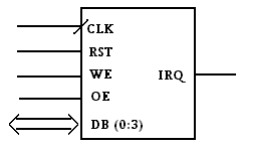
|  |  |
| --- | --- |
|  | |
| МИНОБРНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждениевысшего профессионального образования"Московский технологический университет" **МИРЭА** | |
| Институт информационных технологий (ИТ) | |
| Кафедра вычислительной техники (ВТ) | |

|  |  |  |
| --- | --- | --- |
| **ОТЧЕТ**  **О выполнении лабораторной работы №3** | | |
| **по дисциплине** | | |
| **«**Моделирование**»**  *Вариант 39* | | |
|  | | |
| Выполнил студент группы ИВБО-03-14 | *Кривошея М.С.* | |
| Преподаватель | *Казанцева Л.В.* | |

**Постановка задачи**

Требуется построить цифровой узел с элементами памяти, представляющий собой программируемый генератор последовательности, который выдает процессору серию из 16 прерываний, сопровождаемых статусными комбинациями из последовательности варианта, причем временной интервал между прерываниями в последовательности задается программным способом в виде записи количества тактов синхросигнала. После записи начинается выдача 46 прерываний с заданным интервалом до тех пор, пока не произойдет 16-е прерывание. К устройству предъявляется требование синхронности (т.е. оно должно срабатывать по синхросигналу). Поэтому требуется использовать описание устройства в 1 процессе. Последовательность соответствует таблице заданий. Требуется построить схему при помощи языка описания аппаратуры Verilog.



Интерфейс разрабатываемого устройства

Таблица задания:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **F** | **E** | **D** | **C** | **B** | **A** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| 8 | B | 1 | 3 | 9 | 9 | 6 | 6 | C | F | 5 | 4 | E | 0 | C | E |

**Решение**

**1. ТOP-модуль**

`timescale 1ns / 1ps

module Top(

input CLK,

input RST,

input WE,

input OE,

output reg IRQ,

inout [3:0] DB

);

reg [3:0] T;

reg [3:0] STATE;

reg [3:0] TIMER;

wire [3:0] OUTPUT;

Code CL1(STATE, OUTPUT);

assign DB = OE ? OUTPUT : 4'bZ;

always @ (DB) begin

if(WE)

T <= DB;

end

always @ (posedge RST) begin

STATE <= 4'hF;

TIMER <= 4'h0;

T <= 4'h0;

end

always @ (posedge CLK) begin

IRQ <= 0;

if(!RST && !WE) begin

if(OE) begin

STATE <= STATE + 1;

TIMER <= T;

end else begin

if(TIMER != 0) begin

if(TIMER == 1)

IRQ <= 1;

TIMER <= TIMER - 1;

end

end

end

end

endmodule

**2. Модуль Code.v**

`timescale 1ns / 1ps

module Code(

input [3:0] X,

output reg [3:0] Y

);

always @ (X)

begin

case(X)

4'h0: Y <= 4'h8;

4'h1: Y <= 4'hB;

4'h2: Y <= 4'h1;

4'h3: Y <= 4'h3;

4'h4: Y <= 4'h9;

4'h5: Y <= 4'h9;

4'h6: Y <= 4'h6;

4'h7: Y <= 4'h6;

4'h8: Y <= 4'hC;

4'h9: Y <= 4'hF;

4'hA: Y <= 4'h5;

4'hB: Y <= 4'h4;

4'hC: Y <= 4'hE;

4'hD: Y <= 4'h0;

4'hE: Y <= 4'hC;

4'hF: Y <= 4'hE;

endcase

end

endmodule

**3. Тестовое окружение**

`timescale 1ns / 1ps

module Test;

reg CLK;

reg RST;

reg WE;

reg OE;

wire IRQ;

wire [3:0] DB;

Top uut (

.CLK(CLK),

.RST(RST),

.WE(WE),

.OE(OE),

.IRQ(IRQ),

.DB(DB)

);

always begin

CLK = ~CLK; #50;

end

always begin

OE = 0; #550;

OE = 1; #50;

end

assign DB = (WE && !OE) ? 4'h5 : 4'bZ;

initial begin

CLK = 0;

WE = 0;

OE = 0;

RST = 0; #50;

RST = 1; #50;

RST = 0; #50;

WE = 1; #50;

WE = 0; #150;

end

endmodule

**4. Результаты верификации**

