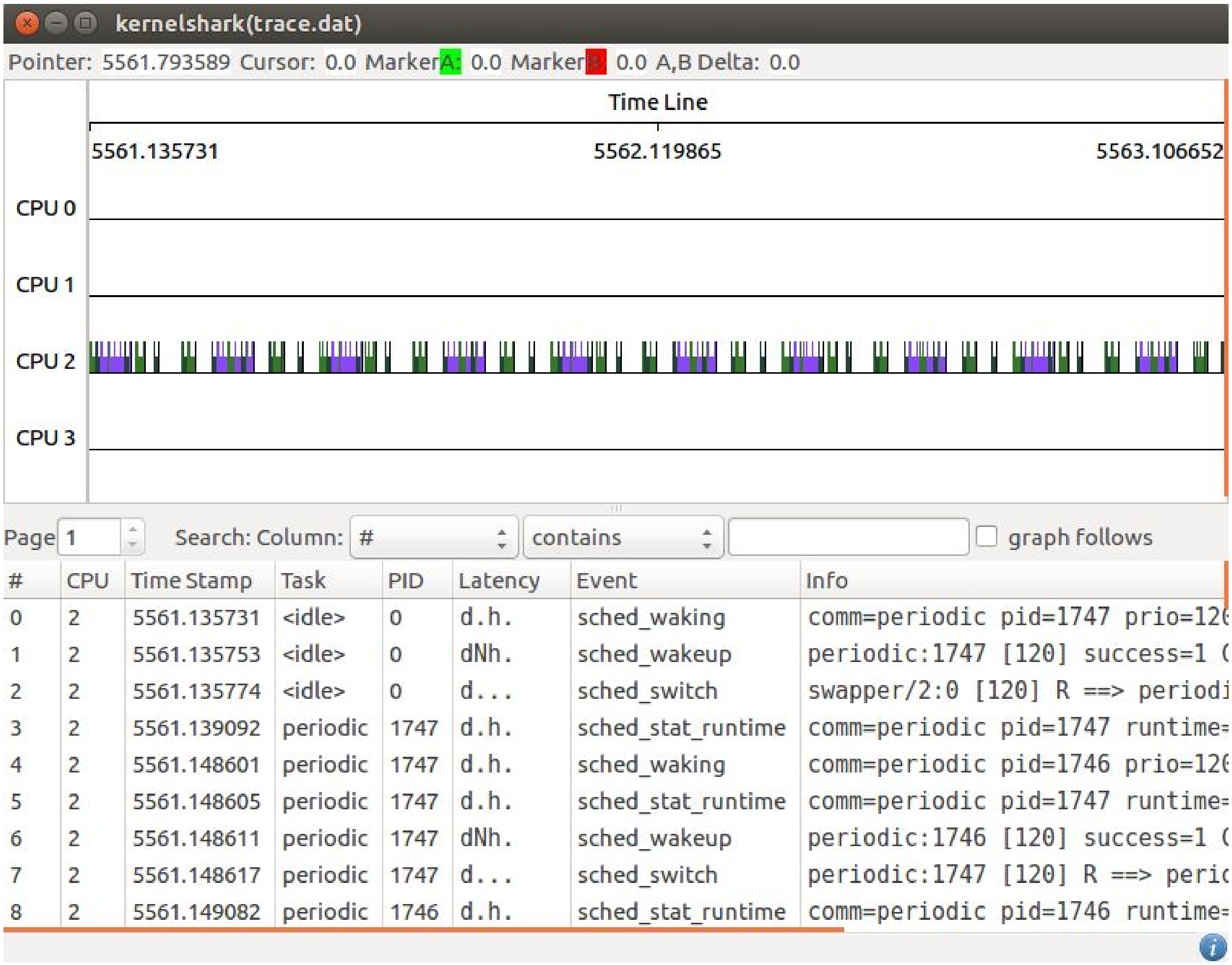
1. Run three instances of your periodic program (periodic.c) on RPI and capture their execution traces for 2 seconds using trace-cmd. Use the following task parameters:
   a. Task 1: C = 10 ms, T = 50 ms, CPUID = 2
   b. Task 2: C = 20 ms, T = 80 ms, CPUID = 2
   c. Task 3: C = 50 ms, T = 200 ms, CPUID = 2



