**In the name of god**

**Arrash Tahery**

**Soudabeh Allahdadi**

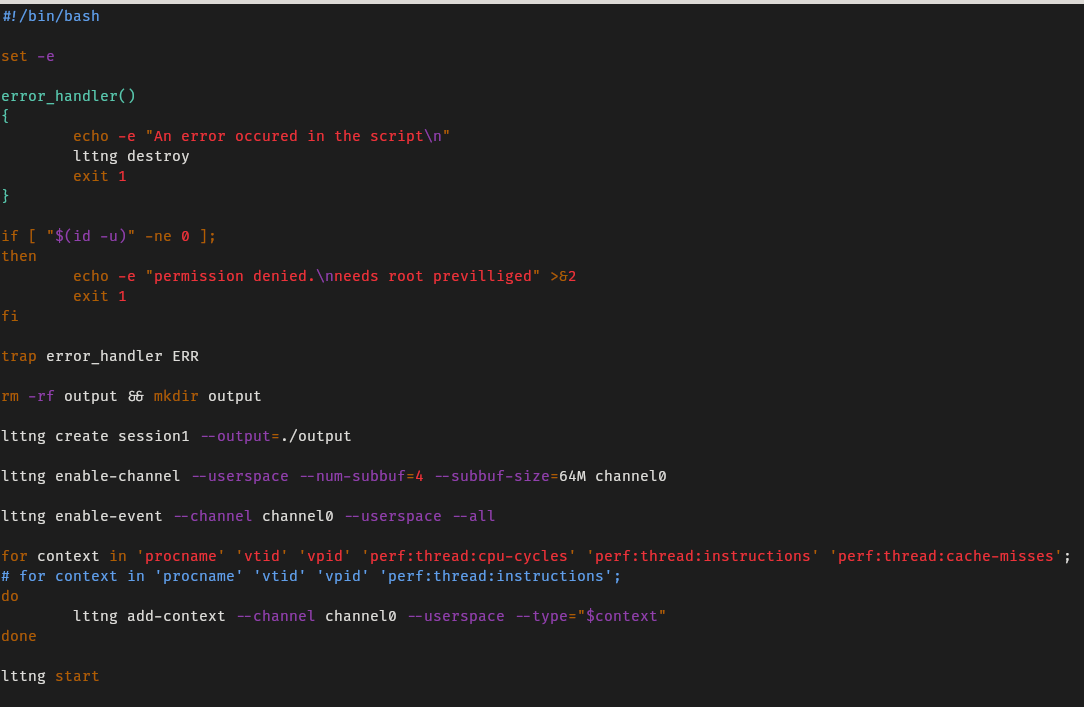
**subject: dpdk tracing**

**requirements:**

1. **dpdk**
2. **lttng**
3. **tracecompass(or something for see the traces)**

**preperation:**

1. compiling dpdk with -finstrument-functions(because we want to use function tracing this flag in gcc will help us to capture data and for capturing data only in special parts of program we add this flag in meson.setup of modules drivers and lib)
2. for capturning data we need to preload liblttng-ust-cyg-profile.so library before running our application.
3. we need to configure lttng. here is mine:

****

1. **running testpmd as server and client using these scripts**

**server:**

**sudo LD\_PRELOAD=/usr/lib/x86\_64-linux-gnu/liblttng-ust-cyg-profile.so ./app/dpdk-testpmd -l 0-1 --proc-type=primary --file-prefix=pmd1 --vdev=net\_memif,role=server -- -i**

