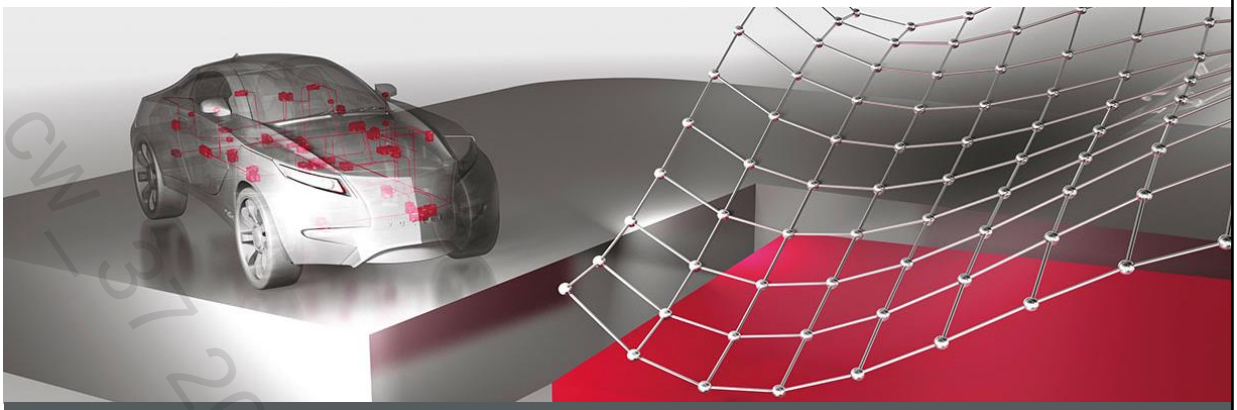


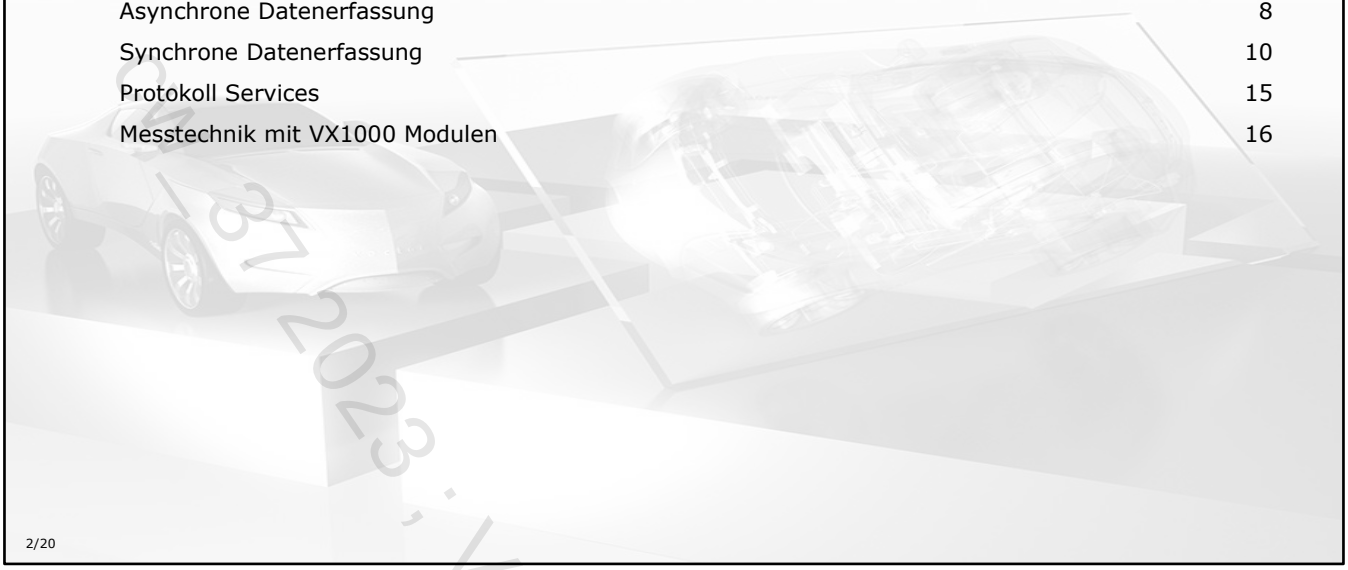
CANape Training

Einführung in das CCP und XCP Protokoll

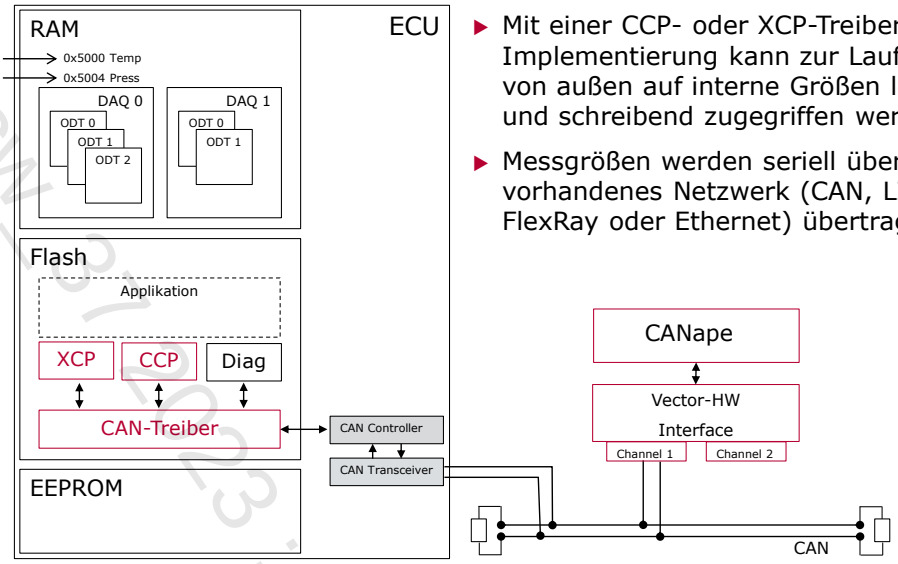
V5.1 | 2020-09-29

Agenda

- **CCP / XCP Protokoll Grundlagen** **3**
- Asynchrone Datenerfassung 8
- Synchrone Datenerfassung 10
- Protokoll Services 15
- Messtechnik mit VX1000 Modulen 16



Übersicht



- Mit einer CCP- oder XCP-Treiber Implementierung kann zur Laufzeit von außen auf interne Größen lesend und schreibend zugegriffen werden.
- Messgrößen werden seriell über ein vorhandenes Netzwerk (CAN, LIN, FlexRay oder Ethernet) übertragen.

