

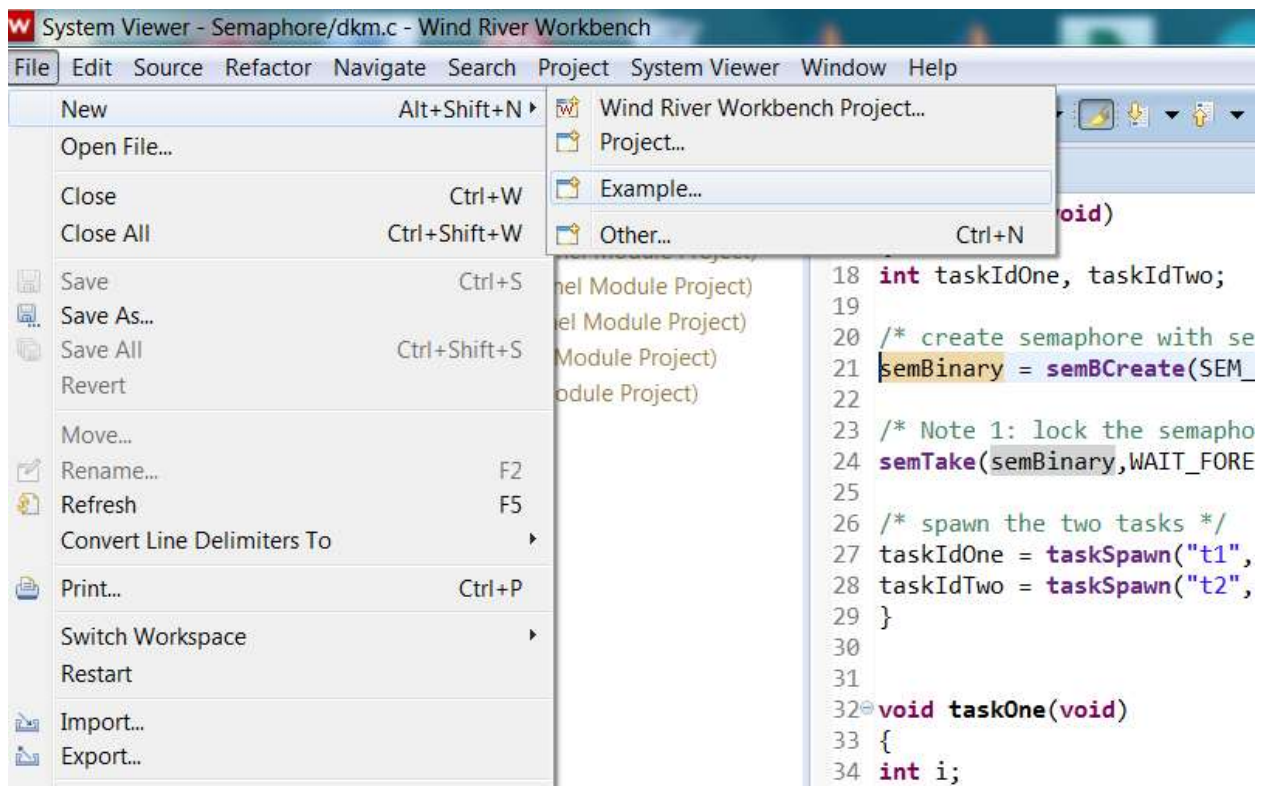
System Viewer Example Code Lab

In this lab you'll look at more specifics of performance measurement using the System Viewer capability.

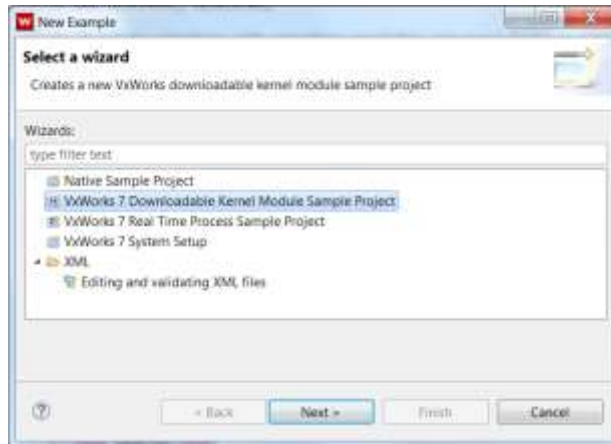
Using the Event Viewer

Let's start with a simple program to expose how to look at the event viewer, then you'll do an example to see how it can show you what your system is doing dynamically.

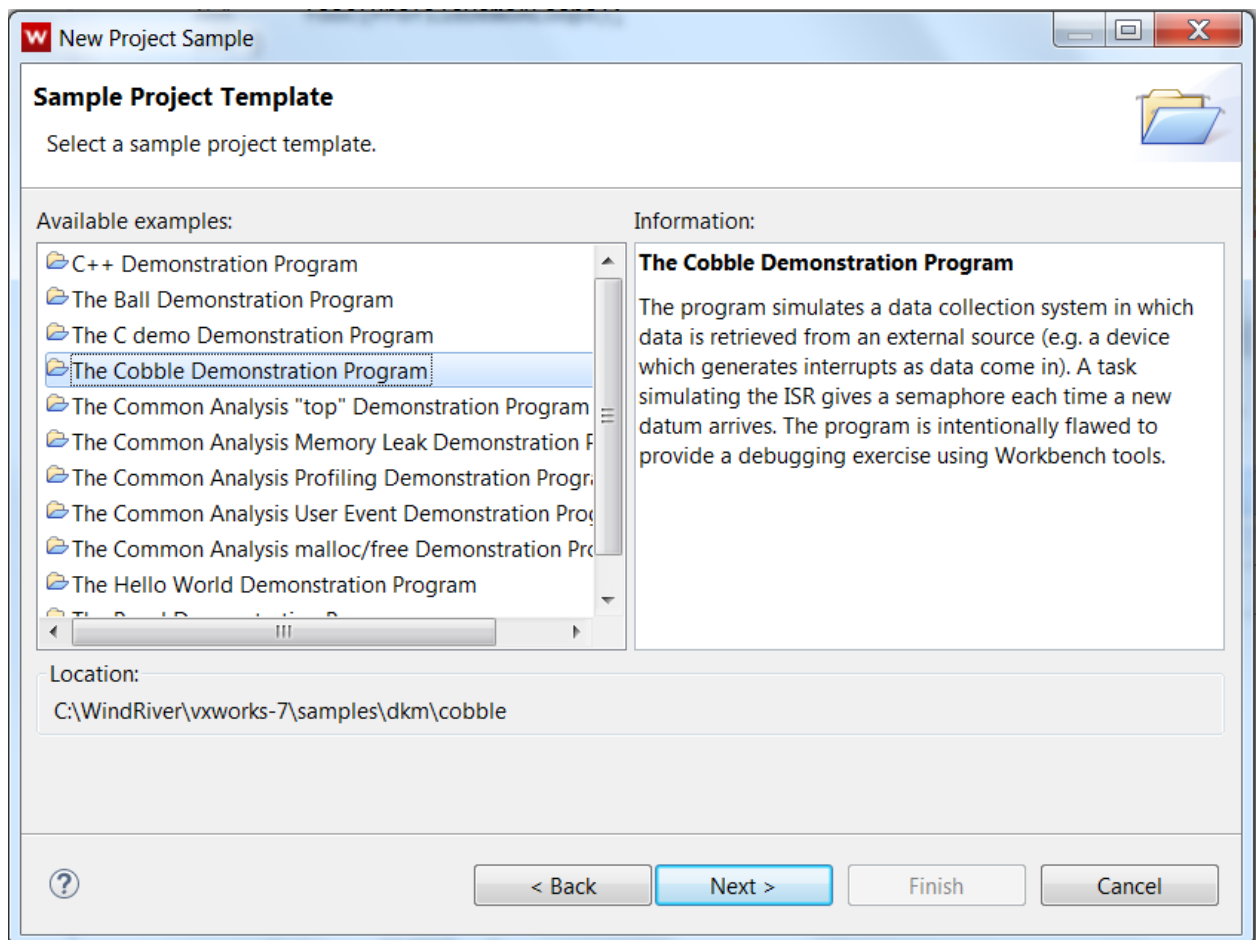
1. Start the VxWorks System
2. Open an example program that can be used to illustrate the Profiling capabilities of VxWorks.
To do this select New->Example



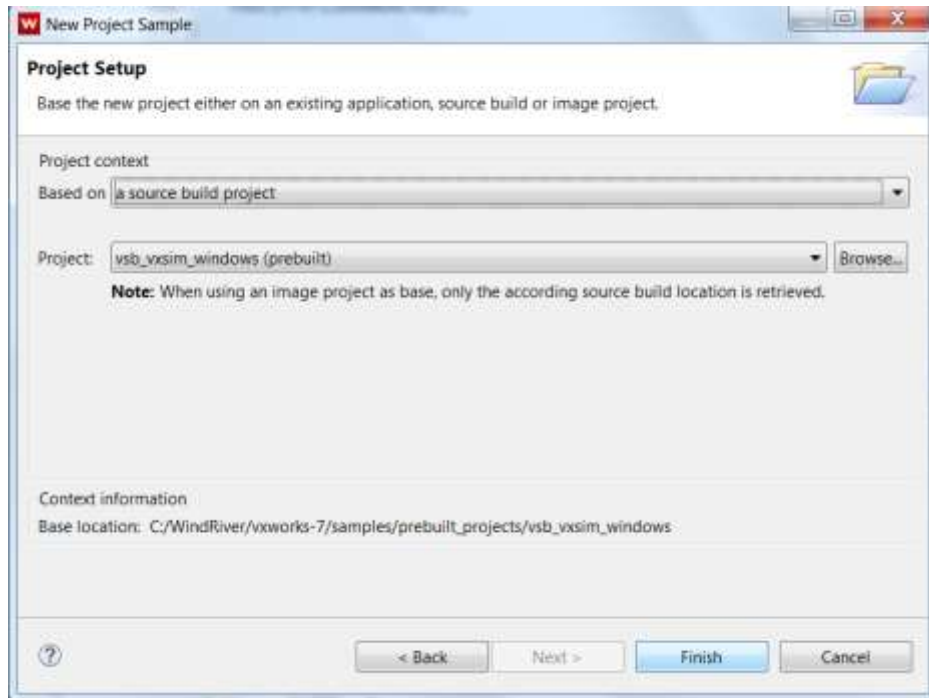
3. Select VxWorks7 Downloadable Kernel Module Sample Project



