ПАК «Вентилятор»
для ускорения моделирования RTL

Монахов А.М.
ООО «Мальт Систем»

MALT system

Содержание

РОССИЙСКИЙ ФОРУМ
МИКРОЭЛЕКТРОНИКА 2025

Введение

Введение

- В диалоге про цифровые САПР многие делают акцент на инструментах имплементации, но верификация RTL не менее важна;
- Аппаратное ускорение верификации необходимо для больших проектов и является общемировым трендом;
- Мы в компании Мальт Систем разработали ПАК для ускорения моделирования RTL «Вентилятор»;
- В настоящей презентации мы обсудим преимущественно программную сторону вопроса и возможные точки взаимодействия, по коммерческим вопросам была отдельная презентация на круглом столе.

Проблема верификации RTL

- Современные микропроцессоры, системы на кристалле (СнК) и другие сложные цифровые микросхемы содержат в себе сотни миллионов базовых логических элементов (вентилей);
- В процессе разработки таких схем неизбежно встаёт задача верификации корректности функционирования их HDL-кода. Особенно остро эта проблема стоит для СнК, внутри которых процессор совмещен с высокоскоростными интерфейсами и периферийными устройствами;
- В процессе разработки СнК помимо верификации непосредственно RTL, необходимо проводить верификацию ОС, драйверов и других сложных программных компонентов;
- По статистике верификация RTL-дизайна, верификация синтезированных списков межсоединений и совместная отладка RTL-кода и системного ПО СнК занимают более 70% времени разработки СнК;
- При этом для сокращения цикла разработки микросхемы и снижения вероятности ошибок в конечном устройстве, необходимо начинать совместную верификацию RTL и ПО как можно на более ранних этапах проекта.

Моделирование в RTL симуляторах

Преимущества:

- Не требует дополнительного оборудования;
- Возможность моделировать дизайн с самых ранних стадий реализации, в том числе проводить модульные тесты;
- Доступ к полной отладочной информации о дизайне, в том числе о состоянии всех сигналов внутри в каждый момент времени;
- Быстрота итерирования - время запуска дизайна на моделирование редко превышает 15 мин.

Основной недостаток RTL симуляторов - ограниченная производительность моделирования, для больших СнК эффективная частота может составлять 10 Гц и менее.

Прототипирование на ПЛИС

Преимущества:

- Скорость моделирования СнК обычно составляет 10-100 МГц;
- Это позволяет проводить сложные тесты вроде загрузки ОС;
- ПЛИС предоставляют возможность проверять интерфейсы к реальным устройствам.

Недостатки:

- ПЛИС предоставляют очень ограниченный доступ к информации о проекте;
- Даже самые большие и дорогие ПЛИС не способны вместить HDL проект размером более чем порядка 100 млн эквивалентных вентилей;
- Не любой код исполняемый в RTL симуляторах может быть реализован на ПЛИС;
- Время сборки одного проекта на большую ПЛИС может превышать 10ч, что ограничивает скорость итерации верификации.

ПАК ”Вентилятор”

Технические характеристики

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2025

- Поддерживаемые HDL языки: Verilog-2005, SV-2015 (с огр.), VHDL-2008;
- Языки написания тестбенчей: Verilog-2005, SV-2015 (с огр.), Cocotb (с огр.);
- Интерфейсы для подключения моделей: SystemVerilog DPI, Cocotb, C++ API;
- Поддержка моделирования как поведенческого RTL, так и нетлистов без задержек;
- Максимальный объем проекта: 100 млн вентилей / 1U;
- Максимальный объём DRAM-памяти: 32 ГБайт / 1U;
- Скорость компиляции дизайна не менее 50 млн экв. вентилей/ч;
- Максимальное количество IO: до 8 тыс;
- Максимальное количество частотных доменов: до 100.

