Operating system concepts

Threads & Concurrency-2
Slides Set #8

By Prof K R Chowdhary

JNV University

2023

Code for Creating a thread in Unix

```
/* thread.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>//Header file for sleep()
#include <pthread.h>
// A normal C function that is executed as a thread
// when its name is specified in pthread_create()
void *myThreadFun(void *vargp){
    sleep(1);
    printf("Printing from inside of a Thread\n");
    return NULL;
}
// main() in next slide
```

Code for Creating a thread in Unix....

```
/* thread.c */
/....
int main(){
  pthread_t thread_id;
  printf("Before Thread\n");
  pthread_create(&thread_id, NULL, myThreadFun, NULL);
  pthread_join(thread_id, NULL);
  printf("After Thread\n");
  exit(0);
}
```

Compiling and running a thread program

```
$ gcc thread.c
$ ./a.out
Before Thread
Printing from inside of a Thread
After Thread
```

- Questions:
  - Is this synchronous or asynchronous thread?
  - What statement is executed after after thread termination?

A thread sharing a global data

```
/* multi_th.c */
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
int sum;   /* this data is shared by the thread(s) */
void *runner(void *param); /*threads call this function*/
int main(int argc, char *argv[]) {
  pthread_t tid;
  pthread_attr_t attr; /* set of thread attributes */
  pthread_attr_init(&attr); /*get default attributes*/
  /* create the thread */
  pthread_create(&tid, &attr, runner, argv[1]);
 ...
} // next slide
```

A thread sharing a global data....

```
/* multi_th.c */
....
 /* wait for the thread to exit */
  pthread_join(tid, NULL);
  printf("sum = %d\n", sum);
return 0;
/* The thread will begin control in this function */
void *runner(void *param) {
  int i, num=atoi(param);
  for (i = 1; i <= num; i++)
    sum += i;
  pthread_exit(0);
}
```

Compiling a running the thread program

We are running the compiled program first with argument value 5 (i.e., argv[1], shown in main() of this program) and then running with argv[1]= 10. The threads sums integers up to this limit.

```
$ gcc multi_th.c
$ ./a.out 5
sum  = 15
$ ./a.out 10
sum = 55
```

- Questions:
  - What is global data here?
  - What is thread type here?
  - What are argv[1], *param?
  - What is *runner?

Code for Multiple threads creation

```
/* mthreads.c */
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define NUM_THREADS 5
void *printhello(void *threadid){
   long tid;
   tid = (long)threadid;
   printf("Hello World! Thread ID, %ld\n", tid);
   pthread_exit(NULL);
}
// main() in next...
```

Code for Multiple threads creation....

```
/* mthreads.c */
//....
int main () {
 pthread_t threads[NUM_THREADS];
 long i;
 for( i = 0; i < NUM_THREADS; i++ ) {
   printf("main() : creating thread %ld\n", i);
   pthread_create(&threads[i], NULL, printhello,
                                  (void *)i);
  }
  pthread_exit(NULL);
}
```

Running the thread code

```
$ ./a.out
main() : creating thread 0
main() : creating thread 1
Hello World! Thread ID, 0
main() : creating thread 2
Hello World! Thread ID, 1
main() : creating thread 3
Hello World! Thread ID, 2
main() : creating thread 4
Hello World! Thread ID, 3
Hello World! Thread ID, 4
```

- Questions:
  - What is (long)threadid?
  - Is there any global shared variable?