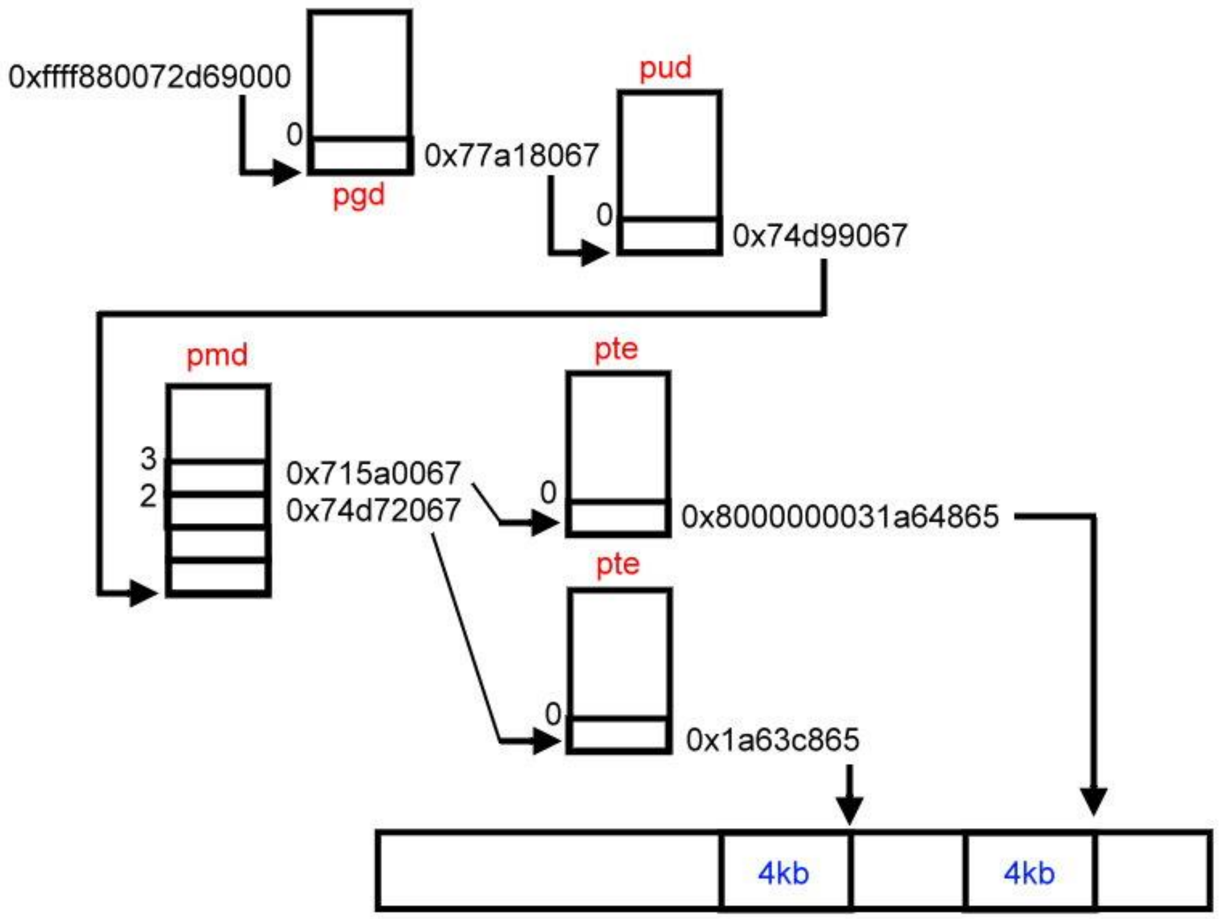
OS 作業 HW3 資工 07 0316316 謝明恩

PART A

1. We need 5 memory pages for the page table structure. We do not need 8 memory pages for the page table structure. The first three page tables can share memory spaces in this example.

2.



PART B

When forking a program, the memory location of the data remains the same (both virtual and physical), but the permission of that memory location will change to read only. When the child or parent wants to write to those locations, it will trigger a page fault, which will trigger a handler in the kernel to do Copy on Write. After writing data to stackbuf1, the virtual address will not change but the data will be copied and the physical address of stackbuf1 for the child will change (In the same time the mapping table entry of the child will change). At this moment, the physical address of stackbuf2 will not change. After writing data to stackbuf2, the physical address of stackbuf2 will change. The main purpose of the program is to see the Copy on Write process.

PART C

We first set the library path so the loader can find the shared library. Before writing to the data segment, the code and data segment of the two programs will share. But when we write to the data segment, it will trigger the copy on write process and it will copy the data into a new address. This address will not be shared between processes meaning that the two processes will write to different physical address. The main purpose of the program is to observe the share library phenomenon between processes, and the Copy on Write process when we write to the data.