Возможности ПАК ”Вентилятор”

Рыночная ниша:

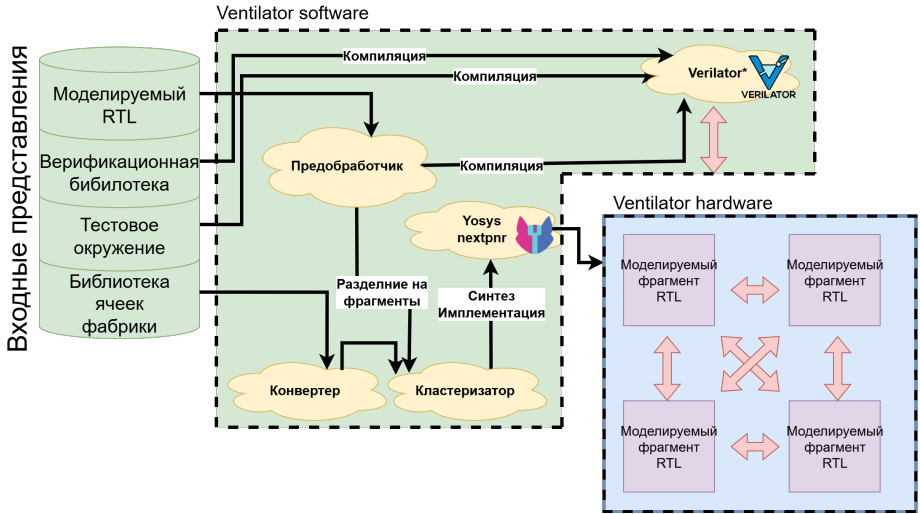
- Верификация СнК размером до 100 млн. вентилей (в перспективе до 1 млрд.);
- Отладка ПО и драйверов параллельно с доработкой RTL;
- Моделирование синтезированных нетлистов без учета временных задержек.

Ограничения:

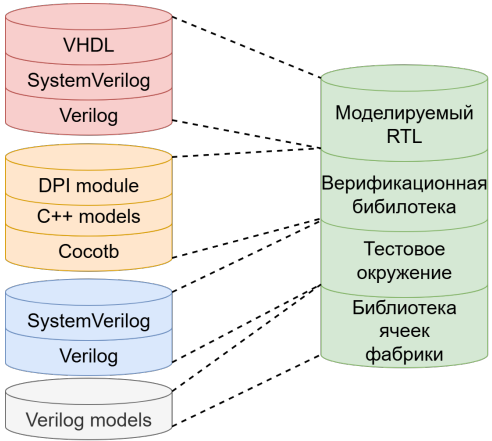
- Не является заменой event-driven симуляторов;
- Невозможность симуляции нетлистов с SDF задержками;
- Предназначен для моделирования синхронных дизайнов;
- Процент синтезируемых модулей в дизайне существенно влияет на производительность;
- Отсутствие поддержки X-состояния сигнала (аналогично ПЛИС).

Программные компоненты ПАК ”Вентилятор”

Схема работы – программная абстракция



Поддерживаемые представления дизайнов



- В синтезируемой части SystemVerilog поддерживается также как в yosys_slang
- В несинтезируемой части SystemVerilog поддерживается как в Verilator

Предобработка Verilog/System Verilog

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2025

Действия выполняемые на этапе предобработки:

- Депараметризация Verilog/System Verilog;
- Определение признаков синтезируемости;
- Разрешение иерархических путей к сигналам и параметрам;
- Синтаксическая обработка Verilog для соответствия требованиям Yosys и Verilator.

```
module mod #(parameter N)
    (input [N-1:0] a, ...);
mod # (.N(1)) inst1_0 (...);
mod # (.N(1)) inst1_1 (...);
mod # (.N(2)) inst2 (...);
```

=>

```
module mod_N1 (input [0:0] a, ...);
module mod_N2 (input [1:0] a, ...);
mod_N1 inst1_0 (...);
mod_N1 inst1_1 (...);
mod_N2 inst2 (...);
```

Предобработка Verilog/System Verilog

- Невозможно использовать Yosys, т.к. он не обрабатывает несинтезируемый код;
- Существуют альтернативные Verilog/SV фронтэнды;
- Изначально мы использовали Verible, потом перешли на более комплексный фронтэнд Slang.

	Поддержка стандартов	Парсинг	Представление дизайна	Обработка дизайна	Контроль ошибок	Удобство редактирования
Verible	IEEE 1800-2017	Bison/Yacc	Обычное синтаксическое дерево	Только препроцессор	Только синтаксические	Удобное синт. дерево, Нет механизма контроля памяти
Slang	IEEE 1800-2017 IEEE 1800-2023	Проприетарный парсер	Сложные проприетарные форматы	Разрешение параметров, Инстанцирование модулей, пр.	Синтаксические, Контроль типов, Контроль портов, пр.	Есть контроль памяти, неудобное представление дизайна

Конвертация библиотек ячеек в синт. Verilog

Для возможности аппаратного ускорения моделирования нетлистов, ячейки из библиотек от вендоров преобразуются в синтезируемые модули.

