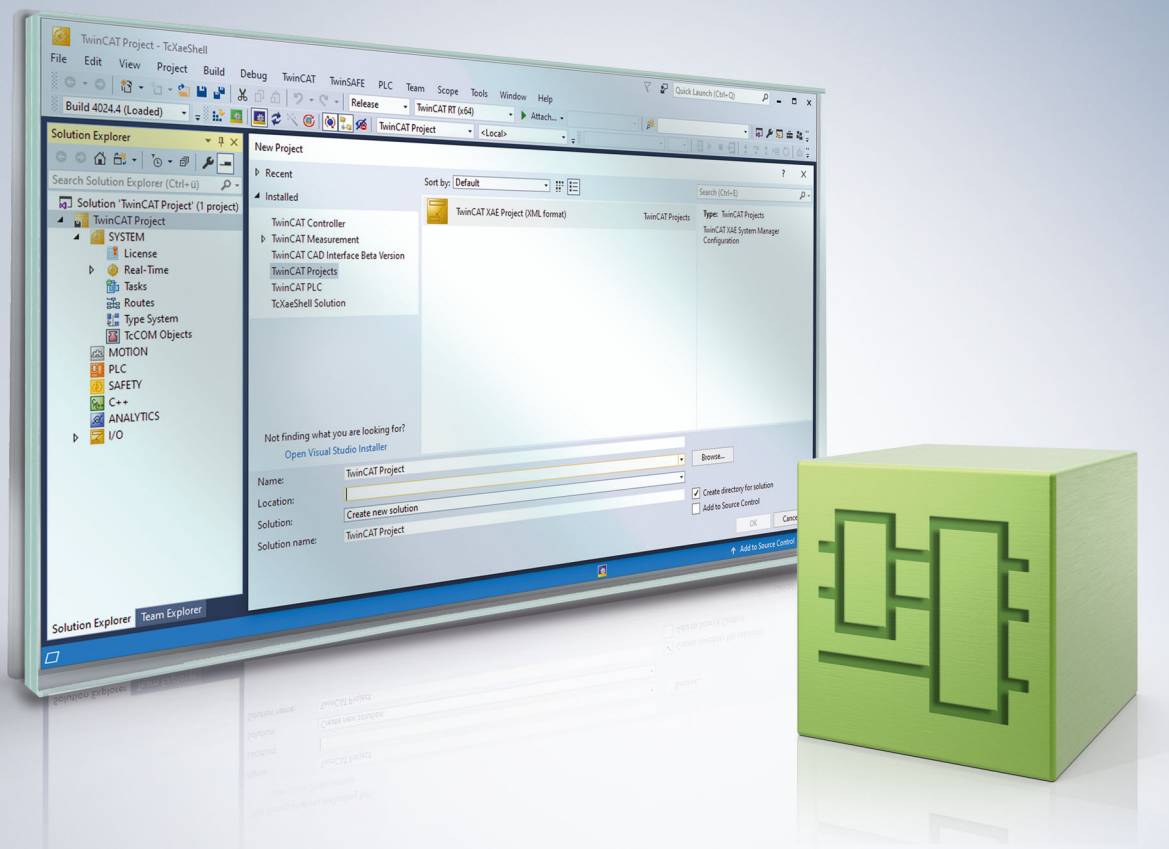
BECKHOFF New Automation Technology

Handbuch | DE

TE1000

TwinCAT 3 | PLC-Bibliothek: Tc2_DMX



23.04.2024 | Version: 1.7.2

Inhaltsverzeichnis

1 Vorwort

1.1 Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, stets die aktuell gültige Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiterentwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

1.2 Zu Ihrer Sicherheit

Sicherheitsbestimmungen

Lesen Sie die folgenden Erklärungen zu Ihrer Sicherheit.
Beachten und befolgen Sie stets produktspezifische Sicherheitshinweise, die Sie gegebenenfalls an den entsprechenden Stellen in diesem Dokument vorfinden.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen vertraut ist.

Signalwörter

Im Folgenden werden die Signalwörter eingeordnet, die in der Dokumentation verwendet werden. Um Personen- und Sachschäden zu vermeiden, lesen und befolgen Sie die Sicherheits- und Warnhinweise.

Warnungen vor Personenschäden

⚠ GEFAHR

Es besteht eine Gefährdung mit hohem Risikograd, die den Tod oder eine schwere Verletzung zur Folge hat.

⚠ WARNUNG

Es besteht eine Gefährdung mit mittlerem Risikograd, die den Tod oder eine schwere Verletzung zur Folge haben kann.

⚠ VORSICHT

Es besteht eine Gefährdung mit geringem Risikograd, die eine mittelschwere oder leichte Verletzung zur Folge haben kann.

Warnung vor Umwelt- oder Sachschäden

HINWEIS

Es besteht eine mögliche Schädigung für Umwelt, Geräte oder Daten.

Information zum Umgang mit dem Produkt

Diese Information beinhaltet z. B.:
Handlungsempfehlungen, Hilfestellungen oder weiterführende Informationen zum Produkt.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

2 Einleitung

Für den Nutzer dieser Bibliothek werden folgende Grundkenntnisse vorausgesetzt:

- TwinCAT XAE
- PC und Netzwerkkenntnisse
- Aufbau und Eigenschaften der Beckhoff Embedded-PC und deren Busklemmensystem
- Technologie von DMX-Geräten

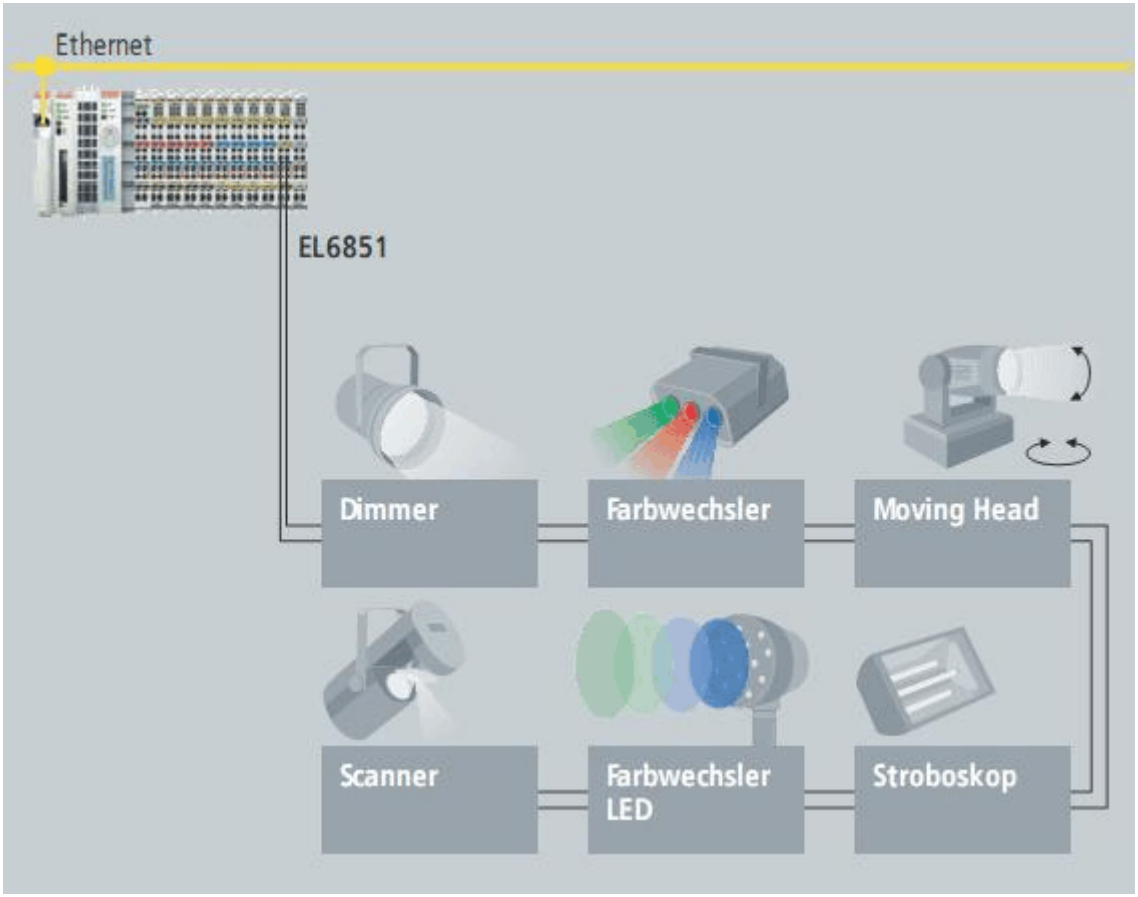
Die Tc2_DMX-Bibliothek ist auf allen Hardware-Plattformen einsetzbar, die TwinCAT 3.1 oder höher unterstützen.

Hardware Dokumentation EL6851 im Beckhoff Information System.

3 DMX

DMX ist das Standardprotokoll für die Ansteuerung von Geräten der professionellen Bühnen- und Effektbeleuchtung, die beispielsweise zur dynamischen Beleuchtung von Show- und Verkaufsräumen, sowie für exklusive Licht- und Farbspiele in prestigeträchtigen Gebäuden, wie Hotels und Veranstaltungszentren, eingesetzt werden. Bei DMX-Geräten in Form von statischen Lichtquellen werden Farbmischungs- und Helligkeitswerte übermittelt, bei bewegten Lichtquellen werden zusätzlich Raumkoordinaten kommuniziert. Die hohe Übertragungsrate von EtherCAT ermöglicht eine höhere Aktualisierungsrate der Lichteinstellungen, und bewirkt, dass die Licht- und Farbwechsel vom Auge harmonischer wahrgenommen werden. Mit der EL6851 können DMX-Geräte mit drei Achsen wie z. B. Scanner, Moving Heads oder Spotlights angesteuert werden. Die Implementierung des RDM-Protokolls (**R**emote **D**evice **M**anagement) für DMX-interne Diagnose und Parametrierung ist mit TwinCAT-Bausteinen möglich.

Der DMX-Master sendet zyklisch mit 250 kBaud neue Einstellungen an Slaves, um dynamische Lichtwechsel und Farbspiele zu generieren. Im DMX-Protokoll sind maximal 32 Slaves an einem Strang ohne Repeater möglich. Der 512 Byte lange Frame im DMX-Protokoll wird als „Universe“ bezeichnet. In ihm stehen 512 Kanäle zur Verfügung, von denen jeweils einer mit 8 Bit Auflösung, also in 256 Stufen, eine Geräte-Einstellung repräsentiert z.B. Dimmen, Farbe, Fokus etc. Bei bewegten Lichtquellen nehmen weitere Einstellungen wie Neigung, Schwenken und Geschwindigkeit (in 8- oder 16 Bit Auflösung) zusätzliche Kanäle in Anspruch, so dass die 512 Kanäle nur indirekt für 32 Teilnehmer ausreichen. Weiterhin benötigt ein Frame bei voller Ausnutzung der Universe 22ms zur internen DMX-Zirkulation, was eine Refresh-Rate von 44 Hz bedeutet. Lichtänderungen bei dieser Frequenz werden als unharmonisch wahrgenommen. Erst ab einer Frequenz von >200Hz erscheinen die Übergänge harmonisch. Durch Verringerung der Nutzdatenmenge kann die Zirkulationsdauer des Frames reduziert werden. Als optimal hat sich eine Ausnutzung von 64 Byte (Frequenz >300 Hz) erwiesen. Damit stehen 64 Kanäle pro Universe zu Verfügung.



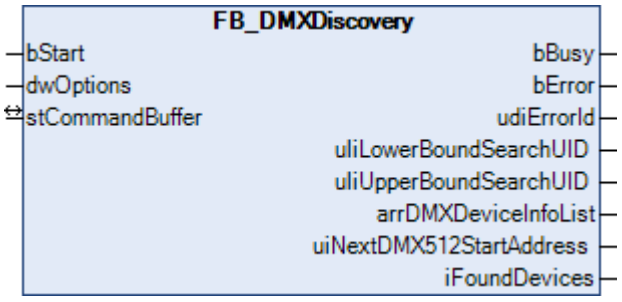
Mit der EL6851 wird die Einbindung mehrerer Universen in eine Steuerung einfach: EtherCAT kann große Datenmengen schnell übermitteln, das EtherCAT-Protokoll bleibt bis in die Klemme erhalten und die Klemme unterstützt verschieden große Mappings (64 bis 512 Byte). So kann bei Anschaltung mehrerer Masterklemmen, jeweils als eigene Universe, der Zeitversatz bei der Übertragung von der Steuerung zum Master deutlich minimiert werden.

4 Programmierung

4.1 POUs

4.1.1 High Level

4.1.1.1 FB_DMXDiscovery



Dieser Funktionsbaustein sucht nach bis zu 50 DMX-Geräten und stellt optional die Startadresse automatisch ein. Die wichtigsten Informationen der gefundenen Geräte werden in einer Struktur angezeigt.

Eingänge

```
VAR_INPUT
  bStart    : BOOL;
  dwOptions : DWORD;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
dwOptions	DWORD	Optionen (siehe Tabelle). Die einzelnen Konstanten müssen miteinander ODER-verknüpft werden.

Konstante	Beschreibung
DMX_OPTION_COMPLETE_NEW_DISCOVERY	Es werden alle DMX-Geräte berücksichtigt.
DMX_OPTION_SET_START_ADDRESS	Bei allen DMX-Geräten die gefunden werden, wird die Startadresse gesetzt. Fortlaufend beginnend mit 1.
DMX_OPTION_OPTICAL_FEEDBACK	Nachdem ein DMX-Gerät gefunden wurde, wird für zwei Sekunden die Funktion IDENTIFY_DEVICE aufgerufen.

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

```
VAR_OUTPUT
  bBusy                 : BOOL;
  bError                : BOOL;
  udiErrorId            : UDINT;
```

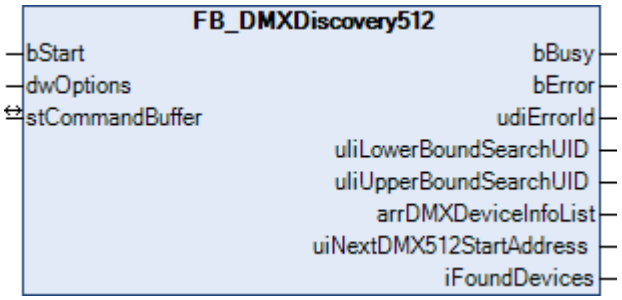
```
  uliLowerBoundSearchUID   : T_ULARGE_INTEGER;
  uliUpperBoundSearchUID   : T_ULARGE_INTEGER;
  arrDMXDeviceInfoList     : ARRAY [1..50] OF ST_DMXDeviceInfo;
  uiNextDMX512StartAddress : UINT;
  iFoundDevices            : INT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
uliLowerBoundSearchUID	T_ULARGE_INTEGER	Während der Suche wird an diesem Ausgang die untere Suchadresse ausgegeben.
uliUpperBoundSearchUID	T_ULARGE_INTEGER	Während der Suche wird an diesem Ausgang die obere Suchadresse ausgegeben.
arrDMXDeviceInfoList	ARRAY OF ST_DMXDeviceInfo [▶ 71]	Array mit den wichtigsten Informationen der gefundenen DMX-Geräte.
uiNextDMX512StartAddress	UINT	Ist die Option DMX_OPTION_SET_START_ADDRESS aktiviert, so wird an diesem Ausgang die Startadresse angezeigt, die dem nächsten DMX-Gerät zugewiesen wird.
iFoundDevices	INT	Während der Suche wird an diesem Ausgang die aktuelle Anzahl der gefunden Geräte ausgegeben.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.1.2 FB_DMXDiscovery512



Dieser Funktionsbaustein sucht nach bis zu 512 DMX-Geräten und stellt optional die Startadresse automatisch ein. Die wichtigsten Informationen der gefundenen Geräte werden in einer Struktur angezeigt.

Eingänge

```
VAR_INPUT
  bStart    : BOOL;
  dwOptions : DWORD;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
dwOptions	DWORD	Optionen (siehe Tabelle). Die einzelnen Konstanten müssen miteinander ODER-verknüpft werden.

Konstante	Beschreibung
DMX_OPTION_COMPLETE_NEW_DISCOVERY	Es werden alle DMX-Geräte berücksichtigt.
DMX_OPTION_SET_START_ADDRESS	Bei allen DMX-Geräten die gefunden werden, wird die Startadresse gesetzt. Fortlaufend beginnend mit 1.
DMX_OPTION_OPTICAL_FEEDBACK	Nachdem ein DMX-Gerät gefunden wurde, wird für zwei Sekunden die Funktion IDENTIFY_DEVICE aufgerufen.

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

```
VAR_OUTPUT
  bBusy                    : BOOL;
  bError                   : BOOL;
  udiErrorId               : UDINT;
  uliLowerBoundSearchUID   : T_ULARGE_INTEGER;
  uliUpperBoundSearchUID   : T_ULARGE_INTEGER;
  arrDMXDeviceInfoList     : ARRAY [1..512] OF ST_DMXDeviceInfo;
  uiNextDMX512StartAddress : UINT;
  iFoundDevices            : INT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
uliLowerBoundSearchUID	T_ULARGE_INTEGER	Während der Suche wird an diesem Ausgang die untere Suchadresse ausgegeben.
uliUpperBoundSearchUID	T_ULARGE_INTEGER	Während der Suche wird an diesem Ausgang die obere Suchadresse ausgegeben.
arrDMXDeviceInfoList	ARRAY OF ST_DMXDeviceInfo [▶ 71]	Array mit den wichtigsten Informationen der gefundenen DMX-Geräte.
uiNextDMX512StartAddress	UINT	Ist die Option DMX_OPTION_SET_START_ADDRESS aktiviert, so wird an diesem Ausgang die Startadresse angezeigt, die dem nächsten DMX-Gerät zugewiesen wird.
iFoundDevices	INT	Während der Suche wird an diesem Ausgang die aktuelle Anzahl der gefunden Geräte ausgegeben.

