Task1

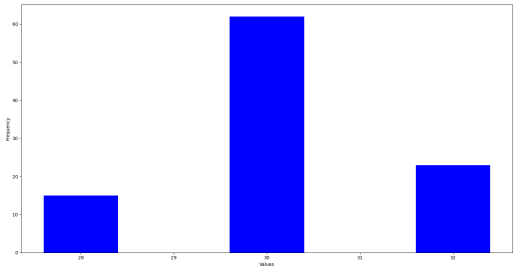
The source file name is a1t1.c. In this file the function map can be found at line 22.

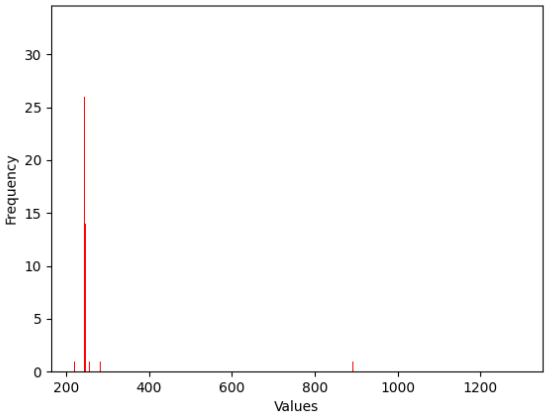
Task2

The source file name is a1t2.c. In this file the function flush_reload can be found at line 47.

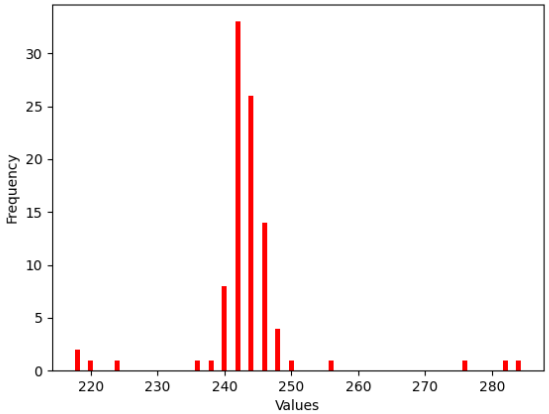
Average and meadian memory access times for cached and non-cached locations are printed in the console. Example (in cycles):
Averages: 30.16 (cached), 262.82 (uncaced)
Means: 30 (cached), 244 (uncached)

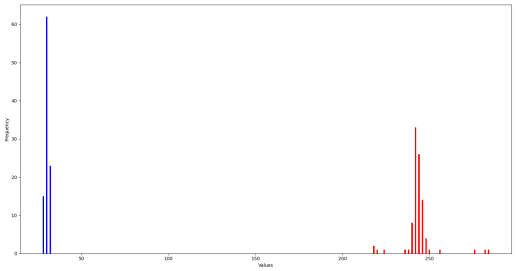
Plots: (cached, uncached)





(uncached without outliers, all without outliers)





Task3

The source file name is a1t3.c. In this file the function monitor can be found at line 67.

I started with a period of 500. That had some odd behavior, e.g. being always 10-40 cycles late. By trying to reduce the period manually the latency remained the same until it jumped to 460-480. Thus, I abandoned the idea of compensating cycles here.

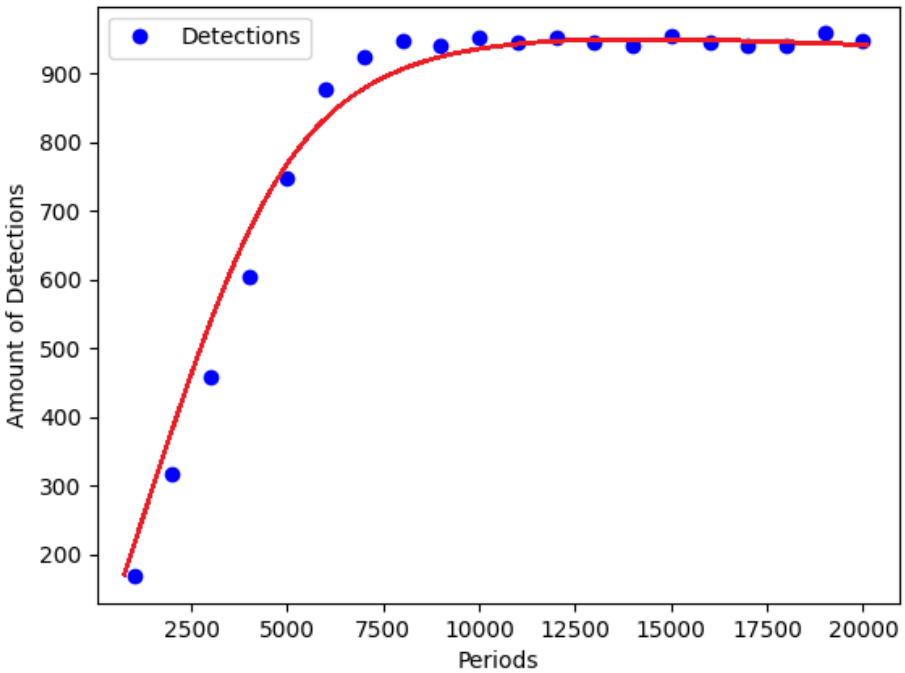
Later, I figured out that setting the period to 10,000 makes it more reliable. I can detect 95 % of activity or accesses and the waiting period I could print was between 0 and 40 only which in my eyes does not require improvement. I think the assignment of the timestamp value to my variable could have caused little latency sometimes.

