Synchronization and Ordering

Consume-release semantics

- Consume-release semantics is the acquire-release semantics without ordering guarantees.

- Has a legendary reputation
  - Very complicated
  - `std::memory_order_consume` is by the compiler mapped to `std::memory_order_acquire`.
  - No compiler implements it (a short exception was *GCC*)

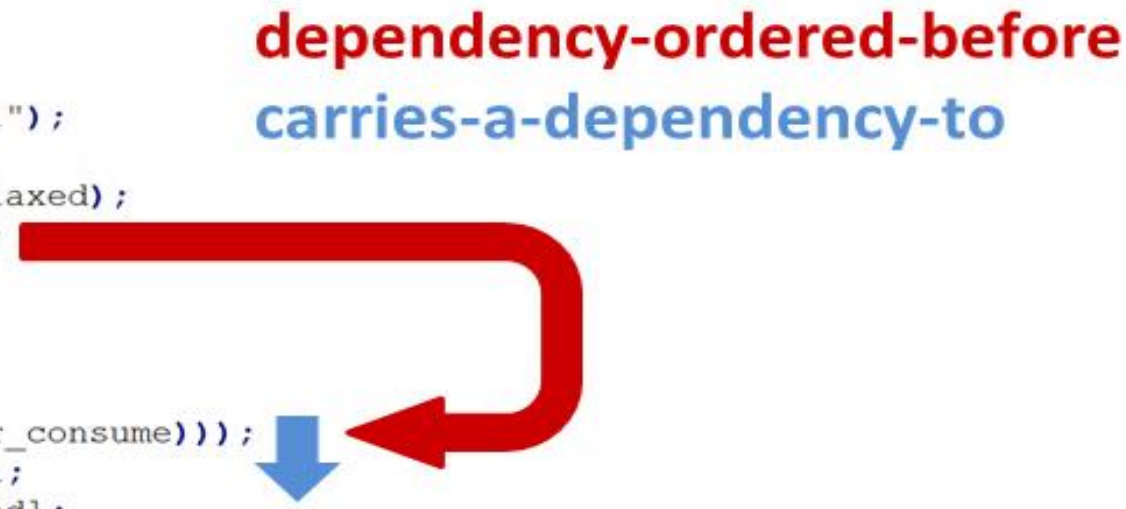
- Deals with data-dependencies
  - In a thread: *carries-a-dependency-to*
  - Between threads: *dependency-ordered-before*

Synchronization and Ordering

```
std::atomic<std::string*> ptr;
int data;
std::atomic<int> atoData;

void producer(){
    std::string* p  = new std::string("C++11");
    data = 2011;
    atoData.store(2014,std::memory_order_relaxed);
    ptr.store(p, std::memory_order_release);
}

void consumer(){
    std::string* p2;
    while (!(p2 = ptr.load(std::memory_order_consume)));
    std::cout << "*p2: " << *p2 << std::endl;
    std::cout << "data: " << data << std::endl;
    std::cout << "atoData: " << atoData.load(std::memory_order_relaxed) << std::endl;
}
```



acquireConsume.cpp