FAQ COMBIVIS studio 6

Device OD Erweiterung mit benutzerdefinierten Unterobjekten und einem Callback Verfahren

FAQ No.0017

Part	Version	Revision	Date	Status
de	6.4.0.0	002	2020-03-13	Released

Inhalt

KEB Automation KG
Südstraße 38
32683 Barntrup, Germany
E-mail: info@keb.de
Tel. +49 5263 401-0
Fax +49 5263 401-116
www.keb.de

District Court Lemgo HRA 5649
DUNS-No. 314108728
VAT-No. DE309087075
Bank Details: Sparkasse Paderborn-Detmold
IBAN DE 19 4765 0130 0000 0060 07
BIC WELADE3L

General Partner:
Vittorio Tavella
KEB Verwaltungs-GmbH, Barntrup
District Court: Lemgo HRB 8965
Directors: Curt Bauer CMO,
Ralf Lutter COO, Vittorio Tavella CFO,
Wolfgang Wiele CTO

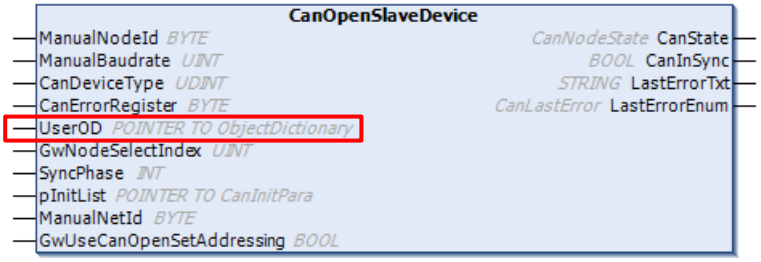
Einführung

Dieses Dokument gibt einen allgemeinen Überblick darüber, wie ein bereits vorhandenes Objekt in einen internen CANopen Device OD (Object Dictionary) um ein benutzerdefiniertes Unterobjekt erweitert werden kann. Dazu muss in dem COMBIVIS studio 6 Projekt eine OD Callback Methode eingeführt werden. Die ist nützlich für die Arbeit mit beliebigen Teilobjekten, die nur in ausgewählten KEB Geräten enthalten sind (z.B. Unterobjekt 1800:3 inhibit time)

Die weiterführende illustrierte Lösung gibt Informationen zu einigen Hintergrundteilen und eine exemplarische Implementierung. Es wurde auf der Basis eines CANopen KEB_CanSlave 3.5.9.72 eingeführt.

OD Zugriff auf Objekte und Unterobjekte

Jeder CANopen Slave Gerät hat ein Geräte-OD, der alle benötigten gerätespezifischen Objekte enthält. Es kann um einen benutzerdefinierten OD erweitert werden, der den Anforderungen des Benutzers an zusätzliche Parameter und Werte entspricht. Dieses Benutzer-OD muss direkt in dem CvS6 Projekt angelegt werden und am UserOD VAR_INPUT des CANopen Slave Geräts registriert werden.



KEB_CanSlave CanOpenSlaveDevice: VAR_INPUT + VAR_OUTPUT

Die Benutzer OD Instanz bietet einen einfachen und flexiblen Zugriff auf ihren Inhalt durch die Applikationsvariablen. Das Geräte OD ist im Vergleich zu einem Benutzer OD fest und begrenzt, da er nur von dem Gerät selbst direkt zugänglich ist. Der Zugriff vom CAN-Bus, zum Abrufen oder Setzen von CANopen-Werten eines beliebigen OD, erfolgt über die Bibliothek KEB_CanSlave und wird automatisch über den KEB Channelhandler abgewickelt.

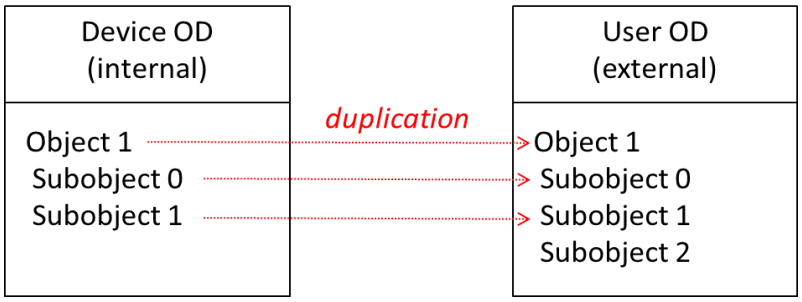
Im Detail werden die CanOpenSlaveDevice Methoden GetCanValue() und SetCanValue() vom CAN-Bus aufgerufen und versuchen zuerst auf das angeforderte Objekt im Benutzer OD zuzugreifen, sofern dieses registriert ist.

