(2)

```
icount   100   104      ax    bx   >= >  <= <  != ==        Thread 0                Thread 1

     0     0     0       0     0   0  0  0  0  0  0
     0     0     0       0     0   0  0  0  0  0  0  1000 mov  flag, %ax
     1     0     0       0     0   1  0  1  0  0  1  1001 test $0, %ax
     2     0     0       0     0   1  0  1  0  0  1  1002 jne  .acquire
     3     1     0       0     0   1  0  1  0  0  1  1003 mov  $1, flag
     4     1     0       0     0   1  0  1  0  0  1  1004 mov  count, %ax
     5     1     0       1     0   1  0  1  0  0  1  1005 add  $1, %ax
     6     1     1       1     0   1  0  1  0  0  1  1006 mov  %ax, count
     7     0     1       1     0   1  0  1  0  0  1  1007 mov  $0, flag
     8     0     1       1    -1   1  0  1  0  0  1  1008 sub  $1, %bx
     9     0     1       1    -1   0  0  1  1  1  0  1009 test $0, %bx
    10     0     1       1    -1   0  0  1  1  1  0  1010 jgt .top
    11     0     1       1    -1   0  0  1  1  1  0  1011 halt
    12     0     1       0     0   0  0  0  0  0  0  ----- Halt;Switch -----  ----- Halt;Switch -----
    12     0     1       0     0   0  0  0  0  0  0                          1000 mov  flag, %ax
    13     0     1       0     0   1  0  1  0  0  1                          1001 test $0, %ax
    14     0     1       0     0   1  0  1  0  0  1                          1002 jne  .acquire
    15     1     1       0     0   1  0  1  0  0  1                          1003 mov  $1, flag
    16     1     1       1     0   1  0  1  0  0  1                          1004 mov  count, %ax
    17     1     1       2     0   1  0  1  0  0  1                          1005 add  $1, %ax
    18     1     2       2     0   1  0  1  0  0  1                          1006 mov  %ax, count
    19     0     2       2     0   1  0  1  0  0  1                          1007 mov  $0, flag
    20     0     2       2    -1   1  0  1  0  0  1                          1008 sub  $1, %bx
    21     0     2       2    -1   0  0  1  1  1  0                          1009 test $0, %bx
    22     0     2       2    -1   0  0  1  1  1  0                          1010 jgt .top
    23     0     2       2    -1   0  0  1  1  1  0                          1011 halt

STATS:: Instructions    24
STATS:: Emulation Rate  47.90 kinst/sec
```

```
ASSIGN VAR .var --> flag 100
ASSIGN VAR .var --> count 104
ASSIGN LABEL .main --> 1000
ASSIGN LABEL .top --> 1000
ASSIGN LABEL .acquire --> 1000
```

```
opcode mov
pc:1000 LOADING       mov  flag, %ax --> self.move_m_to_r(100, 0, 0, 1, 1)
opcode test
pc:1001 LOADING         test $0, %ax --> self.test_i_r(0, 1)
opcode jne
pc:1002 LOADING        jne  .acquire --> self.jump_notequal(1000)
opcode mov
pc:1003 LOADING        mov  $1, flag --> self.move_i_to_m(1, 100, 0, 0, 1)
opcode mov
pc:1004 LOADING      mov  count, %ax --> self.move_m_to_r(104, 0, 0, 1, 1)
opcode add
pc:1005 LOADING         add  $1, %ax --> self.add_i_to_r(1, 1)
opcode mov
pc:1006 LOADING      mov  %ax, count --> self.move_r_to_m(1, 104, 0, 0, 1)
opcode mov
pc:1007 LOADING        mov  $0, flag --> self.move_i_to_m(0, 100, 0, 0, 1)
opcode sub
pc:1008 LOADING         sub  $1, %bx --> self.sub_i_to_r(1, 2)
opcode test
pc:1009 LOADING         test $0, %bx --> self.test_i_r(0, 2)
opcode jgt
pc:1010 LOADING             jgt .top --> self.jump_greaterthan(1000)
opcode halt
pc:1011 LOADING                halt --> self.halt()
```