**client:**

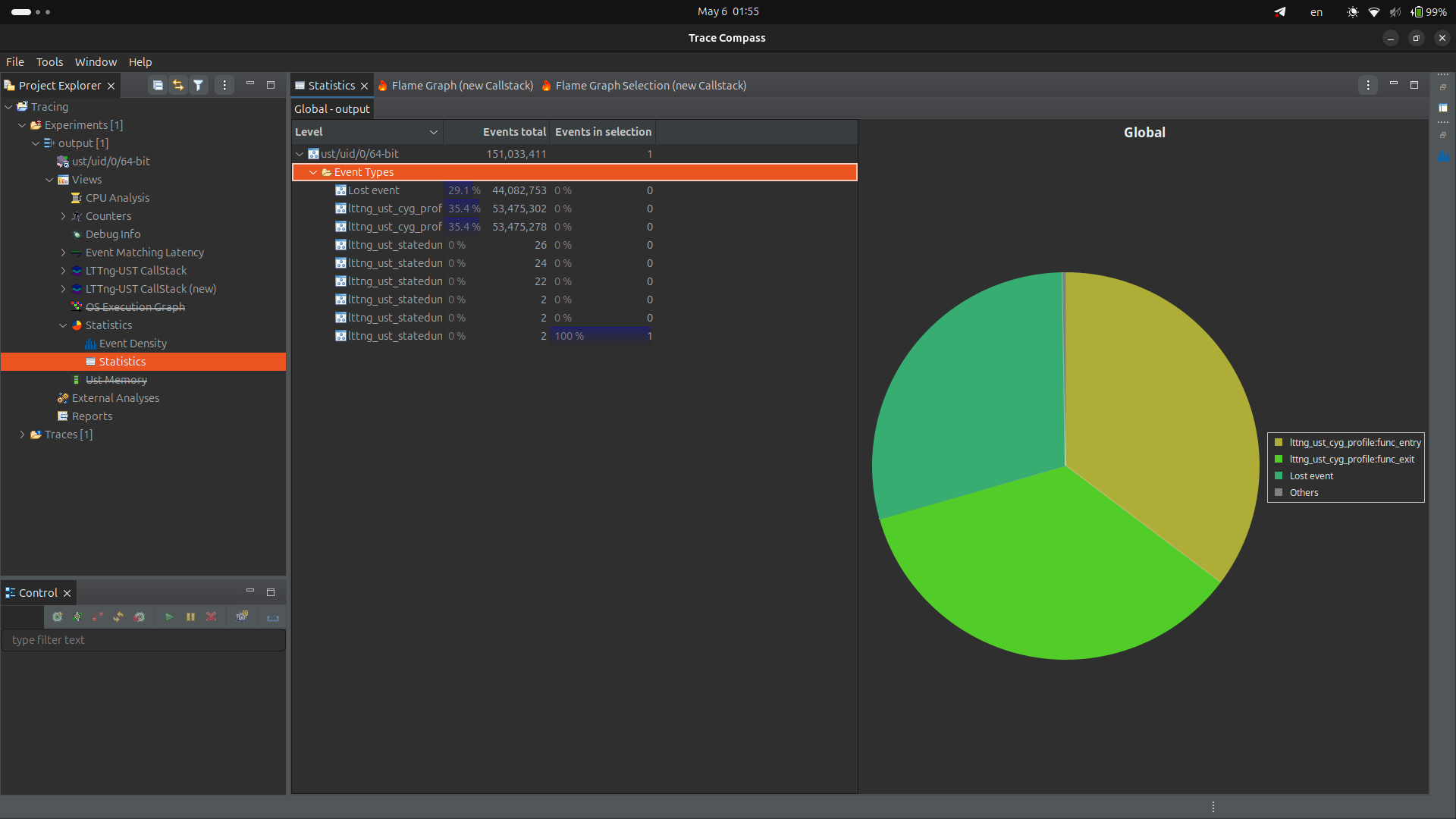
**sudo LD\_PRELOAD=/usr/lib/x86\_64-linux-gnu/liblttng-ust-cyg-profile.so ./app/dpdk-testpmd -l 2-3 --proc-type=primary --file-prefix=pmd2 --vdev=net\_memif -- -i**

1. **after few sconds i will stop this commands and destroy the lttng session:**

**lttng stop**

**lttng destroy**

1. **using tracecompass and spending a little time we will see such results:**



**as you can see i lost 30 percent of my events because i do not have enough memory to capture all the trace data.**

**Analyze:**

**as you can see when we run testpmd it will run five another threads:**

1. **dpdk-testpmd:**

**the main actions of dpdk will happen in this thread. it will create and allocate a ring for revieving packets and pass it to another threads.**

