite	BOOL	如果为 TRUE 且出现缓冲区溢出的情况，则会覆盖最旧的条目。如果为 FALSE 且出现缓冲区溢出的情况（bOk = FALSE），则会报告错误。
putValue	T_MaxString	要写入环形缓冲区的字符串。
pBuffer	BYTE	功能块用作缓冲存储器的 PLC 变量（例如 ARRAY[....] OF BYTES）的地址。使用 ADR 运算符可以确定地址。
cbBuffer	UDINT	要用作缓冲存储器的 PLC 变量的最大字节大小。使用 SIZEOF 运算符可以确定大小。

输出

```
VAR_OUTPUT
    bOk      : BOOL;
    getValue : T_MaxString := '';
    nCount   : UDINT;
    cbSize   : UDINT;
END_VAR
```

名称	类型	描述
bOk	BOOL	如果成功添加或移除一个新的字符串，则返回 TRUE。如果出现缓冲区溢出或缓冲区中没有更多可用条目，则返回 FALSE。
getValue	T_MaxString	该输出可返回从环形缓冲区中读取的最后一个字符串。
nCount	UDINT	返回当前缓冲字符串的数量。
cbSize	UDINT	返回缓冲区中当前分配的数据字节数。分配的数据字节数总是大于实际写入值数据的数量。每个字符串都补充了附加信息，以便以后可以找到它。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.60 FB_SystemTimeToTzSpecificLocalTime

该功能块将 UTC 时间（结构化系统时间格式）转换为本地时间（结构化系统时间格式），同时考虑指定的时区信息。功能块 FB_FileTimeToTzSpecificLocalTime [▶ 27] 具有类似的功能，但使用不同的时间格式（文件时间格式）。

该功能块仅适用于转换**连续的** UTC 时间戳信息。该功能块使用时区信息来计算在本地时间中所需的时间步长（夏令时/冬令时转换）。不允许使用在 UTC 输入时间中的时间步长，否则会导致转换错误。原因：该功能块在内部存储最后转换的时间，因此，当本地时间发生变化时，它可以从 UTC 输入时间和存储值中检测出 B 时间（见下文）。

该功能块与A_Reset()相关联。如果调用该操作，则功能块输出和本地存储（最后转换）的时间将被重置为零。

输入

```
VAR_INPUT
    in     : TIMESTRUCT;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
in	TIMESTRUCT [▶ 347]	要转换的 UTC 时间（结构化系统时间格式）。
tzInfo	ST_TimeZoneInformation [▶ 342]	带有操作系统的当前时区信息的结构变量。

输出

```
VAR_OUTPUT
    out    : TIMESTRUCT;
    eTzID  : E_TimeZoneID := eTimeZoneID_Unknown;
    bB     : BOOL;
END_VAR
```

名称	类型	描述
out	TIMESTRUCT [▶ 347]	转换后的本地时间（结构化系统时间格式）
eTzID	E_TimeZoneID [▶ 329]	附加的夏令时/冬令时信息。
bB	BOOL	TRUE => B 时间（例如：**02:05:00 CET B**），FALSE => 其他时间（例如：**02:05:00 CEST A**）。当本地时间发生回跳时，该输出被置位，并在重复的本地时间通过后复位。

示例：

```
PROGRAM MAIN
VAR
    in         : TIMESTRUCT := ( wYear := 2011, wMonth := 4, wDay := 29, wHour := 14, wMinute := 46,
wSecond := 31, wMilliseconds := 99 );(* UTC time *)
    out        : TIMESTRUCT; (* Local time result is:= ( wYear := 2011, wMonth := 4, wDay := 29, wHou
r := 16, wMinute := 46, wSecond := 31, wMilliseconds := 99 ) *)
    fbToLocal : FB_SystemTimeToTzSpecificLocalTime;
END_VAR
```

```
fbToLocal( in := in, tzInfo := WEST_EUROPE_TZI, out => out );
```

其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

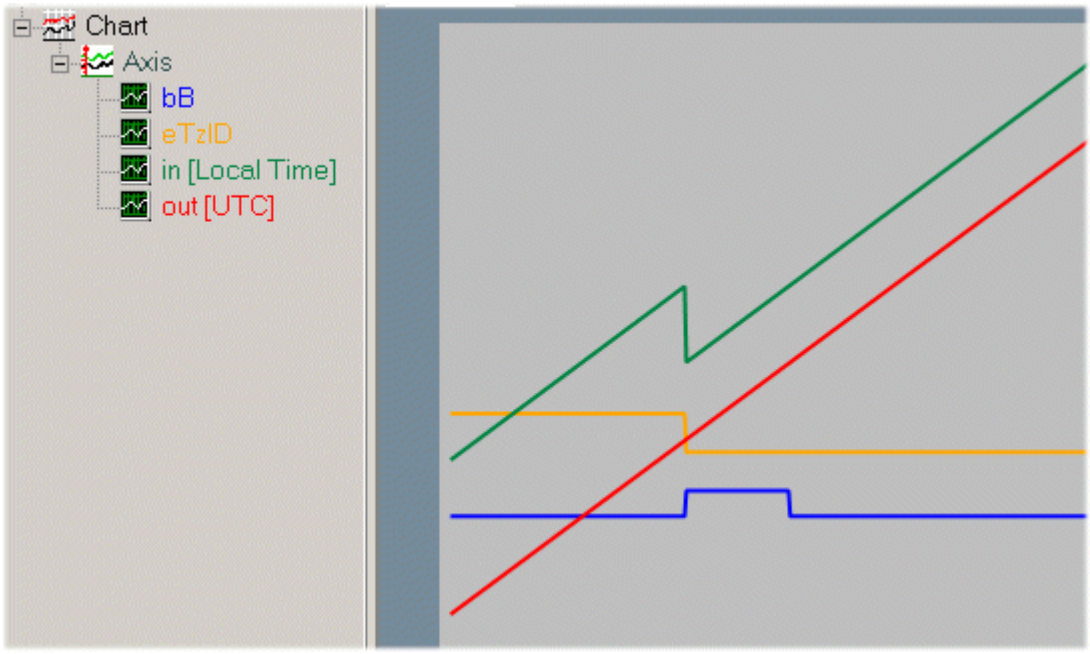
3.61 FB_TzSpecificLocalTimeToFileTime64

该功能块将本地时间（文件时间格式）转换为 UTC 时间（文件时间格式），同时考虑指定的时区信息。功能块 FB_TzSpecificLocalTimeToSystemTime [▶ 115] 具有类似的功能，但转换不同的时间格式（结构化系统时间格式）。

该功能块仅适用于转换**连续的**本地时间戳信息。允许存在因夏令时/冬令时转换而引起的本地时间的阶跃变化，并且功能块可以进行正确检测。但是任意改变本地时间会导致转换错误。原因：在功能块内部存储最后转换的时间，以便在重置本地时间时能够识别夏令时/冬令时信息和 B 时间（见下文）。该功能块与A_Reset()相关联。如果调用该操作，则功能块输出和本地存储（最后转换）的时间将被重置为零。

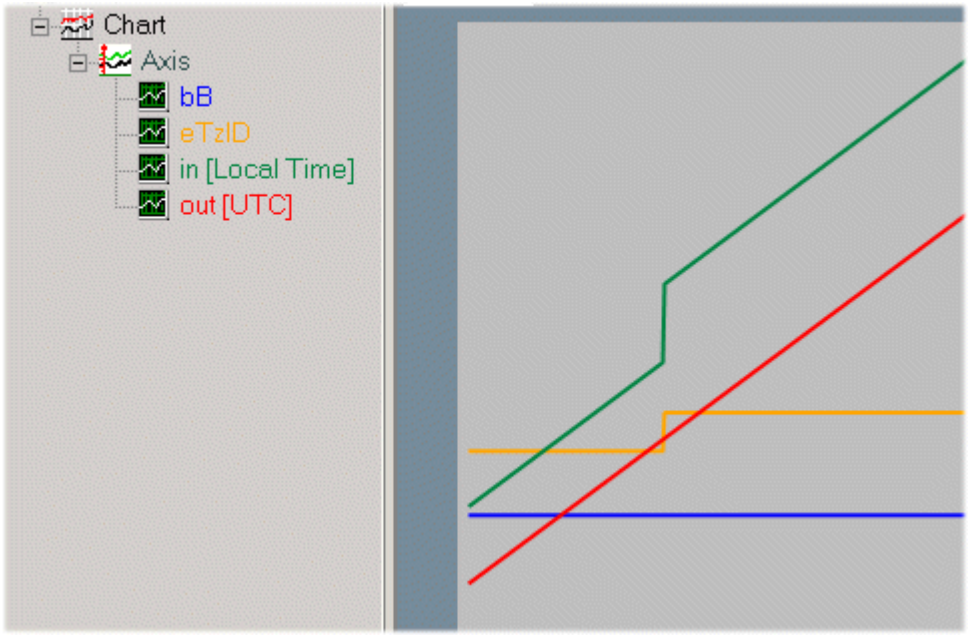
本地时间的阶跃变化有问题，因为它们必须转换为线性 UTC 时间。因此，建议使用（连续的）UTC 时间执行时间戳任务，并且仅在显示目的（例如在可视化中）时才将时间转换为相应的本地时间。

1. 从夏令时到冬令时转换的图示（tzInfo = WEST_EUROPE_TZI）：



本地时间（绿色）跳回。UTC 输出时间（红色）是连续的。本地时间：**02h:59m:59s:999ms..** 后面直接是：**02h:00m:00s:000ms..** 2h 和 3h 之间的时间出现两次。例如，时间转换前的重复时间被称为 **02:05:00 CEST A**，转换后的时间被称为 **02:05:00 CET B**。输出变量 *bB* 指示它是第一次还是第二次*经过*。第二次*经过*时，输出变量 *bB*（蓝色）将被设置为 TRUE。在经过重复时间后，自动重置 *bB* 输出变量。时区 ID（橙色）从 *eTimeZoneID_Daylight*（夏令时）变为 *eTimeZoneID_Standard*（冬令时）。

2. 从冬令时到夏令时转换期间的时间行为图示（tzInfo = WEST_EUROPE_TZI）：



本地时间（绿色）向前跳转。UTC 输出时间（红色）是连续的。本地时间：**2h:59m:59s:999ms..** 后面直接是：**3h:00m:00s:000ms..** 时区 ID（橙色）从 *eTimeZoneID_Standard*（冬令时）变为 *eTimeZoneID_Daylight*（夏令时）。

输入

```
VAR_INPUT
    in     : T_FILETIME64;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
in	T_FILETIME [▶ 344]	要转换的本时间（文件时间格式）。
tzInfo	ST_TimeZoneInformation [▶ 342]	带有操作系统的当前时区信息的结构变量。

输出

```
VAR_OUTPUT
    out   : T_FILETIME64;
    eTzID : E_TimeZoneID := eTimeZoneID_Unknown;
    bB    : BOOL;
END_VAR
```

名称	类型	描述
out	T_FILETIME [▶ 344]	转换后的 UTC 时间（文件时间格式）
eTzID	E_TimeZoneID [▶ 329]	附加的夏令时/冬令时信息。
bB	BOOL	TRUE => B 时间（例如：**02:05:00 CET B**），FALSE => 其他时间（例如：**02:05:00 CEST A**）。当本地时间发生回跳时，该输出被置位，并在重复的本地时间通过后复位。

示例：

本地时间：DT#2011-09-02-11:01:31 被转换为 UTC 时间：DT#2011-09-02-09:01:31。

```
PROGRAM MAIN
VAR
    in      : DT := DT#2011-09-02-11:01:31;(* Local time *)
    out     : DT;(* UTC time *)
    fbToUTC : FB_TzSpecificLocalTimeToFileTime64;
END_VAR
```

```
fbToUTC( in := DT_TO_FILETIME64( in ), tzInfo := WEST_EUROPE_TZI );
out := FILETIME64_TO_DT( fbToUTC.out );
```

其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTime64ToTzSpecificLocalTime [▶ 61]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTime64Bias [▶ 154]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

3.62 FB_TzSpecificLocalTimeToSystemTime

该功能块将本地时间（结构化系统时间格式）转换为 UTC 时间（结构化系统时间格式），同时考虑指定的时区信息。功能块：FB_TzSpecificLocalTimeToFileTime [▶ 29] 具有类似的功能，但使用不同的时间格式（文件时间格式）。

该功能块仅适用于转换**连续的**本地时间戳信息。允许存在因夏令时/冬令时转换而引起的本地时间的阶跃变化，并且功能块可以进行正确检测。但是任意改变本地时间会导致转换错误。原因：在功能块内部存储最后转换的时间，以便在重置本地时间时能够识别夏令时/冬令时信息和 B 时间（见下文）。该功能块与A_Reset()相关联。如果调用该操作，则功能块输出和本地存储（最后转换）的时间将被重置为零。

本地时间的阶跃变化有问题，因为它们必须转换为线性 UTC 时间。因此，建议使用（连续的）UTC 时间执行时间戳任务，并且仅在显示目的（例如在可视化中）时才将时间转换为相应的本地时间。

有关更多信息，请参见 FB_TzSpecificLocalTimeToFileTime [▶ 29] 功能块的文档。

输入

```
VAR_INPUT
    in     : TIMESTRUCT;
    tzInfo : ST_TimeZoneInformation;
END_VAR
```

名称	类型	描述
in	TIMESTRUCT [▶ 347]	要转换的本地时间（结构化系统时间格式）
tzInfo	ST_TimeZoneInformation [▶ 342]	带有操作系统的当前时区信息的结构变量。

输出

```
VAR_OUTPUT
    out    : TIMESTRUCT;
    eTzID  : E_TimeZoneID := eTimeZoneID_Unknown;
    bB     : BOOL;
END_VAR
```

名称	类型	描述
out	TIMESTRUCT [▶ 347]	转换后的 UTC 时间（结构化系统时间格式）
eTzID	E_TimeZoneID [▶ 329]	附加的夏令时/冬令时信息。
bB	BOOL	TRUE => B 时间（例如：**02:05:00 CET B**），FALSE => 其他时间（例如：**02:05:00 CEST A**）。当本地时间发生回跳时，该输出被置位，并在重复的本地时间通过后复位。

示例：

```
PROGRAM MAIN
VAR
    in      : TIMESTRUCT := ( wYear := 2011, wMonth := 4, wDay := 29, wHour := 16, wMinute := 46, wS
econd := 31, wMilliseconds := 99 );(* Local time *)
    out     : TIMESTRUCT;
(* UTC time result is:= ( wYear := 2011, wMonth := 4, wDay := 29, wHour := 14, wMinute := 46, wSecon
d := 31, wMilliseconds := 99 ) *)
    fbToUTC : FB_TzSpecificLocalTimeToSystemTime;
END_VAR
fbToUTC( in := in, tzInfo := WEST_EUROPE_TZI, out => out );
```

其他时间和时区函数和功能块：

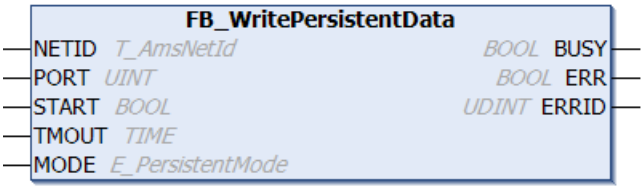
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]

- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.63 FB_WritePersistentData



功能块 FB_WritePersistentData 是 WritePersistentData [▶ 146] 功能块的扩展版本。通过 MODE 参数可以影响写入持久性数据的系统行为 [▶ 373]（数据一致性/任务循环时间超限）。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    PORT    : UINT;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
    MODE    : E_PersistentMode;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要保存其持久性数据的 TwinCAT 计算机的网络地址。对于本地计算机，可以输入一个空字符串。
PORT	UINT	PORT 参数可指定要存储其持久性数据的运行时系统。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。
Mode	E_PersistentMode [▶ 327]	要写入持久性数据的模式。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

示例：

请参见：写入持久性数据：系统行为。 [▶ 373]

请参见：WritePersistentData 功能块文档中的示例。 [▶ 146]

要求

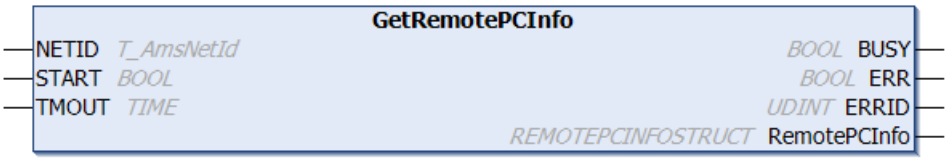
开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.64

还请参阅有关此

全局参数 [▶ 351]

3.65 GetRemotePCInfo



功能块 GetRemotePCInfo 可以用于读取有关 TwinCAT 路由器中已配置的远程 PC 的信息。在成功执行后，“RemotePCInfo”结构将以字符串形式包含远程 PC 的 NetID 和名称，并按其在 TwinCAT 路由器中的存储顺序排列。该功能块允许读取与本地或远程 TwinCAT 系统有关的路由器信息。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定通过已配置的远程 PC 要读取其路由器信息的TwinCAT 计算机的网络地址。要确定本地 TwinCAT 系统的远程 PC，可以指定一个空字符串。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

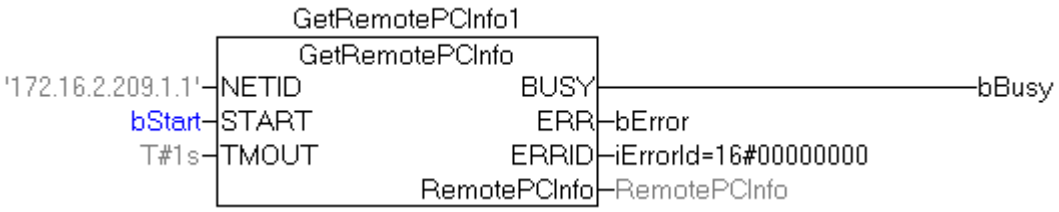
```
VAR_OUTPUT
    BUSY         : BOOL;
    ERR          : BOOL;
    ERRID        : UDINT;
    RemotePCInfo : REMOTEPCINFOSTRUCT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。
RemotePCInfo	REMOTEPCINFOSTRUCT [▶ 332]	包含有关已配置的远程 PC 的信息的结构。

示例：

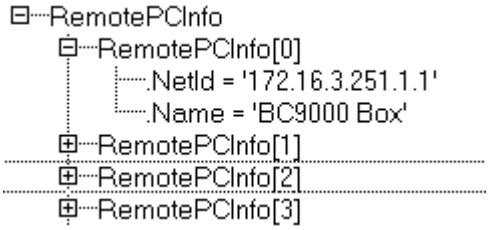
```
PROGRAM MAIN
VAR
    GetRemotePCInfo1     : GetRemotePCInfo;
    RemotePCInfo         : REMOTEPCINFOSTRUCT;
    bBusy                : BOOL;
```

```
    bError              : BOOL;
    iErrorId            : UDINT;
    bStart              : BOOL;
END_VAR
```



在线视图：

已配置的远程 PC 的 NetID 和名称。



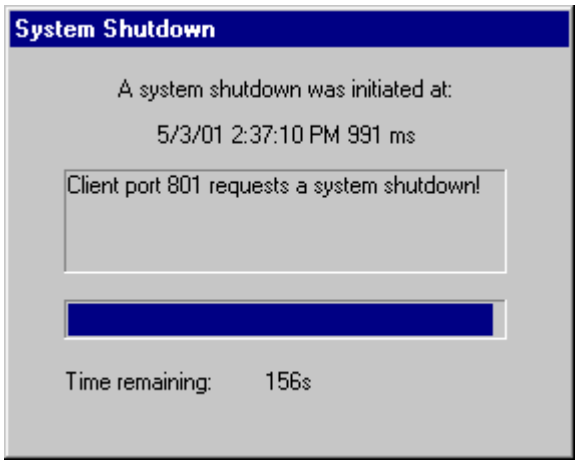
要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.66 NT_AbortShutdown

功能块 NT_AbortShutdown 可以用于终止之前通过 NT_Shutdown [▶ 125] 功能块调用的关闭命令。

在调用 NT_Shutdown [▶ 125] 功能块时，可通过参数指定关机前的延迟时间。在消息窗口中会显示剩余时间：



在延迟时间过后，才会关闭操作系统。在此期间，借助功能块“NT_AbortShutdown”可以从 PLC 中断关闭过程。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要中止关闭过程的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

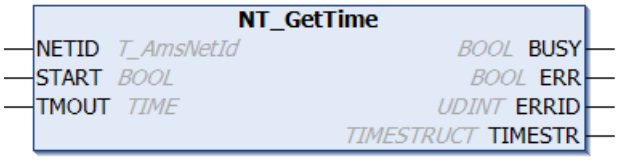
```
VAR_OUTPUT
    BUSY          : BOOL;
    ERR           : BOOL;
    ERRID         : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7）	Tc2_Utilities（系统）

3.67 NT_GetTime



功能块 NT_GetTime 可以用于确定 TwinCAT 系统的本地 Windows 系统时间（在任务栏中可显示本地 Windows 系统时间）。年、月、日、星期、小时、分钟、秒钟和毫秒存储在 TIMESTRUCT [▶ 347] 结构的变量中。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要确定其本地 Windows 系统时间的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY          : BOOL;
    ERR           : BOOL;
    ERRID         : UDINT;
    TIMESTR       : TIMESTRUCT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。
TIMESTR	TIMESTRUCT [▶ 347]	具有本地 Windows 系统时间的结构。

示例:

请参见：示例：软件时钟（RTC、RTC_EX、RTC_EX2） [▶ 371]。

其他时间和时区函数和功能块:

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64、TC OS ARMT2）	Tc2_Utilities（系统）

3.68 NT_Reboot

借助功能块 NT_Reboot 可以重新启动 Windows NT 操作系统。该功能在逻辑上等效于Windows任务栏中的‘重启’命令。通过参数 DELAY 可以定义执行重新启动命令前的延迟时间。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要重新启动的 TwinCAT 计算机的网络地址。如果在本地计算机上执行重新启动，可以输入一个空字符串。
DELAY	DWORD	执行重新启动命令前的延迟时间（以秒为单位）。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY          : BOOL;
    ERR           : BOOL;
    ERRID         : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7）	Tc2_Utilities（系统）

3.69 NT_SetLocalTime

功能块 NT_SetLocalTime 可以用于设置 TwinCAT 系统的本地 Windows 系统时间（在任务栏中可显示本地 Windows 系统时间）。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    TIMESTR : TIMESTRUCT;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要设置其本地 Windows 系统时间的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
TIMESTR	TIMESTRUCT [▶ 347]	具有新的本地 Windows 系统时间的结构。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY          : BOOL;
    ERR           : BOOL;
    ERRID         : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7）	Tc2_Utilities（系统）

3.70 NT_SetTimeToRTCTime

在 Windows CE 下的 PLC 运行时系统中不提供该功能！

功能块 NT_SetTimeToRTCTime 可以用于将本地 Windows 系统时间（在任务栏中显示）与 PC 的实时时钟（BIOS 中的 RTC 时间）同步。

注释

在调用该功能块时，会将 TwinCAT PC 的实时时钟与本地 Windows 系统时间进行比较，并根据差值对本地 Windows 系统时间进行校正。会考虑时区和夏令时。请注意，此校正操作可能导致测量数据或日志记录中出现时间跳变。

在设置本地 Windows 系统时间时，操作系统会自动将 RTC 时间设置为新的本地 Windows 系统时间。由于转换和延迟时间的原因，新的 RTC 时间不可避免地会出现微小的误差。误差在毫秒范围内。即，每次调用 NT_SetTimeToRTCTime 时，实时时钟都会出现微小的误差。为了尽量减少长期偏差，应每 24 小时执行一次校准，而不是在每个 PLC 周期内进行校准。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    SET     : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要同步本地 Windows 系统时间的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
SET	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY        : BOOL;
    ERR         : BOOL;
    ERRID       : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64）	Tc2_Utilities（系统）

3.71 NT_Shutdown

在 Windows CE 下不提供该功能!

借助功能块 NT_Shutdown 可以关闭 Windows NT 操作系统。该功能与 Windows 任务栏上的关闭命令大致对应。可通过DELAY参数定义关机命令执行前的延迟时间。

注意:

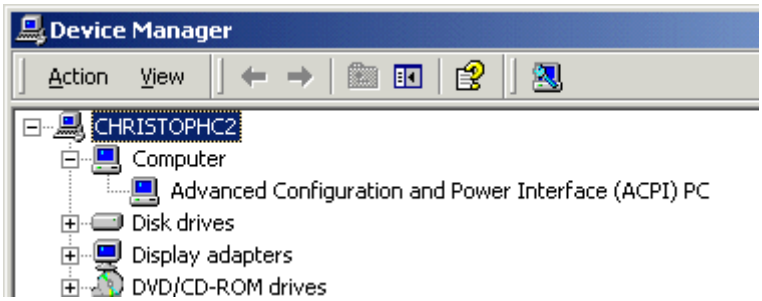
借助“NT_Shutdown”功能块，较新的操作系统（例如 Windows 2000）可以执行“关机并断电”功能（计算机关闭其电源）。仅可在符合 ACPI（高级配置和电源接口）的系统上使用该功能。在安装操作系统之前，应在 BIOS 中激活 ACPI 功能。

PC 的主板和电源必须支持 ACPI 功能。操作系统无法识别之后的更改。如果有支持 ACPI 的 PC，您可以按照以下方式检查 Windows 2000 等系统:

1. 在“System Manager”（系统管理器）中打开“system”（系统）文件夹。

2. 在“Hardware”（硬件）选项卡上选择“Device Manager”（设备管理器）。

在带有设备的导航树中，您现在可以在“Computer”（计算机）处读取：“高级配置和电源接口（ACPI）PC”。



默认的 TwinCAT 设置是在关闭电源时执行关机。您可以在 Windows 注册表中禁用关闭电源的功能。请添加以下条目:

在注册表中，“**DisableACPIPowerOff**”**REG_DWORD = 0x00000001**，位于:
“**HKEY_LOCAL_MACHINE\SOFTWARE\Beckhoff\TwinCAT\System**”其下

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    DELAY   : DWORD;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要设置其本地 Windows 系统时间的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
DELAY	DWORD	执行关闭命令前的延迟时间（以秒为单位）。
START	BOOL	功能块由该输入的正沿（上升沿）启用。

名称	类型	描述
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

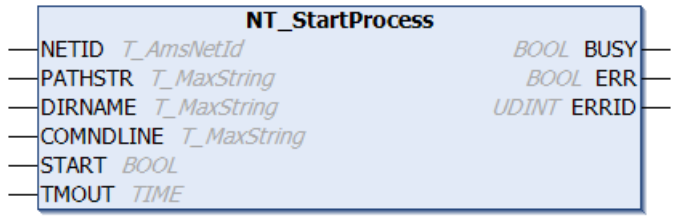
```
VAR_OUTPUT
    BUSY        : BOOL;
    ERR         : BOOL;
    ERRID       : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7）	Tc2_Utilities（系统）

3.72 NT_StartProcess



功能块 NT_StartProcess 可以用于从 PLC 启动 Windows 应用程序。该功能块还可以用于在远程 PC 上运行应用程序。

输入

```
VAR_INPUT
    NETID     : T_AmsNetId;
    PATHSTR   : T_MaxString;
    DIRNAME   : T_MaxString;
    COMNDLINE : T_MaxString;
    START     : BOOL;
    TMOUT     : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要启动应用程序的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PATHSTR	T_MaxString	要运行的应用程序的完整路径，以字符串形式表示（例如“C:\WINNT\NOTEPAD.EXE”）。
DIRNAME	T_MaxString	待执行应用程序的工作目录（以字符串形式表示，例如："C:\WINNT"）。
COMNDLINE	T_MaxString	命令行参数（例如：“win.ini”）。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

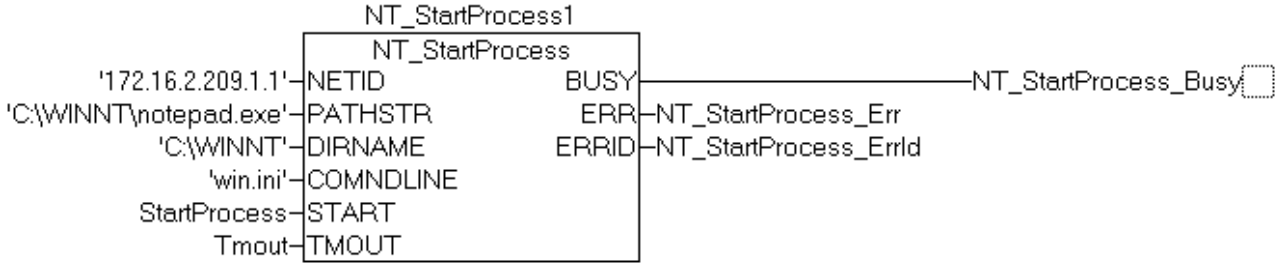
输出

```
VAR_OUTPUT
    BUSY          : BOOL;
    ERR           : BOOL;
    ERRID         : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]，Win32 错误代码（平台 SDK：Win32 API）。	如果设置了 ERR 输出，则返回错误代码。

示例：

```
NT_StartProcess1       : NT_StartProcess;
NT_StartProcess_Busy   : BOOL;
NT_StartProcess_Err    : BOOL;
NT_StartProcess_ErrId  : UDINT;
StartProcess           : BOOL;
Tmout                  : TIME;
```



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（WES7/Win7/Win10：TC RT x86/x64，WEC6/7：TC RT x86，WEC7：TC CE7 ARMV7，TC/BSD：TC RT x64）	Tc2_Utilities（系统）

3.73 PLC_ReadSymInfo

功能块 PLC_ReadSymInfo 可以用于确定有关 PLC 运行时系统的符号（变量）的信息。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    PORT    : T_AmsPort;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要确定其符号信息的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PORT	T_AmsPort	PLC 运行时系统的端口编号。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

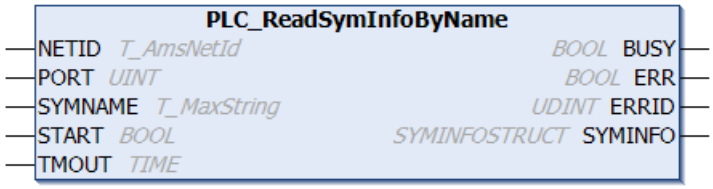
```
VAR_OUTPUT
    BUSY     : BOOL;
    ERR      : BOOL;
    ERRID    : UDINT;
    SYMCOUNT : UDINT;
    SYMSIZE  : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。
SYMCOUNT	UDINT	PLC 运行时系统中的符号数
SYMSIZE	UDINT	存储符号信息的数据的长度（以字节为单位）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.74 PLC_ReadSymInfoByName



功能块 PLC_ReadSymInfoByName 允许使用符号名称读取有关 PLC 符号变量的附加信息（例如，数据类型标识、索引组、索引偏移量、注释.....）。在成功执行后，数据将在数据结构 **SymInfo** 中可用，其类型为 SYMINFOSTRUCT。作为参数传输到功能块的符号名称的最大长度限值为 255 个字符。

输入

```
VAR_INPUT
    NETID    : T_AmsNetId;
    PORT     : T_AmsPort;
    SYMNAME  : T_MaxString;
    START    : BOOL;
    TMOUT    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要执行功能的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PORT	T_AmsPort	符号变量所属的 PLC 运行时系统的端口编号。
SYMNAME	T_MaxString	要读取其信息的 PLC 变量的符号名称（最多 255 个字符，包括完整路径，例如“MAIN.INIT_TASK.VARINT”）。

名称	类型	描述
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

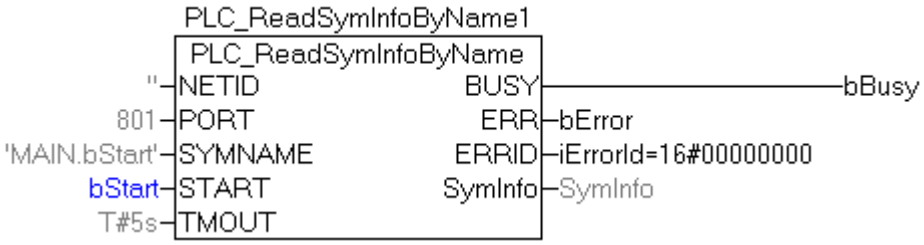
输出

```
VAR_OUTPUT
    BUSY     : BOOL;
    ERR      : BOOL;
    ERRID    : UDINT;
    SymInfo  : SYMINFOSTRUCT;
END_VAR
```

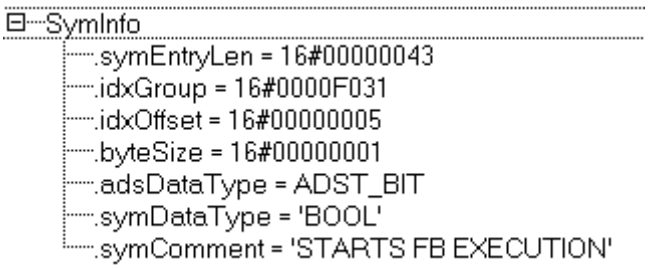
名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。
SymInfo	SYMINFOSTRUCT [▶ 343]	带有关于符号变量的附加信息的结构。

示例：

```
PROGRAM MAIN
VAR
    PLC_ReadSymInfoByName1  : PLC_ReadSymInfoByName;
    bStart AT%QX0.5         : BOOL; (*Starts FB execution*)
    bBusy                   : BOOL;
    bError                  : BOOL;
    iErrorId                : UDINT;
    SymInfo                 : SYMINFOSTRUCT;(*Structure with sombol information*)
END_VAR
```



在线视图：



通过这种方式获得的数据具有以下含义：

symEntryLen = 16#43：符号表中的条目的实际长度为 67 字节；

idxGroup = 16#F031：它是物理输出的 PLC 过程映像中的变量；

idxOffset = 16#5：变量位于字节偏移 0 和位偏移 5 处；

byteSize = 16#1：变量的值在内存中占 1 字节；

adsDataType = ADST_BIT：ADS 数据类型 ID；

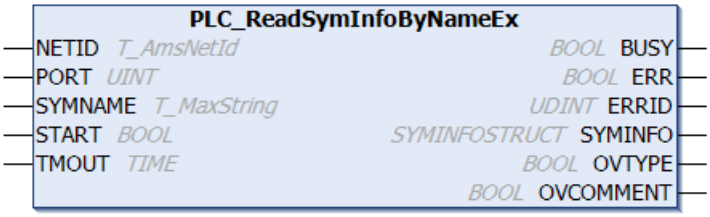
symDataType = BOOL：PLC 中的数据类型标识；

symComment = 'STARTS FB EXECUTION'：用户添加到变量定义行中的注释。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.75 PLC_ReadSymInfoByNameEx



功能块 PLC_ReadSymInfoByNameEx 具有与功能块 PLC_ReadSymInfoByName [▶ 128] 类似的功能。这两个功能块都可以通过符号名称读取符号信息。这两个功能块的区别在于，如果超出可用缓冲区大小，在这里描述的功能块不会返回错误，并且可能会输出不完整的信息。在这种情况下，注释和/或数据类型标识可能已被截断。另外两个输出变量（即 *OVTYPE* 和 *OVCOMMENT*）会对此做出指示，以便应用程序做出相应的响应。

输入

```
VAR_INPUT
    NETID    : T_AmsNetId;
    PORT     : T_AmsPort;
    SYMNAME  : T_MaxString;
    START    : BOOL;
    TMOUT    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要执行功能的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PORT	T_AmsPort	符号变量所属的 PLC 运行时系统的端口编号。
SYMNAME	T_MaxString	要读取其信息的 PLC 变量的符号名称（最多 255 个字符，包括完整路径，例如“MAIN.INIT_TASK.VARINT”）。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY      : BOOL;
    ERR       : BOOL;
    ERRID     : UDINT;
    SymInfo   : SYMINFOSTRUCT;
    OVTYPE    : BOOL;
    OVCOMMENT : BOOL;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。
SymInfo	SYMINFOSTRUCT [▶ 343]	带有关于符号变量的附加信息的结构。

名称	类型	描述
OVTYPE	BOOL	表示带有数据类型标识的字符串是否导致溢出（TRUE）。带有数据类型标识的字符串可能已被截断。
OVCOMMENT	BOOL	表示带有符号注释的字符串是否已导致溢出（TRUE）。带有注释的字符串可能已被截断。

示例：

请参见功能块文档：PLC_ReadSymInfoByName [▶ 128]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.76 PLC_Reset

功能块 PLC_Reset 可以用于重置 PLC 运行时系统。当 PLC 被重置时，PLC 变量将被填入其初始值，同时停止执行 PLC 程序。

输入

```
VAR_INPUT
    NETID    : T_AmsNetId;
    PORT     : T_AmsPort;
    RESET    : BOOL;
    TMOUT    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要执行功能的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PORT	T_AmsPort	符号变量所属的 PLC 运行时系统的端口编号。
RESET	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY         : BOOL;
    ERR          : BOOL;
    ERRID        : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.77 PLC_Start

功能块 PLC_Start 可以用于在 TwinCAT 计算机上启动 PLC 运行时系统。例如，该功能块可以用于在远程 PC 上启动 PLC。

输入

```
VAR_INPUT
    NETID    : T_AmsNetId;
    PORT     : T_AmsPort;
    START    : BOOL;
    TMOUT    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要启动 PLC 的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PORT	T_AmsPort	符号变量所属的 PLC 运行时系统的端口编号。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY         : BOOL;
    ERR          : BOOL;
    ERRID        : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.78 PLC_Stop

功能块 PLC_Stop 可以用于停止 TwinCAT 计算机上的 PLC 运行时系统。例如，该功能块可以用于在远程或本地 PC 上停止 PLC。

输入

```
VAR_INPUT
    NETID    : T_AmsNetId;
    PORT     : T_AmsPort;
    RESET    : BOOL;
    TMOUT    : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定要停止 PLC 的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
PORT	T_AmsPort	包含要停止的 PLC 运行时系统的 ADS 端口编号。
RESET	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY         : BOOL;
    ERR          : BOOL;
    ERRID        : UDINT;
END_VAR
```

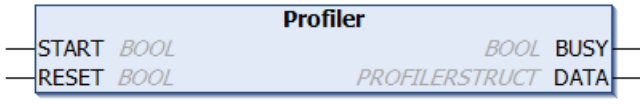
名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	ADS 错误编号 [▶ 376]	如果设置了 ERR 输出，则返回错误代码。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.79 Profiler

在 Windows CE 下的 PLC 中不提供该功能！



使用功能块 Profiler 可以测量 PLC 代码的执行时间。在内部调用 GETCPUACCOUNT 功能块的实例。测量由 START 输入的上升沿启动，并由下降沿停止。在内部对测量结果进行评估，然后以 PROFILERSTRUCT 类型的结构在 DATA 输出处进行进一步处理。除当前、最短和最长执行时间之外，该功能块还可以计算最近 10 次测量的平均执行时间。通过全局变量 MAX_AVERAGE_MEASURES [▶ 350] 可以将平均测量值的数量配置在 2 和 100 之间。以微秒为单位提供测量时间。输出变量 DATA.MeasureCycle [▶ 331] 可提供已执行的测量数量的信息。要测量 PLC 程序的特定程序段的执行时间，必须在要测量的程序段开始时通过 START 输入的上升沿启动测量，并在程序段结束时通过 START 输入的下降沿停止测量。如果 RESET 输入处与 START 处同时产生一个上升沿，则可以重置 DATA 输出处的所有值。当开始新的测量时，旧的测量值将被重置，并通过功能块的后续调用进行重新计算。

注释：

由于调用 GETCPUACCOUNT 功能块需要一定的时间，因此测量的时间可能与实际值不同。该时间取决于特定的计算机，并且包含在找到的时间中。

输入

```
VAR_INPUT
    START   : BOOL;
    RESET   : BOOL;
END_VAR
```

名称	类型	描述
START	BOOL	在该输入端出现上升沿（正跳变）时，开始测量执行时间。该输入的下降沿可停止测量，并导致重新计算当前、最短、最长和平均执行时间。同时，变量 DATA.MeasureCycle [▶ 331] 会递增。
RESET	BOOL	如果在该输入处与 START 输入处同时产生一个上升沿，则可以重置 DATA 输出处的所有变量。当前、最短、最长和平均执行时间的旧值将被重置，并在后续测量中进行重新计算。

输出

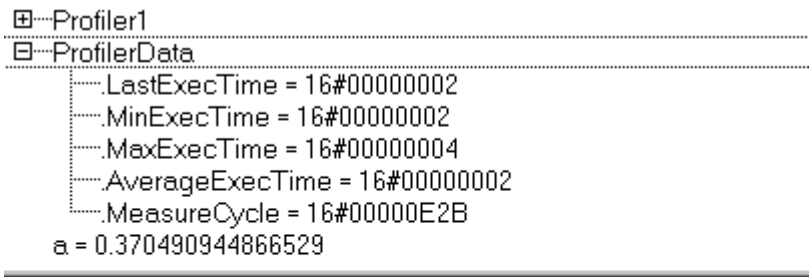
```
VAR_OUTPUT
    BUSY   : BOOL;
    DATA   : PROFILERSTRUCT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在测量过程开始时，该输出将被设置并保持设置状态，直至时间测量完成。在 BUSY 输出被重置后，在 DATA 输出中可以获取最新时间。
DATA	PROFILERSTRUCT [▶ 331]	带有测量时间（以 [µs] 为单位）的结构。

示例 1：

```
PROGRAM ProfilerTest_ST
VAR
   Profiler1     : PROFILER;
   ProfilerData  : PROFILERSTRUCT;
   a             : LREAL;
END_VAR
```

在线显示测量时间：



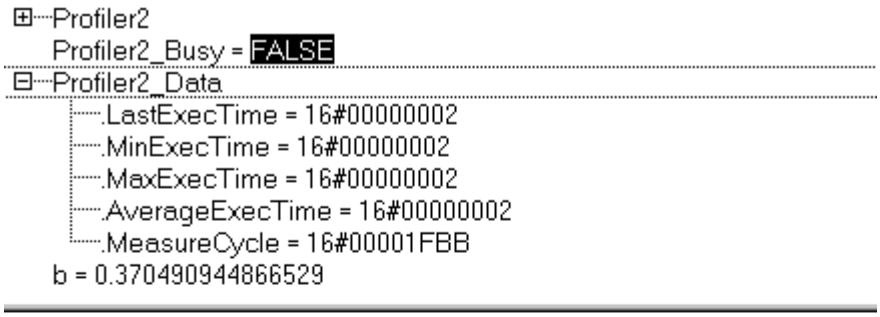

```
Profiler1(Start:=TRUE, Reset:=TRUE);

a := SIN(COS(TAN(12*0.4)));

Profiler1(Start:=FALSE);
ProfilerData:=Profiler1.Data;
```

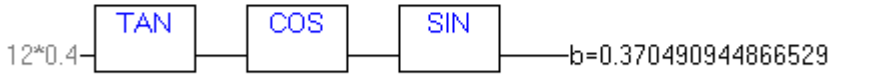
a = 0.370490944866529

示例 2：

```
PROGRAM ProfilerTest_FUP
VAR
   Profiler2      :PROFILER;
   Profiler2_Busy :BOOL;
   Profiler2_Data :PROFILERSTRUCT;
   b              :LREAL;
END_VAR
```

在线显示测量时间：





要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64）	Tc2_Utilities（系统）

3.80 RTC

功能块 RTC（实时时钟）可以用于在 TwinCAT PLC 中实现内部软件时钟。必须使用起始日期和时间对时钟进行初始化。在初始化后，每次调用功能块都会更新时间和日期。CPU 系统时钟可用于计算当前时间和日期。在每个PLC周期中都调用该功能块，以便能够计算当前时间。在该功能块的输出处，以通常的 DATE_AND_TIME（DT）格式可以提供当前日期和时间。在一个 PLC 程序中可以创建多个 RTC 功能块实例。

RTC 时间与参考时间的偏差

系统的工作方式意味着 RTC 时间可能与参考时间不同。偏差取决于 PLC 的循环时间、基本系统时钟周期的值以及所使用的硬件。

为避免出现较大的偏差，应该周期性地同步 RTC 实例（例如，使用无线电时钟或本地 Windows 系统时间）。通过 SNTP 协议可以同步本地 Windows 系统时间与参考时间。

输入

```
VAR_INPUT
   EN  : BOOL;
   PDT : DATE_AND_TIME;
END_VAR
```

名称	类型	描述
EN	BOOL	通过该输入的上升沿，使用指定的日期、时间和毫秒对 RTC_EX2 功能块进行重新初始化。
PDT	DATE_AND_TIME	（预设日期和时间） 功能块的日期和时间的初始化值。EN 输入的上升沿将导致功能块采用该值。

输出

```
VAR_OUTPUT
   Q    : BOOL;
   CDT  : DATE_AND_TIME;
END_VAR
```

名称	类型	描述
Q	BOOL	如果已经对功能块至少进行一次初始化，则此输出被置位。如果已设置输出，则 PDT 输出处的日期、时间和毫秒值均有效。
CDT	TIMESTRUCT [▶ 347]	RTC_EX2 实例的 当前日期和时间。只有在调用功能块时才会更新 CDT 输出。因此，在每个 PLC 周期中会调用一次功能块的实例。

示例：

请参见：示例：软件时钟（RTC、RTC_EX、RTC_EX2）。 [▶ 371]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.81 RTC_EX

功能块 RTC_EX（扩展实时时钟）允许在 TwinCAT PLC 中实现内部软件时钟。必须使用起始日期和时间对时钟进行初始化。在初始化后，每次调用功能块都会更新时间和日期。CPU 系统时钟可用于计算当前时间和日期。在每个PLC周期中都调用该功能块，以便能够计算当前时间。在功能块的输出处，以常见的 DATE_AND_TIME（DT）格式可以提供当前日期和时间。与 RTC [▶ 135] 功能块相比，RTC_EX 的精度为一毫秒。在一个 PLC 程序中可以创建多个 RTC_EX 功能块实例。

RTC_EX 时间与参考时间的偏差

系统的工作方式意味着 RTC_EX 时间可能与参考时间不同。偏差取决于 PLC 的循环时间、基本系统时钟周期的值以及所使用的硬件。

为避免出现较大的偏差，应该周期性地同步 RTC_EX 实例（例如，使用无线电时钟或本地 Windows 系统时间）。通过 SNTP 协议可以同步本地 Windows 系统时间与参考时间。

输入

```
VAR_INPUT
   EN    : BOOL;
   PDT   : DATE_AND_TIME;
   PMSEK : DWORD;
END_VAR
```

名称	类型	描述
EN	BOOL	通过该输入的上升沿，使用指定的日期、时间和毫秒对 RTC_EX2 功能块进行重新初始化。
PDT	TIMESTRUCT [▶ 347]	（预设日期和时间） 功能块的日期和时间的初始化值。EN 输入的上升沿将导致功能块采用该值。
PMSEK	DWORD	（预设微秒） 微秒的初始化值。EN 输入的上升沿将导致功能块采用该值。

输出

```
VAR_OUTPUT
   Q     : BOOL;
   CDT   : DATE_AND_TIME;
   CMSEK : DWORD;
END_VAR
```

名称	类型	描述
Q	BOOL	如果已经对功能块至少进行一次初始化，则此输出被置位。如果该输出被置位，则 PDT 和 CMICRO 输出处的日期、时间和毫秒值均有效。
CDT	DATE_AND_TIME	RTC_EX2 实例的 当前日期和时间。只有在调用功能块时才会更新 CDT 输出。因此，在每个 PLC 周期中会调用一次功能块的实例。
CMSEK	DWORD	（当前毫秒）毫秒输出。

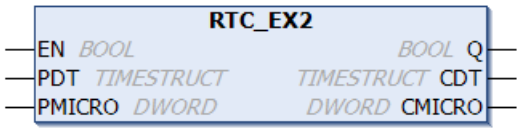
示例：

请参见：示例：软件时钟（RTC、RTC_EX、RTC_EX2）。 [▶ 371]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.82 RTC_EX2



功能块 RTC_EX2（扩展实时时钟）允许在 TwinCAT PLC 中实现内部软件时钟。必须使用起始日期和时间对时钟进行初始化。在初始化后，每次调用功能块都会更新时间和日期。CPU 系统时钟可用于计算当前时间和日期。在每个PLC周期中都调用该功能块，以便能够计算当前时间。在功能块的输出处，以 Windows 系统时间格式可以提供当前日期和时间。与 RTC [▶ 135] 功能块相比，RTC_EX2 的精度为一微秒。在一个 PLC 程序中可以创建多个 RTC_EX2 功能块实例。

RTC_EX2 时间与参考时间的偏差

系统的工作方式意味着 RTC_EX2 时间可能与参考时间不同。偏差取决于 PLC 的循环时间、基本系统时钟周期的值以及所使用的硬件。

为避免出现较大的偏差，应该周期性地同步 RTC_EX2 实例（例如，使用无线电时钟或本地 Windows 系统时间）。通过 SNTP 协议可以同步本地 Windows 系统时间与参考时间。

输入

```
VAR_INPUT
   EN       : BOOL;
   PDT      : TIMESTRUCT;
   PMICRO   : DWORD;
END_VAR
```

名称	类型	描述
EN	BOOL	通过该输入的上升沿，使用指定的日期、时间和毫秒对 RTC_EX2 功能块进行重新初始化。
PDT	TIMESTRUCT [▶ 347]	（预设日期和时间） 功能块的日期和时间的初始化值。EN 输入的上升沿将导致功能块采用该值。
PMICRO	DWORD	（预设微秒） 微秒的初始化值。EN 输入的上升沿将导致功能块采用该值。

输出

```
VAR_OUTPUT
   Q        : BOOL;
   CDT      : TIMESTRUCT;
   CMICRO   : DWORD;
END_VAR
```

名称	类型	描述
Q	BOOL	如果已经对功能块至少进行一次初始化，则此输出被置位。如果该输出被置位，则 PDT 和 CMICRO 输出处的日期、时间和毫秒值均有效。
CDT	TIMESTRUCT [▶ 347]	RTC_EX2 实例的 当前日期和时间。只有在调用功能块时才会更新 CDT 输出。因此，在每个 PLC 周期中会调用一次功能块的实例。
CMICRO	DWORD	（当前微秒） 微秒输出

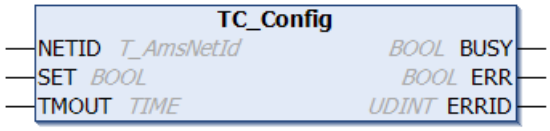
示例：

请参见：示例：软件时钟（RTC、RTC_EX、RTC_EX2）。 [▶ 371]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.83 TC_Config



通过功能块“TC_Config”可以将处于 RUN 模式（绿色 TwinCAT 系统图标）的 TwinCAT 系统切换到 CONFIG 模式（蓝色 TwinCAT 系统图标）。如果系统已处于 CONFIG 模式，则它首先会切换到 STOP 模式（红色 TwinCAT 系统图标），然后再切换到 CONFIG 模式。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    SET     : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	要执行 ADS 命令的 TwinCAT 计算机的网络地址。对于本地计算机，可以输入一个空字符串。
SET	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

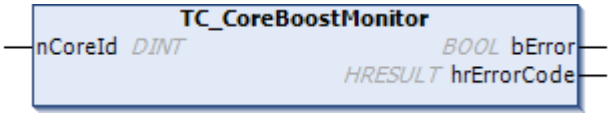
```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现 ADS 错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	UDINT	在设置 ERR 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.84 TC_CoreBoostMonitor



TwinCAT Core Boost 功能提供了提高单个实时内核时钟频率的选项（请参见实时设置的设置选项卡）。这会导致实时内核的功耗增加，从而也可能会导致温度升高。为确保系统设定的限值不会永久性被超出，建议对系统进行监控。

如果使用并激活 TwinCAT Core Boost，则 `TC_CoreBoostMonitor` 功能块可以用于查询有关单个 CPU 内核的信息。也可使用该功能块对各个CPU内核的频率节流（调控）情况进行监控。

硬件

该功能目前仅支持第 11 代及以上的 Intel® Core™ i CPU。

输入

名称	类型	描述
nCoreId	DINT	该输入可用于设置要监控的 CPU 内核。ID 从零开始（0..n）。 如果指定 -1，则会自动设置处理功能块调用或其方法的内核。

输出

名称	类型	描述
bError	BOOL	如果发生错误，则变为 TRUE。
hrErrorCode	HRESULT	返回已发生错误的错误代码。

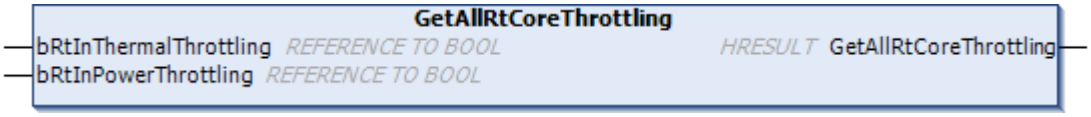
方法

名称	描述
GetAllRtCoreThrottling [▶ 140]	通过引用返回信息，指示是否由于热限制或功耗限制而被触发，导致任意实时内核上正在发生频率节流（调控）。
GetCoreFrequency [▶ 140]	通过引用返回已定义实时内核的已配置和当前时钟频率。
GetCoreTemperature [▶ 141]	通过引用返回当前温度、自实时开始以来出现的最高温度以及已定义实时内核的温度限值。
GetCoreThrottling [▶ 141]	通过引用返回信息，指示是否由于热限制或功耗限制被超出，导致当前指定的实时内核正在被节流（调控）。
GetPowerConsumption [▶ 142]	通过引用返回已定义实时内核的组件的当前功耗或功耗限值。对于同一个组件的所有实时内核，值都是相同的。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4026.6	PC 或 CX（x64）	Tc2_Utilities（系统） >= 3.7.4.0

3.84.1 GetAllRtCoreThrottling



该方法通过引用返回任何实时内核是否由于超出热量或功率限制而发生频率限制。

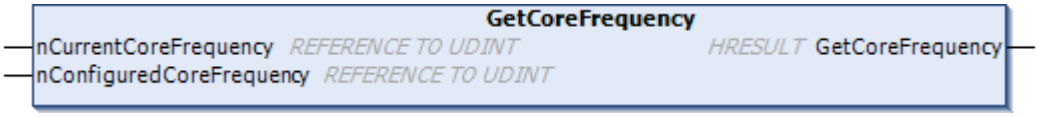
返回值

名称	类型	描述
GetAllRtCoreThrottling	HRESULT	如果出现错误，则返回已发生错误的错误代码。

输入

名称	类型	描述
bRtInThermalThrottling	REFERENCE_TO_BOOL	如果其中一个实时内核当前由于超出温度限值而发生频率限制，则为 TRUE。
bRtInPowerThrottling	REFERENCE_TO_BOOL	如果其中一个实时内核当前由于超出平台组件的性能限值而发生频率限制，则为 TRUE。

3.84.2 GetCoreFrequency



该方法通过引用返回已定义实时内核的已设置和当前时钟频率。如果内核必须限制其频率，则这些参数可能会有所不同。

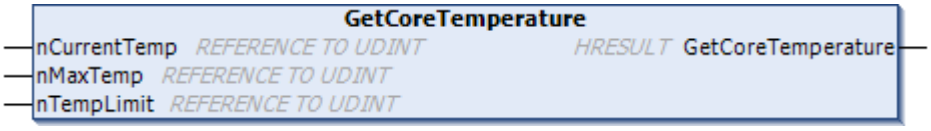
返回值

名称	类型	描述
GetCoreFrequency	HRESULT	如果出现错误，则返回已发生错误的错误代码。

输入

名称	类型	描述
nCurrentCoreFrequency	REFERENCE TO UDINT	返回 CPU 内核的当前频率。
nConfiguredCoreFrequency	REFERENCE TO UDINT	返回 CPU 内核的设定频率。

3.84.3 GetCoreTemperature



该方法通过引用返回当前温度、自实时开始以来出现的最高温度以及已定义实时内核的温度限值。

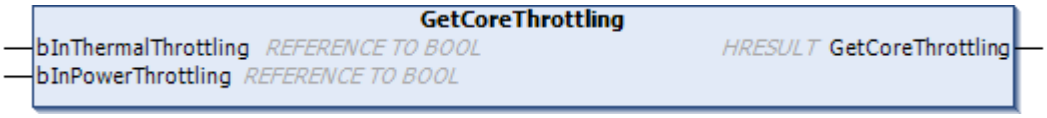
返回值

名称	类型	描述
GetCoreTemperature	HRESULT	如果出现错误，则返回已发生错误的错误代码。

输入

名称	类型	描述
nCurrentTemp	REFERENCE TO UDINT	返回 CPU 内核的当前温度 [°C]。
nMaxTemp	REFERENCE TO UDINT	返回自 TwinCAT XAR 启动以来观察到的 CPU 内核的最高温度 [°C]。
nTempLimit	REFERENCE TO UDINT	返回 CPU 内核的温度限值 [°C]。超过该限值会导致内核的频率限制。

3.84.4 GetCoreThrottling



该方法通过引用返回已定义实时内核当前是否由于超出热量或功率限值而受到限制。

返回值

名称	类型	描述
GetCoreThrottling	HRESULT	如果出现错误，则返回已发生错误的错误代码。

输入

名称	类型	描述
bInThermalThrottling	REFERENCE TO BOOL	如果 CPU 内核当前由于超出温度限值而发生频率限制，则为 TRUE。
bInPowerThrottling	REFERENCE TO BOOL	如果 CPU 内核当前由于超出平台组件 的性能限值而发生频率限制，则为 TRUE。

3.84.5 GetPowerConsumption

该方法通过引用返回已定义实时内核的组件的当前功耗或功耗限值。对于同一个组件的所有实时内核，值都是相同的。

返回值

名称	类型	描述
GetPowerConsumption	HRESULT	如果出现错误，则返回已发生错误的错误代码。

输入

名称	类型	描述
nCurrentPackagePowerConsumption	REFERENCE TO UDINT	返回平台组件的总功率，以瓦特为单位。
nPackagePowerLimit	REFERENCE TO UDINT	返回平台组件的性能限值，以瓦特为单位。超过这些限值可能会导致单个 CPU 内核的频率限制。

3.85 TC_CpuUsage

功能块 TC_CpuUsage 允许确定 TwinCAT 系统的当前 CPU 使用率。该功能与实时设置下的 TwinCAT 系统菜单中显示的 CPU 使用率相对应。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	要执行 ADS 命令的 TwinCAT 计算机的网络地址。对于本地计算机，可以输入一个空字符串
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

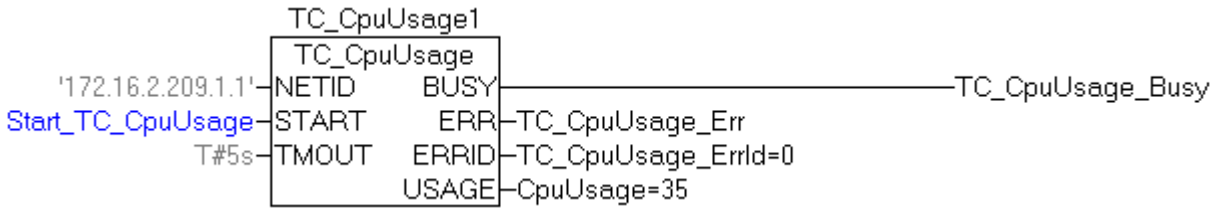
```
VAR_OUTPUT
    BUSY   : BOOL;
    ERR    : BOOL;
    ERRID  : UDINT;
    USAGE  : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。

名称	类型	描述
ERR	BOOL	如果在命令传输过程中出现 ADS 错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	UDINT	在设置 ERR 输出时，返回 ADS 错误编号 [▶ 376]。
USAGE	UDINT	TwinCAT 系统的当前 CPU 使用率（%）。

示例：

```
PROGRAM MAIN
VAR
    TC_CpuUsage1        : TC_CpuUsage;
    Start_TC_CpuUsage   : BOOL;
    TC_CpuUsage_Busy    : BOOL;
    TC_CpuUsage_Err     : BOOL;
    TC_CpuUsage_ErrId   : UDINT;
    CpuUsage            : UDINT;
END_VAR
```



在该示例中，TwinCAT 系统使用了总可用 CPU 计算时间的 35%。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.86

3.87 TC_Restart

功能块 TC_Restart 可以用于重新启动 TwinCAT 系统。该功能与 TwinCAT 系统菜单（位于 Windows 任务栏右侧）上的重新启动命令相对应。重新启动 TwinCAT 系统需要首先停止 TwinCAT 系统，然后立即重新启动。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    RESTART : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	要执行 ADS 命令的 TwinCAT 计算机的网络地址。对于本地计算机，可以输入一个空字符串
RESTART	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现 ADS 错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	UDINT	在设置 ERR 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.88 TC_Stop

功能块 TC_Stop 可以用于停止 TwinCAT 系统。该功能与 TwinCAT 系统菜单（位于 Windows 任务栏右侧）上的停止命令相对应。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    STOP    : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定一个字符串，其中包含要执行 TwinCAT 系统停止的 TwinCAT 计算机的网络地址。如果要停止的 TwinCAT 系统在本地计算机上，可以输入一个空字符串。
STOP	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

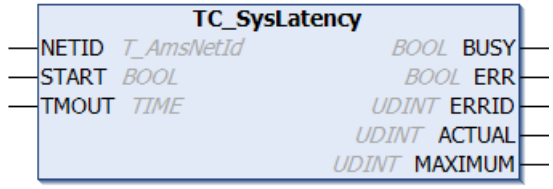
```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现 ADS 错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	UDINT	在设置 ERR 输出时，返回 ADS 错误编号 [▶ 376]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.89 TC_SysLatency



功能块 TC_SysLatency 可以用于确定 TwinCAT 系统的当前和最大延迟时间。该功能与实时设置下的 TwinCAT 系统菜单中显示的 TwinCAT 延迟时间相对应。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	在这里可以指定一个字符串，其中包含要确定其延迟时间的 TwinCAT 计算机的网络地址。对于本地计算机，可以指定一个空字符串。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
    ACTUAL  : UDINT;
    MAXIMUM : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现 ADS 错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	UDINT	在设置 ERR 输出时，返回 ADS 错误编号 [▶ 376]。
ACTUAL	UDINT	TwinCAT 系统的当前延迟时间，以 µs 为单位。
MAXIMUM	UDINT	TwinCAT 系统的最大延迟时间，以 µs 为单位（自 TwinCAT 系统上次启动以来的最大延迟时间）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

3.90

3.91 WritePersistentData

如果在 PLC 运行时系统中定义了持久变量，则在停止/关闭 TwinCAT 系统时（在最后一个 PLC 周期之后），它们的当前值通常会保存在 TwinCAT\Boot 文件夹中的 .bootdata 文件中。在将当前的持久性数据写入文件之前，应将系统的旧 .bootdata 文件重命名为 .bootdata-old，以备份旧的持久性数据。

对于每个已配置的运行时系统，都会创建一个文件。

在下一次系统启动时，将读取 .bootdata 文件，并使用该文件中的值初始化运行时系统中的持久化变量。

如果在系统启动时包含持久性数据的文件（.bootdata）不存在，则会读取持久化数据的备份文件（.bootdata-old）。例如若未配备不间断电源（UPS）的工业 PC（IPC）遭遇断电故障，导致 TwinCAT 系统无法正常关闭。

通过功能块 WritePersistentData，您可以从 PLC 程序中启动持久性数据的保存，并确保包含持久性数据的最新 .bootdata 文件可用。PORT 输入参数可指定要保存其持久性数据的运行时系统。

输入

```
VAR_INPUT
    NETID   : T_AmsNetId;
    PORT    : T_AmsPort;
    START   : BOOL;
    TMOUT   : TIME := DEFAULT_ADS_TIMEOUT;
END_VAR
```

名称	类型	描述
NETID	T_AmsNetID	要执行 ADS 命令的 TwinCAT 计算机的网络地址。对于本地计算机，可以输入一个空字符串
PORT	T_AmsPort	要保存其持久性数据的 PLC 运行时系统的 ADS 端口编号。
START	BOOL	功能块由该输入的正沿（上升沿）启用。
TMOUT	TIME	在执行 ADS 命令时不得超过的超时时间。

输出

```
VAR_OUTPUT
    BUSY    : BOOL;
    ERR     : BOOL;
    ERRID   : UDINT;
END_VAR
```

名称	类型	描述
BUSY	BOOL	在启用功能块时，该输出将被置位并保持置位状态，直至收到反馈。
ERR	BOOL	如果在命令传输过程中出现 ADS 错误，则在 BUSY 输出被复位后，该输出将被置位。
ERRID	UDINT	在设置 ERR 输出时，返回 ADS 错误编号 [▶ 376]。

示例：

```
PROGRAM MAIN
VAR
    bStart                  : BOOL;
    bError                  : BOOL;
    bBusy                   : BOOL;
    nErrorId                : UDINT;
```

```
    fbWritePersistentData : WritePersistentData;
    fbR_Trig              : R_TRIG;
END_VAR
```

```
VAR PERSISTENT
    perA  : INT;
    perB  : BOOL;
    perC  : BYTE;
    perD  : STRING;
    perE  : ARRAY[0..10] OF INT;
    perF  : ARRAY[0..10] OF UDINT;
END_VAR
```

```
fbR_Trig( CLK:=bStart );
IF fbR_Trig.Q THEN
    perA := 24443;
    perB := TRUE;
    perC := 7;
    perD := 'Switch ON/OFF';
    perE[ 0 ] := 1;
    perE[ 10 ] := 11;
    perF[ 0 ] := 263;
    perF[ 10 ] := 23323;
    fbWritePersistentData(NETID:='', PORT:=851, START:=bStart, TMOUT:=T#1s );
ELSE
    fbWritePersistentData( START:=FALSE);
END_IF;

bBusy := fbWritePersistentData.BUSY;
bError := fbWritePersistentData.ERR;
nErrorId := fbWritePersistentData.ERRID;
```

另请参见：附录 > 写入持久性数据时的系统行为 [▶ 373]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4 函数

4.1 时间函数

4.1.1 DT_TO_FILETIME64

函数“DT_TO_FILETIME64”可以用于将 DATE_AND_TIME 格式（DT）的 PLC 变量转换为 FILETIME 格式（64 位）。

函数 DT_TO_FILETIME64：T_FILETIME64 [▶ 344]

输入

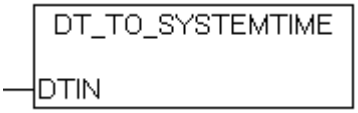
```
VAR_INPUT
    DTIN  : DT;
END_VAR
```

名称	类型	描述
DTIN	DT	以 DATE_AND_TIME 格式表示的要转换的日期和时间

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

4.1.2 DT_TO_SYSTEMTIME



“DT_TO_SYSTEMTIME”函数允许将 DATE_AND_TIME 格式（DT）的 PLC 变量转换为 Windows 系统时间结构。系统时间的分辨率为 1 ms，而 DATE_AND_TIME 的分辨率为 1 s。因此，系统时间结构中的“wMilliseconds”变量总是返回零值。

函数 DT_TO_SYSTEMTIME：TIMESTRUCT [▶ 347]

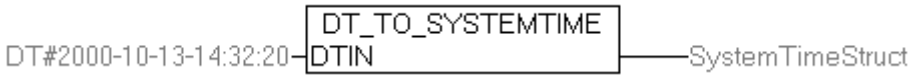
输入

```
VAR_INPUT
    DTIN  : DT;
END_VAR
```

名称	类型	描述
DTIN	DT	以 DATE_AND_TIME 格式表示的要转换的日期和时间

示例：

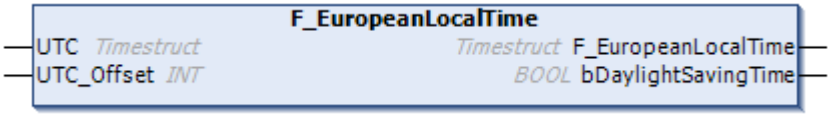
```
PROGRAM SystemTimeTest
VAR
    SystemTimeStruct : TIMESTRUCT;
END_VAR
```



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.3 F_EuropeanLocalTime



该函数将 UTC 时间（结构化系统时间格式）转换为本地时间（结构化系统时间格式），同时考虑指定的偏移信息。该函数仅可在欧洲使用，对世界任何时区均无效。

功能块 FB_SystemTimeToTzSpecificLocalTime [▶ 112] 具有类似的功能，不同之处在于它可以在全球范围内使用。F_EuropeanLocalTime 在欧洲范围内的应用具有处理时间更短的优势，因此该函数特别适用于小型控制系统。

除本地时间外，该函数还可提供有关夏令时或标准时间的信息。该功能块使用时区信息或偏移量来计算在本地时间中所需的时间步长（夏令时/标准时间转换）。

返回值

名称	类型	描述
F_EuropeanLocalTime	TIMESTRUCT [▶ 347]	本地时间的输出

输入

```
VAR_INPUT
    UTC        : TIMESTRUCT;
    UTC_Offset : INT;
END_VAR
```

名称	类型	描述
UTC	TIMESTRUCT [▶ 347]	要转换的 UTC 时间（结构化系统时间格式）。
UTC_Offset	INT	时区偏移，以分钟为单位。

输出

```
VAR_OUTPUT
    bDaylightSavingTime   : BOOL;
END_VAR
```

名称	类型	描述
bDaylightSavingTime	BOOL	附加的夏令时/标准时间信息。如果输出为 TRUE，则这是夏令时。否则，这是标准时间。

示例：

```
PROGRAM MAIN
VAR
    in     : TIMESTRUCT := ( wYear := 2011, wMonth := 4, wDay := 29, wHour := 14, wMinute := 46, wSec
ond := 31, wMilliseconds := 99 );(* UTC time *)
    out    : TIMESTRUCT; (* Local time result is:= ( wYear := 2011, wMonth := 4, wDay := 29, wHour :=
 16, wMinute := 46, wSecond := 31, wMilliseconds := 99 ) *)
    bSummerTime : BOOL;
END_VAR
```

```
out := F_EuropeanLocalTime( UTC := in, UTC_Offset := 60 (*Germany*), bDaylightSavingTime =>
bSummerTime );
```

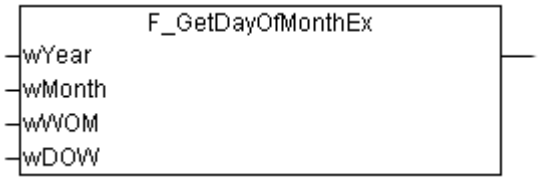
其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime [▶ 29]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTimeTimeToTzSpecificLocalTime [▶ 27]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- F_TranslateFileTimeBias [▶ 262]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT >= v3.1.4024.63	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= v3.9.1.0

4.1.4 F_GetDayOfMonthEx



该函数计算特定月份和年份的第一个、第二个……星期几的日期（例如，2011 年 1 月第二个星期一的日期）。

函数 F_GetDayOfMonthEx：WORD

输入

```
VAR_INPUT
    wYear  : WORD(1601..30827);
    wMonth : WORD(1..12);
    wWOM   : WORD(1..5);
    wDOW   : WORD(0..6);
END_VAR
```

名称	类型	描述
wYear	WORD	年份（1601 至 30827）
wMonth	WORD	月份（1 至 12）
wWOM	WORD	每月的周数（1 至 5）。值 1 代表第 1 周，2 代表第 2 周，5 代表最后一周（即使该月没有 5 周）。
wDOW	WORD	星期几（0 至 6）。0 = 星期日，1 = 星期一…… 6 = 星期六

返回参数	描述
0	错误、错误或无效的函数参数
> 0	无错误。月份中的某一天

示例：

该示例确定的日期是 2011 年 8 月的第二个星期一。结果为：8。

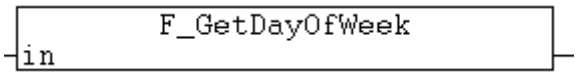
```
PROGRAM P_Dok_F_GetDayOfMonthEx
VAR
    wYear   : WORD := 2011;
    wMonth  : WORD := 8;
    wWOM    : WORD(1..5) := 2;(* Week of month: 2 = Second week *)
    wDOW    : WORD(0..6) := 1;(* Day of week 1 = Monday *)
    wDay    : WORD;(* Day of month *)
END_VAR
```

```
wDay := F_GetDayOfMonthEx( wYear, wMonth, wWOM, wDOW );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.5 F_GetDayOfWeek



该函数根据 DIN 1355 / ISO 8601 标准返回星期几。根据该标准，星期几的编号如下：星期一 = 1，星期二 = 2，…… 星期日 = 7。

函数 F_GetDayOfWeek：WORD

输入

```
VAR_INPUT
    in : DT;
END_VAR
```

名称	类型	描述
in	DT	要确定星期几的日期。

示例：

```
PROGRAM MAIN
VAR
    dtFirst   : DT := DT#2008-01-01-00:00;
    dayOfWeek : WORD;
END_VAR
```

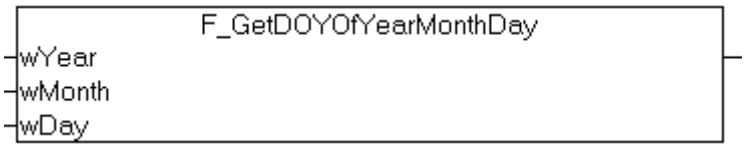
```
dayOfWeek := F_GetDayOfWeek(dtFirst);
```

结果为 2（星期二）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.6 F_GetDOYOfYearMonthDay



该函数计算一年中的天数。

函数 F_GetDOYOfYearMonthDay：WORD

输入

```
VAR_INPUT
    wYear  : WORD;
    wMonth : WORD;
    wDay   : WORD;
END_VAR
```

名称	类型	描述
wYear	WORD	年份（0 ~ 2999）
wMonth	WORD	月份（1 ~ 12）
wDay	WORD	日（1 ~ 31）

返回参数	描述
0	错误，错误的 wYear、wMonth 或 wDay 参数值
> 0	无错误。一年中的天数（1 ~ 366）

示例：

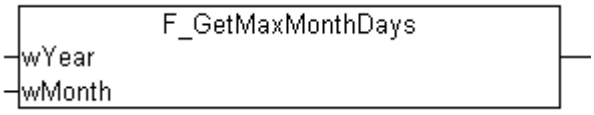
```
PROGRAM P_TEST_DOY
VAR
    wYear  : WORD;
    wDOY   : WORD;
    wMonth : WORD;
    wDay   : WORD;
END_VAR
```

```
wYear := 2009;
wMonth := 1;
wDay := 31;
wDOY := F_GetDOYOfYearMonthDay( wYear, wMonth, wDay );(* wDOY = 31 *)

wYear := 2009;
wMonth := 2;
wDay := 1;
wDOY := F_GetDOYOfYearMonthDay( wYear, wMonth, wDay );(* wDOY = 32 *)

wYear := 2009;
wMonth := 3;
wDay := 1;
wDOY := F_GetDOYOfYearMonthDay( wYear, wMonth, wDay );(* wDOY = 60 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.7 F_GetMaxMonthDays



该函数返回特定月份和年份中的该月中的最大天数。

函数 F_GetMaxMonthDays：WORD

输入

```
VAR_INPUT
    wYear  : WORD;
    wMonth : WORD;
END_VAR
```

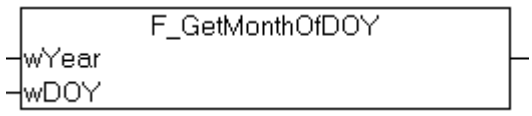
名称	类型	描述
wYear	WORD	年份
wMonth	WORD	月份（1 至 12）

返回参数	描述
0	错误，错误的 wMonth 参数值
> 0	无错误。该月中的最大天数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.8 F_GetMonthOfDOY



该函数根据一年中的天数计算月份。

函数 F_GetMonthOfDOY：WORD

输入

```
VAR_INPUT
    wYear  : WORD;
    wDOY   : WORD;
END_VAR
```

名称	类型	描述
wYear	WORD	年份（0 ~ 2999）
wDOY	WORD	要确定其月份的指定年份中的天数（1 ~ 366）。

返回参数	描述
0	错误，错误的 wYear 或 wDOY 参数值。
> 0	无错误。月份（1 ~ 12）。

示例：

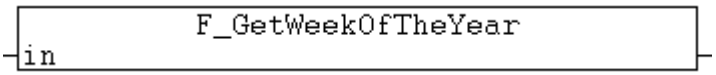
```
PROGRAM P_TEST_DOY
VAR
    wYear  : WORD;
    wDOY   : WORD;
    wMonth : WORD;
END_VAR
```

```
wYear   := 2009;
wDOY    := 31;
wMonth  := F_GetMonthOfDOY( wYear, wDOY );(* wMonth = 1 *)

wYear   := 2009;
wDOY    := 32;
wMonth  := F_GetMonthOfDOY( wYear, wDOY );(* wMonth = 2 *)

wYear   := 2009;
wDOY    := 60;
wMonth  := F_GetMonthOfDOY( wYear, wDOY );(* wMonth = 3 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.9 F_GetWeekOfTheYear



该函数根据 DIN 1355 / ISO 8601 标准返回指定日期的日历周数。

- 第一个日历周被定义为**至少包含新年中的 4 天的第一周（DIN 1355 / ISO 8601）**。
- 日历周从星期一开始。每个日历周包含 7 天。
- 第一个日历周中的返回值为数字 1。
- 12 月 29 日、30 日和 31 日可能属于下一年的第一个日历周。
- 1 月 1 日、2 日和 3 日可能属于上一年的最后一个日历周。

函数 F_GetWeekOfTheYear：WORD

输入

```
VAR_INPUT
    in : DT;
END_VAR
```

名称	类型	描述
in	DT	要确定其日历周的日期。

示例：

```
PROGRAM MAIN
VAR
    dtNow      : DT := DT#2008-03-17-12:00;
    weekOfYear : WORD;
END_VAR
weekOfYear := F_GetWeekOfTheYear(dtNow);
```

结果为 12。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.10 F_TranslateFileTime64Bias

该函数根据指定的偏差时间偏移，将输入时间转换为另一个时区中的时间。例如，该函数可以用于将本地时间转换为 UTC 时间（世界标准时间），反之亦然。

函数 F_TranslateFileTime64Bias：T_FILETIME64 [▶ 344]

输入

```
VAR_INPUT
    in    : T_FILETIME64;
    bias  : DINT;
    toUTC : BOOL;
END_VAR
```

名称	类型	描述
in	T_FILETIME64 [▶ 344]	要转换的输入时间。

名称	类型	描述
bias	DINT	UTC 时间与本地时间之间的差值，以分钟为单位（允许使用正值或负值）
toUTC	BOOL	该参数可以用于指定输入时间的转换方向。

toUTC	方向	系统计算公式
FALSE	UTC -> 本地时间	本地时间 := UTC - 偏差
TRUE	本地时间 -> UTC	UTC := 本地时间 + 偏差

示例：

in 变量包含要转换的时间。*bToUTC* 变量决定了转换方向。如果 *bToUTC*= TRUE，则本地时间将转换为 UTC 时间；如果 *bToUTC*= FALSE，则 UTC 时间将转换为本地时间。*WEST_EUROPE_TZI* 常量包含西欧的时区信息。根据常量中的时区信息和当前的 bDST 设置（夏令时）可计算得出所需的偏差值。或者，通过功能块可以确定 TwinCAT 系统的当前时区信息：FB_GetTimeZoneInformation [▶ 78]。

重要说明：由于在线模式下的可视化控件选项，因此为输入时间选择了数据类型 DT。不建议转换为其他时间格式，因为转换功能可能需要大量计算。

```
PROGRAM MAIN
VAR
    bDST    : BOOL := TRUE;(* TRUE => Daylight saving time, FALSE => Standard time *)
    bToUTC  : BOOL := FALSE;
(* TRUE => Convert local time to UTC time, FALSE => Convert UTC time to local time *)
    in      : DT := DT#2011-08-29-15:15:31;
    out     : DT;
    bias    : DINT;
END_VAR
```

```
IF bDST THEN
    bias := WEST_EUROPE_TZI.bias + WEST_EUROPE_TZI.daylightBias;
ELSE
    bias := WEST_EUROPE_TZI.bias + WEST_EUROPE_TZI.standardBias;
END_IF

out := FILETIME64_TO_DT( F_TranslateFileTime64Bias( DT_TO_FILETIME64( in ), bias, bToUTC ) );
```

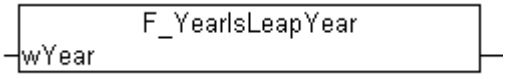
其他时间和时区函数和功能块：

- FB_TzSpecificLocalTimeToSystemTime [▶ 115]
- FB_TzSpecificLocalTimeToFileTime64 [▶ 113]
- FB_SystemTimeToTzSpecificLocalTime [▶ 112]
- FB_FileTime46ToTzSpecificLocalTime [▶ 61]
- FB_GetTimeZoneInformation [▶ 78]
- FB_SetTimeZoneInformation [▶ 109]
- NT_SetLocalTime [▶ 122]
- NT_GetTime [▶ 120]
- NT_SetTimeToRTCTime [▶ 123]
- FB_LocalSystemTime [▶ 90]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

4.1.11 F_YearIsLeapYear



该函数可确定某一年份是否为闰年。

函数 F_YearIsLeapYear：BOOL

输入

```
VAR_INPUT
    wYear : WORD;
END_VAR
```

名称	类型	描述
wYear	WORD	年份

返回参数	描述
TRUE	闰年
FALSE	无闰年

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.12 FILETIME64_TO_DT

函数“FILETIME64_TO_DT”将 FILETIME 格式的时间转换为 DATE_AND_TIME 格式（DT）。与 FILETIME 格式相比，DT 格式具有较小的值范围，而且只能提供秒级精度。因此，要转换的 FILETIME 值受限。允许的最小值与值 *DT#1970-01-01-00:00:00* 对应，最大值与值 *DT#2106-02-06-06:28:15* 对应。在转换时不考虑毫秒，并将毫秒向下取整为相应的 DATE_AND_TIME 返回值。

函数 FILETIME64_TO_DT：DT

输入

```
VAR_INPUT
    fileTime : T_FILETIME64;
END_VAR
```

名称	类型	描述
fileTime	T_FILETIME64 [▶ 344]	以 FILETIME 格式表示的要转换的时间

示例：

```
PROGRAM MAIN
VAR
    timeAsFileTime : T_FILETIME64;
    timeAsDT       : DT;
END_VAR
```

```
timeAsFileTime := F_GetSystemTime();
timeAsDT := FILETIME64_TO_DT( timeAsFileTime );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

4.1.13 FILETIME64_TO_ISO8601

该函数将 T_FILETIME64 格式的 Windows 系统时间转换为 ISO 8601 格式的字符串。

结果与以下方案相对应：**YYYY-MM-DDThh:mm:ss.xxxTZD**

函数 FILETIME64_TO_ISO8601：STRING(39)

输入

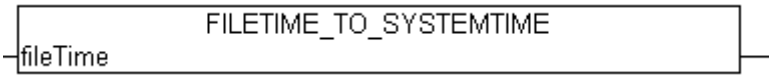
```
VAR_INPUT
    fileTime   : T_FILETIME64; (* Time to be converted (file time format), 64-
bit value representing the number of 100-nanosecond intervals since January 1, 1601 *)
    nBias      : INT;          (* Specifies the current bias, in minutes, for local time translation
 on this computer.
                                The bias is the difference, in minutes, between Coordinated Universa
l Time (UTC) and local time.
                                UTC = local time + bias *)
    bUTC       : BOOL;         (* Specifies whether the fileTime is UTC or local time. *)
    nPrecision : USINT(0..9);  (* Precision. Number of decimal places of seconds. (0..9) *)
END_VAR
```

名称	类型	描述
fileTime	T_FILETIME64 [▶ 344]	以 FILETIME 格式表示的要转换的时间
nBias	INT	表示世界时钟（UTC）与本地时间之间的当前时间偏移，以分钟为单位。以下规则适用：UTC = 本地时间 + 时间偏移
bUTC	BOOL	表示在输入中指定的时间是 UTC 时间还是本地时间。
nPrecision	USINT(0..9)	指定秒显示的精度为小数位数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.46.0

4.1.14 FILETIME64_TO_SYSTEMTIME



函数“FILETIME64_TO_SYSTEMTIME”将 FILETIME 格式的时间转换为“可读的”SYSTEMTIME 格式。如果设置了 64 位 fileTime 变量的最高有效位，则转换失败。在这种情况下，TIMESTRUCT 成员变量的值为零。

函数 FILETIME64_TO_SYSTEMTIME：TIMESTRUCT [▶ 347]

输入

```
VAR_INPUT
    fileTime : T_FILETIME64;
END_VAR
```

名称	类型	描述
fileTime	T_FILETIME64 [▶ 344]	以 FILETIME 格式表示的要转换的时间

示例：

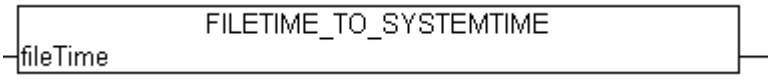
```
PROGRAM MAIN
VAR
    timeAsFileTime   : T_FILETIME64;
    timeAsSystemTime : TIMESTRUCT;
END_VAR
```

```
timeAsFileTime := F_GetSystemTime();
timeAsSystemTime := FILETIME64_TO_SYSTEMTIME( timeAsFileTime );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

4.1.15 FILETIME64_TO_TOD



该函数可以用于从 FILETIME 格式的时间中提取“当日时间”。

函数 FILETIME64_TO_TOD：TOD

输入

```
VAR_INPUT
    fileTime : T_FILETIME64;
END_VAR
```

名称	类型	描述
fileTime	T_FILETIME64 [▶ 344]	以 FILETIME 格式表示的要转换的时间

返回参数	描述
0	错误。如果设 64 位 fileTime 变量的最高有效位被置位，则函数执行。
> 0	无错误。显示当日时间。

示例：

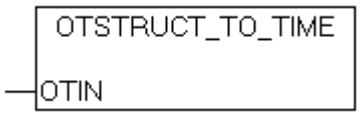
```
PROGRAM MAIN
VAR
    timeAsFileTime : T_FILETIME64;
    timeOfDay      : TOD;
END_VAR
```

