RTEMS and EPICS

Past Present and Future

The present defines the future. The future builds on the foundation of the past (Lailah Gifty Akita)

Chris Johns
Contemporary Software
chris@contemporary.software

Joel Sherrill
OAR Corporation
joel.sherrill@oarcorp.com

RTEMS Releases

RTEMS 4.5 to RTEMS 4.10

Classic API, POSIX, Networking (IPv4)

RTEMS 4.11

RSB, Classic API, POSIX, Legacy Networking (IPv4), Dynamic loading

RTEMS 5

RSB, SMP, Classic API, POSIX, LibBSD Networking (IPv6, IPv4), Dynamic loading, Capture and Tracing, RTEMS Test, No pre-installed include files, new BSP source tree, faster build times, more tests

RTEMS 5

- Smart system initialization. No special linking to disable managers
- BSP support for function and data sections with linker garbage collection
- Production quality Symmetric Multiprocessing (SMP)
- Flat Device Tree (FDT) support
- GCC 7.5.0, recent Newlib
  - `time_t` is 64-bit to address the 2038 problem
  - newlib internal locks are supported for FILE objects
  - `stdin`, `stdout` and `stderr` are global and no longer thread specific
  - support for C11 and C++11 and newer atomics
  - support for C++17 `std::aligned_alloc`
- Improved User Manual and Quick start guide

RTEMS 5

- Programming Interfaces
  - More complete and better compliance to POSIX
  - Changes
    - POSIX services enabled by default. The configure option only controls POSIX timers, signals, and the sporadic scheduling policy
    - Termios supports generation of signals
    - POSIX key destructors called during thread restart
    - Thread Local Storage (TLS)
    - Interrupt API changes. Classic API disable is for all cores and the new API is local core only
  - API Removals
    - These cannot support SMP, task preemption disable, task notepads and task variables

RTEMS 5

- SMP
  - Performance improvements to Multiprocessor Resource Sharing Protocol (MrsP)
  - Add support for O(m) Independence-Preserving Protocol (OMIP)
  - Support for thread pinning (enables support for Epoch Based Reclamation)
  - FreeBSD Timercounter implementation for scalable SMP timers
  - Scheduler support for EDF, one-to-one and one-to-all thread to processor affinities and thread pinning
  - Timers (watchdogs) use per-processor data structures

RTEMS 5

- LibBSD
  - Library of FreeBSD kernel and user land code. Includes networking, wifi, USB, mass storage support, and a number of standard FreeBSD shell commands
  - Imported FreeBSD files is 3329, file unchanged is 2842 (85.4%) and changed is 487. Opacity rating is 2.3% or 97.7% of the FreeBSD is unchanged
  - Production quality
  - Recent FreeBSD security fixes
  - SMP and internal threading support with fine grain locking
  - IPv4, IPv6, VLAN, bridging, packet filtering, PCAP, hardware checksum offloading, kqueue, OpenSSL
  - We track the current FreeBSD release and master branches. RTEMS releases are made with the current FreeBSD release. Master is tracked so we can quickly move to the next FreeBSD

RTEMS Today

What is happening in RTEMS?

- License Changes
- Quality
- Dynamic Loading (`libdl`)
- Debugging Support (`libdebugger`)
- Capture and tracing
- Ecosystem
    - RTEMS Source Builder
    - RSB Vertical Software Stack
    - Deployment
- Foundation

License Change

- Started the move of RTEMS from GPLv2+runtime exception to 2-clause BSD
- GPLv2+runtime exception license has worked well for RTEMS but it worked like a permissive license so why not simplify and use a permissive license
- 2-Clause BSD license lowers the compliance cost for an RTEMS system
  - No need to have a legal team consider the RTEMS GPL and exception license. There were a number of variations in use at the time (`libgcc`, `libgnat`, RTEMS, etc)
- Merged changes and fixes are looked after by the RTEMS project
  - RTEMS is active and evolving and this makes maintaining changes outside hard and costly
  - Unmerged changes leaving the maintenance to the holder of the change
  - Consider using support services to help merge changes if you do not know how to do this

Quality

- Working to streamline and improve the quality of RTEMS release
- Improve the onboarding experience for new users
- Tiers
  - Architecture and BSP tiers provide users with a simple scorecard to evaluate BSP quality
  - Used to remove BSPs that lack support and cannot be maintained
- Testing
  - Users can validate the tools and RTEMS by running the testsuite and comparing results with posted RTEMS test results
  - The `rtems-test` command supports simulator and hardware testing of over 500 test executables in a session
- Build mailing list activity is 600-1300 posts per month

Quality

- Hardware Testing
    - Quality can only be measure by running the tests on hardware
    - Testing during RTEMS development lowers overheads and costs for everyone
    - RTEMS developers want to release working and tested code. Most user are only interested in releases. Working together to find a solution
    - Common drivers and shared code requires regular testing
    - RTEMS is known for long lived support on older hardware and as hardware ages support gets harder
- RTEMS Qualification
    - ESA is working with RTEMS to provide a support structure that facilitates a formal qualification process for RTEMS users
    - You can generate the same data sets a formal qualification process uses

