

Conference Call 5/4/2005 Minutes

Attendees:

John Cherry - OSDL
James Perkins - WindRiver
Richard Griffiths - WindRiver
Daniel Walker - MontaVista
Sven Dietrich - MontaVista
Khem Raj - MontaVista
Vicky Xu - MontaVista

Informal Mutex SIG meeting at OLS

Sven, Inaky, John Cherry, James Perkins, Manas Saksena
- set something up

Bull Robust Futex Implementation

APPLICATION

- * Bull working on a processor emulator
- * Bull needs inter-process synchronization and HA

IMPLEMENTATION

- * Supports uncontested (fast-path) locking
 - ⚡ is this possible with robustness?
- * Pending authorization for (C) assignment to glibc

INTERFACE

- * Futex system call interface has been described as oversimplified, behaving like the ioctl interface.
 - ⚡ Bull added additional op-codes for 'get_robust', 'set_robust'
- * Is Futex system call appropriate / sufficient for implementation of robust priority-oriented mutexes?

INTEGRATION

- * Bull should probably submit code to LKML, as soon as glibc copyright is resolved.
- * How much of this code could be reconciled with Fusyn?
 - Fusyn can use system call or Futex interface
 - Integration with RT PI and RT deadlock-detect would be problem subset as for Fusyn
 - There is only going to be one pthread, binary-compatible interface.

Fusyn and RT CVS

Status: Inaky's work merging Fusyn patch with RT patch
- is it possible to extract Fusyn patch apart from RT patch in CVS?

Fusyn patch on Kernel 2.6.11-rc3
- CVS Branch: fusyn___branch

Fusyn+RT patch on Kernel 2.6.11-rc2-RT-V0.7.45-01
- CVS Branch: fusyn_rtp___branch

Current Development

- * errno patches submitted to Andrew Morton; status?
- * plist patch submitted to Ingo Molnar
 - incorporated into RT-preempt
 - some bug reports generated
 - currently removed from RT patch
 - Daniel is debugging performance issue and will push back to Ingo / RT patch
- * Userspace - Status / Library Updates (status 4/20)
 - new timeout structure
 - set clock ID to CLOCK_REALTIME
 - funlock and fuqueue_wait - 2 new kernel args
 - glibc awareness of userspace flag bits (up-to-date)
- * GLIBC - plans to upgrade Fusyn to 2.3.5 ?

Issues / Next Steps

----- Approaches to Reconciling Fusyn and RT

1. Drop RT Mutex or fulock entirely, migrate PI / DD functionality.

The Fusyn is superset to all implementations. Possibly bring up the RT kernel using the Fusyn in place of the RT mutex, for experimentation and performance comparison.

- * Can RT Mutex replace fulock?
- * Can fulock replace RT Mutex?
 - Better SMP scalability
- * What's required for everything to shim properly with user-space?

2. Gradual merge / reconciliation of implementations
Convergence between RT mutex and Fusyn requires abstraction of PI and D/D, regardless of Bull Futex or Fusyn choice.

- * Abstract Generic PI
 - provide per-mutex PI disable
 - global PI disable switch
 - Extend generic PI for priority-protection (PCP)
 - Implications of userspace PCP and kernel PI
- * Abstract Generic Priority Wait Queues
 - Fuqueue implementation is race-free
- * Abstract Deadlock-detection
 - Evaluate user-space / kernel cross-over:
 - can kernel tasks lock user mutexes?
- * Abstract / Generic vlocator and hash tables?

Issues / Requirements

High Level User Mutex Requirements (from FUSYN capabilities)

Availability of per-mutex PI global switch

- * Posix Priority-inheritance.
 - PI always enabled on kernel locks.
- * fulock / RT mutex exposure in both address spaces.
- * Creating hooks into fulock code for RT functionality
- * Robust try-lock
- * Support atomic access to userspace
- * errno recoverability from existing syscalls
 - see pending -mm patches
- * Compatibility with system semaphore:
 - see RT compat_semaphore implementation

Library Requirements

-
- * Binary interface - pthread support - can it be retained?
 - * Robustness must be switchable or configurable. Per mutex?
 - * PI must be switchable or configurable. Per mutex?
 - PI doesn't hurt any application. When needed, PI is needed regardless of cost.
 - * Uncontested mutex-locking without kernel access
 - not compatible with robustness
 - no performance impact when PI enabled.
 - * Memory pages storing Mutexes must not swapped.
 - eliminates get_user race condition with page fault
 - * Secondary interface for RT/PI mutex
 - possibly problematic with community.
 - * glibc changes (using pid) for mutex ownership

Community Acceptance

-
- * Robust Mutex capabilities are no-drag kernel patches, such that if not configured, no overhead is incurred
 - * Library is not as easily configurable / PI / DD / Robustness switches should be runtime, but no drag.

Current Mutex Option Features and Overview

<i>Feature</i>	<i>RT Mutex</i>	<i>Fusyn</i>	<i>Bull Futex</i>
SMP Scalability	Marginal	Good	?
Priority Inheritance	Yes	Yes	No
Priority Protection	No	Yes	No
Priority Queues	In progress	Yes	No
O(1) Wake-up	No	Yes	No
Robustness	No	Yes	Yes
Deadlock Detection	Yes	Yes	No
User space fast-path locking	N/A	Yes	Yes
User library	No	NPTL	NPTL
User Linux API	No	Fusyn/Futex	Futex
User Posix API	No	Yes	No
Posix Timeout Support	No	Yes	No
Binary compatibility	semaphore	pthread	pthread
Architecture Specific Implementation	No	Library	Library
Community Acceptance	Likely	Low	?