МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
УНИВЕРСИТЕТ ИТМО

П. В. Кустарев, С. В. Быковский

Функциональная схемотехника

Методические указания к лабораторной работе №2

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург

2019

Содержание

Лабораторная работа №2. «Разработка арифметических блоков на RTL-уровне»

1.1 Цель работы

Получить навыки описания арифметических блоков на RTL-уровне с использованием языка описания аппаратуры Verilog HDL.

1.2 Указания к выполнению работы

Лабораторная работа посвящена знакомству с техниками описания схем арифметических блоков на RTL уровне с использованием языка Verilog HDL. Работа выполняется в Vivado Design Suite.

По результатам выполнения лабораторной работы составляется отчет в соответствии с требованиями, приведенными в разделе «Требования к оформлению отчета».

1.3 Порядок выполнения

1. Разработайте и опишите на Verilog HDL схему, вычисляющую значение функции в соответствии с заданными ограничениями согласно варианту задания.
2. Определите диапазон допустимых значений операндов.
3. Разработайте тестовое окружение для разработанной схемы. Тестовое окружение должно проверять схему на значениях внутри диапазона допустимых значений (не менее 10 значений), а также на границах.
4. Проведите моделирование работы схемы и определите время вычисления результата. Схема должна тактироваться от сигнала с частотой 100 МГц.
5. Составьте отчет по результатам выполнения работы.

1.4 Задания по вариантам

№ Варианта	Функция	Ограничения	Формат чисел
1	$y = \sqrt[3]{x}$	1 сумматор и 2 умножителя	Беззнаковые числа 8-бит
2	$y = \sqrt[3]{x}$	2 сумматора и 1 умножитель	Беззнаковые числа 8-бит
3	$y = \sqrt{a^2 + b^2}$	1 сумматор и 2 умножителя	Операнды - беззнаковые числа 8-бит Результат - беззнаковое число 16-бит
4	$y = \sqrt{a^2 + b^2}$	2 сумматора и 1 умножитель	Операнды - беззнаковые числа 8-бит Результат - беззнаковое число 16-бит
5	$y = x - \frac{x^3}{3!} + \frac{x^5}{5!}$	1 сумматор и 2 умножителя	Знаковые числа 8-бит с фиксированной точкой
6	$y = x - \frac{x^3}{3!} + \frac{x^5}{5!}$	2 сумматора и 1 умножитель	Знаковые числа 8-бит с фиксированной точкой
7	$y = x - \frac{x^3}{3!} + \frac{x^5}{5!}$	2 сумматора и 2 умножителя	Знаковые числа 8-бит с фиксированной точкой
8	$y = 1 - \frac{x^2}{2!} + \frac{x^4}{4!}$	1 сумматор и 2 умножителя	Знаковые числа 8-бит с фиксированной точкой
9	$y = 1 - \frac{x^2}{2!} + \frac{x^4}{4!}$	2 сумматора и 1 умножитель	Знаковые числа 8-бит с фиксированной точкой
10	$y = 1 - \frac{x^2}{2!} + \frac{x^4}{4!}$	2 сумматора и 2 умножителя	Знаковые числа 8-бит с фиксированной точкой

1.5 Комментарии к заданиям

Ограничения накладываются на количество используемых блоков суммирования и умножения. В разработанной схеме должен быть использован блок умножения, реализующий последовательных алгоритм умножения «в столбик». В качестве основы можно взять реализацию, представленную в разделе 3.3.

Сумматор реализуется с помощью встроенного в Verilog оператора суммы «+». Сдвиги реализуются также с помощью встроенных операторов сдвига «<<», «>>».

Для вычисления квадратного и кубического корня могут использоваться алгоритмы, представленные в разделе 3.

В формате числа с фиксированной точкой записывается только дробная часть. Целая часть равна 0 и не записывается. Например, число в двоичной системе счисления 0.1010000_2 равно 0.625_{10} в десятичной. Ноль перед точкой не записывается в формате числа. Старший разряд отводится под знак.

Требования к оформлению отчета

Отчет выполняется в виде самодостаточного документа. Материал, изложенный в отчете, должен быть понятным без дополнительных комментариев со стороны исполнителей.

На защиту предоставляется только распечатанный титульный лист. Электронная версия отчета высылается на почту преподавателю.

Отчет должен содержать:

- Титульный лист, на котором указываются:
  - название университета;
  - название факультета;
  - название дисциплины;
  - номер и тема лабораторной работы;
  - вариант лабораторной работы;
  - фамилия, инициалы и номер группы каждого исполнителя;
  - фамилия и инициалы преподавателя;
  - текущий год.
- Содержание.
- Цель работы.
- Задание в соответствии с вариантом.
- Схема (рисунок) разработанного блока вычисления функции, заданной вариантом, в терминах базовых операционных элементов (БОЭ) (т.е. используя мультиплексоры/демультиплексоры, шифраторы/дешифраторы, компараторы, регистры, счетчики и др.).
- Описание работы разработанного блока, начиная от подачи входных данных и заканчивая получением результата.

- Область допустимых входных значений и диапазон выходных значений для разработанного блока.
- Результат тестирования (временные диаграммы) разработанного блока на значениях внутри диапазона допустимых значений (не менее 10 значений) и на границах. Демонстрируются факты корректного и некорректного вычисления результата.
- Время вычисления результата при тактовом сигнале в 100МГц.

