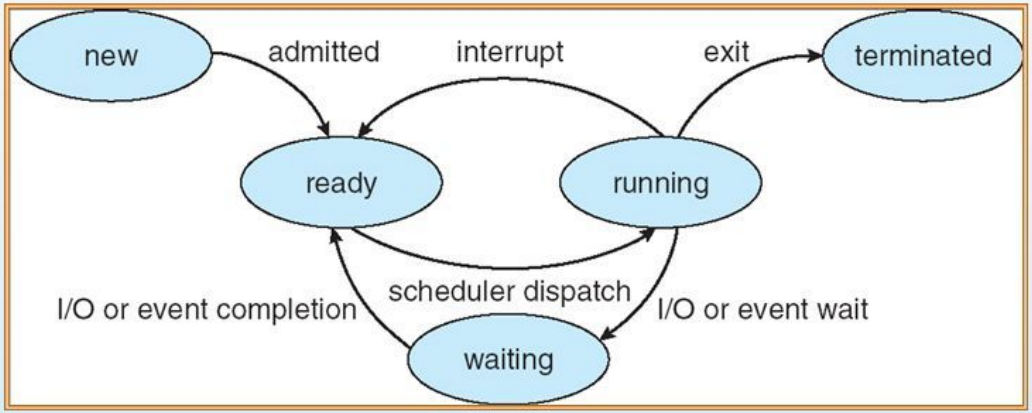
LinuxInternals.org

Process wakeup Path

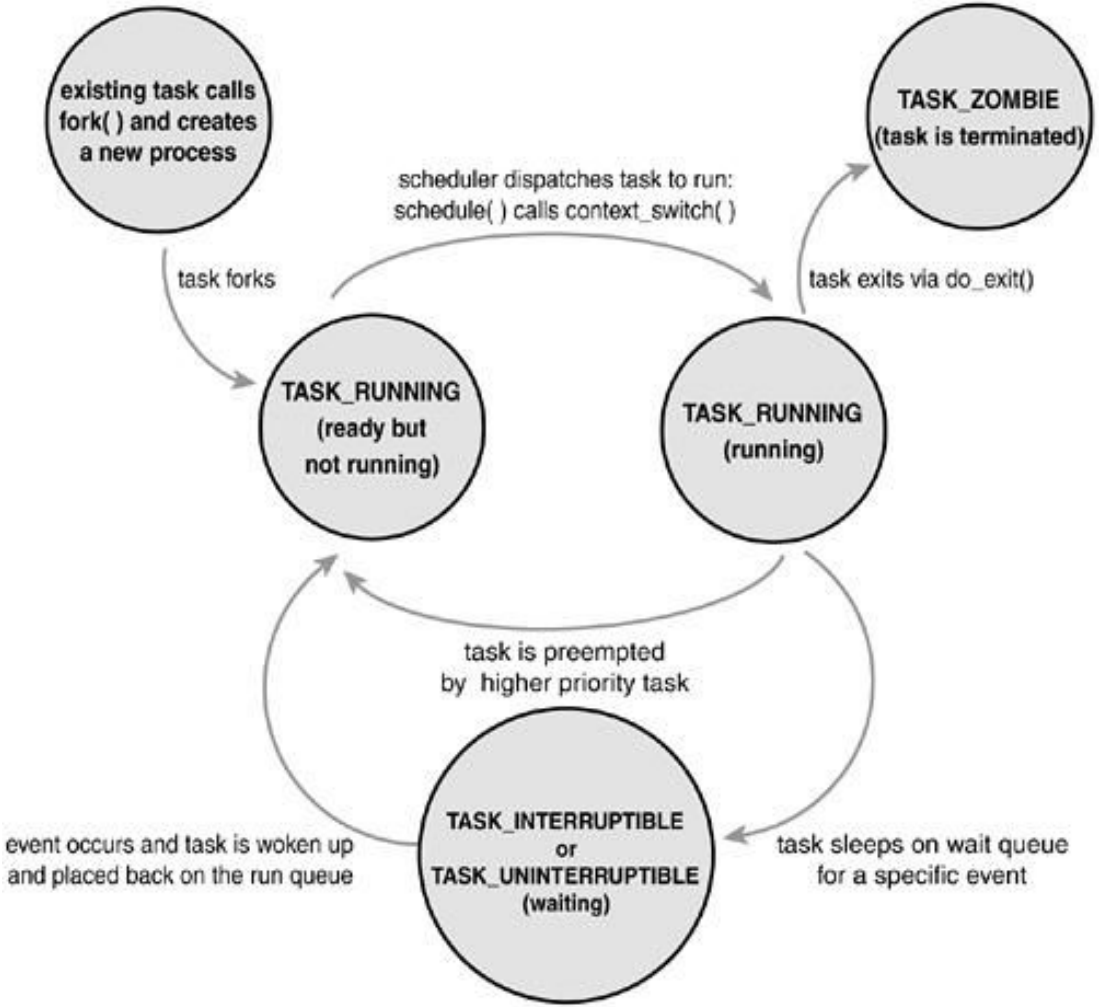
By Joel Fernandes (joel@joelfernandes.org)

