Project 1

- **Description:**
    -
        - Question (100 points)
            1. Requirement:
                - In this project, you need to write a new system call void * my_get_physical_addresses(void *) so that a process can use it to get the physical address of a virtual address of a process.
                - The return value of this system call is either 0 or an address value. 0 means that an error occurs when executing this system call. A non-zero value means the physical address of the logical address submitted to the system call as its parameter.
            2.

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

void * my_get_physical_addresses(void *)
```

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.

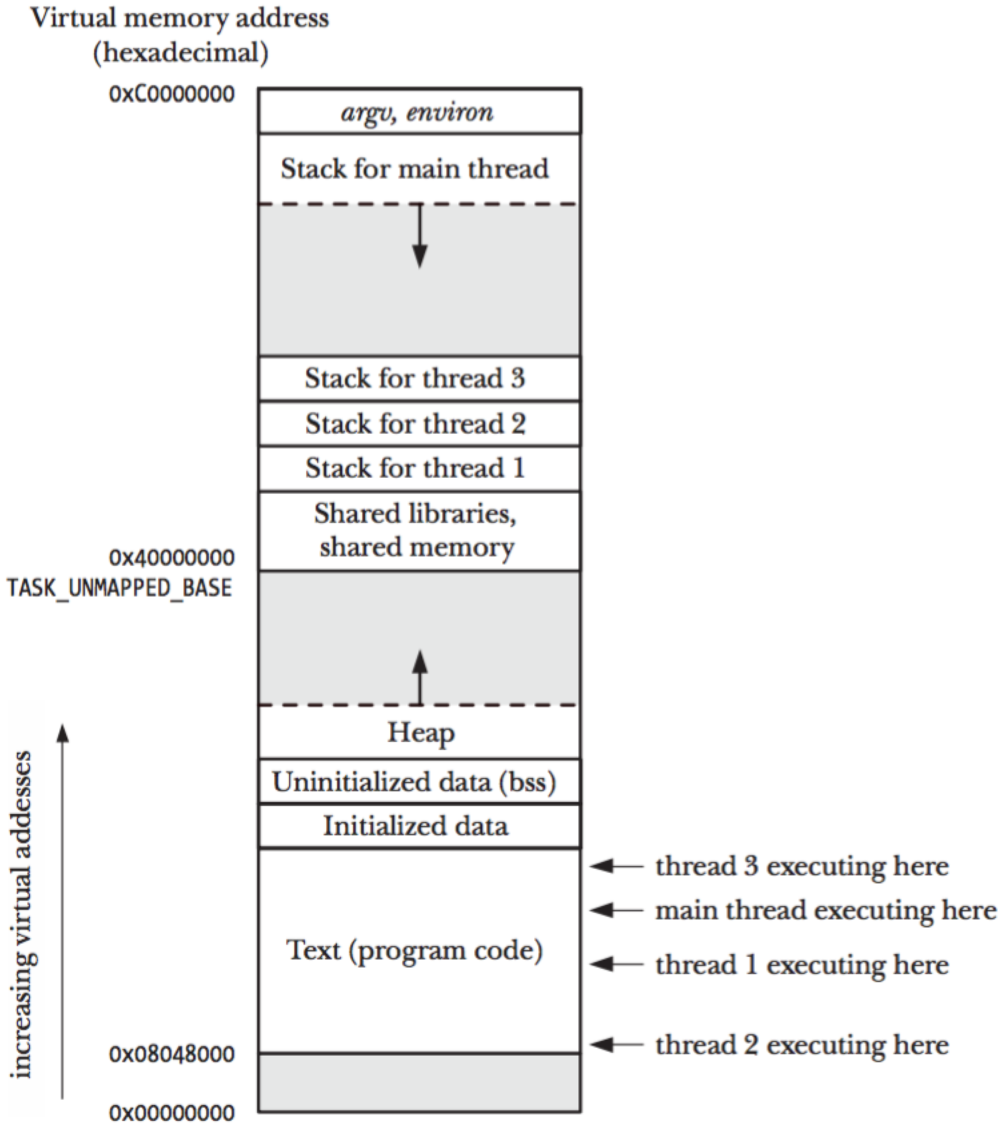

Figure 29-1: Four threads executing in a process (Linux/x86-32)

-- by Jason/cntofu.com

4. What follows is an example code which you can use to check how the threads of a multi-thread application share their memory. However, you need to add extra code to check whether the threads of a multi-thread application share their heap areas. The heap area of a thread is created by malloc() and released by free().

