POSIX Threads programming (pthread)

Multicore Programming

Introduction

- What is Pthread?
- Pthread API
- Example

What is Pthread?

- 병렬적으로 작동하는 소프트웨어의 작성을 위해서 제공되는 표준 API
- 모든 UNIX 계열 POSIX 시스템에서 일반적으로 이용되는 라이브러리
- C 프로그래밍 언어에서 사용할 수 있는 함수들의 모음으로 제공

Pthread API

- pthread_create
- pthread_join
- pthread_exit
- pthread_self
- more APIs, but not today

Pthread API – pthread_create

```
int pthread_create(pthread_t *thread,
                   const pthread_attr_t *attr,
                   void *(*start_routine)(void *),
                   void  *arg);
```

- 새로운 Thread를 생성한다.

@param [out]	thread	새로운 Thread의 ID
@param [in]	attr	Thread의 attribute 설정할 때 사용.(e.g.,Stack size). 기본값 0.
@param[in]	start_routine	실행할 Thread function
@param[in]	arg	start_routine으로 전달할 argument
@return		Thread 생성이 성공하면 0

Pthread API – pthread_join

```
int pthread_join(pthread_t thread,
                 void **ret_val);
```

- 특정 Thread의 종료를 기다린다. 이미 종료된 Thread의 경우 즉시 return.

@param [in]	thread	종료를 기다릴 Thread의 ID
@param [out]	ret_val	종료한 Thread의 return value
@return		Thread join이 성공하면 0

Pthread API – pthread_exit

```
void pthread_exit(void *ret_val);
```

- 호출한 Thread를 종료한다. Thread start function에서의 return call은 pthread_exit의 암시적 호출이다.

@param [in] ret_val Thread의 return value. 다른 thread가 pthread_join을 통해 이 값을 받아갈 수 있다.

Pthread API – pthread_self

```
pthread_t pthread_self(void);
```

- 호출한 Thread의 ID를 반환한다.

@return 호출한 Thread의 ID.

Example

< prac_pthread.cpp >

```
 1 #include <stdio.h>
 2 #include <pthread.h>
 3
 4 #define NUM_THREAD      10
 5 #define NUM_INCREASE    1000000
 6
 7 int cnt_global = 0;
 8
 9 void *ThreadFunc(void *arg) {
10     long cnt_local = 0;
11
12     for (int i = 0; i < NUM_INCREASE; i++) {
13         cnt_global++;            // increase global value
14         cnt_local++;             // increase local value
15     }
16
17     return (void*)cnt_local;
18 }
19
```

Example (continue..)

```
20 int main(void) {
21     pthread_t threads[NUM_THREAD];
22
23     // create threads
24     for (int i = 0; i < NUM_THREAD; i++) {
25         if (pthread_create(&threads[i], 0, ThreadFunc, NULL) < 0) {
26             printf("pthread_create error!\n");
27             return 0;
28         }
29     }
30
31     // wait threads end
32     long ret;
33     for (int i = 0; i < NUM_THREAD; i++) {
34         pthread_join(threads[i], (void**)&ret);
35         printf("thread %d, local count: %d\n", threads[i], ret);
36     }
37
38     printf("global count: %d\n", cnt_global);
39
40     return 0;
41 }
```

Example (continue..)

< Result >

```
mrbin2002@ubuntu:~/TA_multicore/prac_pthread$ g++ prac_pthread.cpp -lpthread
mrbin2002@ubuntu:~/TA_multicore/prac_pthread$ ./a.out
thread 139638316529408, local count: 1000000
thread 139638308136704, local count: 1000000
thread 139638299744000, local count: 1000000
thread 139638291351296, local count: 1000000
thread 139638282958592, local count: 1000000
thread 139638274565888, local count: 1000000
thread 139638266173184, local count: 1000000
thread 139638257780480, local count: 1000000
thread 139638249387776, local count: 1000000
thread 139638240995072, local count: 1000000
global count: 2503957
```

Example (continue..)

< cnt_global++에 해당하는 assembly instruction >

```
23      jmp .L2
24 .L3:
25      movl    cnt_global(%rip), %eax
26      addl    $1, %eax
27      movl    %eax, cnt_global(%rip)
28      addq    $1, -8(%rbp)
29      addl    $1, -12(%rbp)
30 .L2:
31      cmpl    $999999, -12(%rbp)
32      jle .L3
```

Thank You