Лекция 8

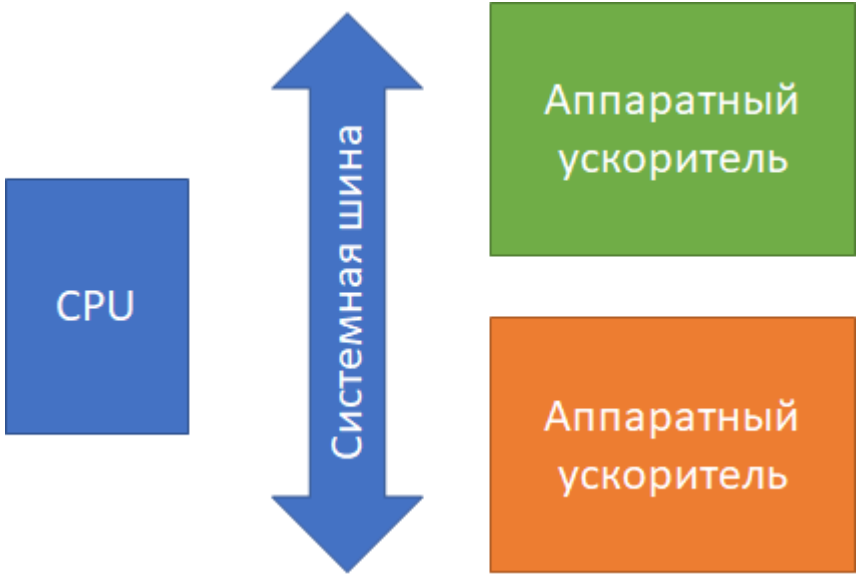
Схемотехника устройств компьютерных систем
Семестр 2

Тема: ПРОЕКТИРОВАНИЯ АППАРАТНОГО ОБЕСПЕЧЕНИЯ С ПРИМЕНЕНИЕМ ЯЗЫКОВ ВЫСОКОГО УРОВНЯ.

Люлява Даниил Вячеславович, старший преподаватель кафедры ВТ

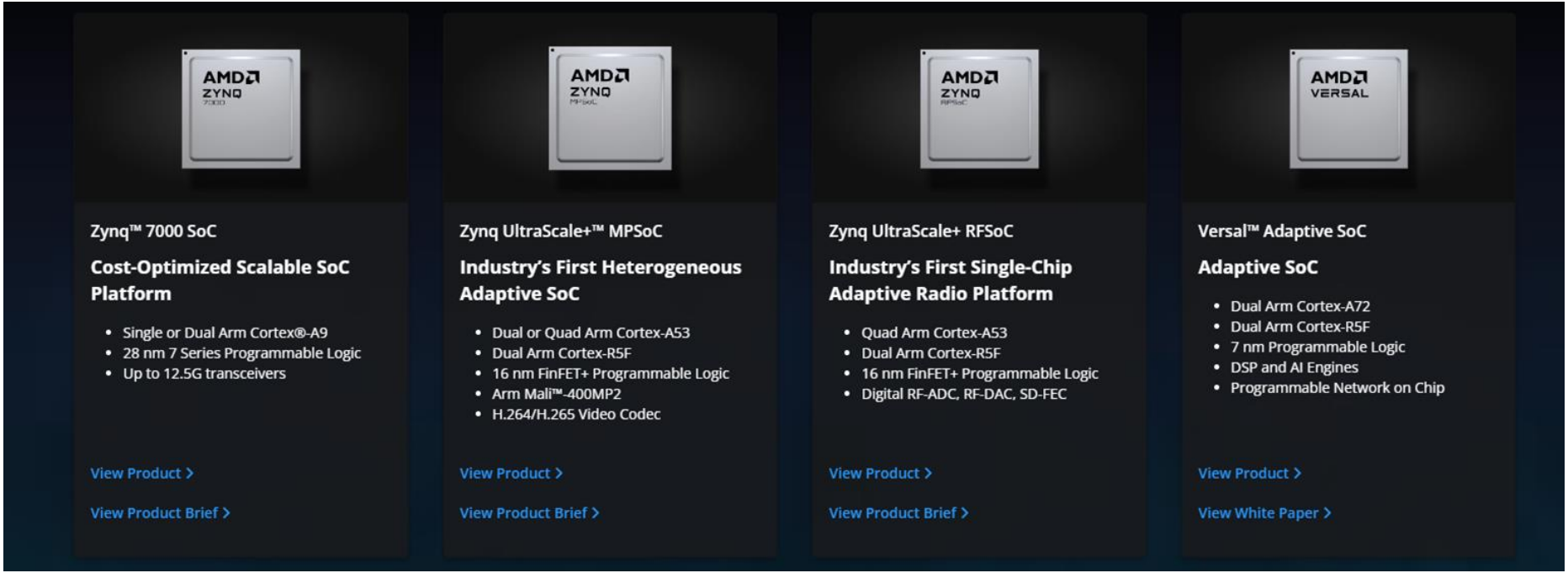
Дуксин Никита Александрович, преподаватель кафедры ВТ

