TOSHIBA
Leading Innovation >>>

Improvement of Scheduling Granularity for Deadline Scheduler

Yoshitake Kobayashi
Advanced Software Technology Group
Corporate Software Engineering Center
TOSHIBA CORPORATION

Copyright 2012, Toshiba Corporation.

Outline

- Motivation
- Deadline scheduler
- SCHED_DEADLINE and its evaluation
- Budget management
- Conclusion

Motivation

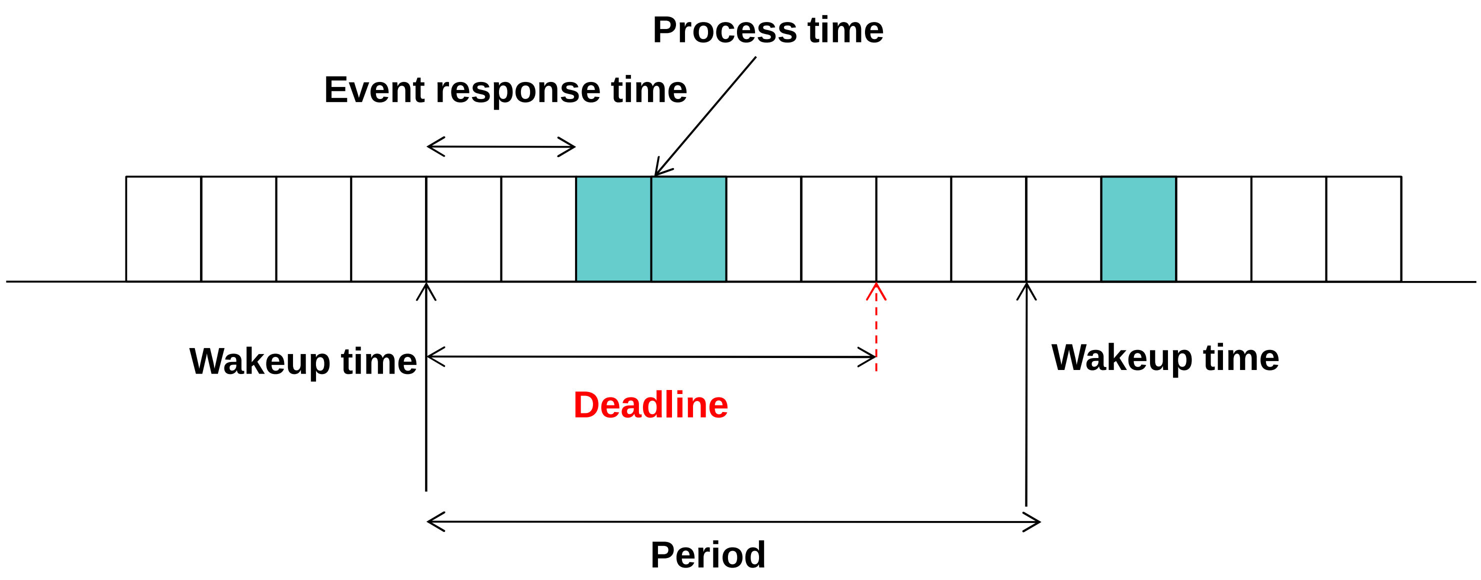
- **We would like to use Linux on control systems**
- **Real-time is one of the most critical topic**

- **Problem statement**
  - Need to evaluate deeply to meet the deadline
  - CPU resource used too much by higher priority tasks

EDF scheduler

Definition of deadline

- **Wakeup time: The time of an event occurred (Ex. Timer interrupt) and target task's state changed to runnable.**
- **Event response time: Interrupt latency**
- **Deadline: The time for a target task must finish**



Earliest Deadline Fisrt scheduling（EDF）

- **The earliest deadline task has the highest priority**
- **Task's priority is dynamically changed and managed**
  - SCHED_FIFO is static priority management
- **Theoretically the total CPU usage by all tasks is up to 100%**
  - Includes the kernel overhead
  - <u>If usage of CPU by all tasks is less than 100%, all tasks meet the deadline</u>

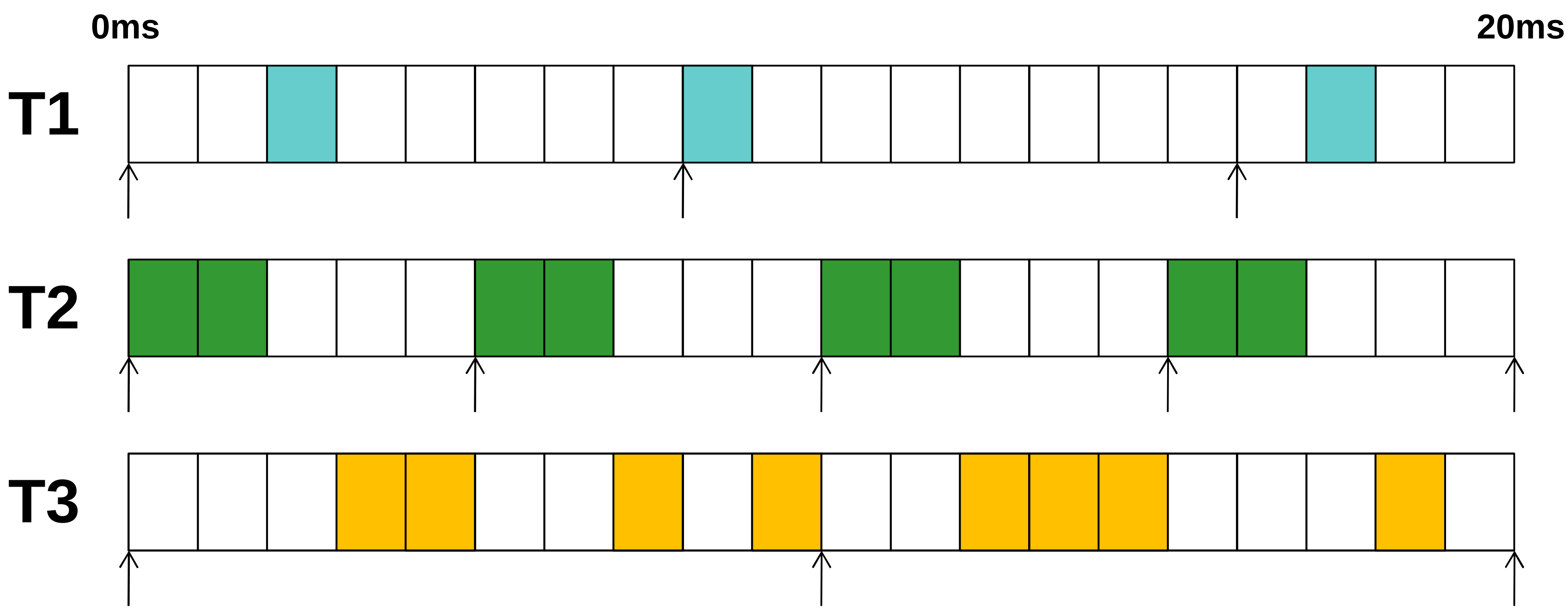
- **Reference**
  - http://en.wikipedia.org/wiki/Earliest_deadline_first_scheduling

An example of EDF Scheduling

- **Task1: budget 1ms period 8ms**
- **Task2: budget 2ms period 5ms**
- **Task3: budget 4ms period 10ms**

CPU usage＝0.925%

＜ 100%



Rate-Monotonic Scheduling（RMS）

- **One of the popular scheduling algorithm for RTOS**
- **Assumptions for task behavior**
  - NO resource sharing such as hardware, a queue, or any kind of semaphore
  - Deterministic deadlines are exactly equal to periods
  - Static priorities (the task with the highest static priority that is runnable immediately preempts all other tasks)
  - Static priorities assigned according to the rate monotonic conventions (tasks with shorter periods/deadlines are given higher priorities)
  - Context switch times and other thread operations are free and have no impact on the model
- **CPU utilization**
  - n: number of periodic tasks, Ti: Release period, Ci: Computation time

$$U = \sum_{i=0}^{n} C_i / T_i \leq n(\sqrt[n]{2} - 1) \xrightarrow[n=\infty]{} \ln 2 \approx 0.69$$

