

Grok Prompt: how can I debug my Sierra Chart Advanced Custom Study .dll

Debugging a Sierra Chart Advanced Custom Study DLL involves setting up your development environment properly and using tools like Visual Studio to step through your code. Here's a straightforward guide to help you get started:

First, ensure your DLL is built with debugging support. Sierra Chart allows you to compile your custom study using Visual C++ in debug mode, which generates the necessary files to trace your code. Open Sierra Chart, go to the "Analysis" menu, and select "Build Custom Studies DLL." In the window that appears, choose "File" then "Select Files" to pick your .cpp source file from the ACS_Source folder (usually found in your Sierra Chart installation directory, like C:\SierraChart\ACS_Source). Next, under the "Build" menu, select "Build with Visual C++ - Debug." If you haven't installed the Visual C++ compiler yet, pick "Install Visual C++ Compiler" from the same menu and follow the prompts—it takes a bit of time, but it's a one-time setup. Once the build succeeds, you'll see a .dll and a .pdb file (Program Debug Database) in your Data folder (typically C:\SierraChart\Data). These files are crucial for debugging.

Now, launch Microsoft Visual Studio—I assume you've got it installed since it's the standard for this process. Open your .cpp file by going to "File" > "Open" > "File" and navigating to the ACS_Source folder. Add your custom study to a chart in Sierra Chart via "Analysis" > "Studies" > "Add Custom Study," selecting your DLL and study name. Back in Visual Studio, attach it to Sierra Chart's process: go to "Debug" > "Attach to Process," ensure "Attach to" is set to "Native," and pick either SierraChart.exe (32-bit) or SierraChart_64.exe (64-bit) from the list, depending on your Sierra Chart version, then hit "Attach."

To inspect your code, set a breakpoint. In Visual Studio, click in the gutter (the gray strip left of the code lines) next to a line in your study function—say, inside the scsf_ function where your logic lives. A red dot appears, marking the spot where execution will pause. Return to Sierra Chart and force a recalculation by selecting "Chart" > "Recalculate." This triggers your study, and Visual Studio will halt at the breakpoint. From there, use "Step Over" (F10) to move line-by-line without diving into function calls, "Step Into" (F11) to dig into functions, or "Continue" (F5) to run until the next breakpoint. Watch variables in Visual Studio's "Locals" or "Watch" windows to see what's happening.

If nothing pauses, double-check: Is your study on the chart? Did the DLL build correctly with debug settings? Is Visual Studio attached to the right process? Sierra Chart only calls your study function during specific events (like chart updates), so recalculating is key to hitting your breakpoint. Also, keep Sierra Chart running normally while attached—GUI freezing only happens when paused at a breakpoint, which is expected.

For trickier issues, log messages to Sierra Chart's Message Log. In your code, add something like `sc.AddMessageToLog("Checkpoint reached", 1)` to track execution

without breakpoints. Open the log via "Window" > "Message Log" to see your output. This helps confirm your study's running and where it might derail.

That's the gist—build in debug mode, attach Visual Studio, set breakpoints, and step through. It's hands-on, but you'll catch bugs faster once you get the rhythm. Anything specific tripping you up with your study?

Grok Prompt: how can I watch variable values

To watch variable values while debugging your Sierra Chart Advanced Custom Study DLL in Visual Studio, you can use a few built-in tools that make it easy to track what's happening in your code as it runs. Here's how to do it:

Once you've attached Visual Studio to SierraChart.exe (or the 64-bit version) and set a breakpoint in your study function (like I described earlier), trigger the breakpoint by recalculating the chart in Sierra Chart ("Chart" > "Recalculate"). When execution pauses at the breakpoint, you've got several options to inspect your variables.

The simplest is the **Locals** window. In Visual Studio, go to "Debug" > "Windows" > "Locals" (or press Alt+4 if it's not already visible). This automatically shows all variables in the current scope—like your study's SCSubgraphRef or SCInputRef objects, or any local integers, floats, etc. As you step through the code with F10 (Step Over) or F11 (Step Into), the values update in real-time. For example, if you've got float myValue = sc.Close[sc.Index], you'll see myValue change as the chart processes each bar.

If you want to focus on specific variables, use the **Watch** window. Right-click a variable in your code (say, sc.Index or a custom one like int counter), select "Add to Watch," and it'll appear in the Watch 1 window (open it via "Debug" > "Windows" > "Watch" > "Watch 1" if needed). You can also manually type expressions, like sc.Close[sc.Index - 1] or counter + 5, to evaluate them on the fly. The Watch window persists across debugging sessions, so it's handy for variables you check often.

For quick checks without a window, hover your mouse over a variable in the code while paused. A tooltip pops up with its current value. If it's an array like sc.Close, hover over it, and you can expand the tooltip to see values at different indices (though it's limited to a small range). This is fast but less flexible than Locals or Watch.

If your study uses Sierra Chart's SCStudyInterfaceRef sc structure, you can dig into its members—like sc.Close, sc.Open, or sc.ArraySize. In the Locals window, expand the sc object by clicking the little arrow next to it. You'll see all its fields, though some (like arrays) might need a Watch expression to inspect specific elements (e.g., sc.Close[0]).

One catch: Sierra Chart runs studies in a single-threaded context tied to chart updates, so variable values only change when the study recalculates. If values aren't updating,

ensure your breakpoint's in an active code path and the chart's triggering the study. Also, if a variable shows gibberish, your DLL might be out of sync—rebuild it in debug mode ("Build with Visual C++ - Debug") and reload it in Sierra Chart.

That's it—Locals for an automatic overview, Watch for custom tracking, and tooltips for quick peeks. Which variables are you trying to monitor? I can tailor this more if you've got specifics in mind.