μ-second precision timer support for the Linux kernel

Uwe Walter, Vincent Oberle
walter@tm.uka.de, vincent@oberle.com

Institute of Telematics
University of Karlsruhe
D–76128 Karlsruhe, Germany

1 Introduction

Nowadays with rising demand on speed and precision in hard- and software technology, a lot of applications need the possibility to execute actions at exactly defined moments of time. Unfortunately the standard timer offered by the Linux kernel for such purposes does not adequately fulfill these requirements because its resolution is far too low. So a new and more precise means for timed execution of programmed functions is desirable.

2 APIC-Timer

Since the first IBM PC all compatible personal computers have been equipped with a programmable interrupt controller (PIC 8259A) which offers the possibility to generate interrupts with a configurable frequency. The Linux kernel uses this periodical moments of time to schedule and execute timer functions. Unfortunately the maximum frequency supported by the standard PIC is 8192 Hz, which would result in interrupts generated each 122 μs. However most of the issued interrupts are wasted if no timer function is attached to them and occupy the system with a tremendous unnecessary workload.

To reduce this inefficiency, the Linux kernel programs the PIC on i386-based architectures with a default value of 100 Hz. Consequently the time interval between two generated interrupts is 10 ms, which automatically results in the same value for the resolution of timer functions execution. By using this standard means it is therefore not possible to issue programmed actions at points of time more accurately than with a worst-case deviation of 10 ms (which results in a mean error of 5 ms).

But help is on its way: With the Pentium processor family, Intel introduced the Advanced Programmable Interrupt Controller (APIC) to replace the old PIC. This APIC is referred to as the local APIC to distinguish it from the I/O-APIC, which is an external controller to manage interrupts on SMP (symmetric multiprocessor) systems. There is no I/O-APIC on UP (uniprocessor) boards but since Intel's P54C all CPUs contain an on-chip local APIC.

Nevertheless, the local APIC is generally disabled on P5 chips and cannot be enabled by software. Only P6 chips (i.e. starting with Pentium Pro) allow to enable the local APIC by software means.

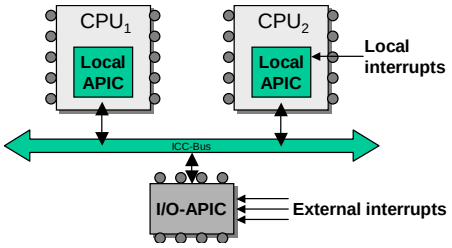

Fig. 1. Local APIC and I/O APIC in P6-PCs

Figure 1 shows the architecture used in i386-compatible PCs. It shows the case of a multiprocessor board, where the I/O APIC is integrated. Nevertheless, in the case of a uniprocessor system the local APIC exists also on board of the CPU.

As described in [1] the local APIC unit contains a 32-bit timer accessible by the CPU. The timer can be configured by a special timer register. The time base can be derived from the processor's bus clock and can optionally be divided. It supports one-shot and periodic modes. For the problem of issuing non-frequent timer

functions the one-shot is interesting, because it does not put unnecessary workload on the system. One-shot mode means, that an initial value is copied into the register and counted down to zero with bus speed and an interrupt is generated when the timer reaches value zero.

Based on this APIC timer a kernel module was implemented which exports the basic functionality of the APIC timer to other modules. It is realized as a linked list of events, ordered according to the expiration time. This allows to issue the timers as fast as possible without a search through the complete list. If a timer expires, a user-defined function is invoked, like it would be expected from any normal system timer.

The minimum resolution with this APIC-based timer is in the magnitude of microseconds. As the bus speed of modern x86-based PCs is at least 100 MHz, the minimum resolution should be 0,1 microseconds (1/100 MHz), but due to the calculation time needed for switching to the interrupt service routine (saving context information etc.) the achievable accuracy is about 1 microsecond. Consequently this is a 1,000 to 10,000 times higher precision than the default PC timer.

3 Implementation

The APIC timer consists of two patches and a module for the standard Linux kernel tree. The module can be loaded and unloaded dynamically during runtime and handles all administrative tasks, i.e. adding, modifying and deleting timers. One objective during the development has been to put as much functionality as possible into the module and not into the kernel itself to keep maintenance and overall view easy.

Unfortunately patching the kernel is still necessary. One patch is needed to activate the local APIC on uniprocessor systems and the second patch integrates the necessary interfaces into the kernel. Without these modifications it is not possible to register an interrupt handler for the APIC timer interrupt, because the default handler smp_apic_timer_interrupt is hard-wired. This is the reason why the APIC timer cannot be used in SMP systems, because then its functionality is needed for synchronization between the different CPUs.

The APIC timer module consists of program functions and interfaces for the users of the APIC timer. The difference between the usage of the standard kernel timer and the APIC timer was kept to a minimum.