1. **eal-intr-thread:**

**it will handle iterrupts from memif socket and listeners.**

1. **rte-mp-handle:**

**it will handle run time environment parameters and will process messages.**

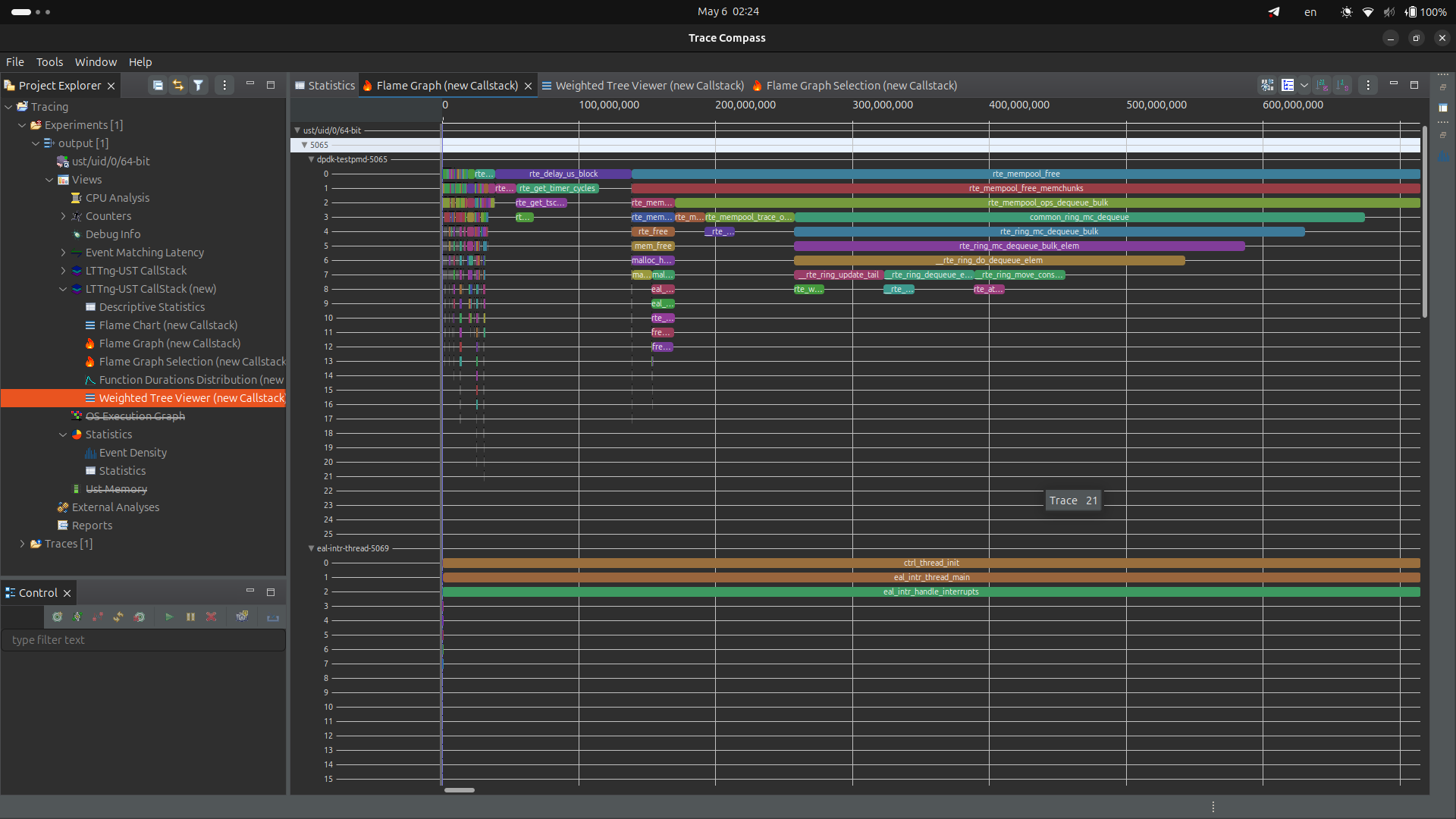
