New event timers infrastructure

Alexander Motin
mav@FreeBSD.org

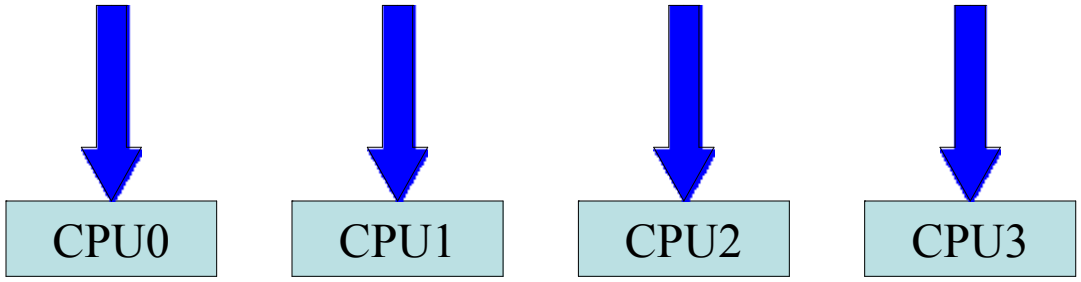
Karlsruhe, October 8, 2010

- Before:
  - each platform has own event timers management code;
  - x86 timers code very tangled, no HPET support;
  - timer interrupts have fixed freqency from HZ to 4 * HZ;
  - statclock often aliased or equal to hardclock;
  - profclock often equal to hardclock;
  - high interrupt rate increases idle power consumption, while lowering HZ increases time granularity.

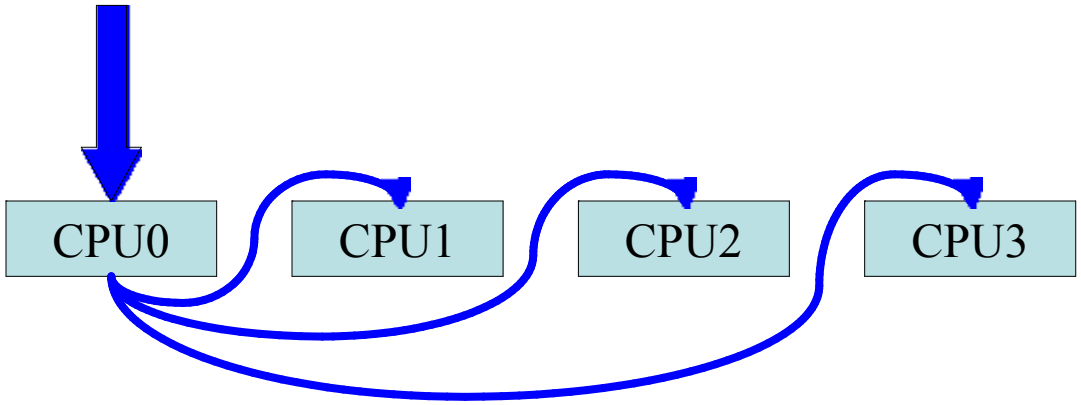
- Project consists of several parts:
  - create MI event timer driver API (done);
  - write MI event timers management code (done);
  - port MD event timer drivers to new MI API and remove MD management code:
    - arm
      - Marvell (done)
      - others (todo)
    - amd64 (done)
    - i386 (done)
      - XEN PV (todo)
    - ia64 (todo)
    - mips (done)
    - pc98 (done)
    - powerpc (done)
    - sparc64 (done)
    - sun4v (done)

- MI event timer driver API (timeet.h, kern_et.c):
  - Driver:
    - struct eventtimer;
    - et_register(struct eventtimer *et);
    - et_deregister(struct eventtimer *et);
  - Consumer:
    - et_find(const char *name, int check, int want);
    - et_init(struct eventtimer *et, et_event_cb_t *event, et_deregister_cb_t *deregister, void *arg);
    - et_start(struct eventtimer *et, struct bintime *first, struct bintime *period);
    - et_stop(struct eventtimer *et);
    - et_free(struct eventtimer *et);