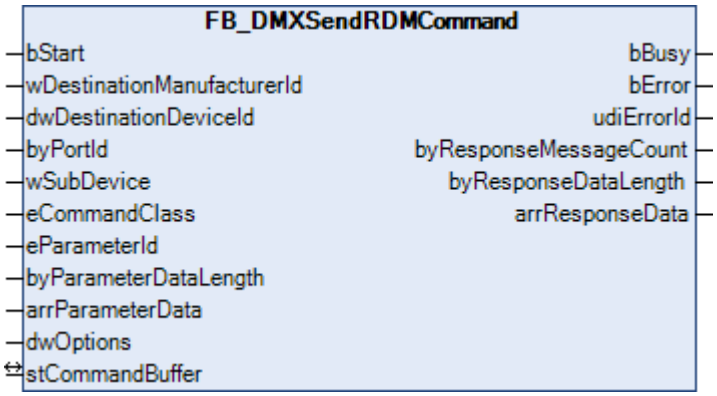
Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2 Low Level

4.1.2.1 Base

4.1.2.1.1 FB_DMXSendRDMCommand



Dieser Funktionsbaustein dient zum allgemeinen Senden eines RDM-Kommandos, definiert per Befehlsnummer und, falls erforderlich, Übergabeparameter.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  wSubDevice                : WORD;
  eCommandClass             : E_DMXCommandClass;
  eParameterId              : E_DMXParameterId;
  byParameterDataLength     : BYTE;
  arrParameterData          : ARRAY [0..255] OF BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
wSubDevice	WORD	Untergeräte (Sub-Devices) sollten in Geräten mit sich wiederholender Anzahl von Modulen verwendet werden, wie bei einem Dimmer Rack. Der Untergeräte (Sub-Device) Eingang erlaubt es Parameternachrichten an ein bestimmtes Modul innerhalb des Gerätes zu senden, um Eigenschaften von dem Modul zu lesen oder zu setzen.
eCommandClass	E_DMXCommandClass [▶ 64]	Command Class (CC) gibt die Aktion der Nachricht an (siehe E_DMXCommandClass).
eParameterId	E_DMXParameterId [▶ 65]	Parameter Id ist eine 16-Bit Nummer, die einen bestimmten Typ von Parameterdaten identifiziert (siehe E_DMXParameterId).
byParameterDataLength	BYTE	Die Datenlänge der Parameter (PDL) ist die vorausgegangene Anzahl an Slots enthalten im Parameterdatenbereich. Ist dieser Eingang 0x00, so folgen keine Parameterdaten.
arrParameterData	ARRAY OF BYTE	Parameterdaten von variabler Länge. Das Format vom Inhalt ist PID abhängig.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

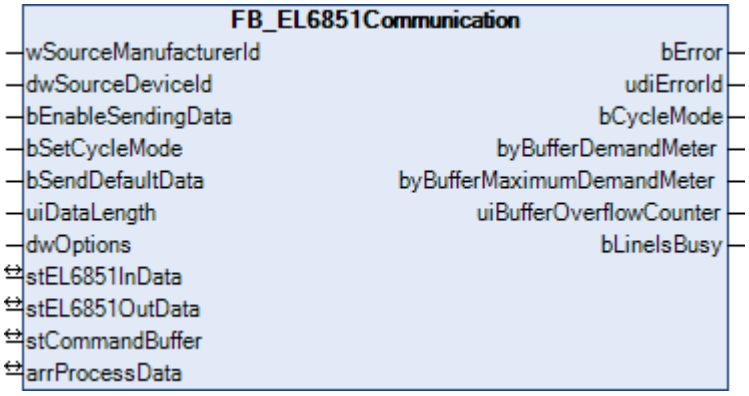
```
VAR_OUTPUT
  bBusy                  : BOOL;
  bError                 : BOOL;
  udiErrorId             : UDINT;
  byResponseMessageCount : BYTE;
  byResponseDataLength   : BYTE;
  arrResponseData        : ARRAY [0..255] OF BYTE;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
byResponseMessageCount	BYTE	Dieser Ausgang zeigt an, dass sich in den DMX-Slave weitere Nachrichten befinden. Mit dem RDM-Befehl Get: QUEUED_MESSAGE werden diese Nachrichten ausgelesen.
byResponseDataLength	BYTE	Enthält die Anzahl der Bytes, die durch den RDM-Befehl zurückgegeben werden.
arrResponseData	ARRAY OF BYTE	Dieser Ausgang enthält die Daten der Rückantwort vom RDM-Befehl. Die Länge ist variabel und das Format der Daten ist abhängig vom RDM-Befehl.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.1.2 FB_EL6851Communication



Veraltet

Dieser Baustein ist veraltet. Nutzen Sie stattdessen den FB_EL6851CommunicationEx() [▶ 18].

Der Zugriff auf die EL6851 sollte immer über diesen Baustein erfolgen. Dieses betrifft sowohl das Versenden der zyklischen DMX-Daten, als auch das Versenden der RDM-Befehle.

Sollen Daten zyklisch zu den DMX-Geräten versendet werden, so setzen sie den Eingang *bEnableSendingData* auf TRUE, den Eingang *bSetCycleMode* auf TRUE, den Eingang *bSendDefaultData* auf FALSE und den Eingang *uiDataLength* auf die entsprechende Länge (in Byte). Die Daten, die versendet werden sollen, können über die Variable *arrProcessData* vorgegeben werden.

Sollen RDM-Befehle versendet werden, so setzen sie den Eingang *bEnableSendingData* auf FALSE und den Eingang *bSetCycleMode* auf FALSE. Die Bausteine für die DMX-RDM-Befehle greifen nicht direkt auf das Prozessabbild der EL6851 zu, sondern legen die einzelnen DMX-RDM-Befehle in einen Puffer ab. Der Baustein FB_EL6851Communication() liest sequentiell die Befehle aus diesen Puffer aus und gibt diese zu der EL6851 weiter. Hierdurch wird sichergestellt, dass nicht mehrere Bausteine gleichzeitig auf das

Prozessabbild der EL6851 zugreifen. Der Puffer, in dem die DMX-RDM-Befehle abgelegt werden, ist in einer Variablen vom Typ ST_DMXCommandBuffer enthalten. Pro EL6851 gibt es eine Instanz vom Baustein FB_EL6851Communication() und eine Variable vom Typ ST_DMXCommandBuffer.

Über die Ausgänge des Bausteins kann ermittelt werden, wie stark der Puffer ausgelastet ist. Sollten Sie feststellen, dass der Puffer regelmäßig überläuft, so sollten Sie mit Hilfe des TwinCAT System Managers die Auslastung der SPS-Task analysieren.

Der Baustein FB_EL6851Communication() kann, wenn nötig in einer separaten, schnelleren Task aufgerufen werden. In diesem Fall sollte die schnellere Task, in der der Baustein FB_EL6851Communication() aufgerufen wird, eine höhere Priorität haben, als die TASK, in der die Bausteine für die RDM-Befehle aufgerufen werden.

Zu beiden Betriebsarten finden sie auch entsprechende Beispiele im Anhang.

Anmerkung zu den IDs von DMX-Geräten

Jedes DMX-Gerät hat eine eindeutige, feste, 48-Bit lange Adresse, auch Unique ID oder kurz UID genannt. Diese Adresse besteht aus der Manufacturer ID (16 Bit) und der Device ID (32 Bit). Die Manufacturer ID identifiziert den Hersteller des Gerätes und wird von der ESTA (Entertainment Services and Technology Association) vergeben. Eine Liste mit allen bekannten Manufacturer IDs ist zu finden unter http://www.esta.org/tsp/working_groups/CP/mfctrIDs.php. Die Device ID wird frei vom Hersteller festgelegt. Hierdurch soll sichergestellt werden, dass jede UID weltweit einmalig vorhanden ist. Die UID kann in der Regel nicht verändert werden. Von der ESTA wurde Beckhoff Automation die Hersteller ID 0x4241 zugeordnet. Da auch ein DMX-Master eine UID besitzt, sollte diese entsprechend der ESTA angegeben werden (Eingang *wSourceManufacturerId*).

Eingänge

```
VAR_INPUT
  wSourceManufacturerId : WORD := 16#42_41;
  dwSourceDeviceId      : DWORD := 16#12_13_14_15;
  bEnableSendingData    : BOOL := TRUE;
  bSetCycleMode         : BOOL := TRUE;
  bSendDefaultData      : BOOL;
  uiDataLength          : UINT;
  dwOptions             : DWORD;
END_VAR
```

Name	Typ	Beschreibung
wSourceManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät. Sollte gemäß der ESTA 0x4241 sein.
dwSourceDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät. Kann frei vergeben werden.
bEnableSendingData	BOOL	Ist die Klemme im CycleMode (Ausgang CycleMode = TRUE), so kann das Versenden mit diesem Baustein aktiviert (TRUE) oder gesperrt (FALSE) werden.
bSetCycleMode	BOOL	Aktiviert den CycleMode. Im CycleMode können die zyklischen Prozessdaten an die DMX-Geräte versendet werden. Zum Versenden der RDM-DMX-Befehle muss der CycleMode deaktiviert werden.
bSendDefaultData	BOOL	Ist dieser Eingang aktiv (TRUE), so werden im CycleMode die Standardwerte versendet.
uiDataLength	UINT	Dieser Eingang ist nur relevant, wenn der CycleMode aktiv ist. Er gibt die Länge des DMX512-Frames in Bytes an.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stEL6851InData  : ST_EL6851InData;
  stEL6851OutData : ST_EL6851OutData;
```

```
  stCommandBuffer : ST_DMXCommandBuffer;
  arrProcessData  : ARRAY [1..512] OF BYTE;
END_VAR
```

Name	Typ	Beschreibung
stEL6851InData	ST_EL6851InData [▶ 75]	Struktur im Eingangsprozessabbild der EL6851. Sie dient zur Kommunikation von der EL6851 zur SPS.
stEL6851OutData	ST_EL6851OutData [▶ 75]	Struktur im Ausgangsprozessabbild der EL6851. Sie dient zur Kommunikation von der SPS zur EL6851.
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]
arrProcessData	ARRAY OF BYTE	Über diese Variable werden dem Baustein die Daten übergeben, die zyklisch an die DMX-Geräte versendet werden sollen. Hierzu muss der CycleMode aktiv sein (siehe auch Eingang *bSetCycleMode*).

Ausgänge

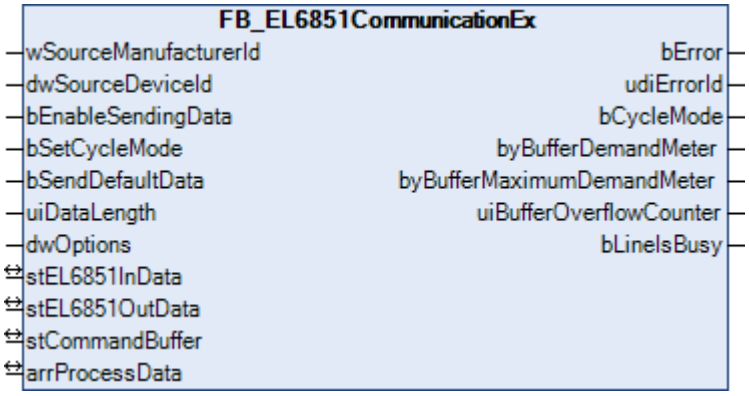
```
VAR_OUTPUT
  bError                    : BOOL;
  udiErrorId                : UDINT;
  bCycleMode                : BOOL;
  byBufferDemandMeter       : BYTE;
  byBufferMaximumDemandMeter : BYTE;
  uiBufferOverflowCounter   : UINT;
  bLineIsBusy               : BOOL;
END_VAR
```

Name	Typ	Beschreibung
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
bCycleMode	BOOL	Ist auf TRUE, wenn der CycleMode aktiv ist (siehe auch Eingang *bSetCycleMode*).
byBufferDemandMeter	BYTE	Belegung des jeweiligen Puffers (0 - 100%).
byBufferMaximumDemandMeter	BYTE	Bisherige maximale Auslastung des jeweiligen Puffers (0 - 100%).
uiBufferOverflowCounter	UINT	Bisherige Anzahl der Pufferüberläufe.
bLineIsBusy	BOOL	Solange der Baustein FB_EL6851Communication() DMX-RDM-Befehle bearbeitet, ist dieser Ausgang gesetzt.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.1.3 FB_EL6851CommunicationEx



Der Zugriff auf die EL6851 sollte immer über diesen Baustein erfolgen. Dieses betrifft sowohl das Versenden der zyklischen DMX-Daten, als auch das Versenden der RDM-Befehle.

Sollen Daten zyklisch zu den DMX-Geräten versendet werden, so setzen sie den Eingang *bEnableSendingData* auf TRUE, den Eingang *bSetCycleMode* auf TRUE, den Eingang *bSendDefaultData* auf FALSE und den Eingang *uiDataLength* auf die entsprechende Länge (jn Byte). Die Daten, die versendet werden sollen, können über die Variable *arrProcessData* vorgegeben werden.

Sollen RDM-Befehle versendet werden, so setzen sie den Eingang *bEnableSendingData* auf FALSE und den Eingang *bSetCycleMode* auf FALSE. Die Bausteine für die DMX-RDM-Befehle greifen nicht direkt auf das Prozessabbild der EL6851 zu, sondern legen die einzelnen DMX-RDM-Befehle in einen Puffer ab. Der Baustein FB_EL6851CommunicationEx() liest sequentiell die Befehle aus diesen Puffer aus und gibt diese zu der EL6851 weiter. Hierdurch wird sichergestellt, dass nicht mehrere Bausteine gleichzeitig auf das Prozessabbild der EL6851 zugreifen. Der Puffer, in dem die DMX-RDM-Befehle abgelegt werden, ist in einer Variablen vom Typ ST_DMXCommandBuffer enthalten. Pro EL6851 gibt es eine Instanz vom Baustein FB_EL6851CommunicationEx() und eine Variable vom Typ ST_DMXCommandBuffer.

Über die Ausgänge des Bausteins kann ermittelt werden, wie stark der Puffer ausgelastet ist. Sollten Sie feststellen, dass der Puffer regelmäßig überläuft, so sollten mit Hilfe des TwinCAT System Managers die Auslastung der SPS-Task analysieren.

Der Baustein FB_EL6851CommunicationEx() kann, wenn nötig in einer separaten, schnelleren Task aufgerufen werden. In diesem Fall sollte die schnellere Task, in der der Baustein FB_EL6851CommunicationEx() aufgerufen wird, eine höhere Priorität haben, als die TASK, in der die Bausteine für die RDM-Befehle aufgerufen werden.

Zu beiden Betriebsarten finden sie auch entsprechende Beispiele im Anhang.

Anmerkung zu den IDs von DMX-Geräten

Jedes DMX-Gerät hat eine eindeutige, feste, 48-Bit lange Adresse, auch Unique ID oder kurz UID genannt. Diese Adresse besteht aus der Manufacturer ID (16 Bit) und der Device ID (32 Bit). Die Manufacturer ID identifiziert den Hersteller des Gerätes und wird von der ESTA (Entertainment Services and Technology Association) vergeben. Eine Liste mit allen bekannten Manufacturer IDs ist zu finden unter http://www.esta.org/tsp/working_groups/CP/mfctrIDs.php. Die Device ID wird frei vom Hersteller festgelegt. Hierdurch soll sichergestellt werden, dass jede UID weltweit einmalig vorhanden ist. Die UID kann in der Regel nicht verändert werden. Von der ESTA wurde Beckhoff Automation die Hersteller ID 0x4241 zugeordnet. Da auch ein DMX-Master eine UID besitzt, sollte diese entsprechend der ESTA angegeben werden (Eingang *wSourceManufacturerId*).

Eingänge

```
VAR_INPUT
  wSourceManufacturerId : WORD := 16#42_41;
  dwSourceDeviceId      : DWORD := 16#12_13_14_15;
  bEnableSendingData    : BOOL := TRUE;
  bSetCycleMode         : BOOL := TRUE;
  bSendDefaultData      : BOOL;
```

```
  uiDataLength          : UINT;
  dwOptions             : DWORD;
END_VAR
```

Name	Typ	Beschreibung
wSourceManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät. Sollte gemäß der ESTA 0x4241 sein.
dwSourceDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät. Kann frei vergeben werden.
bEnableSendingData	BOOL	Ist die Klemme im CycleMode (Ausgang CycleMode = TRUE), so kann das Versenden mit diesem Baustein aktiviert (TRUE) oder gesperrt (FALSE) werden.
bSetCycleMode	BOOL	Aktiviert den CycleMode. Im CycleMode können die zyklischen Prozessdaten an die DMX-Geräte versendet werden. Zum Versenden der RDM-DMX-Befehle muss der CycleMode deaktiviert werden.
bSendDefaultData	BOOL	Ist dieser Eingang aktiv (TRUE), so werden im CycleMode die Standardwerte versendet.
uiDataLength	UINT	Dieser Eingang ist nur relevant, wenn der CycleMode aktiv ist. Er gibt die Länge des DMX512-Frames in Bytes an.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

Ein-/Ausgänge

```
VAR_IN_OUT
  stEL6851InData  : ST_EL6851InData;
  stEL6851OutData : ST_EL6851OutData;
  stCommandBuffer : ST_DMXCommandBuffer;
  arrProcessData  : ARRAY [1..512] OF BYTE;
END_VAR
```

Name	Typ	Beschreibung
stEL6851InData	ST_EL6851InData [▶ 75]	Struktur im Eingangsprozessabbild der EL6851. Sie dient zur Kommunikation von der EL6851 zur SPS.
stEL6851OutData	ST_EL6851OutData [▶ 75]	Struktur im Ausgangsprozessabbild der EL6851. Sie dient zur Kommunikation von der SPS zur EL6851.
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]
arrProcessData	ARRAY OF BYTE	Über diese Variable werden dem Baustein die Daten übergeben, die zyklisch an die DMX-Geräte versendet werden sollen. Hierzu muss der CycleMode aktiv sein (siehe auch Eingang *bSetCycleMode*).

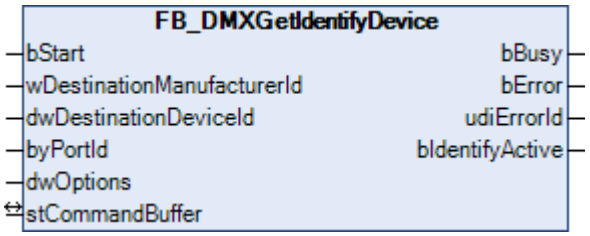
Ausgänge

```
VAR_OUTPUT
  bError                    : BOOL;
  udiErrorId                : UDINT;
  bCycleMode                : BOOL;
  byBufferDemandMeter       : BYTE;
  byBufferMaximumDemandMeter : BYTE;
  uiBufferOverflowCounter   : UINT;
  bLineIsBusy               : BOOL;
END_VAR
```

Name	Typ	Beschreibung
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
bCycleMode	BOOL	Ist auf TRUE, wenn der CycleMode aktiv ist (siehe auch Eingang *bSetCycleMode*).
byBufferDemandMeter	BYTE	Belegung des jeweiligen Puffers (0 - 100%).
byBufferMaximumDemandMeter	BYTE	Bisherige maximale Auslastung des jeweiligen Puffers (0 - 100%).
uiBufferOverflowCounter	UINT	Bisherige Anzahl der Pufferüberläufe.
bLineIsBusy	BOOL	Solange der Baustein FB_EL6851Communication() DMX-RDM-Befehle bearbeitet, ist dieser Ausgang gesetzt.

4.1.2.2 Device Control Parameter Message

4.1.2.2.1 FB_DMXGetIdentifyDevice



Dieser Funktionsbaustein fragt ab, ob von einem DMX-Gerät die Identifizierung aktiv ist.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *bIdentifyActive* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                  : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId   : DWORD;
  byPortId                : BYTE;
  dwOptions               : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

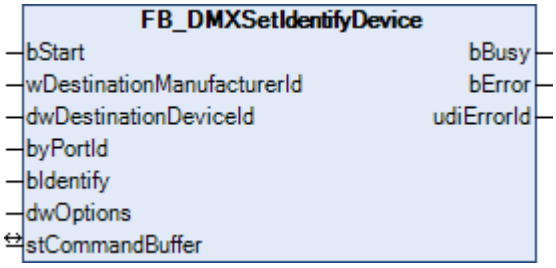
```
VAR_OUTPUT
  bBusy           : BOOL;
  bError          : BOOL;
  udiErrorId      : UDINT;
  bIdentifyActive : BOOL;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
bIdentifyActive	BOOL	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang der Zustand der Identifizierung von dem DMX-Gerät angezeigt.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.2.2 FB_DMXSetIdentifyDevice



Dieser Funktionsbaustein aktiviert oder deaktiviert die Identifizierung eines DMX-Gerätes.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, so geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError* und *udiErrorId* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  bIdentify                 : BOOL := FALSE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
bIdentify	BOOL	Legt fest, ob die Identifizierung aktiviert (TRUE) oder deaktiviert (FALSE) werden soll.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

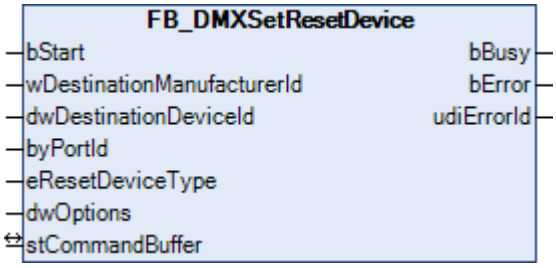
```
VAR_OUTPUT
  bBusy         : BOOL;
  bError        : BOOL;
  udiErrorId    : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.2.3 FB_DMXSetResetDevice



Dieser Funktionsbaustein aktiviert einen Reset bei einem DMX-Gerät.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError* und *udiErrorId* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  eResetDeviceType          : E_DMXResetDeviceType := eDMXResetDeviceTypeWarm;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
eResetDeviceType	E_DMXResetDeviceType [▶ 67]	Legt fest, ob ein Warmstart (eDMXResetDeviceTypeWarm) oder ein Kaltstart (eDMXResetDeviceTypeCold) ausgeführt werden soll. Andere Werte sind für diesen Eingang nicht möglich.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

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

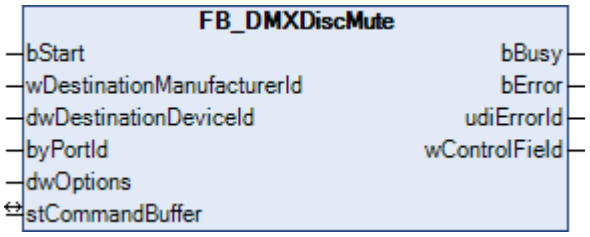
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.3 Discovery Messages

4.1.2.3.1 FB_DMXDiscMute



Dieser Funktionsbaustein setzt das Mute-Flag von einem DMX-Gerät. Das Mute-Flag legt fest, ob ein DMX-Gerät auf den Befehl FB_DMXDiscUniqueBranch() [▶ 26] reagiert (Mute-Flag ist nicht gesetzt) oder nicht (Mute-Flag ist gesetzt).

