Задание №6. Модель памяти

Рубаненко Евгений

Апрель 2017

1 Lazy Value

Корректная реализация ленивой инициализации:

```
#pragma once

#include <atomic>
#include <mutex>
#include <functional>

template <typename T>
class LazyValue {
    using Factory = std::function<T*()>;
public:
    explicit LazyValue(Factory create)
        : create_(create) {
    }

    T& Get() {
        // double checked locking pattern
        T* curr_ptr =                                              (1)
            ptr_to_value_.load(std::memory_order_acquire);
        if (curr_ptr == nullptr) {
            std::lock_guard<std::mutex> guard(mutex_);
            curr_ptr =                                             (2)
                ptr_to_value_.load(std::memory_order_relaxed);
            if (curr_ptr == nullptr) {
                curr_ptr = create_();
                ptr_to_value_.store(curr_ptr,                      (3)
                    std::memory_order_release);
            }
        }
        return *curr_ptr;
    }

    ~LazyValue() {
        if (ptr_to_value_.load() != nullptr) {
            delete ptr_to_value_;
        }
    }

private:
    Factory create_;
    std::mutex mutex_;
    std::atomic<T*> ptr_to_value_{nullptr};
};
```

Неатомарные чтения и записи

В данной программе нам хотелось бы, чтобы *load* знал о *store*. В связи с этим в (1) и (3) мы используем $std :: memory\_order\_acquire$ и $std :: memory\_order\_release$ соответсвенно. Но вот в (2) мы используем $std :: memory\_order\_relaxed$. Нам этого хватает, так как второе чтение может испортить только *store*, но так как мы под мьютексом - то проблема не возникает.

Happens-before

Как и раньше, $std :: memory\_order\_acquire$ и $std :: memory\_order\_release$ строят стрелку $Synchronized - with$. Остальные стрелки возникают в силу $Program - order$. По транзитивному замыканию получаем отношение $Happens - before$.

Гарантии упорядочивания

Использовать только $std :: memory\_order\_relaxed$ нельзя, так как тогда первый *load* может не увидеть результат третьего *store* - тогда программа будет работать некорректно.