[patch 09/14] syslets: x86, add move_user_context() method

Source: http://linux.derkeiler.com/Mailing−Lists/Kernel/2007−02/msg05306.html

- *From*: Ingo Molnar <mingo@xxxxxxx>
- *Date*: Thu, 15 Feb 2007 17:52:42 +0100

From: Ingo Molnar <mingo@xxxxxxx>

add the move_user_context() method to move the user−space
context of one kernel thread to another kernel thread.
User−space might notice the changed TID, but execution,
stack and register contents (general purpose and FPU) are
still the same.

An architecture must implement this interface before it can turn
CONFIG_ASYNC_SUPPORT on.

Signed−off−by: Ingo Molnar <mingo@xxxxxxx>
Signed−off−by: Arjan van de Ven <arjan@xxxxxxxxxxxxxx>
−−−
arch/i386/kernel/process.c | 21 ++++++++++++++++++++
include/asm−i386/system.h | 7 +++++++
2 files changed, 28 insertions(+)

Index: linux/arch/i386/kernel/process.c
===
−−− linux.orig/arch/i386/kernel/process.c
+++ linux/arch/i386/kernel/process.c
@@ −820,6 +820,27 @@ unsigned long get_wchan(struct task_stru
}

/*
+ * Move user−space context from one kernel thread to another.
+ * This includes registers and FPU state. Callers must make
+ * sure that neither task is running user context at the moment:
+ */
+void
+move_user_context(struct task_struct *new_task, struct task_struct *old_task)
+{
+ struct pt_regs *old_regs = task_pt_regs(old_task);
+ struct pt_regs *new_regs = task_pt_regs(new_task);
+ union i387_union *tmp;
+
+ *new_regs = *old_regs;

```
+ /*
+ * Flip around the FPU state too:
+ */
+ tmp = new_task−>thread.i387;
+ new_task−>thread.i387 = old_task−>thread.i387;
+ old_task−>thread.i387 = tmp;
+}
+
+/*
* sys_alloc_thread_area: get a yet unused TLS descriptor index.
*/
static int get_free_idx(void)
Index: linux/include/asm−i386/system.h
===================================================================
−−− linux.orig/include/asm−i386/system.h
+++ linux/include/asm−i386/system.h
@@ −33,6 +33,13 @@ extern struct task_struct * FASTCALL(__s
"2" (prev), "d" (next)); \
} while (0)

+/*
+ * Move user−space context from one kernel thread to another.
+ * This includes registers and FPU state for now:
+ */
+extern void
+move_user_context(struct task_struct *new_task, struct task_struct *old_task);
+
#define _set_base(addr,base) do { unsigned long __pr; \
__asm__ __volatile__ ("movw %%dx,%1\n\t" \
"rorl $16,%%edx\n\t" \
−
```