Computer Science & Information Systems

**Lab Sheet- Multithreading**

1. **Objective:**

At the end of this lab session student will be able to:

* Write a source code that creates threads
* Use the simulator to compile and run the code
* Display the list of processes/threads and the tree of processes showing the parent/child process relationship
* Demonstrate that threads share their parent’s data areas

1. **Theory:**

A thread is the smallest unit of execution that can be independently scheduled by the OS. Most modern operating systems support threads. A process can have multiple threads or sequences of executions. A thread is also known as **light weight** process and is a dynamic component of a process. This is because thread creation and management is not as demanding and time‐consuming as processes. Also, threads normally share their parents’ (and grandparents’) global data spaces and other resources. Several threads are usually created within a single process. These threads share part of the program code and the resources that have been allocated to the process.

Most modern operating systems support **multithreading**, a feature that allows multiple threads to execute within a single process. Multithreading enables a programmer to define several tasks in a single process and each task is assigned to a thread.

1. **Steps to perform :**
   1. Enter the source code in the compiler using compiler’s PROGRAM SOURCE[INPUT] editor window and compile the source code to generate the executable Binary Code
   2. Load the executable code in memory.
   3. Create a single process from program in the OS simulator and note down process id of the process that is created.
   4. Select Round Robin scheduling algorithm, set RR Time slice to 3 ticks and the simulation speed is set at medium.
   5. Make the console window visible by clicking on the **INPUT/OUTPUT…** button. Also make sure the console window stays on top by checking the **Stay on top** check box.
   6. Start executing the process and observe console window display.
   7. Click on the Views tab and click on the VIEW PROCESS LIST.. button. Observe the contents of the window while process is getting executed.
   8. In the Process list window press the VIEW TREE.. button. Observe the contents of the window while process is getting executed.
2. **Examples :**

**Example 1 :** Consider the following source code “Thread\_EX1.txt”

program ThreadTest1

sub thread1 as thread

for i = 1 to 10

writeln("In thread1: ", i)

next

end sub

sub thread2 as thread

for i = 1 to 10

writeln("In thread1: ", i)

next

end sub

call thread1

call thread2

writeln("In main")

do

loop

end

Compile the above source and load it in main memory. Create a single process, choose RR scheduling algorithm with time quantum of 3 ticks. Run the Process.

Answer the following questions:

1. How many processes and how many threads are created?
2. Identify the name of the processes and threads.
3. What is the PID and PPID of the processes and threads created?
4. Represent the parent and child relationship using tree representation.

**Example 2 :** This example demonstrates that threads share their parent’s data areas.

program ThreadTestver3

var s1 string (6)

var s2 string (6)

sub thread1 as thread

s1 = “hello1”

for i = 1 to 10

writeln("In thread1: ", i)

next

end sub

sub thread2 as thread

call thread1

s2 = “hello2”

for j = 10 to 20

writeln("In thread2 :", j)

next

end sub

call thread2

writeln("In main")

wait

writeln (s1)

writeln (s2)

end

Perform the following :

1. Compile the above program. Load the program in the main memory starting at address 100. Create a process. Note down the name of the process. Choose RR scheduling algorithm with time quantum equals to 3 ticks.
2. Click on the **SYMBOL TABLE…** button in the compiler window. In the displayed window, observe the information on variables **s1** and **s2**. Where variables s1, s2, i and j are stored?
3. Start the process. List the names of the threads created.
4. Press VIEW PROCESS STATES.. button in Views panel and observe how process and two threads are executed by the processor.
5. When the main process is in Waiting state, first thread is running and second thread is in ready state, click on SUSPEND button on OS Control panel. Observe the memory allotted for main process, and two threads.
6. Now resume execution by pressing the RESUME button on OS Control panel.

**Exercise:** Consider the following source code

program ThreadTestver2

sub thread1 as thread

for i = 1 to 10

writeln("In thread1: ", i)

next

end sub

sub thread2 as thread

call thread1

for i = 11 to 20

writeln("In thread2 :", i)

next

end sub

call thread2

writeln("In main")

wait

end

Compile the above source and load it in main memory. Create a single process, choose RR scheduling algorithm with time quantum of 3 ticks. Run the Process.

Answer the following questions:

1. How many processes and how many threads are created?
2. Identify the name of the processes and threads.
3. What is the PID and PPID of the processes and threads created?
4. Represent the parent and child relationship using tree representation.