struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) {

#ifdef [**CONFIG\_THREAD\_INFO\_IN\_TASK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_THREAD_INFO_IN_TASK)

*/\**

*\* For reasons of header soup (see current\_thread\_info()), this*

*\* must be the first element of task\_struct.*

*\*/*

struct [**thread\_info**](https://elixir.bootlin.com/linux/latest/C/ident/thread_info) [**thread\_info**](https://elixir.bootlin.com/linux/latest/C/ident/thread_info);

#endif

unsigned int [**\_\_state**](https://elixir.bootlin.com/linux/latest/C/ident/__state);

#ifdef [**CONFIG\_PREEMPT\_RT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PREEMPT_RT)

*/\* saved state for "spinlock sleepers" \*/*

unsigned int [**saved\_state**](https://elixir.bootlin.com/linux/latest/C/ident/saved_state);

#endif

*/\**

*\* This begins the randomizable portion of task\_struct. Only*

*\* scheduling-critical items should be added above here.*

*\*/*

[**randomized\_struct\_fields\_start**](https://elixir.bootlin.com/linux/latest/C/ident/randomized_struct_fields_start)

void \*[**stack**](https://elixir.bootlin.com/linux/latest/C/ident/stack);

[**refcount\_t**](https://elixir.bootlin.com/linux/latest/C/ident/refcount_t) [**usage**](https://elixir.bootlin.com/linux/latest/C/ident/usage);

*/\* Per task flags (PF\_\*), defined further below: \*/*

unsigned int flags;

unsigned int [**ptrace**](https://elixir.bootlin.com/linux/latest/C/ident/ptrace);

#ifdef [**CONFIG\_SMP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_SMP)

int [**on\_cpu**](https://elixir.bootlin.com/linux/latest/C/ident/on_cpu);

struct [**\_\_call\_single\_node**](https://elixir.bootlin.com/linux/latest/C/ident/__call_single_node) [**wake\_entry**](https://elixir.bootlin.com/linux/latest/C/ident/wake_entry);

unsigned int [**wakee\_flips**](https://elixir.bootlin.com/linux/latest/C/ident/wakee_flips);

unsigned long [**wakee\_flip\_decay\_ts**](https://elixir.bootlin.com/linux/latest/C/ident/wakee_flip_decay_ts);

struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) \***[last\_wakee](https://elixir.bootlin.com/linux/latest/C/ident/last_wakee)**;

*/\**

*\* recent\_used\_cpu is initially set as the last CPU used by a task*

*\* that wakes affine another task. Waker/wakee relationships can*

*\* push tasks around a CPU where each wakeup moves to the next one.*

*\* Tracking a recently used CPU allows a quick search for a recently*

*\* used CPU that may be idle.*

*\*/*

int [**recent\_used\_cpu**](https://elixir.bootlin.com/linux/latest/C/ident/recent_used_cpu);

int [**wake\_cpu**](https://elixir.bootlin.com/linux/latest/C/ident/wake_cpu);

#endif

int [**on\_rq**](https://elixir.bootlin.com/linux/latest/C/ident/on_rq);

int [**prio**](https://elixir.bootlin.com/linux/latest/C/ident/prio);

int [**static\_prio**](https://elixir.bootlin.com/linux/latest/C/ident/static_prio);

int [**normal\_prio**](https://elixir.bootlin.com/linux/latest/C/ident/normal_prio);

unsigned int [**rt\_priority**](https://elixir.bootlin.com/linux/latest/C/ident/rt_priority);

struct [**sched\_entity**](https://elixir.bootlin.com/linux/latest/C/ident/sched_entity) [**se**](https://elixir.bootlin.com/linux/latest/C/ident/se);

struct [**sched\_rt\_entity**](https://elixir.bootlin.com/linux/latest/C/ident/sched_rt_entity) [**rt**](https://elixir.bootlin.com/linux/latest/C/ident/rt);

struct [**sched\_dl\_entity**](https://elixir.bootlin.com/linux/latest/C/ident/sched_dl_entity) [**dl**](https://elixir.bootlin.com/linux/latest/C/ident/dl);

const struct [**sched\_class**](https://elixir.bootlin.com/linux/latest/C/ident/sched_class) \***[sched\_class](https://elixir.bootlin.com/linux/latest/C/ident/sched_class)**;

#ifdef [**CONFIG\_SCHED\_CORE**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_SCHED_CORE)

struct [**rb\_node**](https://elixir.bootlin.com/linux/latest/C/ident/rb_node) [**core\_node**](https://elixir.bootlin.com/linux/latest/C/ident/core_node);

unsigned long [**core\_cookie**](https://elixir.bootlin.com/linux/latest/C/ident/core_cookie);

unsigned int [**core\_occupation**](https://elixir.bootlin.com/linux/latest/C/ident/core_occupation);

#endif

#ifdef [**CONFIG\_CGROUP\_SCHED**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_CGROUP_SCHED)

struct [**task\_group**](https://elixir.bootlin.com/linux/latest/C/ident/task_group) \***[sched\_task\_group](https://elixir.bootlin.com/linux/latest/C/ident/sched_task_group)**;

#endif

#ifdef [**CONFIG\_UCLAMP\_TASK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_UCLAMP_TASK)

*/\**

*\* Clamp values requested for a scheduling entity.*

*\* Must be updated with task\_rq\_lock() held.*

*\*/*

struct [**uclamp\_se**](https://elixir.bootlin.com/linux/latest/C/ident/uclamp_se) [**uclamp\_req**](https://elixir.bootlin.com/linux/latest/C/ident/uclamp_req)[[**UCLAMP\_CNT**](https://elixir.bootlin.com/linux/latest/C/ident/UCLAMP_CNT)];

*/\**

*\* Effective clamp values used for a scheduling entity.*

*\* Must be updated with task\_rq\_lock() held.*

*\*/*

struct [**uclamp\_se**](https://elixir.bootlin.com/linux/latest/C/ident/uclamp_se) [**uclamp**](https://elixir.bootlin.com/linux/latest/C/ident/uclamp)[[**UCLAMP\_CNT**](https://elixir.bootlin.com/linux/latest/C/ident/UCLAMP_CNT)];

#endif

struct [**sched\_statistics**](https://elixir.bootlin.com/linux/latest/C/ident/sched_statistics) [**stats**](https://elixir.bootlin.com/linux/latest/C/ident/stats);

#ifdef [**CONFIG\_PREEMPT\_NOTIFIERS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PREEMPT_NOTIFIERS)

*/\* List of struct preempt\_notifier: \*/*

struct [**hlist\_head**](https://elixir.bootlin.com/linux/latest/C/ident/hlist_head) [**preempt\_notifiers**](https://elixir.bootlin.com/linux/latest/C/ident/preempt_notifiers);

#endif

#ifdef [**CONFIG\_BLK\_DEV\_IO\_TRACE**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_BLK_DEV_IO_TRACE)

unsigned int [**btrace\_seq**](https://elixir.bootlin.com/linux/latest/C/ident/btrace_seq);

#endif

unsigned int [**policy**](https://elixir.bootlin.com/linux/latest/C/ident/policy);

int [**nr\_cpus\_allowed**](https://elixir.bootlin.com/linux/latest/C/ident/nr_cpus_allowed);

const [**cpumask\_t**](https://elixir.bootlin.com/linux/latest/C/ident/cpumask_t) \***[cpus\_ptr](https://elixir.bootlin.com/linux/latest/C/ident/cpus_ptr)**;

[**cpumask\_t**](https://elixir.bootlin.com/linux/latest/C/ident/cpumask_t) \***[user\_cpus\_ptr](https://elixir.bootlin.com/linux/latest/C/ident/user_cpus_ptr)**;

[**cpumask\_t**](https://elixir.bootlin.com/linux/latest/C/ident/cpumask_t) [**cpus\_mask**](https://elixir.bootlin.com/linux/latest/C/ident/cpus_mask);

void \***[migration\_pending](https://elixir.bootlin.com/linux/latest/C/ident/migration_pending)**;

#ifdef [**CONFIG\_SMP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_SMP)

unsigned [**short**](https://elixir.bootlin.com/linux/latest/C/ident/short) [**migration\_disabled**](https://elixir.bootlin.com/linux/latest/C/ident/migration_disabled);

