Staying ahead of the multi-core revolution with CDT Debug

Patrick Chuong, Texas Instruments

Dobrin Alexiev, Texas Instruments

Marc Khouzam, Ericsson Canada

Agenda

- The multi-core problem
- Multicore Debug Workgroup
- New features
  - Multi-process
  - Pin&Clone
  - Enhanced breakpoints
  - Grouping
  - Visualizer view
- Plans
- Live Demos

Multi-core problems – scalability and complexity

- Scalability: more CPUs, DSPs, processes, threads...
- Complexity of user tasks:
  - Users still focus mostly on few entities –CPUs, threads
  - Sometimes they need to see the whole system: all CPUs, all threads
  - Users need to be able to customize their view of the system
    - Group or hide CPUs, threads, types of nodes
      - step these threads simultaneously
      - set breakpoints that apply only to these CPUs
    - Define different layouts that fit their current task
      - See the system in terms of JTAG connectivity, power
      - Separate system threads from user threads
- Debug View is getting crowded…
  - but still is the main view for debugging the system

Multi-core – solutions

- Ways of managing complexity and scalability
  - The users can switch between multiple layouts in the Debug View
    - JTAG layout, Power layout, CPU affinity layout
  - The user can hide nodes or node types in the Debug View
    - JTAG connections, all processes nodes, system threads, etc.
  - The user can group nodes in the Debug View
    - Stepping the group node will step all threads in the group
    - Breakpoints can be added to all threads in the group
  - Display the system in multiple views simultaneously
    - Visualizer view, Status view, etc.
    - The user can perform synchronized run control operations
    - The selection in these views can drive the data displayed in the other debugger views: Registers, Variables, Memory, etc.

CDT's Multicore Debug Workgroup

- Joint effort by people/companies interested in bringing 'multicore' debugging to the CDT
- Goals
  - Provide a good user experience for multicore debugging in CDT
  - Determine debugging features of interest
  - Collaborate to bring each feature to CDT: design, implementation, review, test.
- History
  - Workgroup was first proposed at the CDT Summit 2010 and was created in early October 2010
  - Work began with the first conference call on November 16th.
  - 10 different companies are regular participants

CDT's Multicore Debug Workgroup

- Wiki page can be found in the "Workgroups" section of the CDT wiki, or directly at:
  - http://wiki.eclipse.org/CDT/MultiCoreDebugWorkingGroup
  - Conference Call scheduling
  - Completed features
  - Features currently of interest
  - Other proposed features