(3)

40	1	4	4	1	1	0	1	0	0	1	1006 mov %ax, count
41	0	4	4	1	1	0	1	0	0	1	1007 mov $0, flag
42	0	4	4	0	1	0	1	0	0	1	1008 sub $1, %bx
43	0	4	4	0	1	0	1	0	0	1	1009 test $0, %bx
44	0	4	4	0	1	0	1	0	0	1	1010 jgt .top
45	0	4	4	0	1	0	1	0	0	1	1011 halt

(4) i = 11, 15 Good outcome = 200
i = 1, 2 bad outcome = 200
other i's, 133, 167 ...

2196	0	100	100	0	1	0	1	0	0	1	------ Interrupt ------	------ Interrupt ------
2196	0	100	100	0	1	0	1	0	0	1	1009 test $0, %bx	
2197	0	100	100	0	1	0	1	0	0	1	1010 jgt .top	
2198	0	100	100	0	1	0	1	0	0	1	------ Interrupt ------	------ Interrupt ------
2198	0	100	100	0	1	0	1	0	0	1		1009 test $0, %bx
2199	0	100	100	0	1	0	1	0	0	1		1010 jgt .top
2200	0	100	100	0	1	0	1	0	0	1	------ Interrupt ------	------ Interrupt ------
2200	0	100	100	0	1	0	1	0	0	1	1011 halt	
2201	0	100	100	0	1	0	1	0	0	1	----- Halt;Switch -----	----- Halt;Switch -----
2201	0	100	100	0	1	0	1	0	0	1		1011 halt

2196	0	133	133	0	1	0	1	0	0	1	1009 test $0, %bx	
2197	0	133	133	0	1	0	1	0	0	1	1010 jgt .top	
2198	0	133	133	0	1	0	1	0	0	1	1011 halt	
2199	0	133	133	0	1	0	1	0	0	1	----- Halt;Switch -----	----- Halt;Switch -----
2199	0	133	133	0	1	0	1	0	0	1	------ Interrupt ------	------ Interrupt ------
2199	0	133	133	0	1	0	1	0	0	1		1009 test $0, %bx
2200	0	133	133	0	1	0	1	0	0	1		1010 jgt .top
2201	0	133	133	0	1	0	1	0	0	1		1011 halt

2990	1	199	200	1	1	0	1	0	0	1	1005 add $1, %ax
2991	1	200	200	1	1	0	1	0	0	1	1006 mov %ax, count
2992	0	200	200	1	1	0	1	0	0	1	1007 mov $0, flag
2993	0	200	200	0	1	0	1	0	0	1	1008 sub $1, %bx
2994	0	200	200	0	1	0	1	0	0	1	1009 test $0, %bx
2995	0	200	200	0	1	0	1	0	0	1	1010 jgt .top
2996	0	200	200	0	1	0	1	0	0	1	1011 halt

(6) Returns right value, but CPU is running more instructions

```
3000    1   200    199    0   1  0  1  0  0  1  1011 halt
3001    1   200    200    1   1  0  1  0  0  1  ----- Halt;Switch -----  ----- Halt;Switch -----
3001    0   200    200    1   1  0  1  0  0  1                          1007 mov  $0, mutex
3002    0   200    200    0   1  0  1  0  0  1                          1008 sub  $1, %bx
3003    0   200    200    0   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
3003    0   200    200    0   1  0  1  0  0  1                          1009 test $0, %bx
3004    0   200    200    0   1  0  1  0  0  1                          1010 jgt .top
3005    0   200    200    0   1  0  1  0  0  1                          1011 halt
```

