BECKHOFF New Automation Technology

Handbuch | DE

TwinCAT 3

Q-Sys - QRC

13.12.2022 | Version: 1.3.2

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, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche 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 weiter entwickelt.
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®

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

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

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.

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ WARNUNG

Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen geschädigt werden!

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

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 Übersicht

QSC ist ein professioneller Anbieter von Audio-/Videosystemlösungen. Seine softwarebasierte Plattform heißt Q-SYS. Q-SYS soll Drittsystemen die Steuerung und/oder Überwachung verschiedener Aspekte des Systems ermöglichen, indem mit Hilfe verschiedener Kommunikationsprotokolle ein eigener Code geschrieben wird.

Die Q-SYS-Software unterstützt folgende Arten der externen Steuerung:

- Named Controls (Benannte Steuerungen) – Steuerungen, die im Bereich Named Controls hinterlegt worden sind. Die Namen der Steuerungen müssen unterschiedlich sein. [Dies ist Teil der Steuerungsebene von Q-SYS.]
- Component Control (Komponentensteuerung) – Ansteuerung aller Steuerungen innerhalb einer Komponente, indem der Name der Komponente so angepasst wird, dass er eindeutig ist. [Dies ist Teil der Komponentenebene von Q-SYS.]
- Mixer Control (Mischersteuerung) – Spezialisierte Steuerung von Mischern mit Hilfe von Mischerkonzepten. [Dies ist Teil der Komponentenebene von Q-SYS.]

Grundsätzlich gibt es zwei verschiedene Protokolle, die von QSC bereitgestellt werden, um auf die drei oben genannten externen Steuerungen für Q-SYS zuzugreifen. Diese heißen „Q-SYS External Control Protocol“ und „**Q**-SYS **R**emote **C**ontrol“ (im Folgenden QRC).

- **Q-SYS External Control Protocol:**

  Q-SYS External Control Protocol basiert auf ASCII und nutzt eine TCP/IP-Verbindung auf Port 1702. Es erfordert die Verwendung von Named Controls für jede Steuerung, die extern angesteuert werden soll. Dies bedeutet, dass es nur die Funktionen der Steuerungsebene von Q-SYS unterstützt.

- **QRC:**

  QRC ist das neueste und fortschrittlichste Protokoll von QSC, das einem externen Steuerungssystem (z. B. TwinCAT) die Steuerung verschiedener Funktionen in Q-SYS ermöglicht. Das QRC-Protokoll basiert auf JSON-RPC Version 2.0 und nutzt eine TCP/IP-Verbindung auf Port

1710. QRC unterstützt die Verwendung aller drei oben genannten Steuerungen: Named Controls, Component Control und Mixer Control. Auf dieser Grundlage ermöglicht es den externen Zugriff auf Steuerungsebene und Komponentenebene.

Voraussetzung für den externen Zugriff auf **Steuerungsebene** ist, dass jede Steuerung in Q-SYS, die angesteuert werden soll, in den Bereich Named Controls gezogen werden und ihr Name eindeutig sein muss.

In diesem Dokument wird erläutert, wie QRC mit Beckhoff-Steuerungen (TwinCAT-Software) verwendet werden kann. Ein Beispielcode mit der Bezeichnung Bibliothek `Tc3_Qrc` wird ebenfalls beigefügt.

Die Bibliothek `Tc3_Qrc` ermöglicht die Implementierung eines oder mehrerer externer QRC-Clients in der TwinCAT-SPS. Mit ihrer Hilfe kann ein Q-SYS Core direkt aus einem TwinCAT-Programm angesteuert werden.

QRC-Steuerungen können beliebigen Datentypen in TwinCAT zugeordnet werden. Dadurch ergibt sich ein breites Spektrum an Kommunikationsmöglichkeiten für den Systemintegrator.

Die QRC-Spezifikation findet sich hier.

Die QRC-Spezifikation und ihre Eigenschaften werden von QSC gestaltet und entwickelt, die Spezifikation kann in Zukunft geändert werden.

Weitere Informationen über die Aktivitäten von Beckhoff im Marktsegment Bühne und Show befinden sich auf unserer Website unter: PC-based Control für Bühnen- und Showtechnik

Systemanforderungen:

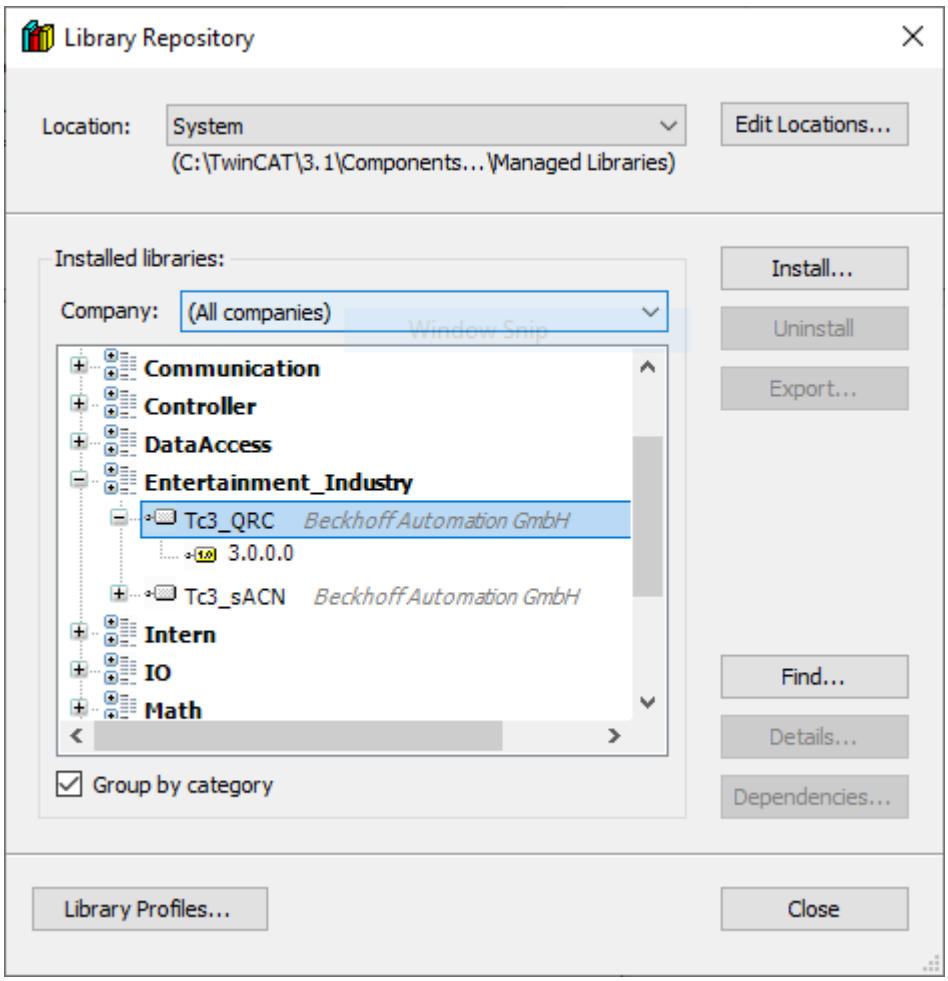
Technische Daten	Anforderung
TwinCAT-Version	TwinCAT 3,1 Build 4022.20 oder höher
Visual Studio-Version	Visual Studio 2013 oder höher
Erforderliche TwinCAT-Lizenz	TF6310-Lizenz

2.1 Update History

Version 3.0.0.0] – 15.12.2020

Geändert:

- Die Hauptversionsnummer dieser Bibliothek wurde aufgrund von TwinCAT 3 in 3.x.x.x geändert.
- Diese Bibliotheksdatei wurde in die Bibliothekskategorie „Entertainment_Industry“ verschoben.



[Version 1.1.2.0] - 12.11.2020

Hinzugefügt:

- Eine Methode FB_exit [▶ 16] zur Online-Änderung der Eingabeparameter von FB_init [▶ 14] hinzugefügt.

Geändert:

- Behebung eines Fehlers.

[Version 1.1.0.0] - 10.03.2020

Hinzugefügt:

- Einen neuen Funktionsbaustein FB_QRC_Snapshot [▶ 39] für Snapshot Bank hinzugefügt.
- Erweiterung des Funktionsbausteins FB_QRC_ChangeGroup [▶ 28] mit einer zusätzlichen Methode AddSnapshotControl [▶ 32] zum Hinzufügen einer Snapshot-Komponente zu einer Änderungsgruppe.
- Unterstützung für das Auslesen des Frames einer Snapshot-Steuerung hinzugefügt. Lesen Sie den Abschnitt Arbeitsablauf für das Auslesen von Snapshot-Eigenschaften [▶ 21] für weitere Informationen.
- Erweiterung des Funktionsbausteins FB_QRC_ResExtract [▶ 17] mit einer zusätzlichen Methode Clear [▶ 18] zum Löschen des internen Speichers.
- Einen Modifikator für jede Methode hinzugefügt. (Auf interne Methoden kann ab dieser Version nicht mehr zugegriffen werden.)
- Eigenschaft `sTxFrame` zu allen Funktionsbausteinen der QRC-Befehle hinzugefügt, um den QRC-Sende-Frame ohne den Verbindungsbaustein einfach lesen zu können.

Geändert:

- Anpassung der Eingangsvariable des Funktionsbausteins FB_QRC_LoopPlayer [▶ 37].

- Anpassung der Severity einiger Ereignisse.
- Anpassung des Variablennamens und Typs der Struktur ST_FileSpec [▶ 42] und ST_JobSpec [▶ 42] zum besseren Verständnis.
- Anpassung des Präfixes des Eigenschaftsnamens mit dem Typ ARRAY zur Angleichung an die TwinCAT 3-Programmierkonventionen.
- Behebung eines Fehlers.

Gestrichen:

- Streichung der Methode Get aus der Eigenschaft sTxFrame von I_Connect [▶ 15].

3 Installation

Der Q-SYS Core wird als Server und die TwinCAT-Automatisierungsplattform als Client betrachtet. Der Q-SYS Core sollte eine Q-SYS-Designdatei laden und in den Run-Modus wechseln, um sich mit der TwinCAT-Automatisierungsplattform zu verbinden. (In Q-SYS Designer wird dieser Prozess als Run-Modus bezeichnet).

Wenn keine Q-SYS-Hardware verfügbar ist, kann alternativ ohne Hardware mit der Q-SYS Designer Software eine Q-SYS-Designdatei simuliert werden (in Q-SYS Designer wird der Simulationsprozess als Emulate-Modus bezeichnet). Weitere Informationen befinden sich auf der Website Q-SYS Help Portal.

Vor Verwendung dieser Bibliothek `tc3_Qrc` müssen die Zielsteuerungen und -komponenten in Q-SYS Designer eingerichtet werden:

- Bei Zielsteuerungen müssen diese in den Bereich Named Controls gezogen werden.
- Bei Zielkomponenten müssen deren Namen angepasst werden und eindeutig sein.
- Bei Mixer Control und Snapshot Control handelt es sich auch um Arten der Komponentensteuerung, die wie Zielkomponenten vorbereitet werden sollten.

In den folgenden Abschnitten stehen die Worte **Q-SYS Gerät** für den Q-SYS Core im Run-Modus oder die Q-SYS Designer Software im Emulate-Modus.

4 Programmierung

Dieses Beispielprojekt besteht im Allgemeinen aus drei Modulen, einem Codierungsmodul, einem Kommunikationsmodul und einem Decodierungsmodul. Außerdem werden zwei Schnittstellen angelegt. Eine Schnittstelle dient dazu, den Datenaustausch zwischen Codierungsmodul und Kommunikationsmodul, die andere Schnittstelle dazu, den Datenaustausch zwischen Kommunikationsmodul und Decodierungsmodul zu ermöglichen.

4.1 Funktionsbausteine

In diesem Projekt sind alle Funktionsbausteine im Wesentlichen in 3 Teile aufgeteilt. Der Funktionsbaustein FB_Connect [▶ 13], der zum Kommunikationsmodul gehört, wird für die Herstellung einer TCP-Verbindung verwendet; 7 Funktionsbausteine [▶ 23], die zum Codierungsmodul gehören, dienen zur Codierung des QRC-Frames. Des Weiteren wird ein Helfer-Funktionsbaustein FB_QRC_ResExtract [▶ 17], der zum Decodierungsmodul gehört, für die Auslesung des QRC-Antwort-Frames verwendet.

4.1.1 FB_Connect

Dieser Funktionsbaustein ermöglicht die Herstellung oder Beendigung einer TCP-Verbindung.

Syntax

```
FUNCTION_BLOCK FB_Connect IMPLEMENTS I_Connect
VAR_OUTPUT
    bBusy                       : BOOL;
    bError                      : BOOL;
    eState                      : E_SocketConnectionState;
    ipResultMessage             : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bBusy	BOOL	Ist gleich TRUE, solange die asynchrone Anforderung noch aktiv ist. Ist gleich FALSE, wenn die Anforderung abgeschlossen wurde oder ein Fehler auftritt.
bBusy	BOOL	Wird gesetzt, wenn bei der Ausführung des Funktionsbausteins ein Fehler auftritt. Fehlerdetails befinden sich im Fenster „Error List“.
eState	E_SocketConnectionState	Gibt den aktuellen Verbindungsstatus zurück. • eSOCKET_DISCONNECTED: Unterbrochen • eSOCKET_SUSPENDED: Status zwischen verbunden und unterbrochen • eSOCKET_CONNECTED: Verbunden
ipResult Message	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem `Tc3_EventLogger`.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
Connect	Eine TCP-Verbindung herstellen.
Disconnect	Eine TCP-Verbindung beenden.
Send	Den QRC-Frame senden.
Receive	Den QRC-Frame empfangen.
FB_exit	Online-Change-Methode

Da alle Methoden asynchron sind und mehr als einen Zyklus benötigen, um abgeschlossen zu werden, könnte nur jeweils eine Methode aufgerufen werden. Prüfen Sie daher den Ausgabeparameter `bBusy`, wenn eine dieser Methoden aufgerufen wird.

Eigenschaften

Eigenschaften	Typ	Zugriff	Beschreibung
aRxFrame	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString	Get	Sobald die fallende Flanke von `bBusy` auftritt und `bError` gleich FALSE ist, kann der empfangene QRC-Antwort-Frame mit dieser Eigenschaft abgefragt werden.
sTxFrame	STRING(QRC_BUFFER_SIZE)	Set	Sobald die fallende Flanke von `bBusy` auftritt und `bError` gleich FALSE ist, kann der zu sendende QRC-Frame mit dieser Eigenschaft gesetzt werden.

Schnittstelle

Name	Beschreibung
I_Connect	Die Schnittstelle, die kommunikationsbezogene Methoden definiert.

4.1.1.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    sSrvNetID                   : T_AmsNetID := '';
    sRemoteHost                 : T_IPv4Addr := '127.0.0.1';
    tReconnect                  : TIME       := T#30s;
    iResExtract                 : I_ResExtract;
END_VAR
```

VAR_INPUT

sSrvNetID: AMS Net Id. Für den lokalen Rechner (Default) kann auch ein Leerstring angegeben werden.

sRemoteHost: IPv4-Zieladresse.

tReconnect: Abkühlzeit für die Wiederherstellung einer TCP-Verbindung, nachdem eine TCP-Verbindung beendet wurde.

iResExtract: Der Funktionsbaustein, der die Schnittstelle I_ResExtract [▶ 18] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_Connect`:

```
PROGRAM MAIN
VAR
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbResExtract        : FB_QRC_ResExtract;
END_VAR
```

4.1.1.2 I_Connect

METHODEN

Connect: Eine TCP-Verbindung herstellen.

Disconnect: Eine TCP-Verbindung beenden.

Send: QRC-Frames senden.

M_Receive: QRC-Frames empfangen.

EIGENSCHAFTEN

Eigenschaften	Typ	Zugriff	Beschreibung
aRxFrame	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString	Get	Sobald die fallende Flanke von `bBusy` auftritt und `bError` gleich FALSE ist, kann der empfangene QRC-Antwort-Frame mit dieser Eigenschaft abgefragt werden.
sTxFrame	STRING(QRC_BUFFER_SIZE)	Set	Sobald die fallende Flanke von `bBusy` auftritt und `bError` gleich FALSE ist, kann der zu sendende QRC-Frame gesetzt werden.