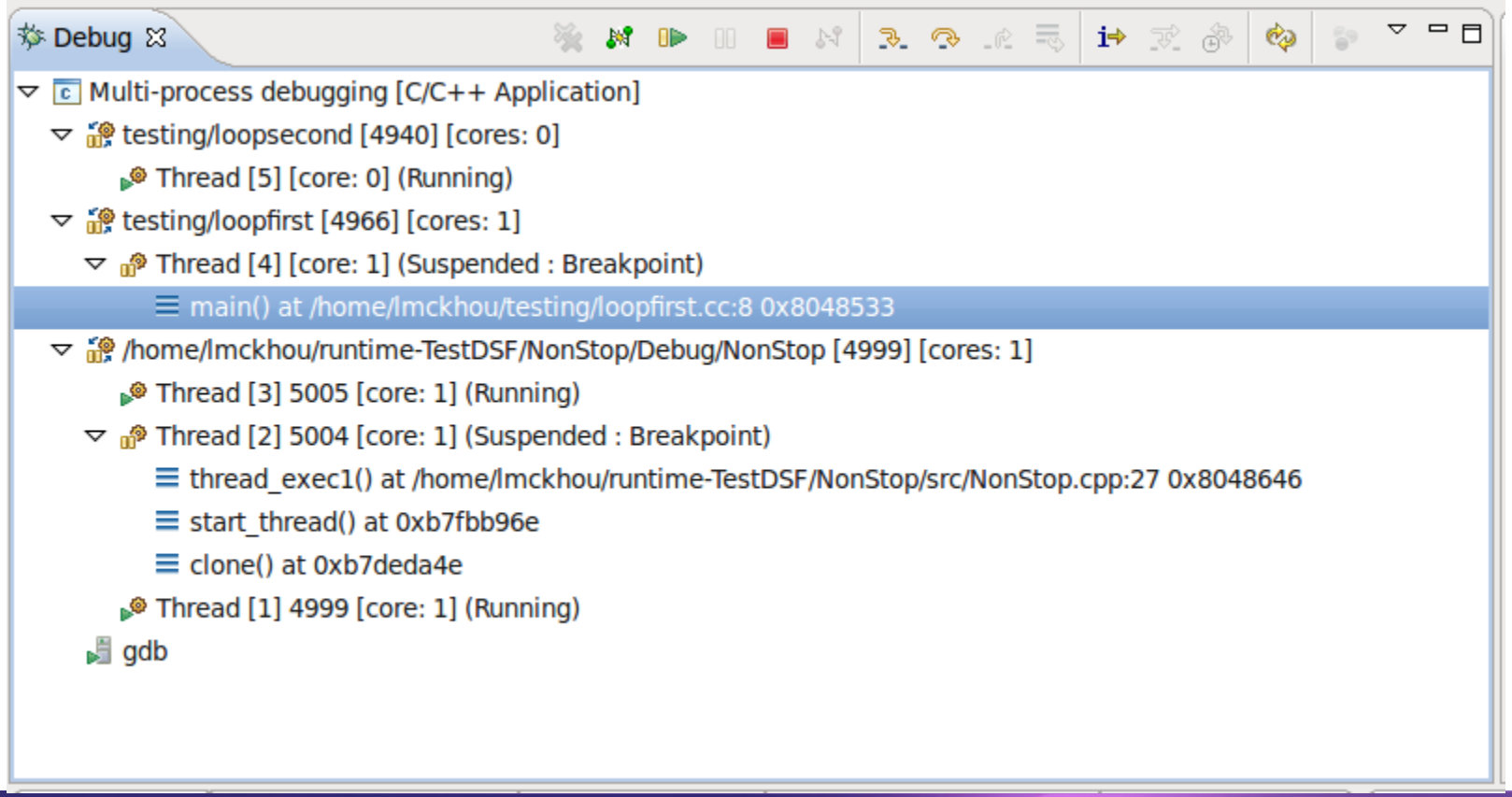
- Conference calls are held every two weeks with minutes of meetings posted on the wiki
- Open to anyone interested (its free ☺)

New features

- **Multi-process**
- Pin & Clone
- Enhanced breakpoints
- Grouping
- Visualizer view

