EPOS Command Library:
Welche Zykluszeiten sind beim Datenaustausch möglich?

Thema:

- Warum können über USB (oder andere Schnittstellen) die Daten nicht im 1 ms Takt ausgetauscht werden?
- Welche Zykluszeiten können mit der EPOS Command Library bei einem Datenaustausch erreicht werden?

Anmerkungen

- Der Begriff “EPOS” bezieht sich nachfolgend auf die EPOS, EPOS2 und EPOS4 Produktreihen.
- Der Begriff „EposCmd Library“ wird nachfolgend für die „EPOS Command Library“ verwendet. Detailinformationen zu der Libary und den Funktionen finden sich in dem Handbuch „EPOS Command Library.pdf“.

Ausgangslage:

Ein Applikationsprogramm basiert auf der EPOS Command Library (-> EposCmd.dll) und wird auf einem Master (z.B. PC, Raspberry Pi, BeagleBone) mit Windows oder Linux Betriebssystem ausgeführt. Das Programm tauscht Daten mit einer EPOS über RS232, USB oder CAN aus. Die Ausführungszeit einer typischen EposCmd Funktion, welche eine Bewegung konfiguriert oder Daten ausliest, wurde mit ca. 10 – 20 ms gemessen. Diese Zeitspanne erscheint ziemlich lange insbesondere da die verwendete Schnittstelle (z.B. USB) sehr hohe Datentransfer-Raten bietet und maxon eine Zykluszeit von 1 ms bei der EPOS4 auf dem Bus angibt.

Warum ist die gemessene Zeit (z.B. 10 – 40 ms) für den Datenaustausch soviel höher als die erwartete Zeit (z.B. 1 – 2 ms) und wie kann dies verbessert werden?

Lösung:

Etwas technischer Hintergrund zuerst:

- Die EPOS kann CAN Nachrichten typisch im 1 ms austauschen und verarbeiten.
- Eine periodischer Datenaustausch mehrerer Dateninhalte im 1 ms Takt (oder weniger als 10 ms) erfordert die Erfüllung weiterer Vorbedingungen betreffend der Systemumgebung:
  - Vorbedingung 1:
    Echtzeitfähiges Bus-System (wie CAN oder EtherCAT).
  - Vorbedingung 2:
    Ein Datenaustausch basierend auf sogenannten PDOs (= Process Data Object).
  - Vorbedingung 3:
    Ein echtzeitfähiger Master für die schnelle, zyklische Datenverarbeitung und Kommunikation.

  Zusammengefasst bedeutet dies, dass ein echtzeitfähiger Master den sogenannten “synchronen” PDO basierenden Datenaustausch über CAN oder EtherCAT unterstützen muss.

- Windows und Linux sind keine echtzeitfähigen Betriebssysteme und bieten kein exakt definierte Bearbeitungs- und Reaktionszeit, was Voraussetzungen für einen schnellen periodischen Datenaustausch sind.
  - Nicht echtzeitfähige Betriebssysteme wie Windows oder Linux teilen die Rechenleistung des Computers zwischen verschiedenen Aufgaben und Prozessen auf. Für Anwendungsprogramme und Library-Funktionen können keine fixen Zeitfenster oder Prioritäten definiert werden. Dies kann dazu führen, dass die Datenverarbeitung vor oder nach einer Kommunikation mit der EPOS für einige Millisekunden aufgrund der Ausführung von priorisierte Prozesse (z.B. Festplattenzugriffe oder Grafikfunktionen) durch das Betriebssystem unterbrochen oder blockiert wird. Das Starten des Datenaustausch und die Bereitstellung der Informationen im Anwendungsprogramm wird hierdurch verzögert und die Ausführungszeit ist deutlich grösser als die eigentliche Reaktionszeit der EPOS.
- PC-basierende Entwicklungswerkzeuge (z.B. Microsoft Visual Studio, NI LabView) und hiermit erzeugte Anwendungsprogramme, wie auch die EposCmd Library unterliegen der Prozessorzuteilung und Priorisierung durch das Betriebssystem.
  - Solche auf der EposCmd Library basierenden Anwendungsprogramme ermöglichen keinen schnellen zyklischen Datenaustausch und es kann kein fixes Antwortzeitverhalten spezifiziert werden, da dieses durch das Betriebssystem und weitere aktive Prozesse beeinflusst wird.
- Die EposCmd Library ist ausschliesslich für die Verwendung mit Windows und Linux entwickelt, d.h. nicht für den Einsatz bei zeitkritischen Anwendungen.
  - Die EposCmd Library unterstützt den PDO-basierenden Datenaustausch nicht, weil dieser in den meisten Fällen das definierte Reaktionsverhalten eines Echtzeitbetriebssystems voraussetzt.
  - Die EposCmd Library tauscht Daten auf der Basis von sogenannten SDO (= Service Data Object) Zugriffen aus. Dies ermöglicht das Lesen und die Konfiguration jedes beliebigen Objekts, d.h. SDOs sind sehr universell einsetzbar.
    Der SDOs nutzen einen quittierten Datenaustausch. Der Master sendet eine Information oder Anfrage an die EPOS und wartet auf die Rückantwort. Eine Störung der Übertragung oder unzulässige Datenwerte können an einen Timeout oder einer Fehlermeldung der EPOS erkannt werden. Auf dieser Basis ist ein sehr zuverlässiger Datenaustausch möglich.
    Im Vergleich zu dem PDO Datenaustausch kann mit SDOs jeweils immer nur auf ein Objekt zugegriffen werden, d.h. die Abfrage von verschiedenen Objekten via SDO erfolgt sequentiell und es muss immer die Rückantwort abgewartet werden bevor auf das nächste Objekt zugegriffen werden darf. Der Zugriff mit SDOs benötigt hierdurch mehrere Aufrufe und Zeit für den Datenaustausch als PDOs, welche mehrere Objekte beinhalten können und keine Quittierung des Datenaustausch vorhanden ist.