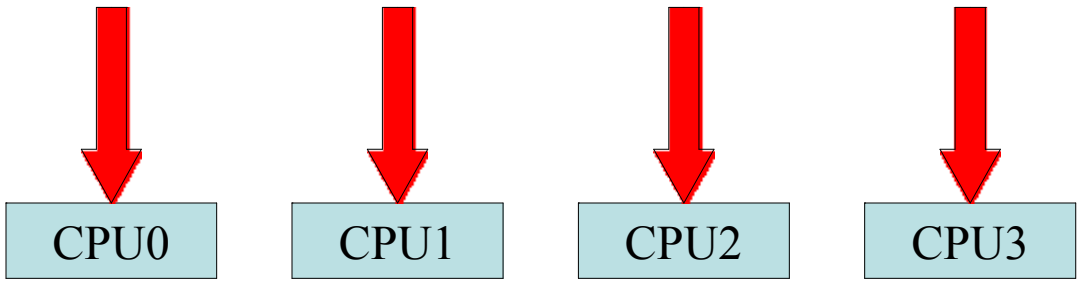
- MI event timers management code supports several modes:
  - one-shot per-CPU mode (prefered):
    - each CPU uses own one-shot capable even timer;
    - timers each time reprogrammed for the time of the next hardclock/statclock/proflock event;
    - when CPU idle -- timer programmed to skip events when no callouts scheduled (up to 1/4s);
    - IPI_HARDCLOCK may be used to wake up sleeping CPU to reprogram it's timer on inter-CPU callout scheduling;
    - binuptime() used to track time.



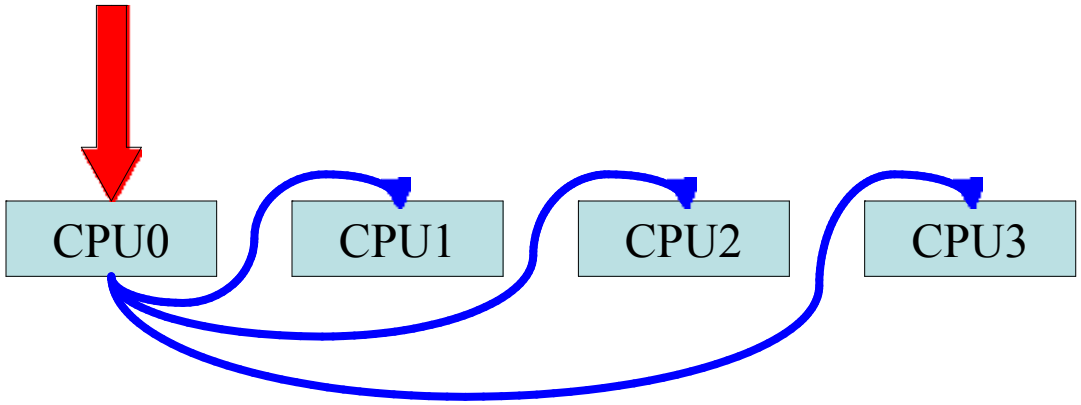
– one-shot global mode:
  - single one-shot capable even timer used;
  - timer each time reprogrammed for the time of the next hardclock/statclock/proflock events for all CPUs;
  - one CPU handles timer interrupts; IPI_HARDCLOCK used to redistribute events to others, when needed;
  - when CPU idle -- skip events when no callouts scheduled (up to 1/4s);
  - binuptime() used to track time.



– periodic per-CPU mode:
  - each CPU uses own periodic even timer;
  - timers programmed to generate fixed interrupt rate (1-4 * HZ, increased to about 8KHz when profiling);
  - no interrupts could be skipped;
  - periodic timer interrupts used to track time.



– periodic global mode:
  - single periodic capable even timer used;
  - timer programmed to generate fixed interrupt rate (1-4 * HZ, increased to about 8KHz when profiling);
  - one CPU handles timer interrupts; IPI_HARDCLOCK used to redistribute events to others, when needed;
  - when CPU idle (except first) -- skip hardclock events when no callouts scheduled up to 1/4s;
  - periodic timer interrupts used to track time.



- Operation mode depends on hardware capabilities, but in most cases can be tuned via sysctl and loader tunables.
- As soon as BSP may not receive interrupts for a long time:
  - hardclock_anycpu() implemented to replace hardclock(); it may be called at any CPU to properly update system time and do other global routine jobs for any number of hardclock events;
  - if currently active timecounter wraps often -- BSP will wake up frequent enough to handle it.
  - if kernel built with DEVICE_POLLING -- BSP will not skip events.

- Implemented MD event timer drivers:
  - arm (Marvell):
    - CPUTimer0 (periodic and one-shot);
  - mips
    - MIPS32 (periodic and one-shot, per-CPU);
  - powerpc
    - decrementer (periodic and one-shot, per-CPU);
  - sparc64
    - tick/stick (periodic and one-shot, per-CPU);
  - sun4v
    - tick (periodic and one-shot, per-CPU);
  - x86:
    - HPET (periodic and one-shot, optionally per-CPU);
    - i8254 (periodic, optionally one-shot);
    - LAPIC (periodic and one-shot, per-CPU, stops in C3);
    - RTC (periodic).