Connect

Diese Methode ermöglicht die Herstellung einer TCP-Verbindung.

```
Method Connect          : BOOL
```

Dieser Prozess ist abgeschlossen, sobald der Rückgabewert TRUE ist.

Disonnect

Diese Methode ermöglicht die Beendigung einer TCP-Verbindung.

```
Method Disonnect        : BOOL
```

Dieser Prozess ist abgeschlossen, sobald der Rückgabewert TRUE ist.

Send

Diese Methode ermöglicht das Senden eines QRC-Frames und das automatische Abrufen des Antwort-Frames vom Q-SYS-Gerät nach dem Senden.

```
Method Send             : I_ResExtract
```

Diese Methode ist abgeschlossen, sobald die fallende Flanke von `bBusy` auftritt und die Eigenschaft `aRxFrame` nicht leer ist. Der Antwort-Frame kann bei der Eigenschaft `aRxFrame` abgerufen werden.

Receive

Diese Methode ermöglicht den Empfang eines QRC-Frames.

```
Method Receive            : I_ResExtract
```

Diese Methode ist abgeschlossen, sobald der fallende Trigger von `bBusy` ausgelöst wird und die Eigenschaft `aRxFrame` nicht leer ist. Der Antwort-Frame kann bei der Eigenschaft `aRxFrame` abgerufen werden.

aRxFrame

Liste der empfangenen QRC-Antwort-Frames.

```
PROPERTY aRxFrame       : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString
```

sTxFrame

Ein QRC-Frame, der bereit zum Senden an das Q-SYS-Gerät ist.

```
PROPERTY sTxFrame         : STRING(QRC_BUFFER_SIZE)
```

4.1.1.3 FB_exit

Syntax

```
Method FB_exit : BOOL
```

Variablen, die am Eingang von `FB_init` angegeben werden, können nach dem Aufruf dieser Methode online geändert werden. Normalerweise kann diese Methode verwendet werden, um dynamisch Verbindungen zu mehreren Q-SYS-Kernen herzustellen.

Beispiel:

Schalten Sie den Zielserver von „192.168.0.110“ auf „192.168.0.100“:

```
PROGRAM MAIN
VAR
    fbConnect           : FB_Connect('', '192.168.1.110', T#15S, fbResExtract);
    fbResExtract        : FB_QRC_ResExtract;
    nStep               : INT;
    bChangeTarget       : BOOL;
END_VAR
```

```
CASE nStep OF
    0:
        fbConnect.Connect();
        IF NOT fbConnect.bBusy AND NOT fbConnect.bError THEN
            nStep := nStep + 1;
        END_IF
    1:
        IF bChangeTarget THEN
            bChangeTarget := FALSE;
            nStep := nStep + 1;
        ELSE
            nStep := 3;
        END_IF
    2:
        fbConnect.FB_exit(FALSE);
        fbConnect.FB_init(FALSE, FALSE, '', '192.168.0.100', T#15S, fbResExtract);
        nStep := 0;
    3:
        (*Rest of Codes*)
END_CASE
```

4.1.2 FB_QRC_ResExtract

Dieser Funktionsbaustein ermöglicht die Auslesung der empfangenen QRC-Frames.

Dieser Auslesebaustein ist nur für QRC-Antwort-Frames der folgenden QRC-Befehle ausgelegt:

- Befehl Status.Get [▶ 24]
- Steuerungsbezogene Befehle (Control.Set [▶ 26] & Control.Get [▶ 25])
- Komponentenbezogene Befehle (Component.Set [▶ 27] & Component.Get [▶ 27])
- „Change Control“-bezogene Befehle (Alle Methoden von FB_QRC_ChangeGroup [▶ 28])
- Snapshot-Komponente (weitere Informationen befinden sich im Abschnitt Steuerschaltfläche ‚Load‘ der Snapshot-Komponente [▶ 53] und Snapshot-Status und zugehörige Eigenschaften [▶ 53])

Die Antwort-Frames anderer QRC-Befehle können mit der Eigenschaft `aRxFrame` direkt abgerufen werden.

Syntax

```
FUNCTION_BLOCK FB_QRC_ResExtract IMPLEMENTS I_ResExtract
VAR_OUTPUT
    bError                  : BOOL;
    ipResultMessage         : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn bei der Ausführung des Funktionsbausteins ein Fehler auftritt. Fehlerdetails befinden sich im Fenster „Error List“.
ipResult Message	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem `Tc3_EventLogger`.

Methoden

Name	Beschreibung
ResExtract	Auslesen der empfangenen QRC-Antwort-Frames.
Clear	Löschen des internen Speichers.

Eigenschaften

Eigenschaften	Typ	Zugriff	Beschreibung
aCtrlProp	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_ControlEx	Get	Mit dieser Eigenschaft können ausgelesene Steuerungseigenschaften abgefragt werden.
aRxFrame	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString	Set, Get	Mit dieser Eigenschaft kann die Auslesung von QRC-Frames gesetzt oder abgefragt werden.
sEngineStatus	T_MaxString	Get	Mit dieser Eigenschaft können Q-SYS-Geräteinformationen abgefragt werden.

Schnittstelle

Name	Beschreibung
I_ResExtract	Die Schnittstelle, die die Auslesemethode definiert.

Sehen Sie dazu auch

Das Attribut bSavOldRes [▶ 19]

4.1.2.1 sEngineStatus

Diese Eigenschaft ermöglicht die Abfrage der Statusinformationen des Q-SYS-Geräts.

Syntax

```
PROPERTY sEngineStatus : T_MaxString
```

4.1.2.2 Clear

Diese Methode ermöglicht das Löschen aller gespeicherten Snapshot-Eigenschaften, die über Poll [▶ 30] oder AutoPoll [▶ 32] abgefragt wurden.

Syntax

```
METHOD Clear : BOOL
```

Diese Methode ist sinnvoll, wenn der verwendete Snapshot veraltet ist. Lesen Sie den Abschnitt Arbeitsablauf für das Auslesen von Snapshot-Eigenschaften [▶ 21] für weitere Informationen.

4.1.2.3 I_ResExtract

METHODEN

ResExtract: Empfangene QRC-Antwort-Frames aus dem Q-SYS-Gerät auslesen.

EIGENSCHAFTEN

Eigenschaften	Typ	Zugriff	Beschreibung
aCtrlProp	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_ControlEx	Get	Mit dieser Eigenschaft werden die ausgelesenen Steuerungseigenschaften abgerufen.
aRxFrame	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString	Set, Get	Mit dieser Eigenschaft können auszulesende QRC-Frames gesetzt oder abgerufen werden.

aCtrlProp

Liste der Steuerungseigenschaften, die durch ResExtract ausgelesen wurden.

```
PROPERTY aCtrlProp       : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_ControlEx
```

arrRxFrame

Mit dieser Eigenschaft kann ein QRC-Antwort-Frame, der bereit zum Auslesen ist, gesetzt oder abgerufen werden.

Wie bereits erwähnt [▶ 17], kann dieser Funktionsbaustein begrenzte Typen von QRC-Antwort-Frames auslesen. Ein Antwort-Frame, der nicht durch den Funktionsbaustein ausgelesen werden kann, kann vor der Auslesung mit der „Getter“-Funktion abgerufen werden. Des Weiteren können Benutzer auch ihren eigenen QRC-Frame bei der „Setter“-Funktion schreiben, um Informationen aus ihrem eigenen QRC-Frame auszulesen.

```
PROPERTY aRxFrame        : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString
```

4.1.2.3.1 ResExtract

ResExtract

Diese Methode ermöglicht die Auslesung von Steuerungseigenschaften aus einem QRC-Antwort-Frame.

Syntax