Auf den Geräte OD wird nur zugegriffen, wenn das Benutzer OD nicht registriert wurde oder wenn das angeforderte Objekt nicht enthalten / nicht zugänglich ist. Diese Reihenfolge hat den Grund dass ein Objekt, das in beiden ODs enthalten ist, geändert werden könnte, um den Zugriff im Benutzer OD abzulehnen und bei Bedarf auf den Geräte OD umzuleiten. In diesem Fall gibt es auch keine andere Möglichkeit, direkt mit dem internen Objekten zu arbeiten.

Dieser gemischte Zugriff auf ein Objekt ist für das Hauptthema dieses Dokuments nützlich und wird in den nächsten Abschnitten erläutert. Die Art und Weise, wie man eine benutzerdefinierte OD erstellt, ist nicht Inhalt dieses Dokuments.
Für weitere Informationen zur KEB_ObjektDictionary Bibliothek beachten Sie bitte den Abschnitt Implementierung der OD Callback Methode.

Geräte OD Erweiterung mit einem benutzerdefinierten Unterobjekt

Um ein bestehendes Objekt im Geräte OD um ein zusätzliches Teilobjekt zu erweitern, muss das gesamte Objekt im Benutzer OD dupliziert werden. Dies führt zu einem Zugriffsproblem, das auch vom Benutzer gelöst werden muss.



Objektverzeichnis: Vervielfältigung von Objekt 1, Erweiterung mit Unterobjekt 2

Das Zugriffsproblem verursacht zwei Fragen:

1. Welches Unterobjekt ist in welchem OD gültig?
2. Wie kann der Zugriff auf das betroffene OD für dieses Unterobjekt gewährt werden?

OD Objektwerte: Gültigkeit und Zugriff

Für jedes duplizierte oder hinzugefügte Unterobjekt im Benutzer OD zeigt die nächste Liste die Gültigkeit und die Zugriffseigenschaften der Werte. Das Benutzer OD hat eine höhere Priorität und ist ohne weitere Anpassung zugänglich.

1. Teilobjekt 0: Gültig im Benutzer OD (Anzahl der weiteren Teilobjekte).

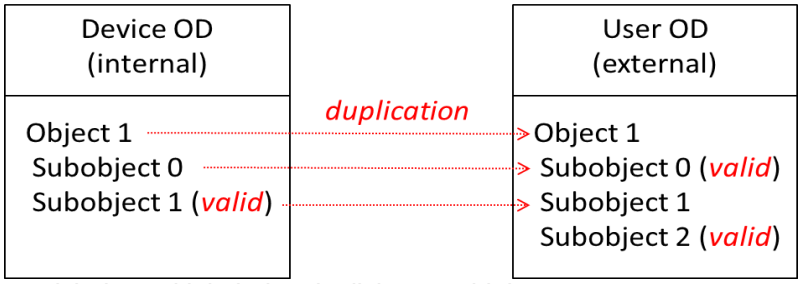
Das Teilobjekt 0 enthält die Anzahl der weiteren Unterobjekte (Array). Da zuerst im Benutzer OD auf alle Objekte zugegriffen wird und die Gesamtzahl der Unterobjekte erhöht wurde, ist es notwendig, den aktualisierten Wert im Benutzer OD zu verwenden. Das OD Subobjekt 0 des Gerätes bleibt auf dem vorherigen Wert für den internen Zugriff des Geräts.

2. Unterobjekt 2: Gültig im Benutzer OD (weitere Unterobjekte).

Weitere Unterobjekte sind nur im Benutzer OD vorhanden und uneingeschränkt zugänglich.

3. Unterobjekt 1: Gültig im Geräte OD (duplizierte Unterobjekte).

Weitere duplizierte Unterobjekte, die in beiden ODs enthalten sind, müssen auf den Geräte OD umgeleitet werden, da das Gerät nur mit seinen internen Objekten arbeitet.



Objektverzeichnis: Gültigkeit des duplizierten Objekts

Für die dritte Option (duplizierte Unterobjekte) ist es notwendig, den gemeinsamen Zugriff auf die Geräte und Benutzerteile des gesamten Objekts zu organisieren.