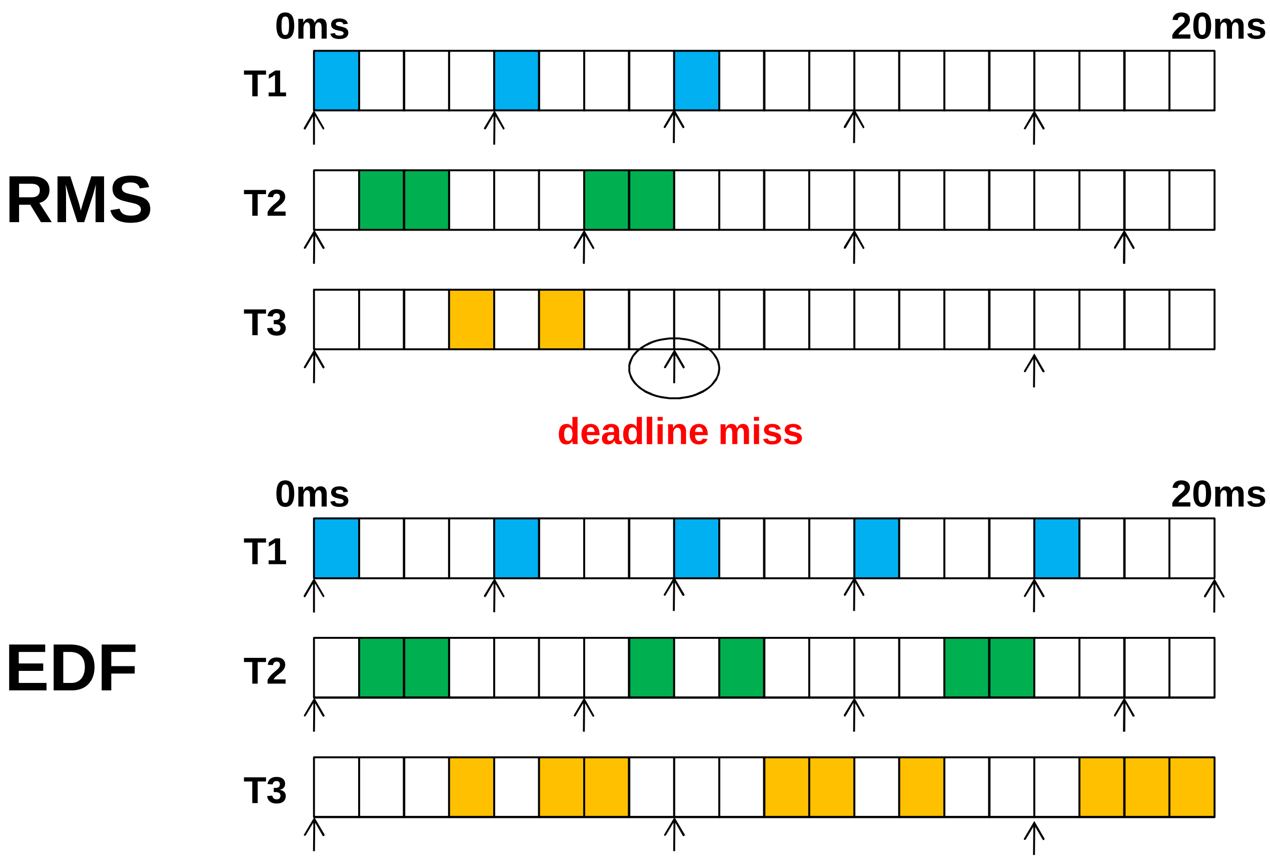
  - CPU utilization depends on the combination of periodic tasks and it is possible to meet the deadline even the CPU utilization is around 80%

- **Reference**
  - http://en.wikipedia.org/wiki/Earliest_deadline_first_scheduling

Compared with the RMS scheduling

- Task1: budget 1ms period 4ms
- Task2: budget 2ms period 6ms
- Task3: budget 3ms period 8ms

CPU usage＝0.958%



Comparison of deadline algorithms

	Advantage	Disadvantage
RMS	Easier to implement	Evaluation for scheduling possibility is required to meet the deadline
EDF	No evaluation for scheduling possibility is required to meet the deadline	Difficult to implement

SCHED_DEADLINE

- **http://www.evidence.eu.com/sched_deadline.html**
  - Implements the EDF scheduling algorithm
  - Posted to LKML by Dario Faggioli and Juri Lelli
  - Latest version is V6 (2012/10/24)
    - But V2 and V4 are used in our evaluation.

- **Key features of SCHED_DEADLINE**
  - <u>Temporal isolation</u>
    - The temporal behavior of each task (i.e., its ability to meet its deadlines) is not affected by the behavior of any other task in the system
    - Each task is characterized by the following aspects:
      - Budget: sched_runtime
      - Period: sched_period, equal to its deadline

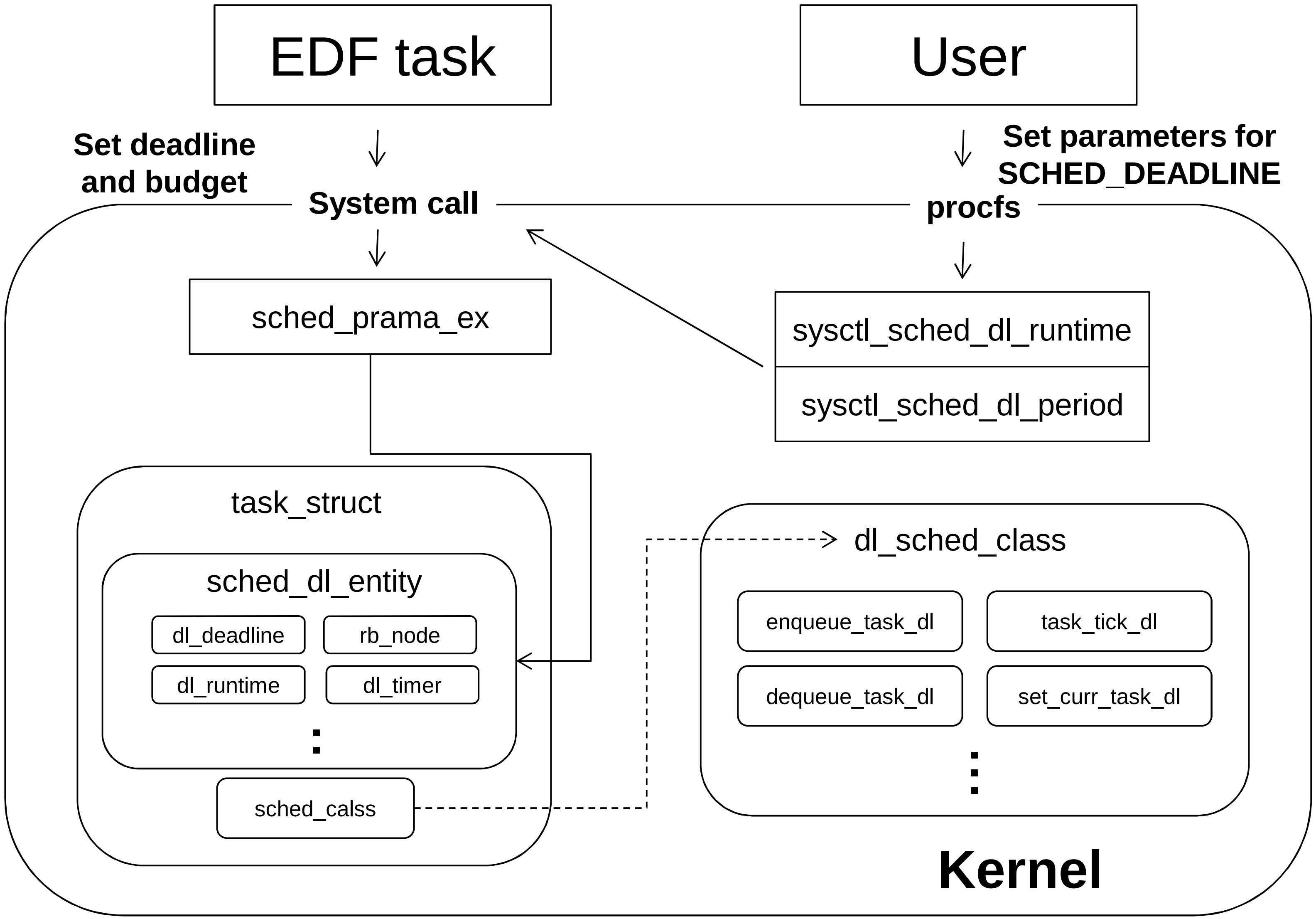
Build SCHED_DEADLINE(linux kernel)

- **Get rt-deadline from the following place**
  - git clone git://gitorious.org/rt-deadline (for V2)