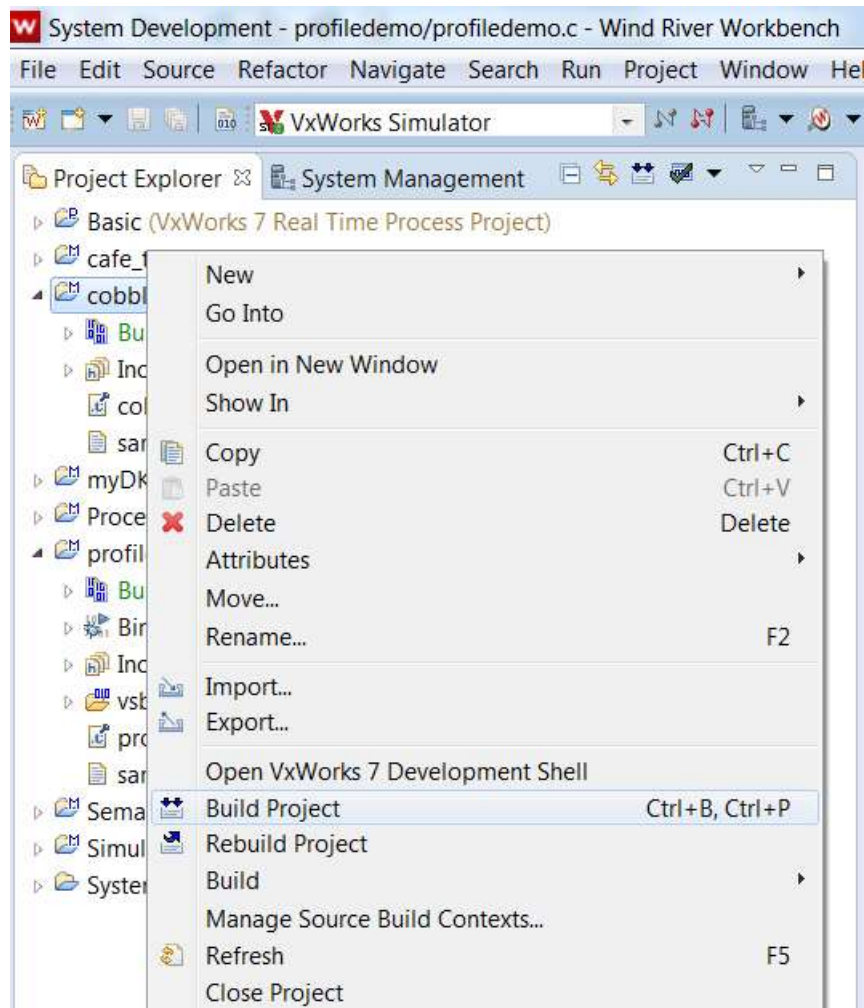
4. Select Next
5. Now select **The Cobble Demonstration Program**



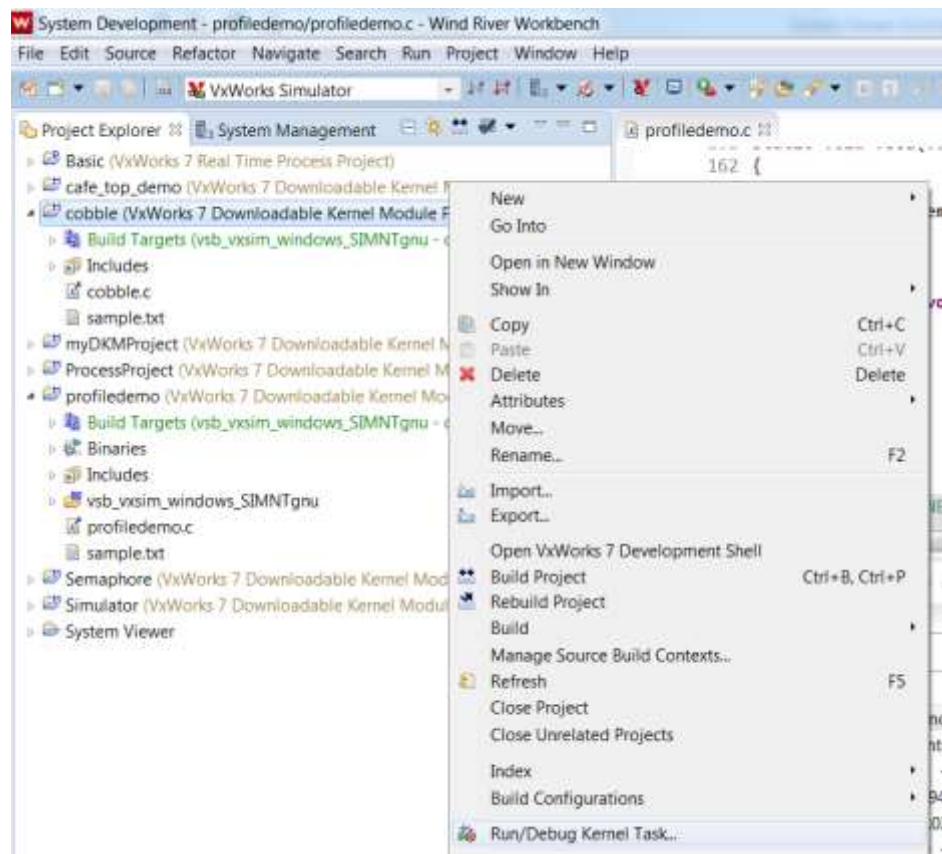
6. Now select a **source build project**, and the **vsb_vxsim_windows(prebuilt)** on the New Project Sample page, and then click **Finish**.



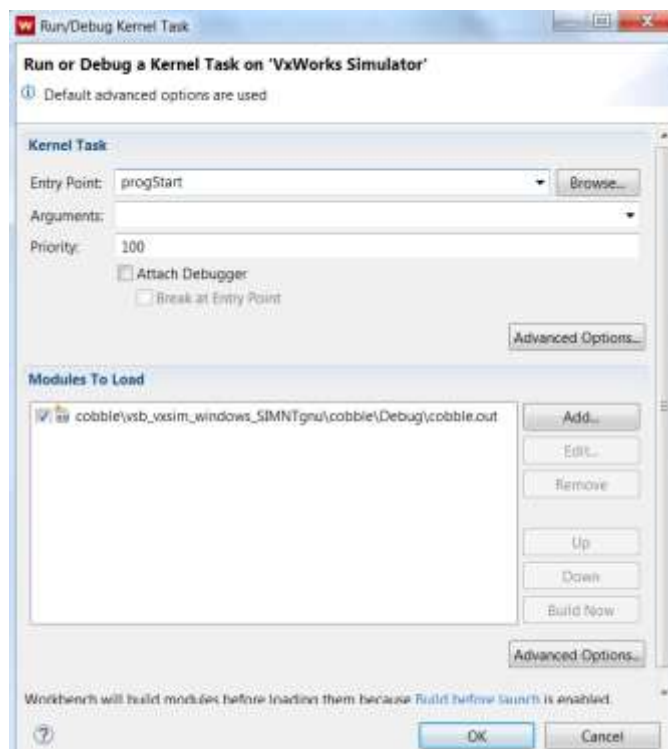
7. Now you are prepared to look at the events in your code. To do this right click on the project, and the select **Build Project**



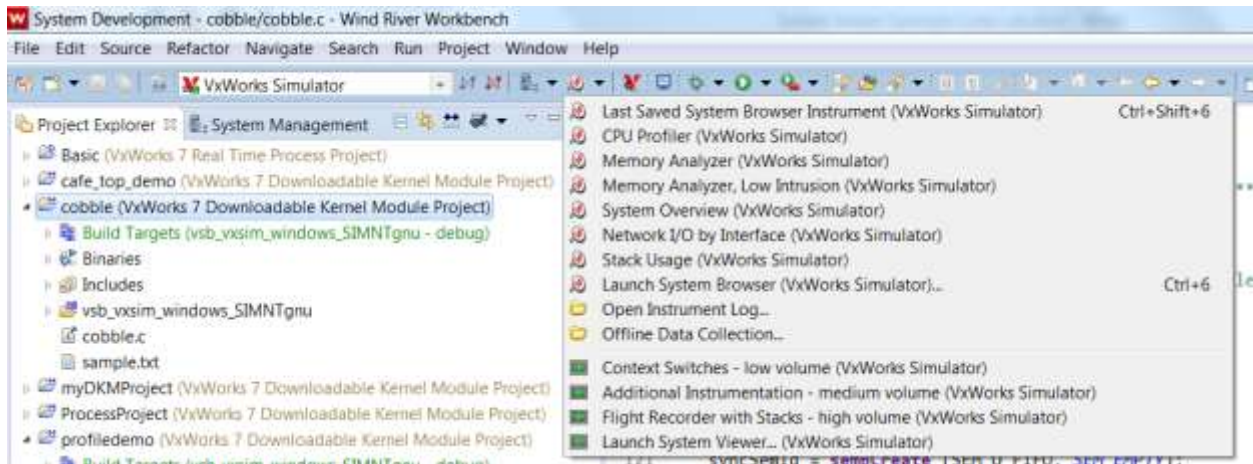
8. Connect to the VxWorks Simulator. Then run the code by right clicking on the project and then selecting **Run/Debug Kernel Task**