```
timeAsFileTime := F_GetSystemTime();
timeOfDay := FILETIME64_TO_TOD( timeAsFileTime );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.0	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.7.3.0

4.1.16 OTSTRUCT_TO_TIME



函数“OTSTRUCT_TO_TIME”可以用于将具有解析的毫秒、秒钟、分钟、小时、日和周的结构转换为 TIME 变量。

函数 OTSTRUCT_TO_TIME：TIME

输入

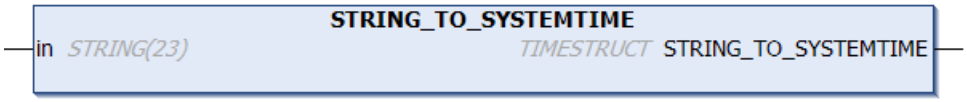
```
VAR_INPUT
    OTIN : OTSTRUCT;
END_VAR
```

名称	类型	描述
OTIN	OTSTRUCT [▶ 331]	要转换的结构

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.17 STRING_TO_SYSTEMTIME



该函数将字符串转换为 Windows SYSTEMTIME 格式。

函数 STRING_TO_SYSTEMTIME：TIMESTRUCT [▶ 347]

输入

```
VAR_INPUT
    in : STRING(23);
END_VAR
```

名称	类型	描述
in	STRING(23)	要转换的字符串。字符串必须具有以下格式： **'YYYY-MM-DD-hh:mm:ss.xxx'**

- YYYY：年份（1601..9999）
- MM：月份（01..12）
- DD：日（01..31）
- hh：小时（00..23）
- mm：分钟（00..59）
- ss：秒钟（00..59）
- xxx：毫秒（000..999）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.18 SYSTEMTIME_TO_DT

“SYSTEMTIME_TO_DT”函数允许将 Windows 系统时间结构转换为 PLC 中常见的 DATE_AND_TIME 格式（DT）。系统时间的分辨率为 1 ms，DATE_AND_TIME 的分辨率为 1 s。在转换期间会考虑系统时间中的毫秒，并将毫秒向上取整为相应的 DATE_AND_TIME 返回值。要禁用四舍五入，请将 Windows 系统时间结构中的 wMilliseconds 元素设置为零。

函数 SYSTEMTIME_TO_DT：DT

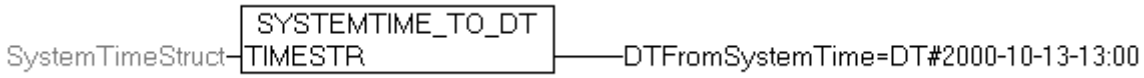
输入

```
VAR_INPUT
    TIMESTR : TIMESTRUCT;
END_VAR
```

名称	类型	描述
TIMESTR	TIMESTRUCT [▶ 347]	具有要转换的 Windows 系统时间的结构

示例：

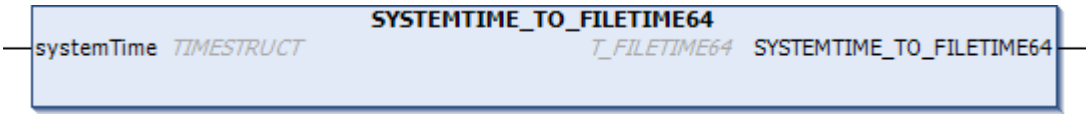
```
PROGRAM SystemTimeTest
VAR
    SystemTimeStruct  : TIMESTRUCT;
    DTFromSystemTime  : DT;
END_VAR
```



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.19 SYSTEMTIME_TO_FILETIME64



该函数可以用于将 Windows 系统时间结构转换为 FILETIME 格式。SystemTime 变量的星期几 wDayOfWeek 将被忽略。系统时间年份必须大于 1601 且 小于 30827。

函数 SYSTEMTIME_TO_FILETIME64：T_FILETIME64 [▶ 344]

输入

```
VAR_INPUT
    systemTime : TIMESTRUCT;
END_VAR
```

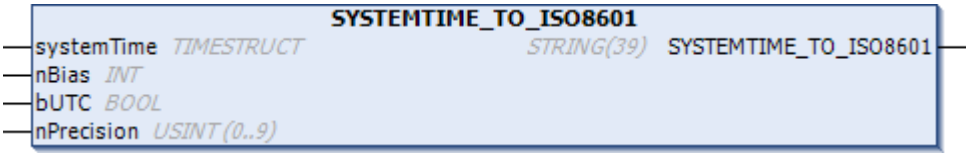
名称	类型	描述
systemTime	TIMESTRUCT [▶ 347]	具有要转换的 Windows 系统时间的结构

返回参数	描述
0	错误，错误的 SystemTime 参数值
> 0	无错误，文件时间

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

4.1.20 SYSTEMTIME_TO_ISO8601



该函数将 Windows 系统时间结构转换为 ISO 8601 格式的字符串。

结果与以下方案相对应：**YYYY-MM-DDThh:mm:ss.xxxTZD**

函数 SYSTEMTIME_TO_ISO8601：STRING(39)

输入

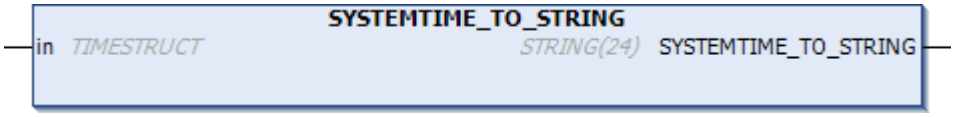
```
VAR_INPUT
    systemTime   : TIMESTRUCT;  (* Input time in system time format (struct) *)
    nBias        : INT;         (* Specifies the current bias, in minutes, for local time translation
 on this computer.
                                 The bias is the difference between Coordinated Universal Time (UTC)
and local time.
                                 UTC = local time + bias *)
    bUTC         : BOOL;        (* Specifies whether the systemTime is UTC or local time. *)
    nPrecision   : USINT(0..9); (* Precision. Number of decimal places of seconds. (0..9) *)
END_VAR
```

名称	类型	描述
systemTime	TIMESTRUCT [▶ 347]	具有要转换的 Windows 系统时间的结构
nBias	INT	表示世界时钟（UTC）与本地时间之间的当前时间偏移，以分钟为单位。以下规则适用：UTC = 本地时间 + 时间偏移。
bUTC	BOOL	表示在输入中指定的时间是 UTC 时间还是本地时间。
nPrecision	USINT(0..9)	指定秒显示的精度为小数位数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.46.0

4.1.21 SYSTEMTIME_TO_STRING



该函数将 Windows 系统时间结构转换为具有以下格式的字符串：**YYYY-MM-DD-hh:mm:ss.xxx**：

- YYYY：年份（1601..9999）
- MM：月份（01..12）
- DD：日（01..31）
- hh：小时（00..23）
- mm：分钟（00..59）
- ss：秒钟（00..59）
- xxx：毫秒（000..999）

函数 SYSTEMTIME_TO_STRING：STRING(24)

输入

```
VAR_INPUT
    in : TIMESTRUCT;
END_VAR
```

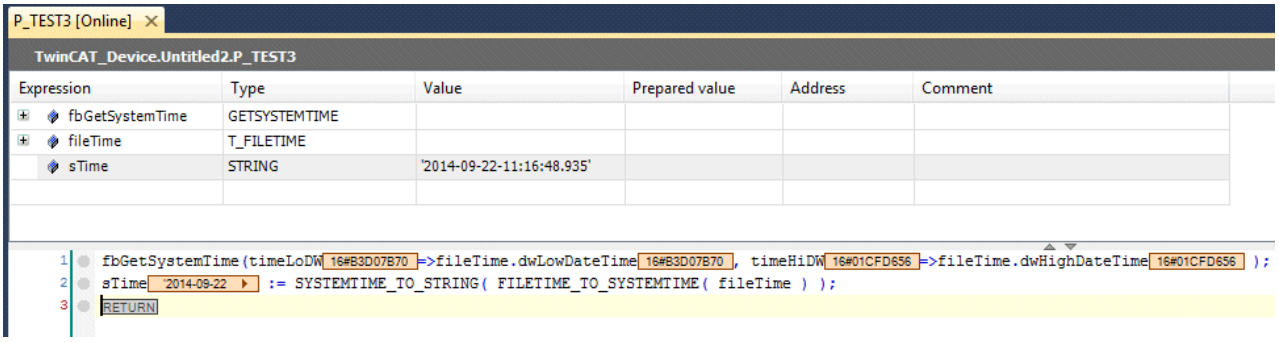
名称	类型	描述
in	TIMESTRUCT [▶ 347]	具有要转换的 Windows 系统时间的结构

示例：

```
PROGRAM MAIN
VAR
    fbGetSystemTime  : GETSYSTEMTIME;
    fileTime         : T_FILETIME;
    sTime            : STRING;
END_VAR
```

```
fbGetSystemTime(timeLoDW=>fileTime.dwLowDateTime, timeHiDW=>fileTime.dwHighDateTime );
sTime := SYSTEMTIME_TO_STRING( FILETIME_TO_SYSTEMTIME( fileTime ) );
```

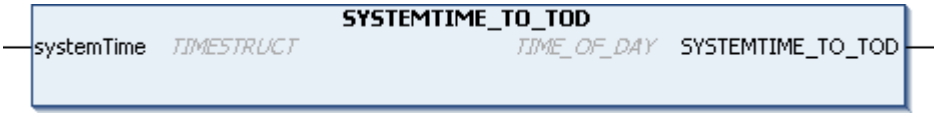
在线视图：



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.1.22 SYSTEMTIME_TO_TOD



该函数可以用于从 Windows 系统时间结构中提取“当日时间”。

函数 SYSTEMTIME_TO_TOD：TOD

输入

```
VAR_INPUT
    systemTime : TIMESTRUCT;
END_VAR
```

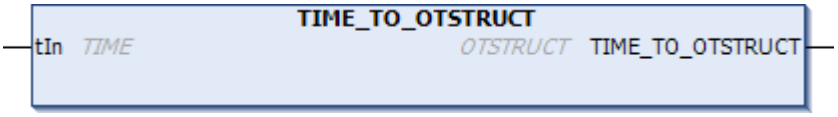
名称	类型	描述
systemTime	TIMESTRUCT [▶ 347]	具有要转换的 Windows 系统时间的结构

返回参数	描述
0	错误，错误的 SystemTime 参数值。
> 0	无错误，显示当日时间。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.0	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.7.3.0

4.1.23 TIME_TO_OTSTRUCT



函数“TIME_TO_OTSTRUCT”可以用于将 TIME 常量或变量转换为具有解析的毫秒、秒钟、分钟、小时、日和周的结构。

函数 TIME_TO_OTSTRUCT：OTSTRUCT [▶ 331]

输入

```
VAR_INPUT
    TIN : TIME;
END_VAR
```

名称	类型	描述
TIN	TIME	要转换的 TIME 变量

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.2 扩展 STRING 函数

4.2.1 CHAR_TO_WCHAR

该函数将 STRING 数据类型的变量转换为 WSTRING 数据类型的变量（以空字符结尾）。

返回值

名称	类型	描述
CHAR_TO_WCHAR	WSTRING(1)	将 STRING 数据类型的变量转换为 WSTRING 数据类型的变量。

输入

```
VAR_INPUT
    sTextIn : STRING(1);
END_VAR
```

名称	类型	描述
sTextIn	STRING(1)	要转换的 STRING 变量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.2 CONCAT2

该函数可连接 2 个任意长度的 STRING 数据类型的字符串，并检查结果字符串是否比指定的输出字符串长。在这种情况下，字符串会被截断。

函数返回：

- TRUE，如果连接成功的话。

- 。FALSE，如果结果字符串比输出字符串长且不适合给定的输出缓冲区的话 此时结果字符串的内存需求将大于输出字符串的内存需求。然后，字符串被截断。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
CONCAT2	BOOL	连接 2 个任意长度的 STRING 数据类型的字符串，并检查结果字符串是否比指定的输出字符串长。

输入

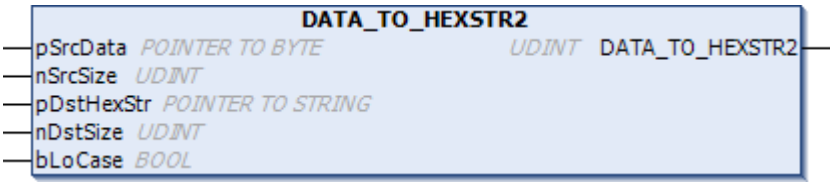
```
VAR_INPUT
    pSrcString1 : POINTER TO STRING;
    pSrcString2 : POINTER TO STRING;
    pDstString  : POINTER TO STRING;
    nDstSize    : UDINT;
END_VAR
```

名称	类型	描述
pSrcString1	POINTER TO STRING	指向要连接的第一个 STRING 变量的指针（输入字符串）
pSrcString2	POINTER TO STRING	指向要连接的第二个 STRING 变量的指针（输入字符串）
pDstString	POINTER TO STRING	指向连接后的结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.3 DATA_TO_HEXSTR2



该函数将二进制数据转换为十六进制字符串。该函数可以用于转换简单的数据类型和结构变量。如果超过输出的最大可能长度，则会在结果字符串中添加一个点字符（“.”），并中止转换。剩余的数据字节不会进行转换。

返回值

名称	类型	描述
DATA_TO_HEXSTR2	UDINT	将二进制数据转换为十六进制字符串。

输入

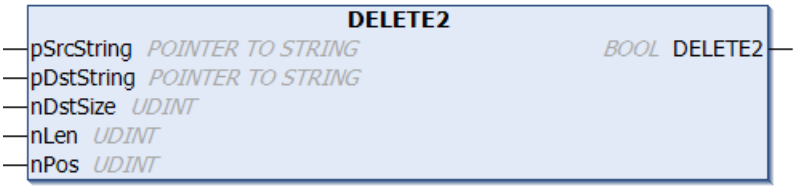
```
VAR_INPUT
    pSrcData    : POINTER TO BYTE;  // pointer to data buffer
    nSrcSize    : UDINT;            // size of data buffer in bytes (= number of bytes to be convert
ed)
    pDstHexStr  : POINTER TO STRING;// pointer to destination buffer
    nDstSize    : UDINT;            // size of destination buffer in bytes
    bLoCase     : BOOL;             // default: use "ABCDEF", if TRUE use "abcdef" characters
END_VAR
```

名称	类型	描述
pSrcData	POINTER TO BYTE	要转换的二进制数据的起始地址（指针）。使用 ADR 运算符可以确定地址。
nSrcSize	UDINT	要转换的二进制数据的最大大小（以字节为单位）。使用 SIZEOF 运算符可以确定大小。
pDstHexStr	POINTER TO STRING	要写入转换后的十六进制字符串的目标缓冲区的起始地址（指针）。使用 ADR 运算符可以确定地址。
nDstSize	UDINT	目标缓冲区的最大可用大小（以字节为单位）。使用 SIZEOF 运算符可以确定大小。
bLoCase	BOOL	该参数指定在转换时使用大写还是小写。TRUE = 小写字母，FALSE = 大写字母。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.0	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.5.1.0

4.2.4 DELETE2



该函数从 nPos 位置开始，移除字符串中的 nLen 字符。

函数返回：

- TRUE，如果成功移除字符的话。
- 。FALSE，如果结果字符串比输出字符串长且不适合给定的输出缓冲区的话 此时结果字符串的内存需求将大于输出字符串的内存需求。然后，字符串被截断。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
DELETE2	BOOL	从 nPos 位置开始，删除字符串中的 nLen 字符。

输入

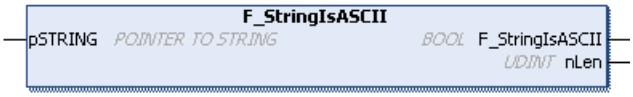
```
VAR_INPUT
    pSrcString : POINTER TO STRING;
    pDstString : POINTER TO STRING;
    nDstSize   : UDINT;
    nLen       : UDINT;
    nPos       : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向 STRING 变量的指针（输入字符串）
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。
nLen	UDINT	要移除的字符数。
nPos	UDINT	要移除的第一个字符的位置，包括以下字符（nPos = 1 = 第一个字符）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.5 F_StringIsASCII



该函数检查字符串是否只包含 ASCII 字符（0x000 至 0x7F），并返回 ASCII 字符数。如果字符串只包含 ASCII 字符，则该字符串直接兼容 UTF-8。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
F_StringIsASCII	BOOL	如果字符串只包含 ASCII 字符，则返回值为 TRUE。

输入

```
VAR_INPUT
    pSTRING : POINTER TO STRING;
END_VAR
```

名称	类型	描述
pString	POINTER TO STRING	指向 STRING 变量的指针

输出

```
VAR_OUTPUT
    nLen : UDINT;
END_VAR
```

名称	类型	描述
nLen	UDINT	字符串中的 ASCII 字符数

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.6 FIND2

该函数在另一个字符串中查找一个可能出现多次的字符串。

函数返回：

- 找到的第一个字符串的第一个字符的位置。
- 如果未找到字符串，则值为 0。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FIND2	UDINT	在另一个字符串中查找一个可能出现多次的字符串。

输入

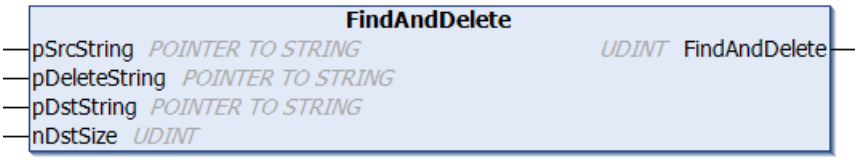
```
VAR_INPUT
    pSrcString  : POINTER TO STRING;
    pFindString : POINTER TO STRING;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针。
pFindString	POINTER TO STRING	指向正在搜索其字符串的 STRING 变量的指针。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.7 FindAndDelete



该函数在另一个字符串中查找一个可能出现多次的字符串，并将其移除。

函数返回：

- 已移除的字符串数。
- 如果未找到字符串，则值为 0。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FindAndDelete	UDINT	在另一个字符串中查找一个可能出现多次的字符串，并将其删除。

输入

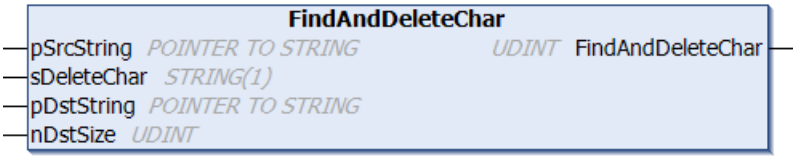
```
VAR_INPUT
    pSrcString    : POINTER TO STRING;
    pDeleteString : POINTER TO STRING;
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针（输入字符串）。
pDeleteString	POINTER TO STRING	指向要搜索并移除其字符串的 STRING 变量的指针（输入字符串）。
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.8 FindAndDeleteChar



该函数在一个字符串中查找一个可能出现多次的字符，并将其移除。

函数返回：

- 已移除的字符数。
- 如果未找到字符，则值为 0。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FindAndDeleteChar	UDINT	在字符串中查找一个可能出现多次的字符，并将其删除

输入

```
VAR_INPUT
    pSrcString    : POINTER TO STRING;
    sDeleteChar   : STRING(1);
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针（输入字符串）。
sDeleteChar	STRING(1)	要移除的字符。
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.9 FindAndReplace

该函数在另一个字符串中查找一个可能出现多次的字符串，并使用另一个字符串替换它。

函数返回：

- 已替换的字符串数。
- 如果未找到字符串，则值为 0。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FindAndReplace	UDINT	在另一个字符串中查找一个可能出现多次的字符串，并使用另一个字符串替换它。

输入

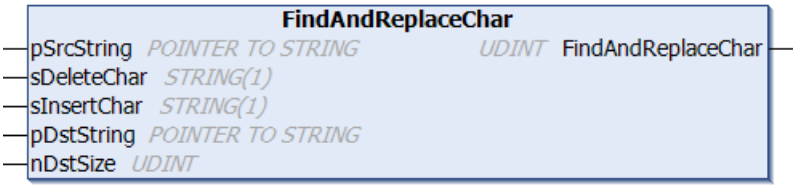
```
VAR_INPUT
    pSrcString    : POINTER TO STRING;
    pDeleteString : POINTER TO STRING;
    pInsertString : POINTER TO STRING;
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针（输入字符串）。
pDeleteString	POINTER TO STRING	指向要替换其字符串的 STRING 变量的指针（输入字符串）。
pInsertString	POINTER TO STRING	指向其字符串要替换其他字符串的 STRING 变量的指针（输入字符串）。
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位；SIZEOF() 运算符可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.10 FindAndReplaceChar



该函数在一个字符串中查找一个可能出现多次的字符，并使用另一个字符替换它。

函数返回

- 已替换的字符数。
- 如果未找到该字符，则值为 0。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FindAndReplaceChar	UDINT	在一个字符串中查找一个可能出现多次的字符，并使用另一个字符替换它。

输入

```
VAR_INPUT
    pSrcString    : POINTER TO STRING;
    sDeleteChar   : STRING(1);
    sInsertChar   : STRING(1);
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针（输入字符串）。
sDeleteChar	STRING(1)	要移除的字符。
sInsertChar	STRING(1)	要替换其他字符的字符。
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位；SIZEOF() 运算符可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.11 FindAndSplit

该函数将一个字符串拆分成 2 个字符串。

从左到右搜索字符串中的分隔符（例如“\”）。输出第一次出现的分隔符两侧的字符串。

使用参数 bSearchFromRight 可以改变搜索方向，以便从右到左搜索字符串。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FindAndSplit	BOOL	如果找到分隔符并成功拆分和输出字符串，则返回值为 TRUE。

输入

```
VAR_INPUT
    pSeparator        : POINTER TO STRING;
    pSrcString        : POINTER TO STRING;
    pLeftString       : POINTER TO STRING;
    nLeftSize         : UDINT;
    pRightString      : POINTER TO STRING;
```

```
    nRightSize        : UDINT;
    bSearchFromRight  : BOOL;
END_VAR
```

名称	类型	描述
pSeparator	POINTER TO STRING	指向代表分隔符的 STRING 变量的指针。
pSrcString	POINTER TO STRING	指向代表源字符串的 STRING 变量的指针。
pLeftString	POINTER TO STRING	指向要将分隔的左侧字符串输出到其中的 STRING 变量的指针。
nLeftSize	UDINT	分隔的左侧字符串的最大大小。
pRightString	POINTER TO STRING	指向要将分隔的右侧字符串输出到其中的 STRING 变量的指针。
nRightSize	UDINT	分隔的右侧字符串的最大大小
bSearchFromRight	BOOL	如果设置了输入，则会改变搜索方向，从右到左搜索字符串中的分隔符。

示例“拆分成多个子字符串”：

本示例显示了如何将字符串“machines/machine1/module2/data/tx”拆分成多个字符串 [“machines”、“machine1”、“module2”、“data”、“tx”]。为此，在循环中反复调用函数 FindAndSplit()。

```
PROGRAM MAIN
VAR
    sSrc         : STRING(255) := 'machines/machine1/module2/data/tx';
    sSeparator   : STRING(1) := '/';
    aSplit       : ARRAY[1..cMax] OF STRING(255);
    bResultSplit : BOOL;
    i            : UDINT;
END_VAR
VAR CONSTANT
    cMax         : UDINT := 9;
END_VAR
```

```
aSplit[1] := sSrc;
FOR i:=1 TO cMax-1 DO
    bResultSplit := FindAndSplit( pSeparator     := ADR(sSeparator),  pSrcString := ADR(aSplit[i]),
                                  pLeftString    := ADR(aSplit[i]),   nLeftSize  := SIZEOF(aSplit[i]
),
                                  pRightString   := ADR(aSplit[i+1]), nRightSize := SIZEOF(aSplit[i+
1]),
                                  bSearchFromRight := FALSE );
    IF NOT bResultSplit THEN
        EXIT;
    END_IF
END_FOR
```

示例“合并多个字符串”：

本示例显示了如何将多个字符串 [“machines”、“machine1”、“module2”、“data”、“tx”] 组合成一个字符串“machines/machine1/module2/data/tx”。

```
PROGRAM MAIN
VAR
    sSeparator    : STRING(1) := '/';
    aSplit        : ARRAY[1..cMax] OF STRING(255) := ['machines', 'machine1', 'module2', 'data',
'tx'];
    sJoined       : STRING(255);
    bResultConcat : BOOL;
    i             : UDINT;
END_VAR
VAR CONSTANT
    cMax          : UDINT := 5;
END_VAR
```

```
sJoined := aSplit[1];
FOR i:=1 TO cMax-1 DO
    bResultConcat := CONCAT2(pSrcString1 := ADR(sJoined), pSrcString2 := ADR(sSeparator), pDstString
 := ADR(sJoined), nDstSize := SIZEOF(sJoined));
    IF NOT bResultConcat THEN
        EXIT;
    END_IF
    bResultConcat := CONCAT2(pSrcString1 := ADR(sJoined), pSrcString2 := ADR(aSplit[i+1]), pDstStrin
g := ADR(sJoined), nDstSize := SIZEOF(sJoined));
    IF NOT bResultConcat THEN
        EXIT;
    END_IF
END_FOR
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.11	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.39.0

4.2.12 FindAndSplitChar

该函数将一个字符串拆分成 2 个字符串。

从左到右搜索字符串中的分隔符（例如“\”）。输出第一次出现的分隔符两侧的字符串。

使用参数 bSearchFromRight 可以改变搜索方向，以便从右到左搜索字符串。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
FindAndSplitChar	BOOL	如果找到分隔符并成功拆分和输出字符串，则返回值为 TRUE。

输入

```
VAR_INPUT
    sSeparatorChar    : STRING(1);
    pSrcString        : POINTER TO STRING;
    pLeftString       : POINTER TO STRING;
    nLeftSize         : UDINT;
    pRightString      : POINTER TO STRING;
    nRightSize        : UDINT;
    bSearchFromRight  : BOOL;
END_VAR
```

名称	类型	描述
sSeparatorChar	STRING(1)	代表分隔符的字符。
pSrcString	POINTER TO STRING	指向代表源字符串的 STRING 变量的指针。
pLeftString	POINTER TO STRING	指向要将分隔的左侧字符串输出到其中的 STRING 变量的指针。
nLeftSize	UDINT	分隔的左侧字符串的最大大小
pRightString	POINTER TO STRING	指向要将分隔的右侧字符串输出到其中的 STRING 变量的指针。
nRightSize	UDINT	分隔的右侧字符串的最大大小
bSearchFromRight	BOOL	如果设置了输入，则会改变搜索方向，从右到左搜索字符串中的分隔符。

示例“拆分成多个子字符串”：

本示例显示了如何将字符串“machines/machine1/module2/data/tx”拆分成多个字符串 [“machines”、“machine1”、“module2”、“data”、“tx”]。为此，在循环中反复调用函数 FindAndSplitChar()。

```
PROGRAM MAIN
VAR
    sSrc         : STRING(255) := 'machines/machine1/module2/data/tx';
    aSplit       : ARRAY[1..cMax] OF STRING(255);
    bResultSplit : BOOL;
    i            : UDINT;
END_VAR
```

```
VAR CONSTANT
    cMax         : UDINT := 9;
END_VAR
```

```
aSplit[1] := sSrc;
FOR i:=1 TO cMax-1 DO
    bResultSplit := FindAndSplitChar( sSeparatorChar := '/',  pSrcString := ADR(aSplit[i]),
                                      pLeftString    := ADR(aSplit[i]),   nLeftSize  := SIZEOF(aSpli
t[i]),
                                      pRightString   := ADR(aSplit[i+1]), nRightSize := SIZEOF(aSpli
t[i+1]),
                                      bSearchFromRight := FALSE );
    IF NOT bResultSplit THEN
        EXIT;
    END_IF
END_FOR
```

示例“合并多个字符串”：

本示例显示了如何将多个字符串 [“machines”、“machine1”、“module2”、“data”、“tx”] 组合成一个字符串“machines/machine1/module2/data/tx”。

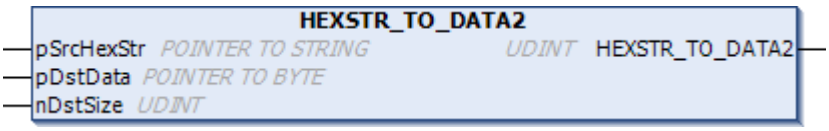
```
PROGRAM MAIN
VAR
    sSeparator    : STRING(1) := '/';
    aSplit        : ARRAY[1..cMax] OF STRING(255) := ['machines', 'machine1', 'module2', 'data',
'tx'];
    sJoined       : STRING(255);
    bResultConcat : BOOL;
    i             : UDINT;
END_VAR
VAR CONSTANT
    cMax          : UDINT := 5;
END_VAR
```

```
sJoined := aSplit[1];
FOR i:=1 TO cMax-1 DO
    bResultConcat := CONCAT2(pSrcString1 := ADR(sJoined), pSrcString2 := ADR(sSeparator), pDstString
 := ADR(sJoined), nDstSize := SIZEOF(sJoined));
    IF NOT bResultConcat THEN
        EXIT;
    END_IF
    bResultConcat := CONCAT2(pSrcString1 := ADR(sJoined), pSrcString2 := ADR(aSplit[i+1]), pDstStrin
g := ADR(sJoined), nDstSize := SIZEOF(sJoined));
    IF NOT bResultConcat THEN
        EXIT;
    END_IF
END_FOR
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.11	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.39.0

4.2.13 HEXSTR_TO_DATA2



该函数将十六进制字符串转换为二进制数据，并返回成功转换的数据字节数作为结果。在要转换的十六进制字符串中，仅限空格可以用作分隔符。允许使用小写字母和大写字母作为十六进制字符。如果出现错误或非法字符，则会中止转换，并返回零长度作为结果。

返回值

名称	类型	描述
HEXSTR_TO_DATA2	UDINT	将十六进制字符串转换为二进制数据。

输入

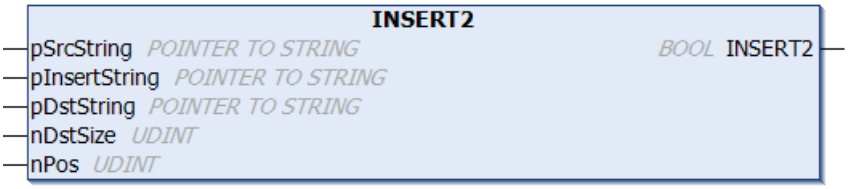
```
VAR_INPUT
    pSrcHexStr   : POINTER TO STRING;  // hex string to convert (Example: "AF 34 55 EC")
    pDstData     : POINTER TO BYTE;    // pointer to destination buffer
    nDstSize     : UDINT;              // size of destination buffer in bytes
END_VAR
```

名称	类型	描述
pSrcHexStr	POINTER TO STRING	要转换的十六进制字符串的起始地址（指针）（例如：“AB CD 01 23”）。使用 ADR 运算符可以确定地址。
pDstData	POINTER TO BYTE	要写入转换后的数据字节的目标缓冲区的起始地址（指针）。使用 ADR 运算符可以确定地址。
nDstSize	UDINT	目标缓冲区的最大可用大小（以字节为单位）。使用 SIZEOF 运算符可以确定大小。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.0	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.5.1.0

4.2.14 INSERT2



该函数在 nPos 位置后，将一个字符串插入另一个字符串。如果 nPos = 0，则会将字符串插入另一个字符串的第一个字符前。

函数返回

- TRUE，如果成功插入字符串的话。
- 。FALSE，如果结果字符串比输出字符串长且不适合给定的输出缓冲区的话 此时结果字符串的内存需求将大于输出字符串的内存需求。然后，字符串被截断。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
INSERT2	BOOL	在 nPos 位置后，将一个字符串插入另一个字符串。

输入

```
VAR_INPUT
    pSrcString    : POINTER TO STRING;
    pInsertString : POINTER TO STRING;
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
    nPos          : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针（输入字符串）。
pInsertString	POINTER TO STRING	指向要将其字符串插入其他字符串的 STRING 变量的指针（输入字符串）。
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位；SIZEOF() 运算符可以用于赋值。
nPos	UDINT	要在其后插入字符串的字符的位置。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.15 LEN2

该函数返回字符串中的字符数（STRING 的长度）。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
LEN2	UDINT	返回字符串中的字符数（STRING 的长度）。

输入

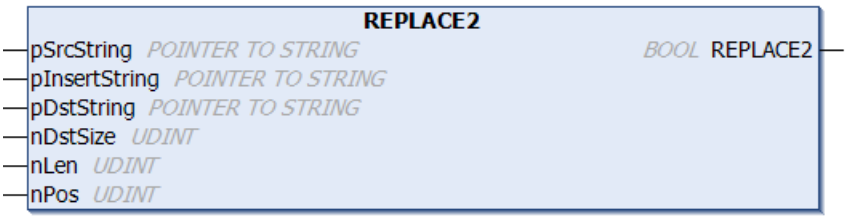
```
VAR_INPUT
    pSTRING : POINTER TO STRING;
END_VAR
```

名称	类型	描述
pSTRING	POINTER TO STRING	指向 STRING 变量的指针

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.16 REPLACE2



该函数从 nPos 位置开始，使用另一个字符串替换一个字符串的 nLen 字符。

函数返回

- TRUE，如果成功替换字符的话。
- 。FALSE，如果结果字符串比输出字符串长且不适合给定的输出缓冲区的话 此时结果字符串的内存需求将大于输出字符串的内存需求。然后，字符串被截断。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
REPLACE2	BOOL	从 nPos 位置开始，使用另一个字符串替换一个字符串的 nLen 字符。

输入

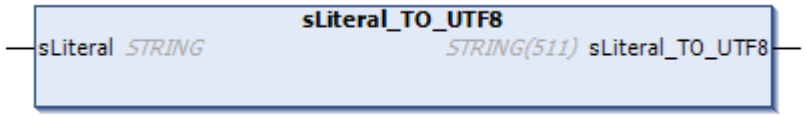
```
VAR_INPUT
    pSrcString    : POINTER TO STRING;
    pInsertString : POINTER TO STRING;
    pDstString    : POINTER TO STRING;
    nDstSize      : UDINT;
    nLen          : UDINT;
    nPos          : UDINT;
END_VAR
```

名称	类型	描述
pSrcString	POINTER TO STRING	指向要搜索其字符串的 STRING 变量的指针（输入字符串）。
pInsertString	POINTER TO STRING	指向其字符串要替换字符的 STRING 变量的指针（输入字符串）。
pDstString	POINTER TO STRING	指向结果 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。
nLen	UDINT	要替换的字符数。
nPos	UDINT	要移除的字符的位置，包括以下字符（nPos = 1 = 第一个字符）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >=3.3.35.0

4.2.17 sLiteral_TO_UTF8



该函数将 STRING 数据类型的任何字符串转换为 UTF-8 格式的字符串。该函数特别适用于分配字面量。

在将字面量分配给 UTF-8 STRING 时，规则如下：

- 对于仅使用 ASCII 字符集的字面量，可以直接进行分配。
- 对于使用 STRING 字符集的字面量，可以通过 sLiteral_TO_UTF8() 进行分配。
- 对于使用 WSTRING 字符集的字面量，可以通过 wsLiteral_TO_UTF8() [▶ 184] 进行分配。

如果字面量比可能的输出字符串长，则会返回一个空字符串。

返回值

名称	类型	描述
sLiteral_TO_UTF8	STRING(511)	将 STRING 数据类型的任何字符串转换为 UTF-8 格式的字符串。

输入

```
VAR_IN_OUT CONSTANT
    sLiteral    : STRING;
END_VAR
```

名称	类型	描述
sLiteral	STRING	要转换的 STRING。

示例

```
{attribute 'TcEncoding' := 'UTF-8'}
sMyText : STRING := sLiteral_TO_UTF8('Hühner legen Eier.');
```

```
{attribute 'TcEncoding' := 'UTF-8'}
sMyText1 : STRING := sLiteral_TO_UTF8('The dinner costs 30 €.');
```

属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统） >=3.3.34.0

4.2.18 STRING_TO_UTF8

该函数将 STRING 数据类型的变量的任何字符串转换为 UTF-8 格式的字符串。

函数返回

- TRUE，如果可以转换的话。
- FALSE，如果给定的字符集而无法转换的话。

如果输入字符串比输出字符串长，则会截断字符串。输入字符串过长，无法编码到输出字符串。在转换为 UTF-8 时，输出字符串的内存需求可能高于输入字符串的内存需求。

该函数可停止转换为 `Tc2_Utilities.Parameterlist.cMaxCharacters`。
通过适当的参数设置，可以避免出现无限循环。

返回值

名称	类型	描述
STRING_TO_UTF8	BOOL	将 STRING 数据类型的变量的任何字符串转换为 UTF-8 格式的字符串。

输入

```
VAR_INPUT
    pDstUTF8   : PVOID;
    pSrcSTRING : POINTER TO STRING;
    nDstSize   : UDINT;
END_VAR
```

名称	类型	描述
pDstUTF8	PVOID	指向 UTF-8 格式的字符串的指针（输出字符串）
pSrcSTRING	POINTER TO STRING	指向要转换的 STRING 变量的指针（输入字符串）
nDstSize	UDINT	结果变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。

属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.19 STRING_TO_WSTRING2

该函数将 STRING 数据类型的变量转换为 WSTRING 数据类型的变量，并检查输入字符串是否比输出字符串长。在这种情况下，字符串会被截断。

函数返回

- TRUE，如果可以转换完整字符串的话。
- FALSE，如果输入字符串比输出字符串长且结果不适合给定的输出缓冲区的话。输出字符串的内存需求高于输入字符串的内存需求。然后，字符串被截断。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止转换，以避免出现无限循环。

返回值

名称	类型	描述
STRING_TO_WSTRING2	BOOL	将 STRING 数据类型的变量转换为 WSTRING 数据类型的变量。

输入

```
VAR_INPUT
    pDstWSTRING : POINTER TO WSTRING;
    pSrcSTRING  : POINTER TO STRING;
    nDstSize    : UDINT;
END_VAR
```

名称	类型	描述
pDstWSTRING	POINTER TO WSTRING	指向转换后的 WSTRING 变量的指针（输出字符串）
pSrcSTRING	POINTER TO STRING	指向要转换的 STRING 变量的指针（输入字符串）
nDstSize	UDINT	生成 WSTRING 变量的大小（输出字符串），以字节为单位。SIZEOF() 运算符可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.20 STRNCPY

该函数复制 STRING 数据类型的变量的字符串，并检查是否完全复制该字符串。

函数返回

- TRUE，如果可以复制完整字符串（源数组的内容）的话。
- FALSE，如果在复制时字符串被截断的话。如果输入字符串比输出字符串长，则仅会复制与输出字符串的长度相对应的字符数（包括空字符结尾）。

返回值

名称	类型	描述
STRNCPY	BOOL	复制 STRING 数据类型的变量的字符串，并检查是否完全复制该字符串。

输入

```
VAR_INPUT
    pDst     : POINTER TO STRING;
    pSrc     : POINTER TO STRING;
    nDstSize : UDINT;
END_VAR
```

名称	类型	描述
pDst	POINTER TO STRING	指向复制的 STRING 变量的指针（输入字符串）
pSrc	POINTER TO STRING	指向要复制的 STRING 变量的指针（输出字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位； SIZEOF() 运算符可以用于赋值。

输出

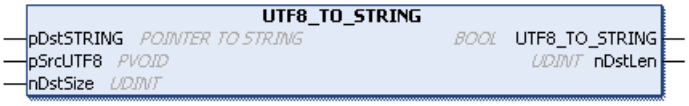
```
VAR_OUTPUT
    nSrcLen  : UDINT;
    nDstLen  : UDINT;
END_VAR
```

名称	类型	描述
nSrcLen	UDINT	要复制的指定 STRING 变量的长度
nDstLen	UDINT	复制的 WSTRING 变量的长度

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.21 UTF8_TO_STRING



该函数将 UTF-8 格式的字符串（PVOID 数据类型的指针变量）转换为 STRING 数据类型的字符串（变量）。

函数返回

- TRUE，如果可以转换的话。
- FALSE，如果给定的字符集而无法转换的话。

如果输入字符串比输出字符串长，则会截断字符串。可跳过未知字符。

该函数可停止转换为 `Tc2_Utilities.Parameterlist.cMaxCharacters`。通过适当的参数设置，可以避免出现无限循环。

返回值

名称	类型	描述
UTF8_TO_STRING	BOOL	如果转换成功，则返回值为 TRUE。

输入

```
VAR_INPUT
    pDstSTRING : POINTER TO STRING;
    pSrcUTF8   : PVOID;
    nDstSize   : UDINT;
END_VAR
```

名称	类型	描述
pDstSTRING	POINTER TO STRING	指向转换后的 STRING 变量的指针（输出字符串）
pSrc UTF8	PVOID	指针变量（输入字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位；SIZEOF() 运算符可以用于赋值。

输出

```
VAR_OUTPUT
    nDstLen        : UDINT;
END_VAR
```

名称	类型	描述
nDstLen	UDINT	输出字符串的实际长度（以字符数表示）

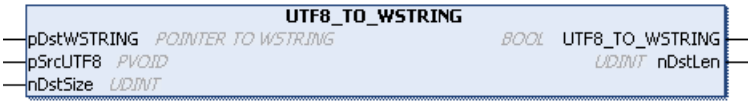
属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.22 UTF8_TO_WSTRING



该函数将 UTF-8 格式的字符串转换为 WSTRING 数据类型的字符串（变量）。

函数返回

- TRUE，如果可以转换的话。
- FALSE，如果给定的字符集而无法转换的话。

如果输入字符串比输出字符串长，则会截断字符串。输入字符串过长，无法编码到输出字符串。在转换为 UTF-8 时，输出字符串的内存需求可能高于输入字符串的内存需求。可跳过未知字符。

该函数可停止转换为
`Tc2_Utilities.Parameterlist.cMaxCharacters`。
通过适当的参数设置，可以避免出现无限循环。

返回值

名称	类型	描述
UTF8_TO_WSTRING	BOOL	如果转换成功，则返回值为 TRUE。

输入

```
VAR_INPUT
    pDstWSTRING : POINTER TO WSTRING;
    pSrcUTF8    : PVOID;
    nDstSize    : UDINT;
END_VAR
```

名称	类型	描述
pDstWSTRING	POINTER TO WSTRING	指向转换后的 WSTRING 变量的指针（输出字符串）
pSrc UTF8	PVOID	指针变量（输入字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位；SIZEOF() 运算符可以用于赋值。

输出

```
VAR_OUTPUT
    nDstLen : UDINT;
END_VAR
```

名称	类型	描述
nDstLen	UDINT	输出字符串的实际长度（以字符数表示）

属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.23 UTF8Len

该函数返回 UTF-8 字符串中的字符数。

如果字符串不符合 UTF-8 格式，则该函数将返回 0 值。

此外，该函数会检查所有字符是否为有效的 ASCII 字符，并通过 `bASCII` 输出将其输出。

函数在 `Tc2_Utilities.Parameterlist.cMaxCharacters` 之后停止检查。通过适当的参数设置，可以避免出现无限循环。

返回值

名称	类型	描述
UTF8Len	UDINT	返回值返回 UTF-8 字符串中的字符数。

输入

```
VAR_INPUT
    pUTF8 : PVOID;
END_VAR
```

名称	类型	描述
pUTF8	PVOID	指向以空字符结尾的 UTF-8 字符串的指针

输出

```
VAR_OUTPUT
    bASCII    : BOOL;
    nSize     : UDINT;
END_VAR
```

名称	类型	描述
bASCII	BOOL	TRUE，如果 UTF-8 字符是有效的 ASCII 字符的话。
nSize	UDINT	字符串的大小，以字节为单位（不以空字符结尾）。根据字符的不同，以字节为单位的大小可能大于字符串的长度。

属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.24 WCHAR_TO_CHAR

该函数将 WSTRING 数据类型的变量转换为 STRING 数据类型的变量（以空字符结尾）。只有当 WSTRING 字符与 STRING 字符相对应时，才能进行转换。否则，不会返回任何字符。

返回值

名称	类型	描述
WCHAR_TO_CHAR	STRING(1)	将 WSTRING 数据类型的变量转换为 STRING 数据类型的变量（以空字符结尾）。

输入

```
VAR_INPUT
    sTextIn : WSTRING(1);
END_VAR
```

名称	类型	描述
sTextIn	WSTRING(1)	要转换的 WSTRING 变量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.25 WCONCAT2

该函数可连接 2 个任意长度的 WSTRING 数据类型的字符串，并检查得到的字符串是否比指定的输出字符串长。在这种情况下，字符串会被截断。

函数返回

- TRUE，如果连接成功的话。
- 。FALSE，如果结果字符串比输出字符串长且不适合给定的输出缓冲区的话 此时结果字符串的内存需求将大于输出字符串的内存需求。然后，字符串被截断。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环

返回值

名称	类型	描述
WCONCAT2	BOOL	连接 2 个任意长度的 WSTRING 数据类型的字符串，并检查结果字符串是否比指定的输出字符串长。

输入

```
VAR_INPUT
    pSrcWString1 : POINTER TO WSTRING;
    pSrcWString2 : POINTER TO WSTRING;
    pDstWString  : POINTER TO WSTRING;
    nDstSize     : UDINT;
END_VAR
```

名称	类型	描述
pSrcWString1	POINTER TO WSTRING	指向要连接的第一个 WSTRING 变量的指针（输入字符串）
pSrcWString2	POINTER TO WSTRING	指向要连接的第二个 WSTRING 变量的指针（输入字符串）
pDstWString	POINTER TO WSTRING	指向连接后的结果 WSTRING 变量的指针（输出字符串）
nDstSize	UDINT	生成 WSTRING 变量的大小（输出字符串），以字节为单位。SIZEOF() 运算符可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.26 WLEN2

该函数返回 WSTRING 数据类型的 Unicode 字符串中的字符数（WSTRING 的长度）。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止检查输入长度，以避免出现无限循环。

返回值

名称	类型	描述
WLEN2	UDINT	返回 WSTRING 数据类型的 Unicode 字符串中的字符数（WSTRING 的长度）。

输入

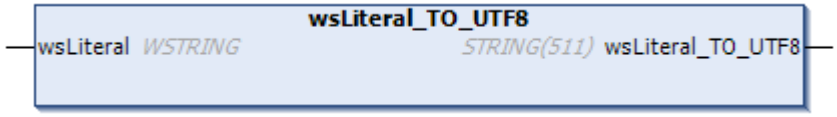
```
VAR_INPUT
    pWSTRING : POINTER TO WSTRING;
END_VAR
```

名称	类型	描述
pWSTRING	POINTER TO WSTRING	指向 WSTRING 变量的指针

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.27 wsLiteral_TO_UTF8



该函数将 WSTRING 数据类型的任何字符串转换为 UTF-8 格式的字符串。该函数特别适用于分配字面量。

在将字面量分配给 UTF-8 STRING 时，规则如下：

- 对于仅使用 ASCII 字符集的字面量，可以直接进行分配。
- 对于使用 STRING 字符集的字面量，可以通过 sLiteral_TO_UTF8() [▶ 176] 进行分配。
- 对于使用 WSTRING 字符集的字面量，可以通过 wsLiteral_TO_UTF8() 进行分配。

如果字面量比可能的输出字符串长，则会返回一个空字符串。

返回值

名称	类型	描述
wsLiteral_TO_UTF8	STRING(511)	将 WSTRING 数据类型的任何字符串转换为 UTF-8 格式的字符串。

输入

```
VAR_IN_OUT CONSTANT
    wsLiteral   : WSTRING;
END_VAR
```

名称	类型	描述
wsLiteral	WSTRING	要转换的 WSTRING。

示例

```
{attribute 'TcEncoding' := 'UTF-8'}
sMyText : STRING := wsLiteral_TO_UTF8("Hühner legen Eier.");
```

```
{attribute 'TcEncoding' := 'UTF-8'}
sMyText2 : STRING := wsLiteral_TO_UTF8("The dinner costs 30 €.");
```

属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统） >=3.3.34.0

4.2.28 WSTRING_TO_STRING2

该函数将 WSTRING 数据类型的变量转换为 STRING 数据类型的变量。

函数返回

- TRUE，如果可以转换完整字符串的话。
- FALSE，如果输入字符串比输出字符串长且结果不适合给定的输出缓冲区的话。

在转换期间可跳过无法转换的字符。

该函数在 `Parameterlist.cMaxCharacters` 个字符后会停止转换，以避免出现无限循环。

返回值

名称	类型	描述
WSTRING_TO_STRING2	BOOL	将 WSTRING 数据类型的变量转换为 STRING 数据类型的变量。

输入

```
VAR_INPUT
    pDstString  : POINTER TO STRING;
    pSrcWString : POINTER TO WSTRING;
    nDstSize    : UDINT;
END_VAR
```

名称	类型	描述
pDstString	POINTER TO WSTRING	指向转换后的 STRING 变量的指针（输出字符串）
pSrcWString	POINTER TO WSTRING	指向要转换的 WSTRING 变量的指针（输入字符串）
nDstSize	UDINT	结果 STRING 变量的大小（输出字符串），以字节为单位；SIZEOF() 运算符可以用于赋值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.29 WSTRING_TO_UTF8

该函数将 WSTRING 数据类型的变量的字符串转换为 UTF-8 格式的字符串。

函数返回

- TRUE，如果可以转换的话。
- FALSE，如果给定的字符集而无法转换的话。

如果输入字符串比输出字符串长，则会截断字符串。输入字符串过长，无法编码到输出字符串。（在转换为 UTF-8 时，输出字符串的内存需求可能高于输入字符串的内存需求）。可跳过未知字符。

该函数可停止转换为 `Tc2_Utilities.Parameterlist.cMaxCharacters`。通过适当的参数设置，可以避免出现无限循环。

返回值

名称	类型	描述
WSTRING_TO_UTF8	BOOL	将 WSTRING 数据类型的变量的字符串转换为 UTF-8 格式的字符串。

输入

```
VAR_INPUT
    pDstUTF8    : PVOID;
    pSrcWSTRING : POINTER TO WSTRING;
    nDstSize    : UDINT;
END_VAR
```

名称	类型	描述
pDstUTF8	PVOID	指针变量（输出字符串）
pSrcWSTRING	POINTER TO WSTRING	指向 WSTRING 变量的指针（输入字符串）
nDstSize	UDINT	结果变量（输出字符串）的大小，以字节为单位；SIZEOF() 运算符可以用于赋值。

属性 'TcEncoding' := 'UTF-8' 的文档

有关 UTF-8 格式的字符串的更多信息，请参见“TcEncoding”的属性文档。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.2.30 WSTRNCPY

该函数复制 WSTRING 数据类型的变量的字符串，并检查是否完全复制该字符串。

函数返回

- TRUE，如果可以复制完整字符串（源数组的内容）的话。
- FALSE，如果在复制时字符串被截断的话。如果输入字符串比输出字符串长，则仅会复制与输出字符串的长度相对应的字符数（包括空字符结尾）。

返回值

名称	类型	描述
WSTRNCPY	BOOL	复制 WSTRING 数据类型的变量的字符串，并检查是否完全复制该字符串。

输入

```
VAR_INPUT
    pDst     : POINTER TO WSTRING;
    pSrc     : POINTER TO WSTRING;
    nDstSize : UDINT;
END_VAR
```

名称	类型	描述
pDst	POINTER TO WSTRING	指向复制的 WSTRING 变量的指针（输入字符串）
pSrc	POINTER TO WSTRING	指向要复制的 WSTRING 变量的指针（输出字符串）

名称	类型	描述
nDstSize	UDINT	生成 WSTRING 变量的大小（输出字符串），以字节为单位。运算符 SIZEOF() 可以用于赋值。

输出

```
VAR_OUTPUT
    nSrcLen  : UDINT;
    nDstLen  : UDINT;
END_VAR
```

名称	类型	描述
nSrcLen	UDINT	要复制的指定 WSTRING 变量的长度
nDstLen	UDINT	复制的 WSTRING 变量的长度

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.3 字节顺序转换函数

4.3.1 主机字节顺序/网络字节顺序

字节顺序在网络协议中是固定的。这被称为网络字节顺序。TwinCAT 系统中的自然字节顺序被称为主机字节顺序。在大多数情况下，所需的网络字节顺序与大端序格式（MOTOROLA）相对应。不过，TwinCAT PLC 系统使用的是小端序格式（Intel）。为了在 TwinCAT PLC 系统和不同平台之间进行无差错的数据交换，必须对应用程序中的字节顺序进行相应的转换。

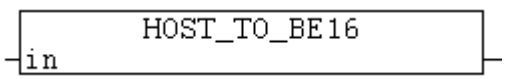
使用以下函数可以将通过网络协议从 TwinCAT 系统（主机）传输到外部系统的数据转换为网络格式：

- HOST_TO_BE16 [▶ 187]
- HOST_TO_BE32 [▶ 188]
- HOST_TO_BE64 [▶ 188]
- HOST_TO_BE64EX [▶ 189]
- HOST_TO_BE128 [▶ 189]

反之，使用以下函数可以将接收的网络数据（外部系统）转换为主机格式（TwinCAT 系统）：

- BE16_TO_HOST [▶ 190]
- BE32_TO_HOST [▶ 190]
- BE64_TO_HOST [▶ 191]
- BE64_TO_HOSTEX [▶ 191]
- BE128_TO_HOST [▶ 192]

4.3.2 HOST_TO_BE16



该函数执行 16 位数字的主机到网络转换。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
HOST_TO_BE16	WORD	

输入

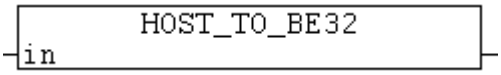
```
VAR_INPUT
    in : WORD;
END_VAR
```

名称	类型	描述
in	WORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.3 HOST_TO_BE32



该函数执行 32 位数字的主机到网络转换。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
HOST_TO_BE32	DWORD	

输入

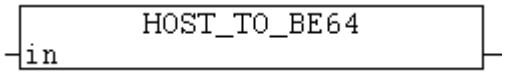
```
VAR_INPUT
    in : DWORD;
END_VAR
```

名称	类型	描述
in	DWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.4 HOST_TO_BE64



该函数执行 64 位数字的主机到网络转换（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
HOST_TO_BE64	T_ULARGE_INTEGER	

输入

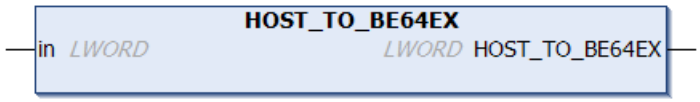
```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER [▶ 347]	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.5 HOST_TO_BE64EX



该函数执行 64 位数字的主机到网络转换（“本地”类型：LWORD）。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
HOST_TO_BE64EX	LWORD	

输入

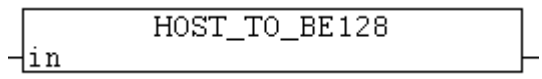
```
VAR_INPUT
    in : LWORD;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.6 HOST_TO_BE128



该函数执行 128 位数字的主机到网络转换（“传统”类型：T_UHUGE_INTEGER [▶ 347]）。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
HOST_TO_BE128	T_UHUGE_INTEGER	

输入

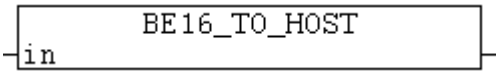
```
VAR_INPUT
    in : T_UHUGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_UHUGE_INTEGER [▶ 347]	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.7 BE16_TO_HOST



该函数执行 16 位数字的网络到主机转换。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
BE16_TO_HOST	WORD	

输入

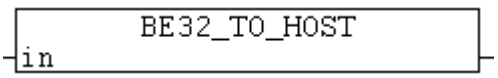
```
VAR_INPUT
    in : WORD;
END_VAR
```

名称	类型	描述
in	WORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.8 BE32_TO_HOST



该函数执行 32 位数字的网络到主机转换。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
BE32_TO_HOST	DWORD	

输入

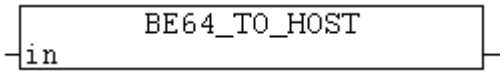
```
VAR_INPUT
    in : DWORD;
END_VAR
```

名称	类型	描述
in	DWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.9 BE64_TO_HOST



该函数执行 64 位数字的网络到主机转换（“传统”类型：T_ULARGE_INTEGER）。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
BE64_TO_HOST	T_ULARGE_INTEGER	

输入

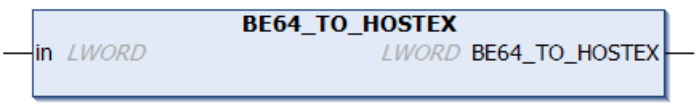
```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER [▶ 347]	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.10 BE64_TO_HOSTEX



该函数执行 64 位数字的网络到主机转换（“本地”类型：LWORD）。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
BE64_TO_HOSTEX	LWORD	

输入

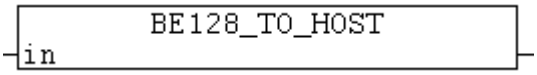
```
VAR_INPUT
    in : LWORD;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.3.11 BE128_TO_HOST



该函数执行 128 位数字的网络到主机转换（“传统”类型：T_UHUGE_INTEGER [▶ 347]）。另请参见：字节顺序 [▶ 187]。

返回值

名称	类型	描述
BE128_TO_HOST	T_UHUGE_INTEGER	

输入

```
VAR_INPUT
    in : T_UHUGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_UHUGE_INTEGER [▶ 347]	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4 FLOAT 函数

4.4.1 [已过时]

4.4.1.1 BOOL_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
BOOL_TO_FLOAT	LREAL	.

输入

```
VAR_INPUT
    in     : BOOL;
END_VAR
```

名称	类型	描述
in	BOOL	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.2 DINT_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
DINT_TO_FLOAT	FLOAT	

输入

```
VAR_INPUT
    in     : DINT;
END_VAR
```

名称	类型	描述
in	DINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.3 FLOAT_TO_BOOL

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_BOOL	BOOL	

输入

```
VAR_INPUT
    in     : LREAL;
END_VAR
```

名称	类型	描述
in	LREAL	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.4 FLOAT_TO_DINT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_DINT	DINT	

输入

```
VAR_INPUT
    in         : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.5 FLOAT_TO_INT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_INT	INT	

输入

```
VAR_INPUT
    in              : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.6 FLOAT_TO_SINT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_SINT	SINT	

输入

```
VAR_INPUT
    in              : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.7 FLOAT_TO_STRING

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_STRING	STRING	

输入

```
VAR_INPUT
    in              : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.8 FLOAT_TO_TIME

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_TIME	TIME	

输入

```
VAR_INPUT
    in              : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.9 FLOAT_TO_UDINT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_UDINT	UDINT	

输入

```
VAR_INPUT
    in              : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.10 FLOAT_TO_UINT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
FLOAT_TO_UINT	UINT	

输入

```
VAR_INPUT
    in          : FLOAT;
END_VAR
```

名称	类型	描述
in	FLOAT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.11 INT_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
INT_TO_FLOAT	FLOAT	

输入

```
VAR_INPUT
    in    : INT;
END_VAR
```

名称	类型	描述
in	INT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.12 SINT_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
SINT_TO_FLOAT	FLOAT	

输入

```
VAR_INPUT
    in     : SINT;
END_VAR
```

名称	类型	描述
in	SINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.13 TIME_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
TIME_TO_FLOAT	FLOAT	

输入

```
VAR_INPUT
    in:        TIME;
END_VAR
```

名称	类型	描述
in	TIME	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.14 UDINT_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
UDINT_TO_FLOAT	FLOAT	

输入

```
VAR_INPUT
    in     : UDINT;
END_VAR
```

名称	类型	描述
in	UDINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.1.15 UINT_TO_FLOAT

过时的函数

该函数已过时。请使用函数“..._TO_LREAL”或“LREAL_TO_...” 相反，它们可以产生相同的结果。

返回值

名称	类型	描述
UINT_TO_FLOAT	FLOAT	

输入

```
VAR_INPUT
    in     : UINT;
END_VAR
```

名称	类型	描述
in	UINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.4.2 LrealIsFinite

返回值

名称	类型	描述
LrealIsFinite	BOOL	

输入

```
VAR_INPUT
    x        : REFERENCE TO LREAL;
END_VAR
```

名称	类型	描述
x	REFERENCE TO LREAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4020	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统） >=3.3.16.0

4.4.3 LrealIsNaN

该函数可测试某个值是否为 NaN（非数字）。如果返回值为 TRUE，则值为 NaN。

返回值

名称	类型	描述
LrealIsNaN		

输入

```
FUNCTION LrealIsNaN : BOOL
VAR_INPUT
    x        : REFERENCE TO LREAL;
END_VAR
```

名称	类型	描述
x	REFERENCE TO LREAL	

以下要点列出了 NaN 值的主要特征：

- 所有使用 NaN 作为输入数据的算术运算都会返回 NaN 作为结果。
- 如果至少有一个操作数是 NaN，则所有关系运算符 =、!=、> < >= <= 始终返回值 False。
- 如果参数具有 NaN 值，则标准 C 函数 `isnan()` 或 `_isnan()` 或者 PLC 函数 LrealIsNaN() [▶ 200] 和 RealIsNaN [▶ 201]（Tc2_Utilities 库）会返回 True 值。
- 表达式 `isnan(a)` 相当于表达式 `!(a == a)` 或 `NOT(a = a)`。

在进一步计算中使用 NaN 值时，NaN 值会自我重现，这一事实具有优势，因为无效值不会被忽略。

⚠ 谨慎

软件故障

仅可在 PLC 库中使用 NaN 值，特别是在运动控制和驱动控制的功能中作为控制值使用时，但它们必须经过明确批准！否则，NaN 值可能会导致相关软件出现潜在的危险故障！

⚠ 谨慎

浮点异常

如果要在应用程序中使用和处理 NaN，则必须关闭 FP 异常。否则，与 NaN 进行比较可能会导致异常，从而造成运行时停止运行，可能还会造成机器损坏。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4020	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统） >=3.3.16.0

4.4.4 RealIsFinite

返回值

名称	类型	描述
RealIsFinite	BOOL	

输入

```
VAR_INPUT
    x        : REFERENCE TO REAL;
END_VAR
```

名称	类型	描述
x	REFERENCE TO REAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.32	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.50.0

4.4.5 RealIsNaN

该函数可测试某个值是否为 NaN（非数字）。如果返回值为 TRUE，则值为 NaN。

返回值

名称	类型	描述
RealIsNaN		

输入

```
FUNCTION RealIsNaN : BOOL
VAR_INPUT
    x        : REFERENCE TO REAL;
END_VAR
```

名称	类型	描述
x	REFERENCE TO REAL	

以下要点列出了 NaN 值的主要特征：

- 所有使用 NaN 作为输入数据的算术运算都会返回 NaN 作为结果。
- 如果至少有一个操作数是 NaN，则所有关系运算符 =、!=、> < >= <= 始终返回值 False。
- 如果参数具有 NaN 值，则标准 C 函数 `isnan()` 或 `_isnan()` 或者 PLC 函数 LrealIsNaN() [▶ 200] 和 RealIsNaN [▶ 201]（Tc2_Utilities 库）会返回 True 值。

- 表达式 `isnan(a)` 相当于表达式 `!(a == a)` 或 `NOT(a = a)`。

在进一步计算中使用 NaN 值时，NaN 值会自我重现，这一事实具有优势，因为无效值不会被忽略。

⚠ 谨慎

软件故障

仅可在 PLC 库中使用 NaN 值，特别是在运动控制和驱动控制的功能中作为控制值使用时，但它们必须经过明确批准！否则，NaN 值可能会导致相关软件出现潜在的危险故障！

⚠ 谨慎

浮点异常

如果要在应用程序中使用和处理 NaN，则必须关闭 FP 异常。否则，与 NaN 进行比较可能会导致异常，从而造成运行时停止运行，可能还会造成机器损坏。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.32	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.50.0

4.5 LCOMPLEX 函数

4.5.1 LcomplexIsNaN

如果 LCOMPLEX 类型的参数有一个未定义值（NaN），则该函数返回 TRUE。

返回值

名称	类型	描述
LcomplexIsNaN	BOOL	

输入

```
VAR_INPUT
    Z   :  REFERENCE TO LCOMPLEX;
END_VAR
```

名称	类型	描述
Z	REFERENCE TO LCOMPLEX	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4020	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统） >=3.3.16.0

4.5.2 LcomplexAbs

该函数返回传输的复数的绝对值。

返回值

名称	类型	描述
LcomplexAbs	LREAL	

输入

```
VAR_INPUT
    Z   :  LCOMPLEX;
END_VAR
```

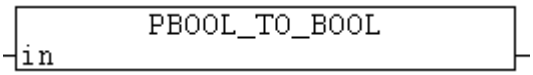
名称	类型	描述
Z	LCOMPLEX	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.21.0

4.6 P[TYPE]_TO_[TYPE] 转换函数

4.6.1 PBOOL_TO_BOOL



该函数返回 BOOL 指针变量的内容。

返回值

名称	类型	描述
PBOOL_TO_BOOL	BOOL	

输入

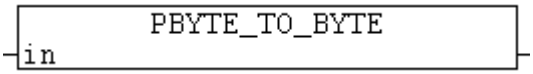
```
VAR_INPUT
    in : POINTER TO BOOL;
END_VAR
```

名称	类型	描述
in	POINTER TO BOOL	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.2 PBYTE_TO_BYTE



该函数返回 BYTE 指针变量的内容。

返回值

名称	类型	描述
PBYTE_TO_BYTE	BYTE	

输入

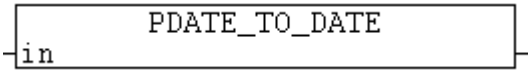
```
VAR_INPUT
    in : POINTER TO BYTE;
END_VAR
```

名称	类型	描述
in	POINTER TO BYTE	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.3 PDATE_TO_DATE



该函数返回 DATE 指针变量的内容。

返回值

名称	类型	描述
PDATE_TO_DATE	DATE	

输入

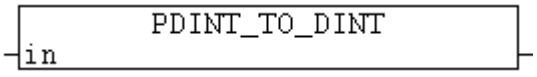
```
VAR_INPUT
    in : POINTER TO DATE;
END_VAR
```

名称	类型	描述
in	POINTER TO DATE	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.4 PDINT_TO_DINT



该函数返回 DINT 指针变量的内容。

返回值

名称	类型	描述
PDINT_TO_DINT	DINT	

输入

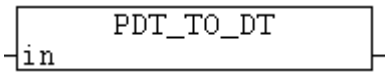
```
VAR_INPUT
    in : POINTER TO DINT;
END_VAR
```

名称	类型	描述
in	POINTER TO DINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.5 PDT_TO_DT



该函数返回 DT 指针变量的内容。

返回值

名称	类型	描述
PDT_TO_DT	DT	

输入

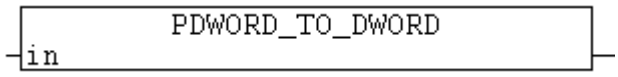
```
VAR_INPUT
    in : POINTER TO DT;
END_VAR
```

名称	类型	描述
in	POINTER TO DT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.6 PDWORD_TO_DWORD



该函数返回 DWORD 指针变量的内容。

返回值

名称	类型	描述
PDWORD_TO_DWORD	DWORD	

输入

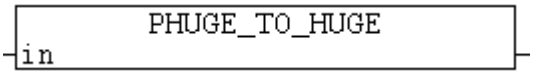
```
VAR_INPUT
    in : POINTER TO DWORD;
END_VAR
```

名称	类型	描述
in	POINTER TO DWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.7 PHUGE_TO_HUGE



该函数返回 T_HUGE_INTEGER [▶ 346] 指针变量的内容。

返回值

名称	类型	描述
PHUGE_TO_HUGE	T_HUGE_INTEGER	

输入

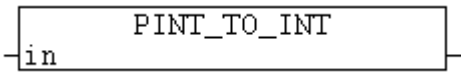
```
VAR_INPUT
    in : POINTER TO T_HUGE_INTEGER;
END_VAR
```

名称	类型	描述
in	POINTER TO T_HUGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.8 PINT_TO_INT



该函数返回 INT 指针变量的内容。

返回值

名称	类型	描述
PINT_TO_INT	INT	

输入

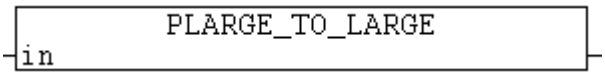
```
VAR_INPUT
    in : POINTER TO INT;
END_VAR
```

名称	类型	描述
in	POINTER TO INT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.9 PLARGE_TO_LARGE



该函数返回 T_LARGE_INTEGER [▶ 346] 指针变量的内容。

返回值

名称	类型	描述
PLARGE_TO_LARGE	T_LARGE_INTEGER	

输入

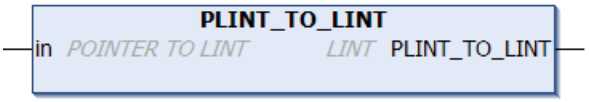
```
VAR_INPUT
    in : POINTER TO T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	POINTER TO T_LARGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.10 PLINT_TO_LINT



该函数返回 LINT 指针变量的内容。

返回值

名称	类型	描述
PLINT_TO_LINT	LINT	

输入

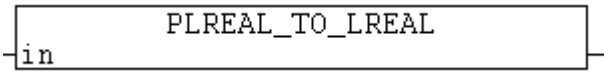
```
VAR_INPUT
    in : POINTER TO LINT;
END_VAR
```

名称	类型	描述
in	POINTER TO LINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.11 PLREAL_TO_LREAL



该函数返回 LREAL 指针变量的内容。

返回值

名称	类型	描述
PLREAL_TO_LREAL	LREAL	

输入

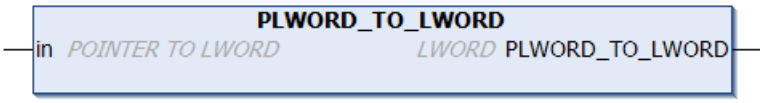
```
VAR_INPUT
    in : POINTER TO LREAL;
END_VAR
```

名称	类型	描述
in	POINTER TO LREAL	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.12 PLWORD_TO_LWORD



该函数返回 LWORD 指针变量的内容。

返回值

名称	类型	描述
PLWORD_TO_LWORD	LWORD	

输入

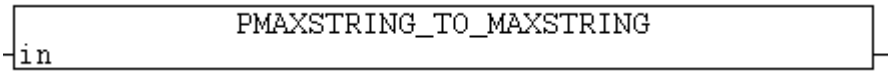
```
VAR_INPUT
    in : POINTER TO LWORD;
END_VAR
```

名称	类型	描述
in	POINTER TO LWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.13 PMAXSTRING_TO_MAXSTRING



该函数返回 T_MaxString 指针变量的内容。

返回值

名称	类型	描述
PMAXSTRING_TO_MAXSTRING	T_MaxString	

输入

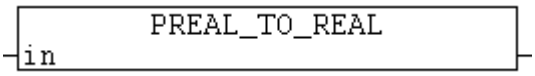
```
VAR_INPUT
    in : POINTER TO T_MaxString;
END_VAR
```

名称	类型	描述
in	POINTER TO T_MaxString	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.14 PREAL_TO_REAL



该函数返回 REAL 指针变量的内容。

返回值

名称	类型	描述
PREAL_TO_REAL	REAL	

输入

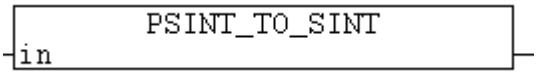
```
VAR_INPUT
    in : POINTER TO REAL;
END_VAR
```

名称	类型	描述
in	POINTER TO REAL	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.15 PSINT_TO_SINT



该函数返回 SINT 指针变量的内容。

返回值

名称	类型	描述
PSINT_TO_SINT	SINT	

输入

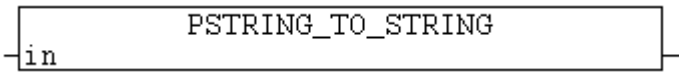
```
VAR_INPUT
    in : POINTER TO SINT;
END_VAR
```

名称	类型	描述
in	POINTER TO SINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.16 PSTRING_TO_STRING



该函数返回 STRING 指针变量的内容。

返回值

名称	类型	描述
PSTRING_TO_STRING	STRING	

输入

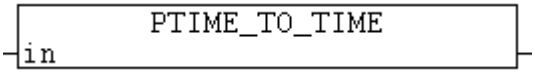
```
VAR_INPUT
    in : POINTER TO STRING;
END_VAR
```

名称	类型	描述
in	POINTER TO String	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.17 PTIME_TO_TIME



该函数返回 TIME 指针变量的内容。

返回值

名称	类型	描述
PTIME_TO_TIME	TIME	

输入

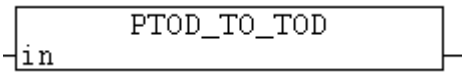
```
VAR_INPUT
    in : POINTER TO TIME;
END_VAR
```

名称	类型	描述
in	POINTER TO TIME	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.18 PTOD_TO_TOD



该函数返回 TOD 指针变量的内容。

返回值

名称	类型	描述
PTOD_TO_TOD	TOD	

输入

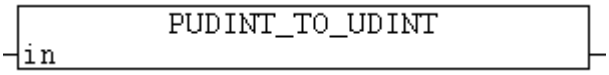
```
VAR_INPUT
    in : POINTER TO TOD;
END_VAR
```

名称	类型	描述
in	POINTER TO TOD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.19 PUDINT_TO_UDINT



该函数返回 UDINT 指针变量的内容。

返回值

名称	类型	描述
PUDINT_TO_UDINT	UDINT	

输入

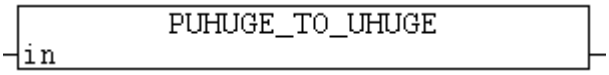
```
VAR_INPUT
    in : POINTER TO UDINT;
END_VAR
```

名称	类型	描述
in	POINTER TO UDINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.20 PUHUGE_TO_UHUGE



该函数返回 T_UHUGE_INTEGER [▶ 347] 指针变量的内容。

返回值

名称	类型	描述
PUHUGE_TO_UHUGE	_UHUGE_INTEGER	

输入

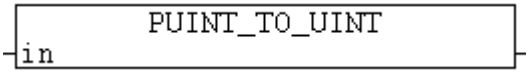
```
VAR_INPUT
    in : POINTER TO T_UHUGE_INTEGER;
END_VAR
```

名称	类型	描述
in	POINTER TO T_UHUGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.21 PUINT_TO_UINT



该函数返回 UINT 指针变量的内容。

返回值

名称	类型	描述
PUINT_TO_UINT	UINT	

输入

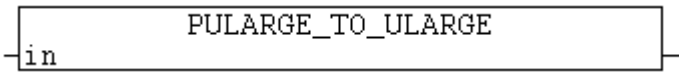
```
VAR_INPUT
    in : POINTER TO UINT;
END_VAR
```

名称	类型	描述
in	POINTER TO UINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.22 PULARGE_TO_ULARGE



该函数返回 T_ULARGE_INTEGER [▶ 347] 指针变量的内容。

返回值

名称	类型	描述
PULARGE_TO_ULARGE	_ULARGE_INTEGER	

输入

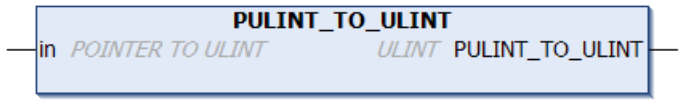
```
VAR_INPUT
    in : POINTER TO T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	POINTER TO T_ULARGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.23 PULINT_TO_ULINT



该函数返回 ULINT 指针变量的内容。

返回值

名称	类型	描述
PULINT_TO_ULINT	ULINT	

输入

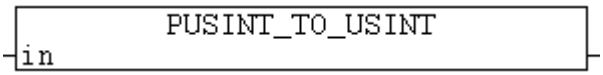
```
VAR_INPUT
    in : POINTER TO ULINT;
END_VAR
```

名称	类型	描述
in	POINTER TO ULINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.24 PUSINT_TO_USINT



该函数返回 USINT 指针变量的内容。

返回值

名称	类型	描述
PUSINT_TO_USINT	USINT	

输入

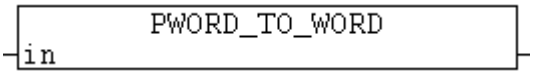
```
VAR_INPUT
    in : POINTER TO USINT;
END_VAR
```

名称	类型	描述
in	POINTER TO USINT	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.25 PWORD_TO_WORD



该函数返回 WORD 指针变量的内容。

返回值

名称	类型	描述
PWORD_TO_WORD	WORD	

输入

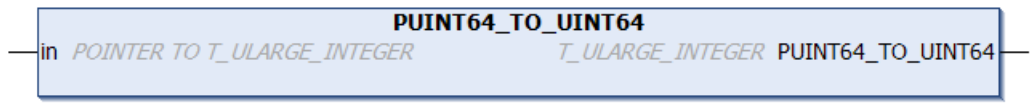
```
VAR_INPUT
    in : POINTER TO WORD;
END_VAR
```

名称	类型	描述
in	POINTER TO WORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.6.26 PUINT64_TO_UINT64



函数 PUINT64_TO_UINT64 返回 T_ULARGE_INTEGER [▶ 347] 指针变量的内容。

返回值

名称	类型	描述
PUINT64_TO_UINT64	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    in    :  POINTER TO T_ULARGE_INTEGER;
END_VAR
```

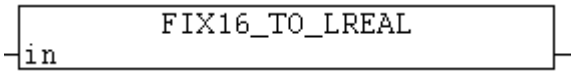
名称	类型	描述
in	POINTER TO T_ULARGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7 16 位定点数函数（有符号）

4.7.1 FIX16_TO_LREAL



将有符号的 16 位定点数转换为 LREAL 类型的浮点数。

返回值

名称	类型	描述
FIX16_TO_LREAL	LREAL	

输入

```
VAR_INPUT
    in : T_FIX16;
END_VAR
```

名称	类型	描述
in	T_FIX16 [▶ 344]	要转换的定点数。

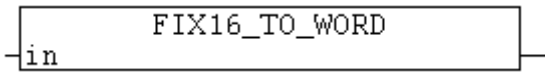
示例：

请参见函数说明：LREAL_TO_FIX16 [▶ 220]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.2 FIX16_TO_WORD



该函数将 16 位定点数转换为 WORD 变量（WORD 变量包含定点数的位数和小数位）。

返回值

名称	类型	描述
FIX16_TO_WORD	WORD	

输入

```
VAR_INPUT
    in : T_FIX16;
END_VAR
```

名称	类型	描述
in	T_FIX16 [▶ 344]	要转换的定点数

示例：

```
PROGRAM FIX_TO_WORD
VAR
    fp16 : WORD;
END_VAR
```

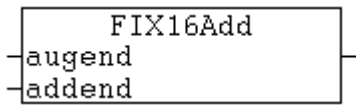
```
fp16 := FIX16_TO_WORD(LREAL_TO_FIX16(12.5, 8));
```

fp16 变量的值为：2#0000110010000000。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.3 FIX16Add



该函数将 2 个有符号的 16 位定点数相加。数字的小数位（精度）不必相同。在相加之前，小数位数较多的数字的精度会降低。即截断其多余的小数位。加法运算的结果是一个有符号的 16 位定点数。

返回值

名称	类型	描述
FIX16Add	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    augend : T_FIX16;
    addend : T_FIX16;
END_VAR
```

名称	类型	描述
augend	T_FIX16 [▶ 344]	第一个被加数
addend	T_FIX16 [▶ 344]	第二个被加数

示例：

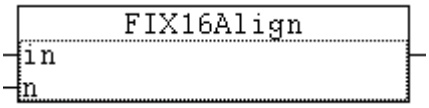
```
PROGRAM FIXADD
VAR
    a, b   : T_FIX16;
    result : LREAL;
END_VAR
```

```
a := LREAL_TO_FIX16( 0.5, 8 );
b := LREAL_TO_FIX16( -0.25, 8 );

result := FIX16_TO_LREAL( FIX16Add( a, b ) );(* The result is: 0.25 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.4 FIX16Align



该函数可以用于更改有符号的 16 位定点数的分辨率（小数位数）。该函数提供新的定点数作为返回参数。

返回值

名称	类型	描述
FIX16Align	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    in : T_FIX16;
    n  : BYTE(0..15);
END_VAR
```

名称	类型	描述
in	T_FIX16 [▶ 344]	要修改其分辨率的定点数
n	BYTE(0..15)	要修改其分辨率的定点数

示例：

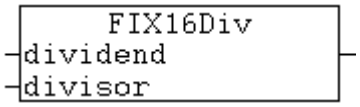
```
PROGRAM FIXALIGN
VAR
    q8, q4 : T_FIX16;
    result : LREAL;
END_VAR
```

```
q8 := LREAL_TO_FIX16( 0.6, 8 );
result := FIX16_TO_LREAL( q8 );(* The result is: 0.6015625 *)

q4 := FIX16Align( q8, 4 );
result := FIX16_TO_LREAL( q4 );(* The result is: 0.5625 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.5 FIX16Div



该函数将 2 个有符号的 16 位定点数相除。数字的小数位（精度）不必相同。在相除之前，小数位数较多的数字的精度会降低。即截断其多余的小数位。除法运算的结果是一个有符号的 16 位定点数。

返回值

名称	类型	描述
FIX16Div	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    dividend : T_FIX16;
    divisor  : T_FIX16;
END_VAR
```

名称	类型	描述
dividend	T_FIX16 [▶ 344]	被除数
divisor	T_FIX16 [▶ 344]	被除数除以的数字

示例：

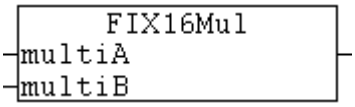
```
PROGRAM FIXDIV
VAR
    a, b   : T_FIX16;
    result : LREAL;
END_VAR
```

```
a := LREAL_TO_FIX16( -22.5, 8 );
b := LREAL_TO_FIX16( 10.0, 8 );

result := FIX16_TO_LREAL( FIX16Div( a, b ) ); (* The result is: -2.25 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.6 FIX16Mul



该函数将 2 个有符号的 16 位定点数相乘。数字的小数位（精度）不必相同。在相乘之前，小数位数较多的数字的精度会降低。即截断其多余的小数位。乘法运算的结果是一个有符号的 16 位定点数。

返回值

名称	类型	描述
FIX16Mul	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    multiA : T_FIX16;
    multiB : T_FIX16;
END_VAR
```

名称	类型	描述
multiA	T_FIX16 [▶ 344]	第一个乘数
multiB	T_FIX16 [▶ 344]	第二个乘数

示例：

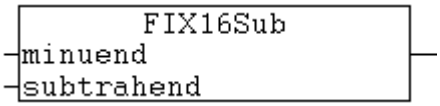
```
PROGRAM FIXMUL
VAR
    a, b   : T_FIX16;
    result : LREAL;
END_VAR
```

```
a := LREAL_TO_FIX16( 0.25, 8 );
b := LREAL_TO_FIX16( 10.0, 8 );

result := FIX16_TO_LREAL( FIX16Mul( a, b ) );(* The result is: 2.5 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.7 FIX16Sub



该函数将 2 个有符号的 16 位定点数相减。数字的小数位（精度）不必相同。在相减之前，小数位数较多的数字的精度会降低。即截断其多余的小数位。减法运算的结果是一个有符号的 16 位定点数。

返回值

名称	类型	描述
FIX16Sub	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    minuend    : T_FIX16;
    subtrahend : T_FIX16;
END_VAR
```

名称	类型	描述
minuend	T_FIX16 [▶ 344]	从其中减去一个值的数字
subtrahend	T_FIX16 [▶ 344]	被减数

示例：

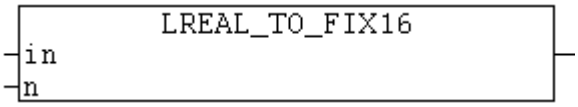
```
PROGRAM FIXSUB
VAR
    a, b   : T_FIX16;
    result : LREAL;
END_VAR
```

```
a := LREAL_TO_FIX16( 0.5, 8 );
b := LREAL_TO_FIX16( 0.75, 8 );

result := FIX16_TO_LREAL( FIX16Sub( a, b ) ); (* The result is: -0.25 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.8 LREAL_TO_FIX16



将 LREAL 类型的浮点数转换为具有所需小数位数的有符号的 16 位定点数。

返回值

名称	类型	描述
LREAL_TO_FIX16	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    in : LREAL;
    n  : WORD(0..15) := 15;
END_VAR
```

名称	类型	描述
in	LREAL	要转换的 LREAL 数字。
n	WORD(0..15)	所需的小数位数。

示例：

在下面的示例中，几个常量被转换为定点数。在转换时可以指定小数位数。请注意（与浮点数的转换类似），可能会出现舍入误差（在我们的示例中为 q2 和 q15）。

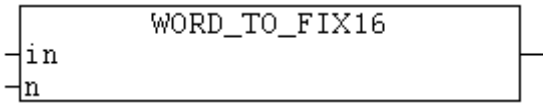
```
PROGRAM TEST
VAR
    q2, q4, q8, q12, q15 : T_FIX16;
    r2, r4, r8, r12, r15 : LREAL;
END_VAR
```

```
q2  := LREAL_TO_FIX16( 0.6, 2 );
q4  := LREAL_TO_FIX16( -0.25, 4 );
q8  := LREAL_TO_FIX16( -0.75, 8 );
q12 := LREAL_TO_FIX16( 2.30078125, 12 );
q15 := LREAL_TO_FIX16( 0.6, 15 );

r2  := FIX16_TO_LREAL( q2 );(* 0.5 *)
r4  := FIX16_TO_LREAL( q4 );(* -0.25 *)
r8  := FIX16_TO_LREAL( q8 );(* -0.75 *)
r12 := FIX16_TO_LREAL( q12 );(* 2.30078125 *)
r15 := FIX16_TO_LREAL( q15 );(* 0.600006103515625 *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.7.9 WORD_TO_FIX16



该函数将 WORD 变量转换为 16 位定点数（WORD 变量包含定点数的编码位数和小数位）。

返回值

名称	类型	描述
WORD_TO_FIX16	T_FIX16 [▶ 344]	

输入

```
VAR_INPUT
    in : WORD;(* 16 bit fixed point number *)
    n  : WORD(0..15);(* number of fractional bits *)
END_VAR
```