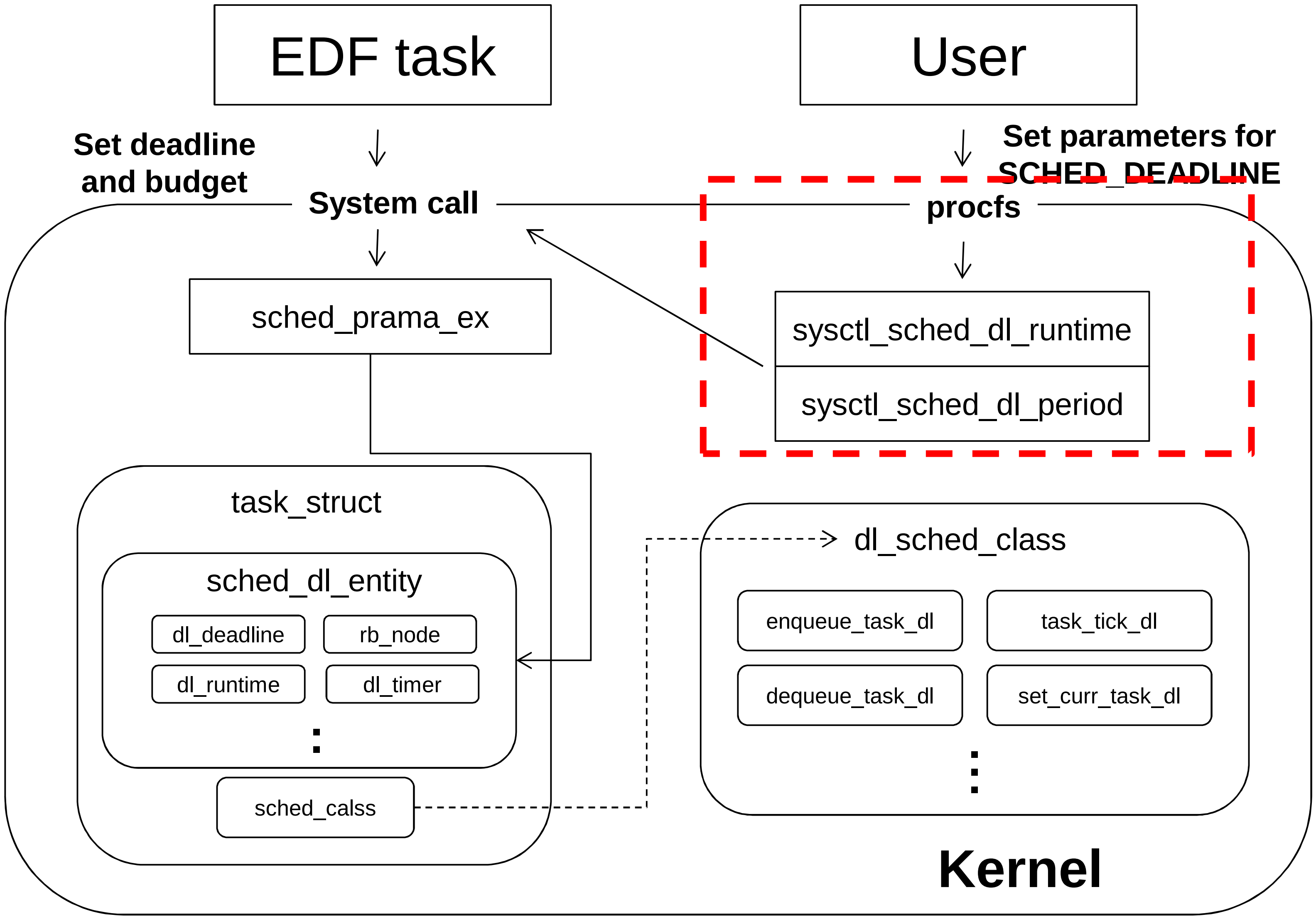
- **Kernel configuration**
  - CONFIG_EXPERIMENTAL = y
  - CONFIG_CGROUPS = y
  - CONFIG_CGROUP_SCHED = n
  - CONFIG_HIGH_RES_TIMERS = y
  - CONFIG_PREEMPT = y
  - CONFIG_PREEMPT_RT = y
  - CONFIG_HZ_1000 = y

- **Note: For V6**
  - git clone git://github.com/jlelli/sched-deadline.git

Overview of SCHED_DEADLINE



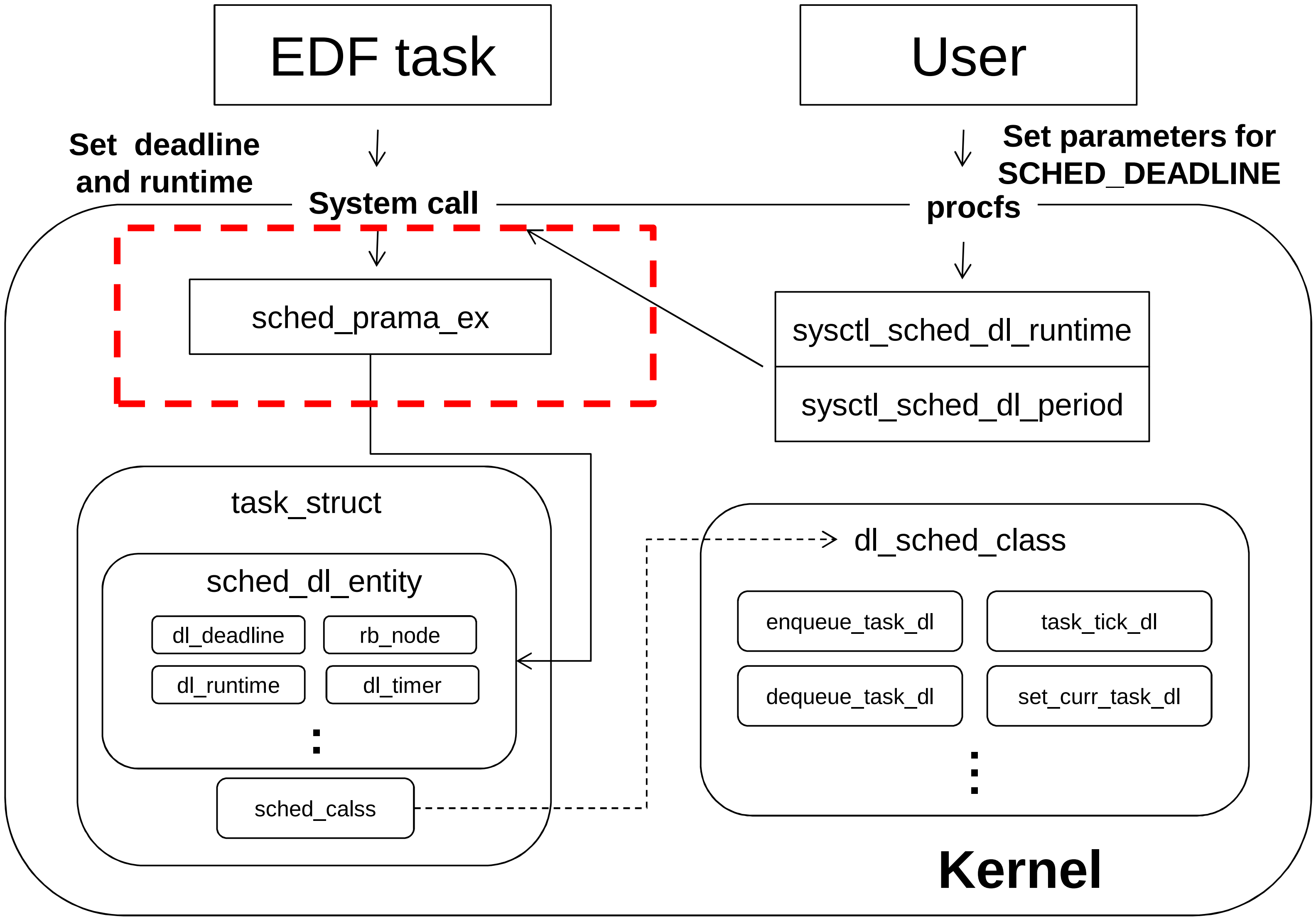
Overview of SCHED_DEADLINE



Setting CPU utilization for EDF tasks

- **Parameters can be setted via procfs**
  - CPU utilization for rt(SCHED_FIFO and SCHED_RR) and dl(SCHED_DEADLINE) should be under 100%
  - Parameters for EDF scheduler
    - /proc/sys/kernel/sched_dl_period_us
    - /proc/sys/kernel/sched_dl_runtime_us
- **<u>When a task requires more than above limit, the task cannot submit to run</u>**
- **An example setting (rt: 50%, dl:50%)**
  - # echo 500000 > /proc/sys/kernel/sched_rt_runtime_us (500ms)
    # echo 100000 > /proc/sys/kernel/sched_dl_period_us (100ms)
    # echo 50000 > /proc/sys/kernel/sched_dl_runtime_us (50ms)

