Abspeicherung von IR Daten

Dokumentation der aktuellen Implementierung

Fuer jede Instanz eines CommObject kann die Abspeicherung der Daten eines InformationReport aktiviert werden. Dies geschieht durch Auswahl des entsprechenden PopUp-Menus auf dem Instanzbaum und der anschließenden Auswahl einer passenden Konfigurationsdatei.

Aufbau der Konfigurationsdatei

Leerzeilen werden übersprungen. Die erste (nicht leere) Zeile muss den Dateinamen der Zieldatei enthalten. Diese wird sofort nach dem Einlesen geöffnet.
Alle weiteren Zeilen beschreiben die Datenstruktur des CommObjekt und was mit den einzelnen Datenelementen geschehen soll.
Die Beschreibung muss alle Elemente, von der obersten Struktur an bis zum letzten Element, von dem Informationen abgespeichert werden sollen, enthalten.
Datenelemente am Ende, von denen keine Informationen abgespeichert werden sollen, können und sollten weggelassen werden. Das beschleunigt auf jeden Fall den Ablauf.

Jede dieser Formatierungszeilen besteht aus 3 Teilen, dem Level, dem Namen mit Überspring-Marker sowie dem eigentlichen, optionalen Formatierungsstring. Diese Teile muessen durch Tab oder Leerzeichen getrennt sein. Führende und folgende Leerezeichen bzw. Tabs sowie die Anzahl dieser zwischen den einzelnen Teilen sind egal.

Angabe des Levels

Derzeit erfolgt (noch) keine Gegenprüfung gegen die DCD. Es wird jedoch der "Einrückungslevel" der Datentypen benötigt. Dieser als erstes angegebende Level ist eine (positive) Zahl die mit zunehmender Einrückungstiefe(Datetyptiefe) ansteigen muss. Alle Elemente des gleichen Levels (an einer bestimmten Stelle im Baum der Datenstruktur) müssen die gleiche Levelangabe tragen. Es sind Lücken in der Nummerierung erlaubt, die Zahlenfolge muss lediglich aufsteigend sein (z.B. 1, 3, 122,123,124).

Ueberspringen und Elementname

Der zweite Teil ist der Name (Elementname der Struktur bzw. Name des CommObject selbst oder Name des Union-Zweig usw.) des Datenelelements. Derzeit erfolgt keine Gegenprüfung gegen die DCD, so dass auch falsche Namen akzeptiert werden. Als Folge dessen dürfen keine Member vergessen oder hinzugefuegt werden, es erfolgt keine Resynchronisation zu den Namen. Es wird empfohlen die korrekten Namen zu verwenden, dass erleichtert die Eingabe und erzeugt Kompatibilitaet zu späteren Versionen die gegebenenfalls zur DCD gegenprüfen.

Die Elemente von Sequenzen und Arrays haben keine eigenen Namen, es ist dafür der Name der Sequenz bzw. des Arrays zu wiederholen.

Wird dem Namen ein '@' vorangestellt, so wird das entsprechende Element übersprungen. Das heißt, dass der Formatierungsstring nicht ausgewertet wird und keine Subelemente (tiefere Level) verarbeitet werden.
ES IST ZU BEACHTEN, dass keine Subelemente mit angegeben werden dürfen wenn z.B. eine Struktur übersprungen wird. Diese würden als Formatierung der Elemente nach dem übersprungenen Datenelelement (z.B. der Struktur) missinterpretiert werden!

Ein nicht übersprungener Datentyp kann auch ohne Formatierungscodes angegeben werden. In diesem Falle erfolgt zwar keinerlei Abspeicherung von Information, es werden jedoch die Sub-

Elemente verarbeitet (und müssen angegeben werden). Es wird jedoch empfohlen Zweige der Datentypen, die keine Ausgaben enthalten sollen, mit hilfe der Überspring-Markierung abzuschneiden. Das beschleunigt die Verarbeitung.

Formatierungscodes

Der dritte Teil beinhaltet einen ununterbrochenen String aus meist einbuchstabigen Formatierungscodes. Welche Codes sinnvolle Ergebnisse liefern hängt vom Datentyp des Datenelelemnts ab. Die angegebenen Codes werden von links nach rechts abgearbeitet. Es können bestimmte Ausgaben auch mehrfach erfolgen (3x als Dezimalzahl z.B. - wems gefällt). Nach jedem abgearbeitetn Datenelement wird der Inhalt in die Datei geschrieben (fflush()). Die folgenden Abschnitte beschreiben die implementierten (oder vorgesehenen) Codes.