Konsequenzen in der Praxis?

Der “Flaschenhals” bei der Kommandierung und dem Datenaustausch mit der EPOS ist weder das Bus-System noch die EPOS selber (welche typisch innert 1 ms nach Empfang der Anfrage antwortet). Die Verzögerung wird hauptsächlich durch das Betriebssystem und die Ausführungszeit des Anwendungsprogramms und der EposCmd Library verursacht, welche die Kommunikation starten, auswerten und die Daten weiterleiten muss.

Falls eine Anwendung einen Datenaustausch im Zyklus schneller als 10 – 20 ms erfordert, muss ein Echtzeit-Master (z.B. SPS, NI’s cRIO, zub’s MACS, …) und CAN oder EtherCAT als Bus-Schnittstelle eingesetzt werden.

maxon motor ag
Brünigstrasse 220
CH – 6072 Sachseln
www.maxonmotor.com

EPOS Command Library: Zykluszeit des Datenaustausch

Version: 1.0 (Dt.)
Author : WJ
Date : 2018-06-21

Funktionsaufrufe und Reaktionszeiten bei Nutzung der EposCmd Library:

Die verschiedenen von maxon unterstützten Programmierumgebungen (z.B. C++, C#, Basic, LabView) für die Anwendungsentwicklung nutzen die EposCmd Library. Die EposCmd Library wiederum führt einen SDO-basierenden Datenaustausch durch, welcher eine Anfrage an die EPOS stellt und auf die Rückantwort wartet. Die Reaktion auf das SDI erfolgt in der EPOS intern typisch innerhalb 1 – 2 ms.

Das Anwendungsprogramm und die EposCmd Library muss die Kommunikationszugriffe auf die verschiedenen Informationen aufbauen und nach der Rückantwort auswerten und verarbeiten. Die hierbei auftretenden Verzögerungen werden durch das Betriebssystem des Masters und andere Programme verursacht, die alle gleichzeitig aktiv sind und sich dabei die Computerleistung teilen müssen.

Typischerweise benötigt jeder Zugriff auf eine Information inkl. der Datenverarbeitung im Master ca. 2 – 10 ms. Diese Zeitspanne kann stark variieren je nachdem welche weiteren Prozesse aktuell ebenfalls die Computer Rechenleistung nutzen und eventuell durch das Betriebssystem priorisiert ausgeführt werden. Bei einem nicht echtzeitfähigen Betriebssystem kann hierauf nur begrenzt Einfluss genommen werden und das Reaktionsverhalten kann nicht mit einem fixen Wert spezifiziert werden.

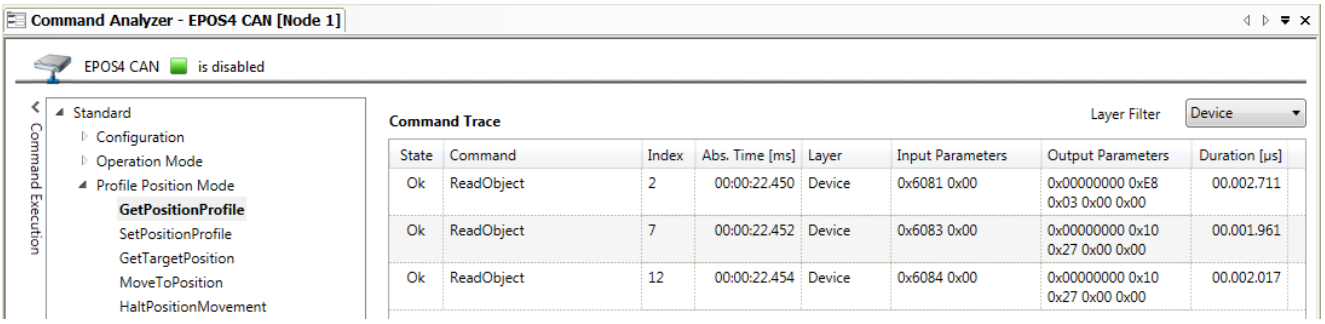
Die komplexen Funktionen der EposCmd Library führen teilweise mehrere Zugriffe auf verschiedene Informationen in den EPOS Objekten aus. Diese Zugriffe können nur schrittweise nacheinander erfolgen. Bei den Funktionen „SetPositionProfile()“ oder „GetPositionProfile()“ als Beispiel werden intern die Daten der Objekte „Profile Velocity“, „Profile Acceleration“ und „Profile Deceleration“ ausgetauscht. Letztendlich kann dies zu einer totalen Bearbeitungszeit von typisch 10 – 20 ms, aber in Extremfällen auch bis 40 ms führen, obwohl jede einzelne der 3 Datenanfragen in der EPOS intern jeweils innerhalb 1 ms verarbeitet wird. Die genannten 40 ms entsprechen zum Beispiel einem typischen Erfahrungswert bei der Ausführung des VI‘s „SetPositionProfile“ in einer LabView Systemumgebung.