名称	类型	描述
in	WORD	16 位定点数
n	WORD(0..15)	分数位数

示例：

```
PROGRAM WORD_TO_FIX
VAR
    double : LREAL;
END_VAR
double := FIX16_TO_LREAL(WORD_TO_FIX16(2#0000110010000000, 8));
```

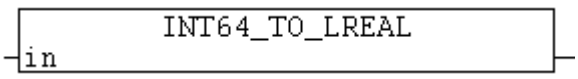
双变量的值为：12.5

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8 64 位函数（有符号）

4.8.1 INT64_TO_LREAL



该函数将 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）转换为 LREAL 类型的浮点数。

返回值

名称	类型	描述
INT64_TO_LREAL	LREAL	

输入

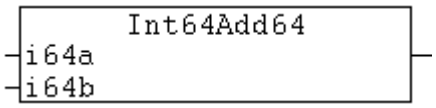
```
VAR_INPUT
    in : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.2 Int64Add64



该函数将 2 个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）相加。结果是一个有符号的 64 位数字。

返回值

名称	类型	描述
Int64Add64	T_LARGE_INTEGER	

输入

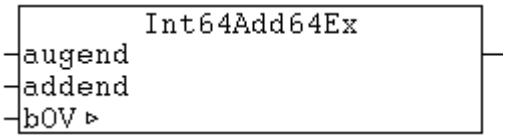
```
VAR_INPUT
i64a : T_LARGE_INTEGER;
i64b : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
i64a	T_LARGE_INTEGER	
i64b	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.3 Int64Add64Ex



该函数将 2 个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）相加。结果是一个有符号的 64 位数字。

返回值

名称	类型	描述
Int64Add64Ex	T_LARGE_INTEGER	

输入

```
VAR_INPUT
    augend : T_LARGE_INTEGER;
    addend : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
augend	T_LARGE_INTEGER	
addend	T_LARGE_INTEGER	

/ 输入/输出

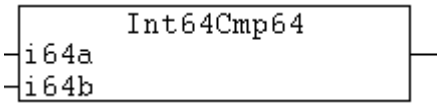
```
VAR_IN_OUT
    bOV    : BOOL;(* TRUE => arithmetic overflow, FALSE => no overflow *)
END_VAR
```

名称	类型	描述
bOV	BOOL	算术溢出。TRUE => 溢出，FALSE => 无溢出。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.4 Int64Cmp64



该函数将 2 个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）进行比较。

返回值

名称	类型	描述
Int64Cmp64	DINT	

输入

```
VAR_INPUT
    i64a : T_LARGE_INTEGER;
    i64b : T_LARGE_INTEGER;
END_VAR
```

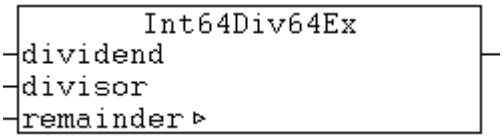
名称	类型	描述
i64a	T_LARGE_INTEGER	如果返回参数 -1：i64a 小于 i64b 0：i64a 与 i64b 相同 1：i64a 大于 i64b
i64b	T_LARGE_INTEGER	

返回参数	描述
-1	i64a 小于 *i64b*
0	i64a 与 *i64b* 相同
1	i64a 大于 *i64b*

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.5 Int64Div64Ex



该函数将 2 个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）相除。结果是一个有符号的 64 位数字。

返回值

名称	类型	描述
Int64Div64Ex	T_LARGE_INTEGER	

输入

```
VAR_INPUT
dividend  : T_LARGE_INTEGER;
divisor   : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
dividend	T_LARGE_INTEGER	被除数
divisor	T_LARGE_INTEGER	被除数除以的数字

输入/输出

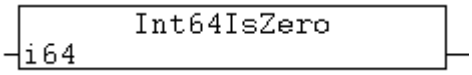
```
VAR_IN_OUT
remainder : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
remainder	T_LARGE_INTEGER	剩余部分

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.6 Int64IsZero



如果 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）的值为零，则该函数返回 TRUE。

返回值

名称	类型	描述
Int64isZero	BOOL	

输入

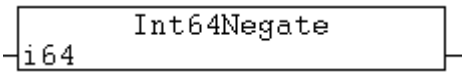
```
VAR_INPUT
    i64 : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
i64	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.7 Int64Negate



该函数对一个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）取反。

返回值

名称	类型	描述
Int64Negate	T_LARGE_INTEGER	

输入

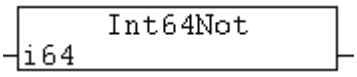
```
VAR_INPUT
    i64 : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
i64	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.8 Int64Not



TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）的按位非。结果是一个有符号的 64 位数字。

返回值

名称	类型	描述
Int64Not	T_LARGE_INTEGER	

输入

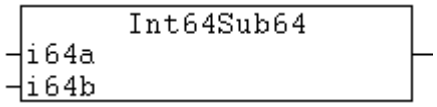
```
VAR_INPUT
    i64 : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
i64	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.9 Int64Sub64



该函数将 2 个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）相减。结果是一个有符号的 64 位数字。

返回值

名称	类型	描述
Int64Sub64	T_LARGE_INTEGER	

输入

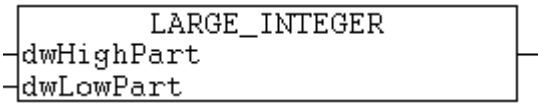
```
VAR_INPUT
    i64a : T_LARGE_INTEGER;
    i64b : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
i64a	T_LARGE_INTEGER	
i64b	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.10 LARGE_INTEGER



该函数对一个 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）进行初始化。

返回值

名称	类型	描述
LARGE_INTEGER	T_LARGE_INTEGER	

输入

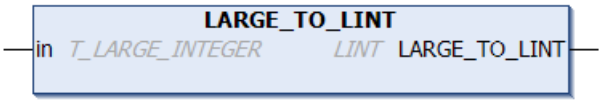
```
VAR_INPUT
    dwHighPart : DWORD;
    dwLowPart  : DWORD;
END_VAR
```

名称	类型	描述
dwHighPart	DWORD	高 32 位
dwLowPart	DWORD	低 32 位

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.11 LARGE_TO_LINT



该函数将 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）转换为 TwinCAT 3 有符号的 64 位数字（“本地”类型）。

返回值

名称	类型	描述
LARGE_TO_LINT	LINT	

输入

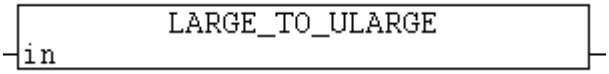
```
VAR_INPUT
    in : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.12 LARGE_TO_ULARGE



该函数将 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）转换为 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
LARGE_TO_ULARGE	T_ULARGE_INTEGER	

输入

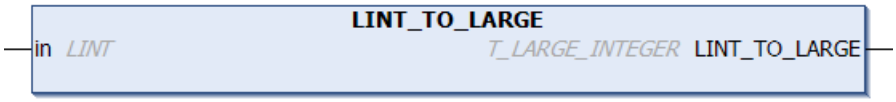
```
VAR_INPUT
    in : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.13 LINT_TO_LARGE



该函数将 TwinCAT 3 有符号的 64 位数字（“本地”类型）转换为 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）。

返回值

名称	类型	描述
LINT_TO_LARGE	T_LARGE_INTEGER	

输入

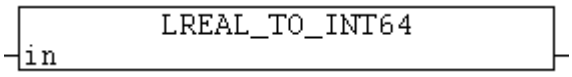
```
VAR_INPUT
    in : LINT;
END_VAR
```

名称	类型	描述
in	LINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.14 LREAL_TO_INT64



该函数将 LREAL 数字转换为 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）。

返回值

名称	类型	描述
LREAL_TO_INT64	T_LARGE_INTEGER	

输入

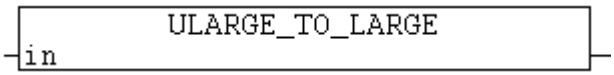
```
VAR_INPUT
    in : LREAL;
END_VAR
```

名称	类型	描述
in	LREAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.8.15 ULARGE_TO_LARGE



该函数将 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）转换为 TwinCAT 2 有符号的 64 位数字（“传统”类型：T_LARGE_INTEGER [▶ 346]）。

返回值

名称	类型	描述
ULARGE_TO_LARGE	T_LARGE_INTEGER	

输入

```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

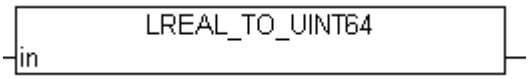
名称	类型	描述
in	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9 64 位整数函数（无符号）

4.9.1 LREAL_TO_UINT64



该函数将 LREAL 数字转换为 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
LREAL_TO_UINT64	T_ULARGE_INTEGER	

输入

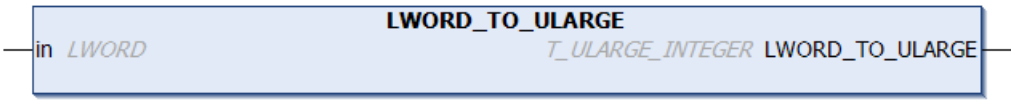
```
VAR_INPUT
    in : LREAL;
END_VAR
```

名称	类型	描述
in	LREAL	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.2 LWORD_TO_ULARGE



该函数将 TwinCAT 3 无符号的 64 位数字（“本地”类型）转换为 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
LWORD_TO_ULARGE	T_ULARGE_INTEGER	

输入

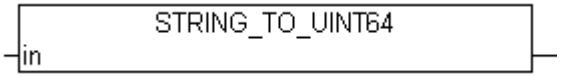
```
VAR_INPUT
    in : LWORD;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.3 STRING_TO_UINT64



该函数将字符串转换为 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
STRING_TO_UINT64	T_ULARGE_INTEGER	

输入

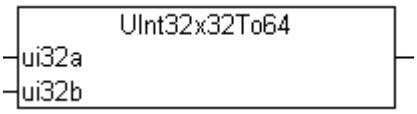
```
VAR_INPUT
    in : STRING(21);
END_VAR
```

名称	类型	描述
in	STRING(21)	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.4 UInt32x32To64



该函数将 2 个无符号的 32 位数字相乘。结果是一个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
UInt32x32To64	T_ULARGE_INTEGER	

输入

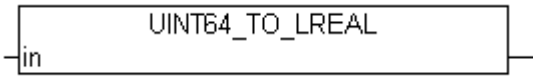
```
VAR_INPUT
    ui32a : DWORD;
    ui32b : DWORD;
END_VAR
```

名称	类型	描述
ui32a	DWORD	
ui32b	DWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.5 UINT64_TO_LREAL



该函数将 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）转换为 LREAL 类型的浮点数。

返回值

名称	类型	描述
UINT64_TO_LREAL	LREAL	

输入

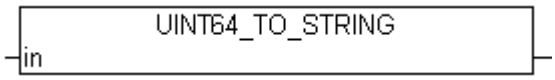
```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.6 UINT64_TO_STRING



该函数将 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）转换为字符串。

返回值

名称	类型	描述
UINT64_TO_STRING	STRING(21)	

输入

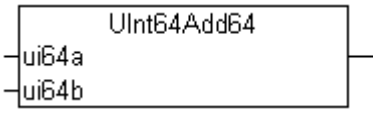
```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER	要转换的数字。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.7 UInt64Add64



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相加。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Add64	T_ULARGE_INTEGER	

输入

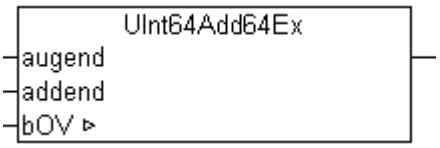
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.8 UInt64Add64Ex



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相加。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Add64Ex	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    augend : T_ULARGE_INTEGER;
    addend : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
augend	T_ULARGE_INTEGER	
addend	T_ULARGE_INTEGER	

/ 输入/输出

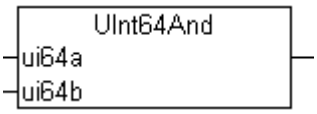
```
VAR_IN_OUT
    bOV    : BOOL;(* TRUE => arithmetic overflow, FALSE => no overflow *)
END_VAR
```

名称	类型	描述
bOV	BOOL	算术溢出。TRUE => 溢出，FALSE => 无溢出。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.9 UInt64And



2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位与。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64And	T_ULARGE_INTEGER	

输入

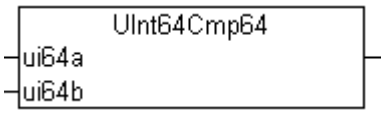
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.10 UInt64Cmp64



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）进行比较。

返回值

名称	类型	描述
UInt64Cmp64	DINT	

VAR_INPUT

```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	-1：ui64a 小于 ui64b
ui64b	T_ULARGE_INTEGER	0：ui64a 与 ui64b 相同
		1：ui64a 大于 ui64b

返回参数	描述
-1	ui64a 小于 *ui64b*
0	ui64a 与 *ui64b* 相同
1	ui64a 大于 *ui64b*

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.11 UInt64Div16Ex

该函数将一个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）除以一个无符号的 16 位数字。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Div16Ex	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    dividend : T_ULARGE_INTEGER;
    divisor  : WORD;
END_VAR
```

名称	类型	描述
dividend	T_ULARGE_INTEGER	被除数
divisor	WORD	被除数除以的数字

/ 输入/输出

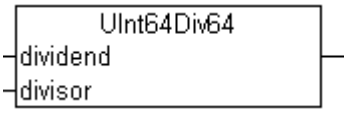
```
VAR_IN_OUT
    remainder : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
remainder	T_ULARGE_INTEGER	剩余部分

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.12 UInt64Div64



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相除。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Div64	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    dividend  : T_ULARGE_INTEGER;
    divisor   : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
dividend	T_ULARGE_INTEGER	被除数
divisor	T_ULARGE_INTEGER	被除数除以的数字

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.13 UInt64Div64Ex

该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相除。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Div64Ex	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    dividend  : T_ULARGE_INTEGER;
    divisor   : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
dividend	T_ULARGE_INTEGER	被除数
divisor	T_ULARGE_INTEGER	被除数除以的数字

输入/输出

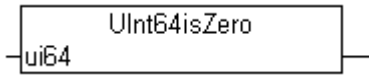
```
VAR_IN_OUT
    remainder : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
remainder	T_ULARGE_INTEGER	剩余部分

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.14 UInt64isZero



如果 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的值为零，则该函数返回 TRUE。

返回值

名称	类型	描述
UInt64isZero	BOOL	

输入

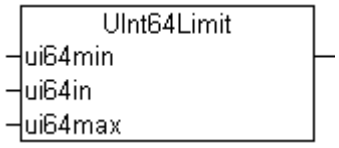
```
VAR_INPUT
    ui64 : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.15 UInt64Limit



限值。结果是一个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
UInt64Limit	T_ULARGE_INTEGER	

输入

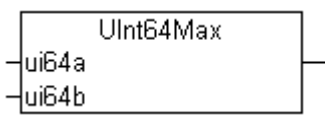
```
VAR_INPUT
    ui64min : T_ULARGE_INTEGER;
    ui64in  : T_ULARGE_INTEGER;
    ui64max : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64min	T_ULARGE_INTEGER	
ui64in	T_ULARGE_INTEGER	
ui64max	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.16 UInt64Max



最大函数。返回 2 个值中的较大值（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
UInt64Max	T_ULARGE_INTEGER	

输入

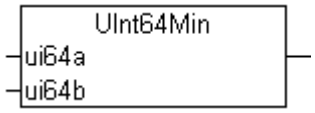
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.17 UInt64Min



最小函数。返回 2 个值中的较小值（“传统”类型：T_ULARGE_INEGER [▶ 347]）。

返回值

名称	类型	描述
UInt64Min	T_ULARGE_INTEGER	

输入

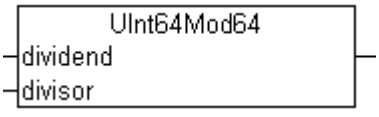
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.18 UInt64Mod64



一个 TwinCAT 2 无符号的 64 位数字与另一个数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）进行模除。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Mod64	T_ULARGE_INTEGER	

输入

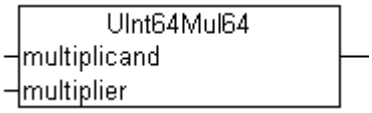
```
VAR_INPUT
    dividend : T_ULARGE_INTEGER;
    divisor  : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
dividend	T_ULARGE_INTEGER	被除数
divisor	T_ULARGE_INTEGER	被除数除以的数字

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.19 UInt64Mul64



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相乘。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Mul64	T_ULARGE_INTEGER	

输入

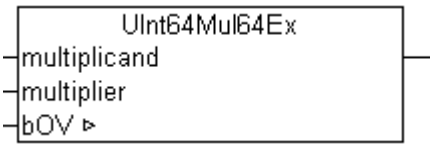
```
VAR_INPUT
    multiplicand : T_ULARGE_INTEGER;
    multiplier   : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
multiplicand	T_ULARGE_INTEGER	
multiplier	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.20 UInt64Mul64Ex



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相乘。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Mul64Ex	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    multiplicand  : T_ULARGE_INTEGER;
    multiplier    : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
multiplicand	T_ULARGE_INTEGER	
multiplier	T_ULARGE_INTEGER	

输入/输出

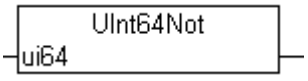
```
VAR_IN_OUT
    bOV : BOOL;(* TRUE => Arithmetic overflow, FALSE => no overflow *)
END_VAR
```

名称	类型	描述
bOV	BOOL	算术溢出。TRUE => 溢出，FALSE => 无溢出。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.21 UInt64Not



TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）进行按位取反。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Not	T_ULARGE_INTEGER	

输入

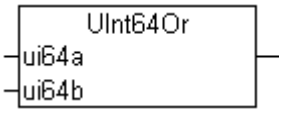
```
VAR_INPUT
    ui64 : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.22 UInt64Or



2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位或运算。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Or	T_ULARGE_INTEGER	

输入

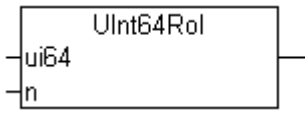
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.23 UInt64Rol



TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位左旋转。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Rol	T_ULARGE_INTEGER	

输入

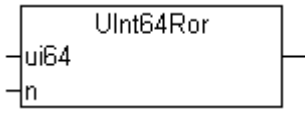
```
VAR_INPUT
    ui64 : T_ULARGE_INTEGER;
    n    : DWORD;
END_VAR
```

名称	类型	描述
ui64	T_ULARGE_INTEGER	
n	DWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.24 UInt64Ror



TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位右旋转。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Ror	T_ULARGE_INTEGER	

输入

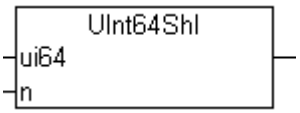
```
VAR_INPUT
    ui64 : T_ULARGE_INTEGER;
    n    : DWORD;
END_VAR
```

名称	类型	描述
ui64	T_ULARGE_INTEGER	
n	DWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.25 UInt64Shl



TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位左移。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Shl	T_ULARGE_INTEGER	

输入

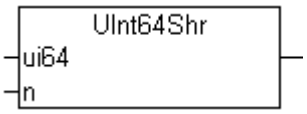
```
VAR_INPUT
    ui64 : T_ULARGE_INTEGER;
    n    : DWORD;
END_VAR
```

名称	类型	描述
ui64	T_ULARGE_INTEGER	
n	DWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.26 UInt64Shr



TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位右移。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Shr	T_ULARGE_INTEGER	

输入

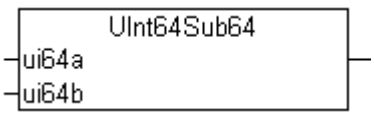
```
VAR_INPUT
    ui64 : T_ULARGE_INTEGER;
    n    : DWORD;
END_VAR
```

名称	类型	描述
ui64	T_ULARGE_INTEGER	
n	DWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.27 UInt64Sub64



该函数将 2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）相减。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Sub64	T_ULARGE_INTEGER	

输入

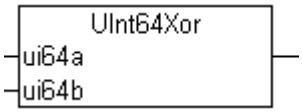
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.28 UInt64Xor



2 个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）的按位异或。结果是一个无符号的 64 位数字。

返回值

名称	类型	描述
UInt64Xor	T_ULARGE_INTEGER	

输入

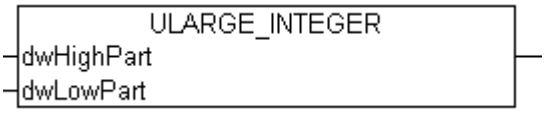
```
VAR_INPUT
    ui64a : T_ULARGE_INTEGER;
    ui64b : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
ui64a	T_ULARGE_INTEGER	
ui64b	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.29 ULARGE_INTEGER



该函数对一个 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）进行初始化。

返回值

名称	类型	描述
ULARGE_INTEGER	T_ULARGE_INTEGER	

输入

```
VAR_INPUT
    dwHighPart : DWORD;
    dwLowPart  : DWORD;
END_VAR
```

名称	类型	描述
dwHighPart	DWORD	高 32 位
dwLowPart	DWORD	低 32 位

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.30 ULARGE_TO_ULINT

该函数将 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）转换为 TwinCAT 3 无符号的 64 位数字（“本地”类型）。

返回值

名称	类型	描述
ULARGE_TO_ULINT	ULINT	

输入

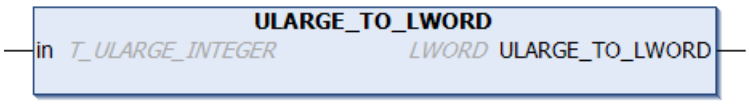
```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.9.31 ULARGE_TO_LWORD



该函数将 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）转换为 TwinCAT 3 无符号的 64 位数字（“本地”类型）。

返回值

名称	类型	描述
ULARGE_TO_LWORD	LWORD	

输入

```
VAR_INPUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10 T_Arg 帮助函数

4.10.1 F_ARGCMP

该函数将 2 个 T_Arg 类型的变量进行比较，并提供比较结果作为返回参数。

返回值

名称	类型	描述
F_ARGCMP	DINT	

输入

```
VAR_INPUT
typeSafe  : BOOL;
arg1      : T_Arg;
arg2      : T_Arg;
END_VAR
```

名称	类型	描述
typeSafe	BOOL	如果为 TRUE => 可以比较相同的类型（类型安全比较）。FALSE => 可以比较不同的类型（与类型无关的比较）。
arg1	T_Arg	要比较的第一个变量（类型：T_Arg [▶ 343]）。
arg2	T_Arg	要比较的第二个变量（类型：T_Arg [▶ 343]）。

返回参数	第一个变量和第二个变量中的第一个不同字节（类型、长度、值）的关系
-3	*arg1* 的长度小于 *arg2*
-2	*arg1* 的类型小于 *arg2*
-1	*arg1* 的值小于 *arg2*
0	arg1 与 *arg2* 相同
1	*arg1* 的值大于 *arg2*
2	*arg1* 的类型大于 *arg2*
3	*arg1* 的长度大于 *arg2*
0xFF	错误的参数值、类型、长度，*arg1* 或 *arg2* 的值 = 0

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.2 F_ARGCPY

该函数将 T_Arg 类型的变量的值复制到另一个变量中，并提供成功复制的数据字节数作为返回参数。

返回值

名称	类型	描述
F_ARGCPY	UDINT	

输入

```
VAR_INPUT
typeSafe : BOOL;
END_VAR
```

名称	类型	描述
typeSafe	BOOL	如果为 TRUE => 可以比较相同的类型（类型安全比较）。FALSE => 可以比较不同的类型（与类型无关的比较）。

输入/输出

```
VAR_IN_OUT
dest : T_Arg;
src  : T_Arg;
END_VAR
```

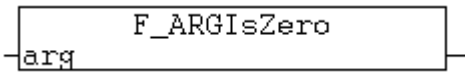
名称	类型	描述
dest	T_Arg	复制操作的目标变量（类型：T_Arg [▶ 343]）。
src	T_Arg	复制操作的源变量（类型：T_Arg [▶ 343]）。

返回参数	含义
0	不正确的参数值。*dest* 或 *src* 的类型、长度或值 == 0
> 0	如果成功，则为复制的字节数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.3 F_ARGISZERO



如果其中一个 T_Arg 成员变量的值为零或未初始化，则该函数返回 TRUE。

返回值

名称	类型	描述
F_ARGISZERO	BOOL	

输入

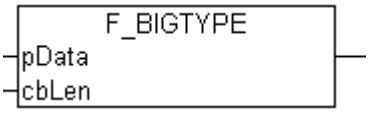
```
VAR_INPUT
    arg : T_Arg;
END_VAR
```

名称	类型	描述
arg	T_Arg	要检查的变量（类型：T_Arg [▶ 343]）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.4 F_BIGTYPE



辅助函数，返回一个具有关于结构或数组变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_BIGTYPE	T_Arg	

输入

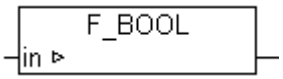
```
VAR_INPUT
pData : POINTER TO BYTE;
cbLen : DWORD;
END_VAR
```

名称	类型	描述
pData	POINTER TO BYTE	地址指针（使用 ADR 运算符可以确定）。
cbLen	DWORD	在内存中占用的字节数（使用 SIZEOF 运算符可以确定）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.5 F_BOOL



辅助函数，返回一个具有关于 BOOL 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_BOOL	T_Arg	

/ 输入/输出

```
VAR_IN_OUT
in : BOOL;
END_VAR
```

名称	类型	描述
in	BOOL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.6 F_BYTE

辅助函数，返回一个具有关于 BYTE 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_BYTE	T_Arg	

/ 输入/输出

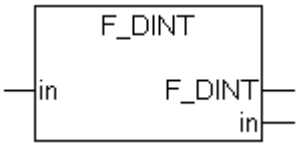
```
VAR_IN_OUT
in : BYTE;
END_VAR
```

名称	类型	描述
in	BYTE	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.7 F_DINT



辅助函数，返回一个具有关于 DINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_DINT	T_Arg	

/ 输入/输出

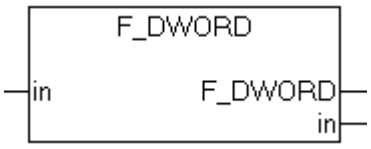
```
VAR_IN_OUT
in : DINT;
END_VAR
```

名称	类型	描述
in	DINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.8 F_DWORD



辅助函数，返回一个具有关于 DWORD 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_DWORD	T_Arg	

/ 输入/输出

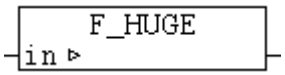
```
VAR_IN_OUT
in : DWORD;
END_VAR
```

名称	类型	描述
in	DWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.9 F_HUGE



辅助函数，返回一个具有关于 T_HUGE_INTEGER [▶ 346] 变量（有符号的 128 位整数，TwinCAT 2“传统”类型）的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_HUGE	T_Arg	

/ 输入/输出

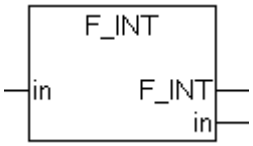
```
VAR_IN_OUT
    in : T_HUGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_HUGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.10 F_INT



辅助函数，返回一个具有关于 INT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_INT	T_Arg	

/ 输入/输出

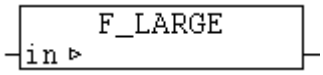
```
VAR_IN_OUT
in : INT;
END_VAR
```

名称	类型	描述
in	INT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.11 F_LARGE



辅助函数，返回一个具有关于 T_LARGE_INTEGER [▶ 346] 变量（有符号的 64 位整数，TwinCAT 2“传统”类型）的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_LARGE	T_Arg	

输入/输出

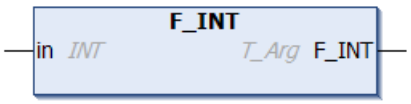
```
VAR_IN_OUT
    in : T_LARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_LARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.12 F_LINT



辅助函数，返回一个具有关于 LINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_LINT	T_Arg	

输入/输出

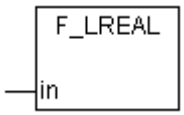
```
VAR_IN_OUT
    in : LINT;
END_VAR
```

名称	类型	描述
in	LINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.13 F_LREAL



辅助函数，返回一个具有关于 LREAL 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_LREAL	T_Arg	

/ 输入/输出

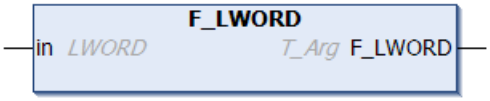
```
VAR_IN_OUT
    in : LREAL;
END_VAR
```

名称	类型	描述
in	LREAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.14 F_LWORD



辅助函数，返回一个具有关于 LWORD 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_LWORD	T_Arg	

/ 输入/输出

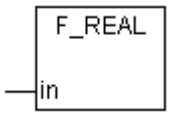
```
VAR_IN_OUT
    in : LWORD;
END_VAR
```

名称	类型	描述
in	LWORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.15 F_REAL



辅助函数，返回一个具有关于 REAL 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_REAL	T_Arg	

/ 输入/输出

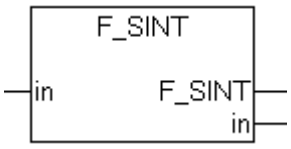
```
VAR_IN_OUT
    in : REAL;
END_VAR
```

名称	类型	描述
in	REAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.16 F_SINT



辅助函数，返回一个具有关于 SINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_SINT	T_Arg	

/ 输入/输出

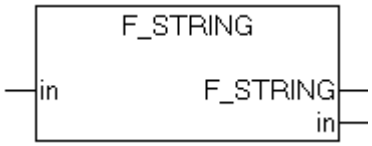
```
VAR_IN_OUT
    in : SINT;
END_VAR
```

名称	类型	描述
in	SINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.17 F_STRING



辅助函数，返回一个具有关于 T_MaxString 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_STRING	T_Arg	

/ 输入/输出

```
VAR_IN_OUT
    in : T_MaxString;
END_VAR
```

名称	类型	描述
in	T_MaxString	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.18 F_STRINGEx

辅助函数，返回一个具有关于 STRING 变量的信息（类型：T_Arg [▶ 343]）的结构。与 F_STRING [▶ 255] 函数不同，传输的 STRING 变量的长度是任意的。

返回值

名称	类型	描述
F_STRINGEx	T_Arg	

/ 输入/输出

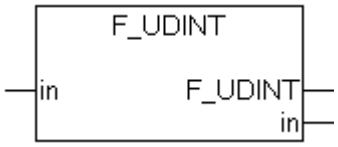
```
VAR_IN_OUT CONSTANT
    in : STRING;
END_VAR
```

名称	类型	描述
in	STRING	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4022	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统） >= v3.3.34.0

4.10.19 F_UDINT



辅助函数，返回一个具有关于 UDINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_UDINT	T_Arg	

/ 输入/输出

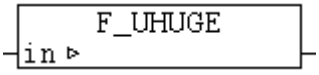
```
VAR_IN_OUT
    in : UDINT;
END_VAR
```

名称	类型	描述
in	UDINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.20 F_UHUGE



辅助函数，返回一个具有关于 T_UHUGE_INTEGER [▶ 347] 变量（无符号的 128 位整数，TwinCAT 2“传统”类型）的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_UHUGE	T_Arg	

/ 输入/输出

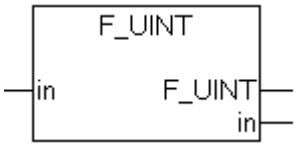
```
VAR_IN_OUT
    in : T_UHUGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_UHUGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.21 F_UINT



辅助函数，返回一个具有关于 UINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_UINT	T_Arg	

/ 输入/输出

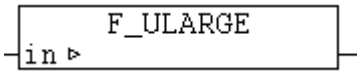
```
VAR_IN_OUT
    in : UINT;
END_VAR
```

名称	类型	描述
in	UINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.22 F_ULARGE



辅助函数，返回一个具有关于 T_LARGE_INTEGER [▶ 347] 变量（无符号的 64 位整数，TwinCAT 2“传统”类型）的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_ULARGE	T_Arg	

/ 输入/输出

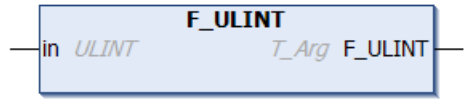
```
VAR_IN_OUT
    in : T_ULARGE_INTEGER;
END_VAR
```

名称	类型	描述
in	T_ULARGE_INTEGER	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.23 F_ULINT



辅助函数，返回一个具有关于 ULINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_ULINT	T_Arg	

/ 输入/输出

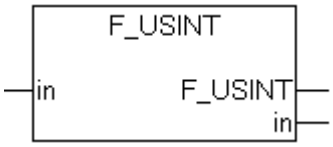
```
VAR_IN_OUT
    in : ULINT;
END_VAR
```

名称	类型	描述
in	ULINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.24 F_USINT



辅助函数，返回一个具有关于 USINT 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_USINT	T_Arg	

/ 输入/输出

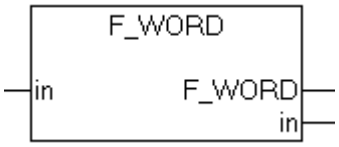
```
VAR_IN_OUT
    in : USINT;
END_VAR
```

名称	类型	描述
in	USINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.25 F_WORD



辅助函数，返回一个具有关于 WORD 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_WORD	T_Arg	

输入/输出

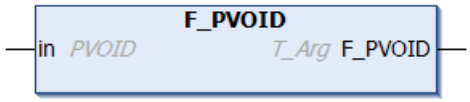
```
VAR_IN_OUT
    in : WORD;
END_VAR
```

名称	类型	描述
in	WORD	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.26 F_PVOID



辅助函数，返回一个具有关于 PVOID 变量的信息（类型：T_Arg [▶ 343]）的结构。

返回值

名称	类型	描述
F_PVOID	T_Arg	

输入/输出

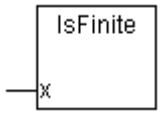
```
VAR_IN_OUT
    in : PVOID;
END_VAR
```

名称	类型	描述
in	PVOID	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.10.27 IsFinite



如果函数 IsFinite() 的参数有一个有限值（INF < x < +INF），则该函数返回 TRUE。如果参数为无限值或 NaN（NaN = 非数字），则该函数返回 FALSE。IsFinite() 检查 LREAL 或 REAL 变量的格式是否符合 IEEE 标准。

如果数学运算的结果超出了可以表示的范围，则在运行时系统中可能会出现 INF 数字。例如：

```
PROGRAM MAIN
VAR
    fSingle : REAL := 12.34;
END_VAR
```

```
(*Cyclic called program code*)
fSingle := fSingle*2;
```

如果通过非法访问（例如通过使用MEMCPY或MEMSET函数）覆盖了 NaN 数字的实际格式（内存内容），则在运行时系统中可能会出现 NaN 数字。例如：

```
PROGRAM MAIN
VAR
    fSingle : REAL := 12.34;
END_VAR
```

```
(*Cyclic called program code*)
MEMSET( ADR( fSingle ), 16#FF, SIZEOF( fSingle ) ); (* Invalid initialization of REAL variable *)
```

调用以 NaN 或 INF 数字作为参数的转换函数会导致在 PC 系统（x86、x64）上出现 FPU 异常。该异常随后会导致 PLC 停止。函数 IsFinite() 可以检查变量的值，从而避免 FPU 异常，并继续执行程序。

返回值

名称	类型	描述
IsFinite	BOOL	

输入

```
VAR_INPUT
    x : T_Arg;
END_VAR
```

名称	类型	描述
x	T_Arg	

x：辅助结构，具有关于要检查的 REAL 或 LREAL 变量的信息（类型：T_Arg [▶ 343]）。从辅助函数 F_REAL [▶ 254] 或 F_LREAL [▶ 253] 调用 IsFinite() 时，必须生成结构参数，并将其作为参数传输。

示例 1：

在下面的示例中，检查了 REAL 和 LREAL 变量的格式，从而避免了 FPU 异常。

```
PROGRAM MAIN
VAR
    fSingle         : REAL := 12.34;
    fDouble         : LREAL := 56.78;
    singleAsString  : STRING;
    doubleAsString  : STRING;
END_VAR
```

```
fSingle := fSingle*2;
IF IsFinite( F_REAL( fSingle ) ) THEN
    singleAsString := REAL_TO_STRING( fSingle );
ELSE
    (* report error !*)
    fSingle := 12.34;
END_IF

fDouble := fDouble*2;
IF IsFinite( F_LREAL( fDouble ) ) THEN
    doubleAsString := LREAL_TO_STRING( fDouble );
ELSE
    (* report error !*)
    fDouble := 56.78;
END_IF
```

示例 2：

在以下情况中，通过检查 IsFinite()，无法避免 FPU 异常：

```
PROGRAM MAIN
VAR
    bigFloat   : LREAL := 3.0E100;
    smallDigit : INT;
END_VAR
```

```
IF IsFinite( F_LREAL( bigFloat ) ) THEN
    smallDigit := LREAL_TO_INT( bigFloat );
END_IF
```

虽然 bigFloat 变量具有正确的格式，但变量值过大，无法转换为 INT 类型。在 PC 系统（x86、x64）上触发异常，运行时系统停止。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11 [废弃]

4.11.1 时间函数

4.11.1.1 DT_TO_FILETIME

函数“DT_TO_FILETIME”可以用于将 DATE_AND_TIME 格式（DT）的 PLC 变量转换为 FILETIME 格式（64 位）。

返回值

名称	类型	描述
DT_TO_FILETIME	T_FILETIME [▶ 344]	

输入

```
VAR_INPUT
    DTIN  : DT;
END_VAR
```

名称	类型	描述
DTIN	DT	以 DATE_AND_TIME 格式表示的要转换的日期和时间。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.1.2 F_TranslateFileTimeBias

该函数根据指定的偏差时间偏移，将输入时间转换为另一个时区中的时间。例如，该函数可以用于将本地时间转换为 UTC 时间（世界标准时间），反之亦然。

返回值

名称	类型	描述
F_TranslateFileTimeBias	T_FILETIME [▶ 344]	

输入

```
VAR_INPUT
    in    : T_FILETIME;
    bias  : DINT;
    toUTC : BOOL;
END_VAR
```

名称	类型	描述
in	T_FILETIME	要转换的输入时间（类型：T_FILETIME [▶ 344]）。
bias	DINT	UTC 时间与本地时间之间的差值，以分钟为单位（允许使用正值或负值）。
toUTC	BOOL	该参数可以用于指定输入时间的转换方向。

toUTC	方向	系统计算公式
FALSE	UTC -> 本地时间	本地时间 := UTC - 偏差
TRUE	本地时间 -> UTC	UTC := 本地时间 + 偏差

示例:

in 变量包含要转换的时间。*bToUTC* 变量决定了转换方向。如果 *bToUTC*= TRUE，则本地时间将转换为 UTC 时间；如果 *bToUTC*= FALSE，则 UTC 时间将转换为本地时间。*WEST_EUROPE_TZI* 常量包含西欧的时区信息。根据常量中的时区信息和当前的 bDST 设置（夏令时）可计算得出所需的偏差值。或者，通过功能块可以确定 TwinCAT 系统的当前时区信息：FB_GetTimeZoneInformation [▶ 78]。

重要说明：由于在线模式下的可视化控件选项，因此为输入时间选择了数据类型 DT。不建议转换为其他时间格式，因为转换功能可能需要大量计算。

```
PROGRAM MAIN
VAR
    bDST    : BOOL := TRUE;(* TRUE => Daylight saving time, FALSE => Standard time *)
    bToUTC  : BOOL := FALSE;
(* TRUE => Convert local time to UTC time, FALSE => Convert UTC time to local time *)
    in      : DT := DT#2011-08-29-15:15:31;
    out     : DT;
    bias    : DINT;
END_VAR
```

```
IF bDST THEN
    bias := WEST_EUROPE_TZI.bias + WEST_EUROPE_TZI.daylightBias;
ELSE
    bias := WEST_EUROPE_TZI.bias + WEST_EUROPE_TZI.standardBias;
END_IF

out := FILETIME_TO_DT( F_TranslateFileTimeBias( DT_TO_FILETIME( in ), bias, bToUTC ) );
```

其他时间和时区函数和功能块：

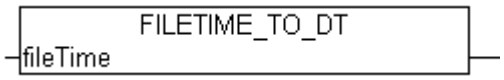
- FB_TzSpecificLocalTimeToSystemTime
- FB_TzSpecificLocalTimeToFileTime
- FB_SystemTimeToTzSpecificLocalTime

- FB_FileTimeTimeToTzSpecificLocalTime
- FB_GetTimeZoneInformation
- FB_SetTimeZoneInformation
- NT_SetLocalTime
- NT_GetTime
- NT_SetTimeToRTCTime
- FB_LocalSystemTime

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.1.3 FILETIME_TO_DT



函数“FILETIME_TO_DT”将 FILETIME 格式的时间转换为 DATE_AND_TIME 格式（DT）。与 FILETIME 格式相比，DT 格式具有较小的值范围，而且只能提供秒级精度。因此，要转换的 FILETIME 值受限。允许的最小值与值 *DT#1970-01-01-00:00:00* 对应，最大值与值 *DT#2106-02-06-06:28:15* 对应。在转换时不考虑毫秒，并将毫秒向下取整为相应的 DATE_AND_TIME 返回值。

返回值

名称	类型	描述
FILETIME_TO_DT	DT	

输入

```
VAR_INPUT
    fileTime : T_FILETIME;
END_VAR
```

名称	类型	描述
fileTime	T_FILETIME [▶ 344]	以 FILETIME 格式表示的要转换的时间（类型：T_FILETIME）。

示例：

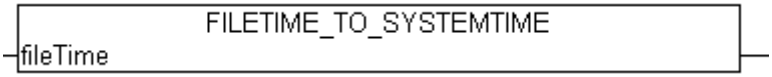
```
PROGRAM MAIN
VAR
    fbSystemTime    : GETSYSTEMTIME;
    timeAsFileTime  : T_FILETIME;
    timeAsDT        : DT;
END_VAR
```

```
fbSystemTime( timeLoDW=>timeAsFileTime.dwLowDateTime, timeHiDW=>timeAsFileTime.dwHighDateTime );
timeAsDT := FILETIME_TO_DT( timeAsFileTime );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.1.4 FILETIME_TO_SYSTEMTIME



函数“FILETIME_TO_SYSTEMTIME”将 FILETIME 格式的时间转换为“可读的”SYSTEMTIME 格式。如果设置了 64 位 fileTime 变量的最高有效位，则转换失败。在这种情况下，TIMESTRUCT 成员变量的值为零。

返回值

名称	类型	描述
FILETIME_TO_SYSTEMTIME	TIMESTRUCT [▶ 347]	

输入

```
VAR_INPUT
    fileTime : T_FILETIME;
END_VAR
```

名称	类型	描述
fileTime	T_FILETIME [▶ 344]	以 FILETIME 格式表示的要转换的时间（类型：T_FILETIME）。

示例：

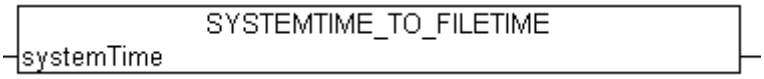
```
PROGRAM MAIN
VAR
    fbSystemTime     : GETSYSTEMTIME;
    timeAsFileTime   : T_FILETIME;
    timeAsSystemTime : TIMESTRUCT;
END_VAR
```

```
fbSystemTime( timeLoDW=>timeAsFileTime.dwLowDateTime, timeHiDW=>timeAsFileTime.dwHighDateTime );
timeAsSystemTime := FILETIME_TO_SYSTEMTIME( timeAsFileTime );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.1.5 SYSTEMTIME_TO_FILETIME



该函数可以用于将 Windows 系统时间结构转换为 FILETIME 格式。SystemTime 变量的星期几 wDayOfWeek 将被忽略。系统时间年份必须大于 1601 且 小于 30827。

返回值

名称	类型	描述
SYSTEMTIME_TO_FILETIME	T_FILETIME [▶ 344]	

输入

```
VAR_INPUT
    systemTime : TIMESTRUCT;
END_VAR
```

名称	类型	描述
systemTime	TIMESTRUCT [▶ 347]	具有要转换的 Windows 系统时间的结构（类型：TIMESTRUCT）。

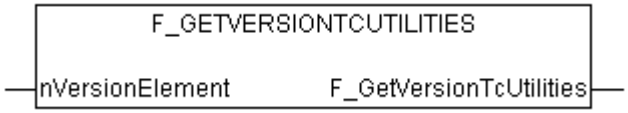
返回参数	描述
0	错误，错误的 SystemTime 参数值。
> 0	无错误。文件时间。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.2 函数 F_GetVersionTcUtilities

该函数已废弃，不应使用。请使用全局常量 stLibVersion_Tc2_Utilities [▶ 349] 从 PLC 库中读取版本信息。



该函数可以用于读取 PLC 库版本信息。

返回值

名称	类型	描述
F_GetVersionTcUtilities	UINT	

输入

```
VAR_INPUT
nVersionElement : INT;
END_VAR
```

名称	类型	描述
nVersionElement	INT	要读取的版本元素。 可能的参数： 1：主要版本号 2：次要版本号 3：修订版本号

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.3 FLOATIsFinite

过时的函数

使用 LrealIsFinite [▶ 200]() 函数代替。

返回值

名称	类型	描述
FLOATIsFinite	BOOL	

输入

```
VAR_INPUT
    x        : LREAL;
END_VAR
```

名称	类型	描述
x	LREAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.11.4 FLOATIsNaN

过时的函数

使用 LrealIsNaN [▶ 200]() 函数代替。

返回值

名称	类型	描述
FLOATIsNaN	BOOL	

输入

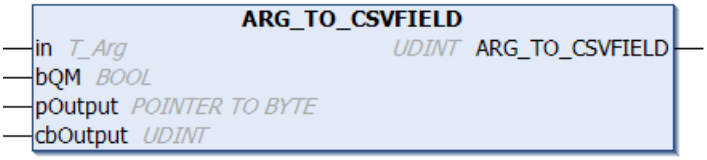
```
VAR_INPUT
    x        : LREAL;
END_VAR
```

名称	类型	描述
x	LREAL	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.12 ARG_TO_CSVFIELD



该函数将 PLC 变量的值转换为 CSV 格式的数据字段。源文件中的单引号被替换为双引号。如果设置了 bQM 参数（QM = 引号），则还会添加外层引号（将 CSV 数据字段括起来）。如果成功，则该函数会返回转换后的数据的长度作为结果。如果发生转换错误或数据丢失，则该函数会返回零。结果将被写入所提供的字节缓冲区。应用程序必须确保缓冲区足够大，以便能够容纳结果。

该函数通常与功能块 FB_CSVMemBufferWriter [▶ 50] 一起使用，在 PLC 内存中生成 CSV 格式的数据集。在下一步中，可以将内存内容写入文件。与 STRING_TO_CSVFIELD [▶ 311] 函数相比，该函数还可以用于将带有二进制数据的 PLC 变量转换为 CSV 数据字段。

返回值

名称	类型	描述
ARG_TO_CSVFIELD	UDINT	

输入

```
VAR_INPUT
    in        : T_Arg;
    bQM       : BOOL;
    pOutput   : POINTER TO BYTE;
    cbOutput  : UDINT;
END_VAR
```

名称	类型	描述
in	T_Arg	PLC 源变量，其值要转换为 CSV 格式的数据字段（类型：T_Arg [▶ 343]）。
bQM	BOOL	如果该输入为 TRUE，则会用引号将转换后的字段数据括起来。
pOutput	POINTER TO BYTE	输出缓冲区的起始地址（指针）。使用 ADR 运算符可以确定缓冲区地址。结果数据将被写入该缓冲区。
cbOutput	UDINT	输出缓冲区的最大可用大小，以字节为单位。使用 SIZEOF 运算符可以确定输出缓冲区的长度。

示例：

下面的示例说明了如何将不同类型的 PLC 变量转换为 CSV 格式，反之亦然。通过 ARG_TO_CSVFIELD 转换，将结果复制到字节缓冲区（field1..field6）中。通过 CSVFIELD_TO_ARG [▶ 272] 转换，源数据位于字节缓冲区（field1..field6）中，并将结果复制到 TwinCAT PLC 变量中。

```
PROGRAM P_ArgToConvExample
VAR
    (* PLC data to be converted to or from CSV format *)
    bOperating : BOOL := TRUE;
    fAxPos     : LREAL := 12.2;
    nCounter   : UDINT := 7;
    sName      : T_MaxString := 'Module: "XAF", $04$05, 20';
    binData    : ARRAY[0..9] OF BYTE := [0, 1, 2, 3, 4, 5, 6, 7, 8, 9];
    sShort     : STRING(10) := 'XAF';

    (* conversion buffer *)
    field1 : ARRAY[0..50 ] OF BYTE;
    field2 : ARRAY[0..50 ] OF BYTE;
    field3 : ARRAY[0..50 ] OF BYTE;
    field4 : ARRAY[0..50 ] OF BYTE;
    field5 : ARRAY[0..50 ] OF BYTE;
    field6 : ARRAY[0..50 ] OF BYTE;

    cbField1 : UDINT;
    cbField2 : UDINT;
    cbField3 : UDINT;
    cbField4 : UDINT;
    cbField5 : UDINT;
    cbField6 : UDINT;

    cbVar1 : UDINT;
    cbVar2 : UDINT;
    cbVar3 : UDINT;
    cbVar4 : UDINT;
    cbVar5 : UDINT;
    cbVar6 : UDINT;
END_VAR
```

```
cbField1 := ARG_TO_CSVFIELD( F_BOOL( bOperating ), TRUE, ADR( field1 ), SIZEOF( field1 ) );
cbField2 := ARG_TO_CSVFIELD( F_LREAL( fAxPos ), TRUE, ADR( field2 ), SIZEOF( field2 ) );
cbField3 := ARG_TO_CSVFIELD( F_UDINT( nCounter ), TRUE, ADR( field3 ), SIZEOF( field3 ) );
cbField4 := ARG_TO_CSVFIELD( F_STRING( sName ), TRUE, ADR( field4 ), SIZEOF( field4 ) );
cbField5 := ARG_TO_CSVFIELD( F_BIGTYPE( ADR( binData ), SIZEOF( binData ) ), TRUE, ADR( field5 ), SI
ZEOF( field5 ) );
cbField6 := ARG_TO_CSVFIELD( F_BIGTYPE( ADR( sShort ), LEN( sShort ) ), TRUE, ADR( field6 ), SIZEOF(
 field6 ) );

cbVar1 := CSVFIELD_TO_ARG( ADR( field1 ), cbField1, TRUE, F_BOOL( bOperating ) );
cbVar2 := CSVFIELD_TO_ARG( ADR( field2 ), cbField2, TRUE, F_LREAL( fAxPos ) );
```

```
cbVar3 := CSVFIELD_TO_ARG( ADR( field3 ), cbField3, TRUE, F_UDINT( nCounter ) );
cbVar4 := CSVFIELD_TO_ARG( ADR( field4 ), cbField4, TRUE, F_STRING( sName ) );
cbVar5 := CSVFIELD_TO_ARG( ADR( field5 ), cbField5, TRUE, F_BIGTYPE( ADR( binData ), SIZEOF( binData
 ) ) );
cbVar6 := CSVFIELD_TO_ARG( ADR( field6 ), cbField6, TRUE, F_BIGTYPE( ADR( sShort ), LEN( sShort ) )
);
```

结果（十六进制字符串形式的字节缓冲区）：

cbField1 = 3，field1 = '22 01 22'

cbField2 = 10，field2 = '22 66 66 66 66 66 66 28 40 22'

cbField3 = 6，field3 = '22 07 00 00 00 22'

cbField4 = 25，field4 = '22 4D 6F 64 75 6C 65 3A 20 22 22 58 41 46 22 22 2C 20 04 05 2C 20 32 30 22'

cbField5 = 12，field5 = '22 00 01 02 03 04 05 06 07 08 09 22'

cbField6 = 5，field6 = '22 58 41 46 22'

cbVar1 = 1

cbVar2 = 8

cbVar3 = 4

cbVar4 = 22

cbVar5 = 10

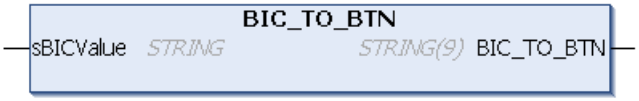
cbVar6 = 3

有关更多信息，请访问：示例：写入/读取 CSV 文件 [▶ 369]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.13 BIC_TO_BTN



函数 BIC_TO_BTN 从 sBICValue 中的倍福识别码中提取倍福可追溯性编号（BTN），并将其作为返回值返回。BTN 末尾的空格会被自动移除。如果没有找到 BTN，则会返回一个空字符串。

返回值

名称	类型	描述
BIC_TO_BTN	STRING(9)	

VAR_INPUT

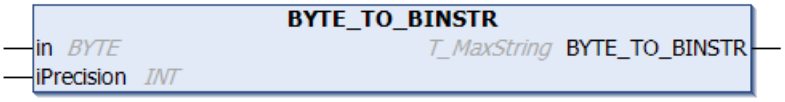
```
VAR_INPUT
    sBICValue    : STRING;
END_VAR
```

名称	类型	描述
sBICValue	**STRING**	该输入必须包含倍福识别码（BIC），函数 BIC_TO_BTN 可从中提取倍福可追溯性编号（BTN），例如，对于 BIC“1P193995SBTN0002agdw1KEL7411 Q1 2P112104020018”，BIC_TO_BTN 可返回值“0002agdw”。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.14 BYTE_TO_BINSTR



该函数将十进制数转换为二进制字符串（以 2 为基数）。

返回值

名称	类型	描述
BYTE_TO_BINSTR	T_MaxString	

输入

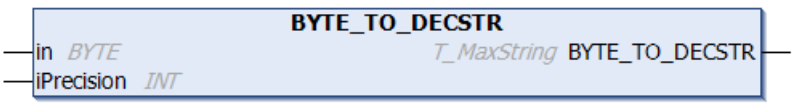
```
VAR_INPUT
    in         : BYTE;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	BYTE	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.15 BYTE_TO_DECSTR



该函数将十进制数转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
BYTE_TO_DECSTR	T_MaxString	

输入

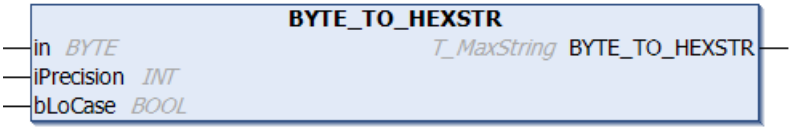
```
VAR_INPUT
    in         : BYTE;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	BYTE	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.16 BYTE_TO_HEXSTR



该函数将十进制数转换为十六进制字符串（以 16 为基数）。

返回值

名称	类型	描述
BYTE_TO_HEXSTR	T_MaxString	

输入

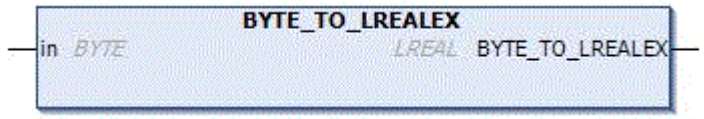
```
VAR_INPUT
    in         : BYTE;
    iPrecision : INT;
    bLoCase    : BOOL := FALSE;
END_VAR
```

名称	类型	描述
in	BYTE	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。
bLoCase	BOOL	该参数确定在转换时使用小写字母还是大写字母。FALSE => "ABCDEF"，TRUE => "abcdef"。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.17 BYTE_TO_LREALEX



Arm® 平台上的 TwinCAT 2 不支持将无符号数字转换为 LREAL 类型的浮点数。已设置最高有效位的无符号数字可能会被隐式转换为负浮点数。在这里描述的函数允许在 TwinCAT 2 中将 BYTE 类型显式转换为 LREAL 类型的正浮点数（即使已设置最高有效位且没有编译器警告）。您只需该函数即可编译转换后的 TwinCAT 2 项目，而无需在 TwinCAT 3 中进行更改。

在 TwinCAT 3 中，BYTE 类型的无符号数字始终会（隐式和显式）转换为正浮点数。因此，可以省去该函数。

返回值

名称	类型	描述
BYTE_TO_LREALEX	LREAL	

输入

```
VAR_INPUT
    in : BYTE;
END_VAR
```

名称	类型	描述
in	BYTE	

示例：

```
PROGRAM MAIN
VAR
    nByte  : BYTE := 16#FF;
    fLreal : LREAL := 0.0;
END_VAR
```

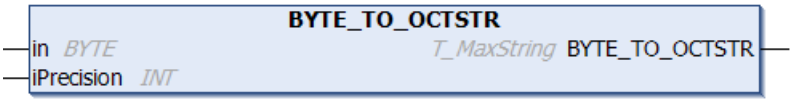
fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := nByte	+255, Warning 1105*	+255	+255
fLreal := BYTE_TO_LREAL(nByte)	+255, Warning 1105*	+255	+255
fLreal := BYTE#16#FF	+255, Warning 1105*	+255	+255
fLreal := 16#FF	+255	+255	+255
fLreal := BYTE_TO_LREALEX(nByte)	+255	+255	+255
fLreal := BYTE_TO_LREALEX(BYTE#16#FF)	+255	+255	+255
fLreal := BYTE_TO_LREALEX(16#FF)	+255	+255	+255

*不支持将无符号整数转换为 LREAL。相反，该值被用作有符号数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.18 BYTE_TO_OCTSTR



该函数将十进制数转换为八进制字符串（以 8 为基数）。

返回值

名称	类型	描述
BYTE_TO_OCTSTR	T_MaxString	

输入

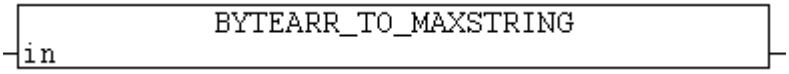
```
VAR_INPUT
    in         : BYTE;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	BYTE	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.19 BYTEARR_TO_MAXSTRING



将字节数组的各个 ASCII 代码转换为字符串。

返回值

名称	类型	描述
BYTEARR_TO_MAXSTRING	T_MaxString	

输入

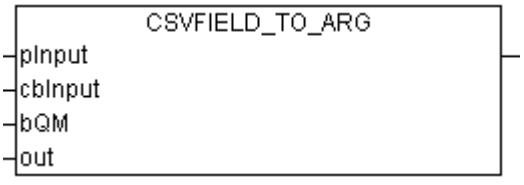
```
VAR_INPUT
    in : ARRAY[0..MAX_STRING_LENGTH] OF BYTE;
END_VAR
```

名称	类型	描述
in	ARRAY[0..MAX_STRING_LENGTH] OF BYTE	字节数组变量（MAX_STRING_LENGTH 默认值：255）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.20 CSVFIELD_TO_ARG



该函数将以字节缓冲区的形式存在的 CSV 格式的数据字段中的值转换为 PLC 变量。数据字段中的双引号被替换为单引号。如果设置了 bQM 参数（QM = 引号），则会从输入数据中移除外层引号（将数据字段括起来）。如果成功，则该函数会返回转换后的数据的长度。如果出现错误或输入数据的长度为零，则该函数会返回零值。应用程序必须确保 PLC 目标变量足够大，以便能够容纳该值。

该函数通常与功能块 FB_CSVMemBufferReader [▶ 48] 一起使用，以读取（解析）以 CSV 格式存储在 PLC 内存中的数据集。在执行此操作前，通常会将 CSV 数据集从文件读取到 PLC 内存中。与 CSVFIELD_TO_STRING [▶ 273] 函数相比，该函数还可以用于将带有二进制数据的 CSV 数据字段转换为 PLC 变量。

返回值

名称	类型	描述
CSVFIELD_TO_ARG	UDINT	

输入

```
VAR_INPUT
    pInput    : POINTER TO BYTE;
    cbInput   : UDINT;
    bQM       : BOOL;
    out       : T_Arg;
END_VAR
```

名称	类型	描述
pInput	POINTER TO BYTE	包含要转换为 CSV 格式的数据字段的字节缓冲区的起始地址（指针）。使用 ADR 运算符可以确定地址。
cbInput	UDINT	要转换的数据字段的长度，以字节为单位。使用 SIZEOF 运算符可以确定长度。
bQM	BOOL	如果该输入为 TRUE，则会将括起来的引号从字段数据中移除。
out	T_Arg	要将数据字段的值写入其中的 PLC 目标变量（类型：T_Arg [▶ 343]）。

示例：

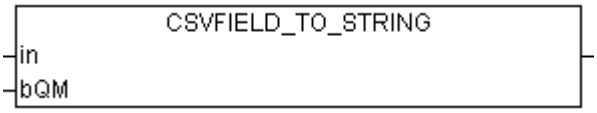
请参见 ARG_TO_CSVFIELD [▶ 266] 功能块文档中的示例。

有关更多信息，请访问：示例：写入/读取 CSV 文件 [▶ 369]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.21 CSVFIELD_TO_STRING



该函数将以源字符串的形式存在的 CSV 数据字段格式的数据字段转换为 PLC 字符串格式的值。源字符串中的双引号被替换为单引号。如果设置了 bQM 参数（QM = 引号），则会从源字符串中移除外层引号（将数据字段括起来）。如果成功，则该函数会返回转换后的字符串作为结果。如果在转换期间发生错误，则该函数会返回一个空字符串，但前提是源字符串不是空字符串。

该函数通常与功能块 FB_CSVMemBufferReader [▶ 48] 一起使用，以读取（解释）以 CSV 格式存储在 PLC 内存中的数据集。在执行此操作前，通常会将 CSV 数据集从文件读取到 PLC 内存中。源字符串不得包含二进制数据。值为零的二进制数据会在错误的位置终止并截断字符串。要转换带有二进制数据的数据字段，请使用函数 CSVFIELD_TO_ARG [▶ 272]。

返回值

名称	类型	描述
CSFIELD_TO_STRING	T_MaxString	

输入

```
VAR_INPUT
    in  : T_MaxString;
    bQM : BOOL;
END_VAR
```

名称	类型	描述
in	T_MaxString	带有 CSV 格式的数据字段的源字符串，要将其转换为 PLC 字符串格式的值（类型：T_MaxString）。
bQM	BOOL	如果该输入为 TRUE，则会将括起来的引号从源字符串中移除。

bQM	描述	源字符串	结果字符串	符合 CSV 标准
FALSE	没有外围引号的源字符串应仅包含字母和数字。在这种情况下，源字符串不得包含任何不可打印的控制字符、引号、分号、逗号（US CSV 格式）或二进制数据。	'Module_XA5' '123456' ''	'Module_XA5' '123456' ''	是 是 是
		'A"""B'	'A""B'	否
		'A""B'	'A"B'	否
		';'	';'	否
		'RN'	'RN'	否
		'AB$00CD'	'AB'（字符串被截断）	否
TRUE	没有用引号括起来的源字符串不应包含任何不可打印的控制字符、引号、分号或逗号（美国 CSV 格式）。不允许使用二进制数据。	'"Module_XA5"'	'Module_XA5'	是
	'"123456"'	'123456'	是	
	'""'	''	是	
	'"A""B"'	'A""B'	是	
	'"A""B"'	'A"B'	是	
	'";"'	';'	是	
	'"RN"'	'RN'	是	
	'"AB$00CD"'	'AB'（字符串被截断）	否	

示例：

```
PROGRAM MAIN
VAR
    s1 : STRING;
    s2 : STRING;
END_VAR
```

```
s1 := CSVFIELD_TO_STRING( '"ab_$04_$05_cd-""ALFA""_5"', TRUE );
s2 := CSVFIELD_TO_STRING( 'Module_50', FALSE );
```

结果：

s1 = 'ab_$04_$05_cd-"ALFA"_5'

s2 = 'Module_50'

有关更多信息，请访问：示例：写入/读取 CSV 文件 [▶ 369]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.22 DATA_TO_HEXSTR

该函数将二进制数据转换为十六进制字符串。该函数可以用于转换简单的数据类型和结构变量。二进制数据的最大长度不得超过 85 字节。如果超出最大长度，则会在结果字符串中添加一个点（“.”），并终止转换。剩余的数据字节不会进行转换。如果函数参数错误（*pData*= 零或 *cbData* = 零），则该函数会返回一个空字符串。

返回值

名称	类型	描述
DATA_TO_HEXSTR	T_MaxString	

输入

```
VAR_INPUT
    pData   : POINTER TO BYTE;
    cbData  : UDINT(0..85);
    bLoCase : BOOL := FALSE;
END_VAR
```

名称	类型	描述
pData	POINTER TO BYTE	要转换的二进制数据的起始地址（指针）。使用 ADR 运算符可以确定地址。
cbData	UDINT(0..85)	要转换的二进制数据的最大长度。长度不可以超过 85 字节。使用 SIZEOF 运算符可以确定长度。
bLoCase	BOOL	该参数指定在转换时使用大写还是小写。TRUE = 小写字母，FALSE = 大写字母。

示例：

请确保*溢出*变量的数据大小不超过 85 字节。因此，在结果字符串 *sH5* 中添加了一个点。

*数字*变量中的字节顺序发生互换，因为计数器变量的内存组织基于小端格式（也被称为 Intel 格式）。

```
PROGRAM MAIN
VAR
    str    : T_MaxString := 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789';
    number : DWORD := 16#BECF1234;
    char   : BYTE := 16#07;
    null   : UDINT := 0;

    overflow   : ARRAY[0..86] OF BYTE; (* data overflow *)
    cbOverflow : UDINT;

    sH1, sH2, sH3, sH4, sH5 : T_MaxString;
END_VAR
```

```
sH1 := DATA_TO_HEXSTR( pData := ADR(str), cbData := LEN(str), FALSE );
sH2 := DATA_TO_HEXSTR( pData := ADR(number), cbData := SIZEOF(number), FALSE );
sH3 := DATA_TO_HEXSTR( pData := ADR(char), cbData := SIZEOF(char), FALSE );
sH4 := DATA_TO_HEXSTR( pData := ADR(null), cbData := SIZEOF(null), FALSE );
cbOverflow:= SIZEOF(overflow);
sH5 := DATA_TO_HEXSTR( pData := ADR(overflow), cbData := cbOverflow, FALSE );
```

结果：

sH1 = '61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 30 31 32 33 34 35 36 37 38 39'

sH2 = '34 12 CF BE'

sH3 = '07'

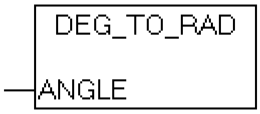
sH4 = '00 00 00 00'

sH5 = '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00.'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.23 DEG_TO_RAD



该函数将度角转换为弧度。

返回值

名称	类型	描述
DEG_TO_RAD	LREAL	

输入

```
VAR_INPUT
    ANGLE   : LREAL;
END_VAR
```

名称	类型	描述
ANGLE	LREAL	要转换的角，以度为单位。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.24 DINT_TO_DECSTR

该函数将有符号的十进制数转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
DINT_TO_DECSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : DINT;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	DINT	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。对于负数，负号将出现在结果字符串中。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    iCnt : INT;
END_VAR
```

```
iCnt := -1234;
s1   := DINT_TO_DECSTR( iCnt, 1);
s2   := DINT_TO_DECSTR( iCnt, 10 );
iCnt := 0;
s3   := DINT_TO_DECSTR( iCnt, 0 );
iCnt := 1234;
s4   := DINT_TO_DECSTR( iCnt, 10 );
```

结果：

s1 = '-1234

s2 = '-0000001234'

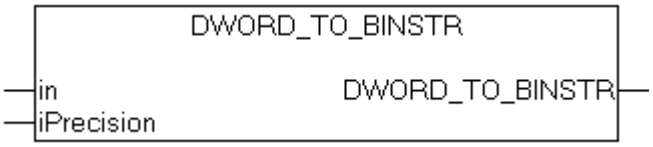
s3 = ''

s4 = '0000001234'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.25 DWORD_TO_BINSTR



该函数将十进制数转换为二进制字符串（以 2 为基数）。

返回值

名称	类型	描述
DWORD_TO_BINSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : DWORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	DWORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1    : STRING;
    s2    : STRING;
    s3    : STRING;
    nCnt  : BYTE;
END_VAR
```

```
s1   := DWORD_TO_BINSTR( 16#81, 16 );
nCnt := 15;
s2   := DWORD_TO_BINSTR( nCnt, 1 );
nCnt := 0;
s3   := DWORD_TO_BINSTR( nCnt, 0 );
```

结果：

s1 = '0000000010000001'

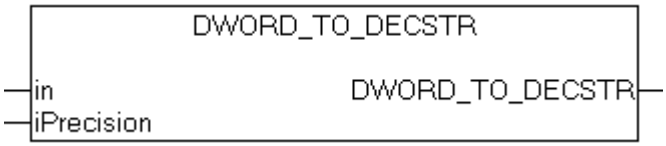
s2 = '1111'

s3 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.26 DWORD_TO_DECSTR



该函数将十进制数转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
DWORD_TO_DECSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : DWORD;
    iPrecision  : INT;
END_VAR
```

名称	类型	描述
in	DWORD	要转换的十进制数。

名称	类型	描述
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    nCnt : WORD;
END_VAR
```

```
nCnt := 43981;
s1   := DWORD_TO_DECSTR( nCnt, 1 );
s2   := DWORD_TO_DECSTR( nCnt, 10 );
nCnt := 0;
s3   := DWORD_TO_DECSTR( nCnt, 0 );
```

结果：

s1 = '43981'

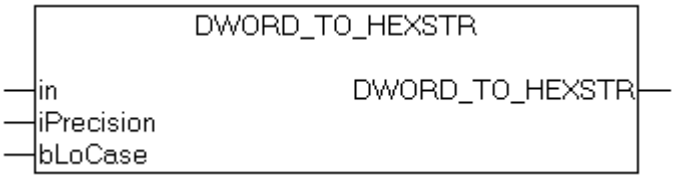
s2 = '0000043981'

s3 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.27 DWORD_TO_HEXSTR



该函数将十进制数转换为十六进制字符串（以 16 为基数）。

返回值

名称	类型	描述
DWORD_TO_HEXSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : DWORD;
    iPrecision : INT;
    bLoCase    : BOOL;
END_VAR
```

名称	类型	描述
in	DWORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。
bLoCase	BOOL	该参数确定在转换时使用小写字母还是大写字母。FALSE => "ABCDEF"，TRUE => "abcdef"。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    nCnt : WORD;
END_VAR
```

```
nCnt := 43981;
s1   := DWORD_TO_HEXSTR( nCnt, 1, FALSE );
s2   := DWORD_TO_HEXSTR( nCnt, 1, TRUE );
nCnt := 15;
s3   := DWORD_TO_HEXSTR( nCnt, 4, FALSE );
nCnt := 0;
s4   := DWORD_TO_HEXSTR( nCnt, 0, FALSE );
```

结果：

s1 = 'ABCD'

s2 = 'abcd'

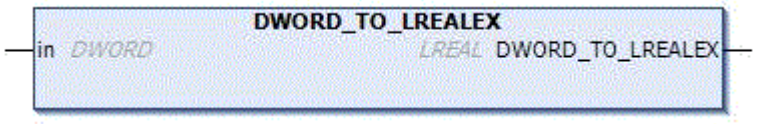
s3 = '000F'

s4 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.28 DWORD_TO_LREALEX



Arm® 平台上的 TwinCAT 2 不支持将无符号数字转换为 LREAL 类型的浮点数。已设置最高有效位的无符号数字可能会被隐式转换为负浮点数。在这里描述的函数允许在 TwinCAT 2 中将 DWORD 类型显式转换为 LREAL 类型的正浮点数（即使已设置最高有效位且没有编译器警告）。您只需该函数即可编译转换后的 TwinCAT 2 项目，而无需在 TwinCAT 3 中进行更改。

在 TwinCAT 3 中，DWORD 类型的无符号数字始终会（隐式和显式）转换为正浮点数。因此，这个函数可以被省略。

返回值

名称	类型	描述
DWORD_TO_LREALEX	LREAL	

输入

```
VAR_INPUT
    in : DWORD;
END_VAR
```

名称	类型	描述
in	DWORD	

示例：

```
PROGRAM MAIN
VAR
    nDword : DWORD := 16#FFFFFFFF;
    fLreal : LREAL := 0.0;
END_VAR
```

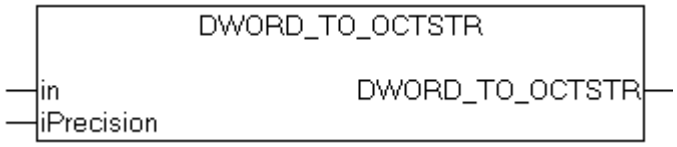
fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := nDword	-*1, Warning 1105**	4294967295	4294967295
fLreal := DWORD_TO_LREAL(nDword)	-*1, Warning 1105**	4294967295	4294967295
fLreal := DWORD#16#FFFFFFFF	-*1, Warning 1105**	4294967295	4294967295
fLreal := 16#FFFFFFFF	4294967295	4294967295	4294967295
fLreal := DWORD_TO_LREALEX(nDword)	4294967295	4294967295	4294967295
fLreal := DWORD_TO_LREALEX(DWORD#16#FFFFFFFF)	4294967295	4294967295	4294967295
fLreal := DWORD_TO_LREALEX(16#FFFFFFFF)	4294967295	4294967295	4294967295

*不支持
**不支持将无符号整数转换为 LREAL。相反，该值被用作有符号数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.29 DWORD_TO_OCTSTR



该函数将十进制数转换为八进制字符串（以 8 为基数）。

返回值

名称	类型	描述
DWORD_TO_OCTSTR	T_MaxString	

输入

```
VAR_INPUT
    in          : DWORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	DWORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1    : STRING;
    s2    : STRING;
```

```
    s3   : STRING;
    nCnt : WORD;
END_VAR
```

```
nCnt := 43981;
s1   := DWORD_TO_OCTSTR( nCnt, 1 );
s2   := DWORD_TO_OCTSTR( nCnt, 10 );
nCnt := 0;
s3   := DWORD_TO_OCTSTR( nCnt, 0 );
```

结果：

s1 = '125715'

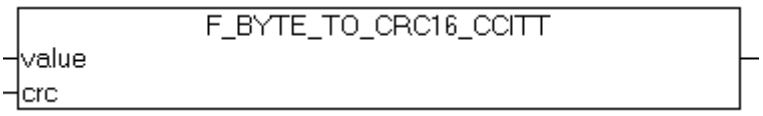
s2 = '0000125715'

s3 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.30 F_BYTE_TO_CRC16_CCITT



函数“F_BYTE_TO_CRC16_CCITT”可以用于确定单个数据字节的 16 位 CRC CCITT（循环冗余检查）。

使用的生成器多项式：名称：CRC-16 CCITT

- 默认值：CRC-CCITT
- 引用：ITU X.25/T.30、ADCCP、SDLC/HDLC、……
- 多项式值：0x1021
- 多项式：x^16 + x^12 + x^5 + 1

返回值

名称	类型	描述
F_BYTE_TO_CRC16_CCITT	WORD	

输入

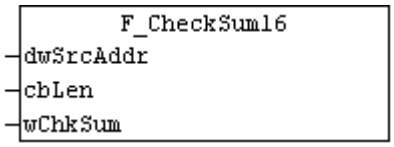
```
VAR_INPUT
    value   : BYTE;(* Data value *)
    crc     : WORD;(* Initial value (16#FFFF or 16#0000) or previous CRC-16 result *)
END_VAR
```

名称	类型	描述
value	BYTE	要转换的数据字节。
crc	WORD	初始值 = 16#FFFF 或 16#0000 或最后一次 CRC。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.31 F_CheckSum16



函数“F_CheckSum16”可以用于确定任何数据的 16 位校验和。

返回值

名称	类型	描述
F_CheckSum16	WORD	

输入

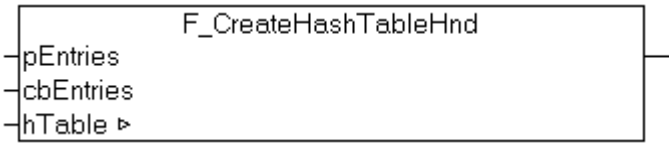
```
VAR_INPUT
    dwSrcAddr   : POINTER TO BYTE;
    cbLen       : UDINT;
    wChkSum     : WORD;
END_VAR
```

名称	类型	描述
dwSrcAddr	POINTER TO BYTE	数据缓冲区的地址。
cbLen	UDINT	数据缓冲区的长度。
wChkSum	WORD	：初始值 = 0 或最后一次校验和。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.32 F_CreateHashTableHnd



该函数对哈希表句柄进行初始化。通过调用 F_CreateHashTableHnd 函数，必须对表句柄进行一次初始化。

返回值

名称	类型	描述
F_CreateHashTableHnd	BOOL	

输入

```
VAR_INPUT
    pEntries    : POINTER TO T_HashTableEntry := 0;
    cbEntries   : UDINT := 0;
END_VAR
```

名称	类型	描述
pEntries	POINTER TO T_HashTableEntry	第一个 T_HashTableEntry 数组元素的地址。使用 ADR 运算符可以确定地址（类型：T_HashTableEntry [▶ 345]）。

名称	类型	描述
cbEntries	UDINT	T_HashTableEntry 字节大小。使用 SIZEOF 运算符可以确定字节大小。

/ 输入/输出

```
VAR_IN_OUT
    hTable : T_HHASHTABLE;
END_VAR
```

名称	类型	描述
hTable	T_HHASHTABLE [▶ 345]	要初始化的哈希表句柄。从功能块 FB_HashTableCtrl [▶ 80] 访问哈希表时需要该句柄。

返回参数	描述
TRUE	无错误
FALSE	错误

示例：

请参见：示例：哈希表（FB_HashTableCtrl）。 [▶ 361]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.33 F_CreateLinkedListHnd

该函数对链表句柄进行初始化。通过调用 F_CreateLinkedListHnd 函数，必须对列表句柄进行一次初始化。

返回值

名称	类型	描述
F_CreateLinkedListHnd	BOOL	

输入

```
VAR_INPUT
    pEntries  : POINTER TO T_LinkedListEntry := 0;
    cbEntries : UDINT := 0;
END_VAR
```

名称	类型	描述
pEntries	POINTER TO T_LinkedListEntry	第一个 T_LinkedListEntry 数组元素的地址。使用 ADR 运算符可以确定地址（类型：T_LinkedListEntry [▶ 347]）。
cbEntries	UDINT	T_LinkedListEntry 数组的字节大小。使用 SIZEOF 运算符可以确定字节大小。

/ 输入/输出

```
VAR_IN_OUT
    hList  : T_HLINKEDLIST;
END_VAR
```

名称	类型	描述
hList	T_HLINKEDLIST [▶ 346]	要初始化的哈希表句柄（类型：T_HLINKEDLIST）。从功能块 FB_LinkedListCtrl [▶ 88] 访问列表时需要该句柄。

返回参数	描述
TRUE	无错误
FALSE	错误

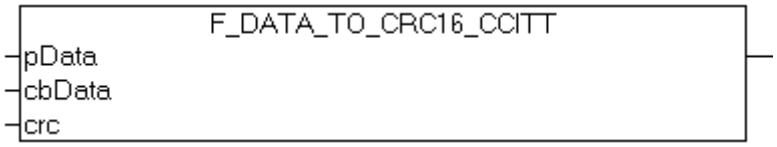
示例：

请参见：示例：链表（FB_LinkedListCtrl）。 [▶ 364]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.34 F_DATA_TO_CRC16_CCITT



函数“F_DATA_TO_CRC16_CCITT”可以用于确定任何数据的 16 位 CRC CCITT（循环冗余检查）。在内部使用函数 F_BYTE_TO_CRC16_CCITT [▶ 282]。

有关所用算法的更多信息，请参见 F_BYTE_TO_CRC16_CCITT [▶ 282] 函数的文档。

返回值

名称	类型	描述
F_DATA_TO_CRC16_CCITT	WORD	

输入

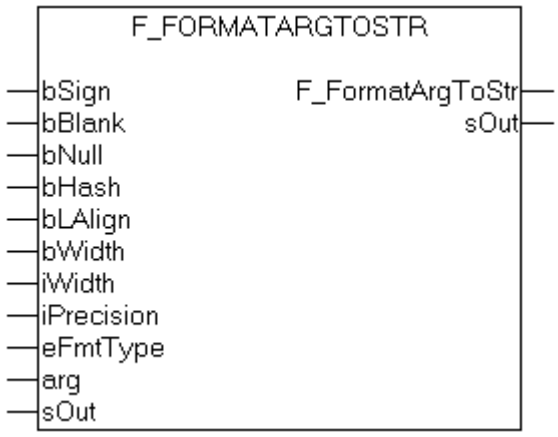
```
VAR_INPUT
    pData  : POINTER TO BYTE;(* Pointer to first data byte *)
    cbData : UDINT;(* Length of data *)
    crc    : WORD;(* Initial value (16#FFFF or 16#0000) or previous CRC-16 result *)
END_VAR
```

名称	类型	描述
pData	POINTER TO BYTE	数据缓冲区的地址。
cbData	UDINT	数据缓冲区的长度。
crc	WORD	初始值 = 16#FFFF 或 16#0000 或最后一次 CRC。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.35 F_FormatArgToStr



格式化辅助函数。该函数由 FB_FormatString [▶ 63] 功能块在内部使用。该函数可以用于按照格式规格 [▶ 373]将 T_Arg [▶ 343] 类型的变量转换为格式化的字符串。

返回值

名称	类型	描述
F_FormatArgToStr	UDINT	

输入

```
VAR_INPUT
   bSign      : BOOL;(* Sign prefix flag *)
   bBlank     : BOOL;(* Blank prefix flag *)
   bNull      : BOOL;(* Null prefix flag *)
   bHash      : BOOL;(* Hash prefix flag *)
   bLAlign    : BOOL;(* FALSE => Right align (default), TRUE => Left align *)
   bWidth     : BOOL;(* FALSE => no width padding, TRUE => blank or zeros padding enabled *)
   iWidth     : INT;(* Width length parameter *)
   iPrecision : INT;(* Precision length parameter *)
   eFmtType   : E_TypeFieldParam;    (* Format type field parameter *)
   arg        : T_Arg;(* Format argument *)
END_VAR
```

名称	类型	描述
bSign	BOOL	符号标志。
bBlank	BOOL	空白标志。
bNull	BOOL	空标志。
bHash	BOOL	哈希前缀标志。
bLAlign	BOOL	对齐标志（TRUE=左对齐）。
bWidth	BOOL	如果为 TRUE，则会评估 iWidth 参数，否则不会评估。
iWidth	INT	宽度参数。
iPrecision	INT	：精度参数。
eFmtType	E_TypeFieldParam	类型参数（类型：E_TypeFieldParam [▶ 330]）。
arg	T_Arg	要格式化的参数。以下辅助函数可以用于将不同类型的 PLC 变量转换为所需的数据类型 T_Arg [▶ 343]：F_BYTE [▶ 249]、F_WORD [▶ 259]、F_DWORD [▶ 250]、F_LWORD [▶ 253]、F_SINT [▶ 254]、F_INT [▶ 251]、F_DINT [▶ 250]、F_LINT [▶ 252]、F_USINT [▶ 258]、F_UINT [▶ 257]、F_UDINT [▶ 256]、F_ULINT [▶ 258]、F_STRING [▶ 255]、F_REAL [▶ 254]、F_LREAL [▶ 253]。

/ 输入/输出

```
VAR_IN_OUT
   sOut     : T_MaxString;
END_VAR
```

名称	类型	描述
sOut	T_MaxString	如果成功，则该变量将返回格式化的输出字符串（类型：T_MaxString）。

返回参数	含义
0	无错误
<> 0	错误。有关错误描述，请参见：格式错误代码 [▶ 375]

示例：

将 BYTE 变量格式化为二进制字符串。

```
PROGRAM MAIN
VAR
    s1      : T_MaxString;
    s2      : T_MaxString;
    s3      : T_MaxString;
    s4      : T_MaxString;
    s5      : T_MaxString;
    errID   : UDINT;
    varByte : BYTE;
    double  : LREAL;
    L1      : INT;
    L2      : INT;
    L3      : INT;
    L4      : INT;
    L5      : INT;
END_VAR
```

```
varByte := 128;
errID   := F_FormatArgToStr(FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, 20, 8, TYPEFIELD_B, F_BYTE( va
rByte ), s1 );
errID   := F_FormatArgToStr(FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, 20, 8, TYPEFIELD_B, F_BYTE( var
Byte ), s2 );
errID   := F_FormatArgToStr(FALSE, FALSE, FALSE, FALSE, TRUE, TRUE, 20, 8, TYPEFIELD_B, F_BYTE( varB
yte ), s3 );
errID   := F_FormatArgToStr(FALSE, FALSE, FALSE, TRUE, TRUE, TRUE, 20, 8, TYPEFIELD_B, F_BYTE( varBy
te ), s4 );
L1 := LEN( s1 );
L2 := LEN( s2 );
L3 := LEN( s3 );
L4 := LEN( s4 );
```

结果：

s1 = '10000000'
s2 = ' 10000000'
s3 = '10000000 '
s4 = '2#10000000 '
L1 = 8
L2 = 20
L3 = 20
L4 = 20

对 LREAL 变量进行格式化。

```
double := 12345.6789;
errID  := F_FormatArgToStr( FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, 20, 8, TYPEFIELD_F, F_LREAL( d
ouble ), s1 );
errID  := F_FormatArgToStr( FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, 20, 8, TYPEFIELD_F, F_LREAL( do
uble ), s2 );
errID  := F_FormatArgToStr( FALSE, FALSE, FALSE, FALSE, TRUE, TRUE, 20, 8, TYPEFIELD_F, F_LREAL( dou
ble ), s3 );
errID  := F_FormatArgToStr( FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, 20, 8, TYPEFIELD_F, F_LREAL( dou
ble ), s4 );
errID  := F_FormatArgToStr( TRUE, FALSE, FALSE, TRUE, TRUE, TRUE, 20, 8, TYPEFIELD_F, F_LREAL( doubl
e ), s5 );
L1 := LEN( s1 );
L2 := LEN( s2 );
L3 := LEN( s3 );
L4 := LEN( s4 );
L5 := LEN( s5 );
```

结果：

s1 = '12345.67890000'
s2 = ' 12345.67890000'
s3 = '12345.67890000 '
s4 = '00000012345.67890000'

s5 = '+12345.67890000 '
L1 = 14
L2 = 20
L3 = 20
L4 = 20
L5 = 20

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.36 F_GenerateHashValue

该函数可以用于计算哈希值。

如果计算成功，则该函数会返回 TRUE。

返回值

名称	类型	描述
F_GenerateHashValue	BOOL	

输入

