Introduction to pthreads

Dr S.Rajarajan

SASTRA

POSIX Threads

- Also known as Pthreads.
- A standard for Unix-like operating systems.
- A library that can be linked with C programs.
- Specifies an application programming interface (API) for multi-threaded programming.

What are pthreads?

- Posix 1003.1c defines a thread interface
  - pthreads
  - defines how threads should be created, managed, and destroyed
- Unix provides a pthreads library
  - API to create and manage threads
  - you don’t need to worry about the implementation details
    - this is a good thing

Creating Threads

- **Prototype:**
  - int pthread_create(pthread_t *tid, const pthread_attr_t *tattr,
    void*(*start_routine)(void *), void *arg);
    - ***tid*****:** an unsigned long integer that indicates a threads id
    - ***tattr*****:** attributes of the thread – usually NULL
    - ***start_routine*****:** the name of the function the thread starts executing
    - ***arg*****:** the argument to be passed to the start routine – only one
  - after this function gets executed, a new thread has been created and is executing the function indicated by *start_routine*

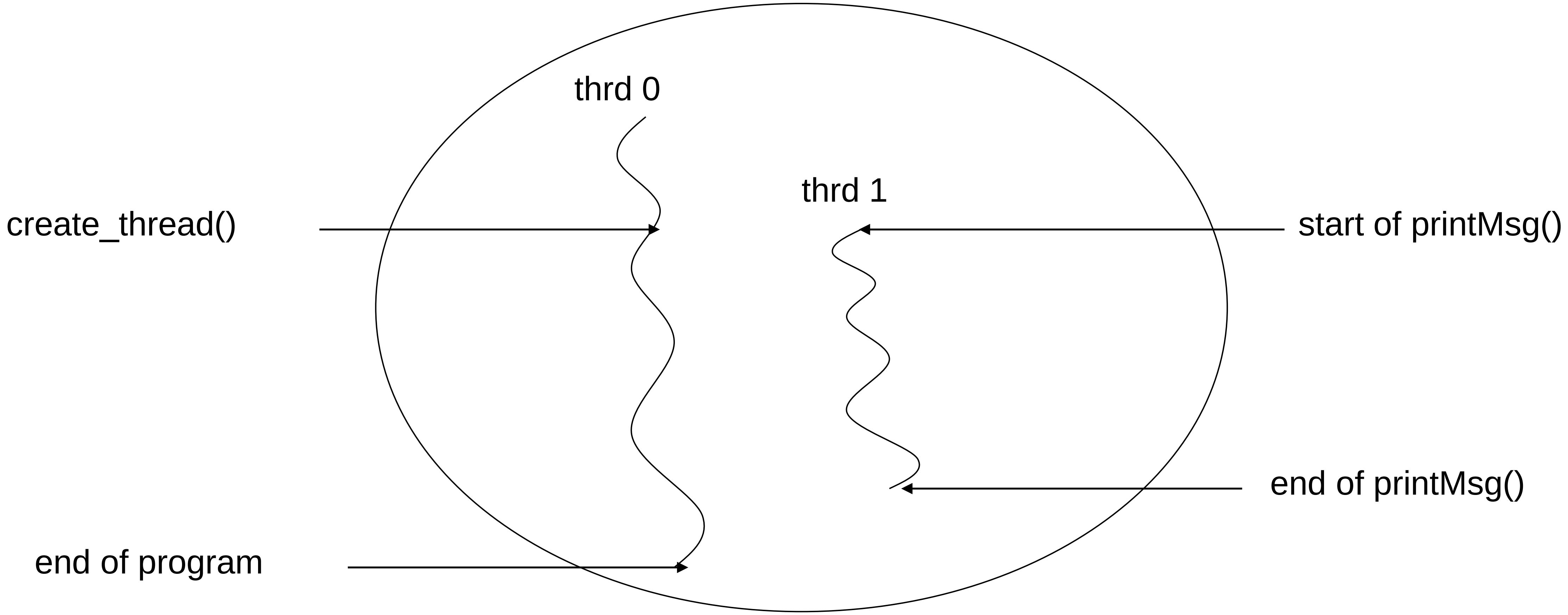
Waiting for a Thread

- **Prototype:**
  - int pthread_join(thread_t tid, void **status);
    - ***tid***: identification of the thread to wait for
    - ***status:*** the exit status of the terminating thread – can be NULL
  - We call the function pthread_join once for each thread.
  - A single call to pthread_join will wait for the thread associated with the pthread_t object to complete.
  - The thread that calls this function blocks its own execution until the thread indicated by *tid* terminates its execution
    - finishes the function it started with or
    - issues a *pthread_exit()* command – more on this in a minute

Example

```
#include <stdio.h>
#include <pthread.h>

void printMsg(char* msg) {
   printf(“%s\n”, msg);
}

int main(int argc, char** argv) {
   pthread_t thrdID;

   printf(“creating a new thread\n”);
   pthread_create(&thrdID, NULL, (void*)printMsg, argv[1]);
   printf(“created thread %d\n”. thrdID);
   pthread_join(thrdID, NULL);

   return 0;
}
```

Example



Note: thrd 0 is the function that contains *main()* – only one *main()* per program

Exiting a Thread

- pthreads **exist in user space** and are seen by the **kernel as a single process**
  - if one issues and *exit()* system call, all the threads are terminated by the OS
  - if the *main()* function exits, all of the other threads are terminated
- To have a thread exit, use *pthread_exit()*
- Prototype:
  - void pthread_exit(void *status);
    - *status:* the exit status of the thread – passed to the *status* variable in the *pthread_join()* function of a thread waiting for this one

Synchronizing Threads

- Three basic synchronization primitives
  1. mutex locks
  2. condition variables
  3. semaphores

Compiling pthread program

```
gcc program.c −lpthread
```

link in the Pthreads library