#endif

unsigned [**short**](https://elixir.bootlin.com/linux/latest/C/ident/short) [**migration\_flags**](https://elixir.bootlin.com/linux/latest/C/ident/migration_flags);

#ifdef [**CONFIG\_PREEMPT\_RCU**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PREEMPT_RCU)

int [**rcu\_read\_lock\_nesting**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_read_lock_nesting);

union [**rcu\_special**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_special) [**rcu\_read\_unlock\_special**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_read_unlock_special);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**rcu\_node\_entry**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_node_entry);

struct [**rcu\_node**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_node) \***[rcu\_blocked\_node](https://elixir.bootlin.com/linux/latest/C/ident/rcu_blocked_node)**;

#endif */\* #ifdef CONFIG\_PREEMPT\_RCU \*/*

#ifdef [**CONFIG\_TASKS\_RCU**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TASKS_RCU)

unsigned long [**rcu\_tasks\_nvcsw**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_tasks_nvcsw);

[**u8**](https://elixir.bootlin.com/linux/latest/C/ident/u8) [**rcu\_tasks\_holdout**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_tasks_holdout);

[**u8**](https://elixir.bootlin.com/linux/latest/C/ident/u8) [**rcu\_tasks\_idx**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_tasks_idx);

int [**rcu\_tasks\_idle\_cpu**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_tasks_idle_cpu);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**rcu\_tasks\_holdout\_list**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_tasks_holdout_list);

#endif */\* #ifdef CONFIG\_TASKS\_RCU \*/*

#ifdef [**CONFIG\_TASKS\_TRACE\_RCU**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TASKS_TRACE_RCU)

int [**trc\_reader\_nesting**](https://elixir.bootlin.com/linux/latest/C/ident/trc_reader_nesting);

int [**trc\_ipi\_to\_cpu**](https://elixir.bootlin.com/linux/latest/C/ident/trc_ipi_to_cpu);

union [**rcu\_special**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_special) [**trc\_reader\_special**](https://elixir.bootlin.com/linux/latest/C/ident/trc_reader_special);

[**bool**](https://elixir.bootlin.com/linux/latest/C/ident/bool) [**trc\_reader\_checked**](https://elixir.bootlin.com/linux/latest/C/ident/trc_reader_checked);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**trc\_holdout\_list**](https://elixir.bootlin.com/linux/latest/C/ident/trc_holdout_list);

#endif */\* #ifdef CONFIG\_TASKS\_TRACE\_RCU \*/*

struct [**sched\_info**](https://elixir.bootlin.com/linux/latest/C/ident/sched_info) [**sched\_info**](https://elixir.bootlin.com/linux/latest/C/ident/sched_info);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**tasks**](https://elixir.bootlin.com/linux/latest/C/ident/tasks);

#ifdef [**CONFIG\_SMP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_SMP)

struct [**plist\_node**](https://elixir.bootlin.com/linux/latest/C/ident/plist_node) [**pushable\_tasks**](https://elixir.bootlin.com/linux/latest/C/ident/pushable_tasks);

struct [**rb\_node**](https://elixir.bootlin.com/linux/latest/C/ident/rb_node) [**pushable\_dl\_tasks**](https://elixir.bootlin.com/linux/latest/C/ident/pushable_dl_tasks);

#endif

struct [**mm\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/mm_struct) \*[**mm**](https://elixir.bootlin.com/linux/latest/C/ident/mm);

struct [**mm\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/mm_struct) \***[active\_mm](https://elixir.bootlin.com/linux/latest/C/ident/active_mm)**;

*/\* Per-thread vma caching: \*/*

struct [**vmacache**](https://elixir.bootlin.com/linux/latest/C/ident/vmacache) [**vmacache**](https://elixir.bootlin.com/linux/latest/C/ident/vmacache);

#ifdef [**SPLIT\_RSS\_COUNTING**](https://elixir.bootlin.com/linux/latest/C/ident/SPLIT_RSS_COUNTING)

struct [**task\_rss\_stat**](https://elixir.bootlin.com/linux/latest/C/ident/task_rss_stat) [**rss\_stat**](https://elixir.bootlin.com/linux/latest/C/ident/rss_stat);

#endif

int [**exit\_state**](https://elixir.bootlin.com/linux/latest/C/ident/exit_state);

int [**exit\_code**](https://elixir.bootlin.com/linux/latest/C/ident/exit_code);

int [**exit\_signal**](https://elixir.bootlin.com/linux/latest/C/ident/exit_signal);

*/\* The signal sent when the parent dies: \*/*

int [**pdeath\_signal**](https://elixir.bootlin.com/linux/latest/C/ident/pdeath_signal);

*/\* JOBCTL\_\*, siglock protected: \*/*

unsigned long [**jobctl**](https://elixir.bootlin.com/linux/latest/C/ident/jobctl);

*/\* Used for emulating ABI behavior of previous Linux versions: \*/*

unsigned int [**personality**](https://elixir.bootlin.com/linux/latest/C/ident/personality);

*/\* Scheduler bits, serialized by scheduler locks: \*/*

unsigned [**sched\_reset\_on\_fork**](https://elixir.bootlin.com/linux/latest/C/ident/sched_reset_on_fork):1;

unsigned [**sched\_contributes\_to\_load**](https://elixir.bootlin.com/linux/latest/C/ident/sched_contributes_to_load):1;

unsigned [**sched\_migrated**](https://elixir.bootlin.com/linux/latest/C/ident/sched_migrated):1;

#ifdef [**CONFIG\_PSI**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PSI)

unsigned [**sched\_psi\_wake\_requeue**](https://elixir.bootlin.com/linux/latest/C/ident/sched_psi_wake_requeue):1;

#endif

*/\* Force alignment to the next boundary: \*/*

unsigned :0;

*/\* Unserialized, strictly 'current' \*/*

*/\**

*\* This field must not be in the scheduler word above due to wakelist*

*\* queueing no longer being serialized by p->on\_cpu. However:*

*\**

*\* p->XXX = X; ttwu()*

*\* schedule() if (p->on\_rq && ..) // false*

*\* smp\_mb\_\_after\_spinlock(); if (smp\_load\_acquire(&p->on\_cpu) && //true*

*\* deactivate\_task() ttwu\_queue\_wakelist())*

*\* p->on\_rq = 0; p->sched\_remote\_wakeup = Y;*

*\**

*\* guarantees all stores of 'current' are visible before*

*\* ->sched\_remote\_wakeup gets used, so it can be in this word.*

*\*/*

unsigned [**sched\_remote\_wakeup**](https://elixir.bootlin.com/linux/latest/C/ident/sched_remote_wakeup):1;

*/\* Bit to tell LSMs we're in execve(): \*/*

unsigned [**in\_execve**](https://elixir.bootlin.com/linux/latest/C/ident/in_execve):1;

unsigned [**in\_iowait**](https://elixir.bootlin.com/linux/latest/C/ident/in_iowait):1;

#ifndef [**TIF\_RESTORE\_SIGMASK**](https://elixir.bootlin.com/linux/latest/C/ident/TIF_RESTORE_SIGMASK)

unsigned [**restore\_sigmask**](https://elixir.bootlin.com/linux/latest/C/ident/restore_sigmask):1;

#endif

#ifdef [**CONFIG\_MEMCG**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_MEMCG)

unsigned [**in\_user\_fault**](https://elixir.bootlin.com/linux/latest/C/ident/in_user_fault):1;

#endif

#ifdef [**CONFIG\_COMPAT\_BRK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_COMPAT_BRK)

unsigned [**brk\_randomized**](https://elixir.bootlin.com/linux/latest/C/ident/brk_randomized):1;

#endif

#ifdef [**CONFIG\_CGROUPS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_CGROUPS)

*/\* disallow userland-initiated cgroup migration \*/*

unsigned [**no\_cgroup\_migration**](https://elixir.bootlin.com/linux/latest/C/ident/no_cgroup_migration):1;

*/\* task is frozen/stopped (used by the cgroup freezer) \*/*

unsigned [**frozen**](https://elixir.bootlin.com/linux/latest/C/ident/frozen):1;

