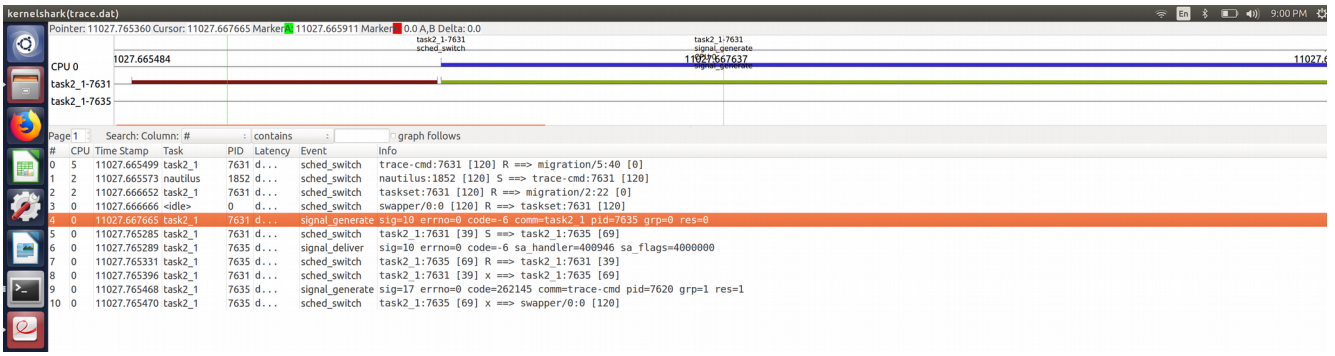
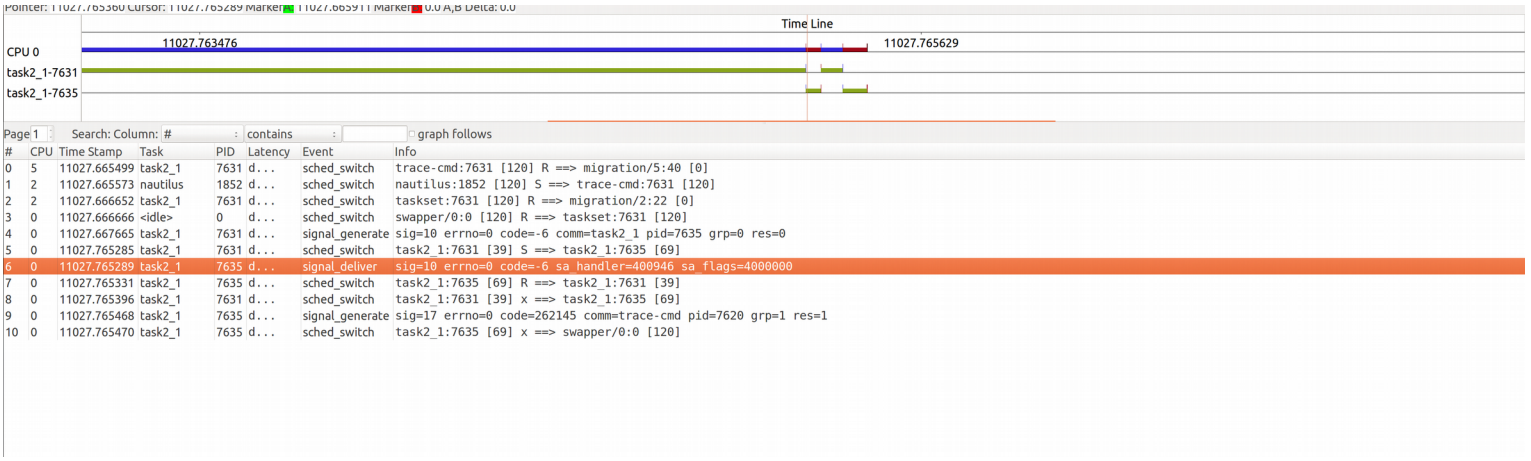
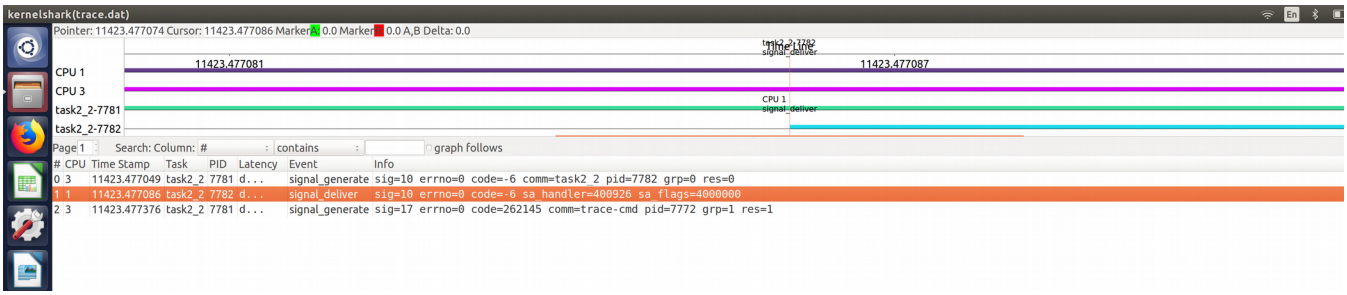
```
//#######################################################
// Project Title : Signaling and event handling in linux
// Created by : Sarvesh Patil & Nagarjun Chinnari
// Date : 1 December 2017
//#######################################################
```