Hinweise zur Evaluation und Optimierung

Tipp 1:
EPOS Studio's "Command Analyzer"

Einen tieferen Einblick in das Zeitverhalten und den Datenaustausch der EposCmd Library Funktionen kann mit dem "Command Analyzer" Tool von EPOS Studio gewonnen werden:

- Starten Sie EPOS Studio's "Command Analyzer" Tool.
- Benutzen Sie die selbe Schnittstelle (z.B. RS232, USB), die auch in Ihrer Anwendung und von Ihrem Master später genutzt wird.
- Benutzen Sie die baumartige Befehlsauswahl und wählen Sie die Funktion aus, welche Sie interessiert, z.B. "Profile Position Mode / GetPositionProfile"
- Bestimmen Sie mit dem "Layer Filter" in der rechten oberen Ecke die Tiefe an Informationen, z.B. …
  - "Device": Zeigt die EPOS Objekt-Nummern, die Parameter und Ausführungszeit jeden Zugriffs.
  - "Protocol": Zeigt den Dateninhalt und die Ausführungszeit jeden Zugriffs.
  - "Interface": Zeigt alle Informationen und den kompletten Datenfluss.



Tipp 2:
Reduktion des Datenverkehrs

Durch einen gezielten Datenaustausch, der nur auf relevante Objekte einbezieht, können die Ausführungszeiten reduziert und Anwendungsprogramme schneller gemacht werden. Komplexe Funktionsaufrufe der EposCmd Library lesen und schreiben mehrere Objekte, die jedoch im konkreten Anwendungsfall eventuell gar nicht von Interesse sind oder nicht aktualisiert werden müssen.

Mit den Fuktionen „SetObject()" und „GetObject()" kann gezielt auf jedes einzelnen Element zugegriffen werden und hierduch eventuell der notwendige Datenverkehr optimiert werden.

Beispiel:
Falls nur die „Profile Velocity" für die nächste Bewegung angepasst werden soll, ist es effizienter nur auf das entsprechende einzelne Objekt mit „SetObject()" zuzugreifen als die Funktion „SetPositionProfile()" zu nutzen, welche „Profile Velocity", „Profile Acceleration" und „Profile Deceleration" aktualisiert. Die Informationen zu den Bezeichnungen und Nummern der einzelnen Objekte finden sich in der jeweiligen „Firmware Specification" der einzelnen EPOS Produkreihen.

maxon motor ag
Brünigstrasse 220
CH – 6072 Sachseln
www.maxonmotor.com

EPOS Command Library: Zykluszeit des Datenaustausch

Version: 1.0 (Dt.)
Author : WJ
Date : 2018-06-21

Zusammenfassung:

- Die EPOS Steuerungen reagieren intern auf Kommandos und Anfragen typisch innerhalb von 1 – 2 ms.
- Windows- und Linux-basierende Master unterstützen kein definiertes Zeitverhalten und keine Priorisierung einzelner Anwendungs- oder Library-Prozesse.
- Jeder Datenaustausch und -verarbeitung benötigt gesamthaft typisch 2 - 10 ms.
- Funktionen der EposCmd Library führen teilweise mehrere unterschiedliche Datenzugriffe aus.
- Die Verarbeitungszeit von komplexen EposCmd Library Funktionen kann 10 – 40 ms benötigen.
- Die Ausführungszeit wird durch den Master und weitere parallel laufende Prozesse und Programme beeinflusst.
- Der “Command Analyzer” von EPOS Studio zeigt eine detaillierte Übersicht über die ausgetauschten Daten und die Ausführungszeit der einzelnen Schritte.
- Mit den Funktionen “SetObject()” und “GetObject()” können gezielt nur einzelne Informationen aktualisiert werden. Im Vergleich mit komplexen Funktionen kann so die Anzahl an Zugriffe (auf eventuell nicht relevante Objekte) und die Ausführungszeiten einer Anwendung reduziert werden.
- Falls eine Anwendung einen Datenaustausch häufiger als alle 10 ms erfordert, muss ein Echtzeit-Master und CAN oder EtherCAT als Bus-Schnittstelle eingesetzt werden.