```
#include <stdio.h>
#include <pthread.h>
#include <string.h>
#include <sys/syscall.h>      /* Definition of SYS_* constants */
#include <unistd.h>

extern void *func1(void *);
extern void *func2(void *);
extern int main();
//void * my_get_physical_addresses(void *);


struct data_
{
  int  id ;
  char name[16] ;
} ;
typedef struct data_ sdata ;
static __thread  sdata  tx ;  //thread local variable

int a=123;  //global variable

void hello(int pid)
{
  int b=10;   //local varialbe

  b=b+pid;
           //global variable
  printf("In thread %d \nthe value of gloable varialbe a is %d, the offset of the logical address of  a is %p, ", pid, a, 
  printf("the physical address of global variable a is %p\n", my_get_physical_addresses(&a) );
           //local variable
  printf("the value of local varialbe b is %d, the offset of the logical address of b is %p, ", b, &b);
  printf("the physical address of local variable b is %p\n", my_get_physical_addresses(&b));
           //thread local variable
  printf("the offset of the logical address of thread local varialbe tx is %p, ", &tx);
  printf("the physical address of thread local variable tx is %p\n", my_get_physical_addresses(&tx));
           //function
  printf("the offset of the logical address of function hello is %p, ", hello);
  printf("the physical address of function hello is %p\n", my_get_physical_addresses(hello));
  printf("the offset of the logical address of function func1 is %p, ", func1);
  printf("the physical address of function func1 is %p\n", my_get_physical_addresses(func1));
  printf("the offset of the logical address of function func2 is %p, ", func2);
  printf("the physical address of function func2 is %p\n", my_get_physical_addresses(func2));
  printf("the offset of the logical address of function main is %p, ", main);
  printf("the physical address of function main is %p\n", my_get_physical_addresses(main));
           //library function
  printf("the offset of the logical address of library function printf is %p, ", printf);
  printf("the physical address of library function printf is %p\n", my_get_physical_addresses(printf));
  printf("=============================================================
}

void *func1(void *arg)
{
  char *p = (char*) arg ;
  int pid  ;
  pid =  syscall( __NR_gettid );
  tx.id = pid ;
  strcpy(tx.name,p) ;
  printf("I am thread with ID %d executing func1().\n",pid);
  hello(pid);
  while(1)
  {
   //printf("(%d)(%s)\n",tx.id,tx.name) ;
   sleep(1) ;
  }
}
```

```
void *func2(void *arg)
{
 char *p = (char*) arg ;
 int pid  ;
 pid =  syscall( __NR_gettid );
 tx.id = pid ;
 strcpy(tx.name,p) ;
 printf("I am thread with ID %d executing func2().\n",pid);
 hello(pid);
 while(1)
 {
   //printf("(%d)(%s)\n",tx.id,tx.name) ;
   sleep(2) ;
 }
}
int main()
{
 pthread_t id[2];
 char p[2][16] ;
 strcpy(p[0],"Thread1") ;
 pthread_create(&id[0],NULL,func1,(void *)p[0]);
 strcpy(p[1],"Thread2") ;
 pthread_create(&id[1],NULL,func2,(void *)p[1] );


 int pid  ;
 pid =  syscall( __NR_gettid );
 tx.id = pid ;
 strcpy(tx.name,"MAIN") ;
 printf("I am main thread with ID %d.\n", pid);
 hello(pid);
 while(1)
 {
  //printf("(%d)(%s)\n",tx.id,tx.name) ;
  sleep(5) ;
 }

}
```

5. 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.

- **Project Submission:** (updated: 31st Oct.)
    - NEW The due day of report submission is 23:55 27th Nov.
    - NEW The demo will be held from 28th Nov. to 1st Dec.
    - NEW Please fill out this form to choose your demo time before 26th Nov.
    - 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