3/20

Ein Steuergerät (ECU) ist Teil eines technischen Systems (z.B. Automatikgetriebe).

Das Verhalten des technischen Systems wird von physikalischen Größen beeinflusst. Über Sensoren werden die Eingangsgrößen vom Steuergerät erfasst und anschließend per Software weiterverarbeitet. Das Steuergeräteprogramm greift dabei auf eine hohe Anzahl von Steuer- und Regelparameter bei der Berechnung komplexer Algorithmen zurück.

Aufgabe des Applikationsingenieurs ist es, diese Steuer- und Regelparameter so zu optimieren, dass das technische System die gestellten Anforderungen erfüllt.

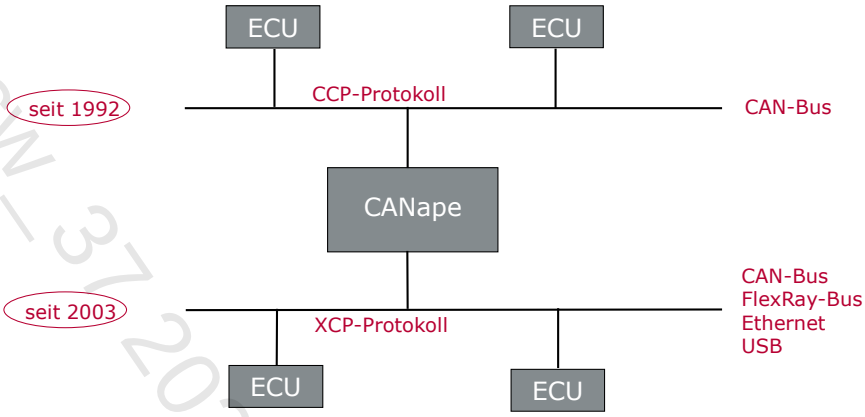
Zu diesem Zweck benötigt der Applikationsingenieur ein Werkzeug, welches ihm zur Laufzeit des Systems erlaubt, gleichzeitig Parameterwerte zu verstellen und Messsignale zu messen.

CANape bietet dem Anwender verschiedene Verfahren an, Parameterwerte im Steuergerät zu verstellen (auch: kalibrieren oder applizieren). Voraussetzung ist dabei, dass die Parameter in einem beschreibbaren Speicherbereich abgelegt sind. Dieser sogenannte applizierbare Speicher könnte z.B. im RAM oder EEPROM liegen.

Der Zugriff in den Gerätespeicher wird bei modernen Applikationssystemen mit Hilfe von ASAM konformen Protokollen ermöglicht.

CAN	Controller Area Network
CANape	CAN application environment
CCP	CAN Calibration Protocol
ECU	Electronic Control Unit
XCP	Universal Calibration Protocol

Chronologie von CCP und XCP



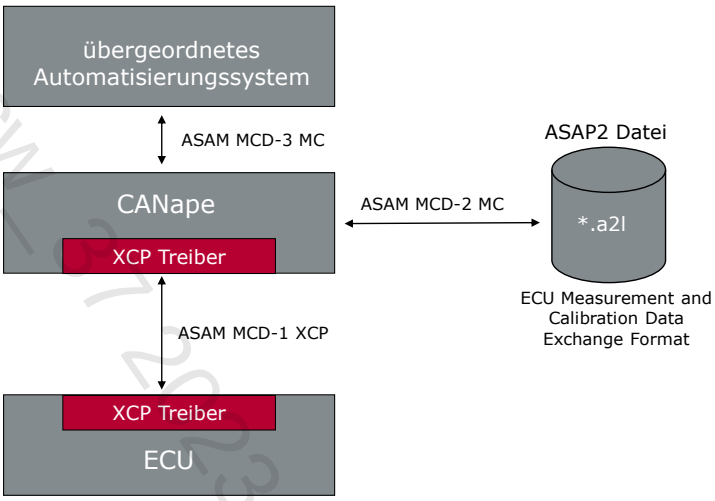
4/20

Im Herbst 1992 wurde das erste Release des CCP-Standards herausgegeben. CCP ist ein auf CAN aufgesetztes Protokoll, welches speziell zum vereinheitlichten Messen und Kalibrieren von Steuergeräten unterschiedlichster Hersteller entwickelt wurde.

Im Laufe der Zeit wurden zur Vernetzung von Steuergeräten neben dem CAN-Bus weitere Bussysteme entwickelt. Die Steuergeräte selbst wurden immer komplexer und leistungsfähiger. Für die notwendigen Mess- und Kalibrieraufgaben in einem Steuergerät wurde ein umfangreicheres, und vor allem ein vom Transportmedium (CAN, FlexRay, Ethernet,…) unabhängiges Protokoll notwendig.

Daher veröffentlichte der ASAM e.V. 2003 den neuen XCP-Standard. Im Vergleich zu CCP stellt XCP ein verbessertes und verallgemeinertes Protokoll dar. Das „X“ deutet auf die verschiedenen Transportschichten hin. XCP ist nicht abwärts kompatibel zu einem bereits eingesetzten CCP.

ASAM Schnittstellenmodell



5/20

Das ASAM-MC Schnittstellenmodell in dieser Folie stellt eine Teilmenge des ASAM-MCD Schnittstellenmodells dar. Das ASAM-MCD-Schnittstellenmodell ist für Mess-, Kalibrier- und für Diagnose-Systeme entwickelt worden.

Für Applizier- bzw. Kalibrieraufgaben sind demnach nur die drei folgenden Schnittstellen von besonderem Interesse:

ASAM MCD 1 MC (ehemals ASAP1a):

Regelt das Protokoll für den Datenaustausch über den Transport-Layer.