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *wControlField* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

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

```
  udiErrorId    : UDINT;
  wControlField : WORD;
END_VAR
```

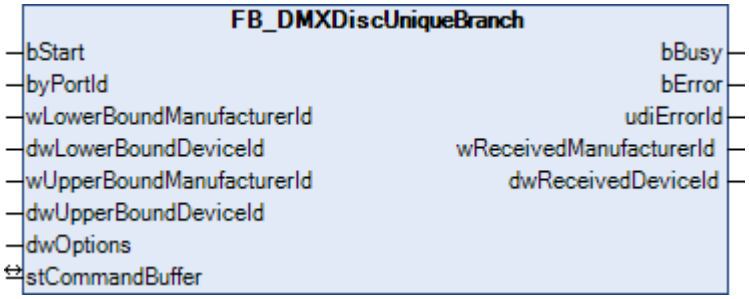
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
wControlField	WORD	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so werden an diesem Ausgang weitere Informationen über das DMX-Gerät ausgegeben. Hierbei ist die Bedeutung der einzelnen Bits wie folgt definiert (s. Tabelle):

Bit	Beschreibung
0 - Managed Proxy Flag	Dieses Bit ist gesetzt, wenn das DMX-Gerät ein Proxygerät ist.
1 - Sub-Device Flag	Dieses Bit ist gesetzt, wenn das DMX-Gerät Untergeräte (Sub-Devices) unterstützt.
2 - Boot-Loader Flag	Dieses Bit ist gesetzt, wenn das DMX-Gerät keine Befehle empfangen kann (z.B. während die Firmware geladen wird).
3 - Proxied Device Flag	Dieses Bit ist gesetzt, wenn die Antwort von einem Proxygerät gesendet wurde.
4 - 15	Reserve (immer 0).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.3.2 FB_DMXDiscUniqueBranch



Dieser Funktionsbaustein fragt ab, ob sich innerhalb eines bestimmten Adressbereichs DMX-Geräte befinden. Dieser Befehl findet Verwendung beim Ermitteln (Discovery) der angeschlossenen DMX-Geräte.

Durch eine positive Flanke am Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wLowerBoundManufacturerId*, *dwLowerBoundDeviceId*, *wUpperBoundManufacturerId* und *dwUpperBoundDeviceId* definieren den Adressbereich, in dem DMX-Geräte gesucht werden. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId*, *wReceivedManufacturerId* und *dwReceivedDeviceId* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Liegt im definierten Adressbereich nur ein DMX-Gerät, so werden über die Ausgänge *wReceivedManufacturerId* und *dwReceivedDeviceId* die 48-Bit UID des DMX-Gerätes zurückgegeben. Liegen in diesem Bereich keine DMX-Geräte, so ist der Ausgang *bError* auf TRUE und *udiErrorId* ist auf 0x8002 (keine Antwort vom DMX-Gerät). Befinden sich in dem Adressbereich zwei oder mehr DMX-Geräte, so ist *bError* auf TRUE und *udiError* enthält eine 0x8006 (Checksummenfehler).

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  byPortId                  : BYTE;
  wLowerBoundManufacturerId : WORD;
  dwLowerBoundDeviceId      : DWORD;
  wUpperBoundManufacturerId : WORD;
  dwUpperBoundDeviceId      : DWORD;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
wLowerBoundManufacturerId	WORD	Eindeutige Hersteller-Id vom unteren Adressbereich.
dwLowerBoundDeviceId	DWORD	Eindeutige Geräte-Id vom unteren Adressbereich.
wUpperBoundManufacturerId	WORD	Eindeutige Hersteller-Id vom oberen Adressbereich.
dwUpperBoundDeviceId	DWORD	Eindeutige Geräte-Id vom oberen Adressbereich.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

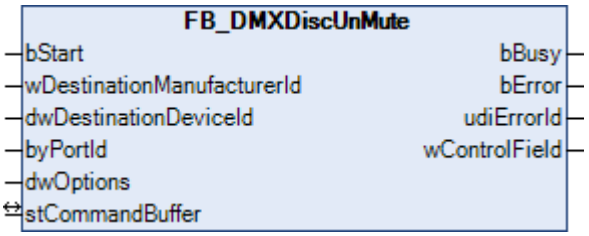
```
VAR_OUTPUT
  bBusy                   : BOOL;
  bError                  : BOOL;
  udiErrorId              : UDINT;
  wReceivedManufacturerId : WORD;
  dwReceivedDeviceId      : DWORD;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
wReceivedManufacturerId	WORD	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang der Zustand der Identifizierung von dem DMX-Gerät angezeigt.
dwReceivedDeviceId	DWORD	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang der Zustand der Identifizierung von dem DMX-Gerät angezeigt.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.3.3 FB_DMXDiscUnMute



Dieser Funktionsbaustein setzt das Mute-Flag von einem DMX-Gerät zurück. Das Mute-Flag legt fest, ob ein DMX-Gerät auf den Befehl FB_DMXDiscUniqueBranch() [▶ 26] reagiert (Mute-Flag ist nicht gesetzt) oder nicht (Mute-Flag ist gesetzt).

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *wControlField* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                   : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                 : BYTE;
  dwOptions                : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

```
VAR_OUTPUT
  bBusy         : BOOL;
  bError        : BOOL;
  udiErrorId    : UDINT;
  wControlField : WORD;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
wControlField	WORD	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so werden an diesem Ausgang weitere Informationen über das DMX-Gerät ausgegeben. Hierbei ist die Bedeutung der einzelnen Bits wie folgt definiert (s. Tabelle):

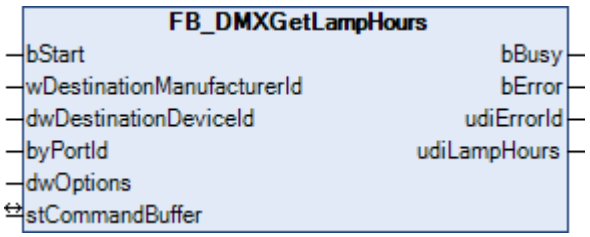
Bit	Beschreibung
0 - Managed Proxy Flag	Dieses Bit ist gesetzt, wenn das DMX-Gerät ein Proxygerät ist.
1 - Sub-Device Flag	Dieses Bit ist gesetzt, wenn das DMX-Gerät Untergeräte (Sub-Devices) unterstützt.
2 - Boot-Loader Flag	Dieses Bit ist gesetzt, wenn das DMX-Gerät keine Befehle empfangen kann (z.B. während die Firmware geladen wird).
3 - Proxied Device Flag	Dieses Bit ist gesetzt, wenn die Antwort von einem Proxygerät gesendet wurde.
4 - 15	Reserve (immer 0).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.4 Power and Lamp Setting Parameter Messages

4.1.2.4.1 FB_DMXGetLampHours



Dieser Funktionsbaustein liest die Anzahl der Stunden aus, in denen die Lampe an war. Durch den Baustein FB_DMXSetLampHours() [▶ 33] kann dieser Stundenzähler verändert werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

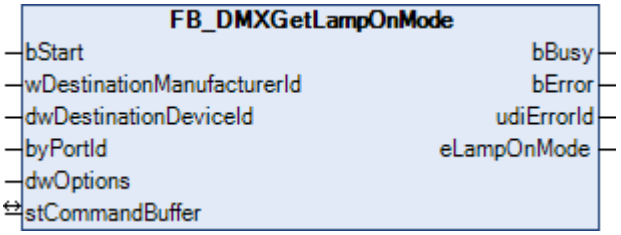
```
VAR_OUTPUT
  bBusy           : BOOL;
  bError          : BOOL;
  udiErrorId      : UDINT;
  udiLampHours    : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
udiLampHours	UDINT	Anzahl der Stunden, in denen die Lampe eingeschaltet war.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.4.2 FB_DMXGetLampOnMode



Dieser Funktionsbaustein liest den Parameter aus, der das Einschaltverhalten des DMX-Gerätes definiert. Mit dem Baustein FB_DMXSetLampOnMode() [▶ 34] kann der Wert verändert werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

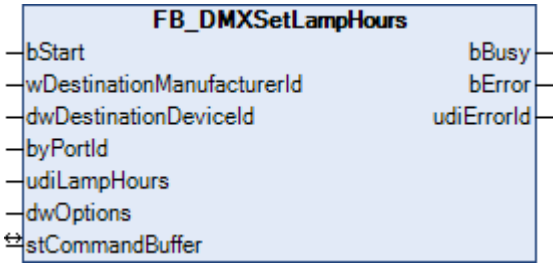
```
VAR_OUTPUT
  bBusy           : BOOL;
  bError          : BOOL;
  udiErrorId      : UDINT;
  eLampOnMode     : E_DMXLampOnMode;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
eLampOnMode	E_DMXLampOnMode [▶ 64]	Enthält den aktuellen Parameter, der das Einschaltverhalten des DMX-Gerätes definiert.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.4.3 FB_DMXSetLampHours



Dieser Funktionsbaustein setzt den Betriebsstundenzähler der Lampe. Durch den Baustein FB_DMXGetLampHours() [▶ 30] kann der Zähler ausgelesen werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  udiLampHours              : UDINT := 0;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
udiLampHours	UDINT	Neuer Wert für den Betriebsstundenzähler.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

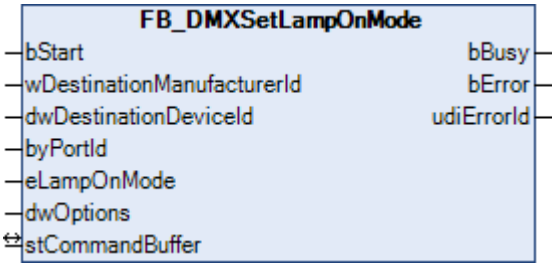
```
VAR_OUTPUT
  bBusy         : BOOL;
  bError        : BOOL;
  udiErrorId    : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.4.4 FB_DMXSetLampOnMode



Dieser Funktionsbaustein legt das Einschaltverhalten des DMX-Gerätes fest. Mit dem Baustein FB_DMXGetLampOnMode() [▶ 31] kann der eingestellte Wert ausgelesen werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  eLampOnMode               : E_DMXLampOnMode := eDMXLampOnModeOff;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
eLampOnMode	E_DMXLampOnMode [▶ 64]	Dieser Parameter legt das Einschaltverhalten des DMX-Gerätes fest.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

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

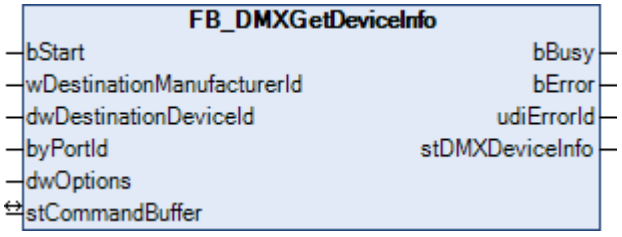
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5 Product Information Messages

4.1.2.5.1 FB_DMXGetDeviceInfo



Dieser Funktionsbaustein fragt alle relevanten Informationen von einem DMX-Gerät ab.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *stDMXDeviceInfo* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
```

```
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

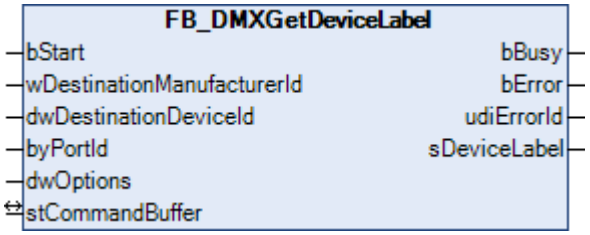
```
VAR_OUTPUT
  bBusy           : BOOL;
  bError          : BOOL;
  udiErrorId      : UDINT;
  stDMXDeviceInfo : ST_DMXDeviceInfo;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
stDMXDeviceInfo	ST_DMXDeviceInfo [▶ 71]	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so werden an diesem Ausgang in einer Struktur alle relevanten Informationen des DMX-Gerätes ausgegeben.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5.2 FB_DMXGetDeviceLabel



Dieser Funktionsbaustein liest aus dem DMX-Gerät einen Text aus, der das Gerät genauer beschreibt. Mit dem Baustein FB_DMXSetDeviceLabel() [▶ 44] kann der Text verändert werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

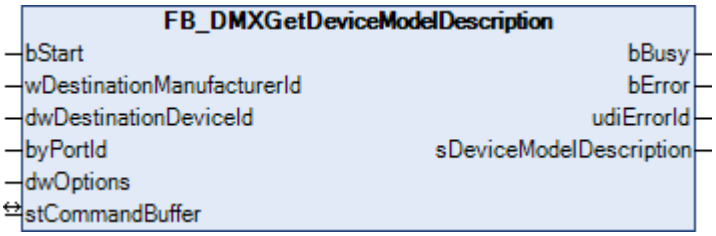
```
VAR_OUTPUT
  bBusy          : BOOL;
  bError         : BOOL;
  udiErrorId     : UDINT;
  sDeviceLabel   : STRING;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
sDeviceLabel	STRING	Beschreibungstext des DMX-Gerätes.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5.3 FB_DMXGetDeviceModelDescription



Dieser Funktionsbaustein fragt die Beschreibung des Gerätetyps ab.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *sDeviceModelDescription* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

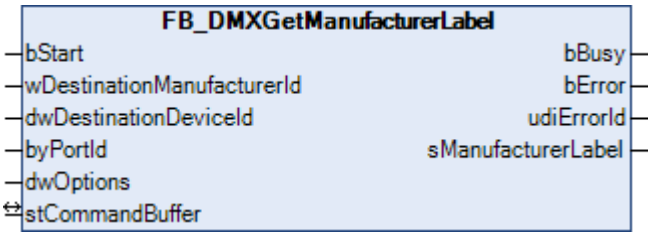
```
VAR_OUTPUT
  bBusy                  : BOOL;
  bError                 : BOOL;
  udiErrorId             : UDINT;
  sDeviceModelDescription : STRING;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
sDeviceModelDescription	STRING	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang eine Beschreibung (maximal 32 Zeichen) des Gerätetyps ausgegeben. Der Inhalt wird vom DMX-Gerätehersteller festgelegt.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5.4 FB_DMXGetManufacturerLabel



Dieser Funktionsbaustein fragt die Beschreibung des DMX-Geräteherstellers ab.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *sManufacturerLabel* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

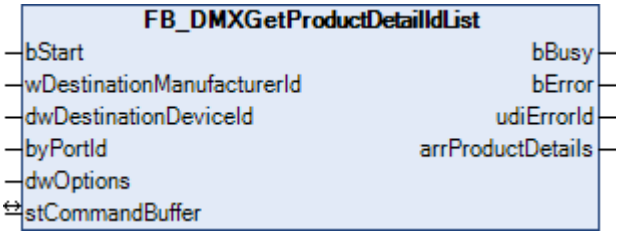
```
VAR_OUTPUT
  bBusy                  : BOOL;
  bError                 : BOOL;
  udiErrorId             : UDINT;
  sManufacturerLabel     : STRING;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
sManufacturerLabel	STRING	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang eine Beschreibung (maximal 32 Zeichen) des DMX-Geräteherstellers ausgegeben. Der Inhalt wird vom DMX-Gerätehersteller festgelegt.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5.5 FB_DMXGetProductDetailIdList



Dieser Funktionsbaustein fragt die Kategorien ab, zu denen das DMX-Gerät zugehörig ist.

RDM definiert verschiedene Gerätekategorien. Jedes DMX-Gerät kann bis zu 6 Kategorien zugeordnet werden. Die Zuordnung erfolgt durch den Gerätehersteller und kann nicht per RDM geändert werden.

Eingänge

```
VAR_INPUT
  bStart                  : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                : BYTE;
  dwOptions               : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

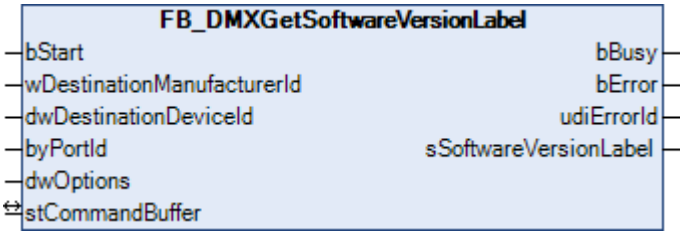
```
VAR_OUTPUT
  bBusy             : BOOL;
  bError            : BOOL;
  udiErrorId        : UDINT;
  arrProductDetails : ARRAY [1..6] OF E_DMXProductDetail;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
arrProductDetails	ARRAY OF E_DMXProductDetail [▶ 66]	Enthält die Liste mit bis zu 6 Gerätekategorien.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5.6 FB_DMXGetSoftwareVersionLabel



Dieser Funktionsbaustein fragt die Beschreibung der Softwareversion des DMX-Gerätes ab.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *sSoftwareVersionLabel* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positiven Flanke am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

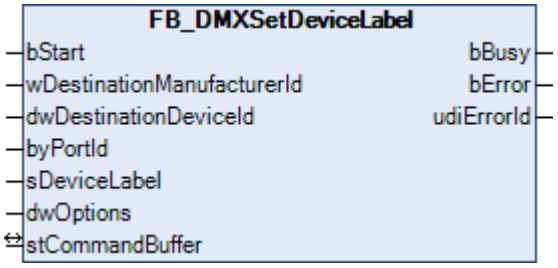
```
VAR_OUTPUT
  bBusy                  : BOOL;
  bError                 : BOOL;
  udiErrorId             : UDINT;
  sSoftwareVersionLabel  : STRING;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
sSoftwareVersionLabel	STRING	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang eine Beschreibung (maximal 32 Zeichen) der Softwareversion des DMX-Gerätes ausgegeben. Der Inhalt wird vom DMX-Gerätehersteller festgelegt.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.5.7 FB_DMXSetDeviceLabel



Dieser Funktionsbaustein schreibt in das DMX-Gerät einen Beschreibungstext. Mit dem Baustein FB_DMXGetDeviceLabel() [▶ 37] kann der Text ausgelesen werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  sDeviceLabel              : STRING := '';
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
sDeviceLabel	STRING	Neuer Beschreibungstext für das DMX-Gerät.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

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

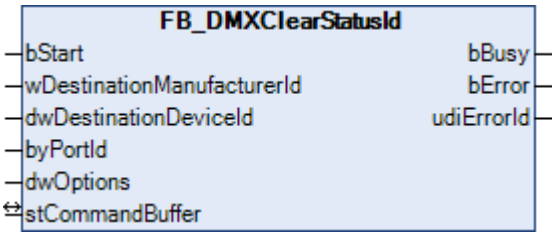
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.6 Queued and Status Messages

