Public Support - Support Request #7789

StreamTime acting in a Playback Config

2019-07-11 11:30 - hidden

Status:	Closed		
Priority:	Normal		
Category:			
Customer:	BOSCH	**Product Issue Numbers:**	https://www.cip.audi.de/jira/browse/ACORE-10107
Department:	CC-DA/ESU	**Affected Products:**	ADTF 2.13.3
Requester's Priority:	Normal	**Platform:**	Windows 7 64bit
Support Level:	3rd Level	**Topic:**	ADTF::DAT
Resolution:	No Customer Feedback	**FAQ Links:**	

Description

Support Anfrage:

Ich habe eine Config mit einem HD-Player und einem Filter, in dem ein cKernelTimer mit 10 ms timeout läuft.
In der Timer-Methode wird geprüft, wie viel Zeit zwischen dem aktuellen und dem letzten Aufruf vergangen ist.
Da kommt öfters mal 0 als Differenz raus, was für den darin laufenden Algo ein Problem darstellt.

Das DAT-File beinhaltet mehrere Streams, die schnellste Update-Rate eines Streams beträgt im Mittel 10 ms.
Der HD-Player aktualisiert die StreamTime nur, wenn er ein MediaSample verschickt. (Richtig?)

Die Datenrate der Streams schwankt aber leicht.
Wenn zwischen zwei Updates nur 9 ms vergangen sind, reicht das natürlich nicht um den cKernelTimer erneut zu triggern.
Danach kommt dann z.B. ein neues Update 11 ms später, die StreamTime wird direkt um 20 ms erhöht und der cKernelTimer direkt zweimal hintereinander getriggert. Zweimal mit der gleichen StreamTime.

Gibt es eine vernünftige Lösung, dass der cKernelTimer trotzdem zwischendrin getriggert wird?

Ich hatte mal aus einem anderen DAT-File einen Stream mit höherer UpdateRate reingemerged. Das löst das Problem, ich betrachte das allerdings nicht als 'vernünftige Lösung' ;)

Lösung:

Die Stream Time wird diskret vor jedem verschickten Sample erhöht. Der Kernel wird synchron über diesen Zeitsprung informiert und überprüft dann welche Timer in der Zeitspanne hätten laufen sollen. Leider hat er dafür nur den letzen und den neuen Zeitstempel zur Verfügung. Er kann also leider nur die Anzahl der notwendigen Timer-Aufrufe feststellen. Selbst wenn er dabei die Zeit interpolieren würde, kann er die Stream Time nicht mehr in die Vergangenheit zurücksetzen, damit ein pClock->GetStreamTime() im Anwender Code den passenden Zeitstempel liefert.

Also ja, die Genauigkeit, der Timer hängt von der maximalen Datenrate im DAT-File ab. (Das ist ein Punkt der in ADTF 3 besser gelöst wird, da hier auch die Trigger von Verarbeitungsketten mit aufgezeichnet werden).

Leider bleiben nur die zwei Möglichkeiten:

- zum einen den Algorithmus so anzupassen, dass eine Zeitdauer zwischen den Aufrufen von 0 kein Problem darstellt (das könnte theoretisch auch im Live-Betrieb auftreten wenn der OS-Kernel durch die Last nicht nachkommt)
- so wie beschrieben höherfrequente Datenströme mit aufzeichnen/einfügen. Da oft ein CAN oder Flexray Bus mit aufgezeichnet wird, kommt das Problem in anderen Use-Cases nicht so sehr zum tragen.

Zusätzlich wurde das Produktticket ACORE-10107 Playback: Improve Timer Handling erstellt.
Siehe Kommentar 14

History

#1 - 2019-07-12 09:28 - hidden

- Project changed from Public Support to 5

- Status changed from New to Customer Feedback Required

- Topic set to ADTF::DAT

- Customer set to BOSCH

Hallo Stephan,

kannst Du uns bitte noch die verwendete ADTF Version nennen?

#2 - 2019-07-12 11:00 - hidden

Win 64 2.13.3

Korrektur: die streamtime wird vermutlich nicht direkt um 20 ms erhöht, sondern erst um 9 und dann um 11. Ansonsten sollten alle Angaben stimmen.

#3 - 2019-07-12 13:02 - hidden

- Status changed from Customer Feedback Required to In Progress

- Department set to CC-DA/ESU

- Affected Products ADTF 2.13.3 added

- Platform Windows 7 64bit added

#5 - 2019-07-15 08:50 - hidden

Hallo,

die Stream Time wird diskret vor jedem verschickten Sample erhöht. Der Kernel wird synchron über diesen Zeitsprung informiert und überprüft dann welche Timer in der Zeitspanne hätten laufen sollen. Leider hat er dafür nur den letzen und den neuen Zeitstempel zur Verfügung. Er kann also leider nur die Anzahl der notwendigen Timer-Aufrufe feststellen. Selbst wenn er dabei die Zeit interpolieren würde, kann er die Stream Time nicht mehr in die Vergangenheit zurücksetzen, damit ein pClock->GetStreamTime() im Anwender Code den passenden Zeitstempel liefert.

Also ja, die Genauigkeit, der Timer hängt von der maximalen Datenrate im DAT-File ab. (Das ist ein Punkt der in ADTF 3 besser gelöst wird, da hier auch die Trigger von Verarbeitungsketten mit aufgezeichnet werden).