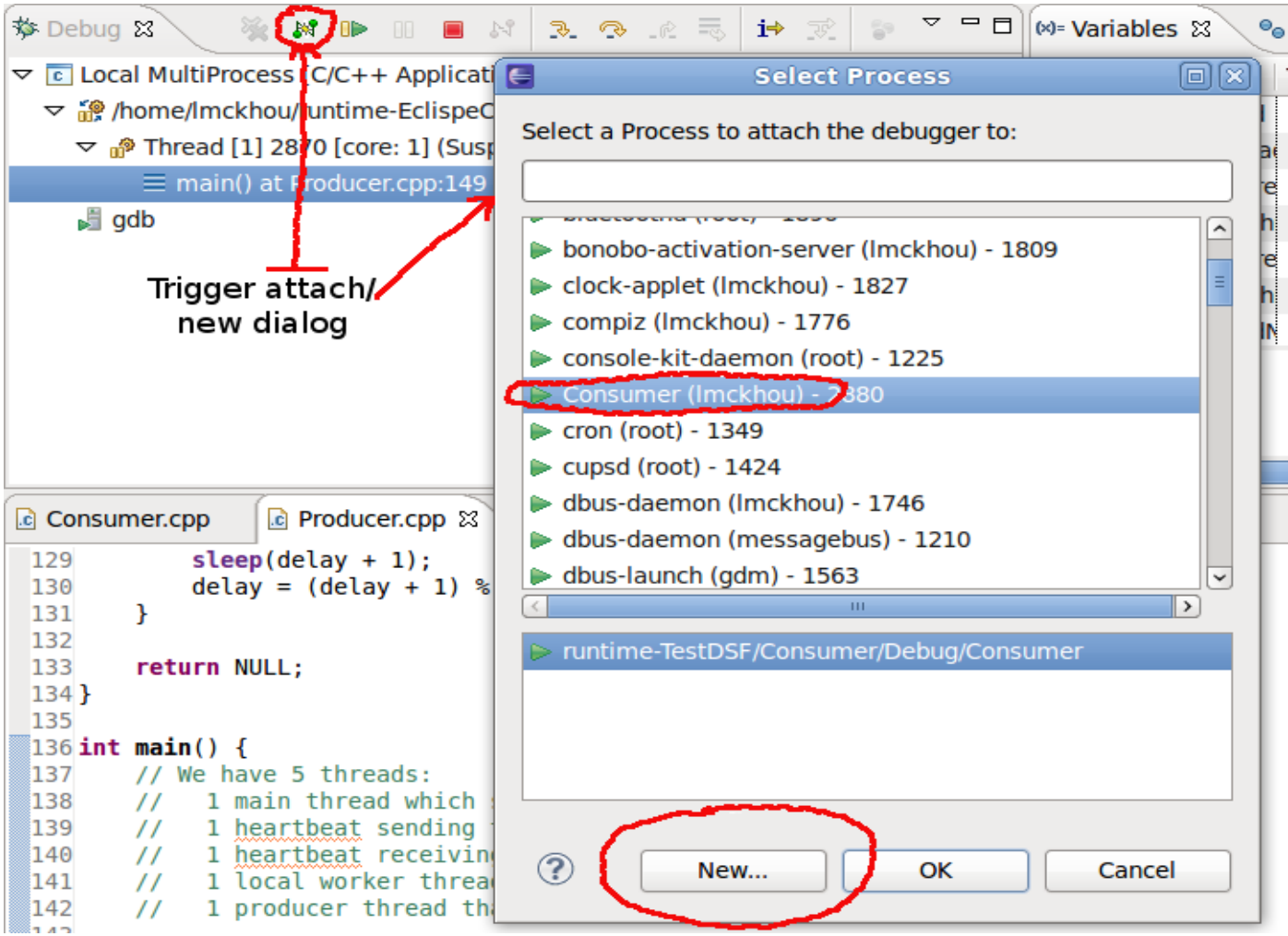
Multi-process debugging

- Allows to debug multiple processes of a target in the same session
- Good for debugging process interactions
- Now available in CDT, for Linux targets when using GDB 7.2



Multi-process debugging

- Combines debugging of running processes and new processes
- Works for local debugging and remote debugging
- Will work in both non-stop or all-stop modes



New features

- Multi-process
- **Pin & Clone**
- Enhanced breakpoints
- Grouping
- Visualizer view

Pin & Clone - Background

- What is Pin & Clone?
  - **Clone**: enable multiple debug view instances in the same workbench window i.e Variables
  - **Pin**: attach a debug view instance to a set of debug context i.e Threads

- **Problem**: ability to compare data from multiple processes/threads

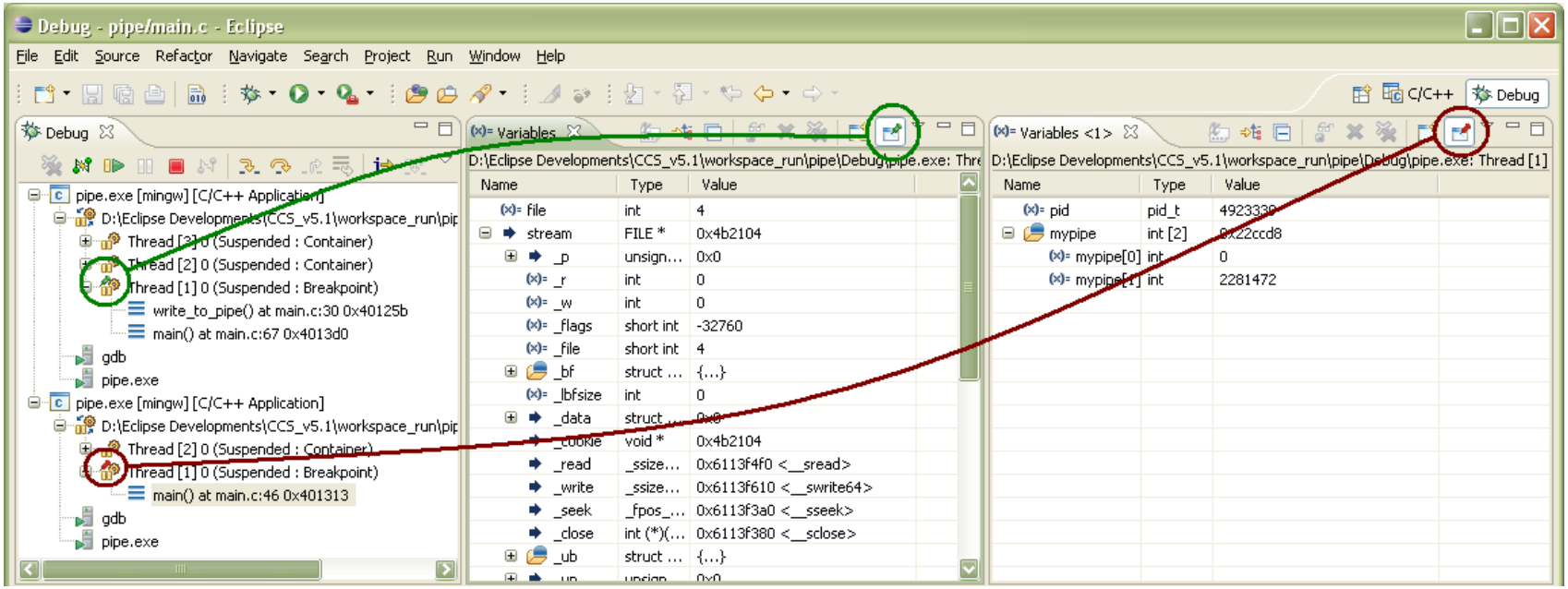
- **Solution**: open multiple view instances and attach the view instances to a set of debug context