Beispiel: Benutzer OD Unterobjekt Umleitung zum Geräte OD

Im nächsten CvS6 Applikationsbeispiel wird davon ausgegangen, dass das Geräteobjekt 1800 mit zwei Unterobjekten im Benutzer OD eines CANopen Gerätes dupliziert wurde. Im Unterobjekt 3 wurde der Eintrag „inhibit time“ (nur bei ausgewählten KEB Geräten enthalten) hinzugefügt.

```
//Pdo1 Tx Communication, subobject 0
(
  Index             :=16#1800,                    //Index
  SubIdxMod         :=16#0000,                    //SubIdxMod
  Itype             :=OD_TBL_ITEM_ENTRIES_3 OR
  //further properties, collapsed [10 lines]
  ReadFuncIdx       :=0,                          //ReadFuncIdx
  WriteFuncIdx      :=0,                          //WriteFuncIdx
  LowerLim          :=0,                          //LowerLim
  UpperLim          :=0,                          //UpperLim
  DefValue          :=3,                          //DefValue
  pData             :=ADR(Para1800_00)            //pData
),
```

Beispiel: Objekt 1800, Unterobjekt 0, Adressierung von 3 weiteren Unterobjekten, gültig im Benutzer OD

Um die beiden Geräteunterobjekte wieder auf das Geräte OD umzuleiten, werden die Objektfunktionen Indexwerte ungleich Null verwendet. Das Unterobjekt 3 benötigt keine Änderung.

Benutzer OD Objekt Funktion Indexmodifikation

Der Indexwert der Lese- und Schreibfunktion ermöglicht eine OD Callback Methode separat für einen Lese- oder Schreibzugriff. Der benutzerdefinierte Wert kann in der Methode interpretiert werden, um zwischen einzelnen Bedeutungen für eine beliebige Zahl zu unterscheiden. Der Standardwert „0“ des Funktionsindex deaktiviert die Callback Methode.

In diesem Beispiel wird davon ausgegangen, dass der Funktionsindexwert des Unterobjekts 1+2 für beide Zugriffsrichtungen auf “1” gesetzt wurde. Die Variablen Para1800_dummy werden nun aus Platzhaltergründen benötigt.

```
//subobject 1
(
  Index             :=16#1800,                    //Index
  SubIdxMod         :=16#0100,                    //SubIdxMod
  Itype             :=OD_TBL_ITEM_ENTRIES_0 OR
  //further properties, collapsed [9 lines]
  ReadFuncIdx       :=1,                          //ReadFuncIdx
  WriteFuncIdx      :=1,                          //WriteFuncIdx
  LowerLim          :=0,                          //LowerLim
  UpperLim          :=65535,                      //UpperLim
  DefValue          :=0,                          //DefValue
  pData             :=ADR(Para1800_dummy)         //pData
),
//subobject 2
(
  Index             :=16#1800,                    //Index
  SubIdxMod         :=16#0200,                    //SubIdxMod
  Itype             :=OD_TBL_ITEM_ENTRIES_0 OR
  //further properties, collapsed [9 lines]
  ReadFuncIdx       :=1,                          //ReadFuncIdx
  WriteFuncIdx      :=1,                          //WriteFuncIdx
  LowerLim          :=0,                          //LowerLim
  UpperLim          :=65535,                      //UpperLim
  DefValue          :=0,                          //DefValue
  pData             :=ADR(Para1800_dummy)         //pData
),
```

Beispiel: Objekt 1800, Unterobjekt 1+2 mit Funktionsindex 1, gültig nur in Geräte OD

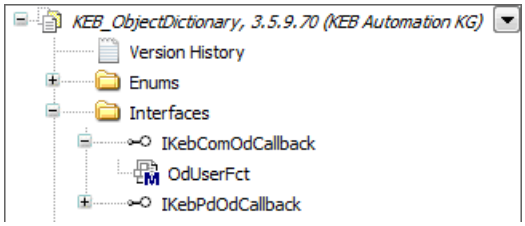
OD Callback Methode Umgebung

Die folgende Tabelle zeigt die Beziehung zwischen den verschiedenen POUs (Program Organisation Unit), die für eine OD Callback Implementierung in CvS6 benötigt werden:

Library / POU	POU used	POU instances
KEB_CanSlave	CanOpenSlaveDevice	KEB_CanOpenSlave
KEB_ObjectDictionary	ObjectDictionary	UserOdCallback
KEB_ObjectDictionary	IKebComOdCallback (interface) with IKebComOdCallback.OdUserFct() (interface method)	
new Function Block	UserOdCallback (FB) impl. IKebComOdCallback with UserOdCallback.OdUserFct()	MyCallback MyCallback. OdUserFct()

Das CANopen Slave Gerät ist hier für den Anschluss eines Benutzer OD in den Initialisierungsschritt der CvS6 Applikation aufgeführt. Eine Übersicht über den Baustein CanOpenSlaveDevice finden Sie im Abschnitt "OD Zugriff auf Objekte und Unterobjekte".

