Cameron Hartling
007735059

CSE 4600 Operating Systems

Homework Assignment 3: Threads

Part I Long Answer Questions. Please provide your own responses to the questions, using original wording, and refrain from directly copying information from the lecture notes.

Question 1. (2 points) Please describe the difference between processes and threads in detail.

A process can have multiple threads. Threads belong to processes and can share code, data, and files between other threads that are a part of the process.

Question 2. (2 points) Please provide detailed explanations of the advantages of employing threads.

Threads are used to keep parts of the process running when other parts are blocked as well as being scalable for multicore architectures.

Question 3. (2 points) Please explain the distinction between concurrency and parallelism.

Concurrency is when a singler processor or core rapidly switches between processes or threads to progress them evenly. Parallelism performs multiple processes at once. This is only possible with multicore architectures.

Question 4. (2 points) Please provide comprehensive explanations for kernel-level threads and user-level threads.

Kernel-level threads are fully supported and managed by the kernel. All TCBs and PDBs are controlled by the kernel as well. User-level threads are managed in the user space. These threads need a private TCB and the switch between these threads are much faster than kernel-level threads.

Question 5. (2 points) Please provide detailed explanations for three multithreading models.

The many-to-one multithreading model is the mapping of many user-level threads to a single kernel thread. The thread management is done in the user space and is very efficient. The one-to-one multithreading model is the mapping of each user-level thread to a kernel thread. The many-to-many is the mapping of several user-level threads to many kernel threads. This model is more flexible but is harder to implement.

Question 6. (2 points) Please investigate and elucidate the function calls pthread_attr_init(), pthread_create(), pthread_join(), pthread_exit().

- pthread_attr_init(): This function initializes the thread attributes object. Once initialized the function pthread_create() can be used on the object.
- pthread_create(): This function will create a new thread. The new thread will start by invoking start_ routine();
- pthread_join(): This function will wait for the specified thread to terminate and then will copy the exit status of the terminated thread.
- pthread_exit(): This function will end the calling thread and return a value by retval.

Part II Programming Questions:

Question 7. (2 points)

```
//Name the file as hw3_q7.c, compile the file using gcc hw3_q7.c
-o q7 -lpthread
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define NUMBER_OF_THREADS 5

void *print_hello(void* tid)
{
      printf("Hello World. Greetings from thread %d\n", tid);
      pthread_exit(NULL);
}
```

```
int main(int argc, char *argv[])
{
     /* The main program creates 10 threads and then exits. */
     pthread_t threads[NUMBER_OF_THREADS];
     int status, i;
     for(i=0; i < NUMBER_OF_THREADS; i++)
     {
          printf("Main here. Creating thread %d\n", i);
          status = pthread_create(&threads[i], NULL,
print_hello, (void*)(unsigned long long)i);
          if (status != 0)
          {
               printf("Oops. pthread create returned error code
%d\n", status);
               exit(-1);
          }
          else{
               printf("The threads id is: %d\n",pthread_self());
          }
      }
}
```

```
[007735059@csusb.edu@jb358-5 hw3]$ ./q7
Main here. Creating thread 0
The threads id is: 1838962176
Main here. Creating thread 1
Hello World. Greetings from thread 0
The threads id is: 1838962176
Main here. Creating thread 2
Hello World. Greetings from thread 1
The threads id is: 1838962176
Main here. Creating thread 3
Hello World. Greetings from thread 2
The threads id is: 1838962176
Main here. Creating thread 4
Hello World. Greetings from thread 3
The threads id is: 1838962176
```

Question 8. (3 points)

```
//Name the file as hw3_q8.c, compile the file using gcc hw3_q8.c
-o q8 -lpthread
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
```

```
int sum; /* this data is shared by the thread(s) */

void *runner(void *param); /* the thread */

int main(int argc, char *argv[])
{
pthread_t tid; /* the thread identifier */
pthread_attr_t attr; /* set of attributes for the thread */

if (argc != 2) {
    fprintf(stderr,"usage: ./a.out <integer value>\n");
    return -1; /*exit(1);*/
}

if (atoi(argv[1]) < 0) {
    fprintf(stderr,"Argument %d must be non-negative\n",
atoi(argv[1]));
    return -1; /*exit(1);*/
}
/* get the default attributes */
pthread_attr_init(&attr);

/* create the thread */
pthread_create(&tid,&attr,runner,argv[1]);

/* print the thread id */
printf("Thread id: %d\n", pthread_self());

/* now wait for the thread to exit */
pthread_join(tid,NULL);

printf("sum = %d\n",sum);
}
/**
 * The thread will begin control in this function
 */
void *runner(void *param)
{
     int i, upper = atoi(param);
     sum = 0;
     if (upper > 0) {
         for (i = 1; i <= upper; i++)
             sum += i;
     }
     pthread_exit(0);
}
```

```
[007735059@csusb.edu@jb358-5 hw3]$ ./q8
usage: ./a.out <integer value>
```

Question 9. (3 points)

```
[007735059@csusb.edu@jb358-5 hw3]$ ./q9
[main process]:->process id:245270 threadid:139734139278848
[new thread]:->process id:245270 threadid:139734135338560
```

The thread IDs if the main process and the new thread are different. The first line is printed by the main process as it is printed prior to the creation of the new thread.