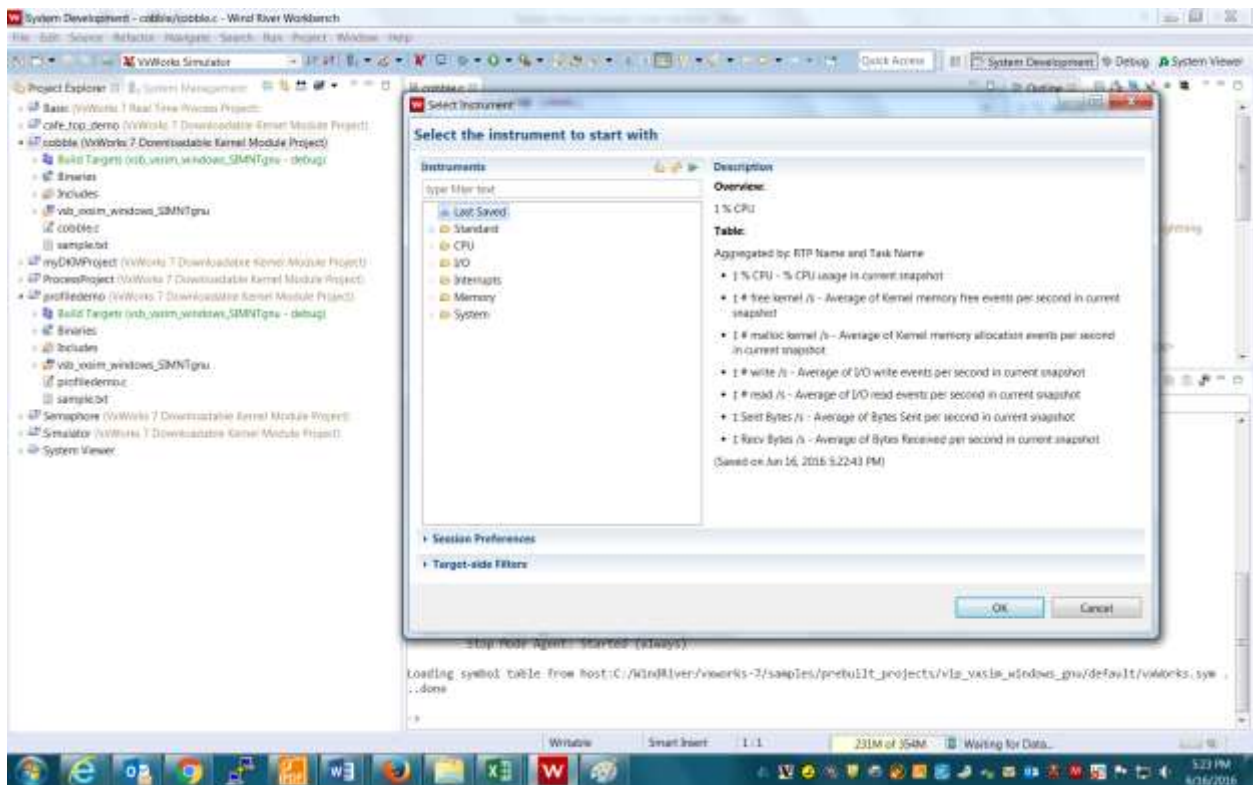
9. You'll want the Entry Point to be the progStart, so set it, like this:



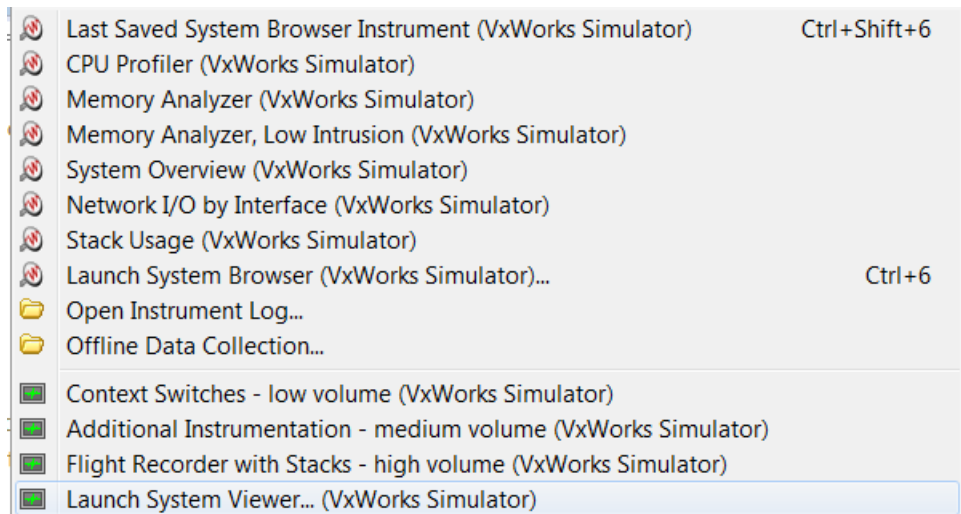
10. Now there are a number of different “instruments” that you can use to evaluate your system. To select these instruments, select the pull down menu by the System Viewer icon:



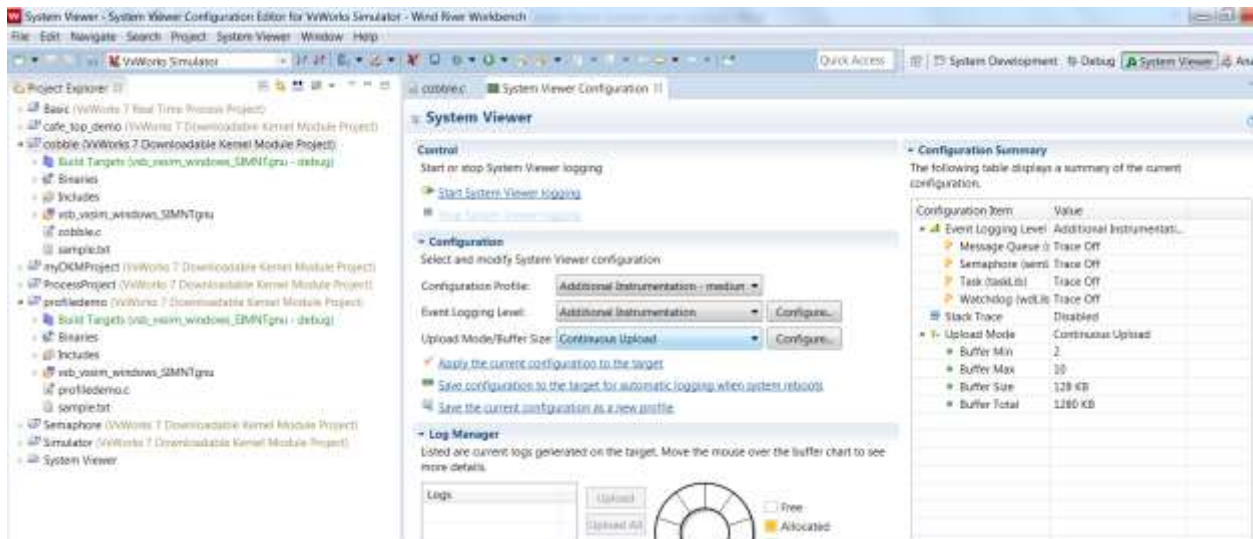
11. If you launch System Browser it will allow you to configure your own custom instruments. When you select that you'll see this pop up:



12. Now you can select what you want to see on your system.
13. However, it is easier to go directly to the System Viewer Configuration by selecting

