Project 2

- **Description:**
  -
    - Question 1 (**60** points)
      1. Requirement:
         - In this project, for both Questions 1 and 2, you need to write a new system call `my_get_physical_addresses(unsigned int * initial, int len_vir, unsigned int * result, int len_phy)` so that a process can use it to get the physical addresses of some virtual addresses.
         - The return value of this system call is either 0 or a positive value. 0 means that an error occurs when executing this system call. A positive value means the system call is executed successfully.
         - The first argument of this system call is the address of an unsigned integer array. Each element of the array stores a virtual address of a process.
         - The second argument of this system call is the number of elements in the array.
         - The third argument is the address of an unsigned integer array. Each element with index `i` of this array stores the physical address of the virtual address stored as element `i` in the array pointed by the first argument.
         - The fourth argument is the number of elements stored in the array pointed by the third argument.

      2.
```
//prototype of the new system call is as follows:

int my_get_physical_addresses(unsigned int * initial, int len_vir, unsigned int * result, int len_phy)
```

      3. Write a multi-thread program with three threads using the new system call to show how the following memory areas are shared by these threads. Your program must use variables with storage class `__thread`. The memory areas include code segments, data segments, BSS segments, heap segments, libraries, stack segments, and thread local storages. You need to draw a figure as follows to show your results.

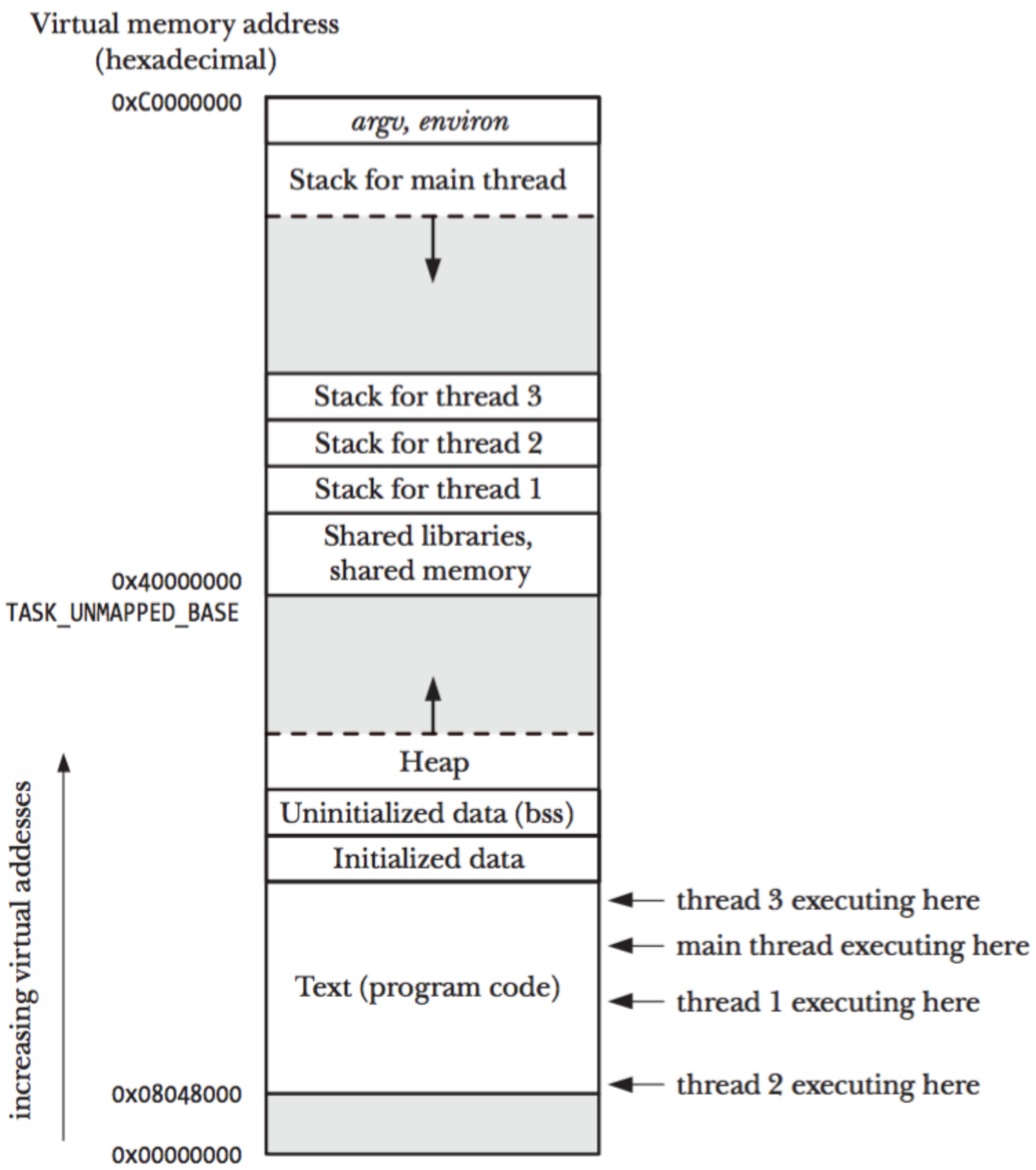

Figure 29-1: Four threads executing in a process (Linux/x86-32)

-- by Jason/cntofu.com

      4. Hint:
         - Two threads show a physical memory cell (one byte) if both of them have a virtual address that is translated into the physical address of the memory cell.
         - The kernel usually does not allocate physical memories to store all code and data of a process when the process starts execution. Hence, if you want kernel to allocate physical memories to a piece of code, execute the code first. If you want kernel to allocate physical memories to a variable, access the variable first.
         - Inside the Linux kernel, you need to use function `copy_from_user()` and function `copy_to_user()` to copy data from/to a user address buffer.
         - Check the "Referenced Material" part of the Course web site to see how to add a new system call in Linux.
    - Question 2 (**50** points)
      - Write a program with the new system call `my_get_physical_addresses(unsigned int * initial, int len_vir, unsigned int * result, int len_phy)` to check how memory areas are shared by two processes that execute this program simultaneously.
      - Hint:
        - When making your check, both related processes must be in progress. Hence you may need to use function `sleep()` to guarantee this requirement.
        - Inside the Linux kernel, you need to use function `copy_from_user()` and function `copy_to_user()` to copy data from/to a user address buffer.
        - Check the "Referenced Material" part of the Course web site to see how to add a new system call in Linux.
- **Project Submission:**
  - NEW The due day of report submission is **23:55 12/12** (updated: 1st Dec.)
  - NEW The demo will be held from **8th Dec. 2021** to **9th Dec. 2021** (updated: 1st Dec.)
  - NEW Please fill out this form to choose your demo time before **5th Dec. 2021** (updated: 1st Dec.)
  - On site demo of this project is required.
  - During on site demo, the TAs will execute several programs written by them to check the correctness of your system calls.
  - When demonstrating your projects, the TAs will ask you some questions regarding to your projects. Part of your project grade is determined by your answers to the questions.
  - You need to submit both an electronic version and a hard-copy of your project report to the TAs.
    - The electronic versions could be sent to the TAs through e-mails.
    - Do not forget writing the names and student IDs of all members in your team.
    - Your report should contain:
      - Your source code
      - the execution results
  - Late submission will **NOT** be accepted.
- **Reference:**
  - G. T. Wang, C 語言 pthread 多執行緒平行化程式設計入門教學與範例。
  - Jason/cntofu.com, 深入 Linux 多線程編程。
  - Will, C pthread_create 傳遞參數的用法。
  - Chin-Hung Liu, Work Note-pthread。
  - MIT, Thread-Local Storage