Аппаратное ускорение вычислений



- Идея аппаратного ускорения – разгрузка центрального процессора путем передачи части вычислений в специализированные аппаратные модули
- Удобные для передачи в ускоритель задачи:
  - GPU
  - КИХ-фильтры
  - БПФ, вейвлет-преобразование
  - Вычисление статистических характеристик сигналов
  - Параллельные вычисления с простыми алгоритмами
- Неподходящие для ускорителей задачи
  - Сложные протоколы
  - Часто меняющиеся алгоритмы

Аппаратные платформы



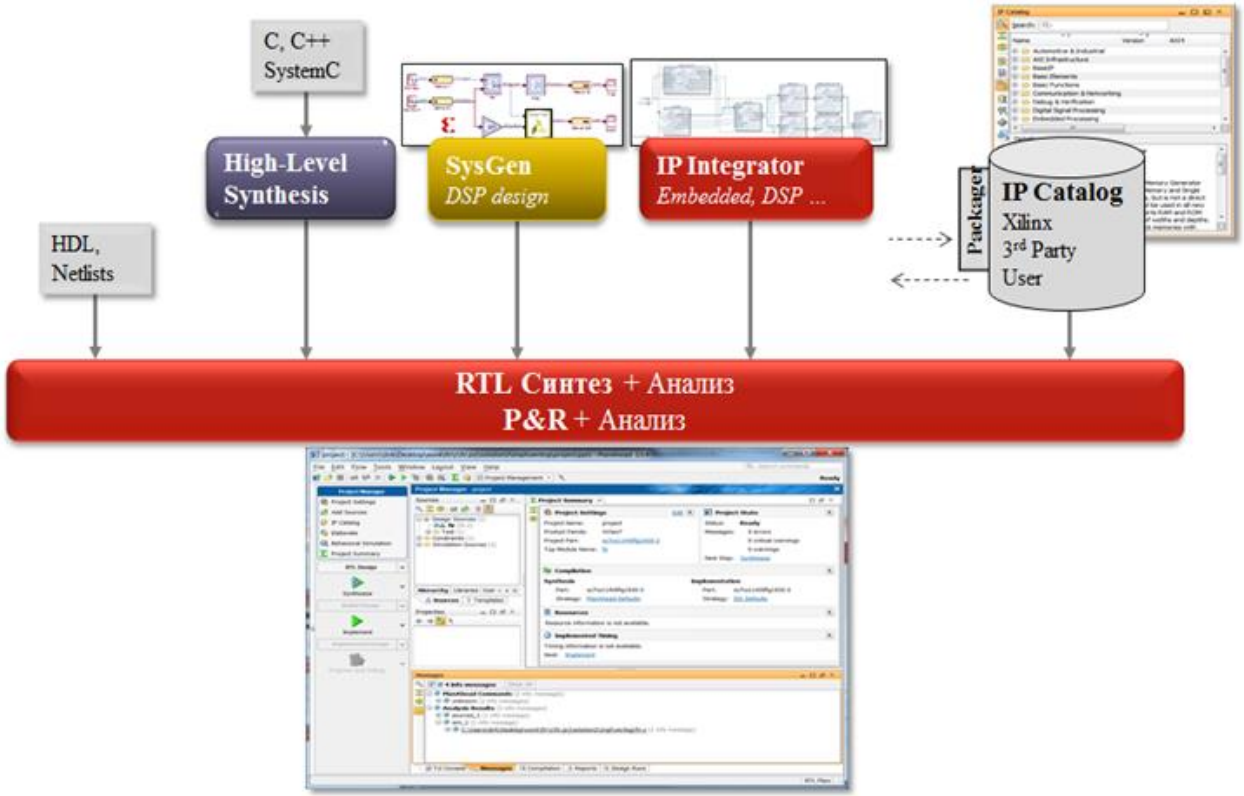
Современные тенденции и проблемы

- Основной инструмент проектирования аппаратного обеспечения – языки описания аппаратуры
- Высокий порог вхождения, необходимость добавления в рабочую группу специалистов по схемотехнике
- Сложность обеспечения взаимодействия, различные области компетенций по сравнению с программистами
- Длительный цикл проектирования по сравнению с программированием, разработка тяготеет с каскадной модели с предварительным формулированием детализированного технического задания
- Требуется:
  - Снижение порога вхождения в разработку и привлечение программистов и специалистов в предметной области

Системы класса HLS

- Следующее поколение языков проектирования аппаратуры, дополняющее HDL
- Языки класса «C-RTL», выполняющие преобразование Си-подобного описания аппаратуры в RTL-представление на одном из языков описания аппаратуры
- Первое поколение (2000) – Catapult-C, Handel-C
- 2012 год – Berkeley Design Technology Inc (BDTI) – HLS для САПР ПЛИС

