Subject: [PATCH] Consolidate default sched_clock()
Posted by adobriyan on Mon, 25 Dec 2006 12:37:01 GMT
View Forum Message <> Reply to Message

Signed-off-by: Alexey Dobriyan <adobriyan@openvz.org>

```
 arch/alpha/kernel/time.c     |   11 -----------
 arch/arm/kernel/time.c       |   10 ----------
 arch/arm26/kernel/time.c     |    8 --------
 arch/avr32/kernel/time.c     |    9 ---------
 arch/cris/kernel/time.c      |    8 --------
 arch/h8300/kernel/time.c     |    6 ------
 arch/m32r/kernel/time.c      |    8 --------
 arch/m68k/kernel/time.c      |    9 ---------
 arch/m68knommu/kernel/time.c |    9 ---------
 arch/mips/kernel/time.c      |    5 -----
 arch/parisc/kernel/time.c    |   11 -----------
 arch/sh/kernel/time.c        |    8 --------
 arch/sh64/kernel/time.c      |    9 ---------
 arch/sparc/kernel/time.c     |    9 ---------
 arch/v850/kernel/time.c      |    8 --------
 arch/xtensa/kernel/time.c    |    9 ---------
 kernel/sched.c               |   10 ++++++++++
 17 files changed, 10 insertions(+), 137 deletions(-)

--- a/arch/alpha/kernel/time.c
+++ b/arch/alpha/kernel/time.c
@@ -91,17 +91,6 @@ static inline __u32 rpcc(void)
 }

 /*
- * Scheduler clock - returns current time in nanosec units.
- *
- * Copied from ARM code for expediency... ;-}
- */
-unsigned long long sched_clock(void)
-{
-       return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
-
-/*
  * timer_interrupt() needs to keep up the real-time clock,
  * as well as call the "do_timer()" routine every clocktick
  */
--- a/arch/arm/kernel/time.c
+++ b/arch/arm/kernel/time.c
```

```
@@ -75,16 +75,6 @@ static unsigned long dummy_gettimeoffset
 }
 #endif

-/*
- * Scheduler clock - returns current time in nanosec units.
- * This is the default implementation.  Sub-architecture
- * implementations can override this.
- */
-unsigned long long __attribute__((weak)) sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 static unsigned long next_rtc_update;

 /*
--- a/arch/arm26/kernel/time.c
+++ b/arch/arm26/kernel/time.c
@@ -89,14 +89,6 @@ static unsigned long gettimeoffset(void)
        return (offset + LATCH/2) / LATCH;
 }

-/*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 static unsigned long next_rtc_update;

 /*
--- a/arch/avr32/kernel/time.c
+++ b/arch/avr32/kernel/time.c
@@ -110,15 +110,6 @@ static void avr32_hpt_init(unsigned int
 }

 /*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- /* There must be better ways...? */
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
```

```
-/*
  * local_timer_interrupt() does profiling and process accounting on a
  * per-CPU basis.
  *
--- a/arch/cris/kernel/time.c
+++ b/arch/cris/kernel/time.c
@@ -217,14 +217,6 @@ #if CONFIG_PROFILING
 #endif
 }

-/*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 static int
 __init init_udelay(void)
 {
--- a/arch/h8300/kernel/time.c
+++ b/arch/h8300/kernel/time.c
@@ -118,9 +118,3 @@ int do_settimeofday(struct timespec *tv)
 }

 EXPORT_SYMBOL(do_settimeofday);
-
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-
-}
--- a/arch/m32r/kernel/time.c
+++ b/arch/m32r/kernel/time.c
@@ -286,11 +286,3 @@ #else
 #error no chip configuration
 #endif
 }
-
-/*
- *  Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
--- a/arch/m68k/kernel/time.c
```

```
+++ b/arch/m68k/kernel/time.c
@@ -159,12 +159,3 @@ int do_settimeofday(struct timespec *tv)
 }

 EXPORT_SYMBOL(do_settimeofday);
-
-/*
- * Scheduler clock - returns current time in ns units.