```
METHOD ResExtract : BOOL
VAR_INPUT
    bSavOldRes      : BOOL;
END_VAR
```

VAR_INPUT

bSavOldRes: Diese Variable bestimmt, ob der referenzierte Funktionsbaustein vergangene Steuerungseigenschaften sichert, die aus früheren QRC-Frames ausgelesen wurden. Weitere Informationen befinden sich im Abschnitt „Das Attribut bSavOldRes [▶ 19]“.

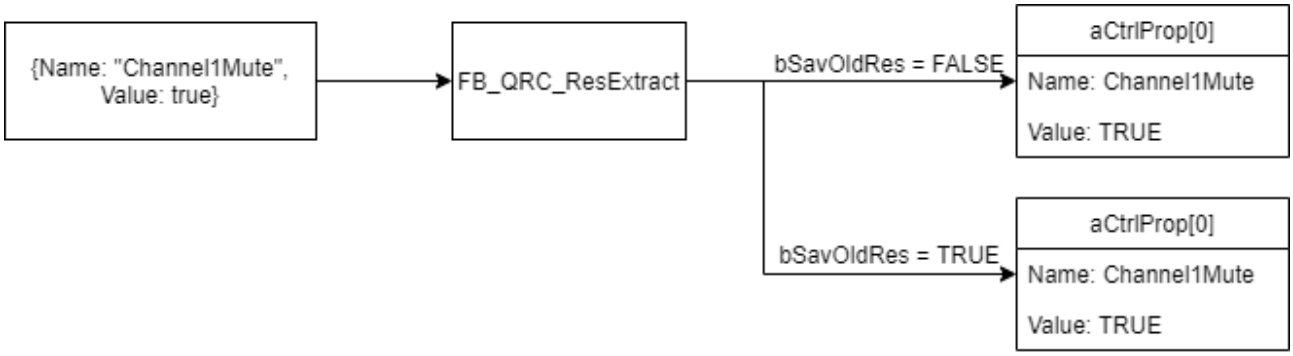
Das Attribut bSavOldRes

Die Eingangsvariable `bSavOldRes` der Methode `ResExtract` wurde implementiert, um eine Konfiguration empfangener Steuerungsinformationen zu ermöglichen. Das Array `aCtrlProp` kann die Anzahl QRC_NUMBER_OF_CONTROL der Steuerungsinformationen speichern. Dieses Attribut kann in der Parameterliste [▶ 43] geändert werden.

- Setzt man das Attribut `bSavOldRes` auf TRUE, werden alle vergangenen Steuerungsinformationen gespeichert. Wenn Steuerungsinformationen anstehen, die bereits gespeichert sind, werden die alten Steuerungsinformationen durch die neuen überschrieben.
- Setzt man das Attribut `bSavOldRes` auf FALSE, werden alle vergangenen Steuerungsinformationen, die im Array `aCtrlProp` gespeichert waren, gelöscht. Nur die neuesten Steuerungsinformationen werden gespeichert.

Für ein besseres Verständnis des Verhaltens wird nachstehend ein Beispiel gezeigt.

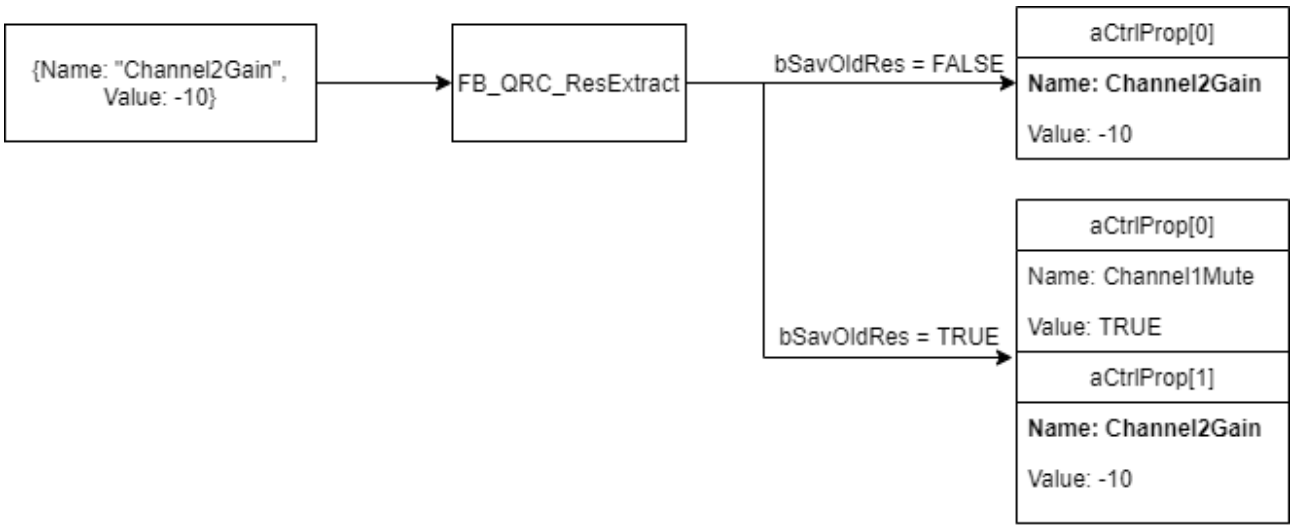
1. Schritt: Steuerungsinformationen von „**Channel1Mute**“ empfangen.



In Schritt 1 wurde ein QRC-Frame bei `aRxFrame` empfangen und die Steuerungsinformationen werden durch FB_QRC_RecExtract [▶ 17] ausgelesen.

Das Array `aCtrlProp` ist leer. Deshalb wird die Steuerung **Channel1Mute** beim Element `aCtrlProp[0]` gespeichert, unabhängig davon, ob `bSavOldRes` gleich TRUE ist oder nicht.

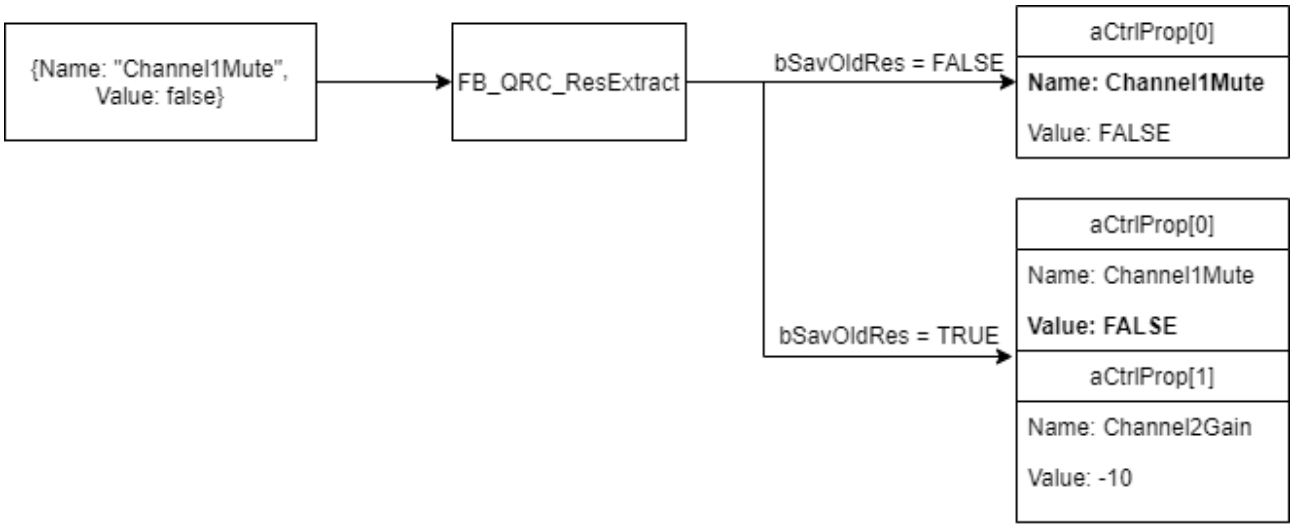
2. Schritt: Steuerungsinformationen von „**Channel2Gain**“ (andere Steuerung) empfangen.



In Schritt 2 wurde ein zweiter QRC-Frame empfangen. Nach der Auslesung der neuen Steuerungsinformationen werden sie abhängig vom Wert von `bSavOldRes` gespeichert.

- Wenn `bSavOldRes` gleich TRUE ist, wird die Steuerung **Channel2Gain** beim Element `aRecProp[1]` gespeichert, weil `aCtrlProp[0]` die andere Steuerung Channel1Mute gespeichert hat.
- Wenn `bSavOldRes` gleich FALSE ist, wird die Steuerung **Channel2Gain** beim Element `aCtrlProp[0]` gespeichert. Die Steuerungsinformationen von Channel1Mute, die beim selben Element gespeichert waren, werden überschrieben.

3. Schritt: Steuerungsinformationen von „**Channel1Mute**“ (Aktualisierung der bereits empfangenen Steuerung) empfangen.



In Schritt 3 wurde ein dritter QRC-Frame empfangen. Nach der Auslesung wurde erkannt, dass der Steuerungsname bereits bei `aCtrlProp[0]` gespeichert war:

- Wenn `bSavOldRes` gleich TRUE ist, werden die neu eingehenden Informationen von **Channel1Mute** beim Element `aRecProp[0]` gespeichert. Dadurch werden die gespeicherten Steuerungsinformationen von **Channel1Mute** aktualisiert und die bei `aCtrlProp[1]` gespeicherten Steuerungsinformationen beibehalten.
- Wenn `bSavOldRes` gleich FALSE ist, werden die neu eingehenden Steuerungsinformationen von **Channel1Mute** beim Element `aCtrlProp[0]` gespeichert. Andere gespeicherte Informationen werden gelöscht.

Alle vergangenen Steuerungseigenschaften werden nur gespeichert, wenn `bSavOldRes` gleich TRUE ist. Falls `bSavOldRes` gleich FALSE ist, werden alle vergangenen Steuerungsinformationen gelöscht.

Arbeitsablauf für das Auslesen von Snapshot-Eigenschaften

Es gibt zwei Möglichkeiten, um einen Snapshot-Status abzufragen – das manuelle Abfragen mit der Methode GetSnapshotState [▶ 41] oder das Beifügen zu einer Änderungsgruppe und Pollen ihrer Änderungen. Basierend auf dem Funktionsprinzip einer Änderungsgruppe meldet die Polling-Funktion in einem Polling-Zyklus nur an die geänderte Steuerung. In einigen Fällen ist es unmöglich, einen Snapshot-Status zu bestimmen. (Ändert sich z. B. ein Snapshot von „geladen“ in „geändert“, dann meldet das Q-SYS-Gerät nur, dass sich die Steuerung „`match`“ von „true“ in „false“ geändert hat. Die andere zugehörige Steuerung „`last`“ bleibt „true“.) Bei jeder Verwendung der Methode GetSnapshotState [▶ 41] werden jedoch alle zugehörigen Steuerungen eines angeforderten Snapshots abgefragt. Mit den vollständigen Informationen kann der Snapshot-Status immer bestimmt werden.

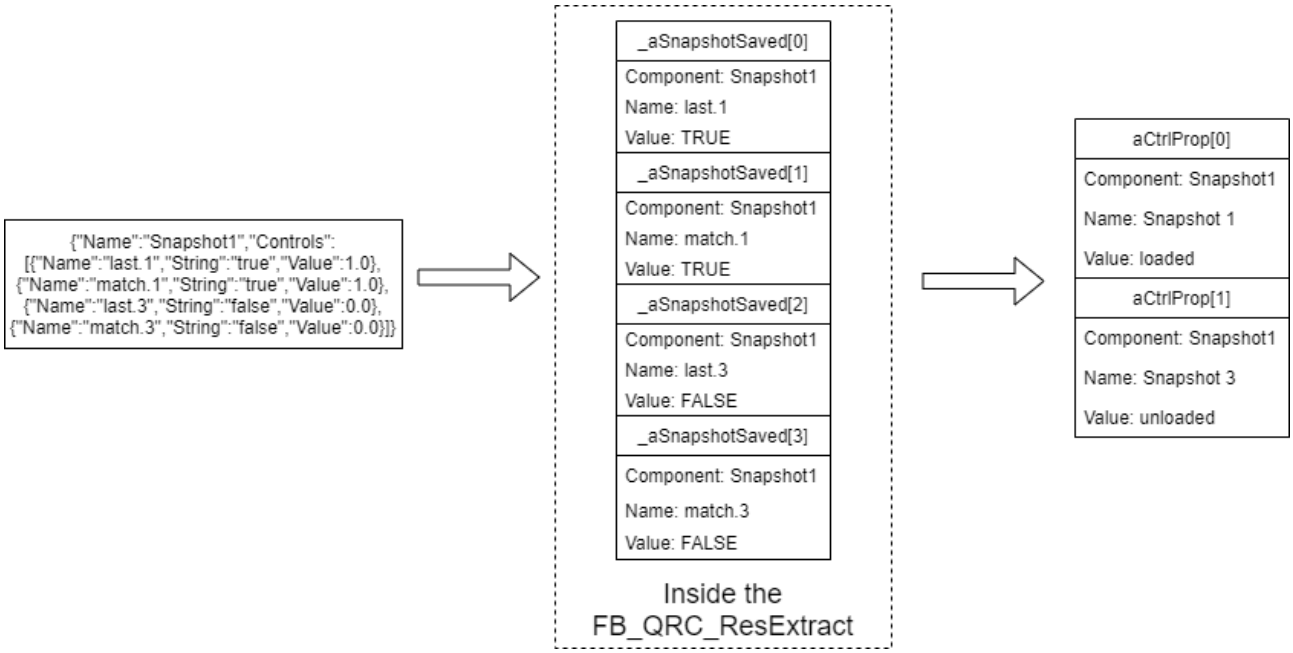
Da jede Snapshot-Eigenschaft, die von einer Polling-Funktion (Poll [▶ 30] oder AutoPoll [▶ 32]) abgefragt wird, intern gespeichert wird, kann die Methode Clear [▶ 31] des Funktionsbausteins FB_QRC_ResExtract [▶ 17] verwendet werden, um diesen Speicher freizugeben.

Nach dem Eintreffen eines Antwort-Frames von einem Q-SYS-Gerät werden alle Snapshot-Steuerungseigenschaften, die durch die Polling-Methode abgerufen werden, intern gespeichert. (Das Attribut `bSavOldRes` hat darauf KEINE Auswirkungen.) Die Snapshot-Steuerungseigenschaften werden aktualisiert. Mit Hilfe der Methode Clear können diese Eigenschaften gelöscht werden.

> Diese Logik hat keine Auswirkungen auf die Logik `bSavOldRes`, die im Abschnitt über das Attribut `bSavOldRes` beschrieben wird. Die Benutzer können jedoch auch `bSavOldRes` auf **TRUE** setzen, um die Steuerungseigenschaften beim Element `aCtrlProp` zu speichern.

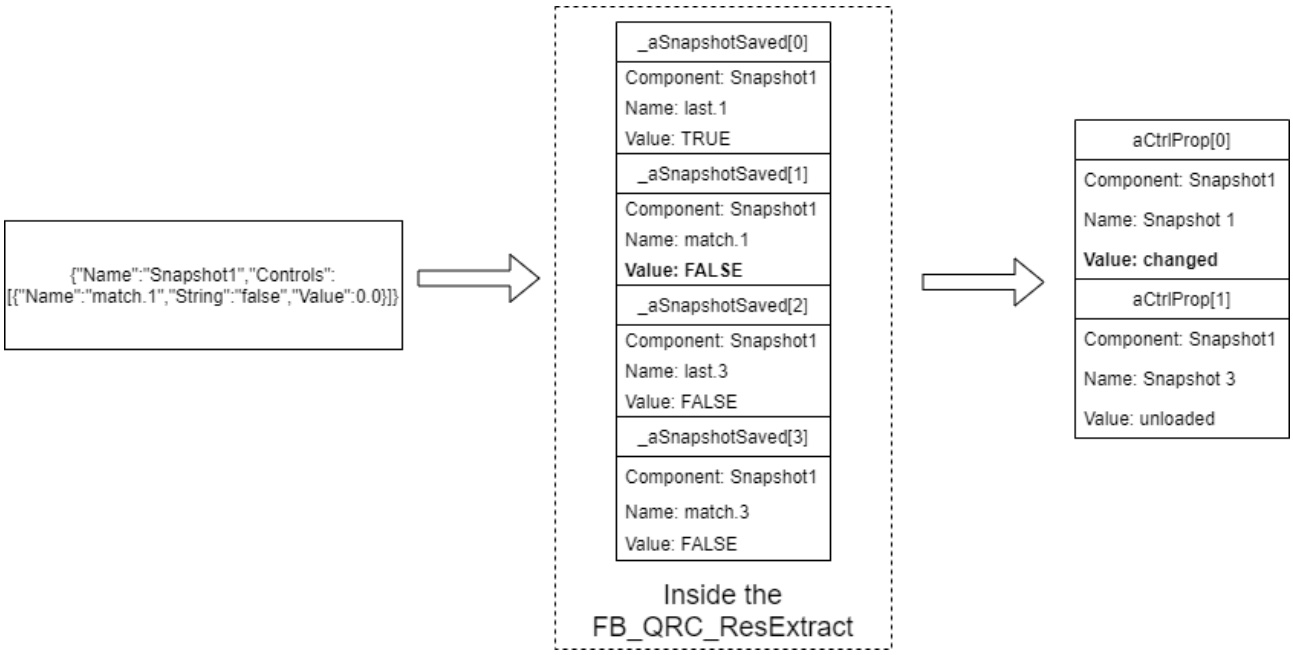
Für ein besseres Verständnis dieses Verhaltens wird nachstehend ein Beispiel gezeigt.

Schritt 1: Nachdem Snapshot 1 und 3 (der Name der Snapshot Bank lautet „Bank1“, der Name der Snapshot-Komponente lautet „Snapshot1“) der Änderungsgruppe („ChangeGroup 1“) beigefügt wurden, wurde der Antwort-Frame empfangen:



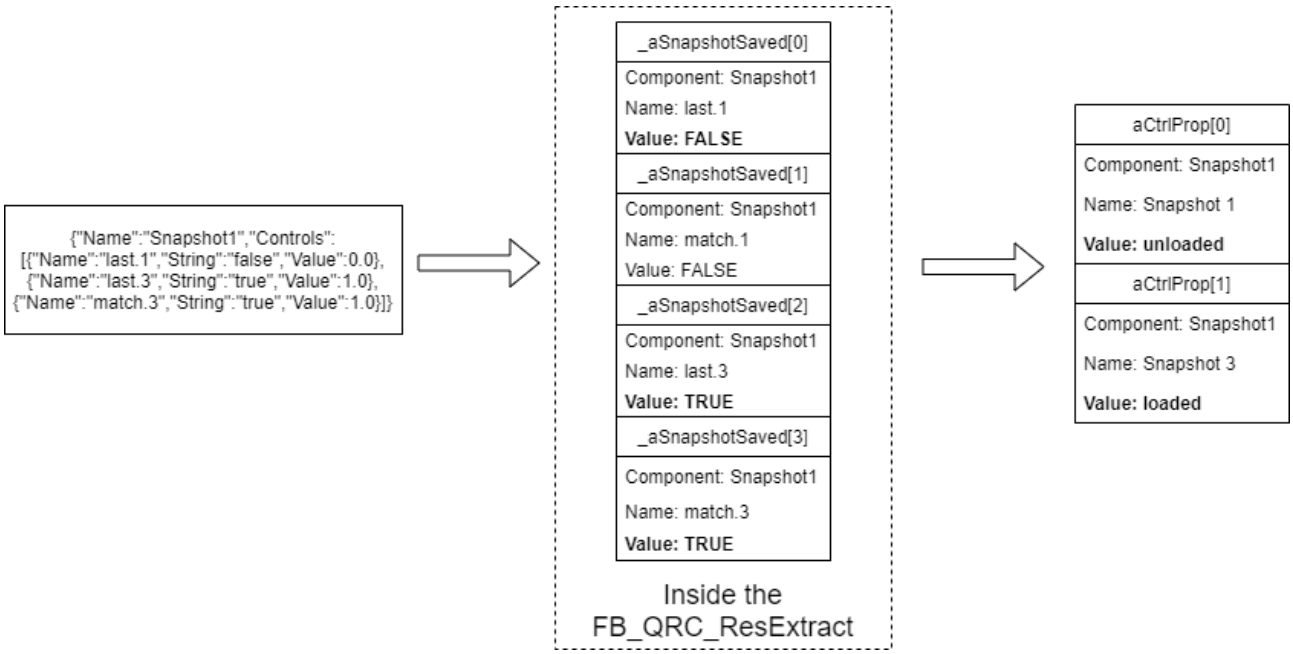
Alle zugehörigen Informationen werden intern in einem Array gespeichert. Die Status der Snapshots werden bestimmt.

Schritt 2: Falls in einem Snapshot enthaltene Steuerungen in einem Polling-Zyklus geändert wurden, trifft ein Polling-Frame ein:



Die Eigenschaft „`match.1`“ wird im internen Array aktualisiert und der Snapshot „Snapshot 1“ ändert seinen Status von „geladen“ in „geändert“. (`aCtrlProp[0]`)

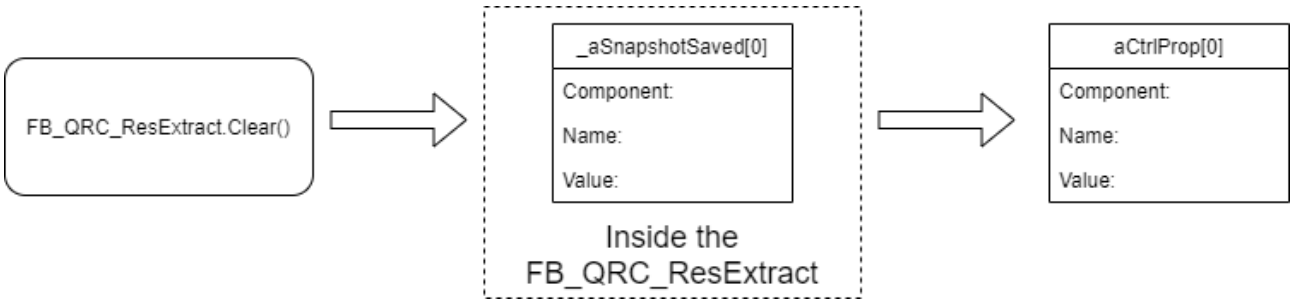
Schritt 3: Snapshot 3 wird ausgelöst.



Der „Snapshot 3“ wurde soeben ausgelöst und der Polling-Frame wurde empfangen. Die zugehörigen Steuerungseigenschaften werden aktualisiert.

Schritt 4: Löschen des internen Arrays.

Wenn Benutzer den Status eines anderen Snapshots pollen möchten und die gespeicherten Eigenschaften nicht mehr nützlich sind, sollte das interne Array mit Hilfe der Methode Clear [▶ 31] zurückgesetzt werden.



Nach dem Löschvorgang sind das interne Array und das Array `aCtrlProp` beide leer.

4.1.3 QRC-Befehle

In den folgenden Abschnitten werden 6 Funktionsbausteine, die sich im Ordner „QRC_Application“ der Bibliothek `Tc3_Qrc` befinden, ausgehend von der QRC-Spezifikation angelegt. Jeder Funktionsbaustein hat dieselbe Methode FB_init [▶ 23], und jede Methode, die die QRC-Spezifikation implementiert hat, hat denselben Rückgabetyp I_Connect [▶ 15].

4.1.3.1 FB_QRC_BasicCommand

Dieser Funktionsbaustein ermöglicht die Codierung eines QRC-Basisbefehls.

Syntax

```
FUNCTION_BLOCK FB_QRC_BasicCommand
VAR_OUTPUT
    bError           : BOOL;
    ipResultMessage  : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
Logon	Q-SYS-Gerät anmelden.
NoOp	TCP-Verbindung aufrechterhalten.
StatusGet	Aktuellen Status des Q-SYS-Geräts abrufen.