HLS в маршруте проектирования



Постановка задачи

Процессор

```
A = 1;
B = 2;
C = A + B;
A = 0;
```

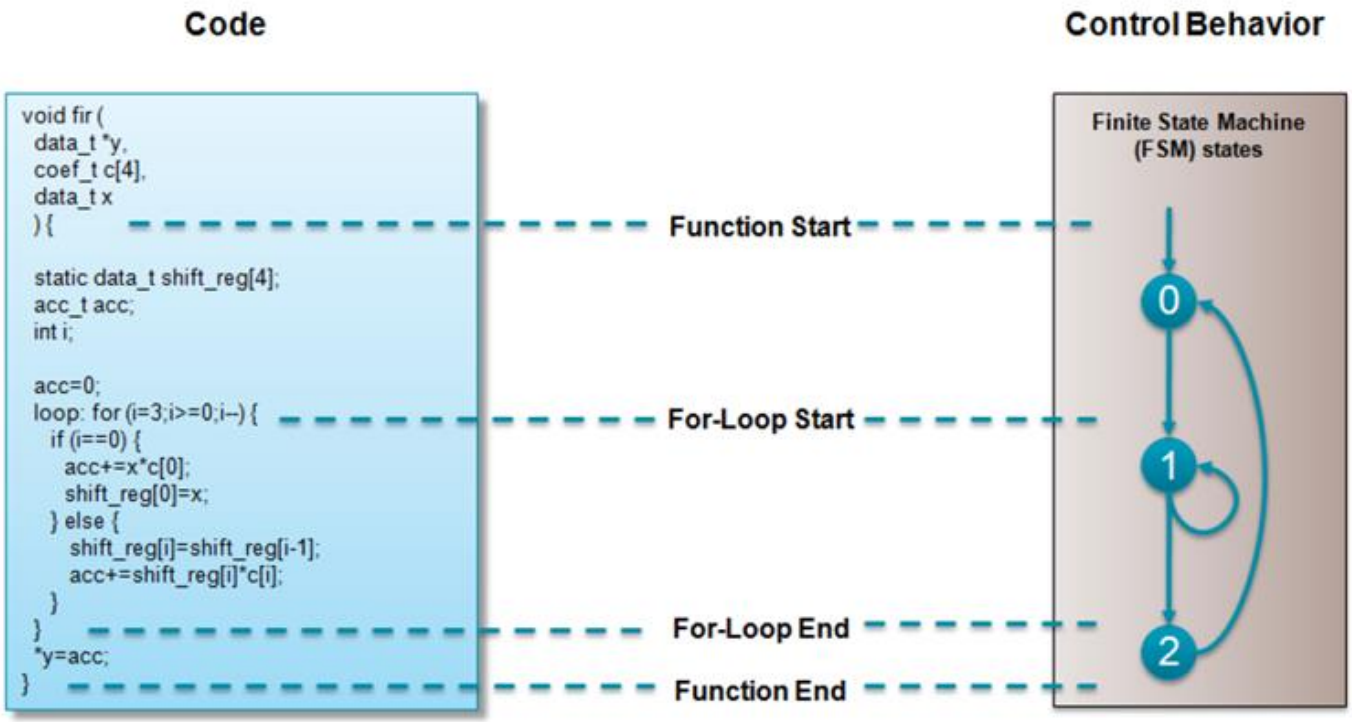
RTL с параллельностью операций

```
A <= 1;
B <= 2;
C <= A + B;
A <= 0;
```

При переносе программы в RTL требуется разрешение конфликтов.
Элементы проекта:

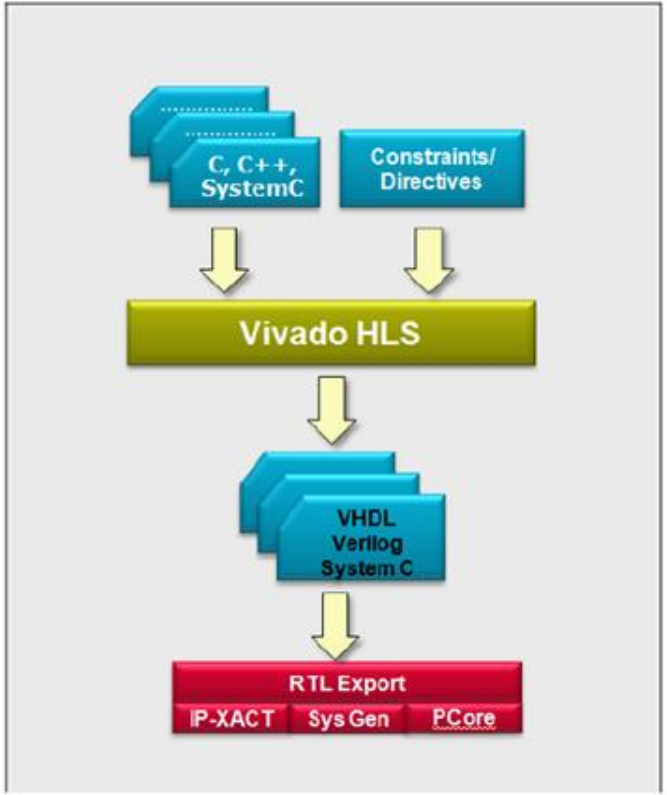
1. Datapath – схема преобразования данных
2. Control flow – автомат управления

