RTEMS Project Ecosystem

Joel Sherrill, Ph.D.

Joel.Sherrill@oarcorp.com
OAR Corporation
Huntsville Alabama USA

December 2014
Ecosystem Background

• Objectives:
  – provide a fully repeatable build from source to a tested, working application base.
  – lower entry barrier and increase host support
  – consistent user interface

• Definition:
  – framework that provides a collection of tools to complete this process

• Status:
  – ongoing and active work, evolving to meet needs
Environment Reproducibility

• User produce and test RTEMS and tools from source
  – complete source
  – RTEMS Source Builder
  – RTEMS Tester

• Same infrastructure used by
  – developers for developmental testing
  – project for Continuous Integration Testing

• Plan to publish results so user can compare their results against official results from RTEMS Project

http://www.rtems.org
Ecosystem Deliverables

• Host tools
  – development tools, simulators, etc. with source, patches, and locally built binaries

• Target libraries
  – RTEMS and third party add-on libraries source, patches, and locally built BSP specific binaries

• Reports
  – configuration, build, and test

• Support smooth integration of RTEMS into project specific configuration management

http://www.rtems.org
Hosting and Infrastructure Refresh

- Half-rack of new hardware purchased with donations for GSoC and GCI participation
- Hosting now at Oregon State University Open Source Laboratory (https://osuosl.org/)
  - couple of hops off Internet2 backbone, very fast
- Multiple single purpose VMs versus multi-purpose single host
  - this did cause some URLs to break
- Conversion from Bugzilla and Mediawiki to Trac
  - Trac provides release status tracking and reports
  - commit closes ticket, sends email to list, updates github, and sends message to IRC
- Future Plans
  - need another build machine dedicated to building tools
  - need more RAM and storage for machines (purchased with space to grow)
  - Mac for testing as host

All volunteer effort! Please pitch in!
Process Refresh

• New infrastructure enables new capabilities
• Future plans
  – incorporate web based patch review and submission system
    • eventually a patch will be tested before a human sees it
  – establish continuous integration and testing so every commit results in an appropriate test sweep
  – provide database of test results
  – refresh user documentation in modern tools
    • current effort to review content in Wiki

Anything else we use computers to help with

http://www.rtems.org
RTEMS Tools Project

RTEMS Users

RTEMS Developers

Continuous Integration Testing

CLI

GUI

RTEMS Source Builder (RSB)

RTEMS Tester

RTEMS LD

RTEMS Trace Linker (TLD)

Capture Trace

Coverage Testing and Reporting

Config

Macro Expansion

ELF/DWARF

Symbol Management

RAP

INI

Izma

http://www.rtems.org
RTEMS Source Builder (RSB)

• RSB contains “recipes” for building host tools and target packages from source
  – enables configuration management of source and patches
• Host independent way to obtain environment
• Full source and patches
• Use RSB to fetch source and build
  – RTEMS tools
  – cross development environment
  – RTEMS (or by hand)
  – simulators if desired
  – third Party Packages for target system

http://www.rtems.org
RSB Example

• Obtain RTEMS RSB from git or release
  – `git clone git://git.rtems.org/rtems-source-builder.git`

• Check basic host environment
  – `cd rtems-source-builder`
  – `source-builder/sb-check`

• Build SPARC toolset
  – `cd rtems`
  – `..//source-builder/sb-set-builder --log=l-sparc.txt \`
    --prefix=$HOME/development/rtems/4.11 4.11/rtems-sparc`

• Takes approximately 17 minutes on a (modest) quad-core machine

http://www.rtems.org
RTEMS Tester

- Automates testing on simulators and hardware using GDB Machine Interface (MI)
  - direct execution using various simulators
  - takes advantage of multiple cores to parallelize simulator testing
- Execute RTEMS Test Suite
  ~/development/rtems/test/rtems-tools.git/tester/rtems-test \
  --log=log_sis_run --rtems-bsp=sis-run \
  --rtems-tools=$HOME/development/rtems/4.11 \ 
  sparc-rtems4.11/c/sis/testsuites
- Takes ~24 minutes single threaded but 11.3 minutes with tester on modest 2.4Ghz quad code
  - Note: end up waiting for longest to complete or timeout
- Future plans
  - reduce run-time of longest running tests
  - automatically calibrate performance of simulators and scale the timeouts
  - execute GCC Test Suite
  - execute tests on other components
  - compare results with RTEMS Project
    - requires RTEMS Project to provide a populated test results database

http://www.rtems.org
RTEMS Coverage Testing

• Tools to analyze and report RTEMS Test Coverage
  – primary tool is covoar

• Test coverage itself is in good shape
  – Improved reporting and integration into framework is needed

• Current status
  – original scripts not in framework. ESA SOCIS student rewrote initial version for framework
  – original scripts had rigid reporting that was not granular enough. New version addresses that

• Future plans
  – bring framework version up to production standards so they can be integrated
  – finer grained (e.g. by directory) coverage reports
RTEMS and Tools Milestone

- Tested RTEMS Environment in place
- Provides fully tested application base with
  - full source and binaries
  - configuration reports
  - test reports including coverage
  - build reports

