

High Resolution POSIX Timers for Linux



MONTAVISTATM
S O F T W A R E

Powering the Embedded Revolution

Real-Time Embedded Forum
Anaheim, CA
Jan 23-24, 2002

John Mehaffey
Technical Marketing





High-Res POSIX Timers for Linux

- What are they
- Why needed
- Specifics
- Issues
- Q&A





High Res Timers

What Are They?

- POSIX timers with microsecond resolution
- Choice of hardware timer sources
- Get at:
www.sourceforge.net/projects/high-res-timers



High Res Timers

Why Are They Needed?

- Better than ± 10 ms timer resolution for embedded apps
- Linux provides down to 1ms clock ticks
 - High overhead
 - Still not enough resolution for some applications
 - Spins for smaller waits
- HRT provides microsecond resolution with lower overhead
- POSIX provides portability



Hi Res Timers

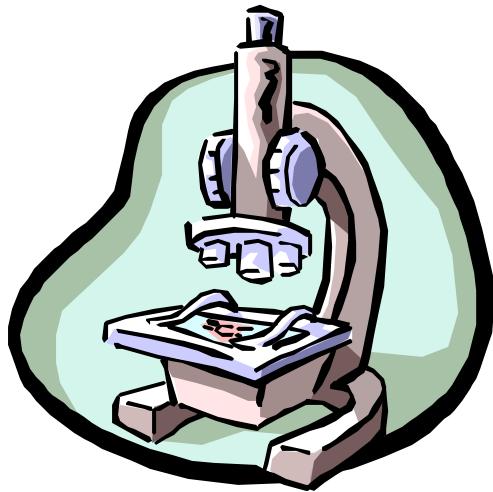
X86 Hardware Time Sources

- PIT – Programmable Interval Timer
 - 838 ns (1.193MHz)
- TSC – CPU Time Stamp Clock
 - CPU MHz
 - Power management issues
- ACPI – power management timer
 - 3x PIT resolution (280 uS)
 - Higher management overhead



Hi Res Timers Comparison

- Old timers
 - Cascading timer lists
 - Spin for <2ms
- Hi res
 - $O(1)$ removal
 - $O(N/512)$ insert
 - "Sub-jiffies" in wall time units

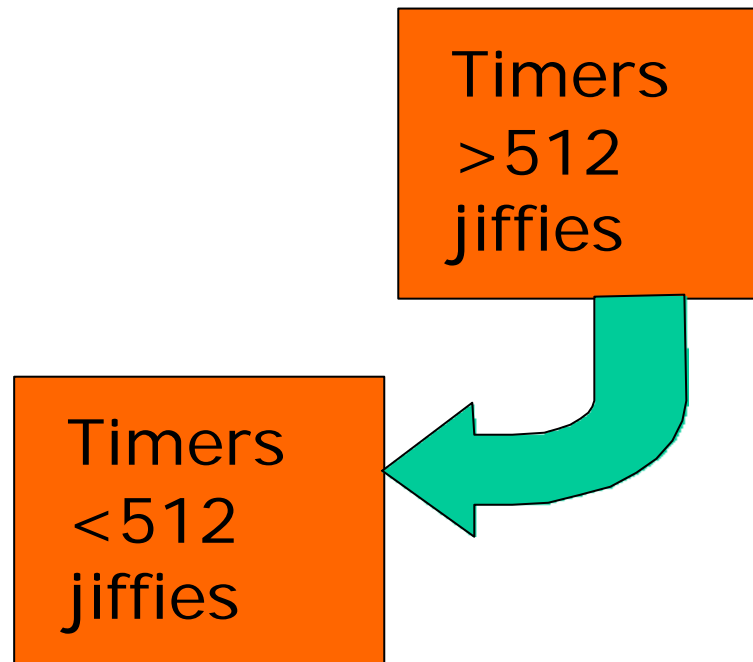




Hi Res Timers

Current Timer Algorithm

Two lists



Linear Insertion/Deletion



Hi Res Timers

Hi-Res Timer Algorithm

n timer lists

Sub-Jiffy 0
Sub-Jiffy 1
Sub-Jiffy 2
• • •
Sub-Jiffy n-2
Sub-Jiffy n-1

Indexed by low $\log_2(n)$ bits
of expiration time

n is configurable



Hi Res Timers

POSIX Clock Types

- `CLOCK_REALTIME`
 - Required clock (wall time)
 - Resolution 1/Hz
- `CLOCK_MONOTONIC`
 - Not affected by time changes
- `CLOCK_REALTIME_HR`,
`CLOCK_MONOTONIC_HR`
 - High resolution versions
 - Resolution depends on HW chosen



Hi Res Timers Programming

Time Values

Struct timespec

```
time_t  tv_sec  
long    tv_nsec
```

Struct itimerspec

```
struct timespec  it_interval  
struct timespec  it_value
```

- `it_interval = 0` ==> One time
- `it_value = 0` ==> Disable timer



Hi Res Timers Programming

Clock Management

```
#include <time.h>

int clock_gettime(clockid_t clock_id,
    const struct timespec *tp);

int clock_gettime(clockid_t clock_id,
    struct timespec *tp);

int clock_getres(clockid_t clock_id,
    struct timespec *res);
```



Hi Res Timers Programming

Timer Management

```
#include <signal.h>
#include <time.h>

int timer_create(clockid_t clock_id,
                struct sigevent *evp, timer_t *timerid);

int timer_delete(timer_t timerid);
```



Hi Res Timers Programming

Timer Usage

```
#include <time.h>
```

```
int timer_settime(timer_t timerid, int  
    flags, const struct itimerspec *value,  
    struct itimerspec *ovalue);
```

```
int timer_gettime(timer_t timerid, struct  
    itimerspec *value);
```

```
int timer_getoverrun(timer_t timerid);
```



Hi Res Timers

Sample Programs

Many programming examples are included with the archive downloadable from

www.sourceforge.net/projects/high-res-timers in the tests directory



MONTAVISTA
SOFTWARE

Hi Res Timers Issues

- Currently only on X86 architecture
- Beta code
- Kernel acceptance





MONTAVISTA
SOFTWARE

Hi Res Timers

Questions

