RTEMS Status and Roadmap

Joel Sherrill, Ph.D.

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

October 2015
Overview

• Roadmap next few release series
• Information on 4.11 release series
  – for background information on RTEMS Ecosystem, see last year’s FSW presentation
• Improvements already merged post 4.11
• Desirable changes
• Hosting and infrastructure status and plans
• An important project announcement
Roadmap Highlights – 4.11 and 4.12

• 4.11 release series
  – branch cut
  – many improvements detailed later in slides
  – pending ftp site clean up

• 4.12 release series
  – remove obsolete architectures and BSPs
  – other improvements as they happen, detailed later in slides
  – last major release on autoconf/automake

http://www.rtems.org
Roadmap Highlights - 5.0

• Focus is on switching to waf based build system from Amar Takhar and process improvements
  – Create devel branch. This branch is open to development. Movement of source should be avoided if possible
  – Update Amar’s waf branch to master
  – Merge Amar’s waf branch onto master
  – Update all BSPs to build with waf
• Buildbot (http://buildbot.net) building all BSPs on each commit on master
• Integrate Phabricator (http://phabricator.org) into project workflow
• When ready, merge the devel branch onto master and delete the devel branch
• This enables any desired source tree restructure to occur
• Add more use of Buildbot:
  – building 3rd party packages (e.g. NTP, netsnmp, graphics, etc.)
  – building development tools (e.g. gcc, binutils, etc.)
• Stability testing

Note the proposed version number scheme
Numbering Past 5.0

- 5.0, 5.1, 5.2... are minor releases
  - comparable 4.10.0, 4.10.1, ...

- 6.0 will be a major release
  - 6.1, 6.2, ... will be follow minor releases

- This scheme has been adopted by GCC and other major projects

http://www.rtems.org
RTEMS 4.11 Features

- SMP is no longer experimental
  - SPARC up to four cores with LEON3 and NGMP
  - PowerPC up to 24 cores on QorIQ
  - ARM on Zynq, Cyclone V, Realview
  - x86 needs algorithm update in context switch

- New architecture ports
  - ehipany, moxie, nios2, or1k, sparc64, v850
  - Collectively added 12 new BSPs

- Many other new features
  - JFFS2, dynamic loading, tracing, warning removal, etc.

http://www.rtems.org
**New BSPs in 4.11**

- 59 more BSPs that in 4.10 even after removing some
- This table is ONLY BSPs added to existing ports

| ARM | altcycv_devkit, altcycv_devkit_smp, beagleboardorig, beagleboardxm, beagleboneblack, beaglebonewhite, lm3s3749, lm3s6965, lm3s6965_qemu, lm4f120, lpc1768_mbed_AHB_ram, lpc1768_mbed_AHB_ram_eth, lpc1768_mbed, lpc17xx_ea_ram, lpc17xx_ea_rom_int, lpc17xx_plx800_ram, lpc17xx_plx800_rom_int, lpc23xx_tli800, lpc24xx_plx800_ram, lpc24xx_plx800_rom_int, lpc40xx_ea_ram, lpc40xx_ea_rom_int, lpc32xx_mzr, lpc32xx_mzr_stage_1, lpc32xx_mzr_stage_2, raspberrypi2, raspberrypi1, realview_pbx_a9_qemu, realview_pbx_a9_qemu_smp, stm32f105rc, stm32f4, tms570ls3137_hdk, tms570ls3137_hdk_intram, tms570ls3137_hdk_sdram, xilinx_zynq_a9_qemu, xilinx_zynq_zc702, xilinx_zynq_zc706, xilinx_zynq_zedboard |
| I386 | edison, pcp4 |
| MIPS | malta |
| PowerPC | brs6l, dp2, br_uid, mpc8309som, qemuprep-altivec, qemuprep, mpc5566evb_spe, mpc5643l_dpu, mpc5643l_evb, mpc5668g, mpc5674f_ecu508_app, mpc5674f_ecu508_boot, mpc5674fevb, mpc5674fevb_spe, mpc5674f_rsm6, phycore_mpc5554, qoriq_core_0, qoriq_core_1, qoriq_p1020rdb, qoriq_t2080rdb, qoriq_t4240rdb, t32mppc, virtex4, virtex5 |
| SPARC | NGMP |

http://www.rtems.org
4.11 Tools Versions

• Cross development tools
  – GCC – 4.9.2
  – GNU Binary Utilities – 2.24
  – GDB – 7.9
  – Newlib – 2.20 snapshot from 20150423
  – Autoconf – 2.69
  – Automake – 1.12.6

• Qemu – 2.3.50

• NOTE: Some may have patches applied by RSB and some targets may not use these versions

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

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 to build cross development tools, supporting tools, and target libraries
  – provides source and reproducible results appropriate for configuration control
• RTEMS Tester to enable users to test on the own hardware or simulators
• 4.11 includes initial release or significant update of
  – RTEMS Trace Linker
  – RTEMS Dynamic Loader and RTEMS LD
  – RTEMS Capture Engine
• Possible: More granular coverage test reporting
4.12 Improvements in Place

• Addition of lightweight API to be used by infrastructure packages
  – GCC run-times, newlib, etc.
• C++11 and C11 support
  – focus is on concurrency support for RTEMS
• Newlib locking
• OpenMP supported and optimized
  – Two cores and 24 core configuration on T4240
  – See RTEMS Trac ticket #2274 for optimizations

http://www.rtems.org
OpenMP Optimization #1

• GCC Improvement from 4.9.3 to 6.0
  – optimized use of malloc()/free()

GCC 4.9.3 on Two Core T4240
• barrier bench 20.6147 seconds
• parallel bench 16.8791 seconds
• static bench 0.852061 seconds
• dynamic bench 0.292199 seconds

