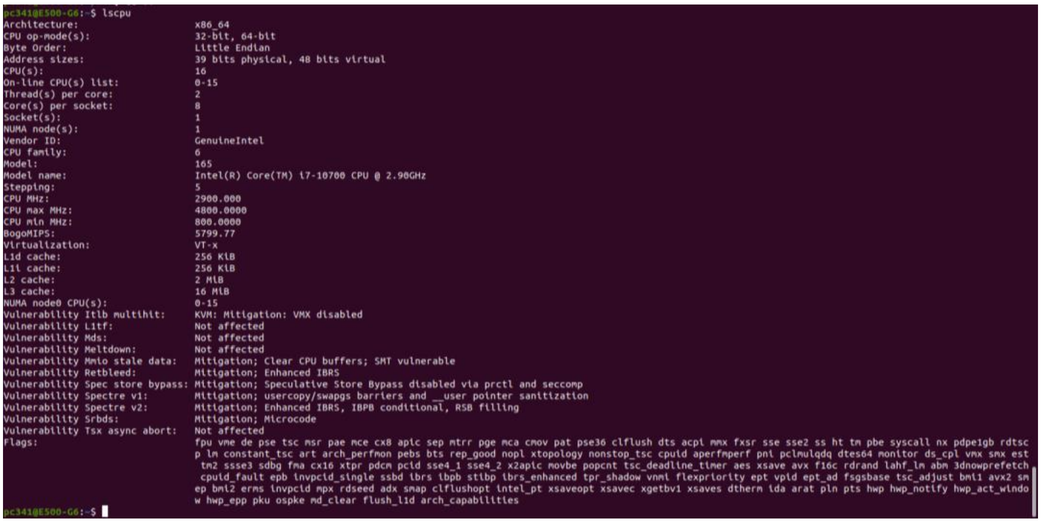
HW9 用 C11 實現 spinlock 並量測公平性

系級：資工三　學號：409410114　姓名：周 x 君

1. gnu_spinlock.c
2. c11_spinlock.c
3. fair.c
4.
   - 硬體的設定

     CPU : Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz

     因為我的電腦不太好觀察數據所以用 pc341 的電腦跑，但 wikichip 只記錄到 8 代前的資訊，沒找到這顆 (應該是 10 代)。

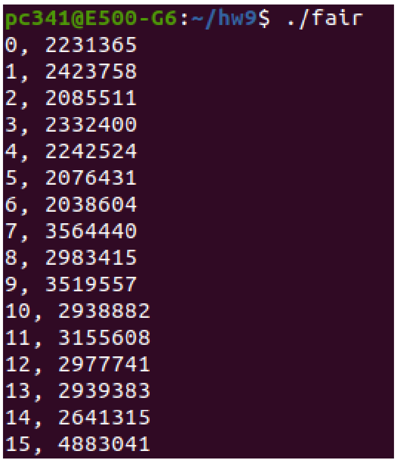


```
pc341@E500-G6:~$ lscpu
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   39 bits physical, 48 bits virtual
CPU(s):                          16
On-line CPU(s) list:             0-15
Thread(s) per core:              2
Core(s) per socket:              8
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           165
Model name:                      Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz
Stepping:                        5
CPU MHz:                         2900.000
CPU max MHz:                     4800.0000
CPU min MHz:                     800.0000
BogoMIPS:                        5799.77
Virtualization:                  VT-x
L1d cache:                       256 KiB
L1i cache:                       256 KiB
L2 cache:                        2 MiB
L3 cache:                        16 MiB
NUMA node0 CPU(s):               0-15
Vulnerability Itlb multihit:     KVM: Mitigation: VMX disabled
Vulnerability L1tf:              Not affected
Vulnerability Mds:               Not affected
Vulnerability Meltdown:          Not affected
Vulnerability Mmio stale data:   Mitigation; Clear CPU buffers; SMT vulnerable
Vulnerability Retbleed:          Mitigation; Enhanced IBRS
Vulnerability Spec store bypass: Mitigation; Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1:        Mitigation; usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:        Mitigation; Enhanced IBRS, IBPB conditional, RSB filling
Vulnerability Srbds:             Mitigation; Microcode
Vulnerability Tsx async abort:   Not affected
Flags:                           fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtsc
                                 p lm constant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est
                                  tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch
                                  cpuid_fault epb invpcid_single ssbd ibrs ibpb stibp ibrs_enhanced tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 sm
                                 ep bmi2 erms invpcid mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_windo
                                 w hwp_epp pku ospke md_clear flush_l1d arch_capabilities
pc341@E500-G6:~$ 
```

