

Public Support - Support Request #3810

DDL struct alignment in DDL3.0

2018-08-21 10:51 - hidden

Status:	Closed	Product Issue Numbers: Affected Products: ADTF 2.14.2 Platform: Windows 7 64bit Topic: ADTF::DDL FAQ Links:
Priority:	Normal	
Category:		
Customer:	AUDI	
Department:	EF	
Requester's Priority:	Normal	
Support Level:	2nd Level	
Resolution:	Solved Issue	

Description

Support Anfrage:

Ich habe zwei Fragen zum SDK-Artikel "DDL Definition File Format". Das Alignment aus <struct> beeinflusst in DDL 2.x **nur** das Alignment in Arrays, wie ich lese.

1. Gilt das nur für Arrays mit arraysize > 1? Oder wird auch das erste Element anhand des <struct>-Alignments ausgerichtet?
2. Betrifft das <struct>-Alignment in **DDL 3.0** weiterhin nur Arrays? Wirkt sich das <struct>-Alignment von 8 in der folgenden DDL 3.0 auf die Größe aus? Oder nicht (wie beobachtet), da die <struct> nicht von einem Element referenziert wird? Die tatsächliche Größe im DAT-File ist 43 bytes gegenüber den von mir vermuteten 48 bytes.

DDL 3.0:

```
<struct alignment="8" name="t_CAN_PCR_01_CAN_PCR_01" version="1">
  <element alignment="8" arraysize="1" byteorder="LE" byteword="0" name="ESP_v_ref" type="tFloat64" />
  <element alignment="1" arraysize="1" byteorder="LE" byteword="8" name="ESP_v_ref_Fahrtrichtung" type="tUInt8" />
  <element alignment="1" arraysize="1" byteorder="LE" byteword="9" name="PCR_Sensorstatus" type="tBool" />
  <element alignment="8" arraysize="1" byteorder="LE" byteword="16" name="PCR_Obj_vx_rel" type="tFloat64" />
  <element alignment="8" arraysize="1" byteorder="LE" byteword="24" name="PCR_Obj_vy_rel" type="tFloat64" />
  <element alignment="8" arraysize="1" byteorder="LE" byteword="32" name="PCR_Obj_TTC" type="tFloat64" />
  <element alignment="1" arraysize="1" byteorder="LE" byteword="40" name="PCR_Obj_Guete" type="tUInt8" />
  <element alignment="1" arraysize="1" byteorder="LE" byteword="41" name="PCR_Crashwahrscheinlichkeit" type="tUInt8" />
  <element alignment="1" arraysize="1" byteorder="LE" byteword="42" name="PCR_Sensorstatus_02" type="tUInt8" />
</struct>
<stream description="streamid_5" name="TargetPCC" type="adtf.core.media_type">
  <struct byteword="0" name="t_CAN_PCR_01_CAN_PCR_01" type="t_CAN_PCR_01_CAN_PCR_01" />
</stream>
```

Lösung:

zu Frage 1.:

Die Position wird immer anhand des Alignments ausgerichtet. In der 2.0 hat der Coder nur den Fehler gemacht, dass er bei Arrays das letzte Element nicht auf seine richtige Größe gepadded hat (wenn das Alignment das gefordert hätte), heißt die Gesamtgröße eine Arrays (auch wenn's nur 1 Element hatte) war nicht notwendigerweise ein Vielfaches des Alignments, die Positionen der Elemente passten aber zum Alignment.

zu Frage 2.:

Hier bist du wie viele andere über die Unterscheidung zwischen serialisierter und deserialisierter Beschreibung gestolpert. Die DDL beschreibt zwei Fassungen in einem :-):

deserialisiert: Positionen und Größen werden auf Basis des Alignments und Typgröße berechnet (alignment + type attribute, byte order ist immer nativ)

serialisiert: Positionen sind durch das bytepos Attribut festgelegt. (bytepos + type + byteorder attribute)

Der Recorder serialisiert die Daten im Speicher der Media Samples mit Hilfe der deserialisierten Beschreibung in die serialisierte Fassung. Daher landen in deinem Beispiel nur 43 Byte auf der Platte.

Details siehe die Kommentare: [#3810-5](#); [#3810-8](#); [#3810-11](#)

History

#1 - 2018-08-21 13:09 - hidden

- Status changed from New to In Progress

- Topic set to ADTF::DDL

Hallo Patrick,

ich kläre gerade wer hier aufgrund der ganzen Abwesenheits-/Urlaubssituation der beste Ansprechpartner ist. Nur für das Ticket/Support grundsätzlich: Auf welche ADTF Version und welches OS bezieht sich die Anfrage?

Gruß
Matthias

#2 - 2018-08-21 13:17 - hidden

Hallo Matthias,

Windows 7 x64, ADTF 2.14

Gruß
Patrick

#3 - 2018-08-21 13:19 - hidden

- Affected Products ADTF 2.14.2 added

- Platform Windows 7 64bit added

#4 - 2018-08-21 16:06 - hidden

Hallo Martin,

kannst Du dir die Anfrage von Patrick bitte einmal ansehen!?