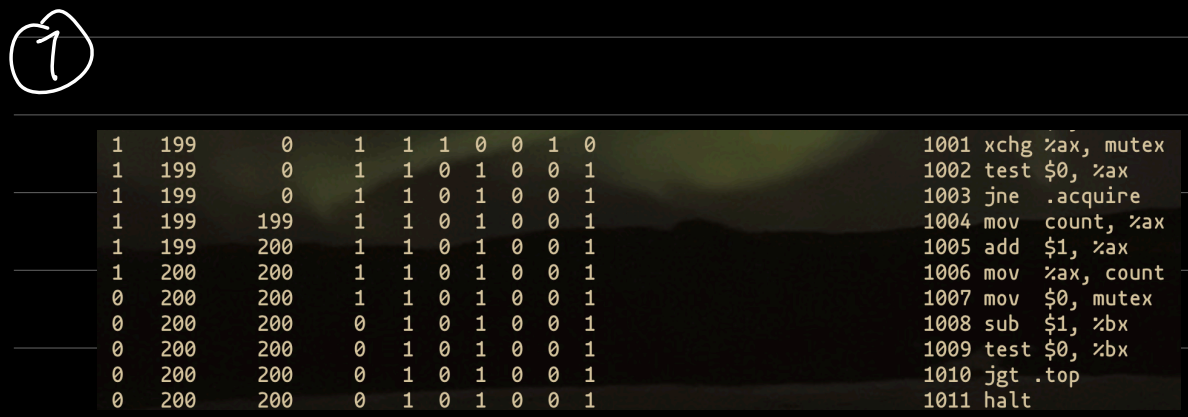
waiting to acquire lock

```
3389    0   200    200    1   1  0  1  0  0  1                          1007 mov  $0, mutex
3390    0   200    200    1   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
3390    0   200    200    0   1  0  1  0  0  1                          1008 sub  $1, %bx
3391    0   200    200    0   1  0  1  0  0  1                          1009 test $0, %bx
3392    0   200    200    0   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
3392    0   200    200    0   1  0  1  0  0  1                          1010 jgt .top
3393    0   200    200    0   1  0  1  0  0  1                          1011 halt
```

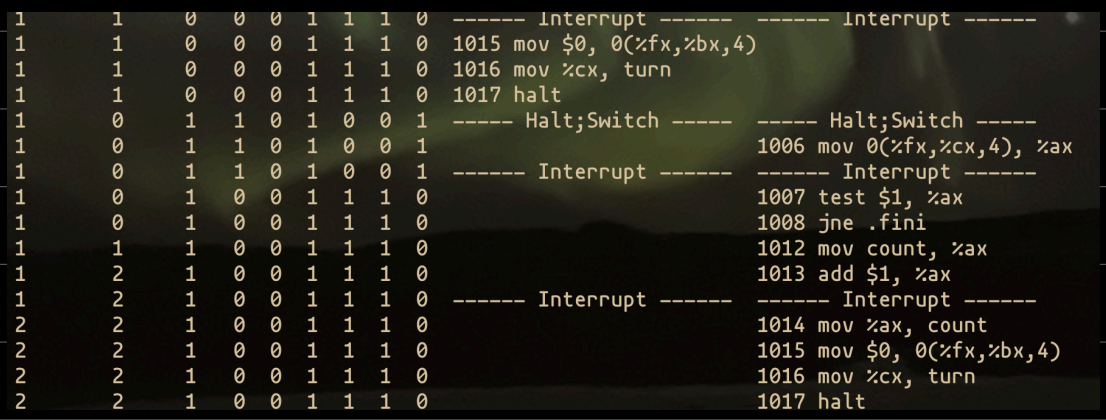
```
3532    1   200    200    1   1  0  1  0  0  1                          1006 mov  %ax, count
3533    0   200    200    1   1  0  1  0  0  1                          1007 mov  $0, mutex
3534    0   200    200    0   1  0  1  0  0  1                          1008 sub  $1, %bx
3535    0   200    200    0   1  0  1  0  0  1                          1009 test $0, %bx
3536    0   200    200    0   1  0  1  0  0  1                          1010 jgt .top
3537    0   200    200    0   1  0  1  0  0  1                          1011 halt
```

```
       Thread 0                Thread 1


1000 mov  $1, %ax
1001 xchg %ax, mutex
------ Interrupt ------  ------ Interrupt ------
                         1000 mov  $1, %ax
                         1001 xchg %ax, mutex
------ Interrupt ------  ------ Interrupt ------
1002 test $0, %ax
1003 jne  .acquire
------ Interrupt ------  ------ Interrupt ------
                         1002 test $0, %ax
                         1003 jne  .acquire
------ Interrupt ------  ------ Interrupt ------
1004 mov  count, %ax
1005 add  $1, %ax
------ Interrupt ------  ------ Interrupt ------
                         1000 mov  $1, %ax
                         1001 xchg %ax, mutex
------ Interrupt ------  ------ Interrupt ------
1006 mov  %ax, count
1007 mov  $0, mutex
------ Interrupt ------  ------ Interrupt ------
                         1002 test $0, %ax
                         1003 jne  .acquire
------ Interrupt ------  ------ Interrupt ------
1008 sub  $1, %bx
1009 test $0, %bx
------ Interrupt ------  ------ Interrupt ------
                         1000 mov  $1, %ax
                         1001 xchg %ax, mutex
------ Interrupt ------  ------ Interrupt ------
1010 jgt .top
1011 halt
----- Halt;Switch -----  ----- Halt;Switch -----
                         1002 test $0, %ax
                         1003 jne  .acquire
                         1004 mov  count, %ax
                         1005 add  $1, %ax
                         1006 mov  %ax, count
```



