Лабораторная работа №1.

Моделирование комбинационных и последовательных схем на языке Verilog. Симуляция и тестирование схем.

Цель лабораторной работы.

Цель лабораторной работы освоить базовые конструкции языка Verilog. Научиться моделировать простые устройства, научиться проводить симуляцию, тестировать и верифицировать смоделированные устройства.

Лабораторная работа состоит из нескольких этапов. В рамках лабораторной работы все студенты выполняют одинаковый набор заданий. При этом устройство, для которого производится моделирование и тестирование у каждого студента свое (свой вариант).

Далее, идет описание этапов лабораторной работы, а после указаны варианты устройств. При этом распределение студентов по вариантам указано в отдельном файле.

Любые вопросы по лабораторной работе отправлять по электронной почте на адрес mikle.shupletsov@gmail.com. Тема письма должна иметь следующий формат: [318][VLSI] Тема письма или вопрос.

Этапы лабораторной работы

Этап 1. Поведенческое (автоматное) моделирование.

Необходимо создать поведенческое (автоматное описание) на языке Verilog устройства для которого дано формальное описание (см. ниже). Формальное описание включает в себя описание входов и выходов моделируемого устройства, а также формальное поведение (функцию или алгоритм, реализуемый устройством).

Этап 2. Логическое (функциональное) моделирование.

По описанию, построенному на предыдущем этапе необходимо создать gate-level описание на языке Verilog. При этом разрешается использовать только следующие стандартные логические элементы языка Verilog: NAND, NOR, AND, OR, XOR, XNOR, BUF, NOT. Элементы единичной задержки (регистры) можно моделировать любым доступным в языке Verilog способом.

Этап 3. Тестирование поведенческого описания.

Необходимо создать тестовый «стенд» (testbench) на языке Verilog, который позволяет протестировать работу смоделированного устройства в разных режимах (сценариях работы). Требуется написать Verilog модель, которая генерирует различные сценарии работы устройства и проверяют корректность работы полученного на первом этапе описания.

Этап 4. Верификация логического описания.

Тестирование логического описания производится при помощи сравнения с поведенческим описанием. На данном этапе требуется построить тестовый «стенд» (testbench), который для заданного набора сценариев симулирует работу и поведенческого и функционального описаний и сравнивает результаты их работы. Если описания эквивалентны, то результаты их работы должны совпадать. На данном этапе требуется написать генератор случайных входных сценариев для проверки описаний.

Этап 5 (дополнительный). Моделирование устройства с учетом задержек.

На данном этапе требуется так модифицировать логическое описание устройство, чтобы оно корректно работало при заданных предположениях о задержках логических элементах и при заданных параметрах тактового генератора.

Предполагаются следующие задержки для логических элементов (одинаковые для всех вариантов):

NOT – 1;

BUF – 2;

NAND – 2;

NOR – 2;

AND – 3;

OR – 3

XOR – 4;

XNOR -4;

Считается, что модель задержки упрощенная, то есть любой переход требует одинаковое количество тактов «глобальных» часов. Кроме того, если элемент единичной задержки (например, D-flipflop) моделируется при помощи стандартных логических элементов, то у него тоже будет некоторая задержка, выраженная в тактах «глобальных» часов.

Тактовый генератор переключается каждые 5 тактов «глобальных часов».

Если логика моделируемого устройства слишком сложная и не все логические элементы «успевают» переключиться за один такт, то работу устройства нужно разделить на несколько тактов, как иногда говорят, добавить регистровые базы.

Варианты моделируемых устройств.

Обязательные входы.

Во всех вариантах предполагается наличие следующих обязательных входов:

1. clock – вход тактового генератора.

2. reset – бит сброса. Когда на указанном входе значение «1», то значение на всех выходах устройства сбрасывается в нулевое значение. Задержка сброса не регламентируется.

3. enable – бит включения устройства. Когда на данном входе значение «1», то устройство работает, когда значение «0», то считается, что устройство выключено.

Вариант 1. Числа Фибоначчи.

По порядковому номеру вычислить двоичное представление соответствующего члена числового ряда Фибоначчи.

Вход: 3-х битовый провод, на который передается порядковый номер числа Фибоначчи.
Выход: 4-х битовый регистр, в котором сохраняется соответствующее число Фибоначчи.

Вариант 2. Блок задержки на переменное число тактов.

Блок содержит два входа: однобитовый провод start и 3-битовый провод count, а также один выход: однобитовый регистр done. Если устройство включено и на входе start подается 0, то устройство находится в состоянии покоя и на выход done подает значение 0. Когда на вход start подается значение 1, то устройство отсчитывает count тактов (подсчет ведется по положительному фронту переключения тактового генератора) и после этого выдает на выходе done значение 1.

Вариант 3. Элементарное арифметическое логическое устройство.

Схема получает на вход два целых числа, и код операции. Далее схема выполняет соответствующую операцию, и результат поступает на выход схемы. Устройство поддерживает следующие операции: сложение, вычитание. Устройство должно корректно обрабатывать возникающие переполнения.

Вход: два 4-х разрядных провода, на которые подаются значения операндов и одноразрядный провод op, на который подается код операции (0 – сложение, 1 - вычитание).

Выход: 4-х разрядный регистр выходного операнда и однобитовый регистр переполнения (overflow).

Вариант 4. Код Грея.

Схема получает на вход двоичное число (лексикографический порядок) и кодирует его в коде Грея. Закодированное число поступает на выход.

Вход: 4-х битовый провод.

Выход: 4-х битовый регистр.

Вариант 5. Блок вычисления квадрата числа.

По целому числу, записанному в двоичной системе исчисления, вычисляет квадрат этого числа и записывается в двоичной системе исчисления.

Вход: 3-х битовый провод.

Выход: 6-ти разрядный регистр.

Вариант 6. Уникальное число в памяти.

Пусть задан массив из 9 ячеек памяти. Каждая ячейка памяти представляет собой 4-х битовый регистр. Предполагается, что массив ячеек памяти заполнен так, что восемь ячеек памяти содержат одинаковые числа, а оставшийся регистр содержит число, отличное от всех остальных. Построить схему, которая находит это число. Если вход схемы не соответствует описанию, то значения на выходах схемы не регламентируются.

Вход: 36-битовый регистр, представляющий массив ячеек памяти.

Выход: 4-х битовый регистр, в котором сохраняется уникальное число.

Вариант 7. Вектор коэффициентов полинома Жегалкина.

Входной вектор, представляющий вектор значений булевой функции при лексикографическом упорядочивании наборов, необходимо перевести в вектор коэффициентов полинома Жегалкина при лексикографическом упорядочивании коэффициентов.

Вход: 8-ми битовый провод, на который подается вектор значений булевой функции.

Выход 8-ми битовый регистр для хранения результирующего вектора коэффициентов полинома Жегалкина.