   - 實驗數據

     沒改 makefile

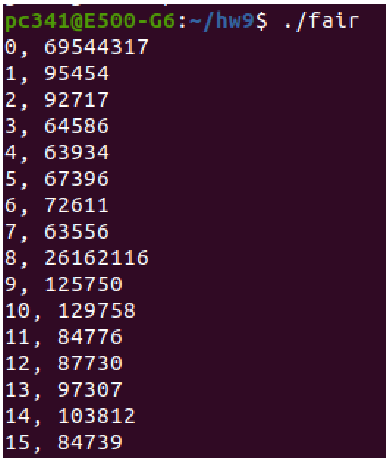
     delay_size = rand_r(&rand_seq)%73



```
pc341@E500-G6:~/hw9$ ./fair
0, 2231365
1, 2423758
2, 2085511
3, 2332400
4, 2242524
5, 2076431
6, 2038604
7, 3564440
8, 2983415
9, 3519557
10, 2938882
11, 3155608
12, 2977741
13, 2939383
14, 2641315
15, 4883041
```

Max=4883041, min=2038604, max/min≒2.4

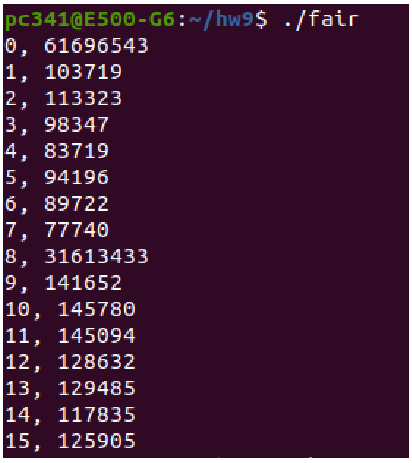
delay_size = rand_r(&rand_seq)%1000



```
pc341@E500-G6:~/hw9$ ./fair
0, 69544317
1, 95454
2, 92717
3, 64586
4, 63934
5, 67396
6, 72611
7, 63556
8, 26162116
9, 125750
10, 129758
11, 84776
12, 87730
13, 97307
14, 103812
15, 84739
```

Max=69544317, min=63556, max/min≒1094

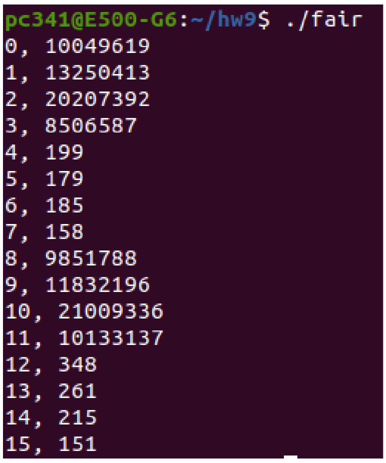
delay_size = rand_r(&rand_seq)%10000



```
pc341@E500-G6:~/hw9$ ./fair
0, 61696543
1, 103719
2, 113323
3, 98347
4, 83719
5, 94196
6, 89722
7, 77740
8, 31613433
9, 141652
10, 145780
11, 145094
12, 128632
13, 129485
14, 117835
15, 125905
```

Max=61696543, min=77740, max/min≒793.6

delay_size = rand_r(&rand_seq)%100000

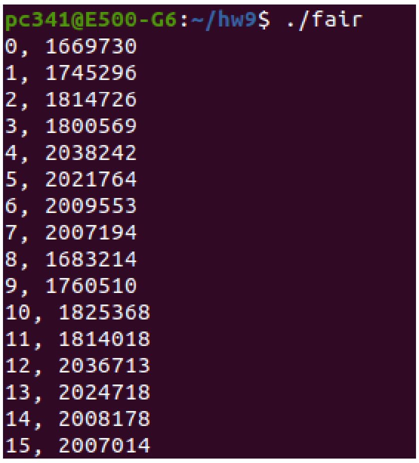


```
pc341@E500-G6:~/hw9$ ./fair
0, 10049619
1, 13250413
2, 20207392
3, 8506587
4, 199
5, 179
6, 185
7, 158
8, 9851788
9, 11832196
10, 21009336
11, 10133137
12, 348
13, 261
14, 215
15, 151
```