4.1.2.6.1 FB_DMXClearStatusId



Dieser Funktionsbaustein löscht den Nachrichtenbuffer im DMX-Gerät.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

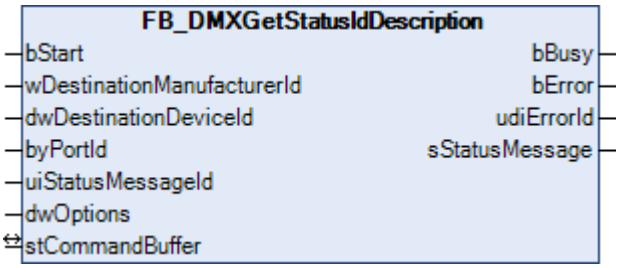
```
VAR_OUTPUT
  bBusy         : BOOL;
  bError        : BOOL;
  udiErrorId    : UDINT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.6.2 FB_DMXGetStatusIdDescription



Dieser Funktionsbaustein liest den Text einer bestimmten Status-Id aus dem DMX-Gerät aus.

RDM definiert einige Standardmeldungen. Jede dieser Meldungen hat eine eindeutige Status-Id. Der zugehörige Text kann mit diesem Baustein aus dem DMX-Gerät ausgelesen werden.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  uiStatusMessageId         : UINT := 1;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
uiStatusMessageId	UINT	Status-Id zu dem der Text ausgelesen werden soll.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

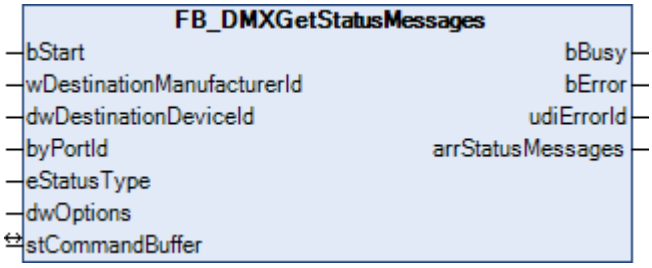
```
VAR_OUTPUT
  bBusy                 : BOOL;
  bError                : BOOL;
  udiErrorId            : UDINT;
  sStatusMessage        : STRING;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
sStatusMessage	STRING	Statusmeldung

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.6.3 FB_DMXGetStatusMessages



Dieser Funktionsbaustein sammelt die Status- oder Fehlerinformation von meinem DMX-Gerät.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  eStatusType               : E_DMXStatusType := eDMXStatusTypeNone;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
eStatusType	E_DMXStatusType [▶ 70]	Statustyp
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

```
VAR_OUTPUT
  bBusy             : BOOL;
  bError            : BOOL;
  udiErrorId        : UDINT;
  arrStatusMessages : ARRAY [0..24] OF ST_DMXStatusMessage;
END_VAR
```

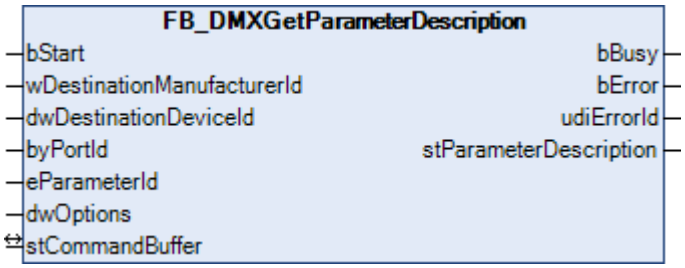
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
arrStatusMessages	ARRAY OF ST_DMXStatusMessage [▶ 74]	Wenn die Ausführung des Befehls abgeschlossen wurde (*bBusy* ist FALSE), dann stehen alle Informationen zum Status/Fehler an diesem Ausgang an.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.7 RDM Information Messages

4.1.2.7.1 FB_DMXGetParameterDescription



Dieser Funktionsbaustein fragt die Definition von herstellerspezifischen PIDs ab.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId     : DWORD;
  byPortId                  : BYTE;
  eParameterId              : E_DMXParameterId;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
eParameterId	E_DMXParameterId [▶ 65]	Angeforderte herstellerspezifische PID.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

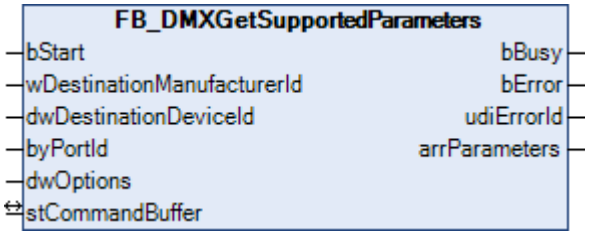
```
VAR_OUTPUT
  bBusy                  : BOOL;
  bError                 : BOOL;
  udiErrorId             : UDINT;
  stParameterDescription : ST_DMXParameterDescription;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
stParameterDescription	ST_DMXParameterDescription [▶ 72]	Wenn die Ausführung des Befehls abgeschlossen wurde (*bBusy* ist FALSE), dann stehen die Informationen zum PID an diesem Ausgang an.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.7.2 FB_DMXGetSupportedParameters



Dieser Funktionsbaustein fragt alle unterstützen Parameter von einem DMX-Gerät ab.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

```
VAR_OUTPUT
  bBusy                  : BOOL;
  bError                 : BOOL;
  udiErrorId             : UDINT;
  arrParameters          : ARRAY [0..114] OF E_DMXParameterId;
END_VAR
```

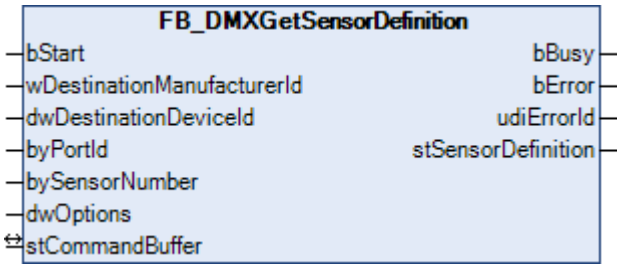
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
arrParameters	ARRAY OF E_DMXParameterId [▶ 65]	Wenn die Ausführung des Befehls abgeschlossen wurde (*bBusy* ist FALSE), dann stehen alle unterstützen Parameter für das DMX-Gerät an diesem Ausgang an.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.8 Sensor Parameter Messages

4.1.2.8.1 FB_DMXGetSensorDefinition



Dieser Funktionsbaustein fragt die Definition eines bestimmten Sensors ab.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId     : DWORD;
  byPortId                  : BYTE;
  bySensorNumber            : BYTE := 0;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
bySensorNumber	BYTE	DMX512 Sensornummer (0 - 254).
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

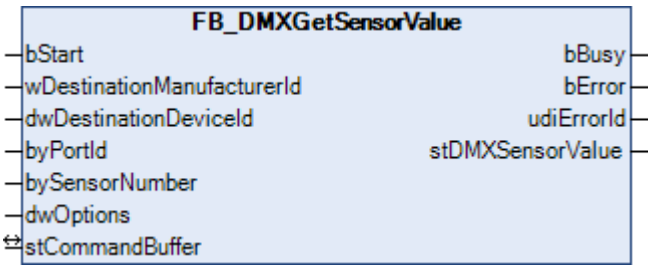
```
VAR_OUTPUT
  bBusy                : BOOL;
  bError               : BOOL;
  udiErrorId           : UDINT;
  stSensorDefinition   : ST_DMXSensorDefinition;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
stSensorDefinition	ST_DMXSensorDefinition [▶ 73]	Wenn die Ausführung des Befehls abgeschlossen wurde (*bBusy* ist FALSE), dann steht die Definition des Sensors an diesem Ausgang an.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.8.2 FB_DMXGetSensorValue



Dieser Funktionsbaustein fragt den aktuellen Wert eines Sensors ab.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId     : DWORD;
  byPortId                  : BYTE;
  bySensorNumber            : BYTE := 0;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät.
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät.
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
bySensorNumber	BYTE	DMX512 Sensornummer (0 - 254).
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

```
VAR_OUTPUT
  bBusy                 : BOOL;
  bError                : BOOL;
  udiErrorId            : UDINT;
  stDMXSensorValue      : ST_DMXSensorValue;
END_VAR
```

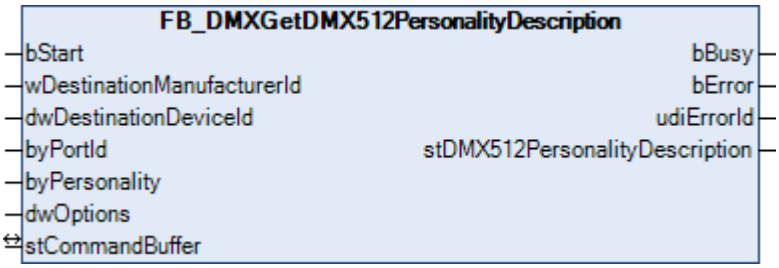
Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
stDMXSensorValue	ST_DMXSensorValue [▶ 74]	Struktur mit Informationen über den aktuellen Zustand des Sensors.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.9 Setup Messages

4.1.2.9.1 FB_DMXGetDMX512PersonalityDescription



Dieser Funktionsbaustein liest aus dem DMX-Gerät weitere Informationen einer Personality aus. Manche DMX-Geräte unterstützen sogenannte Personalities. Ein verändern der Personality kann Einfluss auf bestimmte RDM-Parameter haben.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  byPersonality             : BYTE := 0;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
byPersonality	BYTE	Die Personality dessen Informationen abgefragt werden.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

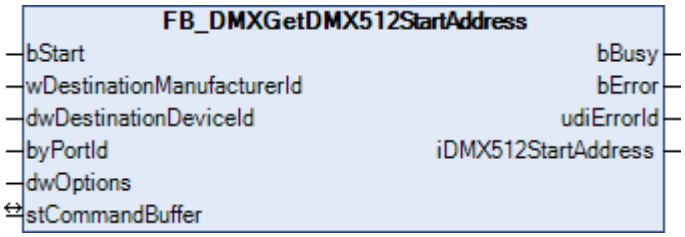
```
VAR_OUTPUT
  bBusy                          : BOOL;
  bError                         : BOOL;
  udiErrorId                     : UDINT;
  stDMX512PersonalityDescription : ST_DMX512PersonalityDescription;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
stDMX512PersonalityDescription	ST_DMX512PersonalityDescription [▶ 70]	Struktur mit Informationen der Personality.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.9.2 FB_DMXGetDMX512StartAddress



Dieser Funktionsbaustein erfragt die DMX512-Startadresse. Diese liegt im Bereich von 1 bis 512. Belegt das DMX-Gerät keinen DMX-Slot, so ist die DMX512-Startadresse 0xFFFF (65535). Jedes Untergerät (Sub-Device) und das Hauptgerät (Root-Device) belegen unterschiedliche DMX512-Startadressen.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError*, *udiErrorId* und *iDMX512StartAddress* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

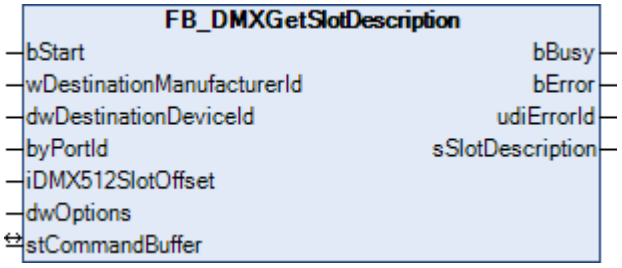
```
VAR_OUTPUT
  bBusy                 : BOOL;
  bError                : BOOL;
  udiErrorId            : UDINT;
  iDMX512StartAddress   : INT;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
iDMX512StartAddress	INT	Ist die Abarbeitung des Befehls abgeschlossen (*bBusy* ist FALSE), so wird an diesem Ausgang die DMX512-Startadresse des DMX-Gerätes ausgegeben.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.9.3 FB_DMXGetSlotDescription



Dieser Funktionsbaustein fragt die Textbeschreibung für Slot Offsets ab.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  iDMX512SlotOffset         : INT := 0;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
iDMX512SlotOffset	INT	DMX512 Slot Offset (0 - 511).
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

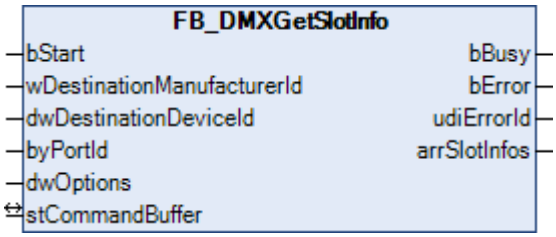
```
VAR_OUTPUT
  bBusy                 : BOOL;
  bError                : BOOL;
  udiErrorId            : UDINT;
  sSlotDescription      : STRING;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
sSlotDescription	STRING	Wenn die Ausführung des Befehls abgeschlossen wurde (*bBusy* ist FALSE), dann steht die Beschreibung (maximal 32 Zeichen) vom Slot an diesem Ausgang an.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.9.4 FB_DMXGetSlotInfo



Dieser Funktionsbaustein fragt die Basisinformationen zur Funktionalität der DMX512 Slots eines DMX-Gerätes ab.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
  byPortId                  : BYTE;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

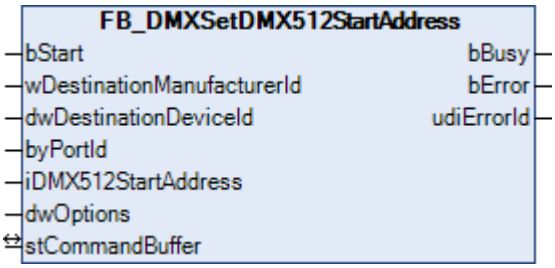
```
VAR_OUTPUT
  bBusy             : BOOL;
  bError            : BOOL;
  udiErrorId        : UDINT;
  arrSlotInfos      : ARRAY [0..45] OF ST_DMXSlotInfo;
END_VAR
```

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).
arrSlotInfos	ARRAY OF ST_DMXSlotInfo [▶ 74]	Wenn die Ausführung des Befehls abgeschlossen wurde (*bBusy* ist FALSE), dann stehen alle relevanten Informationen der DMX512 Slots als Array an diesem Ausgang an.

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.2.9.5 FB_DMXSetDMX512StartAddress



Dieser Funktionsbaustein setzt die DMX512-Startadresse. Diese liegt im Bereich von 1 bis 512. Jedes Untergerät (Sub-Device) und das Hauptgerät (Root-Device) belegen unterschiedliche DMX512-Startadressen.

Durch eine positive Flanke an dem Eingang *bStart* wird der Baustein gestartet und der Ausgang *bBusy* geht auf TRUE. Die Eingänge *wDestinationManufacturerId* und *dwDestinationDeviceId* adressieren das DMX-Gerät. Der Eingang *byPortId* definiert den Kanal innerhalb des adressierten DMX-Gerätes. Ist die Ausführung des Befehls abgeschlossen, geht der Ausgang *bBusy* wieder auf FALSE. Die Ausgänge *bError* und *udiErrorId* können jetzt ausgewertet werden. Solange der Baustein aktiv ist (*bBusy* ist TRUE) werden weitere positive Flanken am Eingang *bStart* ignoriert.

Eingänge

```
VAR_INPUT
  bStart                    : BOOL;
  wDestinationManufacturerId : WORD;
  dwDestinationDeviceId      : DWORD;
```

```
  byPortId                  : BYTE;
  iDMX512Startadresse       : INT;
  dwOptions                 : DWORD := 0;
END_VAR
```

Name	Typ	Beschreibung
bStart	BOOL	Über eine positive Flanke an diesem Eingang wird der Baustein gestartet.
wDestinationManufacturerId	WORD	Eindeutige Hersteller-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
dwDestinationDeviceId	DWORD	Eindeutige Geräte-Id vom DMX-Gerät (Details siehe DMX-Geräteadresse).
byPortId	BYTE	Kanal innerhalb des adressierten DMX-Gerätes. Untergeräte (Sub-Devices) werden durch die Port-Id adressiert. Das Hauptgerät (Root-Device) hat immer die Port-Id 0.
iDMX512StartAddress	INT	DMX512-Startadresse (1 bis 512).
dwOptions	DWORD	Optionen (wird derzeit nicht benutzt).

Ein-/Ausgänge

```
VAR_IN_OUT
  stCommandBuffer : ST_DMXCommandBuffer;
END_VAR
```

Name	Typ	Beschreibung
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Verweis auf die Struktur zur Kommunikation (Puffer) mit dem Baustein FB_EL6851Communication() [▶ 15]

Ausgänge

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

Name	Typ	Beschreibung
bBusy	BOOL	Bei der Aktivierung des Bausteins wird der Ausgang gesetzt und bleibt so lange aktiv, bis der Befehl abgearbeitet wurde.
bError	BOOL	Dieser Ausgang wird auf TRUE geschaltet, wenn bei der Ausführung eines Befehls ein Fehler aufgetreten ist. Der befehlsspezifische Fehlercode ist in *udiErrorId* enthalten. Nur gültig, wenn *bBusy* auf FALSE ist.
udiErrorId	UDINT	Enthält den befehlsspezifischen Fehlercode des zuletzt ausgeführten Befehls. Nur gültig, wenn *bBusy* auf FALSE ist (siehe Fehlercodes [▶ 62]).

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.1.3 Fehlercodes

Wert (hex)	Wert (dez)	Beschreibung
0x0000	0	Kein Fehler.
0x8001	32769	Keine Antwort von der DMX-Klemme.
0x8002	32770	Keine Antwort vom DMX-Gerät.
0x8003	32771	Überlauf Kommunikationspuffer.
0x8004	32772	Keine Antwort von dem Kommunikationsbaustein.
0x8005	32773	Parameter *byPortId* liegt außerhalb des gültigen Bereichs.
0x8006	32774	Fehler Checksumme.
0x8007	32775	Parameter *eResetDeviceType* liegt außerhalb des gültigen Bereichs.
0x8008	32776	Zeitüberschreitung.
0x8009	32777	Parameter *uliLowerBoundUID* ist größer als *uliUpperBoundUID.*
0x800A	32778	Es können keine RDM-Befehle versendet werden, da sich die Klemme im Cycle-Mode befindet.
0x800B	32779	Parameter *iDMX512StartAddress* liegt außerhalb des gültigen Bereichs (1-512).
0x800C	32780	Fehler beim Setzen der DMX512 Start Adresse.
0x800D	32781	Es können keine Prozessdaten versendet werden, da sich die Klemme nicht im Cycle-Mode befindet.
0x800E	32782	Es wurde ein RDM-Telegramm mit der Datenlänge 0 empfangen.
0x800F	32783	RDM-Response: Antwort vom RDM-Telegramm ist ungültig.
0x8010	32784	RDM-Response: Der RDM-Befehl ist nicht im DMX-Gerät implementiert.
0x8011	32785	RDM-Response: Das DMX-Gerät kann den gesendeten RDM-Befehl nicht auswerten, da diese nicht richtig formatiert sind.
0x8012	32786	RDM-Response: Das DMX-Gerät kann auf den RDM-Befehl nicht richtig reagieren, da ein interner Hardwarefehler im DMX-Gerät vorliegt.
0x8013	32787	RDM-Response: Der Proxy ist nicht der RDM-Linienmaster und kann deshalb nicht auf den RDM-Befehl reagieren.
0x8014	32788	RDM-Response: RDM-Schreibbefehle sind im DMX-Gerät derzeit blockiert.
0x8015	32789	RDM-Response: Die *Command Class* (CC) ist nicht korrekt. Evtl. wurde ein Schreibbefehl versendet, obwohl nur Lesebefehle erlaubt sind.
0x8016	32790	RDM-Response: Der Parameter, der mit dem RDM-Befehl versendet wurde, liegt ausserhalb des gültigen Bereichs oder ist ungültig.
0x8017	32791	RDM-Response: interner Bufferüberlauf im DMX-Gerät.
0x8018	32792	RDM-Response: Der RDM-Befehl konnte nicht vollständig empfangen werden, da nicht ausreichend Speicher im DMX-Gerät vorhanden ist.
0x8019	32793	RDM-Response: *Sub Device* (SD) ist nicht korrekt. Evtl. liegt der Wert ausserhalb des gültigen Bereichs oder ist ungültig.
0x801A	32794	Parameter *iDMX512SlotOffset* liegt außerhalb des gültigen Bereichs (0-511).
0x801B	32795	Parameter *bySensorNumber* liegt außerhalb des gültigen Bereichs (0-254).
0x801C	32796	RDM-Response: Das Feld *Parameter Data* (PD) ist zu lang. Es können nicht alle Daten der Rückantwort empfangen werden. Hierzu muss der Baustein FB_EL6851CommunicationEx() [▶ 18] eingesetzt werden.

Wert (hex)	Wert (dez)	Beschreibung
0x801D	32797	Die ADS-Adresse für den Zugriff auf die PDOs ist ungültig. Wurde die Struktur *AdsAddr* der KL6851 auf die entsprechende SPS-Variable gemappt?
0x801E	32798	Beim Lesezugriff auf die PDOs ist ein ADS-Fehler aufgetreten.

4.2 DUTs

4.2.1 Enums

4.2.1.1 E_DMXCommandClass