Danke und Gruß
Matthias

#5 - 2018-08-22 09:30 - hidden

Hi Patrick,

zu 1.: Die Position wird immer anhand des Alignments ausgerichtet. In der 2.0 hat der Coder nur den Fehler gemacht, dass er bei Arrays das letzte Element nicht auf seine richtige Größe gepadded hat (wenn das Alignment das gefordert hätte), heißt die Gesamtgröße eine Arrays (auch wenn's nur 1 Element hatte) war nicht notwendigerweise ein Vielfaches des Alignments, die Positionen der Elemente passten aber zum Alignment.

zu 2.: hier bist du wie viele andere über die Unterscheidung zwischen serialisierter und deserialisierter Beschreibung gestolpert. Die DDL beschreibt zwei Fassungen in einem :-):

- deserialisiert: Positionen und Größen werden auf Basis des Alignments und Typgröße berechnet (alignment + type attribute, byte order ist immer nativ)
- serialisiert: Positionen sind durch das bytepos Attribut festgelegt. (bytepos + type + byteorder attribute)

Der Recorder serialisiert die Daten im Speicher der Media Samples mit Hilfe der deserialisierten Beschreibung in die serialisierte Fassung. Daher landen in deinem Beispiel nur 43 Byte auf der Platte.

Du kannst ja mit folgendem Beispielcode experimentieren, dann siehst du gleich was passiert wenn man was ändert.

```
{
```

```

    cString strDesc = "<struct alignment=\"8\" name=\"t_CAN_PCR_01_CAN_PCR_01\" version=\"1\">"
        "<element alignment=\"8\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"0\" name=\""
ESP_v_ref\" type=\"tFloat64\" />"
        "<element alignment=\"1\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"8\" name=\""
ESP_v_ref_Fahrtrichtung\" type=\"tUInt8\" />"
        "<element alignment=\"1\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"9\" name=\""
PCR_Sensorstatus\" type=\"tBool\" />"
        "<element alignment=\"8\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"16\" name=\""
PCR_Obj_vx_rel\" type=\"tFloat64\" />"
        "<element alignment=\"8\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"24\" name=\""
PCR_Obj_vy_rel\" type=\"tFloat64\" />"
        "<element alignment=\"8\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"32\" name=\""
PCR_Obj_TTC\" type=\"tFloat64\" />"
        "<element alignment=\"1\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"40\" name=\""
PCR_Obj_Guete\" type=\"tUInt8\" />"
        "<element alignment=\"1\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"41\" name=\""
PCR_Crashwahrscheinlichkeit\" type=\"tUInt8\" />"
        "<element alignment=\"1\" arraysize=\"1\" byteorder=\"LE\" bytupos=\"42\" name=\""
PCR_Sensorstatus_02\" type=\"tUInt8\" />"
    "</struct>";

    cMediaCoder oCoder;
    oCoder.Create("t_CAN_PCR_01_CAN_PCR_01", strDesc, 0);
    cObjectPtr<IMediaSerializer> pSerializer;
    oCoder.GetSerializer(&pSerializer);
    LOG_INFO(cString::Format("%d, %d", pSerializer->GetDeserializedSize(), pSerializer->GetSerializedSize()));
}

```

->

48, 43

Grüße,

Martin

#6 - 2018-08-22 13:35 - hidden

Danke dir für deine Erläuterungen.

zu 2. Da gibt es die übersichtliche Grafik in der SDK-Doku "ADTF serialization via Media Description out of the DDL". Also vom DAT-File aus während des Zurücklesens der Samples betrachtet, benötigt man nur die Bytepos-Info, **niemals** das Alignment, richtig? Das Element- und Struct-Alignment ist nur wichtig beim Hin- und Herschieben von Samples zwischen den Filtern zur Laufzeit von ADTF, korrekt?

#7 - 2018-08-22 13:57 - hidden

Hallo Martin,
Patrick hat noch zwei Rückfragen.
Danke und Gruß
Matthias

#8 - 2018-08-23 09:08 - hidden

Hi Patrick,

was willst Du denn machen? Für mich hört sich das so an, dass du selbst aus der DAT Datei lesen willst, richtig? Warum nimmst Du nicht einfach die adtf_file Library, die das alles transparent für dich erledigt?

Beim Lesen braucht man in der Tat nur die bytupos. Wichtig: der Recorder verwendet die DDL aber nur dann zum Serialisieren, wenn der Media Type = 0/0 ist, ansonsten greift die Serialisierung der jeweiligen Media Sample Implementierung.

Grüße,

Martin

#9 - 2018-08-23 10:59 - hidden

Hallo Martin,

Für mich hört sich das so an, dass du selbst aus der DAT Datei lesen willst, richtig?