Elementare Datentypen

- 'B' (Binary) Die Daten werden in der entsprechenden Laenge und in der übertragenen Byteorder binaer in die Datei geschrieben.
- 'C' (Cast) Änderung des Datentyps, siehe unten. Benoetigt 3 weitere Zeichen!
- 'D' (Decimal) Ausgabe des Dateninhalts als Dezimalzahl. Boolean werden dabei als unsigned char und Enums als short interpretiert.
- 'E' (Enum) (Noch nicht implementiert) Ausgabe des Namens eines Enumerationelements bzw. true/false bei Boolean. Fuer alle anderen Datentypen erfolgt keine Ausgabe.
- 'H' (Hexadecimal) Ausgabe des Dateninhalts als Hexadezimalzahl. Dabei wird auch der Inhalt sowohl negativer als auch von Gleitkommazahlen als ganzzahliger Hexadezimalwert ausgegeben. Der Zahl wird nicht automatisch ein "0x" vorangestellt!
- 'I' (Index) Es wird der Index in einem Array als positive Dezimalzahl ausgegeben. Auf allen Datentypen außer Array und Sequenz ist dieser Wert immer 0.
- 'K' (Komma) Es wird ein Komma eingefuegt.
- 'N' (Name) Es wird der Typname des Datenelements eingefuegt.
- 'P' (Punkt) Es wird ein Punkt eingefuegt.
- 'R' (Return) Es wird ein Enterzeichen ('\n') eingefuegt.
- 'S' (Space) Es wird ein Leerzeichen ausgegeben.
- 'T' (Tab) Es wird ein Tabulatorzeichen ausgegeben.
- 'X' (0x) Es wird die Zeichenfolge "0x" ausgegeben.
- 'Y' (Zero) Es wird eine binaere 0 (ein Byte) ausgegeben.
- 'Z' (Zeichen) Das diesem Formatierungzeichen folgende Zeichen wird ausgegeben. Es ist zu beachten, dass derzeit der gesamte Formatierungsstring in Uppercase gewandelt wird und somit auch diese Zeichen.

Casts

Das Formatierungszeichen 'C' loest einen Cast aus. Dieser wurde aufgrund des MDAQ Companion eingefuehrt. Dieser übertraegt seine Messdaten als Bytesequenz. Der Inhalt dieser einzelnen Bytes ist nicht ohne weiteres Interpretierbar. Es werden immer mehrere Bytes fuer ein Datum benötigt. Der hier gewählte Ansatz ist jedoch sehr allgemein gedacht und kann nicht alle Moeglichkeiten des MDAQ Companion abdecken. Er sollte aber ausreichend sein um grundlegende Tests und Datenanalysen durchfuehren zu koennen.

Der "Cast" beruht auf der Angabe eines Zieldatentyps und der zugehörigen Byte(Element)order. Alle Formatierungskommandos im Formatstring, die hinter dem Cast stehen, werden auf den "gecasteten" Wert angewendet. Sie werden nur dann abgearbeitet, wenn der aktuelle Index ein vielfaches des Zieldatentyps (einsprechend der Groesse des Quell und Zieltyps) darstellt. z.B. Char→Long bei jedem 4. Element, also auf Index 3,7,11 usw. oder Short→Float bei jedem 2.Element, also Index 1,3,5 usw. Dadurch ist sichergestellt, dass kein Zugriff auf nicht vorhandenen Speicher erfolgt, wenn z.B. die Laenge des zu castenden Arrays nicht einem Vielfachen der

Typenverhältnisses entspricht. Die letzten Elemente werden dabei jedoch nicht erfasst.

Prinzipiell kann ein Cast auch auf Datenelemente angewendet werden, die nicht Teil eines Array oder Sequenz sind. Mit der Ausnahme von Downcasts wird jedoch keine Ausgabe erfolgen.

Downcasts sind prinzipiell moeglich. Damit sind Casts gemeint, die entweder Casts zum eigenen(quell) Typ bzw. signed/unsigned Aenderung des Typs oder Casts zu kleineren Typen darstellen. In letzterem Fall wird das Quellelement jedoch nur einmal im Zieltyp abgebildet. Der Dateninhalt wird von rechts genommen (unabhängig von der Byteorder. z.B. 31 als Intel-Short dargestellt: 0x1f00 → Cast zu Char → von Rechts genommener Wert = 0x00. Ein Downcast von Float wird keine sinnvollen Ergebnisse liefern und ein Versuch einen Downcast von Double auszulösen wird mit einem Fehler quittiert.

Casts sind jeweils auf (U)Char, (U)Short, (U)Long, Float und Double möglich. Boolean und Enum vierlieren ihre Bedeutung (zuordnung der Datentypen-Objekte nicht moeglich). IFREF kann prinzipiell als Sondertyp nicht gecastet werden. Der Datentyp-Name ist dabei der Typname des Elements auf dem der Cast aktiv wurde (sollte ja im Array bei allen Elementen gleich sein) und stimmt somit nicht mehr mit dem Zieltyp ueberein!

Der Index der gecasteten Elements wird entsprechend der neuen Typbreite angepasst. z.B. Index 11 eines Char Arrays aendert sich nach Cast in Long auf 2.

Die Anpassung der Order bezieht sich nicht auf die Bytes des Quelltyps sondern auf die Anordnung der Elemente. Die Order von Bytes innerhalb des Quelltyps (z.B. Short) bleibt dabei unverändert. Das ist unbedingt zu beachten.