4.1.3.1.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    iConnect                : I_Connect;
END_VAR
```

VAR_INPUT

iConnect: Der Funktionsbaustein, der die Schnittstelle I_Connect [▶ 15] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_QRC_Control`:

```
PROGRAM MAIN
VAR
    fbResExtract        : FB_QRC_ResExtract;
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbQrcControl        : FB_QRC_Control(fbConnect);
END_VAR
```

4.1.3.1.2 LogOn

Diese Methode ermöglicht die Anmeldung des Q-SYS-Geräts.

Syntax

```
METHOD LogOn : I_Connect
VAR_INPUT
    sUserName               : STRING;
    nPassword               : UDINT;
END_VAR
```

VAR_INPUT

sUserName: Benutzername.

nPassword: Passwort.

4.1.3.1.3 NoOp

Diese Methode ermöglicht die Aufrechterhaltung einer TCP-Verbindung.

Syntax

```
METHOD NoOp : I_Connect
```

In `FB_Connect` wird diese Methode intern verwendet, um die TCP-Verbindung aufrechtzuerhalten. Die Zykluszeit der Aufrechterhaltung beträgt 45 Sekunden.

4.1.3.1.4 StatusGet

Diese Methode ermöglicht die Abfrage der Statusinformationen des Q-SYS-Geräts.

Syntax

```
METHOD StatusGet : I_Connect
```

Diese Methode wird vom Q-SYS-Gerät automatisch so implementiert, dass seine Statusinformationen immer zurückgegeben werden, wenn ein Client mit dem Q-SYS-Gerät verbunden wurde oder der Status des Q-SYS-Geräts sich geändert hat. Diese Statusinformationen können durch den Funktionsbaustein `FB_QRC_ResExtract` einfach ausgelesen und abgerufen werden.

4.1.3.2 FB_QRC_Control

Dieser Funktionsbaustein ermöglicht die Codierung von QRC-Frames, die zum Setzen oder Abrufen von Steuerungseigenschaften über Named Controls verwendet werden.

Syntax

```
FUNCTION_BLOCK FB_QRC_Control
VAR_OUTPUT
    bError                      : BOOL;
    ipResultMessage             : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
Get	Steuerungseigenschaften über Named Control abrufen.
Set	Eigenschaften einer Steuerung über Named Control setzen.

4.1.3.2.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    iConnect                : I_Connect;
END_VAR
```

VAR_INPUT

iConnect: Der Funktionsbaustein, der die Schnittstelle I_Connect [▶ 15] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_QRC_Control`:

```
PROGRAM MAIN
VAR
    fbResExtract        : FB_QRC_ResExtract;
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbQrcControl        : FB_QRC_Control(fbConnect);
END_VAR
```

4.1.3.2.2 Get

Diese Methode ermöglicht die Codierung der QRC-Frames, die zum Abrufen von Steuerungseigenschaften über Named Controls verwendet werden.

Syntax

```
METHOD Get : I_Connect
VAR_INPUT
aControlName            : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF STRING;
END_VAR
```

VAR_INPUT

aControlName: Liste der Ziel-Named Controls, die abgefragt werden.

> Named Controls sollten beginnend mit dem ersten Element des Arrays `aControlName` aufgelistet werden.

4.1.3.2.3 Set

Diese Methode ermöglicht die Codierung der QRC-Frames, die zum Setzen von Steuerungseigenschaften über Named Controls verwendet werden.

Syntax

```
METHOD Set : I_Connect
VAR_INPUT
    stControlValue          : ST_Control;
END_VAR
```

VAR_INPUT

stControlValue: Eigenschaften der Ziel-Named Control

4.1.3.3 FB_QRC_Component

Dieser Funktionsbaustein ermöglicht die Codierung der QRC-Frames, die zum Setzen/Abrufen von Steuerungseigenschaften über Named Component verwendet werden. Außerdem wird er zum Auflisten aller vorhandenen Komponenten über Named Component verwendet.

> Definition von Named Component: Named Component ist eine Komponentensteuerung mit einer eindeutigen Namenseigenschaft.

Syntax

```
FUNCTION_BLOCK FB_QRC_Component
VAR_OUTPUT
    bError                  : BOOL;
    ipResultMessage         : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
Set	Steuerungseigenschaften über eine Named Component setzen.
Get	Steuerungseigenschaften über eine Named Component abrufen.
GetComponent	Steuerungseigenschaften aller vorhandenen Named Components in einem Q-SYS Design abrufen.

4.1.3.3.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    iConnect                : I_Connect;
END_VAR
```

VAR_INPUT

iConnect: Der Funktionsbaustein, der die Schnittstelle I_Connect [▶ 15] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_QRC_Control`:

```
PROGRAM MAIN
VAR
    fbResExtract        : FB_QRC_ResExtract;
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbQrcControl        : FB_QRC_Control(fbConnect);
END_VAR
```

4.1.3.3.2 Set

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Setzen einer oder mehrerer Steuerungseigenschaften einer Named Component verwendet wird.

Syntax

```
METHOD Set : I_Connect
VAR_INPUT
    sComponentName              : STRING;
    aControlValue               : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_Control;
END_VAR
```

VAR_INPUT

sComponentName: Namenseigenschaft der Ziel-Named Component.

aControlValue: Zielsteuerungseigenschaften der Named Component.

Die Steuerungseigenschaften sollten beginnend mit dem ersten Element des Arrays `aControlValue` aufgelistet werden.

4.1.3.3.3 Get

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Abrufen einer oder mehrerer Steuerungseigenschaften einer Named Component verwendet wird.

Syntax

```
METHOD Get : I_Connect
VAR_INPUT
    sComponentName            : STRING;
    aControlName              : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF STRING;
END_VAR
```

VAR_INPUT

sComponentName: Namenseigenschaft der Ziel-Named Component.

aControlName: Zielsteuerungsname der Named Component.

> Steuerungsnamen sollten beginnend mit dem ersten Element des Arrays `aControlName` aufgelistet werden.

4.1.3.3.4 GetComponent

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Abrufen von Steuerungseigenschaften aller vorhandenen Named Components verwendet wird.

Syntax

```
METHOD GetComponent : I_Connect
```

> 1. Der Antwort-Frame dieses Befehls kann nicht durch FB_QRC_ResExtract [▶ 17] ausgelesen werden.
> 2. Normalerweise ist der Antwort-Frame dieses Befehls extrem lang (da jede Steuerung von jeder Named Component angegeben wird), bitte beachten Sie die Puffergröße QRC_BUFFER_SIZE [▶ 43].

4.1.3.4 FB_QRC_ChangeGroup

Dieser Funktionsbaustein ermöglicht die Codierung der QRC-Frames, die zum Bearbeiten oder Pollen einer Änderungsgruppe verwendet werden.

Syntax

```
FUNCTION_BLOCK FB_QRC_ChangeGroup
VAR_OUTPUT
    bError                    : BOOL;
    ipResultMessage           : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
AddControl	Eine oder mehrere Steuerungen über Named Controls zu einer Änderungsgruppe hinzufügen.
AddComponent Control	Eine oder mehrere Steuerungen über eine Named Component zu einer Änderungsgruppe hinzufügen.
Remove	Eine oder mehrere Steuerungen aus einer Änderungsgruppe entfernen.
Poll	Eine Änderungsgruppe pollen, um ihre Änderungen abzurufen.
Destroy	Eine Änderungsgruppe löschen.
Clear	Alle Steuerungen aus einer Änderungsgruppe löschen.
Invalidate	Allen Steuerungen vorgeben, ihre Eigenschaften in der nächsten Polling-Runde zu melden.
AutoPoll	Automatisches Polling einrichten.
AddSnapshot Control	Einen oder mehrere Snapshots über Named Snapshot Component zu einer Änderungsgruppe hinzufügen.

1. Eine Änderungsgruppe ist eine Gruppierung von Named Controls oder Named Components. Mit Hilfe dieses Funktionsbausteins können mehr Steuerungseigenschaften mit nur einem QRC-Frame abgerufen werden.

2. Wenn keine Zieländerungsgruppe vorhanden ist, wird sie automatisch erstellt, nachdem der erste Befehl `AddControl`, `AddComponentControl` oder `AddSnapshotControl` vom Q-SYS-Gerät empfangen wurde.

4.1.3.4.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    iConnect                : I_Connect;
END_VAR
```

VAR_INPUT

iConnect: Der Funktionsbaustein, der die Schnittstelle I_Connect [▶ 15] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_QRC_Control`:

```
PROGRAM MAIN
VAR
    fbResExtract        : FB_QRC_ResExtract;
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbQrcControl        : FB_QRC_Control(fbConnect);
END_VAR
```

4.1.3.4.2 AddControl

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Hinzufügen einer oder mehrerer Steuerungen über Named Control zu einer Änderungsgruppe verwendet wird.

Syntax

```
METHOD AddControl : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
    aControlName              : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

aControlName: Liste der Ziel-Named Controls.

Steuerungsnamen sollten beginnend mit dem ersten Element des Arrays `aControlName` aufgelistet werden.

4.1.3.4.3 AddComponentControl

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Hinzufügen einer oder mehrerer Steuerungen innerhalb einer Named Component zu einer Änderungsgruppe verwendet wird.

Syntax

```
METHOD AddComponentControl : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
    sComponentName            : STRING;
    aControlName              : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

sComponentName: Namenseigenschaft der Ziel-Named Component.

aControlName: Zielsteuerungsnamen der Named Component.

Steuerungsnamen sollten beginnend mit dem ersten Element des Arrays `aControlName` aufgelistet werden.

4.1.3.4.4 Remove

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Entfernen einer oder mehrerer Named Controls aus einer Änderungsgruppe verwendet wird.

Syntax

```
METHOD Remove : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
    aControlName              : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

aControlName: Zielsteuerungsnamen.

Steuerungsnamen sollten beginnend mit dem ersten Element des Arrays `aControlName` aufgelistet werden.

4.1.3.4.5 Poll

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Pollen einer Änderungsgruppe verwendet wird.

Syntax

```
METHOD Poll : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

4.1.3.4.6 Destroy

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Zerstören einer Änderungsgruppe verwendet wird. Diese Änderungsgruppe existiert nicht mehr.

Syntax

```
METHOD Destroy : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

Unterschied zwischen der Methode `Destroy` und der Methode `Clear`:

`Clear` wird verwendet, um alle Named Controls/Named Components aus einer Änderungsgruppe zu löschen. Diese Änderungsgruppe existiert weiterhin, ist jedoch leer.

`Destroy` wird verwendet, um eine Änderungsgruppe zu löschen. Diese Änderungsgruppe existiert nach diesem Vorgang nicht mehr.

4.1.3.4.7 Clear

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Löschen aller Named Controls/ Named Components aus einer Änderungsgruppe verwendet wird. Diese Änderungsgruppe existiert weiterhin.

Syntax

```
METHOD Clear : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

Unterschied zwischen der Methode `Destroy` und der Methode `Clear`:

`Clear` wird verwendet, um alle Named Controls/Named Components aus einer Änderungsgruppe zu löschen. Diese Änderungsgruppe existiert weiterhin, ist jedoch leer.

`Destroy` wird verwendet, um eine Änderungsgruppe zu löschen. Diese Änderungsgruppe existiert nach diesem Vorgang nicht mehr.

4.1.3.4.8 Invalidate

Diese Methode ermöglicht die Codierung eines QRC-Frames, der verwendet wird, um alle Named Controls/ Named Components in den Status „Dirty“ zu setzen.

Syntax

```
METHOD Invalidate : I_Connect
VAR_INPUT
    sChangeGroupId            : STRING;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

Interne Funktionsweise der Änderungsgruppe in Q-SYS

Nachdem einer Änderungsgruppe eine neue Steuerung hinzugefügt wurde, wird diese Steuerung mit dem Status „Dirty“ gekennzeichnet, was bedeutet, dass ihre aktuellen Eigenschaften nicht gemeldet sind. Sobald ihre aktuellen Eigenschaften durch die Methoden Poll [▶ 30] oder AutoPoll [▶ 32] gemeldet werden, ändert sich ihr Status in „Clean“. Nur Steuerungen mit dem Status „Dirty“ werden durch die Polling-Methode gemeldet, Steuerungen mit dem Status „Clean“ werden nicht gemeldet. Der Status einer Steuerung wird nur von „Clean“ in „Dirty“ geändert, wenn die Eigenschaften dieser Steuerung geändert werden.

Diese Methode ermöglicht es, jede Steuerung innerhalb einer Änderungsgruppe in den Status „Dirty“ zu setzen. Sie gibt allen Steuerungen vor, ihre aktuellen Statusinformationen durch die nächste Poll [▶ 30]- oder AutoPoll [▶ 32]-Methode zu melden.

4.1.3.4.9 AutoPoll

Diese Methode ermöglicht die Codierung eines QRC-Frames, der verwendet wird, um alle Named Controls/ Named Components in den Status „Dirty“ zu setzen.

Syntax

```
METHOD Poll : I_Connect
VAR_INPUT
    sChangeGroupId        : STRING;
    fRate                 : REAL;
END_VAR
```

VAR_INPUT

sChangeGroupId: Gruppen-ID ändern.

fRate: Polling-Intervall in Sekunden. Der Mindestwert hierfür beträgt 0,1 s.

4.1.3.4.10 AddSnapshotControl

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Beifügen mehrerer Snapshots zu einer Änderungssteuerung verwendet wird.

Die Snapshot-Steuerung kann einer Änderungsgruppe nicht über Named Control beigefügt werden. In dieser Version werden die Snapshot-bezogenen Teilsteuerungen einer Änderungsgruppe über **Named Component** beigefügt.

Syntax

```
METHOD AddSnapshotControl : I_ResExtract
VAR_INPUT
    sChangeGroupId          : STRING;
    sComponentName          : STRING;
    aSnapshotNr             : ARRAY [0..23] OF USINT;
END_VAR
```

sChangeGroupId: Gruppen-Id ändern.

sComponentName: Name der Snapshot-Komponente.

aSnapshotNr: Sequenznummer des Arrays des Ziel-Snapshots.

4.1.3.5 FB_QRC_Mixer

Dieser Funktionsbaustein ermöglicht die Festlegung mehrerer verschiedener Werte auf einem benannten Mischer.

Definition von einem benannten Mischer: Eine Mischerkomponente mit einem eindeutigen Namen.

Syntax

```
FUNCTION_BLOCK FB_QRC_ChangeGroup
VAR_OUTPUT
    bError                      : BOOL;
    ipResultMessage             : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
SetCrossPointGain	Koppelpunkt-Verstärkungswert für Mischereingänge und -ausgänge festlegen.
SetCrossPointDelay	Koppelpunkt-Verzögerungswert für Mischereingänge und -ausgänge festlegen.
SetCrossPointMute	Koppelpunkt stummschalten oder Stummschaltung aufheben für Mischereingänge und -ausgänge.
SetCrossPointSolo	Koppelpunkt-Solo für Mischereingänge und -ausgänge aktivieren oder deaktivieren.
SetInputGain	Verstärkungswert für Mischereingänge festlegen.
SetInputMute	Mischereingänge stummschalten oder Stummschaltung aufheben.
SetInputSolo	Solo für Mischereingänge aktivieren oder deaktivieren.
SetOutputGain	Verstärkungswert für Mischerausgänge festlegen.
SetOutputMute	Mischerausgänge stummschalten oder Stummschaltung aufheben.
SetCueMute	Mischer-Cues stummschalten oder Stummschaltung aufheben.
SetCueGain	Verstärkungswert für Mischer-Cues festlegen.
SetInputCueEnable	Cues für Mischereingänge aktivieren oder deaktivieren.
SetInputCueAfi	Cue-AFL (After Fader Level) für Mischereingänge aktivieren oder deaktivieren.

Beipiel

Die Syntax unterstützt entweder durch Leerzeichen oder Komma getrennte Nummern, Nummernbereiche oder alle Nummern (*). Sie unterstützt die Negation der Auswahl mit dem Operator „!“.

Hier einige Beispiele:

Eingang/Ausgang	Beschreibung
*	Alles
1 2 3	Kanäle 1, 2, 3
1-6	Kanäle 1 bis 6
1-6 9	Kanal 1 bis 6 und 9
1-3 5-9	Kanal 1 bis 3 und 5 bis 9
1-8 !3	Kanal 1 bis 8 außer 3
* !3-5	Alles außer Kanäle 3 bis 5

4.1.3.5.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    iConnect                : I_Connect;
END_VAR
```

VAR_INPUT

iConnect: Der Funktionsbaustein, der die Schnittstelle I_Connect [▶ 15] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_QRC_Control`:

```
PROGRAM MAIN
VAR
    fbResExtract        : FB_QRC_ResExtract;
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbQrcControl        : FB_QRC_Control(fbConnect);
END_VAR
```

4.1.3.5.2 SetCrossPointGain

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Festlegung des Koppelpunkt-Verstärkungswerts für die Eingänge und Ausgänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetCrossPointGain : I_Connect
VAR_INPUT
    stCrossSpec          : ST_CrossSpec;
END_VAR
```

VAR_INPUT

stCrossSpec: Koppelpunkt-Verstärkungseigenschaften.

4.1.3.5.3 SetCrossPointDelay

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Festlegung des Koppelpunkt-Verzögerungswerts für die Eingänge und Ausgänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetCrossPointDelay : I_Connect
VAR_INPUT
    stCrossSpec           : ST_CrossSpec;
END_VAR
```

VAR_INPUT

stCrossSpec: Koppelpunkt-Verzögerungseigenschaften.