```
VAR_INPUT
    hashMode : E_HashMode;
    pData    : PVOID;
    nData    : UDINT;
    pHash    : PVOID;    // destination buffer for generated hash value
    nHash    : UDINT;    // size of destination buffer in bytes. This needs to match the hash mode.
END_VAR
```

名称	类型	描述
hashMode	E_HashMode	在这里指定一种哈希模式，例如 SHA 512。请参见 E_HashMode [▶ 325]。
pData	PVOID	在这里指定输入数据的地址。
nData	UDINT	在这里指定输入数据的大小，以字节为单位。
pHash	PVOID	在这里指定要存储哈希值的缓冲区的地址。
nHash	UDINT	在这里指定哈希值的缓冲区的大小，以字节为单位。大小取决于哈希模式，另请参见 E_HashMode。必须指定适当的大小。否则，该函数将会失败。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.29	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.51.0

4.37 F_GetClassIdVersioned

该函数根据类 ID 和库 ID 计算版本类 ID。这可用于版本控制的 C++ 项目。

返回值

名称	类型	描述
F_GetClassIdVersioned	BOOL	

输入

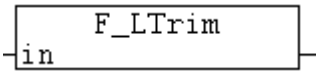
```
VAR_INPUT
    sLibraryId     : STRING(255); // 'vendorName|libraryName|libraryVersion' (e.g. 'C+
+ Module Vendor|IncrementerCpp|0.0.0.1' )
    clsId          : CLSID;
    clsIdVersioned : REFERENCE TO CLSID;
END_VAR
```

名称	类型	描述
sLibraryId	STRING(255)	在这里指定库 ID。
clsId	CLSID	在这里指定类 ID。
clsIdVersioned	REFERENCE TO CLSID	在这里输出所计算出的版本类 ID。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.29	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.51.0

4.38 F_LTrim



从字符串中移除前导空格，并返回缩减后的字符串。

返回值

名称	类型	描述
F_LTrim	T_MaxString	

输入

```
VAR_INPUT
    in  : T_MaxString;
END_VAR
```

名称	类型	描述
in	T_MaxString	要转换的字符串（类型：T_MaxString）。

示例:

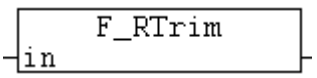
```
PROGRAM MAIN
VAR
    sLTrim : STRING;
END_VAR
```

```
sLTrim := F_LTrim(' <trim ');(* result: '<trim ' *)
sLTrim := F_LTrim(' <trim');(* result: '<trim' *)
sLTrim := F_LTrim('<trim');(* result: '<trim' *)
sLTrim := F_LTrim('');(* result: '' *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.39 F_RTrim



删除指定值的所有尾部空格，并返回结果。

返回值

名称	类型	描述
F_RTrim	T_MaxString	

输入

```
VAR_INPUT
    in : T_MaxString;
END_VAR
```

名称	类型	描述
in	T_MaxString	要转换的字符串（类型：T_MaxString）。

示例：

```
PROGRAM MAIN
VAR
    sRTrim  : STRING;
    sLRTrim : STRING;
END_VAR
```

```
sRTrim := F_RTrim(' trim> ');(* result: ' trim>' *)
sRTrim := F_RTrim('trim> ');(* result: 'trim>' *)
sRTrim := F_RTrim('trim>');(* result: 'trim>' *)
sRTrim := F_RTrim('');(* result: '' *)

sLRTrim := F_RTrim( F_LTrim( ' <trim> '));(* result: '<trim>' *)
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.40 F_SplitBIC

F_SplitBIC 函数使用已知的标识符将倍福识别码（BIC）sBICValue 拆分成不同的部分，并在 ST_SplittedBIC [▶ 339] 结构体中将已识别的子字符串作为返回值返回。子字符串末尾的空格会被自动移除。未使用的子字符串将作为空字符串返回。如果发现未知的标识符，则会在返回结构中将 BIC 的剩余字符串作为 sUndefined 发送。例如，使用功能块 FB_EcCoEReadBIC 或 FB_EcReadBIC（Tc2_EtherCAT 库）可以从 EtherCAT 从站读取 BIC。

返回值

名称	类型	描述
F_SplitBIC	ST_SplittedBIC	

输入

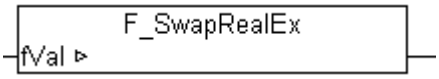
```
VAR_INPUT
    sBICValue   : STRING;
END_VAR
```

名称	类型	描述
sBICValue	**STRING**	该输入必须包含倍福识别码（BIC），使用 F_SplitBIC 函数可将其拆分成子字符串，例如“1P193995SBTN0002agdw1KEL7411 Q1 2P112104020018”。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.41 F_SwapRealEx



总线端子模块控制器上的 REAL 数字的内存表示法（例如 BC2000、BC3100、BC9000）与 x86/x64/Arm® 系统（IPC 或嵌入式控制器）上的 REAL 数字的内存表示法不同。
要在 IPC 上正确表示总线端子模块控制器的 REAL 数字，必须交换 REAL 数字的高位字和低位字。在编程环境下，该操作已经在在线或仿真模式下完成。为了通过网络（ADS 协议、ADSDLL、AdsOcx 等）向总线控制器请求 REAL 数据并在 x86/x64/Arm® IPC 上正确表示它，必须将 REAL 数据转换为正确的格式。该操作可以在总线端子模块控制器端或 IPC 端完成。

函数 F_SwapRealEx 可以用于将 REAL 变量（例如，VB 应用程序要读取的变量或使用 TwinCAT Scope View 记录的变量）转换为 PC 端的合适格式。该函数可更改传输的 *fVal* 参数的内存表示法（VAR_IN_OUT）。

返回值

名称	类型	描述
F_SwapRealEx	BOOL	

/ 输入/输出

```
VAR_IN_OUT
    fVal : REAL;
END_VAR
```

名称	类型	描述
fVal	REAL	要转换的 REAL 值。

返回参数	含义
TRUE	无错误
FALSE	在函数执行期间出错

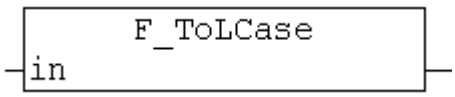
示例：

请参见：示例：通信 BC/BX<->PC/CX（F_SwapRealEx）。 [▶ 352]

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.42 F_ToLCase



F_ToLCase 函数将字符串中的所有大写字母转换为小写字母。

字符集

默认情况下，转换功能使用 Windows 代码页 1252 Latin 1 的字符集，SBCS（单字节字符集）。通过全局变量 GLOBAL_SBCS_TABLE（参见示例）可以在运行时选择不同的字符集（目前只有 Windows 代码页 1250 中欧字符集）。

返回值

名称	类型	描述
F_ToLCase	T_MaxString	

输入

```
VAR_INPUT
    in : T_MaxString;
END_VAR
```

名称	类型	描述
in	T_MaxString	要转换的字符串（类型：T_MaxString）。

示例：

```
PROGRAM MAIN
VAR
    sLCase : STRING;
END_VAR
```

```
sLCase := F_ToLCase( 'TO LOWER CASE 1234567890 ÄÖÜß' );
```

转换的结果为：'to lower case 1234567890 äöüß'

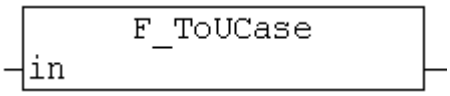
```
GLOBAL_SBCS_TABLE := eSBCS_CentralEuropean;
sLCase := F_ToLCase( 'TO LOWER CASE 1234567890 ĄĘŚĆŻŹŁÓ' );
```

转换的结果为：'to lower case 1234567890 ąęśćżźłó'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.43 F_ToUCase



函数 F_ToUCase 将字符串的所有小写字母转换为大写字母。

字符集

默认情况下，转换功能使用 Windows 代码页 1252 Latin 1 的字符集，SBCS（单字节字符集）。通过全局变量 GLOBAL_SBCS_TABLE（参见示例）可以在运行时选择不同的字符集（目前只有 Windows 代码页 1250 中欧字符集）。

返回值

名称	类型	描述
F_ToUCase	T_MaxString	

输入

```
VAR_INPUT
    in : T_MaxString;
END_VAR
```

名称	类型	描述
in	T_MaxString	要转换的字符串。

示例：

```
PROGRAM MAIN
VAR
    sUCase : STRING;
END_VAR
```

```
sUCase := F_ToUCase( 'to upper case 1234567890 äöüß' );
```

转换的结果为：'TO UPPER CASE 1234567890 ÄÖÜß'

```
GLOBAL_SBCS_TABLE := eSBCS_CentralEuropean;
sUCase := F_ToUCase( 'to upper case 1234567890 ąęśćżźłó' );
```

转换的结果为：'TO UPPER CASE 1234567890 ĄĘŚĆŻŹŁÓ'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.44 GUID_TO_REGSTRING

该函数将结构化 GUID [▶ 331] 变量转换为注册表 GUID 字符串变量（用大括号括起来）。

返回值

名称	类型	描述
GUID_TO_REGSTRING	STRING(38)	

输入

```
VAR_INPUT
    in : GUID;
END_VAR
```

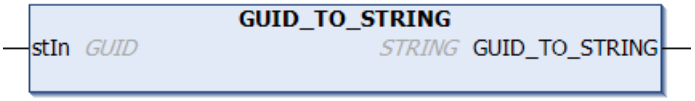
名称	类型	描述
in	GUID	

返回值	含义
'{xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx}'	无错误（“x”为十六进制半字节）
'{00000000-0000-0000-0000-000000000000}'	无错误，GUID 具有初始值（所有字节均为零）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.45 GUID_TO_STRING



该函数将结构化 GUID [▶ 331] 变量转换为 GUID 字符串变量（不带大括号）。

返回值

名称	类型	描述
GUID_TO_STRING	STRING	

输入

```
VAR_INPUT
    stIn : GUID;
END_VAR
```

名称	类型	描述
stIn	GUID	

返回值	含义
'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx'	无错误（“x”为十六进制半字节）
00000000-0000-0000-0000-000000000000'	无错误，GUID 具有初始值（所有字节均为零）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.46 GuidsEqualByVal

该函数比较 2 个 GUID 值。

返回值

名称	类型	描述
GuidsEqualByVal	BOOL	

输入

```
VAR_INPUT
    guidA : GUID;
    guidB : GUID;
END_VAR
```

名称	类型	描述
guidA	GUID	
guidB	GUID	

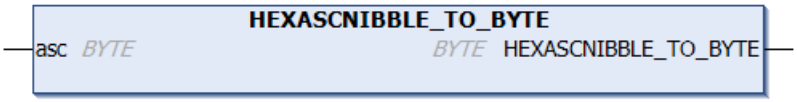
返回值	含义
FALSE	guidA <> guidB

返回值	含义
TRUE	guidA = guidB

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.47 HEXASCNIBBLE_TO_BYTE



该函数将十六进制半字节的 ASCII 代码转换为十进制值。

返回值

名称	类型	描述
HEXASCNIBBLE_TO_BYTE	BYTE	

输入

```
VAR_INPUT
    asc : BYTE;
END_VAR
```

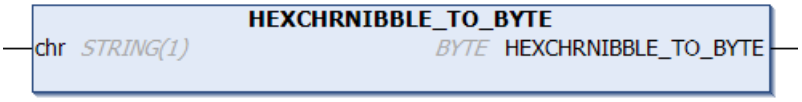
名称	类型	描述
asc	BYTE	十六进制半字节的 ascii 代码（ascii 代码来自：“0”至“9”或“a”至“f”或“A”至“F”）。

返回值	含义
0 至 15	成功，无错误。
255	错误，错误的输入参数值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.48 HEXCHRNIBBLE_TO_BYTE



该函数将十六进制半字节转换为其十进制值。

返回值

名称	类型	描述
HEXCHRNIBBLE_TO_BYTE	BYTE	

输入

```
VAR_INPUT
    chr : STRING(1);
END_VAR
```

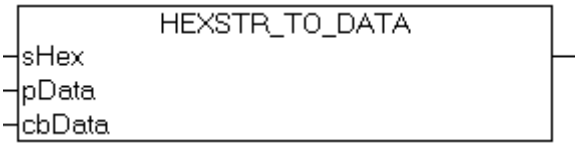
名称	类型	描述
chr	STRING(1)	十六进制半字节（“0”至“9”或“a”至“f”或“A”至“F”）。

返回值	含义
0 至 15	成功，无错误。
255	错误，错误的输入参数值。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.49 HEXSTR_TO_DATA



该函数将十六进制字符串转换为二进制数据，并返回成功转换的数据字节数作为结果。在要转换的十六进制字符串中，仅限空格可以用作分隔符。允许使用小写字母和大写字母作为十六进制字符。如果出现错误或非法字符，则会中止转换，并返回零长度作为结果。

返回值

名称	类型	描述
HEXSTR_TO_DATA	UDINT	

输入

```
VAR_INPUT
    sHex   : T_MaxString;
    pData  : POINTER TO BYTE;
    cbData : UDINT;
END_VAR
```

名称	类型	描述
sHex	T_MaxString	要转换的十六进制字符串（类型：T_MaxString，例如：'AB CD 01 23'）。
pData	POINTER TO BYTE	要写入转换后的数据字节的目标缓冲区的起始地址（指针）。使用 ADR 运算符可以确定地址。
cbData	UDINT	目标缓冲区的最大可用长度。使用 SIZEOF 运算符可以确定长度。

示例：

```
PROGRAM MAIN
VAR
    sH     : STRING := 'AB CD EF 01 23 45 67 89';
    data   : ARRAY[0..10] OF BYTE;
    cbData : UDINT;
END_VAR
```

```
cbData := HEXSTR_TO_DATA( sH, ADR( data ), SIZEOF( data ) );
```

结果（在线）：

TwinCAT_Device.Untitled2.P_TEST2

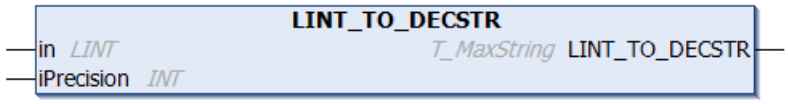
Expression	Type	Value	Prepared value	Address	Comment
sH	STRING	'AB CD EF 01 23 45 67 89'			
data	ARRAY [0..10] OF BYTE				
data[0]	BYTE	16#AB			
data[1]	BYTE	16#CD			
data[2]	BYTE	16#EF			
data[3]	BYTE	16#01			
data[4]	BYTE	16#23			
data[5]	BYTE	16#45			
data[6]	BYTE	16#67			
data[7]	BYTE	16#89			
data[8]	BYTE	16#00			
data[9]	BYTE	16#00			
data[10]	BYTE	16#00			
cbData	UDINT	16#00000008			

```
1  cbData 16#00000008 := HEXSTR_TO_DATA( sH 'AB CD EF 0 ▸, ADR( data ), SIZEOF( data ) );RETURN
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.50 LINT_TO_DECSTR



该函数将有符号的十进制数转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
LINT_TO_DECSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : LINT;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	LINT	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。对于负数，负号将出现在结果字符串中。

示例：

```
PROGRAM MAIN
VAR
    s1    : STRING;
    s2    : STRING;
    s3    : STRING;
```

```
    s4   : STRING;
    iCnt : LINT;
END_VAR
```

```
iCnt := -1234;
s1   := LINT_TO_DECSTR( iCnt, 1 );
s2   := LINT_TO_DECSTR( iCnt, 10 );
iCnt := 0;
s3   := LINT_TO_DECSTR( iCnt, 0 );
iCnt := 1234;
s4   := LINT_TO_DECSTR( iCnt, 10 );
```

结果：

s1 = '-1234'

s2 = '-0000001234'

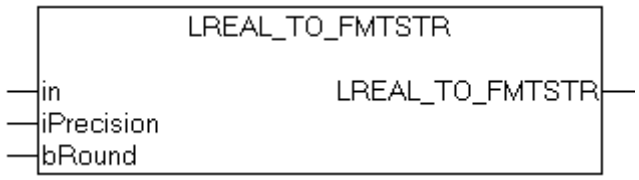
s3 = ''

s4 = '0000001234'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.51 LREAL_TO_FMTSTR



该函数将浮点数转换并格式化为具有以下格式的字符串变量：**[–]dddd.dddd**（dddd 为十进制数）。小数点前的位数取决于浮点数的值。小数点后的位数取决于所需的精度。仅在负值中才会出现符号。对于无限正值，返回 **'#INF'**，对于无限负值，返回 **'-#INF'**。如果传输的变量具有无效值（NaN，非数字），则返回 **'#QNAN'** 或 **'-#QNAN'**。如果格式化的字符串的长度超过了结果字符串的最大允许长度，则返回 **'#OVF'** 或 **'-#OVF'**。

返回值

名称	类型	描述
LREAL_TO_FMTSTR	STRING(510)	

输入

```
VAR_INPUT
    in         : LREAL;
    iPrecision : INT;
    bRound     : BOOL;
END_VAR
```

名称	类型	描述
in	LREAL	要转换和格式化的浮点数。
iPrecision	INT	精度。该值可确定小数点后的位数。对于最小值（零），不会显示小数位。iPrecision 的最大值受小数点前的位数和结果字符串的最大允许长度限制。如果 in = 0 且 iPrecision = 0，则返回字符串“0”。
bRound	BOOL	如果设置了 bRound 参数，则格式化的字符串将根据相应的小数位数（iPrecision）进行四舍五入。以下规则适用于四舍五入：如果所需的最后一个小数位后的小数 >= 5，则该值向上取整，否则不会向上取整。

示例 1：

将 0.46523 转换为带有两位小数的字符串，并进行四舍五入。

```
sOut := LREAL_TO_FMTSTR( 0.46523, 2, TRUE );
```

结果为：“0.47”；

示例：2

LREAL 变量的最大有效位数被限定为 15。

由于浮点数的内部表示和转换期间的舍入误差，结果字符串可能无法与 *in* 变量的值精确对应。

```
PROGRAM MAIN
VAR
    double : LREAL;
    s1     : STRING;
    s2     : STRING;
    s3     : STRING;
    s4     : STRING;
END_VAR
```

```
double := 0.5;
s1     := LREAL_TO_FMTSTR( double, 25, FALSE );
s2     := LREAL_TO_FMTSTR( double, 2, FALSE );
s3     := LREAL_TO_FMTSTR( double, 0, TRUE );
s4     := LREAL_TO_FMTSTR( double, 2, TRUE );
```

结果为：

s1 = '0.4999999999999999756000000' 这是在内部表示双变量的方法。该数字可用作舍入操作的起点。

s2 = '0.49'

四舍五入会导致以下结果：

s3 = '0'

s4 = '0.50'

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.52 LWORD_TO_BASE36STR

该函数将十进制数转换为 Base36 字符串（以 16 为基数）。对于以 16 为基数的数字，除了数字 0-9 之外，还可以使用字母 A-Z。

返回值

名称	类型	描述
LWORD_TO_BASE36STR	T_MaxString	

输入

```
VAR_INPUT
    in          : LWORD;
    iPrecision  : INT;
    bLoCase     : BOOL;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的十进制数。

名称	类型	描述
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。
bLoCase	BOOL	该参数确定在转换时使用小写字母还是大写字母。FALSE => "ABCDEFXY"，TRUE => "abcdefxy"。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    nCnt : LWORD;
END_VAR
```

```
nCnt := 43981;
s1   := LWORD_TO_BASE36STR( nCnt, 1, FALSE );
s2   := LWORD_TO_BASE36STR( nCnt, 1, TRUE );
nCnt := 15;
s3   := LWORD_TO_BASE36STR( nCnt, 4, FALSE );
nCnt := 0;
s4   := LWORD_TO_BASE36STR( nCnt, 0, FALSE );
```

结果：

s1 = 'XXP'

s2 = 'xxp'

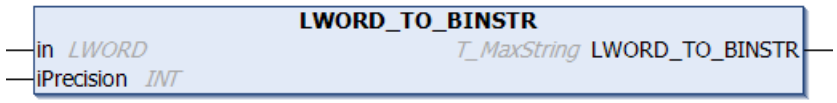
s3 = '000F'

s4 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.47.0

4.53 LWORD_TO_BINSTR



该函数将十进制数转换为二进制字符串（以 2 为基数）。

返回值

名称	类型	描述
LWORD_TO_BINSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : LWORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的十进制数。

名称	类型	描述
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1    : STRING;
    s2    : STRING;
    s3    : STRING;
    nCnt : LWORD;
END_VAR
```

```
s1   := LWORD_TO_BINSTR( 16#81, 16 );
nCnt := 15;
s2   := LWORD_TO_BINSTR( nCnt, 1 );
nCnt := 0;
s3   := LWORD_TO_BINSTR( nCnt, 0 );
```

结果：

s1 = '0000000010000001'

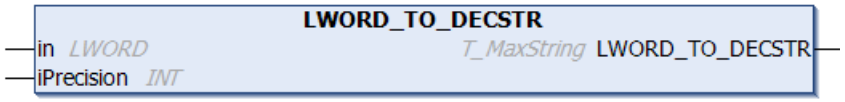
s2 = '1111'

s3 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.54 LWORD_TO_DECSTR



该函数将十进制数转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
LWORD_TO_DECSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : LWORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1    : STRING;
    s2    : STRING;
```

```
    s3   : STRING;
    nCnt : LWORD;
END_VAR
```

```
nCnt := 43981;
s1   := LWORD_TO_DECSTR( nCnt, 1 );
s2   := LWORD_TO_DECSTR( nCnt, 10 );
nCnt := 0;
s3   := LWORD_TO_DECSTR( nCnt, 0 );
```

结果：

s1 = '43981'

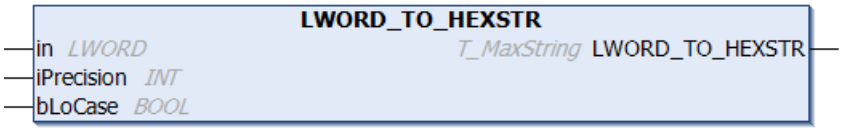
s2 = '0000043981'

s3 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.55 LWORD_TO_HEXSTR



该函数将十进制数转换为十六进制字符串（以 16 为基数）。

返回值

名称	类型	描述
LWORD_TO_HEXSTR	T_MaxString	

输入

```
VAR_INPUT
    in          : LWORD;
    iPrecision  : INT;
    bLoCase     : BOOL;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。
bLoCase	BOOL	该参数确定在转换时使用小写字母还是大写字母。FALSE => "ABCDEF"，TRUE => "abcdef"。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    nCnt : LWORD;
END_VAR
```

```
nCnt := 43981;
s1   := LWORD_TO_HEXSTR( nCnt, 1, FALSE );
s2   := LWORD_TO_HEXSTR( nCnt, 1, TRUE );
nCnt := 15;
s3   := LWORD_TO_HEXSTR( nCnt, 4, FALSE );
nCnt := 0;
s4   := LWORD_TO_HEXSTR( nCnt, 0, FALSE );
```

结果：

s1 = 'ABCD'

s2 = 'abcd'

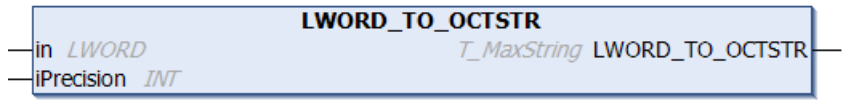
s3 = '000F'

s4 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.56 LWORD_TO_OCTSTR



该函数将十进制数转换为八进制字符串（以 8 为基数）。

返回值

名称	类型	描述
LWORD_TO_OCTSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : LWORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	LWORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    nCnt : LWORD;
END_VAR
```

```
nCnt := 43981;
s1   := LWORD_TO_OCTSTR( nCnt, 1 );
s2   := LWORD_TO_OCTSTR( nCnt, 10 );
nCnt := 0;
s3   := LWORD_TO_OCTSTR( nCnt, 0 );
```

结果：

s1 = '125715'

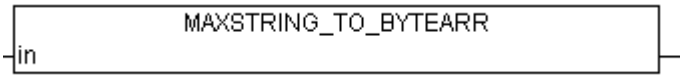
s2 = '0000125715'

s3 = ''

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.57 MAXSTRING_TO_BYTEARR



将字符串转换为字节数组的各个 ASCII 代码。

返回值

名称	类型	描述
MAXSTRING_TO_BYTEARR	ARRAY[0..MAX_STRING_LENGTH] OF BYTE	

输入

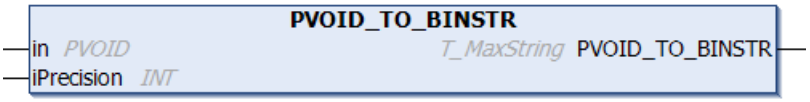
```
VAR_INPUT
    in : T_MaxString;
END_VAR
```

名称	类型	描述
in	T_MaxString	要转换的字符串（类型：T_MaxString）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.58 PVOID_TO_BINSTR



该函数将 PVOID 类型的指针变量的值转换为二进制字符串（以 2 为基数）。

返回值

名称	类型	描述
PVOID_TO_BINSTR	T_MaxString	

输入

```
VAR_INPUT
    in          : PVOID;
    iPrecision  : INT;
END_VAR
```

名称	类型	描述
in	PVOID	要转换的指针变量。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    s5   : STRING;
    s6   : STRING;
    nCnt : BYTE;
    pCnt : PVOID := 0;
END_VAR
```

```
pCnt := 0;
s1   := PVOID_TO_BINSTR( pCnt, 0 );
s2   := PVOID_TO_BINSTR( pCnt, 1 );
s3   := PVOID_TO_BINSTR( pCnt, 32 );

pCnt := ADR( nCnt );
s4   := PVOID_TO_BINSTR( pCnt, 0 );
s5   := PVOID_TO_BINSTR( pCnt, 1 );
s6   := PVOID_TO_BINSTR( pCnt, 32 );
```

结果：

s1 = ''

s2 = '0'

s3 = '00000000000000000000000000000000'

s4 = '10000111110111100000001001010101'（可能有所不同）

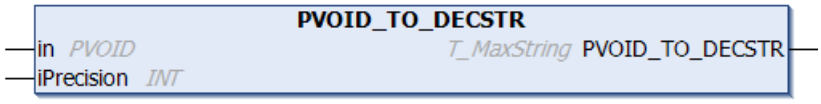
s5 = '10000111110111100000001001010101'（可能有所不同）

s6 = '10000111110111100000001001010101'（可能有所不同）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.59 PVOID_TO_DECSTR



该函数将 PVOID 类型的指针变量的值转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
PVOID_TO_DECSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : PVOID;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	PVOID	要转换的指针变量。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    s5   : STRING;
    s6   : STRING;
    nCnt : WORD;
    pCnt : PVOID := 0;
END_VAR
```

```
pCnt := 0;
s1   := PVOID_TO_DECSTR( pCnt, 0 );
s2   := PVOID_TO_DECSTR( pCnt, 1 );
s3   := PVOID_TO_DECSTR( pCnt, 16 );

pCnt := ADR( nCnt );
s4   := PVOID_TO_DECSTR( pCnt, 0 );
s5   := PVOID_TO_DECSTR( pCnt, 1 );
s6   := PVOID_TO_DECSTR( pCnt, 16 );
```

结果：

s1 = ''

s2 = '0'

s3 = '0000000000000000'

s4 = '2279473749'（可能有所不同）

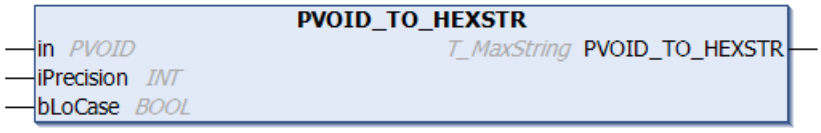
s5 = '2279473749'（可能有所不同）

s6 = '0000002279473749'（可能有所不同）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.60 PVOID_TO_HEXSTR



该函数将 PVOID 类型的指针变量的值转换为十六进制字符串（以 16 为基数）。

返回值

名称	类型	描述
PVOID_TO_HEXSTR	T_MaxString	

输入

```
VAR_INPUT
    in          : PVOID;
    iPrecision  : INT;
    bLoCase     : BOOL;
END_VAR
```

名称	类型	描述
in	PVOID	要转换的指针变量。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

名称	类型	描述
bLoCase	BOOL	该参数确定在转换时使用小写字母还是大写字母。FALSE => "ABCDEF"，TRUE => "abcdef"。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    s5   : STRING;
    s6   : STRING;
    s7   : STRING;
    s8   : STRING;
    s9   : STRING;
    s10  : STRING;
    s11  : STRING;
    s12  : STRING;
    nCnt : WORD;
    pCnt : PVOID := 0;
END_VAR
```

```
pCnt := 0;
s1   := PVOID_TO_HEXSTR( pCnt, 0, FALSE );
s2   := PVOID_TO_HEXSTR( pCnt, 0, TRUE );

s3   := PVOID_TO_HEXSTR( pCnt, 1, FALSE );
s4   := PVOID_TO_HEXSTR( pCnt, 1, TRUE );

s5   := PVOID_TO_HEXSTR( pCnt, 16, FALSE );
s6   := PVOID_TO_HEXSTR( pCnt, 16, TRUE );

pCnt := ADR( nCnt );
s7   := PVOID_TO_HEXSTR( pCnt, 0, FALSE );
s8   := PVOID_TO_HEXSTR( pCnt, 0, TRUE );

s9   := PVOID_TO_HEXSTR( pCnt, 1, FALSE );
s10  := PVOID_TO_HEXSTR( pCnt, 1, TRUE );

s11  := PVOID_TO_HEXSTR( pCnt, 16, FALSE );
s12  := PVOID_TO_HEXSTR( pCnt, 16, TRUE );
```

结果：

s1 = ''

s2 = ''

s3 = '0'

s4 = '0'

s5 = '0000000000000000'

s6 = '0000000000000000'

s7 = '87CBC255'（可能有所不同）

s8 = '87cbc255'（可能有所不同）

s9 = '87CBC255'（可能有所不同）

s10 = '87cbc255'（可能有所不同）

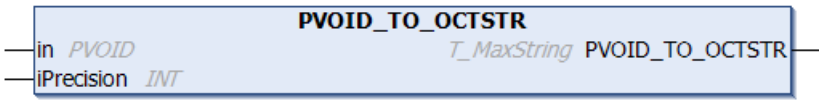
s11 = '0000000087CBC255'（可能有所不同）

s12 = '0000000087cbc255'（可能有所不同）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.61 PVOID_TO_OCTSTR



该函数将 PVOID 类型的指针变量的值转换为八进制字符串（以 8 为基数）。

返回值

名称	类型	描述
PVOID_TO_OCTSTR	T_MaxString	

输入

```
VAR_INPUT
    in         : PVOID;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	PVOID	要转换的指针变量。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    s3   : STRING;
    s4   : STRING;
    s5   : STRING;
    s6   : STRING;
    nCnt : WORD;
    pCnt : PVOID := 0;
END_VAR
```

```
pCnt := 0;
s1   := PVOID_TO_OCTSTR( pCnt, 0 );
s2   := PVOID_TO_OCTSTR( pCnt, 1 );
s3   := PVOID_TO_OCTSTR( pCnt, 16 );

pCnt := ADR( nCnt );
s4   := PVOID_TO_OCTSTR( pCnt, 0 );
s5   := PVOID_TO_OCTSTR( pCnt, 1 );
s6   := PVOID_TO_OCTSTR( pCnt, 16 );
```

结果：

s1 = ''

s2 = '0'

s3 = '0000000000000000'

s4 = '20767501125'（可能有所不同）

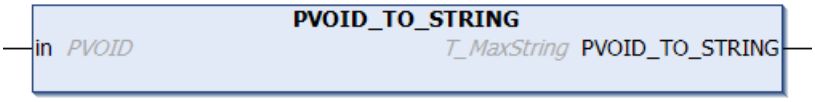
s5 = '20767501125'（可能有所不同）

s6 = '0000020767501125'（可能有所不同）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.62 PVOID_TO_STRING



该函数将 PVOID 类型的指针变量的值转换为十六进制字符串（以 16 为基数）。十六进制字符串具有 PLC 前缀：“16#”。在 32 位系统中，分辨率固定为 8 位，在 64 位系统中，分辨率固定为 16 位。

返回值

名称	类型	描述
PVOID_TO_STRING	T_MaxString	

输入

```
VAR_INPUT
    in : PVOID;
END_VAR
```

名称	类型	描述
in	PVOID	要转换的指针变量。

示例：

```
PROGRAM MAIN
VAR
    s1   : STRING;
    s2   : STRING;
    nCnt : BYTE;
    p1   : POINTER TO BYTE := 0;
    p2   : POINTER TO BYTE := ADR( nCnt );
END_VAR
```

```
s1 := PVOID_TO_STRING( p1 );
s2 := PVOID_TO_STRING( p2 );
```

32 位系统上的结果：

s1 = '16#00000000'

s2 = ''16#87DE0255'（可能有所不同）

64 位系统上的结果：

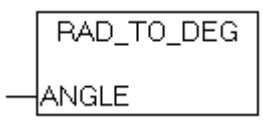
s1 = '16#0000000000000000'

s2 = ''16#8734651087DE0255'（可能有所不同）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.63 RAD_TO_DEG



该函数将弧度转换为度角。

返回值

名称	类型	描述
RAD_TO_DEG	LREAL	

输入

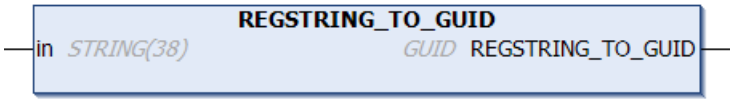
```
VAR_INPUT
    ANGLE : LREAL;
END_VAR
```

名称	类型	描述
ANGLE	LREAL	要转换的弧度。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.64 REGSTRING_TO_GUID



该函数将注册表 GUID 字符串变量（用大括号括起来）转换为结构化 GUID [▶ 331] 变量。

返回值

名称	类型	描述
REGSTRING_TO_GUID	GUID	

输入

```
VAR_INPUT
    in : STRING(38);
END_VAR
```

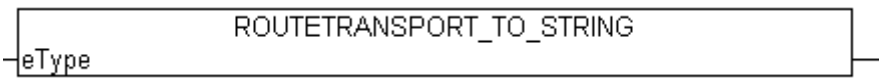
名称	类型	描述
in	STRING(38)	

返回值	含义
'{xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx}'	无错误（“x”为十六进制半字节）。
'{00000000-0000-0000-0000-000000000000}'	转换失败或 GUID 具有初始值（所有字节均为零）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.65 ROUTETRANSPORT_TO_STRING



该函数将 AMS 消息路由器传输层 ID 转换为字符串。

返回值

名称	类型	描述
ROUTETRANSPORT_TO_STRING	STRING	

输入

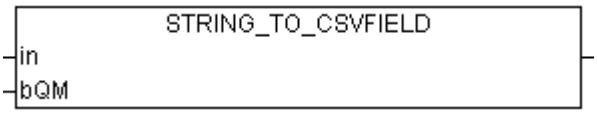
```
VAR_INPUT
    eType : E_RouteTransportType;
END_VAR
```

名称	类型	描述
eType	E_RouteTransportType	要转换的传输层标识符（类型：E_RouteTransportType [▶ 329]）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.66 STRING_TO_CSVFIELD



该函数将 PLC 字符串变量的值转换为字符串形式的 CSV 格式的数据字段。源字符串中的单引号被替换为双引号。如果设置了 bQM 参数（QM = 引号），则还会添加外层引号（将 CSV 数据字段括起来）。如果成功，则该函数会返回转换后的字符串作为结果。如果在转换期间发生错误，则该函数会返回一个空字符串，但前提是源字符串不是空字符串。

该函数通常与功能块 FB_CSVMemBufferWriter [▶ 50] 一起使用，在 PLC 内存中生成 CSV 格式的数据集。在下一步中，可以将内存内容写入文件。

源字符串不得包含二进制数据。值为零的二进制数据会在错误的位置终止并截断字符串。要转换二进制数据，请使用函数：ARG_TO_CSVFIELD [▶ 266]。

返回值

名称	类型	描述
STRING_TO_CSVFIELD	T_MaxString	

输入

```
VAR_INPUT
    in  : T_MaxString;
    bQM : BOOL;
END_VAR
```

名称	类型	描述
in	T_MaxString	源字符串，其值要转换为 CSV 格式的数据字段（类型：T_MaxString）。
bQM	BOOL	如果该输入为 TRUE，则会从结果字符串中添加括起来的引号。

bQM	描述	源字符串	结果字符串	符合 CSV 标准
FALSE	没有外围引号的源字符串应仅包含字母和数字。在这种情况下，源字符串不得包含任何不可打印的控制字符、引号、分号、逗号（US CSV 格式）或二进制数据。			
		'Module_XA5'	'Module_XA5'	是
		'123456'	'123456'	是
		''	''	是
		'A""B'	'A""""B'	否
		'A"B'	'A""B'	否
		';'	';'	否
		'RN'	'RN'	否
		'AB$00CD'	'AB'（字符串被截断）	否
TRUE	没有用引号括起来的源字符串不应包含任何不可打印的控制字符、引号、分号或逗号（美国 CSV 格式）。不允许使用二进制数据。	'Module_XA5'	'"Module_XA5"'	是
		'123456'	'"123456"'	是
		''	'""'	是
		'A""B'	'"A""""B"'	是
		'A"B'	'"A""B"'	是
		';'	'";"'	是
		'RN'	'"RN"'	是
		'AB$00CD'	'"AB"'（字符串被截断）	否

示例：

```
PROGRAM MAIN
VAR
    s1 : STRING;
    s2 : STRING;
END_VAR
```

```
s1 := STRING_TO_CSVFIELD( 'Module_"ALFA_$05"_6', TRUE );
s2 := STRING_TO_CSVFIELD( 'Module_50', FALSE );
```

结果：

s1 = '"Module_""ALFA_$05""_6"'

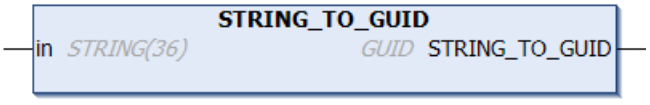
s2 = 'Module_50'

有关更多信息，请访问：示例：写入/读取 CSV 文件 [▶ 369]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.67 STRING_TO_GUID



该函数将 GUID 字符串变量（不带大括号）转换为结构化 GUID [▶ 331] 变量。

返回值

名称	类型	描述
STRING_TO_GUID	GUID	

输入

```
VAR_INPUT
    in : STRING(36);
END_VAR
```

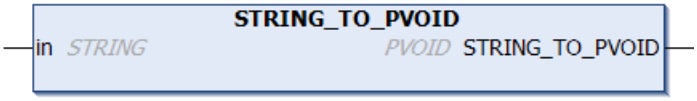
名称	类型	描述
in	STRING(36)	

返回值	含义
'xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx'	无错误（“x”为十六进制半字节）
'00000000-0000-0000-0000-000000000000'	转换失败或 GUID 具有初始值（所有字节均为零）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.68 STRING_TO_PVOID



该函数将字符串变量转换为 PVOID 类型的指针变量。如果输入字符串包含不正确的字符且不能被解释为地址，则该函数会返回零。

返回值

名称	类型	描述
STRING_TO_PVOID	PVOID	

输入

```
VAR_INPUT
    in  : STRING;
END_VAR
```

名称	类型	描述
in	STRING	要转换的字符串变量。

示例：

```
PROGRAM MAIN
VAR
    sP1 : STRING := '16#89345678';
    sP2 : STRING := '8#21115053170';
    sP3 : STRING := '2#10001001001101000101011001111000';
    sP4 : STRING := '2301908600';
    sP5 : STRING := '';
    pP1 : PVOID := 0;
    pP2 : PVOID := 0;
    pP3 : PVOID := 0;
    pP4 : PVOID := 0;
    pP5 : PVOID := 0;
END_VAR
```

```
pP1 := STRING_TO_PVOID( sP1 );
pP2 := STRING_TO_PVOID( sP2 );
pP3 := STRING_TO_PVOID( sP3 );
pP4 := STRING_TO_PVOID( sP4 );
pP5 := STRING_TO_PVOID( sP5 );
```

结果：

pP1 = 2301908600

pP2 = 2301908600

pP3 = 2301908600

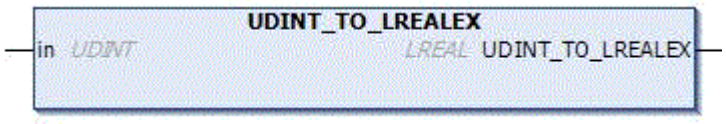
pP4 = 2301908600

pP5 = 0

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.69 UDINT_TO_LREALEX



Arm® 平台上的 TwinCAT 2 不支持将无符号数字转换为 LREAL 类型的浮点数。已设置最高有效位的无符号数字可能会被隐式转换为负浮点数。在这里描述的函数允许在 TwinCAT 2 中将 UDINT 类型显式转换为 LREAL 类型的正浮点数（即使已设置最高有效位且没有编译器警告）。您只需该函数即可编译转换后的 TwinCAT 2 项目，而无需在 TwinCAT 3 中进行更改。

在 TwinCAT 3 中，UDINT 类型的无符号数字始终会（隐式和显式）转换为正浮点数。因此，这个函数可以被省略。

返回值

名称	类型	描述
UDINT_TO_LREALEX	LREAL	

输入

```
VAR_INPUT
    in : UDINT;
END_VAR
```

名称	类型	描述
in	UDINT	

示例：

```
PROGRAM MAIN
VAR
    nUdint : UDINT := 16#FFFFFFFF;
    fLreal : LREAL := 0.0;
END_VAR
```

fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := nUdint	-*1, Warning 1105**	4294967295	4294967295
fLreal := UDINT_TO_LREAL(nUdint)	-*1, Warning 1105**	4294967295	4294967295
fLreal := UDINT#16#FFFFFFFF	-*1, Warning 1105**	4294967295	4294967295
fLreal := 16#FFFFFFFF	4294967295	4294967295	4294967295
fLreal := UDINT_TO_LREALEX(nUdint)	4294967295	4294967295	4294967295
fLreal := UDINT_TO_LREALEX(UDINT#16#FFFFFFFF)	4294967295	4294967295	4294967295

fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := UDINT_TO_LREALEX(16#FFFFFFFF)	4294967295	4294967295	4294967295

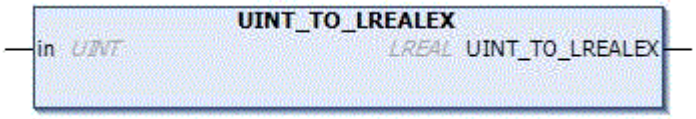
*不支持

**不支持将无符号整数转换为 LREAL。相反，该值被用作有符号数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.70 UINT_TO_LREALEX



Arm® 平台上的 TwinCAT 2 不支持将无符号数字转换为 LREAL 类型的浮点数。已设置最高有效位的无符号数字可能会被隐式转换为负浮点数。在这里描述的函数允许在 TwinCAT 2 中将 UINT 类型显式转换为 LREAL 类型的正浮点数（即使已设置最高有效位且没有编译器警告）。您只需该函数即可编译转换后的 TwinCAT 2 项目，而无需在 TwinCAT 3 中进行更改。

在 TwinCAT 3 中，UINT 类型的无符号数字始终会（隐式和显式）转换为正浮点数。因此，这个函数可以被省略。

返回值

名称	类型	描述
UINT_TO_LREALEX	LREAL	

输入

```
VAR_INPUT
    in : UINT;
END_VAR
```

名称	类型	描述
in	UINT	

示例：

```
PROGRAM MAIN
VAR
    nUint  : UINT  := 16#FFFF;
    fLreal : LREAL := 0.0;
END_VAR
```

fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := nUint	+65535, Warning 1105*	+65535	+65535
fLreal := UINT_TO_LREAL(nUint)	+65535, Warning 1105*	+65535	+65535
fLreal := UINT#16#FFFF	+65535, Warning 1105*	+65535	+65535
fLreal := 16#FFFF	+65535	+65535	+65535

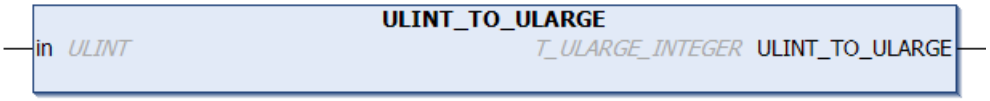
fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := UINT_TO_LREALEX(nUint)	+65535	+65535	+65535
fLreal := UINT_TO_LREALEX(UINT#16#FFFF)	+65535	+65535	+65535
fLreal := UINT_TO_LREALEX(16#FFFF)	+65535	+65535	+65535

*不支持将无符号整数转换为 LREAL。相反，该值被用作有符号数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.71 ULINT_TO_ULARGE



该函数将 TwinCAT 3 无符号的 64 位数字（“本地”类型）转换为 TwinCAT 2 无符号的 64 位数字（“传统”类型：T_ULARGE_INTEGER [▶ 347]）。

返回值

名称	类型	描述
ULINT_TO_ULARGE	T_ULARGE_INTEGER	

输入

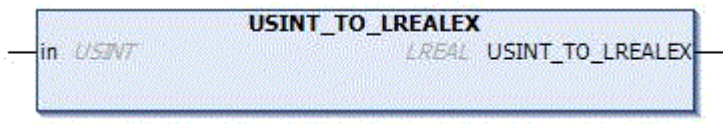
```
VAR_INPUT
    in : ULINT;
END_VAR
```

名称	类型	描述
in	ULINT	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.72 USINT_TO_LREALEX



Arm® 平台上的 TwinCAT 2 不支持将无符号数字转换为 LREAL 类型的浮点数。已设置最高有效位的无符号数字可能会被隐式转换为负浮点数。在这里描述的函数允许在 TwinCAT 2 中将 USINT 类型显式转换为 LREAL 类型的正浮点数（即使已设置最高有效位且没有编译器警告）。您只需该函数即可编译转换后的 TwinCAT 2 项目，而无需在 TwinCAT 3 中进行更改。

在 TwinCAT 3 中，USINT 类型的无符号数字始终会（隐式和显式）转换为正浮点数。因此，这个函数可以被省略。

返回值

名称	类型	描述
USINT_TO_LREALEX	LREAL	

输入

```
VAR_INPUT
    in : USINT;
END_VAR
```

名称	类型	描述
in	USINT	

示例：

```
PROGRAM MAIN
VAR
    nUsint : USINT := 16#FF;
    fLreal : LREAL := 0.0;
END_VAR
```

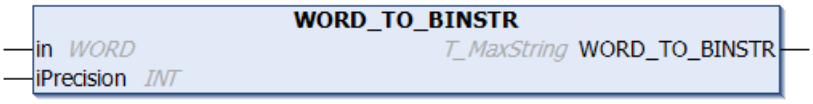
fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := nUsint	+255, Warning 1105*	+255	+255
fLreal := USINT_TO_LREAL(nUsint)	+255, Warning 1105*	+255	+255
fLreal := USINT#16#FF	+255, Warning 1105*	+255	+255
fLreal := 16#FF	+255	+255	+255
fLreal := USINT_TO_LREALEX(nUsint)	+255	+255	+255
fLreal := USINT_TO_LREALEX(USINT#16#FF)	+255	+255	+255
fLreal := USINT_TO_LREALEX(16#FF)	+255	+255	+255

*不支持将无符号整数转换为 LREAL。相反，该值被用作有符号数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.73 WORD_TO_BINSTR



该函数将十进制数转换为二进制字符串（以 2 为基数）。

返回值

名称	类型	描述
WORD_TO_BINSTR	T_MaxString	

输入

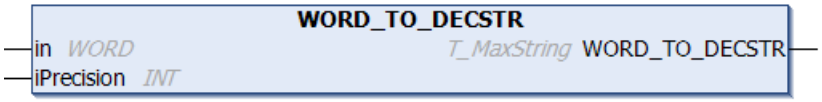
```
VAR_INPUT
    in          : WORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	WORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.74 WORD_TO_DECSTR



该函数将十进制数转换为十进制字符串（以 10 为基数）。

返回值

名称	类型	描述
WORD_TO_DECSTR	T_MaxString	

输入

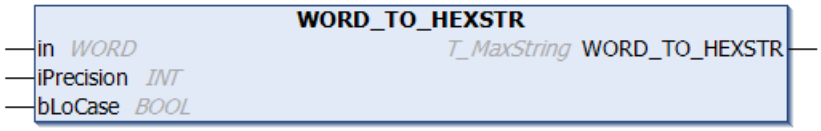
```
VAR_INPUT
    in          : WORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	WORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.75 WORD_TO_HEXSTR



该函数将十进制数转换为十六进制字符串（以 16 为基数）。

返回值

名称	类型	描述
WORD_TO_HEXSTR	T_MaxString	

输入

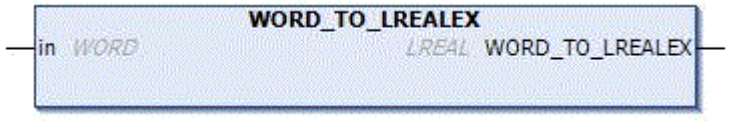
```
VAR_INPUT
    in         : WORD;
    iPrecision : INT;
    bLoCase    : BOOL := FALSE;
END_VAR
```

名称	类型	描述
in	WORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。
bLoCase	BOOL	该参数确定在转换时使用小写字母还是大写字母。FALSE => "ABCDEF"，TRUE => "abcdef"。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.76 WORD_TO_LREALEX



Arm® 平台上的 TwinCAT 2 不支持将无符号数字转换为 LREAL 类型的浮点数。已设置最高有效位的无符号数字可能会被隐式转换为负浮点数。在这里描述的函数允许在 TwinCAT 2 中将 WORD 类型显式转换为 LREAL 类型的正浮点数（即使已设置最高有效位且没有编译器警告）。您只需该函数即可编译转换后的 TwinCAT 2 项目，而无需在 TwinCAT 3 中进行更改。

在 TwinCAT 3 中，WORD 类型的无符号数字始终会（隐式和显式）转换为正浮点数。因此，可以省去该函数。

返回值

名称	类型	描述
WORD_TO_LREALEX	LREAL	

输入

```
VAR_INPUT
    in : WORD;
END_VAR
```

名称	类型	描述
in	WORD	

示例：

```
PROGRAM MAIN
VAR
    nWord  : WORD  := 16#FFFF;
    fLreal : LREAL := 0.0;
END_VAR
```

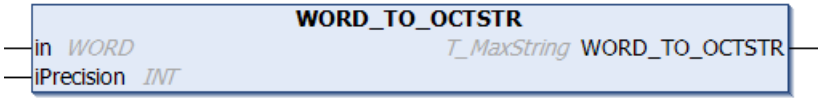
fLreal 值	Tc2.x ARM	Tc2.x X86	Tc3.x ARM、X86、X64
fLreal := nWord	+65535, Warning 1105*	+65535	+65535
fLreal := WORD_TO_LREAL(nWord)	+65535, Warning 1105*	+65535	+65535
fLreal := WORD#16#FFFF	+65535, Warning 1105*	+65535	+65535
fLreal := 16#FFFF	+65535	+65535	+65535
fLreal := WORD_TO_LREALEX(nWord)	+65535	+65535	+65535
fLreal := WORD_TO_LREALEX(WORD#16#FFFF)	+65535	+65535	+65535
fLreal := WORD_TO_LREALEX(16#FFFF)	+65535	+65535	+65535

*不支持将无符号整数转换为 LREAL。相反，该值被用作有符号数。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

4.77 WORD_TO_OCTSTR



该函数将十进制数转换为八进制字符串（以 8 为基数）。

返回值

名称	类型	描述
WORD_TO_OCTSTR	T_MaxString	

输入

```
VAR_INPUT
    in        : WORD;
    iPrecision : INT;
END_VAR
```

名称	类型	描述
in	WORD	要转换的十进制数。
iPrecision	INT	显示的最小位数。如果实际有效位数小于 iPrecision 参数，则从左侧用零填充结果字符串。如果有效位数大于 iPrecision 参数，则结果字符串不会被截断！如果 iPrecision 参数和 in 参数的值为零，则生成字符串是一个空字符串。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5 数据类型

5.1 [已过时]

5.1.1 FLOAT

LREAL 别名类型。

```
TYPE FLOAT :LREAL;
END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.2 ADSDATATYPEID

ADS 数据类型 ID。例如，功能块 FB_ReadSymInfoByNameEx [▶ 130] 使用该数据类型。

```
TYPE ADSDATATYPEID :
(
    ADST_VOID     := 0,
    ADST_INT8     := 16,
    ADST_UINT8    := 17,
    ADST_INT16    := 2,
    ADST_UINT16   := 18,
    ADST_INT32    := 3,
    ADST_UINT32   := 19,
    ADST_INT64    := 20,
    ADST_UINT64   := 21,
    ADST_REAL32   := 4,
    ADST_REAL64   := 5,
    ADST_BIGTYPE  := 65,
    ADST_STRING   := 30,
    ADST_WSTRING  := 31,
    ADST_REAL80   := 32,
    ADST_BIT      := 33,
    ADST_MAXTYPES := 34
);
END_TYPE
```

值	含义
ADST_VOID	预留
ADST_INT8	有符号的 8 位整数
ADST_UINT8	无符号的 8 位整数
ADST_INT16	有符号的 16 位整数
ADST_UINT16	无符号的 16 位整数
ADST_INT32	有符号的 32 位整数
ADST_UINT32	无符号的 32 位整数
ADST_INT64	有符号的 64 位整数
ADST_UINT64	无符号的 64 位整数
ADST_REAL32	32 位浮点数
ADST_REAL64	64 位浮点数
ADST_BIGTYPE	结构化类型
ADST_STRING	字符串类型
ADST_WSTRING	宽字符类型
ADST_REAL80	预留
ADST_BIT	位类型
ADST_MAXTYPES	最大可用类型

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.3 E_AmsLoggerMode

AMS 记录器控制模式。功能块 FB_AmsLogger [▶ 43] 使用该数据类型。

```
TYPE E_AmsLoggerMode :
(
    AMSLOGGER_RUN   := 1,
    AMSLOGGER_STOP  := 2
) UINT;
END_TYPE
```

值	含义
AMSLOGGER_RUN	启动 AMS 记录器
AMSLOGGER_STOP	停止 AMS 记录器

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.4 E_ArgType

内部参数类型 ID。字符串格式化函数/功能块使用该类型。

```
TYPE E_ArgType :
(
    ARGTYPE_UNKNOWN     := 0,
    ARGTYPE_BYTE,
    ARGTYPE_WORD,
    ARGTYPE_DWORD,
    ARGTYPE_REAL,
    ARGTYPE_LREAL,
    ARGTYPE_SINT,
    ARGTYPE_INT,
    ARGTYPE_DINT,
    ARGTYPE_USINT,
    ARGTYPE_UINT,
    ARGTYPE_UDINT,
    ARGTYPE_STRING,
    ARGTYPE_BOOL,
    ARGTYPE_BIGTYPE,
    ARGTYPE_ULARGE,
    ARGTYPE_UHUGE,
    ARGTYPE_LARGE,
    ARGTYPE_HUGE,
    ARGTYPE_LWORD
);
END_TYPE
```

值	相应的 PLC 数据类型
ARGTYPE_UNKNOWN	类型未知或未初始化
ARGTYPE_BYTE	BYTE（8 位）
ARGTYPE_WORD	WORD（16 位）
ARGTYPE_DWORD	DWORD（32 位）
ARGTYPE_REAL	REAL
ARGTYPE_LREAL	LREAL
ARGTYPE_SINT	SINT
ARGTYPE_INT	INT
ARGTYPE_DINT	DINT
ARGTYPE_USINT	USINT
ARGTYPE_UINT	UINT

值	相应的 PLC 数据类型
ARGTYPE_UDINT	UDINT
ARGTYPE_STRING	字符串类型：T_MaxString
ARGTYPE_BOOL	BOOL
ARGTYPE_BIGTYPE	任何数据结构或字节缓冲区
ARGTYPE_ULARGE	T_ULARGE_INTEGER 或 ULINT（无符号的 64 位整数）
ARGTYPE_UHUGE	T_UHUGE_INTEGER（无符号的 128 位整数）
ARGTYPE_LARGE	T_LARGE_INTEGER 或 LINT（有符号的 64 位整数）
ARGTYPE_HUGE	T_HUGE_INTEGER（有符号的 128 位整数）
ARGTYPE_LWORD	LWORD（64 位）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.5 E_DbgContext

协议块可以使用该变量类型。它可确定调试输出的上下文。

```
TYPE E_DbgContext :
(
    eDbgContext_NONE := 0,(* Not used *)
    eDbgContext_USER := 1,(* Service user *)
    eDbgContext_PROV := 2 (* Service provider *)
);
END_TYPE
```

值	含义
eDbgContext_NONE	未使用的参数
eDbgContext_USER	由服务用户触发调试输出
eDbgContext_PROV	由服务提供商触发调试输出

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.6 E_DbgDirection

缓冲区块或协议块可以使用该变量类型，以配置调试输出。

```
TYPE E_DbgDirection :
(
    eDbgDirection_OFF := 0,(* Disabled (no debug oputput) *)
    eDbgDirection_IN  := 1,(* Enabled only for incomming data *)
    eDbgDirection_OUT := 2,(* Enabled only for outgoing data *)
    eDbgDirection_ALL := 3(* Enabled for incomming and outgoing data *)
);
END_TYPE
```

值	含义
eDbgDirection_OFF	禁用调试输出
eDbgDirection_IN	激活传入报文的输出
eDbgDirection_OUT	激活传出报文的输出
eDbgDirection_ALL	激活传入和传出报文的输出

示例：

例如，调试输出本身可以通过 ADSLOGSTR 函数实现。

例如，在环形缓冲区中，按照以下方式通过变量可以控制调试输出：

- 如果值为 *eDbgDirection_IN* 或 *eDbgDirection_ALL*，则在缓冲区中添加新值时会触发调试输出；
- 如果值为 *eDbgDirection_out* 或 *eDbgDirection_ALL*，则从缓冲区中移除值时会触发调试输出；

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.7 E_EnumCmdType

枚举块的控制参数。并非每个枚举块都会使用所有参数！

```
TYPE E_EnumCmdType :
(
    eEnumCmd_First := 0,
    eEnumCmd_Next,
    eEnumCmd_Abort
);
END_TYPE
```

值	含义
eEnumCmd_First	列出第一个元素
eEnumCmd_Next	列出下一个元素
eEnumCmd_Abort	取消列表（关闭已打开的句柄）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.8 E_HashMode

```
{attribute 'qualified_only'}
{attribute 'strict'}
TYPE E_HashMode :
(
    HASH_MD5 := 1,      // generates 16 bytes hash value
    HASH_SHA1,          // generates 20 bytes hash value
    HASH_SHA256,        // generates 32 bytes hash value
    HASH_SHA384,        // generates 48 bytes hash value
    HASH_SHA512         // generates 64 bytes hash value
) DINT;
END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024.29	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.51.0

5.9 E_LDevType

```
TYPE E_LDevType :
(
    eLDT_Unknown        := 0,
    eLDT_Beckhoff,
    eLDT_GenericPC,
    eLDT_TerminalDongle,
    eLDT_UsbDongle
) UDINT;
END_TYPE
```

值	含义
eLDT_Unknown	设备类型未知。
eLDT_Beckhoff	倍福设备（例如 IPC、CX、……）
eLDT_GenericPC	第三方 PC
eLDT_TerminalDongle	授权密钥端子模块，例如 EL6070
eLDT_UsbDongle	授权密钥 U 盘，例如 C9900-L100

5.10 E_LDongleStatus

```
TYPE E_LDongleStatus :
(
    eLDT_Unknown      := 0,
    eLDT_OK,
    eLDT_Pending,
    eLDT_Invalid,
    eLDT_NoConnection
) UDINT;
END_TYPE
```

值	含义
eLDT_Unknown	授权加密狗状态未知
eLDT_OK	授权加密狗已成功验证
eLDT_Pending	正在验证授权加密狗
eLDT_Invalid	授权加密狗无效
eLDT_NoConnection	没有连接至授权加密狗

5.11 E_LicenseHResult

```
TYPE E_LicenseHResult :
(
    //success
    E_LHR_LicenseOK                 := 0,
    E_LHR_LicenseOK_Pending         := 16#203,
    E_LHR_LicenseOK_Demo            := 16#254,
    E_LHR_LicenseOK_OEM             := 16#255,
    //error
    E_LHR_LicenseNoFound            := TO_DINT(16#98110700+16#24),
    E_LHR_LicenseExpired            := TO_DINT(16#98110700+16#25),
    E_LHR_LicenseExceeded           := TO_DINT(16#98110700+16#26),
    E_LHR_LicenseInvalid            := TO_DINT(16#98110700+16#27),
    E_LHR_LicenseSystemIdInvalid    := TO_DINT(16#98110700+16#28),
    E_LHR_LicenseNoTimeLimit        := TO_DINT(16#98110700+16#29),
    E_LHR_LicenseTimeInFuture       := TO_DINT(16#98110700+16#2A),
    E_LHR_LicenseTimePeriodToLong   := TO_DINT(16#98110700+16#2B),
    E_LHR_DeviceException           := TO_DINT(16#98110700+16#2C),
    E_LHR_LicenseDuplicated         := TO_DINT(16#98110700+16#2D),
    E_LHR_SignatureInvalid          := TO_DINT(16#98110700+16#2E),
    E_LHR_CertificateInvalid        := TO_DINT(16#98110700+16#2F),
    E_LHR_LicenseOemNotFound        := TO_DINT(16#98110700+16#30),
    E_LHR_LicenseRestricted         := TO_DINT(16#98110700+16#31),
    E_LHR_LicenseDemoDenied         := TO_DINT(16#98110700+16#32),
    E_LHR_LicensePlatformLevelInv   := TO_DINT(16#98110700+16#33)
) DINT;
END_TYPE
```

值	含义
E_LHR_LicenseOK	授权有效
E_LHR_LicenseOK_Pending	需要验证授权设备（例如，授权密钥端子模块）
E_LHR_LicenseOK_Demo	试用版授权有效
E_LHR_LicenseOK_OEM	OEM 授权有效
E_LHR_LicenseNoFound	缺少授权
E_LHR_LicenseExpired	授权已过期
E_LHR_LicenseExceeded	授权实例过少
E_LHR_LicenseInvalid	授权无效
E_LHR_LicenseSystemIdInvalid	授权的系统 ID 不正确
E_LHR_LicenseNoTimeLimit	授权不受时间限制

值	含义
E_LHR_LicenseTimeInFuture	授权问题：未来的签发时间
E_LHR_LicenseTimePeriodToLong	授权期限过长
E_LHR_DeviceException	系统启动异常
E_LHR_LicenseDuplicated	多次读取授权数据
E_LHR_SignatureInvalid	签名无效
E_LHR_CertificateInvalid	证书无效
E_LHR_LicenseOemNotFound	未知 OEM 的 OEM 授权
E_LHR_LicenseRestricted	授权对系统无效
E_LHR_LicenseDemoDenied	不允许使用试用版授权
E_LHR_LicensePlatformLevelInv	授权的平台级别无效

5.12 E_MIB_IF_Type

管理信息库接口类型。

```
TYPE E_MIB_IF_Type :
(
    MIB_IF_TYPE_OTHER      := 1,
    MIB_IF_TYPE_ETHERNET   := 6,
    MIB_IF_TYPE_TOKENRING  := 9,
    MIB_IF_TYPE_FDDI       := 15,
    MIB_IF_TYPE_PPP        := 23,
    MIB_IF_TYPE_LOOPBACK   := 24,
    MIB_IF_TYPE_SLIP       := 28
);
END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.13 E_NumGroupTypes

数字编号组。例如，功能块 FB_EnumStringNumbers [▶ 56] 使用该数据类型。

```
TYPE E_NumGroupTypes :
(
    eNumGroup_Float,
    eNumGroup_Unsigned,
    eNumGroup_Signed
);
END_TYPE
```

值	含义
eNumGroup_Float	浮点数
eNumGroup_Unsigned	无符号数字
eNumGroup_Signed	有符号数字

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.14 E_PersistentMode

应该写入持久性数据的模式。功能块 FB_WritePersistentData [▶ 117] 使用该数据类型。

```
TYPE E_PersistentMode :
(
    SPDM_2PASS     := 0,
```

```
    SPDM_VAR_BOOST := 1
);
END_TYPE
```

值	含义
SPDM_2PASS	所有数据均来自同一周期
SPDM_VAR_BOOST	各个持久变量的数据来自同一周期

示例：

另请参见：写入持久性数据：系统行为 [▶ 373]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.15 E_RegValueType

注册表值的类型 ID。

```
TYPE E_RegValueType :
(
    REG_NONE := 0,
    REG_SZ,
    REG_EXPAND_SZ,
    REG_BINARY,
    REG_DWORD,
    REG_DWORD_BIG_ENDIAN,
    REG_LINK,
    REG_MULTI_SZ,
    REG_RESOURCE_LIST,
    REG_FULL_RESOURCE_DESCRIPTOR,
    REG_RESOURCE_REQUIREMENTS_LIST,
    REG_QWORD
) UINT;
END_TYPE
```

值	含义
REG_NONE	无 TYPE 值
REG_SZ	以空字符结尾的 Unicode STRING
REG_EXPAND_SZ	以空字符结尾的 Unicode STRING（带有环境变量引用）
REG_BINARY	自由形式二进制
REG_DWORD	32 位数字和 REG_DWORD_LITTLE_ENDIAN（与 REG_DWORD 相同）
REG_DWORD_BIG_ENDIAN	32 位数字
REG_LINK	符号链接（unicode）
REG_MULTI_SZ	多个 Unicode 字符串
REG_RESOURCE_LIST	资源地图中的资源列表
REG_FULL_RESOURCE_DESCRIPTOR	硬件描述中的资源列表
REG_RESOURCE_REQUIREMENTS_LIST	-
REG_QWORD	64 位数字和 REG_QQOERD_LITTLE_ENDIAN（与 REG_QWORD 相同）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.16 E_RouteTransportType

用于承载 AMS 消息的传输层。目前仅支持 TCP/IP 作为传输层。

```
TYPE E_RouteTransportType :
(
    eRouteTransport_None         := 0,
    eRouteTransport_TCP_IP       := 1,
    eRouteTransport_IIO_LIGHTBUS := 2,
    eRouteTransport_PROFIBUS_DP  := 3,
    eRouteTransport_PCI_ISA_BUS  := 4,
    eRouteTransport_ADS_UDP      := 5,
    eRouteTransport_FATP_UDP     := 6,
    eRouteTransport_COM_PORT     := 7,
    eRouteTransport_USB          := 8,
    eRouteTransport_CAN_OPEN     := 9,
    eRouteTransport_DEVICE_NET   := 10,
    eRouteTransport_SSB          := 11,
    eRouteTransport_SOAP         := 12
) UINT;
END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.17 E_SBCSType

Windows SBCS（单字节字符集）代码页类型。

```
TYPE E_SBCSType :
(
    eSBCS_WesternEuropean := 1,
    eSBCS_CentralEuropean := 2
);
END_TYPE
```

值	含义
eSBCS_WesternEuropean	Windows 1252（默认）代码页
eSBCS_CentralEuropean	Windows 1251 代码页

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.18 E_ScopeServerState

```
TYPE E_ScopeServerState
(
   SCOPE_SERVER_IDLE,
   SCOPE_SERVER_CONNECT,
   SCOPE_SERVER_START,
   SCOPE_SERVER_STOP,
   SCOPE_SERVER_SAVE,
   SCOPE_SERVER_DISCONNECT,
   SCOPE_SERVER_RESET
);
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.19 E_TimeZoneID

关于操作系统的配置时区的其他信息。

```
TYPE E_TimeZoneID :
(
    eTimeZoneID_Invalid  := -1,
    eTimeZoneID_Unknown  := 0,
    eTimeZoneID_Standard := 1,
    eTimeZoneID_Daylight := 2
);
END_TYPE
```

值	含义
eTimeZoneID_Invalid	无法读取时区配置
eTimeZoneID_Unknown	可以读取时区配置，但标准时间/夏令时信息未知
eTimeZoneID_Standard	目前使用标准时间
eTimeZoneID_Daylight	目前使用夏令时

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.20 E_TypeFieldParam

字符串格式类型字段。

```
TYPE E_TypeFieldParam :
(
    TYPEFIELD_UNKNOWN := 0,
    TYPEFIELD_B, (* b or B: binary number *)
    TYPEFIELD_O, (* o or O: octal number *)
    TYPEFIELD_U, (* u or U: unsigned decimal number *)
    TYPEFIELD_C, (* c or C: one ASCII character *)
    TYPEFIELD_F, (* f or F: float number ( normalized format )*)
    TYPEFIELD_D, (* d or D: signed decimal number *)
    TYPEFIELD_S, (* s or S: string *)
    TYPEFIELD_XU, (* X: hecadecimal number (upper case characters )*)
    TYPEFIELD_XL, (* x: hecadecimal number (lower case characters )*)
    TYPEFIELD_EU, (* E: float number ( scientific format ) *)
    TYPEFIELD_EL (* e: float number ( scientific format ) *)
);
END_TYPE
```

值	含义
TYPEFIELD_UNKNOWN	未知或未初始化
TYPEFIELD_B	b 或 B：二进制数
TYPEFIELD_O	o 或 O：八进制数
TYPEFIELD_U	u 或 U：无符号十进制数
TYPEFIELD_C	c 或 C：ASCII 字符
TYPEFIELD_F	f 或 F：浮点数（规范化表示法）
TYPEFIELD_D	d 或 D：有符号十进制数
TYPEFIELD_S	s 或 S：字符串
TYPEFIELD_XU	X：十六进制数（大写字符）
TYPEFIELD_XL	x：十六进制数（小写字符）
TYPEFIELD_EU	E：浮点数（科学表示法）
TYPEFIELD_EL	e：浮点数（科学表示法）

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.21 GUID

系统 ID。

```
TYPE GUID :
STRUCT
    Data1 : DWORD;
    Data2 : WORD;
    Data3 : WORD;
    Data4 : ARRAY[0..7] OF BYTE;
END_STRUCT
END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.22 OTSTRUCT

运行时间计数器的时间格式。

```
TYPE OTSTRUCT :
STRUCT
    wWeek         : WORD;
    wDay          : WORD;
    wHour         : WORD;
    wMinute       : WORD;
    wSecond       : WORD;
    wMilliseconds : WORD;
END_STRUCT
END_TYPE
```

值	含义
wWeek	周数：0 ~ 65535
wDay	天数：0 ~ 7
wHour	小时数：0 ~ 23
wMinute	分钟数：0 ~ 59
wSecond	秒钟数：0 ~ 59
wMilliseconds	毫秒数：0 ~ 999

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.23 PROFILERSTRUCT

Profiler 功能块的状态信息。

```
TYPE PROFILERSTRUCT :
STRUCT
    LastExecTime     : DWORD;
    MinExecTime      : DWORD;
    MaxExecTime      : DWORD;
    AverageExecTime  : DWORD;
    MeasureCycle     : DWORD;
END_STRUCT
END_TYPE
```

值	含义
LastExecTime	执行时间的最新测量值，以 [µs] 为单位。
MinExecTime	最短执行时间，以 [µs] 为单位。
MaxExecTime	最长执行时间，以 [µs] 为单位。
AverageExecTime	最近 10 次测量的平均执行时间，以 [µs] 为单位。

值	含义
MeasureCycle	已执行的测量数量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.24 REMOTEPC

远程 PC 配置条目。

```
TYPE REMOTEPC :
STRUCT
    NetId : T_AmsNetId;
    Name  : STRING(31);
END_STRUCT
END_TYPE
```

值	含义
NetId	远程 PC 的网络地址（类型：T_AmsNetID）
名称	远程 PC 名称

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.25 REMOTEPCINFOSTRUCT

带有多个远程 PC 配置条目的列表（类型：REMOTEPC [▶ 332]）。

```
TYPE REMOTEPCINFOSTRUCT : ARRAY[0..99] OF REMOTEPC;
END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.26 ST_AmsRouteEntry

该数据类型包含有关远程 TwinCAT 连接配置的信息。

```
TYPE ST_AmsRouteEntry :
STRUCT
    sName      : STRING(MAX_ROUTE_NAME_LEN);
    sNetID     : T_AmsNetId;
    sAddress   : STRING(MAX_ROUTE_ADDR_LEN);
    eTransport : E_RouteTransportType;
    tTimeout   : TIME;
    dwFlags    : DWORD;
END_STRUCT
END_TYPE
```

值	含义
sName	远程 TwinCAT 系统的符号名称。该名称可以自由选择。通过一个常量限制最大字符串长度（默认值：31 个字符）。
sNetID	远程 TwinCAT 系统的网络地址（类型：T_AmsNetID）。
sAddress	与相应传输层相关的系统地址。如果将 TCP/IP 用作传输层，则会在这里指定 IP 地址。通过一个常量限制最大字符串长度（默认值：79 个字符）。

值	含义
eTransport	传输 AMS 消息所使用的传输层（类型：E_RouteTransportType [▶ 329]）。目前仅支持 TCP/IP 传输层。
tTimeout	超时时间。（当前已预留且未使用）。
dwFlags	附加选项（当前已预留且未使用）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.27 ST_AmsRouteEntryEx

该数据类型包含有关远程 TwinCAT 连接配置的信息。

```
TYPE ST_AmsRouteEntryEx :
STRUCT
    sName          : STRING(MAX_ROUTE_NAME_LEN);
    sNetID         : T_AmsNetId;
    sAddress       : STRING(MAX_ROUTE_ADDR_LEN);
    sVirtualNetID  : T_AmsNetId;
    eTransport     : E_RouteTransportType;
    tTimeout       : TIME;
    dwFlags        : DWORD;
END_STRUCT
END_TYPE
```

值	含义
sName	远程 TwinCAT 系统的符号名称。该名称可以自由选择。通过一个常量限制最大字符串长度（默认值：31 个字符）。
sNetID	远程 TwinCAT 系统的网络地址（类型：T_AmsNetID）。
sAddress	与相应传输层相关的系统地址。如果将 TCP/IP 用作传输层，则会在这里指定 IP 地址。通过一个常量限制最大字符串长度（默认值：79 个字符）。
sVirtualNetID	虚拟网络地址（类型：T_AmsNetID）另请参见 AmsNAT 功能的相关描述。
eTransport	传输 AMS 消息所使用的传输层（类型：E_RouteTransportType [▶ 329]）。目前仅支持 TCP/IP 传输层。
tTimeout	超时时间。（当前已预留且未使用）。
dwFlags	附加选项（当前已预留且未使用）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= v3.3.41.0

5.28 ST_CheckLicense

带有授权信息的结构

```
TYPE ST_CheckLicense :
STRUCT
    stLicenseId     : GUID;
    tExpirationTime : TIMESTRUCT;
    sExpirationTime : STRING(80);
    eResult         : E_LicenseHResult;
    nCount          : UDINT;
END_STRUCT
END_TYPE
```

名称	描述
stLicenseId	授权 ID
tExpirationTime	到期日期

名称	描述
sExpirationTime	到期日期
eResult	授权状态（请参见 E_LicenseHResult [▶ 326]）
nCount	该授权的实例数（0=无限制）

5.29 ST_DeviceIdentification

```
TYPE ST_DeviceIdentification :
STRUCT
    strTargetType       : STRING(30);
    strHardwareModel    : STRING(10);
    strHardwareSerialNo : STRING(12);
    strHardwareVersion  : STRING(8);
    strHardwareDate     : STRING(12);
    strHardwareCPU      : STRING(20);
    strImageDevice      : STRING(48);
    strImageVersion     : STRING(32);
    strImageLevel       : STRING(32);
    strImageOsName      : STRING(48);
    strImageOsVersion   : STRING(8);
    strTwinCATVersion   : STRING(4);
    strTwinCATRevision  : STRING(4);
    strTwinCATBuild     : STRING(8);
    strTwinCATLevel     : STRING(20);
    strAmsNetId         : T_AmsNetId;
END_STRUCT
END_TYPE
```

值	描述
strTargetType	目标系统类型，例如“CX1000 CE”、……
strHardwareModel	硬件型号，例如“1001”。
strHardwareSerialNo	硬件序列号，例如“123”。
strHardwareVersion	硬件版本，例如“1.7”。
strHardwareDate	硬件生产日期，例如“18.8.06”。
strHardwareCPU	硬件 CPU 架构，例如“INTELx86”、“ARM”、“UNKNOWN”或 ''（空字符串）。
strImageDevice	软件平台，例如“CX1000”、……
strImageVersion	软件平台版本，例如“2.15”。
strImageLevel	软件平台级别，例如“HMI”。
strImageOsName	操作系统的名称，例如“Windows CE”。
strImageOsVersion	操作系统版本，例如“5.0”。
strTwinCATVersion	TwinCAT 版本，例如，对于 TwinCAT 2.10.1307：“2”。
strTwinCATRevision	TwinCAT 修订版，例如，对于 TwinCAT 2.10.1307：“10”。
strTwinCATBuild	TwinCAT 版本，例如，对于 TwinCAT 2.10.1307：“1307”。
strTwinCATLevel	注册的 TwinCAT 级别，例如，“PLC”、“NC-PTP”、“NC-I”、……
strAmsNetId	TwinCAT AMS-NetID，例如“5.0.252.31.1.1”。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.30 ST_DeviceIdentificationEx

```
TYPE ST_DeviceIdentificationEx :
STRUCT
    strTargetType       : STRING(30);
    strHardwareModel    : STRING(16);
    strHardwareSerialNo : STRING(16);
    strHardwareVersion  : STRING(8);
    strHardwareDate     : STRING(12);
    strHardwareCPU      : STRING(20);
    strImageDevice      : STRING(48);
    strImageVersion     : STRING(32);
```

```
    strImageLevel       : STRING(32);
    strImageOsName      : STRING(48);
    strImageOsVersion   : STRING(8);
    strTwinCATVersion   : STRING(4);
    strTwinCATRevision  : STRING(4);
    strTwinCATBuild     : STRING(8);
    strTwinCATLevel     : STRING(20);
    strAmsNetId         : T_AmsNetId;
END_STRUCT
END_TYPE
```

值	含义
strTargetType	目标系统类型，例如“CX1000 CE”、……
strHardwareModel	硬件型号，例如“1001”。
strHardwareSerialNo	硬件序列号，例如“123”。
strHardwareVersion	硬件版本，例如“1.7”。
strHardwareDate	硬件生产日期，例如“18.8.06”。
strHardwareCPU	硬件 CPU 架构，例如“INTELx86”、“ARM”、“UNKNOWN”或 ''（空字符串）。
strImageDevice	软件平台，例如“CX1000”、……
strImageVersion	软件平台版本，例如“2.15”。
strImageLevel	软件平台级别，例如“HMI”。
strImageOsName	操作系统的名称，例如“Windows CE”。
strImageOsVersion	操作系统版本，例如“5.0”。
strTwinCATVersion	TwinCAT 版本，例如，对于 TwinCAT 2.10.1307：“2”。
strTwinCATRevision	TwinCAT 修订版，例如，对于 TwinCAT 2.10.1307：“10”。
strTwinCATBuild	TwinCAT 版本，例如，对于 TwinCAT 2.10.1307：“1307”。
strTwinCATLevel	注册的 TwinCAT 级别，例如，“PLC”、“NC-PTP”、“NC-I”、……
strAmsNetId	TwinCAT AMS-NetID，例如“5.0.252.31.1.1”。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.31 ST_FileAttributes

文件或目录属性。

```
TYPE ST_FileAttributes :
STRUCT
    bReadOnly           : BOOL;(* FILE_ATTRIBUTE_READONLY *)
    bHidden             : BOOL;(* FILE_ATTRIBUTE_HIDDEN *)
    bSystem             : BOOL;(* FILE_ATTRIBUTE_SYSTEM *)
    bDirectory          : BOOL;(* FILE_ATTRIBUTE_DIRECTORY *)
    bArchive            : BOOL;(* FILE_ATTRIBUTE_ARCHIVE *)
    bDevice             : BOOL;
(* FILE_ATTRIBUTE_DEVICE. Under CE: FILE_ATTRIBUTE_INROM or FILE_ATTRIBUTE_ENCRYPTED *)
    bNormal             : BOOL;(* FILE_ATTRIBUTE_NORMAL *)
    bTemporary          : BOOL;(* FILE_ATTRIBUTE_TEMPORARY *)
    bSparseFile         : BOOL;(* FILE_ATTRIBUTE_SPARSE_FILE *)
    bReparsePoint       : BOOL;(* FILE_ATTRIBUTE_REPARSE_POINT *)
    bCompressed         : BOOL;(* FILE_ATTRIBUTE_COMPRESSED *)
    bOffline            : BOOL;(* FILE_ATTRIBUTE_OFFLINE. Under CE: FILE_ATTRIBUTE_ROMSTATICREF *)
    bNotContentIndexed  : BOOL;
(* FILE_ATTRIBUTE_NOT_CONTENT_INDEXED. Under CE: FILE_ATTRIBUTE_ROMMODULE *)
    bEncrypted          : BOOL;(* FILE_ATTRIBUTE_ENCRYPTED *)
END_STRUCT
END_TYPE
```

值	含义
bReadOnly	文件或目录为只读。应用程序可以读取文件，但无法写入或删除文件。对于目录，应用程序无法删除它们。
bHidden	文件或目录被隐藏，不会在标准列表中显示。
bSystem	文件或目录是操作系统的一部分，或由操作系统专用。

值	含义
bDirectory	该属性可以用于识别目录。
bArchive	文件或目录属于存档。应用程序使用该属性来标记文件，以便进行备份或删除。
bDevice	预留
bNormal	文件或目录没有设置其他属性。该属性仅在专用时才有效。
bTemporary	文件仅用于临时存储数据。
bSparseFile	文件是一个稀疏文件。
bReparsePoint	一个“重解析点”与文件或目录相关联。
bCompressed	文件或目录已被压缩。文件包含已压缩的数据。对于目录，默认情况下会对新的文件或子目录进行压缩。
bOffline	文件并非始终可用。
bNotContentIndexed	文件未被索引服务编入索引。
bEncrypted	文件或目录已加密。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.32 ST_FileRBufferHead

环形缓冲区文件头状态。功能块 FB_FileRingBuffer [▶ 59] 使用该结构。在打开环形缓冲区文件时读取该结构，并在关闭时将其保存在环形缓冲区文件中。在读取/写入数据集时，始终会更新该结构。

```
TYPE ST_FileRBufferHead :
STRUCT
    status   : DWORD := 0;(* buffer status flags Bit 0 = 1 => Opened, Bit 0 = 0 => Closed, Bit 1 = 1
file corrupted, all other bits are reserved *)
    access   : UDINT := 0;(* access counter, increments every time the buffer is reopened *)
    nID      : UDINT := 0;(* user defined value *)
    cbBuffer : UDINT := 16#100000;(* max. buffer size (1MB) *)
    nCount   : UDINT := 0;(* number of fifo entries *)
    cbSize   : UDINT := 0;(* current (used) file buffer data byte length *)
    ptrFirst : UDINT := 0;(* seek pointer start position of first (oldest) buffer entry *)
    ptrLast  : UDINT := 0;(* seek pointer end position of last (newest) buffer entry *)
    rsrv0    : UDINT := 0;(* reserved *)
    rsrv1    : UDINT := 0;(* reserved *)
    rsrv2    : UDINT := 0;(* reserved *)
    rsrv3    : UDINT := 0;(* reserved *)
END_STRUCT
END_TYPE
```

值	含义
status	状态标志。位 0 = 1 => 文件打开，位 0 = 0 => 文件关闭。位 1 = 1 => 文件损坏（没有正确关闭，或最大缓冲区大小不匹配）。
acces	访问计数器。每次打开文件时，该计数器都会递增。
nID	用户定义的 32 位值。
cbBuffer	最大环形缓冲区文件大小。
nCount	当前存储的数据集的数量。
cbSize	当前存储的数据字节数。
ptrFirst	最旧数据集的文件指针位置。
ptrLast	最新数据集的文件指针位置。
rsrv0..rsrv3	预留

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.33 ST_FindFileEntry

文件搜索功能块 FB_EnumFindFileEntry [▶ 51] 和 FB_EnumFindFileList [▶ 52] 使用该数据类型。

```
TYPE ST_FindFileEntry :
STRUCT
    sFileName          : T_MaxString;
    sAlternateFileName : STRING(13);
    fileAttributes     : ST_FileAttributes;
    fileSize           : T_ULARGE_INTEGER;
    creationTime       : T_FILETIME;
    lastAccessTime     : T_FILETIME;
    lastWriteTime      : T_FILETIME;
END_STRUCT
END_TYPE
```

值	含义
sFileName	以空字符结尾的字符串，带有文件或目录的名称（类型：T_MaxString）。
sAlternateFileName	以空字符结尾的字符串，表示文件或目录的替代名称，采用常规 8.3 格式（filename.ext）。
fileAttributes	带有文件/目录属性的结构（类型：ST_FileAttributes [▶ 335]）。
fileSize	文件的字节大小（64 位数字，类型：T_ULARGE_INTEGER [▶ 347]）。
creationTime	结构变量表示创建文件或目录的时间（类型：T_FILETIME [▶ 344]）。
lastAccessTime	对于文件，该结构表示最后读取或写入文件的时间（类型：T_FILETIME [▶ 344]）。对于目录，该结构表示创建它的时间。
lastWriteTime	对于文件，该结构表示最后写入访问的时间（类型：T_FILETIME [▶ 344]）。对于目录，该结构表示创建它的时间。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.34 ST_IPAdapterHwAddr

物理地址（MAC）。

```
TYPE ST_IPAdapterHwAddr :
STRUCT
    length : UDINT := 0;
    b      : ARRAY[0..MAX_ADAPTER_ADDRESS_LENGTH] OF BYTE;
END_STRUCT
END_TYPE
```

值	含义
length	物理硬件地址的字节长度。
b	MAC 地址字节。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.35 ST_IPAdapterInfo

网络适配器信息。