Постановка задачи



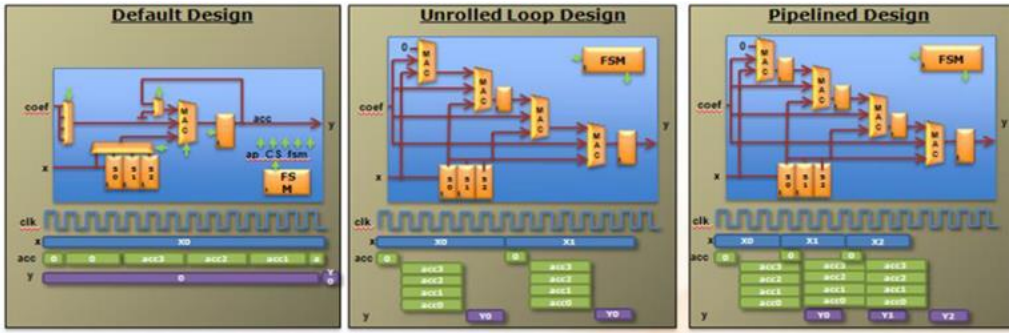
Взаимодействие HLS и Vivado

- Vivado HLS создает RTL-представление из исходного высокоуровневого кода на языке C/C++/SystemC
- Генерация RTL производится с учетом директив компилятора и возможностей ПЛИС
- Обработка RTL-представления производится в САПР Vivado



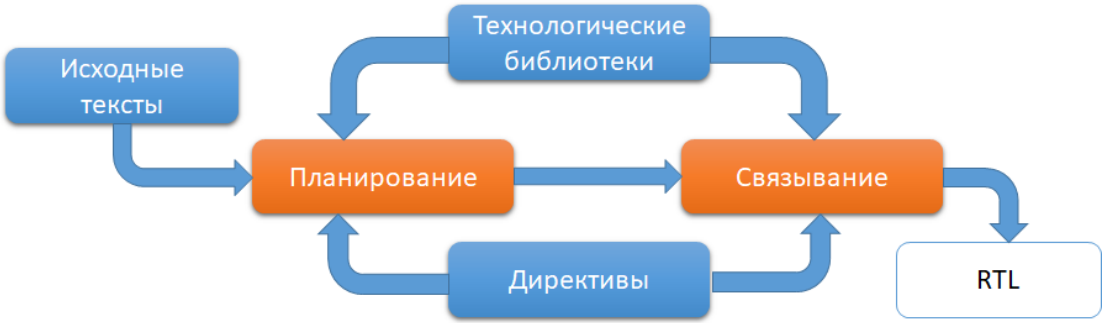
Директивы

- Один и тот же исходный текст может быть реализован различными схемами, различающимися числом тактов на реализацию и объемом аппаратных ресурсов
- Управление реализацией не является частью С-подобного языка и достигается добавлением директив компилятора



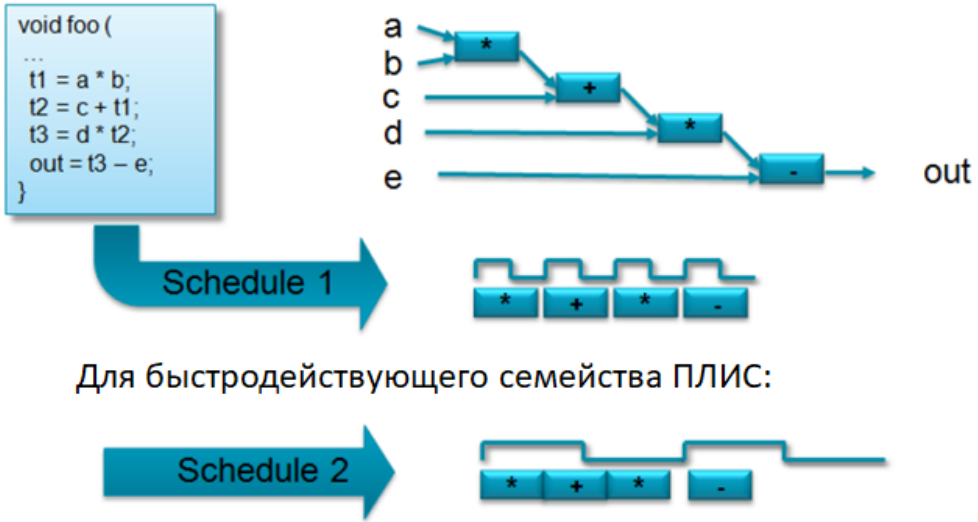
Планирование и связывание (scheduling and binding)