Overview of SCHED_DEADLINE

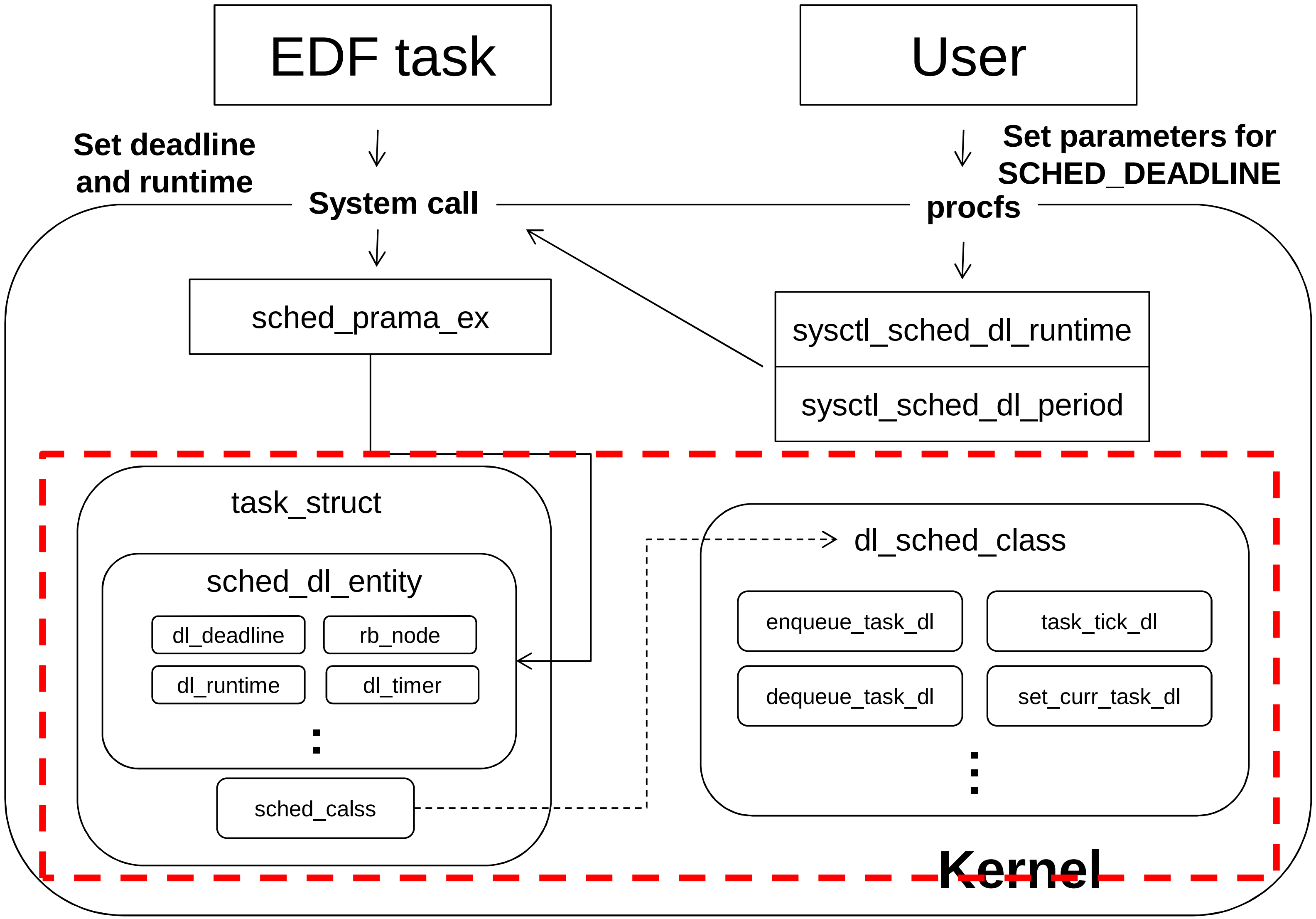


Run a EDF task

- **Schedtool**
  - # schedtool -E -t 10000:100000 -a 0 -e ./yes
  - Options
    -E： a task runs on SCHED_DEADLINE
    -t： <execution time> and <period> in micro seconds
    -a： Affinity mask
    -e： command

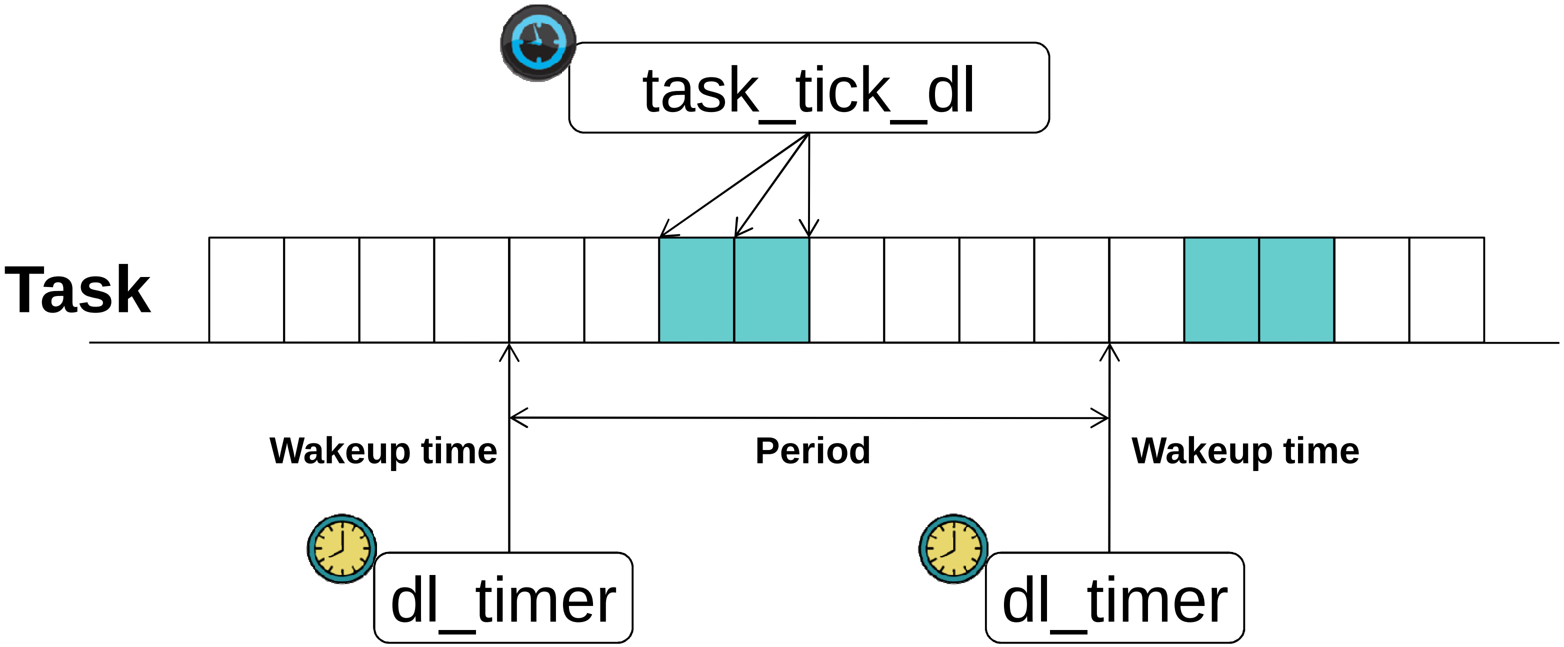
- **System call**
  - sched_setscheduler_ex()