```
TYPE ST_IPAdapterInfo :
STRUCT
    bDefault      : BOOL;
    sAdapterName  : STRING(MAX_ADAPTER_NAME_LENGTH) := '';
    sDescription  : STRING(MAX_ADAPTER_DESCRIPTION_LENGTH) := '';
    physAddr      : ST_IPAdapterHwAddr;
```

```
    dwIndex       : DWORD;
    eType         : E_MIB_IF_Type;
    sIpAddr       : T_IPv4Addr;
    sSubNet       : T_IPv4Addr;
    sDefGateway   : T_IPv4Addr;
    bDhcpEnabled  : BOOL;
    sDhcpSrv      : T_IPv4Addr;
    bHaveWins     : BOOL;
    sPrimWinsSrv  : T_IPv4Addr;
    sSecWinsSrv   : T_IPv4Addr;
    tLeaseObt     : DT;
    tLeaseExp     : DT;
END_STRUCT
END_TYPE
```

值	含义
bDefault	目前，仅可在 Windows CE 下使用该变量！如果为 TRUE，则 TwinCAT 会将网络适配器用作默认适配器。
sAdapterName	字符串形式的适配器名称。
sDescription	字符串形式的适配器描述。
physAddr	物理硬件地址。（类型：ST_IPAdapterHwAddr [▶ 337]）
dwIndex	内部适配器系统索引。
eType	适配器类型（类型：E_MIB_IF_Type [▶ 327]）。
sIpAddr	IP 地址（类型：T_Ipv4Addr）。
sSubNet	IP 网络掩码（类型：T_Ipv4Addr）。
sDefGateway	默认网关的 IP 地址（类型：T_Ipv4Addr）。
bDhcpEnabled	表示是否已为该适配器激活 DHCP。
sDhcpSrv	DHCP 服务器的 IP 地址（类型：T_Ipv4Addr）。
bHaveWins	表示是否使用 Windows 互联网名称服务（WINS）。
sPrimWinsSrv	主 WINS 服务器的 IP 地址（类型：T_Ipv4Addr）。
sSecWinsSrv	：辅助 WINS 服务器的 IP 地址（类型：T_Ipv4Addr）。
tLeaseObt	表示 DHCP 服务器“租用”IP 地址的时间（UTC）。
tLeaseExp	表示在 DHCP 服务器需要请求“延期”之前，DHCP 服务器可以“租用”IP 地址多长时间（UTC）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.36 ST_LicenseDongle

带有所有连接授权加密狗的识别数据的结构

```
TYPE ST_LicenseDongle :
STRUCT
    stAmsAddr     : AMSADDR;
    eDevType      : E_LDevType;
    nFlags        : UDINT;
    eDongleStatus : E_LDongleStatus;
    nSerialNo     : UDINT;
    nReserved1    : UDINT;
    nReserved2    : UDINT;
END_STRUCT
END_TYPE
```

名称	描述
stAmsAddr	授权加密狗的网络 ID（AmsNetId 和端口）
eDevType	请参见 E_LDevType [▶ 325]
nFlags	0：静态配置

名称	描述
	1：动态加密狗
eDongleStatus	授权加密狗的验证状态（请参见 E_LDongleStatus） [▶ 326]
nSerialNo	加密狗/授权密钥端子模块的 ID 编号
nReserved1	留待将来使用
nReserved2	留待将来使用

5.37 ST_ReadEvent

通过 FB_AdsReadEvents [▶ 39] 读取的关于消息的信息。

```
TYPE ST_ReadEvent :
STRUCT
    nSourceId       : UDINT;
    nEventId        : UDINT;
    nClass          : DWORD;
    nConfirmState   : DWORD;
    nResetState     : DWORD;
    sSource         : STRING(255);
    sDate           : STRING(23);
    sTime           : STRING(23);
    sComputer       : STRING;
    sMessageText    : STRING(255);
    bQuitMessage    : BOOL;
    bConfirmable    : BOOL;
END_STRUCT
END_TYPE
```

值	含义
nSourceId	
nEventId	
nClass	
nConfirmState	
nResetState	
sSource	
sDate	
sTime	
sComputer	
sMessageText	
bQuitMessage	
bConfirmable	

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.38 ST_SplittedBIC

使用倍福识别码（BIC），通过 Tc2_Utilities 的 F_SplitBIC [▶ 290] 函数可以填充 ST_SplittedBIC 结构。在结构元素中，用于分解 BIC 的子字符串的标识符和数据大小作为注释提供。

```
TYPE ST_SplittedBIC :
STRUCT
    sItemNo : STRING(6);  //1, '1P' 8(2+6)
    sBTN : STRING(8); //2, 'SBTN' 12(4+8)
    sDescription : STRING(30); //3, '1K' 32(2+30)
    sQuantity : STRING(5); //4, 'Q' 6(1+5)
    sBatchNo : STRING(12); //5, '2P' 14(2+12)
    sIdSerialNo : STRING(9); //6, '51S' 12(3+9)
    sVariantNo : STRING(9); //7, '30P' 12(3+9)
    sDataCode : STRING(6); //8, '9D' 8(2+6)
    sOrderBatchNo : STRING(12); //9, '1T' 14(2+12)
    sSerialNo : STRING(20); //10, '52S' 23(3+20)
```

```
    sPackUnitQuantity : STRING(5); //11, '4QPU' 9(4+5)
    sDataElement : STRING(11); //12, '6D' 13(2+11)
    sMoistSensLevel : STRING(4); //13, 'Z' 5(1+4)
    sPlatingMaterial : STRING(2); //14, 'E' 3(1+2)
    sManufacturer : STRING(9); //15, '12V' 12(3+9)
    sCountryOfOrigin : STRING(2); //16, '4L' 4(2+2)
    sCustomSpecItemNo : STRING(16); //17, 'P' 17(1+16)
    sUndefined : STRING(819); //remaining undefined BIC string (1023 + /0)
END_STRUCT
END_TYPE
```

名称	描述
sItemNo	项目编号，最大长度为 6 个字符，例如“072222”
sBTN	倍福可追溯性编号（BTN），最大长度为 8 个字符，例如“k4p562d7”
sDescription	描述，最大长度为 30 个字符，例如“KL9010”
sQuantity	数量，最大长度为 5 个字符，例如“1”
sBatchNo	批号，最大长度为 12 个字符，例如“401503180016”
sIdSerialNo	ID/序列号，最大长度为 9 个字符，例如“678294104”
sVariantNo	变体，最大长度为 9 个字符，例如“000048443”
sDataCode	供应商特定的数据代码，最大长度为 6 个字符，例如“G0118”
sOrderBatchNo	供应商特定的订单/批号，最大长度为 12 个字符，例如“FA12345678”
sSerialNo	供应商特定的序列号，最大长度为 20 个字符，例如“2304853-1-004”
sPackUnitQuantity	包装单位数量，最大长度为 5 个字符，例如“1000”
sDataElement	日期代码 YYYYMMDD，指定内容，最大长度为 11 个字符，例如“20190315146”
sMoistSensLevel	MS 干燥级别，最大长度为 4 个字符，例如“MSL3”
sPlatingMaterial	涂层材料（根据 JEDEC J-STD-609 标准），最大长度为 2 个字符，例如“e1”
sManufacturer	制造商（DUNS 编号），最大长度为 9 个字符，例如“313291892”
sCountryOfOrigin	原产国，最大长度为 2 个字符，例如“CN”
sCustomSpecItemNo	客户特定的货号，最大长度为 16 个字符，例如“0000000107353052”
sUndefined	无法识别的 BIC 剩余字符串（最大长度为 819 个字符），例如，在通过附加元素扩展 BIC 后，在必要时检查 Tc2_Utilities 库的新版本是否可用

5.39 ST_TcOnlineLicenseInfoDataEx

```
TYPE ST_TcOnlineLicenseInfoDataEx :
STRUCT
    stLicenseId       : GUID
    stLicenseName     : STRING(80);
    tExpirationTime   : TIMESTRUCT;
    sExpirtaionTime   : STRING(80);
    nMaxCount         : UDINT;
    nUsedCount        : UDINT;
    eResult           : E_LicenseHResult;
    nVolumeNo         : UDINT;
    nOptInfo          : WORD;
    nRestriction      : WORD;
    bOemLicense       : BOOL;
    bBeckhoffLicense  : BOOL;
    bBeckhoffPC       : BOOL;
    bEtherCATDongle   : BOOL;
    bUSBDongle        : BOOL;
    bGenDevTypeLic    : BOOL;
END_STRUCT
END_TYPE
```

名称	描述
stLicenseId	授权 ID
stLicenseName	授权名称
tExpirationTime	到期日期
sExpirationTime	到期日期
nMaxCount	此授权可以使用的最大实例数（0=无限制）
nUsedCount	使用的授权实例数

名称	描述
eResult	授权状态（请参见 E_LicenseHResult [▶ 326]）
nVolumeNo	（仅适用于批量授权）：批量许可包编号 0：无许可包编号
nOptInfo	（内部）
nRestriction	授权有使用限制
bOemLicense	OEM 授权
bBeckhoffLicense	倍福授权
bBeckhoffPC	倍福硬件（IPC、CX 等）的授权
bEtherCATDongle	授权适用于 EtherCAT 授权密钥端子模块，例如 EL6070
bUSBDongle	授权适用于 USB 授权加密狗，例如 C9900-L100
bGenDevTypeLic	授权适用于设备类型

5.40 ST_TcOnlineLicensesInfoData

授权信息。

```
TYPE ST_TimeZoneInformation :
STRUCT
    stLicenseId     : GUID;
    sLicenseName    : STRING(80);
    tExpirationTime : TIMESTRUCT;
    sExpirationTime : STRING(80);
    nMaxCount       : UDINT;
    nUsedCount      : UDINT;
    eResult         : E_LicenseHResult;
END_STRUCT
END_TYPE
```

值	含义
stLicenseId	将授权 ID 定义为 GUID [▶ 331]。
sLicenseName	字符串形式的授权名称。
tExpirationTime	表示授权的到期时间（类型：TIMESTRUCT [▶ 347]）。
sExpirationTime	表示字符串形式的授权到期时间。
nMaxCount	如果相关授权包含实例限制，则表示允许的最大实例数（例如 TC3 NC PTP Axes Pack 25）。
nUsedCount	如果相关授权包含实例限制，则表示使用的实例数。
eResult	以 HResult 枚举的形式输出该授权的错误代码。（在这里，错误用负值表示）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0.4018	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）v3.3.9.0 或更高

5.41 ST_TcRouterStatusInfo

TwinCAT 路由器状态信息。

```
TYPE ST_TcRouterStatusInfo :
STRUCT
    maxMem      : DWORD;(* Max. router memory byte size *)
    maxMemAvail : DWORD;(* Available router memory byte size *)
    regPorts    : DWORD;(* Number of registered ports *)
    regDrivers  : DWORD;(* Number of registered TwinCAT server ports *)
    amsDebugLog : BOOL;(* TRUE = Ams logging/debugging enabled, FALSE = Ams logging/
debugging disabled *)
END_STRUCT
END_TYPE
```

值	含义
maxMem	路由器内存的最大字节大小
maxMemAvail	可用路由器内存字节大小
regPorts	注册端口号
regDrivers	注册 TwinCAT 服务器端口号
amsDebugLog	TRUE = 已启用 AMS 日志记录/调试，FALSE = 已禁用 AMS 日志记录/调试

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.42 ST_TimeZoneInformation

时区信息。标准时间也被称为冬令时。偏差参数也可以取负值。

```
TYPE ST_TimeZoneInformation :
STRUCT
    bias         : DINT
    standardName : STRING(31);
    standardDate : TIMESTRUCT;
    standardBias : DINT;
    daylightName : STRING(31);
    daylightDate : TIMESTRUCT;
    daylightBias : DINT;
END_STRUCT
END_TYPE
```

值	含义
bias	定义本地时间与 UTC 时间的当前差值，以分钟为单位。UTC = 本地时间 + 偏差。
standardName	字符串形式的标准时间名称。
standardDate	该结构包含有关从夏令时过渡到标准时间的信息（类型：TIMESTRUCT [▶ 347]）。如果不使用该值，则结构参数 *wMonth* 为零。如果使用该参数，则还必须使用 *daylightDate* 参数。为了能够配置 *standardDate*，应将 *wYear* 参数设置为零，为 *wDayOfWeek* 选择所需的星期几，并为 *wDay* 选择一个介于 1 和 5 之间的值（每月的周数，5 为最后一周）。
standardBias	以分钟为单位的时间差，用于计算标准时间中的本地时间。该值通常为零。
daylightName	字符串形式的夏令时名称。
daylightDate	该结构包含有关从标准时间过渡到夏令时的信息（类型：TIMESTRUCT [▶ 347]）。如果不使用该值，则结构参数 *wMonth* 为零。如果使用该参数，则还必须使用 *standardDate* 参数。为了能够配置 *daylightDate*，应将 *wYear* 参数设置为零，为 *wDayOfWeek* 选择所需的星期几，并为 *wDay* 选择一个介于 1 和 5 之间的值（每月的周数，5 为最后一周）。
daylightBias	以分钟为单位的时间差，用于计算夏令时中的本地时间。

示例：

请参见：FB_SetTimeZoneInformation [▶ 109]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.43

5.44 SYMINFOSTRUCT

TwinCAT PLC 符号信息。

```
TYPE SYMINFOSTRUCT :
STRUCT
    symEntryLen  : UDINT;
    idxGroup     : UDINT;
    idxOffset    : UDINT;
    byteSize     : UDINT;
    adsDataType  : ADSDATATYPEID;
    symDataType  : T_MaxString;
    symComment   : T_MaxString;
END_STRUCT
END_TYPE
```

值	含义
symEntryLen	符号表中的符号条目的实际长度，以字节为单位。符号存储在符号表中。各个条目的长度是变量，取决于符号名称、类型名称和注释的长度。
idxGroup	符号变量的索引组
idxOffset	符号变量的索引偏移量
byteSize	符号变量的值实际占用的内存量，以字节为单位。例如，一个布尔 PLC 变量占用 1 字节，而一个带有 20 个字符的字符串实际上占用 21 字节（用于字符的 20 字节加上用于标记字符串末尾的空值的 1 字节）。
adsDataType	ADS 数据类型 ID。（类型：ADSDATATYPEID [▶ 322]）该类型名称可用于 ADS 访问符号变量。所有 PLC 结构和数组（用户定义的数据类型）的 ADS 数据类型名称均为 ADST_BIGTYPE，且无法通过该数据类型常量进行识别。为了能够识别用户定义的数据类型，可以使用 *symDataType* 变量，或者读取结构中的各个变量的基本类型。
symDataType	字符串形式的符号变量的数据类型名称。例如，用户定义的 PLC 数据结构的类型名称（类型：T_MaxString，最多 255 个字符）。
symComment	用户添加到 PLC 变量定义行的符号变量的注释（类型：T_MaxString，最多 255 个字符）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.45 T_Arg

字符串格式函数/功能块的参数类型。

```
TYPE T_Arg :
STRUCT
    eType : E_ArgType := ARGTYPE_UNKNOWN;
    cbLen : UDINT     := 0;
    pData : PVOID     := 0;
END_STRUCT
END_TYPE
```

值	含义
eType	数据类型标识符（类型：E_ArgType [▶ 323]）。
cbLen	在内存中分配的字节数。
pData	地址指针

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.46 T_FILETIME

该类型的变量为 64 位数字。该值与自 1601 年 1 月 1 日（UTC）以来的 100 纳秒间隔数相对应。

```
TYPE T_FILETIME :
STRUCT
    dwLowDateTime  : DWORD;
    dwHighDateTime : DWORD;
END_STRUCT
END_TYPE
```

值	含义
dwLowDateTime	低 32 位
dwHighDateTime	高 32 位

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.47 T_FILETIME64

该类型的变量为 64 位数字。该值与自 1601 年 1 月 1 日（通常为 UTC）以来的 100 纳秒间隔数相对应。

```
TYPE T_FILETIME64 : ULINT; END_TYPE
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.4024	PC 或 CX（x86、x64、Arm®）	Tc2_Utilities（系统） >= 3.3.44.0

5.48 T_FIX16

该类型的变量表示有符号的 16 位定点数。通常，没有 FPU 单元的系统会使用该数据类型（例如：微控制器或遥测领域中的设备）。例如，如果要通过串行接口传输定点数格式的数据，则必须将这些数据转换为合适的格式。

根据所需的数字范围和分辨率，选择小数位数。对于 15 个小数位的情况，可以显示以下范围内的定点数：-1..1-2^15。这大致相当于浮点数范围：-1..0.999969482421875。

与浮点数不同，定点数的分辨率在整个数字范围内是恒定的。遗憾的是，定点数的数字显示范围较小。对于可能产生正溢出或负溢出的数学运算，必须小心处理。

```
TYPE T_FIX16 :
STRUCT
    value  : INT := 0;
    n      : WORD(0..15);
    status : DWORD := 0;
END_STRUCT
END_TYPE
```

值	含义
value	该成员变量包含定点数的实际值（小数点前后各 16 位）。
n	小数位数。允许的范围：0..15。高阶位留给符号位。
status	状态标志（已保留，当前未使用）。

示例 1：

A/D-C 提供的值为有符号的 16 位定点数，带有 15 个小数位。这些测量值被导入 PLC 后，应转换为 LREAL 数据类型。

```
PROGRAM FIX_TO_FLOAT
VAR
    adc_0        : WORD := 2#1010000000000000;(* = -0.75 (Q0.15) *)
    adc_1        : WORD := 2#0111000000000000;(* = +0.875 (Q0.15) *)
```

```
    fix_0, fix_1 : T_FIX16;
    dbl_0, dbl_1 : LREAL;
END_VAR
```

```
fix_0 := WORD_TO_FIX16( adc_0, 15 );
fix_1 := WORD_TO_FIX16( adc_1, 15 );
dbl_0 := FIX16_TO_LREAL( fix_0 );
dbl_1 := FIX16_TO_LREAL( fix_1 );
```

示例 2：

微型控制器的参数为有符号的 16 位定点数，带有 8 个小数位。PLC 中的 LREAL 参数应转换为该格式。

```
PROGRAM FLOAT_TO_FIX
VAR
    dbl_0          : LREAL := +3.5;
    dbl_1          : LREAL := -3.5;
    fix_0, fix_1   : T_FIX16;
    ctrl_0, ctrl_1 : WORD;
END_VAR
```

```
fix_0  := LREAL_TO_FIX16( dbl_0, 8 );
fix_1  := LREAL_TO_FIX16( dbl_1, 8 );
ctrl_0 := FIX16_TO_WORD( fix_0 );
ctrl_1 := FIX16_TO_WORD( fix_1 );
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.49 T_HashTableEntry

哈希表条目/元素。

```
TYPE T_HashTableEntry :
STRUCT
    key   : DWORD := 0;(* Entry key *)
    value : PVOID := 0;(* Entry value *)
END_STRUCT
END_TYPE
```

值	含义
key	键（无符号 32 位数字或 32 位指针）。
value	值（可以是无符号 32/64 位数字或指针）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.50 T_HHASHTABLE

哈希表句柄。功能块 FB_HashTableCtrl [▶ 80] 使用哈希表句柄。

```
TYPE T_HHASHTABLE :
STRUCT
    nCount : UDINT := 0;
    nFree  : UDINT := 0;
END_STRUCT
END_TYPE
```

值	含义
nCount	占用元素的数量。
nFree	空闲元素的数量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.51 T_HLINKEDLIST

链表句柄。功能块 FB_LinkedListCtrl [▶ 88] 使用链表句柄。

```
TYPE T_HLINKEDLIST :
STRUCT
    nCount : UDINT := 0;
    nFree  : UDINT := 0;
END_STRUCT
END_TYPE
```

值	含义
nCount	占用元素的数量。
nFree	空闲元素的数量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.52 T_HUGE_INTEGER

该类型的变量表示 TwinCAT 2 有符号的 128 位数字（“传统”类型）。

```
TYPE T_HUGE_INTEGER :
STRUCT
   qwLowPart  : T_ULARGE_INTEGER;
   qwHighPart : T_ULARGE_INTEGER;
END_STRUCT
END_TYPE
```

值	含义
qwLowPart	低 64 位。
qwHighPart	高 64 位。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.53 T_LARGE_INTEGER

该类型的变量表示 TwinCAT 2 有符号的 64 位数字（“传统”类型）。

```
TYPE T_LARGE_INTEGER :
STRUCT
    dwLowPart  : DWORD;
    dwHighPart : DWORD;
END_STRUCT
END_TYPE
```

值	含义
dwLowPart	低 32 位。
dwHighPart	高 32 位。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.54 T_LinkedListEntry

该类型的变量表示链表的一个节点/元素。

```
TYPE T_LinkedListEntry :
STRUCT
    value : PVOID := 0;
END_STRUCT
END_TYPE
```

值	含义
value	值（可以是无符号 32/64 位数字或指针）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.55 T_UHUGE_INTEGER

该类型的变量表示 TwinCAT 2 无符号的 128 位数字（“传统”类型）。

```
TYPE T_UHUGE_INTEGER :
STRUCT
    qwLowPart  : T_ULARGE_INTEGER;
    qwHighPart : T_ULARGE_INTEGER;
END_STRUCT
END_TYPE
```

值	含义
qwLowPart	低 64 位。
qwHighPart	高 64 位。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.56 T_ULARGE_INTEGER

该类型的变量表示 TwinCAT 2 无符号的 64 位数字（“传统”类型）。

```
TYPE T_ULARGE_INTEGER :
STRUCT
    dwLowPart  : DWORD;
    dwHighPart : DWORD;
END_STRUCT
END_TYPE
```

值	含义
dwLowPart	低 32 位。
dwHighPart	高 32 位。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

5.57 TIMESTRUCT

系统时间格式的时间。

```
TYPE TIMESTRUCT
STRUCT
   wYear         : WORD;
   wMonth        : WORD;
   wDayOfWeek    : WORD;
   wDay          : WORD;
   wHour         : WORD;
   wMinute       : WORD;
   wSecond       : WORD;
   wMilliseconds : WORD;
END_STRUCT
END_TYPE
```

值	含义
wYear	年份：1970 ~ 2106
wMonth	月份：1 ~ 12（一月 = 1，二月 = 2 等）
wDayOfWeek	星期几：0 ~ 6（星期日 = 0，星期一 = 1 等）
wDay	月份中的某一天：1 ~ 31
wHour	小时：0 ~ 23
wMinute	分钟：0 ~ 59
wSecond	秒钟：0 ~ 59
wMilliseconds	毫秒：0 ~ 999

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

6 全局常量

6.1 库版本

所有库都有特定版本。例如，版本会在 PLC 功能库中显示。全局常量包含有关库版本的信息：

Global_Version

```
VAR_GLOBAL CONSTANT
    stLibVersion_Tc2_Utilities : ST_LibVersion;
END_VAR
```

值	含义
stLibVersion_Tc2_Utilities	Tc2_Utilities 库的版本号（类型：ST_LibVersion）。

要检查您拥有的版本是否是所需的版本，请使用 F_CmpLibVersion 函数（在 Tc2_System 库中定义）。

您可能从 TwinCAT 2 中了解到的用于比较库版本的所有其他选项都已过时。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

7 全局变量

VAR_GLOBAL

名称	类型	值	使用	含义
MAX_AVERAGE_MEASURES	INT	10	Profiler [▶ 133]	平均测量读数的数量。可能的值：2..100
GLOBAL_FORMAT_HASH_PREFIX_TYPE	E_HashPrefixTypes	HASHPREFIX_IEC	FB_FormatString [▶ 63]、F_FormatArgToStr [▶ 286]	二进制、八进制或十六进制格式的标准 IEC 前缀
GLOBAL_SBCS_TABLE	E_SBCSType [▶ 329]	eSBCS_WesternEuropean	F_ToLCase [▶ 291]、F_ToUCase [▶ 292]	Windows SBCS（单字节字符集）代码页表
GLOBAL_DCF77_PULSE_SPLIT	TIME	T#140ms	DCF77_TIME [▶ 32]	脉冲长度。0 == 脉冲 < 140ms，1 == 脉冲 > 140
GLOBAL_DCF77_SEQUENCE_CHECK	BOOL	FALSE	DCF77_TIME [▶ 32]	2 个连续报文的可信度检查：TRUE = 已启用，FALSE = 已禁用。
DEFAULT_CSV_FIELD_SEP	BYTE	16#2C	FB_CSVMemBufferWriter [▶ 50]、FB_CSVmemBufferReader [▶ 48]	数据字段分隔符。分号= 16#3B => 德语字段分隔符，逗号 = 16#2C => US 字段分隔符

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

8 全局参数

VAR_GLOBAL 常量

名称	类型	值	使用	含义
nMaxLicenses	UINT	50	FB_GetLicenses [▶ 72]、FB_GetLicensesEx [▶ 73]	授权的最大数量
nMaxLicenseDevices	UINT	16	FB_GetLicenseDongles [▶ 71]	连接的授权加密狗的最大数量
nMaxCpuCount	USINT	64	FB_GetRtPerformanceData	CPU 的最大数量
cMaxCharacters	UDINT	16#FFFFFFF0		扩展字符串函数将检查或转换的最大字符数。
nMaxFilesOnDongle	UDINT	20		在 USB 加密狗或 EL6070 加密狗中可以存储的最大文件数。
cMaxCSVFieldValueSize	UINT(255..4000)	255	FB_CSVMemBufferReader、FB_CSVMemBufferWritter	CSV 字段的最大大小，以字节为单位。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

还请参阅有关此

- [▶ 118]
- FB_CSVMemBufferReader [▶ 48]
- FB_CSVMemBufferWriter [▶ 50]

9 示例

9.1 示例：通信 BC/BX<->PC/CX（F_SwapRealEx）

该示例演示了 F_SwapRealEx [▶ 291] 函数的应用。该示例包含 2 个组件：TwinCAT 2.xx BC/BX（总线端子模块控制器）应用程序和 TwinCAT 3.xx PC/CX（x86）应用程序。PC/CX 应用程序从 BC/BX 的标志区域读取结构变量/将结构变量写入 BC/BX 的标志区域。结构变量包含 REAL 元素。在 PC/CX 上使用这些元素或将其传输到 BC/BX 之前，必须将它们转换为正确的格式。

您可以在这里将完整的来源解压缩：

TwinCAT 2.xx - BC/BX（总线端子模块控制器）应用程序/项目文件：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803333131.zip

TwinCAT 3.xx - PC/CX（x86、x64、ARM）应用程序/存档文件：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803336971.zip

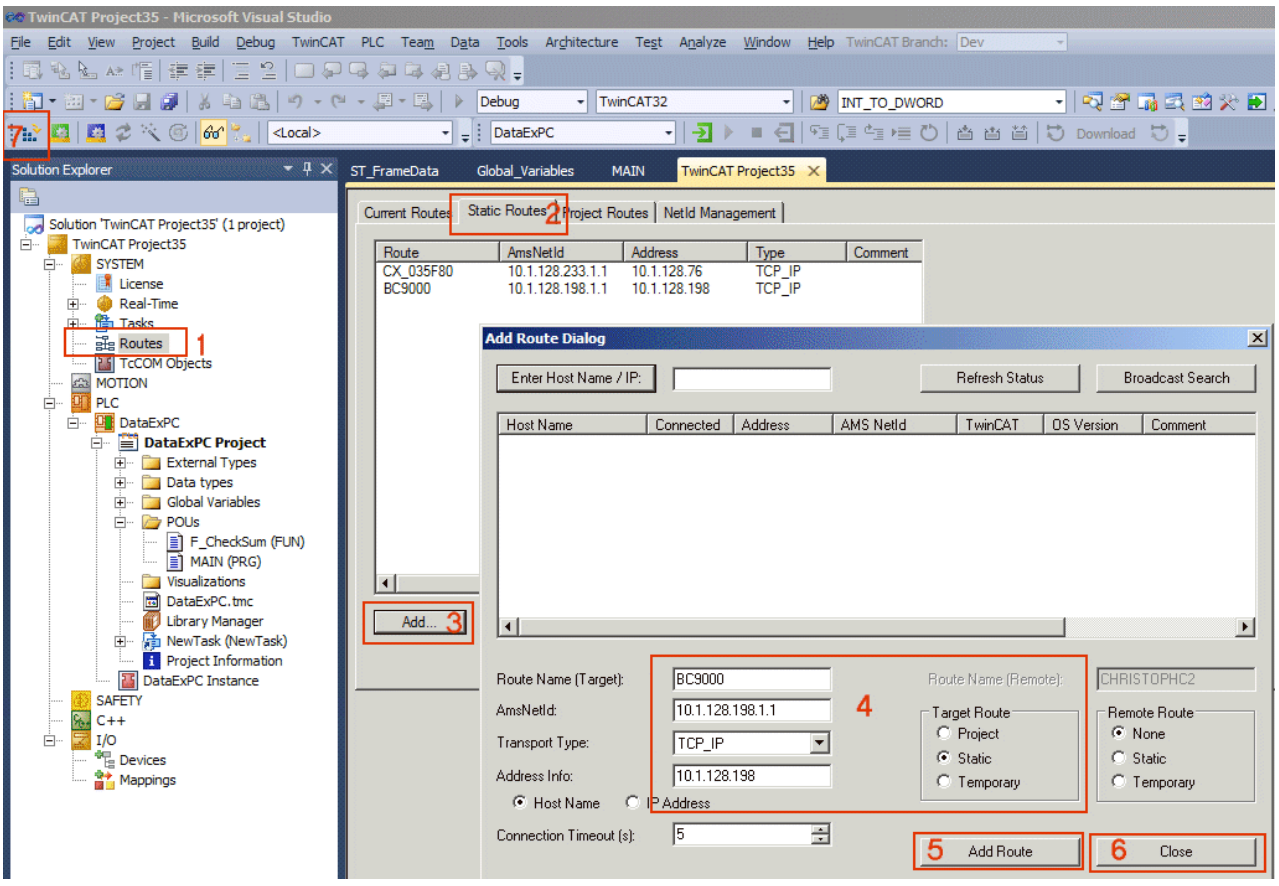
系统要求：

- TwinCAT 2.xx PLC（下载 BC/BX 应用程序所需） + BC/BX 硬件（例如 BC9000）；
- TwinCAT 3.xx 开发环境和运行时系统（下载 PC/CX 应用程序所需）；

下载项目

使用 TwinCAT 2.xx PLC 将 BC/BX 应用程序下载到总线端子模块控制器（例如 BC9000）的运行时系统中。创建一个启动项目并启动 PLC。在下一步中，使用 TwinCAT 3.xx 创建一个新的 XAE 项目。使用鼠标右键点击 *PLC* 节点，再点击 *Add existing item*（添加现有项目），可将存档文件导入 TwinCAT XAE。

要通过 ADS 访问 BC/BX（总线端子模块控制器），必须将其作为设备输入 TwinCAT AMS 路由连接（路由）列表中。创建一个新的静态路由（按图中所示的步骤操作）。务必对 BC/BX 的 AmsNetID 和 IP 地址进行相应配置（请记住 PLC 程序代码中的 AmsNetID）。



重要说明!

BC/BX（总线端子模块控制器）和 PC/CX（x86、x64、ARM）采用不同的内存对齐方式（数据对齐方式）。对于 BC/BX <-> PC/CX 数据交换，请定义采用 8 字节内存对齐方式的结构。

- TwinCAT 2.xx + PC/CX（x86）平台 => 数据结构采用 1 字节内存对齐方式；
- TwinCAT 2.xx + CX（ARM）平台 => 数据结构采用 4 字节（DWORD）内存对齐方式；
- TwinCAT 2.xx + BC/BX（总线端子模块控制器）平台 => 数据结构采用 2 字节（WORD）内存对齐方式；
- TwinCAT 3.xx + PC/CX（x86、x64、ARM）平台 => 数据结构采用 8 字节内存对齐方式；

在两个系统上使用的结构变量定义：

```
(* 8 byte aligned structure, byte size := 152 byte *)
TYPE ST_FrameData :
STRUCT
    nFrameSize: DWORD;(*Frame byte size, member byte size := 4 byte*)
    nTxFrames : DWORD;(*Tx frame number, member byte size := 4 byte*)
    nRxFrames : DWORD;(*Rx frame number, member byte size := 4 byte*)
    nCounter  : DWORD;(*Number value, member byte size := 4 byte*)
    fU        : REAL;(*Floating point number, member byte size := 4 byte*)
    fV        : REAL;(*Floating point number, member byte size := 4 byte*)
    fW        : REAL;(*Floating point number, member byte size := 4 byte*)
    aFloats   : ARRAY[0..9] OF REAL;(* Array of floating point numbers, array byte size := 40 byte*)
    sMsg      : STRING;(*String variable, member byte size := 81 byte incl. String null delimiter*)
    bEnable   : BOOL;(*Boolean flag, member byte size := 1 byte*)
    nRsv0     : BYTE;(*Reserved byte to meet the 8 byte alignment, member byte size := 1 byte*)
    nCRC      : BYTE;(*CRC checksum byte, member byte size := 1 byte*)
END_STRUCT
END_TYPE
```

BC/BX（总线端子模块控制器）应用程序

每次从 PC/CX 进行写入访问后，都会检查数据长度和校验和。然后会生成用于读取访问的全新随机值，这些值也与简单的校验和相关联。

```
PROGRAM MAIN
VAR
    stRxFrame AT%MB500   : ST_FrameData;(* Data transported from PC/CX (x86) to BC/
BX (Bus Terminal Controller) *)
    stTxFrame AT%MB0     : ST_FrameData;(* Data transported from BC/
BX (Bus Terminal Controller) to PC/CX (x86) *)
    nReceivedFrame       : UDINT;
    i                    : INT;
    nRxErrors            : UDINT;
END_VAR
```

```
(* New frame from PC/CX received? *)
IF stRxFrame.nTxFrames <> nReceivedFrame THEN
    (* Frame length OK? *)
    IF stRxFrame.nFrameSize = SIZEOF( stRxFrame) THEN
        (* Checksum OK? *)
        IF stRxFrame.nCRC = F_CheckSum( ADR( stRxFrame), SIZEOF( stRxFrame) - 1 ) THEN (* => OK *)
            (* Create/modify the tx data *)
            stTxFrame.nFrameSize  := SIZEOF( stTxFrame);(* Set frame byte size *)
            stTxFrame.nTxFrames   := stTxFrame.nTxFrames + 1;(* Increment the send frame number *)
            stTxFrame.nRxFrames   := stRxFrame.nTxFrames;(* Report the received frame number *)
            stTxFrame.bEnable     := NOT stRxFrame.bEnable;(* Toggle bool flag *)
            stTxFrame.nCounter    := stTxFrame.nCounter + 1;(* Send some counter value *)
            stTxFrame.sMsg        := CONCAT( 'Message from BC/
BX, counter:', DWORD_TO_STRING( stTxFrame.nCounter ) );(* Create any string message *)
            stTxFrame.fU          := stRxFrame.fU + 10.0;(* Modify some floating point values *)
            stTxFrame.fV          := stRxFrame.fV + 100.0;
            stTxFrame.fW          := stRxFrame.fW + 1000.0;
            FOR i:= 0 TO 9 DO
                stTxFrame.aFloats[i] := stTxFrame.aFloats[i] + i + 3.141592;
            END_FOR
            stTxFrame.nCRC        := F_CheckSum( ADR( stTxFrame), SIZEOF( stTxFrame) - 1 );
(* Create checksum *)
        ELSE(* => Checksum error *)
            nRxErrors := nRxErrors + 1;
        END_IF
    ELSE(* => Invalid frame length *)
        nRxErrors := nRxErrors + 1;
    END_IF
    nReceivedFrame := stRxFrame.nTxFrames;
END_IF
```

PC/CX（x86、x64、ARM）应用程序

bWrite 的上升沿会启动写入过程。在写入操作之前，将 REAL 元素转换为 BC/BX 格式。确定并设置数据长度和校验和。bRead 的上升沿会启动读取过程。在读取操作成功之后，将检查数据长度，然后进行简单的校验和。然后，将 REAL 元素转换为 PC/CX 格式。

```
PROGRAM MAIN
VAR
    bWrite        : BOOL;(* Rising edge at this variable writes data to the BC/
BX (Bus Terminal Controller) *)
    bRead         : BOOL;(* Rising edge at this variable reads data from BC/
BX (Bus Terminal Controller) *)
    stTxFrame     : ST_FrameData; (* Data transported from PC/CX (x86) to BC/
BX (Bus Terminal Controller) *)
    stRxFrame     : ST_FrameData; (* Data transported from BC/BX (Bus Terminal Controller) to PC/
CX (x86) *)
    fbWrite       : ADSWRITE :=  ( NETID := '172.17.61.50.1.1', PORT := 800, IDXGRP := 16#4020, IDXO
FFS := 500, TMOUT := DEFAULT_ADS_TIMEOUT );
    fbRead        : ADSREAD :=  ( NETID := '172.17.61.50.1.1', PORT := 800, IDXGRP := 16#4020, IDXOF
FS := 0, TMOUT := DEFAULT_ADS_TIMEOUT );
    (* Temporary used variables *)
    stTxToBC      : ST_FrameData;
    stRxFromBC    : ST_FrameData;
    i             : INT;
    nTxState      : UDINT;
    nRxState      : UDINT;
    nTxErrors     : UDINT;
    nRxErrors     : UDINT;
END_VAR
```

```
(*##########################################################################################*)
CASE nTxState OF
    0:
        IF bWrite THEN(* Write BC/BX data *)
            bWrite := FALSE;

            (* Prepare/modify tx data *)
            stTxFrame.nFrameSize    := SIZEOF( stTxFrame );(* Set frame byte size *)
            stTxFrame.nTxFrames     := stTxFrame.nTxFrames + 1;(* Increment the send frame number *)
            stTxFrame.nRxFrames     := stRxFrame.nTxFrames;(* Report the received frame number *)
            stTxFrame.bEnable    := NOT stTxFrame.bEnable;(* Toggle bool flag *)
            stTxFrame.nCounter      := stTxFrame.nCounter + 1;(* Increment counter value *)
            stTxFrame.sMsg          := CONCAT( 'Message from PC/
CX, counter: ', DWORD_TO_STRING( stTxFrame.nCounter ) );(* Create some string message *)
            stTxFrame.fU            := stTxFrame.fU + 1.2;(* Modify some floating point values *)
            stTxFrame.fV            := stTxFrame.fV + 3.4;
            stTxFrame.fW            := stTxFrame.fW + 5.6;
            FOR i:= 0 TO 9 DO
                stTxFrame.aFloats[i] := stTxFrame.aFloats[i] + i;
            END_FOR
            stTxFrame.nCRC          := 0;

            (* Create temporary copy of tx data *)
            stTxToBC := stTxFrame;

            (* Swap REAL variables to BC/BX (Bus Terminal Controller) format *)
            F_SwapRealEx( stTxToBC .fU );
            F_SwapRealEx( stTxToBC .fV );
            F_SwapRealEx( stTxToBC .fW );
            FOR i:= 0 TO 9 DO
                F_SwapRealEx( stTxToBC .aFloats[i] );
            END_FOR

            (* Create CRC check number *)
            stTxToBC .nCRC     := F_CheckSum( ADR( stTxToBC ), SIZEOF( stTxToBC ) - 1 );

            (* Send *)
            fbWrite( WRITE := FALSE );
            fbWrite( LEN := SIZEOF( stTxToBC ), SRCADDR := ADR( stTxToBC ), WRITE := TRUE );
            nTxState := 1;
        END_IF

    1:(* Wait until ads write command not busy *)
        fbWrite( WRITE := FALSE );
        IF NOT fbWrite.BUSY THEN
            IF NOT fbWrite.ERR THEN
                nTxState := 0;
            ELSE(* Ads error *)
                nTxState := 100;
            END_IF
        END_IF

    100: (* TODO: Error state, add error handling *)
        nTxErrors := nTxErrors + 1;
        nTxState := 0;

END_CASE

(*##########################################################################################*)
CASE nRxState OF
    0:
        IF bRead THEN(* Read BC/BX data *)
            bRead := FALSE;

            fbRead( READ := FALSE );
            fbRead( LEN := SIZEOF( stRxFromBC ), DESTADDR := ADR( stRxFromBC ), READ := TRUE );
            nRxState := 1;
        END_IF

    1:(* Wait until ads read command not busy *)
        fbRead( READ := FALSE );
```

```
        IF NOT fbRead.BUSY THEN
            IF NOT fbRead.ERR THEN
                (* Perform simple frame length check *)
                IF stRxFromBC.nFrameSize = SIZEOF( stRxFromBC ) THEN (* Check frame length *)
                    (* Perform simple CRC check *)
                    IF stRxFromBC.nCRC = F_CheckSum( ADR( stRxFromBC ), SIZEOF( stRxFromBC ) -
 1 ) THEN

                        (* Swap REAL variables to PC/CX (x86) format *)
                        F_SwapRealEx( stRxFromBC.fU );
                        F_SwapRealEx( stRxFromBC.fV );
                        F_SwapRealEx( stRxFromBC.fW );
                        FOR i:= 0 TO 9 DO
                            F_SwapRealEx( stRxFromBC.aFloats[i] );
                        END_FOR

                        stRxFrame := stRxFromBC;
                        nRxState     := 0;

                    ELSE(* => Checksum error *)
                        nRxState := 100;
                    END_IF
                ELSE(* => Invalid frame length *)
                    nRxState := 100;
                END_IF
            ELSE(* => Ads error *)
                nRxState := 100;
            END_IF
        END_IF

    100: (* TODO: Error state, add error handling *)
        nRxErrors := nRxErrors + 1;
        nRxState := 0;

END_CASE
```

应用程序测试

打开 PC/CX 应用程序，并将 TRUE 写入 bWrite 变量。在下一步中，将 TRUE 写入 bRead 变量。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.2 示例：文件搜索（FB_EnumFindFileEntry，FB_EnumFindFileList）

您可以在这里将完整的来源解压缩：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803340811.zip

示例：FB_EnumFindFileEntry（ST）

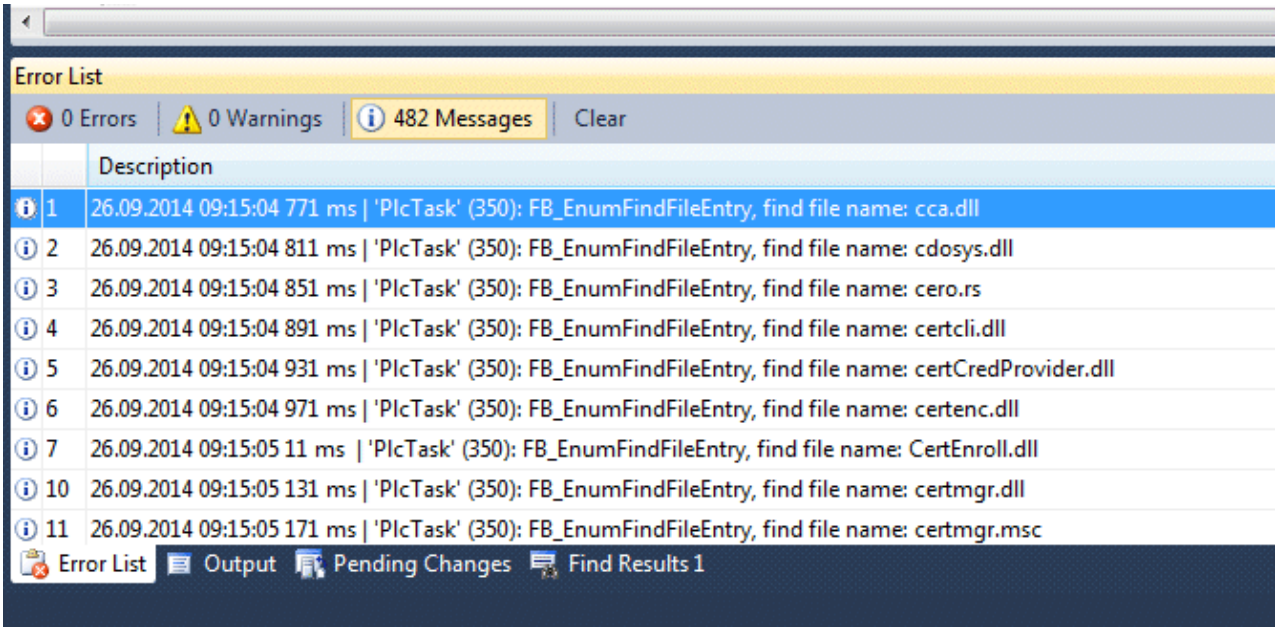
在本地 TwinCAT 系统中，所有文件均应在以下目录中列出：C:\Windows\system32\。文件名应作为消息写入 TwinCAT XAE 错误列表。应该可以取消该过程。*bEnum* 变量的上升沿会开始列出找到的文件。*bAbort* 变量的上升沿会中止该过程。

```
PROGRAM P_TestEnumEntry
VAR
    fbEnum: FB_EnumFindFileEntry := ( sNetID := '', tTimeout := T#5s, sPathName := 'C:
\Windows\System32\*.*' );
    bEnum : BOOL;
    bAbort: BOOL;
    nState: BYTE;
END_VAR
```

```
CASE nState OF
    0:
        IF bEnum THEN              (* flag set ? *)
            bEnum       := FALSE;          (* reset flag *)
            fbEnum.eCmd := eEnumCmd_First;     (* enum first entry *)
            nState      := 1;
        END_IF

    1:    (* enum one entry *)
        IF bAbort THEN
            bAbort      := FALSE;
            fbEnum.eCmd := eEnumCmd_Abort;
        END_IF
        fbEnum( bExecute := FALSE );
```

```
        fbEnum( bExecute := TRUE );
        nState     := 2;

    2:    (* wait until function block not busy *)
        fbEnum( bExecute := FALSE );
        IF NOT fbEnum.bBusy THEN
            IF NOT fbEnum.bError THEN
                IF NOT fbEnum.bEOE THEN
                    ADSLOGSTR( ADSLOG_MSGTYPE_HINT OR ADSLOG_MSGTYPE_LOG, 'FB_EnumFindFileEntry, fin
d file name: %s', fbEnum.stFindFile.sFileName );
                    fbEnum.eCmd := eEnumCmd_Next; (* enum next entry *)
                    nState      := 1;
                ELSE (* no more entries *)
                    nState      := 0;
                END_IF
            ELSE (* log error *)
                ADSLOGSTR( ADSLOG_MSGTYPE_ERROR OR ADSLOG_MSGTYPE_LOG, 'FB_EnumFindFileEntry error:
%s', DWORD_TO_HEXSTR( fbEnum.nErrID, 0, FALSE ) );
                nState     := 0;
            END_IF
        END_IF
END_CASE
```

写入 TwinCAT XAE 错误列表的日志消息：

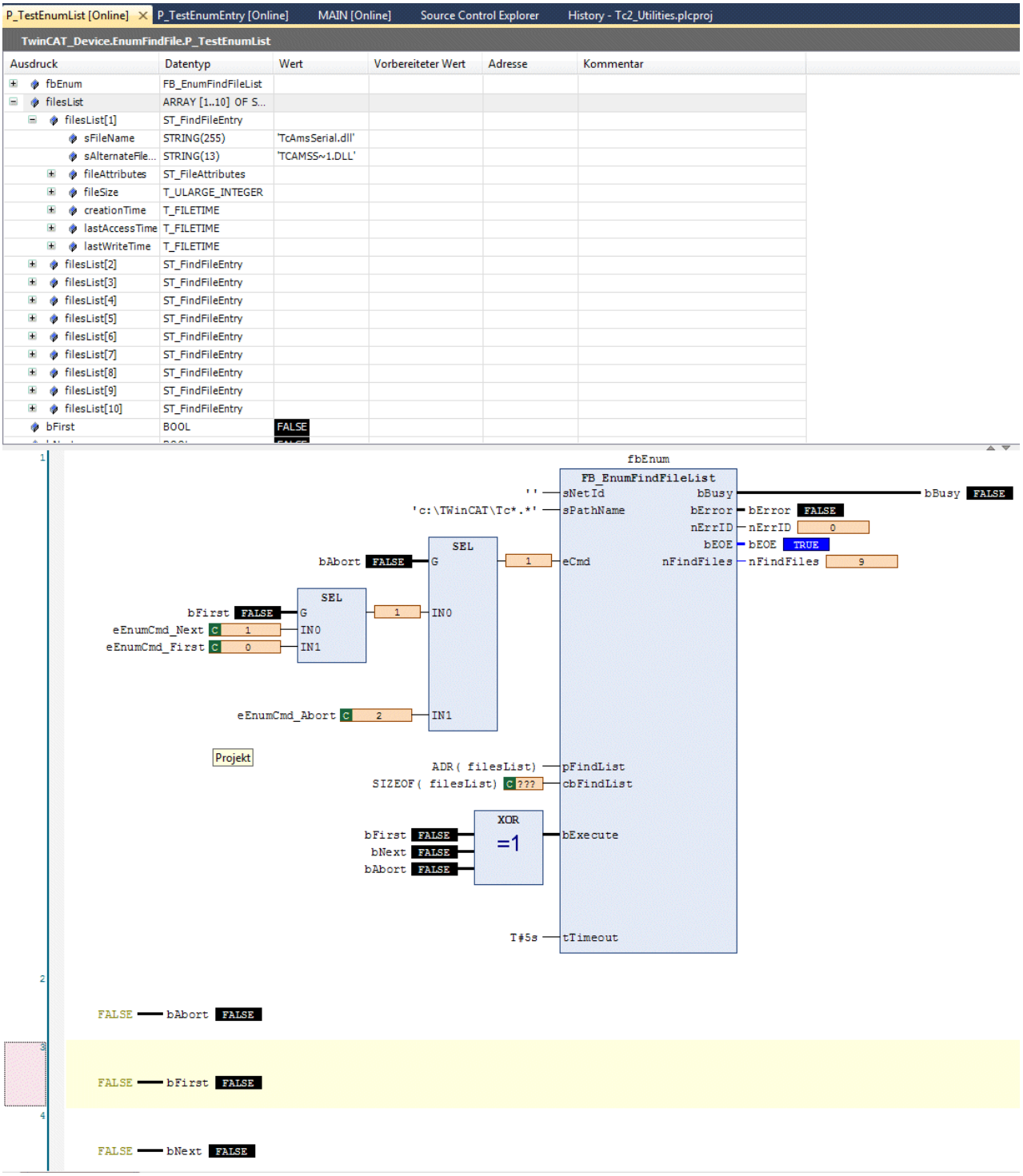


示例：FB_EnuFindFileList（FBD）

bFirst 变量的上升沿会激活该过程。如果成功，则会将文件名输入到 *fileList* 数组变量中。

```
PROGRAM P_TestEnumList
VAR
    fbEnum      : FB_EnumFindFileList;
    filesList   : ARRAY[1..10] OF ST_FindFileEntry;
    bFirst      : BOOL;
    bNext       : BOOL;
    bAbort      : BOOL;
    bBusy       : BOOL;
    bError      : BOOL;
    nErrID      : UDINT;
    bEOE        : BOOL;
    nFindFiles  : UDINT;
END_VAR
```

在线视图：



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.3 示例：文件环 FIFO（FB_FileRingBuffer）

您可以在这里找到完整的来源：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803395851.zip

下面的示例说明了功能块的简单应用。*bOpen* 的上升沿会打开一个现有的环形缓冲区文件。如果文件不存在，则会创建一个新文件。*bClose* 的上升沿会关闭一个打开的文件。*bCreate* 的上升沿会创建一个新文件。如果您设置 *bAdd* = TRUE，则会将新的数据记录写入环形缓冲区文件；如果 *bRemove* = TRUE，则会将最旧的数据记录移除。

```
PROGRAM MAIN
VAR
    bOpen    : BOOL;
    bClose   : BOOL;
    bCreate  : BOOL;
    bAdd     : BOOL;
    bRemove  : BOOL;
    bGet     : BOOL;
    bReset   : BOOL;

    fbFileBuffer    : FB_FileRingBuffer := ( sNetId     := '',
                                             sPathName  := 'c:\temp\Data.dat',
                                             ePath      := PATH_GENERIC,
                                             nID        := 1,
                                             cbBuffer   := 100, (*cbBuffer := 16#80000000, 2GB*)
                                             bOverwrite := TRUE,
                                             pWriteBuff := 0,
                                             cbWriteLen := 0,
                                             pReadBuff  := 0,
                                             cbReadLen  := 0,
                                             tTimeout   := t#5s );

    storeData : ARRAY[1..10] OF BYTE :=[10(0)];
    cbStore   : UDINT;
    loadData  : ARRAY[1..10] OF BYTE :=[10(0)];
    cbLoad    : UDINT;
    i         : INT;
END_VAR
```

```
fbFileBuffer( cbReturn => cbLoad );

IF NOT fbFileBuffer.bBusy THEN

    IF bOpen THEN
        bOpen := FALSE;
        fbFileBuffer.A_Open();
    END_IF

    IF bClose THEN
        bClose := FALSE;
        fbFileBuffer.A_Close();
    END_IF

    IF bCreate THEN
        bCreate := FALSE;
        fbFileBuffer.A_Create();
    END_IF

    IF bAdd THEN
        bAdd := FALSE;

        (* modify data *)
        FOR i:=1 TO 10 BY 1 DO
            storeData[i] := storeData[i] + 1;
        END_FOR

        cbStore := SEL( cbStore > 1, SIZEOF(storeData), cbStore -
 1 ); (* modify the data chunk length *)
        fbFileBuffer.A_AddTail( pWriteBuff := ADR(storeData), cbWriteLen := cbStore,
                                pReadBuff := 0, cbReadLen:=0 );
    END_IF

    IF bRemove THEN
        bRemove := FALSE;
        fbFileBuffer.A_RemoveHead( pWriteBuff := 0, cbWriteLen := 0,
                                   pReadBuff := ADR(loadData), cbReadLen := SIZEOF(loadData));
    END_IF

    IF bGet THEN
        bGet := FALSE;
        fbFileBuffer.A_GetHead( pWriteBuff := 0, cbWriteLen := 0,
                                pReadBuff := ADR(loadData), cbReadLen := SIZEOF(loadData));
    END_IF

    IF bReset THEN
        bReset := FALSE;
        fbFileBuffer.A_Reset();
    END_IF

END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.4 示例：内存环 FiFo（FB_MemRingBuffer）

您可以在这里找到完整的来源：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803399691.zip

下面的示例说明了功能块的简单应用。具有相同长度的数据集将被缓冲（尽管这不是强制性的）。数据集的结构如下：

```
TYPE ST_DataSetEntry :
STRUCT
    bFlag  : BOOL;
    nValue : BYTE;
    sMsg   : STRING(20) := 'Unknown';
END_STRUCT
END_TYPE
```

FB_Data setFifo 功能块的接口：

在项目示例中使用的应用程序特定的功能块 FB_Data setFifo 在内部使用 FB_MemRingBuffer 功能块。该模块可简化数据集的添加/移除。此外，新功能块还可以提供缓冲区当前的百分比填充状态和覆盖选项。如果设置了 *bOverwrite* 输入且缓冲区已满，则会从缓冲区中移除最旧的条目，并使用新的条目将其覆盖。

```
VAR_GLOBAL CONSTANT
    MAX_BUFFER_SIZE : UDINT := 1000;
END_VAR
```

```
FUNCTION_BLOCK FB_DataSetFifo
VAR_INPUT
    bOverwrite : BOOL;
    in         : ST_DataSetEntry;
END_VAR
VAR_OUTPUT
    bOk        : BOOL;
    nCount     : UDINT;
    nLoad      : UDINT;
    out        : ST_DataSetEntry;
END_VAR
VAR
    arrBuffer  : ARRAY[0..MAX_BUFFER_SIZE] OF BYTE; (* Buffer memory used by FB_MemRingBuffer functi
on block *)
    fbBuffer   : FB_MemRingBuffer;
END_VAR
```

主要程序：

bReset 的上升沿会删除所有缓冲区条目。如果您设置 *bAdd* = TRUE，则会将新的数据记录写入环形缓冲区；如果 *bRemove* = TRUE，则会将最旧的数据记录移除。*bGet* 的上升沿会导致读取最旧的数据集，但不会将其移除。

```
PROGRAM MAIN
VAR
    fbFifo      : FB_DataSetFifo := ( bOverwrite := TRUE );
    newEntry    : ST_DataSetEntry;
    oldEntry    : ST_DataSetEntry;
    bSuccess    : BOOL;
    nCount      : UDINT;
    nLoad       : UDINT;

    bReset      : BOOL := TRUE;
    bAdd        : BOOL := TRUE;
    bGet        : BOOL := TRUE;
    bRemove     : BOOL := TRUE;
END_VAR
```

```
IF bReset THEN
    bReset := FALSE;
    (* reset fifo (clear all entries) *)
    fbFifo.A_Reset( in := newEntry, bOk=>bSuccess, nCount=> nCount, nLoad => nLoad );
END_IF

IF bAdd THEN
    bAdd := FALSE;

    (* create new or modify data set entry *)
    newEntry.bFlag  := NOT newEntry.bFlag;
    newEntry.nValue := newEntry.nValue + 1;
```

```
    newEntry.sMsg   := BYTE_TO_STRING(newEntry.nValue);

    (* add new entry to the fifo *)
    fbFifo.A_Add( in := newEntry, bOk=>bSuccess, nCount=> nCount, nLoad => nLoad );
END_IF

IF bGet THEN
    bGet := FALSE;
    (* get (but not delete) oldest entry *)
    fbFifo.A_Get( out => oldEntry, bOk => bSuccess, nCount => nCount, nLoad => nLoad );
END_IF

IF bRemove THEN
    bRemove:= FALSE;
    (* remove oldest entry *)
    fbFifo.A_Remove( out => oldEntry, bOk => bSuccess, nCount => nCount, nLoad => nLoad );
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.5 示例：内存环 FiFo（FB_MemRingBufferEx）

您可以在这里将完整的来源解压缩：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803404811.zip

bAdd 的上升沿会导致将新的数据元素（pubObj 数组）存储到环形缓冲区中。然后，通过 bGet 的上升沿可以将最旧的数据元素复制到 getObj 变量中。

通过 bRelease 的上升沿可以将不需要的数据元素从缓冲区中移除。

```
PROGRAM MAIN
VAR
    bReset  : BOOL := TRUE;
    bAdd, bGet, bRelease, bGetFree : BOOL;
    putObj  : ARRAY[0..3] OF BYTE :=[ 16#00, 16#AA, 16#BB, 16#CC];
    getObj  : ARRAY[0..3] OF BYTE :=[ 4(0)];
    bOk     : BOOL;
    nCount  : UDINT;
    cbSize  : UDINT;
    cbFree  : UDINT;
    fbBuffer: FB_MemRingBufferEx;
    buffer  : ARRAY[0..30] OF BYTE;
END_VAR
```

```
IF bReset THEN
    bReset := FALSE;
    fbBuffer.A_Reset( pBuffer := ADR( buffer ), cbBuffer := SIZEOF( buffer ),
                    bOk=>bOk,nCount=>nCount, cbSize=>cbSize, cbFree=>cbFree );
END_IF

IF bAdd THEN
    bAdd := FALSE;
    putObj[0] := putObj[0] + 1;(* modify data *)

    fbBuffer.A_AddTail( pBuffer := ADR( buffer ), cbBuffer := SIZEOF( buffer ),
                        pWrite := ADR( putObj ), cbWrite := SIZEOF( putObj ),
                        bOk=>bOk, nCount=>nCount, cbSize=>cbSize, cbFree=>cbFree );
    IF fbBuffer.bOk THEN
        ;(* Success *)
    ELSE
        ;(* Buffer overflow *)
    END_IF
END_IF

IF bGet THEN
    bGet := FALSE;
    fbBuffer.A_GetHead( pBuffer := ADR( buffer ), cbBuffer := SIZEOF( buffer ),
                        bOk=>bOk, nCount=>nCount, cbSize=>cbSize, cbFree=>cbFree );
    IF fbBuffer.bOk THEN
        (* Success *)
        MEMCPY( ADR( getObj ), fbBuffer.pRead, MIN( SIZEOF( getObj ), fbBuffer.cbRead ) );
    ELSE
        ;(* Buffer empty *)
    END_IF

END_IF


IF bRelease THEN
    bRelease := FALSE;
    fbBuffer.A_FreeHead( pBuffer := ADR( buffer ), cbBuffer := SIZEOF( buffer ),
                        bOk=>bOk, nCount=>nCount, cbSize=>cbSize, cbFree=>cbFree );
```

```
    IF fbBuffer.bOk THEN
        ;(* Success *)
    ELSE
        ;(* Buffer empty *)
    END_IF
END_IF

IF bGetFree THEN
    bGetFree := FALSE;
    fbBuffer.A_GetFreeSize( pBuffer := ADR( buffer ), cbBuffer := SIZEOF( buffer ),
                            bOk=>bOk, nCount=>nCount, cbSize=>cbSize, cbFree=>cbFree );
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.6 示例：哈希表（FB_HashTableCtrl）

您可以在这里将完整的来源解压缩：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803421707.zip

示例项目有两个程序部分：

- P_TABLE_OF_UDINT 是一个简单的示例程序，仅会处理哈希表中的 32 位值。
- P_TABLE_OF_STRUCTDATA 说明了如何处理哈希表中的其他数据类型（例如，结构化数据类型）。

在运行时不能更改表格元素的最大数量，该数量在示例项目中受 MAX_DATA_ELEMENTS 限制。如果需要更多元素，则可以相应地扩大表格数组（即增加常量的值）。

```
VAR_GLOBAL CONSTANT
    MAX_DATA_ELEMENTS   : UDINT := 100;(* Max. number of elements in the list *)
    MAX_NAME_LENGTH     : UDINT := 30;(* Max. length of article name *)
END_VAR
```

程序 P_TABLE_OF_UDINT

在第一个 PLC 周期中，商品编号和商品名称存储在表格中。商品编号可作为键，商品名称的数组索引可作为值。
通过 bLookup 的上升沿，可以通过商品编号找到商品名称。

```
PROGRAM P_TABLE_OF_UDINT
VAR
    sInfo    : T_MaxString := '';
    bAdd     : BOOL := TRUE;
    bLookup  : BOOL := TRUE;
    bRemove  : BOOL := TRUE;
    bEnum    : BOOL := TRUE;
    bCount   : BOOL := TRUE;

    search   : UDINT := 11111;(* article number *)

    fbTable  : FB_HashTableCtrl;(* basic hash table control function block *)
    hTable   : T_HHASHTABLE;(* hash table handle *)
    table    : ARRAY[0..MAX_DATA_ELEMENTS] OF T_HashTableEntry;
(* Max. number of hash table entries. The value of hash table entry = 32 bit integer *)
    names    : ARRAY[0..MAX_DATA_ELEMENTS] OF STRING(MAX_NAME_LENGTH);
    bInit    : BOOL := TRUE;
END_VAR
```

```
IF bInit THEN
    bInit := FALSE;
    F_CreateHashTableHnd( ADR( table ), SIZEOF( table ), hTable );(* Intialize table handle *)
END_IF

IF bAdd THEN
    bAdd := FALSE;

    (* Fill table. Article number is the key. Array index number is the value (article name) *)
    names[0] := 'Chair';
    fbTable.A_Add( key := 12345, putValue := 0(* array index*), hTable := hTable );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF

    names[1] := 'Table';
    fbTable.A_Add( key := 67890, putValue := 1, hTable := hTable );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
```

```
    END_IF

    names[2] := 'Couch';
    fbTable.A_Add( key := 11111, putValue := 2, hTable := hTable );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF

    names[3] := 'TV set';
    fbTable.A_Add( key := 22222, putValue := 3, hTable := hTable );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF
END_IF


IF bLookup THEN    (* search for the article name by article number *)
    bLookup := FALSE;
    sInfo := '';
    fbTable.A_Lookup( key := search, hTable := hTable );
    IF fbTable.bOk THEN
        sInfo := names[fbTable.getValue];
    ELSE
        ;(* Entry not found *)
    END_IF
END_IF

IF bRemove THEN(* remove one entry from the table *)
    bRemove := FALSE;
    sInfo := '';
    fbTable.A_Remove( key := search, hTable := hTable );
    IF fbTable.bOk THEN
        sInfo := names[fbTable.getValue];
    ELSE
        ;(* Entry not found *)
    END_IF
END_IF


IF bEnum THEN(* enumerate table entries *)
    bEnum := FALSE;
    sInfo := '';

    fbTable.A_GetFirst( putPosPtr := 0, hTable := hTable );
    IF fbTable.bOk THEN
        sInfo := names[fbTable.getValue];

        REPEAT
            fbTable.A_GetNext( putPosPtr := fbTable.getPosPtr , hTable := hTable );
            IF fbTable.bOk THEN
                sInfo := names[fbTable.getValue];
            END_IF
        UNTIL NOT fbTable.bOk
        END_REPEAT

    END_IF
END_IF

IF bCount THEN(* count entries in the table *)
    bCount := FALSE;
    sInfo := UDINT_TO_STRING( hTable.nCount );
END_IF
```

程序 P_TABLE_OF_STRUCTDATA

程序的该部分说明了如何在表格中操作结构化数据集，以取代简单的 32 位数字。32 位元素值仅可用作指向实际元素值的引用指针。引用指针能够指向结构化变量或其他数据类型的实例。功能被封装在一个功能块中。功能块 *FB_SpecialHashTableCtrl* 可以被视为功能块 *FB_HashTableCtrl* 的专用版本。FB（即专用 FB）也可在内部使用 *FB_HashTableCtrl* 块。

DATAELEMENT_TO_STRING 函数仅可用于允许节点值的可视化输出。

以 *ST_DataElement* 类型的结构化变量为例。亮点：您可以将更多成员变量添加到 ST_DataElement 的数据类型声明中，而无需对程序或 FB_SpecialHashTableCtrl 功能块进行任何更改。

ST_DataElement 的类型声明：

```
TYPE ST_DataElement :(* Structured application data entry *)
STRUCT
    (* Adapt this structure to match your application needs *)
    number        : UDINT := 0;
    name        : STRING(MAX_NAME_LENGTH) := '';
    price        : REAL := 0.0;
END_STRUCT
END_TYPE
```

32 位元素值如何成为指向 ST_DataElement 数组实例的引用指针?

表格的最大大小受 MAX_DATA_ELEMENTS 常量限制。因此，在表格中存储的引用指针不能超过 MAX_DATA_ELEMENTS 个。在 *FB_SpecialHashTableCtrl* 块内部有一个 ST_DataElement 数组变量，其数组大小与 T_HashTableEntry 数组变量相同。为了简化操作，两个数组的数组索引都是一样的!

每个 T_HashTableEntry 数组元素仅可在表格中使用一次。*FB_HashTableCtrl* 功能块会搜索空闲/未使用的 T_HashTableEntry 数组元素。如果成功，则会将元素添加到表格中。操作 *A_GetIndexAtPosPtr* 可以用于确定 T_HashTableEntry 数组的索引。在下一步中，可将刚刚添加的 32 位节点值分配到 ST_DataElement 数组中相同数组元素的地址。在项目示例中，通过第二个操作调用：*A_Add*。

nodes[index].value := ADR(dataPool[index])

例如，在 *FB_SpecialHashTableCtrl->A_Add* 操作中实现分配：

```
(* Adds entry to the table *)
MEMSET( ADR( getValue ), 0, SIZEOF( getValue ) );
getPosPtr := 0;

fbTable.A_Add( hTable := hTable, key := key, putValue := 16#00000000(* we will set this value later
*), getPosPtr=>getPosPtr, bOk=>bOk );
(* Add new element to the table, getPosPtr points to the new entry *)
IF fbTable.bOk THEN(* Success *)
    fbTable.A_GetIndexAtPosPtr( hTable := hTable, putPosPtr := getPosPtr, getValue =>indexOfElem, bO
k=>bOk );(* Get array index of getPosPtr entry *)
    IF fbTable.bOk THEN(* Success *)
        pRefPtr     := ADR( dataPool[indexOfElem] );(* Get pointer to the data element *)

        pRefPtr^ := putValue;(* copy application value *)

        fbTable.A_Add( hTable := hTable, key := key, putValue := pRefPtr, bOk=>bOk );
(* Assign the entry value = pointer to the data element *)
        IF fbTable.bOk THEN(* Success *)
            getValue := putValue;
        END_IF
    END_IF
END_IF
```

```
PROGRAM P_TABLE_OF_STRUCTDATA
VAR
    sInfo       : T_MaxString := '';
    bAdd        : BOOL := TRUE;
    bLookup     : BOOL := TRUE;
    bRemove     : BOOL := TRUE;
    bEnum       : BOOL := TRUE;
    bCount      : BOOL := TRUE;

    search      : UDINT := 11111;(* article number *)

    fbTable     : FB_SpecialHashTableCtrl;(* Specialized hash table control function block *)
    putValue    : ST_DataElement;
    getValue    : ST_DataElement;
    getPosPtr   : POINTER TO T_HashTableEntry := 0;
    bInit       : BOOL := TRUE;
END_VAR
```

```
IF bInit THEN
    bInit     := FALSE;
    fbTable.A_Reset();(* reset / initialize table *)
END_IF


IF bAdd THEN
    bAdd := FALSE;

    (* Fill table. Article number is the key and data structure is the value *)
    putValue.number := 12345;
    putValue.name := 'Chair';
    putValue.price := 44.98;
    fbTable.A_Add( key := 12345, putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF

    putValue.number := 67890;
    putValue.name := 'Table';
    putValue.price := 99.98;
    fbTable.A_Add( key := 67890, putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF

    putValue.number := 11111;
    putValue.name := 'Couch';
    putValue.price :=  99.98;
    fbTable.A_Add( key := 11111, putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF
```

```
    putValue.number := 22222;
    putValue.name := 'TV set';
    putValue.price :=  99.98;
    fbTable.A_Add( key := 22222, putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbTable.bOk THEN
        ;(* Table overflow *)
    END_IF

END_IF

IF bLookup THEN(* search for the article name by article number *)
    bLookup := FALSE;
    sInfo := '';
    fbTable.A_Lookup( key := search, getPosPtr=>getPosPtr, getValue=>getValue );
    IF fbTable.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );
    ELSE
        ;(* Entry not found *)
    END_IF
END_IF

IF bRemove THEN(* remove one entry from the table *)
    bRemove := FALSE;
    sInfo := '';
    fbTable.A_Remove( key := search, getPosPtr=>getPosPtr, getValue=>getValue );
    IF fbTable.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );
    ELSE
        ;(* Entry not found *)
    END_IF
END_IF

IF bEnum THEN(* enumerate table entries *)
    bEnum := FALSE;
    sInfo := '';

    fbTable.A_GetFirst( putPosPtr := 0, getPosPtr=>getPosPtr, getValue=>getValue );
    IF fbTable.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );

        REPEAT
            fbTable.A_GetNext( putPosPtr := fbTable.getPosPtr , getPosPtr=>getPosPtr, getValue=>getV
alue );
            IF fbTable.bOk THEN
                sInfo := DATAELEMENT_TO_STRING( getValue );
            END_IF
        UNTIL NOT fbTable.bOk
        END_REPEAT

    END_IF
END_IF

IF bCount THEN(* count entries in the table *)
    bCount := FALSE;
    fbTable.A_Count();
    IF fbTable.bOk THEN
        sInfo := UDINT_TO_STRING( fbTable.nCount );
    END_IF
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.7 示例：链表（FB_LinkedListCtrl）

您可以在这里将完整的来源解压缩：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803425547.zip

示例项目有两个程序部分：

- P_LIST_OF_UDINT 是一个简单的示例程序，仅会编辑链表中的 32 位值。
- P_LIST_OF_STRUCTDATA 说明了如何在链表中管理其他数据类型（例如，结构化数据类型）。

在运行时不能更改节点元素的最大数量，该数量在示例项目中受 MAX_DATA_ELEMENTS 限制。如果您需要更多节点，则您必须相应地增加节点数组的大小（即增加常量的值）。

```
VAR_GLOBAL CONSTANT
    MAX_DATA_ELEMENTS : UDINT := 100;(* Max. number of elements in the list *)
    MAX_NAME_LENGTH   : UDINT := 30;(* Max. length of article name *)
END_VAR
```

程序 P_LIST_OF_UDINT

在第一个 PLC 周期中对链表的句柄进行初始化。在访问列表时，该句柄可作为 VAR_IN_OUT 变量传递给 FB_LinkedListCtrl 功能块。通过功能块的操作调用，可对链表进行操作。这样可以添加、移除和搜索节点元素。当相关布尔变量的上升沿出现时，就会执行所需的操作。在您运行程序时，所有操作都会执行一次。

```
PROGRAM P_LIST_OF_UDINT
VAR
    sInfo               : T_MaxString := '';
    bAddTailValue       : BOOL := TRUE;
    bAddHeadValue       : BOOL := TRUE;
    bGetTail            : BOOL := TRUE;
    bGetHead            : BOOL := TRUE;
    bFind               : BOOL := TRUE;
    bRemoveHeadValue    : BOOL := TRUE;
    bRemoveTailValue    : BOOL := TRUE;
    bCount              : BOOL := TRUE;

    search              : UDINT := 12345;

    fbList              : FB_LinkedListCtrl;(* basic linked list control function block *)
    hList               : T_HLINKEDLIST;(* linked list handle *)
    nodes               : ARRAY[0..MAX_DATA_ELEMENTS] OF T_LinkedListEntry;
(* Max. number of linked list nodes. The value of list node = 32 bit integer *)
    putValue            : PVOID;(* Pointer or integer value (x86=>32bit, x64=>64bit)*)
    getValue            : PVOID;(* Pointer or integer value (x86=>32bit, x64=>64bit)*)
    getPosPtr           : POINTER TO T_LinkedListEntry := 0;
    bInit               : BOOL := TRUE;
END_VAR
```

```
IF bInit THEN
    bInit     := FALSE;
    F_CreateLinkedListHnd( ADR( nodes ), SIZEOF( nodes ), hList );
END_IF

IF bAddTailValue THEN(* add some nodes to the list *)
    bAddTailValue := FALSE;

    putValue := 22222;
    fbList.A_AddTailValue( hList := hList, putValue := putValue, getPosPtr=>getPosPtr, getValue=>get
Value );
    putValue := 11111;
    fbList.A_AddTailValue( hList := hList, putValue := putValue, getPosPtr=>getPosPtr, getValue=>get
Value );
    putValue := 12345;
    fbList.A_AddTailValue( hList := hList, putValue := putValue, getPosPtr=>getPosPtr, getValue=>get
Value );
    putValue := 67890;
    fbList.A_AddTailValue( hList := hList, putValue := putValue, getPosPtr=>getPosPtr, getValue=>get
Value );
END_IF

IF bAddHeadValue THEN
    bAddHeadValue := FALSE;

    putValue := 33333;
    fbList.A_AddHeadValue( hList := hList, putValue := putValue, getPosPtr=>getPosPtr, getValue=>get
Value );
    putValue := 44444;
    fbList.A_AddHeadValue( hList := hList, putValue := putValue, getPosPtr=>getPosPtr, getValue=>get
Value );
END_IF


IF bGetTail THEN(* enumerate all nodes in list (start at tail node) *)
    bGetTail := FALSE;
    sInfo := '';

    fbList.A_GetTail( hList := hList, getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := PVOID_TO_STRING( getValue );

        REPEAT
            fbList.A_GetPrev( hList := hList, putPosPtr := getPosPtr, getValue=>getValue, getPosPtr=
>getPosPtr );
            IF fbList.bOk THEN
                sInfo := PVOID_TO_STRING( getValue );
            ELSE
                EXIT;
            END_IF
        UNTIL NOT fbList.bOk
        END_REPEAT
    END_IF
END_IF

IF bGetHead THEN(* enumerate all nodes in list (start at head node) *)
    bGetHead := FALSE;
    sInfo := '';

    fbList.A_GetHead( hList := hList, getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := PVOID_TO_STRING( getValue );
```

```
        REPEAT
            fbList.A_GetNext( hList := hList, putPosPtr := getPosPtr, getValue=>getValue, getPosPtr=
>getPosPtr );
            IF fbList.bOk THEN
                sInfo := PVOID_TO_STRING( getValue );
            ELSE
                EXIT;
            END_IF
        UNTIL NOT fbList.bOk
        END_REPEAT
    END_IF
END_IF


IF bFind  THEN(* search for node in the list by node value*)
    bFind     := FALSE;
    getPosPtr := 0;(* start from first node element *)
    sInfo := '';

    REPEAT
        fbList.A_FindNext( hList := hList, putPosPtr := getPosPtr, putValue := search, getValue=>get
Value, getPosPtr=>getPosPtr );
        IF fbList.bOk THEN
            sInfo := PVOID_TO_STRING( getValue );
        ELSE
            EXIT;
        END_IF
    UNTIL NOT fbList.bOk
    END_REPEAT

END_IF


IF bRemoveTailValue THEN(* remove tail node from node list *)
    bRemoveTailValue := FALSE;
    sInfo := '';
    fbList.A_RemoveTailValue( hList := hList, getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := PVOID_TO_STRING( getValue );
    END_IF
END_IF


IF bRemoveHeadValue THEN(* remove head node from node list *)
    bRemoveHeadValue := FALSE;
    sInfo := '';
    fbList.A_RemoveHeadValue( hList := hList, getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := PVOID_TO_STRING( getValue );
    END_IF
END_IF

IF bCount THEN(* count nodes in list *)
    bCount := FALSE;
    sInfo := UDINT_TO_STRING( hList.nCount );
END_IF
```

程序 P_LIST_OF_STRUCTDATA

程序的该部分说明了如何在列表中操作结构化数据集，以取代简单的 32 位数字。在这种情况下，32 位节点值仅可用作指向实际节点值的引用指针。引用指针能够指向结构化变量或其他数据类型的实例。功能被封装在一个功能块中。功能块 *FB_SpecialLinkedListCtrl* 可以被视为功能块 FB_*LinkedListCtrl* 的专用版本。专用 FB 也可在内部使用 *FB_LinkedListCtrl* 块。

DATAELEMENT_TO_STRING 函数仅可用于允许节点值的可视化输出。

以 *ST_DataElement* 类型的结构化变量为例。亮点：您可以将更多成员变量添加到 ST_DataElement 的数据类型声明中，而无需对程序或 FB_SpecialLinkedListCtrl 功能块进行任何更改。

ST_DataElement 的类型声明：

```
(* Structured application data entry *)
TYPE ST_DataElement :
STRUCT
    (* Adapt this structure to match your application needs *)
    number   : UDINT := 0;
    name     : STRING(MAX_NAME_LENGTH) := '';
    price    : REAL := 0.0;
END_STRUCT
END_TYPE
```

实现了简单的搜索功能。您可以搜索具有特定*名称*、*编号*或*价格的*节点。

32 位节点值如何成为指向 ST_DataElement 数组实例的引用指针?

列表的最大大小受 MAX_DATA_ELEMENTS 常量限制。因此，在列表中存储的引用指针不能超过 MAX_DATA_ELEMENTS 个。*FB_SpecialLinkedListCtrl* 功能块有一个内部的 ST_DataElement 数组变量，其大小与 T_LinkedListEntry 数组变量相同。为了简化操作，两个数组的数组索引都是一样的!

每个 T_LinkedListEntry 数组元素仅可在列表中插入一次。因此，*FB_LinkedListCtrl* 功能块会搜索空闲/未使用的 T_LinkedListEntry 数组元素，如果成功，则会将其插入列表中。通过操作 *A_GetIndexAtPosPtr*，可以确定正在使用的 T_LinkedListEntry 的索引。在下一步中，可将刚刚添加的 32 位节点值分配到 ST_DataElement 数组中相同数组元素的地址。在项目示例中，通过操作调用：*A_SetValueAtPosPtr*。

nodes[index].value := ADR(dataPool[index])

例如，在 *FB_SpecialLinkedListCtrl->A_AddHeadValue* 操作中执行分配：

```
(* Adds head to the node list *)
MEMSET( ADR( getValue ), 0, SIZEOF( getValue ) );
getPosPtr := 0;

fbList.A_AddHeadValue( hList := hList, putValue := 16#00000000(* we will set this value later *), ge
tPosPtr=>getPosPtr, bOk=>bOk );
(* Add new element to the list, getPosPtr points to the new list node *)
IF fbList.bOk THEN(* Success *)
    fbList.A_GetIndexAtPosPtr( hList := hList, putPosPtr := getPosPtr, getValue =>indexOfElem, bOk=>
bOk );(* Get array index of getPosPtr *)
    IF fbList.bOk THEN(* Success *)
        pRefPtr     := ADR( dataPool[indexOfElem] );(* Get pointer to the data element *)
        pRefPtr^     := putValue;(* set element value *)

        fbList.A_SetValueAtPosPtr( hList := hList, putPosPtr := getPosPtr, putValue := pRefPtr, bOk=
>bOk );(* Assign the node value = pointer to the data element *)
        IF fbList.bOk THEN(* Success *)
            getValue := putValue;
        END_IF
    END_IF
END_IF
```

```
PROGRAM P_LIST_OF_STRUCTDATA
VAR
    sInfo              : T_MaxString := '';
    bAddTailValue      : BOOL := TRUE;
    bAddHeadValue      : BOOL := TRUE;
    bGetTail           : BOOL := TRUE;
    bGetHead           : BOOL := TRUE;
    bFind              : BOOL := TRUE;
    bRemoveHeadValue   : BOOL := TRUE;
    bRemoveTailValue   : BOOL := TRUE;
    bCount             : BOOL := TRUE;

    search             : ST_DataElement := ( name := 'Couch', price := 99.98, number := 12345 );
(* search value ( by name, by price or by number ) *)
    eSearch            : E_SEARCH_CRITERIA := eSEARCH_BY_NAME;
(* / eSEARCH_BY_PRICE / eSEARCH_BY_NUMBER *)

    fbList             : FB_SpecialLinkedListCtrl;
(* Specialized linked list control function block *)
    putValue           : ST_DataElement;
    getValue           : ST_DataElement;
    getPosPtr          : POINTER TO T_LinkedListEntry := 0;
    bInit              : BOOL := TRUE;
END_VAR
```

```
IF bInit THEN
    bInit     := FALSE;
    fbList.A_Reset();(* reset / initialize list *)
END_IF


IF bAddTailValue THEN(* add some nodes to the list *)
    bAddTailValue := FALSE;

    putValue.number := 22222;
    putValue.name := 'TV set';
    putValue.price :=  99.98;
    fbList.A_AddTailValue( putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbList.bOk THEN
        ;(* List overflow *)
    END_IF


    putValue.number := 11111;
    putValue.name := 'Couch';
    putValue.price :=  99.98;
    fbList.A_AddTailValue( putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbList.bOk THEN
        ;(* List overflow *)
    END_IF

    putValue.number := 12345;
```

```
    putValue.name := 'Chair';
    putValue.price := 44.98;
    fbList.A_AddTailValue( putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbList.bOk THEN
        ;(* List overflow *)
    END_IF

    putValue.number := 67890;
    putValue.name := 'Table';
    putValue.price := 99.98;
    fbList.A_AddTailValue( putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbList.bOk THEN
        ;(* List overflow *)
    END_IF
END_IF

IF bAddHeadValue THEN
    bAddHeadValue := FALSE;

    putValue.number := 33333;
    putValue.name := 'Couch';
    putValue.price := 199.98;
    fbList.A_AddHeadValue( putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbList.bOk THEN
        ;(* List overflow *)
    END_IF

    putValue.number := 44444;
    putValue.name := 'Couch';
    putValue.price := 299.98;
    fbList.A_AddHeadValue( putValue := putValue, getPosPtr=>getPosPtr, getValue=>getValue );
    IF NOT fbList.bOk THEN
        ;(* List overflow *)
    END_IF
END_IF



IF bGetTail THEN(* enumerate all nodes in list (start at tail node) *)
    bGetTail := FALSE;
    sInfo := '';

    fbList.A_GetTail( getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );

        REPEAT
            fbList.A_GetPrev( putPosPtr := getPosPtr, getValue=>getValue, getPosPtr=>getPosPtr );
            IF fbList.bOk THEN
                sInfo := DATAELEMENT_TO_STRING( getValue );
            ELSE
                EXIT;
            END_IF
        UNTIL NOT fbList.bOk
        END_REPEAT
    END_IF
END_IF

IF bGetHead THEN(* enumerate all nodes in list (start at head node) *)
    bGetHead := FALSE;
    sInfo := '';

    fbList.A_GetHead( getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );

        REPEAT
            fbList.A_GetNext( putPosPtr := getPosPtr, getValue=>getValue, getPosPtr=>getPosPtr );
            IF fbList.bOk THEN
                sInfo := DATAELEMENT_TO_STRING( getValue );
            ELSE
                EXIT;
            END_IF
        UNTIL NOT fbList.bOk
        END_REPEAT
    END_IF
END_IF


IF bFind  THEN(* search for node in the list by node value (name, price, number... )*)
    bFind     := FALSE;
    getPosPtr := 0;(* start from first node element *)
    sInfo := '';

    REPEAT
        fbList.A_Find( eSearch := eSearch, putPosPtr := getPosPtr, putValue := search, getValue=>get
Value, getPosPtr=>getPosPtr );
        IF fbList.bOk THEN
            sInfo := DATAELEMENT_TO_STRING( getValue );
        ELSE
            EXIT;
        END_IF
    UNTIL NOT fbList.bOk
    END_REPEAT
```

```
END_IF


IF bRemoveTailValue THEN(* remove tail node from node list *)
    bRemoveTailValue := FALSE;
    sInfo := '';
    fbList.A_RemoveTailValue( getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );
    END_IF
END_IF


IF bRemoveHeadValue THEN(* remove head node from node list *)
    bRemoveHeadValue := FALSE;
    sInfo := '';
    fbList.A_RemoveHeadValue( getValue=>getValue, getPosPtr=>getPosPtr );
    IF fbList.bOk THEN
        sInfo := DATAELEMENT_TO_STRING( getValue );
    END_IF
END_IF

IF bCount THEN(* count nodes in list *)
    bCount := FALSE;
    sInfo := '';
    fbList.A_Count( );
    IF fbList.bOk THEN
        sInfo := UDINT_TO_STRING( fbList.nCount );
    END_IF
END_IF
```

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

9.8 示例：写入/读取 CSV 文件

您可以在这里将完整的来源解压缩到示例项目中：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803601163.zip

使用项目示例生成的 CSV 文件：

不含二进制数据的数据字段：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803605003.zip
含有二进制数据的数据字段：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803430923.zip（请注意，该文件需要特殊软件才能正确解读）

CSV 代表逗号分隔值。以下文档介绍了如何借助辅助的 PLC CSV 功能写入和读取 CSV 文件。CSV 文件基本上是文本文件，可以存储简单的结构化数据集，这些数据集可以用于两个系统之间的数据交换。该格式可以存储不同长度的表格或列表。一个表格行与 CSV 文件中的一个数据集（或行）相对应。一个表格单元格与 CSV 文件中的一个数据字段相对应。

关于支持的 CSV 格式的一般信息

- CSV 格式的文件的扩展名应为 **.csv**。
- CRLF 字符（CR= 回车，LF= 换行）可用于分隔各个数据集（行）（Windows 操作系统）。即，每个数据集后面必须有一个 CRLF。
- CSV 文件必须以 CRLF 字符结束。
- 二进制数据必须用单引号括起来。如果不使用单引号，则数据字段仅可包含数字和/或字母。
- 包含特殊字符/控制字符的数据字段用双引号括起来。如果数据字段包含双引号，则会添加第二个双引号。
- 特殊字符可用于分隔数据字段（列）。辅助功能所使用的各个数据字段的标准分隔符是分号。在德国和欧洲，分号可用作数据字段分隔符，而在美国则倾向于使用逗号。通过 PLC 全局变量 **DEFAULT_CSV_FIELD_SEP** 可以将分隔符从分号配置为逗号。
- 每个数据集应该具有相同数量的数据字段（列）。

具有 m 列 和 n 行 的 CSV 文件的基本配置（CRLF 字符通常不可见，在图中用字母 **CRLF** 表示）

```
"Field1Record1";"Field2Record1"; ... ;"Field(m)Record1"CRLF
"Field1Record2";"Field2Record2"; ... ;"Field(m)Record2"CRLF

...