- Планирование определяет, на каком именно такте производится каждая из операций
- Учитываются построенные datapath и control flow, а также директивы и возможности технологической библиотеки
- Связывание определяет ресурсы ПЛИС, которые будут использованы для выполнения операций



Планирование и связывание (scheduling and binding)

Планирование операций производится с учетом технологических библиотек



Связывание

- Распределение ресурсов производится с учетом директив компилятора
- Пример 1: требуется два умножителя, т.к. обе операции умножения привязаны к одному и тому же такту

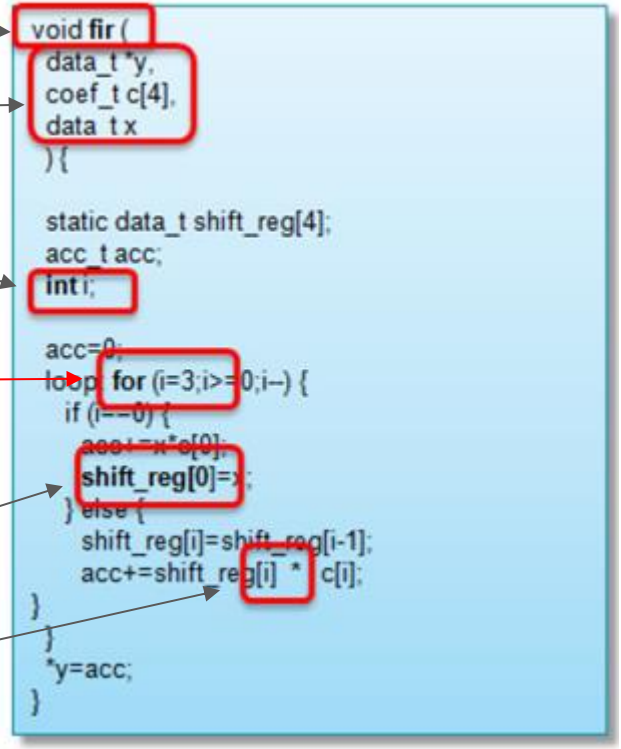
- Пример 2: операции умножения выполняются на разных тактах

Процесс синтеза в HLS

- Синтезатор HLS определяет, на каком такте выполняется каждое из действий
- Определяются устройства, выполняющие каждую из операций
- Производится оптимизация:
  - Производительность (частота и пропускная способность)
  - Минимизируется латентность
  - Минимизируется размер

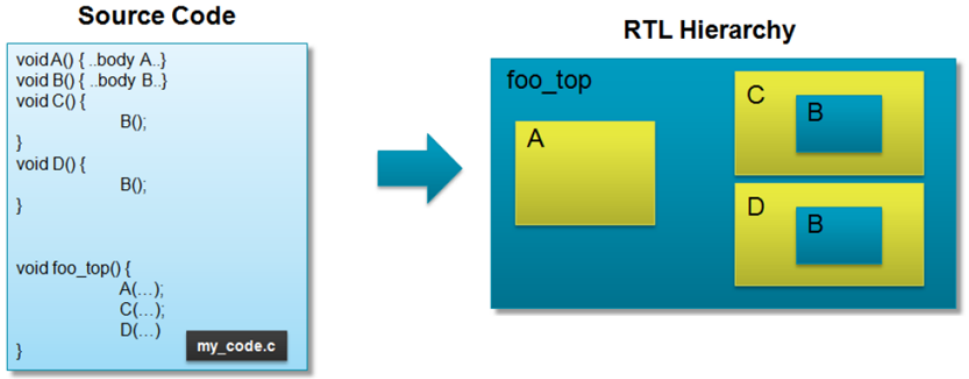
Преобразования кода на языке C

- **Функции** соответствуют аппаратным компонентам
- **Аргументы** функции верхнего уровня преобразуются в порты ввода-вывода
- **Типы** операндов описываются явно и влияют на размер и производительность
- **Циклы** могут быть реализованы различными способами (в зависимости от директив и устанавливаемых целевых показателей) и существенно влияют на размер и производительность
- **Массивы** также существенно влияют на характеристики проекта
- **Операторы** могут использовать разделяемые ресурсы или специфические аппаратные компоненты