#endif

#ifdef [**CONFIG\_BLK\_CGROUP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_BLK_CGROUP)

unsigned [**use\_memdelay**](https://elixir.bootlin.com/linux/latest/C/ident/use_memdelay):1;

#endif

#ifdef [**CONFIG\_PSI**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PSI)

*/\* Stalled due to lack of memory \*/*

unsigned [**in\_memstall**](https://elixir.bootlin.com/linux/latest/C/ident/in_memstall):1;

#endif

#ifdef [**CONFIG\_PAGE\_OWNER**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PAGE_OWNER)

*/\* Used by page\_owner=on to detect recursion in page tracking. \*/*

unsigned [**in\_page\_owner**](https://elixir.bootlin.com/linux/latest/C/ident/in_page_owner):1;

#endif

#ifdef [**CONFIG\_EVENTFD**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_EVENTFD)

*/\* Recursion prevention for eventfd\_signal() \*/*

unsigned [**in\_eventfd\_signal**](https://elixir.bootlin.com/linux/latest/C/ident/in_eventfd_signal):1;

#endif

unsigned long [**atomic\_flags**](https://elixir.bootlin.com/linux/latest/C/ident/atomic_flags); */\* Flags requiring atomic access. \*/*

struct [**restart\_block**](https://elixir.bootlin.com/linux/latest/C/ident/restart_block) [**restart\_block**](https://elixir.bootlin.com/linux/latest/C/ident/restart_block);

[**pid\_t**](https://elixir.bootlin.com/linux/latest/C/ident/pid_t) [**pid**](https://elixir.bootlin.com/linux/latest/C/ident/pid);

[**pid\_t**](https://elixir.bootlin.com/linux/latest/C/ident/pid_t) [**tgid**](https://elixir.bootlin.com/linux/latest/C/ident/tgid);

#ifdef [**CONFIG\_STACKPROTECTOR**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_STACKPROTECTOR)

*/\* Canary value for the -fstack-protector GCC feature: \*/*

unsigned long [**stack\_canary**](https://elixir.bootlin.com/linux/latest/C/ident/stack_canary);

#endif

*/\**

*\* Pointers to the (original) parent process, youngest child, younger sibling,*

*\* older sibling, respectively. (p->father can be replaced with*

*\* p->real\_parent->pid)*

*\*/*

*/\* Real parent process: \*/*

struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[real\_parent](https://elixir.bootlin.com/linux/latest/C/ident/real_parent)**;

*/\* Recipient of SIGCHLD, wait4() reports: \*/*

struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \*[**parent**](https://elixir.bootlin.com/linux/latest/C/ident/parent);

*/\**

*\* Children/sibling form the list of natural children:*

*\*/*

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**children**](https://elixir.bootlin.com/linux/latest/C/ident/children);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**sibling**](https://elixir.bootlin.com/linux/latest/C/ident/sibling);

struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) \***[group\_leader](https://elixir.bootlin.com/linux/latest/C/ident/group_leader)**;

*/\**

*\* 'ptraced' is the list of tasks this task is using ptrace() on.*

*\**

*\* This includes both natural children and PTRACE\_ATTACH targets.*

*\* 'ptrace\_entry' is this task's link on the p->parent->ptraced list.*

*\*/*

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**ptraced**](https://elixir.bootlin.com/linux/latest/C/ident/ptraced);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**ptrace\_entry**](https://elixir.bootlin.com/linux/latest/C/ident/ptrace_entry);

*/\* PID/PID hash table linkage. \*/*

struct [**pid**](https://elixir.bootlin.com/linux/latest/C/ident/pid) \***[thread\_pid](https://elixir.bootlin.com/linux/latest/C/ident/thread_pid)**;

struct [**hlist\_node**](https://elixir.bootlin.com/linux/latest/C/ident/hlist_node) [**pid\_links**](https://elixir.bootlin.com/linux/latest/C/ident/pid_links)[[**PIDTYPE\_MAX**](https://elixir.bootlin.com/linux/latest/C/ident/PIDTYPE_MAX)];

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**thread\_group**](https://elixir.bootlin.com/linux/latest/C/ident/thread_group);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**thread\_node**](https://elixir.bootlin.com/linux/latest/C/ident/thread_node);

struct [**completion**](https://elixir.bootlin.com/linux/latest/C/ident/completion) \***[vfork\_done](https://elixir.bootlin.com/linux/latest/C/ident/vfork_done)**;

*/\* CLONE\_CHILD\_SETTID: \*/*

int [**\_\_user**](https://elixir.bootlin.com/linux/latest/C/ident/__user) \***[set\_child\_tid](https://elixir.bootlin.com/linux/latest/C/ident/set_child_tid)**;

*/\* CLONE\_CHILD\_CLEARTID: \*/*

int [**\_\_user**](https://elixir.bootlin.com/linux/latest/C/ident/__user) \***[clear\_child\_tid](https://elixir.bootlin.com/linux/latest/C/ident/clear_child_tid)**;

*/\* PF\_IO\_WORKER \*/*

void \***[pf\_io\_worker](https://elixir.bootlin.com/linux/latest/C/ident/pf_io_worker)**;

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**utime**](https://elixir.bootlin.com/linux/latest/C/ident/utime);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**stime**](https://elixir.bootlin.com/linux/latest/C/ident/stime);

#ifdef [**CONFIG\_ARCH\_HAS\_SCALED\_CPUTIME**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_ARCH_HAS_SCALED_CPUTIME)

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**utimescaled**](https://elixir.bootlin.com/linux/latest/C/ident/utimescaled);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**stimescaled**](https://elixir.bootlin.com/linux/latest/C/ident/stimescaled);

#endif

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**gtime**](https://elixir.bootlin.com/linux/latest/C/ident/gtime);

struct [**prev\_cputime**](https://elixir.bootlin.com/linux/latest/C/ident/prev_cputime) [**prev\_cputime**](https://elixir.bootlin.com/linux/latest/C/ident/prev_cputime);

#ifdef [**CONFIG\_VIRT\_CPU\_ACCOUNTING\_GEN**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_VIRT_CPU_ACCOUNTING_GEN)

struct [**vtime**](https://elixir.bootlin.com/linux/latest/C/ident/vtime) [**vtime**](https://elixir.bootlin.com/linux/latest/C/ident/vtime);

#endif

#ifdef [**CONFIG\_NO\_HZ\_FULL**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_NO_HZ_FULL)

[**atomic\_t**](https://elixir.bootlin.com/linux/latest/C/ident/atomic_t) [**tick\_dep\_mask**](https://elixir.bootlin.com/linux/latest/C/ident/tick_dep_mask);

#endif

*/\* Context switch counts: \*/*

unsigned long [**nvcsw**](https://elixir.bootlin.com/linux/latest/C/ident/nvcsw);

unsigned long [**nivcsw**](https://elixir.bootlin.com/linux/latest/C/ident/nivcsw);

*/\* Monotonic time in nsecs: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**start\_time**](https://elixir.bootlin.com/linux/latest/C/ident/start_time);

*/\* Boot based time in nsecs: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**start\_boottime**](https://elixir.bootlin.com/linux/latest/C/ident/start_boottime);

*/\* MM fault and swap info: this can arguably be seen as either mm-specific or thread-specific: \*/*

unsigned long [**min\_flt**](https://elixir.bootlin.com/linux/latest/C/ident/min_flt);

unsigned long [**maj\_flt**](https://elixir.bootlin.com/linux/latest/C/ident/maj_flt);

*/\* Empty if CONFIG\_POSIX\_CPUTIMERS=n \*/*

struct [**posix\_cputimers**](https://elixir.bootlin.com/linux/latest/C/ident/posix_cputimers) [**posix\_cputimers**](https://elixir.bootlin.com/linux/latest/C/ident/posix_cputimers);

#ifdef [**CONFIG\_POSIX\_CPU\_TIMERS\_TASK\_WORK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_POSIX_CPU_TIMERS_TASK_WORK)

struct [**posix\_cputimers\_work**](https://elixir.bootlin.com/linux/latest/C/ident/posix_cputimers_work) [**posix\_cputimers\_work**](https://elixir.bootlin.com/linux/latest/C/ident/posix_cputimers_work);