Pin & Clone - Features

- Debug views that support Pin & Clone
  - Variables, Expressions, Registers, Memory Browser, and Disassembly
- Main feature set
  - Views that are pinned will have the pin context(s) label shown in it's description area and the pin toolbar bar icon will have matching overlay icon in the Debug View
  - View toolbar icon shows multi-pin contexts when more than one debug context is selected, extendable by the backend
  - View tab label will be numerically indexed
  - Pinned context(s) will reattach itself when a launch is terminate and restart
- Available in CDT 8.0 M6 for DSF-GDB backend
- Extendable and customizable by debugger backend

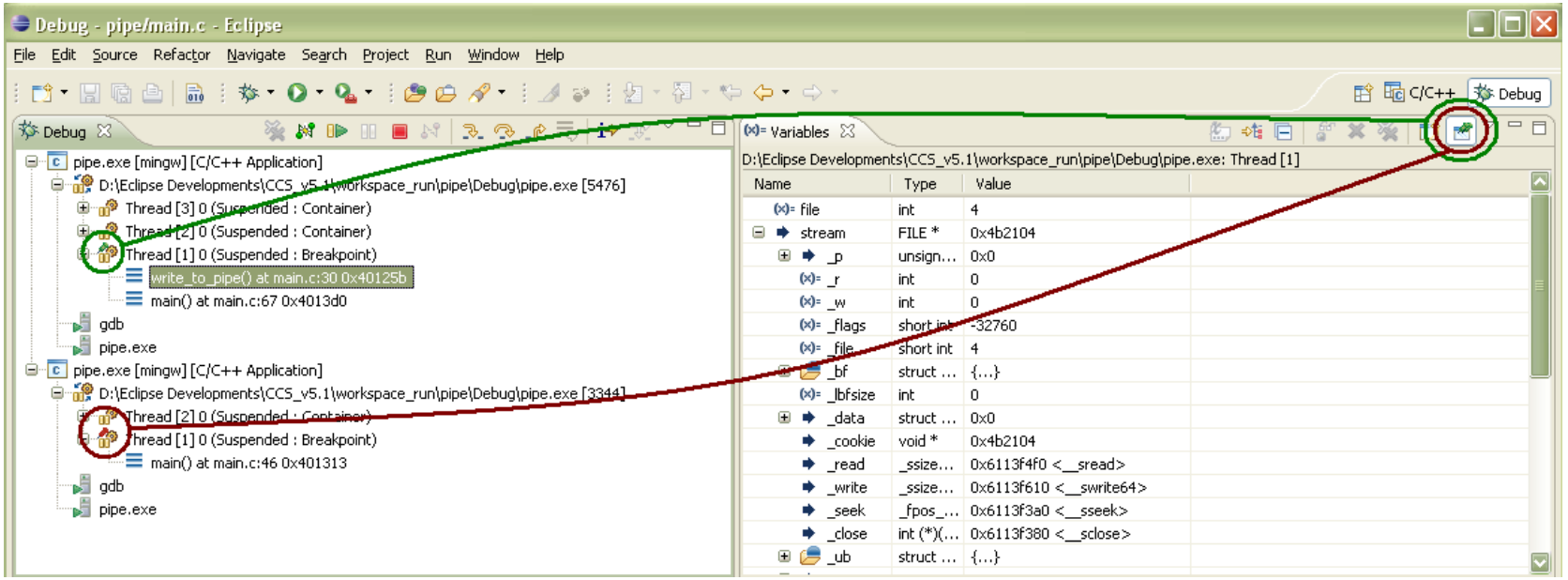
Pin & Clone – Multi-View Instances

- Compare data between two debug sessions
- Two Variables views
  - Views are pinned to a thread in each debug session



