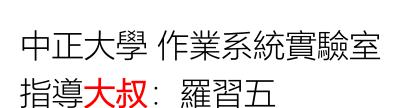
作業九: 用C11實現spinlock 並量測公平性





作業目標及負責助教

- 作業目標:
 - ●查看pthread的spinlock實作方式
 - ●使用C11將重新實現spinlock
 - 使用自行設計的spinlock, 試試看在不保障bounded waiting的情況下, 公平性怎樣?

查閱pthread的spinlock實作方式

- 可以使用google的關鍵字搜尋「gnu pthread spinlock source code」
- 利用上述關鍵字可以查到下列網頁
 - https://elixir.bootlin.com/glibc/latest/source/nptl/pthread_spin_lock.c

GNU的程式碼

```
int __pthread_spin_lock (pthread_spinlock_t *lock) {
1.
2.
      int val = 0;
3.
       if (__glibc_likely (atomic_exchange_acquire (lock, 1) == 0))
4.
        return 0:
5.
       do {
6.
         do {
         /* TODO Back-off. */
7.
8.
         atomic_spin_nop ();
9.
         val = atomic_load_relaxed (lock);
10.
        } while (val != 0);
       } while (!atomic_compare_exchange_weak_acquire (lock, &val, 1));
11.
12.
       return 0;
13.
     int pthread spin unlock (pthread spinlock t *lock) {
14.
15.
       atomic store release (lock, 0);
    return 0;
中正大學 – 羅習五
16.
17.
```

使用 C11實現的程式碼範例

```
static inline int my_spin_lock (atomic_int *lock) {
1.
2.
         int val=0;
        if (likely(atomic_exchange_explicit(lock, 1, memory_order acq rel) == 0))
4.
           return 0;
5.
         do {
           do {
6.
7.
             asm("pause");
          } while (*lock != 0);
8.
          val = 0;
9.
10.
         } while (!atomic compare exchange weak explicit(lock, &val, 1, memory order acq rel, memory order relaxed));
11.
         return 0;
12.
13.
      static inline int my_spin_unlock(atomic_int *lock) {
14.
         atomic store explicit(lock, 0, memory order release);
15.
         return 0;
16.
                                                               創作共用-姓名 標示-非商業性-相同方式分享
              中正大學 - 羅習五
```

量測每個core進入CS的次數是否公平

```
void thread(void *givenName) {
1.
2.
           int givenID = (intptr_t)givenName;
           srand((unsigned)time(NULL));
3.
4.
           unsigned int rand_seq;
5.
           cpu_set_t set; CPU_ZERO(&set); CPU_SET(givenID, &set);
6.
           sched_setaffinity(gettid(), sizeof(set), &set);
7.
           while(atomic_load_explicit(&wait, memory_order_acquire));
8.
           while(1) {
9.
             my_spin_lock(&a_lock);
10.
             atomic fetch add(&in cs, 1);
11.
             atomic_fetch_add_explicit(&count_array[givenID], 1, memory_order_relaxed);
             if (in_cs != 1) {
12.
               printf("violation: mutual exclusion\n"); exit(0);
13.
14.
15.
             atomic_fetch_add(&in_cs, -1);
16.
             my_spin_unlock(&a_lock);
17.
             int delay_size = rand_r(&rand_seq)%73;
18.
             for (int i=0; i<delay size; i++);</pre>
19.
20.
```

作業繳交

- 1. (15pt) 請對GNU的spinlock撰寫簡短的註解(大致上每一行 都要)
- 2. (15pt) 請對C11的spinlock撰寫簡短的註解(大致上每一行都要)
- 3. (20pt) 對如何量測各個核心進入CS的次數是否公平的程式碼 撰寫註解(大致上每一行都要)
- 4. (50pt) 說明你的硬體的設定,列出實驗數據,並解釋實驗數據
- 5. 上述1~3繳交程式碼, 4繳交pdf