Средства симуляции ЦП и ОС и изучение поведения программ

Лекция №8

Державин Андрей
Шурыгин Антон

- **Квиз**
- Потактовые модели
- Общая схема симуляции
- Реализации потактовых моделей

Код викторины:

00725045

- Квиз
- **Потактовые модели**
- Общая схема симуляции
- Реализации потактовых моделей

Потактовые модели

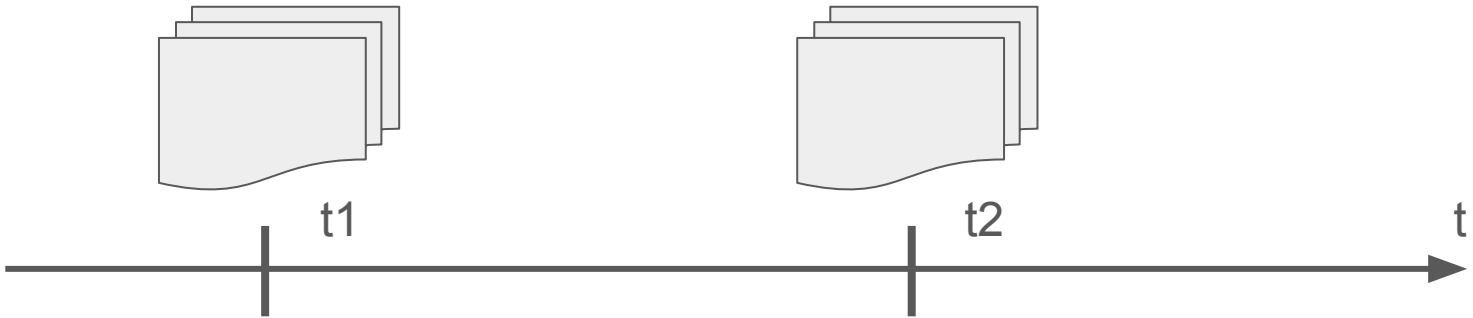
- Модель таймера фокусировалась только на внешних эффектах
- Иногда требуется иметь возможность рассмотреть внутренние процессы

Потактовые модели

- Рассмотрим сценарии моделирования
  - Одно устройство, события на каждом такте
    - Интерпретатор+ (CPU)
  - Много устройств, события значительно реже
    - Система дискретных событий
  - Много устройств, события на каждом такте
    - Сценарий хорошо ложится на модели цифровых схем
    - Каждый внутренний блок есть устройство

Потактовые модели

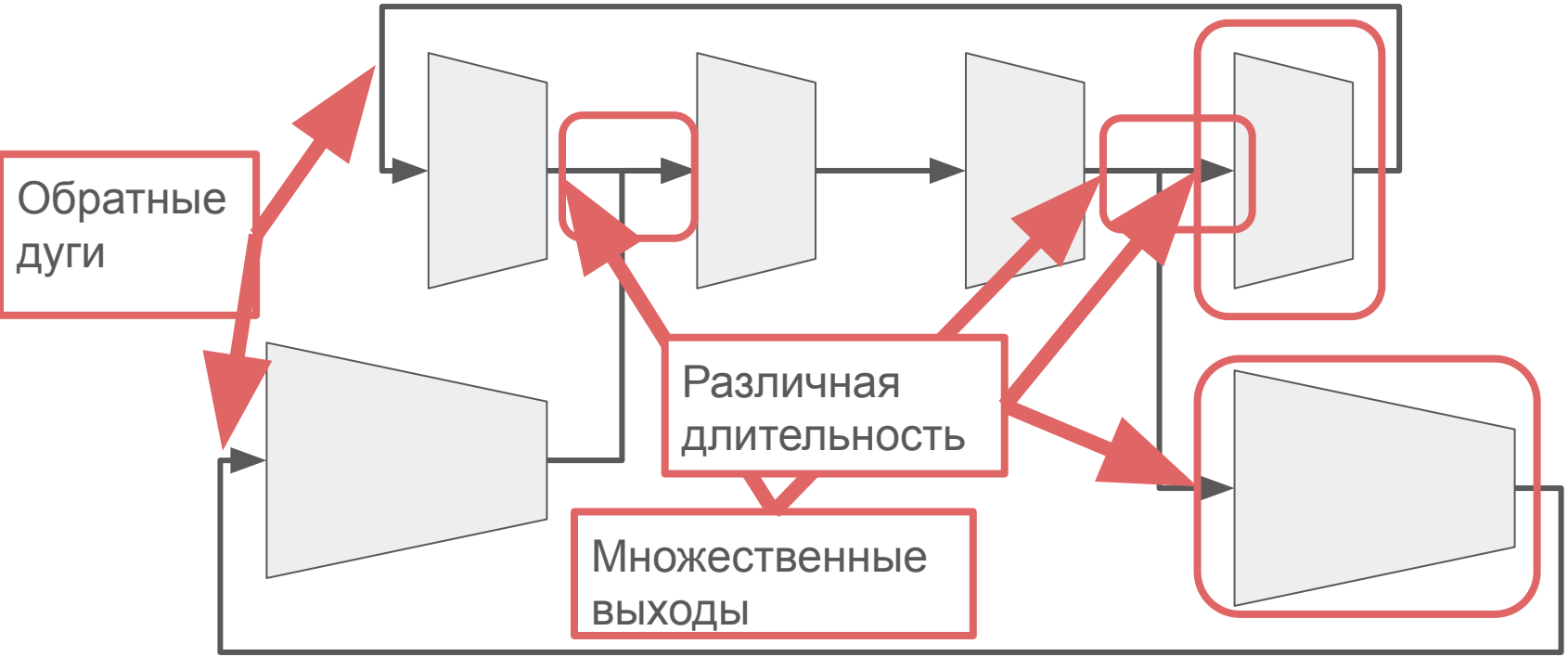
- Используем систему очередей, как для таймера
  - Каждый такт происходит много событий
  - Тяжело отслеживать перемещение данных и взаимодействие устройств
- Современные цифровые схемы - синхронные
  - Темп задаётся единым тактовым генератором
  - Состояния всех элементов меняются по фронту(спаду) тактового сигнала
  - Результаты текущего такта не будут переданы до наступления следующего