1. **lcore-worker:**

**reading data from memif and copy that in its own memroy.**

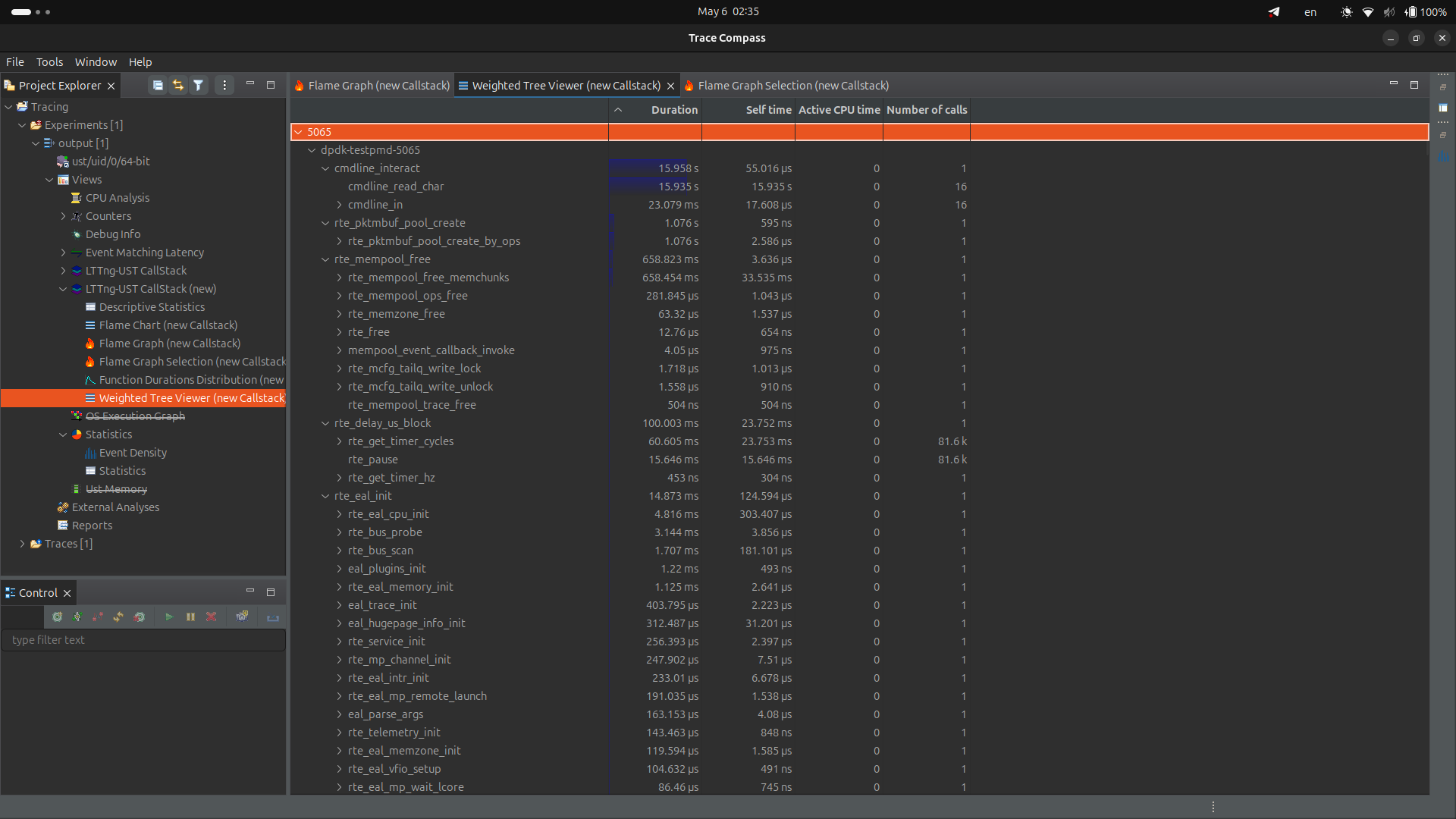
1. **telemetry:**

**listenning on socket and capturing data**

**these all threads are exist in both of