"Field1Record(n)";"Field2Record(n)"; ... ;"Field(m)Record(n)"CRLF
```

可用的功能块和函数

- STRING_TO_CSVFIELD [▶ 311]、ARG_TO_CSVFIELD [▶ 266]：将 PLC 数据转换为 CSV 格式的数据字段；
- CSVFIELD_TO_STRING [▶ 273]、CSVFIELD_TO_ARG [▶ 272]：将 CSV 格式的数据字段转换为 PLC 数据；
- FB_CSVMemBufferWriter [▶ 50]：从多个数据字段生成字节缓冲区中的数据集；
- FB_CSVMemBufferReader [▶ 48]：将字节缓冲区中的数据集拆分成单个数据字段；

在文本模式或二进制模式下写入/读取 CSV 文件

借助用于文件访问的 PLC 功能块，可以在文本或二进制模式下读取或写入 CSV 文件。根据所选模式的不同，优点和缺点也有所不同。

在 99% 的情况下，在文本模式下可以读取/写入 CSV 文件。只有在极少数情况下才需要使用二进制模式。

	文本模式	二进制模式
用于文件读取访问的功能块	**FB_FileGets**（特殊功能：在读取访问期间，该功能块可从文件中自动移除最后一个数据集末尾的 CR 字符。务必恢复/重新插入该字符，以确保 FB_CSVMemBufferReader 功能块能够解释此类数据集）	**FB_FileRead**
用于文件写入访问的功能块	**FB_FilePuts**（特殊功能：在写入访问期间，该功能块可在最后一个数据集的末尾自动添加一个 CR 字符。不过，FB_CSVMemBufferWriter 也会生成 CR 字符。为了避免 CSV 文件中的字符重复，必须在写入访问之前将其从缓冲区中移除）	**FB_FileWrite**
编程工作	较少	较多
数据字段中的特殊字符、不可打印的控制字符	不允许	允许
可以写入/读取的最大数据集长度	限值为 253 个字符（数据集 + CRLF）。即数据集长度不得超过 253 个字符。	理论上，最大数据集长度不受限制。不过，功能块（FB_CSVMemBufferReader 和 FB_CSVMemBufferWriter）会将 CSV 字段限制为在全局参数 cMaxCSVFieldValueSize 中指定的最大大小。
使用写入访问功能块可以写入一个完整的数据集	是	是
使用读取访问功能块可以读取一个完整的数据集	是。纯文本文件中的数据集以 CRLF 结尾。在此类文件中，CRLF 表示一行的结束。FB_FileGets 功能块可读取直到 CRLF 的数据。	否
数据字段中的二进制数据	不允许	允许
辅助功能可将 PLC 数据转换为 CSV 格式，反之亦然	CSVFIELD_TO_STRING [▶ 273] STRING_TO_CSVFIELD [▶ 311]	CSVFIELD_TO_ARG [▶ 272] ARG_TO_CSVFIELD [▶ 266]

	文本模式	二进制模式
支持可以直接写入/读取的 PLC 变量类型	T_MaxString（带有 255 个字符的 STRING），其他数据类型必须首先转换为字符串，然后再作为字符串格式的数据字段进行读取/写入。	可以写入/读取任何数据类型
示例代码	P_TextModeRead() P_TextModeWrite()	P_BinaryModeRead() P_BinaryModeWrite()

示例项目

项目示例实际上包含 4 个示例：2 个用于在文本模式下的写入/读取访问（首选），2 个用于在二进制模式下的写入/读取访问（罕见）：

P_TextModeRead();

P_TextModeWrite();

P_BinaryModeRead();

P_BinaryModeWrite();

在文本模式下读取 CSV 文件的基本程序序列：

第 1 步：在文本模式下打开 CSV 文件（FB_FileOpen）。如果成功，则转到第 2 步。

第 2 步：使用功能块 FB_FileGets 读取一行。添加一个 CR 字符（请参见表格中的注释）。如果成功，则转到第 3 步，否则转到第 4 步（到达文件末尾或发生错误）。

第 3 步：使用功能块 FB_CSVMemBufferReader 解析读取的行。读取各个数据字段。然后转到第 2 步，读取下一行。重复第 2 步和第 3 步，直至到达文件末尾或发生错误。

第 4 步：关闭 CSV 文件（FB_FileClose）。

在文本模式下写入 CSV 文件的基本程序序列。

第 1 步：在文本模式下打开 CSV 文件（FB_FileOpen）。如果成功，则转到第 2 步。

第 2 步：使用功能块 FB_CSVMemBufferWriter 生成一个新的数据集。将各个数据字段写入缓冲区。该缓冲区可能是一个较大的字符串。移除数据集末尾的 CR 字符，然后转到第 3 步。

第 3 步：使用功能块 FB_FilePuts 写入一行。重复第 2 步和第 3 步，直至写入所有数据集。然后转到第 4 步。

第 4 步：关闭文件（FB_FileClose）。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

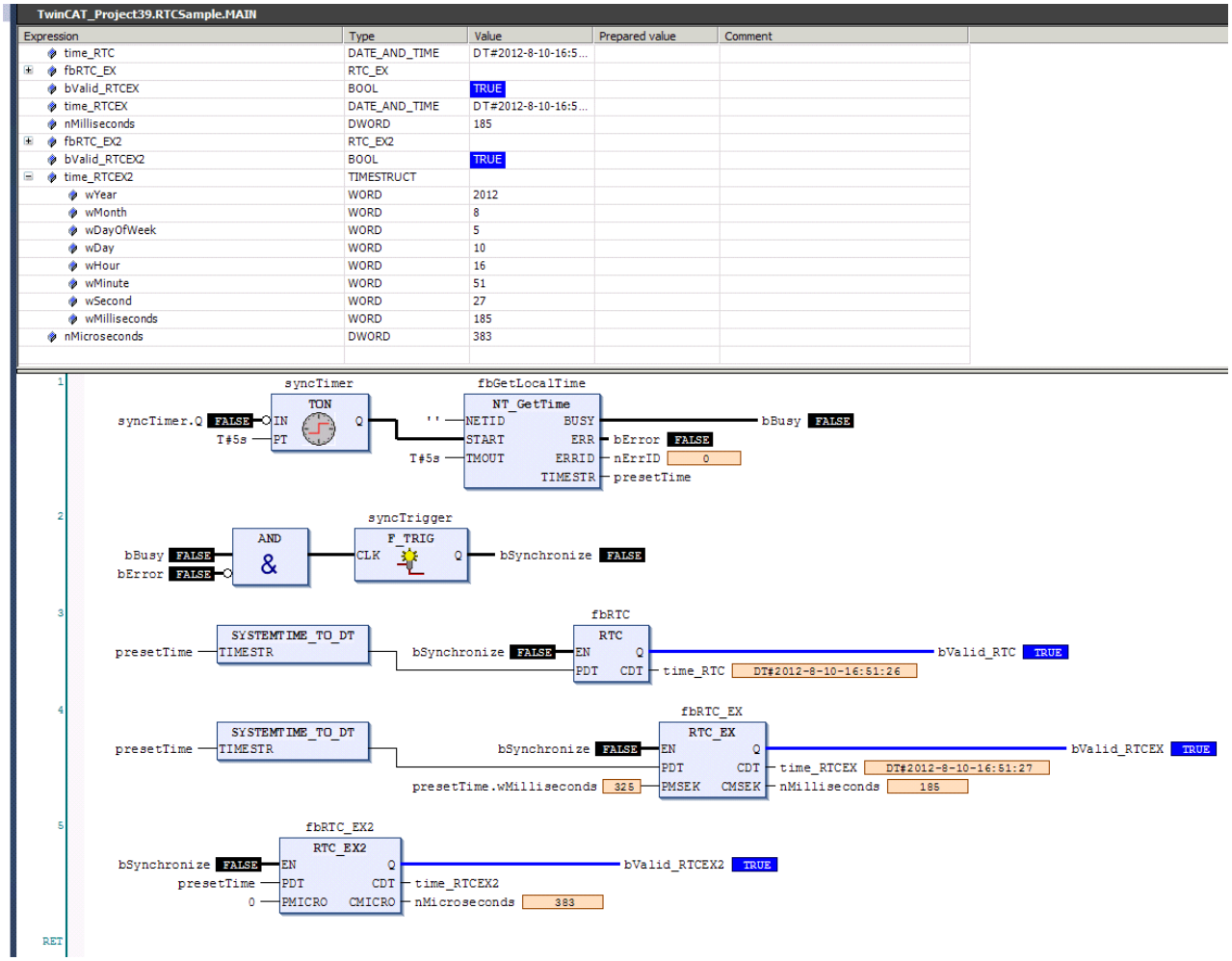
9.9 示例：软件时钟（RTC、RTC_EX、RTC_EX2）

您可以在这里将完整的来源解压缩：https://infosys.beckhoff.com/content/1033/TcPlcLib_Tc2_Utilities/Resources/803608843.zip

在下面的示例中，3 个软件时钟每 5 秒钟与本地 Windows 系统时间同步一次（在任务栏中显示本地 Windows 系统时间）。

```
PROGRAM MAIN
VAR
    fbGetLocalTime  : NT_GetTime;
    bBusy           : BOOL;
    bError          : BOOL;
    nErrID          : UDINT;
    presetTime      : TIMESTRUCT;

    syncTimer       : TON;
```

```
    syncTrigger     : F_TRIG;
    bSynchronize    : BOOL;

    fbRTC           : RTC;
    bValid_RTC      : BOOL;
    time_RTC        : DT;

    fbRTC_EX        : RTC_EX;
    bValid_RTCEX    : BOOL;
    time_RTCEX      : DT;
    nMilliseconds   : DWORD;

    fbRTC_EX2       : RTC_EX2;
    bValid_RTCEX2   : BOOL;
    time_RTCEX2     : TIMESTRUCT;
    nMicroseconds   : DWORD;
END_VAR
```



要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

10 附录

10.1 写入持久性数据时的系统行为

写入触发器	持久性数据访问的内部优化	持久性数据一致性	超出 Plc 循环时间
功能块 WritePersistentData [▶ 146]	无	所有数据均来自同一 plc 周期。	是，如果写入所有数据的时间超过 plc 循环时间的话。
功能块 FB_WritePersistentData [▶ 117] 和 SPDM_2PASS [▶ 327]	是	所有数据均来自同一 plc 周期。	是，如果写入所有数据的时间超过 plc 循环时间的话。
功能块 FB_WritePersistentData [▶ 117] 和 SPDM_VAR_BOOST [▶ 327]	是	每个变量的数据均来自同一 plc 周期。	罕见，如果写入最大的持久变量的时间超过 plc 循环时间的话。
TwinCAT 系统停止（在 TwinCAT 系统停止时，自动写入所有持久性数据）。	是	所有数据均来自同一 plc 周期。	无

10.2 格式规范

FB_FormatString [▶ 63] 和 FB_FormatString2 [▶ 64] 功能块以及 F_FormatArgToStr [▶ 286] 函数会使用该格式规格。格式说明通过字符串输入变量传递给功能块，而通过各个函数参数传递给函数。

格式规范包括各种必填和可选的参数字段：

- 类型 [▶ 373]
- 标志 [▶ 374]
- 宽度 [▶ 375]
- 精度 [▶ 375]

最简单的格式说明仅包含百分号和类型字段（例如 %s）。直到类型字段为止百分号后面的所有字符均被评估为参数字段，直至。百分号之前的字符和类型字段之后的字符会被复制到输出字符串中。如果出现无法识别的字符或非法字符，则会因出错而中止格式化。要在输出字符串中输出百分号，应使用 2 个连续的百分号（%%）。

类型

必填的参数字段。类型字段包含一个 ASCII 字符，该字符可指定相关参数会被解释为字符串、整数，还是浮点数。请注意，一些类型字段参数区分大小写。

类型	参数	输出
b、B	BYTE、WORD、DWORD、REAL[1]、SINT[2]、INT[2]、DINT[2]、USINT、UINT、UDINT	二进制字符串（例如 '101010111000'）
o、O	BYTE、WORD、DWORD、REAL[1]、SINT[2]、INT[2]、DINT[2]、USINT、UINT、UDINT	八进制字符串

类型	参数	输出
u、U	BYTE、WORD、DWORD、SINT[2]、INT[2]、DINT[2]、USINT、UINT、UDINT	不带符号的十进制字符串
c、C	BYTE、USINT	单（ASCII）字节字符
f、F	REAL[3]、LREAL	浮点数 字符串的形式为 **[–]dddd.dddd**，（dddd 为十进制数）。 小数点前的位数取决于浮点数的值。小数点后的位数取决于所需的精度。 仅在负值中才会出现符号。对于无限正值，返回 **'#INF'**，对于无限负值，返回 **'-#INF'**。 如果传输的变量具有非法值（NaN，非数字），则返回 **'#QNAN'** 或 **'-#QNAN'**。 如果格式化的字符串的长度超过了结果字符串的最大允许长度，则返回 **'#OVF'** 或 **'-#OVF'**。
d、D	BYTE、WORD、DWORD、SINT、INT、DINT、USINT、UINT、UDINT	十进制字符串 仅在负值中才会出现符号。
s、S	STRING	单字节字符字符串 输出字符，直至达到最后的零或精度字段参数。
X	BYTE、WORD、DWORD、REAL[1]、SINT[2]、INT[2]、DINT[2]、USINT、UINT、UDINT	十六进制字符串 大写字母（'ABCDEF'）可用于格式化。
x	BYTE、WORD、DWORD、REAL[1]、SINT[2]、INT[2]、DINT[2]、USINT、UINT、UDINT	十六进制字符串 小写字母（'abcdef'）可用于格式化。
E	未执行。留待将来使用。	科学记数法中的浮点数
e	未执行。留待将来使用。	科学记数法中的浮点数

[1] 以二进制、八进制、十六进制或十进制字符串的形式返回 REAL 变量的内容。

[2] 以二进制、八进制、十六进制或十进制字符串的形式返回有符号类型的内容。

[3] 将 REAL 变量转换为 LREAL 类型，然后进行格式化。

标志

可选的参数字段。在标志字段中可以按照任何所需顺序指定一个或多个标志。标志字段参数可指定格式化的值的对齐方式、符号、空格和二进制/八进制/十六进制前缀的输出。

标志	含义	类型	标准
-	左对齐标志。 格式化的值在宽度参数中对齐的宽度内左对齐。	可与所有类型配合使用。	右对齐。
+	符号标志。 强制输出有符号正数的正号。	仅可与 **e**、**E**、**f**、**F**、**d**、**D** 配合使用，否则标志将被忽略。	仅在负值中才会出现负号。
0	零标志。	仅可与 **e**、**E**、**f**、**F**、**s**、**S** 配合使用，否则标志将被忽略。	不用零填充。

标志	含义	类型	标准
	如果该标志位于宽度参数之前，则从左侧用零填充结果字符串，直至达到所需的宽度。	如果额外设置了左对齐标志（-），则零标志也将被忽略。	
空格（' '）	空格标志 正值前面为空白。	仅可与 **e**、**E**、**f**、**F**、**d**、**D** 配合使用，否则标志将被忽略。 如果同时设置了左符号标志（+），则空白标志也将被忽略	无空白。
#	前缀标志。 在格式化的值前可加上 IEC 或标准 C 前缀。 IEC 前缀：**2#**、**8#**、**16#**（默认值） 标准 C 前缀：**0**、**0x**、**0X**	仅可与 **b**、**B**、**o**、**O**、**x**、**X** 配合使用，否则标志将被忽略。 通过在程序中设置全局变量 GLOBAL_FORMAT_HASH_PREFIX_TYPE 可以激活标准 C 前缀类型：GLOBAL_FORMAT_HASH_PREFIX_TYPE := HASHPREFIX_STDC;	无前缀。

宽度

可选的参数字段。宽度字段包含一个正十进制值，该值可指定在输出字符串中输出的最小字符数。

根据对齐标志的不同，将在输出字符串的左侧或右侧添加空格，直至达到所需的宽度。如果在宽度字段参数前放置零标志，则将从左侧用零填充结果字符串，直至达到所需的宽度。输出字符串永远不会被宽度字段参数截断到所需的长度。

还可以为宽度字段参数输入星号（*）。然后，由参数提供所需的值（允许的类型：BYTE、WORD、DWORD、USINT、UINT、UDINT）。然后，宽度字段参数的参数紧跟要格式化的值的参数。

精度

可选的参数字段。精度字段位于点（.）之后，包含一个正十进制值。如果在点后面没有值，则会使用默认精度值（请参见表格）。

类型	含义	标准
b、**B**、**o**、**O**、**u**、**U**、**x**、**X**、**d**、**D**	精度字段参数可指定在输出字符串中输出多少个十进制字符（位数）。 如果没有足够的位数，则从左侧用零填充字符串。输出字符串永远不会被切断。	标准：1
c、**C**	没有意义，可被忽略。	输出一个字符。
f、**F**	精度字段参数可指定小数位数。 参数值总是向上取整为相应的小数位数。	标准：6 个小数位
s、**S**	精度字段参数可指定从参数字符串中输出多少个字符。 不会输出超过精度值的字符。	所有字符都会输出，直至达到最后的零。

还可以为精度字段参数输入星号（*）。然后，由参数提供所需的值（允许的类型：BYTE、WORD、DWORD、USINT、UINT、UDINT）。然后，精度字段参数的参数紧跟要格式化的值的参数。

10.3 格式错误代码

FB_FormatString [▶ 63] 和 FB_FormatString2 [▶ 64] 功能块以及 F_FormatArgToStr [▶ 286] 函数会返回以下错误代码。如果使用多个参数，则除错误代码外，还会返回参数编号（1..9）。参数编号可提供有关在格式化期间检测到错误的具体位置的信息。

错误代码	含义
16#00000000	无错误

错误代码	含义
16#00000010 + 参数编号（1..9）	百分号（%）处于无效的位置
16#00000020 + 参数编号（1..9）	星号参数处于无效的位置
16#00000040 + 参数编号（1..9）	宽度字段值无效
16#00000080 + 参数编号（1..9）	精度字段值无效
16#00000100 + 参数编号（1..9）	其中一个标志处于无效的位置
16#00000200 + 参数编号（1..9）	宽度或精度字段值处于无效的位置
16#00000400 + 参数编号（1..9）	点“.” 精度字段的符号处于无效的位置
16#00000800 + 参数编号（1..9）	类型字段值无效（不支持）
16#00001000 + 参数编号（1..9）	不同的类型字段和参数
16#00002000 + 参数编号（1..9）	格式字符串参数无效
16#00004000 + 参数编号（1..9）	格式字符串中参数过多
16#00008000 + 参数编号（1..9）	目标字符串缓冲区溢出（格式化的字符串过长）
16#00010000 + 参数编号（1..9）	指针输入无效

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

10.4 Scope Server 错误代码

FB_ScopeServerControl [▶ 107] 功能块会返回以下错误代码。

