Public Support - Support Request #6259

Read out wrong size of CAN data

2019-02-21 16:43 - hidden

Status:	Closed		
Priority:	Normal		
Category:			
Customer:	AUDI	**Product Issue Numbers:**	
Department:	TKI	**Affected Products:**	ADTF Streaming Library 2.9.0
Requester's Priority:	Normal	**Platform:**	Windows 10 64bit
Support Level:	2nd Level	**Topic:**	StreamingLib::Reader
Resolution:	Solved Issue	**FAQ Links:**	

Description

Hallo,

ich möchte gern mit adtf-streaming lib flexray und Can Daten aus .dat lesen.

Bei der benutzung des Beispiels candump.dat auf dem Trace erzeugt vom Beispiel canwritter, habe ich die can Daten mit ID und 8 bytes richtig bekommen.

Aber beim Traces von ADTF bekomme ich CAN Daten mit mehr als 8 Bytes, und ID passen auch nicht?? pCanMessage->ui8Length (siehe code unter) ist oft mehr als 32 bytes

```
if (pDataBlock->GetStreamId() == nCanStream)
{
    nMessageCounter++;

    //get the time the block was received as a sample
    tTimeStamp tsReceived = pDataBlock->GetTime();

    //get data from datablock
    tCanMessage *pCanMessage = NULL;

    //get data from block
    pDataBlock->GetData((const tVoid**)&pCanMessage);

    //show can message
    cout << "#" << nMessageCounter << " -- " << tsReceived << " : C" << (int)pCanMessage->
ui8Channel << " ID=" << hex << pCanMessage->ui16Id << dec << "[";

    for (int nCounter = 0; nCounter < pCanMessage->ui8Length; nCounter++)
    {
        cout << " " << hex << (int)pCanMessage->pui8Data[nCounter] << dec;
    }

    cout << "]" << endl;
}
```

Lösung

Ich denke du verwendest die falsche Datenstruktur. In ADTF wird schon lange nicht mehr tCanMessage verwendet sondern tCANData, die ist in der Streaming Library als tADTFCANData verfügbar. Welche CAN Struktur im Sample enthalten ist kannst du über die Größe der Daten ermitteln:

```
const tVoid* pBlockData;
tInt64 nBlockSize = pDataBlock->GetData(&pBlockData);

if (nBlockSize == sizeof(tADTFCANData))
{
```

```
    const tADTFCANData* pCanData = reinterpret_cast<const tADTFCANData*>(pBlockData);
    ...
}
else if (nBlockSize == sizeof(tCANMessage))
{
    const tCANMessage* pCanMessage = reinterpret_cast<const tCANMessage*>(pBlockData);
    ...
}
...
```

History

#1 - 2019-02-22 09:40 - hidden

- Project changed from Public Support to 11

- Status changed from New to In Progress

- Topic set to StreamingLib::Reader

#2 - 2019-02-22 09:41 - hidden

- Status changed from In Progress to Customer Feedback Required

Hallo Cedric,

kann es sein, dass es sich nicht um CAN, sondern CAN-FD Daten handelt, wenn du mehr Bytes bekommst ?
Was sagen die Traces in ADTF ?
Hast du eine Beispiel-Datei und dazu passende DBC für uns ?

CAN-FD wird so nicht in ADTF 2.x (und damit Streaming Library) unterstützt.

#3 - 2019-02-22 10:12 - hidden

- File can_trace.dat added

- File adtf_streaming_can.JPG added

Hallo Florian,

danke für die schnelle Anwort.

Ich kann dir eine dummy trace mit 2 Channele schicken, wo ich sicher bin, dass es kein FD Daten gibt. Mit ADTF kann ich die Signale lesen.
Screenshot von den Ergebnissen habe ich auch angehängt.
dbc wäre KonfortCan von Audi. Ich darf nicht weiter schicken.

Vielen Dank

#4 - 2019-02-25 17:21 - hidden

Hallo Florian,

konntest du was anfangen, mit was ich dir geschickt (trace + screenshot) habe?

Dnake,
Grüße,
Cédric

#5 - 2019-02-26 08:40 - hidden

- Description updated

#6 - 2019-02-26 09:00 - hidden

- Status changed from Customer Feedback Required to In Progress

@Martin: Hast du noch eine Idee ? Schneidet der Trace View ggf. nach 8 Byte ab und es sind doch CAN FD Daten ?
Oder berücksichtigt der Trace View ggf. ein "New Line" (die CAN Nutzdaten an sich können ja länger sein als 8 Byte, müssen dann eben immer aufgeteilt werden)

#7 - 2019-02-26 10:45 - hidden

Hi Cedric,

ich denke du verwendest die falsche Datenstruktur. In ADTF wird schon lange nicht mehr tCanMessage verwendet sondern tCANData, die ist in der Streaming Library als tADTFCANData verfügbar. Welche CAN Struktur im Sample enthalten ist kannst du über die Größe der Daten ermitteln:

```
const tVoid* pBlockData;
tInt64 nBlockSize = pDataBlock->GetData(&pBlockData);

if (nBlockSize == sizeof(tADTFCANData))
{
    const tADTFCANData* pCanData = reinterpret_cast<const tADTFCANData*>(pBlockData);
    ...
}
else if (nBlockSize == sizeof(tCANMessage))
{
    const tCANMessage* pCanMessage = reinterpret_cast<const tCANMessage*>(pBlockData);
    ...
}
...
```

Grüße,

Martin

#9 - 2019-02-26 10:53 - hidden

Hallo Martin, Florian,

danke für die Antwort. Ich habe die Canstruct.h benutzt vom Beispiel (candump, canwriter). Ich probiere mit tADTFCANData, und gebe Rückmeldung.

Grüße,
Cédric

#10 - 2019-02-26 13:03 - hidden

Hallo zusammen,

ich bekomme jetzt meine 8 bytes. Es funktionniert.

noch eine Frage dazu, um das gleiche zu machen mit Flexray Daten welche typ muss ich benutzen? Im adtf_streaminglib_types gibt es kein Flexraydatatype. kann ich einfach
den flexray_frame.h file vom ADTF-device-toolbox nehmen?

Danke
Grüße,
Cédric

#11 - 2019-02-26 13:35 - hidden

- Status changed from In Progress to To Be Closed

- Resolution set to Solved Issue

- Affected Products ADTF Streaming Library 2.9.0 added

- Platform Windows 10 64bit added

Hi,

ja das geht.

Grüße,

Martin

#12 - 2019-02-26 13:36 - hidden

danke,
der Ticket kann geschlossen werden.

Grüße,
Cédric

#13 - 2019-02-26 15:50 - hidden

- Project changed from 11 to Public Support

- Subject changed from adtf-streaming can/flexray lesen to Read out wrong size of CAN data

- Description updated

- Private changed from Yes to No

#14 - 2019-03-08 13:35 - hidden

- Status changed from To Be Closed to Closed

Files

can_trace.dat	3.95 MB	2019-02-22	hidden
adtf_streaming_can.JPG	322 KB	2019-02-22	hidden