Потактовые модели. Особенности реализации

- Узлы реальной системы работают параллельно
  - Симулятор обрабатывает узлы последовательно
- Связи между узлами могут быть сложными
  - Выход может быть связан с предыдущим узлом, или даже с самим собой
  - Сложно установить глобальный порядок симуляции
- Некоторые операции занимают более 1 такта
- Узлы могут состоять из более мелких устройств
- Необходим подход, учитывающий все особенности

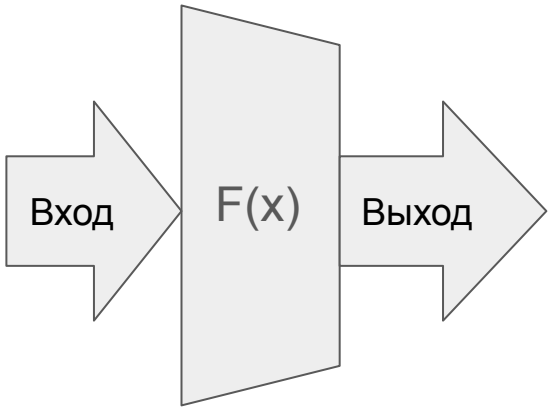
Потактовые модели. Особенности реализации



- Квиз
- Потактовые модели
- **Общая схема симуляции**
- Реализации потактовых моделей

Схема симуляции

- Попробуем функциональную и временнУю части
- Узлы выдают результат мгновенно (ведут себя как функции)
- Задержка моделируется с помощью линии задержки (*порты*) с фиксированной длинной и пропускной способностью



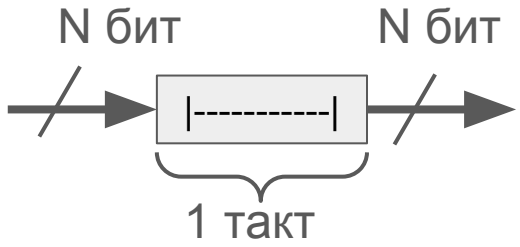
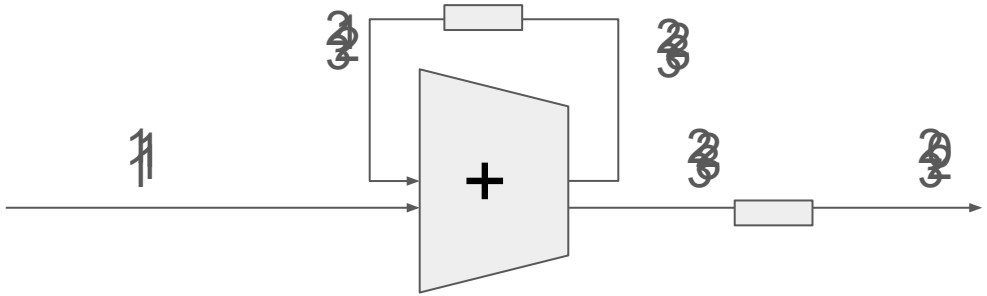