Max=21009336, min=151, max/min≒139134.6

Makefile 拿掉-O3

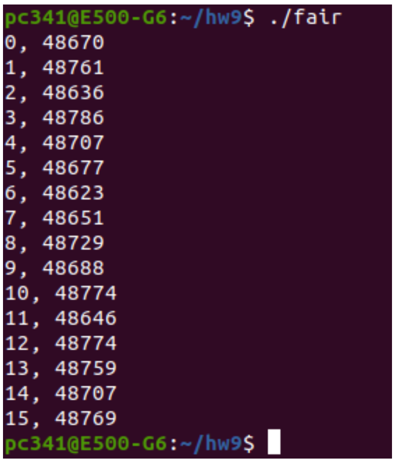
delay_size = rand_r(&rand_seq)%1000



```
pc341@E500-G6:~/hw9$ ./fair
0, 1669730
1, 1745296
2, 1814726
3, 1800569
4, 2038242
5, 2021764
6, 2009553
7, 2007194
8, 1683214
9, 1760510
10, 1825368
11, 1814018
12, 2036713
13, 2024718
14, 2008178
15, 2007014
```

Max=2038242, min=1669730, max/min≒1.2

delay_size = rand_r(&rand_seq)%100000

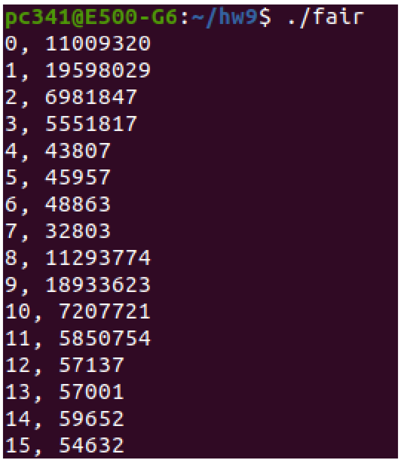


```
pc341@E500-G6:~/hw9$ ./fair
0, 48670
1, 48761
2, 48636
3, 48786
4, 48707
5, 48677
6, 48623
7, 48651
8, 48729
9, 48688
10, 48774
11, 48646
12, 48774
13, 48759
14, 48707
15, 48769
pc341@E500-G6:~/hw9$ █
```

Max=48786, min=48623, max/min≒1.00335

加入參數(march=skylake-avx512)到 makefile

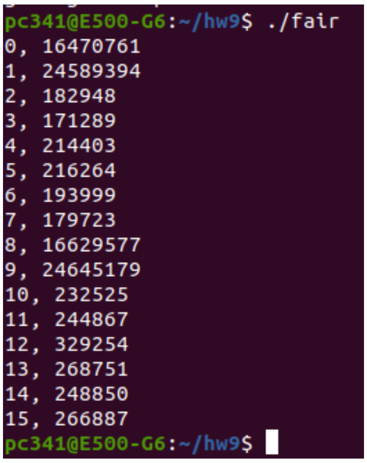
delay_size = rand_r(&rand_seq)%100000



```
pc341@E500-G6:~/hw9$ ./fair
0, 11009320
1, 19598029
2, 6981847
3, 5551817
4, 43807
5, 45957
6, 48863
7, 32803
8, 11293774
9, 18933623
10, 7207721
11, 5850754
12, 57137
13, 57001
14, 59652
15, 54632
```

Max=19598029, min=32803, max/min≒579.4

delay_size = rand_r(&rand_seq)%1000



```
pc341@E500-G6:~/hw9$ ./fair
0, 16470761
1, 24589394
2, 182948
3, 171289
4, 214403
5, 216264
6, 193999
7, 179723
8, 16629577
9, 24645179
10, 232525
11, 244867
12, 329254
13, 268751
14, 248850
15, 266887
pc341@E500-G6:~/hw9$
```

Max=24645179, min=171289, max/min≒143.8

觀察以上截圖可以發現單純只修改 delay_size，delay 增加公平性越低; 把-O3 拿掉的效果是最好的，公平性最高; 加入 march=skylake-avx512 效果次好。

Delay_size\參數	沒改	拿掉-O3	加 march=skylake-avx512
1000	1094	1.2	143.8
100000	793.6	1.00335	579.4
結論(公平性)	最不公平	最公平	中間