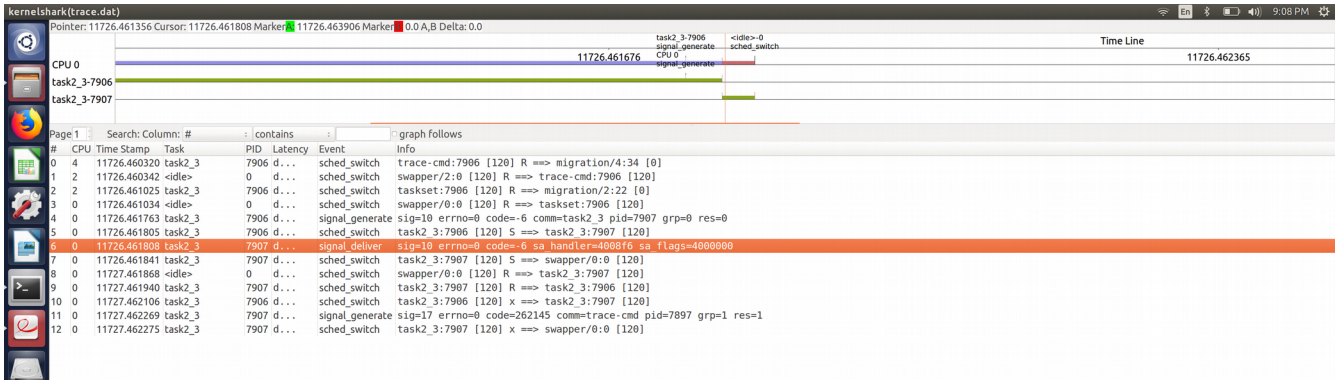
In this case a runnable thread is created and a signal is generated for that particular thread, the thread waits until the higher priority thread exits and then the signal handler is called by the thread of lower priority.

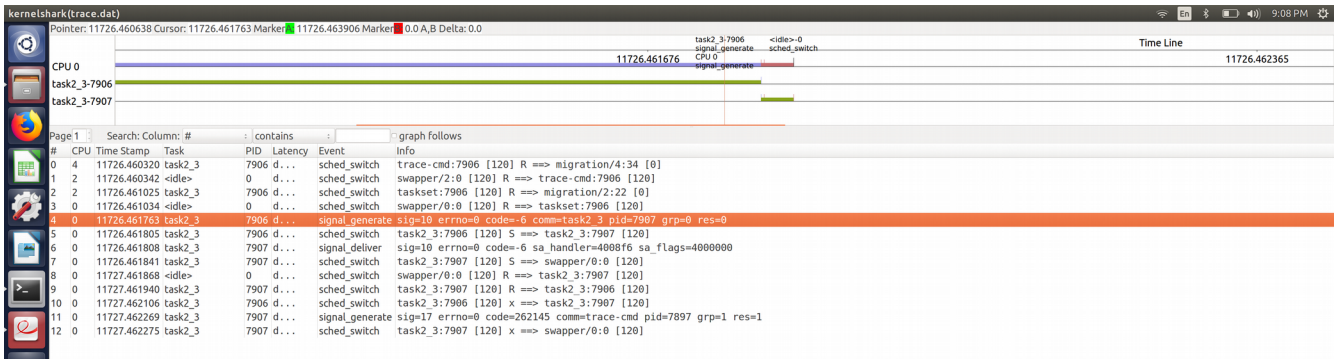
The same can be seen in the above images where signal is generated in the high priority thread(7631) and when it completes its task the low priority thread (7635) receives the signal and the process is killed.



In this case a thread is created and a semaphore is initialized. now when a signal is generated from the main thread to the thread which is in blocked state it receives the signal and gets killed and the program is exited.

The same is shown in the above image where the signal is sent by the main thread(7781) and is received by the thread (7782) which is in blocked state.





In this case a thread is created and a is in sleep state using nanosleep. now when a signal is generated from the main thread to the thread which is in sleeping state it receives the signal and gets killed and the program is exited.

The same is shown in the above image where the signal is sent by the main thread(7906) and is received by the thread (7907) which is in sleeping state.