#endif

*/\* Process credentials: \*/*

*/\* Tracer's credentials at attach: \*/*

const struct [**cred**](https://elixir.bootlin.com/linux/latest/C/ident/cred) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[ptracer\_cred](https://elixir.bootlin.com/linux/latest/C/ident/ptracer_cred)**;

*/\* Objective and real subjective task credentials (COW): \*/*

const struct [**cred**](https://elixir.bootlin.com/linux/latest/C/ident/cred) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[real\_cred](https://elixir.bootlin.com/linux/latest/C/ident/real_cred)**;

*/\* Effective (overridable) subjective task credentials (COW): \*/*

const struct [**cred**](https://elixir.bootlin.com/linux/latest/C/ident/cred) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \*[**cred**](https://elixir.bootlin.com/linux/latest/C/ident/cred);

#ifdef [**CONFIG\_KEYS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KEYS)

*/\* Cached requested key. \*/*

struct [**key**](https://elixir.bootlin.com/linux/latest/C/ident/key) \***[cached\_requested\_key](https://elixir.bootlin.com/linux/latest/C/ident/cached_requested_key)**;

#endif

*/\**

*\* executable name, excluding path.*

*\**

*\* - normally initialized setup\_new\_exec()*

*\* - access it with [gs]et\_task\_comm()*

*\* - lock it with task\_lock()*

*\*/*

char [**comm**](https://elixir.bootlin.com/linux/latest/C/ident/comm)[[**TASK\_COMM\_LEN**](https://elixir.bootlin.com/linux/latest/C/ident/TASK_COMM_LEN)];

struct [**nameidata**](https://elixir.bootlin.com/linux/latest/C/ident/nameidata) \***[nameidata](https://elixir.bootlin.com/linux/latest/C/ident/nameidata)**;

#ifdef [**CONFIG\_SYSVIPC**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_SYSVIPC)

struct [**sysv\_sem**](https://elixir.bootlin.com/linux/latest/C/ident/sysv_sem) [**sysvsem**](https://elixir.bootlin.com/linux/latest/C/ident/sysvsem);

struct [**sysv\_shm**](https://elixir.bootlin.com/linux/latest/C/ident/sysv_shm) [**sysvshm**](https://elixir.bootlin.com/linux/latest/C/ident/sysvshm);

#endif

#ifdef [**CONFIG\_DETECT\_HUNG\_TASK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_DETECT_HUNG_TASK)

unsigned long [**last\_switch\_count**](https://elixir.bootlin.com/linux/latest/C/ident/last_switch_count);

unsigned long [**last\_switch\_time**](https://elixir.bootlin.com/linux/latest/C/ident/last_switch_time);

#endif

*/\* Filesystem information: \*/*

struct [**fs\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/fs_struct) \*[**fs**](https://elixir.bootlin.com/linux/latest/C/ident/fs);

*/\* Open file information: \*/*

struct [**files\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/files_struct) \*[**files**](https://elixir.bootlin.com/linux/latest/C/ident/files);

#ifdef [**CONFIG\_IO\_URING**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_IO_URING)

struct [**io\_uring\_task**](https://elixir.bootlin.com/linux/latest/C/ident/io_uring_task) \***[io\_uring](https://elixir.bootlin.com/linux/latest/C/ident/io_uring)**;

#endif

*/\* Namespaces: \*/*

struct [**nsproxy**](https://elixir.bootlin.com/linux/latest/C/ident/nsproxy) \***[nsproxy](https://elixir.bootlin.com/linux/latest/C/ident/nsproxy)**;

*/\* Signal handlers: \*/*

struct [**signal\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/signal_struct) \*[**signal**](https://elixir.bootlin.com/linux/latest/C/ident/signal);

struct [**sighand\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/sighand_struct) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[sighand](https://elixir.bootlin.com/linux/latest/C/ident/sighand)**;

[**sigset\_t**](https://elixir.bootlin.com/linux/latest/C/ident/sigset_t) [**blocked**](https://elixir.bootlin.com/linux/latest/C/ident/blocked);

[**sigset\_t**](https://elixir.bootlin.com/linux/latest/C/ident/sigset_t) [**real\_blocked**](https://elixir.bootlin.com/linux/latest/C/ident/real_blocked);

*/\* Restored if set\_restore\_sigmask() was used: \*/*

[**sigset\_t**](https://elixir.bootlin.com/linux/latest/C/ident/sigset_t) [**saved\_sigmask**](https://elixir.bootlin.com/linux/latest/C/ident/saved_sigmask);

struct [**sigpending**](https://elixir.bootlin.com/linux/latest/C/ident/sigpending) [**pending**](https://elixir.bootlin.com/linux/latest/C/ident/pending);

unsigned long [**sas\_ss\_sp**](https://elixir.bootlin.com/linux/latest/C/ident/sas_ss_sp);

[**size\_t**](https://elixir.bootlin.com/linux/latest/C/ident/size_t) [**sas\_ss\_size**](https://elixir.bootlin.com/linux/latest/C/ident/sas_ss_size);

unsigned int [**sas\_ss\_flags**](https://elixir.bootlin.com/linux/latest/C/ident/sas_ss_flags);

struct [**callback\_head**](https://elixir.bootlin.com/linux/latest/C/ident/callback_head) \***[task\_works](https://elixir.bootlin.com/linux/latest/C/ident/task_works)**;

#ifdef [**CONFIG\_AUDIT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_AUDIT)

#ifdef [**CONFIG\_AUDITSYSCALL**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_AUDITSYSCALL)

struct [**audit\_context**](https://elixir.bootlin.com/linux/latest/C/ident/audit_context) \***[audit\_context](https://elixir.bootlin.com/linux/latest/C/ident/audit_context)**;

#endif

[**kuid\_t**](https://elixir.bootlin.com/linux/latest/C/ident/kuid_t) [**loginuid**](https://elixir.bootlin.com/linux/latest/C/ident/loginuid);

unsigned int [**sessionid**](https://elixir.bootlin.com/linux/latest/C/ident/sessionid);

#endif

struct [**seccomp**](https://elixir.bootlin.com/linux/latest/C/ident/seccomp) [**seccomp**](https://elixir.bootlin.com/linux/latest/C/ident/seccomp);

struct [**syscall\_user\_dispatch**](https://elixir.bootlin.com/linux/latest/C/ident/syscall_user_dispatch) [**syscall\_dispatch**](https://elixir.bootlin.com/linux/latest/C/ident/syscall_dispatch);

*/\* Thread group tracking: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**parent\_exec\_id**](https://elixir.bootlin.com/linux/latest/C/ident/parent_exec_id);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**self\_exec\_id**](https://elixir.bootlin.com/linux/latest/C/ident/self_exec_id);

*/\* Protection against (de-)allocation: mm, files, fs, tty, keyrings, mems\_allowed, mempolicy: \*/*

[**spinlock\_t**](https://elixir.bootlin.com/linux/latest/C/ident/spinlock_t) [**alloc\_lock**](https://elixir.bootlin.com/linux/latest/C/ident/alloc_lock);

*/\* Protection of the PI data structures: \*/*

[**raw\_spinlock\_t**](https://elixir.bootlin.com/linux/latest/C/ident/raw_spinlock_t) [**pi\_lock**](https://elixir.bootlin.com/linux/latest/C/ident/pi_lock);

struct [**wake\_q\_node**](https://elixir.bootlin.com/linux/latest/C/ident/wake_q_node) [**wake\_q**](https://elixir.bootlin.com/linux/latest/C/ident/wake_q);

#ifdef [**CONFIG\_RT\_MUTEXES**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_RT_MUTEXES)

*/\* PI waiters blocked on a rt\_mutex held by this task: \*/*

struct [**rb\_root\_cached**](https://elixir.bootlin.com/linux/latest/C/ident/rb_root_cached) [**pi\_waiters**](https://elixir.bootlin.com/linux/latest/C/ident/pi_waiters);

*/\* Updated under owner's pi\_lock and rq lock \*/*

struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) \***[pi\_top\_task](https://elixir.bootlin.com/linux/latest/C/ident/pi_top_task)**;