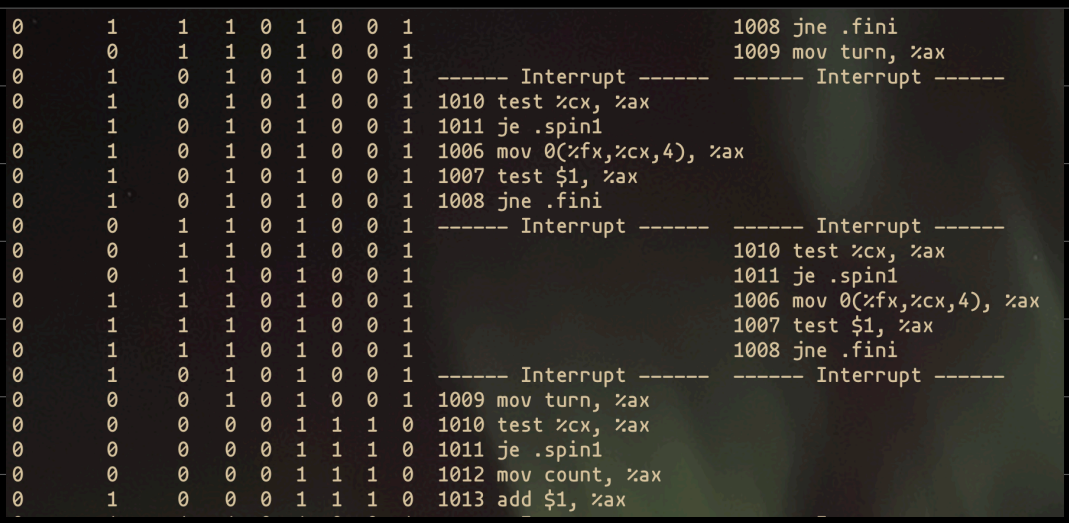
```
1   199     0     1   1  1  0  0  1  0                    1001 xchg %ax, mutex
1   199     0     1   1  0  1  0  0  1                    1002 test $0, %ax
1   199     0     1   1  0  1  0  0  1                    1003 jne  .acquire
1   199   199     1   1  0  1  0  0  1                    1004 mov  count, %ax
1   199   200     1   1  0  1  0  0  1                    1005 add  $1, %ax
1   200   200     1   1  0  1  0  0  1                    1006 mov  %ax, count
0   200   200     1   1  0  1  0  0  1                    1007 mov  $0, mutex
0   200   200     0   1  0  1  0  0  1                    1008 sub  $1, %bx
0   200   200     0   1  0  1  0  0  1                    1009 test $0, %bx
0   200   200     0   1  0  1  0  0  1                    1010 jgt .top
0   200   200     0   1  0  1  0  0  1                    1011 halt
```



```
1     1     0   0  0  1  1  1  0  ------ Interrupt ------  ------ Interrupt ------
1     1     0   0  0  1  1  1  0  1015 mov $0, 0(%fx,%bx,4)
1     1     0   0  0  1  1  1  0  1016 mov %cx, turn
1     1     0   0  0  1  1  1  0  1017 halt
1     0     1   1  0  1  0  0  1  ----- Halt;Switch -----  ----- Halt;Switch -----
1     0     1   1  0  1  0  0  1                          1006 mov 0(%fx,%cx,4), %ax
1     0     1   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
1     0     1   0  0  1  1  1  0                          1007 test $1, %ax
1     0     1   0  0  1  1  1  0                          1008 jne .fini
1     1     1   0  0  1  1  1  0                          1012 mov count, %ax
1     2     1   0  0  1  1  1  0                          1013 add $1, %ax
1     2     1   0  0  1  1  1  0  ------ Interrupt ------  ------ Interrupt ------
2     2     1   0  0  1  1  1  0                          1014 mov %ax, count
2     2     1   0  0  1  1  1  0                          1015 mov $0, 0(%fx,%bx,4)
2     2     1   0  0  1  1  1  0                          1016 mov %cx, turn
2     2     1   0  0  1  1  1  0                          1017 halt
```

