САПР ДЛЯ РАЗРАБОТКИ IP-CORE НА ЯЗЫКАХ ВЫСОКОГО УРОВНЯ VIVADO HIGH-LEVEL SINTHESYS

Черепов А.А.
Научный руководитель: МальчуковА.Н.
Томский политехнический университет
e-mail: aac26@tpu.ru

Введение

ПЛИС –разновидность интегральных схем, логика работы которых определяется не при изготовлении, а задается посредством программирования. Современные ПЛИС содержат до двух миллионов логических ячеек, которые могут быть сконфигурированы для задания множества алгоритмов. При этом скорость работы алгоритма на ПЛИС сравнима со скоростью работы этого же алгоритма на обычной ИС, однако важным преимуществом ПЛИС является возможность динамического изменения структуры логических ячеек.

Практическое использование ПЛИС часто вызывает трудности для «чистых» программистов, которые сталкиваются с целым рядом непривычных задач: необходимостью помнить о правильном формировании тактовых сигналов, учитывать латентность, а также вообще понимать, что операторы языков описания аппаратуры не вполне эквивалентны операторам языков программирования.

САПР VivadoHLS, предназначена для создания цифровых устройств с применением языков высокого уровня.С помощью данной среды разработки возможно написать программу, или перенести уже готовый алгоритм для его дальнейшего распараллеливания и конвейеризации, чтобы получить все преимущества от использования ПЛИС.

Важным преимуществом данного ПО является то, что инженеры, которые разрабатывают программы для ПЛИС, смогут работать на более высоком уровне абстракции, а разработчики прикладного ПО получают возможность реализации аппаратного ускорения вычислений при помощи ПЛИС, затрачивая при этом минимальное количество времени и ресурсов.

Шаги разработки

Разработка HLS проектов подразумевает следующие шаги:

- Создание алгоритма на языке высокого уровня и его проверка при помощи C-симуляции.
- Имплементация данного алгоритма на RTL при помощи C-синтеза и его верификация. Шаги которой показаны на рисунке 1.
- Оптимизация полученного RTL определения при помощи специальных директив.
- Создание нескольких решений, отличающихся быстродействием и количеством занимаемых ресурсов ПЛИС.
- Экспорт данных решений как IP-core для дальнейшего использования.
- Использование ресурсов данного проекта в других проектах.

Каждый проект Vivado HLS содержит в себе следующие компоненты:

- Функцию, написанную на C, C++ или SystemC, которая будет преобразована в RTL
- Ограничения и требования, включающие в себя период тактов, погрешность периода и ПЛИС, для которой создается IP-core. По умолчанию погрешность приравнивается к 12,5% от периода.
- Опциональные директивы, которые определяют особенности функционирования IP-core, такие как интерфесы ввода/вывода, конвейеризацию циклов, ресурсы для размещения данных и т. д.
- Файлы с кодом для тестирования синтезируемой функции и заголовочные файлы

Исходный код в Vivado HLS делится на две части:

- Код, который будет синтезирован в RTL.
- Весь остальной код, который используется для тестирования преобразуемого кода.

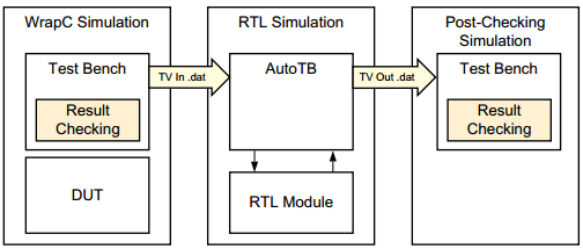

Рис. 1. Шаги RTLверификации.

Для каждого проекта необходимо выбрать одну функцию, которая будет определять использоваться для синтеза. Этой функцией не может быть функция main(). Также будут синтезированы все подфункции, вызываемые основной функцией.

Разработчику предоставляется возможность гибко определять интерфейсы ввода/вывода данных, используемые ресурсы памяти, осуществлять конвейеризацию и распараллеливание при помощи специальных директив.