```
TYPE E_DMXCommandClass :
(
  eDMXCommandClassNotDefined               := 16#0000,
  eDMXCommandClassDiscoveryCommand         := 16#0010,
  eDMXCommandClassDiscoveryCommandResponse := 16#0011,
  eDMXCommandClassGetCommand               := 16#0020,
  eDMXCommandClassGetCommandResponse       := 16#0021,
  eDMXCommandClassSetCommand               := 16#0030,
  eDMXCommandClassSetCommandResponse       := 16#0031
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.2 E_DMXDataType

```
TYPE E_DMXDataType :
(
  eDMXDataTypeNotDefined    := 16#0000,
  eDMXDataTypeBitField      := 16#0001,
  eDMXDataTypeASCII         := 16#0002,
  eDMXDataTypeUnsignedByte  := 16#0003,
  eDMXDataTypeSignedByte    := 16#0004,
  eDMXDataTypeUnsignedWord  := 16#0005,
  eDMXDataTypeSignedWord    := 16#0006,
  eDMXDataTypeUnsignedDWord := 16#0007,
  eDMXDataTypeSignedDWord   := 16#0008
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.3 E_DMXLampOnMode

```
TYPE E_DMXLampOnMode :
(
  eDMXLampOnModeOff      := 0,
  eDMXLampOnModeDMX      := 1,
  eDMXLampOnModeOn       := 2,
  eDMXLampOnModeAfterCal := 3
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.4 E_DMXParameterDescriptionCommandClass

```
TYPE E_DMXParameterDescriptionCommandClass :
(
  eDMXParameterDescriptionCommandClassGet    := 16#0001,
  eDMXParameterDescriptionCommandClassSet    := 16#0002,
  eDMXParameterDescriptionCommandClassGetSet := 16#0003
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.5 E_DMXParameterId

```
TYPE E_DMXParameterId :
(
  eDMXParameterIdNone                          := 16#0000,
  eDMXParameterIdDiscUniqueBranch              := 16#0001,
  eDMXParameterIdDiscMute                      := 16#0002,
  eDMXParameterIdDiscUnMute                    := 16#0003,
  eDMXParameterIdProxiedDevices                := 16#0010,
  eDMXParameterIdProxiedDeviceCount            := 16#0011,
  eDMXParameterIdCommsStatus                   := 16#0015,
  eDMXParameterIdQueuedMessage                 := 16#0020,
  eDMXParameterIdStatusMessages                := 16#0030,
  eDMXParameterIdStatusIdDescription           := 16#0031,
  eDMXParameterIdClearStatusId                 := 16#0032,
  eDMXParameterIdSubDeviceStatusReportThreshold := 16#0033,
  eDMXParameterIdSupportedParamaters           := 16#0050,
  eDMXParameterIdParameterDescription          := 16#0051,
  eDMXParameterIdDeviceInfo                    := 16#0060,
  eDMXParameterIdProductDetailIdList           := 16#0070,
  eDMXParameterIdDeviceModelDescription        := 16#0080,
  eDMXParameterIdManufacturerLabel             := 16#0081,
  eDMXParameterIdDeviceLabel                   := 16#0082,
  eDMXParameterIdFactoryDefaults               := 16#0090,
  eDMXParameterIdLanguageCapabilities          := 16#00A0,
  eDMXParameterIdLanguage                      := 16#00B0,
  eDMXParameterIdSoftwareVersionLabel          := 16#00C0,
  eDMXParameterIdBootSoftwareVersionId         := 16#00C1,
  eDMXParameterIdBootSoftwareVersionLabel      := 16#00C2,
  eDMXParameterIdDMXPersonality                := 16#00E0,
  eDMXParameterIdDMXPersonalityDescription     := 16#00E1,
  eDMXParameterIdDMXStartAddress               := 16#00F0,
  eDMXParameterIdSlotInfo                      := 16#0120,
  eDMXParameterIdSlotDescription               := 16#0121,
  eDMXParameterIdDefaultSlotValue              := 16#0122,
  eDMXParameterIdSensorDefinition              := 16#0200,
  eDMXParameterIdSensorValue                   := 16#0201,
  eDMXParameterIdRecordSensors                 := 16#0202,
  eDMXParameterIdDeviceHours                   := 16#0400,
  eDMXParameterIdLampHours                     := 16#0401,
  eDMXParameterIdLampStrikes                   := 16#0402,
  eDMXParameterIdLampState                     := 16#0403,
  eDMXParameterIdLampOnMode                    := 16#0404,
  eDMXParameterIdDevicePowerCycles             := 16#0405,
  eDMXParameterIdDisplayInvert                 := 16#0500,
  eDMXParameterIdDisplayLevel                  := 16#0501,
  eDMXParameterIdPanInvert                     := 16#0600,
  eDMXParameterIdTiltInvert                    := 16#0601,
  eDMXParameterIdPanTiltSwap                   := 16#0602,
  eDMXParameterIdRealTimeClock                 := 16#0603,
  eDMXParameterIdIdentifyDevice                := 16#1000,
  eDMXParameterIdResetDevice                   := 16#1001,
  eDMXParameterIdPowerState                    := 16#1010,
  eDMXParameterIdPerformSelftest               := 16#1020,
  eDMXParameterIdSelfTestDescription           := 16#1021,
  eDMXParameterIdCapturePreset                 := 16#1030,
  eDMXParameterIdPresetPlayBack                := 16#1031
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.6 E_DMXProductDetail

```
TYPE E_DMXProductDetail :
(
  eDMXProductDetailNotDeclared          := 16#0000,
  eDMXProductDetailArc                  := 16#0001,
  eDMXProductDetailMetalHalide          := 16#0002,
  eDMXProductDetailIncandescent         := 16#0003,
  eDMXProductDetailLED                  := 16#0004,
  eDMXProductDetailFluroescent          := 16#0005,
  eDMXProductDetailColdcathode          := 16#0006,
  eDMXProductDetailElectroluminescent   := 16#0007,
  eDMXProductDetailLaser                := 16#0008,
  eDMXProductDetailFlashtube            := 16#0009,
  eDMXProductDetailColorscroller        := 16#0100,
  eDMXProductDetailColorwheel           := 16#0101,
  eDMXProductDetailColorchange          := 16#0102,
  eDMXProductDetailIrisDouser           := 16#0103,
  eDMXProductDetailDimmingShutter       := 16#0104,
  eDMXProductDetailProfileShutter       := 16#0105,
  eDMXProductDetailBarndoorShutter      := 16#0106,
  eDMXProductDetailEffectsDisc          := 16#0107,
  eDMXProductDetailGoboRotator          := 16#0108,
  eDMXProductDetailVideo                := 16#0200,
  eDMXProductDetailSlide                := 16#0201,
  eDMXProductDetailFilm                 := 16#0202,
  eDMXProductDetailOilwheel             := 16#0203,
  eDMXProductDetailLCDGate              := 16#0204,
  eDMXProductDetailFoggerGlycol         := 16#0300,
  eDMXProductDetailFoggerMineraloil     := 16#0301,
  eDMXProductDetailFoggerWater          := 16#0302,
  eDMXProductDetailCO2                  := 16#0303,
  eDMXProductDetailLN2                  := 16#0304,
  eDMXProductDetailBubble               := 16#0305,
  eDMXProductDetailFlamePropane         := 16#0306,
  eDMXProductDetailFlameOther           := 16#0307,
  eDMXProductDetailOlefactoryStimulator := 16#0308,
  eDMXProductDetailSnow                 := 16#0309,
  eDMXProductDetailWaterJet             := 16#030A,
  eDMXProductDetailWind                 := 16#030B,
  eDMXProductDetailConfetti             := 16#030C,
  eDMXProductDetailHazard               := 16#030D,
  eDMXProductDetailPhaseControl         := 16#0400,
  eDMXProductDetailReversePhaseControl  := 16#0401,
  eDMXProductDetailSine                 := 16#0402,
  eDMXProductDetailPWM                  := 16#0403,
  eDMXProductDetailDC                   := 16#0404,
  eDMXProductDetailHfballast            := 16#0405,
  eDMXProductDetailHfhvNeonballast      := 16#0406,
  eDMXProductDetailHfhvEl               := 16#0407,
  eDMXProductDetailMhrBallast           := 16#0408,
  eDMXProductDetailBitangleModulation   := 16#0409,
  eDMXProductDetailFrequencyModulation  := 16#040A,
  eDMXProductDetailHighfrequency12V     := 16#040B,
  eDMXProductDetailRelayMechanical      := 16#040C,
  eDMXProductDetailRelayElectronic      := 16#040D,
  eDMXProductDetailSwitchElectronic     := 16#040E,
  eDMXProductDetailContactor            := 16#040F,
  eDMXProductDetailMirrorballRotator    := 16#0500,
  eDMXProductDetailOtherRotator         := 16#0501,
  eDMXProductDetailKabukiDrop           := 16#0502,
  eDMXProductDetailCurtain              := 16#0503,
  eDMXProductDetailLineset              := 16#0504,
  eDMXProductDetailMotorControl         := 16#0505,
  eDMXProductDetailDamperControl        := 16#0506,
  eDMXProductDetailSplitter             := 16#0600,
  eDMXProductDetailEthernetNode         := 16#0601,
  eDMXProductDetailMerge                := 16#0602,
  eDMXProductDetailDatapatch            := 16#0603,
  eDMXProductDetailWirelessLink         := 16#0604,
  eDMXProductDetailProtocolConvertor    := 16#0701,
  eDMXProductDetailAnalogDemultiplex    := 16#0702,
```

```
  eDMXProductDetailAnalogMultiplex      := 16#0703,
  eDMXProductDetailSwitchPanel          := 16#0704,
  eDMXProductDetailRouter               := 16#0800,
  eDMXProductDetailFader                := 16#0801,
  eDMXProductDetailMixer                := 16#0802,
  eDMXProductDetailChangeoverManual     := 16#0900,
  eDMXProductDetailChangeoverAuto       := 16#0901,
  eDMXProductDetailTest                 := 16#0902,
  eDMXProductDetailGfiRcd               := 16#0A00,
  eDMXProductDetailBattery              := 16#0A01,
  eDMXProductDetailControllableBreaker  := 16#0A02,
  eDMXProductDetailOther                := 16#7FFF
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.7 E_DMXResetDeviceType

```
TYPE E_DMXResetDeviceType :
(
  eDMXResetDeviceTypeWarm := 1,
  eDMXResetDeviceTypeCold := 255
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.8 E_DMXResponseType

```
TYPE E_DMXResponseType :
(
  eDMXResponseTypeAck         := 16#0000,
  eDMXResponseTypeAckTimer    := 16#0001,
  eDMXResponseTypeNackReason  := 16#0002,
  eDMXResponseTypeAckOverflow := 16#0003
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.9 E_DMXSensorType

```
TYPE E_DMXSensorType :
(
  eDMXSensorTypeTemperature       := 16#00,
  eDMXSensorTypeVoltage           := 16#01,
  eDMXSensorTypeCurrent           := 16#02,
  eDMXSensorTypeFrequency         := 16#03,
  eDMXSensorTypeResistance        := 16#04,
  eDMXSensorTypePower             := 16#05,
  eDMXSensorTypeMass              := 16#06,
  eDMXSensorTypeLength            := 16#07,
  eDMXSensorTypeArea              := 16#08,
  eDMXSensorTypeVolume            := 16#09,
  eDMXSensorTypeDensity           := 16#0A,
  eDMXSensorTypeVelocity          := 16#0B,
  eDMXSensorTypeAcceleration      := 16#0C,
  eDMXSensorTypeForce             := 16#0D,
  eDMXSensorTypeEnergy            := 16#0E,
  eDMXSensorTypePressure          := 16#0F,
  eDMXSensorTypeTime              := 16#10,
```

```
  eDMXSensorTypeAngle             := 16#11,
  eDMXSensorTypePositionX         := 16#12,
  eDMXSensorTypePositionY         := 16#13,
  eDMXSensorTypePositionZ         := 16#14,
  eDMXSensorTypeAngularVelocity   := 16#15,
  eDMXSensorTypeLuminousIntensity := 16#16,
  eDMXSensorTypeLuminousFlux      := 16#17,
  eDMXSensorTypeIlluminance       := 16#18,
  eDMXSensorTypeChrominanceRed    := 16#19,
  eDMXSensorTypeChrominanceGreen  := 16#1A,
  eDMXSensorTypeChrominanceBlue   := 16#1B,
  eDMXSensorTypeContacts          := 16#1C,
  eDMXSensorTypeMemory            := 16#1D,
  eDMXSensorTypeItems             := 16#1E,
  eDMXSensorTypeHumidity          := 16#1F,
  eDMXSensorTypeCounter16Bit      := 16#20,
  eDMXSensorTypeOther             := 16#7F
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.10 E_DMXSensorUnit

```
TYPE E_DMXSensorUnit :
(
  eDMXSensorUnitNone                    := 16#00,
  eDMXSensorUnitCentigrade              := 16#01,
  eDMXSensorUnitVoltsDC                 := 16#02,
  eDMXSensorUnitVoltsACPeak             := 16#03,
  eDMXSensorUnitVoltsACRms              := 16#04,
  eDMXSensorUnitAmpereDC                := 16#05,
  eDMXSensorUnitAmpereACPeak            := 16#06,
  eDMXSensorUnitAmpereACRms             := 16#07,
  eDMXSensorUnitHertz                   := 16#08,
  eDMXSensorUnitOhm                     := 16#09,
  eDMXSensorUnitWatt                    := 16#0A,
  eDMXSensorUnitKilogram                := 16#0B,
  eDMXSensorUnitMeters                  := 16#0C,
  eDMXSensorUnitMetersSquared           := 16#0D,
  eDMXSensorUnitMetersCubed             := 16#0E,
  eDMXSensorUnitKilogrammesPerMeterCubed := 16#0F,
  eDMXSensorUnitMetersPerSecond         := 16#10,
  eDMXSensorUnitMetersPerSecondSquared  := 16#11,
  eDMXSensorUnitNewton                  := 16#12,
  eDMXSensorUnitJoule                   := 16#13,
  eDMXSensorUnitPascal                  := 16#14,
  eDMXSensorUnitSecond                  := 16#15,
  eDMXSensorUnitDegree                  := 16#16,
  eDMXSensorUnitSteradian               := 16#17,
  eDMXSensorUnitCandela                 := 16#18,
  eDMXSensorUnitLumen                   := 16#19,
  eDMXSensorUnitLux                     := 16#1A,
  eDMXSensorUnitIre                     := 16#1B,
  eDMXSensorUnitByte                    := 16#1C
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.11 E_DMXSensorUnitPrefix

```
TYPE E_DMXSensorUnitPrefix :
(
  eDMXSensorUnitPrefixNone   := 16#00,
  eDMXSensorUnitPrefixDeci   := 16#01,
  eDMXSensorUnitPrefixCenti  := 16#02,
  eDMXSensorUnitPrefixMilli  := 16#03,
```

```
  eDMXSensorUnitPrefixMicro  := 16#04,
  eDMXSensorUnitPrefixNano   := 16#05,
  eDMXSensorUnitPrefixPico   := 16#06,
  eDMXSensorUnitPrefixFempto := 16#07,
  eDMXSensorUnitPrefixAtto   := 16#08,
  eDMXSensorUnitPrefixZepto  := 16#09,
  eDMXSensorUnitPrefixYocto  := 16#0A,
  eDMXSensorUnitPrefixDeca   := 16#11,
  eDMXSensorUnitPrefixHecto  := 16#12,
  eDMXSensorUnitPrefixKilo   := 16#13,
  eDMXSensorUnitPrefixMega   := 16#14,
  eDMXSensorUnitPrefixGiga   := 16#15,
  eDMXSensorUnitPrefixTerra  := 16#16,
  eDMXSensorUnitPrefixPeta   := 16#17,
  eDMXSensorUnitPrefixExa    := 16#18,
  eDMXSensorUnitPrefixZetta  := 16#19,
  eDMXSensorUnitPrefixYotta  := 16#1A
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.12 E_DMXSlotDefinition

```
TYPE E_DMXSlotDefinition :
(
  eDMXSlotDefinitionIntensity       := 16#0001,
  eDMXSlotDefinitionIntensityMaster := 16#0002,
  eDMXSlotDefinitionPan             := 16#0101,
  eDMXSlotDefinitionTilt            := 16#0102,
  eDMXSlotDefinitionColorWheel      := 16#0201,
  eDMXSlotDefinitionColorSubCyan    := 16#0202,
  eDMXSlotDefinitionColorSubYellow  := 16#0203,
  eDMXSlotDefinitionColorSubMagenta := 16#0204,
  eDMXSlotDefinitionColorAddRed     := 16#0205,
  eDMXSlotDefinitionColorAddGreen   := 16#0206,
  eDMXSlotDefinitionColorAddBlue    := 16#0207,
  eDMXSlotDefinitionColorCorrection := 16#0208,
  eDMXSlotDefinitionColorScroll     := 16#0209,
  eDMXSlotDefinitionColorSemaphore  := 16#0210,
  eDMXSlotDefinitionStaticGoboWheel := 16#0301,
  eDMXSlotDefinitionRotoGoboWheel   := 16#0302,
  eDMXSlotDefinitionPrismWheel      := 16#0303,
  eDMXSlotDefinitionEffectsWheel    := 16#0304,
  eDMXSlotDefinitionBeamSizeIris    := 16#0401,
  eDMXSlotDefinitionEdge            := 16#0402,
  eDMXSlotDefinitionFrost           := 16#0403,
  eDMXSlotDefinitionStrobe          := 16#0404,
  eDMXSlotDefinitionZoom            := 16#0405,
  eDMXSlotDefinitionFramingShutter  := 16#0406,
  eDMXSlotDefinitionShutterRotate   := 16#0407,
  eDMXSlotDefinitionDouser          := 16#0408,
  eDMXSlotDefinitionBarnDoor        := 16#0409,
  eDMXSlotDefinitionLampControl     := 16#0501,
  eDMXSlotDefinitionFixtureControl  := 16#0502,
  eDMXSlotDefinitionFixtureSpeed    := 16#0503,
  eDMXSlotDefinitionMacro           := 16#0504,
  eDMXSlotDefinitionUndefined       := 16#FFFF
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.13 E_DMXSlotType