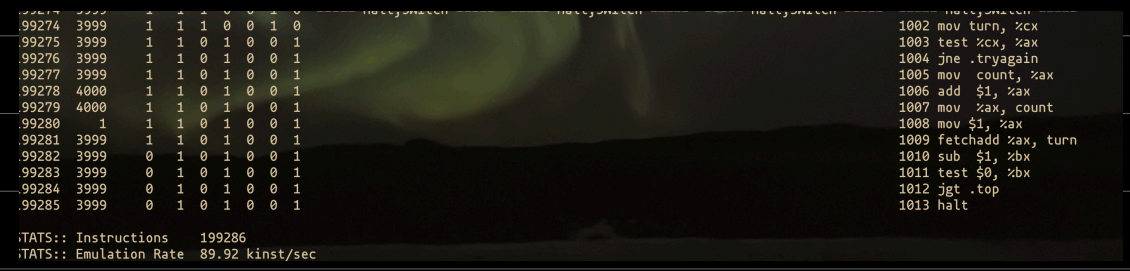
```
1     1     1   0  0  0  0  0  0  ------ Interrupt ------  ------ Interrupt ------
1     1     1   1  0  1  0  0  1                          1007 test $1, %ax
1     1     1   1  0  1  0  0  1                          1008 jne .fini
1     1     1   1  0  1  0  0  1                          1009 mov turn, %ax
1     1     1   1  1  0  0  1  0                          1010 test %cx, %ax
1     1     1   1  1  0  0  1  0                          1011 je .spin1
1     1     1   1  1  0  0  1  0                          1012 mov count, %ax
1     2     1   1  1  0  0  1  0                          1013 add $1, %ax
1     1     0   0  0  1  1  1  0  ------ Interrupt ------  ------ Interrupt ------
1     1     0   0  0  1  1  1  0  1017 halt
1     2     1   1  1  0  0  1  0  ----- Halt;Switch -----  ----- Halt;Switch -----
2     2     1   1  1  0  0  1  0                          1014 mov %ax, count
2     2     1   1  1  0  0  1  0                          1015 mov $0, 0(%fx,%bx,4)
2     2     1   1  1  0  0  1  0                          1016 mov %cx, turn
2     2     1   1  1  0  0  1  0                          1017 halt
```