Budget management for EDF tasks

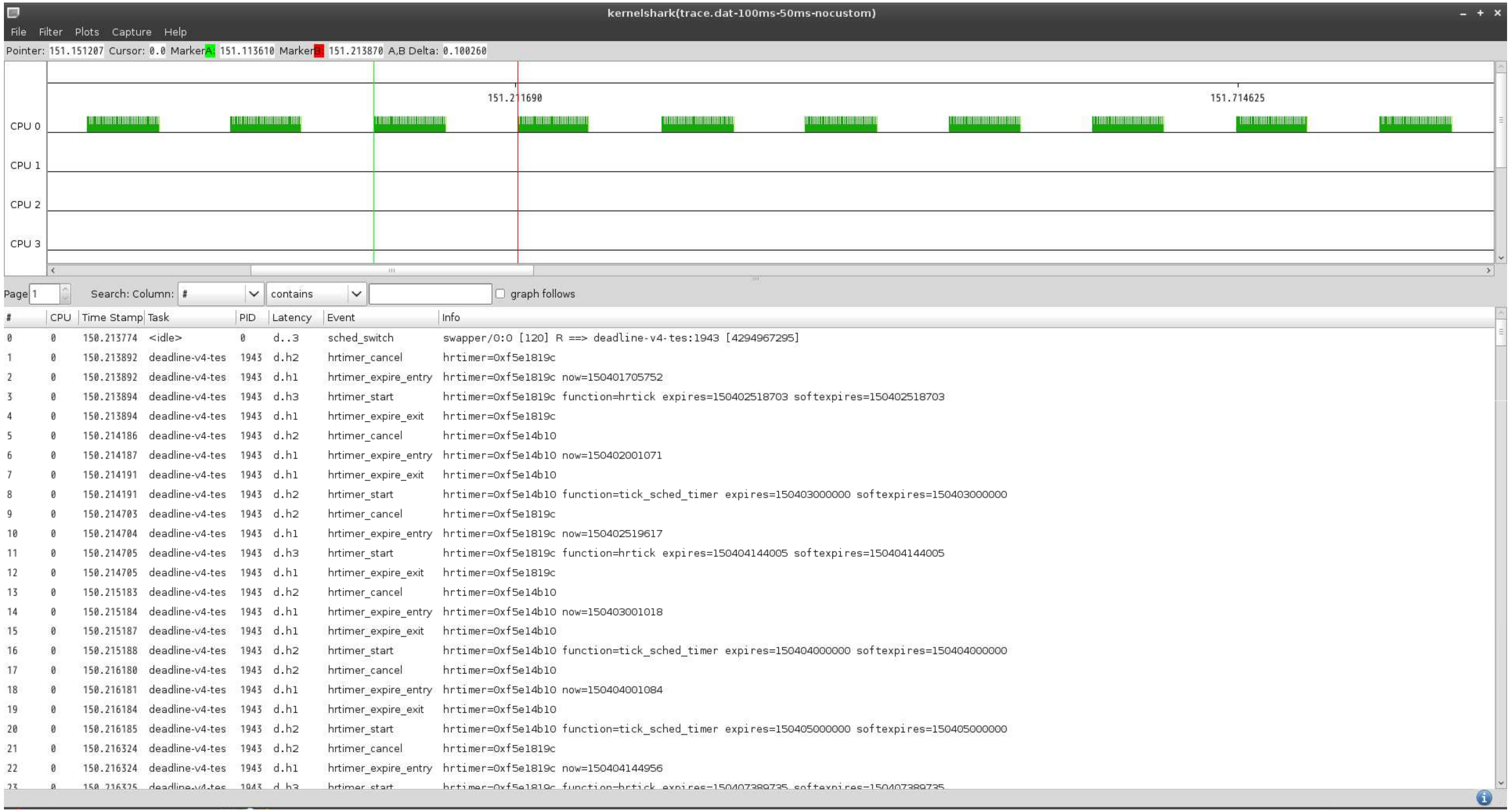


Budget management for EDF tasks

- **Each task on SCHED_DEADLINE has budget which allows it to use CPU**
- **Budget management**
  - Refill budget : dl_timer (high resolution timer)
  - Use budget : task_tick_dl (tick bsed)

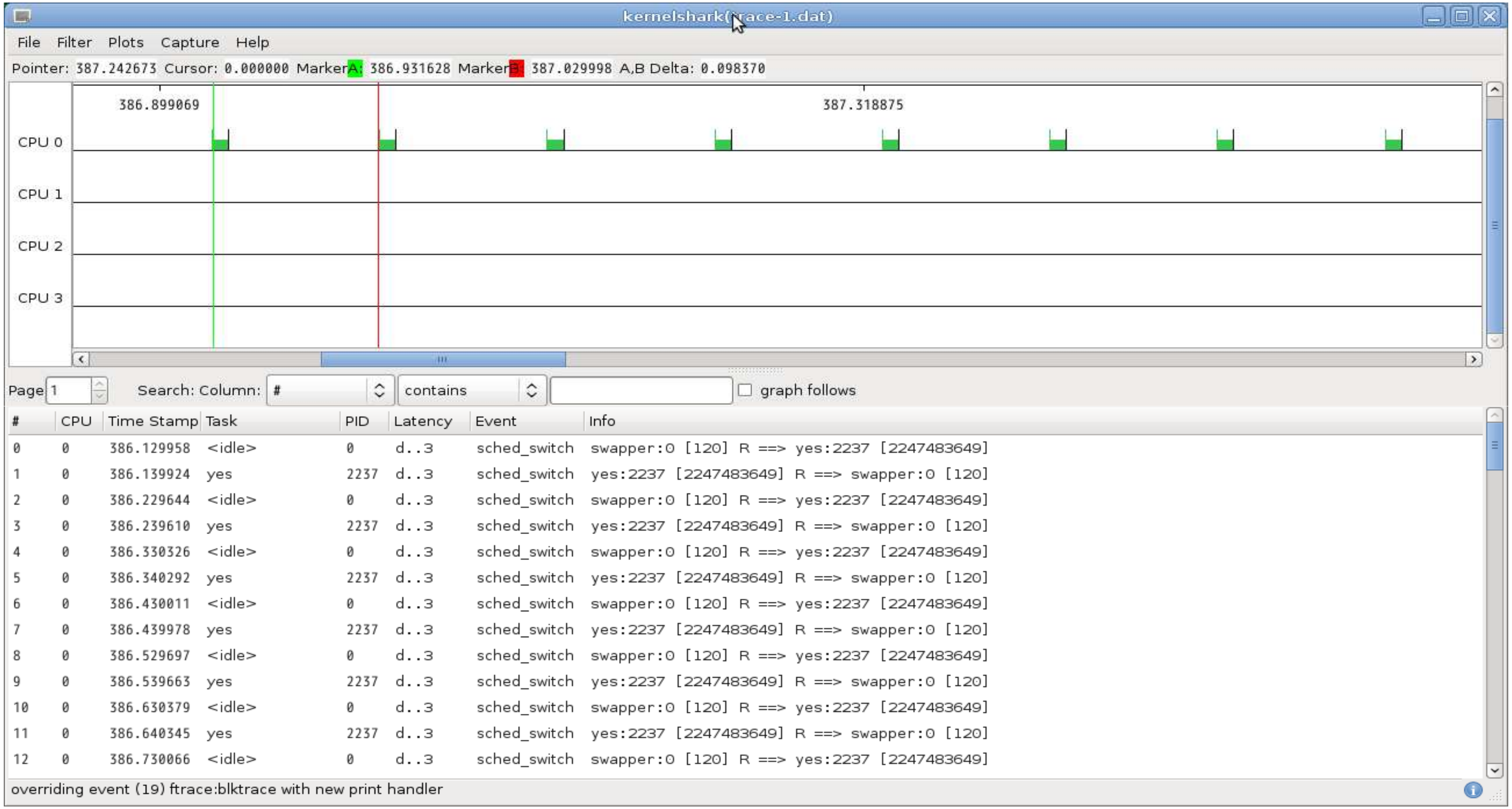


Evaluation (Period:100ms, Budget: 50ms)