ASAM MCD 2 MC (in Kurzform auch als ASAP2-Interface bezeichnet):

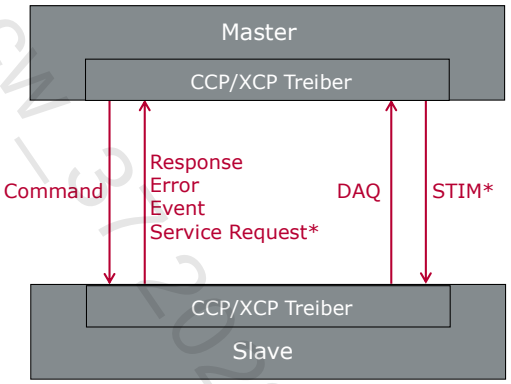
In einem standardisierten Umfeld muss natürlich auch der Zugriff auf die Parameter, Kennlinien/Kennfelder und Messwerte geregelt sein. Die ASAP2 Spezifikation beschreibt, wie die Informationen für ein Messwerkzeug zur Verfügung gestellt werden müssen. Dies bedeutet, dass die Syntax für die Steuergerätebeschreibungsdatei exakt vorgeschrieben ist.

ASAM MCD 3 MC (in Kurzform auch als ASAP3-Interface bezeichnet):

Ein Kalibrierwerkzeug wie CANape kann auch von außen über einen Prüfstand gesteuert werden. CANape stellt in diesem Fall seine ASAM MCD 1 MC - Schnittstelle zur Verfügung und ermöglicht somit einem Prüfstand oder einer anderen Applikation den Zugriff auf den Steuergeräte-Code und dessen Parameter.

Aufbau des Treibers

- CCP und XCP verwenden das Master-Slave Prinzip



- DAQ=Data Akquisition
- STIM=Data Stimulation
  - * nur bei XCP spezifiziert

6/20

Master des CCP/XCP Protokolls ist das Mess- und Kalibrierwerkzeug, die angeschlossenen Steuergeräte sind die Slaves. Grundsätzlich initiiert der Master die Kommunikation mit den Slaves. Dazu baut der Master zu einem Slave eine kontinuierliche logische Verbindung über den Bus auf. Über diese logische Verbindung können dann Informationen ausgetauscht werden: Jedes Kommando muss vom betreffenden Slave quittiert werden (positiv oder durch eine Fehlermeldung). Der Master kann zu mehreren Slaves gleichzeitig logische Verbindungen aufgebaut haben. Bei CCP und XCP sind die ausgetauschten Informationen unterschiedlich strukturiert, im folgenden eine kurze Übersicht zu den beiden Protokollen:

CCP

Auf dem CAN-Bus werden alle Informationen in sogenannten „Message Objects“ übertragen. In jedem Message Object haben maximal 8 Byte Daten Platz.

Für das auf CAN basierende CCP-Protokoll werden zwei Arten von Message Objects verwendet:

Command Receive Object (CRO): sendet Kommandos vom Master zum Slave

Data Transmission Object (DTO): sendet positive Kommandoquittungen, Fehlermeldungen und Ereignisse vom Slave zum Master

Jedem Message Object ist ein CAN-Identifier zugewiesen, d.h. CCP benötigt nur zwei Identifier.

XCP

XCP ist vom Transportprotokoll unabhängig und bietet erweiterte Funktionen im Vergleich zu CCP. Aus diesem Grund wurde die Terminologie geändert. Bei XCP werden alle Informationen in XCP Paketen übertragen, die maximale Länge der zu übertragenden Daten hängt vom verwendeten Transportprotokoll ab (bei CAN weiterhin maximal 8 Byte). XCP kennt zwei Pakettypen:

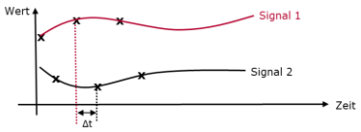
Command Transfer Object (CTO): überträgt Kommandos vom Master zum Slave, überträgt positive Kommandoquittungen, Fehlermeldungen, Ereignisse und Service Requests vom Slave zum Master

Data Transfer Object (DTO): überträgt alle Daten für die synchrone Datenerfassung und Datenstimulation (DAQ und STIM Pakete)

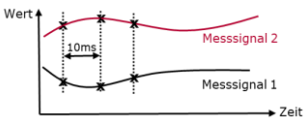
Datenerfassung

- Je nach Steuergeräte-Implementierung können über CCP / XCP folgende Datenerfassungsmodi (Messraster) verwendet werden:

- Asynchrone Messdatenerfassung
  - Polling Mode: Anfrage / Antwort



- Synchrone Messdatenerfassung (optional)
  - Cyclic Mode: Periodische Tasks steuern die Messdatenerfassung (z.B. alle 10 ms)
  - Event Mode: Spontanes Ereignis steuert die Messdatenerfassung (z.B. Kurbelwellenposition)



7/20

Polling Mode:

Im Polling Messwerterfassungs-Mode werden die Messwerte eines Signals auf Anfrage von dem Steuergerät zurückgegeben. CANape schickt entsprechend der eingestellten Abtastrate eine Anfrage an das Steuergerät und bekommt von diesem den aktuellen Signalwert zurückgeliefert.

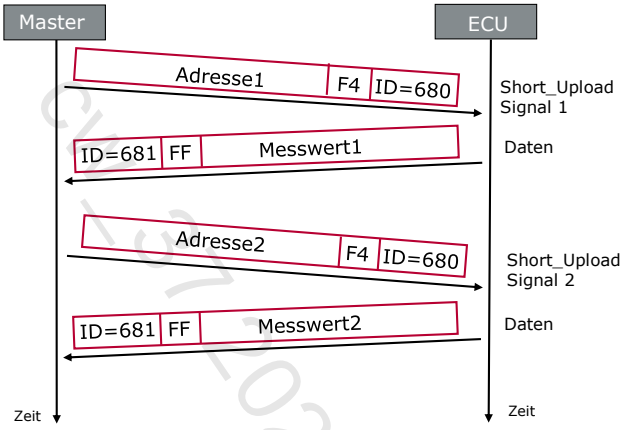
Cyclic Mode:

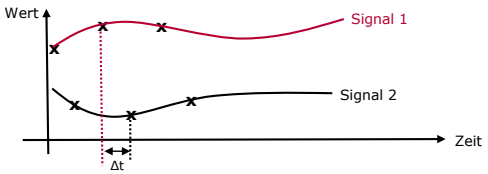
Bei der zyklischen Messdatenerfassung wird von CANape nur einmalig eine Anforderung zum zyklischen Senden der Signalwerte entsprechend der eingestellten Rate an das Steuergerät gesendet. Das Steuergerät schickt daraufhin regelmäßig die aktuellen Messwerte. Dieser Mode wird nur unterstützt, wenn der Steuergeräteentwickler diesen Mode im CCP/XCP-Protokoll aktiviert hat.

