Die Bibliothek SysLibPLCConfig.lib

Diese Bibliothek unterstützt das Auslesen der Konfigurationsdaten der Steuerungskonfiguration. Diese werden beim Download der Applikation ebenfalls zur Steuerung übertragen und vom Laufzeitsystem in Strukturen geschrieben. Die Bibliothek bietet Funktionen, um Zeiger auf diese Strukturen zu erhalten. Die Abarbeitung erfolgt synchron.

- Da man Zeiger auf die Originalstrukturen des Laufzeitsystems erhält, gilt folgendes:
- Die Struktur (Zeiger auf Unterelemente) darf nicht verändert werden.
- Werden die Default-Werte von Parametern in der Struktur verändert, bleibt dies ohne Einfluss.

Wenn das Zielsystem die Funktionalität beinhaltet, können folgende Bibliotheksfunktionen verwendet werden:

- CfgCCGetError
- CfgCCGetHeader
- CfgCCGetRootModule
- CfgCCGetRootModuleByModuleId
- CfgCCGetRootModuleByNodeId

CfgCCGetError

Hinweis: Momentan im Laufzeitsystem noch nicht implementiert. Fehlercode immer 0.

Diese Funktion liefert Informationen zum beim Laden der Konfiguration aufgetretenen Fehler.

Als Rückgabewert erhält man dazu einen Zeiger auf die Struktur CCLoadError.

Struktur CCLoadError

```
TYPE CCLoadError :
STRUCT
    ulLastError: UDINT;          (* Fehlercode des letzten Fehlers. *)
    ulAddInfo1: UDINT;           (* Entsprechend ulLastError, Bedeutung ändert sich. *)
    ulAddInfo2: UDINT;           (* Entsprechend ulLastError, Bedeutung ändert sich. *)
    szLastError: STRING(32);     (* Letzte Fehlermeldung, erleichtert ev. Debugging. *)
END_STRUCT
END_TYPE
```

CfgCCGetHeader

Diese Funktion liefert einen Zeiger auf die Header-Struktur der Steuerungskonfiguration CCHeader.

Struktur CCHeader:

```
TYPE CCHeader :
STRUCT

   szTag: STRING(10); (* Enthält nullterminierte Zeichenfolge STRING "CommConf" *)

   cByteOrder: BYTE; (* Die Dateidaten liegen im Intel ('I') oder Motorola Format ('M') vor *)

   ulSize: UDINT; (* Größe der folgenden Daten *)

   lVersion: UDINT; (* Versionsnummer der Datei *)

END_STRUCT
END_TYPE
```

CfgCCGetRootModule

Diese Funktion liefert Information zum Root-Modul der Steuerungskonfiguration. Als Rückgabewert erhält man dazu einen Zeiger auf die Struktur CCModule.

Struktur CCModule

```
TYPE CCModule :

STRUCT

    ucEntryTag: BYTE;                (* 'M' = Modul*)
    ucDummy1: BYTE;
    ucDummy2: BYTE;
    ucDummy3: BYTE;
    ulModuleId: UDINT;               (* Id des Moduls, die in der *cfg-Datei definiert ist. *)
    sModuleNumber: UINT;             (* Nummer des Moduls innerhalb des Eltern-Moduls (-1 wenn es sich um das Root-Modul handelt) *)
    usModuleTag: UINT;               (* Modultyp: 0=3S-Module, 1=DP-Master, 2=DP-Slave, 3=CAN-Master, 4=CAN-Slave, 5=DP-SingleSlave *)
    byDeviceDriver: BYTE;            (* Angabe, ob das Modul einen Gerätetreiber benötigt: 0=FALSE, 1=TRUE*)
    ucDummy4: BYTE;
    ucDummy5: BYTE;
    ucDummy6: BYTE;
    ulNodeId: UDINT;                 (* Knotennummer des Moduls*)
    byDefinedWithStruct: BYTE;       (* Das Modul ist über eine Struktur definiert: 0=FALSE, 1=TRUE *)
    ucDummy7: BYTE;
    ucDummy8: BYTE;
```

```
    ucDummy9: BYTE;
    ulBitOffsetInput: UDINT;                (* Offset des Modul-Eingangsbereichs *)
    ulBitSizeInput: UDINT;                  (* Größe des Modul-Eingangsbereichs in Bit *)
    ulBitOffsetOutput: UDINT;               (* Offset des Modul-Ausgangsbereichs *)
    ulBitSizeOutput: UDINT;                 (* Größe des Modul-Ausgangsbereichs in Bit *)
    ulRefIdCommonDiag: UDINT;               (* RefId des gemeinsamen Diagnosebereichs der
                                               Module *)
    ulBitOffsetCommonDiag: UDINT;           (* Offset des gemeinsamen Diagnosebereichs der
                                               Module *)
    ulBitSizeDiag: UDINT;                   (* Größe des gemeinsamen Diagnosebereichs der
                                               Module in Bit*)
    usParameterCount: UINT;                 (* Anzahl der Parameter *)
    usDummy: UINT;
    ppccpModuleParams: POINTER
    TO POINTER TO ccParam;                  (* <ccParam [0..usParameterCount]> Zeiger
                                               auf ein Array von Zeigern auf
                                               CCModuleParam-Strukturen. (Definition der
                                               Struktur CCParam siehe unten).
                                               Dereferenzierung des Zeigers mit
                                               ppccpModuleParams^ liefert den Zeiger auf
                                               die erste Parameter-Struktur.
                                               (ppccpModuleParams+4)^ liefert den
                                               Zeiger auf die nächste Parameter-Struktur.
                                               Siehe auch Beispielprojekt, Kommentar (*
                                               Read pointer to parameters *).
    ulSizeOfSpecificData: UDINT;            (* Größe der modulspezifischen Daten in Bytes *)
    pModuleData: POINTER TO
    BYTE;                                   (*<MODULE_SPECIFIC_DATA> Hier liegen die
                                               Daten, entsprechend usModuleTag: pModuleData
                                               kann ein Zeiger auf PBSlave, CANSlave,
                                               PBMaster, PBSlave, PBSingleSlave, sein; siehe
                                               untenstehende Definitionen.*)
    usChannelCount: UINT;                   (* Anzahl der konfigurierten Kanäle *)
    usModuleCount: UINT;                    (* Anzahl der konfigurierten Module *)
(*  Im Folgenden die Angaben der Kanäle und Sub-Module des Moduls in der konfigurierten
Reihenfolge! (DP-Slaves sind nach Stationsnummer angeordnet!) Somit kann hier auch eine
weitere CCModule-Struktur enthalten sein. *)
    ppcccChannels: POINTER TO
    POINTER TO ccChannel;                   (* <ccChannel [0..usChannelCount]> Definition der
                                               Struktur CCChannel siehe unten. Dereferenzierung
                                               des Zeigers mit ppccpChannels^ liefert den Zeiger
                                               auf die erste Kanal-Struktur. (ppccpChannels+4)^
                                               liefert den Zeiger auf die nächste Kanal-Struktur.
                                               Siehe auch Beispielprojekt, Kommentar "(*Read
                                               pointer to parameters *)"
    ppccmSubModules: POINTER
    TO POINTER TO BYTE;                     (*<ccModule [0..usModuleCount]> Zeigt auf ein
                                               Array von Variablen vom Typ POINTER TO
                                               ccModule. Um den Inhalt anzuzeigen, muss der
                                               Wert einer Variable vom Typ "POINTER TO
                                               CCModule" zugewiesen werden. Definition der
                                               Struktur CCModule siehe unten. Dereferenzierung
                                               des Zeigers mit ppccpSubModules^ liefert den
                                               Zeiger auf die erste Submodul-Struktur.
                                               (ppccpSubModules+4)^ liefert den Zeiger auf die
                                               nächste Submodul-Struktur. Siehe auch
                                               Beispielprojekt, Kommentar "(*Read pointer to
                                               parameters *)" *)
END_STRUCT
```

```
END_TYPE
```

