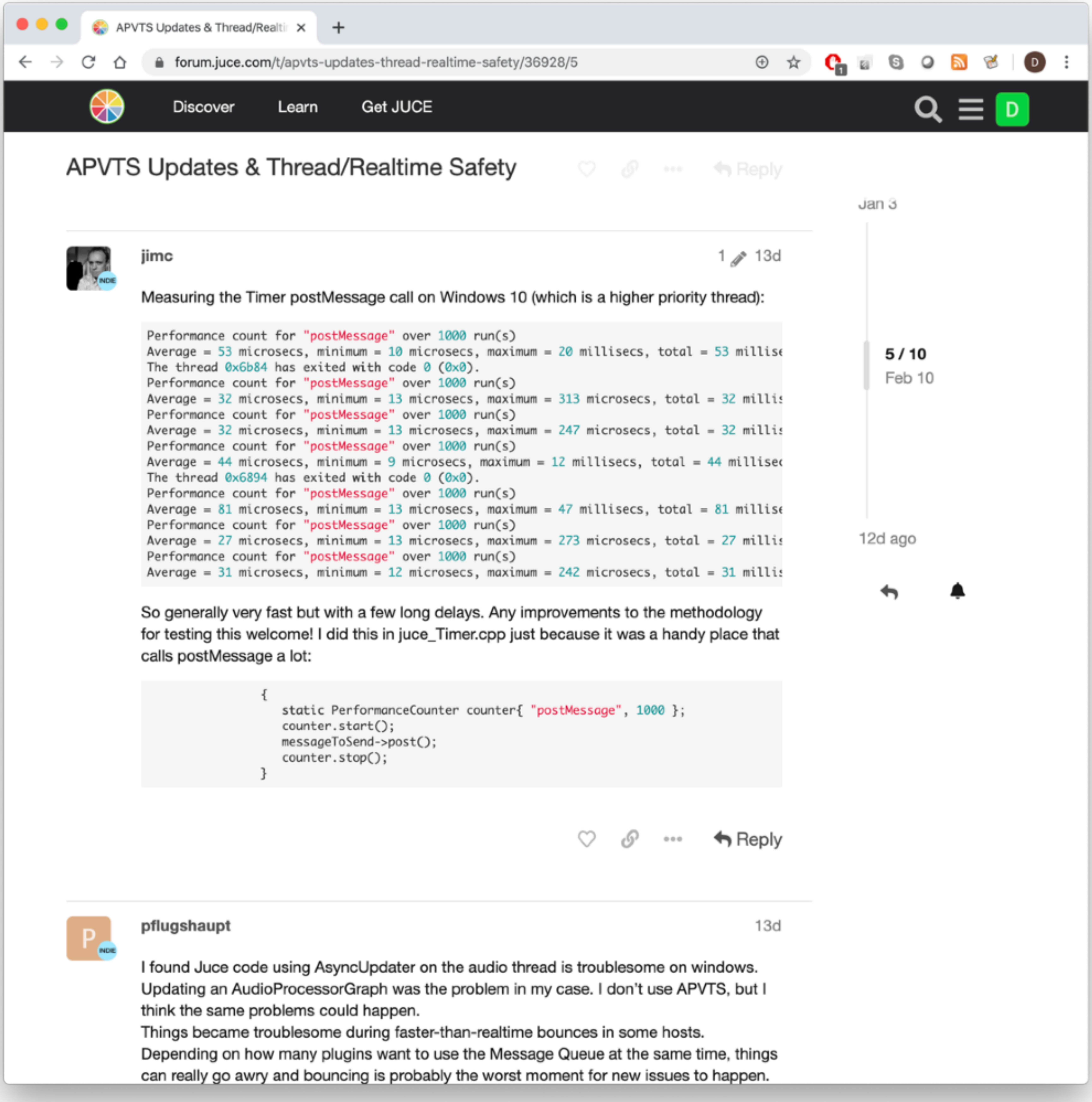

APVTS Updates & Thread/Realti
forum.juce.com/t/apvts-updates-thread-realtime-safety/36928/5
Discover
Learn
Get JUCE
D