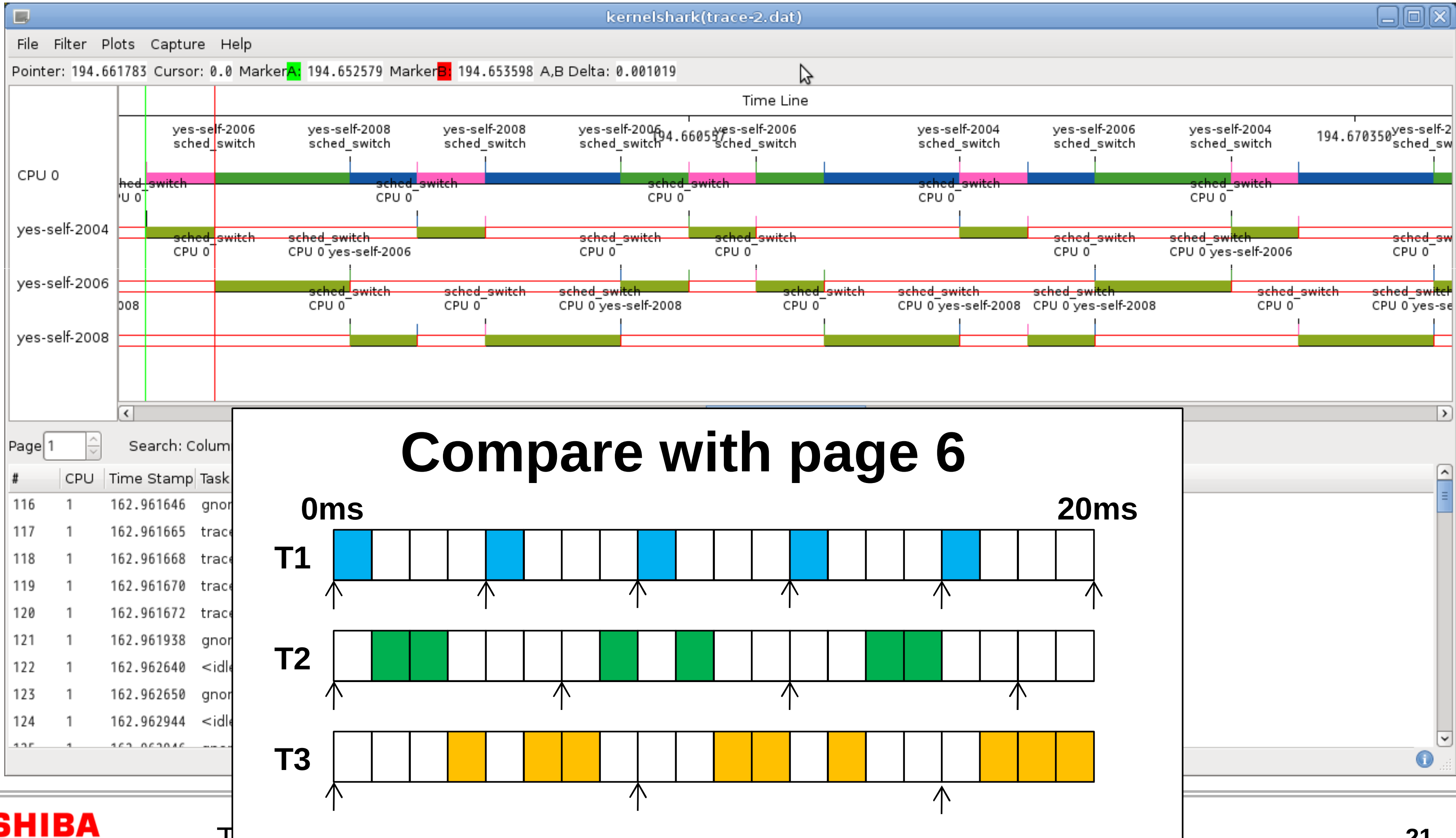
#	CPU	Time Stamp	Task	PID	Latency	Event	Info
0	0	150.213774	<idle>	0	d..3	sched_switch	swapper/0:0 [120] R ==> deadline-v4-tes:1943 [4294967295]
1	0	150.213892	deadline-v4-tes	1943	d.h2	hrtimer_cancel	hrtimer=0xf5e1819c
2	0	150.213892	deadline-v4-tes	1943	d.h1	hrtimer_expire_entry	hrtimer=0xf5e1819c now=150401705752
3	0	150.213894	deadline-v4-tes	1943	d.h3	hrtimer_start	hrtimer=0xf5e1819c function=hrtick expires=150402518703 softexpires=150402518703
4	0	150.213894	deadline-v4-tes	1943	d.h1	hrtimer_expire_exit	hrtimer=0xf5e1819c
5	0	150.214186	deadline-v4-tes	1943	d.h2	hrtimer_cancel	hrtimer=0xf5e14b10
6	0	150.214187	deadline-v4-tes	1943	d.h1	hrtimer_expire_entry	hrtimer=0xf5e14b10 now=150402001071
7	0	150.214191	deadline-v4-tes	1943	d.h1	hrtimer_expire_exit	hrtimer=0xf5e14b10
8	0	150.214191	deadline-v4-tes	1943	d.h2	hrtimer_start	hrtimer=0xf5e14b10 function=tick_sched_timer expires=150403000000 softexpires=150403000000
9	0	150.214703	deadline-v4-tes	1943	d.h2	hrtimer_cancel	hrtimer=0xf5e1819c
10	0	150.214704	deadline-v4-tes	1943	d.h1	hrtimer_expire_entry	hrtimer=0xf5e1819c now=150402519617
11	0	150.214705	deadline-v4-tes	1943	d.h3	hrtimer_start	hrtimer=0xf5e1819c function=hrtick expires=150404144005 softexpires=150404144005
12	0	150.214705	deadline-v4-tes	1943	d.h1	hrtimer_expire_exit	hrtimer=0xf5e1819c
13	0	150.215183	deadline-v4-tes	1943	d.h2	hrtimer_cancel	hrtimer=0xf5e14b10
14	0	150.215184	deadline-v4-tes	1943	d.h1	hrtimer_expire_entry	hrtimer=0xf5e14b10 now=150403001018
15	0	150.215187	deadline-v4-tes	1943	d.h1	hrtimer_expire_exit	hrtimer=0xf5e14b10
16	0	150.215188	deadline-v4-tes	1943	d.h2	hrtimer_start	hrtimer=0xf5e14b10 function=tick_sched_timer expires=150404000000 softexpires=150404000000
17	0	150.216180	deadline-v4-tes	1943	d.h2	hrtimer_cancel	hrtimer=0xf5e14b10
18	0	150.216181	deadline-v4-tes	1943	d.h1	hrtimer_expire_entry	hrtimer=0xf5e14b10 now=150404001084
19	0	150.216184	deadline-v4-tes	1943	d.h1	hrtimer_expire_exit	hrtimer=0xf5e14b10
20	0	150.216185	deadline-v4-tes	1943	d.h2	hrtimer_start	hrtimer=0xf5e14b10 function=tick_sched_timer expires=150405000000 softexpires=150405000000
21	0	150.216324	deadline-v4-tes	1943	d.h2	hrtimer_cancel	hrtimer=0xf5e1819c
22	0	150.216324	deadline-v4-tes	1943	d.h1	hrtimer_expire_entry	hrtimer=0xf5e1819c now=150404144956
23	0	150.216325	deadline-v4-tes	1943	d.h3	hrtimer_start	hrtimer=0xf5e1819c function=hrtick expires=150407389735 softexpires=150407389735

Evaluation (Period: 100ms , Budget: 10ms)

