



# Programming with POSIX\* Threads

Based on slides from Intel Software College

and

*Multi-Core Programming –*

*increasing performance through software multi-threading*

by Shameem Akhter and Jason Roberts



## Objectives

Explore Pthreads “core” functions to create and synchronize threads



2

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## What is Pthreads?

POSIX.1c standard

C language interface

Threads exist within same process

All threads are peers

- No explicit parent-child model
- Exception: "main thread" holds process information



3

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## pthread\_create

```
int pthread_create(tid, attr, function, arg);
```

**pthread\_t \*tid**

- handle of created thread

**const pthread\_attr\_t \*attr**

- attributes of thread to be created

**void \*(\*function)(void \*)**

- function to be mapped to thread

**void \*arg**

- single argument to function



4

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## pthread\_create Explained

Spawn a thread running the function

Thread handle returned via `pthread_t` structure

- Specify `NULL` to use default attributes

Single argument sent to function

- If no arguments to function, specify `NULL`

Check error codes!

**EAGAIN** - insufficient resources to create thread  
**EINVAL** - invalid attribute



5

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## Example: Thread Creation

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

void *hello (void * arg) {
    printf("Hello Thread\n");
}

main() {
    pthread_t tid;

    pthread_create(&tid, NULL, hello, NULL);
}
```

**What Happens?**



6

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## Waiting for a Thread

```
int pthread_join(tid, val_ptr);
```

`pthread_t tid`

- handle of *joinable* thread

`void **val_ptr`

- exit value returned by joined thread



7

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## pthread\_join Explained

Calling thread waits for thread with handle `tid` to terminate

- Only one thread can be joined
- Thread must be *joinable*

Exit value is returned from joined thread

- Type returned is (`void *`)
- Use `NULL` if no return value expected

```
ESRCH - thread (pthread_t) not found  
EINVAL - thread (pthread_t) not joinable
```



8

Copyright © 2006, Intel Corporation. All rights reserved.

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

Programming with POSIX\* Threads



## Thread States

Pthreads threads have two states

- *joinable* and *detached*

Threads are joinable by default

- Resources are kept until `pthread_join`
- Can be reset with attributes or API call

Detached threads cannot be joined

- Resources can be reclaimed at termination
- Cannot reset to be *joinable*



9

Programming with POSIX\* Threads

Copyright © 2006, Intel Corporation. All rights reserved.  
Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.



## Example: Multiple Threads

```
#include <stdio.h>
#include <pthread.h>
#define NUM_THREADS 4

void *hello (void *arg) {
    printf("Hello Thread\n");
}

main() {
    pthread_t tid[NUM_THREADS];
    for (int i = 0; i < NUM_THREADS; i++)
        pthread_create(&tid[i], NULL, hello, NULL);

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



10

Programming with POSIX\* Threads

Copyright © 2006, Intel Corporation. All rights reserved.  
Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.



## What's Wrong?

What is printed for myNum?

```
void *threadFunc(void *pArg) {
    int* p = (int*)pArg;
    int myNum = *p;
    printf( "Thread number %d\n", myNum);
}
. . .
// from main():
for (int i = 0; i < numThreads; i++) {
    pthread_create(&tid[i], NULL, threadFunc, &i);
}
```



11

Programming with POSIX\* Threads



Copyright © 2006, Intel Corporation. All rights reserved.  
Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.

## Solution – “Local” Storage

```
void *threadFunc(void *pArg)
{
    int myNum = *((int*)pArg);
    printf( "Thread number %d\n", myNum);
}
. . .

// from main():
for (int i = 0; i < numThreads; i++) {
    tNum[i] = i;
    pthread_create(&tid[i], NULL, threadFunc, &tNum[i]);
}
```



12

Programming with POSIX\* Threads



Copyright © 2006, Intel Corporation. All rights reserved.  
Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States or other countries. \*Other brands and names are the property of their respective owners.