Pointer: 5561.793589 Cursor: 0.0 MarkerA: 0.0 MarkerB: 0.0 A,B Delta: 0.0

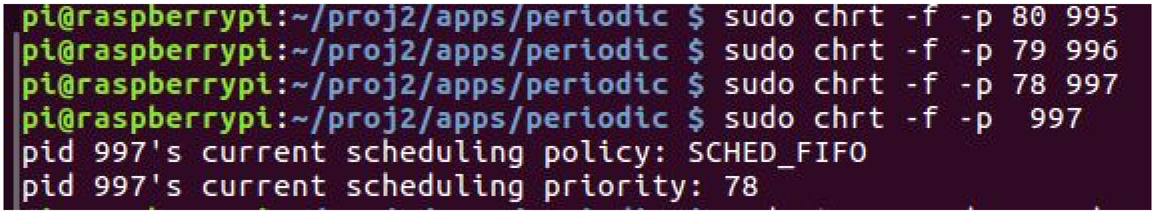
Page 1 Search: Column: # contains

#	CPU	Time Stamp	Task	PID	Latency	Event	Info
0	2	5561.135731	<idle>	0	d.h.	sched_waking	comm=periodic pid=1747 prio=12
1	2	5561.135753	<idle>	0	dNh.	sched_wakeup	periodic:1747 [120] success=1 (
2	2	5561.135774	<idle>	0	d...	sched_switch	swapper/2:0 [120] R ==> periodi
3	2	5561.139092	periodic	1747	d.h.	sched_stat_runtime	comm=periodic pid=1747 runtime=
4	2	5561.148601	periodic	1747	d.h.	sched_waking	comm=periodic pid=1746 prio=12
5	2	5561.148605	periodic	1747	d.h.	sched_stat_runtime	comm=periodic pid=1747 runtime=
6	2	5561.148611	periodic	1747	dNh.	sched_wakeup	periodic:1746 [120] success=1 (
7	2	5561.148617	periodic	1747	d...	sched_switch	periodic:1747 [120] R ==> peri
8	2	5561.149082	periodic	1746	d.h.	sched_stat_runtime	comm=periodic pid=1746 runtime=

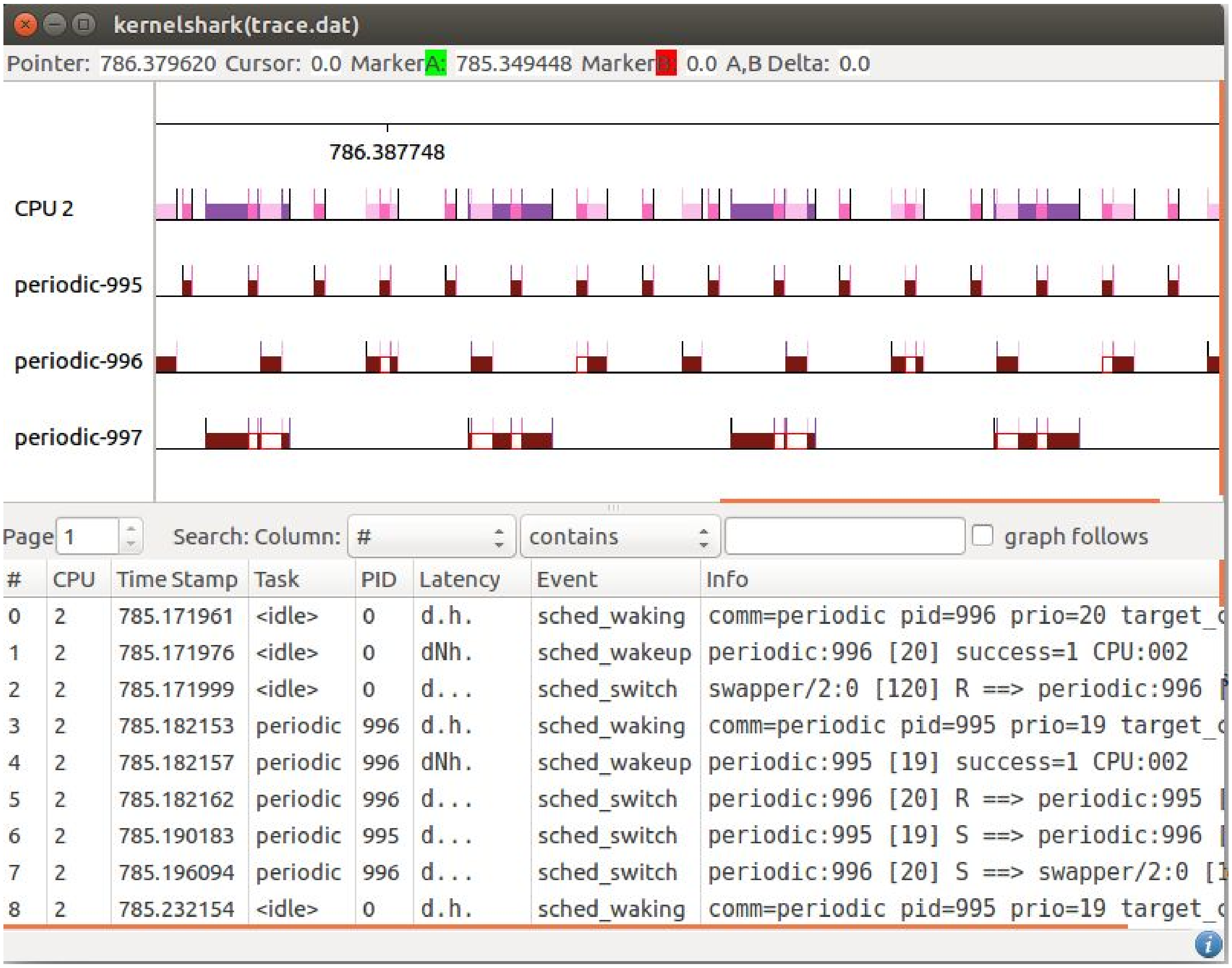
We believe this is happening because in the code we did not account for context switching or priority.