Diagram of Process State



KLDP wiki
https://wiki.kldp.org/wiki.php/ProcessManagement

- Linux process states


existing task calls fork( ) and creates a new process
TASK_ZOMBIE (task is terminated)
scheduler dispatches task to run: schedule( ) calls context_switch( )
task forks
task exits via do_exit()
TASK_RUNNING (ready but not running)
TASK_RUNNING (running)
task is preempted by higher priority task
TASK_INTERRUPTIBLE or TASK_UNINTERRUPTIBLE (waiting)
event occurs and task is woken up and placed back on the run queue
task sleeps on wait queue for a specific event

When/what path does a task goto sleep?

Steps to sleep

1. Add to wait queue
2. Change process state from RUNNING -> SLEEP
3. Call scheduler
4. Change process state from SLEEP -> RUNNING
5. Remove from wait queue

Steps 2 to 4 are repeated in a loop to handle signal interruptions. Many system calls return an error if a signal is received while its running.

Example: Using nanosleep call

```
 trace-cmd record -F -e "sched:sched_switch"
                     -R "stacktrace" sleep 1
```

```
          sleep-1044  [003]  1552.918189: kernel_stack:
<stack trace>
=> schedule (ffffffff8191d5c6)
=> do_nanosleep (ffffffff81920540)
=> hrtimer_nanosleep (ffffffff810bcae6)
=> SyS_nanosleep (ffffffff810bcbec)
=> entry_SYSCALL_64_fastpath (ffffffff819214e0)
```

Code for timer (sleep call):

http://lxr.free-electrons.com/source/kernel/time/hrtimer.c?v=4.8#L1471

- t->task = NULL means wakeup path completed timer
- t->task != NULL means signal interrupted it
- So if t->task != NULL and signal is still pending, then abort the sleep and return remaining time to userspace

Nanosleep wakeup path

```
  trace-cmd record -F -e "sched:sched_wakeup" -R "stacktrace" sleep 1

          <idle>-0     [002]  2099.399289: sched_wakeup:         comm=sleep pid=1094 prio=120
target_cpu=002
          <idle>-0     [002]  2099.399311: kernel_stack:         <stack trace>
=> ttwu_do_activate (ffffffff8108387f)
=> try_to_wake_up (ffffffff8108439d)
=> wake_up_process (ffffffff810844a5)
=> hrtimer_wakeup (ffffffff810bbd92)
=> __hrtimer_run_queues (ffffffff810bc3ef)
=> hrtimer_interrupt (ffffffff810bc888)
=> local_apic_timer_interrupt (ffffffff8103f565)
=> smp_apic_timer_interrupt (ffffffff81923a1d)
=> apic_timer_interrupt (ffffffff81922d2f)
=> arch_cpu_idle (ffffffff8102654f)
=> default_idle_call (ffffffff81920fe5)
=> cpu_startup_entry (ffffffff81098c3c)
```

nanosleep Wakeup Path

http://lxr.free-electrons.com/source/kernel/time/hrtimer.c?v=4.8#L1451

Example: reading from console using cat

```
trace-cmd record -F -e "sched:sched_switch"
                    -R "stacktrace" cat
```

```
            cat-1085  [002]  1885.918221: kernel_stack:
<stack trace>
=> schedule (ffffffff8191d5c6)
=> schedule_timeout (ffffffff81920284)
=> wait_woken (ffffffff810982e5)
=> n_tty_read (ffffffff813ec0db)
=> tty_read (ffffffff813e6382)
=> __vfs_read (ffffffff811929c8)
=> vfs_read (ffffffff8119316c)
=> SyS_read (ffffffff81194556)
```

Code for TTY driver (read call):

TTY driver code calling wait_woken
http://lxr.free-electrons.com/source/drivers/tty/n_tty.c?v=4.8#L2109

wait_woken:
http://lxr.free-electrons.com/source/kernel/sched/wait.c?v=4.8#L326

Example: Waiting on I/O to a page read from flash

```
trace-cmd record -e “sched:switch”
                 -F -R “stacktrace”
                 dd if=/dev/sda35 of=/dev/null
```

Dd goes to sleep waiting for I/O to complete on a page..
__lock_page_killable causes sleep

```
<stack trace>
=> __schedule (ffffffc000f078b0)
=> schedule (ffffffc000f079b4)
=> io_schedule (ffffffc000f07a2c)
=> bit_wait_io (ffffffc000f08398)
=> __wait_on_bit_lock (ffffffc000f081e8)
=> __lock_page_killable (ffffffc000177308)
=> generic_file_read_iter (ffffffc0001791cc)
=> blkdev_read_iter (ffffffc0001fd064)
=> new_sync_read (ffffffc0001c5948)
=> vfs_read (ffffffc0001c6164)
=> SyS_read (ffffffc0001c696c)
```

Code:

http://lxr.free-electrons.com/source/mm/filemap.c#L1840