Subject: WhenAction() -> ProcessEvents() -> WhenAction() -> hang, crash :-)
Posted by xrysf03 on Sun, 08 Dec 2019 14:21:49 GMT
View Forum Message <> Reply to Message

Dear everybody,

while playing with the GUI in Ultimate++ (still without forking my own threads), lately it feels like I'm stuck... like I've found myself in a place where I'm stretching things a little too far, and I need to establish a new direction.

To be more specific:

I'll attach just a screenshot. Mind the large "Go" button in the upper right corner, and the large ScatterCtrl. After setting up some parameters, the user needs to press "Go" and the key things start to happen: the proggie tunes into a sequence of frequencies, spaced by 2 MHz in this setup, calculates an FFT for each frequency and applies the data to the final plot (ScatterCtrl).

In other words, the callback handler, installed for the "Go" button's WhenAction, does a lot of time-consuming work, blocks the GUI for quite some time. To allow for a gradual display of the data, I've learned to call Ctrl::ProcessEvents() at suitable instants during the band scan: namely, just after one partial FFT gets applied to the visual output (the spectrum visually grows from left to right). So the band scanning callback is calling Ctrl::ProcessEvents() every now and then.

Here's the catch: when the user presses the "Go" button again, while the band scanning routine is still at work (= the previous GUI callback hasn't returned yet), the program stops working - hangs and either has to be killed or segfaults on its own in a few seconds.

I figured right away that I probably should not call the button callback again, even if this is just a recursive reentrancy, rather than multi-threaded reentrancy. I tried to avoid the problem in two ways:

1) I tried installing a "latch" at the very beginning of the heavy "band scan" function, with a boolean flag (member of my main window class) to signal if the band scan is already running - in which case the second (and further) invocations of that heavy function would return immediately, without doing anything. That hasn't helped - and I could hazard a guess that this has to do with compiler options such as a missing -D_REENTRANT for GCC. This possibly controls how local variables are allocated and initialized (= not necessarily on stack etc).

2) I tried re-installing a different callback on start of the heavy worker function (itself the initial "Go" button callback). I just used the THISBACK macro at the start of this fat callback, to install *another* dummy callback. (I'd also change the label on the button to "Stop", and tried raising another flag, that would signal to the fat band scan function to give up the sweep after it's finished processing the current frequency slot.) For some reason, installing a different callback didn't work either, the result is the same = a hang upon the second "Go" button click.

Mind the checkbox/Option labeled "loop around". That one works fine. It doesn't have its WhenAction() hooked: the code of my fat band scan function just polls its value every now and then, and quits looping if the check mark is removed. So: that works fine, which makes me

believe, that it's the auto-recursive calling of the WhenAction of the "Go" button that causes the hang.

If I don't violate the rule that "I should not click the Go button while already scanning", the program works fine and is actually getting useful already.

I don't believe very much that these "hangs when clicking Go too early again" would be a problem inside my own code. I'm inclined to believe that this behavior is a "feature" = I'm trying to twist the GUI clockwork a little too far.

And, I should probably "do the right thing" = instead of abusing ProcessEvents() excessively, I should probably just fork my own thread for the heavy work, and return control to the GUI in the foreground thread ASAP. The GUI callbacks should not block. That way, the GUI will remain alive and snappy. I recall reading an UPP forum post about some way to "ping back" the GUI front end thread to run some code in its context, and that there's a GUI lock that I need to take if I mess with the objects from a different thread directly (actually maybe that's not possible at all).

= I probably know roughly what I need to do.
And the reason for me to post this topic is that: any comments are welcome :)

Frank

File Attachments

1) rtlsdr_skyline_GUI.png, downloaded 329 times