|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

Институт Информационных Технологий

Кафедра Вычислительной Техники (ВТ)

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №** 7

«Реализация конечных автоматов, заданных автоматным графом»

по дисциплине

«Архитектура вычислительных машин и систем»

|  |  |
| --- | --- |
| Выполнил студент группы  ИВБО-11-23 | Туктаров Т.А |
| Принял ассистент кафедры ВТ | Дуксина И.И. |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_2025 г. |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_2025 г. |

Москва 2025

АННОТАЦИЯ

Данная работа включает в себя 1 рисунок, 3 листинга. Количество страниц в работе — 12.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc181462601)

[1 ПОСТАНОВКА ЗАДАЧИ 5](#_Toc181462602)

[2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ 6](#_Toc181462603)

[2.1 Реализация автомата Мили 6](#_Toc181462604)

[2.2 Реализация автомата Мура 6](#_Toc181462605)

[3 ВЕРИФИКАЦИЯ 9](#_Toc181462606)

[ЗАКЛЮЧЕНИЕ 10](#_Toc181462607)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 11](#_Toc181462608)

ВВЕДЕНИЕ

Verilog HDL — это язык описания аппаратуры, используемый для описания и моделирования электронных систем. Verilog был создан Филом Мурби (Phil Moorby) и Прабху Гоэлем (Prabhu Goel) зимой 1983–1984 годов в фирме Automated Integrated Design Systems (с 1985 года — Gateway Design Automation) как язык моделирования аппаратуры. Verilog HDL наиболее часто используется в проектировании, верификации и реализации (например, в виде СБИС) аналоговых, цифровых и смешанных электронных систем на различных уровнях абстракции.

Автомат Мили — синхронный автомат, у которого вход и выход не развязаны во времени, т.е. хотя бы один выход зависит от текущего значения на входе.

Автомат Мура — синхронный автомат, у которого выходные значения определяются только состоянием автомата в тот же дискретный момент времени. При этом комбинационная схема, вычисляющая выходные значения, не связана непосредственно с входными сигналами.

# 1 ПОСТАНОВКА ЗАДАЧИ

Задание: Спроектировать модуль, описывающий автомат Мили согласно персональному варианту, выданному преподавателем при помощи Verilog HDL. Произвести преобразование автомата Мили в эквивалентный ему автомат Мура. Создать модуль описывающий автомат Мура при помощи Verilog HDL. Произвести верификацию полученных устройства средствами САПР Vivado.

Персональный вариант (Рисунок 1.1):