При запуске среды разработки открывается стартовая страница, изображенная на рисунке 2, которая позволяет создать новый проект, открыть созданные ранее проекты, в том числе проекты с примерами, а также позволяет получить быстрый доступ к различным видам документации.

Рис. 2. Стартовая страница VivadoHLS.

Особенности использования C/C++

Трансляция исходного кода высокого уровня на язык описания аппаратуры накладывает специфические требования и ограничения на использование языковых конструкций. Также вместе с VivadoHLSпоставляется набор библиотек для языков C/C++/SystemC, содержащие в себе ряд полезных функций и типов, таких как: знаковые/беззнаковые целые числа произвольной разрядности, дробные числа с фиксированной точкой произвольной разрядности, библиотеки для работы с потоками, мат. функциями, для обработки видео и т.д.

Программисту при использовании VivadoHLSжелательно иметь хотя бы общее представление об интерфейсах взаимодействия между отдельными блоками RTL, и знать об AXI стандартах, handshakeпротоколе и т.д.

Рассмотрим, как создаются интерфейсы при преобразовании функции на языке Си в RTL блок на примере простейшей функции.

```
dout_t sum_io
(din_t in1, din_t in2, dio_t *sum)
{
    dout_t temp;
    *sum = in1 + in2 + *sum;
    temp = in1 + in2;
    return temp;
}
```

Результат синтеза примера в RTLблок показан на рисунке 3.

Этот пример содержит в себе:

- Два входных аргумента, передаваемых напрямую.
- Указатель *sum*позволяющий читать и записывать данные.
- Возвращаемую посредством *return*переменную *temp*.

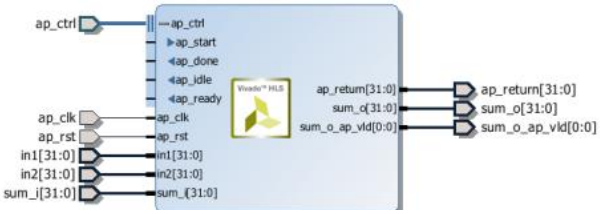

Рис. 3. RTLблок, полученный в результате синтеза примера.

По умолчанию VivadoHLScоздает порты ap_clkи ap_rst для тактирования и сброса соответственно. Также для реализации протокола блочного уровня создается шина ap_ctrl, содержащая в себе сигналы для индикации о занятости, готовности, завершении работы и сигнал старта.

Также VavadoHLS по умолчаниюсоздает порты для переменных, которые объявлены как аргументы функции. Программист может при помощи специальных директив назначать аргументам функции конкретный тип порта ввода или вывода данных. При этом для разных типов входных данных доступны разные типы портов. Все входные данные разделяются на три типа: скалярный, массив, указатель или ссылка. При этом как порты ввода/вывода могут использоваться только массивы или указатели/ссылки. Скалярный тип данных определяется как порт только для ввода или только для вывода данных. Таблицу с соответствием интерфейсов типам входных переменных можно найти в документе [2].

Заключение

Vivado HLS является мощным инструментом, который позволяет в кратчайшие сроки разработать IP-Core прикладному программисту для ускорения вычисления своих алгоритмов на ПЛИС.

Списоклитературы:

1. Introduction to FPGA Design with Vivado High-Level Synthesis(UG998). – 89 p

2. Vivado Design Suite User Guide High-Level Synthesis UG902(v2015.4) November 24, 2015. – 671 p

3. Vivado High-Level Synthesis // Xilinx Inc // URL: https://www.xilinx.com/products/design-tools/vivado/integration/esl-design.html // Датаобращения: 10.10.16

4. Getting started with Vivado High-Level Synthesis // Xilinx Inc. // URL: http://www.xilinx.com/video/hardware/getting-started-vivado-high-level-synthesis.html// `Датаобращения 10.10.16