server and client role of dpdk-testpmd.**

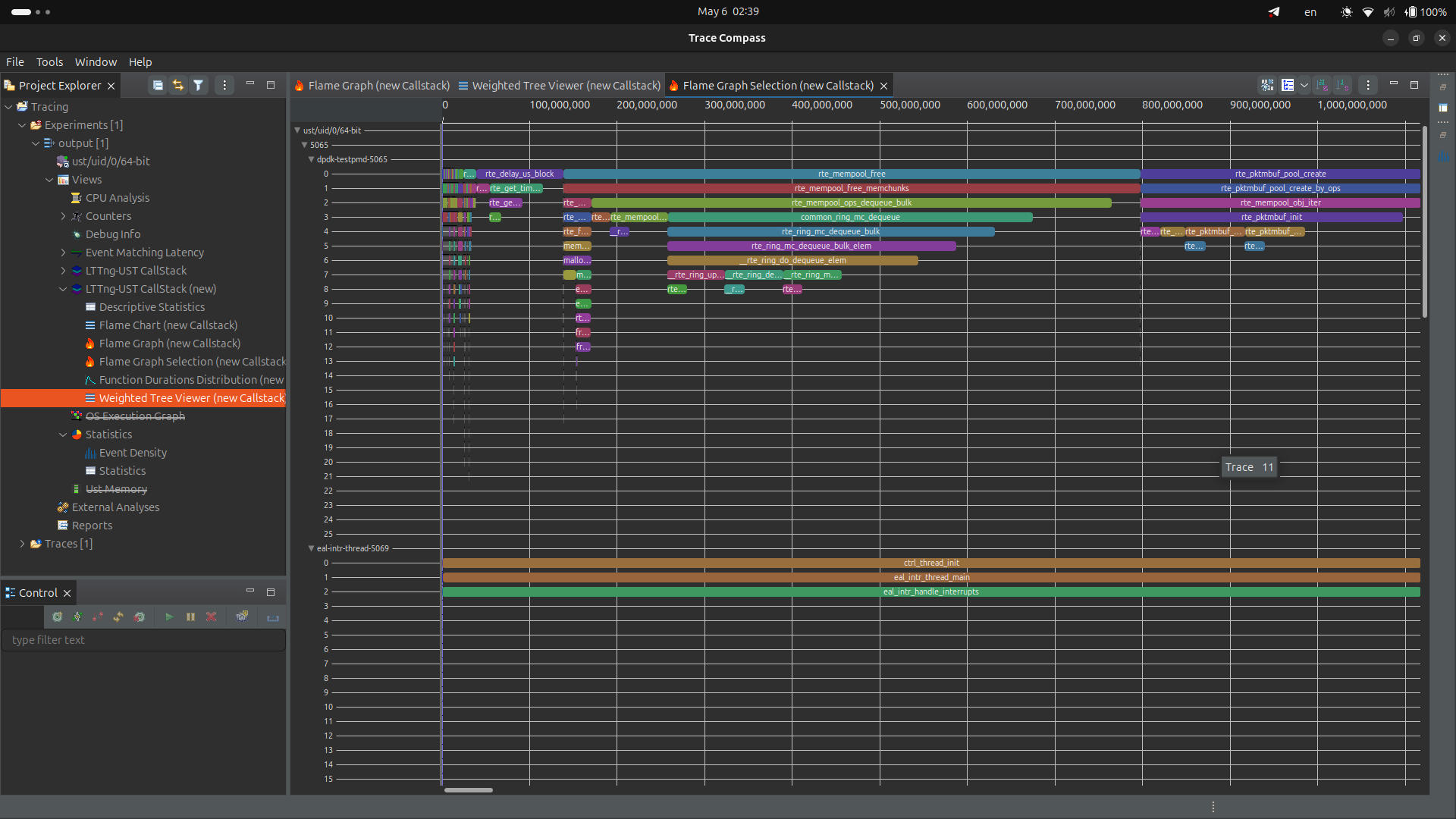


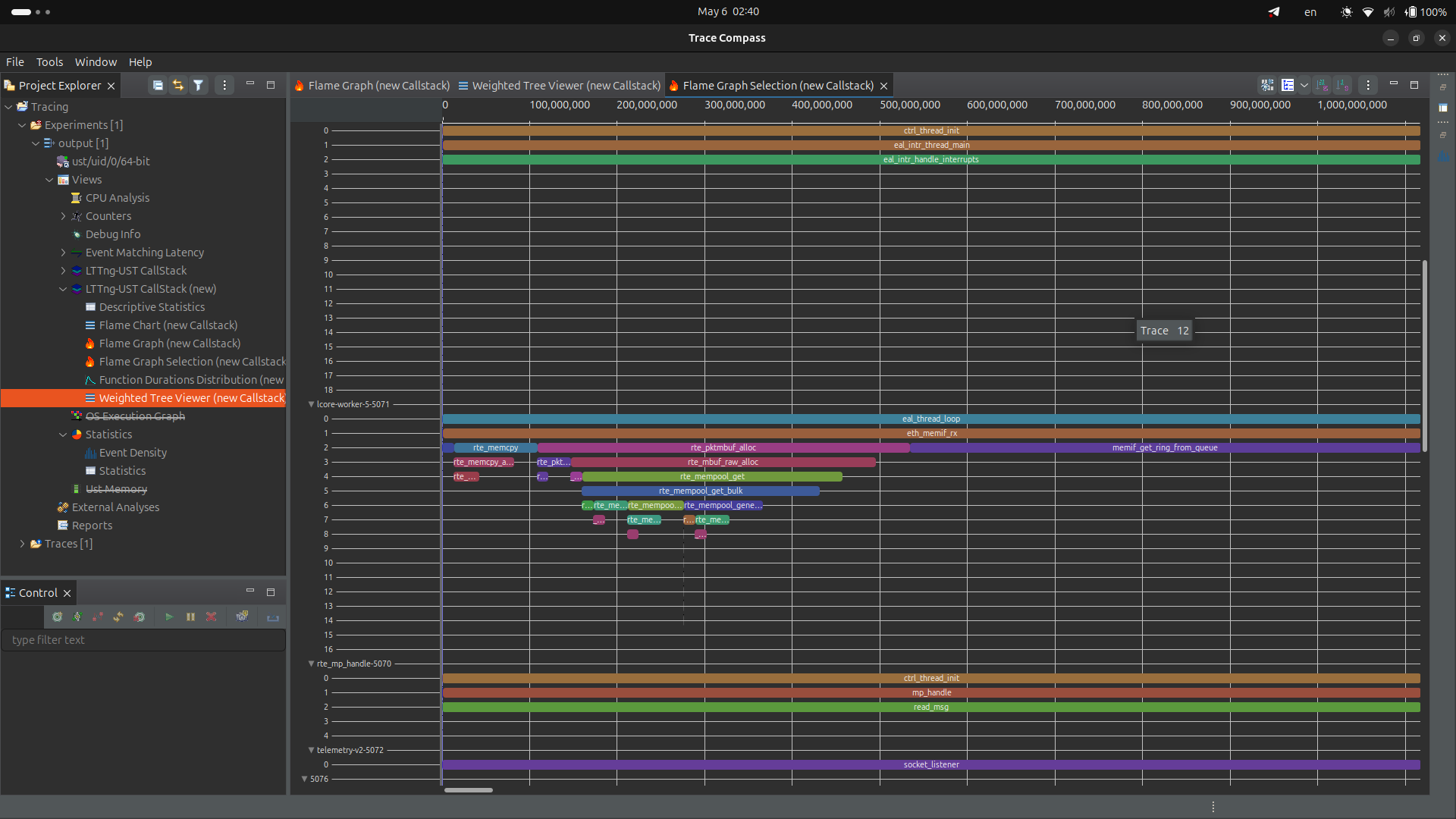
**as we can see in its server most of its execution time was because of interacting with command line.**