Схема симуляции

- Нельзя напрямую соединять два функциональных блока
- Симуляция происходит в два этапа
  - Вычисление функицональных блоков
  - Передача данных через порты



- Квиз
- Потактовые модели
- Общая схема симуляции
- ➢ **Реализации потактовых моделей**

Реализация моделей. Порт

- Как мы можем реализовать модель порта?
- Что-то похожее мы уже делали…

```
template <typename T>
class Port {
    T m_input{}, m_output{};
public:
    void input(const T &data) { m_input = data; }
    const T &output() const { return m_output; }
    void on_clk() { m_output = m_input; }
};
```

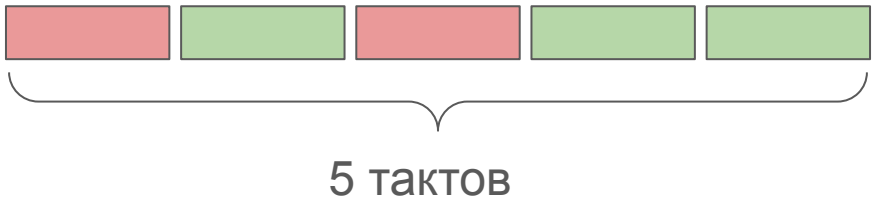
Реализация моделей. Порт

- Операции могут занимать более одного такта
- Как должен вести себя порт?
- Добавляем признак валидности данных

```
template <typename T>
class Port {
    std::optional<T> m_input{};
    std::optional<T> m_output{};
public:
    void input(const T &data) { m_input = data; }
    const auto &output() const { return m_output; }
    void on_clk() { m_output = m_input; }
};
```

Реализация моделей. Длительные задержки.

- Как моделировать более длительные задержки?
- Соединим несколько последовательных портов

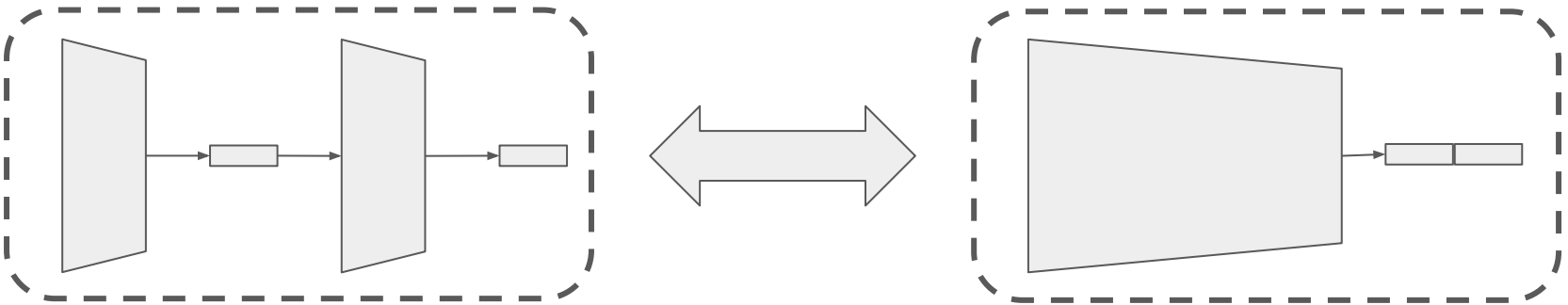


Реализация моделей. Длительные задержки.

- Все ли хорошо?
  - Можем нарушить изолированность стадий
- Как решить эту проблему?
  - Разделить порты повторителями
  - Установить порядок симуляции портов
  - Объединить в один многотактный порт