Event Mode:

Im Event Mode wird die Datenübertragung von definierten Ereignissen ausgelöst. Diese Ereignisse müssen im Steuergeräte-Code abgelegt sein. Tritt das Ereignis auf, schickt das Steuergerät die aktuellen Messwerte.

Polling Mode (Beispiel CCP)





8/20

In obigen Beispiel werden zwei Messsignale im Polling Mode in regelmäßigen Abständen, z.B. alle 10ms, gemessen.

Für jedes der Messsignale gilt folgender Ablauf:

Der Master sendet eine Anfrage in Form eines Short Upload Kommando zum Slave

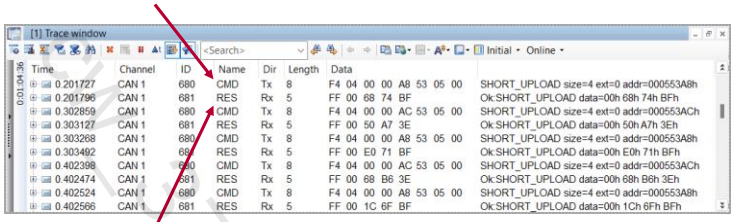
Der Slave antwortet mit einer positiven Response und dem gewünschten Messwert

Auf diese Weise erhält der Master von jedem Messsignal alle 10ms den aktuellen Wert. Allerdings gibt es zwischen den Messpunkten der Signale 1 und 2 einen Zeitversatz Δt. Hierbei spricht man von einer asynchronen Datenerfassung.

Polling Mode (Beispiel CCP)

- Tracing einer Datenerfassung im Polling-Modus

Command



Response

- 0xF4 im Byte(0) der Command Botschaft steht für das Short_Upload Kommando
- Werte des Messsignals stehen in der Response

Structure of data in CRO

Position	Type	Description
0	byte	Command Code = SHORT_UP 0x0F
1	byte	Command Counter = CTR
2	byte	Size of data block to be uploaded in bytes (1...5)
3	byte	Address extension
4	unsigned long	Address

The master device sends a SHORT_UP CRO to the slave device. The command counter CTR currently is 0x23, the size of data is 0x04 and the source adress is 0x12345678:

byte 0	1	2	3	4	5	6	7
0x0F	0x23	0x04	0x00	0x12	0x34	0x56	0x78

The following return information is expected (Contents of returned DTO:)

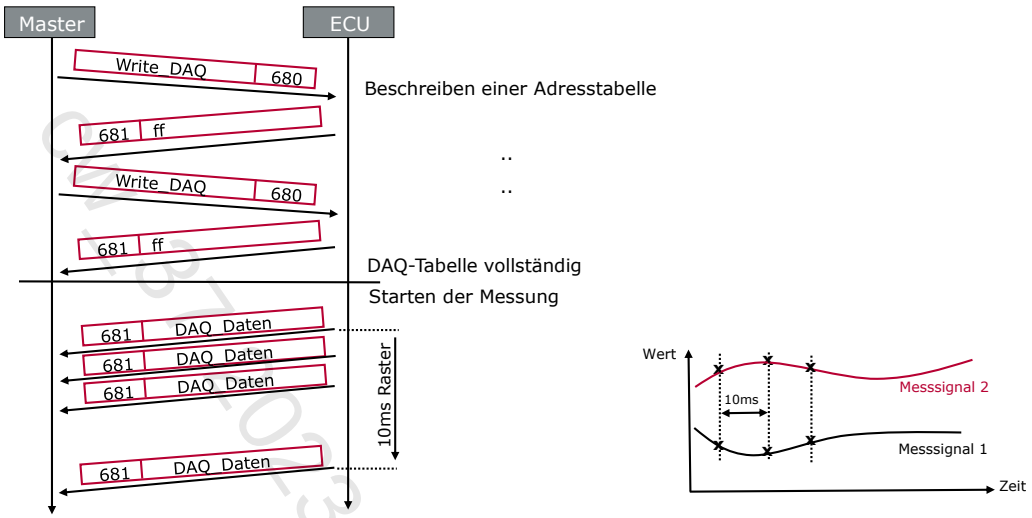
Position	Type	Description
0	byte	Packet ID: 0xFF
1	byte	Command Return Code
2	byte	Command Counter = CTR
3 ... 7	bytes	requested data bytes

The slave device answers with a DTO containing ACKNOWLEDGE (0x00), the CTR of the CRO and the requested data bytes:

byte 0	1	2	3	4	5	6	7
0xFF	0x00	0x23	0x10	0x11	0x12	0x13	--

Synchrone Datenerfassung

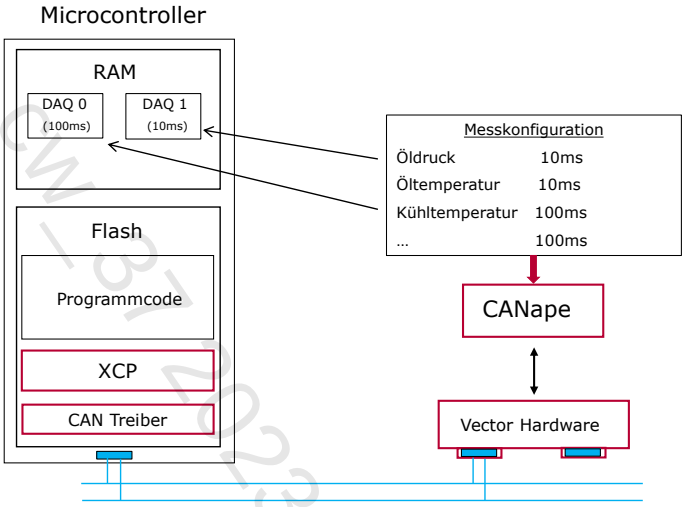
Cyclic Mode



10/20

In der Zeichnung werden zwei Messsignale im zyklischen Mode alle 10ms gemessen. Da die Werte der Messsignale 1 und 2 zum gleichen Zeitpunkt abgetastet werden, spricht man von einer synchronen Datenerfassung.

DAQ-Listen Konzept



- XCP-Master generiert entsprechend der Messkonfiguration DAQ-Listen im Speicher der ECU
- Größe der DAQ-Listen und demnach die maximale Anzahl der möglichen Adresseinträge ist implementierungsabhängig

11/20

Die Vorbereitung der synchronen Messdatenerfassung läuft in einem Messwerkzeug wie CANape in mehreren Schritten ab.