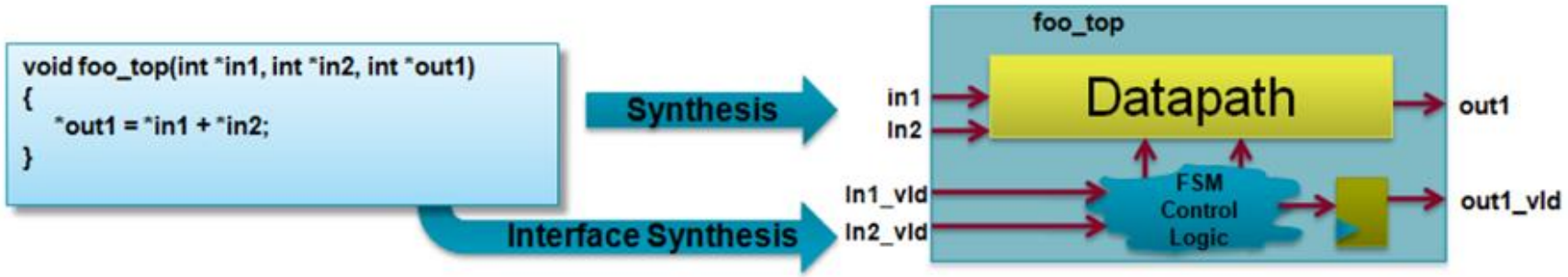
Функции

- Функции преобразуются в модули RTL
  - По умолчанию, каждая функция преобразуется в отдельный экземпляр схемы
  - Небольшие функции могут быть развернуты (inlining)



Аргументы

- Аргументы функции верхнего уровня становятся интерфейсными сигналами
- Добавляются сигналы для автоматической синхронизации

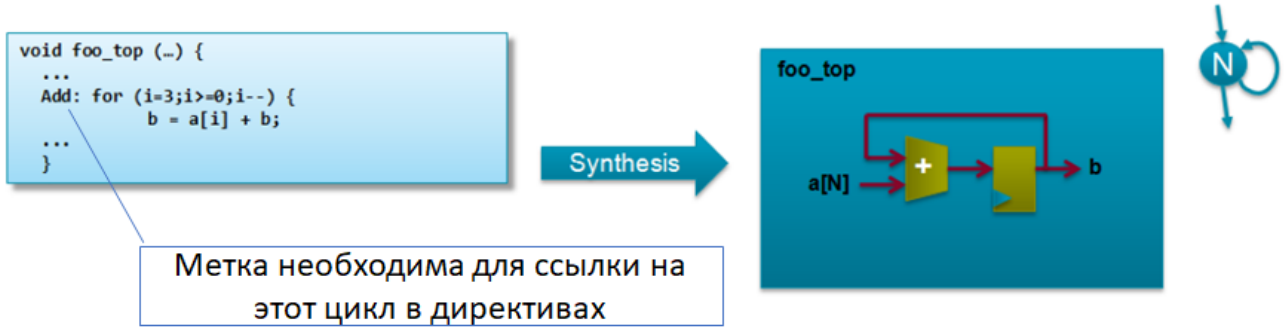


Типы

- Размер операндов указывается явно
  - Стандартные типы C:
    - Long long (64 bit), int (32 bit), short (16 bit), char (8 bit), float (32 bit), double (64 bit)
  - Типы с явным указанием разрядности (arbitrary precision)
    - C: int(1-1024), uint(1-1024)
    - C++: ap_(u)int(1-1024), ap_fixed
    - SystemC: sc_(u)int(1-1024), sc_fixed(1-1024)

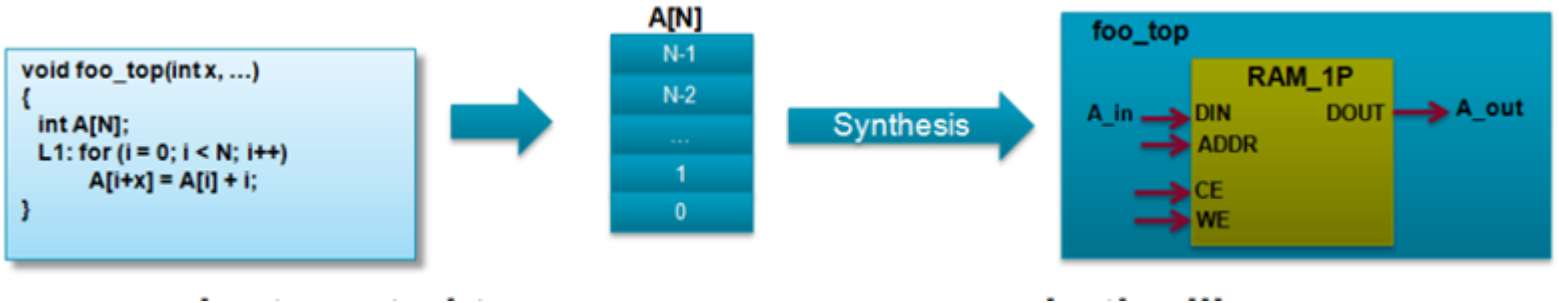
Циклы

- По умолчанию циклы не развернуты
  - Каждая итерация выполняется одним и тем же аппаратным модулем
- Циклы могут быть развернуты, если число итераций является константой, известной на этапе компиляции



Массивы

- По умолчанию массивы реализуются в памяти
- Можно указывать конкретный тип аппаратного примитива памяти
- Массивы можно объединять и разделять (для упаковки нескольких массивов в один аппаратный примитив или увеличения пропускной способности путем реализации в нескольких примитивах)