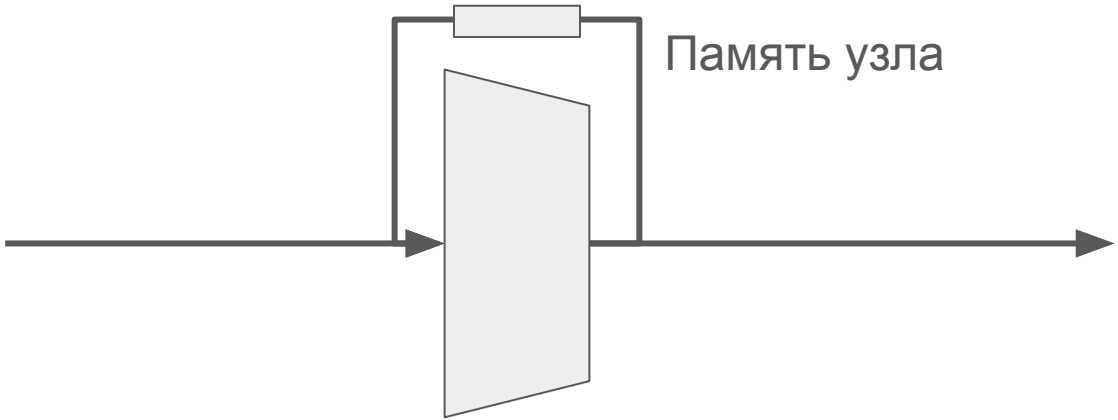
Реализация моделей. Композиция узлов

- Как можно оптимизировать симуляцию, если внутренние процессы между двумя узлами не представляют интереса?
  - Можно объединить их в композицию



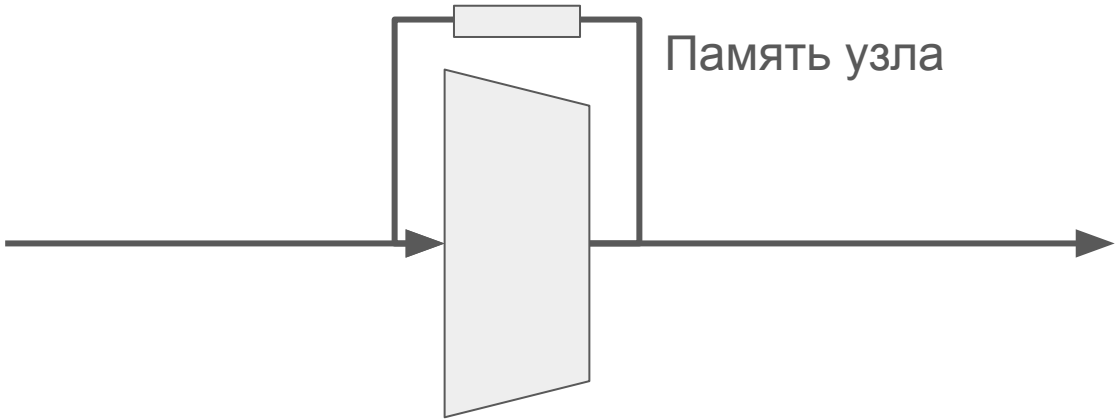
Реализация моделей. Память узлов

- Для реализации некоторых моделей устройств требуется хранить внутренне состояние узлов
- Используем порты с обратной связью!



Реализация моделей. Память узлов

- Если состояние узла очень объемное, то перегон памяти будет медленным
- Отмечаем изменившиеся данные на выходе



Реализация моделей. Ускорение симуляции

- Имеющаяся реализация сильно замедляет процесс симуляции
  - Функциональная симуляция ~100 MIPS
  - Потактовая симуляция ~1 MIPS
- Все узлы работают параллельно и независимо (благодаря двум этапам)
- Как можно ускорить симуляцию?
  - Параллельная симуляция
    - Число узлов может превышать число ядер процессора
  - Использование FPGA (ПЛИС)
    - Популярное решение
    - Сложно переносить модель

Реализация моделей. Модели задержек

- Создание потактового симулятора обычно начинается с реализации функциональной модели
- Хочется переиспользовать имеющуюся функциональность для уменьшения объема работы
- Моделируем только время конкретных операций
- Функциональная часть остается за функциональной моделью

Реализация моделей

- Потактовая симуляция ~1 MIPS
- Вообще говоря, нам не всегда нужна 100% точность подсчета тактов
- В таких случаях используются приближенные модели, которые работают быстрее засчет менее детального моделирования архитектуры

Итоги

- Зачем вообще нужны потактовые модели?
  - Позволяют понять как влияет изменение микроархитектуры на производительность (IPC)
  - Используется на финальных стадиях дизайна процессора

Домашнее задание

- Продолжаем писать симуляторы)

Литература

- Pellauer, Michael, et al. "A-Ports: An efficient abstraction for cycle-accurate performance models on FPGAs." Proceedings of the 16th international ACM/SIGDA symposium on Field programmable gate arrays. 2008.
- Pellauer, Michael, et al. "HAsim: FPGA-based high-detail multicore simulation using time-division multiplexing." 2011 IEEE 17th International Symposium on High Performance Computer Architecture. IEEE, 2011.