[patch 02/23] GTOD: persistent clock support, core

Source: http://linux.derkeiler.com/Mailing−Lists/Kernel/2006−09/msg08953.html

- *From*: Thomas Gleixner <tglx@xxxxxxxxxxxxx>
- *Date*: Fri, 29 Sep 2006 23:58:21 −0000

From: John Stultz <johnstul@xxxxxxxxxx>

persistent clock support: do proper timekeeping across suspend/resume.

Signed−off−by: John Stultz <johnstul@xxxxxxxxxx>
Signed−off−by: Thomas Gleixner <tglx@xxxxxxxxxxxxx>
Signed−off−by: Ingo Molnar <mingo@xxxxxxx>
−−
include/linux/hrtimer.h | 3 +++
include/linux/time.h | 1 +
kernel/hrtimer.c | 8 ++++++++
kernel/timer.c | 34 ++++++++++++++++++++++++++++++++−−−
4 files changed, 43 insertions(+), 3 deletions(−)

linux−2.6.18−rc6_timeofday−persistent−clock−generic_C6.patch
Index: linux−2.6.18−mm2/include/linux/hrtimer.h
===
−−− linux−2.6.18−mm2.orig/include/linux/hrtimer.h 2006−09−30 01:41:14.000000000 +0200
+++ linux−2.6.18−mm2/include/linux/hrtimer.h 2006−09−30 01:41:15.000000000 +0200
@@ −146,6 +146,9 @@ extern void hrtimer_init_sleeper(struct
/* Soft interrupt function to run the hrtimer queues: */
extern void hrtimer_run_queues(void);

+/* Resume notification */
+void hrtimer_notify_resume(void);
+
/* Bootup initialization: */
extern void __init hrtimers_init(void);

Index: linux−2.6.18−mm2/include/linux/time.h
===
−−− linux−2.6.18−mm2.orig/include/linux/time.h 2006−09−30 01:41:14.000000000 +0200
+++ linux−2.6.18−mm2/include/linux/time.h 2006−09−30 01:41:15.000000000 +0200
@@ −92,6 +92,7 @@ extern struct timespec xtime;
extern struct timespec wall_to_monotonic;
extern seqlock_t xtime_lock;

+extern unsigned long read_persistent_clock(void);
void timekeeping_init(void);

```
static inline unsigned long get_seconds(void)
Index: linux−2.6.18−mm2/kernel/hrtimer.c
===================================================================
−−− linux−2.6.18−mm2.orig/kernel/hrtimer.c 2006−09−30 01:41:14.000000000 +0200
+++ linux−2.6.18−mm2/kernel/hrtimer.c 2006−09−30 01:41:15.000000000 +0200
@@ −287,6 +287,14 @@ static unsigned long ktime_divns(const k
#endif /* BITS_PER_LONG >= 64 */

/*
+ * Timekeeping resumed notification
+ */
+void hrtimer_notify_resume(void)
+{
+ clock_was_set();
+}
+
+/*
* Counterpart to lock_timer_base above:
*/
static inline
Index: linux−2.6.18−mm2/kernel/timer.c
===================================================================
−−− linux−2.6.18−mm2.orig/kernel/timer.c 2006−09−30 01:41:14.000000000 +0200
+++ linux−2.6.18−mm2/kernel/timer.c 2006−09−30 01:41:15.000000000 +0200
@@ −41,6 +41,9 @@
#include <asm/timex.h>
#include <asm/io.h>

+/* jiffies at the most recent update of wall time */
+unsigned long wall_jiffies = INITIAL_JIFFIES;
+
u64 jiffies_64 __cacheline_aligned_in_smp = INITIAL_JIFFIES;

EXPORT_SYMBOL(jiffies_64);
@@ −743,12 +746,20 @@ int timekeeping_is_continuous(void)
return ret;
}

+/* Weak dummy function for arches that do not yet support it.
+ * XXX − Do be sure to remove it once all arches implement it.
+ */
+unsigned long __attribute__((weak)) read_persistent_clock(void)
+{
+ return 0;
+}
+
/*
* timekeeping_init − Initializes the clocksource and common timekeeping values
*/
void __init timekeeping_init(void)
```

```
{
− unsigned long flags;
+ unsigned long flags, sec = read_persistent_clock();

write_seqlock_irqsave(&xtime_lock, flags);

@@ −758,11 +769,18 @@ void __init timekeeping_init(void)
clocksource_calculate_interval(clock, tick_nsec);
clock−>cycle_last = clocksource_read(clock);

+ xtime.tv_sec = sec;
+ xtime.tv_nsec = (jiffies % HZ) * (NSEC_PER_SEC / HZ);
+ set_normalized_timespec(&wall_to_monotonic,
+ −xtime.tv_sec, −xtime.tv_nsec);
+
write_sequnlock_irqrestore(&xtime_lock, flags);
}


static int timekeeping_suspended;
+static unsigned long timekeeping_suspend_time;
+
/**
* timekeeping_resume − Resumes the generic timekeeping subsystem.
* @dev: unused
@@ −773,14 +791,23 @@ static int timekeeping_suspended;
*/
static int timekeeping_resume(struct sys_device *dev)
{
− unsigned long flags;
+ unsigned long flags, now = read_persistent_clock();

write_seqlock_irqsave(&xtime_lock, flags);
− /* restart the last cycle value */
+
+ if (now && (now > timekeeping_suspend_time)) {
+ unsigned long sleep_length = now − timekeeping_suspend_time;
+ xtime.tv_sec += sleep_length;
+ jiffies_64 += sleep_length * HZ;
+ }
+ /* re−base the last cycle value */
clock−>cycle_last = clocksource_read(clock);
clock−>error = 0;
timekeeping_suspended = 0;
write_sequnlock_irqrestore(&xtime_lock, flags);
+
+ hrtimer_notify_resume();
+
return 0;
}
```

@@ −790,6 +817,7 @@ static int timekeeping_suspend(struct sy

write_seqlock_irqsave(&xtime_lock, flags);
timekeeping_suspended = 1;
+ timekeeping_suspend_time = read_persistent_clock();
write_sequnlock_irqrestore(&xtime_lock, flags);
return 0;
}

−−

−
To unsubscribe from this list: send the line "unsubscribe linux−kernel" in
the body of a message to majordomo@xxxxxxxxxxxxxxx
More majordomo info at http://vger.kernel.org/majordomo−info.html
Please read the FAQ at http://www.tux.org/lkml/