Zuerst definiert der CANape–Anwender seine gewünschten Messsignale und deren Messraster in der Messkonfiguration.

Anschließend wird CANape sämtliche Adressinformationen zu den gewünschten Messdaten in sogenannte DAQ-Listen schreiben. Wenn beispielsweise der CANape Anwender zwei verschiedene Messraster bei der Definition seiner Messsignale ausgewählt hat, wird CANape zwei DAQ-Listen formulieren und diese temporär ins RAM des Steuergerätes schreiben. Ab diesem Zeitpunkt kennt der CCP / XCP-Treiber des Steuergerätes “ WAS “ gemessen werden soll!

Im nächsten Schritt kommuniziert CANape dem CCP / XCP-Treiber des Steuergerätes welche DAQ-Liste mit welchem internen Messraster (EVENTKANAL -> siehe nächste Folie) abgearbeitet werden soll.

Bei der Auswahl der Messraster innerhalb der Messsignalliste kann der CANape Anwender nur auf Messraster zurück greifen, die innerhalb des Steuergeräte Treibers vorbereitet wurden.

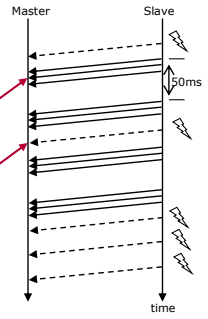
Demzufolge kann der CANape Anwender zwar frei wählen welche Signale ermittelt werden sollen, aber die Definition des Messrasters obliegt indirekt dem Steuergeräteentwickler.

Datenerfassungssteuerung

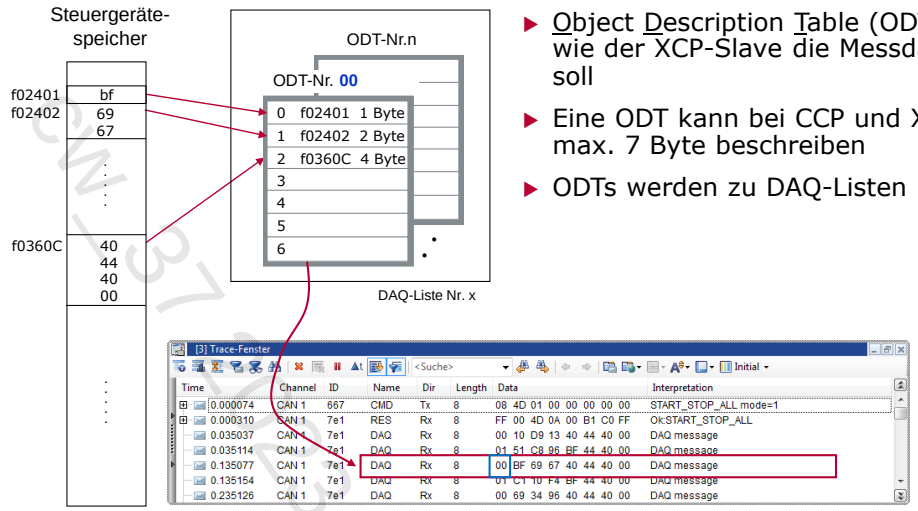
- xcpEvent()-Aufrufe im Programmcode steuern den Zeitpunkt der Datenerfassung

Programmcode

void main(void){	Task1() {	Task2() {
.....		
xcpInit();	....	....
....	....	....
//activate every 50ms Task1();	…	....
SetCyclicTimer (50);	a=.....;	u=.....;
....	b=.....;	v=.....;
While (1) {	.....	....
If (crankShaft==OT) Task2();}	xcpEvent(1);	xcpEvent(2);
}	}	}



Organisation der DAQ - Listen



- Object Description Table (ODT) beschreibt wie der XCP-Slave die Messdaten verpacken soll
- Eine ODT kann bei CCP und XCP on CAN max. 7 Byte beschreiben
- ODTs werden zu DAQ-Listen gruppiert

13/20

Geht man davon aus, dass der CANape Anwender sehr viele Signale für die Messung definiert hat, wird es bei einem Transportmedium wie CAN notwendig sein, die gemessenen Signalwerte über mehrere XCP-Pakete verteilt, segmentiert zum Master zu senden. Bei einem Diagnosetreiber wie z.B. KWP2000 on CAN wird für die Übertragung größerer Datenmengen ein Transportprotokoll notwendig.

Bei CCP und XCP wird bereits in der DAQ-Liste (Adressliste) festgelegt, wie die einzelnen XCP-Pakete (Segmente) aufgebaut sind. In der Zeichnung aus obiger Folie ist erkennbar, dass eine DAQ-Liste im Grunde nur eine Hülle für viele kleine ODT-Listen darstellt, wobei eine ODT-Liste beschreibt, welche Signalwerte gemeinsam in einem XCP-Paket von der ECU an das Messwerkzeug versendet werden sollen.

Die Größe einer ODT-Liste, d.h. die Anzahl der Signalwerte, beschrieben durch eine Adress- und Längen-Angabe, ist bei XCP abhängig von dem Transportmedium. Bei XCP on CAN können maximal nur sieben Signale gruppiert übertragen werden. Bei den acht Datenbyte einer CAN-Message könnte man meinen es wären acht Signale, aber auf Grund einer sogenannten Paket-Idendifikation (PID) mit einer Länge von mindestens einem Byte sind maximal nur sieben Nutzbyte in dem CAN-Paket nutzbar für die Übertragung von Messwerten.

Performance

- Die Performance der synchronen Messdatenerfassung ist begrenzt durch:
  - Größe des Speicherbereichs für die DAQ-Listen
  - Im besonderen Maße durch die Bandbreite des Transport Medium
  - Rechenleistung, die für den XCP-Treiber zur Verfügung steht
    - Treiber sollte die eigentliche Applikation möglichst nicht beeinflussen

Die Bandbreite der verwendeten Transport Layer stellt eine entscheidende Rolle für die Performance dar. Bei CCP oder XCP on CAN hängt die Anzahl der Messdaten, die synchron gemessen werden können, von verschiedenen Faktoren ab. Ein wesentlicher Faktor stellt die eingestellte Baudrate des Netzwerks dar. Wenn beispielsweise eine Baudrate von 500KBaud eingestellt ist und von einer durchschnittlichen Botschaftslänge von 125 Bitzeiten ausgegangen wird, können ungefähr 4000 CAN-Pakete pro Sekunde versendet werden. Allerdings stellt dies ein grober Verstoß an die praxisüblichen Buslasten dar. In einem CAN-Netzwerk, welches rein für die Messtechnik vorhanden ist, ist diese Obergrenze an CAN-Paketen denkbar. Ein zweites entscheidendes Kriterium ist das Messraster. Möchte man z.B. im 10ms Raster Messdaten erfassen, so wären gerade mal 40 CAN-Pakete in einem reinen Messtechnik CAN-Netzwerk möglich. Vor dem Hintergrund das mindestens ein Nutzbyte für die Adressierung notwendig ist, könnten maximal 40x7Byte also 280 Byte Messdaten pro 10 Millisekunden transportiert werden.