Registered timer handlers are stored in a double linked list, similar to the storage of the standard timer in the Linux kernel. The list elements are structured like this:

```
 struct apic_timer_list { struct apic_timer_list *next,
*prev; unsigned long long expires; unsigned long data; void
(*function)(unsigned long long, unsigned long) };
```

- next und prev are used to link the apic_timer_list-entries.
- The variable expires holds the value of the timestamp-counter-register (TSC) at which the timer function shall be called and executed. (The TSC is a processor register which is incremented by one at each processor cycle. It can be compared to a discrete clock with processor speed precision.) The necessary conversion into bus cycle accuracy is done automatically by the APIC timer module. To improve performance, the linked list is ordered according to execution (expires) time.
- data is simply a pointer which can point to private data of the registered timer handler function. This is extremely useful if multiple instances of a handler function are called and need to keep private data.
- function is the pointer to the timer function which shall be called when the timer is due. After the expires moment of time has been reached function is called with the additional parameter data (see above).

When the local APIC issues an interrupt, the function do_apic_irq is called, which, after a few administrative tasks, itself calls run_apic_timer. That function compares the actual moment of time to the planned expires-value, because that difference will be used for error correction of the following timers to improve accuracy. After that, the expired timer is removed from the list and its handler function is invoked. When their job is done, the local APIC will be programmed for the next due timer in the list.

The APIC timer module exports the following functions (which are quite similar to that of the standard timer, described for example in [2]) as interface for its usage:

- init_apic_timer(struct apic_timer_list *timer) initializes the given structure of type apic_timer_list.
- add_apic_timer(struct apic_timer_list *timer) registers the given structure of type apic_timer_list and inserts it into the double linked list of registered timers, which await their execution. The timer handler function timer->function will be called when the timer->expires time is reached.
- del_apic_timer(struct apic_timer_list *timer) removes a given apic_timer_list-structure from the list of registered (and waiting) timers.
- mod_apic_timer(struct apic_timer_list *timer, unsigned long long expires) modifies the expiration time (expires) of the given registered timer structure apic_timer_list. Probably its position inside the registered timer list has to be updated.

4 Using the APIC-timer for high precision traffic shaping

High precision timers can be used for –and are even needed by– a lot of applications. As an example their benefits are demonstrated by using them for network traffic shaping. Traffic shapers are an important building block for the support of quality of service (QoS) in packet networks, because they reduce packet bursts by delaying packets which arrive too fast for their configured bandwidth. It is obvious that traffic shapers need to be able to transmit waiting data packets at precisely calculated points of time to perform in a satisfying way. The accuracy of these timed actions is crucial to the performance of traffic shapers.

To show the impact of timer precision on the shaping smoothing level, research was done using the following three timers on a standard 700 MHz PC with a 100 Mbps Ethernet connection.

1. The standard PC-timer. Its trigger frequency is left unchanged at the default of 100 Hz which limits its accuracy to 10 ms.
2. A modified PC-timer. By issuing interrupts with 5000 Hz the resolution is improved to 0,2 ms but the system workload increases dramatically because of the amount of unnecessary interrupts.
3. The newly developed APIC-timer.

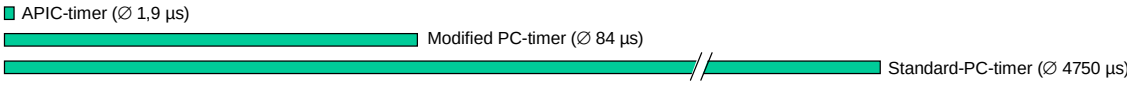

Fig. 2. Mean deviation of the actual to the planned transmission time of data packets for the different timers.

Figure 2 shows the mean deviation of the planned time a packet should be transmitted to the actual value when it is sent. Figure 3 presents histograms of the trigger error for differently delayed packets. It is obvious how the timer resolution affects the deviation.

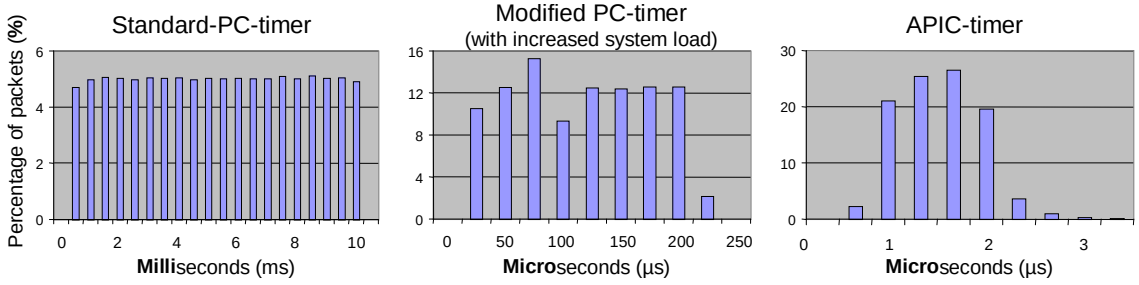

Fig. 3. Deviation of the actual transmission time to the planned sending time for the different timers.