- `systat -vm 1` on 8-core system before:

```
    1 users    Load  0.50  0.28  0.11                  Sep 22 11:15

Mem:KB    REAL            VIRTUAL                       VN PAGER   SWAP PAGER
        Tot   Share      Tot    Share    Free           in   out     in   out
Act   33932    7432   613508     8848 3795100  count
All  154568    8832 1074444k    33404          pages
Proc:                                                          Interrupts
  r   p   d   s   w   Csw  Trp  Sys  Int  Sof  Flt        cow   16421 total
            40       174    4  135    5   74              zfod         ehci0 16
                                                          ozfod     2 ehci1 23
 0.0%Sys   0.0%Intr  0.0%User  0.0%Nice  100%Idle        %ozfod  2052 cpu0:timer
|    |    |    |    |    |    |    |    |    |         daefr     3 re0 256
                                                          prcfr       ahci0 257
                                        4 dtbuf           totfr  2052 cpu1:timer
Namei     Name-cache   Dir-cache    142132 desvn          react  2052 cpu6:timer
   Calls    hits   %    hits   %       658 numvn          pdwak  2052 cpu3:timer
       3       3 100                  90 frevn          pdpgs  2052 cpu2:timer
                                                          intrn  2052 cpu4:timer
Disks  ada0  ada1  ada2   cd0 pass0 pass1 pass2    154764 wire   2052 cpu5:timer
KB/t   0.00  0.00  0.00  0.00  0.00  0.00  0.00     20728 act    2052 cpu7:timer
tps       0     0     0     0     0     0     0     12388 inact
MB/s   0.00  0.00  0.00  0.00  0.00  0.00  0.00        84 cache
%busy     0     0     0     0     0     0     0   3795016 free
                                                    13232 buf
```

- `systat -vm 1` on 8-core system after:

```
    1 users    Load  0.76  0.31  0.12                  Sep 22 13:18

Mem:KB    REAL            VIRTUAL                       VN PAGER   SWAP PAGER
        Tot   Share      Tot    Share    Free           in   out     in   out
Act   33928    7432   614856     8848 3794480  count
All  154796    8832 1074446k    33404          pages
Proc:                                                          Interrupts
  r   p   d   s   w   Csw  Trp  Sys  Int  Sof  Flt        cow     92 total
            40       171    6  147   93   66              zfod        uart1 3
                                                          ozfod        uart0 4
 0.0%Sys   0.0%Intr  0.0%User  0.0%Nice  100%Idle        %ozfod        ehci0 16
|    |    |    |    |    |    |    |    |    |          daefr      2 ehci1 23
                                                          prcfr      3 re0 256
                                        28 dtbuf          totfr      2 ahci0 257
Namei     Name-cache   Dir-cache    142132 desvn          react     52 hpet0:t0
   Calls    hits   %    hits   %       656 numvn          pdwak      3 hpet0:t1
       3       3 100                  91 frevn          pdpgs      4 hpet0:t2
                                                          intrn      4 hpet0:t3
Disks  ada0  ada1  ada2   cd0 pass0 pass1 pass2    155400 wire       5 hpet0:t4
KB/t   2.00  0.00  0.00  0.00  0.00  0.00  0.00     20860 act        4 hpet0:t5
tps       2     0     0     0     0     0     0     12236 inact      5 hpet0:t6
MB/s   0.00  0.00  0.00  0.00  0.00  0.00  0.00        68 cache      8 hpet0:t7
%busy     0     0     0     0     0     0     0   3794412 free
                                                    13168 buf
```

- Results:
- Temperature of Core i7-870 with boxed cooler with 25C at the room:
  - full load: 85C;
  - idle without PM: 55C;
  - idle w/ P-states+C-states: 32C.
- Time to build net/mpd5 port in one thread on Core i7-870:
  - default: 12,02c;
  - w/ C6 state used: 10,79c (10% more TurboBoost).

- Problems/further work directions:
  - some kernel subsystems generate too much events; it would be nice to remove or group some of them; callout(9) API may need to be extended to allow precision specified;
  - callout(9) call wheel optimized for periodic ticks; difficult to get next scheduled tick time; switch to some tree structure?
  - scheduler depends on both hardclock (via sched_tick()) and statclock (via sched_clock()); it would be nice to be able skip some hardclock/statclock calls also when CPU is active;
  - scheduler unaware about sleeping cores; it would be nice to not schedule to sleeping cores without real need;
  - cache/TLB invalidation IPIs sent to every CPU; it would be nice to avoid it, if possible;
  - write more efficient cpu_idle() methods for some platforms.
  - implement powertop alternative.

- Questions?