GCC 6.0 on Two Core T4240
• barrier bench 23.3409 seconds
• parallel bench 9.60804 seconds
• static bench 0.472419 seconds
• dynamic bench 0.223881 seconds
• guided bench 0.00999273 seconds
• runtime bench 0.229282 seconds
• single bench 2.18316 seconds

http://www.rtems.org
OpenMP Optimization #2

- RTEMS Improvement post 4.11
  - use new lightweight API
  - results are GCC 6.0 on 24 core T4240

Using pthread API
- barrier bench 783.888 seconds
- parallel bench 115.901 seconds
- static bench 5.7876 seconds
- dynamic bench 0.262251 seconds
- guided bench 0.0133215 seconds
- runtime bench 0.261378 seconds
- single bench 57.3227 seconds

Using new lightweight API
- barrier bench 5.74687 seconds
- parallel bench 2.38893 seconds
- static bench 0.118236 seconds
- dynamic bench 0.2516 seconds
- guided bench 0.00146854 seconds
- runtime bench 0.250789 seconds
- single bench 0.543456 seconds

- Most tests showed 10-100x improvement
- Two core configuration showed similar improvements

http://www.rtems.org
Goals for 4.12

• Removal of obsolete ports
• Removal of obsolete BSPs

• Improvements like OpenMP, C11, C++11 work will be included
• Other work included as it appears
• Last major release on autoconf/automake

Help us ensure no BSPs in use are removed
5.0 Changes

• Minimum goal conversion to waf build system

• Requires reorganization of include files to avoid preinstall phase of build
  – improves dependency tracking of builds

• Desirable to reorganize c/ subdirectory
  – make source easier to navigate and understand
  – shed legacy of not moving directories due to CVS

• Desirable to convert documentation to something more modern than texinfo
Waf Background

• Build system from Samba written in Python
  – https://github.com/waf-project/waf

• Computes full dependency tree, maximally parallelizes build using all available cores

• Already used on other RTEMS repositories

• Much faster than current build system
  – takes 12 seconds to 2 minutes from end of *git clone* to all tests built for sparc/sis depending on host OS and hardware
Desirable Changes

• RSB support for all ported third party packages
  – most known are in place now or patches posted
  – allows removal of two RTEMS git repositories

• TCP/IP stack desirable changes
  – More BSPs support new FreeBSD TCP/IP stack
  – More BSPs support port of LWIP
  – Current FreeBSD TCP/IP stack builds separate from RTEMS

• BSP Specific SMP Issues
  – Raspberry Pi 2: GSoC student worked on support, didn’t finish
  – Update x86 to account for SMP context switch locking algorithm and update IRQ to APIC
Hosting and Infrastructure Status

• Hosting now at Oregon State University Open Source Laboratory (https://osuosl.org/) using multiple single purpose VMs versus multi-purpose single host
  – this did cause some URLs to break
• Since 4.10, project has switched to git from CVS, Trac from Mediawiki and Bugzilla, and updated to latest Texinfo allowing us to eliminate our own tool
• Future Hardware Infrastructure 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!
Future Process Infrastructure Plans

• New infrastructure enables new capabilities

• Future plans
  – incorporate web based patch review and submission system (Phabricator)
    • 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 (Buildbot)
  – provide database of test results
  – refresh user documentation in modern tools

Anything else we can use computers to help with
Micromonitor

- Micromonitor or uMon ([https://git.rtems.org/umon](https://git.rtems.org/umon)) is an Apache licensed alternative to U-Boot developed over 15 years by Ed Sutter.
- was only recently relicensed to Apache
  - has history of use with RTEMS
  - now absorbed into the RTEMS Project
- GSoC student Jarielle Catbagan worked with Ed Sutter to move the target independent code into a new git repository and ensure it could be built by anyone
  - older, pre Apache licensed code at [http://www.umonfw.com](http://www.umonfw.com)
- Jarielle ported uMon to the BeagleBone Black
- Low level ARM CPU initialization specific to the BBB. This include DDR3 memory, UART console I/O, and eMMC/SD card.
- uMon capable of booting from eMMC/SD card and via UART
  - network support not exercised in new repository
- Current developments include Ethernet functionality, RSB Integration, and application booting
Announcement #1
Relicensing to Two Paragraph BSD

• Original rationale for *GPL v2 + exception*
  – felt needed to establish a community
  – “stick” to encourage submissions
  – license was used by GCC language run-times

• Rationale to switch
  – RTEMS has a strong community
  – users understand business value of submitting
    • we have explained the “carrot”
  – GPL v3 has soured some on GPL in general

http://www.rtems.org
Mechanics of Switching

• Permission from each submitter must be obtained
  – permission obtained from top contributors
• All contributors to a file must give permission before that file can be relicensed
• Web form with confirmation email to grant permission to relicense
• Effort will be made to locate the “missing”
When Permission is Not Obtained

• Some files may never be relicensed
• If user funds, replacement may be written
• Older BSPs and ports may be obsoleted and removed on technical grounds
  – These are the most likely to contain files to have contributors that are hard to find

http://www.rtems.org
Generation of License Report

• We are considering marking files with Linux Foundation sponsored [SPDX.org](http://www.spdx.org) annotation
• Standard annotation for open source licenses
  – Examples: NASA-1.3, GPL-2.0
• Can annotate exceptions like current RTEMS license
  – Example: GPL-2.0-with-linking-exception
• Tools exist to generate overall license reports
  – RTEMS would have to integrate one with build system
Conclusion

• Work is ongoing and evolving

• Evolution is driven by feedback

• Community involvement and sponsorship is needed to keep technical work from dragging

• License change is important to community

http://www.rtems.org
Contact Information

Joel Sherrill, Ph.D.

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