ГУАП

КАФЕДРА № 44

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Доцент, к.т.н. |  |  |  | В.А. Ненашев |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №4 |
| КОНЕЧНЫЕ АВТОМАТЫ |
| по курсу: СХЕМОТЕХНИКА |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4941 |  |  |  | Н. С. Горбунов |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

**Цель работы**: в данной лабораторной работе требуется разработать последовательностную схему (синхронная логика), реализующую конечный автомат в соответствии с заданием. Провести моделирование разработанной схемы в среде Modelsim Lattice FPGA. Для моделирования создать соответствующий тест.

**Задание по лабораторной работе**

Разработать последовательностную схему (синхронная логика), реализующую конечный автомат в соответствии с заданием.

**Вариант задания:**

|  |  |
| --- | --- |
| 7 | Инкрементировать значение на 9-разрядном выходе до значения, указанного на входе max. По достижении max, производить декремент значения на выходе до значения, указанного на входе min. Затем цикл повторяется. Схема начинает работы со значения max. Значения min и max могут меняться в процессе работы, но не должны влиять на текущий процесс инкремента или декремента |

**Ход выполнения работы**

Листинг 1. Gorbunov4941.v

module state\_machine (  
input clock,  
input reset,  
input [8:0] max,  
input [8:0] min,  
output reg [8:0] out\_pin);  
  
reg flag;  
reg [3:0] state;  
  
localparam s\_flagCheck = 0, s\_inc = 1, s\_dec = 2, s\_maxCheck = 3, s\_minCheck = 4, s\_flagInv = 5;  
  
  
always @(posedge clock) begin  
if (reset) begin  
out\_pin <= max;  
flag <=0;  
state <=0;  
end  
else begin  
case(state)  
s\_flagCheck: begin  
if (flag) state <= s\_inc;  
else state <= s\_dec;  
end  
s\_inc: begin  
out\_pin <= out\_pin + 1;  
state <= s\_maxCheck;  
end  
s\_dec: begin  
out\_pin <= out\_pin - 1;  
state <= s\_minCheck;  
end  
s\_maxCheck: begin  
if (out\_pin == max) state <= s\_flagInv;  
else state <= s\_flagCheck;  
end  
s\_minCheck: begin  
if (out\_pin == min) state <= s\_flagInv;  
else state <= s\_flagCheck;  
end  
s\_flagInv: begin  
flag <= !flag;  
state <= s\_flagCheck;  
end  
endcase  
end  
end  
endmodule

Листинг 2. Test.sv

`timescale 1ns / 1ps  
module test;  
reg t\_clock;  
reg t\_reset;  
reg [8:0] t\_max;  
reg [8:0] t\_min;  
wire [8:0] t\_out\_pin;  
state\_machine uut(  
.clock(t\_clock),  
.reset(t\_reset),  
.max(t\_max),  
.min(t\_min),  
.out\_pin(t\_out\_pin) );  
initial t\_clock = 0;  
initial forever #5  
t\_clock <= !t\_clock;  
initial begin  
t\_max = 50;  
t\_min = 20;  
t\_reset = 1;  
#20;  
t\_reset = 0;  
#1860;  
t\_max = 40;  
t\_min = 25;