⑨, ⑩



```
0       1     1   1  0  1  0  0  1                          1008 jne .fini
0       0     1   1  0  1  0  0  1                          1009 mov turn, %ax
0       1     0   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
0       1     0   1  0  1  0  0  1  1010 test %cx, %ax
0       1     0   1  0  1  0  0  1  1011 je .spin1
0       1     0   1  0  1  0  0  1  1006 mov 0(%fx,%cx,4), %ax
0       1     0   1  0  1  0  0  1  1007 test $1, %ax
0       1     0   1  0  1  0  0  1  1008 jne .fini
0       0     1   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
0       0     1   1  0  1  0  0  1                          1010 test %cx, %ax
0       0     1   1  0  1  0  0  1                          1011 je .spin1
0       1     1   1  0  1  0  0  1                          1006 mov 0(%fx,%cx,4), %ax
0       1     1   1  0  1  0  0  1                          1007 test $1, %ax
0       1     1   1  0  1  0  0  1                          1008 jne .fini
0       1     0   1  0  1  0  0  1  ------ Interrupt ------  ------ Interrupt ------
0       0     0   1  0  1  0  0  1  1009 mov turn, %ax
0       0     0   0  0  1  1  1  0  1010 test %cx, %ax
0       0     0   0  0  1  1  1  0  1011 je .spin1
0       0     0   0  0  1  1  1  0  1012 mov count, %ax
0       1     0   0  0  1  1  1  0  1013 add $1, %ax
```

```
STATS:: Instructions    99463
STATS:: Emulation Rate  91.57 kinst/sec
```



```
.99274  3999     1  1  0  0  1  0                          1002 mov turn, %cx
.99275  3999     1  1  0  1  0  0  1                       1003 test %cx, %ax
.99276  3999     1  1  0  1  0  0  1                       1004 jne .tryagain
.99277  3999     1  1  0  1  0  0  1                       1005 mov  count, %ax
.99278  4000     1  1  0  1  0  0  1                       1006 add  $1, %ax
.99279  4000     1  1  0  1  0  0  1                       1007 mov  %ax, count
.99280     1     1  1  0  1  0  0  1                       1008 mov $1, %ax
.99281  3999     1  1  0  1  0  0  1                       1009 fetchadd %ax, turn
.99282  3999     0  1  0  1  0  0  1                       1010 sub  $1, %bx
.99283  3999     0  1  0  1  0  0  1                       1011 test $0, %bx
.99284  3999     0  1  0  1  0  0  1                       1012 jgt .top
.99285  3999     0  1  0  1  0  0  1                       1013 halt

TATS:: Instructions    199286
TATS:: Emulation Rate  89.92 kinst/sec
```

```
STATS:: Instructions    299010
STATS:: Emulation Rate  85.83 kinst/sec
```

⑪, ⑫ ticket.s spends less time spinning. Adds ordering to threads, which

doesn't happen in "test-and-set.s" leading to more fairness in "ticket.s"

(13) clearly lesser instructions, especially when long processes and other threads yield instead of spinning

```
267945  6000     0   1  0  1  0  0  1                                    1011 halt

STATS:: Instructions    267946
STATS:: Emulation Rate  84.81 kinst/sec
```

```
----  ----- Halt;Switch -----
 85984  5998     0   1  0  1  0  0  1                                    1012 jgt .top
 85985  5998     0   1  0  1  0  0  1                                    1013 halt

STATS:: Instructions    85986
STATS:: Emulation Rate  80.73 kinst/sec
```

(14) test-and-test-and-set.s, has a read first before test-and-set hence potentially saving on avoiding to run more expensive atomic operations like "xchng".