Pin & Clone – Multiple pinned contexts

- Limit a view to a sub-set of context
- One Variables view
  - View is pinned to two threads, switching between these two thread will cause the view to update



New features

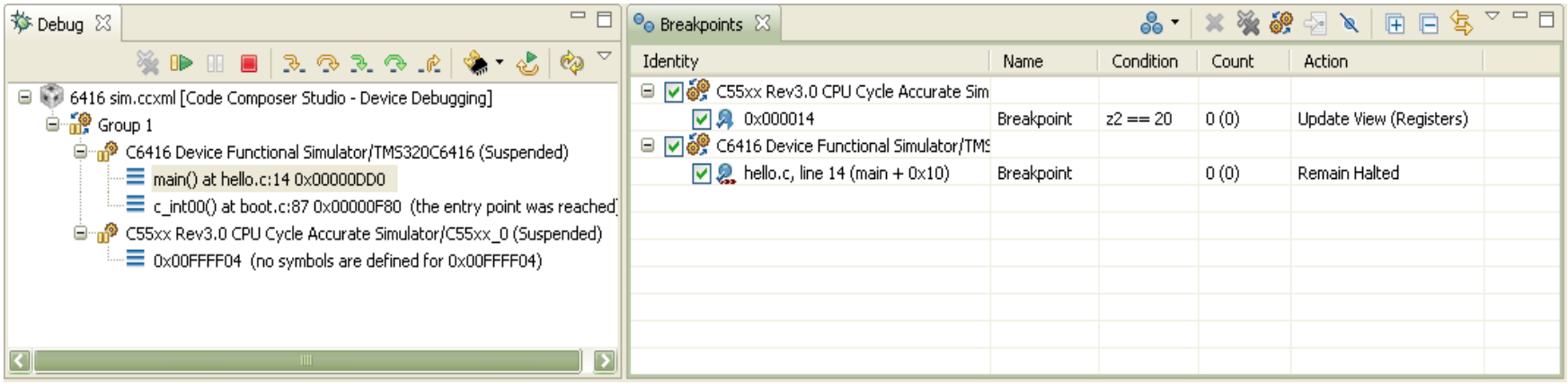
- Multi-process
- Pin & Clone
- **Enhanced breakpoints**
- Grouping
- Visualizer view

Enhanced Breakpoint Support

- Current limitations:
  - Installed for all threads/cores
  - Can’t configure properties before install to backend
  - Not scalable, restricted UI
    - New h/w capability isn’t dynamically exposed in UI
    - Does not make use of flexible viewer from Platform Debug
  - No indication which thread/core the breakpoint is hit

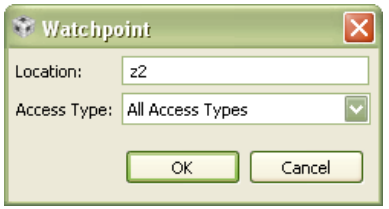
Breakpoint View Example: TI UBM

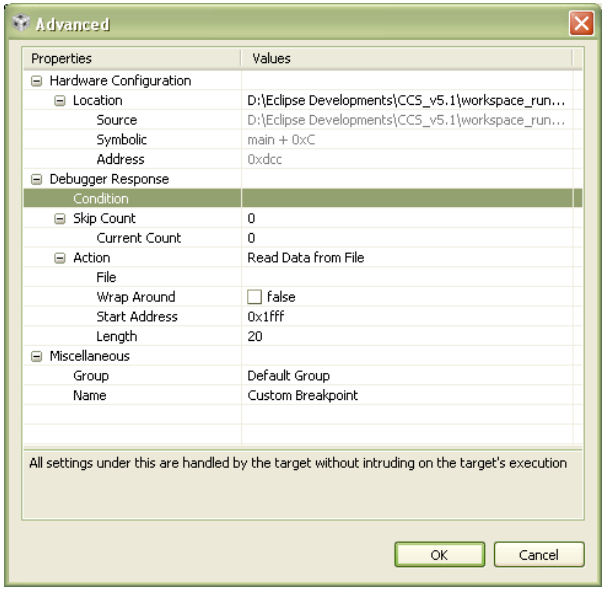
- A debug session with two cores
- An address breakpoint is installed on C55xx core
  - With condition z2 == 20 and action to update Register view
- A source line breakpoint is installed on C6416 core
  - With action to Halt (suspend)
- Marker icon with >>> indicates breakpoint has recently hit

