



# OPERATING SYSTEMS

## Threads and Concurrency

---

**Suresh Jamadagni**

Department of Computer Science

# OPERATING SYSTEMS

## Slides Credits for all the PPTs of this course

---



- The slides/diagrams in this course are an **adaptation**, **combination**, and **enhancement** of material from the following resources and persons:
  1. Slides of Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne - 9<sup>th</sup> edition 2013 and some slides from 10<sup>th</sup> edition 2018
  2. Some conceptual text and diagram from Operating Systems - Internals and Design Principles, William Stallings, 9<sup>th</sup> edition 2018
  3. Some presentation transcripts from A. Frank – P. Weisberg
  4. Some conceptual text from Operating Systems: Three Easy Pieces, Remzi Arpaci-Dusseau, Andrea Arpaci Dusseau

# OPERATING SYSTEMS

---

## Pthreads and Windows Threads

**Suresh Jamadagni**

Department of Computer Science

- Provides the programmer with an API for creating and managing threads.
- Two primary ways of implementing a thread library.
  - Provide a library entirely in user space with no kernel support.
    - All code and data structures for the library exist in user space.
  - Implement a kernel-level library
    - Code and data structures for the library exist in kernel space.
    - Invoking a function directly by the operating system.

- Three main thread libraries are in use today:
  - (1) POSIX Pthreads,
  - (2) Win32,
  - (3) Java.
- Pthreads, the threads extension of the POSIX standard, may be provided as either a user- or kernel-level library.
- The Win32 thread library is a kernel-level library available on Windows systems.
- The Java thread API allows threads to be created and managed directly in Java programs.

# OPERATING SYSTEMS

## Thread Libraries: Pthreads

---



- May be provided either as user-level or kernel-level
- A POSIX standard (IEEE 1003.1c) API for thread creation and synchronization
- ***Specification***, not ***implementation***
- API specifies behavior of the thread library, implementation is up to development of the library
- Common in UNIX operating systems (Solaris, Linux, Mac OS X)

```
#include <pthread.h>
#include <stdio.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; /* the thread identifier */
    pthread_attr_t attr; /* set of thread attributes */

    if (argc != 2) {
        fprintf(stderr, "usage: a.out <integer value>\n");
        return -1;
    }
    if (atoi(argv[1]) < 0) {
        fprintf(stderr, "%d must be >= 0\n", atoi(argv[1]));
        return -1;
    }
}
```

```
/* get the default attributes */
pthread_attr_init(&attr);
/* create the thread */
pthread_create(&tid,&attr,runner,argv[1]);
/* 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;

    for (i = 1; i <= upper; i++)
        sum += i;

    pthread_exit(0);
}
```



# OPERATING SYSTEMS

## Pthreads Code for Joining 10 Threads

---



```
#define NUM_THREADS 10

/* an array of threads to be joined upon */
pthread_t workers[NUM_THREADS];

for (int i = 0; i < NUM_THREADS; i++)
    pthread_join(workers[i], NULL);
```

- ❑ Windows implements the Win32 API as its primary API.
- ❑ A Windows application runs as a separate process, and each process may contain one or more threads.
- ❑ Windows uses the one-to-one mapping.
- ❑ Windows also provides support for a **fiber** library, which provides the functionality of the many-to-many model

- ❑ The general components of a thread include:
  - A thread ID uniquely identifying the thread
  - A register set representing the status of the processor.
  - A user stack, employed when the thread is running in user mode, and a kernel stack, employed when the thread is running in kernel mode
  - A private storage area various run-time libraries and dynamic link libraries (DLLs)

- ❑ Threads are created in the Win32 API using the `CreateThread()` function the Win32 API.
- ❑ Using the `WaitForSingleObject()` function, which causes the creating thread to block until the child thread has exited

```
#include <windows.h>
#include <stdio.h>
DWORD Sum; /* data is shared by the thread(s) */

/* the thread runs in this separate function */
DWORD WINAPI Summation(LPVOID Param)
{
    DWORD Upper = *(DWORD*)Param;
    for (DWORD i = 0; i <= Upper; i++)
        Sum += i;
    return 0;
}

int main(int argc, char *argv[])
{
    DWORD ThreadId;
    HANDLE ThreadHandle;
    int Param;

    if (argc != 2) {
        fprintf(stderr, "An integer parameter is required\n");
        return -1;
    }
    Param = atoi(argv[1]);
    if (Param < 0) {
        fprintf(stderr, "An integer >= 0 is required\n");
        return -1;
    }
}
```

```
/* create the thread */
ThreadHandle = CreateThread(
    NULL, /* default security attributes */
    0, /* default stack size */
    Summation, /* thread function */
    &Param, /* parameter to thread function */
    0, /* default creation flags */
    &ThreadId); /* returns the thread identifier */

if (ThreadHandle != NULL) {
    /* now wait for the thread to finish */
    WaitForSingleObject(ThreadHandle, INFINITE);

    /* close the thread handle */
    CloseHandle(ThreadHandle);

    printf("sum = %d\n", Sum);
}
}
```



**THANK YOU**

---

**Suresh Jamadagni**

Department of Computer Science Engineering

**[sureshjamadagni@pes.edu](mailto:sureshjamadagni@pes.edu)**