Dynamic Loading (`libdl`)

- Improved loading of ELF format files
  - Support for C++ exception throws across and from loaded modules
  - Extended address range fix-up for architectures that use short range addressing, for example ARM and PowerPC
  - Support for SDATA on the PowerPC added
- Linking from archives
  - Loading of dependent object files based on symbol resolution
  - Recursive resolution until all symbols resolved
  - Lacks function and data section support to minimised loaded footprint
- TLS support missing
- Dynamically loaded debugging support in GDB missing, `libdl` has support

Embedded With RTEMS www.rtems.org

Debugging (`libdebugger`)

- Debugging Agent
  - Designed for application debugging
  - Uses GDB remote protocol, no special GDB patches
  - Stop all model
  - Thread aware with the ability to switch threads and inspect the stack
  - Catch a crash
  - Implement in software on the target so not all bugs can be debugged with this agent
  - Limited architectures supported and fragile support on ARM especially SMP
  - No PowerPC support
- Tracing
  - High performance trace support added
  - Flexible target upload options

Ecosystem

- A maturing and important part of RTEMS
- Provides a framework of tools and commands with interfaces that can be maintained from release to release
- Provide users with the confidence to build infrastructure and workflows around documented commands will not change on them requiring further investment
- Users use the tool the RTEMS developers use
- Framework for user to build deployable systems

RTEMS Source Builder

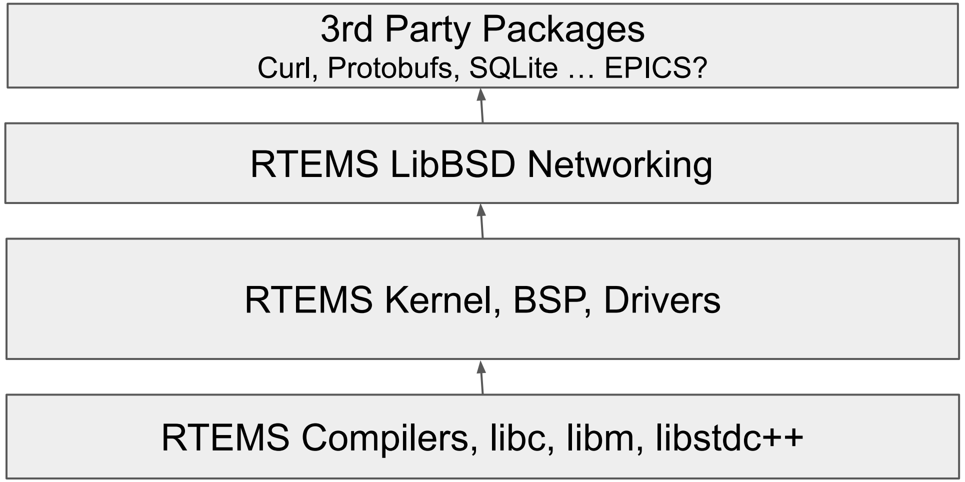
Embedded With RTEMS
www.rtems.org

- Repeatable builds of complex packages that change over time
- Wide range of supported host operating systems. Designed to handle fast change and evolving host operating systems and hardware
- Released or deployed sources are self hosted to protect against upstream home site changes
- All source, patches and options captured at the start of a project. Easily integrated into a project’s long term archiving and configuration management
- Supports building vertical software stacks
- Designed to be deployed

CONTEMPORARY SOFTWARE OAR

RSB Vertical Software Stack

- Simple software stack based on interface dependencies. The layer above depends on the layers below

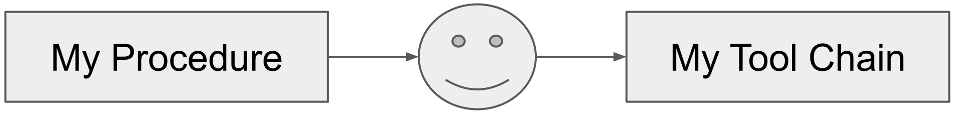

3rd Party Packages
Curl, Protobufs, SQLite … EPICS?
RTEMS LibBSD Networking
RTEMS Kernel, BSP, Drivers
RTEMS Compilers, libc, libm, libstdc++

Deployment

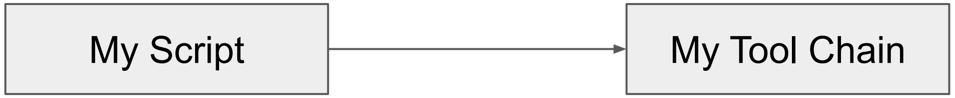
- Creating and distributing binary packages containing a tool chain, BSP plus any 3rd party packages your application needs
- Control the quality
- Match your project or your organisation’s packaging system
- Wrap the RTEMS ecosystem tools and commands to support your workflows, team structures and quality systems
- Run `rtems-test` and the testsuite on your hardware to produce a baseline set of test results
- The RTEMS Project is not involved in deployment. Deployment is a whole project on it’s own. Work with your support services to find a solution