2. Run the same taskset but now, assign real-time priorities to each task, by either chrt or sched_setscheduler. Use SCHED_FIFO policy.
    a. Task 1: C = 10 ms, T = 50 ms, CPUID = 2 -- real-time priority: 80
    b. Task 2: C = 20 ms, T = 80 ms, CPUID = 2 -- real-time priority: 79
    c. Task 3: C = 50 ms, T = 200 ms, CPUID = 2 -- real-time priority: 78

    Attach a screenshot of kernelshark, and give an explanation on task behavior.



```
pi@raspberrypi:~/proj2/apps/periodic $ sudo chrt -f -p 80 995
pi@raspberrypi:~/proj2/apps/periodic $ sudo chrt -f -p 79 996
pi@raspberrypi:~/proj2/apps/periodic $ sudo chrt -f -p 78 997
pi@raspberrypi:~/proj2/apps/periodic $ sudo chrt -f -p  997
pid 997's current scheduling policy: SCHED_FIFO
pid 997's current scheduling priority: 78
```



#	CPU	Time Stamp	Task	PID	Latency	Event	Info
0	2	785.171961	<idle>	0	d.h.	sched_waking	comm=periodic pid=996 prio=20 target_
1	2	785.171976	<idle>	0	dNh.	sched_wakeup	periodic:996 [20] success=1 CPU:002
2	2	785.171999	<idle>	0	d...	sched_switch	swapper/2:0 [120] R ==> periodic:996
3	2	785.182153	periodic	996	d.h.	sched_waking	comm=periodic pid=995 prio=19 target_
4	2	785.182157	periodic	996	dNh.	sched_wakeup	periodic:995 [19] success=1 CPU:002
5	2	785.182162	periodic	996	d...	sched_switch	periodic:996 [20] R ==> periodic:995
6	2	785.190183	periodic	995	d...	sched_switch	periodic:995 [19] S ==> periodic:996
7	2	785.196094	periodic	996	d...	sched_switch	periodic:996 [20] S ==> swapper/2:0 [
8	2	785.232154	<idle>	0	d.h.	sched_waking	comm=periodic pid=995 prio=19 target_

The tasks are now following a distinct pattern. This is now happening because there is a scheduling method in place (FIFO) that gives higher priority. You can clearly see that when the task periodic-995 is released, it takes priority over both other tasks.