4. Other Hardware Primitives

Compare-and-swap:

```
int CompareAndSwap(int *ptr, int expected, int new);
```

returns the old value and simultaneously updates to the new value if old == expected.

Fetch-and-add:

```
int FetchAndAdd(int *ptr);
```

returns the old value and simultaneously adds 1 and stores the new value.

Can be used to implement ticket lock and ensure fairness.

```
1  typedef struct __lock_t {
2      int ticket;
3      int turn;
4  } lock_t;
5
6  void lock_init(lock_t *lock) {
7      lock->ticket = 0;
8      lock->turn   = 0;
9  }
10
11 void lock(lock_t *lock) {
12     int myturn = FetchAndAdd(&lock->ticket);
13     while (lock->turn != myturn)
14         ; // spin
15  }
16
17 void unlock(lock_t *lock) {
18     lock->turn = lock->turn + 1;
19 }
```

Figure 28.7: **Ticket Locks**