Последовательность преобразования:

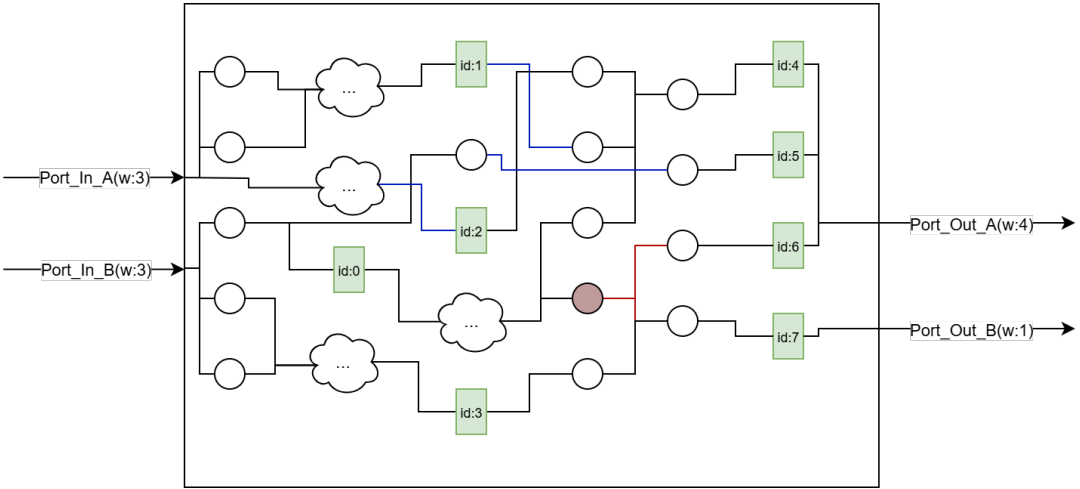
- Трансформация Verilog table в классическую таблицу истинности;
- Формирование и упрощение логической функции;
- Определение списка чувствительности и CLK-портов;
- Преобразование логической функции в Verilog код;
- Сравнение полученной модели с исходной в симуляции.

```
NAND2V2_140P9T35L g642(
   .A1 (n_12), .A2 (n_13),
   .ZN (n_16));
```

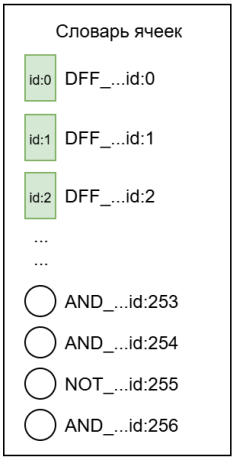
=>

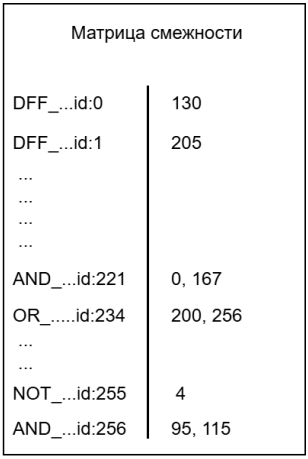
```
// модуль эквивалентный:
assign n_16 = ~(n_12 & n_13);
```

Кластеризация



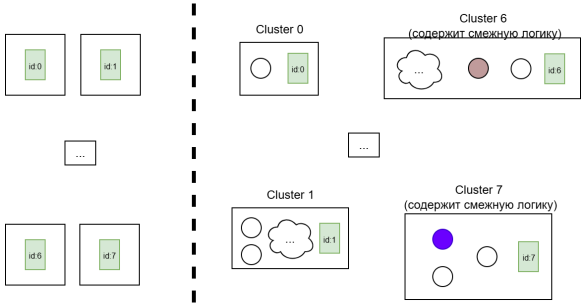
Кластеризация





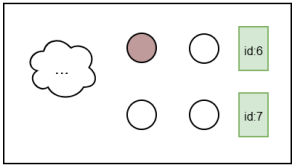
Выбор стартовых ячеек-триггеров для кластеров

