CS 251: Real-Time Embedded Systems Homework 3

Team 09: Manish Chugani, Yeahn Kim, Jesus Martinez Vega

1. Report the memory access time of mem_alloc and mem_alloc_lock (assignment 4.2.1 & 4.2.2) with the memory size of 1 MB, 10MB, and 100 MB. For each case, run at least 10 times and provide average memory access. Give a brief discussion comparing the results.

trial	mem_alloc, 1 MB (ns)	mem_alloc, 10 MB (ns)	mem_alloc, 100 MB (ns)
1	1664602	18147076	136507226
2	1730504	18045258	91429310
3	1658112	18149447	139517351
4	1694577	17681706	91715093
5	1826674	18134259	92313232
6	1654501	18353106	91749636
7	1648556	18165001	91713958
8	1757857	17959952	139549052
9	1832690	17926531	91682994
10	1707194	17884710	91682994
average	**1717527**	**18044705**	**105786085**

Table 1. Average memory access times (in ns) of mem_alloc with memory sizes of 1 MB, 10 MB, and 100 MB. The average values are rounded to the nearest whole number.

trial	mem_alloc_l, 1 MB (ns)	mem_alloc_l, 10 MB (ns)	mem_alloc_l, 100 MB (ns)
1	19351	215387	1274609
2	19630	194332	1274647
3	19093	195294	1259665
4	19111	196035	1237813
5	19370	194554	1268739
6	19666	195739	1241868
7	19500	196054	1217646
8	19667	197055	1268832
9	19241	194647	1211442
10	19519	194832	1254572
average	**19415**	**197393**	**1250983**

Table 2. Average memory access times (in ns) of mem_alloc_lock with memory sizes of 1 MB, 10 MB, and 100 MB. The average values are rounded to the nearest whole number.

The average memory access times for mem_alloc_lock are overall shorter than the average memory access times for mem_alloc (mem_alloc_lock is around ~90x faster for all memory sizes). Interestingly, the average memory access times appears to grow at the same rate for both mem_alloc and mem_alloc_lock (~10x increase and ~6x increase from 1 MB to 10 MB and 10 MB to 100 MB respectively).

2. Report the kernel logs of /dev/segment_info (assignment 4.3.1) for mem_alloc_lock (assignment 4.2.2) with the memory size of 1 KB and 100 MB.

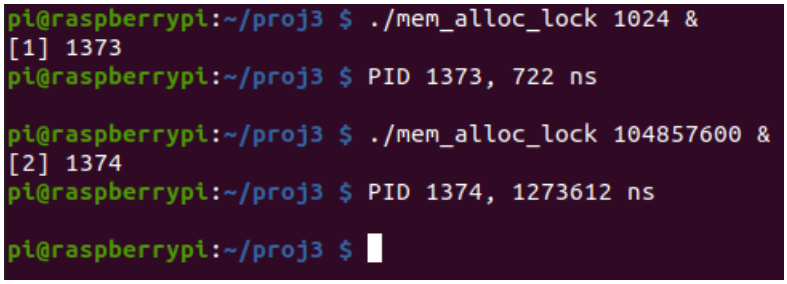

Figure 1. Process 1373 is mem_alloc_lock run with 1 KB. Process 1374 is mem_alloc_lock run with 100 MB.

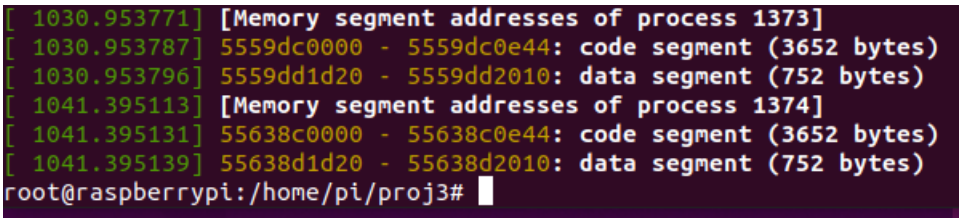

Figure 2. Kernel logs of /dev/segment_info run with processes 1373 and 1374.

3. Report the kernel logs of /dev/vm_areas (assignment 4.3.2) for mem_alloc_lock (assignment 4.2.2) with the memory size of 1 KB and 100 MB.

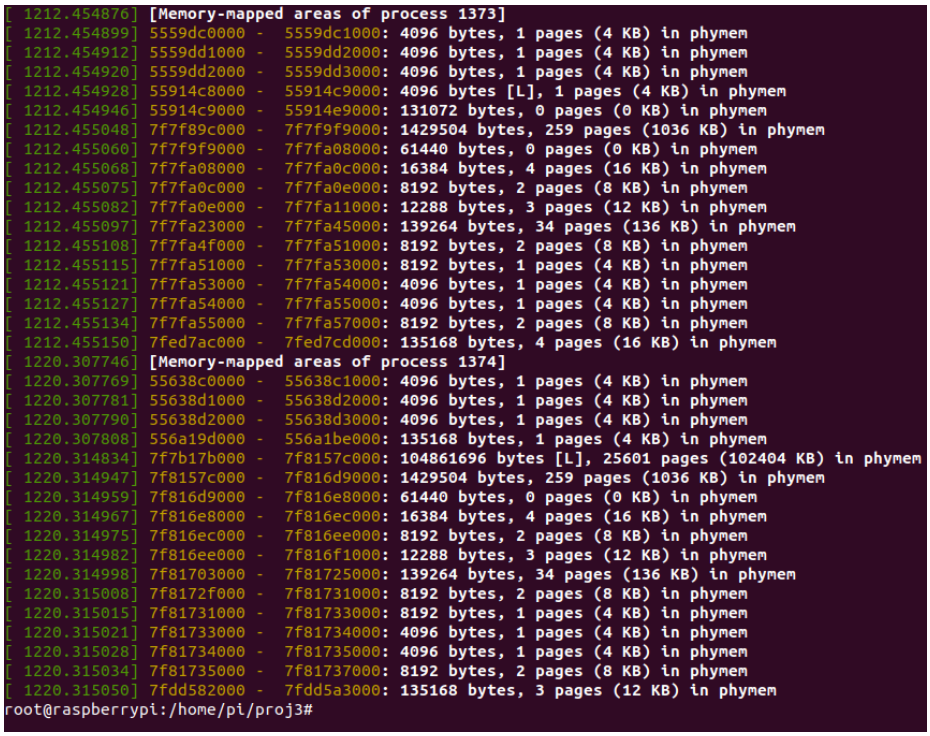

Figure 3. Kernel logs of /dev/vm_areas run with processes 1373 and 1374. Processes 1373 and 1374 are the same as seen in figure 1.

4. If the OS kernel uses virtual memory with demand paging but does not provide mlock-like functions, then what would be a workaround that a user-level program can do to mitigate unpredictable memory access delay at runtime? Recall lecture slides.

During the initialization phase of your program, you could allocate some memory (either on the stack or heap) and then touch the allocated block of memory using memset(). Just allocating the memory isn't enough, it must also be written to (using memset) for the operating system to bring the pages into physical memory.

5. Give a brief summary of the contributions of each member.
    - Manish Chugani: Wrote and tested mem_alloc.c and mem_alloc_lock.c.
    - Yeahn Kim: Wrote and tested task_alloc.c.
    - Jesus Martinez Vega: Wrote and tested segment_info.c and vm_areas.c. Wrote the lab report.