Überblick CCP / XCP Features

- CCP und XCP bieten neben der synchronen und asynchronen Datenerfassung weitere optionale Dienste an:
  - Zugriffsschutz (Seed&Key)
  - Automatische Auswahl der *.a2l Beschreibungsdatei
- XCP unterstützt darüber hinaus neue Features
  - Synchrone Datenstimulation und Bypass-Funktionalität
  - Automatische Erkennung und Konfiguration der Slaves
  - Erfassung von Messdaten mit Zeitstempel
  - Resume Mode / Power-on Messung

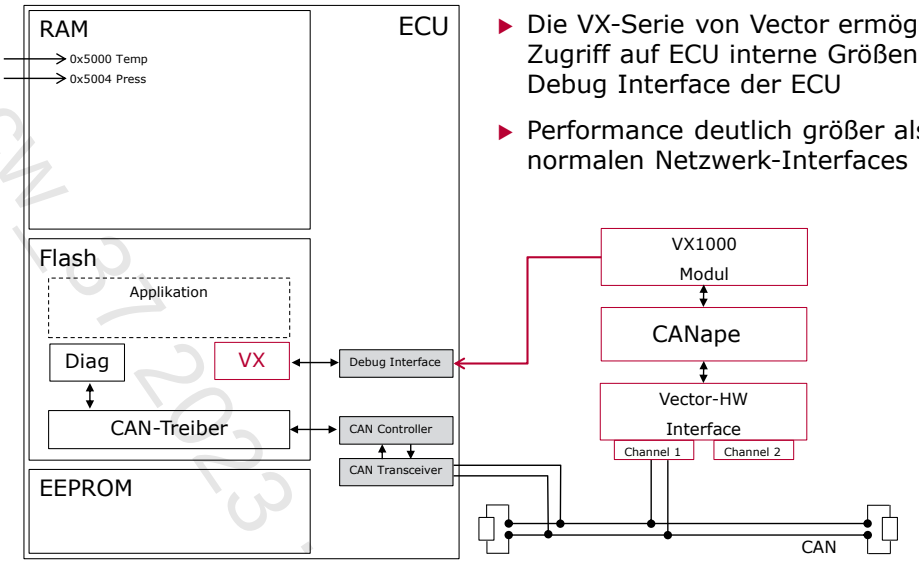
15/20

Die Grundfunktionen Upload und Download zum Lesen und Schreiben in den Gerätespeicher sind in jeder Implementierung zu finden.

Die Seed&Key–Funktionalität für den Zugriffsschutz ist dagegen optional. Immer dann, wenn der CCP / XCP-Treiber im Serien-Steuergerät verbleibt, ist es sinnvoll, unautorisierten Personen den Zugriff zu unterbinden.

Während der Steuergeräteentwicklung verändert sich der Source Code zwangsläufig sehr häufig. Da sich bei der Entwicklung des Steuergerätecode nach jedem Kompiliervorgang unter Umständen die Adressen der Mess- und Verstell-Größen ändern können, ist es sinnvoll, hier einen Mechanismus zu verwenden, der automatisch die richtige Beschreibungsdatei auswählt. So gibt es in den beiden Treibern Funktionen, die den Namen der Beschreibungsdatei aus dem Steuergerätespeicher auslesen können, vorausgesetzt dieser wird vom Steuergeräteentwickler darin auch abgelegt.

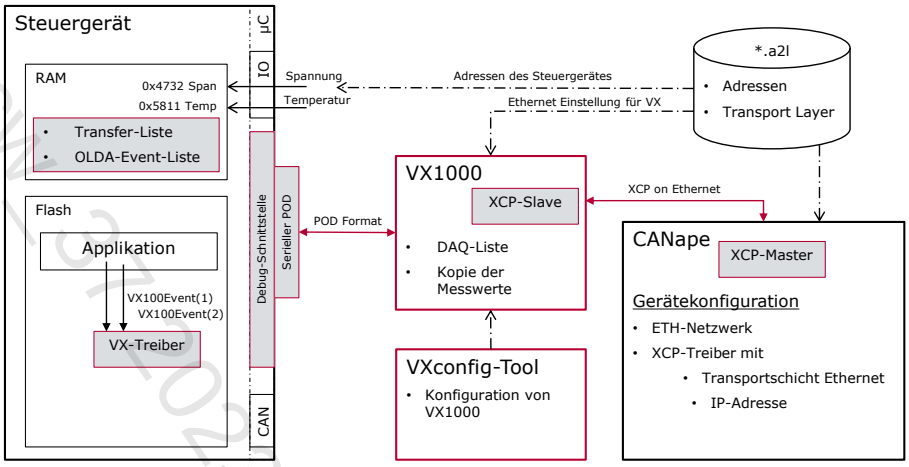
Überblick



- Die VX-Serie von Vector ermöglicht den Zugriff auf ECU interne Größen über das Debug Interface der ECU
- Performance deutlich größer als über die normalen Netzwerk-Interfaces

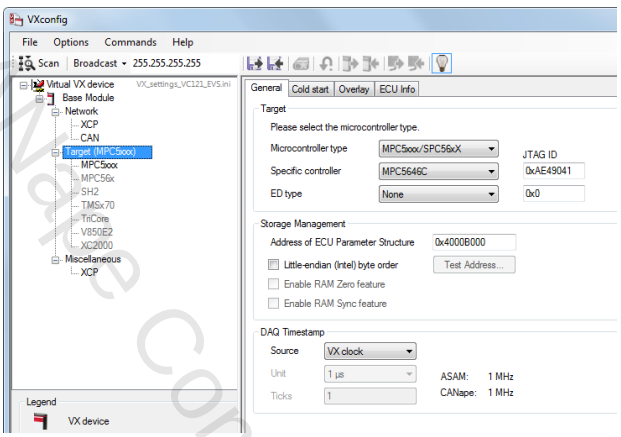
16/20

Übersicht (Serieller POD / RAM-Copy Verfahren)

