ICS Homework 12

May 31, 2020

1 Organization

1.1

If we only use 0x400000-0x800000, compare the memory usage of page table between single level and 4 level.

1.2

If we have enough physical memory and use the whole 2^{48}B virtual space, compare the memory usage again.

1.3

Given the following page table and cr3=0x1000, please translate the virtual addresses to physical addresses. Assume the machine is x86_64, which uses 4 level page table.

Physical Address of PTE	Address Part in PTE
0x1000	0x4000
0x1008	0x4000
0x4000	0x5000
0x5000	0x7000
0x5008	0x8000
0x5010	0x9000
0x5018	0x3000
0x5020	0x5000
0x7008	0x8000
0x7010	0x10000

Physical Address	Virtual Address
0x1234	
0x2234	
0x8000802135	

1.4

Assume on a x86_64 machine without cache, we have an int `a[70]` at virtual address `0x8000344f00`, a program access `a[0], a[1], ..., a[69]` one by one. Before the first access, we have only an empty L1 page table.

1.4.1

How many physical page is used after the program access `a[0], a[64], a[69]`? (Remember the page table is stored in physical page too)

1.4.2

How many memory accesses and page fault happened in the program if we have no TLB?

1.4.3

How many memory accesses and page fault happened in the program if we have a full associative infinite TLB?

2 System Software

2.1

Using the progress graph in Figure 12-21 of file "badcnt.c", draw the following trajectories out and point out the value of `cnt` after the execution (assume the value of `cnt` is 0 initially for each trajectory).

1. H1,L1,H2,L2,U2,U1,S2,T2,S1,T1
2. H2,L2,U2,H1,S2,L1,T2,U1,S1,T1
3. H1,L1,U1,H2,L2,S1,U2,S2,T1,T2
4. H1,H2,L1,U1,S1,L2,U2,T1,S2,T2

2.2

```
1  #include "csapp.h"
2  #define N 4
3
4  void *thread(void *vargp) {
5      int myid = *((int)vargp);
6          printf("in thread %d\n", myid);
7          return NULL;
8  }
9
10 int main() {
11     pthread_t tid[N];
12         int *ptr;
13
14         for (int i = 0; i < N; i++) {
```

```
15          ptr = malloc(sizeof(int));
16          *ptr = i;
17          // create a thread running
18          // with argument ptr
19          // your code here
20          pthread_create(&tid[i], NULL,
21                  thread, ptr);
22
23          free(ptr);
24          }
25
26          for (int i = 0; i < N; i++)
27              pthread_join(tid[i], NULL);
28  }
```

1. Complete the previous code according to the comment.
2. Is there any race condition in the previous code? Why or why not?