Die Callback Methode OdUserFct() ist in der Schnittstelle IKebComOdCallback der Bibliothek KEB_ObjectDictionary enthalten. Diese Schnittstelle muss in einem neuen Funktionsblock implementiert werden. So kann die Interface Methode sofort an die Bedürfnisse des Benutzers angepasst werden.



KEB_ObjectDictionary mit Schnittstelle IKebComOdCallback und Methode OdUSerFct()

Zuerst muss die Instanz der benutzer OD im CANopen Slave Gerät und due Instanz des neuen Funktionsblocks im Benutzer OD´s.callbackInst Var_INPU registriert werden.

```
//created and initialized a user OD before
KEB_CanOpenSlave.UserOD := ADR(UserOD);
UserOd.callbackInst := MyCallBack;
```

Registrierung des Benutzer OD und der Benutzer Callback Instanzmethode

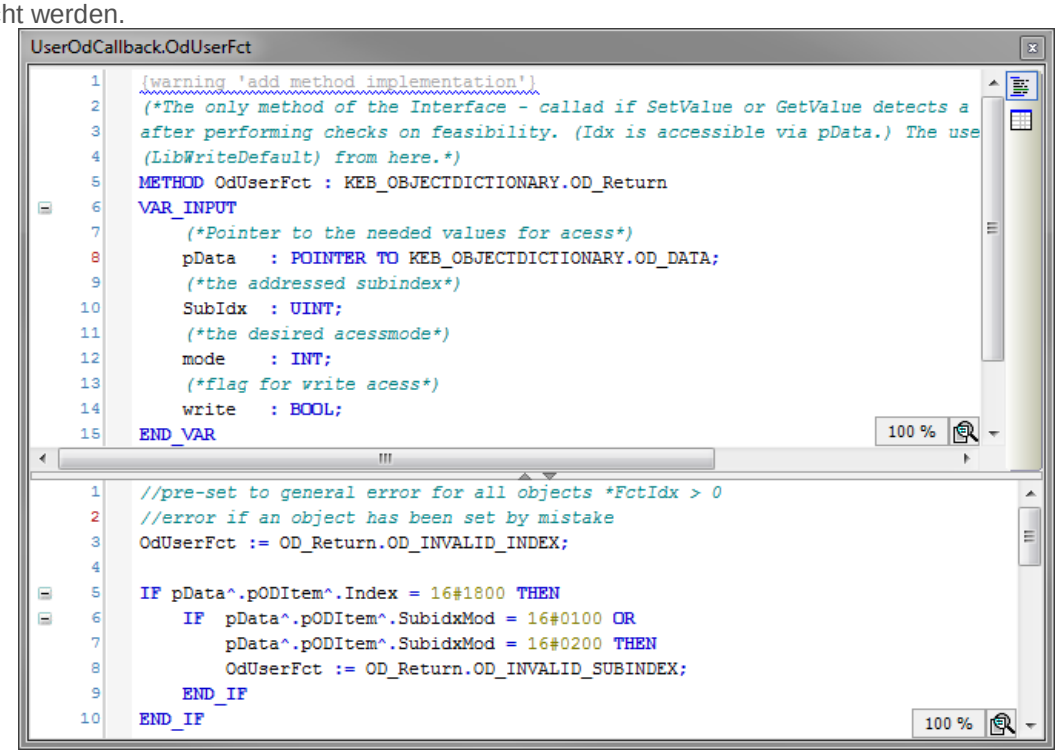
Die Art und Weise, wie man ein benutzerdefiniertes OD erstellt, ist nicht Inhalt dieses Dokuments.

Implementierung der OD Callback Methode

Nun muss der Inhalt der Interface Methode in der Funktionsblockinstanz um die Anforderungen des in den letzten Anschnitt vorgestellten Beispiels geändert werden.