Es ist möglich, Casts hintereinander zu legen. Der Index wird dabei weiter heruntergebrochen. Es ist jedoch zu beachten, dass als Quelle fuer die Daten der unveränderte Datenbereich des darunter liegenden, ursprünglichen Arrays benutzt wird. Das hat Einfluss auf die Byte(Element)order.

Kennung Vorzeichen (1.Zusatzzeichen)

- 'U' (Unsigned) Vorzeichenloser Datentyp.
- 'S' (Signed) Vorzeichenbehafteter Datentyp. Auf diesen Kennbuchstaben wird NICHT geprüft. Alles was nicht 'U' ist gilt als 'S'.

Kennung Typbreite (2.Zusatzzeichen)

Zieltyp des Casts.

- 'L' (Long)
- 'S' (Short)
- 'C' (Char)
- 'F' (Float)
- 'D' (Double)

Kennung Elementorder (3.Zusatzzeichen)

- 'I' (Intel - LSB first) Elementorder LSB first.
- 'M' (Motorola - MSB first) Elementorder MSB first. Auf diesen Kennbuchstaben wird NICHT geprüft. Alles was nicht 'I' ist gilt als 'M'.

IFREF

Der Typ IFREF ist derzeit noch nicht implementiert. Angegebene Formatierungen werden übergangen.

Prinzipiell ist nicht angedacht Instanznamen zu dekodieren, das würde sehr viel Zeit kosten. Zukünftige Versionen sollen aber die einzelnen Elemente von IFREF als Zahlen ausgeben können. Bitte um Rückmeldung fuer abweichende Wünsche (die halbwegs begründet sind).

Komplexe Datentypen

Komplexe Typen (Struktur, Union, Array, Sequenz) besitzen spezielle, eigene Formatierungszeichen. Mit ihnen sollen die speziellen Eigenschaften der Typen ausgegeben werden können.

- 'R' (Return) Noch nicht implementiert. Es wird ein Enterzeichen (\n) ausgegeben.
- 'A' (Afterbreak) Noch nicht implementiert. Diese Formatierung soll ein Enterzeichen nach Abarbeitung des komplexen Typs einfügen. Das ist aber nur zwingend erforderlich wenn z.B. der Typ aus einem einzelnen Array besteht. Alternativ kann man aber auch ein Enter vorher abgesetzt werden.

Union

Die Verarbeitung einer Union zeigt einige Besonderheiten die im Folgenden erläutert werden sollen:

- Der Selektor muss mit der Bezeichnung "Selector" in der Elementliste angegeben werden. Er kann mit den ganz normalen Formatierungszeichen fuer elementare Typen ausgegeben werden.
- Es müssen grundsätzlich alle Zweige einer Union angegeben werden (ausser die ganze Union wird übersprungen). Es wird jedoch nur der Unionzweig in die Ausgebae ubertragen, der zum aktuellen Selektor passt.
- Es koennen einzelne Unionzweige zum Überspringen markiert werden. Sind diese aktiv werden keinerlei Informationen ausgegeben.
- Selektor und Unionzweige befinden sich auf dem gleiche Level.
- Die Union wird auch korrekt ausgegeben, wenn der Selektor zum überspringen markiert ist, das hat keinen Einfluss auf die interne Verarbeitung, lediglich auf die Ausgabe des Selektors selbst.

Was geht nicht

- Querbeziehungen zu anderen Elementen der DCD (z.B. uMeasurementValueTypeDesc im MDAQ) können nicht hergestellt werden. Das würde eine Spezialbehandlung fuer bestimmte DCDs darstellen. Die entsprechend nötige Semantik ist nicht in der ASAM DCD enthalten.
- Variable Casts z.B. abwechselnde Übertragung von Long- und Short-werten ist nicht möglich. Entweder ein Array wird gecastet oder nicht.
- Casts von komplexen Typen sind nicht vorgesehen.
- Casts dürfen keine Lücken enthalten. So ist es z.B. nicht möglich alle 5Char einen Long zu extrahieren.
- Verkämmen von Datenelementen (z.B. EndOffsets und Messdaten des MDAQ Companion).

Beispiel1 MDAQ MeasurementValues)

Die folgende Definition bewirkt die Ausgabe der Messdaten mit Enter nach jedem empfangenen Block als Hex unsigned long Werte, interpretiert als Intel mit Tabs dazwischen (0x00000001 0x00000002 Enter…).

<Dateiname>

1 MeasurementValues
2 @Strukturmember
2 Data
3 auchValueData A
4 auchValueData XHCULIT
3 @Skip

Beispiel2 MDAQ MeasurementValues)

Ausgabe der Endoffsets Dezimal und Hexadezimal. Ausgabe der Messdaten gecastet in unsigned long - LSB first dezimal. Jeweils mit nachfolgendem Enter.

D:\Project\Temp\TestIRData.txt

1 MeasurementValue
2 @uTransferStatus
2 Data
3 auchValueData
4 aucValueData CULIDR
3 @aulValueEndOffsets
4 @aulValueEndOffsets DSHR