*/\* Deadlock detection and priority inheritance handling: \*/*

struct [**rt\_mutex\_waiter**](https://elixir.bootlin.com/linux/latest/C/ident/rt_mutex_waiter) \***[pi\_blocked\_on](https://elixir.bootlin.com/linux/latest/C/ident/pi_blocked_on)**;

#endif

#ifdef [**CONFIG\_DEBUG\_MUTEXES**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_DEBUG_MUTEXES)

*/\* Mutex deadlock detection: \*/*

struct [**mutex\_waiter**](https://elixir.bootlin.com/linux/latest/C/ident/mutex_waiter) \***[blocked\_on](https://elixir.bootlin.com/linux/latest/C/ident/blocked_on)**;

#endif

#ifdef [**CONFIG\_DEBUG\_ATOMIC\_SLEEP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_DEBUG_ATOMIC_SLEEP)

int [**non\_block\_count**](https://elixir.bootlin.com/linux/latest/C/ident/non_block_count);

#endif

#ifdef [**CONFIG\_TRACE\_IRQFLAGS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TRACE_IRQFLAGS)

struct [**irqtrace\_events**](https://elixir.bootlin.com/linux/latest/C/ident/irqtrace_events) [**irqtrace**](https://elixir.bootlin.com/linux/latest/C/ident/irqtrace);

unsigned int [**hardirq\_threaded**](https://elixir.bootlin.com/linux/latest/C/ident/hardirq_threaded);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**hardirq\_chain\_key**](https://elixir.bootlin.com/linux/latest/C/ident/hardirq_chain_key);

int [**softirqs\_enabled**](https://elixir.bootlin.com/linux/latest/C/ident/softirqs_enabled);

int [**softirq\_context**](https://elixir.bootlin.com/linux/latest/C/ident/softirq_context);

int [**irq\_config**](https://elixir.bootlin.com/linux/latest/C/ident/irq_config);

#endif

#ifdef [**CONFIG\_PREEMPT\_RT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PREEMPT_RT)

int [**softirq\_disable\_cnt**](https://elixir.bootlin.com/linux/latest/C/ident/softirq_disable_cnt);

#endif

#ifdef [**CONFIG\_LOCKDEP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_LOCKDEP)

# define [**MAX\_LOCK\_DEPTH**](https://elixir.bootlin.com/linux/latest/C/ident/MAX_LOCK_DEPTH) 48UL

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**curr\_chain\_key**](https://elixir.bootlin.com/linux/latest/C/ident/curr_chain_key);

int [**lockdep\_depth**](https://elixir.bootlin.com/linux/latest/C/ident/lockdep_depth);

unsigned int [**lockdep\_recursion**](https://elixir.bootlin.com/linux/latest/C/ident/lockdep_recursion);

struct [**held\_lock**](https://elixir.bootlin.com/linux/latest/C/ident/held_lock) [**held\_locks**](https://elixir.bootlin.com/linux/latest/C/ident/held_locks)[[**MAX\_LOCK\_DEPTH**](https://elixir.bootlin.com/linux/latest/C/ident/MAX_LOCK_DEPTH)];

#endif

#if [**defined**](https://elixir.bootlin.com/linux/latest/C/ident/defined)([**CONFIG\_UBSAN**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_UBSAN)) && ![**defined**](https://elixir.bootlin.com/linux/latest/C/ident/defined)([**CONFIG\_UBSAN\_TRAP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_UBSAN_TRAP))

unsigned int [**in\_ubsan**](https://elixir.bootlin.com/linux/latest/C/ident/in_ubsan);

#endif

*/\* Journalling filesystem info: \*/*

void \***[journal\_info](https://elixir.bootlin.com/linux/latest/C/ident/journal_info)**;

*/\* Stacked block device info: \*/*

struct [**bio\_list**](https://elixir.bootlin.com/linux/latest/C/ident/bio_list) \***[bio\_list](https://elixir.bootlin.com/linux/latest/C/ident/bio_list)**;

*/\* Stack plugging: \*/*

struct [**blk\_plug**](https://elixir.bootlin.com/linux/latest/C/ident/blk_plug) \*[**plug**](https://elixir.bootlin.com/linux/latest/C/ident/plug);

*/\* VM state: \*/*

struct [**reclaim\_state**](https://elixir.bootlin.com/linux/latest/C/ident/reclaim_state) \***[reclaim\_state](https://elixir.bootlin.com/linux/latest/C/ident/reclaim_state)**;

struct [**backing\_dev\_info**](https://elixir.bootlin.com/linux/latest/C/ident/backing_dev_info) \***[backing\_dev\_info](https://elixir.bootlin.com/linux/latest/C/ident/backing_dev_info)**;

struct [**io\_context**](https://elixir.bootlin.com/linux/latest/C/ident/io_context) \***[io\_context](https://elixir.bootlin.com/linux/latest/C/ident/io_context)**;

#ifdef [**CONFIG\_COMPACTION**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_COMPACTION)

struct [**capture\_control**](https://elixir.bootlin.com/linux/latest/C/ident/capture_control) \***[capture\_control](https://elixir.bootlin.com/linux/latest/C/ident/capture_control)**;

#endif

*/\* Ptrace state: \*/*

unsigned long [**ptrace\_message**](https://elixir.bootlin.com/linux/latest/C/ident/ptrace_message);

[**kernel\_siginfo\_t**](https://elixir.bootlin.com/linux/latest/C/ident/kernel_siginfo_t) \***[last\_siginfo](https://elixir.bootlin.com/linux/latest/C/ident/last_siginfo)**;

struct [**task\_io\_accounting**](https://elixir.bootlin.com/linux/latest/C/ident/task_io_accounting) [**ioac**](https://elixir.bootlin.com/linux/latest/C/ident/ioac);

#ifdef [**CONFIG\_PSI**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PSI)

*/\* Pressure stall state \*/*

unsigned int [**psi\_flags**](https://elixir.bootlin.com/linux/latest/C/ident/psi_flags);

#endif

#ifdef [**CONFIG\_TASK\_XACCT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TASK_XACCT)

*/\* Accumulated RSS usage: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**acct\_rss\_mem1**](https://elixir.bootlin.com/linux/latest/C/ident/acct_rss_mem1);

*/\* Accumulated virtual memory usage: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**acct\_vm\_mem1**](https://elixir.bootlin.com/linux/latest/C/ident/acct_vm_mem1);

*/\* stime + utime since last update: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**acct\_timexpd**](https://elixir.bootlin.com/linux/latest/C/ident/acct_timexpd);

#endif

#ifdef [**CONFIG\_CPUSETS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_CPUSETS)

*/\* Protected by ->alloc\_lock: \*/*

[**nodemask\_t**](https://elixir.bootlin.com/linux/latest/C/ident/nodemask_t) [**mems\_allowed**](https://elixir.bootlin.com/linux/latest/C/ident/mems_allowed);

*/\* Sequence number to catch updates: \*/*

seqcount\_spinlock\_t [**mems\_allowed\_seq**](https://elixir.bootlin.com/linux/latest/C/ident/mems_allowed_seq);

int [**cpuset\_mem\_spread\_rotor**](https://elixir.bootlin.com/linux/latest/C/ident/cpuset_mem_spread_rotor);

int [**cpuset\_slab\_spread\_rotor**](https://elixir.bootlin.com/linux/latest/C/ident/cpuset_slab_spread_rotor);

#endif

#ifdef [**CONFIG\_CGROUPS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_CGROUPS)

*/\* Control Group info protected by css\_set\_lock: \*/*

struct [**css\_set**](https://elixir.bootlin.com/linux/latest/C/ident/css_set) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[cgroups](https://elixir.bootlin.com/linux/latest/C/ident/cgroups)**;

*/\* cg\_list protected by css\_set\_lock and tsk->alloc\_lock: \*/*

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**cg\_list**](https://elixir.bootlin.com/linux/latest/C/ident/cg_list);

#endif

#ifdef [**CONFIG\_X86\_CPU\_RESCTRL**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_X86_CPU_RESCTRL)

[**u32**](https://elixir.bootlin.com/linux/latest/C/ident/u32) [**closid**](https://elixir.bootlin.com/linux/latest/C/ident/closid);