Leider bleiben nur die zwei Möglichkeiten:

- zum einen den Algorithmus so anzupassen, dass eine Zeitdauer zwischen den Aufrufen von 0 kein Problem darstellt (das könnte theoretisch auch im Live-Betrieb auftreten wenn der OS-Kernel durch die Last nicht nachkommt)
- so wie beschrieben höherfrequente Datenströme mit aufzeichnen/einfügen. Da oft ein CAN oder Flexray Bus mit aufgezeichnet wird, kommt das Problem in anderen Use-Cases nicht so sehr zum tragen.

Es tut mir leid, dass es da im Moment in ADTF 2 keine besseren Möglichkeiten gibt.

Grüße,

Martin

#6 - 2019-07-15 09:02 - hidden

Der Vollständigkeit halber: als unzureichende Alternative gibt es noch cCyclicThread.

Der läuft aber komplett losgelößt von der Stream Time (also vom Playbackspeed). Aber auch dort wird GetStreamTime() im Falle, dass zwei Sample mehr als einen Timer-Intervall auseinander liegen zweimal die gleiche Zeit zurückliefern, die diese ja nur diskret erhöht wird (auch da gibts in ADTF 3 neue Möglichkeiten zur Interpolation :-)).

#7 - 2019-07-15 10:23 - hidden

- Status changed from In Progress to Customer Feedback Required

#8 - 2019-07-17 11:32 - hidden

Hallo Stephan Stühmer,

wir haben kein Feedback erhalten.
Bitte bis spätestens 19.07. um ein Feedback.

Danke und Gruß

Matthias

#9 - 2019-07-17 13:15 - hidden

Hallo,

Danke für die ausführliche Antwort.
Ein Umstieg auf ADTF3 ist zur Laufzeit des Projektes leider nicht möglich.
Werde also einen der Workarounds bemühen müssen.

Ein Gedanke noch: wir spielen Messdaten aus dem Fahrzeug ab, eine Aufzeichnung des Timer/Task-Schemas ist also auch mit ADTF3 nicht möglich, weil der Algo nicht mitläuft.
Könnte man evtl. die StreamTime Erhöhung vom Player abfangen, ermitteln wann die Timer/Threads dran wären, die StreamTime auf die ermittelten Zeitpunkte erhöhen und die Timer/Threads damit aufrufen, bevor auch die neuen Samples verschickt werden? Dann vor dem zweiten Timer-Aufruf die Samples losschicken. Klar, kein Update für ADTF2, aber vielleicht könnte so etwas in ADTF3 landen….

Ich habe hier nämlich auch noch nen Filter, der CAN-Busdaten einliest und auch mit nem 10 ms cKernelTimer den Algo aufruft. Der produziert immer unterschiedliche Ergebnisse. Wenn ich ihn statt mit Timer auf eine 10 ms zyklische Botschaft triggere, dann produziert er immer die gleichen Ausgangsdaten. Irgendwie scheint das Senden der MediaSamples und das Aufrufen des Timers nicht deterministisch…

Mit freundlichen Grüßen / Best regards

Stephan Stuehmer

#10 - 2019-07-18 11:06 - hidden

- Status changed from Customer Feedback Required to In Progress

#11 - 2019-07-26 08:36 - hidden

@Martin: Kannst du hier noch einen Blick drauf werfen ?

#12 - 2019-08-01 08:54 - hidden

Auf Feedback von Martin nach seinen Urlaub warten

#14 - 2019-08-13 09:01 - hidden

- Product Issue Numbers set to ACORE-10107

Hallo Stephan Stuehmer,

ja, so ein Mechanismus ist sicher sinnvoll und auch schon mal angedacht worden. Im Moment passt es nicht ganz zur Architektur und Trennung von Player und Kernel Service. Dazu ist eine etwas gröbere Anpassung nötig und ich habe dafür das Ticket ACORE-10107 erstellt.

Deterministisch ist das Verhalten im Playback Fall aber auch jetzt schon, nur halt mit einer eventuellen Zeitdifferenz von 0. Aber die Timeraufrufe sollten jedes mal exakt in der gleichen Abfolge stattfinden.

Wann wird denn der cKernelTimer erzeugt, in Start() oder in Init()? In ADTF 2 übernimmt der Player das Streamtime-Handling erst nach Init(StageGraphReady). Wenn Ihr Filter vorher schon einen Timer registriert, beginnt der mit der Systemzeit zu laufen und wird dann "umgehangen", da ist aber der Intervallstartpunkt abhängig von der Systemzeit.

Grüße,

Martin

#15 - 2019-08-14 09:50 - hidden

- Status changed from In Progress to Customer Feedback Required

#16 - 2019-08-16 14:32 - hidden

@Matthias
Bitte beobachten

#17 - 2019-08-21 12:40 - hidden

Hallo Stephan Stuehmer,

ist die Information von Martin (13.08.) angekommen?
Bitte um ein kurzes Feedback

Danke und Gruß
Matthias

#18 - 2019-08-26 09:20 - hidden

Hallo Stephan Stuehmer,

bitte um Feedback bis zum 28.08.
Danach würden wir das Ticket sonst schließen.

Gruß
Matthias

#19 - 2019-09-02 09:21 - hidden

- Subject changed from StreamTime Vehalten in einer Playback Config to StreamTime acting in a Playback Config

- Description updated

- Status changed from Customer Feedback Required to To Be Closed

- Resolution set to No Customer Feedback

- Product Issue Numbers changed from ACORE-10107 to https://www.cip.audi.de/jira/browse/ACORE-10107

#22 - 2020-07-07 13:52 - hidden

- Project changed from 5 to Public Support

- Private changed from Yes to No

- Support Level changed from 2nd Level to 3rd Level

#23 - 2020-07-07 16:39 - hidden

- Status changed from To Be Closed to Closed