4.1.3.5.4 SetCrossPointMute

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Stummschaltung oder Aufhebung der Stummschaltung des Koppelpunkts für die Eingänge und Ausgänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetCrossPointMute : I_Connect
VAR_INPUT
    stCrossSpec          : ST_CrossSpec;
END_VAR
```

VAR_INPUT

stCrossSpec: Koppelpunkt-Stummschaltungseigenschaften.

4.1.3.5.5 SetCrossPointSolo

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Aktivierung oder Deaktivierung des Koppelpunkt-Solos für die Eingänge und Ausgänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetCrossPointSolo : I_Connect
VAR_INPUT
    stCrossSpec          : ST_CrossSpec;
END_VAR
```

VAR_INPUT

stCrossSpec: Koppelpunkt-Solo-Eigenschaften.

4.1.3.5.6 SetInputGain

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Festlegung des Verstärkungswerts für die Eingänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetInputGain : I_Connect
VAR_INPUT
    stInputSpec     : ST_InputSpec;
END_VAR
```

VAR_INPUT

stInputSpec: Eingangsverstärkung Eigenschaften.

4.1.3.5.7 SetInputMute

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Stummschaltung oder Aufhebung der Stummschaltung der Eingänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetInputMute : I_Connect
VAR_INPUT
    stInputSpec     : ST_InputSpec;
END_VAR
```

VAR_INPUT

stInputSpec: Eingangsstummschaltung Eigenschaften.

4.1.3.5.8 SetInputSolo

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Aktivierung oder Deaktivierung des Solos für die Eingänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetInputSolo : I_Connect
VAR_INPUT
    stInputSpec     : ST_InputSpec;
END_VAR
```

VAR_INPUT

stInputSpec: Eingangs-Solo Eigenschaften.

4.1.3.5.9 SetOutputGain

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Festlegung des Verstärkungswerts für die Ausgänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetOutputGain : I_Connect
VAR_INPUT
    stOutputSpec     : ST_OutputSpec;
END_VAR
```

VAR_INPUT

stOutputSpec: Ausgangsverstärkung Eigenschaften.

4.1.3.5.10 SetOutputMute

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Stummschaltung oder Aufhebung der Stummschaltung der Ausgänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetOutputMute : I_Connect
VAR_INPUT
    stOutputSpec     : ST_OutputSpec;
END_VAR
```

VAR_INPUT

stOutputSpec: Ausgangsstummschaltung Eigenschaften.

4.1.3.5.11 SetCueMute

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Stummschaltung oder Aufhebung der Stummschaltung für Mischer-Cues verwendet wird.

Syntax

```
METHOD SetCueMute : I_Connect
VAR_INPUT
    stCueSpec     : ST_CueSpec;
END_VAR
```

VAR_INPUT

stCueSpec: Cue-Stummschaltung Eigenschaften.

4.1.3.5.12 SetCueGain

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Festlegung des Verstärkungswerts für Mischer-Cues verwendet wird.

Syntax

```
METHOD SetCueGain : I_Connect
VAR_INPUT
    stCueSpec     : ST_CueSpec;
END_VAR
```

VAR_INPUT

stCueSpec: Cue-Verstärkung Eigenschaften.

4.1.3.5.13 SetInputCueEnable

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Aktivierung oder Deaktivierung der Cues und Eingänge eines benannten Mischers verwendet wird.

Syntax

```
METHOD SetInputCueGain : I_Connect
VAR_INPUT
    stInputCueSpec     : ST_InputCueSpec;
END_VAR
```

VAR_INPUT

stInputCueSpec: Eingangs- und Cue-Eigenschaften.

4.1.3.5.14 SetInputCueAfi

Diese Methode ermöglicht die Codierung eines QRC-Frames, der für die Aktivierung oder Deaktivierung von Cue-AFL (After Fader Level) für Mischereingänge verwendet wird.

Syntax

```
METHOD SetInputCueAfi : I_Connect
VAR_INPUT
    stInputCueSpec    : ST_InputCueSpec;
END_VAR
```

VAR_INPUT

stInputCueSpec: Cue-AFL Eigenschaften.

4.1.3.6 FB_QRC_LoopPlayer

Dieser Funktionsbaustein ermöglicht die Abfrage einer Dateiwiedergabe auf einem benannten Loop-Player.

Definition von einem benannten Loop-Player: Ein benannter Loop-Player ist eine Loop-Player-Komponente mit einem eindeutigen Namen.

Syntax

```
FUNCTION_BLOCK FB_QRC_LoopPlayer
VAR_OUTPUT
    bError                      : BOOL;
    ipResultMessage             : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
Start	Wiedergabe starten.
Stop	Wiedergabe anhalten.
Cancel	Wiedergabe abbrechen.

4.1.3.6.1 FB_init

Syntax

```
Method FB_init : BOOL
VAR_INPUT
    iConnect                : I_Connect;
END_VAR
```

VAR_INPUT

iConnect: Der Funktionsbaustein, der die Schnittstelle I_Connect [▶ 15] implementiert.

Beispiel:

Deklaration des Funktionsbausteins `FB_QRC_Control`:

```
PROGRAM MAIN
VAR
    fbResExtract        : FB_QRC_ResExtract;
    fbConnect           : FB_Connect('', '192.168.1.101', T#15S, fbResExtract);
    fbQrcControl        : FB_QRC_Control(fbConnect);
END_VAR
```

4.1.3.6.2 Start

Diese Methode ermöglicht den Start der Wiedergabe auf einem benannten Loop-Player.

Syntax

```
METHOD Start : I_Connect
VAR_INPUT
    stJobSpec           : ST_JobSpec;
END_VAR
```

VAR_INPUT

stJobSpec: Eigenschaften des Jobs, der auf einem benannten Loop-Player wiedergegeben wird.

4.1.3.6.3 Stop

Diese Methode ermöglicht das Anhalten der Wiedergabe auf einem benannten Loop-Player.

Syntax

```
METHOD Stop : I_Connect
VAR_INPUT
    sName           : STRING;
    aOutput         : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF USINT;
    bLog            : BOOL := FALSE;
END_VAR
```

VAR_INPUT

sName: Der Name des Loop-Players.

aOutput: Array von Ausgangskanälen.

bLog: Optionales Attribut für eine Ereignismeldung, standardmäßig FALSE.

4.1.3.6.4 Cancel

Diese Methode ermöglicht den Abbruch eines Jobs auf einem benannten Loop-Player.

Syntax

```
METHOD Cancel : I_Connect
VAR_INPUT
    sName           : STRING;
    aOutput         : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF USINT;
    bLog            : BOOL := FALSE;
END_VAR
```

VAR_INPUT

sName: Der Name des Loop-Players.

aOutput: Array von Ausgangskanälen.

bLog: Optionales Attribut für eine Ereignismeldung, standardmäßig FALSE.

4.1.3.7 FB_QRC_Snapshot

Dieser Funktionsbaustein ermöglicht die Codierung von QRC-Frames, die zum Laden/Speichern von Snapshots verwendet werden. Zudem kann er zum Abfragen mehrerer Snapshot-Status verwendet werden.

Dieser Funktionsbaustein ist ab Version 1.1.0.0 verfügbar.
Dieser Befehl ist in der QRC-Spezifikation nicht aufgeführt.

- Lesen Sie vor Verwendung dieses Funktionsbausteins oder zugehöriger Methoden zuerst den Abschnitt Snapshot-Status und zugehörige Eigenschaften [▶ 19].

Syntax

```
FUNCTION_BLOCK FB_QRC_Snapshot
VAR_OUTPUT
    bError                      : BOOL;
    ipResultMessage             : I_TcMessage;
END_VAR
```

Ausgänge

Name	Typ	Beschreibung
bError	BOOL	Wird gesetzt, wenn ein Fehler aufgetreten ist. Fehlerdetails befinden sich im Fenster **Error List**.
ipResultMessage	I_TcMessage	Ermöglicht die Fehlerbehandlung mit dem Tc3_EventLogger.

Methoden

Name	Beschreibung
FB_init	Initialisierungsmethode
Load	Einen Snapshot auslösen.
Save	Einen Snapshot speichern. (Alten Snapshot überschreiben.)
GetSnapshotState	Mehrere Snapshot-Status in einer Snapshot Bank abrufen.

4.1.3.7.1 Load

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Auslösen eines Snapshots verwendet wird.

Syntax

```
METHOD Load : I_Connect
VAR_INPUT
    sBankName       : STRING;
    nSnapshotNr     : USINT;
    fRamp           : REAL := 0; (*Optional*)
END_VAR
```

sBankName: Name der Snapshot Bank.

nSnapshotNr: Sequenznummer des Ziel-Snapshots.

fRamp: Optionale Rampenzeit.

> Der **Name einer Snapshot Bank** ist vollkommen anders als der **Name der Snapshot-Komponente**. Wie man den Namen einer Snapshot Bank findet, wird im Abschnitt Einfache Möglichkeit, Steuerungsnamen, Komponentennamen und Snapshot Bank-Namen zu finden [▶ 51], beschrieben.

4.1.3.7.2 Save

Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Speichern eines Snapshots verwendet wird.

Syntax

```
METHOD Save : I_Connect
VAR_INPUT
    sBankName       : STRING;
    nSnapshotNr     : USINT;
    fRamp           : REAL := 0; (*Optional*)
END_VAR
```

sBankName: Name der Snapshot Bank.

nSnapshotNr: Sequenznummer des Ziel-Snapshots.

fRamp: Optionale Rampenzeit.

Der **Name einer Snapshot Bank** ist vollkommen anders als der **Name der Snapshot-Komponente**. Wie man den Namen einer Snapshot Bank findet, wird im Abschnitt Einfache Möglichkeit, Steuerungsnamen, Komponentennamen und Snapshot Bank-Namen zu finden [▶ 51], beschrieben.

4.1.3.7.3 GetSnapshotState

Jeder Snapshot-Status kann von zwei zugehörigen Steuerungen in der Snapshot-Komponente bestimmt werden. Diese Methode ermöglicht die Codierung eines QRC-Frames, der zum Abfragen mehrerer Snapshot-Status verwendet wird.

Syntax

```
METHOD GetSnapshotState : I_Connect
VAR_INPUT
    sComponentName      : STRING;
    aSnapshotNr         : ARRAY[0..23] OF USINT;
END_VAR
```

sComponentName: Name der Snapshot-Komponente.

aSnapshotNr: Sequenznummer des Arrays des angeforderten Snapshots.

1. Der **Name der Snapshot-Komponente** ist vollkommen anders als der **Name einer Snapshot Bank**. Wie man den Namen der Snapshot-Komponente findet, wird im Abschnitt Einfache Möglichkeit, Steuerungsnamen, Komponentennamen und Snapshot Bank-Namen zu finden [▶ 51], beschrieben.
2. Setzen Sie das Attribut `bSavOldRes` des Funktionsbausteins `FB_QRC_RecExtract` auf TRUE, um die vergangenen Snapshot-Status bei der Eigenschaft `aCtrlProp` zu speichern.

4.2 Strukturen, Aufzählungen, GVL

4.2.1 E_FileMode

```
Type E_FileMode
{
    mono,
    stereo
} USINT;
END_TYPE
```

4.2.2 ST_Control

```
TYPE ST_CONTROL
STRUCT
    sName         : STRING := '';  (*Name of Named Control*)
    sValue        : STRING := '';  (*Value of Named Control*)
    sString       : STRING := '';  (*String of Named Control*)
    fRamp         : REAL   := 0;   (*Optional ramp time of Named Control*)
END_STRUCT
END_TYPE
```

4.2.3 ST_ControlEx

Diese Struktur erweitert St_Control [▶ 41] und ist nur für die Eigenschaft `arrCtrlProp` des Funktionsbausteins FB_QRC_ResExtract [▶ 17] ausgelegt.

```
TYPE ST_ControlEx EXTENDS ST_Control
STRUCT
    sComponent       : STRING := '';  (*Component name*)
    fPosition        : STRING := '';  (*Control position*)
END_STRUCT
END_TYPE
```

4.2.4 Struktur über Mischer

ST_CrossSpec

```
TYPE ST_CrossSpec:
STRUCT
    sName           : STRING := '';  (*Name of named mixer*)
    sInputs         : STRING := '';  (*Input channel of named mixer*)
    sOutputs        : STRING := '';  (*Output channel of named mixer*)
    sValue          : STRING := '';  (*value of named mixer*)
    fRamp           : REAL   := 0;   (*Optional ramp time of named mixer*)
END_STRUCT
END_TYPE
```

ST_InputSpec

```
TYPE ST_InputSpec:
STRUCT
    sName           : STRING := '';  (*Name of named mixer*)
    sInputs         : STRING := '';  (*Input channel of named mixer*)
    sValue          : STRING := '';  (*value of named mixer*)
    fRamp           : REAL   := 0;   (*Optional ramp time of named mixer*)
END_STRUCT
END_TYPE
```

ST_OutputSpec

```
TYPE ST_OutputSpec:
STRUCT
    sName           : STRING := '';  (*Name of named mixer*)
    sOutputs        : STRING := '';  (*Output channel of named mixer*)
    sValue          : STRING := '';  (*value of named mixer*)
    fRamp           : REAL   := 0;   (*Optional ramp time of named mixer*)
END_STRUCT
END_TYPE
```

ST_CueSpec

```
TYPE ST_CueSpec:
STRUCT
    sName           : STRING := '';  (*Name of named mixer*)
    sCues           : STRING := '';  (*Cue of named mixer*)
    sValue          : STRING := '';  (*value of named mixer*)
    fRamp           : REAL   := 0;   (*Optional ramp time of named mixer*)
END_STRUCT
END_TYPE
```

ST_InputCueSpec

```
TYPE ST_InputCueSpec EXTENDS ST_CueSpec:
STRUCT
    sInputs         : STRING  := '';(*Input channel of named mixer*)
END_STRUCT
END_TYPE
```

4.2.5 ST_FileSpec

```
TYPE ST_FileSpec:
STRUCT
    sFileName    : T_MaxString;
    eMode        : E_FileMode;
    nOutput      : USINT; (*Output Channel*)
END_STRUCT
END_TYPE
```

4.2.6 ST_JobSpec

```
TYPE ST_JobSpec:
STRUCT
    sName           : STRING :='';
    nStartTime         : UDINT  := 0;
    aFiles             : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_FileSpec;
```

```
    bLoop               : BOOL;
    nSeek               : UDINT := 0;
    bLog                : BOOL;
END_STRUCT
END_TYPE
```

4.2.7 Param

Name	Defaultwert	Beschreibung
QRC_RECEIVE_POLLING_TIME	100 ms	Polling-Zeit für die TCP-Verbindung
QRC_RECEIVE_TIMEOUT	10 s	Zeit für Empfänger-Zeitüberschreitung
QRC_BUFFER_SIZE	2500	QRC-Frame-Puffergröße in Byte
QRC_NUMBER_OF_CONTROL	50	Die maximale Anzahl von Steuerungen, die senden, empfangen und auslesen dürfen.

1. QRC_BUFFER_SIZE Die definierte Länge des Sendepuffers `sTxFrame`. Vor der Übertragung jedes QRC-Frames wurde dieser QRC-Frame gemessen und geprüft, ob die Länge dieses Frames größer als QRC_BUFFER_SIZE ist. In einigen Fällen (z. B. wenn Hunderte von Steuerungen über Control.Set [▶ 26] oder Component.Set [▶ 27] übertragen werden) ist der Puffer schnell überlastet. Bei einem Überlauf des Puffers tritt vor der Übertragung dieses QRC-Frames ein Fehler „Buffer overflowed“ auf. Dieser QRC-Frame wird ignoriert, bis der Wert von QRC_BUFFER_SIZE erhöht wurde.