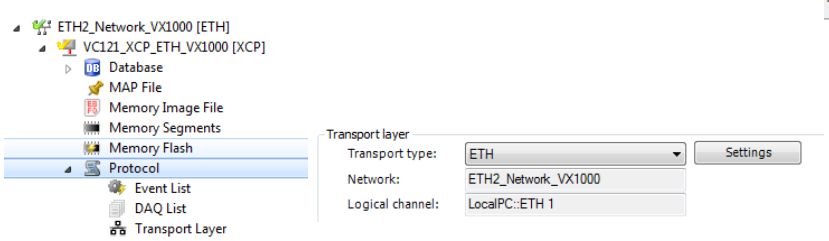


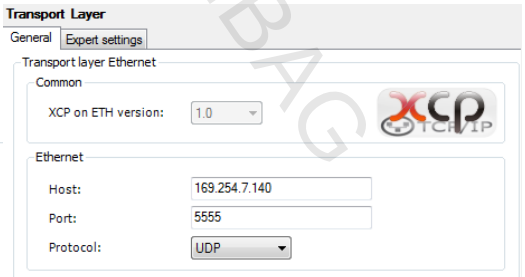
Folgende Vorgehensweise empfiehlt sich:

- Das VX1000 sollte über das zugehörige Konfigurationstool VXconfig konfiguriert werden:
  - Spannung des POD einstellen
  - Mikrocontroller und Frequenz einstellen usw.

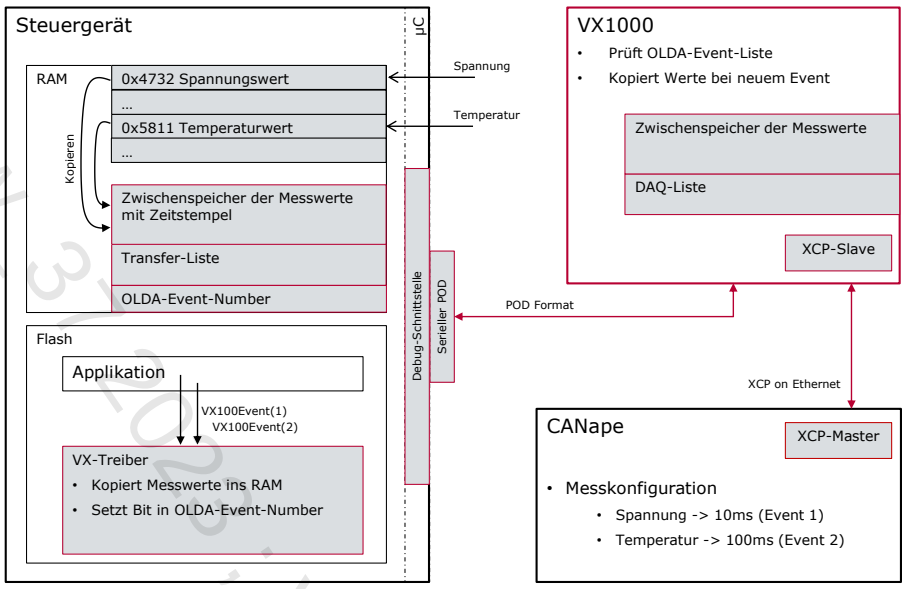


- Für den CANape Anwender verhält sich das VX1000 wie ein XCPonEthernet-Steuergerät
  - Ethernet-Netzwerk definieren
  - XCP-Treiber definieren und Ethernet-Netzwerk zuweisen
    - > IP-Adresse des VX1000 angeben (enthalten in A2L)





Ablauf (Serieller POD / RAM-Copy Verfahren)



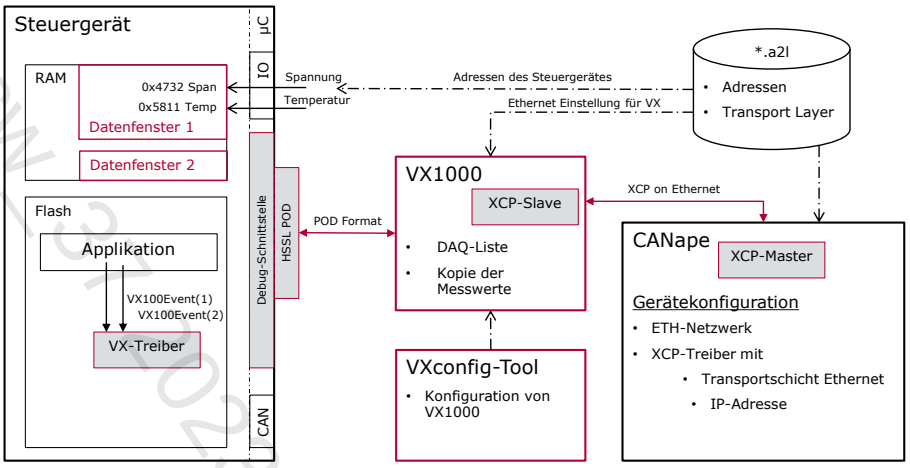
18/20

- Ablauf einer DAQ-Messung mit CANape (Seriell POD / RAM-Copy Verfahren):

- CANape schreibt DAQ Liste ins VX.
- DAQ-Liste von CANape wird vom VX-Modul als Transfer-Liste ins RAM des Steuergerätes geschrieben.
- Kopiervorgänge innerhalb der ECU werden über Event-Aufrufe aus der Applikationssoftware heraus gesteuert (ähnlich der xcpEvent()-Aufrufe bei XCPonCAN werden VX1000Event()-Aufrufe in den Applikations-Code instrumentiert).
- VX-Treiber kopiert Messgrößen entsprechend der Transferliste in einen Zwischenspeicher im RAM des Steuergerätes.
- VX-Treiber signalisiert über eine 32 Bit Liste (Olda-Event-Number) welche VX1000Event()-Kanäle aufgerufen wurden.
- VX-Basismodul überwacht das 32 Bit Register jede 10µs und liest die neuen Messdaten über die Schnittstelle aus und schreibt diese in den Speicher des VX-Basismoduls.
- Anschließend werden die Daten zu CANape über XCP on Ethernet Pakete entsprechend der DAQ-Listen verschickt.
- Beim Polling-Verfahren holt sich der XCP-Slave die Daten aus dem Steuergeräte RAM und sendet sie an CANape.