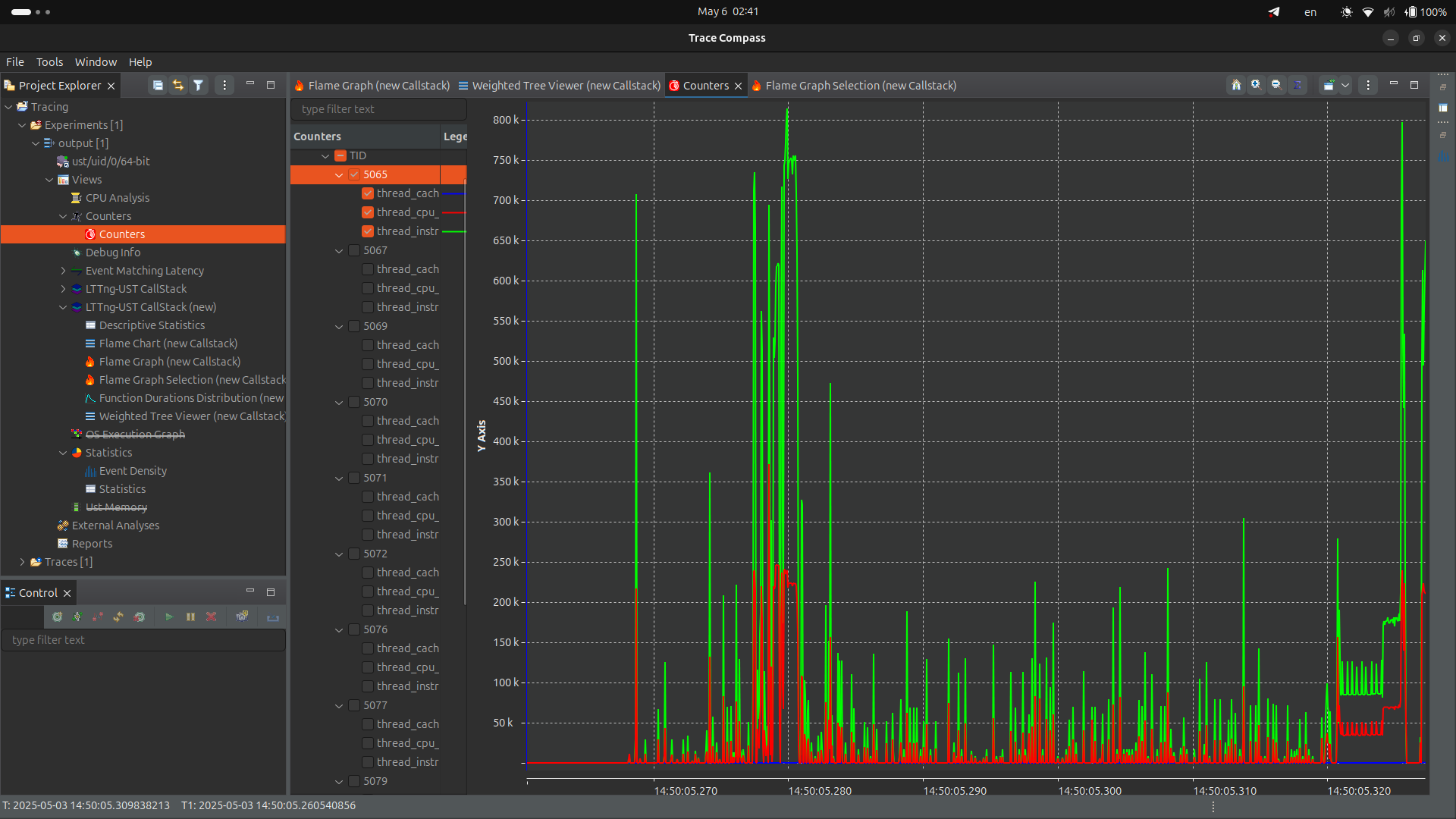
**another big part is for creating and making free memory.**

**when we see more carefully to flame graphs we see that dpdk will allocate large memory and then it creates a queue in it and when it trying to recieve a packet it will get a bucket fill it and then copy that in another part of memory and works on it.**