[**u32**](https://elixir.bootlin.com/linux/latest/C/ident/u32) [**rmid**](https://elixir.bootlin.com/linux/latest/C/ident/rmid);

#endif

#ifdef [**CONFIG\_FUTEX**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_FUTEX)

struct [**robust\_list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/robust_list_head) [**\_\_user**](https://elixir.bootlin.com/linux/latest/C/ident/__user) \***[robust\_list](https://elixir.bootlin.com/linux/latest/C/ident/robust_list)**;

#ifdef [**CONFIG\_COMPAT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_COMPAT)

struct [**compat\_robust\_list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/compat_robust_list_head) [**\_\_user**](https://elixir.bootlin.com/linux/latest/C/ident/__user) \***[compat\_robust\_list](https://elixir.bootlin.com/linux/latest/C/ident/compat_robust_list)**;

#endif

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**pi\_state\_list**](https://elixir.bootlin.com/linux/latest/C/ident/pi_state_list);

struct [**futex\_pi\_state**](https://elixir.bootlin.com/linux/latest/C/ident/futex_pi_state) \***[pi\_state\_cache](https://elixir.bootlin.com/linux/latest/C/ident/pi_state_cache)**;

struct [**mutex**](https://elixir.bootlin.com/linux/latest/C/ident/mutex) [**futex\_exit\_mutex**](https://elixir.bootlin.com/linux/latest/C/ident/futex_exit_mutex);

unsigned int [**futex\_state**](https://elixir.bootlin.com/linux/latest/C/ident/futex_state);

#endif

#ifdef [**CONFIG\_PERF\_EVENTS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PERF_EVENTS)

struct [**perf\_event\_context**](https://elixir.bootlin.com/linux/latest/C/ident/perf_event_context) \***[perf\_event\_ctxp](https://elixir.bootlin.com/linux/latest/C/ident/perf_event_ctxp)**[**[perf\_nr\_task\_contexts](https://elixir.bootlin.com/linux/latest/C/ident/perf_nr_task_contexts)**];

struct [**mutex**](https://elixir.bootlin.com/linux/latest/C/ident/mutex) [**perf\_event\_mutex**](https://elixir.bootlin.com/linux/latest/C/ident/perf_event_mutex);

struct [**list\_head**](https://elixir.bootlin.com/linux/latest/C/ident/list_head) [**perf\_event\_list**](https://elixir.bootlin.com/linux/latest/C/ident/perf_event_list);

#endif

#ifdef [**CONFIG\_DEBUG\_PREEMPT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_DEBUG_PREEMPT)

unsigned long [**preempt\_disable\_ip**](https://elixir.bootlin.com/linux/latest/C/ident/preempt_disable_ip);

#endif

#ifdef [**CONFIG\_NUMA**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_NUMA)

*/\* Protected by alloc\_lock: \*/*

struct [**mempolicy**](https://elixir.bootlin.com/linux/latest/C/ident/mempolicy) \***[mempolicy](https://elixir.bootlin.com/linux/latest/C/ident/mempolicy)**;

[**short**](https://elixir.bootlin.com/linux/latest/C/ident/short) [**il\_prev**](https://elixir.bootlin.com/linux/latest/C/ident/il_prev);

[**short**](https://elixir.bootlin.com/linux/latest/C/ident/short) [**pref\_node\_fork**](https://elixir.bootlin.com/linux/latest/C/ident/pref_node_fork);

#endif

#ifdef [**CONFIG\_NUMA\_BALANCING**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_NUMA_BALANCING)

int [**numa\_scan\_seq**](https://elixir.bootlin.com/linux/latest/C/ident/numa_scan_seq);

unsigned int [**numa\_scan\_period**](https://elixir.bootlin.com/linux/latest/C/ident/numa_scan_period);

unsigned int [**numa\_scan\_period\_max**](https://elixir.bootlin.com/linux/latest/C/ident/numa_scan_period_max);

int [**numa\_preferred\_nid**](https://elixir.bootlin.com/linux/latest/C/ident/numa_preferred_nid);

unsigned long [**numa\_migrate\_retry**](https://elixir.bootlin.com/linux/latest/C/ident/numa_migrate_retry);

*/\* Migration stamp: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**node\_stamp**](https://elixir.bootlin.com/linux/latest/C/ident/node_stamp);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**last\_task\_numa\_placement**](https://elixir.bootlin.com/linux/latest/C/ident/last_task_numa_placement);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**last\_sum\_exec\_runtime**](https://elixir.bootlin.com/linux/latest/C/ident/last_sum_exec_runtime);

struct [**callback\_head**](https://elixir.bootlin.com/linux/latest/C/ident/callback_head) [**numa\_work**](https://elixir.bootlin.com/linux/latest/C/ident/numa_work);

*/\**

*\* This pointer is only modified for current in syscall and*

*\* pagefault context (and for tasks being destroyed), so it can be read*

*\* from any of the following contexts:*

*\* - RCU read-side critical section*

*\* - current->numa\_group from everywhere*

*\* - task's runqueue locked, task not running*

*\*/*

struct [**numa\_group**](https://elixir.bootlin.com/linux/latest/C/ident/numa_group) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[numa\_group](https://elixir.bootlin.com/linux/latest/C/ident/numa_group)**;

*/\**

*\* numa\_faults is an array split into four regions:*

*\* faults\_memory, faults\_cpu, faults\_memory\_buffer, faults\_cpu\_buffer*

*\* in this precise order.*

*\**

*\* faults\_memory: Exponential decaying average of faults on a per-node*

*\* basis. Scheduling placement decisions are made based on these*

*\* counts. The values remain static for the duration of a PTE scan.*

*\* faults\_cpu: Track the nodes the process was running on when a NUMA*

*\* hinting fault was incurred.*

*\* faults\_memory\_buffer and faults\_cpu\_buffer: Record faults per node*

*\* during the current scan window. When the scan completes, the counts*

*\* in faults\_memory and faults\_cpu decay and these values are copied.*

*\*/*

unsigned long \***[numa\_faults](https://elixir.bootlin.com/linux/latest/C/ident/numa_faults)**;

unsigned long [**total\_numa\_faults**](https://elixir.bootlin.com/linux/latest/C/ident/total_numa_faults);

*/\**

*\* numa\_faults\_locality tracks if faults recorded during the last*

*\* scan window were remote/local or failed to migrate. The task scan*

*\* period is adapted based on the locality of the faults with different*

*\* weights depending on whether they were shared or private faults*

*\*/*

unsigned long [**numa\_faults\_locality**](https://elixir.bootlin.com/linux/latest/C/ident/numa_faults_locality)[3];

unsigned long [**numa\_pages\_migrated**](https://elixir.bootlin.com/linux/latest/C/ident/numa_pages_migrated);

#endif */\* CONFIG\_NUMA\_BALANCING \*/*

#ifdef [**CONFIG\_RSEQ**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_RSEQ)

struct [**rseq**](https://elixir.bootlin.com/linux/latest/C/ident/rseq) [**\_\_user**](https://elixir.bootlin.com/linux/latest/C/ident/__user) \***[rseq](https://elixir.bootlin.com/linux/latest/C/ident/rseq)**;

[**u32**](https://elixir.bootlin.com/linux/latest/C/ident/u32) [**rseq\_sig**](https://elixir.bootlin.com/linux/latest/C/ident/rseq_sig);

*/\**

*\* RmW on rseq\_event\_mask must be performed atomically*

*\* with respect to preemption.*

*\*/*

unsigned long [**rseq\_event\_mask**](https://elixir.bootlin.com/linux/latest/C/ident/rseq_event_mask);

#endif

struct [**tlbflush\_unmap\_batch**](https://elixir.bootlin.com/linux/latest/C/ident/tlbflush_unmap_batch) [**tlb\_ubc**](https://elixir.bootlin.com/linux/latest/C/ident/tlb_ubc);

union {

[**refcount\_t**](https://elixir.bootlin.com/linux/latest/C/ident/refcount_t) [**rcu\_users**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_users);

struct [**rcu\_head**](https://elixir.bootlin.com/linux/latest/C/ident/rcu_head) [**rcu**](https://elixir.bootlin.com/linux/latest/C/ident/rcu);

};