Операторы

- Размер операторов используется для синтеза достаточного аппаратного решения (есть возможность экономии ресурсов)
- Директивами можно контролировать максимальное количество блоков определенного типа (например, DSP48)

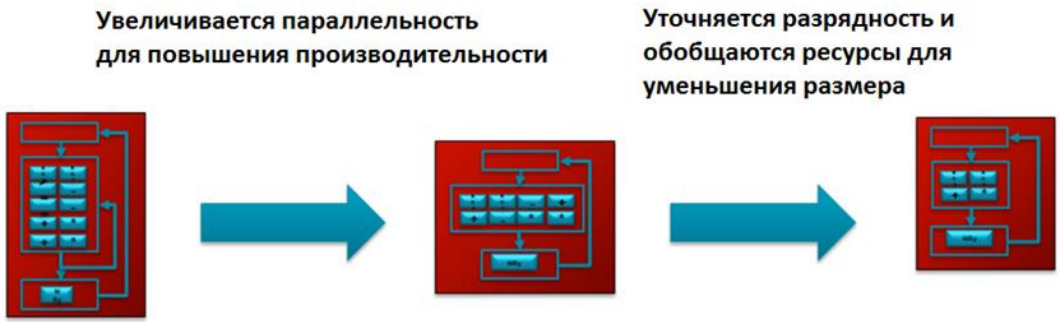
Ограничения

Не синтезируются:

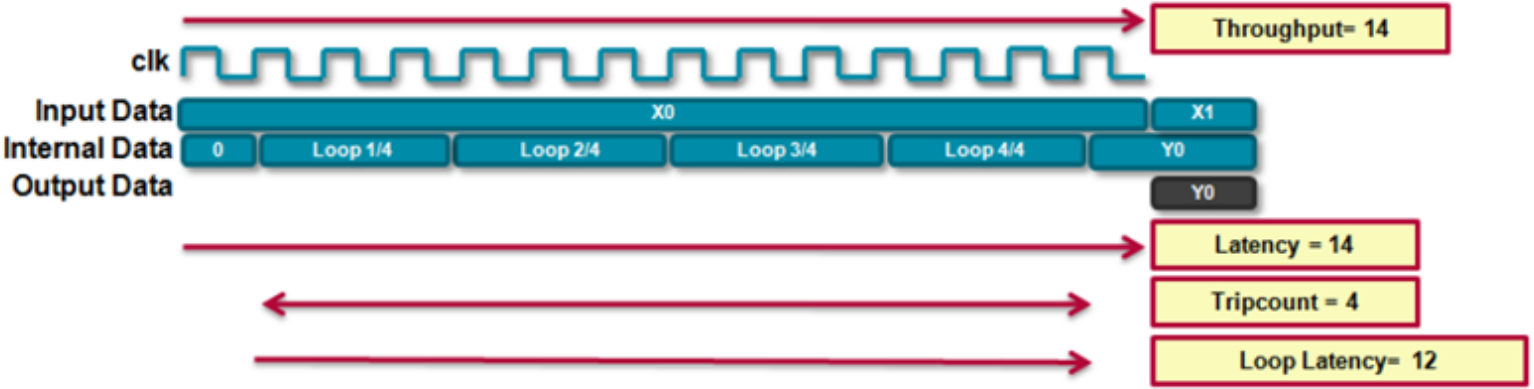
- функции динамического управления памятью (malloc/free);
- операции ввода-вывода (printf/scanf);
- системные вызовы (опрос таймера).

Порядок оптимизации в HLS

- Синтезируется начальный вариант
- Проверяется достижение заданной производительности
- Уменьшается размер



Термины для описания конвейеризованной схемы в HLS



Латентность - число тактов от входа до выхода

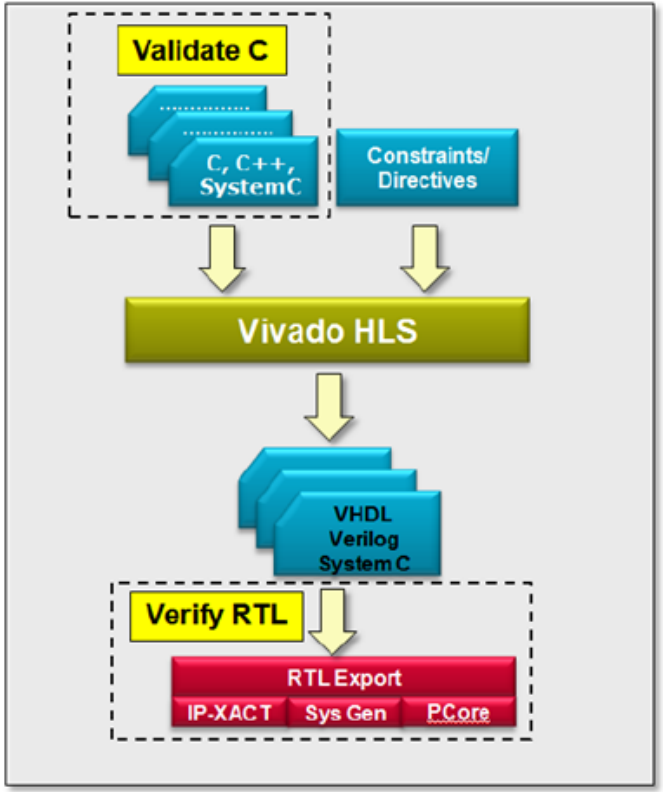
Пропускная способность (throughput) - число тактов между соседними входными отсчетами