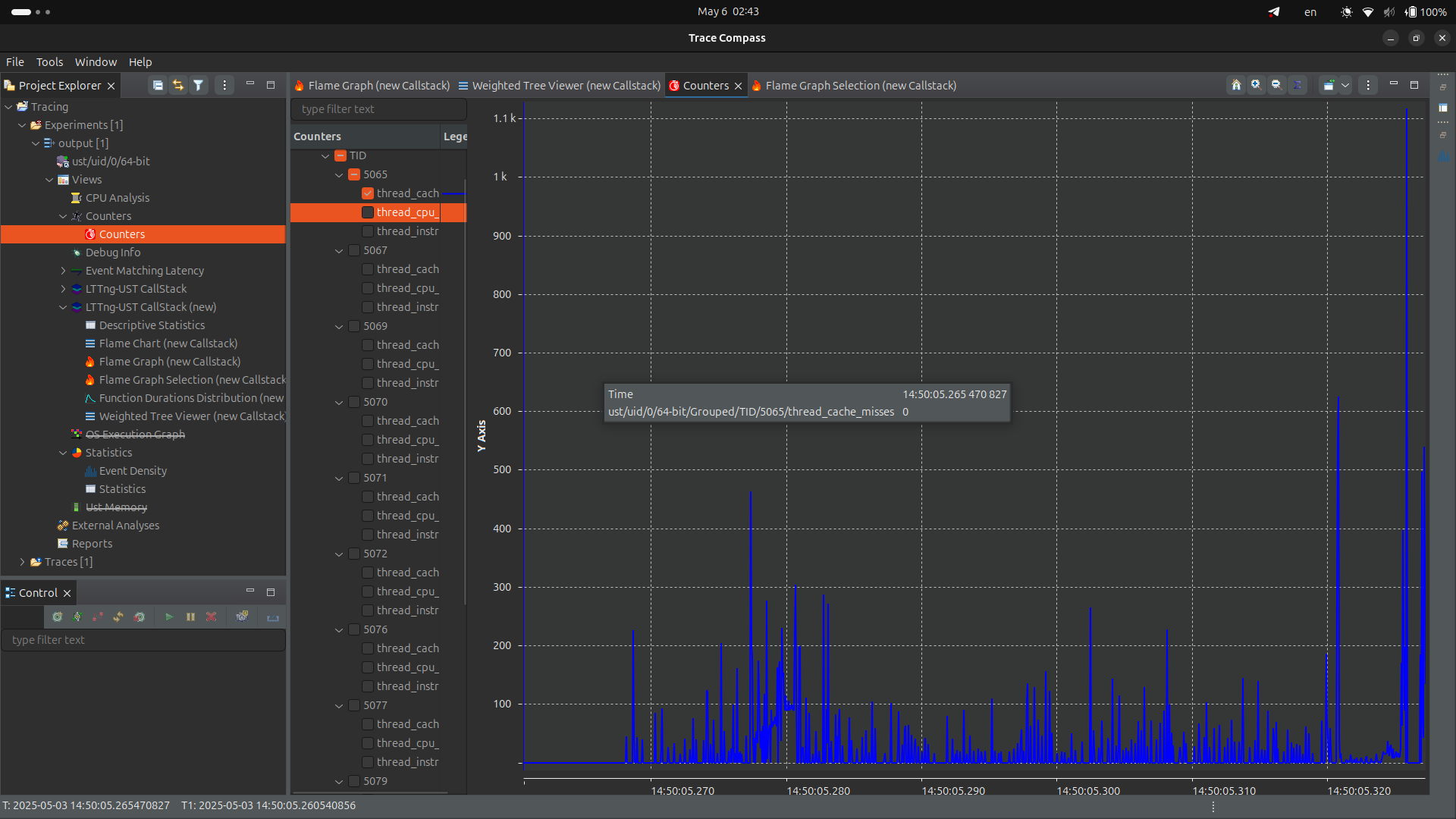
**Analyzing hardware counters**



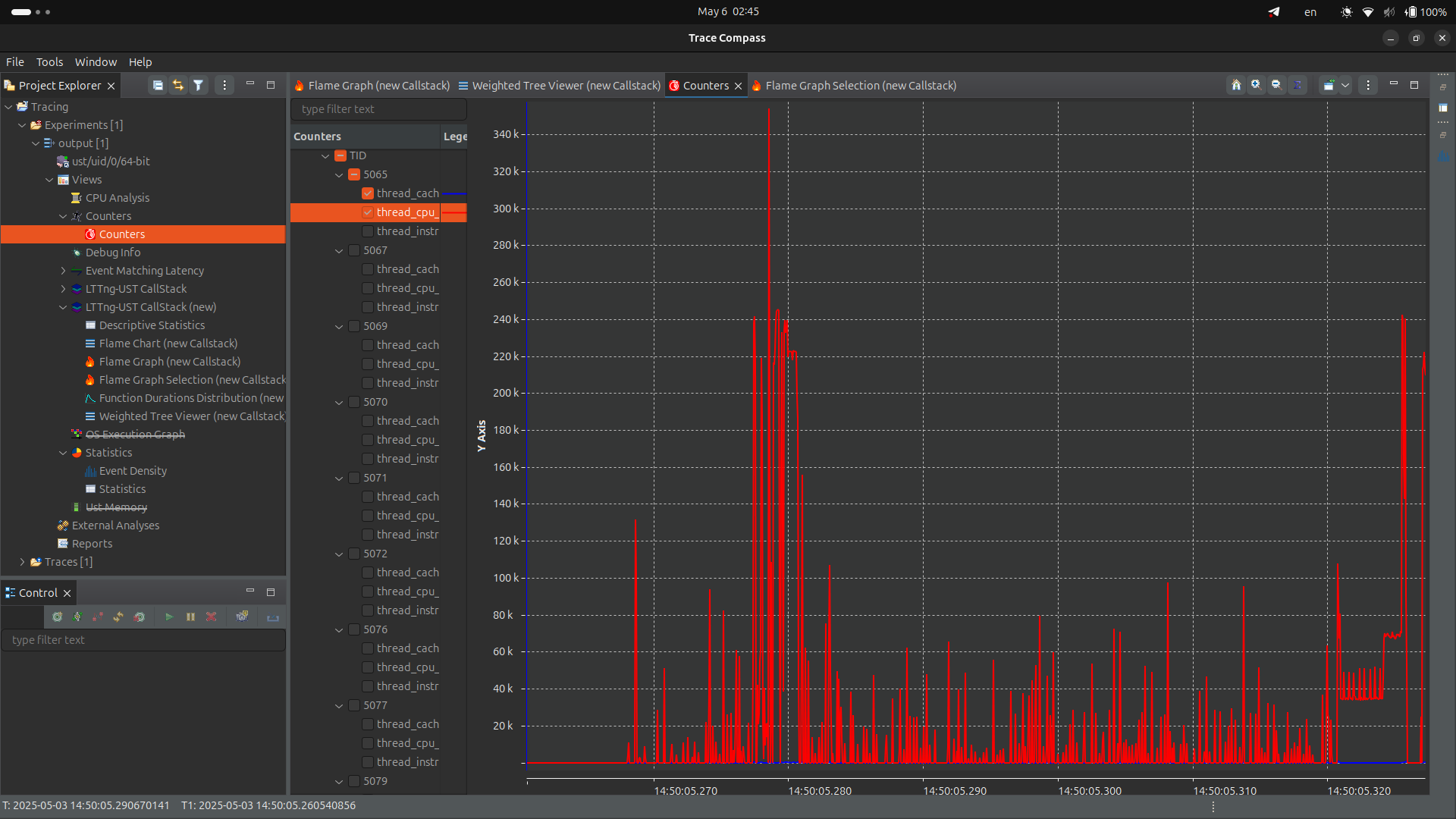
**we want to analyze three types of hardware events:**

