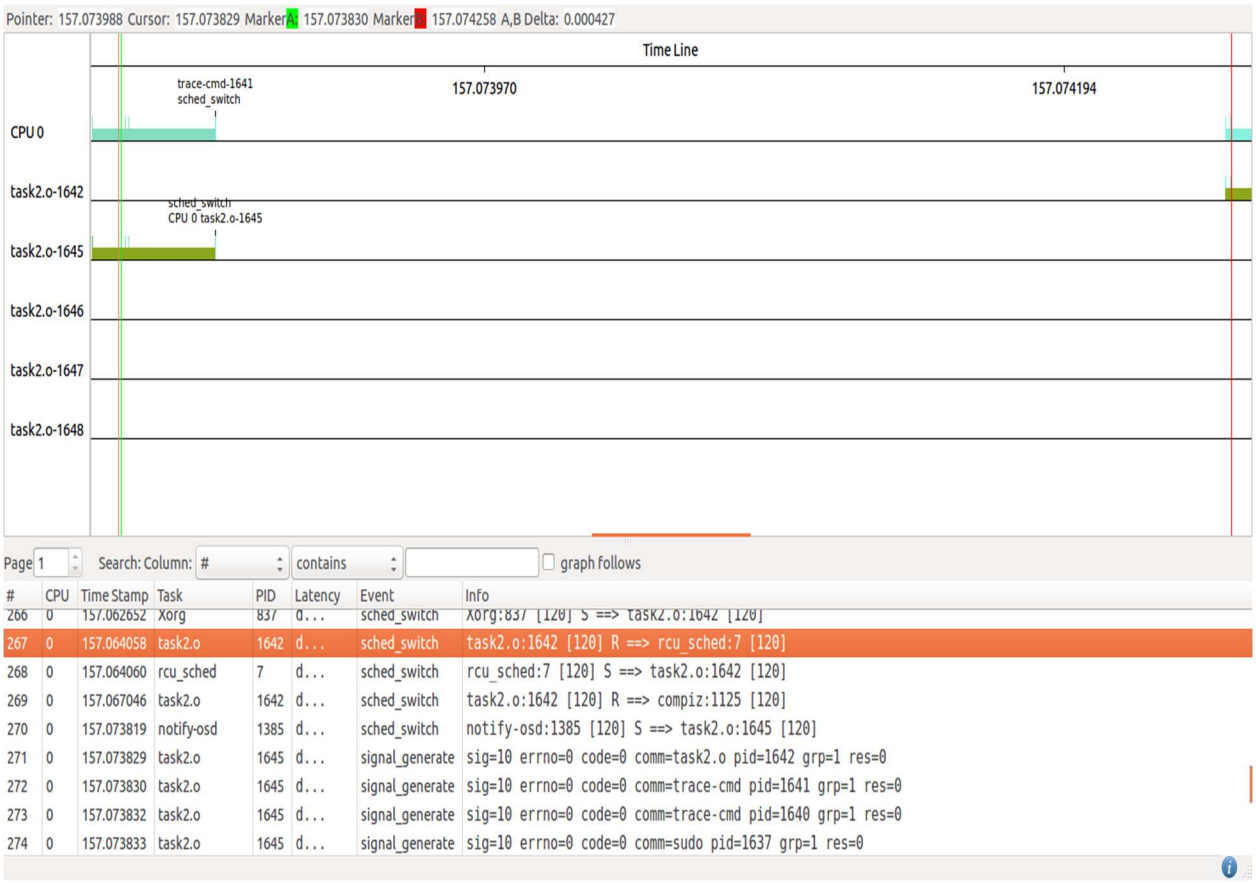
Assignment 5: Event Handle and Signaling
Kernelshark Report