Task4

The source file name is monitor.c.

With a period of 10,000 in the monitor program I measured following amounts of accesses when using fr_test with a period of 1000, 2000, ..., 20000.

168, 316, 459, 604, 748, 876, 924, 947, 941, 952, 945, 951, 945, 940, 954, 946, 940, 940, 958, 948

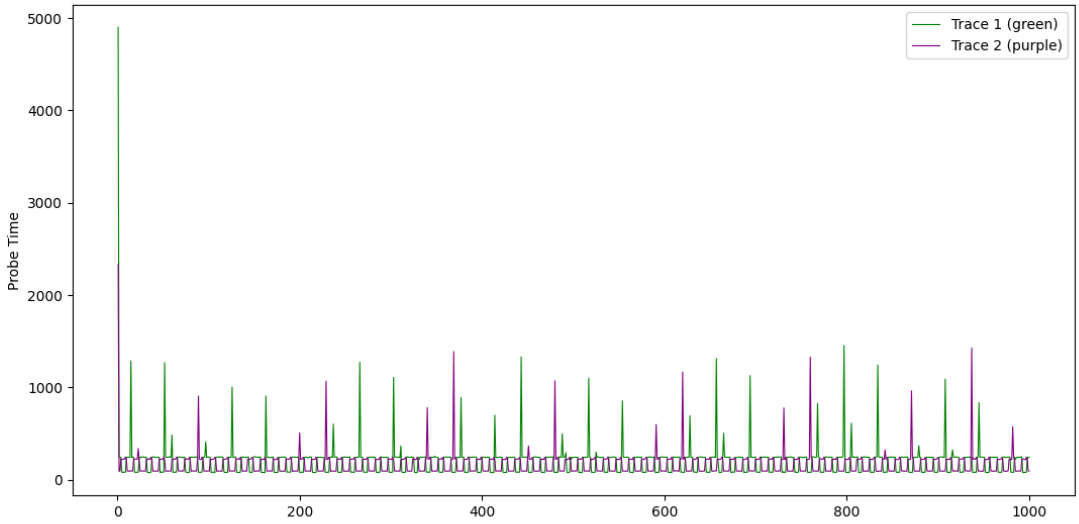


Task5

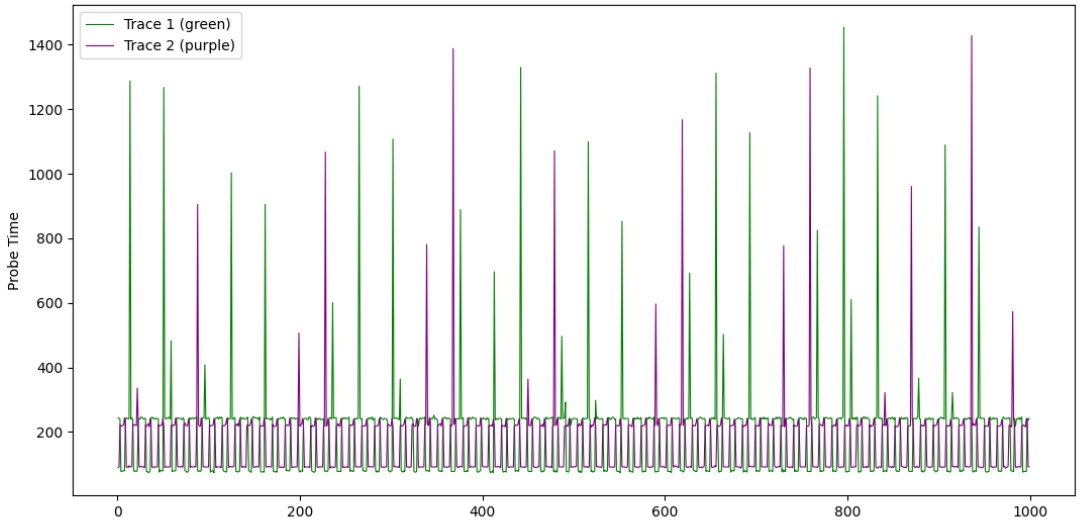
The source file name is fr_trace.c, trace.out contains the measured probe times.

Here are the plots, one with all 1000 probe times and one with the first measurement removed since the long probe time probably comes from loading from memory.

all



first probe time removed



Task6

The source file name is madgpg_monitor.c, gnupg.txt contains the measured probe times.

A figure of a section of probes