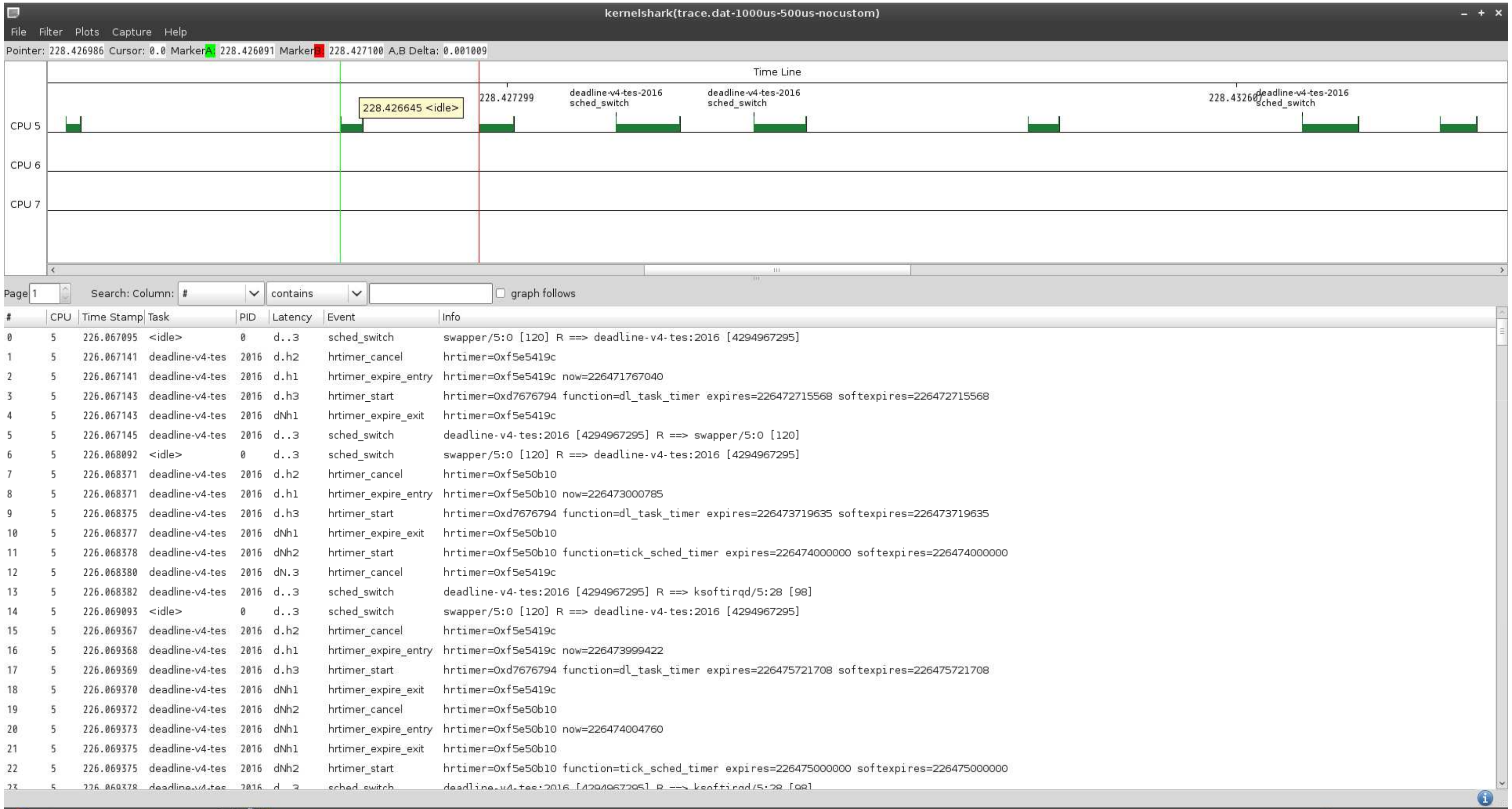


Evaluation

- **Task T1: budget 1ms period 4ms**
- **Task T2: budget 2ms period 6ms**
- **Task T3: budget 3ms period 8ms**

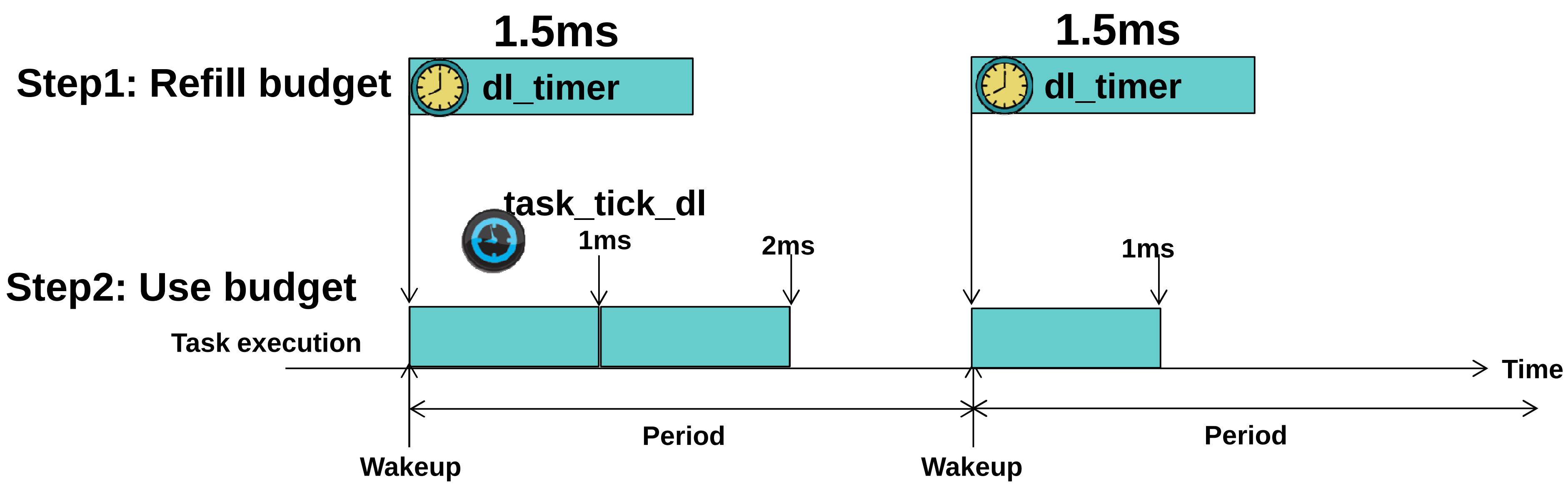


Evaluation (Period: 1ms, Budget: 0.5ms)



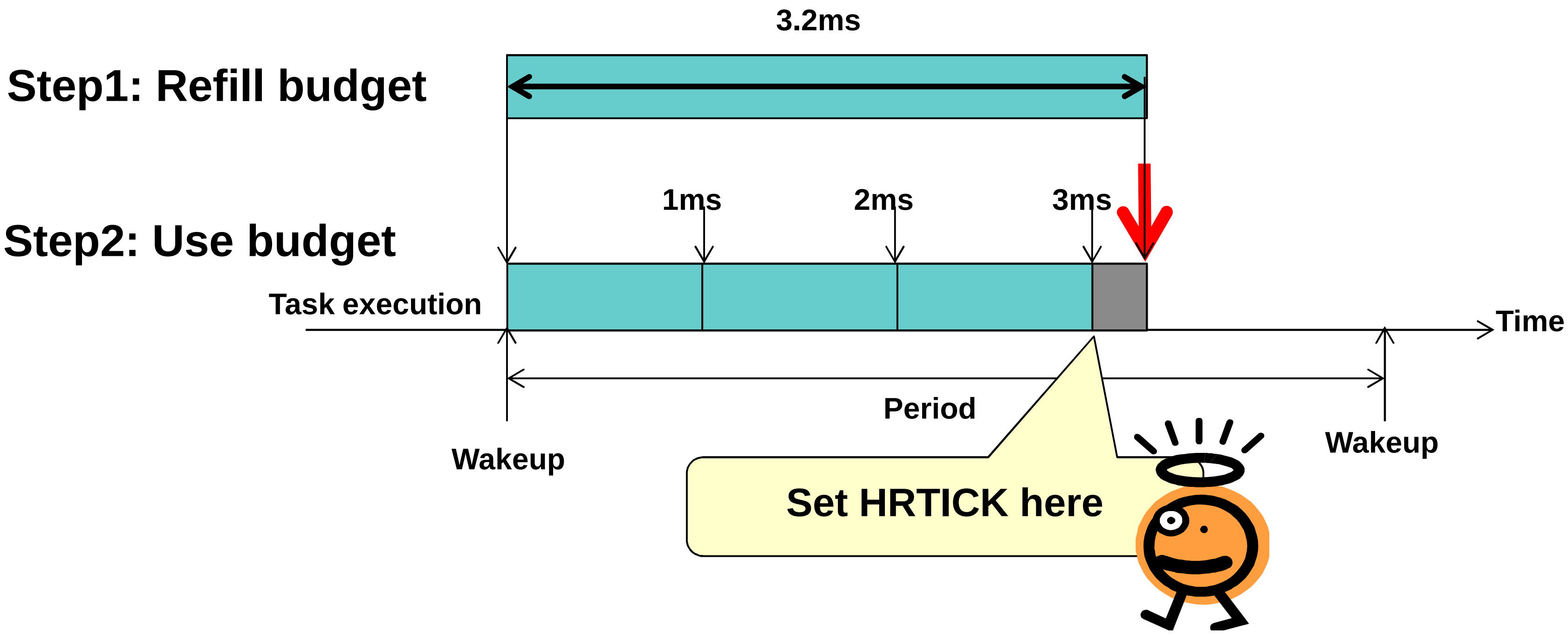
Budget management for EDF tasks

- **Each task on SCHED_DEADLINE has budget which allows it to use CPU**
- **Budget management**
  - Refill budget : dl_timer (high resolution timer)
  - Use budget : task_tick_dl (tick based)
- **An Issue**
  - Difficult to keep task's budget if the budget has micro seconds granularity