1. **Signal handling when thread is running**

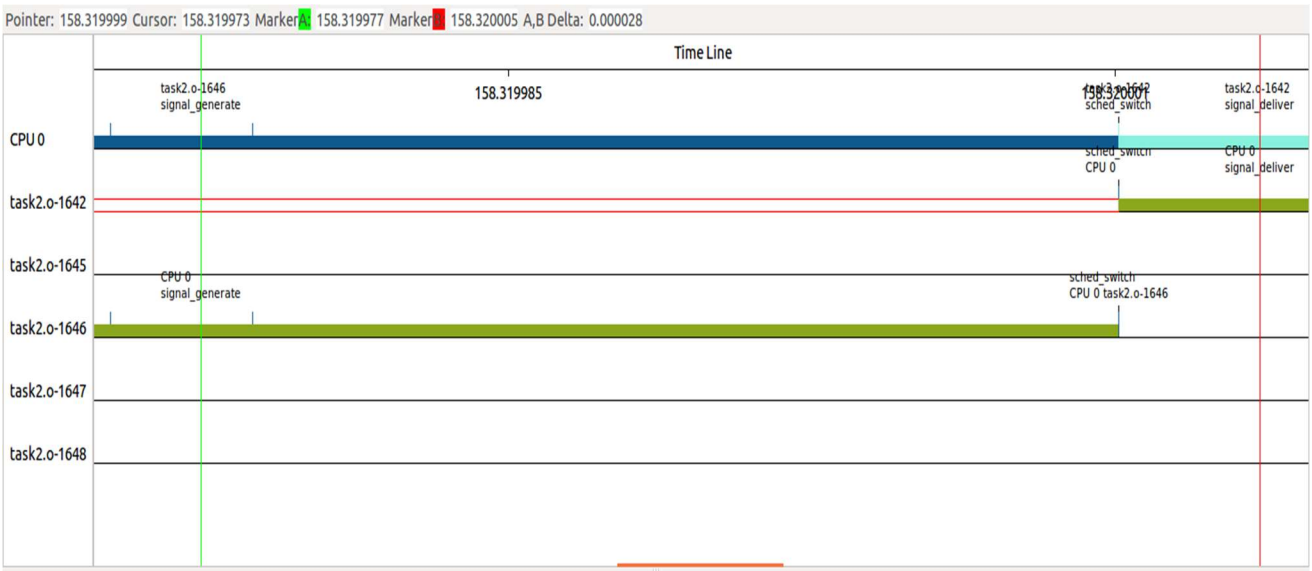
   Thread is created when main function is running continuously and sends SIGUSR1 after 1 second delay. The signal generated is marked with Marker A (threaded 1645) (157.07830) using kill function to the kernel. Kernel waits for high priority tasks to finish and delivers the signal Marker B (157.074258) time difference between generation and delivery is 427us.

   After signal delivery, signal handler sets flag to main to stop the loop.



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
266	0	157.062652	Xorg	837	d...	sched_switch	Xorg:837 [120] S ==> task2.o:1642 [120]
267	0	157.064058	task2.o	1642	d...	sched_switch	task2.o:1642 [120] R ==> rcu_sched:7 [120]
268	0	157.064060	rcu_sched	7	d...	sched_switch	rcu_sched:7 [120] S ==> task2.o:1642 [120]
269	0	157.067046	task2.o	1642	d...	sched_switch	task2.o:1642 [120] R ==> compiz:1125 [120]
270	0	157.073819	notify-osd	1385	d...	sched_switch	notify-osd:1385 [120] S ==> task2.o:1645 [120]
271	0	157.073829	task2.o	1645	d...	signal_generate	sig=10 errno=0 code=0 comm=task2.o pid=1642 grp=1 res=0
272	0	157.073830	task2.o	1645	d...	signal_generate	sig=10 errno=0 code=0 comm=trace-cmd pid=1641 grp=1 res=0
273	0	157.073832	task2.o	1645	d...	signal_generate	sig=10 errno=0 code=0 comm=trace-cmd pid=1640 grp=1 res=0
274	0	157.073833	task2.o	1645	d...	signal_generate	sig=10 errno=0 code=0 comm=sudo pid=1637 grp=1 res=0