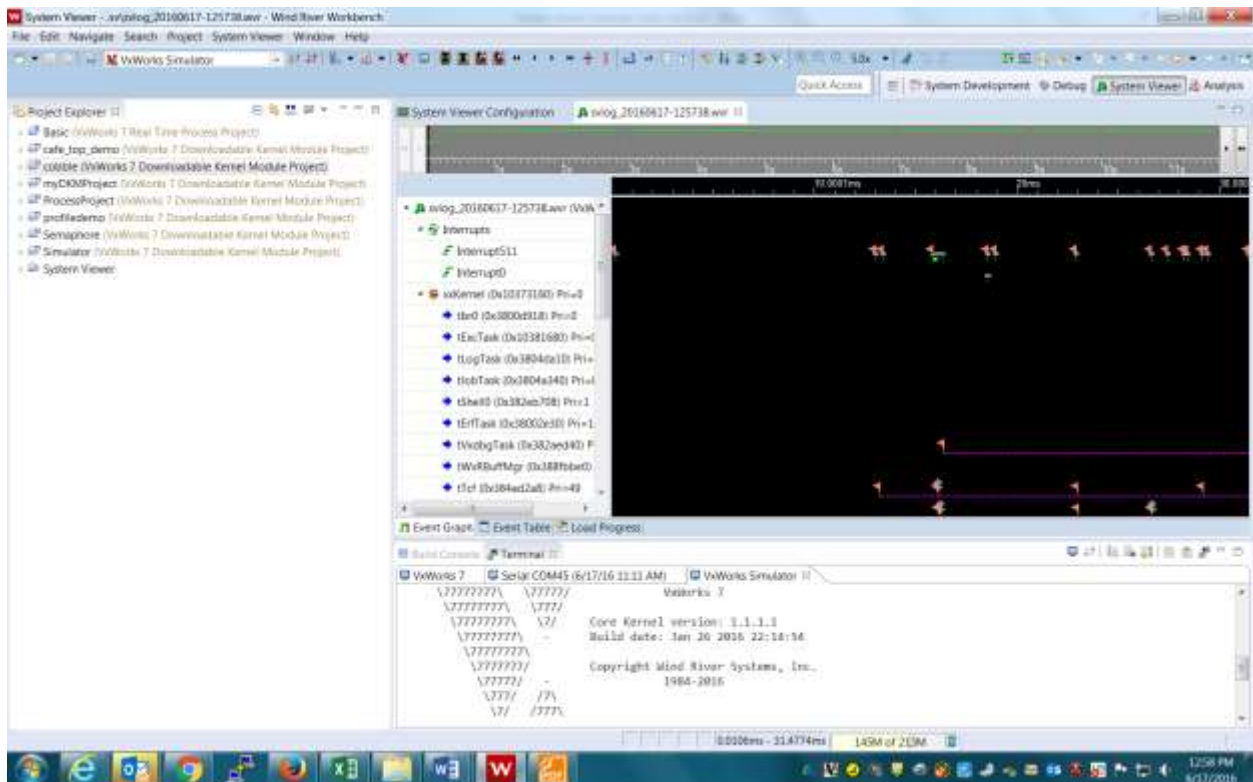


14. You should now see the configuration screen for the System Viewer:

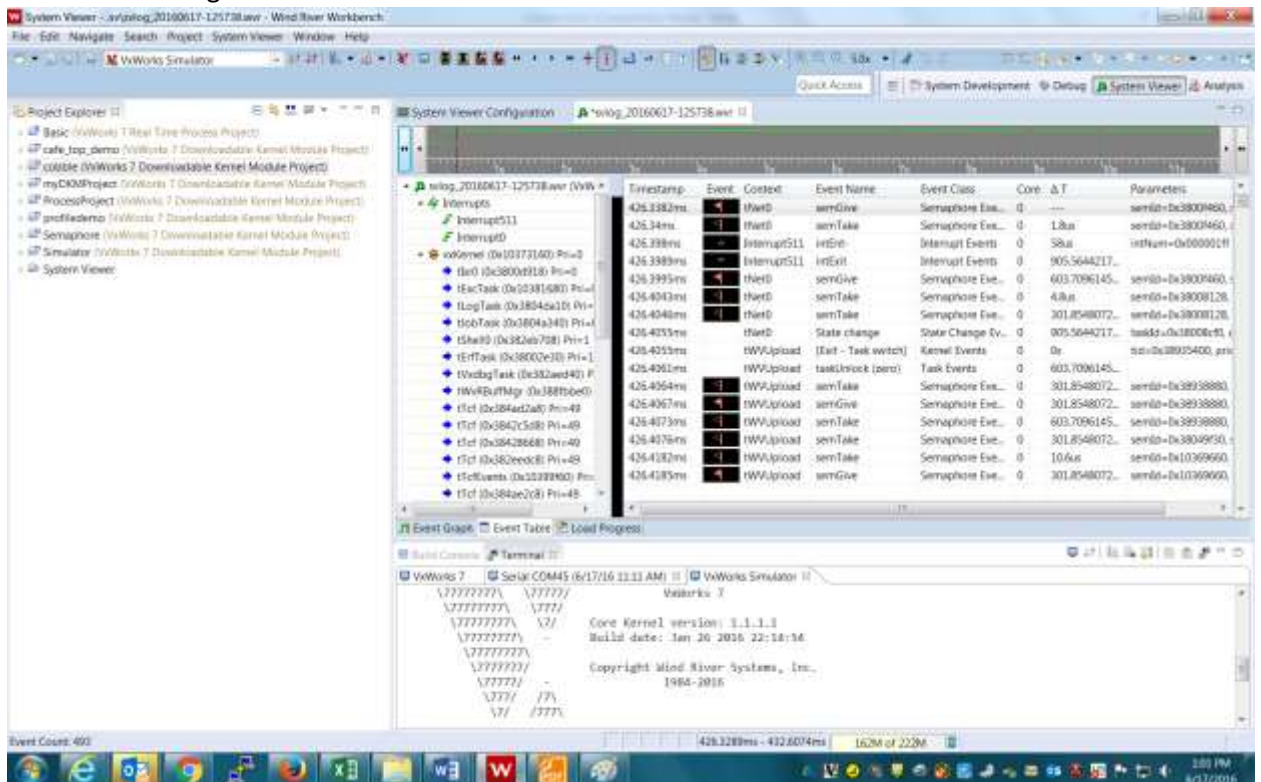


15. If you select different Configuration Profiles you'll see the Configuration summary change on the left side. Select one where Semaphores are available.

16. Now select Start System Viewer Logging. If you watch the Log Manager you will see it loading up with system events. When it is a quarter or half full you can select Stop System Viewer logging. Then the system viewer will pop us with all of the events you've asked for.

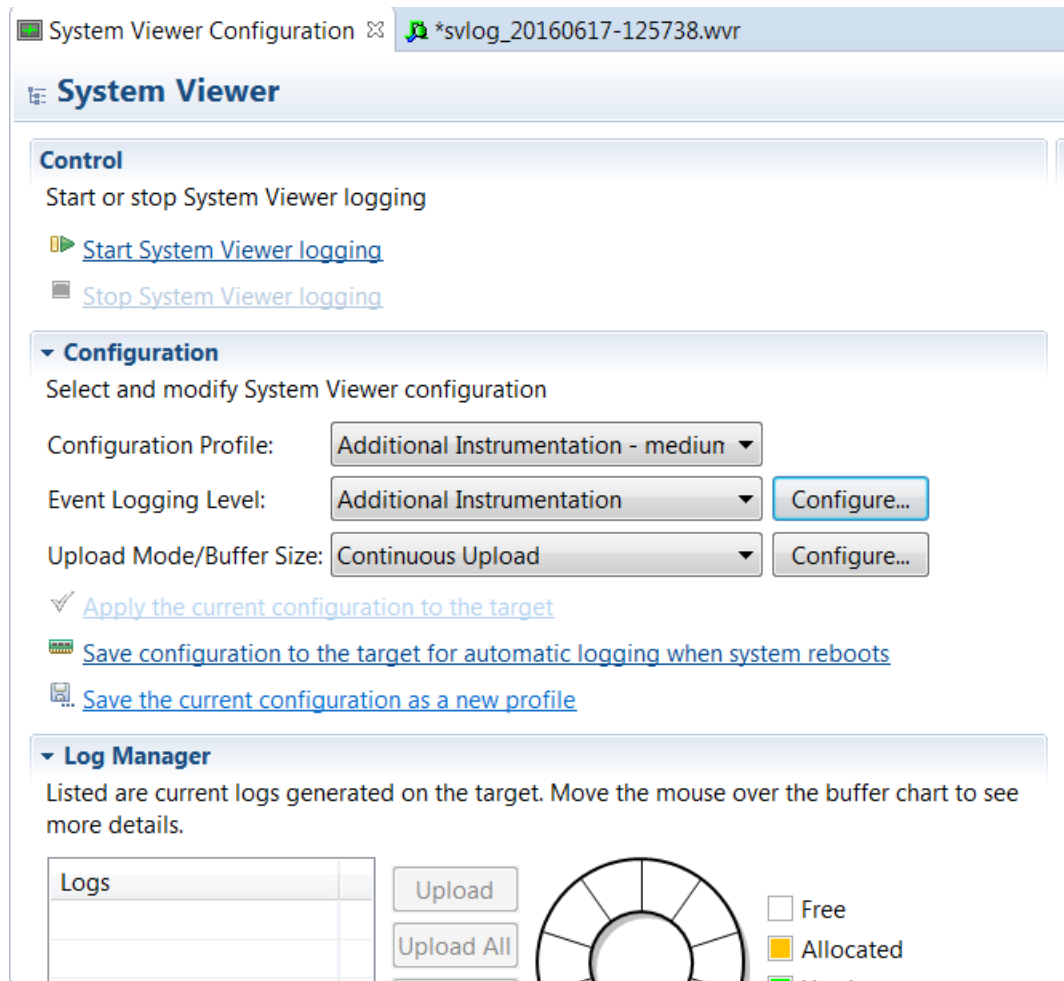


17. You might want to zoom in on a specific set of events, in this case let's look at the interrupts that are occurring. The Event Table can be useful for this:

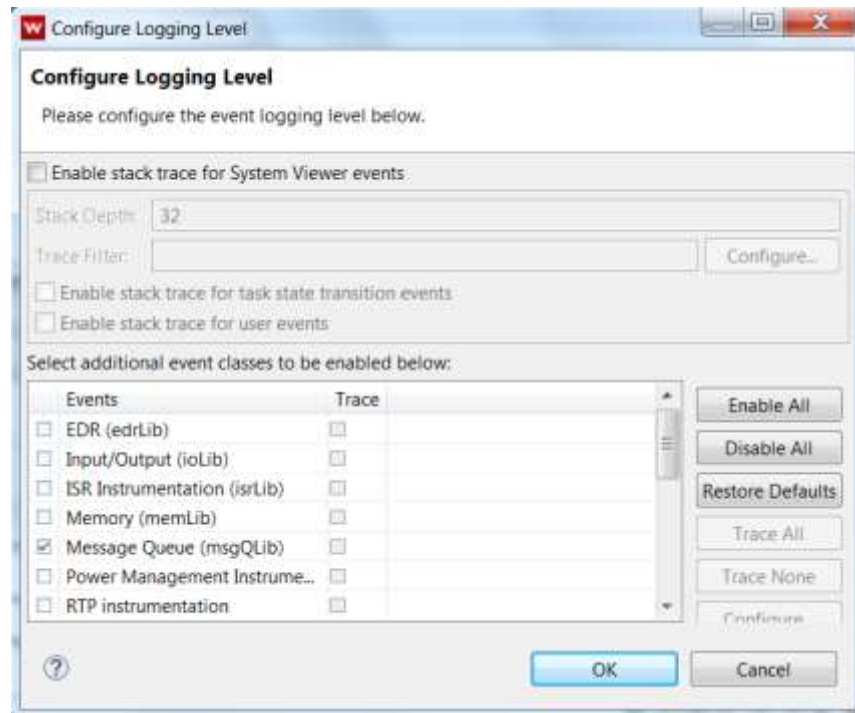


18. Here you can see the interrupts giving and taking the semaphores. You can also see when the interrupts are happening in the system.

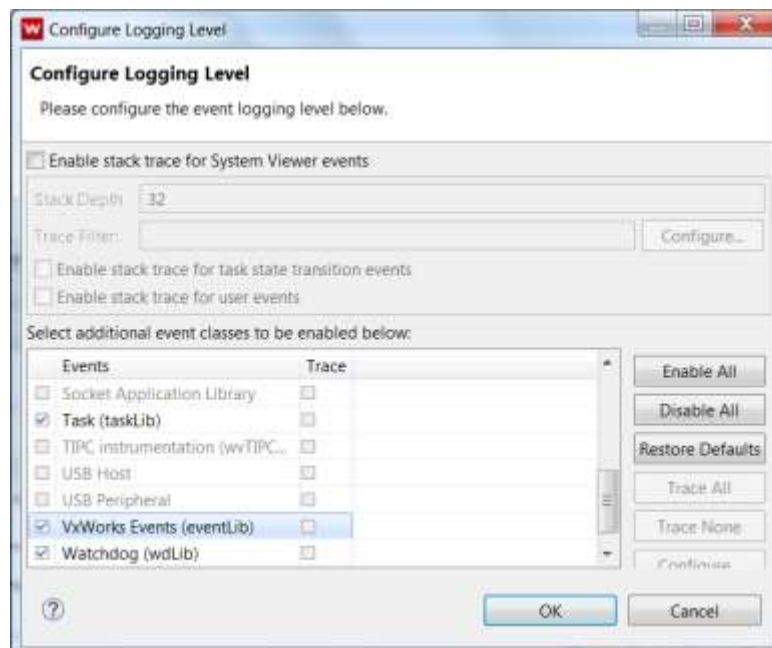
19. Let's configure what you want to see here. Go to the System Viewer Configuration Tab