2. QRC_BUFFER_SIZE ist von hoher Relevanz für QRC_NUMBER_OF_CONTROL. Folglich muss der Wert von QRC_BUFFER_SIZE entsprechend geändert werden. (Das Verhältnis 1:50 (1 Steuerung - 50 Byte) wird empfohlen).

4.3 Schnittstellen

4.3.1 I_Connect

METHODEN

Connect: Eine TCP-Verbindung herstellen.

Disconnect: Eine TCP-Verbindung beenden.

Send: QRC-Frames senden.

M_Receive: QRC-Frames empfangen.

EIGENSCHAFTEN

Eigenschaften	Typ	Zugriff	Beschreibung
aRxFrame	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString	Get	Sobald die fallende Flanke von `bBusy` auftritt und `bError` gleich FALSE ist, kann der empfangene QRC-Antwort-Frame mit dieser Eigenschaft abgefragt werden.
sTxFrame	STRING(QRC_BUFFER_SIZE)	Set	Sobald die fallende Flanke von `bBusy` auftritt und `bError` gleich FALSE ist, kann der zu sendende QRC-Frame gesetzt werden.

Connect

Diese Methode ermöglicht die Herstellung einer TCP-Verbindung.

```
Method Connect            : BOOL
```

Dieser Prozess ist abgeschlossen, sobald der Rückgabewert TRUE ist.

Disonnect

Diese Methode ermöglicht die Beendigung einer TCP-Verbindung.

```
Method Disonnect          : BOOL
```

Dieser Prozess ist abgeschlossen, sobald der Rückgabewert TRUE ist.

Send

Diese Methode ermöglicht das Senden eines QRC-Frames und das automatische Abrufen des Antwort-Frames vom Q-SYS-Gerät nach dem Senden.

```
Method Send               : I_ResExtract
```

Diese Methode ist abgeschlossen, sobald die fallende Flanke von `bBusy` auftritt und die Eigenschaft `aRxFrame` nicht leer ist. Der Antwort-Frame kann bei der Eigenschaft `aRxFrame` abgerufen werden.

Receive

Diese Methode ermöglicht den Empfang eines QRC-Frames.

```
Method Receive            : I_ResExtract
```

Diese Methode ist abgeschlossen, sobald der fallende Trigger von `bBusy` ausgelöst wird und die Eigenschaft `aRxFrame` nicht leer ist. Der Antwort-Frame kann bei der Eigenschaft `aRxFrame` abgerufen werden.

aRxFrame

Liste der empfangenen QRC-Antwort-Frames.

```
PROPERTY aRxFrame       : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString
```

sTxFrame

Ein QRC-Frame, der bereit zum Senden an das Q-SYS-Gerät ist.

```
PROPERTY sTxFrame         : STRING(QRC_BUFFER_SIZE)
```

4.3.2 I_ResExtract

METHODEN

ResExtract: Empfangene QRC-Antwort-Frames aus dem Q-SYS-Gerät auslesen.

EIGENSCHAFTEN

Eigenschaften	Typ	Zugriff	Beschreibung
aCtrlProp	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_ControlEx	Get	Mit dieser Eigenschaft werden die ausgelesenen Steuerungseigenschaften abgerufen.
aRxFrame	ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString	Set, Get	Mit dieser Eigenschaft können auszulesende QRC-Frames gesetzt oder abgerufen werden.

aCtrlProp

Liste der Steuerungseigenschaften, die durch ResExtract ausgelesen wurden.

```
PROPERTY aCtrlProp        : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF ST_ControlEx
```

arrRxFrame

Mit dieser Eigenschaft kann ein QRC-Antwort-Frame, der bereit zum Auslesen ist, gesetzt oder abgerufen werden.

Wie bereits erwähnt [▶ 17], kann dieser Funktionsbaustein begrenzte Typen von QRC-Antwort-Frames auslesen. Ein Antwort-Frame, der nicht durch den Funktionsbaustein ausgelesen werden kann, kann vor der Auslesung mit der „Getter“-Funktion abgerufen werden. Des Weiteren können Benutzer auch ihren eigenen QRC-Frame bei der „Setter“-Funktion schreiben, um Informationen aus ihrem eigenen QRC-Frame auszulesen.

```
PROPERTY aRxFrame         : ARRAY[0..QRC_NUMBER_OF_CONTROL] OF T_MaxString
```

4.3.2.1 ResExtract

ResExtract

Diese Methode ermöglicht die Auslesung von Steuerungseigenschaften aus einem QRC-Antwort-Frame.

Syntax

```
METHOD ResExtract : BOOL
VAR_INPUT
    bSavOldRes      : BOOL;
END_VAR
```

VAR_INPUT

bSavOldRes: Diese Variable bestimmt, ob der referenzierte Funktionsbaustein vergangene Steuerungseigenschaften sichert, die aus früheren QRC-Frames ausgelesen wurden. Weitere Informationen befinden sich im Abschnitt „Das Attribut bSavOldRes [▶ 45]“.

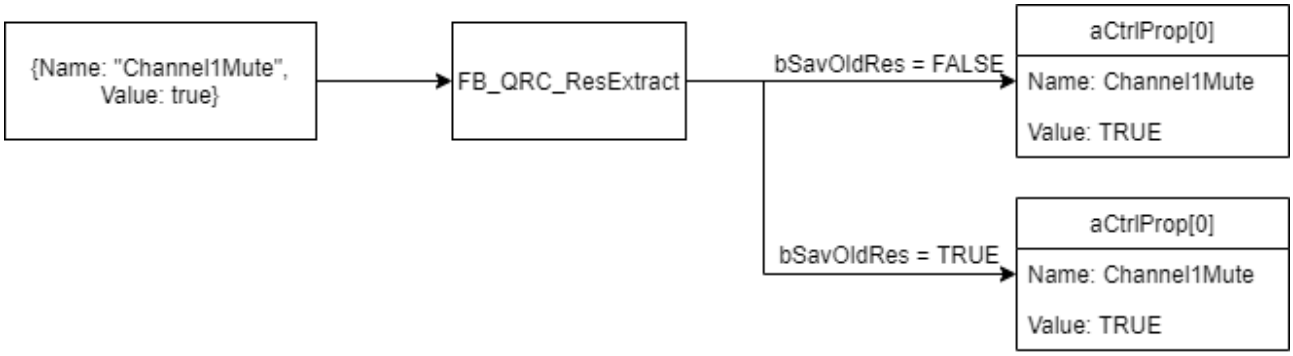
4.3.2.1.1 Das Attribut bSavOldRes

Die Eingangsvariable `bSavOldRes` der Methode `ResExtract` wurde implementiert, um eine Konfiguration empfangener Steuerungsinformationen zu ermöglichen. Das Array `aCtrlProp` kann die Anzahl QRC_NUMBER_OF_CONTROL der Steuerungsinformationen speichern. Dieses Attribut kann in der Parameterliste [▶ 43] geändert werden.

- Setzt man das Attribut `bSavOldRes` auf TRUE, werden alle vergangenen Steuerungsinformationen gespeichert. Wenn Steuerungsinformationen anstehen, die bereits gespeichert sind, werden die alten Steuerungsinformationen durch die neuen überschrieben.
- Setzt man das Attribut `bSavOldRes` auf FALSE, werden alle vergangenen Steuerungsinformationen, die im Array `aCtrlProp` gespeichert waren, gelöscht. Nur die neuesten Steuerungsinformationen werden gespeichert.

Für ein besseres Verständnis des Verhaltens wird nachstehend ein Beispiel gezeigt.

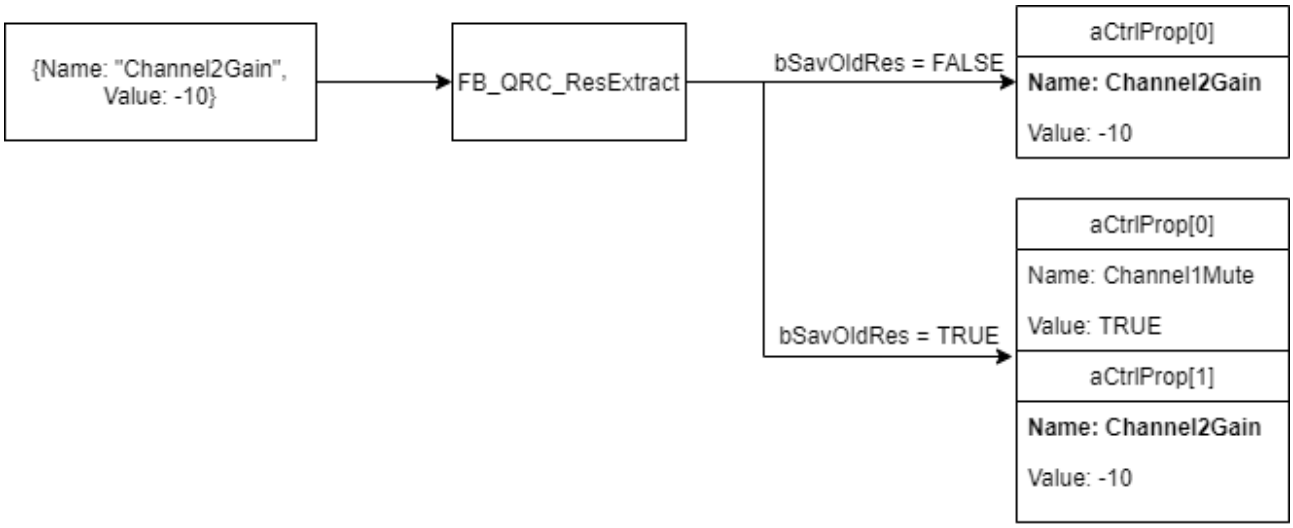
1. Schritt: Steuerungsinformationen von „**Channel1Mute**“ empfangen.



In Schritt 1 wurde ein QRC-Frame bei `aRxFrame` empfangen und die Steuerungsinformationen werden durch FB_QRC_RecExtract [▶ 17] ausgelesen.

Das Array `aCtrlProp` ist leer. Deshalb wird die Steuerung **Channel1Mute** beim Element `aCtrlProp[0]` gespeichert, unabhängig davon, ob `bSavOldRes` gleich TRUE ist oder nicht.

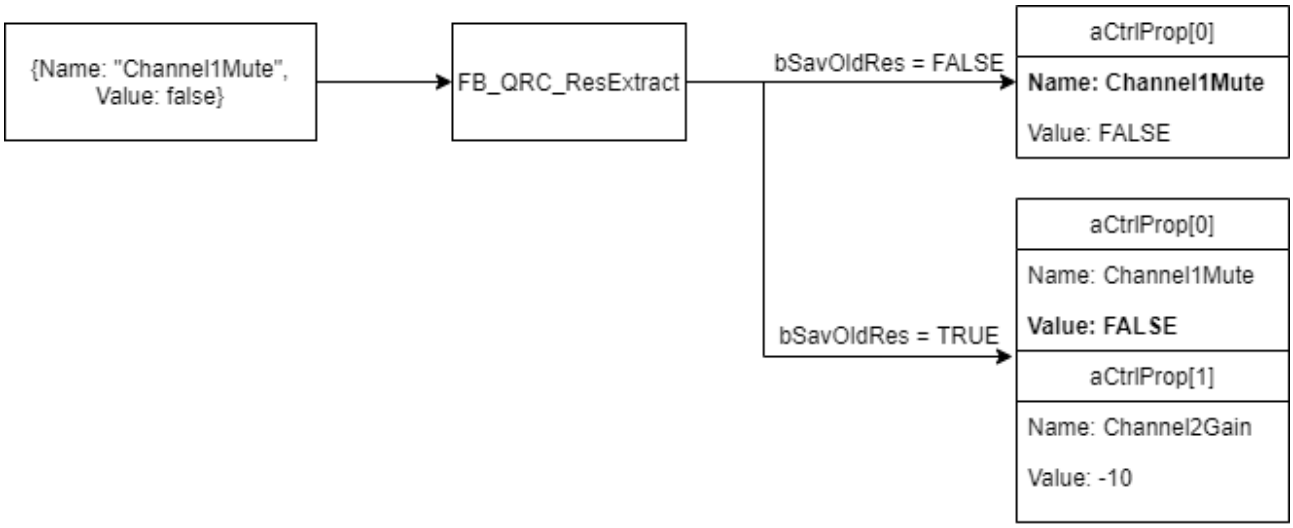
2. Schritt: Steuerungsinformationen von „**Channel2Gain**“ (andere Steuerung) empfangen.



In Schritt 2 wurde ein zweiter QRC-Frame empfangen. Nach der Auslesung der neuen Steuerungsinformationen werden sie abhängig vom Wert von `bSavOldRes` gespeichert.

- Wenn `bSavOldRes` gleich TRUE ist, wird die Steuerung **Channel2Gain** beim Element `aRecProp[1]` gespeichert, weil `aCtrlProp[0]` die andere Steuerung Channel1Mute gespeichert hat.
- Wenn `bSavOldRes` gleich FALSE ist, wird die Steuerung **Channel2Gain** beim Element `aCtrlProp[0]` gespeichert. Die Steuerungsinformationen von Channel1Mute, die beim selben Element gespeichert waren, werden überschrieben.

3. Schritt: Steuerungsinformationen von „**Channel1Mute**“ (Aktualisierung der bereits empfangenen Steuerung) empfangen.



In Schritt 3 wurde ein dritter QRC-Frame empfangen. Nach der Auslesung wurde erkannt, dass der Steuerungsname bereits bei `aCtrlProp[0]` gespeichert war:

- Wenn `bSavOldRes` gleich TRUE ist, werden die neu eingehenden Informationen von **Channel1Mute** beim Element `aRecProp[0]` gespeichert. Dadurch werden die gespeicherten Steuerungsinformationen von **Channel1Mute** aktualisiert und die bei `aCtrlProp[1]` gespeicherten Steuerungsinformationen beibehalten.

- Wenn `bSavOldRes` gleich FALSE ist, werden die neu eingehenden Steuerungsinformationen von **Channel1Mute** beim Element `aCtrlProp[0]` gespeichert. Andere gespeicherte Informationen werden gelöscht.

Alle vergangenen Steuerungseigenschaften werden nur gespeichert, wenn `bSavOldRes` gleich TRUE ist. Falls `bSavOldRes` gleich FALSE ist, werden alle vergangenen Steuerungsinformationen gelöscht.

4.3.2.1.2 Arbeitsablauf für das Auslesen von Snapshot-Eigenschaften

Es gibt zwei Möglichkeiten, um einen Snapshot-Status abzufragen – das manuelle Abfragen mit der Methode GetSnapshotState [▶ 41] oder das Beifügen zu einer Änderungsgruppe und Pollen ihrer Änderungen. Basierend auf dem Funktionsprinzip einer Änderungsgruppe meldet die Polling-Funktion in einem Polling-Zyklus nur an die geänderte Steuerung. In einigen Fällen ist es unmöglich, einen Snapshot-Status zu bestimmen. (Ändert sich z. B. ein Snapshot von „geladen“ in „geändert“, dann meldet das Q-SYS-Gerät nur, dass sich die Steuerung „`match`“ von „true“ in „false“ geändert hat. Die andere zugehörige Steuerung „`last`“ bleibt „true“.) Bei jeder Verwendung der Methode GetSnapshotState [▶ 41] werden jedoch alle zugehörigen Steuerungen eines angeforderten Snapshots abgefragt. Mit den vollständigen Informationen kann der Snapshot-Status immer bestimmt werden.