```
TYPE E_UTILITIES_ERRORCODES :
(
   eUtilError_NoError                 := 0,
   eUtilError_ScopeServerNotAvailable := 16#8001,
   eUtilError_ScopeServerStateChange  := 16#8002
);
END_TYPE
```

错误代码	枚举	含义
0x0000	eUtilError_NoError	无错误
0x8001	eUtilError_ScopeServerNotAvailable	TwinCAT Scope Server 不可用。可能尚未安装它。
0x8002	eUtilError_ScopeServerStateChange	不允许请求更改状态。有关有效更改状态的列表，请参见 B_ScopeServerControl 状态图 [▶ 107]。

要求

开发环境	目标平台	要集成的 PLC 库（类别组）
TwinCAT v3.1.0	PC 或 CX（x86、x64、ARM）	Tc2_Utilities（系统）

10.5 ADS 返回代码

错误代码分组：
全局错误代码：--- FEHLENDER LINK ---... （0x9811_0000 ...）
路由器错误代码：--- FEHLENDER LINK ---... （0x9811_0500 ...）
一般 ADS 错误：--- FEHLENDER LINK ---... （0x9811_0700 ...）
RTime 错误代码：--- FEHLENDER LINK ---... （0x9811_1000 ...）

全局错误代码

Hex	Dec	HRESULT	名称	描述
0x0	0	0x98110000	ERR_NOERROR	无错误。
0x1	1	0x98110001	ERR_INTERNAL	内部错误。
0x2	2	0x98110002	ERR_NORTIME	不具有实时性。
0x3	3	0x98110003	ERR_ALLOCLOCKEDMEM	分配已锁定 – 内存错误。
0x4	4	0x98110004	ERR_INSERTMAILBOX	邮箱已满 – 无法发送 ADS 消息。减少每个周期的 ADS 消息数量将有所帮助。
0x5	5	0x98110005	ERR_WRONGRECEIVEHMSG	HMSG 错误。
0x6	6	0x98110006	ERR_TARGETPORTNOTFOUND	未找到目标端口 – ADS 服务器未启动或无法访问。
0x7	7	0x98110007	ERR_TARGETMACHINENOTFOUND	未找到目标计算机 – 未找到 AMS 路由。
0x8	8	0x98110008	ERR_UNKNOWNCMDID	未知命令 ID。
0x9	9	0x98110009	ERR_BADTASKID	任务 ID 无效。
0xA	10	0x9811000A	ERR_NOIO	No IO。
0xB	11	0x9811000B	ERR_UNKNOWNAMSCMD	未知 AMS 命令。
0xC	12	0x9811000C	ERR_WIN32ERROR	Win32 错误。
0xD	13	0x9811000D	ERR_PORTNOTCONNECTED	端口未连接。
0xE	14	0x9811000E	ERR_INVALIDAMSLENGTH	AMS 长度无效。
0xF	15	0x9811000F	ERR_INVALIDAMSNETID	AMS Net ID 无效。
0x10	16	0x98110010	ERR_LOWINSTLEVEL	安装级别 – TwinCAT 2 授权错误。
0x11	17	0x98110011	ERR_NODEBUGINTAVAILABLE	无调试可用。
0x12	18	0x98110012	ERR_PORTDISABLED	端口已禁用 – TwinCAT 系统服务未启动。
0x13	19	0x98110013	ERR_PORTALREADYCONNECTED	端口已连接。
0x14	20	0x98110014	ERR_AMSSYNC_W32ERROR	AMS Sync Win32 错误。
0x15	21	0x98110015	ERR_AMSSYNC_TIMEOUT	AMS Sync 超时。
0x16	22	0x98110016	ERR_AMSSYNC_AMSERROR	AMS Sync 错误。
0x17	23	0x98110017	ERR_AMSSYNC_NOINDEXINMAP	不存在适用于 AMS Sync 的索引映射。
0x18	24	0x98110018	ERR_INVALIDAMSPORT	AMS 端口无效。
0x19	25	0x98110019	ERR_NOMEMORY	无内存。
0x1A	26	0x9811001A	ERR_TCPSEND	TCP 发送错误。
0x1B	27	0x9811001B	ERR_HOSTUNREACHABLE	主机无法访问。
0x1C	28	0x9811001C	ERR_INVALIDAMSFRAGMENT	AMS 片段无效。
0x1D	29	0x9811001D	ERR_TLSSEND	TLS 发送错误 – ADS 安全连接失败。
0x1E	30	0x9811001E	ERR_ACCESSDENIED	拒绝访问 – 拒绝 ADS 安全访问。

路由器错误代码

Hex	Dec	HRESULT	名称	描述
0x500	1280	0x98110500	ROUTERERR_NOLOCKEDMEMORY	无法分配锁定的内存。
0x501	1281	0x98110501	ROUTERERR_RESIZEMEMORY	路由器内存大小无法更改。
0x502	1282	0x98110502	ROUTERERR_MAILBOXFULL	邮箱已达到最大消息数。
0x503	1283	0x98110503	ROUTERERR_DEBUGBOXFULL	调试邮箱已达到最大消息数。
0x504	1284	0x98110504	ROUTERERR_UNKNOWNPORTTYPE	端口类型未知。
0x505	1285	0x98110505	ROUTERERR_NOTINITIALIZED	路由器未初始化。
0x506	1286	0x98110506	ROUTERERR_PORTALREADYINUSE	端口号已分配。
0x507	1287	0x98110507	ROUTERERR_NOTREGISTERED	端口未注册。
0x508	1288	0x98110508	ROUTERERR_NOMOREQUEUES	已达到最大端口数。
0x509	1289	0x98110509	ROUTERERR_INVALIDPORT	端口无效。
0x50A	1290	0x9811050A	ROUTERERR_NOTACTIVATED	路由未激活。
0x50B	1291	0x9811050B	ROUTERERR_FRAGMENTBOXFULL	邮箱已达到最大片段消息数。
0x50C	1292	0x9811050C	ROUTERERR_FRAGMENTTIMEOUT	发生片段超时。
0x50D	1293	0x9811050D	ROUTERERR_TOBEREMOVED	端口已移除。

一般性 ADS 错误代码

Hex	Dec	HRESULT	名称	描述
0x700	1792	0x98110700	ADSERR_DEVICE_ERROR	一般性设备错误。
0x701	1793	0x98110701	ADSERR_DEVICE_SRVNOTSUPP	服务器不支持该服务。
0x702	1794	0x98110702	ADSERR_DEVICE_INVALIDGRP	索引组无效。
0x703	1795	0x98110703	ADSERR_DEVICE_INVALIDOFFSET	索引偏移量无效。
0x704	1796	0x98110704	ADSERR_DEVICE_INVALIDACCESS	不允许读取或写入。 可能有几种原因。例如，创建路由时输入了错误的密码。
0x705	1797	0x98110705	ADSERR_DEVICE_INVALIDSIZE	参数大小不正确。
0x706	1798	0x98110706	ADSERR_DEVICE_INVALIDDATA	无效数据值。
0x707	1799	0x98110707	ADSERR_DEVICE_NOTREADY	设备尚未准备好运行。
0x708	1800	0x98110708	ADSERR_DEVICE_BUSY	设备正忙。
0x709	1801	0x98110709	ADSERR_DEVICE_INVALIDCONTEXT	操作系统上下文无效。这可能是由于在不同的任务中使用 ADS 函数块造成的。可以通过在 PLC 中进行多任务同步解决这个问题。
0x70A	1802	0x9811070A	ADSERR_DEVICE_NOMEMORY	内存不足。
0x70B	1803	0x9811070B	ADSERR_DEVICE_INVALIDPARM	参数值无效。
0x70C	1804	0x9811070C	ADSERR_DEVICE_NOTFOUND	未找到（文件…）。
0x70D	1805	0x9811070D	ADSERR_DEVICE_SYNTAX	文件或命令中存在语法错误。
0x70E	1806	0x9811070E	ADSERR_DEVICE_INCOMPATIBLE	对象不匹配。
0x70F	1807	0x9811070F	ADSERR_DEVICE_EXISTS	对象已存在。
0x710	1808	0x98110710	ADSERR_DEVICE_SYMBOLNOTFOUND	符号未找到。
0x711	1809	0x98110711	ADSERR_DEVICE_SYMBOLVERSIONINVALID	符号版本无效。这可能是由于联机更改造成的。创建新句柄。
0x712	1810	0x98110712	ADSERR_DEVICE_INVALIDSTATE	设备（服务器）处于无效状态。
0x713	1811	0x98110713	ADSERR_DEVICE_TRANSMODENOTSUPP	不支持 AdsTransMode。
0x714	1812	0x98110714	ADSERR_DEVICE_NOTIFYHNDINVALID	通知句柄无效。
0x715	1813	0x98110715	ADSERR_DEVICE_CLIENTUNKNOWN	通知客户端未注册。
0x716	1814	0x98110716	ADSERR_DEVICE_NOMOREHDLS	没有更多的句柄可用。
0x717	1815	0x98110717	ADSERR_DEVICE_INVALIDWATCHSIZE	通知大小过大。
0x718	1816	0x98110718	ADSERR_DEVICE_NOTINIT	设备未初始化。
0x719	1817	0x98110719	ADSERR_DEVICE_TIMEOUT	设备超时。
0x71A	1818	0x9811071A	ADSERR_DEVICE_NOINTERFACE	接口查询失败。
0x71B	1819	0x9811071B	ADSERR_DEVICE_INVALIDINTERFACE	请求的接口错误。
0x71C	1820	0x9811071C	ADSERR_DEVICE_INVALIDCLSID	Class ID 无效。
0x71D	1821	0x9811071D	ADSERR_DEVICE_INVALIDOBJID	Object ID 无效。
0x71E	1822	0x9811071E	ADSERR_DEVICE_PENDING	请求待定。
0x71F	1823	0x9811071F	ADSERR_DEVICE_ABORTED	请求已中止。
0x720	1824	0x98110720	ADSERR_DEVICE_WARNING	警告信号。
0x721	1825	0x98110721	ADSERR_DEVICE_INVALIDARRAYIDX	数组索引无效。
0x722	1826	0x98110722	ADSERR_DEVICE_SYMBOLNOTACTIVE	符号未激活。
0x723	1827	0x98110723	ADSERR_DEVICE_ACCESSDENIED	拒绝访问。 有几种可能的原因。例如，单向 ADS 路由用于相反方向。
0x724	1828	0x98110724	ADSERR_DEVICE_LICENSENOTFOUND	缺少授权。
0x725	1829	0x98110725	ADSERR_DEVICE_LICENSEEXPIRED	授权已过期。
0x726	1830	0x98110726	ADSERR_DEVICE_LICENSEEXCEEDED	超出授权。
0x727	1831	0x98110727	ADSERR_DEVICE_LICENSEINVALID	授权无效。
0x728	1832	0x98110728	ADSERR_DEVICE_LICENSESYSTEMID	授权问题：系统 ID 无效。
0x729	1833	0x98110729	ADSERR_DEVICE_LICENSENOTIMELIMIT	授权不受时间限制。
0x72A	1834	0x9811072A	ADSERR_DEVICE_LICENSEFUTUREISSUE	授权问题：未来的时间。
0x72B	1835	0x9811072B	ADSERR_DEVICE_LICENSETIMETOLONG	授权期限太长。
0x72C	1836	0x9811072C	ADSERR_DEVICE_EXCEPTION	系统启动异常。
0x72D	1837	0x9811072D	ADSERR_DEVICE_LICENSEDUPLICATED	授权文件读取了两次。
0x72E	1838	0x9811072E	ADSERR_DEVICE_SIGNATUREINVALID	签名无效。
0x72F	1839	0x9811072F	ADSERR_DEVICE_CERTIFICATEINVALID	证书无效。
0x730	1840	0x98110730	ADSERR_DEVICE_LICENSEOEMNOTFOUND	OEM未知公钥。
0x731	1841	0x98110731	ADSERR_DEVICE_LICENSERESTRICTED	此系统 ID 授权无效。

Hex	Dec	HRESULT	名称	描述
0x732	1842	0x98110732	ADSERR_DEVICE_LICENSEDEMODENIED	演示授权已禁止。
0x733	1843	0x98110733	ADSERR_DEVICE_INVALIDFNCID	无效函数 ID。
0x734	1844	0x98110734	ADSERR_DEVICE_OUTOFRANGE	超出有效范围。
0x735	1845	0x98110735	ADSERR_DEVICE_INVALIDALIGNMENT	无效对齐。
0x736	1846	0x98110736	ADSERR_DEVICE_LICENSEPLATFORM	无效平台级别。
0x737	1847	0x98110737	ADSERR_DEVICE_FORWARD_PL	上下文 – 转到被动级别。
0x738	1848	0x98110738	ADSERR_DEVICE_FORWARD_DL	上下文 – 转到调度级别。
0x739	1849	0x98110739	ADSERR_DEVICE_FORWARD_RT	上下文 – 转到实时。
0x740	1856	0x98110740	ADSERR_CLIENT_ERROR	客户端错误。
0x741	1857	0x98110741	ADSERR_CLIENT_INVALIDPARM	服务包含无效参数。
0x742	1858	0x98110742	ADSERR_CLIENT_LISTEMPTY	轮询列表为空。
0x743	1859	0x98110743	ADSERR_CLIENT_VARUSED	Var 连接已在使用中。
0x744	1860	0x98110744	ADSERR_CLIENT_DUPLINVOKEID	调用的 ID 已在使用中。
0x745	1861	0x98110745	ADSERR_CLIENT_SYNCTIMEOUT	已发生超时 – 远程终端在指定的 ADS 超时中未响应。远程终端的路由设置可能配置不正确。
0x746	1862	0x98110746	ADSERR_CLIENT_W32ERROR	Win32 子系统中发生错误。
0x747	1863	0x98110747	ADSERR_CLIENT_TIMEOUTINVALID	客户端超时值无效。
0x748	1864	0x98110748	ADSERR_CLIENT_PORTNOTOPEN	端口未打开。
0x749	1865	0x98110749	ADSERR_CLIENT_NOAMSADDR	无 AMS 地址。
0x750	1872	0x98110750	ADSERR_CLIENT_SYNCINTERNAL	Ads sync 中发生内部错误。
0x751	1873	0x98110751	ADSERR_CLIENT_ADDHASH	哈希表溢出。
0x752	1874	0x98110752	ADSERR_CLIENT_REMOVEHASH	未在表格中找到密钥。
0x753	1875	0x98110753	ADSERR_CLIENT_NOMORESYM	缓存中没有符号。
0x754	1876	0x98110754	ADSERR_CLIENT_SYNCRESINVALID	收到无效响应。
0x755	1877	0x98110755	ADSERR_CLIENT_SYNCPORTLOCKED	同步端口已锁定。
0x756	1878	0x98110756	ADSERR_CLIENT_REQUESTCANCELLED	请求被取消。

RTime 错误代码

Hex	Dec	HRESULT	名称	描述
0x1000	4096	0x98111000	RTERR_INTERNAL	实时系统中发生内部错误。
0x1001	4097	0x98111001	RTERR_BADTIMERPERIODS	计时器值无效。
0x1002	4098	0x98111002	RTERR_INVALIDTASKPTR	任务指针提供了无效值 0（零）。
0x1003	4099	0x98111003	RTERR_INVALIDSTACKPTR	堆栈指针提供了无效值 0（零）。
0x1004	4100	0x98111004	RTERR_PRIOEXISTS	请求任务优先级已分配。
0x1005	4101	0x98111005	RTERR_NOMORETCB	没有可用的空闲 TCB（任务控制块）。TCB 的最大数量为 64。
0x1006	4102	0x98111006	RTERR_NOMORESEMAS	没有可用的空闲信号。信号的最大数量为 64。
0x1007	4103	0x98111007	RTERR_NOMOREQUEUES	队列中没有可用的空闲空间。队列中的最大位置数为 64。
0x100D	4109	0x9811100D	RTERR_EXTIRQALREADYDEF	已应用外部同步中断。
0x100E	4110	0x9811100E	RTERR_EXTIRQNOTDEF	未应用外部同步中断。
0x100F	4111	0x9811100F	RTERR_EXTIRQINSTALLFAILED	外部同步中断应用失败。
0x1010	4112	0x98111010	RTERR_IRQLNOTLESSOREQUAL	在错误的上下文中调用服务函数
0x1017	4119	0x98111017	RTERR_VMXNOTSUPPORTED	不支持 Intel VT-x 扩展。
0x1018	4120	0x98111018	RTERR_VMXDISABLED	BIOS 中未启用 Intel VT-x 扩展。
0x1019	4121	0x98111019	RTERR_VMXCONTROLSMISSING	Intel VT-x 扩展中缺少函数。
0x101A	4122	0x9811101A	RTERR_VMXENABLEFAILS	Intel VT-x 激活失败。

具体的正向 HRESULT 返回代码：

HRESULT	名称	描述
0x0000_0000	S_OK	无错误。
0x0000_0001	S_FALSE	无错误。 示例：处理成功，但有一个负面或不完整的结果。
0x0000_0203	S_PENDING	无错误。 示例：处理成功，但还没有结果。

HRESULT	名称	描述
0x0000_0256	S_WATCHDOG_TIMEOUT	无错误。 示例：处理成功，但发生了超时。

TCP Winsock 错误代码

Hex	Dec	名称	描述
0x274C	10060	WSAETIMEDOUT	发生连接超时 - 在创建连接时发生错误，因为远程终端在特定时间后未正确响应，或者所建立连接因所连接主机未响应而无法维持。
0x274D	10061	WSAECONNREFUSED	连接遭到拒绝 - 无法建立连接，因为目标计算机明确拒绝了该连接。此错误通常由尝试连接到外部主机上处于非活动状态的服务（即没有运行服务器应用程序的服务）导致。
0x2751	10065	WSAEHOSTUNREACH	不存在至主机的路由 - 套接字操作引用了不可用的主机。
更多 Winsock 错误代码：Win32 错误代码 [▶ 380]			

还请参阅有关此

- ADS 返回代码 [▶ 376]
- ADS 返回代码 [▶ 377]
- ADS 返回代码 [▶ 377]
- ADS 返回代码 [▶ 379]

10.6 Win32 错误代码

下表提供了一个 Win32 错误代码的列表。

0 [▶ 380]、100 [▶ 381]、200 [▶ 383]、1001 [▶ 384]、1100 [▶ 386]、1200 [▶ 387]、1400 [▶ 391]、1600 [▶ 392]、1800 [▶ 395]、2000 [▶ 396]、3000 [▶ 397]、5000 [▶ 399]、6000 [▶ 401]、8000 [▶ 402]、8500 [▶ 409]、9001 [▶ 411]、10004 [▶ 413]、12000 [▶ 415]、

错误			描述
十进制	十六进制	名称	
0	0x00000000	ERROR_SUCCESS	该操作成功完成。
1	0x00000001	ERROR_INVALID_FUNCTION	功能不正确。
2	0x00000002	ERROR_FILE_NOT_FOUND	系统无法找到指定的文件。
3	0x00000003	ERROR_PATH_NOT_FOUND	系统无法找到指定的路径。
4	0x00000004	ERROR_TOO_MANY_OPEN_FILES	系统无法打开该文件。
5	0x00000005	ERROR_ACCESS_DENIED	访问遭到拒绝。
6	0x00000006	ERROR_INVALID_HANDLE	句柄无效。
7	0x00000007	ERROR_ARENA_TRASHED	存储控制块遭到破坏。
8	0x00000008	ERROR_NOT_ENOUGH_MEMORY	没有足够的存储空间来处理这个命令。
9	0x00000009	ERROR_INVALID_BLOCK	存储控制块地址无效。
10	0x0000000A	ERROR_BAD_ENVIRONMENT	环境不正确。
11	0x0000000B	ERROR_BAD_FORMAT	试图加载格式不正确的程序。
12	0x0000000C	ERROR_INVALID_ACCESS	访问代码无效。
13	0x0000000D	ERROR_INVALID_DATA	数据无效。
14	0x0000000E	ERROR_OUTOFMEMORY	没有足够的存储空间来完成这一操作。
15	0x0000000F	ERROR_INVALID_DRIVE	系统无法找到指定的驱动器。
16	0x00000010	ERROR_CURRENT_DIRECTORY	该目录无法删除。
17	0x00000011	ERROR_NOT_SAME_DEVICE	系统不能将文件移动到不同的磁盘驱动器。
18	0x00000012	ERROR_NO_MORE_FILES	已经没有文件了。
19	0x00000013	ERROR_WRITE_PROTECT	媒体受到写入保护。
20	0x00000014	ERROR_BAD_UNIT	系统无法找到指定设备。
21	0x00000015	ERROR_NOT_READY	设备还没有准备好。
22	0x00000016	ERROR_BAD_COMMAND	设备不能识别该命令。
23	0x00000017	ERROR_CRC	数据错误（循环冗余检查）。
24	0x00000018	ERROR_BAD_LENGTH	程序发出了一条指令，但指令长度不正确。
25	0x00000019	ERROR_SEEK	驱动器无法定位磁盘上的特定区域或轨道。

错误			描述
十进制	十六进制	名称	
26	0x0000001A	ERROR_NOT_DOS_DISK	无法访问指定的磁盘或软盘。
27	0x0000001B	ERROR_SECTOR_NOT_FOUND	驱动器无法找到要求的扇区。
28	0x0000001C	ERROR_OUT_OF_PAPER	打印机没纸了。
29	0x0000001D	ERROR_WRITE_FAULT	系统不能向指定的设备写入。
30	0x0000001E	ERROR_READ_FAULT	系统不能从指定的设备中读取。
31	0x0000001F	ERROR_GEN_FAILURE	连接到系统上的一个设备没有发挥作用。
32	0x00000020	ERROR_SHARING_VIOLATION	该进程不能访问该文件，因为它正被另一个进程使用。
33	0x00000021	ERROR_LOCK_VIOLATION	该进程不能访问该文件，因为另一个进程锁定了该文件的一部分。
34	0x00000022	ERROR_WRONG_DISK	驱动器中出现了错误的软盘。将 %2（卷序列号：%3）插入驱动器 %1。
36	0x00000024	ERROR_SHARING_BUFFER_EXCEEDED	为共享而打开的文件太多。
38	0x00000026	ERROR_HANDLE_EOF	达到了文件的终点。
39	0x00000027	ERROR_HANDLE_DISK_FULL	磁盘已满。
50	0x00000032	ERROR_NOT_SUPPORTED	该请求不被支持。
51	0x00000033	ERROR_REM_NOT_LIST	远程计算机不可用。
52	0x00000034	ERROR_DUP_NAME	网络上存在一个重复的名字。
53	0x00000035	ERROR_BAD_NETPATH	未找到网络路径。
54	0x00000036	ERROR_NETWORK_BUSY	网络很忙。
55	0x00000037	ERROR_DEV_NOT_EXIST	指定的网络资源或设备已不再可用。
56	0x00000038	ERROR_TOO_MANY_CMDS	已达到网络 BIOS 命令限制。
57	0x00000039	ERROR_ADAP_HDW_ERR	发生了一个网络适配器硬件错误。
58	0x0000003A	ERROR_BAD_NET_RESP	指定的服务器不能执行请求的操作。
59	0x0000003B	ERROR_UNEXP_NET_ERR	发生了一个意外的网络错误。
60	0x0000003C	ERROR_BAD_REM_ADAP	遥控适配器不兼容。
61	0x0000003D	ERROR_PRINTQ_FULL	打印机队列已满。
62	0x0000003E	ERROR_NO_SPOOL_SPACE	在服务器上没有存储等待打印文件的空间。
63	0x0000003F	ERROR_PRINT_CANCELLED	等待打印的文件已被删除。
64	0x00000040	ERROR_NETNAME_DELETED	指定的网络名称已不再可用。
65	0x00000041	ERROR_NETWORK_ACCESS_DENIED	网络访问被拒绝。
66	0x00000042	ERROR_BAD_DEV_TYPE	网络资源类型不正确。
67	0x00000043	ERROR_BAD_NET_NAME	无法找到网络名称。
68	0x00000044	ERROR_TOO_MANY_NAMES	超过了本地计算机网络适配器卡的名称限制。
69	0x00000045	ERROR_TOO_MANY_SESS	超过了网络 BIOS 会话限制。
70	0x00000046	ERROR_SHARING_PAUSED	远程服务器已经暂停或正在启动。
71	0x00000047	ERROR_REQ_NOT_ACCEP	此时不能再与该远程计算机进行连接，因为计算机能接受的连接数已经很多了。
72	0x00000048	ERROR_REDIR_PAUSED	指定的打印机或磁盘设备已暂停使用。
80	0x00000050	ERROR_FILE_EXISTS	该文件存在。
82	0x00000052	ERROR_CANNOT_MAKE	该目录或文件不能被创建。
83	0x00000053	ERROR_FAIL_I24	在 INT 24 上失败。
84	0x00000054	ERROR_OUT_OF_STRUCTURES	处理这一请求的存储空间不可用。
85	0x00000055	ERROR_ALREADY_ASSIGNED	本地设备名称已经在使用。
86	0x00000056	ERROR_INVALID_PASSWORD	指定的网络密码不正确。
87	0x00000057	ERROR_INVALID_PARAMETER	该参数不正确。
88	0x00000058	ERROR_NET_WRITE_FAULT	网络上发生了一个写入故障。
89	0x00000059	ERROR_NO_PROC_SLOTS	系统此时不能启动另一个进程。

错误			描述
十进制	十六进制	名称	
100	0x00000064	ERROR_TOO_MANY_SEMAPHORES	无法创建另一个系统信号。
101	0x00000065	ERROR_EXCL_SEM_ALREADY_OWNED	独占信号被另一个进程所拥有。
102	0x00000066	ERROR_SEM_IS_SET	该信号已设置，且无法关闭。
103	0x00000067	ERROR_TOO_MANY_SEM_REQUESTS	该信号无法再设置。

错误			描述
十进制	十六进制	名称	
104	0x00000068	ERROR_INVALID_AT_INTERRUPT_TIME	无法在中断时请求独占的信号系统。
105	0x00000069	ERROR_SEM_OWNER_DIED	此信号的先前所有权已经结束。
106	0x0000006A	ERROR_SEM_USER_LIMIT	插入驱动器的软盘 %1。
107	0x0000006B	ERROR_DISK_CHANGE	由于没有插入备用软盘，程序停止。
108	0x0000006C	ERROR_DRIVE_LOCKED	该磁盘正在使用中或被其他进程锁定。
109	0x0000006D	ERROR_BROKEN_PIPE	该管道已经结束。
110	0x0000006E	ERROR_OPEN_FAILED	系统无法打开指定的设备或文件。
111	0x0000006F	ERROR_BUFFER_OVERFLOW	文件名太长。
112	0x00000070	ERROR_DISK_FULL	磁盘上没有足够的空间。
113	0x00000071	ERROR_NO_MORE_SEARCH_HANDLES	没有更多的内部文件标识符可用。
114	0x00000072	ERROR_INVALID_TARGET_HANDLE	目标内部文件标识符不正确。
117	0x00000075	ERROR_INVALID_CATEGORY	应用程序进行的 IOCTL 调用不正确。
118	0x00000076	ERROR_INVALID_VERIFY_SWITCH	写入时验证开关的参数值不正确。
119	0x00000077	ERROR_BAD_DRIVER_LEVEL	系统不支持所请求的命令。
120	0x00000078	ERROR_CALL_NOT_IMPLEMENTED	此系统不支持此功能。
121	0x00000079	ERROR_SEM_TIMEOUT	信号的超时时间已过。
122	0x0000007A	ERROR_INSUFFICIENT_BUFFER	传递给系统调用的数据区域太小。
123	0x0000007B	ERROR_INVALID_NAME	文件名、目录名或卷标的语法不正确。
124	0x0000007C	ERROR_INVALID_LEVEL	系统调用级别不正确。
125	0x0000007D	ERROR_NO_VOLUME_LABEL	该磁盘没有卷标。
126	0x0000007E	ERROR_MOD_NOT_FOUND	无法找到指定的模块。
127	0x0000007F	ERROR_PROC_NOT_FOUND	无法找到指定的程序。
128	0x00000080	ERROR_WAIT_NO_CHILDREN	不存在需要等待的子进程。
129	0x00000081	ERROR_CHILD_NOT_COMPLETE	%1 应用程序不能在 Win32 模式下运行。
130	0x00000082	ERROR_DIRECT_ACCESS_HANDLE	试图使用一个文件句柄到一个开放的磁盘分区进行原始磁盘 I/O 以外的操作。
131	0x00000083	ERROR_NEGATIVE_SEEK	试图将文件指针移到文件的开头之前。
132	0x00000084	ERROR_SEEK_ON_DEVICE	无法在指定的设备或文件上设置文件指针。
133	0x00000085	ERROR_IS_JOIN_TARGET	JOIN 或 SUBST 命令不能用于包含先前加入驱动器的驱动器。
134	0x00000086	ERROR_IS_JOINED	试图在已经加入的驱动器上使用 JOIN 或 SUBST 命令。
135	0x00000087	ERROR_IS_SUBSTED	试图在一个已经被替换的驱动器上使用 JOIN 或 SUBST 命令。
136	0x00000088	ERROR_NOT_JOINED	系统试图删除未加入驱动器的 JOIN。
137	0x00000089	ERROR_NOT_SUBSTED	系统试图删除一个没有被替换的驱动器的替换。
138	0x0000008A	ERROR_JOIN_TO_JOIN	系统试图将一个驱动器连接到一个已连接的驱动器上的目录。
139	0x0000008B	ERROR_SUBST_TO_SUBST	系统试图将一个驱动器替换为一个被替换的驱动器上的目录。
140	0x0000008C	ERROR_JOIN_TO_SUBST	系统试图将一个驱动器连接到一个被替换的驱动器上的目录。
141	0x0000008D	ERROR_SUBST_TO_JOIN	系统尝试将一个驱动器替换为一个连接的驱动器上的目录。
142	0x0000008E	ERROR_BUSY_DRIVE	系统此时不能执行 JOIN 或 SUBST。
143	0x0000008F	ERROR_SAME_DRIVE	系统不能将一个驱动器加入或替代同一驱动器上的目录。
144	0x00000090	ERROR_DIR_NOT_ROOT	该目录不是根目录的一个子目录。
145	0x00000091	ERROR_DIR_NOT_EMPTY	该目录不是空的。
146	0x00000092	ERROR_IS_SUBST_PATH	指定的路径正在被替代使用。
147	0x00000093	ERROR_IS_JOIN_PATH	没有足够的资源来处理这个命令。
148	0x00000094	ERROR_PATH_BUSY	此时不能使用指定的路径。
149	0x00000095	ERROR_IS_SUBST_TARGET	试图加入或替代一个驱动器，而该驱动器上的一个目录是先前替代的目标。
150	0x00000096	ERROR_SYSTEM_TRACE	在你的 CONFIG.SYS 文件中没有指定系统跟踪信息，或者跟踪是被禁止的。

错误			描述
十进制	十六进制	名称	
151	0x00000097	ERROR_INVALID_EVENT_COUNT	DosMuxSemWait 指定信号事件的数量不正确。
152	0x00000098	ERROR_TOO_MANY_MUXWAITERS	DosMuxSemWait 没有执行；已经设置了太多的信号。
153	0x00000099	ERROR_INVALID_LIST_FORMAT	DosMuxSemWait 列表不正确。
154	0x0000009A	ERROR_LABEL_TOO_LONG	您输入的卷标超过了目标文件系统的标签字符限制。
155	0x0000009B	ERROR_TOO_MANY_TCBS	无法创建另一个线程。
156	0x0000009C	ERROR_SIGNAL_REFUSED	接收进程已拒绝信号。
157	0x0000009D	ERROR_DISCARDED	该段已被丢弃，且无法锁定。
158	0x0000009E	ERROR_NOT_LOCKED	该段已被解锁。
159	0x0000009F	ERROR_BAD_THREADID_ADDR	线程 ID 的地址不正确。
160	0x000000A0	ERROR_BAD_ARGUMENTS	传递给 DosExecPgm 的参数字符串不正确。
161	0x000000A1	ERROR_BAD_PATHNAME	指定的路径无效。
162	0x000000A2	ERROR_SIGNAL_PENDING	已经有一个信号正在等待。
164	0x000000A4	ERROR_MAX_THRDS_REACHED	系统中不能再创建线程了。
167	0x000000A7	ERROR_LOCK_FAILED	无法锁定一个文件的某个区域。
170	0x000000AA	ERROR_BUSY	请求的资源正在使用中。
173	0x000000AD	ERROR_CANCEL_VIOLATION	对于所提供的取消区域，某个锁请求未完成。
174	0x000000AE	ERROR_ATOMIC_LOCKS_NOT_SUPPORTED	文件系统不支持对锁类型进行原子更改。
180	0x000000B4	ERROR_INVALID_SEGMENT_NUMBER	系统检测到一个不正确的段号。
182	0x000000B6	ERROR_INVALID_ORDINAL	操作系统不能运行 %1。
183	0x000000B7	ERROR_ALREADY_EXISTS	当一个文件已经存在时，不能创建该文件。
186	0x000000BA	ERROR_INVALID_FLAG_NUMBER	所传递的标志不正确。
187	0x000000BB	ERROR_SEM_NOT_FOUND	没有找到指定的系统信号名称。
188	0x000000BC	ERROR_INVALID_STARTING_CODESEG	操作系统不能运行 %1。
189	0x000000BD	ERROR_INVALID_STACKSEG	操作系统不能运行 %1。
190	0x000000BE	ERROR_INVALID_MODULETYPE	操作系统不能运行 %1。
191	0x000000BF	ERROR_INVALID_EXE_SIGNATURE	不能在 Win32 模式下运行 %1。
192	0x000000C0	ERROR_EXE_MARKED_INVALID	操作系统不能运行 %1。
193	0x000000C1	ERROR_BAD_EXE_FORMAT	%1 不是一个有效的 Win32 应用程序。
194	0x000000C2	ERROR_ITERATED_DATA_EXCEEDS_64k	操作系统不能运行 %1。
195	0x000000C3	ERROR_INVALID_MINALLOCSIZE	操作系统不能运行 %1。
196	0x000000C4	ERROR_DYNLINK_FROM_INVALID_RING	操作系统不能运行这个应用程序。
197	0x000000C5	ERROR_IOPL_NOT_ENABLED	操作系统目前没有配置为运行该应用程序。
198	0x000000C6	ERROR_INVALID_SEGDPL	操作系统不能运行 %1。
199	0x000000C7	ERROR_AUTODATASEG_EXCEEDS_64k	操作系统不能运行这个应用程序。

错误			描述
十进制	十六进制	名称	
200	0x000000C8	ERROR_RING2SEG_MUST_BE_MOVABLE	代码段不能大于或等于 64K。
201	0x000000C9	ERROR_RELOC_CHAIN_XEEDS_SEGLIM	操作系统不能运行 %1。
202	0x000000CA	ERROR_INFLOOP_IN_RELOC_CHAIN	操作系统不能运行 %1。
203	0x000000CB	ERROR_ENVVAR_NOT_FOUND	系统无法找到所输入的环境选项。
205	0x000000CD	ERROR_NO_SIGNAL_SENT	命令子树中没有进程有信号处理程序。
206	0x000000CE	ERROR_FILENAME_EXCED_RANGE	文件名或扩展名太长。
207	0x000000CF	ERROR_RING2_STACK_IN_USE	2 号环形堆栈正在使用中。
208	0x000000D0	ERROR_META_EXPANSION_TOO_LONG	全局文件名字符 * 或 ? 输入不正确，或指定了太多的全局文件名字符。
209	0x000000D1	ERROR_INVALID_SIGNAL_NUMBER	所发布的信号并不正确。
210	0x000000D2	ERROR_THREAD_1_INACTIVE	信号处理程序不能被设置。
212	0x000000D4	ERROR_LOCKED	该段被锁定，且不能被重新分配。
214	0x000000D6	ERROR_TOO_MANY_MODULES	该程序或动态链接模块已连接了太多的动态链接模块。
215	0x000000D7	ERROR_NESTING_NOT_ALLOWED	不能对 LoadModule 进行嵌套调用。

错误			描述
十进制	十六进制	名称	
216	0x000000D8	ERROR_EXE_MACHINE_TYPE_MISMATCH	镜像文件 %1 是有效的，但用于当前机器以外的机器类型。
230	0x000000E6	ERROR_BAD_PIPE	管道状态无效。
231	0x000000E7	ERROR_PIPE_BUSY	所有的管道实例都很忙。
232	0x000000E8	ERROR_NO_DATA	该管道正在被关闭。
233	0x000000E9	ERROR_PIPE_NOT_CONNECTED	管道的另一端没有进程。
234	0x000000EA	ERROR_MORE_DATA	有更多的数据。
240	0x000000F0	ERROR_VC_DISCONNECTED	该会话被取消了。
254	0x000000FE	ERROR_INVALID_EA_NAME	指定的扩展属性名称无效。
255	0x000000FF	ERROR_EA_LIST_INCONSISTENT	扩展属性不一致。
258	0x00000102	WAIT_TIMEOUT	等待操作已经超时。
259	0x00000103	ERROR_NO_MORE_ITEMS	没有更多的数据了。
266	0x0000010A	ERROR_CANNOT_COPY	不能使用复制功能。
267	0x0000010B	ERROR_DIRECTORY	目录名称无效。
275	0x00000113	ERROR_EAS_DIDNT_FIT	扩展属性不适合在缓冲区内。
276	0x00000114	ERROR_EA_FILE_CORRUPT	挂载的文件系统上的扩展属性文件已损坏。
277	0x00000115	ERROR_EA_TABLE_FULL	扩展属性表文件已满。
278	0x00000116	ERROR_INVALID_EA_HANDLE	指定的扩展属性句柄无效。
282	0x0000011A	ERROR_EAS_NOT_SUPPORTED	挂载的文件系统不支持扩展属性。
288	0x00000120	ERROR_NOT_OWNER	试图释放不属于调用者的 mutex。
298	0x0000012A	ERROR_TOO_MANY_POSTS	对信号量进行了太多的发布操作。
299	0x0000012B	ERROR_PARTIAL_COPY	仅部分 ReadProcessMemory 或 WriteProcessMemory 请求完成。
300	0x0000012C	ERROR_OPLOCK_NOT_GRANTED	oplock 请求被拒绝。
301	0x0000012D	ERROR_INVALID_OPLOCK_PROTOCOL	系系统收到了无效的操作锁定确认。
302	0x0000012E	ERROR_DISK_TOO_FRAGMENTED	卷的碎片太多，无法完成这个操作。
303	0x0000012F	ERROR_DELETE_PENDING	该文件无法打开，因为它正在被删除过程中。
317	0x0000013D	ERROR_MR_MID_NOT_FOUND	系统无法在 %2 的信息文件中找到信息编号 0x%1 的信息文本。
487	0x000001E7	ERROR_INVALID_ADDRESS	试图访问无效的地址。
534	0x00000216	ERROR_ARITHMETIC_OVERFLOW	计算结果超过了 32 位。
535	0x00000217	ERROR_PIPE_CONNECTED	在通道的另一端有一个过程。
536	0x00000218	ERROR_PIPE_LISTENING	等待一个进程来打开通道的另一端。
994	0x000003E2	ERROR_EA_ACCESS_DENIED	拒绝了对扩展属性的访问。
995	0x000003E3	ERROR_OPERATION_ABORTED	由于线程退出或应用程序请求，I/O 操作已被中止。
996	0x000003E4	ERROR_IO_INCOMPLETE	重叠的 I/O 事件没有处于信号状态。
997	0x000003E5	ERROR_IO_PENDING	正在进行重叠的 I/O 操作。
998	0x000003E6	ERROR_NOACCESS	内存位置访问无效。
999	0x000003E7	ERROR_SWAPERROR	执行页内操作错误。

错误			描述
十进制	十六进制	名称	
1001	0x000003E9	ERROR_STACK_OVERFLOW	递归太深；堆栈溢出。
1002	0x000003EA	ERROR_INVALID_MESSAGE	窗口不能对发送的信息采取行动。
1003	0x000003EB	ERROR_CAN_NOT_COMPLETE	无法完成此功能。
1004	0x000003EC	ERROR_INVALID_FLAGS	无效的标志。
1005	0x000003ED	ERROR_UNRECOGNIZED_VOLUME	卷中不包含被识别的文件系统。请确保所有需要的文件系统驱动程序都已加载，并且卷没有损坏。
1006	0x000003EE	ERROR_FILE_INVALID	一个文件的卷已经被外部更改，所以打开的文件不再有效。
1007	0x000003EF	ERROR_FULLSCREEN_MODE	在全屏模式下无法执行所请求的操作。
1008	0x000003F0	ERROR_NO_TOKEN	试图引用一个不存在的令牌。
1009	0x000003F1	ERROR_BADDB	配置注册表数据库已损坏。
1010	0x000003F2	ERROR_BADKEY	配置注册表的键值无效。
1011	0x000003F3	ERROR_CANTOPEN	配置注册表键无法打开。

错误			描述
十进制	十六进制	名称	
1012	0x000003F4	ERROR_CANTREAD	无法读取配置注册表的键值。
1013	0x000003F5	ERROR_CANTWRITE	配置注册表键无法写入。
1014	0x000003F6	ERROR_REGISTRY_RECOVERED	注册表数据库中的一个文件必须通过使用日志或备用副本来恢复。恢复成功。
1015	0x000003F7	ERROR_REGISTRY_CORRUPT	注册表损坏。包含注册表数据的文件之一的结构已损坏，或者系统内存中该文件的映像已损坏，或者无法恢复该文件，因为备用副本或日志不存在或已损坏。
1016	0x000003F8	ERROR_REGISTRY_IO_FAILED	一个由注册表发起的 I/O 操作失败，不可恢复。注册表无法读取、写入或刷新其中一个包含系统注册表图像的文件。
1017	0x000003F9	ERROR_NOT_REGISTRY_FILE	系统试图加载或恢复一个文件到注册表，但指定的文件不是注册表文件格式。
1018	0x000003FA	ERROR_KEY_DELETED	试图对一个已被标记为删除的注册表键进行非法操作。
1019	0x000003FB	ERROR_NO_LOG_SPACE	系统无法在注册表日志中分配所需空间。
1020	0x000003FC	ERROR_KEY_HAS_CHILDREN	不能在已经有子键或值的注册表键中创建符号链接。
1021	0x000003FD	ERROR_CHILD_MUST_BE_VOLATILE	无法在不稳定的父键下创建稳定的子键。
1022	0x000003FE	ERROR_NOTIFY_ENUM_DIR	一个通知变更请求正在完成，且信息没有被返回到调用者的缓冲区。调用者现在需要列举文件以找到更改。
1051	0x0000041B	ERROR_DEPENDENT_SERVICES_RUNNING	已经向其他正在运行的服务依赖的一个服务发送了停止控制。
1052	0x0000041C	ERROR_INVALID_SERVICE_CONTROL	请求的控制对该服务无效。
1053	0x0000041D	ERROR_SERVICE_REQUEST_TIMEOUT	该服务没有及时响应启动或控制请求。
1054	0x0000041E	ERROR_SERVICE_NO_THREAD	无法为该服务创建一个线程。
1055	0x0000041F	ERROR_SERVICE_DATABASE_LOCKED	服务数据库被锁定。
1056	0x00000420	ERROR_SERVICE_ALREADY_RUNNING	该服务的一个实例已经在运行。
1057	0x00000421	ERROR_INVALID_SERVICE_ACCOUNT	帐户名称无效或不存在，或者对于指定的帐户名称来说密码无效。
1058	0x00000422	ERROR_SERVICE_DISABLED	该服务不能被启动，要么是因为它被禁用，要么是因为它没有与之相关的启用设备。
1059	0x00000423	ERROR_CIRCULAR_DEPENDENCY	指定了循环服务依赖性。
1060	0x00000424	ERROR_SERVICE_DOES_NOT_EXIST	指定的服务不作为已安装的服务而存在。
1061	0x00000425	ERROR_SERVICE_CANNOT_ACCEPT_CTRL	该服务目前不能接受控制信息。
1062	0x00000426	ERROR_SERVICE_NOT_ACTIVE	该服务尚未启动。
1063	0x00000427	ERROR_FAILED_SERVICE_CONTROLLER_CONNECT	服务进程无法连接到服务控制器。
1064	0x00000428	ERROR_EXCEPTION_IN_SERVICE	在处理控制请求时，服务中发生了一个异常。
1065	0x00000429	ERROR_DATABASE_DOES_NOT_EXIST	指定的数据库不存在。
1066	0x0000042A	ERROR_SERVICE_SPECIFIC_ERROR	该服务返回了一个特定于服务的错误代码。
1067	0x0000042B	ERROR_PROCESS_ABORTED	该进程意外终止。
1068	0x0000042C	ERROR_SERVICE_DEPENDENCY_FAIL	依赖服务或组未能启动。
1069	0x0000042D	ERROR_SERVICE_LOGON_FAILED	由于登录失败，该服务没有启动。
1070	0x0000042E	ERROR_SERVICE_START_HANG	启动后，该服务挂在启动等待状态。
1071	0x0000042F	ERROR_INVALID_SERVICE_LOCK	指定的服务数据库锁无效。
1072	0x00000430	ERROR_SERVICE_MARKED_FOR_DELETE	指定的服务已被标记为删除。
1073	0x00000431	ERROR_SERVICE_EXISTS	指定的服务已经存在。
1074	0x00000432	ERROR_ALREADY_RUNNING_LKG	系统目前正以最后已知的良好配置运行。
1075	0x00000433	ERROR_SERVICE_DEPENDENCY_DELETED	依赖性服务不存在或已被标记为删除。
1076	0x00000434	ERROR_BOOT_ALREADY_ACCEPTED	已接受使用当前启动作为最后的有效控制设置。
1077	0x00000435	ERROR_SERVICE_NEVER_STARTED	从上次启动以来，未尝试过启动服务。
1078	0x00000436	ERROR_DUPLICATE_SERVICE_NAME	该名称已经作为服务名称或服务显示名称在使用。
1079	0x00000437	ERROR_DIFFERENT_SERVICE_ACCOUNT	为该服务指定的账户与为在同一进程中运行的其他服务指定的账户不同。
1080	0x00000438	ERROR_CANNOT_DETECT_DRIVER_FAILURE	只能为 Win32 服务设置失败操作，不能为驱动程序设置失败操作。

错误			描述
十进制	十六进制	名称	
1081	0x00000439	ERROR_CANNOT_DETECT_PROCESS_ABORT	该服务与服务控制管理器在同一进程中运行。因此，如果这个服务的进程意外终止，服务控制管理器不能采取行动。
1082	0x0000043A	ERROR_NO_RECOVERY_PROGRAM	没有为这项服务配置恢复程序。
1083	0x0000043B	ERROR_SERVICE_NOT_IN_EXE	该服务被配置为运行的可执行程序并没有实现该服务。
1084	0x0000043C	ERROR_NOT_SAFEBOOT_SERVICE	该服务不能在安全模式下启动。

错误			描述
十进制	十六进制	名称	
1100	0x0000044C	ERROR_END_OF_MEDIA	已经到达磁带的物理终点。
1101	0x0000044D	ERROR_FILEMARK_DETECTED	一个磁带访问达到一个文件标记。
1102	0x0000044E	ERROR_BEGINNING_OF_MEDIA	已达磁带或磁盘分区的开头。
1103	0x0000044F	ERROR_SETMARK_DETECTED	磁带访问已达一组文件的结尾。
1104	0x00000450	ERROR_NO_DATA_DETECTED	磁带上没有更多的数据了。
1105	0x00000451	ERROR_PARTITION_FAILURE	磁带无法分区。
1106	0x00000452	ERROR_INVALID_BLOCK_LENGTH	在访问多卷分区的新磁带时，当前的块大小不正确。
1107	0x00000453	ERROR_DEVICE_NOT_PARTITIONED	在加载磁带时无法找到磁带分区信息。
1108	0x00000454	ERROR_UNABLE_TO_LOCK_MEDIA	无法锁定介质弹出机制。
1109	0x00000455	ERROR_UNABLE_TO_UNLOAD_MEDIA	无法卸下媒体。
1110	0x00000456	ERROR_MEDIA_CHANGED	驱动器中的介质可能已经改变。
1111	0x00000457	ERROR_BUS_RESET	I/O 总线已重置。
1112	0x00000458	ERROR_NO_MEDIA_IN_DRIVE	驱动器中没有介质。
1113	0x00000459	ERROR_NO_UNICODE_TRANSLATION	目标多字节代码页中不存在 Unicode 字符的映射。
1114	0x0000045A	ERROR_DLL_INIT_FAILED	一个动态链接库（DLL）初始化程序失败。
1115	0x0000045B	ERROR_SHUTDOWN_IN_PROGRESS	系统关闭正在进行中。
1116	0x0000045C	ERROR_NO_SHUTDOWN_IN_PROGRESS	无法中止系统关机，因为没有关机正在进行中。
1117	0x0000045D	ERROR_IO_DEVICE	由于 I/O 设备错误，请求无法执行。
1118	0x0000045E	ERROR_SERIAL_NO_DEVICE	没有串行设备成功初始化。串行驱动器将卸载。
1119	0x0000045F	ERROR_IRQ_BUSY	无法打开正在其他设备共享中断请求（IRQ）的设备。至少一个使用该 IRQ 的其他设备已打开。
1120	0x00000460	ERROR_MORE_WRITES	一个串行 I/O 操作已被另一个串行端口的写入完成。（IOCTL_SERIAL_XOFF_COUNTER 已达零。）
1121	0x00000461	ERROR_COUNTER_TIMEOUT	串行I/O操作完成，因为已过超时时间。（IOCTL_SERIAL_XOFF_COUNTER 未达零。）
1122	0x00000462	ERROR_FLOPPY_ID_MARK_NOT_FOUND	在软盘上未发现 ID 地址标记。
1123	0x00000463	ERROR_FLOPPY_WRONG_CYLINDER	软盘扇区 ID 字段与软盘控制器磁道地址不相符。
1124	0x00000464	ERROR_FLOPPY_UNKNOWN_ERROR	软盘控制器报告了软盘驱动程序无法识别的错误。
1125	0x00000465	ERROR_FLOPPY_BAD_REGISTERS	软盘控制器在其寄存器中返回不一致的结果。
1126	0x00000466	ERROR_DISK_RECALIBRATE_FAILED	在访问硬盘时，重新校准操作失败，重试仍然失败。
1127	0x00000467	ERROR_DISK_OPERATION_FAILED	在访问硬盘时，磁盘操作失败，重试仍然失败。
1128	0x00000468	ERROR_DISK_RESET_FAILED	在访问硬盘时，即使失败，仍须复位磁盘控制器。
1129	0x00000469	ERROR_EOM_OVERFLOW	已达磁带的物理终点。
1130	0x0000046A	ERROR_NOT_ENOUGH_SERVER_MEMORY	服务器存储空间不足，无法处理此命令。
1131	0x0000046B	ERROR_POSSIBLE_DEADLOCK	检测到潜在的死锁状态。
1132	0x0000046C	ERROR_MAPPED_ALIGNMENT	指定的基址或文件偏移量未适当对齐。
1140	0x00000474	ERROR_SET_POWER_STATE_VETOED	改变系统电源状态的尝试被另一应用程序或驱动程序否决。
1141	0x00000475	ERROR_SET_POWER_STATE_FAILED	系统 BIOS 改变系统电源状态的尝试失败。
1142	0x00000476	ERROR_TOO_MANY_LINKS	试图在一个文件上创建超过文件系统支持的链接。
1150	0x0000047E	ERROR_OLD_WIN_VERSION	指定程序要求更新的 Windows 版本。
1151	0x0000047F	ERROR_APP_WRONG_OS	指定程序不是 Windows 或 MS-DOS 程序。
1152	0x00000480	ERROR_SINGLE_INSTANCE_APP	无法启动指定程序的一个以上的实例。

错误			描述
十进制	十六进制	名称	
1153	0x00000481	ERROR_RMODE_APP	指定程序为早期版本 Windows 编写。
1154	0x00000482	ERROR_INVALID_DLL	运行该应用程序所需的一个库文件被损坏。
1155	0x00000483	ERROR_NO_ASSOCIATION	没有应用程序与此操作的指定文件相关联。
1156	0x00000484	ERROR_DDE_FAIL	在向应用程序发送命令时发生错误。
1157	0x00000485	ERROR_DLL_NOT_FOUND	无法找到运行该应用程序所需的一个库文件。
1158	0x00000486	ERROR_NO_MORE_USER_HANDLES	当前进程已使用了窗口管理器对象的系统允许的所有句柄。
1159	0x00000487	ERROR_MESSAGE_SYNC_ONLY	该信息只能与同步操作一起使用。
1160	0x00000488	ERROR_SOURCE_ELEMENT_EMPTY	所指源元素没有媒体。
1161	0x00000489	ERROR_DESTINATION_ELEMENT_FULL	所指目标元素已包含媒体。
1162	0x0000048A	ERROR_ILLEGAL_ELEMENT_ADDRESS	所指元素不存在。
1163	0x0000048B	ERROR_MAGAZINE_NOT_PRESENT	所指元素是未显示的存储资源的一部分。
1164	0x0000048C	ERROR_DEVICE_REINITIALIZATION_NEEDED	由于硬件错误，所指设备需要重新初始化。
1165	0x0000048D	ERROR_DEVICE_REQUIRES_CLEANING	设备显示，在尝试进一步操作之前需要清除。
1166	0x0000048E	ERROR_DEVICE_DOOR_OPEN	设备显示门已打开。
1167	0x0000048F	ERROR_DEVICE_NOT_CONNECTED	设备没有连接。
1168	0x00000490	ERROR_NOT_FOUND	找不到元素。
1169	0x00000491	ERROR_NO_MATCH	在索引中没有匹配的指定键。
1170	0x00000492	ERROR_SET_NOT_FOUND	在对象上不存在指定的属性集。
1171	0x00000493	ERROR_POINT_NOT_FOUND	传给 GetMouseMovePointsEx 的点不在缓冲区内。
1172	0x00000494	ERROR_NO_TRACKING_SERVICE	追踪（工作站）服务没有运行。
1173	0x00000495	ERROR_NO_VOLUME_ID	无法找到卷的 ID。
1175	0x00000497	ERROR_UNABLE_TO_REMOVE_REPLACED	无法删除要替换的文件。
1176	0x00000498	ERROR_UNABLE_TO_MOVE_REPLACEMENT	无法将替换文件移到待替换的文件。待替换文件保留原来的名称。
1177	0x00000499	ERROR_UNABLE_TO_MOVE_REPLACEMENT_2	无法将替换文件移到待替换的文件。待替换文件已用备份名称重新命名。
1178	0x0000049A	ERROR_JOURNAL_DELETE_IN_PROGRESS	卷更改日志正在删除。
1179	0x0000049B	ERROR_JOURNAL_NOT_ACTIVE	卷更改日志不处于活动中。
1180	0x0000049C	ERROR_POTENTIAL_FILE_FOUND	找到了一个文件，但可能不是正确文件。
1181	0x0000049D	ERROR_JOURNAL_ENTRY_DELETED	日志项从日志中删除。

错误			描述
十进制	十六进制	名称	
1200	0x000004B0	ERROR_BAD_DEVICE	指定设备名称无效。
1201	0x000004B1	ERROR_CONNECTION_UNAVAIL	设备目前未连接，但其为一个记录连接。
1202	0x000004B2	ERROR_DEVICE_ALREADY_REMEMBERED	本地设备名称存在与另一网络资源的记录连接。
1203	0x000004B3	ERROR_NO_NET_OR_BAD_PATH	无网络供应商接受指定的网络路径。
1204	0x000004B4	ERROR_BAD_PROVIDER	指定的网络供应商名称无效。
1205	0x000004B5	ERROR_CANNOT_OPEN_PROFILE	无法打开网络连接配置文件。
1206	0x000004B6	ERROR_BAD_PROFILE	网络连接配置文件已损坏。
1207	0x000004B7	ERROR_NOT_CONTAINER	无法枚举空载体。
1208	0x000004B8	ERROR_EXTENDED_ERROR	发生扩展错误。
1209	0x000004B9	ERROR_INVALID_GROUPNAME	指定组名的格式无效。
1210	0x000004BA	ERROR_INVALID_COMPUTERNAME	指定计算机名称的格式无效。
1211	0x000004BB	ERROR_INVALID_EVENTNAME	指定事件名称的格式无效。
1212	0x000004BC	ERROR_INVALID_DOMAINNAME	指定域名的格式无效。
1213	0x000004BD	ERROR_INVALID_SERVICENAME	指定服务名称的格式无效。
1214	0x000004BE	ERROR_INVALID_NETNAME	指定网络名称的格式无效。
1215	0x000004BF	ERROR_INVALID_SHARENAME	指定共享名称的格式无效。
1216	0x000004C0	ERROR_INVALID_PASSWORDNAME	指定密码的格式无效。
1217	0x000004C1	ERROR_INVALID_MESSAGENAME	指定信息名称的格式无效。
1218	0x000004C2	ERROR_INVALID_MESSAGEDEST	指定信息目标的格式无效。
1219	0x000004C3	ERROR_SESSION_CREDENTIAL_CONFLICT	所提供的凭证与现有的凭证集相冲突。

错误			描述
十进制	十六进制	名称	
1220	0x000004C4	ERROR_REMOTE_SESSION_LIMIT_EXCEEDED	试图建立网络服务器的会话，但已对该服务器创建过多的会话。
1221	0x000004C5	ERROR_DUP_DOMAINNAME	工作组或域名已被网络上的另一台计算机使用。
1222	0x000004C6	ERROR_NO_NETWORK	网络不存在或未启动。
1223	0x000004C7	ERROR_CANCELLED	操作已被用户取消。
1224	0x000004C8	ERROR_USER_MAPPED_FILE	所请求操作无法在有用户映射部分打开的文件上执行。
1225	0x000004C9	ERROR_CONNECTION_REFUSED	远程系统拒绝网络连接。
1226	0x000004CA	ERROR_GRACEFUL_DISCONNECT	网络连接已经优雅地关闭。
1227	0x000004CB	ERROR_ADDRESS_ALREADY_ASSOCIATED	网络传输端点已有与之相关的地址。
1228	0x000004CC	ERROR_ADDRESS_NOT_ASSOCIATED	地址尚未与网络端点相关联。
1229	0x000004CD	ERROR_CONNECTION_INVALID	企图在不存在的网络连接上进行操作。
1230	0x000004CE	ERROR_CONNECTION_ACTIVE	在活动的网络连接上尝试了无效的操作。
1231	0x000004CF	ERROR_NETWORK_UNREACHABLE	无法访问网络位置。有关网络故障排除的信息，请参见 Windows 帮助。
1232	0x000004D0	ERROR_HOST_UNREACHABLE	无法访问网络位置。有关网络故障排除的信息，请参见 Windows 帮助。
1233	0x000004D1	ERROR_PROTOCOL_UNREACHABLE	无法访问网络位置。有关网络故障排除的信息，请参见 Windows 帮助。
1234	0x000004D2	ERROR_PORT_UNREACHABLE	在远程系统的目标网络端点没有服务在运行。
1235	0x000004D3	ERROR_REQUEST_ABORTED	请求已中止。
1236	0x000004D4	ERROR_CONNECTION_ABORTED	网络连接被本地系统中止。
1237	0x000004D5	ERROR_RETRY	操作无法完成。应该重试。
1238	0x000004D6	ERROR_CONNECTION_COUNT_LIMIT	无法与服务器建立连接，因为该账户的并发连接数已达到限制。
1239	0x000004D7	ERROR_LOGIN_TIME_RESTRICTION	试图在该账户未经授权的时间段内登录。
1240	0x000004D8	ERROR_LOGIN_WKSTA_RESTRICTION	该账户未被授权从该站登录。
1241	0x000004D9	ERROR_INCORRECT_ADDRESS	网络地址不能用于请求的操作。
1242	0x000004DA	ERROR_ALREADY_REGISTERED	该服务已注册。
1243	0x000004DB	ERROR_SERVICE_NOT_FOUND	指定的服务不存在。
1244	0x000004DC	ERROR_NOT_AUTHENTICATED	所请求操作未执行，因为用户没有被认证。
1245	0x000004DD	ERROR_NOT_LOGGED_ON	由于用户没有登录到网络上，所请求的操作没有被执行。指定的服务不存在。
1246	0x000004DE	ERROR_CONTINUE	继续进行中的工作。
1247	0x000004DF	ERROR_ALREADY_INITIALIZED	在初始化已经完成的情况下，试图执行初始化操作。
1248	0x000004E0	ERROR_NO_MORE_DEVICES	没有更多的本地设备。
1249	0x000004E1	ERROR_NO_SUCH_SITE	指定的站点不存在。
1250	0x000004E2	ERROR_DOMAIN_CONTROLLER_EXISTS	具有指定名称的域控制器已经存在。
1251	0x000004E3	ERROR_ONLY_IF_CONNECTED	仅在连接到服务器时才支持这一操作。
1252	0x000004E4	ERROR_OVERRIDE_NOCHANGES	即使没有更改，组策略框架也应该调用扩展。
1253	0x000004E5	ERROR_BAD_USER_PROFILE	指定用户没有一个有效的配置文件。
1254	0x000004E6	ERROR_NOT_SUPPORTED_ON_SBS	在微软小型企业服务器上不支持这种操作。
1255	0x000004E7	ERROR_SERVER_SHUTDOWN_IN_PROGRESS	服务器机器正在关闭。
1256	0x000004E8	ERROR_HOST_DOWN	远程系统无法使用。有关网络故障排除的信息，请参见 Windows 帮助。
1257	0x000004E9	ERROR_NON_ACCOUNT_SID	所提供的安全标识符不是来自一个账户域。
1258	0x000004EA	ERROR_NON_DOMAIN_SID	所提供的安全标识符没有域的成分。
1259	0x000004EB	ERROR_APPHELP_BLOCK	AppHelp 对话框被取消，从而导致应用程序无法启动。
1260	0x000004EC	ERROR_ACCESS_DISABLED_BY_POLICY	您的管理员已经禁止了对所请求资源的访问。
1261	0x000004ED	ERROR_REG_NAT_CONSUMPTION	程序试图使用无效的寄存器值。通常，由未初始化的寄存器引起。该错误为 Itanium 特有。
1262	0x000004EE	ERROR_CSCSHARE_OFFLINE	该共享目前处于脱机状态或不存在。
1300	0x00000514	ERROR_NOT_ALL_ASSIGNED	并非所有引用的权限都分配给了调用者。
1301	0x00000515	ERROR_SOME_NOT_MAPPED	帐户名称和安全 ID 之间的某些映射未完成。

错误			描述
十进制	十六进制	名称	
1302	0x00000516	ERROR_NO_QUOTAS_FOR_ACCOUNT	没有专门为这个账户设置系统配额限制。
1303	0x00000517	ERROR_LOCAL_USER_SESSION_KEY	没有可用的加密密钥。返回了一个已知加密密钥。
1304	0x00000518	ERROR_NULL_LM_PASSWORD	密码太复杂，无法转换为 LAN Manager 密码。返回的 LAN Manager 密码为空字符串。
1305	0x00000519	ERROR_UNKNOWN_REVISION	修订级别未知。
1306	0x0000051A	ERROR_REVISION_MISMATCH	表示两个修订级别不兼容。
1307	0x0000051B	ERROR_INVALID_OWNER	这个安全 ID 不能指定为这个对象的所有者。
1308	0x0000051C	ERROR_INVALID_PRIMARY_GROUP	该安全 ID 不能指派为对象的主要组。
1309	0x0000051D	ERROR_NO_IMPERSONATION_TOKEN	当前并未模拟客户的线程试图操作模拟令牌。
1310	0x0000051E	ERROR_CANT_DISABLE_MANDATORY	该组不能被禁用。
1311	0x0000051F	ERROR_NO_LOGON_SERVERS	目前没有可用的登录服务器为登录请求提供服务。
1312	0x00000520	ERROR_NO_SUCH_LOGON_SESSION	指定的登录会话不存在。可能已被终止。
1313	0x00000521	ERROR_NO_SUCH_PRIVILEGE	指定的权限不存在。
1314	0x00000522	ERROR_PRIVILEGE_NOT_HELD	所要求特权并未由客户持有。
1315	0x00000523	ERROR_INVALID_ACCOUNT_NAME	所提供名称并非正确的帐户名形式。
1316	0x00000524	ERROR_USER_EXISTS	指定的用户已存在。
1317	0x00000525	ERROR_NO_SUCH_USER	指定的用户不存在。
1318	0x00000526	ERROR_GROUP_EXISTS	指定的组已存在。
1319	0x00000527	ERROR_NO_SUCH_GROUP	指定的组不存在。
1320	0x00000528	ERROR_MEMBER_IN_GROUP	指定的用户账户已是指定组的成员，或是因为组包含成员所以无法删除指定的组。
1321	0x00000529	ERROR_MEMBER_NOT_IN_GROUP	指定的用户账户不是指定组账户的成员。
1322	0x0000052A	ERROR_LAST_ADMIN	无法禁用或删除最后剩下的管理账户。
1323	0x0000052B	ERROR_WRONG_PASSWORD	无法更新密码。提供的作为当前密码的值不正确。
1324	0x0000052C	ERROR_ILL_FORMED_PASSWORD	无法更新密码。为新密码提供的值包含密码中不允许的值。
1325	0x0000052D	ERROR_PASSWORD_RESTRICTION	无法更新密码。为新密码提供的值不符合域的长度、复杂性或历史要求。
1326	0x0000052E	ERROR_LOGON_FAILURE	登录失败：未知用户名或错误密码。
1327	0x0000052F	ERROR_ACCOUNT_RESTRICTION	登录失败：用户账户限制。
1328	0x00000530	ERROR_INVALID_LOGON_HOURS	登录失败：违反账户登录时间限制。
1329	0x00000531	ERROR_INVALID_WORKSTATION	登录失败：不允许用户登录此计算机。
1330	0x00000532	ERROR_PASSWORD_EXPIRED	登录失败：指定的账户密码已过期。
1331	0x00000533	ERROR_ACCOUNT_DISABLED	登录失败：账户当前已禁用。
1332	0x00000534	ERROR_NONE_MAPPED	帐户名与安全 ID 间无任何映射完成。
1333	0x00000535	ERROR_TOO_MANY_LUIDS_REQUESTED	一次性请求太多的本地用户标识符（LUID）。
1334	0x00000536	ERROR_LUIDS_EXHAUSTED	无更多可用的本地用户标识符（LUID）。
1335	0x00000537	ERROR_INVALID_SUB_AUTHORITY	对于该特定用途，安全 ID 的次级授权部分无效。
1336	0x00000538	ERROR_INVALID_ACL	访问控制列表（ACL）结构无效。
1337	0x00000539	ERROR_INVALID_SID	安全 ID 结构无效。
1338	0x0000053A	ERROR_INVALID_SECURITY_DESCR	安全描述符结构无效。
1340	0x0000053C	ERROR_BAD_INHERITANCE_ACL	无法创建固有的访问控制列表（ACL）或访问控制条目（ACE）。
1341	0x0000053D	ERROR_SERVER_DISABLED	该服务器目前已被禁用。
1342	0x0000053E	ERROR_SERVER_NOT_DISABLED	该服务器目前已启用。
1343	0x0000053F	ERROR_INVALID_ID_AUTHORITY	提供给标识符授权的值为无效值。
1344	0x00000540	ERROR_ALLOTTED_SPACE_EXCEEDED	没有更多的内存可用于安全信息更新。
1345	0x00000541	ERROR_INVALID_GROUP_ATTRIBUTES	指定属性无效，或与整组的属性不兼容。
1346	0x00000542	ERROR_BAD_IMPERSONATION_LEVEL	所需的模拟级别未提供，或所提供的模拟级别无效。
1347	0x00000543	ERROR_CANT_OPEN_ANONYMOUS	无法打开匿名级别的安全令牌。
1348	0x00000544	ERROR_BAD_VALIDATION_CLASS	请求的验证信息类别无效。
1349	0x00000545	ERROR_BAD_TOKEN_TYPE	令牌的类型不适合其尝试的用途。
1350	0x00000546	ERROR_NO_SECURITY_ON_OBJECT	无法在与安全性无关联的对象上运行安全性操作。
1351	0x00000547	ERROR_CANT_ACCESS_DOMAIN_INFO	无法从域控制器中读取配置信息，这是因为该机器不可用，或者访问被拒绝。

错误			描述
十进制	十六进制	名称	
1352	0x00000548	ERROR_INVALID_SERVER_STATE	安全账户管理器（SAM）或本地安全授权（LSA）服务器在执行安全操作时处于错误状态。
1353	0x00000549	ERROR_INVALID_DOMAIN_STATE	域处于错误的状态，无法执行安全操作。
1354	0x0000054A	ERROR_INVALID_DOMAIN_ROLE	这个操作只允许在域的主域控制器上进行。
1355	0x0000054B	ERROR_NO_SUCH_DOMAIN	指定的域不存在或无法联系。
1356	0x0000054C	ERROR_DOMAIN_EXISTS	指定的域已存在。
1357	0x0000054D	ERROR_DOMAIN_LIMIT_EXCEEDED	试图超过每台服务器的域名数量限制。
1358	0x0000054E	ERROR_INTERNAL_DB_CORRUPTION	由于灾难性介质故障或磁盘上的数据结构损坏，无法完成请求的操作。
1359	0x0000054F	ERROR_INTERNAL_ERROR	发生了内部错误。
1360	0x00000550	ERROR_GENERIC_NOT_MAPPED	通用访问类型包含在已映射到非通用类型的访问掩码中。
1361	0x00000551	ERROR_BAD_DESCRIPTOR_FORMAT	安全描述符格式不正确（绝对或自相关）。
1362	0x00000552	ERROR_NOT_LOGON_PROCESS	所请求的操作被限制为仅由登录进程使用。调用进程没有注册为登录进程。
1363	0x00000553	ERROR_LOGON_SESSION_EXISTS	无法使用已在使用中的 ID 启动新的登录会话。
1364	0x00000554	ERROR_NO_SUCH_PACKAGE	指定验证包未知。
1365	0x00000555	ERROR_BAD_LOGON_SESSION_STATE	登录会话的状态与请求的操作不一致。
1366	0x00000556	ERROR_LOGON_SESSION_COLLISION	登录会话 ID 已经在使用中。
1367	0x00000557	ERROR_INVALID_LOGON_TYPE	登录请求包含无效的登录类型值。
1368	0x00000558	ERROR_CANNOT_IMPERSONATE	在使用命名通道读取数据之前，无法经由该通道模拟。
1369	0x00000559	ERROR_RXACT_INVALID_STATE	注册表子树的交易状态与请求的操作不兼容。
1370	0x0000055A	ERROR_RXACT_COMMIT_FAILURE	出现内部安全数据库损坏。
1371	0x0000055B	ERROR_SPECIAL_ACCOUNT	无法在内置帐户上执行此操作。
1372	0x0000055C	ERROR_SPECIAL_GROUP	无法在内置特殊组上执行此操作。
1373	0x0000055D	ERROR_SPECIAL_USER	无法在内置特殊用户上执行此操作。
1374	0x0000055E	ERROR_MEMBERS_PRIMARY_GROUP	无法从组中删除用户，因为当前组为用户的主组。
1375	0x0000055F	ERROR_TOKEN_ALREADY_IN_USE	该令牌已经作为主令牌在使用。
1376	0x00000560	ERROR_NO_SUCH_ALIAS	指定的本地组不存在。
1377	0x00000561	ERROR_MEMBER_NOT_IN_ALIAS	指定的账户名不是本地组的成员。
1378	0x00000562	ERROR_MEMBER_IN_ALIAS	指定的账户名已经是本地组的成员。
1379	0x00000563	ERROR_ALIAS_EXISTS	指定的本地组已经存在。
1380	0x00000564	ERROR_LOGON_NOT_GRANTED	登录失败：未授予用户在此计算机上的请求登录类型。
1381	0x00000565	ERROR_TOO_MANY_SECRETS	单个系统中可存储的最大密钥数量已超出限制。
1382	0x00000566	ERROR_SECRET_TOO_LONG	秘密的长度超过了允许的最大长度。
1383	0x00000567	ERROR_INTERNAL_DB_ERROR	本地安全颁发机构数据库内部包含不一致性。
1384	0x00000568	ERROR_TOO_MANY_CONTEXT_IDS	在尝试登录的过程中，用户的安全环境积累了过多的安全 ID。
1385	0x00000569	ERROR_LOGON_TYPE_NOT_GRANTED	登录失败：未授予用户在此计算机上的请求登录类型。
1386	0x0000056A	ERROR_NT_CROSS_ENCRYPTION_REQUIRED	更改用户密码时需要交叉加密密码。
1387	0x0000056B	ERROR_NO_SUCH_MEMBER	成员不存在，无法将成员添加到本地组中，也无法从本地组将其删除。
1388	0x0000056C	ERROR_INVALID_MEMBER	无法将新成员加入到本地组中，因为成员的帐户类型错误。
1389	0x0000056D	ERROR_TOO_MANY_SIDS	已经指定了太多的安全 ID。
1390	0x0000056E	ERROR_LM_CROSS_ENCRYPTION_REQUIRED	更改此用户密码时需要交叉加密密码。
1391	0x0000056F	ERROR_NO_INHERITANCE	表明 ACL 未包含可继承的组件。
1392	0x00000570	error_file_corrupt	文件或目录已损坏，无法读取。
1393	0x00000571	ERROR_DISK_CORRUPT	磁盘结构损坏，无法读取。
1394	0x00000572	ERROR_NO_USER_SESSION_KEY	无指定登录会话的用户会话密钥。
1395	0x00000573	ERROR_LICENSE_QUOTA_EXCEEDED	正在访问的服务许可用于特定数量的连接。此时不能再与服务进行连接，因为已经达到可接受的连接数目。
1396	0x00000574	ERROR_WRONG_TARGET_NAME	登录失败：目标账户名称不正确。

错误			描述
十进制	十六进制	名称	
1397	0x00000575	ERROR_MUTUAL_AUTH_FAILED	相互身份验证失败。该服务器在域控制器的密码过期。
1398	0x00000576	ERROR_TIME_SKEW	客户端和服务器之间存在时间差。
1399	0x00000577	ERROR_CURRENT_DOMAIN_NOT_ALLOWED	该操作不能在当前域上执行。

错误			描述
十进制	十六进制	名称	
1400	0x00000578	ERROR_INVALID_WINDOW_HANDLE	无效的窗口句柄。
1401	0x00000579	ERROR_INVALID_MENU_HANDLE	无效的菜单句柄。
1402	0x0000057A	ERROR_INVALID_CURSOR_HANDLE	无效的光标句柄。
1403	0x0000057B	ERROR_INVALID_ACCEL_HANDLE	无效的加速表句柄。
1404	0x0000057C	ERROR_INVALID_HOOK_HANDLE	无效的挂钩句柄。
1405	0x0000057D	ERROR_INVALID_DWP_HANDLE	无效的多重窗口位置结构句柄。
1406	0x0000057E	ERROR_TLW_WITH_WSCHILD	无法创建顶层子窗口。
1407	0x0000057F	ERROR_CANNOT_FIND_WND_CLASS	无法找到窗口类。
1408	0x00000580	ERROR_WINDOW_OF_OTHER_THREAD	无效窗口；它属于其他线程。
1409	0x00000581	ERROR_HOTKEY_ALREADY_REGISTERED	热键已注册。
1410	0x00000582	ERROR_CLASS_ALREADY_EXISTS	类别已存在。
1411	0x00000583	ERROR_CLASS_DOES_NOT_EXIST	类别不存在。
1412	0x00000584	ERROR_CLASS_HAS_WINDOWS	类别仍有打开的窗口。
1413	0x00000585	ERROR_INVALID_INDEX	无效索引。
1414	0x00000586	ERROR_INVALID_ICON_HANDLE	无效的图标句柄。
1415	0x00000587	ERROR_PRIVATE_DIALOG_INDEX	使用专用 DIALOG 窗口字词。
1416	0x00000588	ERROR_LISTBOX_ID_NOT_FOUND	未找到列表框的标识符。
1417	0x00000589	ERROR_NO_WILDCARD_CHARACTERS	找不到通配符。
1418	0x0000058A	ERROR_CLIPBOARD_NOT_OPEN	线程没有打开剪贴板。
1419	0x0000058B	ERROR_HOTKEY_NOT_REGISTERED	热键未注册。
1420	0x0000058C	ERROR_WINDOW_NOT_DIALOG	窗口不是有效的对话窗口。
1421	0x0000058D	ERROR_CONTROL_ID_NOT_FOUND	未找到控制 ID。
1422	0x0000058E	ERROR_INVALID_COMBOBOX_MESSAGE	因为没有编辑控件，所以组合框的消息无效。
1423	0x0000058F	ERROR_WINDOW_NOT_COMBOBOX	窗口不是组合框。
1424	0x00000590	ERROR_INVALID_EDIT_HEIGHT	高度必须小于 256。
1425	0x00000591	ERROR_DC_NOT_FOUND	无效的设备上下文（DC）句柄。
1426	0x00000592	ERROR_INVALID_HOOK_FILTER	无效的钩子程序类型。
1427	0x00000593	ERROR_INVALID_FILTER_PROC	无效的钩子程序。
1428	0x00000594	ERROR_HOOK_NEEDS_HMOD	在没有模块句柄的情况下无法设置非本地钩子。
1429	0x00000595	ERROR_GLOBAL_ONLY_HOOK	该钩子程序仅可全局设置。
1430	0x00000596	ERROR_JOURNAL_HOOK_SET	日志挂钩子程序已安装完毕。
1431	0x00000597	ERROR_HOOK_NOT_INSTALLED	未安装挂钩子程序。
1432	0x00000598	ERROR_INVALID_LB_MESSAGE	单选列表框的信息无效。
1433	0x00000599	ERROR_SETCOUNT_ON_BAD_LB	LB_SETCOUNT 发送到非被动的列表框。
1434	0x0000059A	ERROR_LB_WITHOUT_TABSTOPS	该列表框不支持制表符。
1435	0x0000059B	ERROR_DESTROY_OBJECT_OF_OTHER_THREAD	无法销毁另一线程创建的对象。
1436	0x0000059C	ERROR_CHILD_WINDOW_MENU	子窗口没有菜单。
1437	0x0000059D	ERROR_NO_SYSTEM_MENU	窗口没有系统菜单。
1438	0x0000059E	ERROR_INVALID_MSGBOX_STYLE	无效的消息框样式。
1439	0x0000059F	ERROR_INVALID_SPI_VALUE	无效的系统范围（SPI_*）参数。
1440	0x000005A0	ERROR_SCREEN_ALREADY_LOCKED	屏幕已锁定。
1441	0x000005A1	ERROR_HWNDS_HAVE_DIFF_PARENT	在多窗口位置结构中，所有窗口句柄必须具有相同的上层。
1442	0x000005A2	ERROR_NOT_CHILD_WINDOW	窗口不是子窗口。
1443	0x000005A3	ERROR_INVALID_GW_COMMAND	无效的 GW_* 命令。
1444	0x000005A4	ERROR_INVALID_THREAD_ID	无效的线程标识符。
1445	0x000005A5	ERROR_NON_MDICHILD_WINDOW	无法处理非多文档界面（MDI）窗口的信息。
1446	0x000005A6	ERROR_POPUP_ALREADY_ACTIVE	弹出式菜单已激活。

错误			描述
十进制	十六进制	名称	
1447	0x000005A7	ERROR_NO_SCROLLBARS	窗口没有滚动条。
1448	0x000005A8	ERROR_INVALID_SCROLLBAR_RANGE	滚动条范围不能大于 MAXLONG。
1449	0x000005A9	ERROR_INVALID_SHOWWIN_COMMAND	无法以指定的方式显示或删除窗口。
1450	0x000005AA	ERROR_NO_SYSTEM_RESOURCES	系统资源不足，无法完成请求的服务。
1451	0x000005AB	ERROR_NONPAGED_SYSTEM_RESOURCES	系统资源不足，无法完成请求的服务。
1452	0x000005AC	ERROR_PAGED_SYSTEM_RESOURCES	系统资源不足，无法完成请求的服务。
1453	0x000005AD	ERROR_WORKING_SET_QUOTA	配额不足，无法完成请求的服务。
1454	0x000005AE	ERROR_PAGEFILE_QUOTA	配额不足，无法完成请求的服务。
1455	0x000005AF	ERROR_COMMITMENT_LIMIT	页面文件太小，无法完成操作。
1456	0x000005B0	ERROR_MENU_ITEM_NOT_FOUND	未找到菜单项。
1457	0x000005B1	ERROR_INVALID_KEYBOARD_HANDLE	无效的键盘布局句柄。
1458	0x000005B2	ERROR_HOOK_TYPE_NOT_ALLOWED	钩子类型不允许。
1459	0x000005B3	ERROR_REQUIRES_INTERACTIVE_WINDOWSTATION	该操作需要交互式窗口工作站。
1460	0x000005B4	ERROR_TIMEOUT	该操作返回，因为超时时间已过。
1461	0x000005B5	ERROR_INVALID_MONITOR_HANDLE	无效的监视器句柄。
1500	0x000005DC	ERROR_EVENTLOG_FILE_CORRUPT	事件日志文件已损坏。
1501	0x000005DD	ERROR_EVENTLOG_CANT_START	无法打开事件日志文件，所以事件日志服务没有启动。
1502	0x000005DE	ERROR_LOG_FILE_FULL	事件日志文件已满。
1503	0x000005DF	ERROR_EVENTLOG_FILE_CHANGED	事件日志文件在读取操作之间发生了变化。

错误			描述
十进制	十六进制	名称	
1601	0x00000641	ERROR_INSTALL_SERVICE_FAILURE	无法访问 Windows Installer 服务。请联系技术人员，验证 Windows Installer 服务是否正确注册。
1602	0x00000642	ERROR_INSTALL_USEREXIT	用户取消了安装。
1603	0x00000643	ERROR_INSTALL_FAILURE	安装时出现严重错误。
1604	0x00000644	ERROR_INSTALL_SUSPEND	安装暂停，未完成。
1605	0x00000645	ERROR_UNKNOWN_PRODUCT	该操作仅对当前安装的产品有效。
1606	0x00000646	ERROR_UNKNOWN_FEATURE	特征 ID 未注册。
1607	0x00000647	ERROR_UNKNOWN_COMPONENT	组件 ID 未注册。
1608	0x00000648	ERROR_UNKNOWN_PROPERTY	未知属性。
1609	0x00000649	ERROR_INVALID_HANDLE_STATE	句柄处于不正确的状态。
1610	0x0000064A	ERROR_BAD_CONFIGURATION	该产品的配置数据已损坏。请联系技术人员。
1611	0x0000064B	ERROR_INDEX_ABSENT	组件限定符不存在。
1612	0x0000064C	ERROR_INSTALL_SOURCE_ABSENT	该产品的安装来源不可用。请验证来源是否存在以及是否可以访问。
1613	0x0000064D	ERROR_INSTALL_PACKAGE_VERSION	该安装包无法由 Windows Installer 服务安装。必须安装包含较新版本 Windows Installer 服务的 Windows 服务包。
1614	0x0000064E	ERROR_PRODUCT_UNINSTALLED	产品已卸载。
1615	0x0000064F	ERROR_BAD_QUERY_SYNTAX	SQL 查询语法无效或不支持。
1616	0x00000650	ERROR_INVALID_FIELD	记录字段不存在。
1617	0x00000651	ERROR_DEVICE_REMOVED	设备已删除。
1618	0x00000652	ERROR_INSTALL_ALREADY_RUNNING	另一项安装工作已在进行中。在进行本次安装之前，请完成该安装。
1619	0x00000653	ERROR_INSTALL_PACKAGE_OPEN_FAILED	该安装包无法打开。验证程序包是否存在以及是否可以访问，或者联系应用程序供应商以验证是否有效 Windows Installer 程序包。
1620	0x00000654	ERROR_INSTALL_PACKAGE_INVALID	该安装包无法打开。联系应用程序供应商以验证是否有效的 Windows Installer 程序包。
1621	0x00000655	ERROR_INSTALL_UI_FAILURE	启动 Windows Installer 服务用户界面时出现了错误。请联系技术人员。
1622	0x00000656	ERROR_INSTALL_LOG_FAILURE	打开安装日志文件出错。验证指定的日志文件位置是否存在以及是否可以写入。
1623	0x00000657	ERROR_INSTALL_LANGUAGE_UNSUPPORTED	系统不支持该安装包的语言。

错误			描述
十进制	十六进制	名称	
1624	0x00000658	ERROR_INSTALL_TRANSFORM_FAILURE	应用变换时出现错误。验证指定的变换路径是否有效。
1625	0x00000659	ERROR_INSTALL_PACKAGE_REJECTED	系统政策禁止这种安装。联系系统管理员。
1626	0x0000065A	ERROR_FUNCTION_NOT_CALLED	无法执行函数。
1627	0x0000065B	ERROR_FUNCTION_FAILED	函数在执行过程中失败。
1628	0x0000065C	ERROR_INVALID_TABLE	指定了无效或未知的表。
1629	0x0000065D	ERROR_DATATYPE_MISMATCH	提供的数据类型错误。
1630	0x0000065E	ERROR_UNSUPPORTED_TYPE	不支持这种类型的数据。
1631	0x0000065F	ERROR_CREATE_FAILED	Windows Installer 服务未能启动。请联系支持人员。
1632	0x00000660	ERROR_INSTALL_TEMP_UNWRITABLE	临时文件夹已满或无法访问。验证临时文件夹是否存在以及是否可以写入。
1633	0x00000661	ERROR_INSTALL_PLATFORM_UNSUPPORTED	此处理器类型不支持此安装包。请联系产品供应商。
1634	0x00000662	ERROR_INSTALL_NOTUSED	组件未在此电脑上使用。
1635	0x00000663	ERROR_PATCH_PACKAGE_OPEN_FAILED	补丁包无法打开。验证补丁包是否存在以及是否可以访问，或者联系应用程序供应商以验证是否有效的 Windows Installer 补丁包。
1636	0x00000664	ERROR_PATCH_PACKAGE_INVALID	补丁包无法打开。联系应用程序供应商以验证是否有效的 Windows Installer 补丁包。
1637	0x00000665	ERROR_PATCH_PACKAGE_UNSUPPORTED.	该补丁包无法通过 Windows Installer 服务处理。必须安装包含较新版本 Windows Installer 服务的 Windows 服务包。
1638	0x00000666	ERROR_PRODUCT_VERSION	已安装该产品的另一版本。该版本的安装无法继续。如需配置或删除本产品的现有版本，请使用控制面板上的添加/删除程序。
1639	0x00000667	ERROR_INVALID_COMMAND_LINE	无效的命令行参数。有关详细的命令行帮助，请查阅 Windows Installer SDK。
1640	0x00000668	ERROR_INSTALL_REMOTE_DISALLOWED	在终端服务远程会话中，仅管理员有权添加、删除或配置服务器软件。如果想在服务器上安装或配置软件，请联系网络管理员。
1641	0x00000669	ERROR_SUCCESS_REBOOT_INITIATED	请求的操作已成功完成。系统将重新启动，以便更改能够生效。
1642	0x0000066A	ERROR_PATCH_TARGET_NOT_FOUND	升级补丁无法通过 Windows Installer 服务安装，因为待升级的程序可能丢失，或者升级补丁可能更新了程序的不同版本。验证待升级的程序在你的电脑上是否存在，并且你有正确的升级补丁。
1643	0x0000066B	ERROR_PATCH_PACKAGE_REJECTED	系统政策不允许使用该补丁包。它没有用适当的证书签名。
1644	0x0000066C	ERROR_INSTALL_TRANSFORM_REJECTED	系统政策不允许进行一项或多项定制。它们没有用适当的证书签名。
1700	0x000006A4	RPC_S_INVALID_STRING_BINDING	字符串绑定无效。
1701	0x000006A5	RPC_S_WRONG_KIND_OF_BINDING	绑定句柄类型不正确。
1702	0x000006A6	RPC_S_INVALID_BINDING	绑定句柄无效。
1703	0x000006A7	RPC_S_PROTSEQ_NOT_SUPPORTED	不支持 RPC 协议序列。
1704	0x000006A8	RPC_S_INVALID_RPC_PROTSEQ	RPC 协议序列无效。
1705	0x000006A9	RPC_S_INVALID_STRING_UUID	字符串通用唯一标识符（UUID）无效。
1706	0x000006AA	RPC_S_INVALID_ENDPOINT_FORMAT	端点格式无效。
1707	0x000006AB	RPC_S_INVALID_NET_ADDR	网络地址无效。
1708	0x000006AC	RPC_S_NO_ENDPOINT_FOUND	未找到端点。
1709	0x000006AD	RPC_S_INVALID_TIMEOUT	超时值无效。
1710	0x000006AE	RPC_S_OBJECT_NOT_FOUND	未找到对象的通用唯一标识符（UUID）。
1711	0x000006AF	RPC_S_ALREADY_REGISTERED	对象的通用唯一标识符（UUID）已注册。
1712	0x000006B0	RPC_S_TYPE_ALREADY_REGISTERED	类型的通用唯一标识符（UUID）已注册。
1713	0x000006B1	RPC_S_ALREADY_LISTENING	RPC 服务器已在监听。
1714	0x000006B2	RPC_S_NO_PROTSEQS_REGISTERED	未登记协议序列。
1715	0x000006B3	RPC_S_NOT_LISTENING	RPC 服务器未在监听。
1716	0x000006B4	RPC_S_UNKNOWN_MGR_TYPE	管理器类型未知。
1717	0x000006B5	RPC_S_UNKNOWN_IF	界面未知。

错误			描述
十进制	十六进制	名称	
1718	0x000006B6	RPC_S_NO_BINDINGS	没有绑定。
1719	0x000006B7	RPC_S_NO_PROTSEQS	不存在协议序列。
1720	0x000006B8	RPC_S_CANT_CREATE_ENDPOINT	无法创建端点。
1721	0x000006B9	RPC_S_OUT_OF_RESOURCES	资源不足，无法完成此操作。
1722	0x000006BA	RPC_S_SERVER_UNAVAILABLE	RPC 服务器不可用。
1723	0x000006BB	RPC_S_SERVER_TOO_BUSY	RPC 服务器过忙，无法完成此操作。
1724	0x000006BC	RPC_S_INVALID_NETWORK_OPTIONS	网络选项无效。
1725	0x000006BD	RPC_S_NO_CALL_ACTIVE	这个线程上没有活动的远程过程调用。
1726	0x000006BE	RPC_S_CALL_FAILED	远程过程调用失败。
1727	0x000006BF	RPC_S_CALL_FAILED_DNE	远程过程调用失败且未运行。
1728	0x000006C0	RPC_S_PROTOCOL_ERROR	远程过程调用（RPC）协议出错。
1730	0x000006C2	RPC_S_UNSUPPORTED_TRANS_SYN	RPC 服务器不支持该传输语法。
1732	0x000006C4	RPC_S_UNSUPPORTED_TYPE	不支持通用唯一标识符（UUID）类型。
1733	0x000006C5	RPC_S_INVALID_TAG	标记无效。
1734	0x000006C6	RPC_S_INVALID_BOUND	数组边界无效。
1735	0x000006C7	RPC_S_NO_ENTRY_NAME	绑定不包含条目名称。
1736	0x000006C8	RPC_S_INVALID_NAME_SYNTAX	名称语法无效。
1737	0x000006C9	RPC_S_UNSUPPORTED_NAME_SYNTAX	不支持名称语法。
1739	0x000006CB	RPC_S_UUID_NO_ADDRESS	没有网络地址可用于构建通用唯一标识符（UUID）。
1740	0x000006CC	RPC_S_DUPLICATE_ENDPOINT	该端点是备份。
1741	0x000006CD	RPC_S_UNKNOWN_AUTHN_TYPE	认证类型未知。
1742	0x000006CE	RPC_S_MAX_CALLS_TOO_SMALL	最大调用次数太少。
1743	0x000006CF	RPC_S_STRING_TOO_LONG	字符串太长。
1744	0x000006D0	RPC_S_PROTSEQ_NOT_FOUND	未找到 RPC 协议序列。
1745	0x000006D1	RPC_S_PROCNUM_OUT_OF_RANGE	程序编号超出了范围。
1746	0x000006D2	RPC_S_BINDING_HAS_NO_AUTH	绑定不包含任何认证信息。
1747	0x000006D3	RPC_S_UNKNOWN_AUTHN_SERVICE	认证服务未知。
1748	0x000006D4	RPC_S_UNKNOWN_AUTHN_LEVEL	认证级别未知。
1749	0x000006D5	RPC_S_INVALID_AUTH_IDENTITY	安全上下文无效。
1750	0x000006D6	RPC_S_UNKNOWN_AUTHZ_SERVICE	授权服务未知。
1751	0x000006D7	EPT_S_INVALID_ENTRY	条目无效。
1752	0x000006D8	EPT_S_CANT_PERFORM_OP	服务器端点无法执行该操作。
1753	0x000006D9	EPT_S_NOT_REGISTERED	端点映射器中无更多的可用端点。
1754	0x000006DA	RPC_S_NOTHING_TO_EXPORT	未导出界面。
1755	0x000006DB	RPC_S_INCOMPLETE_NAME	该条目名称不完整。
1756	0x000006DC	RPC_S_INVALID_VERS_OPTION	版本选项无效。
1757	0x000006DD	RPC_S_NO_MORE_MEMBERS	不存在更多成员。
1758	0x000006DE	RPC_S_NOT_ALL_OBJS_UNEXPORTED	没有内容未导出。
1759	0x000006DF	RPC_S_INTERFACE_NOT_FOUND	接口未找到。
1760	0x000006E0	RPC_S_ENTRY_ALREADY_EXISTS	条目已存在。
1761	0x000006E1	RPC_S_ENTRY_NOT_FOUND	未找到条目。
1762	0x000006E2	RPC_S_NAME_SERVICE_UNAVAILABLE	名称服务不可用。
1763	0x000006E3	RPC_S_INVALID_NAF_ID	网络地址族无效。
1764	0x000006E4	RPC_S_CANNOT_SUPPORT	不支持请求的操作。
1765	0x000006E5	RPC_S_NO_CONTEXT_AVAILABLE	无可用的安全上下文以允许模拟。
1766	0x000006E6	RPC_S_INTERNAL_ERROR	在一个远程过程调用（RPC）中发生了内部错误。
1767	0x000006E7	RPC_S_ZERO_DIVIDE	RPC 服务器试图以零除整数。
1768	0x000006E8	RPC_S_ADDRESS_ERROR	RPC 服务器中发生了寻址错误。
1769	0x000006E9	RPC_S_FP_DIV_ZERO	RPC 服务器上的浮点操作导致以零做除数。
1770	0x000006EA	RPC_S_FP_UNDERFLOW	在 RPC 服务器上发生了浮点下溢。
1771	0x000006EB	RPC_S_FP_OVERFLOW	在 RPC 服务器上发生了浮点溢出。
1772	0x000006EC	RPC_X_NO_MORE_ENTRIES	自动句柄绑定的可用 RPC 服务器列表已用完。
1773	0x000006ED	RPC_X_SS_CHAR_TRANS_OPEN_FAIL	无法打开字符翻译表文件。

错误			描述
十进制	十六进制	名称	
1774	0x000006EE	RPC_X_SS_CHAR_TRANS_SHORT_FILE	包含字符翻译表的文件少于 512 字节。
1775	0x000006EF	RPC_X_SS_IN_NULL_CONTEXT	在远程过程调用中，将空的上下文句柄从客户端传给了主机。
1777	0x000006F1	RPC_X_SS_CONTEXT_DAMAGED	在远程过程调用中，上下文句柄发生了变化。
1778	0x000006F2	RPC_X_SS_HANDLES_MISMATCH	传递给远程过程调用的绑定句柄不匹配。
1779	0x000006F3	RPC_X_SS_CANNOT_GET_CALL_HANDLE	承接体无法获得远程过程调用句柄。
1780	0x000006F4	RPC_X_NULL_REF_POINTER	一个空的引用指针被传递给了承接体。
1781	0x000006F5	RPC_X_ENUM_VALUE_OUT_OF_RANGE	枚举值超出了范围。
1782	0x000006F6	RPC_X_BYTE_COUNT_TOO_SMALL	字节数太少。
1783	0x000006F7	RPC_X_BAD_STUB_DATA	存根接收到了错误的数据。
1784	0x000006F8	ERROR_INVALID_USER_BUFFER	提供的用户缓冲区对请求的操作无效。
1785	0x000006F9	ERROR_UNRECOGNIZED_MEDIA	磁盘媒体未被识别。可能未被格式化。
1786	0x000006FA	ERROR_NO_TRUST_LSA_SECRET	工作站没有信任密钥。
1787	0x000006FB	ERROR_NO_TRUST_SAM_ACCOUNT	服务器上的安全数据库没有这个工作站信任关系的计算机账户。
1788	0x000006FC	ERROR_TRUSTED_DOMAIN_FAILURE	主域和受信域之间的信任关系失败。
1789	0x000006FD	ERROR_TRUSTED_RELATIONSHIP_FAILURE	该工作站与主域之间的信任关系失败。
1790	0x000006FE	ERROR_TRUST_FAILURE	网络登录失败。
1791	0x000006FF	RPC_S_CALL_IN_PROGRESS	该线程的远程过程调用已经在进行中。
1792	0x00000700	ERROR_NETLOGON_NOT_STARTED	试图登录，但网络登录服务没有启动。
1793	0x00000701	ERROR_ACCOUNT_EXPIRED	用户账户已过期。
1794	0x00000702	ERROR_REDIRECTOR_HAS_OPEN_HANDLES	转发程序正在使用中，无法卸载。
1795	0x00000703	ERROR_PRINTER_DRIVER_ALREADY_INSTALLED	指定的打印机驱动程序已安装。
1796	0x00000704	ERROR_UNKNOWN_PORT	指定端口未知。
1797	0x00000705	ERROR_UNKNOWN_PRINTER_DRIVER	打印机驱动程序未知。
1798	0x00000706	ERROR_UNKNOWN_PRINTPROCESSOR	打印处理器未知。
1799	0x00000707	ERROR_INVALID_SEPARATOR_FILE	指定的分隔页文件无效。

错误			描述
十进制	十六进制	名称	
1800	0x00000708	ERROR_INVALID_PRIORITY	指定优先级无效。
1801	0x00000709	ERROR_INVALID_PRINTER_NAME	打印机名称无效。
1802	0x0000070A	ERROR_PRINTER_ALREADY_EXISTS	打印机已存在。
1803	0x0000070B	ERROR_INVALID_PRINTER_COMMAND	打印机命令无效。
1804	0x0000070C	ERROR_INVALID_DATATYPE	指定的数据类型无效。
1805	0x0000070D	ERROR_INVALID_ENVIRONMENT	指定环境无效。
1806	0x0000070E	RPC_S_NO_MORE_BINDINGS	没有更多的绑定。
1807	0x0000070F	ERROR_NOLOGON_INTERDOMAIN_TRUST_ACCOUNT	所用帐户为域间信任帐户。请使用你的全局用户账户或本地用户账户访问该服务器。
1808	0x00000710	ERROR_NOLOGON_WORKSTATION_TRUST_ACCOUNT	所用帐户为计算机账户。请使用你的全局用户账户或本地用户账户访问该服务器。
1809	0x00000711	ERROR_NOLOGON_SERVER_TRUST_ACCOUNT	所用账户为服务器信任账户。请使用你的全局用户账户或本地用户账户访问该服务器。
1810	0x00000712	ERROR_DOMAIN_TRUST_INCONSISTENT	指定域的名称或安全标识（SID）与该域的信任信息不一致。
1811	0x00000713	ERROR_SERVER_HAS_OPEN_HANDLES	服务器在使用中且无法卸载。
1812	0x00000714	ERROR_RESOURCE_DATA_NOT_FOUND	指定镜像文件不包含资源区域。
1813	0x00000715	ERROR_RESOURCE_TYPE_NOT_FOUND	在镜像文件中，找不到指定的资源类型。
1814	0x00000716	ERROR_RESOURCE_NAME_NOT_FOUND	在镜像文件中找不到指定的资源名称。
1815	0x00000717	ERROR_RESOURCE_LANG_NOT_FOUND	在镜像文件中找不到指定的资源语言 ID。
1816	0x00000718	ERROR_NOT_ENOUGH_QUOTA	配额不足，无法处理此命令。
1817	0x00000719	RPC_S_NO_INTERFACES	尚未注册界面。
1818	0x0000071A	RPC_S_CALL_CANCELLED	远程过程调用已取消。
1819	0x0000071B	RPC_S_BINDING_INCOMPLETE	绑定句柄不包含所有需要的信息。
1820	0x0000071C	RPC_S_COMM_FAILURE	在一个远程过程调用中发生了通信故障。
1821	0x0000071D	RPC_S_UNSUPPORTED_AUTHN_LEVEL	不支持请求的认证级别。

错误			描述
十进制	十六进制	名称	
1822	0x0000071E	RPC_S_NO_PRINC_NAME	未登记主要名称。
1823	0x0000071F	RPC_S_NOT_RPC_ERROR	指定错误不是有效的 Windows RPC 错误代码。
1824	0x00000720	RPC_S_UUID_LOCAL_ONLY	已经分配了一个只在此计算机上有效的 UUID。
1825	0x00000721	RPC_S_SEC_PKG_ERROR	发生了一个安全包的特定错误。
1826	0x00000722	RPC_S_NOT_CANCELLED	线程未取消。
1827	0x00000723	RPC_X_INVALID_ES_ACTION	对编码/解码句柄的操作无效。
1828	0x00000724	RPC_X_WRONG_ES_VERSION	序列化包的版本不兼容。
1829	0x00000725	RPC_X_WRONG_STUB_VERSION	RPC 承接体的版本不兼容。
1830	0x00000726	RPC_X_INVALID_PIPE_OBJECT	RPC 管道对象无效或已破坏。
1831	0x00000727	RPC_X_WRONG_PIPE_ORDER	试图在 RPC 管道对象上进行无效操作。
1832	0x00000728	RPC_X_WRONG_PIPE_VERSION	不支持的 RPC 管道版本。
1898	0x0000076A	RPC_S_GROUP_MEMBER_NOT_FOUND	未找到组成员。
1899	0x0000076B	EPT_S_CANT_CREATE	无法创建端点映射器数据库条目。
1900	0x0000076C	RPC_S_INVALID_OBJECT	对象通用唯一标识符（UUID）为空的 UUID。
1901	0x0000076D	ERROR_INVALID_TIME	指定时间无效。
1902	0x0000076E	ERROR_INVALID_FORM_NAME	指定的表格名称无效。
1903	0x0000076F	ERROR_INVALID_FORM_SIZE	指定的表格大小无效。
1904	0x00000770	ERROR_ALREADY_WAITING	指定的打印机句柄已在等待
1905	0x00000771	ERROR_PRINTER_DELETED	指定的打印机已删除。
1906	0x00000772	ERROR_INVALID_PRINTER_STATE	打印机的状态无效。
1907	0x00000773	ERROR_PASSWORD_MUST_CHANGE	用户的密码必须在第一次登录前进行修改。
1908	0x00000774	ERROR_DOMAIN_CONTROLLER_NOT_FOUND	无法找到此域的域控制器。
1909	0x00000775	ERROR_ACCOUNT_LOCKED_OUT	所引用账户目前已锁定，且可能无法登录。
1910	0x00000776	OR_INVALID_OXID	未找到指定的对象导出器。
1911	0x00000777	OR_INVALID_OID	未找到指定的对象。
1912	0x00000778	OR_INVALID_SET	未找到指定的对象解析器集。
1913	0x00000779	RPC_S_SEND_INCOMPLETE	在请求缓冲区中还有一些数据需要发送。
1914	0x0000077A	RPC_S_INVALID_ASYNC_HANDLE	无效的异步远程过程调用句柄。
1915	0x0000077B	RPC_S_INVALID_ASYNC_CALL	该操作的异步 RPC 调用句柄无效。
1916	0x0000077C	RPC_X_PIPE_CLOSED	RPC 管道对象已关闭。
1917	0x0000077D	RPC_X_PIPE_DISCIPLINE_ERROR	RPC 调用在所有管道被处理之前完成。
1918	0x0000077E	RPC_X_PIPE_EMPTY	RPC 管道中没有更多的数据。
1919	0x0000077F	ERROR_NO_SITENAME	该机器没有可用的站点名称。
1920	0x00000780	ERROR_CANT_ACCESS_FILE	系统无法访问此文件。
1921	0x00000781	ERROR_CANT_RESOLVE_FILENAME	文件名称不能被系统解析。
1922	0x00000782	RPC_S_ENTRY_TYPE_MISMATCH	条目不属于预期的类型。
1923	0x00000783	RPC_S_NOT_ALL_OBJS_EXPORTED	无法将所有对象的 UUID 导出到指定的条目。
1924	0x00000784	RPC_S_INTERFACE_NOT_EXPORTED	无法将接口导出到指定的条目。
1925	0x00000785	RPC_S_PROFILE_NOT_ADDED	无法添加指定的配置文件条目。
1926	0x00000786	RPC_S_PRF_ELT_NOT_ADDED	无法添加指定的配置文件元素。
1927	0x00000787	RPC_S_PRF_ELT_NOT_REMOVED	无法删除指定的配置文件元素。
1928	0x00000788	RPC_S_GRP_ELT_NOT_ADDED	无法添加组元素。
1929	0x00000789	RPC_S_GRP_ELT_NOT_REMOVED	无法删除组元素。
1930	0x0000078A	ERROR_KM_DRIVER_BLOCKED	打印机驱动程序与您计算机上启用的阻止 NT 4.0 驱动程序的策略不兼容。

错误			描述
十进制	十六进制	名称	
2000	0x000007D0	ERROR_INVALID_PIXEL_FORMAT	像素格式无效。
2001	0x000007D1	ERROR_BAD_DRIVER	指定的驱动程序无效。
2002	0x000007D2	ERROR_INVALID_WINDOW_STYLE	窗口样式或类别属性对该操作无效。
2003	0x000007D3	ERROR_METAFILE_NOT_SUPPORTED	不支持请求的元文件操作。
2004	0x000007D4	ERROR_TRANSFORM_NOT_SUPPORTED	不支持所请求的转换操作。
2005	0x000007D5	ERROR_CLIPPING_NOT_SUPPORTED	不支持所要求的剪裁操作。

错误			描述
十进制	十六进制	名称	
2010	0x000007DA	ERROR_INVALID_CMM	指定的颜色管理模块无效。
2011	0x000007DB	ERROR_INVALID_PROFILE	指定的颜色配置文件无效。
2012	0x000007DC	ERROR_TAG_NOT_FOUND	未找到指定的标签。
2013	0x000007DD	ERROR_TAG_NOT_PRESENT	所需标签不存在。
2014	0x000007DE	ERROR_DUPLICATE_TAG	指定标签已存在。
2015	0x000007DF	ERROR_PROFILE_NOT_ASSOCIATED_WITH_DEVICE	指定的颜色配置文件与任何设备都不相关。
2016	0x000007E0	ERROR_PROFILE_NOT_FOUND	未找到指定的颜色配置文件。
2017	0x000007E1	ERROR_INVALID_COLORSPACE	指定的颜色空间无效。
2018	0x000007E2	ERROR_ICM_NOT_ENABLED	未启用图像颜色管理。
2019	0x000007E3	ERROR_DELETING_ICM_XFORM	在删除颜色转换时出现错误。
2020	0x000007E4	ERROR_INVALID_TRANSFORM	指定的颜色转换无效。
2021	0x000007E5	ERROR_COLORSPACE_MISMATCH	指定的转换与位图的颜色空间不匹配。
2022	0x000007E6	ERROR_INVALID_COLORINDEX	指定的命名颜色索引不存在于配置文件中。
2108	0x0000083C	ERROR_CONNECTED_OTHER_PASSWORD	网络连接成功，但必须提示用户输入最初指定以外的密码。
2202	0x0000089A	ERROR_BAD_USERNAME	指定的用户名无效。
2250	0x000008CA	ERROR_NOT_CONNECTED	该网络连接不存在。
2401	0x00000961	ERROR_OPEN_FILES	该网络连接有打开的文件或请求等待。
2402	0x00000962	ERROR_ACTIVE_CONNECTIONS	使用中的连接仍然存在。
2404	0x00000964	ERROR_DEVICE_IN_USE	设备正由活动进程使用，无法断开。
2500	0x000009C4	ERROR_PKINIT_FAILURE	在智能卡登录期间，kerberos 协议在验证 KDC 证书时遇到了错误。
2501	0x000009C5	ERROR_SMARTCARD_SUBSYSTEM_FAILURE	kerberos 协议在试图利用智能卡子系统时遇到了错误。

错误			描述
十进制	十六进制	名称	
3000	0x00000BB8	ERROR_UNKNOWN_PRINT_MONITOR	指定的打印显示器未知。
3001	0x00000BB9	ERROR_PRINTER_DRIVER_IN_USE	指定的打印机驱动程序目前正在使用中。
3002	0x00000BBA	ERROR_SPOOL_FILE_NOT_FOUND	未找到线轴文件。
3003	0x00000BBB	ERROR_SPL_NO_STARTDOC	未发出 StartDocPrinter 调用。
3004	0x00000BBC	ERROR_SPL_NO_ADDJOB	未发出 AddJob 调用。
3005	0x00000BBD	ERROR_PRINT_PROCESSOR_ALREADY_INSTALLED	指定的打印处理器已安装。
3006	0x00000BBE	ERROR_PRINT_MONITOR_ALREADY_INSTALLED	指定的打印显示器已安装。
3007	0x00000BBF	ERROR_INVALID_PRINT_MONITOR	指定的打印显示器不具备所需功能。
3008	0x00000BC0	使用中的错误_打印_监控	指定的打印显示器目前正在使用中。
3009	0x00000BC1	ERROR_PRINTER_HAS_JOBS_QUEUED	当有作业排到打印机上时，不允许进行所要求的操作。
3010	0x00000BC2	ERROR_SUCCESS_REBOOT_REQUIRED	请求的操作成功。在系统重新启动之前，更改将不会生效。
3011	0x00000BC3	ERROR_SUCCESS_RESTART_REQUIRED	请求的操作成功。在服务重新启动之前，更改将不会生效。
3012	0x00000BC4	ERROR_PRINTER_NOT_FOUND	未找到打印机。
4000	0x00000FA0	ERROR_WINS_INTERNAL	WINS 在处理该命令时遇到了错误。
4001	0x00000FA1	ERROR_CAN_NOT_DEL_LOCAL_WINS	本地 WINS 无法删除。
4002	0x00000FA2	ERROR_STATIC_INIT	从文件中导入失败。
4003	0x00000FA3	ERROR_INC_BACKUP	备份失败。之前是否做了完整的备份?
4004	0x00000FA4	ERROR_FULL_BACKUP	备份失败。检查正在备份数据库的目录。
4005	0x00000FA5	ERROR_REC_NON_EXISTENT	该名称不存在于 WINS 数据库中。
4006	0x00000FA6	ERROR_RPL_NOT_ALLOWED	不允许复制未配置的伙伴。
4100	0x00001004	ERROR_DHCP_ADDRESS_CONFLICT	DHCP 客户端获得了一个已经在网络上使用的 IP 地址。本地接口将被禁用，直到 DHCP 客户端能够获得一个新的地址。
4200	0x00001068	ERROR_WMI_GUID_NOT_FOUND	传递的 GUID 没有被 WMI 数据提供程序识别为有效。

错误			描述
十进制	十六进制	名称	
4201	0x00001069	ERROR_WMI_INSTANCE_NOT_FOUND	传递的实例名称没有被 WMI 数据提供程序识别为有效。
4202	0x0000106A	ERROR_WMI_ITEMID_NOT_FOUND	传递的数据项目 ID 没有被 WMI 数据提供程序识别为有效。
4203	0x0000106B	ERROR_WMI_TRY_AGAIN	WMI 请求无法完成，应该重试。
4204	0x0000106C	ERROR_WMI_DP_NOT_FOUND	无法找到 WMI 数据提供程序。
4205	0x0000106D	ERROR_WMI_UNRESOLVED_INSTANCE_REF	WMI 数据提供程序引用一个尚未注册的实例集。
4206	0x0000106E	ERROR_WMI_ALREADY_ENABLED	WMI 数据块或事件通知已启用。
4207	0x0000106F	ERROR_WMI_GUID_DISCONNECTED	WMI 数据块已不再可用。
4208	0x00001070	ERROR_WMI_SERVER_UNAVAILABLE	WMI 数据服务不可用。
4209	0x00001071	ERROR_WMI_DP_FAILED	WMI 数据提供程序未能执行该请求。
4210	0x00001072	ERROR_WMI_INVALID_MOF	WMI 的 MOF 信息无效。
4211	0x00001073	ERROR_WMI_INVALID_REGINFO	WMI 的注册信息无效。
4212	0x00001074	ERROR_WMI_ALREADY_DISABLED	WMI 数据块或事件通知已禁用。
4213	0x00001075	ERROR_WMI_READ_ONLY	WMI 数据项或数据块为只读。
4214	0x00001076	ERROR_WMI_SET_FAILURE	WMI 数据项或数据块无法更改。
4300	0x000010CC	ERROR_INVALID_MEDIA	该媒体标识符不代表有效的媒体。
4301	0x000010CD	ERROR_INVALID_LIBRARY	库标识符不代表有效的库。
4302	0x000010CE	ERROR_INVALID_MEDIA_POOL	媒体池标识符不代表有效的媒体池。
4303	0x000010CF	ERROR_DRIVE_MEDIA_MISMATCH	驱动器和媒体不兼容或存在于不同的库中。
4304	0x000010D0	ERROR_MEDIA_OFFLINE	媒体目前存在于一个脱机库中，必须联机才能执行此操作。
4305	0x000010D1	ERROR_LIBRARY_OFFLINE	该操作不能在脱机库上执行。
4306	0x000010D2	ERROR_EMPTY	库、驱动器或媒体池是空的。
4307	0x000010D3	ERROR_NOT_EMPTY	库、驱动器或媒体池必须是空的才能执行这个操作。
4308	0x000010D4	ERROR_MEDIA_UNAVAILABLE	该媒体池或库中目前没有任何媒体可用。
4309	0x000010D5	ERROR_RESOURCE_DISABLED	该操作所需的资源被禁用。
4310	0x000010D6	ERROR_INVALID_CLEANER	媒体标识符不代表一个有效的清洁器。
4311	0x000010D7	ERROR_UNABLE_TO_CLEAN	驱动器不能被清除或不支持清除。
4312	0x000010D8	ERROR_OBJECT_NOT_FOUND	该对象标识符不代表一个有效的对象。
4313	0x000010D9	ERROR_DATABASE_FAILURE	无法从数据库读取或写入数据库。
4314	0x000010DA	ERROR_DATABASE_FULL	该数据库已满。
4315	0x000010DB	ERROR_MEDIA_INCOMPATIBLE	媒体与设备或媒体池不兼容。
4316	0x000010DC	ERROR_RESOURCE_NOT_PRESENT	该操作所需的资源不存在。
4317	0x000010DD	ERROR_INVALID_OPERATION	操作标识符无效。
4318	0x000010DE	ERROR_MEDIA_NOT_AVAILABLE	介质未安装或未准备好使用。
4319	0x000010DF	ERROR_DEVICE_NOT_AVAILABLE	设备尚未准备好使用。
4320	0x000010E0	ERROR_REQUEST_REFUSED	操作员或管理员拒绝了该请求。
4321	0x000010E1	ERROR_INVALID_DRIVE_OBJECT	驱动器标识符不代表有效的驱动器。
4322	0x000010E2	ERROR_LIBRARY_FULL	库已满。没有插槽可供使用。
4323	0x000010E3	ERROR_MEDIUM_NOT_ACCESSIBLE	传输程序不能访问媒体。
4324	0x000010E4	ERROR_UNABLE_TO_LOAD_MEDIUM	无法将媒体装入驱动器。
4325	0x000010E5	ERROR_UNABLE_TO_INVENTORY_DRIVE	无法检索到有关驱动器的状态。
4326	0x000010E6	ERROR_UNABLE_TO_INVENTORY_SLOT	无法检索到关于该插槽的状态。
4327	0x000010E7	ERROR_UNABLE_TO_INVENTORY_TRANSPORT	无法检索到关于传输的状态。
4328	0x000010E8	ERROR_TRANSPORT_FULL	无法使用该传输，因为它已经在使用中。
4329	0x000010E9	ERROR_CONTROLLING_IEPORT	无法打开或关闭弹入/弹出端口。
4330	0x000010EA	ERROR_UNABLE_TO_EJECT_MOUNTED_MEDIA	无法弹出介质，因为它在驱动器中。
4331	0x000010EB	ERROR_CLEANER_SLOT_SET	已保留一个更干净的插槽。
4332	0x000010EC	ERROR_CLEANER_SLOT_NOT_SET	未保留清洁器插槽。
4333	0x000010ED	ERROR_CLEANER_CARTRIDGE_SPENT	清洁器滤芯已经进行了最大数量的驱动器清洁。
4334	0x000010EE	ERROR_UNEXPECTED_OMID	介质标识符不匹配。
4335	0x000010EF	ERROR_CANT_DELETE_LAST_ITEM	该组或资源中最后剩下的项目不能被删除。
4336	0x000010F0	ERROR_MESSAGE_EXCEEDS_MAX_SIZE	提供的信息超过了该参数允许的最大尺寸。

错误			描述
十进制	十六进制	名称	
4337	0x000010F1	ERROR_VOLUME_CONTAINS_SYS_FILES	该卷包含系统或分页文件。
4338	0x000010F2	ERROR_INDIGENOUS_TYPE	该媒体类型不能从该库中删除，因为该库中至少有一个驱动器报告它可以支持该媒体类型。
4339	0x000010F3	ERROR_NO_SUPPORTING_DRIVES	这个脱机媒体不能被挂载在这个系统上，因为没有启用的驱动器可以使用。