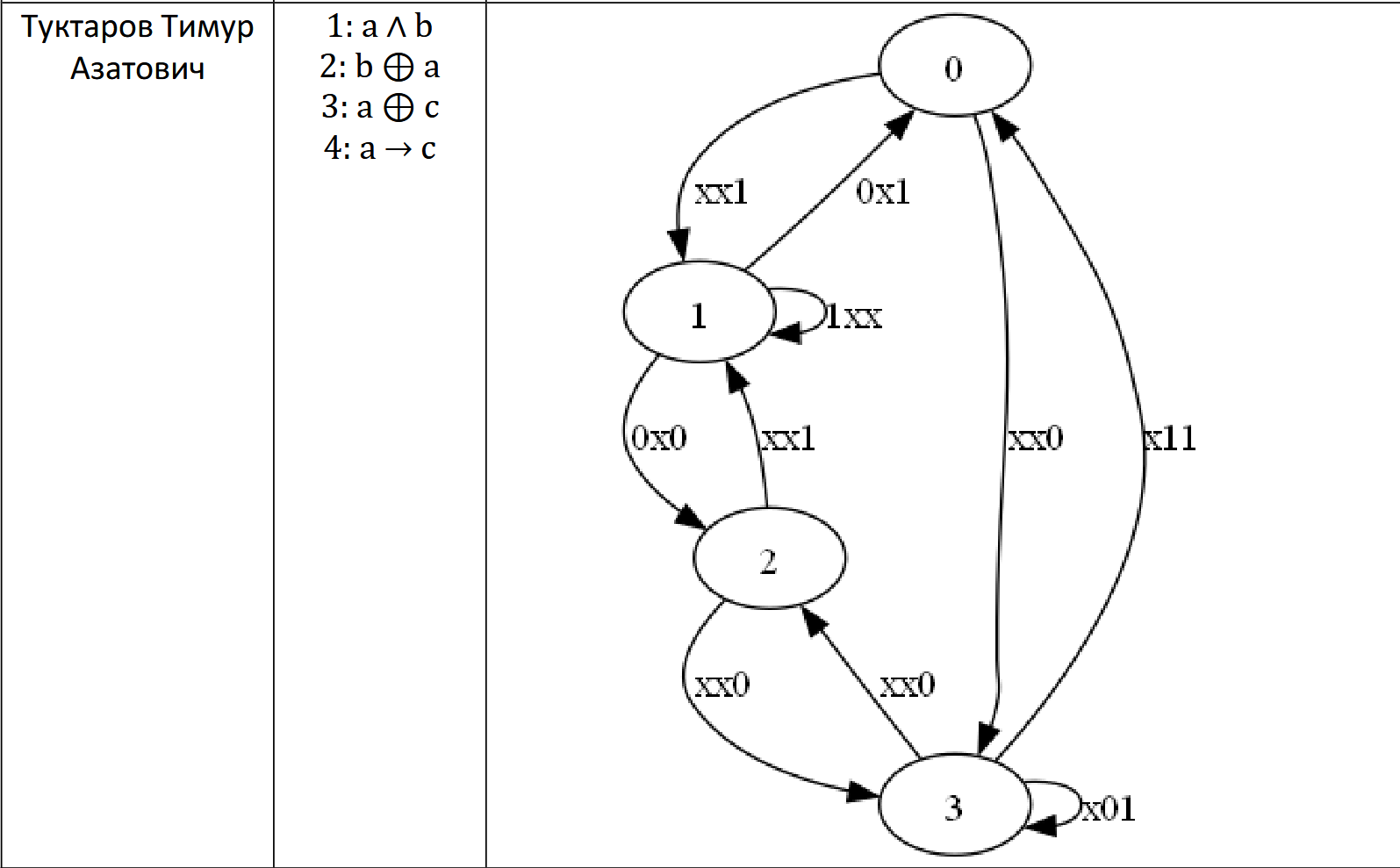


Рисунок 1.1 – Персональный вариант

# 2 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ

## Реализация автомата Мили

При помощи языка описания аппаратуры Verilog средствами САПР Vivado реализуем автомат Мили. Реализация представлена в Листинге 2.1:

Листинг 2.1 — Модуль mili.v

|  |
| --- |
| `timescale 1ns / 1ps  module mili(  input clk, a, b, c,  output reg [1:0] state,  output reg d  );  initial begin  state = 0;  end  always@(posedge clk)  begin  case (state)  2'd0:begin  if(c)state <= 2'd1;  else if(~c)state <= 2'd3;  d <= a & b;  end  2'd1:begin  if(a)state <= 2'd1;  else if(~a && ~c)state <= 2'd2;  else if(~a && c)state <= 2'd0;  d = (~a & b) | (a & ~b);  end  2'd2:begin  if(~c)state <= 2'd3;  else if(c)state <= 2'd1;  d = (~a & c) | (a & ~c);  end  2'd3:begin  if(~c)state <= 2'd2;  else if(b && c)state <= 2'd0;  else if(~b && c)state <= 2'd3;  d = ~a | c;  end  endcase  end  endmodule |

## Реализация автомата Мура

Для реализации автомата Мура, эквивалентного автомату Мили необходимо составить автоматную таблицу для автомата Мили и автомата Мура.

Автоматная таблица для автомата Мили представлена в Таблице 2.1.

Таблица 2.1 — Автоматная таблица автомата Мили

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Состояние/входы | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| S0 | S3 0 | S1 0 | S3 0 | S1 0 | S3 0 | S1 1 | S3 0 | S1 1 |
| S1 | S2 0 | S0 0 | S2 1 | S0 1 | S1 1 | S1 1 | S1 0 | S1 0 |
| S2 | S3 0 | S1 1 | S3 0 | S1 1 | S3 1 | S1 0 | S3 1 | S1 0 |
| S3 | S2 1 | S3 1 | S2 1 | S0 1 | S2 0 | S3 1 | S2 0 | S0 1 |

Автоматная таблица для автомата Мура представлена в Таблице 2.2.