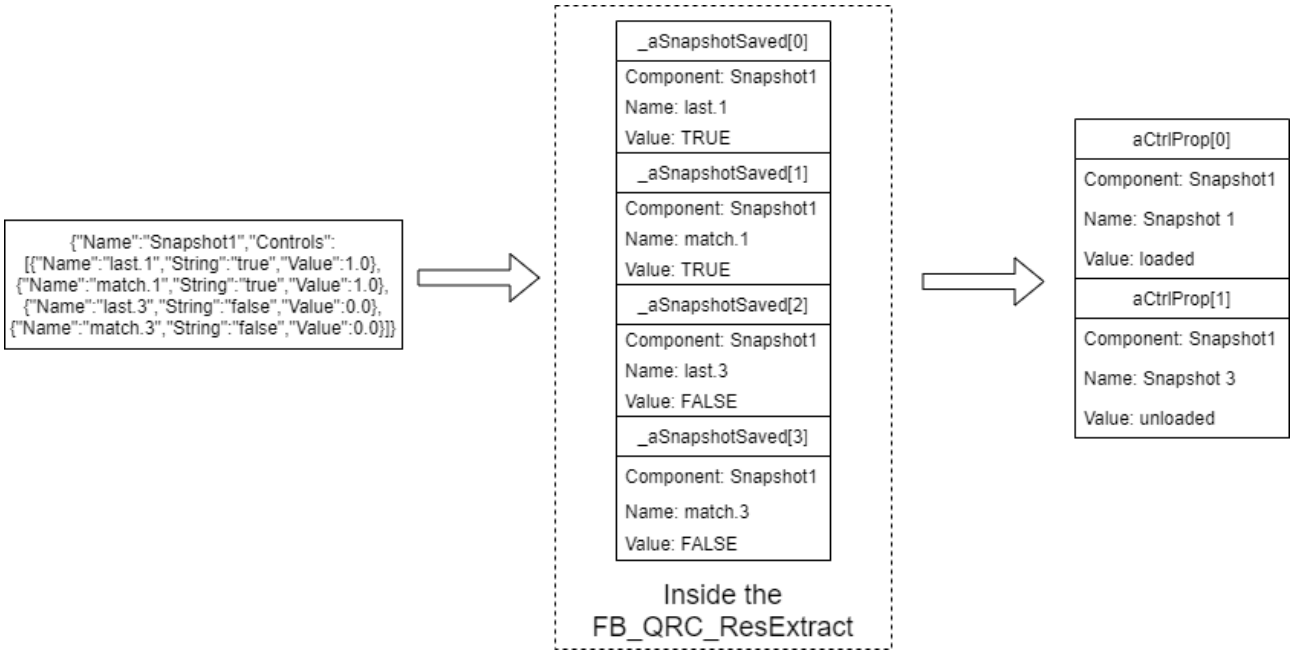
Da jede Snapshot-Eigenschaft, die von einer Polling-Funktion (Poll [▶ 30] oder AutoPoll [▶ 32]) abgefragt wird, intern gespeichert wird, kann die Methode Clear [▶ 31] des Funktionsbausteins FB_QRC_ResExtract [▶ 17] verwendet werden, um diesen Speicher freizugeben.

Nach dem Eintreffen eines Antwort-Frames von einem Q-SYS-Gerät werden alle Snapshot-Steuerungseigenschaften, die durch die Polling-Methode abgerufen werden, intern gespeichert. (Das Attribut `bSavOldRes` hat darauf KEINE Auswirkungen.) Die Snapshot-Steuerungseigenschaften werden aktualisiert. Mit Hilfe der Methode Clear können diese Eigenschaften gelöscht werden.

Diese Logik hat keine Auswirkungen auf die Logik `bSavOldRes`, die im Abschnitt über das Attribut `bSavOldRes` beschrieben wird. Die Benutzer können jedoch auch `bSavOldRes` auf **TRUE** setzen, um die Steuerungseigenschaften beim Element `aCtrlProp` zu speichern.

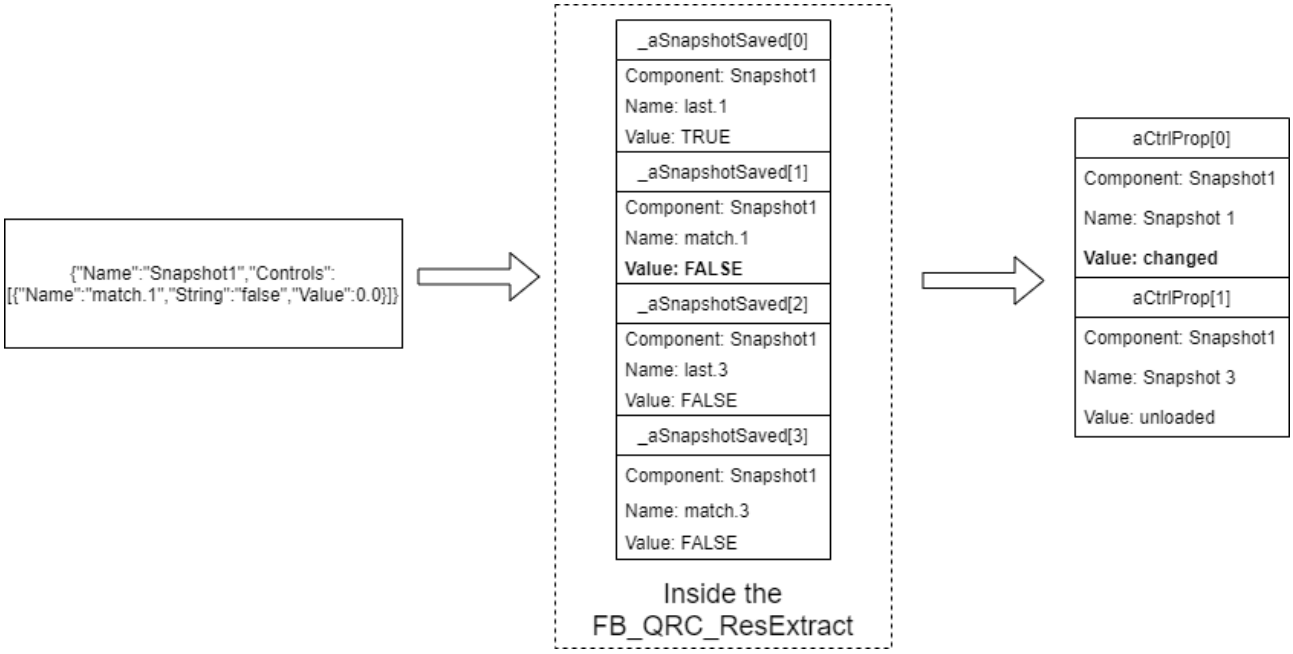
Für ein besseres Verständnis dieses Verhaltens wird nachstehend ein Beispiel gezeigt.

Schritt 1: Nachdem Snapshot 1 und 3 (der Name der Snapshot Bank lautet „Bank1“, der Name der Snapshot-Komponente lautet „Snapshot1“) der Änderungsgruppe („ChangeGroup 1“) beigefügt wurden, wurde der Antwort-Frame empfangen:



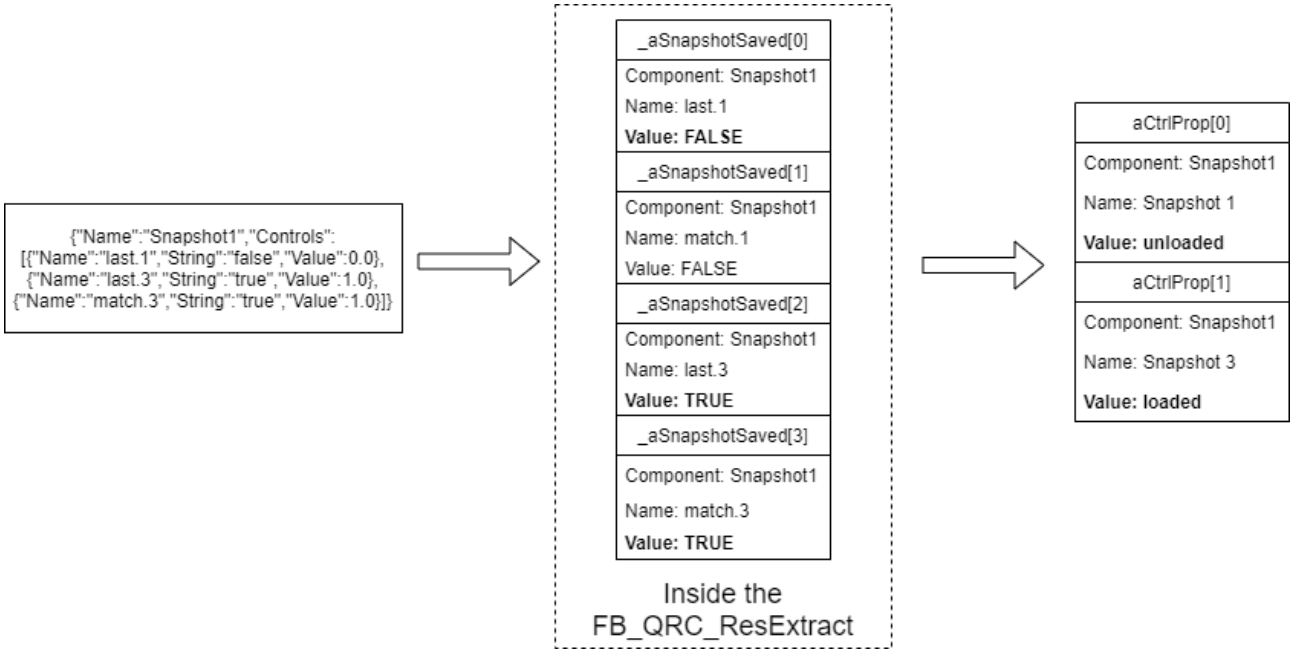
Alle zugehörigen Informationen werden intern in einem Array gespeichert. Die Status der Snapshots werden bestimmt.

Schritt 2: Falls in einem Snapshot enthaltene Steuerungen in einem Polling-Zyklus geändert wurden, trifft ein Polling-Frame ein:



Die Eigenschaft „`match.1`“ wird im internen Array aktualisiert und der Snapshot „Snapshot 1“ ändert seinen Status von „geladen“ in „geändert“. (`aCtrlProp[0]`)

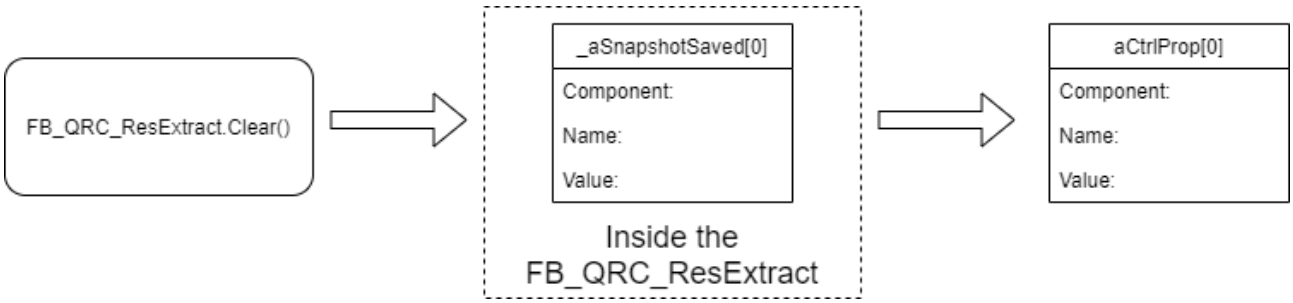
Schritt 3: Snapshot 3 wird ausgelöst.



Der „Snapshot 3“ wurde soeben ausgelöst und der Polling-Frame wurde empfangen. Die zugehörigen Steuerungseigenschaften werden aktualisiert.

Schritt 4: Löschen des internen Arrays.

Wenn Benutzer den Status eines anderen Snapshots pollen möchten und die gespeicherten Eigenschaften nicht mehr nützlich sind, sollte das interne Array mit Hilfe der Methode Clear [▶ 31] zurückgesetzt werden.

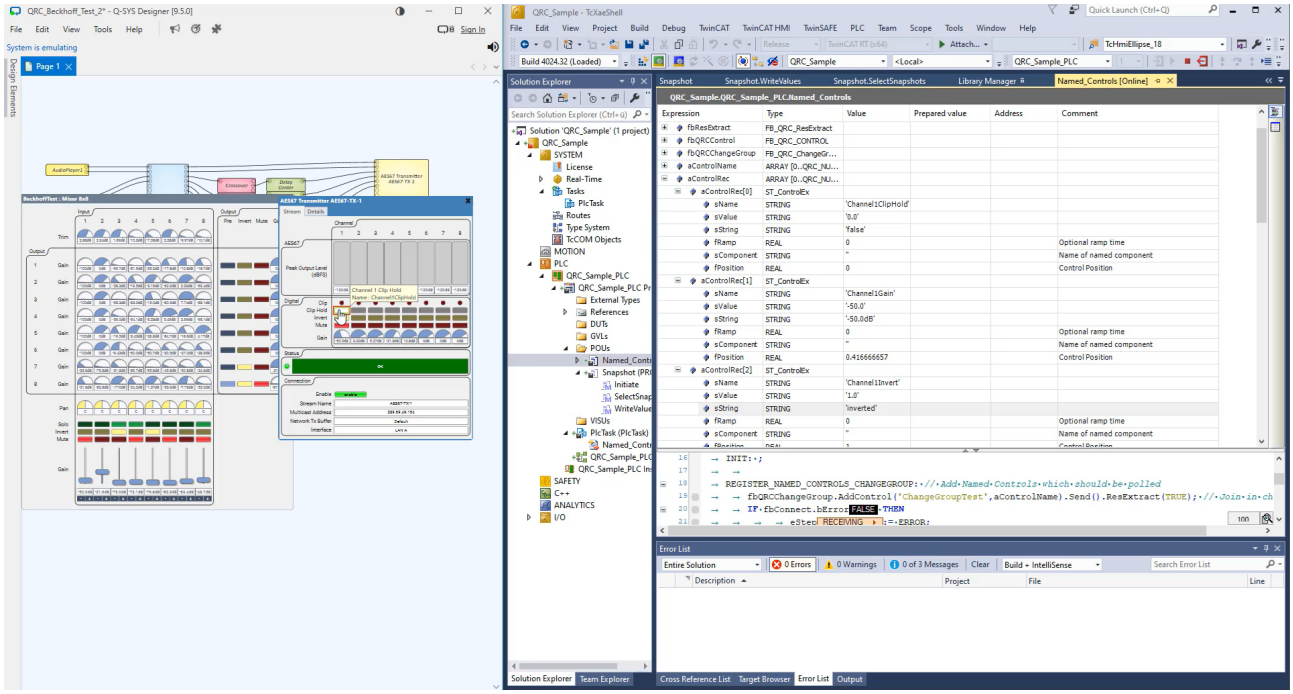


Nach dem Löschvorgang sind das interne Array und das Array `aCtrlProp` beide leer.

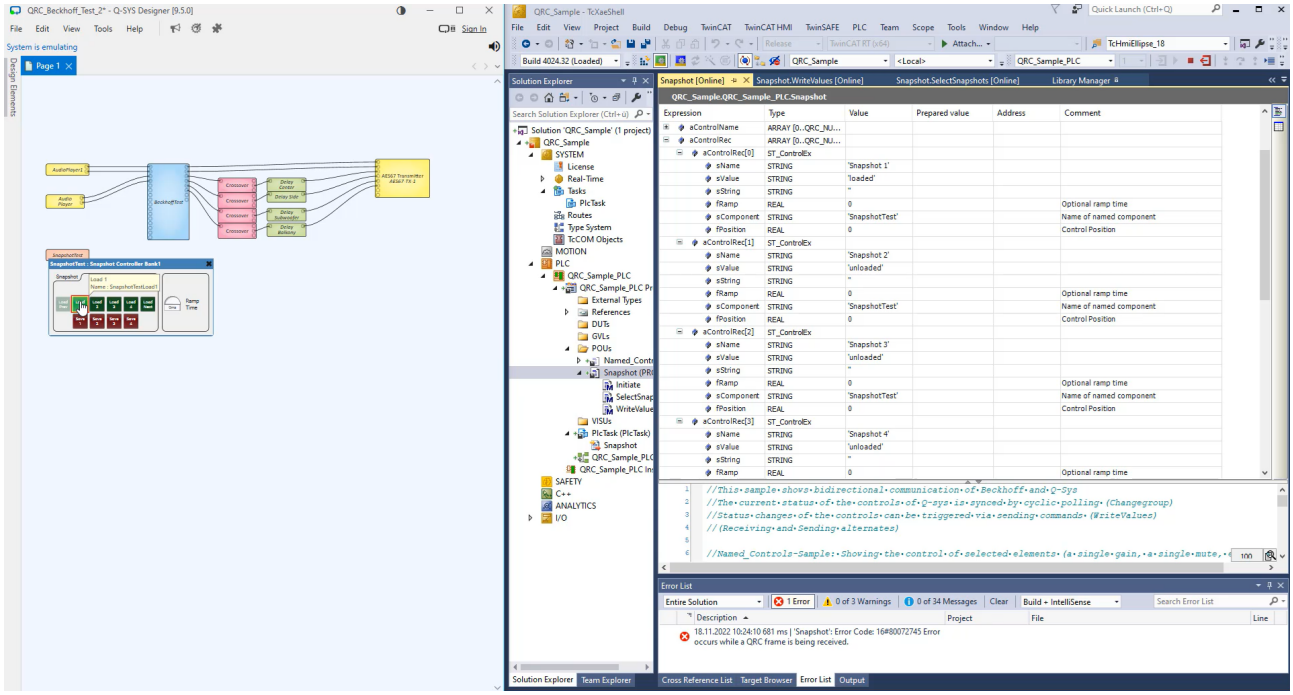
5 Beispiel: AutoPolling und Schreiben von Controls

Das https://infosys.beckhoff.com/content/1031/TF6310_QRC/Resources/13204173963.zip zeigt exemplarisch die Funktionalität der QRC-Integration in TwinCAT. Es besteht aus zwei TwinCAT-Programmen und einer Q-Sys-Designdatei. Zusätzlich sind zwei Videos enthalten, die die Basisfunktionalität der beiden Programme zeigen.