Trip count - число тактов в одной итерации

Loop latency - латентность одного цикла

Моделирование

- Два шага в процессе моделирования
  - RTL validation – выполняется над исходным текстом на C
  - RTL verification – выполняется симулятором RTL-представления после синтеза схемы



Тестовые воздействия

- Тест является модулем на C, находящимся выше верхнего уровня синтезируемой части проекта
- Тест содержит функцию main()
- Рекомендации:
  - Самопроверяющийся проект (сравнение результатов с эталоном встроено в тест)
  - При успехе возвращать 0 (ненулевое значение трактуется в HLS как ошибка моделирования с выводом сообщения в консоль)

```
int main () {

  int ret=0;
  ...
  ret = system("diff  --brief  -w output.dat  output.golden.dat");

  if (ret != 0) {
       printf("Test failed  !!!\n");
       ret=1;
  } else {
       printf("Test passed !\n");
  }
  ...
  return ret;
}
```

Вопросы

Вопросы для подготовки к контрольной работе

1. Основные тенденции вычислительной техники. Проблемы проектирования.

Основные сведения о технологии производства интегральных схем. Технологический процесс, его характеристики. Понятие технологического сдвига. Понятие Power-Delay Product. Современные тенденции и проблемы: темный кремний, GALS, стена памяти. Синхронный стиль проектирования.

2. Особенности аппаратных платформ проектирования.

Маршрут проектирования и уровни описания. Этапы топологического моделирования. Факторы, оказывающие влияние на топологическое представление. Классификация аппаратных платформ. Сравнительная характеристика FPGA и ASIC. Основные риски при выборе платформы. Обзор ПЛИС Xilinx/AMD.

3. Архитектурные аспекты проектирования. Часть 1.

Проблемы расстановки элементов на кристалле. Концепция «No silver bullet». Паттерн проектирования «Комбинационная схема». Паттерн проектирования «Конечный автомат». Кодирование состояний конечного автомата. Элементы памяти. Паттерн проектирования «Конвейер». Средства и стратегии размещения САПР.

4. Архитектурные аспекты проектирования. Часть 2.

Паттерн проектирования «Процессорное ядро». Переход от конечного автомата к процессорному ядру. Понятие абстракции машины Тьюринга. Структура процессорного ядра. Классификация процессоров. Гарвардская архитектура. Архитектура Фон-Неймана. Пример построения простого процессорного ядра.

5. Архитектурные аспекты проектирования. Часть 3.

Проблемы двухтактной архитектуры. 3, 4, 5-и тактные архитектуры. Конвейерное процессорное ядро 2, 3, 4 такта. Основные возникающие конфликты. Архитектура MIPS. Проблема отложенного перехода. Соотношение производительности тракта данных и управляющей схемы. Архитектуры VLIW, EPIC. Регистровая модель процессора. Регистровый файл. Адресность команд. Архитектуры RISC, CISC.

6. Системные шины.

Понятие и предназначение системной шины. Интерфейс простой системной шины. Параллельные шины. Шина ISA. Шина PCI. Шина Wishbone. Мосты для системной шины. Арбитраж системной шины. Коммутатор системной шины. Приёмы проектирования: группировка, memory mapped. Сочетание процессорного управления и автономной работы. AXI. AXI4-Lite. Пример процессорной системы класса СНК.

7. Сопряжение измерительных и силовых устройств с цифровыми системами.

Ввод аналоговых сигналов в компьютерных системах. АЦП, его характеристики. Архитектуры и интерфейсы АЦП. Сопряжение АЦП с цифровыми системами. ЦАП. Интерфейсы ЦАП. Сопряжение ЦАП с цифровыми системами. Управление силовыми устройствами с помощью ШИМ.

8. Проектирования аппаратного обеспечения с применением языков высокого уровня.

Аппаратное ускорение вычислений. Аппаратные платформы. Современные тенденции и проблемы. Системы класса HLS. HLS в маршруте проектирования. Типовая задача HLS. Взаимодействие HLS и Vivado. Директивы. Планирование и связывание. Процесс синтеза в HLS. Порядок оптимизации в HLS. Термины для описания конвейеризованной схемы в HLS. Моделирование и тестовые воздействия.

Спасибо за внимание!