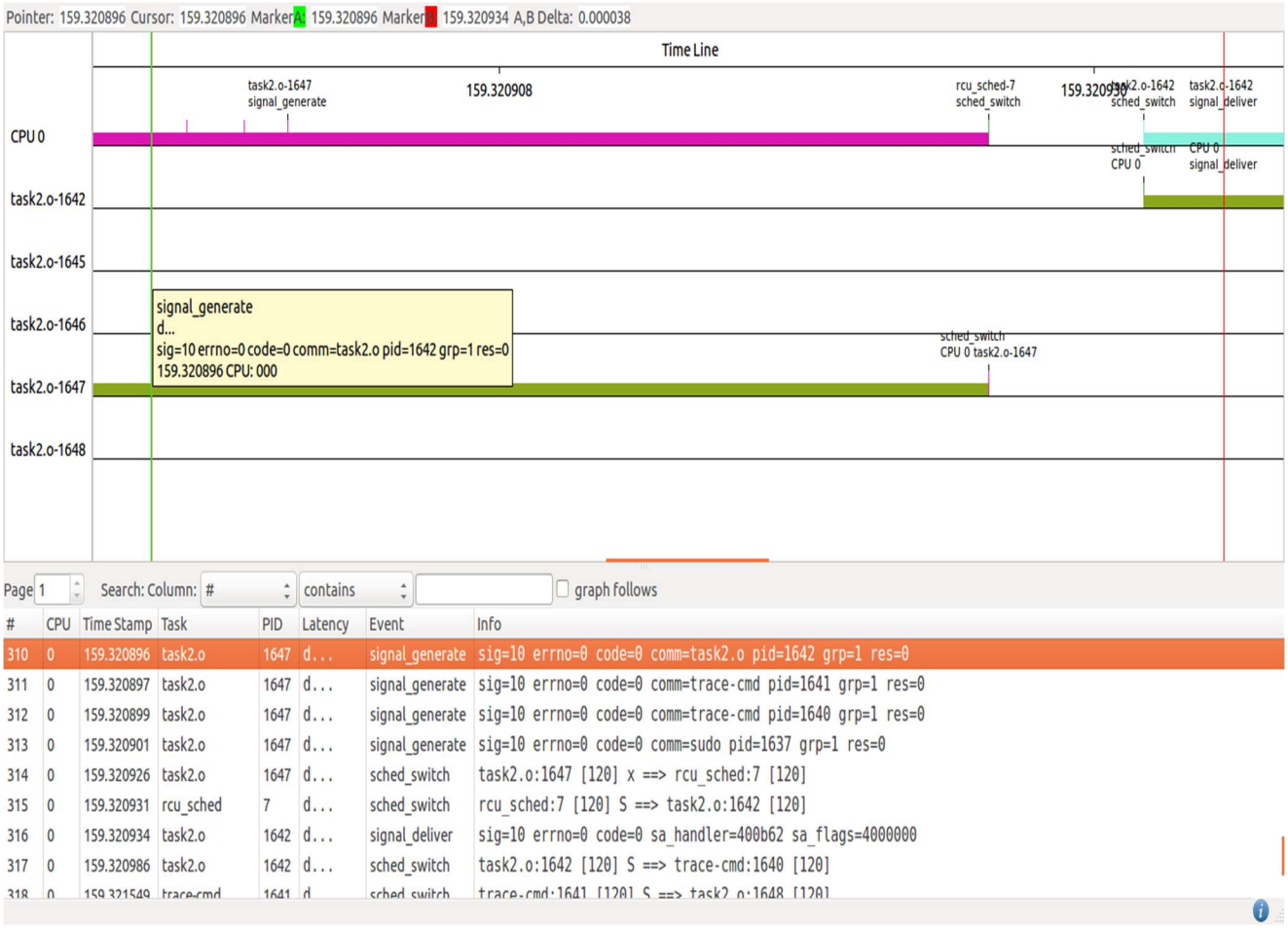
2. **Signal handling when the thread is runnable (Realtime priority)**
   Main thread changes the priority and a thread is created which sends SIGUSR1. Signal is generated (marked with Marker A (threaded 1646) (158.319973)) using kill function to the kernel. Kernel waits for high priority tasks to finish and delivers the signal (Marker B (158.320005)). The time difference between generation and delivery is 28us. After signal delivery, signal handler sets flag to main to stop the loop.