APVTS Updates & Thread/Realtime Safety

Reply
Jan 3
5 / 10
Feb 10
12d ago

jimc 1 13d

Measuring the Timer postMessage call on Windows 10 (which is a higher priority thread):

```
Performance count for "postMessage" over 1000 run(s)
Average = 53 microsecs, minimum = 10 microsecs, maximum = 20 millisecs, total = 53 millise
The thread 0x6b84 has exited with code 0 (0x0).
Performance count for "postMessage" over 1000 run(s)
Average = 32 microsecs, minimum = 13 microsecs, maximum = 313 microsecs, total = 32 millis
Performance count for "postMessage" over 1000 run(s)
Average = 32 microsecs, minimum = 13 microsecs, maximum = 247 microsecs, total = 32 millis
Performance count for "postMessage" over 1000 run(s)
Average = 44 microsecs, minimum = 9 microsecs, maximum = 12 millisecs, total = 44 millisec
The thread 0x6894 has exited with code 0 (0x0).
Performance count for "postMessage" over 1000 run(s)
Average = 81 microsecs, minimum = 13 microsecs, maximum = 47 millisecs, total = 81 millise
Performance count for "postMessage" over 1000 run(s)
Average = 27 microsecs, minimum = 13 microsecs, maximum = 273 microsecs, total = 27 millis
Performance count for "postMessage" over 1000 run(s)
Average = 31 microsecs, minimum = 12 microsecs, maximum = 242 microsecs, total = 31 millis
```

So generally very fast but with a few long delays. Any improvements to the methodology for testing this welcome! I did this in juce_Timer.cpp just because it was a handy place that calls postMessage a lot:

```
            {
                static PerformanceCounter counter{ "postMessage", 1000 };
                counter.start();
                messageToSend->post();
                counter.stop();
            }
```

pflugshaupt 13d

I found Juce code using AsyncUpdater on the audio thread is troublesome on windows. Updating an AudioProcessorGraph was the problem in my case. I don't use APVTS, but I think the same problems could happen.
Things became troublesome during faster-than-realtime bounces in some hosts.
Depending on how many plugins want to use the Message Queue at the same time, things can really go awry and bouncing is probably the worst moment for new issues to happen.

INDIE
5 / 10
Feb 10
12d ago
Reply

Measuring the Timer postMessage call on Windows 10 (which is a higher priority thread):

```
Performance count for "postMessage" over 1000 run(s)
Average = 53 microsecs, minimum = 10 microsecs, maximum = 20 millisecs, total = 53 millise
The thread 0x6b84 has exited with code 0 (0x0).
Performance count for "postMessage" over 1000 run(s)
Average = 32 microsecs, minimum = 13 microsecs, maximum = 313 microsecs, total = 32 millis
Performance count for "postMessage" over 1000 run(s)
Average = 32 microsecs, minimum = 13 microsecs, maximum = 247 microsecs, total = 32 millis
Performance count for "postMessage" over 1000 run(s)
Average = 44 microsecs, minimum = 9 microsecs, maximum = 12 millisecs, total = 44 millisec
The thread 0x6894 has exited with code 0 (0x0).
Performance count for "postMessage" over 1000 run(s)
Average = 81 microsecs, minimum = 13 microsecs, maximum = 47 millisecs, total = 81 millise
Performance count for "postMessage" over 1000 run(s)
Average = 27 microsecs, minimum = 13 microsecs, maximum = 273 microsecs, total = 27 millis
Performance count for "postMessage" over 1000 run(s)
Average = 31 microsecs, minimum = 12 microsecs, maximum = 242 microsecs, total = 31 millis
```

So generally very fast but with a few long delays. Any improvements to the methodology for testing this welcome! I did this in juce_Timer.cpp just because it was a handy place that calls postMessage a lot:

```
                {
                    static PerformanceCounter counter{ "postMessage", 1000 };
                    counter.start();
                    messageToSend->post();
                    counter.stop();
                }
```

How long is an audio buffer?