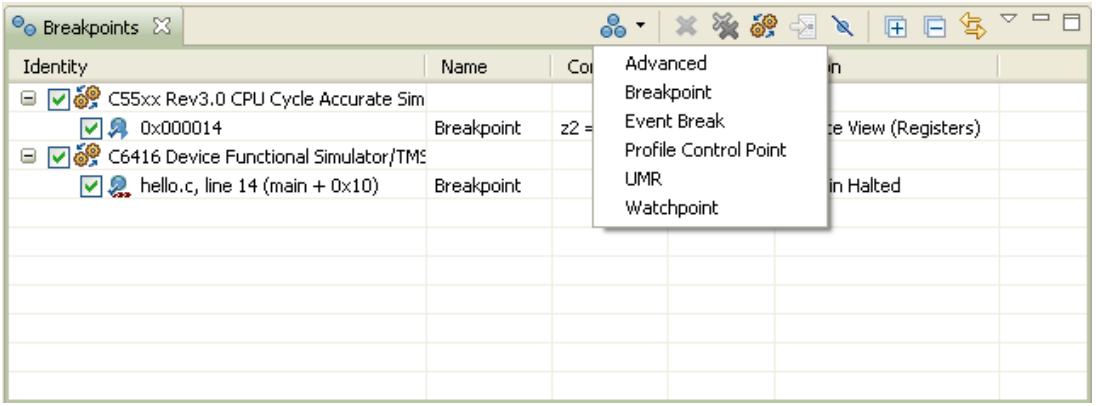


Create Breakpoint Example: TI UBM

- Available supported breakpoint type menu for the active debug context i.e Thread
- New Watchpoint type dialog
- New Advanced type dialog
  - Dynamic breakpoint properties
- Contextual breakpoint type menu support for Editor, Disassembly view, Project view, Outline view, etc…







New features

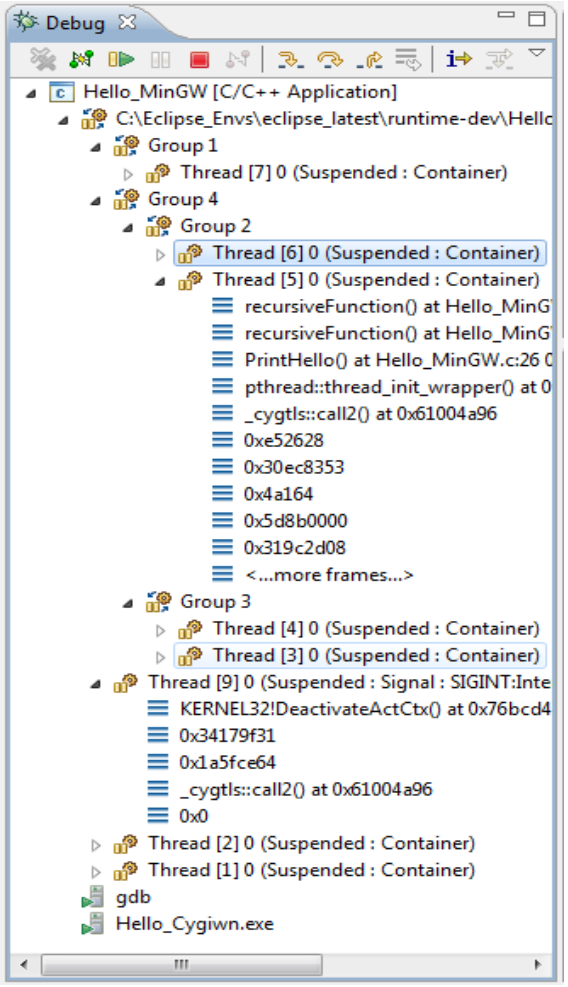
- Multi-process
- Pin & Clone
- Enhanced breakpoints
- **Grouping**
- Visualizer view

Debug View User Groups

- Why?
  - focus on few entities –CPUs, threads
  - step multiple threads simultaneously
  - set breakpoints that apply to multiple CPUs
- How?
  - choose the Debug View context menu “Group” when multiple nodes are selected. A group is created as parent of the selected nodes
  - stepping the group node will step all threads in the group
  - the group icon will show the state of all threads in the group
  - the user can ungroup threads previously grouped
  - the groups will be persisted between debug sessions
  - groups can contain other groups

Debug View User Groups – integration

- Available for CDT-DSF debuggers
- Currently part of the code is in DSF common layer, part is in the DSF-GDB debugger.
- The common layer enables other DSF debuggers to integrate the feature easily.
- Different backend capabilities can be provided:
  - Some backends can support the user groups, for some the user groups will be transparent.
  - For some backends stepping, running or suspending groups will be different than the same command issued to multiple threads.