```
TYPE E_DMXSlotType :
(
  eDMXSlotTypePrinary       := 0,
```

```
  eDMXSlotTypeSecFine        := 1,
  eDMXSlotTypeSecTiming      := 2,
  eDMXSlotTypeSecSpeed       := 3,
  eDMXSlotTypeSecControl     := 4,
  eDMXSlotTypeSecIndex       := 5,
  eDMXSlotTypeSecRotation    := 6,
  eDMXSlotTypeSecIndexRotate := 7,
  eDMXSlotTypeSecUndefined   := 255
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.1.14 E_DMXStatusType

```
TYPE E_DMXStatusType :
(
  eDMXStatusTypeNone             := 16#00,
  eDMXStatusTypeGetLastMessage   := 16#01,
  eDMXStatusTypeAdvisory         := 16#02,
  eDMXStatusTypeWarning          := 16#03,
  eDMXStatusTypeError            := 16#04,
  eDMXStatusTypeAdvisoryCleared  := 16#12,
  eDMXSensorTypeWarningCleared   := 16#13,
  eDMXSensorTypeErrorCleared     := 16#14
);
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2 Structures

4.2.2.1 ST_DMX512Personality

```
TYPE ST_DMX512Personality :
STRUCT
  byCurrentPersonality : BYTE;
  byTotalPersonalities : BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.2 ST_DMX512PersonalityDescription

```
TYPE ST_DMX512PersonalityDescription :
STRUCT
  iDMX512SlotsRequired : INT;
  sDescription         : STRING;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.3 ST_DMXCommandBuffer

```
TYPE ST_DMXCommandBuffer :
STRUCT
  arrMessageQueue     : ST_DMXMessageQueue;
  stResponseTable     : ST_DMXResponseTable;
  byTransactionNumber : BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.4 ST_DMXDeviceInfo

```
TYPE ST_DMXDeviceInfo :
STRUCT
  uliUID               : ST_DMXMac;
  stRDMProtocolVersion : ST_DMXRDMProtocolVersion;
  uiDeviceModelId      : UINT;
  stProductCategory    : ST_DMXProductCategory;
  udiSoftwareVersionId : UDINT;
  uiDMX512Footprint    : UINT;
  stDMX512Personality  : ST_DMX512Personality;
  uiDMX512StartAddress : UINT;
  uiSubDeviceCount     : UINT;
  bySensorCount        : BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.5 ST_DMXMac

```
TYPE ST_DMXMac :
STRUCT
  wHighPart    : WORD;      (* Manufacturer ID / Higher word *)
  dwLowPart    : DWORD;     (* Device ID / Lower double word *)
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.6 ST_DMXMessageQueue

```
TYPE ST_DMXMessageQueue :
STRUCT
  arrBuffer                    : ARRAY [1..20] OF ST_DMXMessageQueueItem;
  byBufferReadPointer          : BYTE;
  byBufferWritePointer         : BYTE;
  byBufferDemandCounter        : BYTE;
  byBufferMaximumDemandCounter : BYTE;
  uiBufferOverflowCounter      : UINT;
  bLockSemaphore               : BOOL;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.7 ST_DMXMessageQueueItem

```
TYPE ST_DMXMessageQueueItem :
STRUCT
  bEntryIsEngaged             : BOOL;
  byMessageLength             : BYTE;
  wDestinationManufacturerID  : WORD;
  dwDestinationDeviceID       : DWORD;
  byTransactionNumber         : BYTE;
  byPortID                    : BYTE;
  byMessageCount              : BYTE;
  wSubDevice                  : WORD;
  byCommandClass              : BYTE;
  wParameterID                : WORD;
  byParameterDataLength       : BYTE;
  arrParameterData            : ARRAY [0..255] OF BYTE;
  bWaitingForDMXSlaveResponse : BOOL;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.8 ST_DMXParameterDescription

```
TYPE ST_DMXParameterDescription :
STRUCT
  byParameterDataLength : BYTE;
  eDataType             : E_DMXDataType;
  ePDCommandClass       : E_DMXParameterDescriptionCommandClass;
  eType                 : E_DMXSensorType;
  eUnit                 : E_DMXSensorUnit;
  eUnitPrefix           : E_DMXSensorUnitPrefix;
  dwMinValidValue       : DWORD;
  dwMaxValidValue       : DWORD;
  dwDefaultValue        : DWORD;
  sDescription          : STRING;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.9 ST_DMXProductCategory

```
TYPE ST_DMXProductCategory :
STRUCT
  byCoarse : BYTE;
  byFine   : BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.10 ST_DMXRDMProtocolVersion

```
TYPE ST_DMXRDMProtocolVersion :
STRUCT
  byMajorVersion : BYTE;
  byMinorVersion : BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.11 ST_DMXResponseTable

```
TYPE ST_DMXResponseTable :
STRUCT
  arrResponseTable               : ARRAY [1..20] OF ST_DMXResponseTableItem;
  byResponseTableCounter         : BYTE;
  byResponseTableMaxCounter      : BYTE;
  uiResponseTableOverflowCounter : UINT;
  bLockSemaphore                 : BOOL;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.12 ST_DMXResponseTableItem

```
TYPE ST_DMXResponseTableItem :
STRUCT
  bEntryIsEngaged            : BOOL;
  uiErrorId                  : UINT;
  iErrorParameter            : INT;
  byMessageLength            : BYTE;
  wDestinationManufacturerID : WORD;
  dwDestinationDeviceID      : DWORD;
  wSourceManufacturerID      : WORD;
  dwSourceDeviceID           : DWORD;
  byTransactionNumber        : BYTE;
  byResponseType             : BYTE;
  byMessageCount             : BYTE;
  wSubDevice                 : WORD;
  byCommandClass             : BYTE;
  wParameterID               : WORD;
  byParameterDataLength      : BYTE;
  arrParameterData           : ARRAY [0..255] OF BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.13 ST_DMXSensorDefinition

```
TYPE ST_DMXSensorDefinition :
STRUCT
  eSensorType          : E_DMXSensorType;
  eSensorUnit          : E_DMXSensorUnit;
  eSensorUnitPrefix    : E_DMXSensorUnitPrefix;
  iRangeMinimumValue   : INT;
  iRangeMaximumValue   : INT;
  iNormalMinimumValue  : INT;
  iNormalMaximumValue  : INT;
  byRecordValueSupport : BYTE;
  sDescription         : STRING;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.14 ST_DMXSensorValue

```
TYPE ST_DMXSensorValue :
STRUCT
  iPresentValue         : INT;
  iLowestDetectedValue  : INT;
  iHighestDetectedValue : INT;
  iRecordedValue        : INT;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.15 ST_DMXSlotInfo

```
TYPE ST_DMXSlotInfo :
STRUCT
  bEntryIsValid   : BOOL;
  eSlotType       : E_DMXSlotType;
  eSlotDefinition : E_DMXSlotDefinition;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.16 ST_DMXStatusMessage

```
TYPE ST_DMXStatusMessage :
STRUCT
  bEntryIsValid    : BOOL;
  iSubDeviceId     : INT;
  eStatusType      : E_DMXStatusType;
  iStatusMessageId : INT;
  iDataValue01     : INT;
  iDataValue02     : INT;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.17 ST_EL6851AdsAddr

```
TYPE ST_EL6851AdsAddr :
STRUCT
  arrNetId : ARRAY [0..5] OF USINT;
  uiPort   : UINT;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.18 ST_EL6851InData

```
TYPE ST_EL6851InData :
STRUCT
  bTransmitAccepted  : BOOL;
  bReceiveToggle     : BOOL;
  bCyclicTxDDisabled : BOOL;
  bDefaultDataSent   : BOOL;
  bFrameSentToggle   : BOOL;
  bTxPDOToggle       : BOOL;
  wChannelLength     : WORD;
  byStartCode        : BYTE;
  byDummy            : BYTE;
  arrData            : ARRAY [1..64] OF BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.19 ST_EL6851InDataEx

```
TYPE ST_EL6851InDataEx :
STRUCT
  bTransmitAccepted  : BOOL;
  bReceiveToggle     : BOOL;
  bCyclicTxDDisabled : BOOL;
  bDefaultDataSent   : BOOL;
  bFrameSentToggle   : BOOL;
  bTxPDOToggle       : BOOL;
  wChannelLength     : WORD;
  byStartCode        : BYTE;
  byDummy1           : BYTE;
  arrData            : ARRAY [1..64] OF BYTE;
  bWcState           : BOOL;
  bInputToggle       : BOOL;
  uiState            : UINT;
  stAdsAddr          : ST_EL6851AdsAddr;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.2.2.20 ST_EL6851OutData

```
TYPE ST_EL6851OutData :
STRUCT
  bTransmitRequest  : BOOL;
  bDisableCyclicTxD : BOOL;
  bSendDefaultData  : BOOL;
  byDummy1          : BYTE;
  wChannelLength    : WORD;
  byStartCode       : BYTE;
  byDummy2          : BYTE;
  arrData           : ARRAY [1..512] OF BYTE;
END_STRUCT
END_TYPE
```

Voraussetzungen

Entwicklungsumgebung	erforderliche TC3 SPS-Bibliothek
TwinCAT ab v3.1.4020.14	Tc2_DMX ab v3.5.3.0

4.3 Integration in TwinCAT

4.3.1 EL6851 mit CX5120

Dieses Beispiel beschreibt, wie ein einfaches SPS-Programm für DMX in TwinCAT geschrieben werden kann und wie es mit der Hardware verknüpft wird. Es soll nach DMX-Geräten gesucht werden.

Beispiel: https://infosys.beckhoff.com/content/1031/tcplclib_tc2_dmx/Resources/6202185099.zip

Hardware

Einrichtung der Komponenten

Es wird folgende Hardware benötigt:

- 1x Embedded-PC CX5120
- 1x DMX-Masterklemme EL6851
- 1x Abschlusskappe EL9011

Richten Sie die Hardware, sowie die DMX-Komponenten wie in den entsprechenden Dokumentationen beschrieben ein.

Software

Erstellung des SPS-Programms

Erstellen Sie ein neues „TwinCAT XAE Project“ und legen Sie ein „Standard PLC Project“ an.

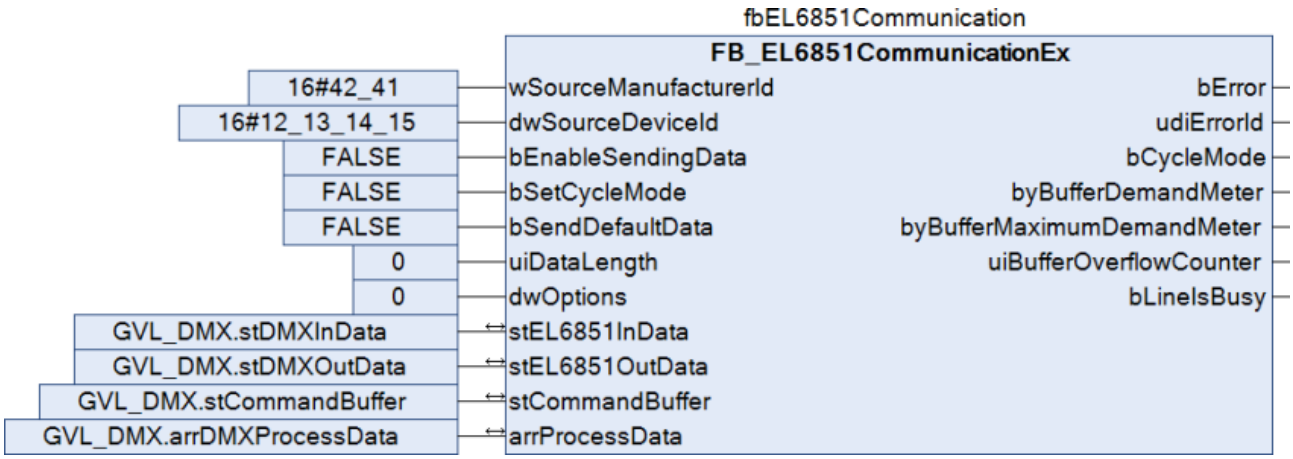
Fügen Sie im SPS-Projekt unter „References“ die Bibliothek Tc2_DMX hinzu.

Erzeugen Sie eine globale Variablenliste mit dem Namen GVL_DMX und legen Sie die folgenden Variablen an:

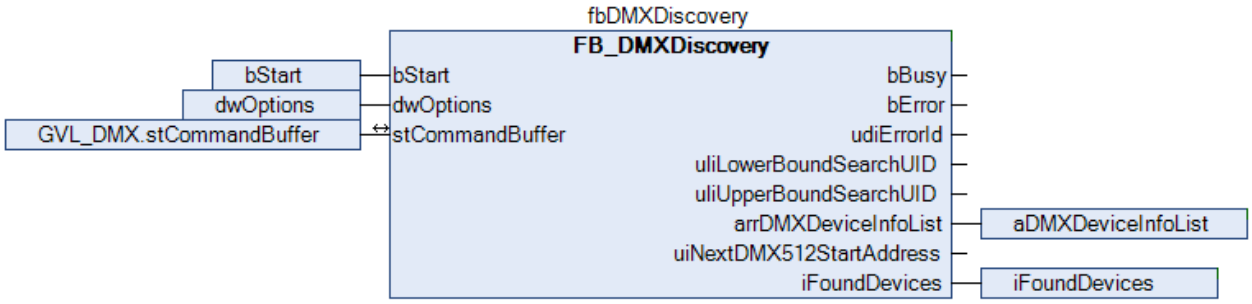
```
VAR_GLOBAL
    stEL6851InData      AT %I* : ST_EL6851InDataEx;
    stKL6851OutData     AT %Q* : ST_EL6851OutData;
    stCommandBuffer            : ST_DMXCommandBuffer;
    arrDMXProcessData          : ARRAY [1..512] OF BYTE;
END_VAR
```

Name	Typ	Beschreibung
stEL6851InData	ST_EL6851InDataEx [▶ 75]	Eingangsvariable für die DMX-Klemme.
stEL6851OutData	ST_EL6851OutData [▶ 75]	Ausgangsvariable für die DMX-Klemme.
stCommandBuffer	ST_DMXCommandBuffer [▶ 71]	Wird für die Kommunikation mit DMX benötigt.
arrDMXProcessData	BYTE	Über diese Variable werden dem Baustein die Daten übergeben, die zyklisch an die DMX-Geräte versendet werden sollen. Hierzu muss der CycleMode aktiv sein (siehe auch Eingang *bSetCycleMode*).

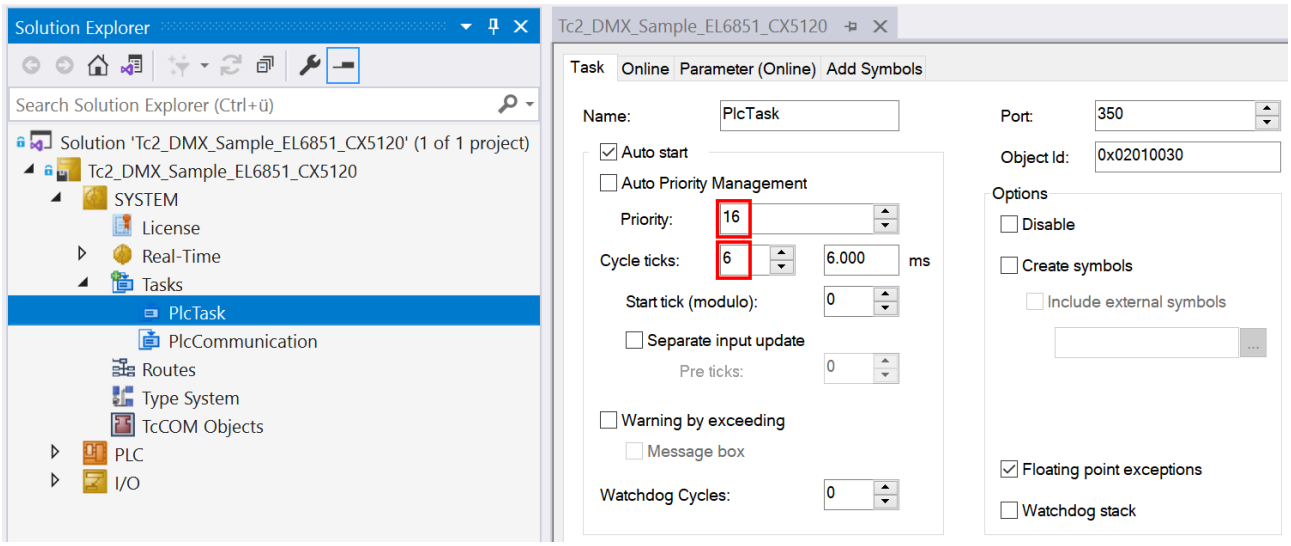
Legen Sie anschließend ein Programm (CFC) für die Hintergrundkommunikation mit DMX an. In diesem wird der Baustein FB_EL6851CommunicationEx [▶ 18] aufgerufen. Achten Sie beim Kommunikationsbaustein darauf, ihn mit *stEL6851InData*, *stEL6851OutData* und *stCommandBuffer* zu verknüpfen.



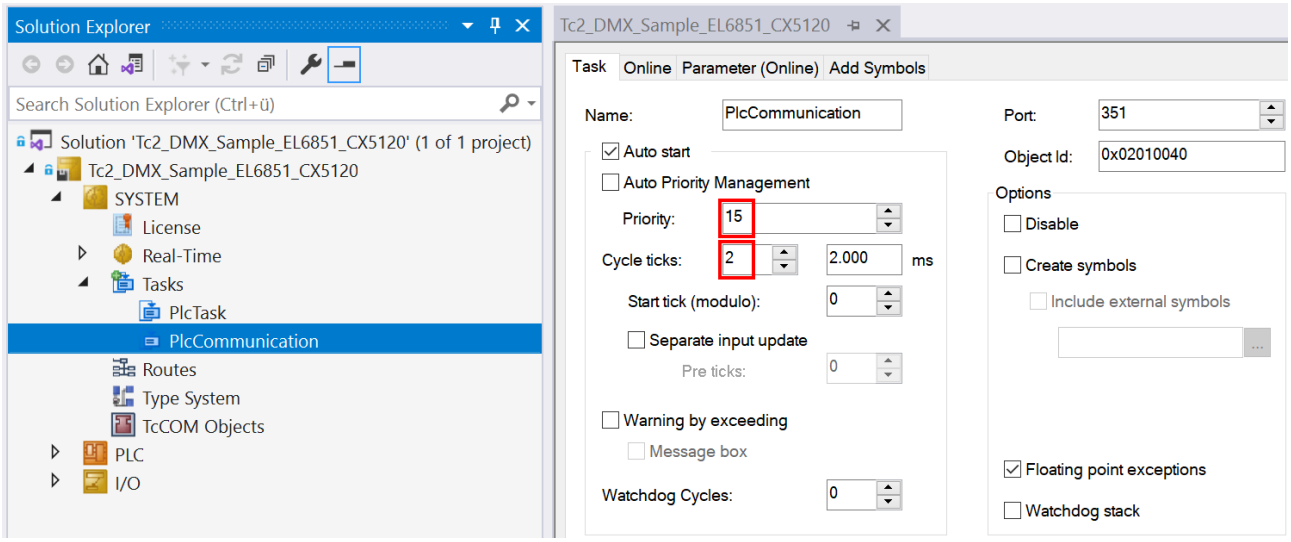
Legen Sie ein MAIN-Programm (CFC) an, in dem der BausteinFB_DMXDiscovery512 [▶ 11] aufgerufen wird. Der Eingang *stCommandBuffer* des Bausteins wird mit der globalen Variable *stCommandBuffer* verknüpft. Wenn die Variable *bStart* TRUE ist wird die Suche nach DMX-Geräten gestartet. Die Anzahl der gefundenen Geräte steht in der Variable *iFoundDevices* und zusätzliche Informationen dazu in *aDMXDeviceInfoList*.



