**6/3/23 Study of Pthreads**

**19I412 Pre-Lab Report**

**Anand Narayanan N**

**21I205**

**PThreads :**

Pthreads, or POSIX threads, is a standard API for creating and manipulating threads in a multi-threaded application. Pthreads provide a simple and efficient way to write parallel programs that can take advantage of modern multi-core processors.

**1 Thread Variable Creation :**

To create a thread, we first need to declare a variable of type pthread\_t, which is used to hold a thread ID. We can then pass this variable as an argument to the pthread\_create() function, which creates a new thread and assigns a unique ID to it.

**Syntax :**

pthread\_t thread\_id;

* thread\_id – variable of type pthread\_t

**Code :**

pthread\_t tid;

**Thread Function**

The thread function is the function that will be executed by the new thread. This function must take a void\* argument and return a void\* result.

**Syntax :**

void \*my\_thread\_function(void \*arg)

* This function takes a void pointer as input and gives out void pointer as output(return value).

**Code :**

#include <pthread.h>

void \*my\_thread\_function(void \*arg) {

int my\_arg = \*(int\*) arg;

// do something with my\_arg

return NULL;

}

In this example, my\_thread\_function takes a pointer to an int as its argument and returns NULL.

**Thread Call**

To create a new thread and execute a thread function, we use the pthread\_create() function. The pthread\_create() function takes four arguments:

**Syntax :**

int pthread\_create(pthread\_t \*thread, const pthread\_attr\_t \*attr, void \*(\*start\_routine)(void\*), void \*arg);

* Thread\_id – pthread variable declared in Step 1
* attr – Pointer to attributes of the Thread such as Stack Size and Scheduling Policy(Can also be given NULL for no attributes)
* start\_routine – Pointer to function that will be executed by the Thread
* arg – Pointer to arguments to be passed to start\_routine

**Code :**

#include <pthread.h>

void \*my\_thread\_function(void \*arg) {

int my\_arg = \*(int\*) arg;

// do something with my\_arg

return NULL;

}

int main() {

pthread\_t my\_thread;

int my\_arg = 42;

pthread\_create(&my\_thread, NULL, my\_thread\_function, &my\_arg);

// do something while my\_thread is running

return 0;

}

In this example, we create a new thread using pthread\_create and pass it a pointer to my\_thread\_function as the thread function.

**Thread Join**

To wait for a thread to finish executing, and to join it with the main thread, we call the pthread\_join function.

**Syntax :**

pthread\_join(thread\_id, NULL);

* Thread\_id – Thread we want to join
* NULL – To retrieve the return value of the Thread function (can also be NULL to return no values)

**Code :**

pthread\_t tid;

pthread\_create(&tid, NULL, thread\_function, NULL);

pthread\_join(tid, NULL);