Struktur CCChannel:

```
TYPE CCChannel :
STRUCT
    ucEntryTag: BYTE;                       (* 'C' = Channel *)
    ulChannelId: UDINT;                     (* Id des Kanals, wie in der Konfigurationsdatei
                                               angegeben *)
    usChannelNumber: UINT;                  (* Nummer des Kanals im Elternmodul *)
    ulRefId: UDINT;                         (* Richtung des Kanals: 1=input, 2=output, 3=input AND
                                               output *)
    usChannelType: UINT;                    (* Kanaltyp (angegeben als CoDeSys "TypeClass") *)
    ulBitOffset: UDINT;                     (* Offset des Kanals im in-/output-Bereich *)
    usParameterCount: UINT;                 (* Anzahl der Parameter *)
    ppccpParams: POINTER TO                 (* PARAMETER[1..usParameterCount]> Zeiger auf ein
    POINTER TO CCParam;                        Array von Zeigern auf CCParam-Strukturen. (Definition
                                               der Struktur CCParam siehe unten) *)
END_STRUCT
END_TYPE
```

Struktur CCParam:

```
TYPE CCParam :
STRUCT
    ulParameterId: UDINT;                   (* Id des Parameters wie in der Konfigurationsdatei
                                               definiert *)
    usParameterNumber: UINT;                (* Anzahl der Parameter des Moduls *)
    byReadOnly: BYTE;                       (* 1=TRUE, 0=FALSE *)
    byDummy: BYTE;
    usParameterType: UINT;                  (* Typ des Parameters (als CoDeSys "TypeClass"
                                               definiert *)
    usDummy: UINT;
    ulSize: UDINT;                          (* Größe des Parameters in Bytes *)
    byValue: BYTE;                          (* Die Speicherbelegung des Parameterwerts startet mit
                                               diesem Byte. Die anderen Bytes folgen unmittelbar
                                               danach, wenn die Größe des Parameterwerts größer
                                               als 1 ist. *)
END_STRUCT
END_TYPE
```

CfgCCGetRootModuleByModuleId

Diese Funktion liefert Informationen zum Root-Modul der aktuellen Steuerungskonfiguration, das über die Modul Id angegeben wird. Die Modul-Id ist in der Konfigurationsdatei mit dem Eintrag "Id" definiert, siehe Dokument Steuerungskonfiguration_D.pdf.

Als Rückgabewert erhält man einen Zeiger auf die Struktur CCModule (siehe oben, Funktion CfgCCGetRootModule)

Input-Variable	**Datentyp**	**Beschreibung**
ulModuleId	UDINT	Modul-Id des Root-Moduls

CfgCCGetRootModuleByNodeId

Diese Funktion liefert Informationen zum Root-Modul der aktuellen Steuerungskonfiguration, das über die Node-Id angegeben wird.

Die Knotennummer des Moduls ergibt sich normalerweise automatisch aus der Position des Moduls in der Konfiguration. Näheres siehe Dokument Steuerungskonfiguration_D.pdf.

Als Rückgabewert erhält man einen Zeiger auf die Struktur CCModule (siehe oben, Funktion CfgCCGetRootModule)

Input-Variable	**Datentyp**	**Beschreibung**
ulNodeId	UDINT	Node-Id des Root-Moduls