New features

- Multi-process
- Pin & Clone
- Enhanced breakpoints
- Grouping
- **Visualizer view**

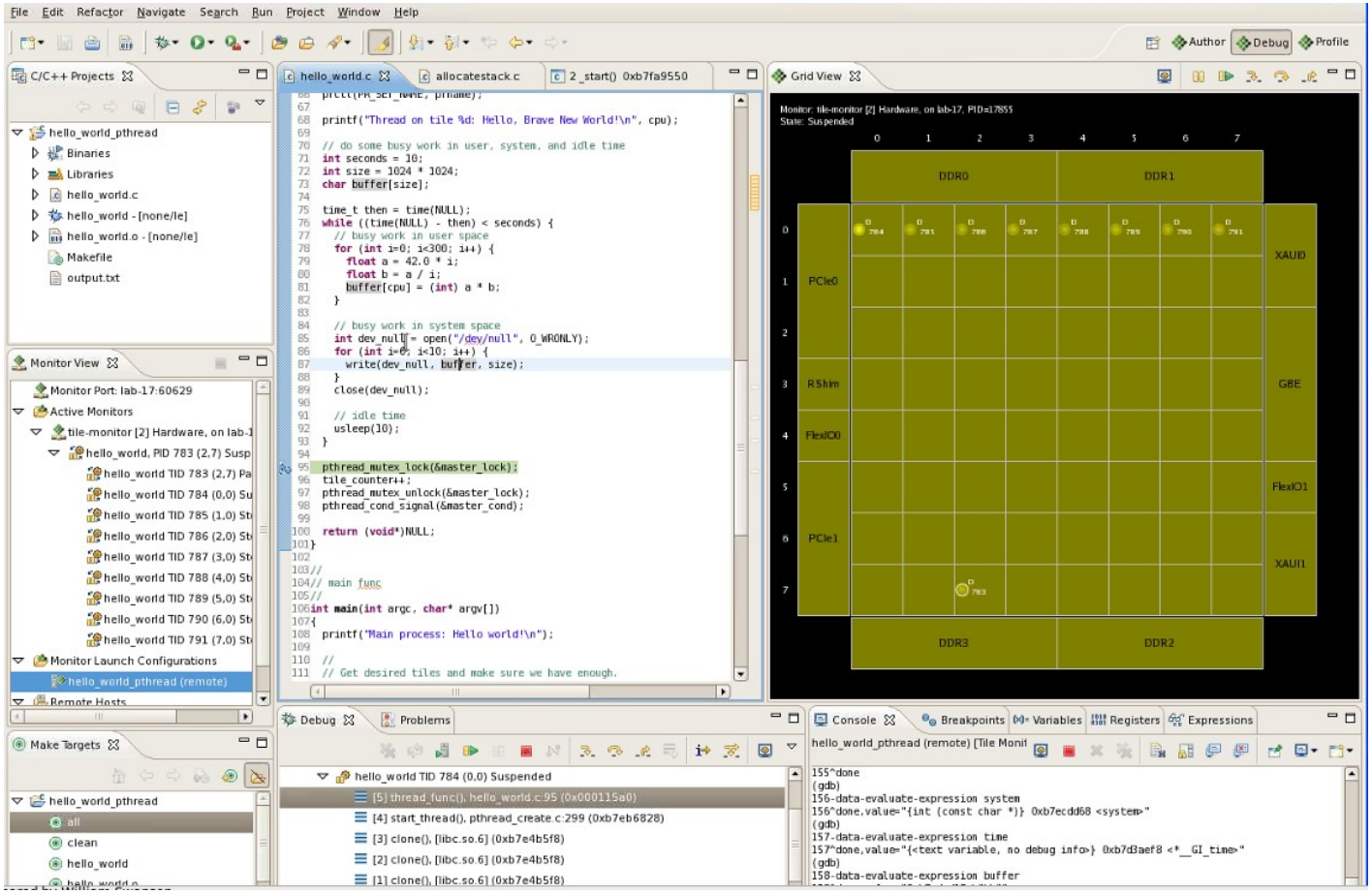
Visualizer View

- Eclipse Debug View centric to debug session
- Debug View has limitations
- Need for a graphical debugging view: Visualizer View
  - Complement to the Debug View
  - Efficient representation of a large amount of data
  - Quick visual access to system state