Кластеризация

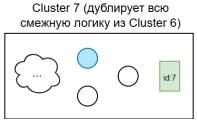


Поиск в ширину (BFS) драйверов триггеров с добавлением их в кластеры

Кластеризация

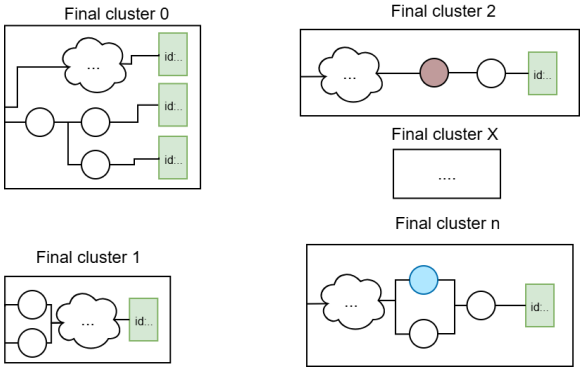


Объединение кластеров со смежной логикой



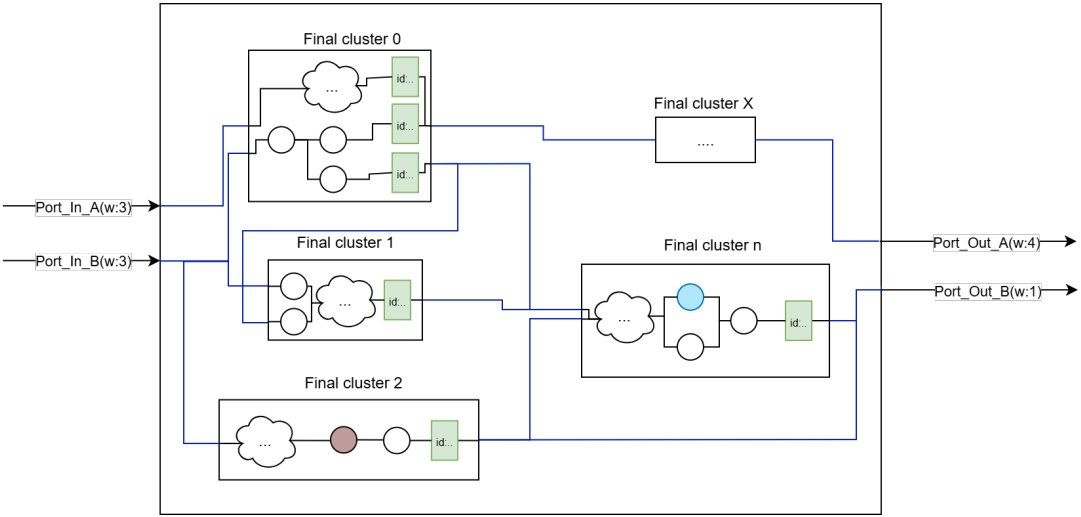
Разделение кластеров содержащих смежную логику с её дублированием