Page 1 Search: Column: # contains graph follows

#	CPU	Time Stamp	Task	PID	Latency	Event	Info
296	0	158.180368	task2.o	1642	d...	sched_switch	task2.o:1642 [0] R ==> kworker/u2:0:5 [120]
297	0	158.228257	compiz	1125	d...	sched_switch	compiz:1125 [120] R ==> task2.o:1642 [0]
298	0	158.319955	task2.o	1642	d...	sched_switch	task2.o:1642 [0] R ==> task2.o:1646 [0]
299	0	158.319973	task2.o	1646	d...	signal_generate	sig=10 errno=0 code=0 comm=task2.o pid=1642 grp=1 res=0
300	0	158.319974	task2.o	1646	d...	signal_generate	sig=10 errno=0 code=0 comm=trace-cmd pid=1641 grp=1 res=0
301	0	158.319976	task2.o	1646	d...	signal_generate	sig=10 errno=0 code=0 comm=trace-cmd pid=1640 grp=1 res=0
302	0	158.319978	task2.o	1646	d...	signal_generate	sig=10 errno=0 code=0 comm=sudo pid=1637 grp=1 res=0
303	0	158.320001	task2.o	1646	d...	sched_switch	task2.o:1646 [0] x ==> task2.o:1642 [0]

3. **Signal handling when the thread is blocked by a semaphore**
   Main thread creates a thread and blocks itself on waiting semaphore. Signal is generated (marked with Marker A (threaded 1647) (159.320896)) using kill function to the kernel. Kernel waits for high priority tasks to finish and delivers the signal (Marker B (159.320934)). The time difference between generation and delivery is 38us. Main task runs before acquiring the semaphore.



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
310	0	159.320896	task2.o	1647	d...	signal_generate	sig=10 errno=0 code=0 comm=task2.o pid=1642 grp=1 res=0
311	0	159.320897	task2.o	1647	d...	signal_generate	sig=10 errno=0 code=0 comm=trace-cmd pid=1641 grp=1 res=0
312	0	159.320899	task2.o	1647	d...	signal_generate	sig=10 errno=0 code=0 comm=trace-cmd pid=1640 grp=1 res=0
313	0	159.320901	task2.o	1647	d...	signal_generate	sig=10 errno=0 code=0 comm=sudo pid=1637 grp=1 res=0
314	0	159.320926	task2.o	1647	d...	sched_switch	task2.o:1647 [120] x ==> rcu_sched:7 [120]
315	0	159.320931	rcu_sched	7	d...	sched_switch	rcu_sched:7 [120] S ==> task2.o:1642 [120]
316	0	159.320934	task2.o	1642	d...	signal_deliver	sig=10 errno=0 code=0 sa_handler=400b62 sa_flags=4000000
317	0	159.320986	task2.o	1642	d...	sched_switch	task2.o:1642 [120] S ==> trace-cmd:1640 [120]
318	0	159.321549	trace-cmd	1641	d...	sched_switch	trace-cmd:1641 [120] S ==> task2.o:1648 [120]

4. **Signal Handling when thread is delayed.**

   Main thread creates thread which is used to send the SIGUSR1 signal. After creating thread main goes to sleep for 10sec.Signal generated (marked with Marker A (threaded 1648) (160.321591)) using kill function to the kernel. Kernel waits for high priority tasks to finish and delivers the signal (Marker B (160.321633)). The time difference between generation and delivery is 41us. After the signal is delivered main wakes up and resume. Remaining time is returned from Nano sleep.