4340	0x000010F4	ERROR_CLEANER_CARTRIDGE_INSTALLED	磁带库中有一个清洁器滤芯。
4350	0x000010FE	ERROR_FILE_OFFLINE	远程存储服务无法调用该文件。
4351	0x000010FF	ERROR_REMOTE_STORAGE_NOT_ACTIVE	目前，远程存储服务不可操作。
4352	0x00001100	ERROR_REMOTE_STORAGE_MEDIA_ERROR	远程存储服务遇到了一个媒体错误。
4390	0x00001126	ERROR_NOT_A_REPARSE_POINT	该文件或目录不是一个重解析点。
4391	0x00001127	ERROR_REPARSE_ATTRIBUTE_CONFLICT	不能设置重解析点属性，因为它与现有属性冲突。
4392	0x00001128	ERROR_INVALID_REPARSE_DATA	重解析点缓冲区中的数据无效。
4393	0x00001129	ERROR_REPARSE_TAG_INVALID	重解析点缓冲区中的标签无效。
4394	0x0000112A	ERROR_REPARSE_TAG_MISMATCH	在请求中指定的标签和重解析点中存在的标签不匹配。
4500	0x00001194	ERROR_VOLUME_NOT_SIS_ENABLED	单一实例存储在这个卷上不可用。

错误			描述
十进制	十六进制	名称	
5001	0x00001389	ERROR_DEPENDENT_RESOURCE_EXISTS	集群资源不能被移到另一个组，因为其他资源都依赖于它。
5002	0x0000138A	ERROR_DEPENDENCY_NOT_FOUND	无法找到集群资源的依赖关系。
5003	0x0000138B	ERROR_DEPENDENCY_ALREADY_EXISTS	不能使集群资源依赖指定的资源，因为它已经处于依赖状态。
5004	0x0000138C	ERROR_RESOURCE_NOT_ONLINE	集群资源未联机。
5005	0x0000138D	ERROR_HOST_NODE_NOT_AVAILABLE	集群节点不能用于此操作。
5006	0x0000138E	ERROR_RESOURCE_NOT_AVAILABLE	该集群资源不可用。
5007	0x0000138F	ERROR_RESOURCE_NOT_FOUND	无法找到集群资源。
5008	0x00001390	ERROR_SHUTDOWN_CLUSTER	该集群正在被关闭。
5009	0x00001391	ERROR_CANT_EVICT_ACTIVE_NODE	集群节点不能退出集群，除非该节点已经停止。
5010	0x00001392	ERROR_OBJECT_ALREADY_EXISTS	对象已存在。
5011	0x00001393	ERROR_OBJECT_IN_LIST	该对象已经在列表中。
5012	0x00001394	ERROR_GROUP_NOT_AVAILABLE	该集群组不能用于任何新的请求。
5013	0x00001395	ERROR_GROUP_NOT_FOUND	无法找到集群组。
5014	0x00001396	ERROR_GROUP_NOT_ONLINE	由于集群组未联机，操作无法完成。
5015	0x00001397	ERROR_HOST_NODE_NOT_RESOURCE_OWNER	该集群节点不是资源的所有者。
5016	0x00001398	ERROR_HOST_NODE_NOT_GROUP_OWNER	集群节点不是组的所有者。
5017	0x00001399	ERROR_RESMON_CREATE_FAILED	无法在指定的资源监视器中创建集群资源。
5018	0x0000139A	ERROR_RESMON_ONLINE_FAILED	集群资源无法通过资源监视器联机。
5019	0x0000139B	ERROR_RESOURCE_ONLINE	操作无法完成，因为集群资源已联机。
5020	0x0000139C	ERROR_QUORUM_RESOURCE	集群资源无法删除或脱机，因为是仲裁资源。
5021	0x0000139D	ERROR_NOT_QUORUM_CAPABLE	集群无法使指定的资源成为仲裁资源，因为它不能够成为仲裁资源。
5022	0x0000139E	ERROR_CLUSTER_SHUTTING_DOWN	集群软件正在关闭。
5023	0x0000139F	ERROR_INVALID_STATE	组或资源不处于正确的状态，无法执行所请求的操作。
5024	0x000013A0	ERROR_RESOURCE_PROPERTIES_STORED	属性已存储，但在下次资源联机前，不是所有的修改将生效。
5025	0x000013A1	ERROR_NOT_QUORUM_CLASS	集群无法使指定的资源成为仲裁资源，因为它不属于共享存储类。
5026	0x000013A2	ERROR_CORE_RESOURCE	集群资源无法删除，因为它是核心资源。
5027	0x000013A3	ERROR_QUORUM_RESOURCE_ONLINE_FAILED	仲裁资源未能联机。
5028	0x000013A4	ERROR_QUORUMLOG_OPEN_FAILED	无法成功创建或挂载仲裁日志。
5029	0x000013A5	ERROR_CLUSTERLOG_CORRUPT	集群日志损坏。
5030	0x000013A6	ERROR_CLUSTERLOG_RECORD_EXCEEDS_MAXSIZE	该记录无法写入集群日志，因为它超过了最大限量。
5031	0x000013A7	ERROR_CLUSTERLOG_EXCEEDS_MAXSIZE	集群日志超过了它的最大限量。

错误			描述
十进制	十六进制	名称	
5032	0x000013A8	ERROR_CLUSTERLOG_CHKPOINT_NOT_FOUND	在集群日志中没有发现检查点记录。
5033	0x000013A9	ERROR_CLUSTERLOG_NOT_ENOUGH_SPACE	记录所需的最小磁盘空间不可用。
5034	0x000013AA	ERROR_QUORUM_OWNER_ALIVE	集群节点未能控制仲裁资源，因为该资源被另一个活动节点拥有。
5035	0x000013AB	ERROR_NETWORK_NOT_AVAILABLE	集群网络不适用于此操作。
5036	0x000013AC	ERROR_NODE_NOT_AVAILABLE	集群节点不能用于此操作。
5037	0x000013AD	ERROR_ALL_NODES_NOT_AVAILABLE	所有的集群节点都必须运行，以执行这一操作。
5038	0x000013AE	ERROR_RESOURCE_FAILED	集群资源失败。
5039	0x000013AF	ERROR_CLUSTER_INVALID_NODE	集群节点无效。
5040	0x000013B0	ERROR_CLUSTER_NODE_EXISTS	集群节点已存在。
5041	0x000013B1	ERROR_CLUSTER_JOIN_IN_PROGRESS	节点正在加入集群。
5042	0x000013B2	ERROR_CLUSTER_NODE_NOT_FOUND	未找到集群节点。
5043	0x000013B3	ERROR_CLUSTER_LOCAL_NODE_NOT_FOUND	未找到集群的本地节点信息。
5044	0x000013B4	ERROR_CLUSTER_NETWORK_EXISTS	集群网络已存在。
5045	0x000013B5	ERROR_CLUSTER_NETWORK_NOT_FOUND	未找到集群网络。
5046	0x000013B6	ERROR_CLUSTER_NETINTERFACE_EXISTS	集群网络接口已存在。
5047	0x000013B7	ERROR_CLUSTER_NETINTERFACE_NOT_FOUND	未找到集群的网络接口。
5048	0x000013B8	ERROR_CLUSTER_INVALID_REQUEST	集群请求对这个对象无效。
5049	0x000013B9	ERROR_CLUSTER_INVALID_NETWORK_PROVIDER	集群网络提供程序无效。
5050	0x000013BA	ERROR_CLUSTER_NODE_DOWN	集群节点已停止。
5051	0x000013BB	ERROR_CLUSTER_NODE_UNREACHABLE	无法连接到集群节点。
5052	0x000013BC	ERROR_CLUSTER_NODE_NOT_MEMBER	该集群节点不是集群成员。
5053	0x000013BD	ERROR_CLUSTER_JOIN_NOT_IN_PROGRESS	集群加入操作未进行。
5054	0x000013BE	ERROR_CLUSTER_INVALID_NETWORK	集群网络无效。
5056	0x000013C0	ERROR_CLUSTER_NODE_UP	集群节点已启动。
5057	0x000013C1	ERROR_CLUSTER_IPADDR_IN_USE	集群 IP 地址已在使用中。
5058	0x000013C2	ERROR_CLUSTER_NODE_NOT_PAUSED	该集群节点未暂停。
5059	0x000013C3	ERROR_CLUSTER_NO_SECURITY_CONTEXT	无可用集群安全上下文。
5060	0x000013C4	ERROR_CLUSTER_NETWORK_NOT_INTERNAL	集群网络未为内部集群通信配置。
5061	0x000013C5	ERROR_CLUSTER_NODE_ALREADY_UP	该集群节点已启动。
5062	0x000013C6	ERROR_CLUSTER_NODE_ALREADY_DOWN	该集群节点已停止。
5063	0x000013C7	ERROR_CLUSTER_NETWORK_ALREADY_ONLINE	集群网络已联机。
5064	0x000013C8	ERROR_CLUSTER_NETWORK_ALREADY_OFFLINE	集群网络已脱机。
5065	0x000013C9	ERROR_CLUSTER_NODE_ALREADY_MEMBER	该集群节点已经是集群的成员。
5066	0x000013CA	ERROR_CLUSTER_LAST_INTERNAL_NETWORK	集群网络是唯一为两个或多个活动集群节点之间的内部集群通信而配置的网络。无法从网络中删除内部通信能力。
5067	0x000013CB	ERROR_CLUSTER_NETWORK_HAS_DEPENDENTS	一个或多个集群资源依赖网络向客户提供服务。无法从网络中删除客户访问能力。
5068	0x000013CC	ERROR_INVALID_OPERATION_ON_QUORUM	该操作不能在集群资源上执行，因为它是仲裁资源。你不能使仲裁资源脱机或修改其可能的所有者列表。
5069	0x000013CD	ERROR_DEPENDENCY_NOT_ALLOWED	集群的仲裁资源不允许有任何依赖关系。
5070	0x000013CE	ERROR_CLUSTER_NODE_PAUSED	集群节点已暂停。
5071	0x000013CF	ERROR_NODE_CANT_HOST_RESOURCE	集群资源无法联机。所有者节点无法运行该资源。
5072	0x000013D0	ERROR_CLUSTER_NODE_NOT_READY	集群节点没有准备好执行请求的操作。
5073	0x000013D1	ERROR_CLUSTER_NODE_SHUTTING_DOWN	集群节点正在关闭。
5074	0x000013D2	ERROR_CLUSTER_JOIN_ABORTED	集群加入操作中止。
5075	0x000013D3	ERROR_CLUSTER_INCOMPATIBLE_VERSIONS	由于加入节点和其支持者之间的软件版本不兼容，集群加入操作失败。
5076	0x000013D4	ERROR_CLUSTER_MAXNUM_OF_RESOURCES_EXCEEDED	该资源无法创建，因为集群已经达到了其可以监控的资源数量的限制。
5077	0x000013D5	ERROR_CLUSTER_SYSTEM_CONFIG_CHANGED	系统配置在集群加入或形成操作中发生了改变。加入或形成操作中止。
5078	0x000013D6	ERROR_CLUSTER_RESOURCE_TYPE_NOT_FOUND	未找到指定的资源类型。

错误			描述
十进制	十六进制	名称	
5079	0x000013D7	ERROR_CLUSTER_RESTYPE_NOT_SUPPORTED	指定节点不支持这种类型的资源。这可能是由于版本不一致或由于该节点上没有资源 DLL。
5080	0x000013D8	ERROR_CLUSTER_RESNAME_NOT_FOUND	指定资源名称不受该资源 DLL 支持。这可能是由于提供给资源 DLL 的名字错误（或改变）。
5081	0x000013D9	ERROR_CLUSTER_NO_RPC_PACKAGES_REGISTERED	无法在 RPC 服务器上注册认证包。
5082	0x000013DA	ERROR_CLUSTER_OWNER_NOT_IN_PREFLIST	你不能将该组联机，因为该组的所有者不在该组的首选列表中。如需改变组的所有者节点，请移动该组。
5083	0x000013DB	ERROR_CLUSTER_DATABASE_SEQMISMATCH	加入操作失败，因为集群数据库序列号已经改变或与锁定程序节点不兼容。如果集群数据库在加入过程中发生变化，这可能在加入操作中发生。
5084	0x000013DC	ERROR_RESMON_INVALID_STATE	当资源处于当前状态时，资源监视器将不允许执行失败操作。如果资源处于等待状态，可能会发生这种情况。
5085	0x000013DD	ERROR_CLUSTER_GUM_NOT_LOCKER	非锁定程序代码得到了保留锁定的请求，以便进行全局更新。
5086	0x000013DE	ERROR_QUORUM_DISK_NOT_FOUND	集群服务无法找到仲裁磁盘。
5087	0x000013DF	ERROR_DATABASE_BACKUP_CORRUPT	备份的集群数据库可能已经损坏。
5088	0x000013E0	ERROR_CLUSTER_NODE_ALREADY_HAS_DFS_ROOT	这个集群节点中已经存在 DFS 根。
5089	0x000013E1	ERROR_RESOURCE_PROPERTY_UNCHANGEABLE	尝试修改资源属性失败，因为与另一个现有的属性冲突。
5890	0x00001702	ERROR_CLUSTER_MEMBERSHIP_INVALID_STATE	试图进行的操作与节点的当前成员状态不兼容。
5891	0x00001703	ERROR_CLUSTER_QUORUMLOG_NOT_FOUND	仲裁资源不包含仲裁日志。
5892	0x00001704	ERROR_CLUSTER_MEMBERSHIP_HALT	成员身份引擎申请关闭该节点上的集群服务。
5893	0x00001705	ERROR_CLUSTER_INSTANCE_ID_MISMATCH	加入操作失败，因为加入节点的集群实例 ID 与支持者节点的集群实例 ID 不匹配。
5894	0x00001706	ERROR_CLUSTER_NETWORK_NOT_FOUND_FOR_IP	无法找到指定 IP 地址的匹配网络。请同时指定子网掩码和集群网络。
5895	0x00001707	ERROR_CLUSTER_PROPERTY_DATA_TYPE_MISMATCH	属性的实际数据类型与属性的预期数据类型不一致。
5896	0x00001708	ERROR_CLUSTER_EVICT_WITHOUT_CLEANUP	该集群节点被成功退出集群。该节点没有被清理，因为它不支持退出清理功能。

错误			描述
十进制	十六进制	名称	
6000	0x00001770	ERROR_ENCRYPTION_FAILED	指定的文件无法加密。
6001	0x00001771	ERROR_DECRYPTION_FAILED	指定的文件无法解密。
6002	0x00001772	ERROR_FILE_ENCRYPTED	指定的文件已加密，但用户没有能力解密。
6003	0x00001773	ERROR_NO_RECOVERY_POLICY	没有为该系统配置有效的加密恢复策略。
6004	0x00001774	ERROR_NO_EFS	该系统没有加载所需的加密驱动程序。
6005	0x00001775	ERROR_WRONG_EFS	文件加密所使用的加密驱动程序与目前加载的加密驱动程序不同。
6006	0x00001776	ERROR_NO_USER_KEYS	没有为该用户定义 EFS 密钥。
6007	0x00001777	ERROR_FILE_NOT_ENCRYPTED	指定的文件未加密。
6008	0x00001778	ERROR_NOT_EXPORT_FORMAT	指定的文件不是定义的 EFS 导出格式。
6009	0x00001779	ERROR_FILE_READ_ONLY	指定的文件是只读文件。
6010	0x0000177A	ERROR_DIR_EFS_DISALLOWED	该目录已禁止加密。
6011	0x0000177B	ERROR_EFS_SERVER_NOT_TRUSTED	该服务器不被信任用于远程加密操作。
6012	0x0000177C	ERROR_BAD_RECOVERY_POLICY	为该系统配置的恢复策略包含无效的恢复证书。
6013	0x0000177D	ERROR_EFS_ALG_BLOB_TOO_BIG	在源文件上使用的加密算法需要比目标文件使用的更大的密钥缓冲区。
6014	0x0000177E	ERROR_VOLUME_NOT_SUPPORT_EFS	该磁盘分区不支持文件加密。
6118	0x000017E6	ERROR_NO_BROWSER_SERVERS_FOUND	该工作组的服务器列表目前尚不可用。
6200	0x00001838	SCHED_E_SERVICE_NOT_LOCALSYSTEM	任务调度程序服务必须被配置为在系统账户中运行才能正常工作。个别任务可以被配置为在其他账户中运行。
7001	0x00001B59	ERROR_CTX_WINSTATION_NAME_INVALID	指定的会话名称无效。

错误			描述
十进制	十六进制	名称	
7002	0x00001B5A	ERROR_CTX_INVALID_PD	指定的协议驱动程序无效。
7003	0x00001B5B	ERROR_CTX_PD_NOT_FOUND	在系统路径中未找到指定的协议驱动程序。
7004	0x00001B5C	ERROR_CTX_WD_NOT_FOUND	在系统路径中未找到指定的终端连接驱动程序。
7005	0x00001B5D	ERROR_CTX_CANNOT_MAKE_EVENTLOG_ENTRY	无法为这个会话创建事件记录的注册表键。
7006	0x00001B5E	ERROR_CTX_SERVICE_NAME_COLLISION	系统中已经存在同名服务。
7007	0x00001B5F	ERROR_CTX_CLOSE_PENDING	在会话上关闭操作正在等待。
7008	0x00001B60	ERROR_CTX_NO_OUTBUF	没有可用的输出缓冲区。
7009	0x00001B61	ERROR_CTX_MODEM_INF_NOT_FOUND	未找到 MODEM.INF 文件。
7010	0x00001B62	ERROR_CTX_INVALID_MODEMNAME	在 MODEM.INF 中没有找到调制解调器的名称。
7011	0x00001B63	ERROR_CTX_MODEM_RESPONSE_ERROR	调制解调器不接受发给它的命令。验证配置的调制解调器名称是否与连接的调制解调器相匹配。
7012	0x00001B64	ERROR_CTX_MODEM_RESPONSE_TIMEOUT	调制解调器没有对发给它的命令作出反应。确认调制解调器已正确接上电缆并接通电源。
7013	0x00001B65	ERROR_CTX_MODEM_RESPONSE_NO_CARRIER	载体检测失败或由于断开连接导致载体被丢弃。
7014	0x00001B66	ERROR_CTX_MODEM_RESPONSE_NO_DIALTONE	在规定时间内未检测到拨号音。验证电话线是否正确连接并正常工作。
7015	0x00001B67	ERROR_CTX_MODEM_RESPONSE_BUSY	回拨时在远程站点检测到忙音信号。
7016	0x00001B68	ERROR_CTX_MODEM_RESPONSE_VOICE	回拨时在远程站点检测到语音。
7017	0x00001B69	ERROR_CTX_TD_ERROR	传输驱动程序错误
7022	0x00001B6E	ERROR_CTX_WINSTATION_NOT_FOUND	无法找到指定的会话。
7023	0x00001B6F	ERROR_CTX_WINSTATION_ALREADY_EXISTS	指定的会话名称已经在使用中。
7024	0x00001B70	ERROR_CTX_WINSTATION_BUSY	请求的操作无法完成，因为终端连接目前正忙于处理连接、断开、重置或删除操作。
7025	0x00001B71	ERROR_CTX_BAD_VIDEO_MODE	试图连接到一个当前客户端不支持视频模式的会话。
7035	0x00001B7B	ERROR_CTX_GRAPHICS_INVALID	该应用程序试图启用 DOS 图形模式。不支持 DOS 图形模式。
7037	0x00001B7D	ERROR_CTX_LOGON_DISABLED	你的交互式登录权限已被禁用。请联系管理员。
7038	0x00001B7E	ERROR_CTX_NOT_CONSOLE	所要求的操作只能在系统控制台进行。这通常是由于驱动程序或系统 DLL 要求直接访问控制台的结果。
7040	0x00001B80	ERROR_CTX_CLIENT_QUERY_TIMEOUT	客户端未能响应服务器的连接信息。
7041	0x00001B81	ERROR_CTX_CONSOLE_DISCONNECT	不支持断开控制台会话的连接。
7042	0x00001B82	ERROR_CTX_CONSOLE_CONNECT	不支持将断开连接的会话重新连接到控制台。
7044	0x00001B84	ERROR_CTX_SHADOW_DENIED	远程控制另一个会话的请求被拒绝。
7045	0x00001B85	ERROR_CTX_WINSTATION_ACCESS_DENIED	请求的会话访问被拒绝。
7049	0x00001B89	ERROR_CTX_INVALID_WD	指定的终端连接驱动程序无效。
7050	0x00001B8A	ERROR_CTX_SHADOW_INVALID	所要求的会话无法远程控制。这可能是因为该会话已断开，或当前没有用户登录。
7051	0x00001B8B	ERROR_CTX_SHADOW_DISABLED	所请求的会话没有被配置为允许远程控制。
7052	0x00001B8C	ERROR_CTX_CLIENT_LICENSE_IN_USE	连接到该终端服务器的请求遭到拒绝。你的终端服务器客户端许可证号码目前被其他用户使用。请致电系统管理员，获得唯一的许可证号码。
7053	0x00001B8D	ERROR_CTX_CLIENT_LICENSE_NOT_SET	连接到该终端服务器的请求遭到拒绝。尚未为这个终端服务器客户端的副本输入你的终端服务器客户端的许可证号码。请联系系统管理员。
7054	0x00001B8E	ERROR_CTX_LICENSE_NOT_AVAILABLE	系统已达到其许可的登录限制。请稍后再试。
7055	0x00001B8F	ERROR_CTX_LICENSE_CLIENT_INVALID	所用客户端未获得使用该系统的许可。登录请求遭到拒绝。
7056	0x00001B90	ERROR_CTX_LICENSE_EXPIRED	系统许可证已过期。登录请求遭到拒绝。
7057	0x00001B91	ERROR_CTX_SHADOW_NOT_RUNNING	无法终止远程控制，因为指定的会话目前没有被远程控制。

错误			描述
十进制	十六进制	名称	
8001	0x00001F41	FRS_ERR_INVALID_API_SEQUENCE	文件复制服务 API 被错误调用。
8002	0x00001F42	FRS_ERR_STARTING_SERVICE	文件复制服务无法启动。
8003	0x00001F43	FRS_ERR_STOPPING_SERVICE	文件复制服务无法停止。

错误			描述
十进制	十六进制	名称	
8004	0x00001F44	FRS_ERR_INTERNAL_API	文件复制服务 API 终止了请求。事件日志可能有更多信息。
8005	0x00001F45	FRS_ERR_INTERNAL	文件复制服务终止了请求。事件日志可能有更多信息。
8006	0x00001F46	FRS_ERR_SERVICE_COMM	无法联系到文件复制服务。事件日志可能有更多信息。
8007	0x00001F47	FRS_ERR_INSUFFICIENT_PRIV	文件复制服务不能满足请求，因为用户的权限不足。事件日志可能有更多信息。
8008	0x00001F48	FRS_ERR_AUTHENTICATION	文件复制服务不能满足请求，因为认证的 RPC 不可用。事件日志可能有更多信息。
8009	0x00001F49	FRS_ERR_PARENT_INSUFFICIENT_PRIV	文件复制服务不能满足请求，因为用户在域控制器上的权限不足。事件日志可能有更多信息。
8010	0x00001F4A	FRS_ERR_PARENT_AUTHENTICATION	文件复制服务不能满足请求，因为认证的 RPC 在域控制器上不可用。事件日志可能有更多信息。
8011	0x00001F4B	FRS_ERR_CHILD_TO_PARENT_COMM	文件复制服务无法与域控制器上的文件复制服务通信。事件日志可能有更多信息。
8012	0x00001F4C	FRS_ERR_PARENT_TO_CHILD_COMM	域控制器上的文件复制服务不能与该计算机上的文件复制服务进行通信。事件日志可能有更多信息。
8013	0x00001F4D	FRS_ERR_SYSVOL_POPULATE	由于内部错误，文件复制服务无法进入系统卷。事件日志可能有更多信息。
8014	0x00001F4E	FRS_ERR_SYSVOL_POPULATE_TIMEOUT	由于内部超时，文件复制服务无法进入系统卷。事件日志可能有更多信息。
8015	0x00001F4F	FRS_ERR_SYSVOL_IS_BUSY	文件复制服务无法处理该请求。系统卷正忙于处理前一个请求。
8016	0x00001F50	FRS_ERR_SYSVOL_DEMOTE	由于内部错误，文件复制服务不能停止对系统卷的复制。事件日志可能有更多信息。
8017	0x00001F51	FRS_ERR_INVALID_SERVICE_PARAMETER	文件复制服务检测到一个无效的参数。
8200	0x00002008	ERROR_DS_NOT_INSTALLED	在安装目录服务时发生了一个错误。关于更多信息，请参见事件日志。
8201	0x00002009	ERROR_DS_MEMBERSHIP_EVALUATED_LOCALLY	目录服务在本地评估了组成员资格。
8202	0x0000200A	ERROR_DS_NO_ATTRIBUTE_OR_VALUE	指定的目录服务属性或值不存在。
8203	0x0000200B	ERROR_DS_INVALID_ATTRIBUTE_SYNTAX	目录服务指定的属性语法无效。
8204	0x0000200C	ERROR_DS_ATTRIBUTE_TYPE_UNDEFINED	目录服务指定的属性类型未定义。
8205	0x0000200D	ERROR_DS_ATTRIBUTE_OR_VALUE_EXISTS	指定的目录服务属性或值已存在。
8206	0x0000200E	ERROR_DS_BUSY	目录服务忙。
8207	0x0000200F	ERROR_DS_UNAVAILABLE	目录服务不可用。
8208	0x00002010	ERROR_DS_NO_RIDS_ALLOCATED	目录服务无法分配相对标识符。
8209	0x00002011	ERROR_DS_NO_MORE_RIDS	目录服务已经用完了相对标识符库。
8210	0x00002012	ERROR_DS_INCORRECT_ROLE_OWNER	由于目录服务不是该类型操作的主控，因此无法执行请求的操作。
8211	0x00002013	ERROR_DS_RIDMGR_INIT_ERROR	目录服务无法初始化分配相对标识符的子系统。
8212	0x00002014	ERROR_DS_OBJ_CLASS_VIOLATION	请求的操作不满足与对象的类别相关的一个或多个约束。
8213	0x00002015	ERROR_DS_CANT_ON_NON_LEAF	目录服务只能在一个叶对象上执行所请求的操作。
8214	0x00002016	ERROR_DS_CANT_ON_RDN	目录服务不能对一个对象的 RDN 属性执行请求的操作。
8215	0x00002017	ERROR_DS_CANT_MOD_OBJ_CLASS	目录服务检测到试图修改一个对象的对象类别。
8216	0x00002018	ERROR_DS_CROSS_DOM_MOVE_ERROR	无法执行请求的跨域移动操作。
8217	0x00002019	ERROR_DS_GC_NOT_AVAILABLE	无法联系全局目录服务器。
8218	0x0000201A	ERROR_SHARED_POLICY	策略对象共享，且只能在根目录上修改。
8219	0x0000201B	ERROR_POLICY_OBJECT_NOT_FOUND	政策对象不存在。
8220	0x0000201C	ERROR_POLICY_ONLY_IN_DS	要求的政策信息只在目录服务中。
8221	0x0000201D	ERROR_PROMOTION_ACTIVE	域控制器升级目前正在进行。
8222	0x0000201E	ERROR_NO_PROMOTION_ACTIVE	域控制器升级目前未进行
8224	0x00002020	ERROR_DS_OPERATIONS_ERROR	发生了一个操作错误。
8225	0x00002021	ERROR_DS_PROTOCOL_ERROR	发生了一个协议错误。
8226	0x00002022	ERROR_DS_TIMELIMIT_EXCEEDED	超过了该请求的时间限制。
8227	0x00002023	ERROR_DS_SIZELIMIT_EXCEEDED	超过了该请求的大小限制。

错误			描述
十进制	十六进制	名称	
8228	0x00002024	ERROR_DS_ADMIN_LIMIT_EXCEEDED	超过了该请求的管理限制。
8229	0x00002025	ERROR_DS_COMPARE_FALSE	比较响应为false。
8230	0x00002026	ERROR_DS_COMPARE_TRUE	比较响应为true。
8231	0x00002027	ERROR_DS_AUTH_METHOD_NOT_SUPPORTED	服务器不支持所请求的认证方法。
8232	0x00002028	ERROR_DS_STRONG_AUTH_REQUIRED	该服务器需要更安全的认证方法。
8233	0x00002029	ERROR_DS_INAPPROPRIATE_AUTH	不适当的认证。
8234	0x0000202A	ERROR_DS_AUTH_UNKNOWN	认证机制不明。
8235	0x0000202B	ERROR_DS_REFERRAL	从服务器返回了一个建议。
8236	0x0000202C	ERROR_DS_UNAVAILABLE_CRIT_EXTENSION	服务器不支持请求的关键扩展。
8237	0x0000202D	ERROR_DS_CONFIDENTIALITY_REQUIRED	该请求需要安全连接。
8238	0x0000202E	ERROR_DS_INAPPROPRIATE_MATCHING	匹配不当。
8239	0x0000202F	ERROR_DS_CONSTRAINT_VIOLATION	发生了一个违反约束的情况。
8240	0x00002030	ERROR_DS_NO_SUCH_OBJECT	服务器上没有这样的对象。
8241	0x00002031	ERROR_DS_ALIAS_PROBLEM	有一个别名问题。
8242	0x00002032	ERROR_DS_INVALID_DN_SYNTAX	指定了一个无效的 dn 语法。
8243	0x00002033	ERROR_DS_IS_LEAF	该对象是叶对象。
8244	0x00002034	ERROR_DS_ALIAS_DEREF_PROBLEM	有一个别名废弃问题。
8245	0x00002035	ERROR_DS_UNWILLING_TO_PERFORM	服务器不愿意处理该请求。
8246	0x00002036	ERROR_DS_LOOP_DETECT	检测到了一个循环。
8247	0x00002037	ERROR_DS_NAMING_VIOLATION	有一个违反命名的问题。
8248	0x00002038	ERROR_DS_OBJECT_RESULTS_TOO_LARGE	结果集太大。
8249	0x00002039	ERROR_DS_AFFECTS_MULTIPLE_DSAS	该操作影响到多个 DSA
8250	0x0000203A	ERROR_DS_SERVER_DOWN	该服务器没有运行。
8251	0x0000203B	ERROR_DS_LOCAL_ERROR	出现一个本地错误。
8252	0x0000203C	ERROR_DS_ENCODING_ERROR	出现一个编码错误。
8253	0x0000203D	ERROR_DS_DECODING_ERROR	发生了一个解码错误。
8254	0x0000203E	ERROR_DS_FILTER_UNKNOWN	无法识别搜索过滤器。
8255	0x0000203F	ERROR_DS_PARAM_ERROR	一个或多个参数非法。
8256	0x00002040	ERROR_DS_NOT_SUPPORTED	不支持指定的方法。
8257	0x00002041	ERROR_DS_NO_RESULTS_RETURNED	未得到任何结果。
8258	0x00002042	ERROR_DS_CONTROL_NOT_FOUND	服务器不支持指定的控件。
8259	0x00002043	ERROR_DS_CLIENT_LOOP	客户端检测到一个引用循环。
8260	0x00002044	ERROR_DS_REFERRAL_LIMIT_EXCEEDED	超过了预设的转介限制。
8261	0x00002045	ERROR_DS_SORT_CONTROL_MISSING	该搜索需要一个 SORT 控件。
8262	0x00002046	ERROR_DS_OFFSET_RANGE_ERROR	搜索结果超过了指定的偏移范围。
8301	0x0000206D	ERROR_DS_ROOT_MUST_BE_NC	根对象必须是一个命名上下文的头。根对象不能有一个实例化的父对象。
8302	0x0000206E	ERROR_DS_ADD_REPLICA_INHIBITED	无法执行添加副本的操作。命名上下文必须可写，以便创建副本。
8303	0x0000206F	ERROR_DS_ATT_NOT_DEF_IN_SCHEMA	引用了模式中没有定义的属性。
8304	0x00002070	ERROR_DS_MAX_OBJ_SIZE_EXCEEDED	超过了一个对象的最大尺寸。
8305	0x00002071	ERROR_DS_OBJ_STRING_NAME_EXISTS	试图向目录中添加一个名称已经被使用的对象。
8306	0x00002072	ERROR_DS_NO_RDN_DEFINED_IN_SCHEMA	试图添加一个在模式中没有定义 RDN 的类的对象。
8307	0x00002073	ERROR_DS_RDN_DOESNT_MATCH_SCHEMA	试图添加一个使用 RDN 的对象，但该 RDN 不是模式中定义的 RDN。
8308	0x00002074	ERROR_DS_NO_REQUESTED_ATTS_FOUND	在这些对象上没有发现所要求的属性。
8309	0x00002075	ERROR_DS_USER_BUFFER_TO_SMALL	用户缓冲区太小。
8310	0x00002076	ERROR_DS_ATT_IS_NOT_ON_OBJ	操作中指定的属性在对象上不存在。
8311	0x00002077	ERROR_DS_ILLEGAL_MOD_OPERATION	修改操作非法。某些方面的修改不允许。
8312	0x00002078	ERROR_DS_OBJ_TOO_LARGE	指定的对象太大。
8313	0x00002079	ERROR_DS_BAD_INSTANCE_TYPE	指定的实例类型无效。
8314	0x0000207A	ERROR_DS_MASTERDSA_REQUIRED	该操作必须在主控 DSA 进行。
8315	0x0000207B	ERROR_DS_OBJECT_CLASS_REQUIRED	必须指定对象类别属性。
8316	0x0000207C	ERROR_DS_MISSING_REQUIRED_ATT	缺少必需的属性。

错误			描述
十进制	十六进制	名称	
8317	0x0000207D	ERROR_DS_ATT_NOT_DEF_FOR_CLASS	试图修改一个对象，以包括一个对其类别不合法的属性。
8318	0x0000207E	ERROR_DS_ATT_ALREADY_EXISTS	指定的属性已经存在于对象上。
8320	0x00002080	ERROR_DS_CANT_ADD_ATT_VALUES	指定的属性不存在，或者没有值。
8321	0x00002081	ERROR_DS_SINGLE_VALUE_CONSTRAINT	为一个只能有一个值的属性指定了多个值。
8322	0x00002082	ERROR_DS_RANGE_CONSTRAINT	该属性的一个值不在可接受的数值范围内。
8323	0x00002083	ERROR_DS_ATT_VAL_ALREADY_EXISTS	指定的值已存在。
8324	0x00002084	ERROR_DS_CANT_REM_MISSING_ATT	属性无法删除，因为它不存在于对象上。
8325	0x00002085	ERROR_DS_CANT_REM_MISSING_ATT_VAL	属性值无法删除，因为它不存在于对象上。
8326	0x00002086	ERROR_DS_ROOT_CANT_BE_SUBREF	指定的根对象不能是子参考。
8327	0x00002087	ERROR_DS_NO_CHAINING	不允许链接。
8328	0x00002088	ERROR_DS_NO_CHAINED_EVAL	不允许进行链接评估。
8329	0x00002089	ERROR_DS_NO_PARENT_OBJECT	操作无法执行，因为该对象的父级未实例化或已删除。
8330	0x0000208A	ERROR_DS_PARENT_IS_AN_ALIAS	不允许拥有一个别名的父类。别名是叶对象。
8331	0x0000208B	ERROR_DS_CANT_MIX_MASTER_AND_REPS	对象和父类必须是相同的类型，不是都是原件就是都是副本。
8332	0x0000208C	ERROR_DS_CHILDREN_EXIST	因为存在子对象，所以不能执行该操作。这个操作只能在一个叶对象上进行。
8333	0x0000208D	ERROR_DS_OBJ_NOT_FOUND	未找到目录对象。
8334	0x0000208E	ERROR_DS_ALIASED_OBJ_MISSING	别名对象丢失。
8335	0x0000208F	ERROR_DS_BAD_NAME_SYNTAX	对象名称语法不对。
8336	0x00002090	ERROR_DS_ALIAS_POINTS_TO_ALIAS	不允许一个别名参考另一个别名。
8337	0x00002091	ERROR_DS_CANT_DEREF_ALIAS	别名不能解除参考。
8338	0x00002092	ERROR_DS_OUT_OF_SCOPE	操作超出了范围。
8339	0x00002093	ERROR_DS_OBJECT_BEING_REMOVED	操作无法继续，因为该对象正在删除。
8340	0x00002094	ERROR_DS_CANT_DELETE_DSA_OBJ	DSA 对象无法删除。
8341	0x00002095	ERROR_DS_GENERIC_ERROR	发生了一个目录服务错误。
8342	0x00002096	ERROR_DS_DSA_MUST_BE_INT_MASTER	操作只能在内部主控 DSA 对象上执行。
8343	0x00002097	ERROR_DS_CLASS_NOT_DSA	对象必须为 DSA 类别。
8344	0x00002098	ERROR_DS_INSUFF_ACCESS_RIGHTS	访问权限不足，无法执行操作。
8345	0x00002099	ERROR_DS_ILLEGAL_SUPERIOR	对象无法添加，因为父级不在可能的上级列表中。
8346	0x0000209A	ERROR_DS_ATTRIBUTE_OWNED_BY_SAM	不允许访问该属性，因为该属性是由安全账户管理器（SAM）拥有。
8347	0x0000209B	ERROR_DS_NAME_TOO_MANY_PARTS	名称包括太多部分。
8348	0x0000209C	ERROR_DS_NAME_TOO_LONG	名称太长。
8349	0x0000209D	ERROR_DS_NAME_VALUE_TOO_LONG	名称值太长。
8350	0x0000209E	ERROR_DS_NAME_UNPARSEABLE	目录服务在解析一个名称时遇到了错误。
8351	0x0000209F	ERROR_DS_NAME_TYPE_UNKNOWN	目录服务无法获得一个名称的属性类型。
8352	0x000020A0	ERROR_DS_NOT_AN_OBJECT	这个名称不能识别一个对象；这个名称识别的是一个幻象。
8353	0x000020A1	ERROR_DS_SEC_DESC_TOO_SHORT	安全描述符太短。
8354	0x000020A2	ERROR_DS_SEC_DESC_INVALID	安全描述符无效。
8355	0x000020A3	ERROR_DS_NO_DELETED_NAME	为删除的对象创建名称失败。
8356	0x000020A4	ERROR_DS_SUBREF_MUST_HAVE_PARENT	一个新的子参考的父类必须存在。
8357	0x000020A5	ERROR_DS_NCNAME_MUST_BE_NC	该对象必须是一个命名上下文。
8358	0x000020A6	ERROR_DS_CANT_ADD_SYSTEM_ONLY	不允许添加系统拥有的属性。
8359	0x000020A7	ERROR_DS_CLASS_MUST_BE_CONCRETE	对象的类必须是结构性的；你不能实例化一个抽象的类。
8360	0x000020A8	ERROR_DS_INVALID_DMD	无法找到模式对象。
8361	0x000020A9	ERROR_DS_OBJ_GUID_EXISTS	已存在具有此 GUID（非活动或活动）的本地对象。
8362	0x000020AA	ERROR_DS_NOT_ON_BACKLINK	该操作不能在后方链接上执行。
8363	0x000020AB	ERROR_DS_NO_CROSSREF_FOR_NC	无法找到指定命名上下文的交叉引用。
8364	0x000020AC	ERROR_DS_SHUTTING_DOWN	由于目录服务正在关闭，该操作无法执行。
8365	0x000020AD	ERROR_DS_UNKNOWN_OPERATION	目录服务请求无效。

错误			描述
十进制	十六进制	名称	
8366	0x000020AE	ERROR_DS_INVALID_ROLE_OWNER	无法读取角色所有者属性。
8367	0x000020AF	ERROR_DS_COULDNT_CONTACT_FSMO	请求的 FSMO 操作失败。无法联系到目前的 FSMO 持有人。
8368	0x000020B0	ERROR_DS_CROSS_NC_DN_RENAME	不允许跨过一个命名上下文修改 DN。
8369	0x000020B1	ERROR_DS_CANT_MOD_SYSTEM_ONLY	属性无法修改，因为它是由系统拥有。
8370	0x000020B2	ERROR_DS_REPLICATOR_ONLY	只有复制器可以执行这一功能。
8371	0x000020B3	ERROR_DS_OBJ_CLASS_NOT_DEFINED	指定的类别未定义。
8372	0x000020B4	ERROR_DS_OBJ_CLASS_NOT_SUBCLASS	指定类别不是子类。
8373	0x000020B5	ERROR_DS_NAME_REFERENCE_INVALID	名称参考无效。
8374	0x000020B6	ERROR_DS_CROSS_REF_EXISTS	已经存在交叉参考。
8375	0x000020B7	ERROR_DS_CANT_DEL_MASTER_CROSSREF	不允许删除主控交叉参考。
8376	0x000020B8	ERROR_DS_SUBTREE_NOTIFY_NOT_NC_HEAD	子树通知只在 NC 头上支持。
8377	0x000020B9	ERROR_DS_NOTIFY_FILTER_TOO_COMPLEX	通知过滤器太复杂。
8378	0x000020BA	ERROR_DS_DUP_RDN	模式更新失败：重复的 RDN。
8379	0x000020BB	ERROR_DS_DUP_OID	模式更新失败：重复的 OID
8380	0x000020BC	ERROR_DS_DUP_MAPI_ID	模式更新失败：重复的 MAPI 标识符。
8381	0x000020BD	ERROR_DS_DUP_SCHEMA_ID_GUID	模式更新失败：重复的模式 ID GUID。
8382	0x000020BE	ERROR_DS_DUP_LDAP_DISPLAY_NAME	模式更新失败：重复的 LDAP 显示名称。
8383	0x000020BF	ERROR_DS_SEMANTIC_ATT_TEST	模式更新失败：范围下限小于范围上限
8384	0x000020C0	ERROR_DS_SYNTAX_MISMATCH	模式更新失败：语法不匹配
8385	0x000020C1	ERROR_DS_EXISTS_IN_MUST_HAVE	模式删除失败：属性在 must-contain 中使用
8386	0x000020C2	ERROR_DS_EXISTS_IN_MAY_HAVE	模式删除失败：属性在 may-contain 中使用
8387	0x000020C3	ERROR_DS_NONEXISTENT_MAY_HAVE	模式更新失败：may-contain 中的属性不存在
8388	0x000020C4	ERROR_DS_NONEXISTENT_MUST_HAVE	模式更新失败：must-contain 中的属性不存在
8389	0x000020C5	ERROR_DS_AUX_CLS_TEST_FAIL	模式更新失败：在辅助类别列表中的类别不存在或不是一个辅助类别
8390	0x000020C6	ERROR_DS_NONEXISTENT_POSS_SUP	模式更新失败：poss-superiors 中的类别不存在
8391	0x000020C7	ERROR_DS_SUB_CLS_TEST_FAIL	模式更新失败：subclassof 列表中的类别不存在或不满足层次规则
8392	0x000020C8	ERROR_DS_BAD_RDN_ATT_ID_SYNTAX	模式更新失败：Rdn-Att-Id 语法不对
8393	0x000020C9	ERROR_DS_EXISTS_IN_AUX_CLS	模式删除失败：类别作为辅助类别使用
8394	0x000020CA	ERROR_DS_EXISTS_IN_SUB_CLS	模式删除失败：类别作为子类使用
8395	0x000020CB	ERROR_DS_EXISTS_IN_POSS_SUP	模式删除失败：类别作为 poss superior 使 用
8396	0x000020CC	ERROR_DS_RECALCSCHEMA_FAILED	模式更新在重新计算验证缓存时失败。
8397	0x000020CD	ERROR_DS_TREE_DELETE_NOT_FINISHED	树的删除未完成。
8398	0x000020CE	ERROR_DS_CANT_DELETE	无法执行请求的删除操作。
8399	0x000020CF	ERROR_DS_ATT_SCHEMA_REQ_ID	无法读取模式记录的管辖类标识符。
8400	0x000020D0	ERROR_DS_BAD_ATT_SCHEMA_SYNTAX	属性模式语法不对。
8401	0x000020D1	ERROR_DS_CANT_CACHE_ATT	属性无法缓存。
8402	0x000020D2	ERROR_DS_CANT_CACHE_CLASS	类别无法缓存。
8403	0x000020D3	ERROR_DS_CANT_REMOVE_ATT_CACHE	属性无法从缓存中删除。
8404	0x000020D4	ERROR_DS_CANT_REMOVE_CLASS_CACHE	类别无法从缓存中删除。
8405	0x000020D5	ERROR_DS_CANT_RETRIEVE_DN	无法读取区分的名称属性。
8406	0x000020D6	ERROR_DS_MISSING_SUPREF	缺少一个必要的子参考。
8407	0x000020D7	ERROR_DS_CANT_RETRIEVE_INSTANCE	无法检索到实例类型属性。
8408	0x000020D8	ERROR_DS_CODE_INCONSISTENCY	发生了内部错误。
8409	0x000020D9	ERROR_DS_DATABASE_ERROR	发生了数据库错误。
8410	0x000020DA	ERROR_DS_GOVERNSID_MISSING	缺少 GOVERNSID 属性。
8411	0x000020DB	ERROR_DS_MISSING_EXPECTED_ATT	缺少一个预期的属性。
8412	0x000020DC	ERROR_DS_NCNAME_MISSING_CR_REF	指定的命名上下文缺少一个交叉引用。
8413	0x000020DD	ERROR_DS_SECURITY_CHECKING_ERROR	发生了一个安全检查错误。
8414	0x000020DE	ERROR_DS_SCHEMA_NOT_LOADED	模式未加载。
8415	0x000020DF	ERROR_DS_SCHEMA_ALLOC_FAILED	模式分配失败。请检查机器内存是否不足。
8416	0x000020E0	ERROR_DS_ATT_SCHEMA_REQ_SYNTAX	未能获得属性模式的所需语法。

错误			描述
十进制	十六进制	名称	
8417	0x000020E1	ERROR_DS_GCVERIFY_ERROR	全局目录验证失败。全局目录不可用或不支持该操作。该目录的某些部分目前尚不可用。
8418	0x000020E2	ERROR_DS_DRA_SCHEMA_MISMATCH	由于所涉及的服务器之间存在模式不匹配，复制操作失败。
8419	0x000020E3	ERROR_DS_CANT_FIND_DSA_OBJ	无法找到 DSA 对象。
8420	0x000020E4	ERROR_DS_CANT_FIND_EXPECTED_NC	无法找到命名上下文。
8421	0x000020E5	ERROR_DS_CANT_FIND_NC_IN_CACHE	在缓存中无法找到命名上下文。
8422	0x000020E6	ERROR_DS_CANT_RETRIEVE_CHILD	无法检索到子对象。
8423	0x000020E7	ERROR_DS_SECURITY_ILLEGAL_MODIFY	由于安全原因，不允许修改。
8424	0x000020E8	ERROR_DS_CANT_REPLACE_HIDDEN_REC	该操作不能取代隐藏的记录。
8425	0x000020E9	ERROR_DS_BAD_HIERARCHY_FILE	层次结构文件无效。
8426	0x000020EA	ERROR_DS_BUILD_HIERARCHY_TABLE_FAILED	尝试建立层次表失败。
8427	0x000020EB	ERROR_DS_CONFIG_PARAM_MISSING	注册表中缺少目录配置参数。
8428	0x000020EC	ERROR_DS_COUNTING_AB_INDICES_FAILED	尝试计算地址簿索引失败。
8429	0x000020ED	ERROR_DS_HIERARCHY_TABLE_MALLOC_FAILED	层次结构表的分配失败。
8430	0x000020EE	ERROR_DS_INTERNAL_FAILURE	目录服务遇到了内部故障。
8431	0x000020EF	ERROR_DS_UNKNOWN_ERROR	目录服务遇到了未知故障。
8432	0x000020F0	ERROR_DS_ROOT_REQUIRES_CLASS_TOP	根对象需要一个‘top’类别。
8433	0x000020F1	ERROR_DS_REFUSING_FSMO_ROLES	这个目录服务器正在关闭，不能接受新的浮动单主操作角色的所有权。
8434	0x000020F2	ERROR_DS_MISSING_FSMO_SETTINGS	目录服务缺少强制性配置信息，无法确定浮动单主操作角色的所有权。
8435	0x000020F3	ERROR_DS_UNABLE_TO_SURRENDER_ROLES	目录服务无法将一个或多个浮动单主操作角色的所有权转移给其他服务器。
8436	0x000020F4	ERROR_DS_DRA_GENERIC	复制操作失败。
8437	0x000020F5	ERROR_DS_DRA_INVALID_PARAMETER	为这个复制操作指定了一个无效的参数。
8438	0x000020F6	ERROR_DS_DRA_BUSY	目录服务太忙，无法在这个时候完成复制操作。
8439	0x000020F7	ERROR_DS_DRA_BAD_DN	为该复制操作指定的区分名称无效。
8440	0x000020F8	ERROR_DS_DRA_BAD_NC	为该复制操作指定的命名上下文无效。
8441	0x000020F9	ERROR_DS_DRA_DN_EXISTS	为该复制操作指定的区分名称已经存在。
8442	0x000020FA	ERROR_DS_DRA_INTERNAL_ERROR	复制系统遇到了内部错误。
8443	0x000020FB	ERROR_DS_DRA_INCONSISTENT_DIT	复制操作遇到了数据库不一致的情况。
8444	0x000020FC	ERROR_DS_DRA_CONNECTION_FAILED	无法联系到为该复制操作指定的服务器。
8445	0x000020FD	ERROR_DS_DRA_BAD_INSTANCE_TYPE	复制操作遇到了一个实例类型无效的对象。
8446	0x000020FE	ERROR_DS_DRA_OUT_OF_MEM	复制操作未能分配内存。
8447	0x000020FF	ERROR_DS_DRA_MAIL_PROBLEM	复制操作在邮件系统中遇到了一个错误。
8448	0x00002100	ERROR_DS_DRA_REF_ALREADY_EXISTS	目标服务器的复制参考信息已经存在。
8449	0x00002101	ERROR_DS_DRA_REF_NOT_FOUND	目标服务器的复制参考信息不存在。
8450	0x00002102	ERROR_DS_DRA_OBJ_IS_REP_SOURCE	命名上下文无法删除，因为它被复制到了另一个服务器上。
8451	0x00002103	ERROR_DS_DRA_DB_ERROR	复制操作遇到了数据库错误。
8452	0x00002104	ERROR_DS_DRA_NO_REPLICA	命名上下文正在被删除或没有从指定的服务器上复制。
8453	0x00002105	ERROR_DS_DRA_ACCESS_DENIED	复制访问被拒绝。
8454	0x00002106	ERROR_DS_DRA_NOT_SUPPORTED	要求的操作不被这个版本的目录服务所支持。
8455	0x00002107	ERROR_DS_DRA_RPC_CANCELLED	复制远程过程调用已取消。
8456	0x00002108	ERROR_DS_DRA_SOURCE_DISABLED	源服务器目前正在拒绝复制请求。
8457	0x00002109	ERROR_DS_DRA_SINK_DISABLED	目标服务器目前正在拒绝复制请求。
8458	0x0000210A	ERROR_DS_DRA_NAME_COLLISION	复制操作由于对象名称冲突而失败。
8459	0x0000210B	ERROR_DS_DRA_SOURCE_REINSTALLED	复制源已被重新安装。
8460	0x0000210C	ERROR_DS_DRA_MISSING_PARENT	复制操作失败，因为缺少一个必要的父对象。
8461	0x0000210D	ERROR_DS_DRA_PREEMPTED	复制操作被抢占。
8462	0x0000210E	ERROR_DS_DRA_ABANDON_SYNC	由于缺乏更新，复制同步的尝试被放弃。
8463	0x0000210F	ERROR_DS_DRA_SHUTDOWN	复制操作被终止，因为系统正在关闭。
8464	0x00002110	ERROR_DS_DRA_INCOMPATIBLE_PARTIAL_SET	复制同步尝试失败，因为目标部分属性集不是源部分属性集的一个子集。

错误			描述
十进制	十六进制	名称	
8465	0x00002111	ERROR_DS_DRA_SOURCE_IS_PARTIAL_REPLICA	复制同步尝试失败，因为一个主副本试图从一个部分副本同步。
8466	0x00002112	ERROR_DS_DRA_EXTN_CONNECTION_FAILED	已联系为这个复制操作指定的服务器，但该服务器无法联系到完成操作所需的另外一个服务器。
8467	0x00002113	ERROR_DS_INSTALL_SCHEMA_MISMATCH	源林的活动目录模式版本与此计算机上的活动目录版本不兼容。必须在源林中的域控制器上升级操作系统，然后才能将这台计算机作为域控制器添加到该林中。
8468	0x00002114	ERROR_DS_DUP_LINK_ID	模式更新失败：已存在具有相同链接标识符的属性。
8469	0x00002115	ERROR_DS_NAME_ERROR_RESOLVING	名称翻译：常见处理错误。
8470	0x00002116	ERROR_DS_NAME_ERROR_NOT_FOUND	名称翻译：找不到名称，或权限不够，无法看到名称。
8471	0x00002117	ERROR_DS_NAME_ERROR_NOT_UNIQUE	名称翻译：输入名称被映射到一个以上的输出名称。
8472	0x00002118	ERROR_DS_NAME_ERROR_NO_MAPPING	名称翻译：找到了输入名称，但没有找到相关的输出格式。
8473	0x00002119	ERROR_DS_NAME_ERROR_DOMAIN_ONLY	名称翻译：无法完全解析，只找到了域名。
8474	0x0000211A	ERROR_DS_NAME_ERROR_NO_SYNTACTICAL_MAPPING	名称翻译：不接到线上，无法在客户端执行纯粹的语法映射。
8475	0x0000211B	ERROR_DS_CONSTRUCTED_ATT_MOD	不允许修改已建 att。
8476	0x0000211C	ERROR_DS_WRONG_OM_OBJ_CLASS	对于具有指定语法的属性，指定 OM-Object-Class 不正确。
8477	0x0000211D	ERROR_DS_DRA_REPL_PENDING	复制请求已经发布，等待回复。
8478	0x0000211E	ERROR_DS_DS_REQUIRED	所请求操作需要目录服务，但没有可用目录服务。
8479	0x0000211F	ERROR_DS_INVALID_LDAP_DISPLAY_NAME	类别或属性的 LDAP 显示名称包含非 ASCII 字符。
8480	0x00002120	ERROR_DS_NON_BASE_SEARCH	所要求的搜索操作仅支持基础搜索。
8481	0x00002121	ERROR_DS_CANT_RETRIEVE_ATTS	搜索未能从数据库中检索到属性。
8482	0x00002122	ERROR_DS_BACKLINK_WITHOUT_LINK	模式更新操作试图添加没有相应前向链接的后向链接属性。
8483	0x00002123	ERROR_DS_EPOCH_MISMATCH	跨域移动的来源和目标在对象日期上不一致。来源或目标没有该对象的最新版本。
8484	0x00002124	ERROR_DS_SRC_NAME_MISMATCH	跨域移动的来源或目标在对象的当前名称上不一致。来源或目标没有该对象的最新版本。
8485	0x00002125	ERROR_DS_SRC_AND_DST_NC_IDENTICAL	跨域移动操作的来源和目标相同。调用程序应该使用本地移动操作，而不是跨域移动操作。
8486	0x00002126	ERROR_DS_DST_NC_MISMATCH	跨域移动的来源和目标在林中的命名上下文上不一致。来源或目标没有最新版本的分区容器。
8487	0x00002127	ERROR_DS_NOT_AUTHORITIVE_FOR_DST_NC	跨域移动的目标不是目标命名上下文的权威。
8488	0x00002128	ERROR_DS_SRC_GUID_MISMATCH	跨域移动的来源和目标在源对象的身份上不一致。来源或目标没有来源对象的最新版本。
8489	0x00002129	ERROR_DS_CANT_MOVE_DELETED_OBJECT	跨域移动的对象已被目标服务器删除。来源服务器没有来源对象的最新版本。
8490	0x0000212A	ERROR_DS_PDC_OPERATION_IN_PROGRESS	另一项需要独占访问 PDC PSMO 的操作已经在进行中。
8491	0x0000212B	ERROR_DS_CROSS_DOMAIN_CLEANUP_REQD	跨域移动操作失败，导致移动对象存在两个版本 - 一个在来源域，一个在目标域。目标对象需要删除，以将系统恢复到一致状态。
8492	0x0000212C	ERROR_DS_ILLEGAL_XDOM_MOVE_OPERATION	此对象可能不会跨域边界移动，因为不允许此类的跨域移动，或者该对象具有某些特殊特征，例如：信任帐户或受限 RID，这会阻止其移动。
8493	0x0000212D	ERROR_DS_CANT_WITH_ACCT_GROUP_MEMBERSHPS	无法跨域移动有成员资格的对象，因为一旦移动，将违反账户组的成员条件。从任何账户组成员中删除该对象并重试。
8494	0x0000212E	ERROR_DS_NC_MUST_HAVE_NC_PARENT	命名上下文标题必须是另一个命名上下文标题的直接子标题，而不是内部节点的子标题。
8495	0x0000212F	ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE	该目录无法验证提议的命名上下文名称，因为它没有持有提议的命名上下文之上的命名上下文副本。请确保域名命名主站的角色由配置为全局目录服务器的服务器担任，并且该服务器的复制伙伴是最新的。（仅适用于 Windows 2000 域名命名主机）

错误			描述
十进制	十六进制	名称	
8496	0x00002130	ERROR_DS_DST_DOMAIN_NOT_NATIVE	目标域必须处于本地模式。
8497	0x00002131	ERROR_DS_MISSING_INFRASTRUCTURE_CONTAINER	因为服务器在指定域中没有基础设施容器，所以无法执行该操作。
8498	0x00002132	ERROR_DS_CANT_MOVE_ACCOUNT_GROUP	不允许非空账户组的跨域移动。
8499	0x00002133	ERROR_DS_CANT_MOVE_RESOURCE_GROUP	不允许跨域移动非空的资源组。

错误			描述
十进制	十六进制	名称	
8500	0x00002134	ERROR_DS_INVALID_SEARCH_FLAG	该属性的搜索标志无效。ANR 位只对 Unicode 或 Teletex 字符串的属性有效。
8501	0x00002135	ERROR_DS_NO_TREE_DELETE_ABOVE_NC	不允许从有 NC 头作为子体的对象开始删除树。
8502	0x00002136	ERROR_DS_COULDNT_LOCK_TREE_FOR_DELETE	目录服务在准备删除树时未能锁定该树，因为该树正在使用中。
8503	0x00002137	ERROR_DS_COULDNT_IDENTIFY_OBJECTS_FOR_TREE_DELETE	目录服务在试图删除树时，未能识别要删除的对象列表。
8504	0x00002138	ERROR_DS_SAM_INIT_FAILURE	安全账户管理器初始化失败，因为存在以下错误：%1。错误状态：0x%2。单击“确定”关闭系统并重新启动到目录服务还原模式。检查事件日志以了解详细信息。
8505	0x00002139	ERROR_DS_SENSITIVE_GROUP_VIOLATION	仅管理员可以修改管理组的成员列表。
8506	0x0000213A	ERROR_DS_CANT_MOD_PRIMARYGROUPID	无法改变域控制器帐户的主组 ID。
8507	0x0000213B	ERROR_DS_ILLEGAL_BASE_SCHEMA_MOD	试图修改基本模式。
8508	0x0000213C	ERROR_DS_NONSAFE_SCHEMA_CHANGE	不允许向现有的类别添加新的强制属性，从现有的类别中删除强制属性，或者向特殊类别 Top 添加不是反向链接属性的可选属性（直接或通过继承，例如通过添加或删除附属类别）。
8509	0x0000213D	ERROR_DS_SCHEMA_UPDATE_DISALLOWED	在这个 DC 上不允许模式更新，因为这个 DC 不是模式 FSMO 的角色所有者。
8510	0x0000213E	ERROR_DS_CANT_CREATE_UNDER_SCHEMA	该类别的对象不能在模式容器下创建。在模式容器下，只能创建属性-模式和类别-模式对象。
8511	0x0000213F	ERROR_DS_INSTALL_NO_SRC_SCH_VERSION	复制/子代安装未能在源 DC 的模式容器上获得 objectVersion 属性。模式容器上缺少该属性，或提供的凭证没有读取该属性的权限。
8512	0x00002140	ERROR_DS_INSTALL_NO_SCH_VERSION_IN_INIFILE	复制/子系统安装时，未能读取 system32 目录下 schema.ini 文件中 SCHEMA 部分的 objectVersion 属性。
8513	0x00002141	ERROR_DS_INVALID_GROUP_TYPE	指定的组类型无效。
8514	0x00002142	ERROR_DS_NO_NEST_GLOBALGROUP_IN_MIXEDDOMAIN	如果组是安全启用的，不能在混合域中嵌套全局组。
8515	0x00002143	ERROR_DS_NO_NEST_LOCALGROUP_IN_MIXEDDOMAIN	如果组是安全启用的，不能在混合域中嵌套本地组。
8516	0x00002144	ERROR_DS_GLOBAL_CANT_HAVE_LOCAL_MEMBER	全局组不能以本地组为成员。
8517	0x00002145	ERROR_DS_GLOBAL_CANT_HAVE_UNIVERSAL_MEMBER	全局组不能以通用组为成员。
8518	0x00002146	ERROR_DS_UNIVERSAL_CANT_HAVE_LOCAL_MEMBER	通用组不能以本地组为成员。
8519	0x00002147	ERROR_DS_GLOBAL_CANT_HAVE_CROSSDOMAIN_MEMBER	全局组不能有跨域成员。
8520	0x00002148	ERROR_DS_LOCAL_CANT_HAVE_CROSSDOMAIN_LOCAL_MEMBER	本地组不能将另一个跨域的本地组作为成员。
8521	0x00002149	ERROR_DS_HAVE_PRIMARY_MEMBERS	包含主要成员的组不能改变为安全禁用的组。
8522	0x0000214A	ERROR_DS_STRING_SD_CONVERSION_FAILED	模式缓存加载未能转换类别-模式对象上的字符串默认 SD。
8523	0x0000214B	ERROR_DS_NAMING_MASTER_GC	只有被配置为全局目录服务器的 DSA 才应该被允许持有域名命名主机 FSMO 角色。（仅适用于 Windows 2000 服务器）
8524	0x0000214C	ERROR_DS_LOOKUP_FAILURE	由于 DNS 查询失败，DSA 操作无法继续。
8525	0x0000214D	ERROR_DS_COULDNT_UPDATE_SPNS	在处理一个对象的 DNS 主机名称的变化时，服务主名称的值不能保持同步。
8526	0x0000214E	ERROR_DS_CANT_RETRIEVE_SD	安全描述符属性无法读取。

错误			描述
十进制	十六进制	名称	
8527	0x0000214F	ERROR_DS_KEY_NOT_UNIQUE.	请求的对象没有找到，但找到了一个具有该键的对象。
8528	0x00002150	ERROR_DS_WRONG_LINKED_ATT_SYNTAX	正在添加的链接属性的语法不正确。正向链接只能有 2.5.5.1、2.5.5.7 和 2.5.5.14 语法，而反向链接只能有 2.5.5.1 语法。
8529	0x00002151	ERROR_DS_SAM_NEED_BOOTKEY_PASSWORD	安全账户管理器需要获得启动密码。
8530	0x00002152	ERROR_DS_SAM_NEED_BOOTKEY_FLOPPY	安全账户管理器需要从软盘中获取启动密钥。
8531	0x00002153	ERROR_DS_CANT_START	目录服务无法启动。
8532	0x00002154	ERROR_DS_INIT_FAILURE	目录服务无法启动。
8533	0x00002155	ERROR_DS_NO_PKT_PRIVACY_ON_CONNECTION	客户端和服务器之间的连接需要数据包保密性或更好。
8534	0x00002156	ERROR_DS_SOURCE_DOMAIN_IN_FOREST	源域可能与目标不在同一林中。
8535	0x00002157	ERROR_DS_DESTINATION_DOMAIN_NOT_IN_FOREST	目标域必须在林中。
8536	0x00002158	ERROR_DS_DESTINATION_AUDITING_NOT_ENABLED	该操作要求启用目标域审计。
8537	0x00002159	ERROR_DS_CANT_FIND_DC_FOR_SRC_DOMAIN	该操作无法为源域找到 DC。
8538	0x0000215A	ERROR_DS_SRC_OBJ_NOT_GROUP_OR_USER	源对象必须是一个组或用户。
8539	0x0000215B	ERROR_DS_SRC_SID_EXISTS_IN_FOREST	源对象的 SID 已经存在于目标林中。
8540	0x0000215C	ERROR_DS_SRC_AND_DST_OBJECT_CLASS_MISMATCH	源对象和目标对象必须是同一类型。
8541	0x0000215D	ERROR_SAM_INIT_FAILURE	安全账户管理器初始化失败，因为存在以下错误：%1。错误状态：0x%2。点击“确定”关闭系统并重新启动到安全模式。检查事件日志以了解详细信息。
8542	0x0000215E	ERROR_DS_DRA_SCHEMA_INFO_SHIP	模式信息无法包含在复制请求中。
8543	0x0000215F	ERROR_DS_DRA_SCHEMA_CONFLICT	由于模式不兼容，复制操作无法完成。
8544	0x00002160	ERROR_DS_DRA_EARLIER_SCHEMA_CONLICT	由于之前的模式不兼容，复制操作无法完成。
8545	0x00002161	ERROR_DS_DRA_OBJ_NC_MISMATCH	复制更新无法应用，因为源和目标还没有收到关于最近跨域移动操作的信息。
8546	0x00002162	ERROR_DS_NC_STILL_HAS_DSAS	请求的域无法删除，因为存在着仍然托管这个域的域控制器。
8547	0x00002163	ERROR_DS_GC_REQUIRED	所要求的操作只能在全局目录服务器上执行。
8548	0x00002164	ERROR_DS_LOCAL_MEMBER_OF_LOCAL_ONLY	本地组只能是同一域中其他本地组的成员。
8549	0x00002165	ERROR_DS_NO_FPO_IN_UNIVERSAL_GROUPS	外部安全主要成员不能是通用组的成员。
8550	0x00002166	ERROR_DS_CANT_ADD_TO_GC	由于安全原因，该属性不允许被复制到 GC。
8551	0x00002167	ERROR_DS_NO_CHECKPOINT_WITH_PDC	无法采取 PDC 的检查点，因为目前有太多的修改正在处理。
8552	0x00002168	ERROR_DS_SOURCE_AUDITING_NOT_ENABLED	该操作要求启用源域审计。
8553	0x00002169	ERROR_DS_CANT_CREATE_IN_NONDOMAIN_NC	安全主体对象只能在域命名上下文中创建。
8554	0x0000216A	ERROR_DS_INVALID_NAME_FOR_SPN	由于提供的主机名不符合必要的格式，因此无法构建服务主名称（SPN）。
8555	0x0000216B	ERROR_DS_FILTER_USES_CONTRUCTED_ATTRS	传递了一个使用构造属性的过滤器。
8556	0x0000216C	ERROR_DS_UNICODEPWD_NOT_IN_QUOTES	unicodePwd 属性值必须用双引号括起来。
8557	0x0000216D	ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED	你的计算机无法加入到域中。超过了这个域中允许创建的最大计算机账户数量。请联系系统管理员，以重新设置或增加这一限制。
8558	0x0000216E	ERROR_DS_MUST_BE_RUN_ON_DST_DC	出于安全原因，该操作必须在目标 DC 上运行。
8559	0x0000216F	ERROR_DS_SRC_DC_MUST_BE_SP4_OR_GREATER	出于安全考虑，源 DC 必须是 NT4SP4 或更高版本。
8560	0x00002170	ERROR_DS_CANT_TREE_DELETE_CRITICAL_OBJ	在树删除操作中，无法删除关键目录服务系统对象。树的删除可能部分完成。
8561	0x00002171	ERROR_DS_INIT_FAILURE_CONSOLE	目录服务无法启动，因为存在以下错误：%1。错误状态：0x%2。请点击“确定”关闭系统。你可以使用恢复控制台来进一步诊断系统。
8562	0x00002172	ERROR_DS_SAM_INIT_FAILURE_CONSOLE	安全账户管理器初始化失败，因为存在以下错误：%1。错误状态：0x%2。请点击“确定”关闭系统。你可以使用恢复控制台来进一步诊断系统。

错误			描述
十进制	十六进制	名称	
8563	0x00002173	ERROR_DS_FOREST_VERSION_TOO_HIGH	Windows 版本太老，无法支持当前的目录林行为。必须在这台服务器上升级操作系统，然后它才能成为这个林中的域控制器。
8564	0x00002174	ERROR_DS_DOMAIN_VERSION_TOO_HIGH	Windows 版本太老，无法支持当前的域行为。必须在这台服务器上升级操作系统，然后才能成为这个域中的域控制器。
8565	0x00002175	ERROR_DS_FOREST_VERSION_TOO_LOW	这个版本的 Windows 不再支持这个目录林中使用的行为版本。在这个服务器成为林中的域控制器之前，必须升级森林行为版本。
8566	0x00002176	ERROR_DS_DOMAIN_VERSION_TOO_LOW	这个版本的 Windows 不再支持这个域中使用的行为版本。在该服务器成为域中的域控制器之前，必须升级域行为版本。
8567	0x00002177	ERROR_DS_INCOMPATIBLE_VERSION	Windows 的版本与域或林的行为版本不兼容。
8568	0x00002178	ERROR_DS_LOW_DSA_VERSION	行为版本不能增加到请求值，因为域控制器仍然存在版本低于请求值的情况。
8569	0x00002179	ERROR_DS_NO_BEHAVIOR_VERSION_IN_MIXEDDOMAIN	当域仍处于混合域模式时，行为版本值不能增加。在增加行为版本之前，必须首先将域改为本地模式。
8570	0x0000217A	ERROR_DS_NOT_SUPPORTED_SORT_ORDER	不支持所要求的排序顺序。
8571	0x0000217B	ERROR_DS_NAME_NOT_UNIQUE	发现一个具有非唯一名称的对象。
8572	0x0000217C	ERROR_DS_MACHINE_ACCOUNT_CREATED_PRENT4	该机器账户在 NT4 之前创建。账户需要重新创建。
8573	0x0000217D	ERROR_DS_OUT_OF_VERSION_STORE	数据库在版本存储之外。
8574	0x0000217E	ERROR_DS_INCOMPATIBLE_CONTROLS_USED	无法继续操作，因为使用了多个冲突的控件。
8575	0x0000217F	ERROR_DS_NO_REF_DOMAIN	无法找到此分区的有效安全描述符参考域。
8576	0x00002180	ERROR_DS_RESERVED_LINK_ID	模式更新失败：链接标识符被保留。
8577	0x00002181	ERROR_DS_LINK_ID_NOT_AVAILABLE	模式更新失败：没有可用的链接标识符。
8578	0x00002182	ERROR_DS_AG_CANT_HAVE_UNIVERSAL_MEMBER	账户组不能以通用组为成员。
8579	0x00002183	ERROR_DS_MODIFYDN_DISALLOWED_BY_INSTANCE_TYPE	不允许对命名上下文头或只读对象进行重命名或移动操作。
8580	0x00002184	ERROR_DS_NO_OBJECT_MOVE_IN_SCHEMA_NC	不允许对模式命名上下文中的对象进行移动操作。
8581	0x00002185	ERROR_DS_MODIFYDN_DISALLOWED_BY_FLAG	在对象上设置了一个系统标志，不允许对象被移动或重命名。
8582	0x00002186	ERROR_DS_MODIFYDN_WRONG_GRANDPARENT	不允许此对象改变其祖容器。在此对象上不禁止移动，但限制于移到兄弟容器。
8583	0x00002187	ERROR_DS_NAME_ERROR_TRUST_REFERRAL	无法完全解决，就会产生转介到另一个林。
8584	0x00002188	ERROR_NOT_SUPPORTED_ON_STANDARD_SERVER	标准服务器不支持所要求的操作。
8585	0x00002189	ERROR_DS_CANT_ACCESS_REMOTE_PART_OF_AD	无法访问位于远程服务器上的活动目录的一个分区。确保至少有一台服务器正在为有关分区运行。
8586	0x0000218A	ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE_V2	目录无法验证提议的命名上下文（或分区）名称，因为它不持有副本，也不能联系提议的命名上下文上面的一个副本。请确保父命名上下文已在 DNS 中正确注册，并且域命名主机至少可以访问此命名上下文的一个副本。。
8587	0x0000218B	ERROR_DS_THREAD_LIMIT_EXCEEDED	已超过请求的线程限制。
8588	0x0000218C	ERROR_DS_NOT_CLOSEST	全局目录服务器不在最近站点中。

错误			描述
十进制	十六进制	名称	
9001	0x00002329	DNS_ERROR_RCODE_FORMAT_ERROR	DNS服务器无法解释格式。
9002	0x0000232A	DNS_ERROR_RCODE_SERVER_FAILURE	DNS 服务器故障。
9003	0x0000232B	DNS_ERROR_RCODE_NAME_ERROR	DNS 名称不存在。
9004	0x0000232C	DNS_ERROR_RCODE_NOT_IMPLEMENTED	名称服务器不支持 DNS 请求。
9005	0x0000232D	DNS_ERROR_RCODE_REFUSED	拒绝 DNS 操作。
9006	0x0000232E	DNS_ERROR_RCODE_YXDOMAIN	本应不存在的 DNS 名称仍然存在。
9007	0x0000232F	DNS_ERROR_RCODE_YXRRSET	不应该存在的 DNS RR 集仍然存在。
9008	0x00002330	DNS_ERROR_RCODE_NXRRSET	应该存在的 DNS RR 集不存在。

错误			描述
十进制	十六进制	名称	
9009	0x00002331	DNS_ERROR_RCODE_NOTAUTH	DNS 服务器对区域没有权威。
9010	0x00002332	DNS_ERROR_RCODE_NOTZONE	更新或先决条件中的 DNS 名称不在区域内。
9016	0x00002338	DNS_ERROR_RCODE_BADSIG	DNS 签名验证失败。
9017	0x00002339	DNS_ERROR_RCODE_BADKEY	DNS 不正确密钥。
9018	0x0000233A	DNS_ERROR_RCODE_BADTIME	DNS 签名有效期过期。
9501	0x0000251D	DNS_INFO_NO_RECORDS	未发现给定的 DNS 查询记录。
9502	0x0000251E	DNS_ERROR_BAD_PACKET	DNS 包不正确。
9503	0x0000251F	DNS_ERROR_NO_PACKET	没有 DNS 包。
9504	0x00002520	DNS_ERROR_RCODE	DNS 错误，检查 rcode。
9505	0x00002521	DNS_ERROR_UNSECURE_PACKET	不安全的 DNS 包。
9551	0x0000254F	DNS_ERROR_INVALID_TYPE	无效 DNS 类型。
9552	0x00002550	DNS_ERROR_INVALID_IP_ADDRESS	无效的 IP 地址。
9553	0x00002551	DNS_ERROR_INVALID_PROPERTY	无效属性。
9554	0x00002552	DNS_ERROR_TRY_AGAIN_LATER	稍后再尝试 DNS 操作。
9555	0x00002553	DNS_ERROR_NOT_UNIQUE	给定名称和类型的记录不唯一。
9556	0x00002554	DNS_ERROR_NON_RFC_NAME	DNS 名称不符合 RFC 规范。
9557	0x00002555	DNS_STATUS_FQDN	DNS 名称是一个完全限定的 DNS 名称。
9558	0x00002556	DNS_STATUS_DOTTED_NAME	DNS 名称以点分隔（多标签）。
9559	0x00002557	DNS_STATUS_SINGLE_PART_NAME	DNS 名称是一个单部分的名称。
9560	0x00002558	DNS_ERROR_INVALID_NAME_CHAR	DSN 名称包含一个无效的字符。
9561	0x00002559	DNS_ERROR_NUMERIC_NAME	DNS 名称完全是数字。
9601	0x00002581	DNS_ERROR_ZONE_DOES_NOT_EXIST	DNS 区域不存在。
9602	0x00002582	DNS_ERROR_NO_ZONE_INFO	DNS 区域信息不可用。
9603	0x00002583	DNS_ERROR_INVALID_ZONE_OPERATION	对 DNS 区域的操作无效。
9604	0x00002584	DNS_ERROR_ZONE_CONFIGURATION_ERROR	无效的 DNS 区域配置。
9605	0x00002585	DNS_ERROR_ZONE_HAS_NO_SOA_RECORD	DNS 区域没有授权开始（SOA）记录。
9606	0x00002586	DNS_ERROR_ZONE_HAS_NO_NS_RECORDS	DNS 区域没有名称服务器（NS）记录。
9607	0x00002587	DNS_ERROR_ZONE_LOCKED	DNS 区域已锁定。
9608	0x00002588	DNS_ERROR_ZONE_CREATION_FAILED	DNS 区域创建失败。
9609	0x00002589	DNS_ERROR_ZONE_ALREADY_EXISTS	DNS 区域已存在。
9610	0x0000258A	DNS_ERROR_AUTOZONE_ALREADY_EXISTS	DNS 自动区域已存在。
9611	0x0000258B	DNS_ERROR_INVALID_ZONE_TYPE	无效的 DNS 区域类型。
9612	0x0000258C	DNS_ERROR_SECONDARY_REQUIRES_MASTER_IP	二级 DNS 区域需要主 IP 地址。
9613	0x0000258D	DNS_ERROR_ZONE_NOT_SECONDARY	DNS 区域不是二级。
9614	0x0000258E	DNS_ERROR_NEED_SECONDARY_ADDRESSES	需要次要 IP 地址。
9615	0x0000258F	DNS_ERROR_WINS_INIT_FAILED	WINS 初始化失败。
9616	0x00002590	DNS_ERROR_NEED_WINS_SERVERS	需要 WINS 服务器。
9617	0x00002591	DNS_ERROR_NBSTAT_INIT_FAILED	NBTSTAT 初始化调用失败。
9618	0x00002592	DNS_ERROR_SOA_DELETE_INVALID	无效删除授权开始（SOA）。
9619	0x00002593	DNS_ERROR_FORWARDER_ALREADY_EXISTS	该名称已经存在一个有条件的转发区。
9651	0x000025B3	DNS_ERROR_PRIMARY_REQUIRES_DATAFILE	主要 DNS 区域需要数据文件。
9652	0x000025B4	DNS_ERROR_INVALID_DATAFILE_NAME	DNS 区域的数据文件名称无效。
9653	0x000025B5	DNS_ERROR_DATAFILE_OPEN_FAILURE	打开 DNS 区域的数据文件失败。
9654	0x000025B6	DNS_ERROR_FILE_WRITEBACK_FAILED	写入 DNS 区域的数据文件失败。
9655	0x000025B7	DNS_ERROR_DATAFILE_PARSING	读取 DNS 区域的数据文件时出现故障。
9701	0x000025E5	DNS_ERROR_RECORD_DOES_NOT_EXIST	DNS 记录不存在。
9702	0x000025E6	DNS_ERROR_RECORD_FORMAT	DNS 记录格式错误。
9703	0x000025E7	DNS_ERROR_NODE_CREATION_FAILED	DNS 中的节点创建失败。
9704	0x000025E8	DNS_ERROR_UNKNOWN_RECORD_TYPE	未知的 DNS 记录类型。
9705	0x000025E9	DNS_ERROR_RECORD_TIMED_OUT	DNS 记录超时。
9706	0x000025EA	DNS_ERROR_NAME_NOT_IN_ZONE	名称不在 DNS 区域。
9707	0x000025EB	DNS_ERROR_CNAME_LOOP	检测到 CNAME 循环。
9708	0x000025EC	DNS_ERROR_NODE_IS_CNAME	Node 是一个 CNAME 的 DNS 记录。

错误			描述
十进制	十六进制	名称	
9709	0x000025ED	DNS_ERROR_CNAME_COLLISION	对于指定名称，已存在 CNAME 记录。
9710	0x000025EE	DNS_ERROR_RECORD_ONLY_AT_ZONE_ROOT	只在 DNS 区域根部记录。
9711	0x000025EF	DNS_ERROR_RECORD_ALREADY_EXISTS	DNS 记录已存在。
9712	0x000025F0	DNS_ERROR_SECONDARY_DATA	二级 DNS 区域数据错误。
9713	0x000025F1	DNS_ERROR_NO_CREATE_CACHE_DATA	无法创建 DNS 缓存数据。
9714	0x000025F2	DNS_ERROR_NAME_DOES_NOT_EXIST	DNS 名称不存在。
9715	0x000025F3	DNS_WARNING_PTR_CREATE_FAILED	无法创建指针（PTR）记录。
9716	0x000025F4	DNS_WARNING_DOMAIN_UNDELETED	DNS 域未删除。
9717	0x000025F5	DNS_ERROR_DS_UNAVAILABLE	目录服务不可用。
9718	0x000025F6	DNS_ERROR_DS_ZONE_ALREADY_EXISTS	DNS 区域已经存在于目录服务中。
9719	0x000025F7	DNS_ERROR_NO_BOOTFILE_IF_DS_ZONE	DNS 服务器未创建或读取目录服务集成 DNS 区域的启动文件。
9751	0x00002617	DNS_INFO_AXFR_COMPLETE	DNS AXFR（区域转移）完成。
9752	0x00002618	DNS_ERROR_AXFR	DNS区域传输失败。
9753	0x00002619	DNS_INFO_ADDED_LOCAL_WINS	增加了本地 WINS 服务器。
9801	0x00002649	DNS_STATUS_CONTINUE_NEEDED	安全更新呼叫需要继续更新请求。
9851	0x0000267B	DNS_ERROR_NO_TCPIP	未安装 TCP/IP 网络协议。
9852	0x0000267C	DNS_ERROR_NO_DNS_SERVERS	没有为本地系统配置 DNS 服务器。
9901	0x000026AD	DNS_ERROR_DP_DOES_NOT_EXIST	指定的目录分区不存在。
9902	0x000026AE	DNS_ERROR_DP_ALREADY_EXISTS	指定的目录分区已存在。
9903	0x000026AF	DNS_ERROR_DP_NOT_ENLISTED	DS 未列在指定的目录分区中。
9904	0x000026B0	DNS_ERROR_DP_ALREADY_ENLISTED	DS 已列在指定的目录分区中。

错误			描述
十进制	十六进制	名称	
10004	0x00002714	WSAEINTR	一个阻塞操作被一个对 WSACancelBlockingCall 的调用所打断。
10009	0x00002719	WSAEBADF	提供的文件句柄无效。
10013	0x0000271D	WSAEACCES	试图以其访问权限所禁止的方式访问一个套接字。
10014	0x0000271E	WSAEFAULT	系统在试图在调用中使用一个指针参数时检测到一个无效的指针地址。
10022	0x00002726	WSAEINVAL	提供了一个无效的参数。
10024	0x00002728	WSAEMFILE	打开的套接字太多。
10035	0x00002733	WSAEWOULDBLOCK	无法立即完成非阻塞的套接字操作。
10036	0x00002734	WSAEINPROGRESS	目前正在执行一个阻塞性操作。
10037	0x00002735	WSAEALREADY	在一个已经有操作在进行的非阻塞套接字上尝试了一个操作。
10038	0x00002736	WSAENOTSOCK	在非套接字上尝试了一个操作。
10039	0x00002737	WSAEDESTADDRREQ	在套接字的操作中遗漏了一个必要的地址。
10040	0x00002738	WSAEMSGSIZE	在数据报套接字上发送的信息大于内部信息缓冲区或其他一些网络限制，或者用于接收数据报的缓冲区小于数据报本身。
10041	0x00002739	WSAEPROTOTYPE	在套接字函数调用中指定了一个协议，该协议不支持所要求的套接字类型的语义。
10042	0x0000273A	WSAENOPROTOOPT	在 getsockopt 或 setockopt 调用中指定了一个未知的、无效的或不支持的选项或级别。
10043	0x0000273B	WSAEPROTONOSUPPORT	所请求的协议没有被配置到系统中，或者不存在对它的实现。
10044	0x0000273C	WSAESOCKTNOSUPPORT	该地址族中不存在对指定套接字类型的支持。
10045	0x0000273D	WSAEOPNOTSUPP	对于所引用的对象类型，尝试的操作不受支持。
10046	0x0000273E	WSAEPFNOSUPPORT	该协议系列没有被配置到系统中，或者不存在该协议的实现。
10047	0x0000273F	WSAEAFNOSUPPORT	使用了一个与请求的协议不兼容的地址。
10048	0x00002740	WSAEADDRINUSE	每个套接字地址（协议/网络地址/端口）通常只允许使用一次。
10049	0x00002741	WSAEADDRNOTAVAIL	请求的地址在其上下文中无效。
10050	0x00002742	WSAENETDOWN	套接字操作遇到了一个无法连接的网络。

错误			描述
十进制	十六进制	名称	
10051	0x00002743	WSAENETUNREACH	试图对一个无法连接的网络进行套接字操作。
10052	0x00002744	WSAENETRESET	由于在操作过程中，keep-alive 活动检测到故障，连接已被中断。
10053	0x00002745	WSAECONNABORTED	已建立的连接被你主机中的软件中止了。
10054	0x00002746	WSAECONNRESET	一个现有的连接被远程主机强行关闭。
10055	0x00002747	WSAENOBUFS	由于系统缺乏足够的缓冲空间或队列已满，无法在套接字上执行操作。
10056	0x00002748	WSAEISCONN	在一个已经连接的套接字上发出了一个连接请求。
10057	0x00002749	WSAENOTCONN	发送或接收数据的请求被拒绝，因为套接字没有连接，而且（当使用 sendto 调用在数据报套接字上发送时）没有提供地址。
10058	0x0000274A	WSAESHUTDOWN	由于以前的关闭调用，套接字在那个方向已经关闭，发送或接收数据的请求被禁止。
10059	0x0000274B	WSAETOOMANYREFS	对某些内核对象的引用太多。
10060	0x0000274C	WSAETIMEDOUT	连接尝试失败，因为被连接方在一段时间后没有正确回应，或者建立的连接失败，因为连接的主机没有回应。
10061	0x0000274D	WSAECONNREFUSED	因为目标机主动拒绝，所以无法进行连接。
10062	0x0000274E	WSAELOOP	无法翻译名称。
10063	0x0000274F	WSAENAMETOOLONG	名称组件或名称太长。
10064	0x00002750	WSAEHOSTDOWN	一个套接字操作失败了，因为目标主机停机了。
10065	0x00002751	WSAEHOSTUNREACH	试图对一个无法连接的主机进行套接字操作。
10066	0x00002752	WSAENOTEMPTY	无法删除一个非空的目录。
10067	0x00002753	WSAEPROCLIM	一个 Windows 套接字的实现可能对同时使用它的应用程序的数量有限制。
10068	0x00002754	WSAEUSERS	配额已用完。
10069	0x00002755	WSAEDQUOT	磁盘配额已用完。
10070	0x00002756	WSAESTALE	文件句柄参考不再可用。
10071	0x00002757	WSAEREMOTE	该项目在本地不可用。
10091	0x0000276B	WSASYSNOTREADY	WSAStartup 目前无法运行，因为它用来提供网络服务的底层系统目前不可用。
10092	0x0000276C	WSAVERNOTSUPPORTED	不支持要求的 Windows 套接字版本。
10093	0x0000276D	WSANOTINITIALISED	应用程序未调用 WSAStartup，或 WSAStartup 失败。
10101	0x00002775	WSAEDISCON	由 WSARecv 或 WSARecvFrom 返回，表示远程主机已经顺利启动了关闭序列。
10102	0x00002776	WSAENOMORE	WSALookupServiceNext 不能再返回任何结果。
10103	0x00002777	WSAECANCELLED	当这个调用仍在处理时，对 WSALookupServiceEnd 的调用已经进行。该调用已被取消。
10104	0x00002778	WSAEINVALIDPROCTABLE	过程调用表无效。
10105	0x00002779	许可证	请求的服务提供商无效。
10106	0x0000277A	WSAEPROVIDERFAILEDINIT	所请求的服务提供程序不能加载或初始化。
10107	0x0000277B	WSASYSCALLFAILURE	一个不应该失败的系统调用失败了。
10108	0x0000277C	WSASERVICE_NOT_FOUND	不知道这种服务。在指定的名称空间中找不到该服务。
10109	0x0000277D	WSATYPE_NOT_FOUND	未找到指定的类别。
10110	0x0000277E	WSA_E_NO_MORE	WSALookupServiceNext 不能再返回任何结果。
10111	0x0000277F	WSA_E_CANCELLED	当这个调用仍在处理时，对 WSALookupServiceEnd 的调用已经进行。该调用已被取消。
10112	0x00002780	WSAEREFUSED	数据库查询失败，因为它被主动拒绝。
11001	0x00002AF9	WSAHOST_NOT_FOUND	不知道这种主机。
11002	0x00002AFA	WSATRY_AGAIN	这通常是主机名解析过程中的一个临时错误，意味着本地服务器没有收到来自权威服务器的响应。
11003	0x00002AFB	WSANO_RECOVERY	在数据库查询过程中发生了一个不可恢复的错误。
11004	0x00002AFC	WSANO_DATA	请求的名称有效，并已在数据库中找到，但它没有正确的相关数据被解析。

错误			描述
十进制	十六进制	名称	
11005	0x00002AFD	WSA_QOS_RECEIVERS	至少有一个保留已经到达。
11006	0x00002AFE	wsa_qos_senders	至少到达了一个路径。
11007	0x00002AFF	WSA_QOS_NO_SENDERS	没有发送者。
11008	0x00002B00	WSA_QOS_NO_RECEIVERS	没有接收者。
11009	0x00002B01	WSA_QOS_REQUEST_CONFIRMED	储备已被确认。
11010	0x00002B02	WSA_QOS_ADMISSION_FAILURE	由于缺乏资源而出错。
11011	0x00002B03	WSA_QOS_POLICY_FAILURE	因管理原因被拒绝 - 证书不对。
11012	0x00002B04	WSA_QOS_BAD_STYLE	不明或冲突的风格。
11013	0x00002B05	WSA_QOS_BAD_OBJECT	一般来说，filterspec 的某些部分或 providerspecific 的缓冲区有问题。
11014	0x00002B06	WSA_QOS_TRAFFIC_CTRL_ERROR	flowspec 的某些部分有问题。
11015	0x00002B07	WSA_QOS_GENERIC_ERROR	一般 QOS 错误。
11016	0x00002B08	WSA_QOS_ESERVICETYPE	在 flowspec 中发现了一个无效的或未被识别的服务类型。
11017	0x00002B09	WSA_QOS_EFLOWSPEC	在 QOS 结构中发现一个无效的或不一致的 flowspec。
11018	0x00002B0A	WSA_QOS_EPROVSPECBUF	无效的 QOS 提供程序专用缓冲区。
11019	0x00002B0B	WSA_QOS_EFILTERSTYLE	使用了一个无效的 QOS 过滤器样式。
11020	0x00002B0C	WSA_QOS_EFILTERTYPE	使用了一个无效的 QOS 过滤器类型。
11021	0x00002B0D	WSA_QOS_EFILTERCOUNT	在 FLOWDESCRIPTOR 中指定的 QOS FILTERSPEC 的数量不正确。
11022	0x00002B0E	WSA_QOS_EOBJLENGTH	在 QOS 提供程序特定的缓冲区中指定了一个具有无效 ObjectLength 字段的对象。
11023	0x00002B0F	WSA_QOS_EFLOWCOUNT	在 QOS 结构中指定了一个不正确的流量描述符数量。
11024	0x00002B10	WSA_QOS_EUNKNOWNPSOBJ	在 QOS 提供程序特定的缓冲区中发现了一个未被识别的对象。
11025	0x00002B11	WSA_QOS_EPOLICYOBJ	在 QOS 提供程序特定的缓冲区中发现了一个无效的策略对象。
11026	0x00002B12	WSA_QOS_EFLOWDESC	在流量描述符列表中发现一个无效的 QOS 流量描述符。
11027	0x00002B13	WSA_QOS_EPSFLOWSPEC	在 QOS 提供程序特定的缓冲区中发现了一个无效的或不一致的 flowspec。
11028	0x00002B14	WSA_QOS_EPSFILTERSPEC	在 QOS 提供程序特定的缓冲区中发现了一个无效的 FILTERSPEC。
11029	0x00002B15	WSA_QOS_ESDMODEOBJ	在 QOS 提供程序特定的缓冲区中发现了一个无效的形状丢弃模式对象。
11030	0x00002B16	WSA_QOS_ESHAPERATEOBJ	在 QOS 提供程序特定的缓冲区中发现了一个无效的成形速率对象。
11031	0x00002B17	WSA_QOS_RESERVED_PETYPE	在 QOS 提供者特定的缓冲区中发现了一个保留的策略元素。

错误			描述
十进制	十六进制	名称	
12000	0x00002EE0	ERROR_SXS_SECTION_NOT_FOUND	要求的部分在激活环境中不存在。
12001	0x00002EE1	ERROR_SXS_CANT_GEN_ACTCTX	这个应用程序未能启动，因为应用程序的配置不正确。重新安装应用程序可能会解决这个问题。
12002	0x00002EE2	ERROR_SXS_INVALID_ACTCTXDATA_FORMAT	应用程序绑定的数据格式无效。
12003	0x00002EE3	ERROR_SXS_ASSEMBLY_NOT_FOUND	系统上未安装引用的程序集。
12004	0x00002EE4	ERROR_SXS_MANIFEST_FORMAT_ERROR	清单文件未以所需的标签和格式信息开头。
12005	0x00002EE5	ERROR_SXS_MANIFEST_PARSE_ERROR	清单文件包含一个或多个语法错误。
12006	0x00002EE6	ERROR_SXS_ACTIVATION_CONTEXT_DISABLED	该应用程序试图激活已禁用的激活上下文。
12007	0x00002EE7	ERROR_SXS_KEY_NOT_FOUND	在任何活动的激活上下文中都未找到所要求的查找密钥。
12008	0x00002EE8	ERROR_SXS_VERSION_CONFLICT	应用程序所需的一个组件版本与另一个已经激活的组件版本相冲突。
12009	0x00002EE9	ERROR_SXS_WRONG_SECTION_TYPE	要求的类型激活上下文部分的类型与使用的查询 API 不匹配。

错误			描述
十进制	十六进制	名称	
12010	0x00002EEA	ERROR_SXS_THREAD_QUERIES_DISABLED	由于缺乏系统资源，需要对当前执行线程禁用独立激活。
12011	0x00002EEB	ERROR_SXS_PROCESS_DEFAULT_ALREADY_SET	设置进程默认激活上下文的尝试失败，因为进程默认激活上下文已经被设置。
12012	0x00002EEC	ERROR_SXS_UNKNOWN_ENCODING_GROUP	无法识别指定的编码组标识符。
12013	0x00002EED	ERROR_SXS_UNKNOWN_ENCODING	无法识别请求的编码。
12014	0x00002EEE	ERROR_SXS_INVALID_XML_NAMESPACE_URI	清单包含对无效 URI 的引用。
12015	0x00002EEF	ERROR_SXS_ROOT_MANIFEST_DEPENDENCY_NOT_INSTALLED	应用程序清单包含对未安装的依赖程序集的引用。
12016	0x00002EF0	ERROR_SXS_LEAF_MANIFEST_DEPENDENCY_NOT_INSTALLED	应用程序使用的程序集的清单引用了未安装的依赖程序集。
12017	0x00002EF1	ERROR_SXS_INVALID_ASSEMBLY_IDENTITY_ATTRIBUTE	清单包含无效的程序集标识属性。
12018	0x00002EF2	ERROR_SXS_MANIFEST_MISSING_REQUIRED_DEFAULT_NAMESPACE	清单缺少程序集元素上所需的默认命名空间规范。
12019	0x00002EF3	ERROR_SXS_MANIFEST_INVALID_REQUIRED_DEFAULT_NAMESPACE	清单具有在程序集元素上指定的默认命名空间，但其值不是“urn:schemas-microsoft-com:asm.v1”。
12020	0x00002EF4	ERROR_SXS_PRIVATE_MANIFEST_CROSS_PATH_WITH_REPARSE_POINT	探测到的私有清单已越过与重分析点关联的路径。
12021	0x00002EF5	ERROR_SXS_DUPLICATE_DLL_NAME	应用程序清单直接或间接引用的两个或多个组件具有相同名称的文件。
12022	0x00002EF6	ERROR_SXS_DUPLICATE_WINDOWCLASS_NAME	应用程序清单直接或间接引用的两个或多个组件具有同名的窗口类。
12023	0x00002EF7	ERROR_SXS_DUPLICATE_CLSID	应用程序清单直接或间接引用的两个或多个组件具有相同的 COM 服务器 CLSID。
12024	0x00002EF8	ERROR_SXS_DUPLICATE_IID	应用程序清单直接或间接引用的两个或多个组件具有相同的 COM 接口 IID 的代理。
12025	0x00002EF9	ERROR_SXS_DUPLICATE_TLBID	应用程序清单直接或间接引用的两个或多个组件具有相同的 COM 类型库 TLBID。
12026	0x00002EFA	ERROR_SXS_DUPLICATE_PROGID	应用程序清单直接或间接引用的两个或多个组件具有相同的 COM ProgID。
12027	0x00002EFB	ERROR_SXS_DUPLICATE_ASSEMBLY_NAME	应用程序清单直接或间接引用的两个或多个组件是同一组件的不同版本，这是不允许的。
12028	0x00002EFC	ERROR_SXS_FILE_HASH_MISMATCH	组件的文件与组件清单中的验证信息不一致。
12029	0x00002EFD	ERROR_SXS_POLICY_PARSE_ERROR	策略清单包含一个或多个语法错误。
12030	0x00002EFE	ERROR_SXS_XML_E_MISSINGQUOTE	清单解析错误：需要字符串文本，但找不到开头引号字符。
12031	0x00002EFF	ERROR_SXS_XML_E_COMMENTSYNTAX	清单解析错误：在注释中使用了不正确的语法。
12032	0x00002F00	ERROR_SXS_XML_E_BADSTARTNAMECHAR	清单解析错误：名称以无效字符开始。
12033	0x00002F01	ERROR_SXS_XML_E_BADNAMECHAR	清单解析错误：名称包含无效字符。
12034	0x00002F02	ERROR_SXS_XML_E_BADCHARINSTRING	清单解析错误：字符串文本包含无效字符。
12035	0x00002F03	ERROR_SXS_XML_E_XMLDECLSYNTAX	清单解析错误：XML 声明的语法无效。
12036	0x00002F04	ERROR_SXS_XML_E_BADCHARDATA	清单解析错误：在文本内容中发现无效的字符。
12037	0x00002F05	ERROR_SXS_XML_E_MISSINGWHITESPACE	清单解析错误：缺少所需的空白。
12038	0x00002F06	ERROR_SXS_XML_E_EXPECTINGTAGEND	清单解析错误：应为字符 ‘>’ 。
12039	0x00002F07	ERROR_SXS_XML_E_MISSINGSEMICOLON	清单解析错误：应为分号字符。
12040	0x00002F08	ERROR_SXS_XML_E_UNBALANCEDPAREN	清单解析错误：括号不平衡。
12041	0x00002F09	ERROR_SXS_XML_E_INTERNALERROR	清单解析错误：内部错误。
12042	0x00002F0A	ERROR_SXS_XML_E_UNEXPECTED_WHITESPACE	清单解析错误：此位置不允许空白。
12043	0x00002F0B	ERROR_SXS_XML_E_INCOMPLETE_ENCODING	清单解析错误：当前编码处于无效状态时到达文件结尾。
12044	0x00002F0C	ERROR_SXS_XML_E_MISSING_PAREN	清单解析错误：缺少括号。
12045	0x00002F0D	ERROR_SXS_XML_E_EXPECTINGCLOSEQUOTE	清单解析错误：缺少单引号或双引号字符（\'或\"）。
12046	0x00002F0E	ERROR_SXS_XML_E_MULTIPLE_COLONS	清单解析错误：名称中不允许有多个冒号。
12047	0x00002F0F	ERROR_SXS_XML_E_INVALID_DECIMAL	清单解析错误：十进制数字的字符无效。
12048	0x00002F10	ERROR_SXS_XML_E_INVALID_HEXIDECIMAL	清单解析错误：十六进制数字的字符无效。

错误			描述
十进制	十六进制	名称	
12049	0x00002F11	ERROR_SXS_XML_E_INVALID_UNICODE	清单解析错误：此平台的 Unicode 字符值无效。
12050	0x00002F12	ERROR_SXS_XML_E_WHITESPACEORQUESTIONMARK	清单解析错误：应为空白或 '?' 。
12051	0x00002F13	ERROR_SXS_XML_E_UNEXPECTEDENDTAG	清单解析错误：此位置不应有结束标记。
12052	0x00002F14	ERROR_SXS_XML_E_UNCLOSEDTAG	清单解析错误：以下标记未关闭：%1。
12053	0x00002F15	ERROR_SXS_XML_E_DUPLICATEATTRIBUTE	清单解析错误：重复属性。
12054	0x00002F16	ERROR_SXS_XML_E_MULTIPLEROOTS	清单解析错误：XML 文档中只允许一个顶级元素。
12055	0x00002F17	ERROR_SXS_XML_E_INVALIDATROOTLEVEL	清单解析错误：文档顶层无效。
12056	0x00002F18	ERROR_SXS_XML_E_BADXMLDECL	清单解析错误：XML 声明无效。
12057	0x00002F19	ERROR_SXS_XML_E_MISSINGROOT	清单解析错误：XML 文档必须具有顶级元素。
12058	0x00002F1A	ERROR_SXS_XML_E_UNEXPECTEDEOF	清单解析错误：意外的文件结尾。
12059	0x00002F1B	ERROR_SXS_XML_E_BADPEREFINSUBSET	清单解析错误：参数实体不能在内部子集的标记声明中使用。
12060	0x00002F1C	ERROR_SXS_XML_E_UNCLOSEDSTARTTAG	清单解析错误：元素未关闭。
12061	0x00002F1D	ERROR_SXS_XML_E_UNCLOSEDENDTAG	清单解析错误：结束元素缺少字 '>' 。
12062	0x00002F1E	ERROR_SXS_XML_E_UNCLOSEDSTRING	清单解析错误：字符串文本未关闭。
12063	0x00002F1F	ERROR_SXS_XML_E_UNCLOSEDCOMMENT	清单解析错误：注释未关闭。
12064	0x00002F20	ERROR_SXS_XML_E_UNCLOSEDDECL	清单解析错误：声明未关闭。
12065	0x00002F21	ERROR_SXS_XML_E_UNCLOSEDCDATA	清单解析错误：CDATA 节未关闭。
12066	0x00002F22	ERROR_SXS_XML_E_RESERVEDNAMESPACE	清单解析错误：命名空间前缀不允许以保留字符串 "xml" 开头。
12067	0x00002F23	ERROR_SXS_XML_E_INVALIDENCODING	清单解析错误：系统不支持指定的编码。
12068	0x00002F24	ERROR_SXS_XML_E_INVALIDSWITCH	清单解析错误：不支持从当前编码切换到指定编码。
12069	0x00002F25	ERROR_SXS_XML_E_BADXMLCASE	清单解析错误：名称 'xml' 是保留的，必须小写。
12070	0x00002F26	ERROR_SXS_XML_E_INVALID_STANDALONE	清单解析错误：独立属性的值必须为 "yes" 或 "no" 。
12071	0x00002F27	ERROR_SXS_XML_E_UNEXPECTED_STANDALONE	清单解析错误：独立属性不能在外部实体中使用。
12072	0x00002F28	ERROR_SXS_XML_E_INVALID_VERSION	清单解析错误：版本号无效。
12073	0x00002F29	ERROR_SXS_XML_E_MISSINGEQUALS	清单解析错误：属性和属性值之间缺少等号。
13000	0x000032C8	ERROR_IPSEC_QM_POLICY_EXISTS	指定的快速模式策略已存在。
13001	0x000032C9	ERROR_IPSEC_QM_POLICY_NOT_FOUND	未找到指定的快速模式策略。
13002	0x000032CA	ERROR_IPSEC_QM_POLICY_IN_USE	正在使用指定的快速模式策略。
13003	0x000032CB	ERROR_IPSEC_MM_POLICY_EXISTS	指定的主模式策略已存在。
13004	0x000032CC	ERROR_IPSEC_MM_POLICY_NOT_FOUND	未找到指定的主模式策略。
13005	0x000032CD	ERROR_IPSEC_MM_POLICY_IN_USE	正在使用指定的主模式策略。
13006	0x000032CE	ERROR_IPSEC_MM_FILTER_EXISTS	指定的主模式过滤器已存在。
13007	0x000032CF	ERROR_IPSEC_MM_FILTER_NOT_FOUND	未找到指定的主模式过滤器。
13008	0x000032D0	ERROR_IPSEC_TRANSPORT_FILTER_EXISTS	指定的传输模式过滤器已存在。
13009	0x000032D1	ERROR_IPSEC_TRANSPORT_FILTER_NOT_FOUND	指定的传输模式过滤器不存在。
13010	0x000032D2	ERROR_IPSEC_MM_AUTH_EXISTS	指定的主模式认证列表已存在。
13011	0x000032D3	ERROR_IPSEC_MM_AUTH_NOT_FOUND	未找到指定的主模式认证列表。
13012	0x000032D4	ERROR_IPSEC_MM_AUTH_IN_USE	正在使用指定的快速模式策略。
13013	0x000032D5	ERROR_IPSEC_DEFAULT_MM_POLICY_NOT_FOUND	未找到指定的主模式策略。
13014	0x000032D6	ERROR_IPSEC_DEFAULT_MM_AUTH_NOT_FOUND	未找到指定的快速模式策略。
13015	0x000032D7	ERROR_IPSEC_DEFAULT_QM_POLICY_NOT_FOUND	清单文件包含一个或多个语法错误。
13016	0x000032D8	ERROR_IPSEC_TUNNEL_FILTER_EXISTS	该应用程序试图激活已禁用的激活上下文。
13017	0x000032D9	ERROR_IPSEC_TUNNEL_FILTER_NOT_FOUND	在任何活动的激活上下文中都未找到所要求的查找密钥。
13018	0x000032DA	ERROR_IPSEC_MM_FILTER_PENDING_DELETION	主模式过滤器正在等待删除。
13019	0x000032DB	ERROR_IPSEC_TRANSPORT_FILTER_PENDING_DELETION	传输过滤器正在等待删除。

错误			描述
十进制	十六进制	名称	
13020	0x000032DC	ERROR_IPSEC_TUNNEL_FILTER_PENDING_DELETION	隧道过滤器正在等待删除。
13021	0x000032DD	ERROR_IPSEC_MM_POLICY_PENDING_DELETION	主模式策略正在等待删除。
13022	0x000032DE	ERROR_IPSEC_MM_AUTH_PENDING_DELETION	主模式认证捆绑包正在等待删除。
13023	0x000032DF	ERROR_IPSEC_QM_POLICY_PENDING_DELETION	快速模式策略正在等待删除。
13801	0x000035E9	ERROR_IPSEC_IKE_AUTH_FAIL	IKE 认证凭证不可接受。
13802	0x000035EA	ERROR_IPSEC_IKE_ATTRIB_FAIL	IKE 的安全属性不可接受。
13803	0x000035EB	ERROR_IPSEC_IKE_NEGOTIATION_PENDING	IKE 协商正在进行中。
13804	0x000035EC	ERROR_IPSEC_IKE_GENERAL_PROCESSING_ERROR	一般处理错误。
13805	0x000035ED	ERROR_IPSEC_IKE_TIMED_OUT	协商超时。
13806	0x000035EE	ERROR_IPSEC_IKE_NO_CERT	IKE 未找到有效的机器证书。
13807	0x000035EF	ERROR_IPSEC_IKE_SA_DELETED	IKE SA 在建立完成前被对等方删除。
13808	0x000035F0	ERROR_IPSEC_IKE_SA_REAPED	IKE SA 在建立完成前删除。
13809	0x000035F1	ERROR_IPSEC_IKE_MM_ACQUIRE_DROP	协商请求在队列中停留时间过长。
13810	0x000035F2	ERROR_IPSEC_IKE_QM_ACQUIRE_DROP	协商请求在队列中停留时间过长。
13811	0x000035F3	ERROR_IPSEC_IKE_QUEUE_DROP_MM	协商请求在队列中停留时间过长。
13812	0x000035F4	ERROR_IPSEC_IKE_QUEUE_DROP_NO_MM	协商请求在队列中停留时间过长。
13813	0x000035F5	ERROR_IPSEC_IKE_DROP_NO_RESPONSE	对等方无回应。
13814	0x000035F6	ERROR_IPSEC_IKE_MM_DELAY_DROP	协商时间太长。
13815	0x000035F7	ERROR_IPSEC_IKE_QM_DELAY_DROP	协商时间太长。
13816	0x000035F8	ERROR_IPSEC_IKE_ERROR	发生未知错误。
13817	0x000035F9	ERROR_IPSEC_IKE_CRL_FAILED	证书吊销检查失败。
13818	0x000035FA	ERROR_IPSEC_IKE_INVALID_KEY_USAGE	证书密钥用法无效。
13819	0x000035FB	ERROR_IPSEC_IKE_INVALID_CERT_TYPE	证书类型无效。
13820	0x000035FC	ERROR_IPSEC_IKE_NO_PRIVATE_KEY	没有机器证书相关的私钥。
13822	0x000035FE	ERROR_IPSEC_IKE_DH_FAIL	Diffie-Helman 计算失败。
13824	0x00003600	ERROR_IPSEC_IKE_INVALID_HEADER	标头无效。
13825	0x00003601	ERROR_IPSEC_IKE_NO_POLICY	未配置政策。
13826	0x00003602	ERROR_IPSEC_IKE_INVALID_SIGNATURE	验证签名失败。
13827	0x00003603	ERROR_IPSEC_IKE_KERBEROS_ERROR	使用 Kerberos 认证失败。
13828	0x00003604	ERROR_IPSEC_IKE_NO_PUBLIC_KEY	对等方的证书没有公钥
13829	0x00003605	ERROR_IPSEC_IKE_PROCESS_ERR	处理错误负载时出错。
13830	0x00003606	ERROR_IPSEC_IKE_PROCESS_ERR_SA	处理 SA 负载时出错。
13831	0x00003607	ERROR_IPSEC_IKE_PROCESS_ERR_PROP	处理建议负载时出错。
13832	0x00003608	ERROR_IPSEC_IKE_PROCESS_ERR_TRANS	处理转换负载时出错。
13833	0x00003609	ERROR_IPSEC_IKE_PROCESS_ERR_KE	处理 KE 负载时出错。
13834	0x0000360A	ERROR_IPSEC_IKE_PROCESS_ERR_ID	处理 ID 负载时出错。
13835	0x0000360B	ERROR_IPSEC_IKE_PROCESS_ERR_CERT	处理 Cert 负载时出错。
13836	0x0000360C	ERROR_IPSEC_IKE_PROCESS_ERR_CERT_REQ	处理证书请求负载时出错。
13837	0x0000360D	ERROR_IPSEC_IKE_PROCESS_ERR_HASH	处理哈希负载时出错。
13838	0x0000360E	ERROR_IPSEC_IKE_PROCESS_ERR_SIG	处理签名负载时出错。
13839	0x0000360F	ERROR_IPSEC_IKE_PROCESS_ERR_NONCE	处理 Nonce 负载时出错。
13840	0x00003610	ERROR_IPSEC_IKE_PROCESS_ERR_NOTIFY	处理通知负载时出错。
13841	0x00003611	ERROR_IPSEC_IKE_PROCESS_ERR_DELETE	处理删除负载时出错。
13842	0x00003612	ERROR_IPSEC_IKE_PROCESS_ERR_VENDOR	处理 VendorId 负载时出错。
13843	0x00003613	ERROR_IPSEC_IKE_INVALID_PAYLOAD	接收到无效负载。
13844	0x00003614	ERROR_IPSEC_IKE_LOAD_SOFT_SA	软 SA 已加载。
13845	0x00003615	ERROR_IPSEC_IKE_SOFT_SA_TORN_DOWN	软 SA 拆卸。
13846	0x00003616	ERROR_IPSEC_IKE_INVALID_COOKIE	收到的 cookie 无效..
13847	0x00003617	ERROR_IPSEC_IKE_NO_PEER_CERT	对方未能发送有效的机器证书。
13848	0x00003618	ERROR_IPSEC_IKE_PEER_CRL_FAILED	对等方证书的认证撤销检查失败。
13849	0x00003619	ERROR_IPSEC_IKE_POLICY_CHANGE	新政策使用旧政策形成的 SA 失效。
13850	0x0000361A	ERROR_IPSEC_IKE_NO_MM_POLICY	没有可用的主模式 IKE 策略。

错误			描述
十进制	十六进制	名称	
13851	0x0000361B	ERROR_IPSEC_IKE_NOTCBPRIV	启用 TCB 权限失败。
13852	0x0000361C	ERROR_IPSEC_IKE_SECLOADFAIL	加载 SECURITY.DLL 失败。
13853	0x0000361D	ERROR_IPSEC_IKE_FAILSSPINIT	未能从 SSPI 获得安全功能表调度地址。
13854	0x0000361E	ERROR_IPSEC_IKE_FAILQUERYSSP	无法查询 Kerberos 包以获得最大令牌大小。
13855	0x0000361F	ERROR_IPSEC_IKE_SRVACQFAIL	无法获取 ISAKMP/ERROR_IPSEC_IKE 服务的 Kerberos 服务器凭证。Kerberos 认证将无法发挥作用。最有可能的原因是缺少域成员资格。如果你的计算机是工作组的成员，这很正常。
13856	0x00003620	ERROR_IPSEC_IKE_SRVQUERYCRED	未能确定 ISAKMP/ERROR_IPSEC_IKE 服务的 SSPI 主体名称（QueryCredentialsAttributes）。
13857	0x00003621	ERROR_IPSEC_IKE_GETSPIFAIL	未能从 Ipsec 驱动程序获得入站 SA 的新 SPI。最常见的原因是，驱动程序没有正确的过滤器。检查政策，以验证过滤器。
13858	0x00003622	ERROR_IPSEC_IKE_INVALID_FILTER	给定的过滤器无效。
13859	0x00003623	ERROR_IPSEC_IKE_OUT_OF_MEMORY	内存分配失败。
13860	0x00003624	ERROR_IPSEC_IKE_ADD_UPDATE_KEY_FAILED	未能向 IPSec 驱动程序添加安全关联。最常见的原因是，IKE 协商完成花费的时间太长。如果问题仍然存在，降低故障机器上的负载。
13861	0x00003625	ERROR_IPSEC_IKE_INVALID_POLICY	无效政策。
13862	0x00003626	ERROR_IPSEC_IKE_UNKNOWN_DOI	无效 DOI。
13863	0x00003627	ERROR_IPSEC_IKE_INVALID_SITUATION	无效情况。
13864	0x00003628	ERROR_IPSEC_IKE_DH_FAILURE	Diffie-Hellman 故障。
13865	0x00003629	ERROR_IPSEC_IKE_INVALID_GROUP	Diffie-Hellman 组无效。
13866	0x0000362A	ERROR_IPSEC_IKE_ENCRYPT	加密有效负载时出错。
13867	0x0000362B	ERROR_IPSEC_IKE_DECRYPT	解密有效负载时出错。
13868	0x0000362C	ERROR_IPSEC_IKE_POLICY_MATCH	策略匹配错误。
13869	0x0000362D	ERROR_IPSEC_IKE_UNSUPPORTED_ID	ID 不受支持。
13870	0x0000362E	ERROR_IPSEC_IKE_INVALID_HASH	哈希验证失败。
13871	0x0000362F	ERROR_IPSEC_IKE_INVALID_HASH_ALG	哈希算法无效。
13872	0x00003630	ERROR_IPSEC_IKE_INVALID_HASH_SIZE	哈希大小无效。
13873	0x00003631	ERROR_IPSEC_IKE_INVALID_ENCRYPT_ALG	加密算法无效。
13874	0x00003632	ERROR_IPSEC_IKE_INVALID_AUTH_ALG	认证算法无效。
13875	0x00003633	ERROR_IPSEC_IKE_INVALID_SIG	证书签名无效。
13876	0x00003634	ERROR_IPSEC_IKE_LOAD_FAILED	加载失败。
13877	0x00003635	ERROR_IPSEC_IKE_RPC_DELETE	通过 RPC 调用已删除。
13878	0x00003636	ERROR_IPSEC_IKE_BENIGN_REINIT	为执行重新初始化而创建的临时状态。这不是真正的故障。
13879	0x00003637	ERROR_IPSEC_IKE_INVALID_RESPONDER_LIFETIME_NOTIFY	响应程序生存期通知中接收的生存期值低于 Windows 2000 配置的最小值。请修复对等机上的策略。
13880	0x00003638	ERROR_IPSEC_IKE_QM_LIMIT_REAP	因为达到了 QM 极限，所以获得 SA。
13881	0x00003639	ERROR_IPSEC_IKE_INVALID_CERT_KEYLEN	证书中的密钥长度太小，无法满足配置的安全要求。
13882	0x0000363A	ERROR_IPSEC_IKE_MM_LIMIT	已超过对等机的最大已建立 MM SA 数。
13883	0x0000363B	ERROR_IPSEC_IKE_NEGOTIATION_DISABLED	IKE 收到了一个禁用协商的策略。
13884	0x0000363C	ERROR_IPSEC_IKE_QM_LIMIT	已达到主模式的最大快速模式限制。新的主模式将启动。

10.7 技术支持和服务

倍福公司及其合作伙伴在世界各地提供全面的技术支持和服务，对与倍福产品和系统解决方案相关的所有问题提供快速有效的帮助。

下载搜索器

我们的下载搜索器包含我们供您下载的所有文件。您可以通过它搜索我们的应用案例、技术文档、技术图纸、配置文件等等。

可供下载的文件格式多种多样。

倍福分公司和代表处

若需要倍福产品的本地支持和服务，请联系倍福分公司或代表处！

倍福遍布世界各地的分公司和代表处地址可在倍福官网上找到：http://www.beckhoff.com.cn

该网页还提供更多倍福产品组件的文档。

倍福技术支持

技术支持部门为您提供全面的技术援助，不仅帮助您应用各种倍福产品，还提供其他广泛的服务：

- 技术支持
- 复杂自动化系统的设计、编程和调试
- 以及倍福系统组件的各种培训课程

热线电话：	+49 5246 963-157
电子邮箱：	support@beckhoff.com

倍福售后服务

倍福服务中心提供所有售后服务：

- 现场服务
- 维修服务
- 备件服务
- 热线服务

热线电话：	+49 5246 963-460
电子邮箱：	service@beckhoff.com

倍福公司总部

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

电话：	+49 5246 963-0
电子邮箱：	info@beckhoff.com
网址：	www.beckhoff.com

Trademark statements

Beckhoff®, ATRO®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, MX-System®, Safety over EtherCAT®, TC/BSD®, TwinCAT®, TwinCAT/BSD®, TwinSAFE®, XFC®, XPlanar® and XTS® are registered and licensed trademarks of Beckhoff Automation GmbH.

Third-party trademark statements

Arm, Arm9 and Cortex are trademarks or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere.

Intel, the Intel logo, Intel Core, Xeon, Intel Atom, Celeron and Pentium are trademarks of Intel Corporation or its subsidiaries.

Microsoft, Microsoft Azure, Microsoft Edge, PowerShell, Visual Studio, Windows and Xbox are trademarks of the Microsoft group of companies.

更多信息:
www.beckhoff.com/te1000

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
电话号码: +49 5246 9630
info@beckhoff.com
www.beckhoff.com