Таблица 2.2 — Автоматная таблица автомата Мура

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Состояние/входы | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
| S0 0 | S3 0 | S1 0 | S3 0 | S1 0 | S3 0 | S1 1 | S3 0 | S1 1 |
| S0 1 | S3 0 | S1 0 | S3 0 | S1 0 | S3 0 | S1 1 | S3 0 | S1 1 |
| S1 0 | S2 0 | S0 0 | S2 1 | S0 1 | S1 1 | S1 1 | S1 0 | S1 0 |
| S1 1 | S2 0 | S0 0 | S2 1 | S0 1 | S1 1 | S1 1 | S1 0 | S1 0 |
| S2 0 | S3 0 | S1 1 | S3 0 | S1 1 | S3 1 | S1 0 | S3 1 | S1 0 |
| S2 1 | S3 0 | S1 1 | S3 0 | S1 1 | S3 1 | S1 0 | S3 1 | S1 0 |
| S3 0 | S2 1 | S3 1 | S2 1 | S0 1 | S2 0 | S3 1 | S2 0 | S0 1 |
| S3 1 | S2 1 | S3 1 | S2 1 | S0 1 | S2 0 | S3 1 | S2 0 | S0 1 |

При помощи языка описания аппаратуры Verilog средствами САПР Vivado реализуем автомат Мура. Реализация представлена в (Листинге 2.2):

Листинг 2.2 — Модуль Moore

|  |
| --- |
| `timescale 1ns / 1ps  module moore#  (parameter  S0\_0=3'b000, S0\_1=3'b001,  S1\_0=3'b010, S1\_1=3'b011,  S2\_0=3'b100, S2\_1=3'b101,  S3\_0=3'b110, S3\_1=3'b111)  (input clk,a,b,c,  output reg [2:0] state,  output reg d);  reg [2:0] new\_state=0;  initial begin  state = 0;  end  always @(posedge clk)  begin  state = new\_state;  casex(state)  3'bxx1: d = 1'b1;  3'bxx0: d = 1'b0;  endcase  casex({state,a,b,c})  {S0\_0, 3'b000}, {S0\_1, 3'b000}: new\_state = S3\_0;  {S0\_0, 3'b001}, {S0\_1, 3'b001}: new\_state = S1\_0;  {S0\_0, 3'b010}, {S0\_1, 3'b010}: new\_state = S3\_0;  {S0\_0, 3'b011}, {S0\_1, 3'b011}: new\_state = S1\_0;  {S0\_0, 3'b100}, {S0\_1, 3'b100}: new\_state = S3\_0;  {S0\_0, 3'b101}, {S0\_1, 3'b101}: new\_state = S1\_1;  {S0\_0, 3'b110}, {S0\_1, 3'b110}: new\_state = S3\_0;  {S0\_0, 3'b111}, {S0\_1, 3'b111}: new\_state = S1\_1;  {S1\_0, 3'b000}, {S1\_1, 3'b000}: new\_state = S2\_0;  {S1\_0, 3'b001}, {S1\_1, 3'b001}: new\_state = S0\_0;  {S1\_0, 3'b010}, {S1\_1, 3'b010}: new\_state = S2\_1;  {S1\_0, 3'b011}, {S1\_1, 3'b011}: new\_state = S0\_1; |