*/\* Cache last used pipe for splice(): \*/*

struct [**pipe\_inode\_info**](https://elixir.bootlin.com/linux/latest/C/ident/pipe_inode_info) \***[splice\_pipe](https://elixir.bootlin.com/linux/latest/C/ident/splice_pipe)**;

struct [**page\_frag**](https://elixir.bootlin.com/linux/latest/C/ident/page_frag) [**task\_frag**](https://elixir.bootlin.com/linux/latest/C/ident/task_frag);

#ifdef [**CONFIG\_TASK\_DELAY\_ACCT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TASK_DELAY_ACCT)

struct [**task\_delay\_info**](https://elixir.bootlin.com/linux/latest/C/ident/task_delay_info) \*[**delays**](https://elixir.bootlin.com/linux/latest/C/ident/delays);

#endif

#ifdef [**CONFIG\_FAULT\_INJECTION**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_FAULT_INJECTION)

int [**make\_it\_fail**](https://elixir.bootlin.com/linux/latest/C/ident/make_it_fail);

unsigned int [**fail\_nth**](https://elixir.bootlin.com/linux/latest/C/ident/fail_nth);

#endif

*/\**

*\* When (nr\_dirtied >= nr\_dirtied\_pause), it's time to call*

*\* balance\_dirty\_pages() for a dirty throttling pause:*

*\*/*

int [**nr\_dirtied**](https://elixir.bootlin.com/linux/latest/C/ident/nr_dirtied);

int [**nr\_dirtied\_pause**](https://elixir.bootlin.com/linux/latest/C/ident/nr_dirtied_pause);

*/\* Start of a write-and-pause period: \*/*

unsigned long [**dirty\_paused\_when**](https://elixir.bootlin.com/linux/latest/C/ident/dirty_paused_when);

#ifdef [**CONFIG\_LATENCYTOP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_LATENCYTOP)

int [**latency\_record\_count**](https://elixir.bootlin.com/linux/latest/C/ident/latency_record_count);

struct [**latency\_record**](https://elixir.bootlin.com/linux/latest/C/ident/latency_record) [**latency\_record**](https://elixir.bootlin.com/linux/latest/C/ident/latency_record)[[**LT\_SAVECOUNT**](https://elixir.bootlin.com/linux/latest/C/ident/LT_SAVECOUNT)];

#endif

*/\**

*\* Time slack values; these are used to round up poll() and*

*\* select() etc timeout values. These are in nanoseconds.*

*\*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**timer\_slack\_ns**](https://elixir.bootlin.com/linux/latest/C/ident/timer_slack_ns);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**default\_timer\_slack\_ns**](https://elixir.bootlin.com/linux/latest/C/ident/default_timer_slack_ns);

#if [**defined**](https://elixir.bootlin.com/linux/latest/C/ident/defined)([**CONFIG\_KASAN\_GENERIC**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KASAN_GENERIC)) || [**defined**](https://elixir.bootlin.com/linux/latest/C/ident/defined)([**CONFIG\_KASAN\_SW\_TAGS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KASAN_SW_TAGS))

unsigned int [**kasan\_depth**](https://elixir.bootlin.com/linux/latest/C/ident/kasan_depth);

#endif

#ifdef [**CONFIG\_KCSAN**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KCSAN)

struct [**kcsan\_ctx**](https://elixir.bootlin.com/linux/latest/C/ident/kcsan_ctx) [**kcsan\_ctx**](https://elixir.bootlin.com/linux/latest/C/ident/kcsan_ctx);

#ifdef [**CONFIG\_TRACE\_IRQFLAGS**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TRACE_IRQFLAGS)

struct [**irqtrace\_events**](https://elixir.bootlin.com/linux/latest/C/ident/irqtrace_events) [**kcsan\_save\_irqtrace**](https://elixir.bootlin.com/linux/latest/C/ident/kcsan_save_irqtrace);

#endif

#endif

#if [**IS\_ENABLED**](https://elixir.bootlin.com/linux/latest/C/ident/IS_ENABLED)([**CONFIG\_KUNIT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KUNIT))

struct [**kunit**](https://elixir.bootlin.com/linux/latest/C/ident/kunit) \***[kunit\_test](https://elixir.bootlin.com/linux/latest/C/ident/kunit_test)**;

#endif

#ifdef [**CONFIG\_FUNCTION\_GRAPH\_TRACER**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_FUNCTION_GRAPH_TRACER)

*/\* Index of current stored address in ret\_stack: \*/*

int [**curr\_ret\_stack**](https://elixir.bootlin.com/linux/latest/C/ident/curr_ret_stack);

int [**curr\_ret\_depth**](https://elixir.bootlin.com/linux/latest/C/ident/curr_ret_depth);

*/\* Stack of return addresses for return function tracing: \*/*

struct [**ftrace\_ret\_stack**](https://elixir.bootlin.com/linux/latest/C/ident/ftrace_ret_stack) \***[ret\_stack](https://elixir.bootlin.com/linux/latest/C/ident/ret_stack)**;

*/\* Timestamp for last schedule: \*/*

unsigned long long [**ftrace\_timestamp**](https://elixir.bootlin.com/linux/latest/C/ident/ftrace_timestamp);

*/\**

*\* Number of functions that haven't been traced*

*\* because of depth overrun:*

*\*/*

[**atomic\_t**](https://elixir.bootlin.com/linux/latest/C/ident/atomic_t) [**trace\_overrun**](https://elixir.bootlin.com/linux/latest/C/ident/trace_overrun);

*/\* Pause tracing: \*/*

[**atomic\_t**](https://elixir.bootlin.com/linux/latest/C/ident/atomic_t) [**tracing\_graph\_pause**](https://elixir.bootlin.com/linux/latest/C/ident/tracing_graph_pause);

#endif

#ifdef [**CONFIG\_TRACING**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_TRACING)

*/\* State flags for use by tracers: \*/*

unsigned long [**trace**](https://elixir.bootlin.com/linux/latest/C/ident/trace);

*/\* Bitmask and counter of trace recursion: \*/*

unsigned long [**trace\_recursion**](https://elixir.bootlin.com/linux/latest/C/ident/trace_recursion);

#endif */\* CONFIG\_TRACING \*/*

#ifdef [**CONFIG\_KCOV**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KCOV)

*/\* See kernel/kcov.c for more details. \*/*

*/\* Coverage collection mode enabled for this task (0 if disabled): \*/*

unsigned int [**kcov\_mode**](https://elixir.bootlin.com/linux/latest/C/ident/kcov_mode);

*/\* Size of the kcov\_area: \*/*

unsigned int [**kcov\_size**](https://elixir.bootlin.com/linux/latest/C/ident/kcov_size);

*/\* Buffer for coverage collection: \*/*

void \***[kcov\_area](https://elixir.bootlin.com/linux/latest/C/ident/kcov_area)**;

*/\* KCOV descriptor wired with this task or NULL: \*/*

struct [**kcov**](https://elixir.bootlin.com/linux/latest/C/ident/kcov) \***[kcov](https://elixir.bootlin.com/linux/latest/C/ident/kcov)**;

*/\* KCOV common handle for remote coverage collection: \*/*

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**kcov\_handle**](https://elixir.bootlin.com/linux/latest/C/ident/kcov_handle);

*/\* KCOV sequence number: \*/*

int [**kcov\_sequence**](https://elixir.bootlin.com/linux/latest/C/ident/kcov_sequence);

*/\* Collect coverage from softirq context: \*/*

unsigned int [**kcov\_softirq**](https://elixir.bootlin.com/linux/latest/C/ident/kcov_softirq);

#endif

#ifdef [**CONFIG\_MEMCG**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_MEMCG)

struct [**mem\_cgroup**](https://elixir.bootlin.com/linux/latest/C/ident/mem_cgroup) \***[memcg\_in\_oom](https://elixir.bootlin.com/linux/latest/C/ident/memcg_in_oom)**;

[**gfp\_t**](https://elixir.bootlin.com/linux/latest/C/ident/gfp_t) [**memcg\_oom\_gfp\_mask**](https://elixir.bootlin.com/linux/latest/C/ident/memcg_oom_gfp_mask);

int [**memcg\_oom\_order**](https://elixir.bootlin.com/linux/latest/C/ident/memcg_oom_order);

