Примерная программа пятидневного семинара вокруг RTL2GDSII, FPGA, ASIC, SoC, uArch

День 1. До обеда: Что из себя представляет электронная индустрия?

Вся электроника ($2T), индустрия полупроводников ($350B), индустрия средств автоматизации проектирования (Electronic Design Automation — EDA), индустрия полупроводниковой интеллектуальной собственности (Semiconductor IP). Уровни абстракции системы на кристалле и соответствующие им специализации инженеров. Уровни программы, архитектуры / системы команд, микроархитектуры / устройства конвейера, уровень регистровых передач (Register Transfer Level — RTL), уровень логических элементов, уровень транзисторов, уровень физики. Как проектируются и производятся микросхемы. Концепция маршрута разработки микросхем RTL-to-GDSII: спецификация, описание на RTL, моделирование и верификация, логический синтез, размещение и трассировка, производство. Экономика разработки и производства разных типов микросхем. Позиции на рынке Application-Specific Integrated Circuit (ASIC) и Field-Programmable Gate Array (FPGA). Баланс между стоимостью, гибкостью, количеством, начальными тратами и тратами за время жизни проекта.

День 1. После обеда: Использование языка Verilog для разработки комбинационной логики

Краткое введение в язык описания аппаратуры Verilog и сравнение его с VHDL. Модули, порты, иерархия. Типы данных, выражения, присваивания. Always-блоки и основные операторы. Подмножества языка для реализации схемы и для ее тестирования / верификации.

Комбинационная логика. Базовые логические элементы, базовые комбинационные блоки (мультиплексоры, дешифраторы), временные характеристики (задержка распространения, задержка реакции). Комбинационные арифметические схемы и варианты их оптимальной реализации.

Лабораторная работа 1. Создание простой комбинационной схемы. Работа с программами Xilinx Vivado и Altera Quartus II для логического синтеза, размещения и трассировки. Конфигурация плат Digilent Nexys4 DDR (или Cmod A7) и Terasic DE0-CV (или DE10-Lite), работа на них простых цифровых схемы. Каждый слушатель получает индивидуальное задание.

День 2. Использование языка Verilog для разработки последовательностной логики

Последовательностная логика. D-триггеры, временные характеристики (время предустановки, время удержания), временные ограничения, максимальная тактовая частота.

Базовые последовательностные блоки (счетчики, сдвигающие регистры), последовательностные арифметические схемы. Память (регистровые файлы, статическая, динамическая).

Конечные автоматы. Диаграммы состояний и реализация конечных автоматов на языке Verilog.

Правила для написания корректного кода на Verilog для уровня регистровых передач. Объяснения, какие проблемы могут происходить (например гонки - race conditions во время моделирования), если не следовать этим правилам.

Концепция конвейера, применимость для повышения пропускной способности. Пример использования конвейера для реализации математических операций.

Лабораторная работа 2. Проектирование последовательностного блока - счетчика, сдвигового регистра или простого конечного автомата.

День 3. Искусство верификации цифровых схем, микропроцессоров и систем

Уровни моделирования и верификации - физический, уровень логических элементов, уровень регистровых передач, уровень транзакций, уровень инструкций при моделировании процессоров. Совместное тестирование аппаратной и программной части. Что происходит внутри программы-симулятора.

Написание сред тестирования. Введение в SystemVerilog и связанные с ним технологии - coverage-driven constrained-random verification methodology. Universal Verification Methodology (UVM) и ее применение в индустрии.

Лабораторная работа 3.1. Использование симуляторов на уровне регистровых передач: Vivado Simulator, ModelSim и Icarus Verilog. Примеры тестового окружения на Verilog 2001, SystemVerilog без UVM и SystemVerilog с UVM.

Лабораторная работа 3.2. Моделирование на уровне архитектуры процессора / инструкций. Использование MARS MIPS simulator. Каждый слушатель получает индивидуальное задание.

День 4. Архитектура и микроархитектура микропроцессорных ядер

Разница между архитектурой (системой команд и видимыми программисту регистрами) и микроархитектурой (аппаратной организации микропроцессора).

Введение в микроархитектуру процессоров:

1. Простейшая организация - однотактный процессор
2. Многотактный процессор - тактовая частота выше, но выполнение команды требуют больше циклов
3. Конвейереная организация процессора и конфликты конвейера

Способы улучшения производительности микроархитектуры:

1. Разрешение конфликтов конвейера с помощью байпасов (bypass / forwarding)

2. Длина конвейера у различных процессоров как компромисс между тактовой частотой и количеством тактов на команду
3. Предсказатель переходов для уменьшения количества очисток (flush) длинных конвейеров во время условного перехода

MIPSfpga - введение. MIPSfpga — это пакет, который содержит процессорное ядро в исходниках на Verilog, которое можно менять, добавлять новые инструкции, строить многопроцессорные системы, менять одновременно программы и аппаратуру.

Сравнение MIPSfpga с другими процессорными ядрами, которые применяются в университетах Сравнение с Xilinx/ARM Zynq 7000, ARM Design Start, Xilinx MicroBlaze, Xilinx PicoBlaze, Altera NIOS II, RISC/V, OpenRISC, ARM-compatible Amber, OpenSPARC / UltraSPARC T1/T2, LEON4.

Введение в различные типы шин используемых внутри и вне системы на кристалле - AHB-Lite, AXI, OCP, SPI, UART, I2C.

Описание структуры MIPSfpga+ - варианта MIPSfpga:

1. Внешняя оболочка для синтеза
2. Структура системы из процессорного ядра, контроллера памяти и периферийных устройств ввода-вывода
3. Периферийное устройство - датчик освещения
4. Описание загрузки программы в синтезированную систему через UART
5. Демонстрация симуляции системы MIPSfpga, работы среды тестирования и анализа временных диаграмм
6. Демонстрация синтеза системы, компиляции примеров программ, конфигурация платы ПЛИС синтезированной системой, загрузка программы в синтезированную систему и совместная работа всех компонент

Лабораторная работа 4. Синтез и использование системы с MIPSfpga. Интеграция датчика освещения.

День 5. Более продвинутые аспекты микроархитектуры процесоров и систем на кристалле

Более продвинутые способы повышения производительности процесоров:

1. Аппаратная поддержка многопоточности
2. Суперскалярные процессоры
3. Векторные расширения, или одиночный поток команд, множественный поток данных (single instruction multiple data, SIMD)

Обзор лицензируемых микропроцессорных ядер MIPS, оптимизированных для различных приложений.

Лабораторная работа 5.1. Использование MIPSfpga для исследования работы конвейера

Введение в кэши. Кэши прямого отображения, многосекционные наборно-ассоциативные кэш, иерархия кэшей.

Многоядерные системы:

1. Проблема когерентностей кэшей и памяти
2. Решение - протоколы MESI и подобные
3. Некогерентные и гетерогентные системы

Лабораторная работа 5.2. Использование MIPSfpga для исследования работы кэша
Прерывания в микропроцессоре и операционной системе.

Лабораторная работа 5.3. Использование MIPSfpga для исследования работы прерываний
Дополнительный проект: использование интерфейса расширения процессора CorExtend для добавления инструкция и создания сопроцессоров.