- Bei diesem Verfahren verhält sich das VX-Basismodul wie ein XCP-Slave in einem Steuergerät. Der CANape Anwender kann auf dem Ethernet Netzwerk eine normale XCPonEthernet Kommunikation mit DAQ-Listen beobachten.

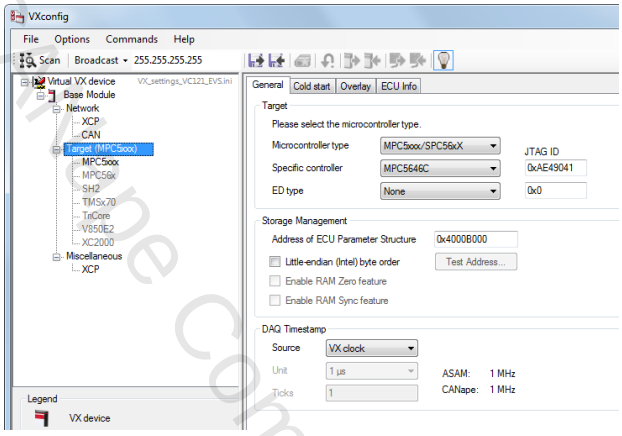
Übersicht (HSSL POD / Datentrace Verfahren)



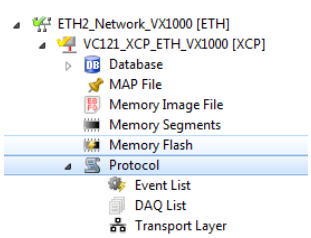
19/20

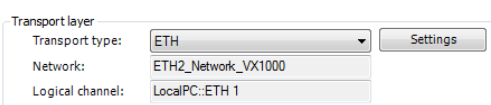
Folgende Vorgehensweise empfiehlt sich:

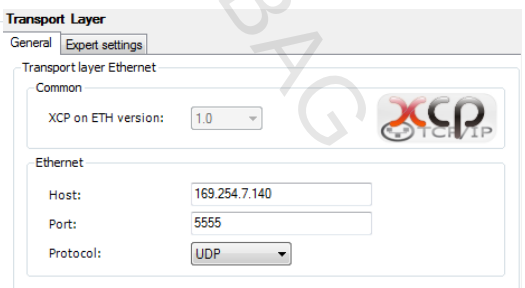
- Das VX1000 sollte über das zugehörige Konfigurationstool VXconfig konfiguriert werden:
  - Spannung des POD einstellen
  - Mikrocontroller und Frequenz einstellen
  - Bis zu 4 Datenfenster definieren usw.



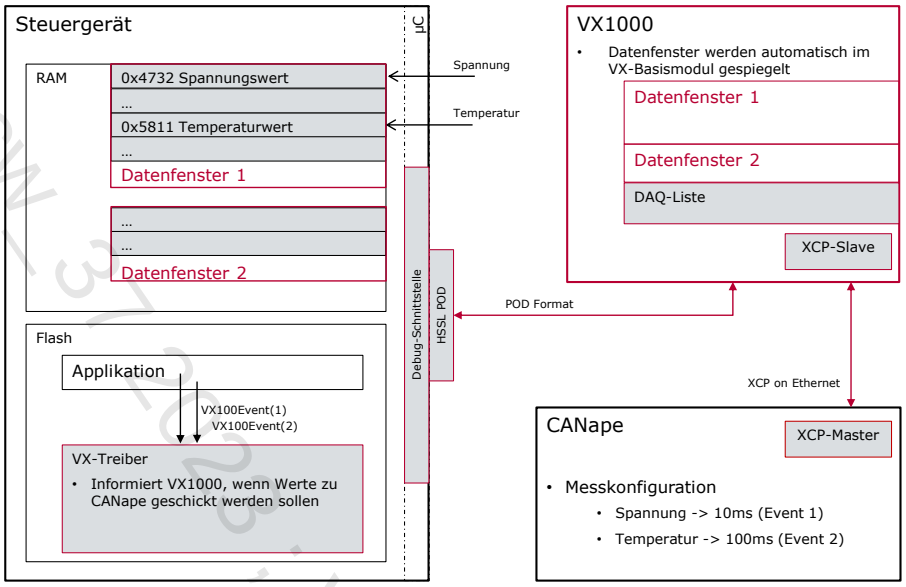
- Für den CANape Anwender verhält sich das VX100 wie ein XCPonEthernet-Steuergerät
  - Ethernet-Netzwerk definieren
  - XCP-Treiber definieren und Ethernet-Netzwerk zuweisen
    - > IP-Adresse des VX1000 angeben







Ablauf (HSSL POD / Datentrace Verfahren)



20/20

- Ablauf einer DAQ-Messung mit CANape (Paralleler POD / Datentrace Verfahren):

- CANape schreibt DAQ Listen entsprechend der CANape-Messkonfiguration in das VX-Basismodul inklusive der gewählten Messraster-Informationen. Genauso wie bei einem normalen XCP-Slave.
- Die Messdaten, die der CANape Anwender auslesen möchte, müssen dabei innerhalb bestimmter Datenfenster (Adressbereichen) liegen, die zuvor über das VXconfig-Tool konfiguriert wurden. Es können bis zu 4 Datenfenster definiert werden.
- Sämtliche Änderungen innerhalb der Datenfenster werden von der Debug-Schnittstelle automatisch in einen VX-Spiegelspeicher geschrieben. Hierzu werden keine Transferlisten benötigt.
- Die Messraster die CANape zuvor den DAQ-Listen zugeordnet hatte, werden im Steuergerät den entsprechenden VX1000event()-Aufrufen zugeordnet. Sobald einer der VXevent()-Aufrufe abgearbeitet wurde, bekommt das VX-Basismodul ein Zeichen, dass aktuell synchrone Daten im Spiegelspeicher vorhanden sind. Zu diesem Zeitpunkt kann nun das VX-Basismodul mit der Übertragung der Messdaten beginnen und zwar über ganz normale XCPonEthernet Pakete. Die VXevent()-Aufrufe haben demnach lediglich noch die Aufgabe die Datenübertragung vom VX-Basismodul zu CANape zu steuern..
- Bei dem Polling-Messraster von CANape holt sich CANape die Daten ebenso aus dem VX-Spiegelspeicher.