Genau. Hier arbeiten einige Leute am MDM-Datenmanagement in der AWS-Cloud. DAT-Files werden dort parallelisiert mittels Apache Spark und mit

der funktionalen Sprache Scala gelesen und verarbeitet.

wenn der Media Type = 0/0 ist, ansonsten greift die Serialisierung der jeweiligen Media Sample Implementierung

Ich habe bspw. Laserscanner-Daten mit benutzerspezifischem Media-Type 0x11901, die mit der folgenden Description (DDL 3.0) kommen (Ausschnitt):

```
<struct alignment="4" name="tScanPointList" version="0">
  <element alignment="1" arraysize="1" byteorder="LE" bytepos="0" name="bLocalTime" type="tBool" />
  <element alignment="4" arraysize="1" byteorder="LE" bytepos="1" name="nTimestamp_sec" type="tUInt32" />
  <element alignment="4" arraysize="1" byteorder="LE" bytepos="5" name="nTimestamp_fractions" type="tUInt32" />
  <element alignment="2" arraysize="1" byteorder="LE" bytepos="9" name="nNumSubScans" type="tUInt16" />
  <element alignment="4" arraysize="16" byteorder="LE" bytepos="11" name="sSubScan" type="tSubScan" />
</struct>
</structs>
<streams>
  <stream description="streamid_1" name="scan_points_x2310" type="adtf.core.media_type">
    <struct bytepos="0" name="tScanPointList" type="tScanPointList" />
  </stream>
</streams>
```

Beim Playback in ADTF 2.14 scheint Bytepos keine Auswirkung zu haben, egal welchen Wert ich dort mit dem Originalwert ersetze. Soweit, so gut da Media-Type != 0/0. Tatsächlich hat aber das Manipulieren der Alignment-Werte Auswirkungen, wie ich im Signal-View anhand falscher Signalwerte erkennen kann. Die Samples im DAT-File sind tatsächlich aligned.

Kann man sagen, bei Media-Type 0/0 zählt Bytepos, bei Media-Type != 0/0 zählt Alignment?

#11 - 2018-08-23 11:14 - hidden

Nochmals die Frage, warum nicht die adtf_file Library verwendet wird? Die ist Open Source, unterstützt DAT Files von ADTF 2 und ADTF 3. Was braucht ihr, was die nicht kann, und wäre der Aufwand nicht besser dort investiert?

Kann man sagen, bei Media-Type 0/0 zählt Bytepos, bei Media-Type != 0/0 zählt Alignment?

Nein, es zählt die Implementierung der Serialisierung in der Media Sample Implementierung, die kann machen was sie will (z.B. bei CAN/Flexray etc.). Genau diese Serialisierungen gibts in der adtf_file Library, es macht keinen Sinn dass Ihr die nachimplementiert!

Natürlich hat das Verändern der Alignments eine Auswirkung in ADTF, da der Media Description Signal Provider versucht die Daten in Samples anhand der Beschreibung (für die deserialisierte Fassung) zu interpretieren. Wenn man die Alignments ändert passt die Beschreibung nicht mehr zur Representation im Sample.

Grüße,

Martin

#12 - 2018-08-23 15:53 - hidden

Nochmals die Frage, warum nicht die adtf_file Library verwendet wird?

Anforderungen für die Lib wäre ein JVM/Scala-Port und die Fähigkeit, DAT-Dateien verteilt verarbeiten zu können. Das DAT-File-Streamen läuft parallelisiert und im Cluster verteilt. Es wird in der gleichen JVM wie Spark verarbeitet, es gibt daher keine Interprozess-Kommunikation, und soll es auch nicht geben. Die Performance ist daher sehr hoch. Die Analyse läuft im Anschluß ebenfalls über parallelisierte Scala-Skripte, ohne Datei-Hin- und Hergeschiebe.

Ich konnte jetzt mit einem Test-Filter nachvollziehen, daß Media-Type 0/0 mittels Bytepos auf der Platte landet. Wenn ich Media-Type zufällig gewählt x/y einstelle, wird das Alignment scheinbar genutzt. Es gibt hierbei keine spezielle Media-Type-Implementierung:

```
// landet mittels DDL-Bytepos auf der Platte
RETURN_IF_FAILED(info.pin->Create(name, new cMediaType(0, 0, 0, name, ddl, IMediaDescription::MDF_DDL030000), this));

// landet mittels DDL-Alignment auf der Platte
RETURN_IF_FAILED(info.pin->Create(name, new cMediaType(0x1f75, 3, 0, name, ddl, IMediaDescription::MDF_DDL030000), this));
```

Das mit der Media-Type-Abhängigkeit war mir zuvor nicht klar. Vielen Dank!

#13 - 2018-08-24 09:20 - hidden

Hallo Patrick,

Ist diese Anfrage damit abgeschlossen?
Können wir das Ticket schließen?

#14 - 2018-08-24 09:20 - hidden

- Status changed from *In Progress* to *Customer Feedback Required*

#15 - 2018-08-24 10:08 - hidden

Ja, kann geschlossen werden. Danke!

#16 - 2018-08-24 10:30 - hidden

- Subject changed from *DDL struct alignment* to *DDL struct alignment in DDL3.0*

- Description updated

- Status changed from *Customer Feedback Required* to *To Be Closed*

- Resolution set to *Solved Issue*

#17 - 2018-09-11 15:29 - hidden

- Project changed from *11* to *Public Support*

- Status changed from *To Be Closed* to *Closed*

#18 - 2018-09-11 17:06 - hidden

- Private changed from *Yes* to *No*