- RTEMS user can use ecosystem to support their project

http://www.rtems.org
RTEMS Continuous Integration Testing

• RSB provides uniform way to build tools, infrastructure, RTEMS and add-on packages
• Eases integration into a Continuous Integration Framework
• Prototype buildbot instance demonstration

• Future Plans
  – web based interface for submitting patches
  – patches tested BEFORE seen by reviewers
  – full test results available to entire community

http://www.rtems.org
Target Focused Tools

- Event tracing and recording
  - capture engine
  - trace wrapper generation
- Run-Time Loader
- GDB Pretty Printers

- Most of these capabilities require combination of target and host side software

http://www.rtems.org
Capture Engine

• Capture trace data from
  – user extensions: thread create, start, restart, delete, switch, begin, exit, and terminate
  – any method invocation with trace wrapper generation

• Trace information is buffered on a per CPU basis to avoid locking during recording

• Can be printed via CLI or transmitted to host computer for further analysis

• Future plans
  – improve trace filtering and triggering

http://www.rtems.org
Capture Engine Usage

- Requires host and target support
- Run-time captures and buffers trace records
- Target has API and CLI to manage trace records
- If buffers transferred to host, other tools can be used to examine records
Thread Migration Capture Example

TaskName:Priority:affinity

affinity = \{cpu,cpu\}

(1) UT1:7:{2,3}
(2) TA1:8:{2,3}
(3) TA2:5:{0,1}
(4) TA3:6:{0,3}
(5) TA4:9:{1}

(6) Raise priority of TA4:4:{1} & trigger series of migrations

Capture Engine Output

<table>
<thead>
<tr>
<th>CPU</th>
<th>TaskID</th>
<th>Name</th>
<th>Priority</th>
<th>Ready Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>TA1</td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>TA2</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>TA4</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>UT1</td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>TA3</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0a10003</td>
<td>TA2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0a10004</td>
<td>TA3</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0a10002</td>
<td>TA1</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

http://www.rtems.org
LTTng/RTEMS Proof of Concept #1

- Aeroflex/Gaisler PoC of MTAPI Demo on NGMP
LTTng/RTEMS Proof of Concept #2
RTEMS Trace Linker

• Eases linking RTEMS applications
• Automated generation of method wrappers
  – user specifies methods to wrap and trace
  – user specifies pattern for wrapper methods
• Proof of concept has pattern which uses `printk()` to display method invocation and return values
• Future plans
  – wrapper pattern for method trace to capture logs
  – extend filter/trace for arbitrary sets of user traces

http://www.rtems.org
Trace Visualization

• GUI visualization of event timeline
• Current status
  – identified existing FOSS tools focused on this
  – discussed with Linux Trace Toolkit developers how to add support for RTEMS trace visualization

• Future Plans
  – framework for mechanism to get trace to host
  – support Common Trace Format (CTF)
  – provide mapping of RTEMS events/objects to visual representation

http://www.rtems.org
RTEMS Run-time Loader

• Base image plus RTEMS Application (RAP) modules loaded and linked at run-time
  – uses host-based tool and the run-time link editor to perform a similar function to static linking
• Designed for modest target hardware
• File Formats supported by
  – ELF - normal ELF objects and executable
  – RAP - LZ77 compressed modules
  – AR – Object libraries with GNU extensions

• Future Plans
  – Support for most architectures is in place testing and feedback are needed

http://www.rtems.org
GDB Pretty Printers

• GDB now supports Python plugins
• Plugins can examine and analyze program variables and data structures
• Initial prototype to demonstrate feasibility
• Future Plans
  – user useful views of internal RTEMS information
  – methods to check consistency
  – replace RTEMS GDB macros with better versions written in a more capable language

http://www.rtems.org
Conclusion

• Work is ongoing and evolving

• Evolution is driven by feedback

• Help us make these tools make your job easier
Contacts and Acknowledgements

Joel Sherrill, Ph.D.
OAR Corporation
Huntsville Alabama USA
Joel.Sherrill@oarcorp.com

Gedare Bloom, Ph.D.
George Washington University
Washington DC USA
gedare@rtems.org

Chris Johns
Contemporary Software
Sydney Australia
chrisj@rtems.org
Backup Slides
Ecosystem Framework (figure)

• Bottom bar is framework
  – Features: configuration, macro expansions, ELF, symbol management, execute for parallelization, RAP (RTEMS Application Package) format, INI file, lzma

• Top bar is user interface, CLI and GUI dashed

• Above top bar: User, Continuous Integration Tester

• Vertical bars: RSB, RTEMS Tester, RTEMS LD, RTEMS Trace Linker (TLD)

http://www.rtems.org
Capture Engine API Example

- Set capture engine to capture and print tasks with priority 0-20 related to TA00 while the application “run_program” is executing.

```c
sc = rtems_capture_open (5000, NULL);
sc = rtems_capture_watch_ceiling (0);
sc = rtems_capture_watch_floor (20);
sc = rtems_capture_watch_global (true);
sc = rtems_capture_set_trigger (0,
  0,
  rtems_build_name('T', 'A', '0', '0'),
  0,
  rtems_capture_from_any,
  rtems_capture_switch);
sc = rtems_capture_control (true);
run_program();
sc = rtems_capture_control (false);
rtems_capture_print_trace_records (22, false);
```