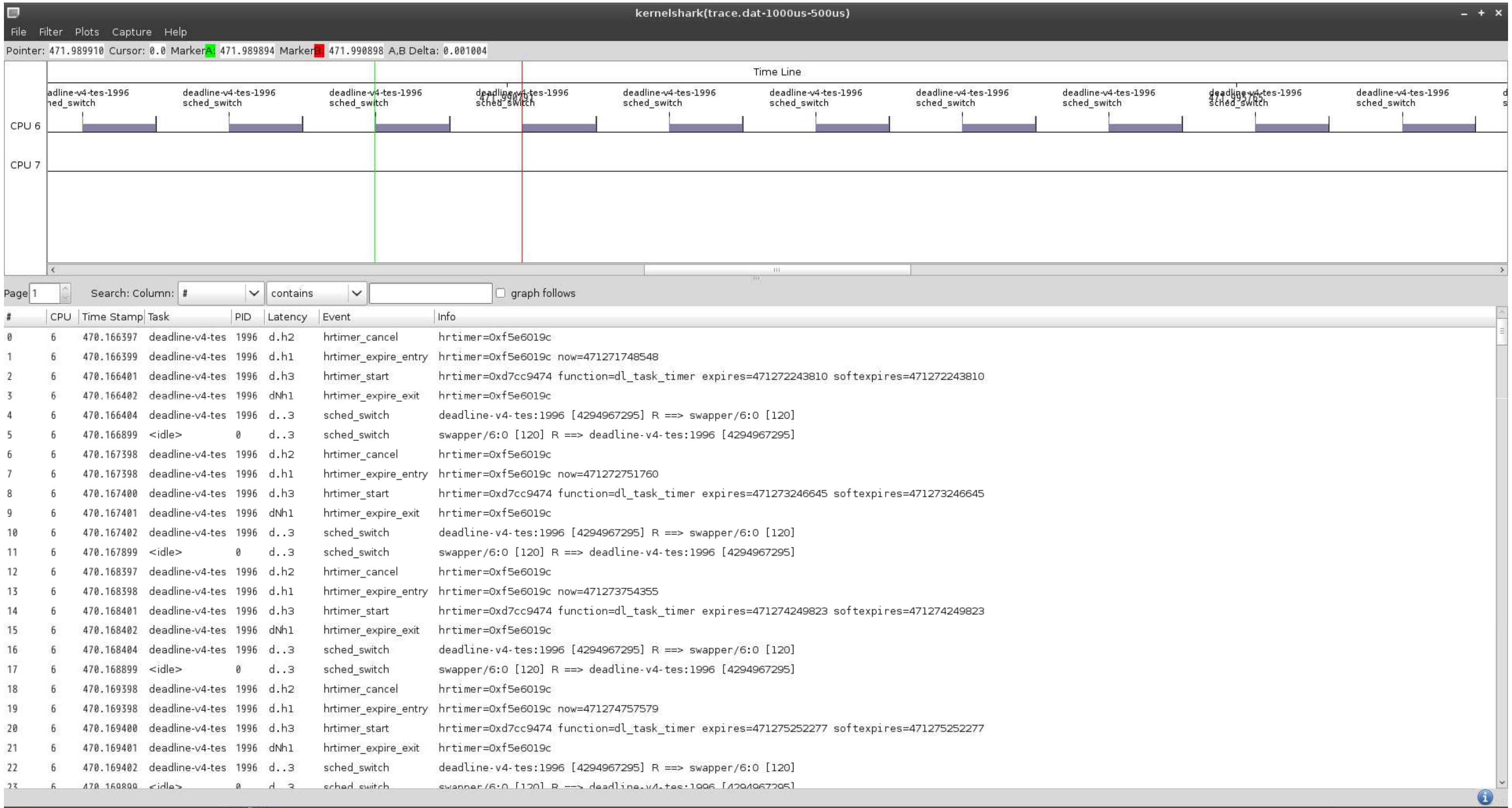


Support for micro seconds granularity

- **Overview**
  - When a task's budget is less than 1ms, set HRTICK for the rest of budget



Evaluation (Cycle: 1ms, Budget: 0.5ms)



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
0	6	470.166397	deadline-v4-tes	1996	d.h2	hrtimer_cancel	hrtimer=0xf5e6019c
1	6	470.166399	deadline-v4-tes	1996	d.h1	hrtimer_expire_entry	hrtimer=0xf5e6019c now=471271748548
2	6	470.166401	deadline-v4-tes	1996	d.h3	hrtimer_start	hrtimer=0xd7cc9474 function=dl_task_timer expires=471272243810 softexpires=471272243810
3	6	470.166402	deadline-v4-tes	1996	dNh1	hrtimer_expire_exit	hrtimer=0xf5e6019c
4	6	470.166404	deadline-v4-tes	1996	d..3	sched_switch	deadline-v4-tes:1996 [4294967295] R ==> swapper/6:0 [120]
5	6	470.166899	<idle>	0	d..3	sched_switch	swapper/6:0 [120] R ==> deadline-v4-tes:1996 [4294967295]
6	6	470.167398	deadline-v4-tes	1996	d.h2	hrtimer_cancel	hrtimer=0xf5e6019c
7	6	470.167398	deadline-v4-tes	1996	d.h1	hrtimer_expire_entry	hrtimer=0xf5e6019c now=471272751760
8	6	470.167400	deadline-v4-tes	1996	d.h3	hrtimer_start	hrtimer=0xd7cc9474 function=dl_task_timer expires=471273246645 softexpires=471273246645
9	6	470.167401	deadline-v4-tes	1996	dNh1	hrtimer_expire_exit	hrtimer=0xf5e6019c
10	6	470.167402	deadline-v4-tes	1996	d..3	sched_switch	deadline-v4-tes:1996 [4294967295] R ==> swapper/6:0 [120]
11	6	470.167899	<idle>	0	d..3	sched_switch	swapper/6:0 [120] R ==> deadline-v4-tes:1996 [4294967295]
12	6	470.168397	deadline-v4-tes	1996	d.h2	hrtimer_cancel	hrtimer=0xf5e6019c
13	6	470.168398	deadline-v4-tes	1996	d.h1	hrtimer_expire_entry	hrtimer=0xf5e6019c now=471273754355
14	6	470.168401	deadline-v4-tes	1996	d.h3	hrtimer_start	hrtimer=0xd7cc9474 function=dl_task_timer expires=471274249823 softexpires=471274249823
15	6	470.168402	deadline-v4-tes	1996	dNh1	hrtimer_expire_exit	hrtimer=0xf5e6019c
16	6	470.168404	deadline-v4-tes	1996	d..3	sched_switch	deadline-v4-tes:1996 [4294967295] R ==> swapper/6:0 [120]
17	6	470.168899	<idle>	0	d..3	sched_switch	swapper/6:0 [120] R ==> deadline-v4-tes:1996 [4294967295]
18	6	470.169398	deadline-v4-tes	1996	d.h2	hrtimer_cancel	hrtimer=0xf5e6019c
19	6	470.169398	deadline-v4-tes	1996	d.h1	hrtimer_expire_entry	hrtimer=0xf5e6019c now=471274757579
20	6	470.169400	deadline-v4-tes	1996	d.h3	hrtimer_start	hrtimer=0xd7cc9474 function=dl_task_timer expires=471275252277 softexpires=471275252277
21	6	470.169401	deadline-v4-tes	1996	dNh1	hrtimer_expire_exit	hrtimer=0xf5e6019c
22	6	470.169402	deadline-v4-tes	1996	d..3	sched_switch	deadline-v4-tes:1996 [4294967295] R ==> swapper/6:0 [120]
23	6	470.169899	<idle>	0	d..3	sched_switch	swapper/6:0 [120] R ==> deadline-v4-tes:1996 [4294967295]

Advantage and Disadvantage

- **Advantage**
  - Easy to support high resolution budget
- **Disadvantage**
  - Increase overhead

Conclusion

- **SCHED_DEADLINE is useful for real time systems**
- **An Enhancement for budget management**
  - Support fine grained budget such as 100 micro seconds
  - HRTICK is needed to support fine grained budget

- **What we've done:**
  - Backport the SCHED_DEADLINE v4 to kernel v3.0-rt
    - Because v3.0 is the base version of LTSI
  - Implement this improvement into SCHED_DEADLINE v4 on kernel v3.0-rt
  - Forwardport it to original SCHED_DEADLINE v4
  - Send a patch to the author of SCHED_DEADLINE

Thank you