Some exercises on Threads

1. Modify the program thread3.c on slide 27 (or close to) such that it uses 8 threads rather than 4 threads. Each thread must also print the content of the variables mytid, sum and mysum after the for loop but before calling the lock synchronization primitive.

2. Design a simulation with threads of a wed browser connecting with a server (web site). The browser ask to download a file name OldFinalExam.pdf from this web site. However, the web site respond "You don't have the permission to download this file". In your design you must specify:

   (a) Which thread is the web browser and which thread is the web server

   (b) How the web browser will deliver its request to the thread simulating the server

   (c) How the thread simulating the web site will deliver its message to the thread simulating the web browser

   (d) The declaration of each function that will be executed as a thread (function name, function parameters)

   (e) How the web browser will wait for the respond from the web server

   You do not need to code this simulation. Your design is not general, only for this specific example. Your design focus on how the two threads communicate with each other.

3. Write the pseudo-code of a parallel matrix multiplication algorithm $A = B \times C$ using threads. The dimensions of the matrices are A = $a \times c$, B = $a \times b$ and C = $b \times c$. The number of threads is a.

4. List the fields of a thread control block (TCB) and explain the purpose of each field.

5. Explain why each thread control block has entries (fields) to store the contain of its run time registers but does not have entries to store the contain of the dynamically allocated variables (variables that are allocated memory on the heap section of a process).

6. Unlike static variables, threads do not share their stack with other threads. Why then the stack is not saved in the TCB when a context switch is performed (thread is de-scheduled)?

7. Can a thread write into the memory stack of another thread in a same process?

8. What is the binary representation of the character 5? What is the binary representation of the integer 5? How many bytes are needed to store the character 5 and how many bytes are needed to store the integer 5?

9. Does the multithreaded web server below exhibit task or data parallelism?

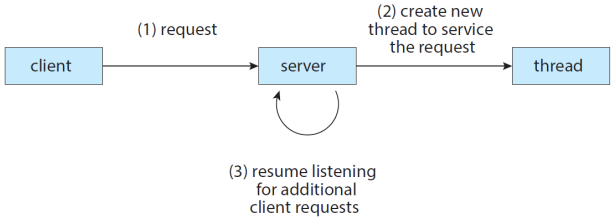

Figure 4.2 Multithreaded server architecture.

10. Is it possible to have concurrency but not parallelism? Explain

11. Consider the following code segment:

```
pid t pid;
pid = fork();
if (pid == 0) { /* child process */
fork();
thread create( . . .);
}
fork();
```

    (a) How many unique processes are created?

    (b) How many unique threads are created?

12. Consider again the program thread3.c in the slides on threads (slide 27 or close to that).

    (a) How many TLBs have been instantiated in the kernel immediately after the execution of the instruction "pthread_t threads[NTHREADS];" (line 1)

    (b) How many thread stacks there are in the process immediately after the execution of the instruction "pthread_t threads[NTHREADS];" (line 1)

    (c) How many TLBs are still instantiated just before the execution of the instruction "for (i=0;i<ARRAYSIZE;i++){" (line 2) of the main thread

    (d) How many thread stacks there are just before the execution of the instruction "for (i=0;i<ARRAYSIZE;i++){" (line 2) of the main thread

    (e) Assume a fork is executed just after the instruction "pthread_t threads[NTHREADS];" (line 1), nothing else change in the code of thread3.c. How many TLBs will be instantiated by the parent and child process all together

    (f) Assume a POSIX fork is executed just before the instruction "for (i=0; i<NTHREADS; i++)" (line 3), nothing else change in the code of thread3.c. Theorize (argue) what could happen when the loop "for (i=0; i<NTHREADS; i++)" (line 3) is executed in the child process (more likely this is implementation dependent).

    (g) Assume the piece of code below is executed just after the instruction

"printf ("Done. Sum= %e \n", sum);"

(line 4). What will be the value of sum printed by the child process. Explain your answer.

```
 pthread_mutex_destroy(&sum_mutex);
 if ((childpid = fork()) <= 0) {
        pthread_mutex_init(&sum_mutex, NULL);
        for (i=0; i<NTHREADS; i++) {
          tids[i] = i;
          pthread_create(&threads[i], &attr, do_work, (void *) &tids[i]);
        }
 for (i=0; i<NTHREADS; i++)
         pthread_join(threads[i], NULL);
 printf ("Done. Sum= %e \n", sum);
}
```

(h) The previous example is not practical as the child process repeat the same as the parent. Give at least one good reason for forking processes which create threads rather than having all the threads in a same process.

13. Assume the image of a process is swapped out from main memory to the external memory. The process has 3 active threads, two in the ready queue and one in the running state. What will happen to those 3 threads?