Im Programm „Named_Controls“ können Named Controls aus dem Q-Sys-Designer selektiert werden. Der Status der Controls wird anschließend in einem beliebigen Intervall abgefragt (AutoPoll) und es können Werte für beliebige Controls gesetzt werden.



Im Programm „Snapshot“ können Sie vordefinierte Snapshots aus dem Q-Sys-Designer selektieren. Der Status der Snapshots wird anschließend in einem beliebigen Intervall abgefragt (AutoPoll) und die einzelnen Snapshots, in diesem Beispiel Snapshot 1-3, können getriggert werden.



6 Anhang

6.1 Fehlercodes

Die folgenden Fehlercodes können zurückgegeben werden. Diese Fehlercodes werden von QSC definiert.

Code (dez)	Beschreibung
-32700	Parse-Fehler. Ungültige JSON wurde vom Server empfangen.
-32600	Ungültige Anforderung. Die gesendete JSON ist kein gültiges Anforderungsobjekt.
-32601	Methode nicht gefunden.
-32602	Ungültige Parameter.
-32603	Serverfehler
2	Ungültige Seitenanforderungs-ID.
3	Ungültige Seitenanforderung / die Anforderung Seitenanforderung konnte nicht erstellt werden.
4	Fehlende Datei
5	Änderungsgruppen ausgeschöpft
6	Unbekannte Änderungsgruppe
7	Unbekannter Komponentenname
8	Unbekannte Steuerung
9	Illegaler Mischerkanalindex
10	Anmeldung erforderlich

Zugehörige Informationen befinden sich auch im Fenster „Error List“.

6.2 Puffergröße

Während des Sendeprozesses oder des Empfangsprozesses wird ein langer TCP-Frame (Länge > `QRC_BUFFER_SIZE`) in mehrere Segmente aufgeteilt. Nach jedem Empfangsprozess prüft der Funktionsbaustein FB_Connect [▶ 13] den empfangenen Frame darauf, ob es sich um einen unabhängigen QRC-Frame oder um ein Segment eines langen QRC-Frames handelt. Der Funktionsbaustein empfängt so lange weiter, bis alle Segmente angekommen sind oder eine Empfangszeitüberschreitung auftritt.

Die Größe des Empfangspuffers beträgt 255 Byte*`QRC_NUMBER_OF_CONTROL`. Wenn der Puffer überläuft, tritt ein Fehler auf und weitere Fehlerdetails befinden sich im Fenster „Error List“.

6.3 String-Funktion

Die STRING-Funktionen (LEN, MID, LEFT usw.) sind nur gültig für den normalen String-Typ (String-Länge <= 255). Für den langen String-Typ (Länge > 255, in diesem Projekt ist `sTxFrame` ein langer String) können stattdessen Speicherfunktionen (MEMSET, MEMCPY, MEMMOVE) verwendet werden.

6.4 Einfache Möglichkeit, Steuerungsnamen, Komponentennamen und Snapshot Bank-Namen zu finden

Aktualisierung:

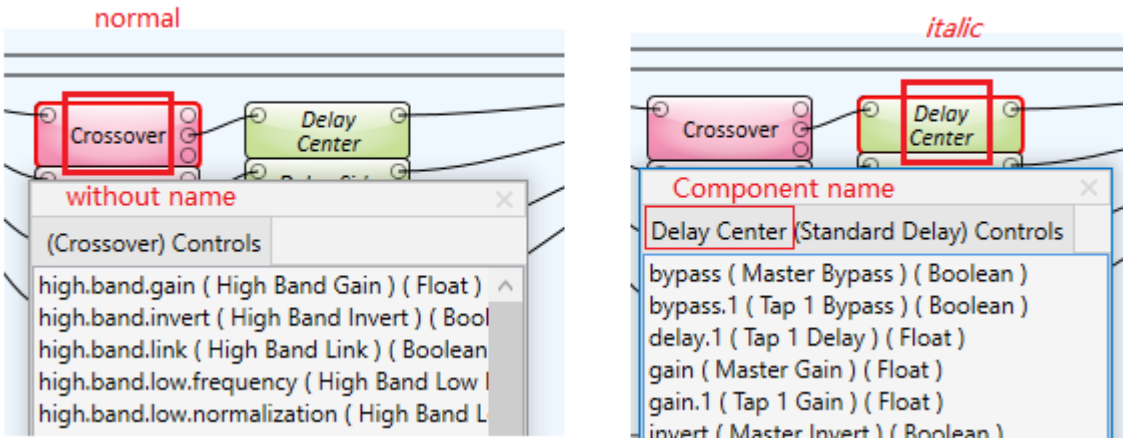
Ab Version 1.1.0.0 wird bei Verwendung des Funktionsbausteins `FB_QRC_Snapshot` der Name einer Snapshot Bank benötigt. Der Name einer Snapshot Bank ist im Bereich Snapshot oder im Fenster Snapshot Eigenschaften zu finden und kann dort konfiguriert werden.

Im Vergleich zum TwinCAT-Programm ist es auch möglich, Steuerungen und Komponenten ohne Benennung in Q-SYS Designer zu instanziieren. Die Steuerungsnamen und Komponentennamen sind jedoch der Schlüssel zur Fernsteuerung. Ohne einen eindeutigen Namen kann nicht auf eine Steuerung oder Komponente zugegriffen werden bzw. können diese nicht angesteuert werden. Vor dem Betrieb muss unbedingt geprüft werden, ob jede Steuerung oder Komponente einen gültigen Namen hat oder nicht.

Daher gilt:

- Bei **Steuerungen** prüft man am besten, ob ihr Name bereits im Bereich „Named Controls“ hinterlegt wurde oder nicht.
- Bei **Komponenten** ist es am einfachsten, die Schriftart dieser Komponente zu prüfen. Ist ihre Schriftart normal, bedeutet dies, dass die Komponente noch nicht benannt worden ist. Der Text einer benannten Komponente wird kursiv dargestellt.

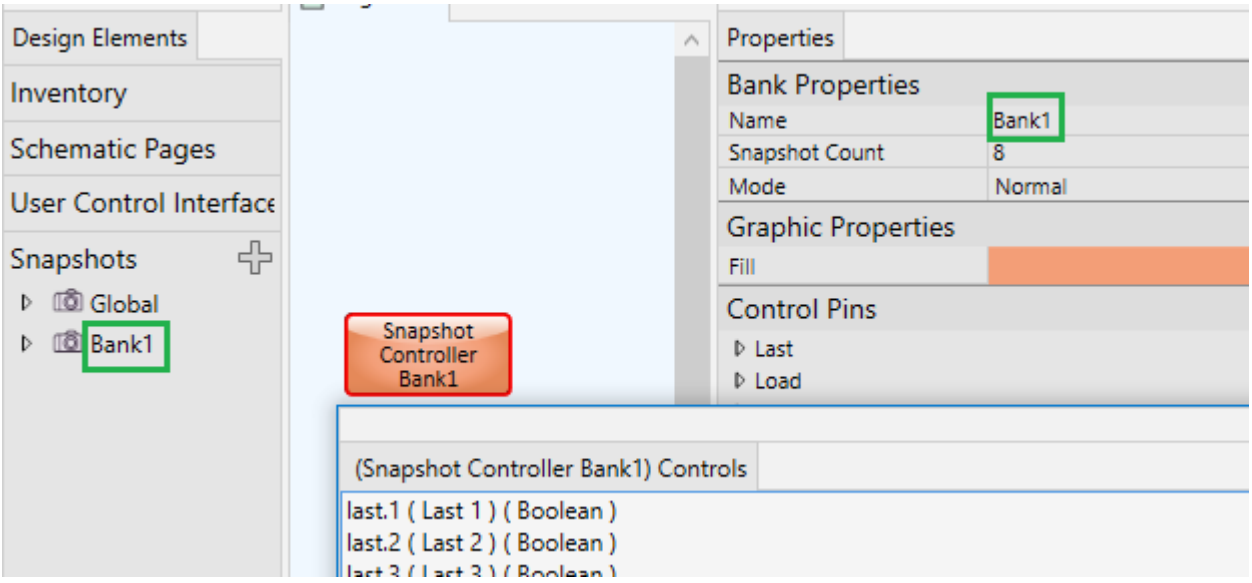
Hier ein Beispiel für eine Komponente:



(normal = nicht benannt, kursiv = benannt)

Auf der linken Seite ist die Schriftart des Textes „Crossover“ normal. Dies bedeutet, dass „Crossover“ der Typ der Komponente ist und diese nicht benannt wurde. Auf der linken Seite ist die Schriftart des Textes „*Delay Center*“ kursiv. Der Typ dieser Steuerung ist „Standard Delay“ und ihr Name lautet „Delay Center“. Sie hat einen gültigen Namen.

Hier ein Beispiel für eine Snapshot Bank:



(Grünes Rechteck = Name einer Snapshot Bank)

6.5 Steuerschaltfläche „Load“ der Snapshot-Komponente

Aktualisierung:

Ab Version 1.1.0.0 können die Methoden Load [▶ 40] und Save [▶ 40] des Funktionsbausteins FB_QRC_Snapshot [▶ 39] verwendet werden, um Snapshots ohne Verwendung des Named Control-Konzepts auszulösen/zu speichern. Die Methode GetSnapshotState [▶ 41] kann verwendet werden, um den Status eines Snapshots manuell abzufragen, oder die Methode AddSnapshotControl [▶ 32] des Funktionsbausteins FB_QRC_ChangeGroup [▶ 28] kann verwendet werden, um ihn einer Änderungsgruppe beizufügen und dann deren Änderungen zu pollen. Der Funktionsbaustein FB_QRC_ResExtract [▶ 17] wird nun dahingehend unterstützt, den Antwort-Frame des Snapshots auszulesen. Lesen Sie den Abschnitt Snapshot-Status und zugehörige Eigenschaften [▶ 53] für weitere Informationen.

Vorwort

Es gibt keine zugehörigen Informationen über die Steuerung von Snapshot-Komponenten in der QRC-Spezifikation. Die Schaltfläche „Load“ (eine Steuerung der Snapshot-Komponente) ist ein „Trigger“-Typ. Es besteht daher keine Möglichkeit, einen Status zurückzuerlangen oder eine Änderungsgruppe hinzuzufügen. Die folgende Lösung ist eine praktische Umgehung, um die Rückmeldung über die Eigenschaft „Farbe“ der Snapshot-Schaltflächen zu erhalten. Auf andere Weise können wir nicht sicherstellen, ob der Prozess „Load“ erfolgreich ausgeführt wurde.

Anhand der Eigenschaft „Farbe“ können Statusänderungen von Schaltflächen innerhalb einer Snapshot-Komponente erkannt werden. Mit Hilfe des Befehls „Control.Get [▶ 25]“ kann die Eigenschaft „Farbe“ abgefragt werden. Aufgrund des Verhaltens der „Save“-Schaltflächen (keine Statusänderung) sind sie von dieser Lösung ausgeschlossen.

Tab. 1: Farbeigenschaft und ihr entsprechender Snapshot-Status

Farbe	Status
‘@7F19’	„entladen“
‘@7F7F’	„geladen“
‘@7F4C’	„geändert“

Der Status „geändert“ bedeutet, dass die betreffenden Steuerungen nach dem Laden des Snapshots geändert wurden. In diesem Status können die „Save“-Schaltflächen zum Überschreiben eines Snapshots verwendet werden.

Diese Logik wurde bereits im Funktionsbaustein FB_QRC_ResExtract [▶ 17] implementiert.

1. Diese Lösung funktioniert nur mit den Standard-Schaltflächenfarben. Die Farbe der Snapshot-Schaltflächen darf NICHT geändert werden. Anderenfalls kann ihr Status vom Funktionsbaustein `FB_QRC_ResExtract` nicht erkannt werden.
2. Diese Methode wurde speziell für Snapshot-Schaltflächen entwickelt, sie funktioniert nicht bei anderen Steuerungen mit Triggertyp.
3. Die Schaltflächen „Load Prev“ und „Load Next“ sind von dieser Lösung ausgeschlossen, da ihre Eigenschaft „Farbe“ nicht vom Q-SYS-Gerät abgefragt werden kann.

6.6 Snapshot-Status und zugehörige Eigenschaften

In der früheren Version des QRC-Beispielprojekts wurde die Eigenschaft „Farbe“ der Schaltfläche „Load“ verwendet, um den Snapshot-Status zu erkennen. Dies ist eine inoffizielle „Umgehung“, die jedoch noch gut funktioniert. Nun wurde eine neue Möglichkeit implementiert. Die Funktionalität „Farbe“ ist nach wie vor vorhanden.

Ab Version 1.1.0.0 kann nun der Funktionsbaustein FB_QRC_Snapshot [▶ 39] verwendet werden, um einen Snapshot direkt zu laden oder zu speichern und um mehrere Snapshots manuell abzufragen. Mit Hilfe der Methode AddSnapshotControl [▶ 32] des Funktionsbausteins FB_QRC_ChangeGroup [▶ 28] können mehrere

Snapshots einer Änderungsgruppe beigefügt werden. Anschließend können ihre Status zyklisch gepollt werden. Der Funktionsbaustein FB_QRC_ResExtract [▶ 17] wurde ebenfalls aktualisiert, um den Antwort-Frame eines Snapshots auszulesen.

Eine Snapshot Bank besteht aus einem Snapshot Controller und allen von Ihnen hinzugefügten Steuerungen und Komponenten. Dieser Snapshot Controller ist auch eine Komponentensteuerung und wird im folgenden Dokument zum besseren Verständnis als „Snapshot-Komponente“ bezeichnet.

In einer Snapshot-Komponente hat jeder Snapshot zwei zugehörige Eigenschaften/ Komponentensteuerungen, die nachstehend aufgeführt sind:

- **last.x**: Beschreibt, ob der Snapshot geladen ist oder nicht.
- **match.x**: Beschreibt, ob Steuerungen im Snapshot nach dem Laden dieses Snapshots geändert wurden.

Diese zwei Eigenschaftsnamen sind über das Menü „View Component Control Info...“ in Q-SYS Designer zu finden. Mit Hilfe dieser zwei Eigenschaften kann der Snapshot-Status bestimmt werden.

Die Eigenschaftswerte „last“ und „match“ und ihre entsprechenden Snapshot-Status sind nachstehend aufgeführt.

	last = false	**last = true**
match = false	entladen	geändert
match = true	-	geladen

Diese Logik wurde im Funktionsbaustein FB_QRC_ResExtract [▶ 17] und den Methoden AddSnapshotControl [▶ 32] und GetSnapshotState [▶ 41] implementiert.

Mehr Informationen:
www.beckhoff.de/entertainment-industrie

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