Das Hauptziel ist es, das Verhalten des OD zu überschreiben, bei dem die Methode registriert wurden. Durch den Aufruf der Methoden GetValue() und SetValue() des OD werden die Funktionsindexwerte des aktuellen Unterobjekts überprüft und, falls sie geändert wurden, die Callback-Methode ausgelöst. Sein OD_RETURN-Wert überschreibt den Rückgabewert der internen OD Methoden und kehrt zum Aufrufer des OD (z.B. dem KEB Channelhandler) zurück.

Die erste Bemerkung, auf die man sich konzentrieren sollte, ist die Warnanweisung über der Variablendeklaration. Es dient der Erinnerung und kann bei der Einrichtung einer Implementierung gelöscht werden.



Beispielhafte Implementierung der Callback Methode OdUserFct() für zwei Teilobjekte

Die VAR_INPUT pData enthält die eigentlichen Objekteigenschaften. Auf diese Struktur muss zugegriffen werden, um eine benutzerdefinierte Operation auszuführen und Rückgabewerte in Abhängigkeit von der Objektidentität mit dem Index, dem Subindex und weiteren Werten einzustellen. Beachten Sie, dass der Subindexwert des MSB (most significant byte) des OD_DATA Elements Subidx_Mod ist.

Im Falle des dritten Unterobjekts von der Liste im Abschnitt OD Objektwerte: Gültigkeit und Zugriff , mit dem Rückgabewert „OD_INVALID_SUBINDEX“ kann der Zugriff auf das Benutzer-OD für die duplizierten Unterobjekte auf diese Weise abgelehnt werden.
Es ist gut, den Rückgabewert zu einem beliebigen, allgemeinen Element der Aufzählung OD_Return zu initiieren, da, wenn ein Funktionsindexwert versehentlich gesetzt wurde, er durch überprüfung dieses Wertes erkannt werden konnte.
Für jede verzeichnisspezifische Prozedur muss das Schreiben VAR_INPUT interpretier werden.

Umleitungsergebnisse

Anfragen für jedes (Unter-)Objekt, das in nur einem der ODs enthalten ist, werden an den passenden OD weitergeleitet. Anfragen für ein fehlendes Objekt werden abgelehnt.

Die Callback Methode wird ausgelöst, wenn die Methode GetValue() oder SetValue() des OD einen Lese- oder Schreibfunktion Indexwert >0 im aktuellen Objekt erkennt. Anschließend kann der Rückgabewert des OD durch die Methode mit einem Fehlerwert überschrieben werden, der den Zugriff auf das aktuelle OD für das aktuelle (Unter-)Objekt ablehnt.

Wenn beide Funktionsindexwerte gleich 0 sind, sind nur die OD Standardmethoden für den Rückgabewert verantwortlich. Wurde ein Funktionsindexwert versehentlich ungleich 0 gesetzt, kann er durch einen unerwarteten OD_RETURN Wert erkannt werden (falls dieser voreingestellt ist), da die Callback Methoden niemals ausgelöst werden sollen, wenn der Benutzer ein Objekt nur mit den OD Standardmethoden lesen möchte.

Einschränkungen

Wie in der Einleitung erwähnt, ist diese Lösung nur für das entwickelte CANopen Slave Gerät anwendbar. Das Verhalten anderer Geräte kann sich in wichtigen Details unterscheiden und muss für jeden Einsatzzweck überprüft werden.

Eine Duplizierung von Objekten ist nur für eine fortlaufende Anzahl von Unterobjekten möglich. Diese muss bei Bedarf geschaffen und organisiert werden.

Disclaimer

KEB Automation KG reserves the right to change/adapt specifications and technical data without prior notification. The safety and warning reference specified in this manual is not exhaustive. Although the manual and the information contained in it is made with care, KEB does not accept responsibility for misprint or other errors or resulting damages. The marks and product names are trademarks or registered trademarks of the respective title owners.

The information contained in the technical documentation, as well as any user-specific advice in verbal or in written form are made to the best of our knowledge and information about the application. However, they are considered for information only without responsibility. This also applies to any violation of industrial property rights of a third-party.

Inspection of our units in view of their suitability for the intended use must be done generally by the user. Inspections are particular necessary, if changes are executed, which serve for the further development or adaption of our products to the applications (hardware, software or download lists). Inspections must be repeated completely, even if only parts of hardware, software or download lists are modified.

Application and use of our units in the target products is outside of our control and therefore lies exclusively in the area of responsibility of the user.

KEB Automation KG
Südstraße 38 • D-32683 Barntrup
fon: +49 5263 401-0 • fax: +49 5263 401-116
net: www.keb.de • mail: info@keb.de