**cache\_misses, cpu\_cycles, thread\_instructions**

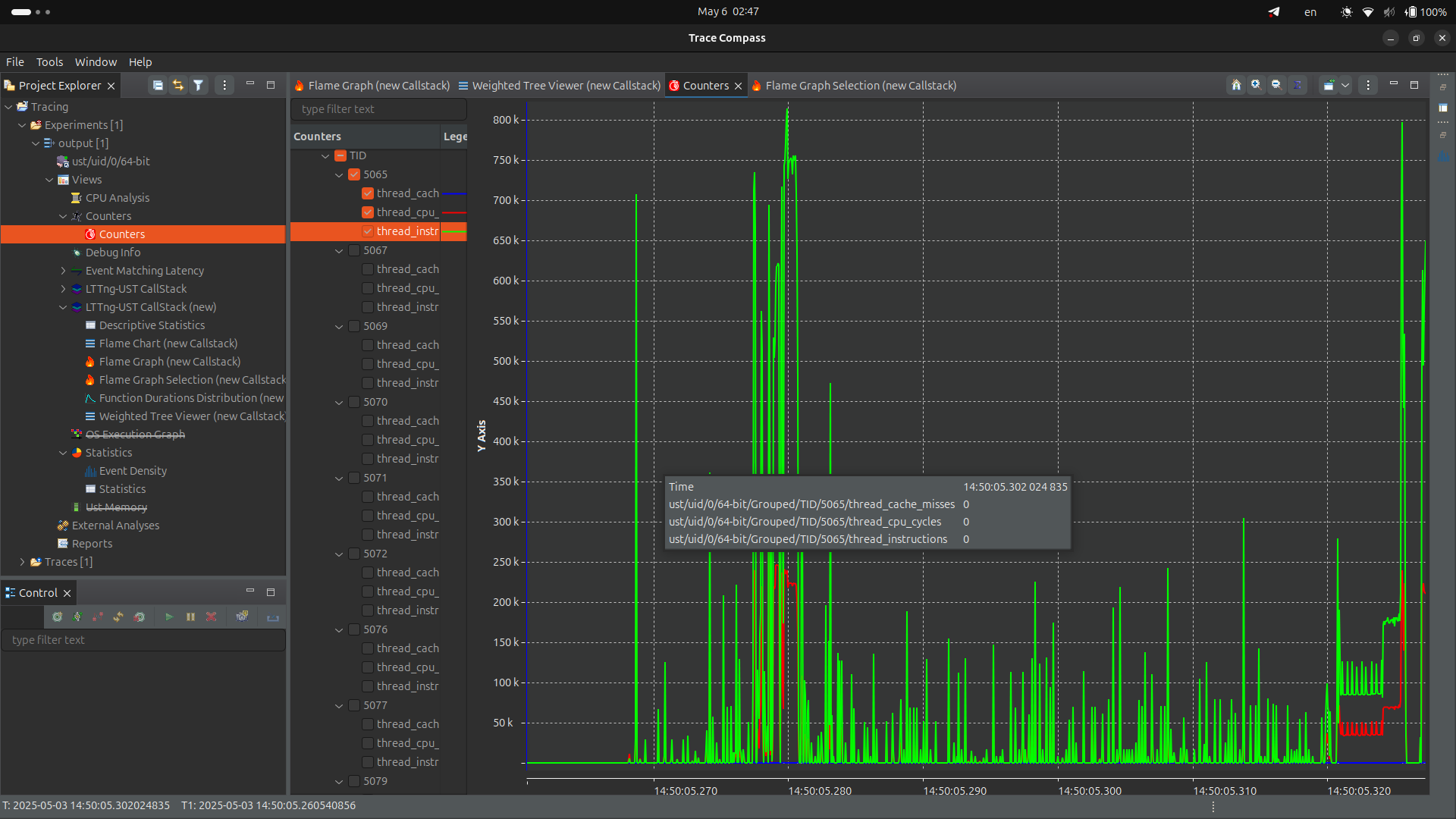
**as we can see number of cache misses will increase at end of the program and when program start to work.**



**if we see this more carefully we will consider that when that cache-misses occured when we have more cpu-cycles:**



**and here there is one interesing thing. number of thread instructions is more greater that cpu cycles but how is this possible?let’s look at this more carefully:**



**in the top of them cpu cycles that program uses is 500k while there are 800k instructions. the response is that modern cpus will use some optimizations on the program for running it in less cycles. for example it could run commands while it fetching another commands from memroy with considering variable dependencies.**