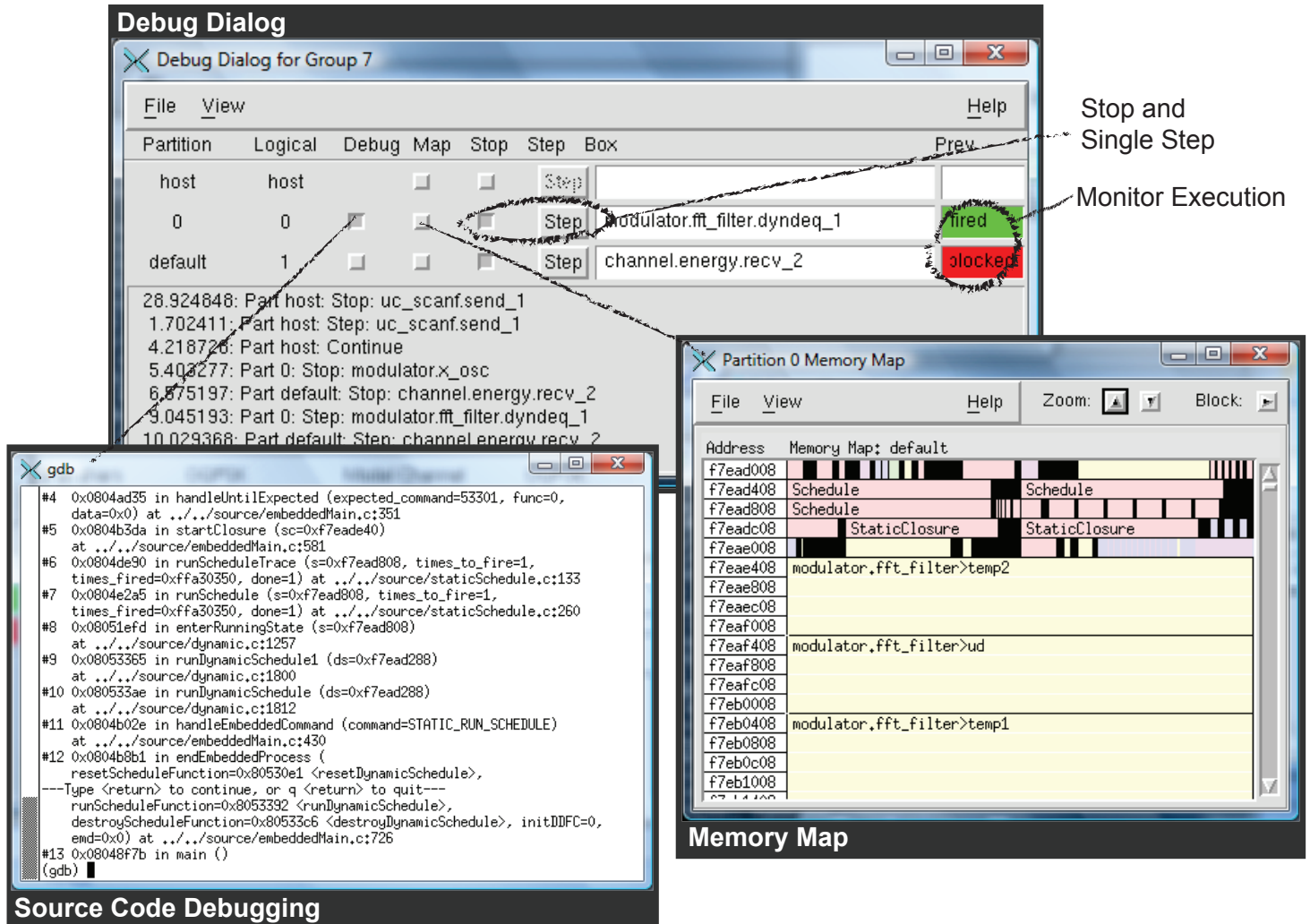


# Gedae Distributed Debugging

## Executive Summary

Gedae provides powerful tools for debugging your distributed software. Traditional debuggers only provide help for addressing the problems of programming a single thread on a single processor. Gedae's tools have been designed to address the programming issues of a multithreaded, multiprocessor application.



## Debug Dialog

The Debug Dialog provides a centralized control panel for debugging distributed software. Source code debuggers can be launched directly from the dialog for each processor element. Execution of individual threads can be stopped, single stepped, and restarted. Breakpoints can be set based on a variety of data flow events, including:

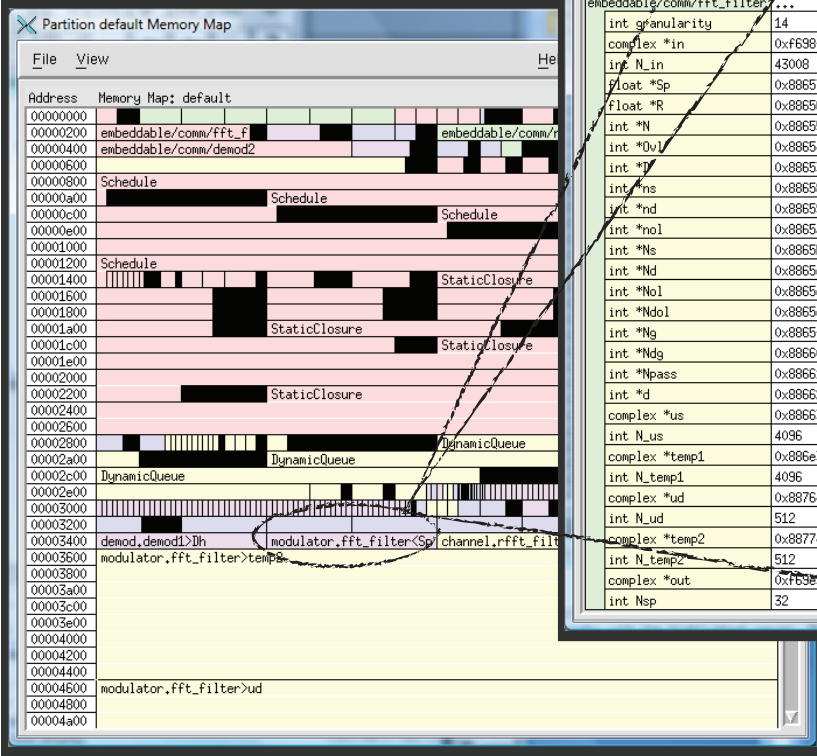
1. When a particular primitive executes

2. On the Nth execution of a particular primitive
3. The beginning (reset) or end of a stream segment
4. When data is produced/consumed on a particular output/input
5. When N tokens of data have been produced/consumed on a particular output/input
6. When a particular data queue becomes full or other queue-need-based conditions
7. On the failure of a particular primitive box
8. On a Trace Table event, including user-generated events

## Memory Map

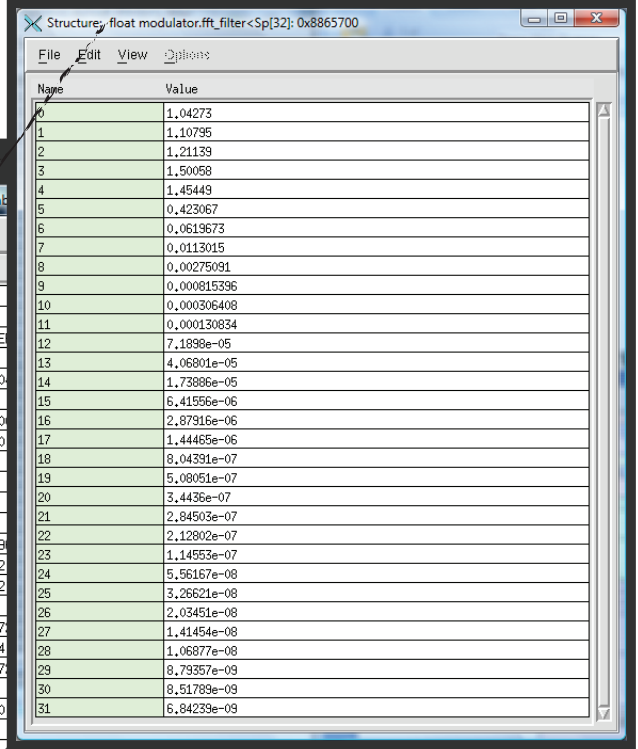
Gedae provides a variety of tools for observing the layout of memory and its data values. Each target core or processor has a Memory Map. The map is the plan of the memory usage on the target core. The memory map can be interactively browsed, zooming in to see smaller data blocks like individual parameters and querying the blocks to view their data structures and values.

### Memory Map

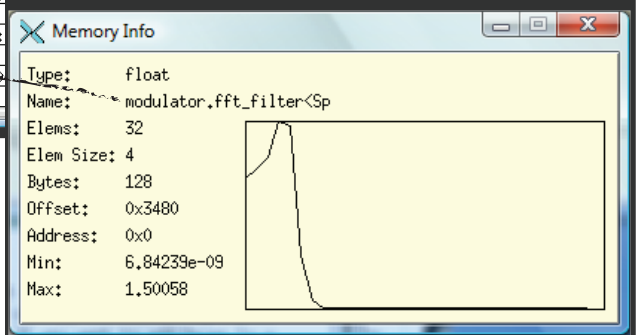


Values in the Memory Map can be displayed in various ways, including by data structure, by value and by plot.

### Array Information



### Pointer Information



## Data Flow Debugging

Gedae provides a comprehensive set of tools to help users detect and fix data flow problems. Using these tools, common distributed processing problems, such as blocking and starvation, can be understood and fixed. After Gedae indicates a data flow error in the distributed application, the Flattened Graph helps visualize the error condition. The Flattened Graph displays the full flow graph in one window and provides many methods for color coding primitives, arcs, and I/Os to help debug data flow issues. Many data flow problems can be visualized in the display, including issues encountered both inside a single thread and between distributed threads.

In this case, blue lines indicate blocking, green arcs indicate starvation, and purple arcs indicate a data flow error.

### Flattened Graph