Кластеризация



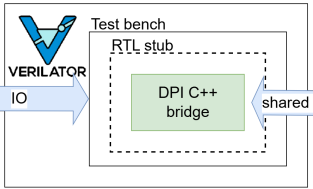
Формирование нетлистов кластеров

Кластеризация



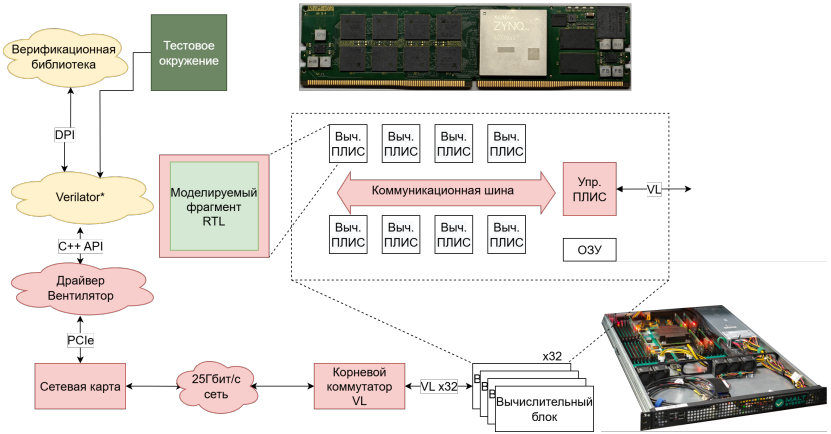
Синхронизация состояния


```
(MALT) worker@Ventilyator:~/egor/emuflow$ ./run.sh ../test_designs/designs/litex_vexriscv8_linux/litex_vexriscv8_linux.json
################################################################################
EmuFlow v.0.1 by MALT System
Starting Ventilator flow...
################################################################################


Flow directory: /home/worker/egor/emuflow/runs/20240705_101036_cluster_verilator_fpga_flow

Running stage 1: Verilog precheck ... completed!  Time: 5.942 s
Running stage 2: Initialization ... completed!  Time: 0.086 s
Running stage 3: Clustering ... completed!  Time: 521.667 s
Running stage 4: Cluster check ... completed!  Time: 1.367 s
Running stage 5.1: Implementation ...
Running stage 5.2: Make binaries ...
Stage Make binaries (5.2)  completed!  Time: 18.085 s
Stage Implementation (5.1)  completed!  Time: 255.105 s
Running stage 6: Run simulation ...

Simulation
[    0.000000] Linux version 5.14.0 (florent@panda)
(riscv32-buildroot-linux-gnu-gcc.br_real (Buildroot 2021.08-381-g279167ee8d)
10.3.0, GNU ld (GNU Binutils) 2.36.1) #1 SMP Tue Sep 21 12:57:31 CEST 2021
[    0.000000] earlycon
Simulation successful!
-
/home/worker/egor/test_designs/designs/litex_vexriscv8_linux/VexRiscvLitexSmpCl_
Verilog $finish
Resulting simulation speed: 8737.75 cycles/s

Stage Run simulation (6)  completed!  Time: 654.051 s
Runtime: 1439.033 s
Flow Ventilator completed succesfully!
```



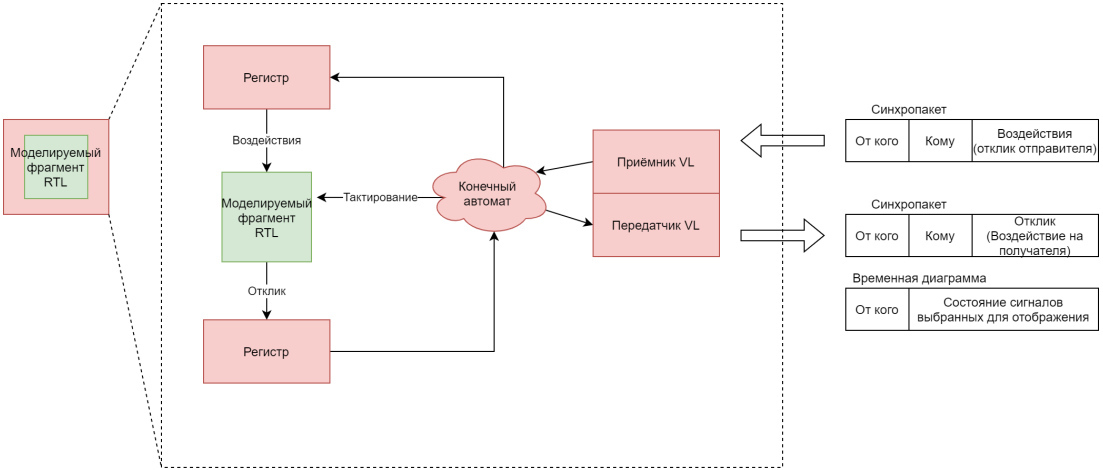
Аппаратная реализация ПАК ”Вентилятор”

Схема работы – аппаратная реализация



Механизм синхронизации и передачи временных диаграмм

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2025



Заключение

Направления взаимодействия

- Ведётся сбор тестовых примеров и пожеланий к архитектуре тестового окружения от сторонних потребителей;
- Готовится запуск открытого бета-тестирования с доступом к ПАК для сторонних дизайн-центров с поддержкой от наших инженеров;
- Уже сейчас мы можем предоставить заинтересованным предварительную версию документации пользователя;
- В ходе этой и других ведущихся в компании работ по САПР на базе открытых инструментов нами наработан богатый опыт, который будет полезен в будущих совместных работах по этой тематике.

Спасибо за внимание!

Фото вычислительного модуля