This timer error impacts the smoothness of the shaped network traffic as it is shown in figure 4. In this experiment a constant-rate packet stream of 1292 byte UDP-packets should be shaped to a bandwidth of 1 Mbps. This would result in one data packet transmitted every 10,3 milliseconds. The figure shows the actual interpacket spacing after the shaping process.

When the standard PC-timer is used, the shaper has only the possibility to send a packet either in 10 or 20 ms. There is no way to transmit a packet in between these timer intervals. The shaper was designed to send packets early, so it transmits most of the packets with a spacing of 10 ms. The configured rate is honored by delaying a few packets for two whole timer intervals (20 ms).

The modified PC-timer can handle this job a lot better because of its finer granularity of 0,2 ms. It, too, oscillates between two adjacent timer intervals (which is hardly visible in the figure).

By using the APIC-timer it becomes possible to produce a very smooth output data stream. Its very small error of approximately 2 μs is invisible in the figure.

Even at this relatively low bandwidth the standard timer can only produce a jittering output stream. The two other timers perform better and are therefore out through a more challenging test. Figure 5 shows the results of the test shaping a 667 byte packet stream to a bandwidth of 50 Mbps, which equals sending a packet every 107 microseconds.

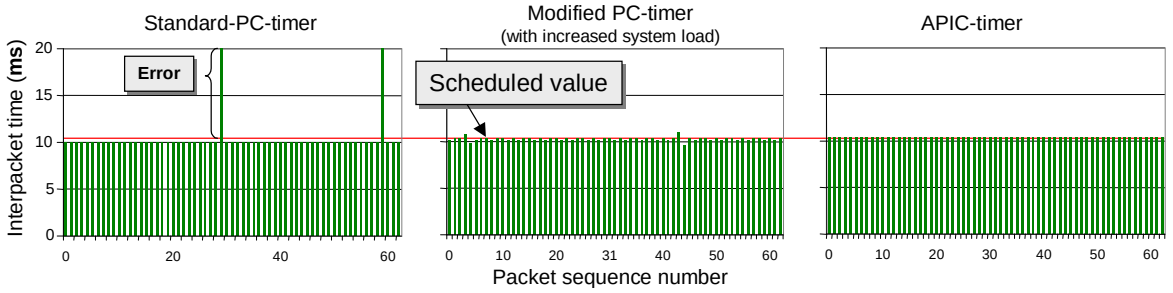

Fig. 4. Interpacket times of a shaped 1 Mbps data stream consisting of 1292 byte packets (nominal spacing 10,3 ms).

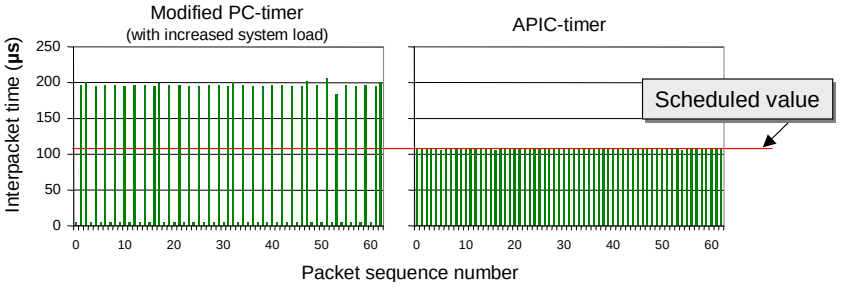

Fig. 5. Interpacket times of a shaped 50 Mbps data stream consisting of 667 byte packets (nominal spacing 107 μs).

In this experiment even the modified PC-timer is operating over its maximum resolution. The scheduled packet spacing cannot be maintained and the shaper has to send packets either immediately or delayed by a whole 0,2 ms timer interval.

Again the APIC-timer shows no signs of decreasing accuracy and produces a smooth packet stream. In following stress tests the APIC-timer was even capable of transmitting packets with a spacing of only 17μs before the test setup reached other limits (packet generation and transmission). This equals a packet frequency of 58,800 packets per second which would –even with small 100 byte packets– result in a bandwidth of more than 47 Mbps.

5 Conclusion

A new and easy-to-use timer module for the Linux kernel has been implemented, which offers the possibility to execute functions at precisely defined moments of time. By making use of the local APIC's functionality an accuracy of about 1 microsecond is achieved for the execution time of programmed actions (although only possible on uniprocessor systems at the moment).

The possible benefits of such an improved timer precision have been shown for traffic shaping as a possible example application.

References

1. Intel architecture software developer's manual, volume 1-3, 2001. http://developer.intel.com/design/pentiumii/manuals/24319{0-2}.htm.
2. Alessandro Rubini and Jonathan Corbet. *Linux Device Drivers*. O'Reilly-Verlag, 2 edition, July 2001. Online version at http://www.oreilly.com/catalog/linuxdrive2/chapter/book.