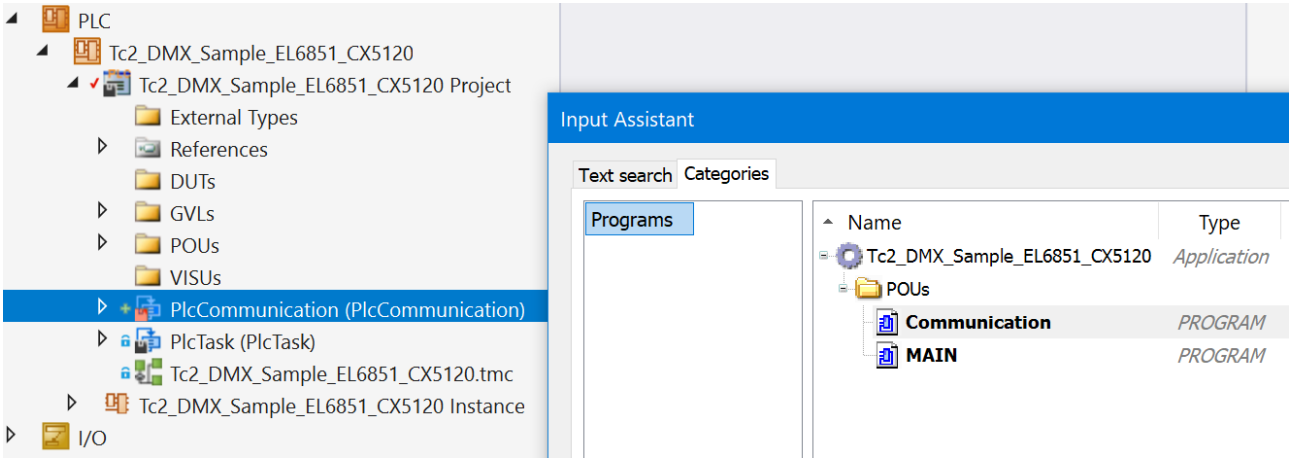
Navigieren Sie in den Bereich der Taskkonfiguration und konfigurieren die PlcTask. Exemplarisch erhält die Task die Priorität 16 und eine Zykluszeit von 6 ms.



Legen Sie eine weitere Task für die Hintergrundkommunikation an. Geben Sie dieser Task eine höhere Priorität (kleinere Zahl) und eine niedrigere Intervall-Zeit als der PlcTask.



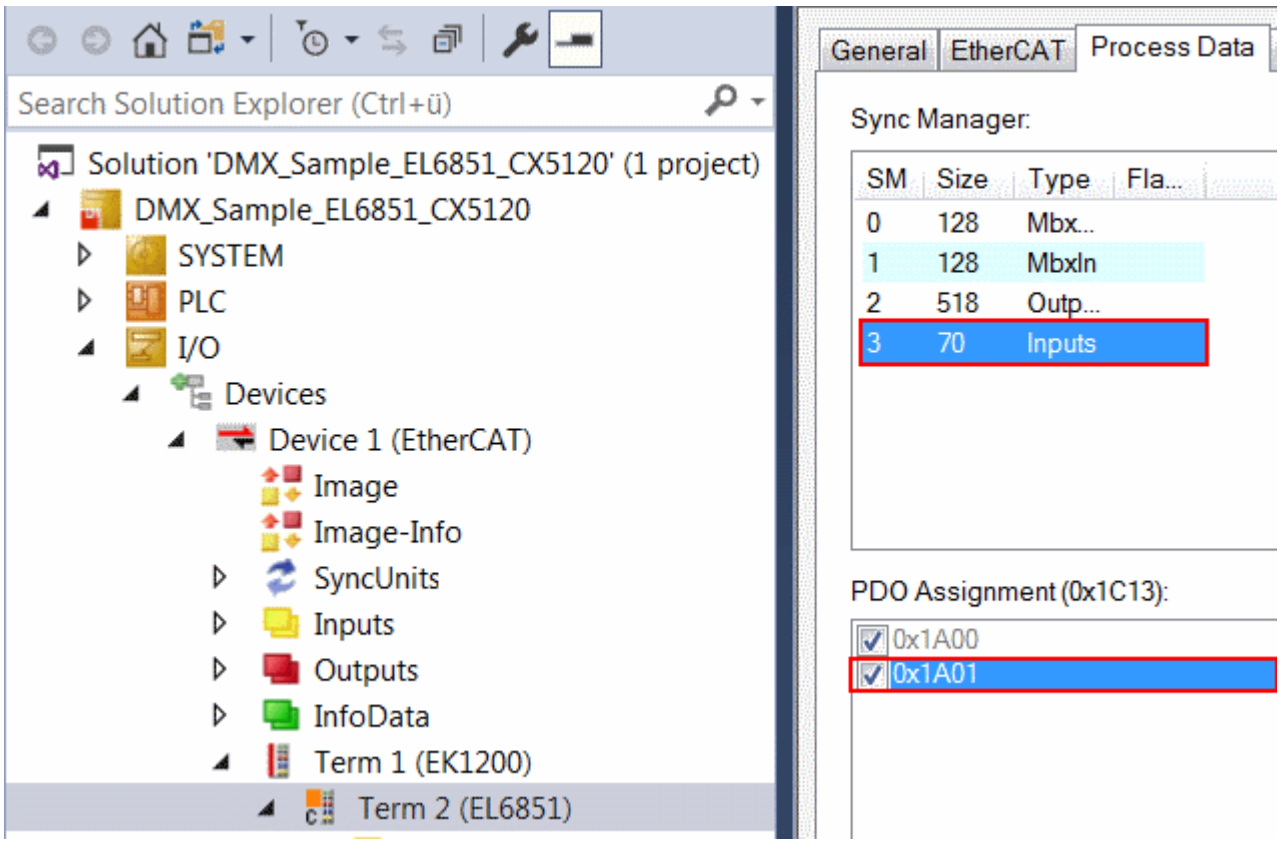
Fügen Sie dieser Task das Programm für die Kommunikation zu. Weitere Information zur Taskkonfiguration finden Sie in der Beschreibung des Bausteins FB_EL6851CommunicationEx [▶ 18].



E/A Konfiguration

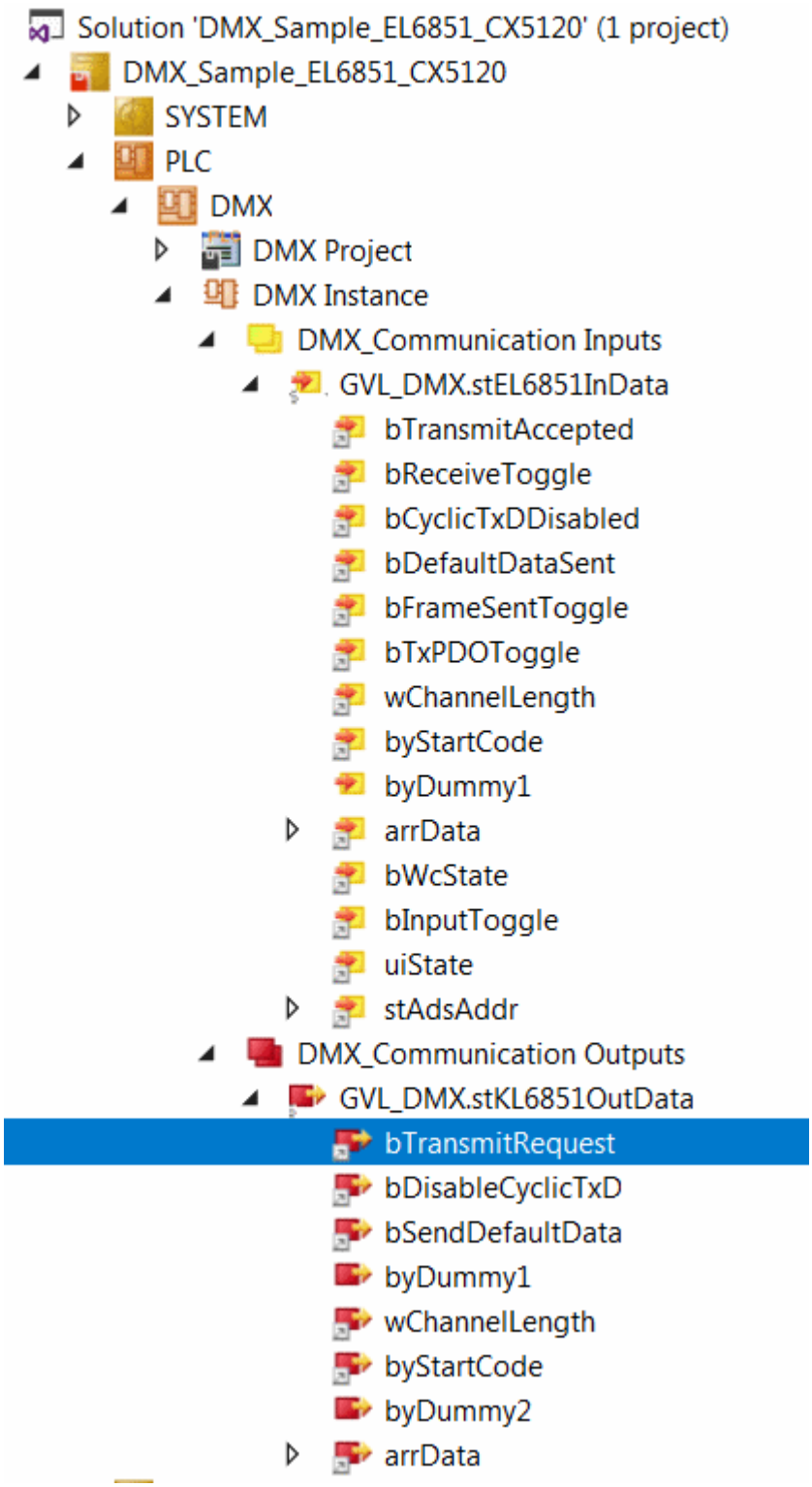
Wählen Sie als Zielsystem den CX und lassen Sie nach dessen Hardware suchen.

Stellen Sie die DMX-Eingänge 1 bis 64 zur Verfügung, indem Sie die Registerkarte *Prozessdaten* der EL6851 öffnen und unter *Inputs* die Option *0x1A01* anwählen.



Im Bereich SPS, in der Instanz des Projektes sehen Sie, dass die Ein- und Ausgangsvariablen der entsprechenden Task zugeordnet sind.

Verknüpfen Sie die globalen Variablen des SPS-Programms mit den Ein- und Ausgängen der EtherCAT-Klemme.



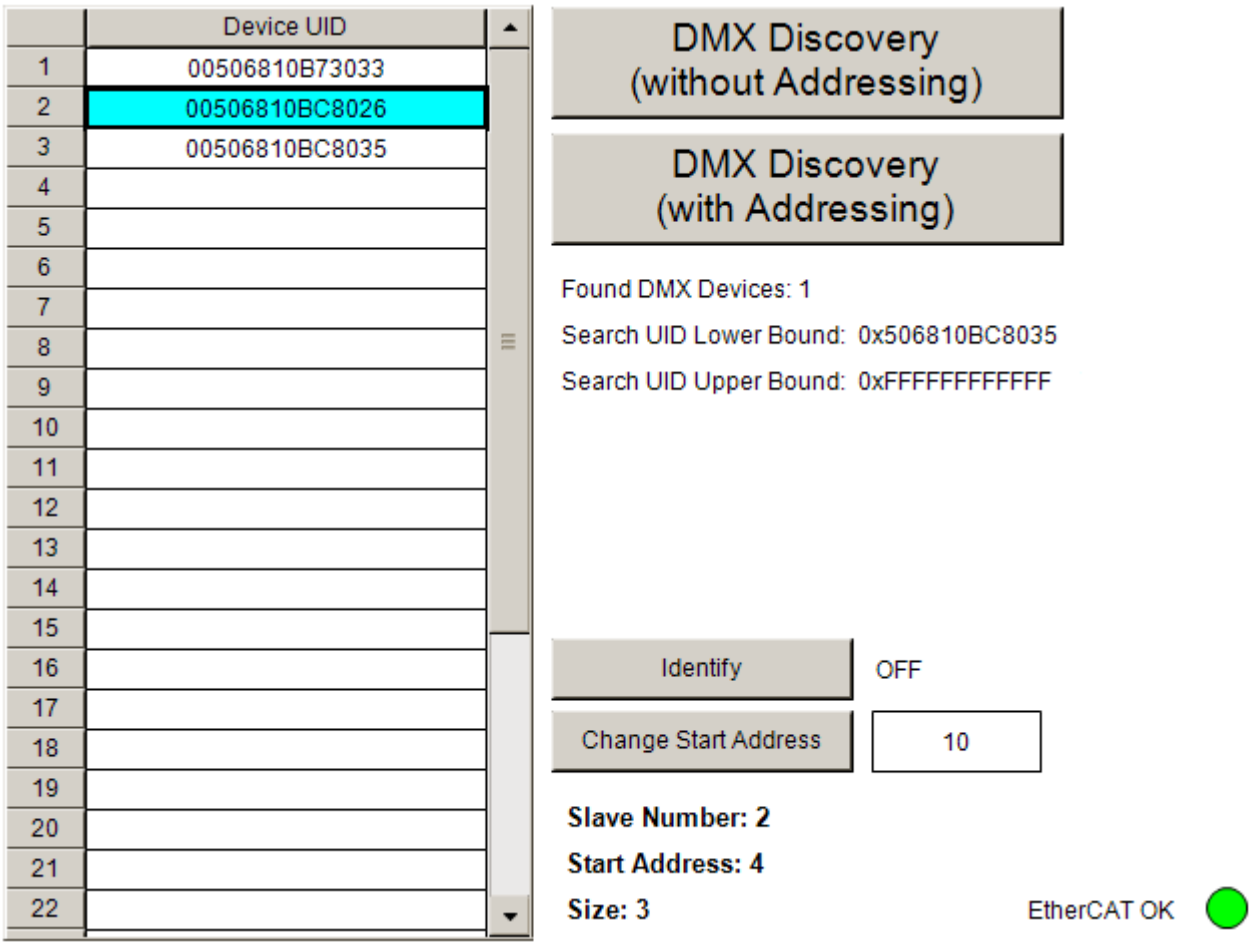
Erstellen Sie die Projektmappe und aktivieren Sie die Konfiguration.

5 Anhang

5.1 Beispiel: Konfigurieren per RDM

Konfigurieren von DMX-Slaves per Remote Device Management (RDM)

https://infosys.beckhoff.com/content/1031/tcplclib_tc2_dmx/Resources/577712139.zip für TwinCAT 3.1.



ErtherCAT-Funktionalität

Der Dialog ist nur dann in Funktion, wenn die "EtherCAT OK"- LED grün leuchtet. Eine rote LED zeigt eine EtherCAT-Kommunikationsstörung an.

Mit der Schaltfläche *'DMX Discovery (without Addressing)*' wird an der DMX-Linie nach DMX-Slaves gesucht. Alle gefundenen DMX-Slaves werden in der linken Liste angezeigt. Durch Anklicken eines Eintrages wird dieser selektiert. Nach Betätigen von '*Identify*' wird der RDM-Befehl zum Identifizieren des jeweiligen DMX-Gerätes versendet. Die Startadresse kann in dem Eingabefeld neben der Schaltfläche *'Change Start Address*' eingegeben werden. Nach Drücken der Schaltfläche wird die neue Startadresse an das selektierte DMX-Gerät gesendet. Im unteren Bereich wird die Nummer des DMX-Gerätes, die Startadresse im DMX512-Frame und die Slotgröße (Anzahl der Bytes im DMX-Frame) für das selektierte Gerät angezeigt.

5.2 Beispiel: DMX-Master

Versenden der zyklischen Prozessdaten als DMX-Master (EL6851)

https://infosys.beckhoff.com/content/1031/tcplclib_tc2_dmx/Resources/530449035.zip für TwinCAT 3.1.

Vorbereitung

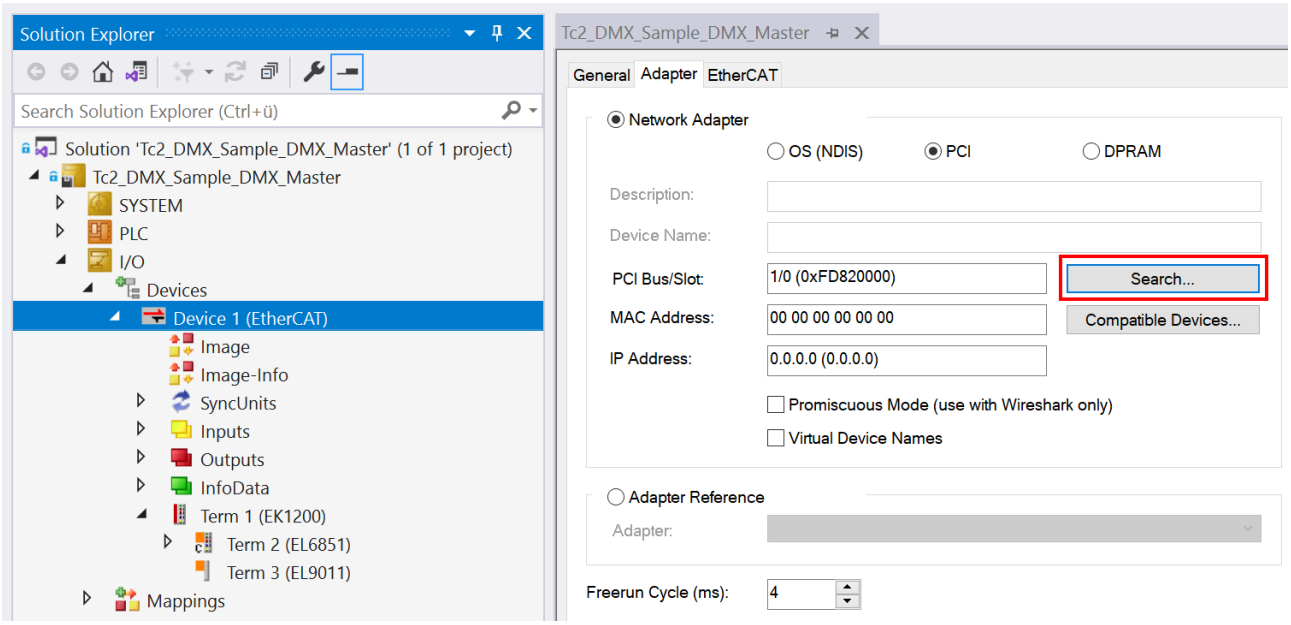
Die Applikationsbeispiele sind mit einem Prüfaufbau getestet und entsprechend beschrieben worden. Etwaige Abweichungen bei der Einrichtung an realen Applikationen sind möglich.

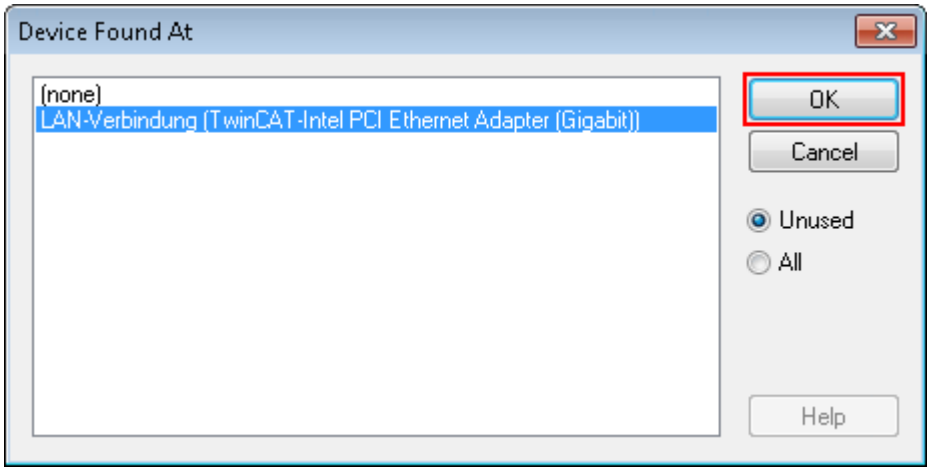
Für den Prüfaufbau wurde folgende Hardware und Software verwendet:

- TwinCAT-PC mit TwinCAT Version 3.1 (Build 4014.2) oder neuer.
- Beckhoff EtherCAT Koppler EK1100, Klemmen EL6851 und EL9011.
- RGB-LED DMX-Slave mit 3 Kanälen (je Farbe einer). Pro Kanal wird ein Slot belegt.