*/\* Number of pages to reclaim on returning to userland: \*/*

unsigned int [**memcg\_nr\_pages\_over\_high**](https://elixir.bootlin.com/linux/latest/C/ident/memcg_nr_pages_over_high);

*/\* Used by memcontrol for targeted memcg charge: \*/*

struct [**mem\_cgroup**](https://elixir.bootlin.com/linux/latest/C/ident/mem_cgroup) \***[active\_memcg](https://elixir.bootlin.com/linux/latest/C/ident/active_memcg)**;

#endif

#ifdef [**CONFIG\_BLK\_CGROUP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_BLK_CGROUP)

struct [**request\_queue**](https://elixir.bootlin.com/linux/latest/C/ident/request_queue) \***[throttle\_queue](https://elixir.bootlin.com/linux/latest/C/ident/throttle_queue)**;

#endif

#ifdef [**CONFIG\_UPROBES**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_UPROBES)

struct [**uprobe\_task**](https://elixir.bootlin.com/linux/latest/C/ident/uprobe_task) \***[utask](https://elixir.bootlin.com/linux/latest/C/ident/utask)**;

#endif

#if [**defined**](https://elixir.bootlin.com/linux/latest/C/ident/defined)([**CONFIG\_BCACHE**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_BCACHE)) || [**defined**](https://elixir.bootlin.com/linux/latest/C/ident/defined)(CONFIG\_BCACHE\_MODULE)

unsigned int [**sequential\_io**](https://elixir.bootlin.com/linux/latest/C/ident/sequential_io);

unsigned int [**sequential\_io\_avg**](https://elixir.bootlin.com/linux/latest/C/ident/sequential_io_avg);

#endif

struct [**kmap\_ctrl**](https://elixir.bootlin.com/linux/latest/C/ident/kmap_ctrl) [**kmap\_ctrl**](https://elixir.bootlin.com/linux/latest/C/ident/kmap_ctrl);

#ifdef [**CONFIG\_DEBUG\_ATOMIC\_SLEEP**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_DEBUG_ATOMIC_SLEEP)

unsigned long [**task\_state\_change**](https://elixir.bootlin.com/linux/latest/C/ident/task_state_change);

# ifdef [**CONFIG\_PREEMPT\_RT**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_PREEMPT_RT)

unsigned long [**saved\_state\_change**](https://elixir.bootlin.com/linux/latest/C/ident/saved_state_change);

# endif

#endif

int [**pagefault\_disabled**](https://elixir.bootlin.com/linux/latest/C/ident/pagefault_disabled);

#ifdef [**CONFIG\_MMU**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_MMU)

struct [**task\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/task_struct) \***[oom\_reaper\_list](https://elixir.bootlin.com/linux/latest/C/ident/oom_reaper_list)**;

#endif

#ifdef [**CONFIG\_VMAP\_STACK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_VMAP_STACK)

struct [**vm\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/vm_struct) \***[stack\_vm\_area](https://elixir.bootlin.com/linux/latest/C/ident/stack_vm_area)**;

#endif

#ifdef [**CONFIG\_THREAD\_INFO\_IN\_TASK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_THREAD_INFO_IN_TASK)

*/\* A live task holds one reference: \*/*

[**refcount\_t**](https://elixir.bootlin.com/linux/latest/C/ident/refcount_t) [**stack\_refcount**](https://elixir.bootlin.com/linux/latest/C/ident/stack_refcount);

#endif

#ifdef [**CONFIG\_LIVEPATCH**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_LIVEPATCH)

int [**patch\_state**](https://elixir.bootlin.com/linux/latest/C/ident/patch_state);

#endif

#ifdef [**CONFIG\_SECURITY**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_SECURITY)

*/\* Used by LSM modules for access restriction: \*/*

void \*[**security**](https://elixir.bootlin.com/linux/latest/C/ident/security);

#endif

#ifdef [**CONFIG\_BPF\_SYSCALL**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_BPF_SYSCALL)

*/\* Used by BPF task local storage \*/*

struct [**bpf\_local\_storage**](https://elixir.bootlin.com/linux/latest/C/ident/bpf_local_storage) [**\_\_rcu**](https://elixir.bootlin.com/linux/latest/C/ident/__rcu) \***[bpf\_storage](https://elixir.bootlin.com/linux/latest/C/ident/bpf_storage)**;

*/\* Used for BPF run context \*/*

struct [**bpf\_run\_ctx**](https://elixir.bootlin.com/linux/latest/C/ident/bpf_run_ctx) \***[bpf\_ctx](https://elixir.bootlin.com/linux/latest/C/ident/bpf_ctx)**;

#endif

#ifdef [**CONFIG\_GCC\_PLUGIN\_STACKLEAK**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_GCC_PLUGIN_STACKLEAK)

unsigned long [**lowest\_stack**](https://elixir.bootlin.com/linux/latest/C/ident/lowest_stack);

unsigned long [**prev\_lowest\_stack**](https://elixir.bootlin.com/linux/latest/C/ident/prev_lowest_stack);

#endif

#ifdef [**CONFIG\_X86\_MCE**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_X86_MCE)

void [**\_\_user**](https://elixir.bootlin.com/linux/latest/C/ident/__user) \***[mce\_vaddr](https://elixir.bootlin.com/linux/latest/C/ident/mce_vaddr)**;

[**\_\_u64**](https://elixir.bootlin.com/linux/latest/C/ident/__u64) [**mce\_kflags**](https://elixir.bootlin.com/linux/latest/C/ident/mce_kflags);

[**u64**](https://elixir.bootlin.com/linux/latest/C/ident/u64) [**mce\_addr**](https://elixir.bootlin.com/linux/latest/C/ident/mce_addr);

[**\_\_u64**](https://elixir.bootlin.com/linux/latest/C/ident/__u64) [**mce\_ripv**](https://elixir.bootlin.com/linux/latest/C/ident/mce_ripv) : 1,

[**mce\_whole\_page**](https://elixir.bootlin.com/linux/latest/C/ident/mce_whole_page) : 1,

[**\_\_mce\_reserved**](https://elixir.bootlin.com/linux/latest/C/ident/__mce_reserved) : 62;

struct [**callback\_head**](https://elixir.bootlin.com/linux/latest/C/ident/callback_head) [**mce\_kill\_me**](https://elixir.bootlin.com/linux/latest/C/ident/mce_kill_me);

int [**mce\_count**](https://elixir.bootlin.com/linux/latest/C/ident/mce_count);

#endif

#ifdef [**CONFIG\_KRETPROBES**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_KRETPROBES)

struct [**llist\_head**](https://elixir.bootlin.com/linux/latest/C/ident/llist_head) [**kretprobe\_instances**](https://elixir.bootlin.com/linux/latest/C/ident/kretprobe_instances);

#endif

#ifdef [**CONFIG\_ARCH\_HAS\_PARANOID\_L1D\_FLUSH**](https://elixir.bootlin.com/linux/latest/K/ident/CONFIG_ARCH_HAS_PARANOID_L1D_FLUSH)

*/\**

*\* If L1D flush is supported on mm context switch*

*\* then we use this callback head to queue kill work*

*\* to kill tasks that are not running on SMT disabled*

*\* cores*

*\*/*

struct [**callback\_head**](https://elixir.bootlin.com/linux/latest/C/ident/callback_head) [**l1d\_flush\_kill**](https://elixir.bootlin.com/linux/latest/C/ident/l1d_flush_kill);

#endif

*/\**

*\* New fields for task\_struct should be added above here, so that*

*\* they are included in the randomized portion of task\_struct.*

*\*/*

[**randomized\_struct\_fields\_end**](https://elixir.bootlin.com/linux/latest/C/ident/randomized_struct_fields_end)

*/\* CPU-specific state of this task: \*/*

struct [**thread\_struct**](https://elixir.bootlin.com/linux/latest/C/ident/thread_struct) [**thread**](https://elixir.bootlin.com/linux/latest/C/ident/thread);

*/\**

*\* WARNING: on x86, 'thread\_struct' contains a variable-sized*

*\* structure. It \*MUST\* be at the end of 'task\_struct'.*

*\**

*\* Do not put anything below here!*

*\*/*

};