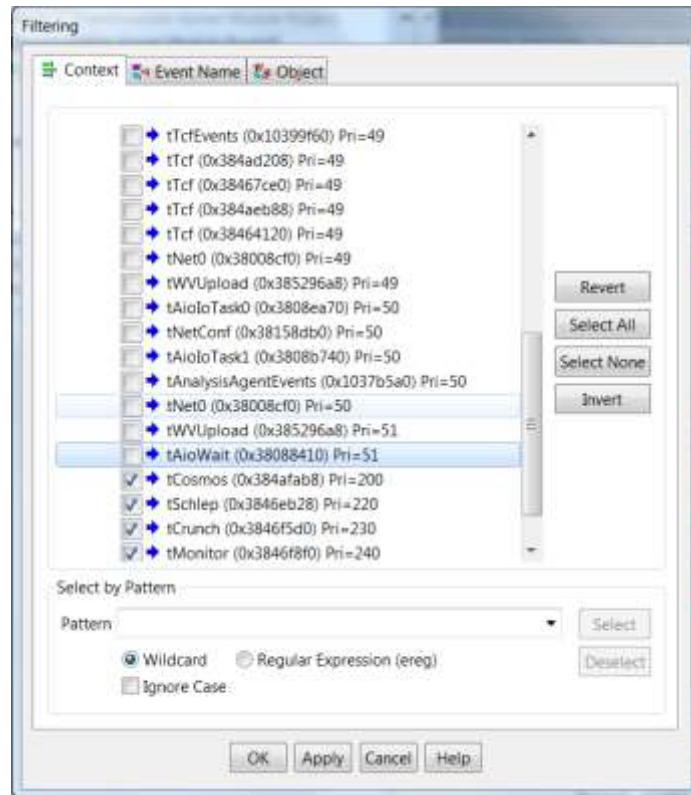
20. Now select the Configure for the Event Logging Level. You should see this:



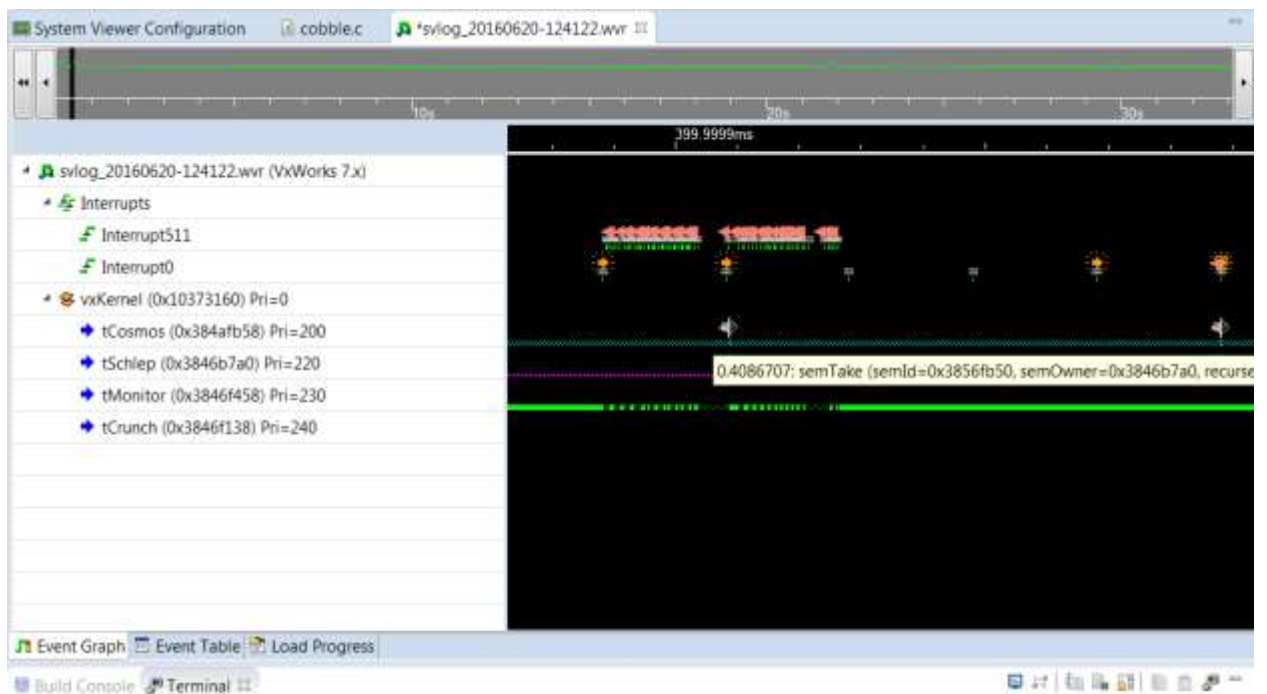
21. Scroll down and add VxWorks Events



22. Now you are set to look at the events. Run a system viewer log and look at the log. You will probably only want to look at your processes, so right click on one of the processes, then select filter ..., and you should be able to only set the processes that you are managing:

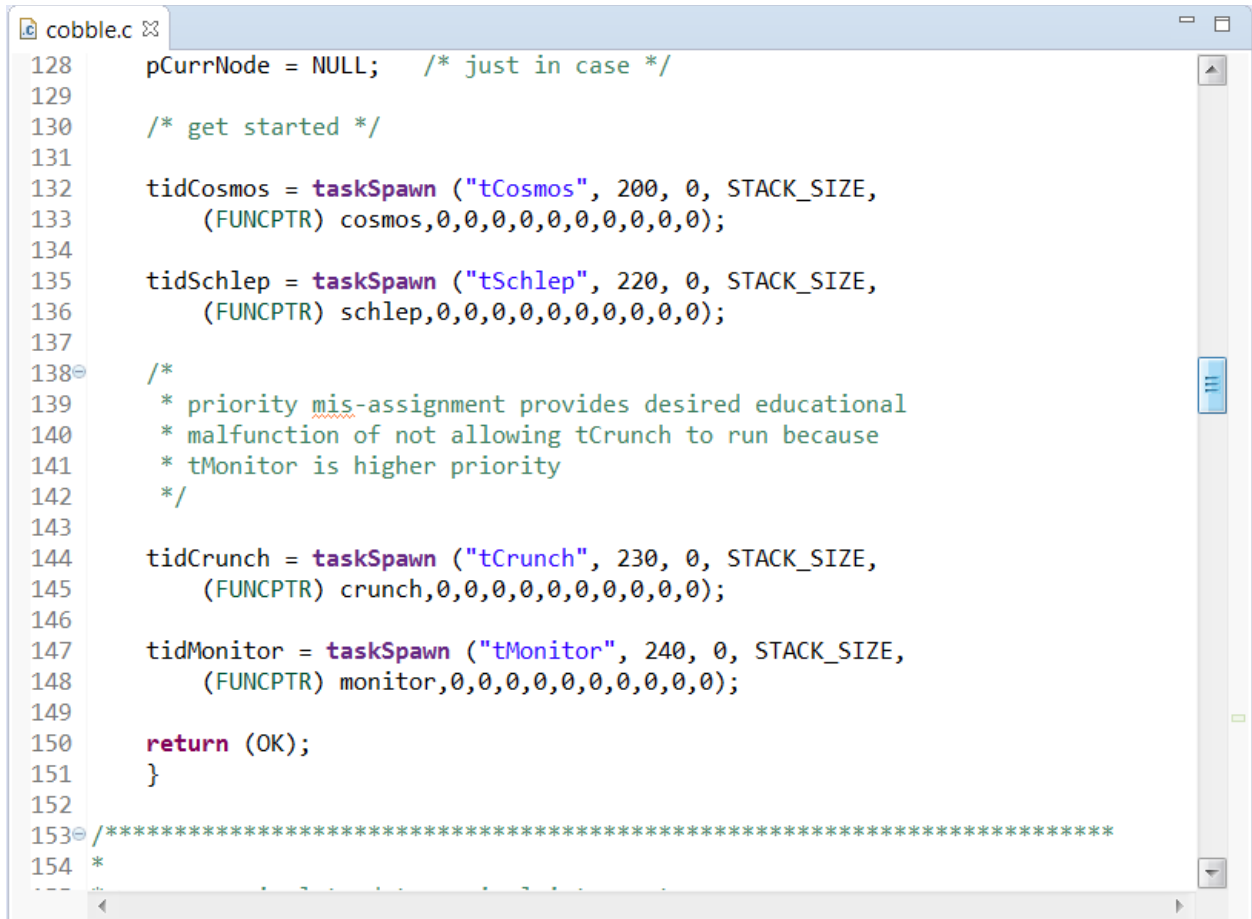


23. , Now look at the event set again, and you can see what is going on with the processes that your created:



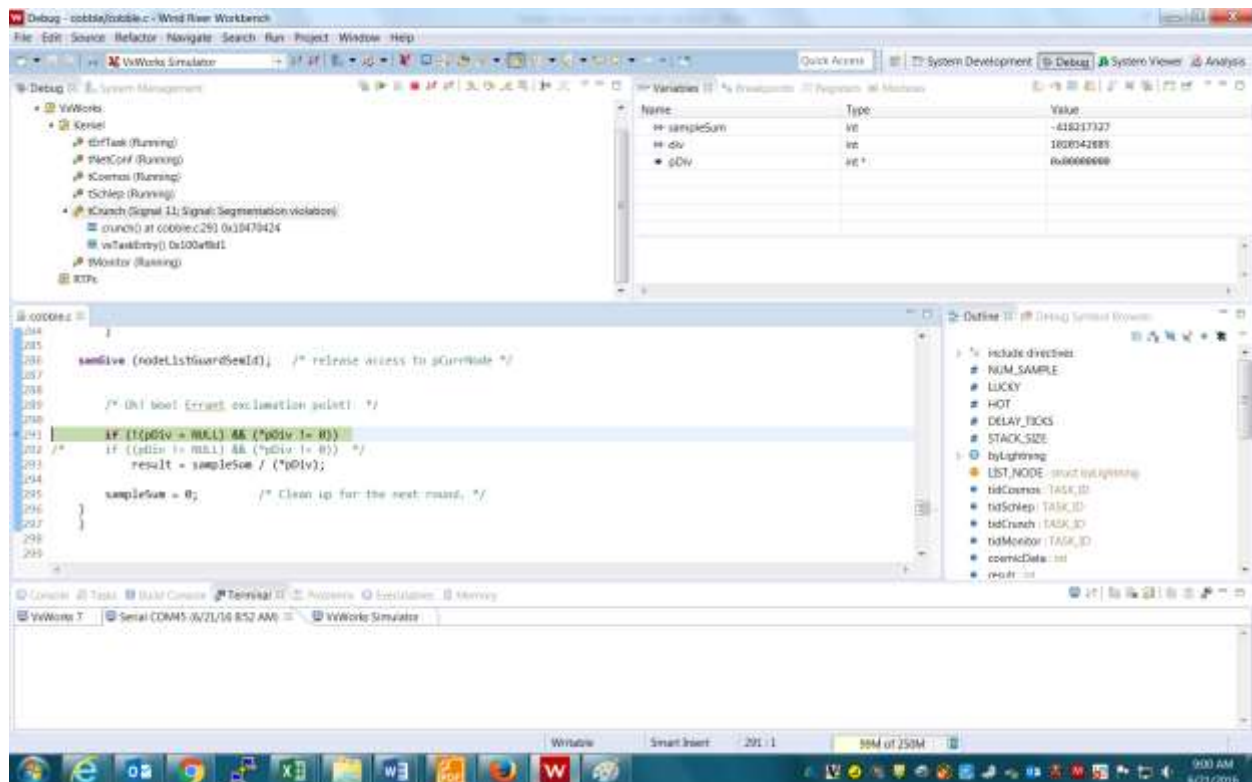
24. If you hover your mouse over one of the events you can see the event detail.

25. As you might suspect, this can be a powerful tool in viewing what is going on, which processes are running, and what events are taking place. Here we see that the tCrunch process is not running. This because the priority of the Monitor and Crunch processes are not correct. So go to the code at cobble.c and change the priority of the tCrunch and tMonitor processes, like this:

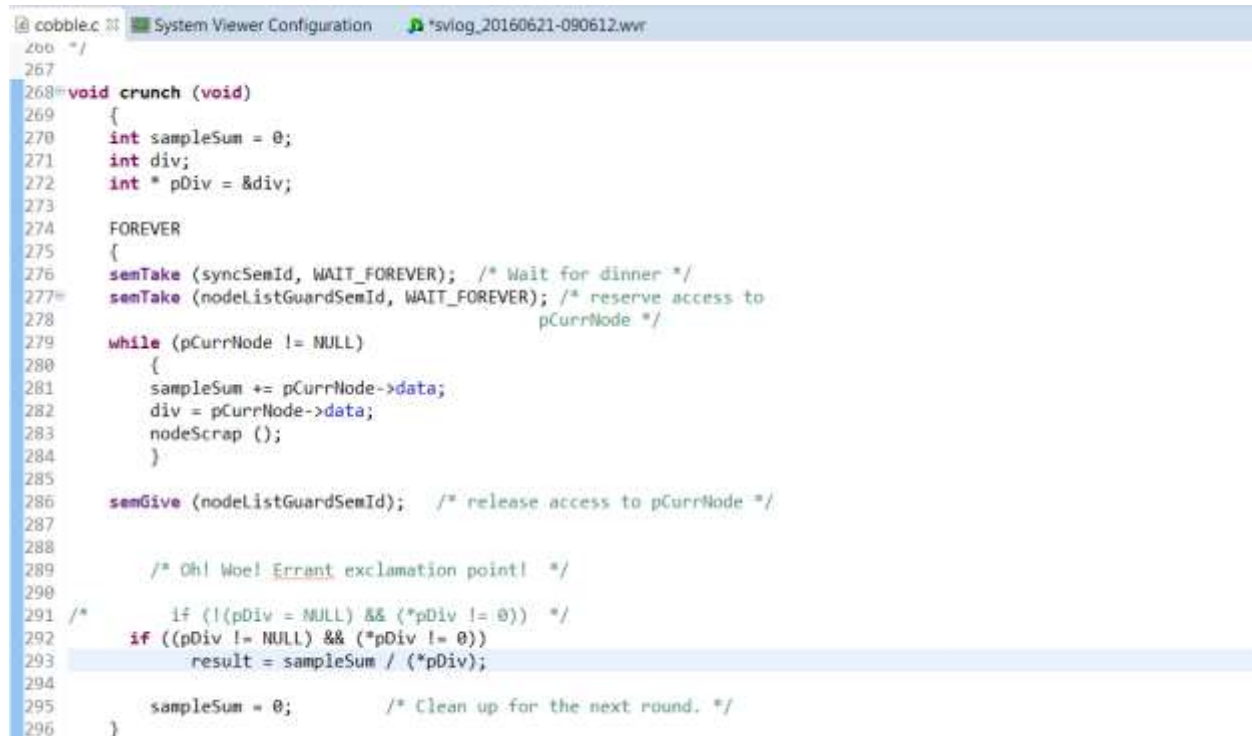


```
cobble.c
128     pCurrNode = NULL;    /* just in case */
129
130     /* get started */
131
132     tidCosmos = taskSpawn ("tCosmos", 200, 0, STACK_SIZE,
133                          (FUNCPTR) cosmos,0,0,0,0,0,0,0,0,0,0);
134
135     tidSchlep = taskSpawn ("tSchlep", 220, 0, STACK_SIZE,
136                          (FUNCPTR) schlep,0,0,0,0,0,0,0,0,0,0);
137
138     /*
139     * priority mis-assignment provides desired educational
140     * malfunction of not allowing tCrunch to run because
141     * tMonitor is higher priority
142     */
143
144     tidCrunch = taskSpawn ("tCrunch", 230, 0, STACK_SIZE,
145                          (FUNCPTR) crunch,0,0,0,0,0,0,0,0,0,0);
146
147     tidMonitor = taskSpawn ("tMonitor", 240, 0, STACK_SIZE,
148                          (FUNCPTR) monitor,0,0,0,0,0,0,0,0,0,0);
149
150     return (OK);
151 }
152
153 /*****
154 *
```

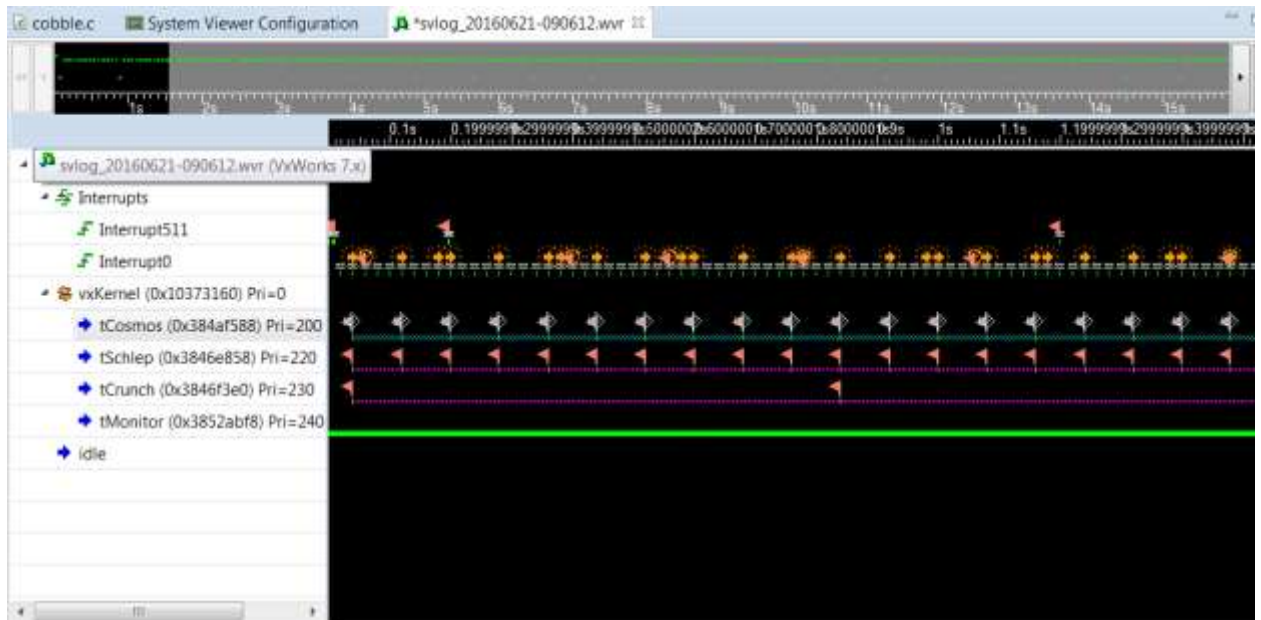
26. Now build the code and run the program.
27. The debug window will pop up and show you that you have an error in the code:



28. Notice the Segmentation violation in tCrunch. pDiv is 0, and you can't divide by 0. So you'll need to fix this code, like this:



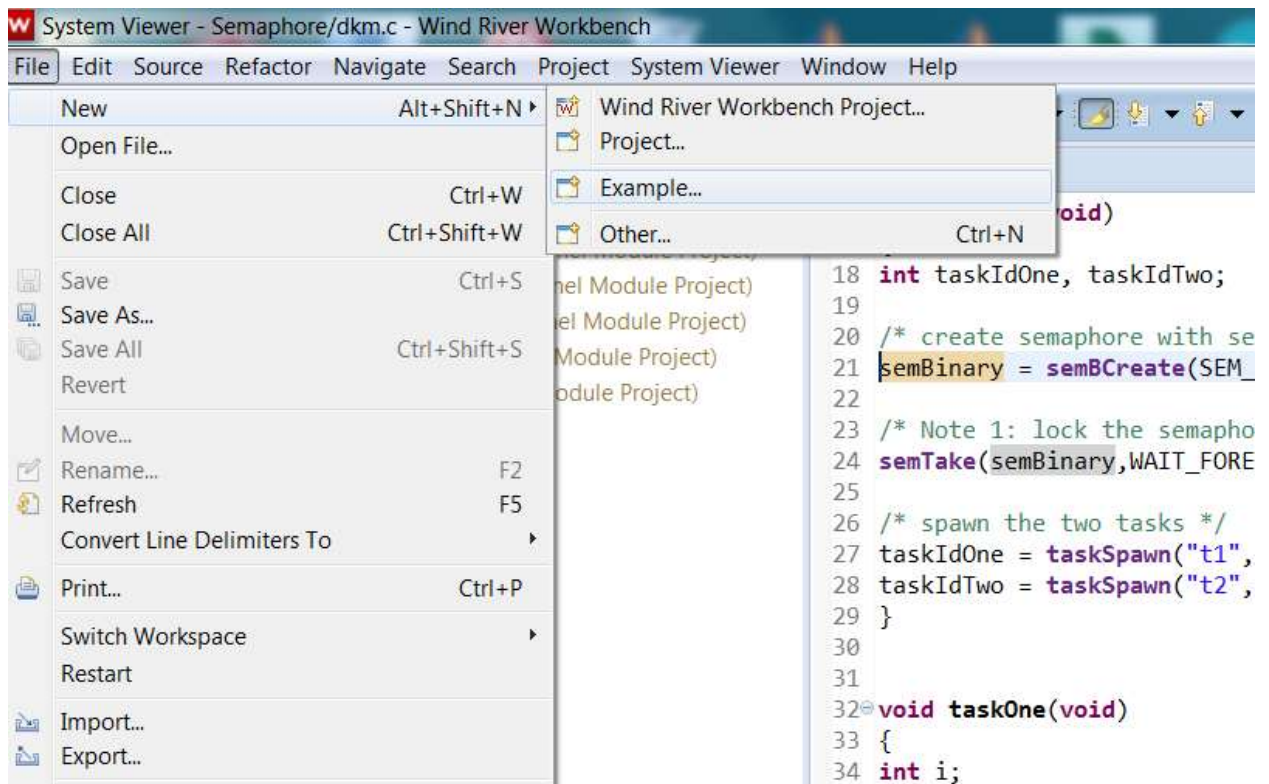
29. Once the code is fixed, Rebuild and download the code to the simulator. Now set up your System Viewer and you should see all your processes running!



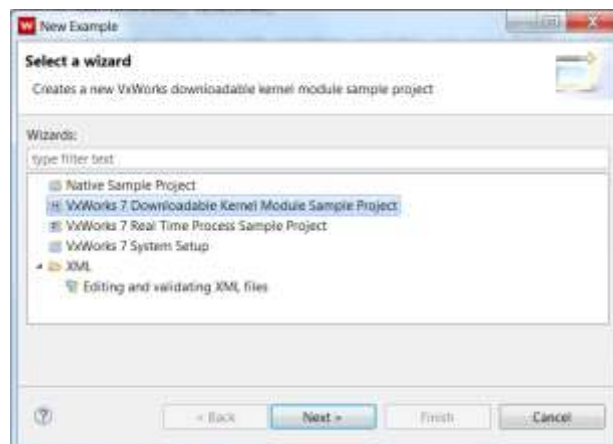
Performance Profiling Capabilities

But you can also profile your system, and see how the system is performing.

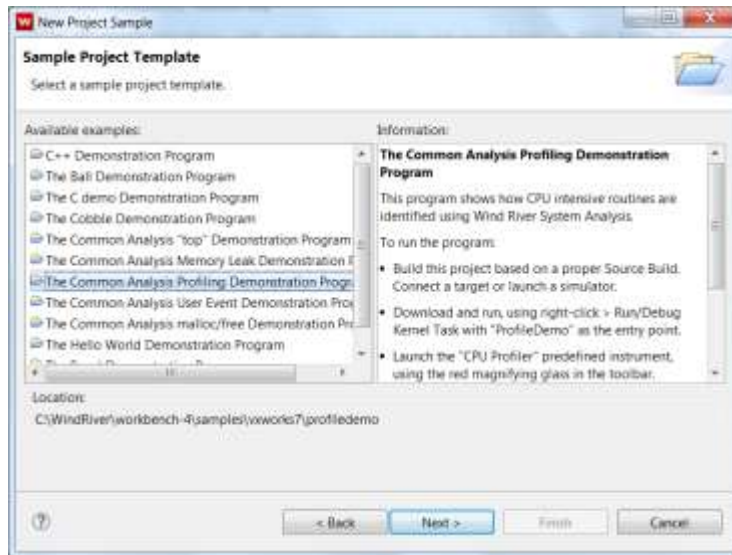
1. Start the VxWorks System
2. Open an example program that can be used to illustrate the Profiling capabilities of VxWorks. To do this select New->Example



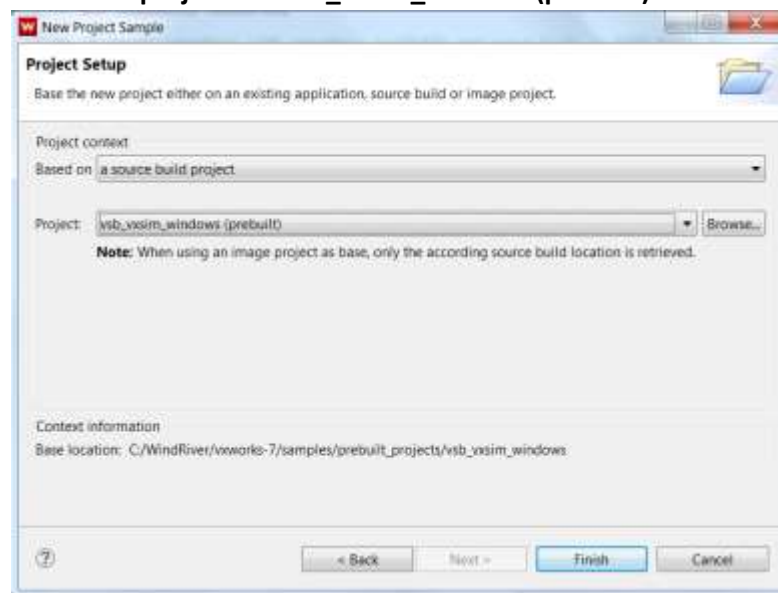
3. Select VxWorks7 Downloadable Kernel Module Sample Project



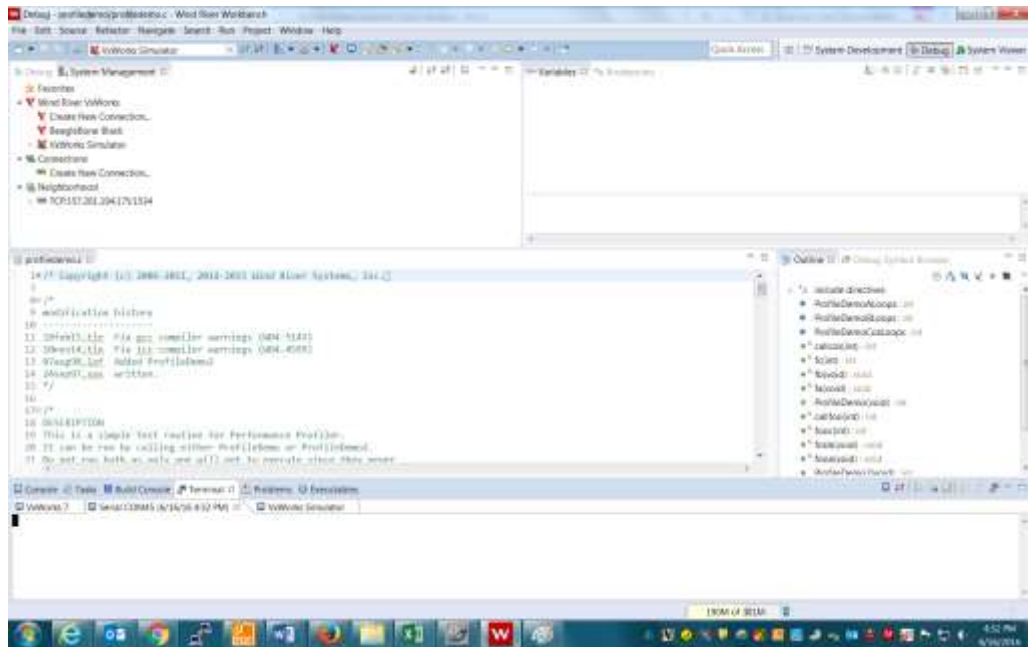
4. Select Next
5. Now Select The Common Analysis Profiling Demonstration Program



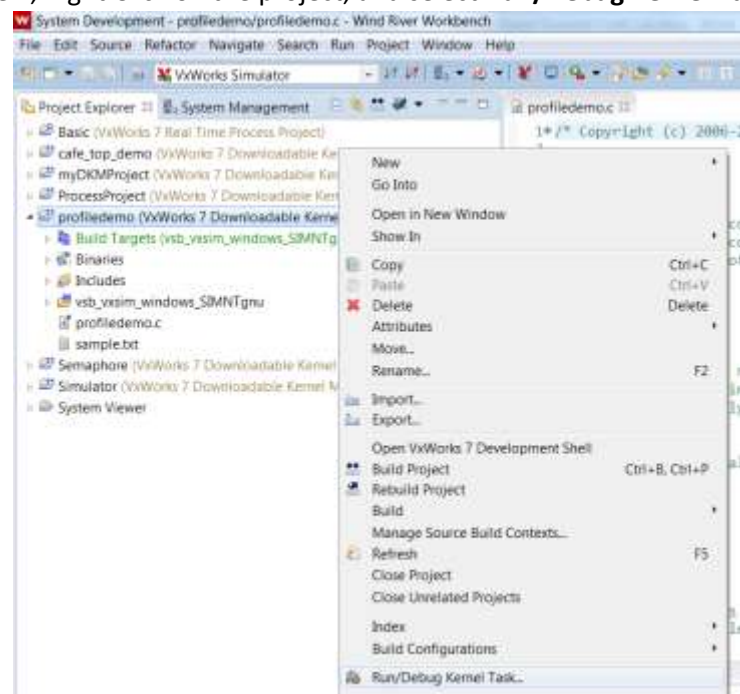
6. Hit Next again.
7. Now select a **source build project** and **vsb_vxsim_windows(prebuilt)**