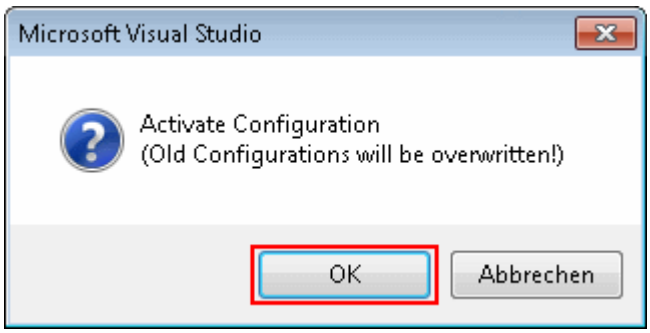
Starten des Beispielprogramms

1. Speichern Sie die https://infosys.beckhoff.com/content/1031/tcplclib_tc2_dmx/Resources/530449035.zip auf Ihrer Festplatte und entpacken Sie diese.
2. Öffnen Sie das Projekt mit TwinCAT XAE.
3. Schließen Sie die Hardware entsprechend an und verbinden Sie den Ethernet-Adapter ihres PCs mit dem EtherCAT-Koppler.
4. Wählen Sie den lokalen Ethernet-Adapter (ggf. mit Echtzeit-Treiber) unter Tc2_DMX_Sample_DMX_Master > I/O > Devices > Device 1 (EtherCAT) aus. Wählen Sie dann im Karteireiter **Adapter** unter **Search...** den entsprechenden Adapter aus und bestätigen Sie diesen.

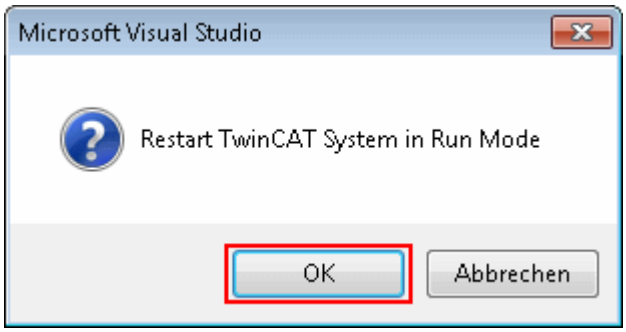




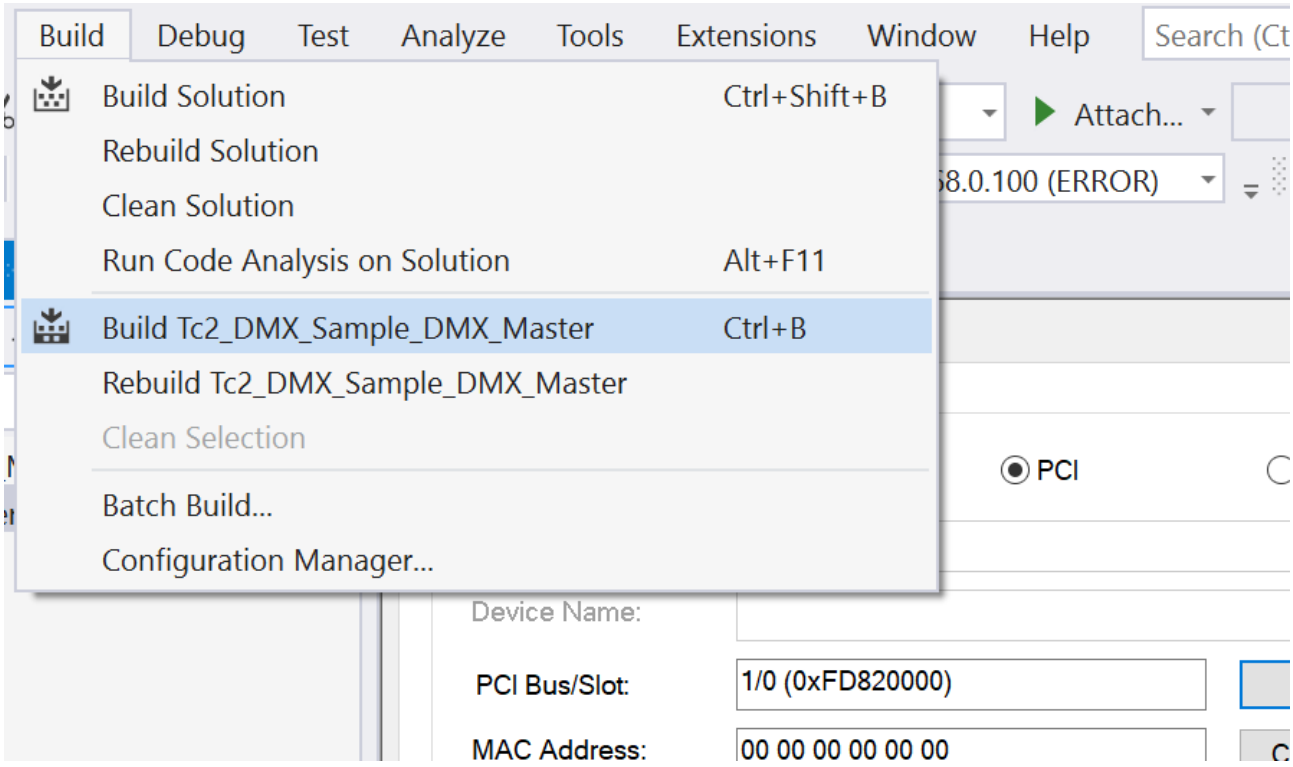
5. Aktivieren Sie die Konfiguration und bestätigen Sie.



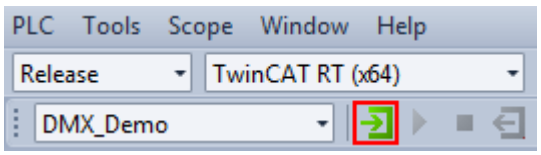
6. Starten Sie TwinCAT im RUN-Modus.

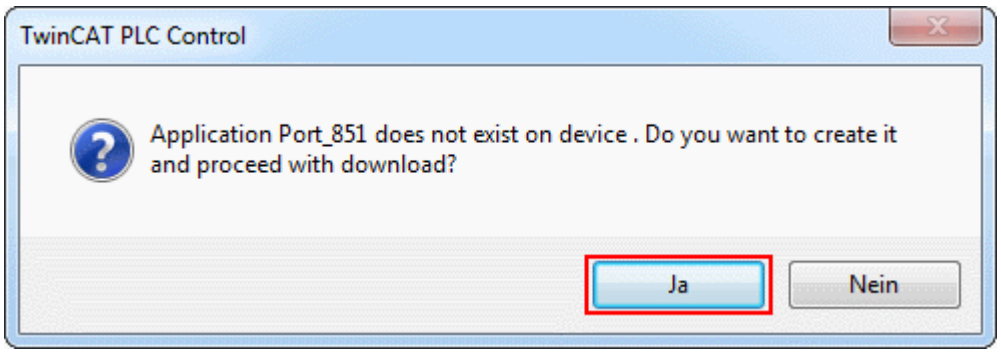


7. Übersetzen Sie das Projekt in TwinCAT XAE, indem Sie im Menü **Build** den Befehl **Build Tc2_DMX_Sample_DMX_Master** auswählen.

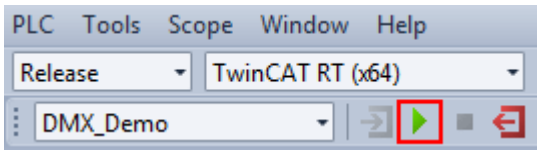


8. Loggen Sie sich ein und bestätigen Sie das Laden des Programms.



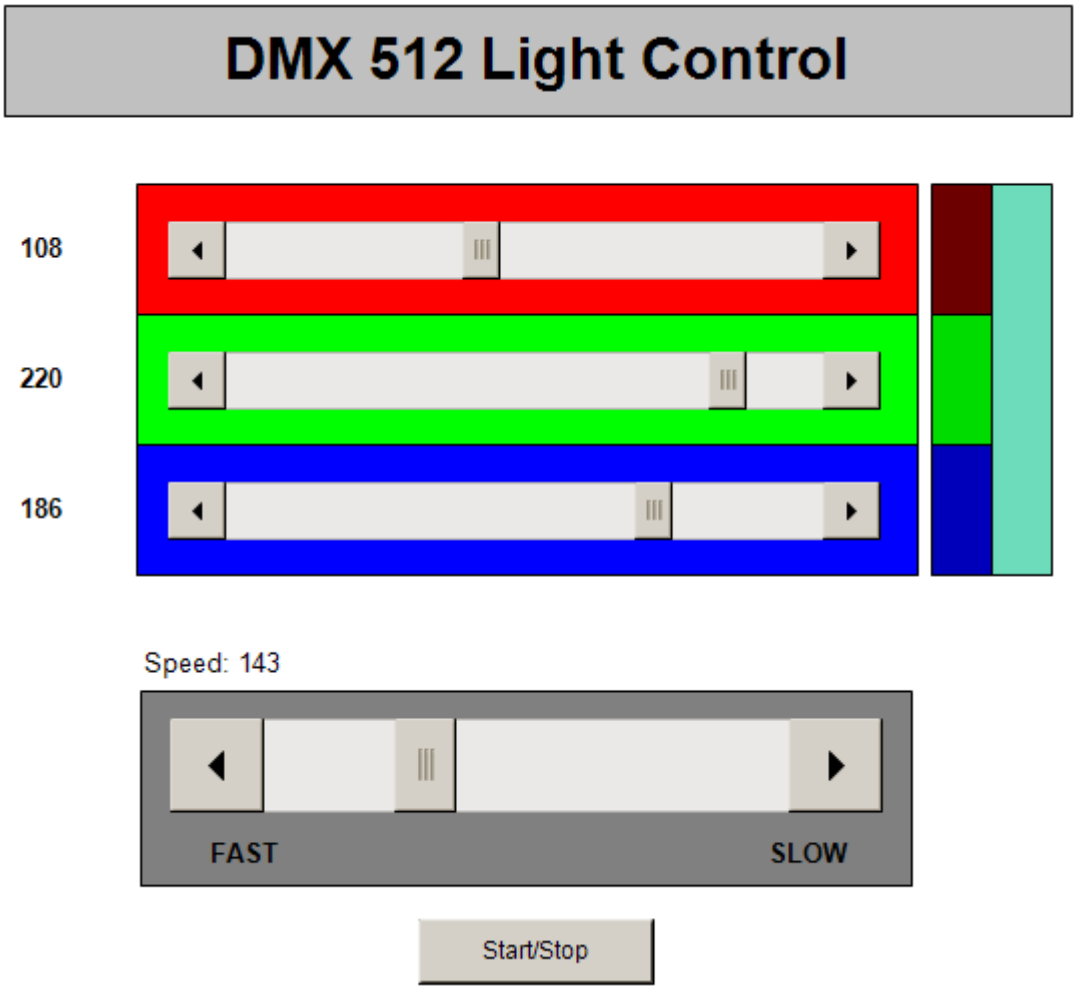


9. Starten Sie das Programm.



Visualisierung

Vorgabe der Stellgrößen für die drei Farben des DMX-Slaves in der TwinCAT XAE:



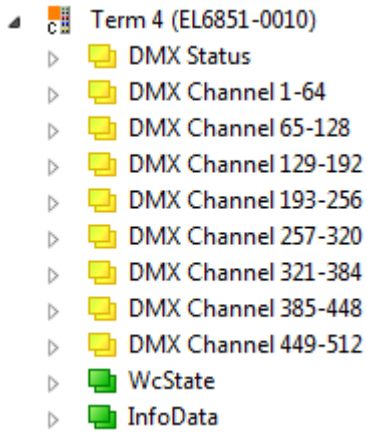
Das Beispiel sendet zyklisch die DMX-Daten zu einem DMX-Slave. Das hier verwendete DMX-Gerät belegt drei Slots (Bytes) im DMX512-Frame. Jeder Slot adressiert eine der drei Farben. Ist die Schaltfläche "Start/Stop" gedrückt, so werden automatisch generierte Daten an das DMX-Gerät gesendet. Die Geschwindigkeit der Änderungen kann durch den waagerechten Schieberegler verändert werden. Ist die Schaltfläche "Start/Stop" nicht gedrückt, so können Sie durch die drei waagerechten Schieberegler die Werte manuell verändern.

5.3 Beispiel: DMX-Slave

Empfangen von jeweils 64 Byte Daten an zwei DMX-Slaves (EL6851-0010)

https://infosys.beckhoff.com/content/1031/tcplclib_tc2_dmx/Resources/578408715.zip für TwinCAT 3.1.

Arrays zu je 64 Byte im Vollausbau (alle PDO angewählt):



Eine EL6851-0010 kann max. 512 Byte lesen (jeweils 64 Byte in acht Arrays, siehe Abb. 1). Die Arrays können in der TwinCAT XAE (Reiter *Prozessdaten*) über das PDO 0x1C13 zugeordnet werden.

Beispiel:

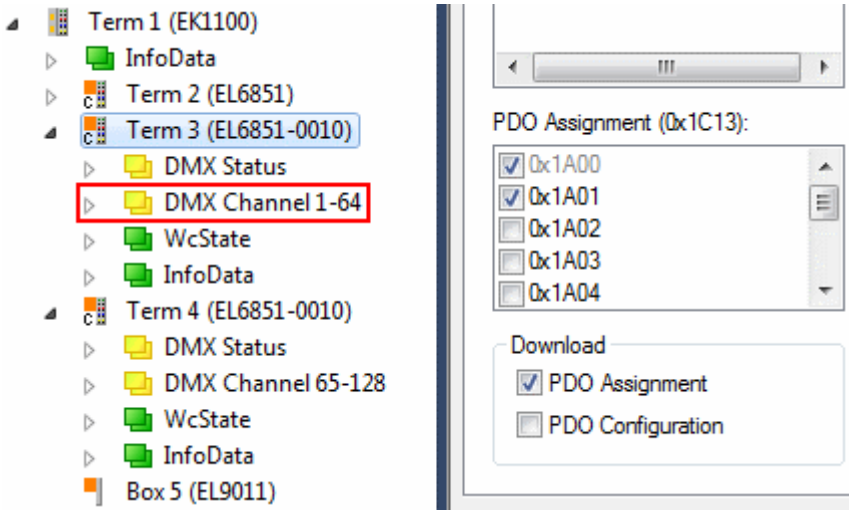
DMX Channel 1 - 64 -- > Index 0x1A01

DMX Channel 65 - 128 --> Index 0x1A02

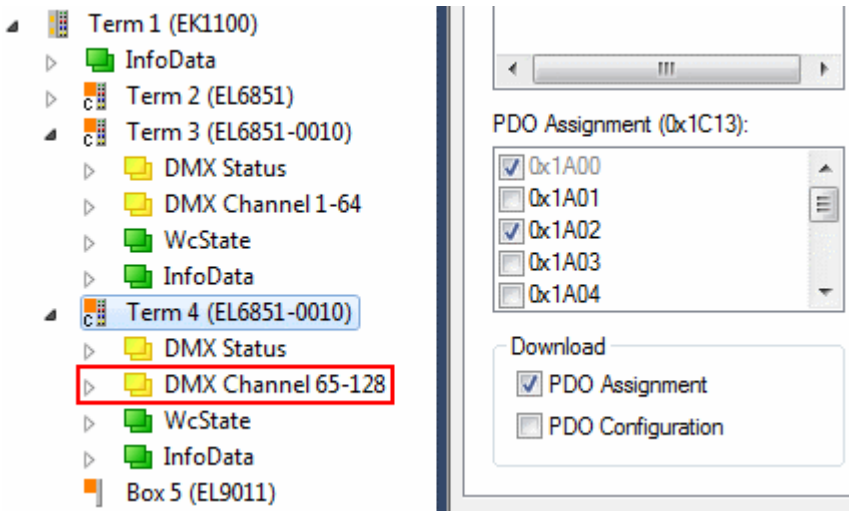
....

DMX Channel 449 - 512 --> Index 0x1A08

- DMX Channel 1 - 64 (default) durch Anwahl PDO 0x1A01:

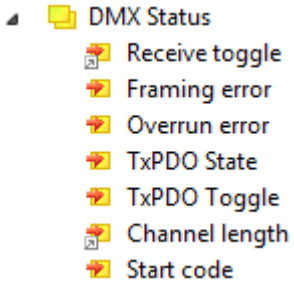


- DMX Channel 65 - 128 durch Anwahl PDO 0x1A02:



Im Beispiel-Programm empfängt der erste DMX-Slave die ersten gesendeten 64 Byte und der zweite die nächsten 64 Byte (Abb. 2 + 3; der Empfang von den gesamten 128 Byte mit einer EL6851-0010 ist auch möglich, im Beispiel ist die Aufteilung bewusst gewählt).

DMX Status Objekt:



Im DMX Status Objekt (Index 0x6000, *DMX-Status*, Abb. 4) ist mit Index 0x6000:11 (*Channel length*) ein Copy Counter angelegt.

Beispiel:

Bei aktiviertem PDO 0x1A01 beträgt der Wert von *Channel length* 64_{dez}. Bei aktiviertem PDO 0x1A02 ist der Wert 128_{dez}. Sind beide PDO aktiviert (0x1A01 und 0x1A02) beträgt der Wert ebenfalls 128_{dez}.

Watchdog DMX Slave 1

Watchdog DMX Slave 2

	P_DMX_Slave_1.DMX_DATA[INDEX]
1	16#71
2	16#00
3	16#00
4	16#00
5	16#00
6	16#00
7	16#00
8	16#00
9	16#00
10	16#00
11	16#00
12	16#00
13	16#00
14	16#00
15	16#00
16	16#00
17	16#00
18	16#00
19	16#00
20	16#00

	P_DMX_Slave_2.DMX_DATA[INDEX]
1	16#71
2	16#00
3	16#00
4	16#00
5	16#00
6	16#00
7	16#00
8	16#00
9	16#00
10	16#00
11	16#00
12	16#00
13	16#00
14	16#00
15	16#00
16	16#00
17	16#00
18	16#00
19	16#00
20	16#00

DMX Slave 1 empfängt 64 Byte Daten auf Kanal 1 des ersten Arrays (*DMX Channel 1 - 64*)

DMX Slave 2 empfängt 64 Byte Daten auf Kanal 1 des zweiten Arrays (*DMX Channel 65 - 128*)

Das *Receive toggle* Bit (Index 0x6000:02) wird jeweils über den FB *fbMonitorToggleBit* ausgewertet und angezeigt (Watchdog DMX Slave).

5.4 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

Downloadfinder

Unser Downloadfinder beinhaltet alle Dateien, die wir Ihnen zum Herunterladen anbieten. Sie finden dort Applikationsberichte, technische Dokumentationen, technische Zeichnungen, Konfigurationsdateien und vieles mehr.

Die Downloads sind in verschiedenen Formaten erhältlich.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unserer Internetseite: www.beckhoff.com

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline:	+49 5246 963-157
E-Mail:	support@beckhoff.com

Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline:	+49 5246 963-460
E-Mail:	service@beckhoff.com

Beckhoff Unternehmenszentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon:	+49 5246 963-0
E-Mail:	info@beckhoff.com
Internet:	www.beckhoff.com

Mehr Informationen:
www.beckhoff.com/te1000

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.com
www.beckhoff.com