Deployment Examples

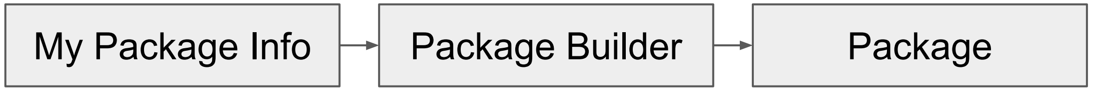
- A procedure



- A script



- A tar file
- A packaging system



- **Deployment support options are available**

EPICS and RTEMS

Our understanding of EPICS and RTEMS?

- EPICS X + RTEMS 4.10 Capabilities
- EPICS 7 + RTEMS 5 Capabilities
- EPICS BSP Network Driver Status
- Improving EPICS+RTEMS Experience

EPICS X + RTEMS 4.10 Capabilities

- Legacy networking stack
  - BSP contained drivers. Some drivers in the ports add-on kit
  - NFSv2
- GeSys external package not in RTEMS
  - BFD (GPL) based dynamic loading (RTEMS has `dlopen`/`dlclose` support)
  - Cexp runtime scripting (No RTEMS replacement)
  - Debugging agent (RTEMS has `libdebugger`)
- EPICS RTEMS build configuration
  - `Makefile.inc`. This interface exposes internal RTEMS build system values
  - Depends on packages not being built by the RSB so not in the RTEMS CI workflow
  - Links `no-*.rel` object files for unwanted managers
  - Links `rtems++`. This library has been removed

EPICS 7 + RTEMS 5 Integration

- BSP
  - We would like RSB package support. Can we build without touching EPICS internal configuration files?
  - Post-link processing depends on BSP defined settings
  - Does you BSP work with RTEMS 5?
- LibBSD Drivers
  - Memory usage has increased. EPICS may need a custom LibBSD build configuration
  - What networking features are needed?
  - NFSv4 (underway)
  - Driver integration and testing means access to hardware for ongoing support

EPICS 7 + RTEMS 5 Integration

- Dynamic Loading
  - Build system support for base image symbols
  - Build support for dynamically loaded modules (replace GeSys)
  - Target library management
- Debugging
  - `libdebugger` is green and needs support resources
  - Backend support is limited
- 3rd Party Packages
  - Should the RSB build `ncurses`, `tecla`, ... ?
  - PTP is underway
- Testing
  - If `rtems-test` can run the RTEMS testsuite can it be used run EPICS test executables?

EPICS BSP Network Roadmap

- Legacy Network Stack (e.g. libnetworking) will be obsoleted and removed
  - Will be placed in “purgatory” repo in case someone needs it and supports adding build system
  - No feature upgrades and limited support even if made to build again
- Libbsd stack is more full-featured and has larger size
- Impacts BSPs which do not yet have LibBSD drivers
- Analysis required per BSP and NIC to determine solution path
  - In libbsd, current FreeBSD, or *BSD -- easier
  - Older NICs can possibly be resurrected from old *BSD
  - Custom drivers require conversion
  - Freeze on RTEMS 5.x and plan to eliminate hardware

BSPs with EPICS Users libbsd Status

RTEMS BSP Family	NIC/Driver	Libbsd Options
Zynq	On SoC, LibBSD	Supported
PC	Various	Supported
Motorola Shared	DEC NIC	Support in process
Beatnik	em, gfe,mve (GT64260)	FreeBSD em, old NetBSD gfe, custom mve
mvme3100	tsec	Custom, maybe FreeBSD tsec
mvme5500	wm, GT64260	FreeBSD wm, custom like mve, same as Beatnik
gen68360	on SoC, custom for RTEMS	refactor, is it in use?
uc5282	on SoC, custom for RTEMS	refactor, is it in use?
mvme162/167	i82596	old FreeBSD, is it in use?

Improving RTEMS+EPICS Experience

- What architecture are still in use, m68k, PowerPC, i386, …?
  - We have not deleted m68k VMEbus boards, can we?
- PowerPC Multithreaded Debugger Support (other CPUs?)
- EPICS RTEMS Source Builder Recipe
- RTEMS.org automation of EPICS+RTEMS Testing
- 3rd party libraries EPICS uses
- What do you want?

Conclusion

- RTEMS and EPICS have a long history of use together
- Two core developers are how engaged in the EPICS community
- Closer cooperation can improve experience

Contacts

Chris Johns
Contemporary Software
e: chris@contemporary.software

Joel Sherrill
OAR Corporation
e: joel.sherrill@oar.com