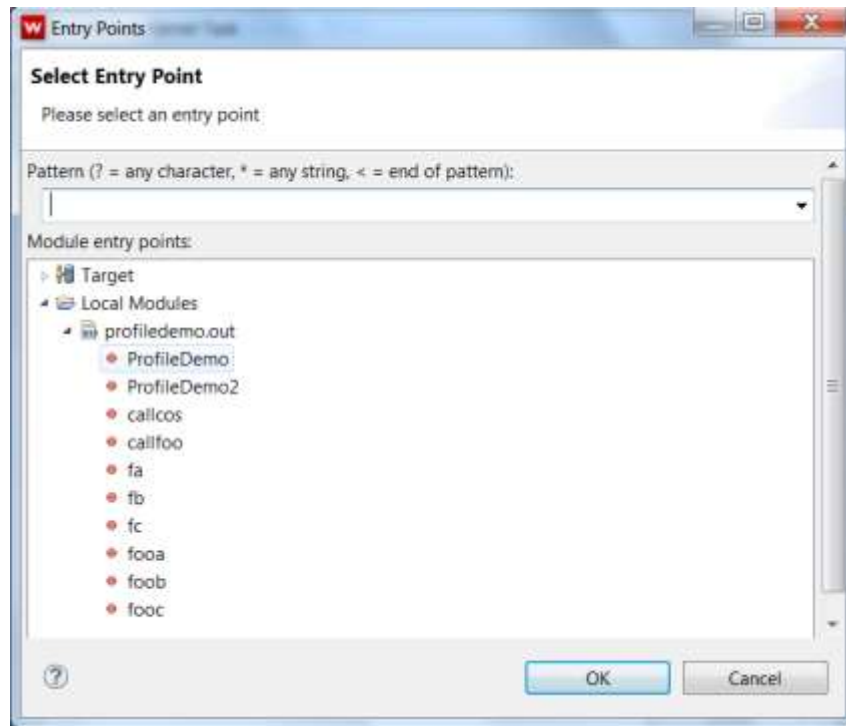
8. Now click Finish
9. You should now see the project in the Project Browser



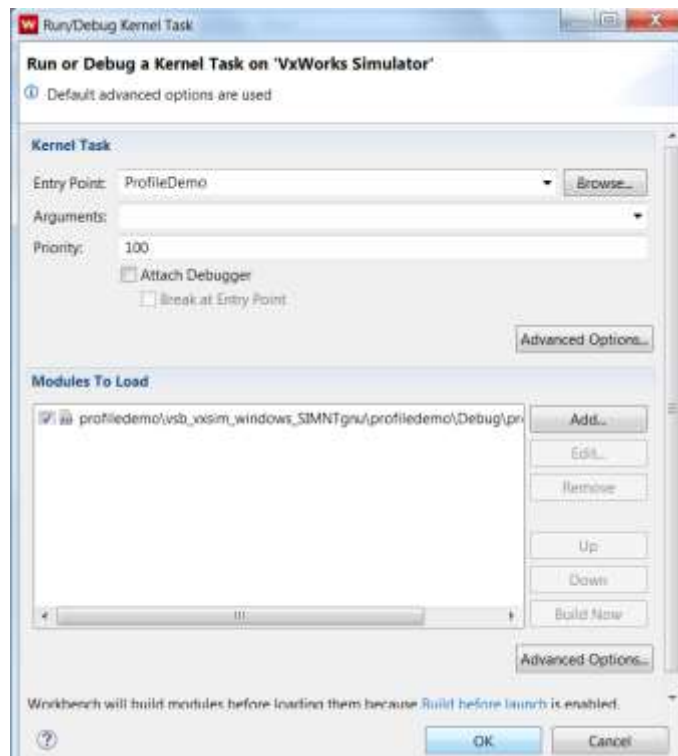
14. Now that you are connected, you'll want to download and run the demo. Go back to the **System Development** view, Right Click on the project, and select **Run/Debug Kernel Task**.



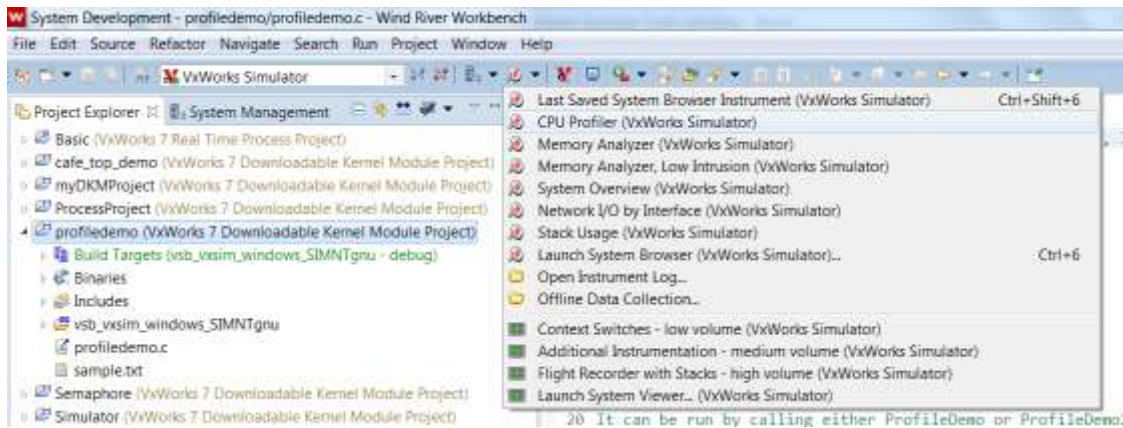
15. This will bring up a dialog box asking for the entry point, if you select the Browse button you can select ProfileDemo, this will be where you start your code execution.



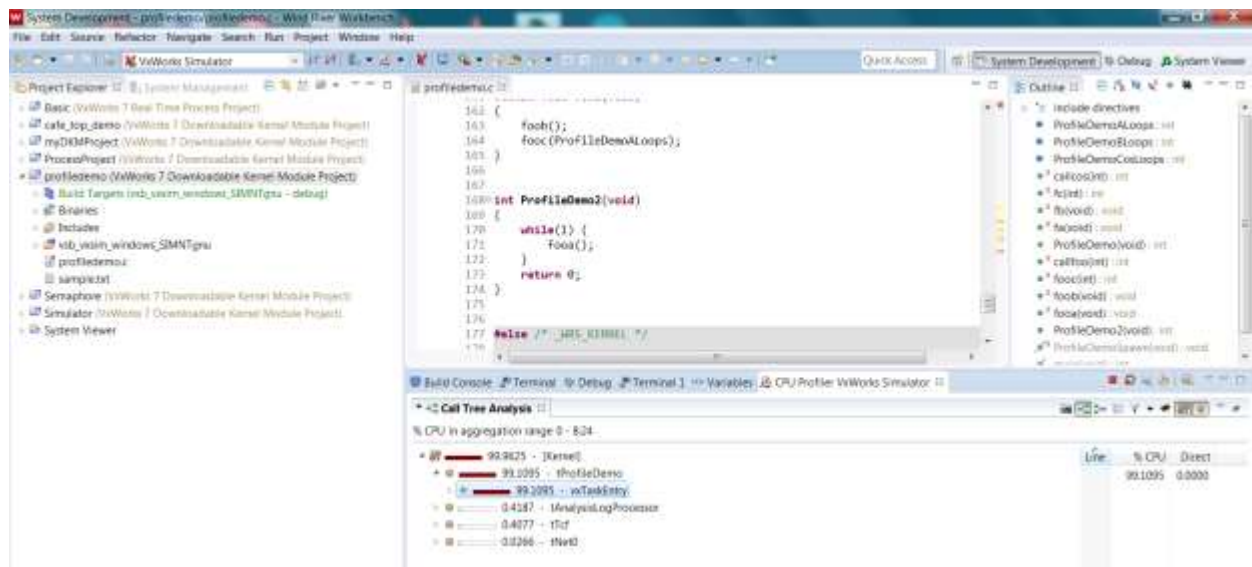
16. Now click OK and you'll be ready to start looking at the profile of your code.



17. Now you'll want to select the CPU Profiler (VxWorks Simulator) from the instrumentation menu:

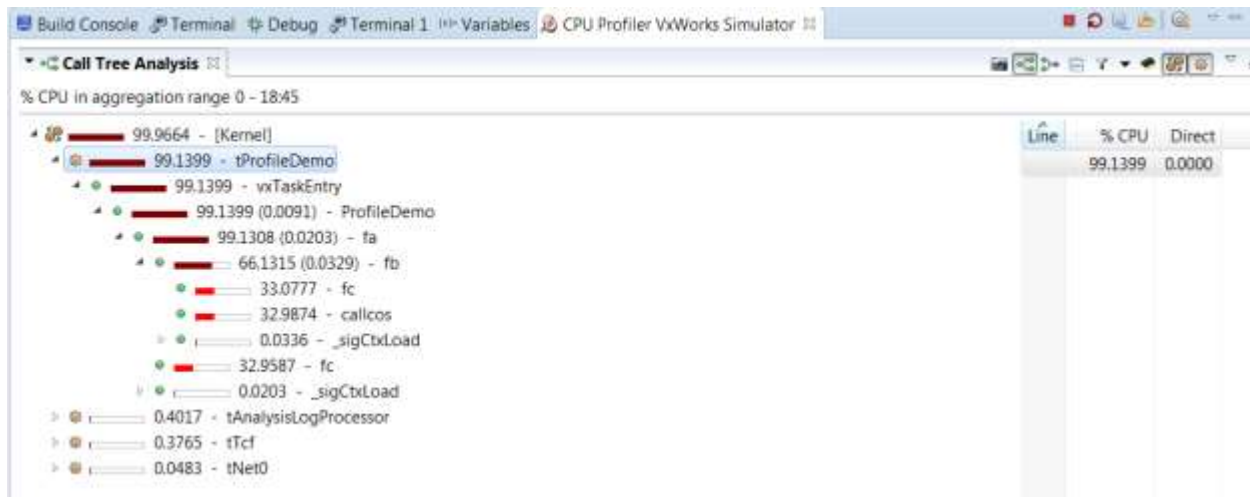


18. Here is what you should see:



19. This program shows how CPU intensive routines are identified using Wind River System Analysis.

20. You can drill down to see more specifics by selecting each task, ultimately it can look like this:



21. Advanced operations: - Switch between hot-spot and top-down views in the profiler - add the history graph for looking back in time - Right-click the profiler tree to replace with different metrics (cache misses, branch misses, ...)