end endmodule

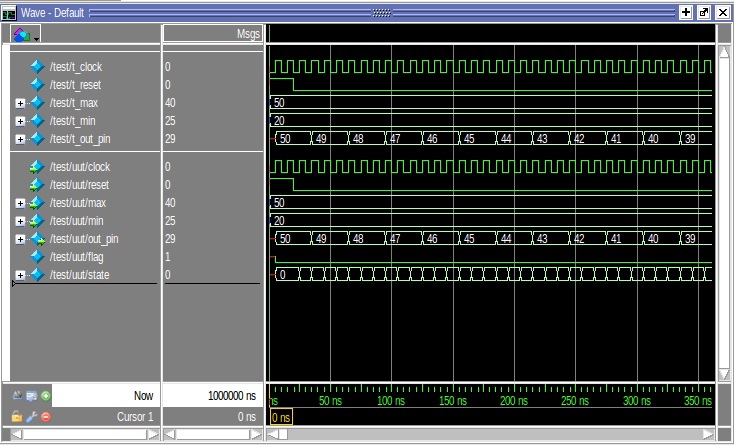


Рисунок 1. Декрементирование от максимума

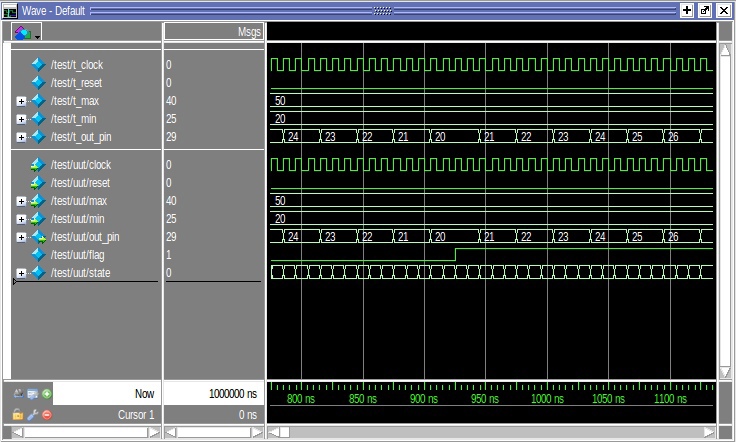


Рисунок 2. Окончание декрементирования на минимуме, начало инкрементирования

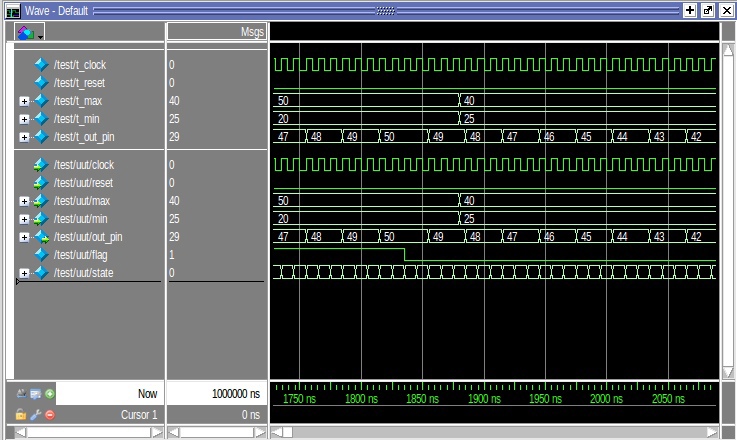


Рисунок 3. Смена минимума и максимума «на лету»

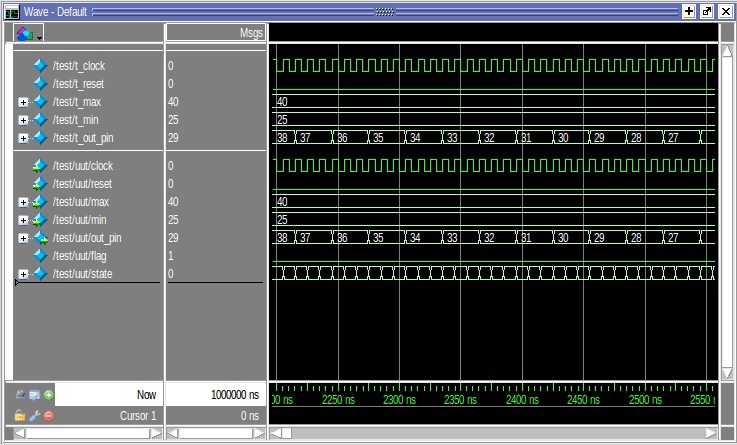


Рисунок 4. Декрементирование до нового минимума



Рисунок 5. Достигнут новый минимум

**Вывод:** разработал последовательностную схему (синхронная логика), реализующую конечный автомат в соответствии с заданием.