- Meant to be a framework to allow for different graphical representations
- Design has been started and posted to the wiki

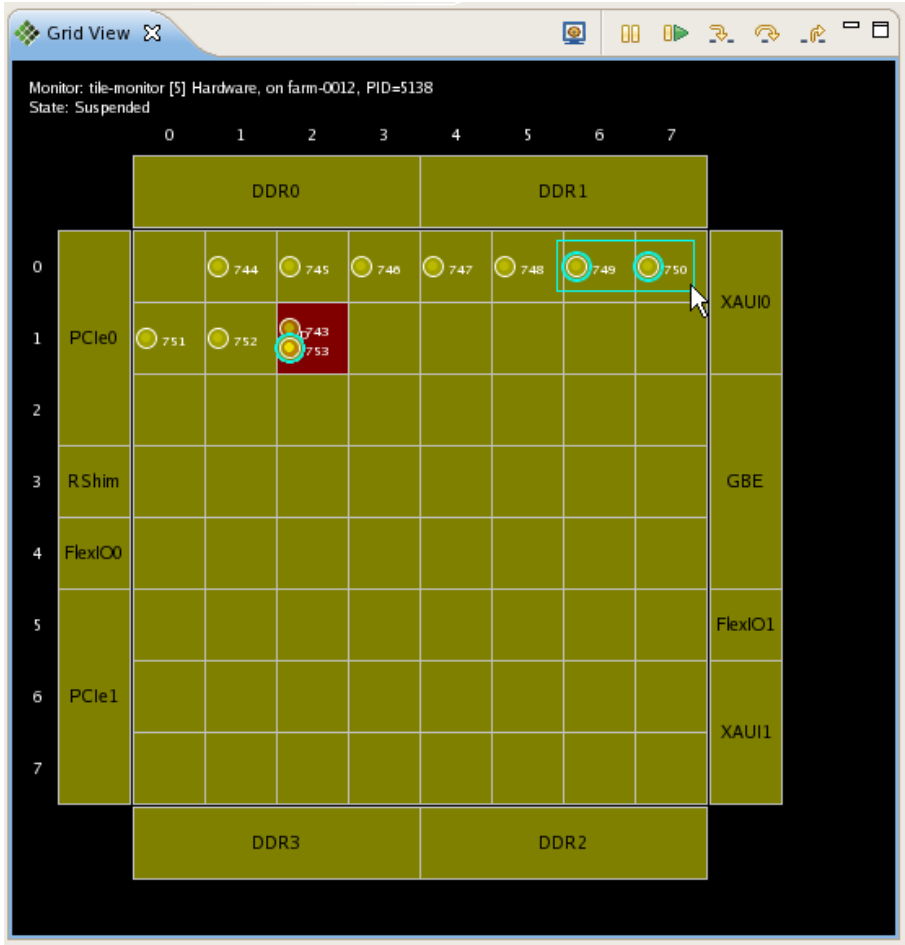
Visualizer View Example: Tilera's Grid

- An instance of a representation. Hardware-centric.
- Part of the goals of the CDT Visualizer View



Visualizer View Example: Tilera's Grid

- Color indicates application state
  - yellow = stopped on a breakpoint
  - red = process crash
- Dots are the processes
- IO and Memory shown on the edges
- Drag selection of processes/threads
- Allows to control execution
  - Resume/Suspend
  - Step
- Selection in the Grid is reflected in Debug view and all other views



Plans

- Indigo CDT 8.0
  - Pin & Clone Support (completed)
  - Multi-Process (working support)
  - Group & Ungroup (preliminary support)
  - Synchronized run control operation
- Post Indigo CDT 8.0+
  - Graphical Visualizer view
  - Enhanced breakpoint support
  - Hiding of debug view elements
  - OS-awareness
  - Global breakpoints

Points of Contact

- CDT Mailing list:
  - cdt-dev@eclipse.org
- CDT Wiki:
  - http://wiki.eclipse.org/CDT
- Multicore Debug Wiki:
  - http://wiki.eclipse.org/CDT/MultiCoreDebugWorkingGroup

Demos

- Multi-process
- Pin & Clone
- Grouping

Questions?