Продолжение листинга 2.2

|  |
| --- |
| {S1\_0, 3'b100}, {S1\_1, 3'b100}: new\_state = S1\_1;  {S1\_0, 3'b101}, {S1\_1, 3'b101}: new\_state = S1\_1;  {S1\_0, 3'b110}, {S1\_1, 3'b110}: new\_state = S1\_0;  {S1\_0, 3'b111}, {S1\_1, 3'b111}: new\_state = S1\_0;  {S2\_0, 3'b000}, {S2\_1, 3'b000}: new\_state = S3\_0;  {S2\_0, 3'b001}, {S2\_1, 3'b001}: new\_state = S1\_1;  {S2\_0, 3'b010}, {S2\_1, 3'b010}: new\_state = S3\_0;  {S2\_0, 3'b011}, {S2\_1, 3'b011}: new\_state = S1\_1;  {S2\_0, 3'b100}, {S2\_1, 3'b100}: new\_state = S3\_1;  {S2\_0, 3'b101}, {S2\_1, 3'b101}: new\_state = S1\_0;  {S2\_0, 3'b110}, {S2\_1, 3'b110}: new\_state = S3\_1;  {S2\_0, 3'b111}, {S2\_1, 3'b111}: new\_state = S1\_0;  {S3\_0, 3'b000}, {S3\_1, 3'b000}: new\_state = S2\_1;  {S3\_0, 3'b001}, {S3\_1, 3'b001}: new\_state = S3\_1;  {S3\_0, 3'b010}, {S3\_1, 3'b010}: new\_state = S2\_1;  {S3\_0, 3'b011}, {S3\_1, 3'b011}: new\_state = S0\_1;  {S3\_0, 3'b100}, {S3\_1, 3'b100}: new\_state = S2\_0;  {S3\_0, 3'b101}, {S3\_1, 3'b101}: new\_state = S3\_1;  {S3\_0, 3'b110}, {S3\_1, 3'b110}: new\_state = S2\_0;  {S3\_0, 3'b111}, {S3\_1, 3'b111}: new\_state = S0\_1;  endcase  end  endmodule |

# 3 ВЕРИФИКАЦИЯ

Реализуем верификацию для реализованных модулей: Mili, Moore, для это создадим модуль testbench.v. Результат представлен в листинге 3.1:

Листинг 3.1 — Модуль testbecnh.v

|  |
| --- |
| `timescale 1ns / 1ps  module testbench();  reg a=0;  reg b=0;  reg c=0;  reg clk=0;  wire [1:0] mili\_state;  wire [2:0] moore\_state;    wire mili\_out;  wire moore\_out;    always #10 clk = ~clk;  initial begin  c = 1; // xx1 (1)  #20 a=0; c=0; // 0x0 (2)  #20 c=0; // xx0 (3)  #20 b=0; c=1; // x01 (3)  #20 c=0; // xx0 (2)  #20 c=1; // xx1 (1)  #20 a=1;// 1xx (1)  #20 a=0; c=1; // 0x1 (0)  #20 c=0;// xx0 (3)  #20 b=1;c=1;// x11 (0)  #50 $stop;  end    mili M1(.clk(clk), .a(a), .b(b), .c(c), .state(mili\_state), .d(mili\_out));  moore M2(.clk(clk), .a(a), .b(b), .c(c), .state(moore\_state), .d(moore\_out));  endmodule |

Результат верификации представлен на Рисунке 3.1.

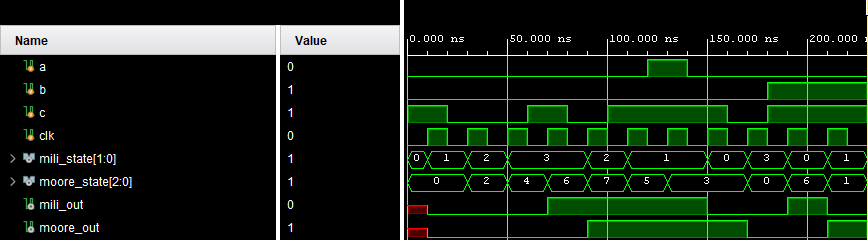


Рисунок 3.1 — Результат верификации созданных схем

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы для заданного варианта были созданы автоматы Мура и Мили при помощи языка описания аппаратуры Verilog средствами САПР Vivado. Произведена верификация полученных схем.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Дуксин, Н. А. Архитектура вычислительных машин и систем. Основы построения вычислительной техники: Практикум : учебное пособие / Н. А. Дуксин, Д. В. Люлява, И. Е. Тарасов. — Москва : РТУ МИРЭА, 2023. — 185 с.

2. Смирнов С.С. Информатика [Электронный ресурс]: Методические указания по выполнению практических и лабораторных работ / С.С. Смирнов — М., МИРЭА — Российский технологический университет, 2018. — 1 электрон. опт. диск (CD-ROM).

*3.* Соловьев В. В. Основы языка проектирования цифровой аппаратуры Verilog. — М.: Горячая линия — Телеком, 2014. — 208 с.