- */
-unsigned long long sched_clock(void)
-{
-       return (unsigned long long)jiffies*(1000000000/HZ);
-}
-
--- a/arch/m68knommu/kernel/time.c
+++ b/arch/m68knommu/kernel/time.c
@@ -173,13 +173,4 @@ int do_settimeofday(struct timespec *tv)
  clock_was_set();
  return 0;
 }
-
-/*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 EXPORT_SYMBOL(do_settimeofday);
--- a/arch/mips/kernel/time.c
+++ b/arch/mips/kernel/time.c
@@ -457,8 +457,3 @@ EXPORT_SYMBOL(rtc_lock);
 EXPORT_SYMBOL(to_tm);
 EXPORT_SYMBOL(rtc_mips_set_time);
 EXPORT_SYMBOL(rtc_mips_get_time);
-
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies*(1000000000/HZ);
-}
--- a/arch/parisc/kernel/time.c
+++ b/arch/parisc/kernel/time.c
@@ -288,17 +288,6 @@ do_settimeofday (struct timespec *tv)
 }
 EXPORT_SYMBOL(do_settimeofday);

-/*
```

```
- * XXX: We can do better than this.
- * Returns nanoseconds
- */
-
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
-
 void __init start_cpu_itimer(void)
 {
  unsigned int cpu = smp_processor_id();
--- a/arch/sh/kernel/time.c
+++ b/arch/sh/kernel/time.c
@@ -41,14 +41,6 @@ static int null_rtc_set_time(const time_
 void (*rtc_sh_get_time)(struct timespec *) = null_rtc_get_time;
 int (*rtc_sh_set_time)(const time_t) = null_rtc_set_time;

-/*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long __attribute__ ((weak)) sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 #ifndef CONFIG_GENERIC_TIME
 void do_gettimeofday(struct timeval *tv)
 {
--- a/arch/sh64/kernel/time.c
+++ b/arch/sh64/kernel/time.c
@@ -579,12 +579,3 @@ #endif
  asm __volatile__ ("nop");
  panic("Unexpected wakeup!\n");
 }
-
-/*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
--- a/arch/sparc/kernel/time.c
+++ b/arch/sparc/kernel/time.c
@@ -436,15 +436,6 @@ static inline unsigned long do_gettimeof
```

```
  return (*master_l10_counter >> 10) & 0x1fffff;
 }

-/*
- * Returns nanoseconds
- * XXX This is a suboptimal implementation.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 /* Ok, my cute asm atomicity trick doesn't work anymore.
  * There are just too many variables that need to be protected
  * now (both members of xtime, et al.)
--- a/arch/v850/kernel/time.c
+++ b/arch/v850/kernel/time.c
@@ -28,14 +28,6 @@ #include "mach.h"
 #define TICK_SIZE (tick_nsec / 1000)

 /*
- * Scheduler clock - returns current time in nanosec units.
- */
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
-/*
  * timer_interrupt() needs to keep up the real-time clock,
  * as well as call the "do_timer()" routine every clocktick
  */
--- a/arch/xtensa/kernel/time.c
+++ b/arch/xtensa/kernel/time.c
@@ -38,15 +38,6 @@ #endif
 unsigned int last_ccount_stamp;
 static long last_rtc_update = 0;

-/*
- * Scheduler clock - returns current tim in nanosec units.
- */
-
-unsigned long long sched_clock(void)
-{
- return (unsigned long long)jiffies * (1000000000 / HZ);
-}
-
 static irqreturn_t timer_interrupt(int irq, void *dev_id);
```

```
 static struct irqaction timer_irqaction = {
  .handler = timer_interrupt,
--- a/kernel/sched.c
+++ b/kernel/sched.c
@@ -57,6 +57,16 @@ #include <asm/tlb.h>
 #include <asm/unistd.h>

 /*
+ * Scheduler clock - returns current time in nanosec units.
+ * This is default implementation.
+ * Architectures and sub-architectures can override this.
+ */
+unsigned long long __attribute__((weak)) sched_clock(void)
+{
+ return (unsigned long long)jiffies * (1000000000 / HZ);
+}
+
+/*
  * Convert user-nice values [ -20 ... 0 ... 19 ]
  * to static priority [ MAX_RT_PRIO..MAX_PRIO-1 ],
  * and back.
```