Требования к оформлению:

- отчет выполняется как текстовый документ в соответствии с ГОСТ 2.105-95;
- шрифт Times New Roman 12-14 pt, межстрочный интервал 1-1,5, поля с краев листа – не менее 2 см;
- сквозная нумерация страниц;
- обязательны нумерация и подписи к рисункам и таблицам, а также ссылки на них в тексте отчета;
- схемы и временные диаграммы должны быть темными на светлом фоне; если наложение временных диаграмм нескольких сигналов мешает их однозначному восприятию, они должны разноситься на отдельные координатные сетки;
- в распечатанном отчете линии на схемах и временных диаграммах должны быть четко видны.

Алгоритмы машинной арифметики

3.1 Извлечение квадратного корня

В данном разделе представлен целочисленный алгоритм извлечения квадратного корня с округлением до ближайшего меньшего целого числа.

$$y = \lfloor \sqrt{a} \rfloor$$

Округление до ближайшего меньшего числа означает то, что, если мы извлекаем квадратный корень из числа, для которого квадратный корень не является целым, то в качестве результата берется наименьшая целая часть. Например, при извлечении квадратного корня из 10 получим 3.16 и в качестве результата берем 3.

$$\lfloor \sqrt{10} \rfloor = \lfloor 3.16 \rfloor = 3$$

Синтезируемый алгоритм вычисления квадратного корня с использованием только операций сложения, вычитания, сравнения и сдвига, написанный на языке python (функция **sqrt_hw**), представлен в листинге 3.1. В листинге описан алгоритм вычисления корня из 8-разрядного целого числа.

```
import math

def sqrt_alg(x):
    return math.floor(math.sqrt(x))

def sqrt_hw(x):
    m = 0x40
    y = 0

    while (m != 0):
        b = y | m
        y >>= 1

        if (x >= b):
            x -= b
            y |= m
```

```
        m >>= 2

    return y

# Test
print("Start stimulus generation")
for x in range(0, 10, 1):
    a = x*x
    alg_val = sqrt_alg(a)
    hw_val = sqrt_hw(a)

    if (alg_val == hw_val):
        print("Correct! x: ", str(a).ljust(6), "; y: ", hex(hw_val).ljust
            (6))
    else:
        print("ERROR! x: ", hex(x).ljust(6), "; y(model): ",
        hex(alg_val).ljust(6), "; y(hw): ", hex(hw_val).ljust(6))
```

Листинг 3.1: Синтезируемый алгоритм вычисления квадратного корня на языке python

3.2 Извлечение кубического корня

В данном разделе представлен целочисленный алгоритм извлечения кубического корня с округлением до ближайшего меньшего целого числа.

$$y = \lfloor \sqrt[3]{a} \rfloor$$

Синтезируемый алгоритм вычисления кубического корня с использованием только операций умножения, сложения, вычитания, сравнения и сдвига, написанный на языке python (функция **cube_hw**), представлен в листинге 3.2.

```
import math
import numpy

def cube_alg(x):
    return math.floor(numpy.cbrt(x))

def cube_hw(x):
    y = 0;
    for s in range(30, -3, -3):
        y = 2*y;
        b = (3*y*(y + 1) + 1) << s;
        if (x >= b):
```

```
            x = x - b;
            y = y + 1;
    return y;

# Test
for x in range(0, 5, 1):
    a = pow(x, 3)
    alg_val = cube_alg(a)
    hw_val = cube_hw(a)
    if (alg_val == hw_val):
        print("Correct! x: ", str(a).ljust(6), "; y: ", str(hw_val).ljust
            (6))
    else:
        print("ERROR! x: ", str(a).ljust(6), "; y(model): ", str(alg_val).
            ljust(6), "; y(hw): ", hex(hw_val).ljust(6))
```

Листинг 3.2: Синтезируемый алгоритм вычисления кубического корня на языке python

3.3 Алгоритм умножения

Реализация алгоритма умножения «в столбик» на языке Verilog HDL представлена в листинге 3.3.

```
module mult (
    input clk_i,
    input rst_i,

    input [7:0] a_bi,
    input [7:0] b_bi,
    input start_i,

    output busy_o,
    output reg [15:0] y_bo
);

    localparam IDLE = 1'b0;
    localparam WORK = 1'b1;

    reg  [2:0]  ctr;
    wire [2:0]  end_step;
    wire [7:0]  part_sum;
    wire [15:0] shifted_part_sum;
    reg [7:0]   a, b;
```

```
    reg [15:0]   part_res;
    reg state;

    assign part_sum = a & {8{b[ctr]}};
    assign shifted_part_sum = part_sum << ctr;
    assign end_step = (ctr == 3'h7);
    assign busy_o = state;

    always@(posedge clk_i)
        if(rst_i) begin
            ctr      <= 0;
            part_res <= 0;
            y_bo     <= 0;

            state <= IDLE;
        end else begin

            case(state)
                IDLE:
                    if(start_i) begin
                        state  <= WORK;

                        a        <= a_bi;
                        b        <= b_bi;
                        ctr      <= 0;
                        part_res <= 0;
                    end
                WORK:
                    begin
                        if(end_step) begin
                            state  <= IDLE;
                            y_bo   <= part_res;
                        end

                        part_res = part_res + shifted_part_sum;
                        ctr <= ctr + 1;
                    end
            endcase
        end

endmodule
```

Листинг 3.3: Описание блока умножения на Verilog HDL