МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

**«Вятский государственный университет»**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

РАЗРАБОТКА ФУНКЦИОНАЛЬНОЙ СХЕМЫ ОПЕРАЦИОННОГО АВТОМАТА В САПР QUARTUS

Отчет по лабораторной работе дисциплины

«Теория автоматов»

Выполнил студент группы ИВТ-22 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Родыгин И.А./

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Мельцов В.Ю./

Киров 2017

1 Цель работы

Построить функциональную схему операционного автомата в системе автоматического проектирования Quartus, с целью проверки правильности работы операционного автомата.

2 Постановка задачи

Построить функциональную схему операционного автомата, выполняющего алгоритм умножения 2 способом в двоичной системе счисления в ДК с ПЗ с характеристиками с автоматической коррекцией.

3 Разработка функциональной схемы операционного автомата

Функциональная схема операционного автомата представлена на рисунке 1.

4 Разработка содержательной ГСА

Содержательная ГСА представлена на рисунке 2.

5 Разработка микропрограммы

Листинг микропрограммы приведен в приложении А.

6 Результаты работы

Результаты работы программы представлены на рисунках 3-6.

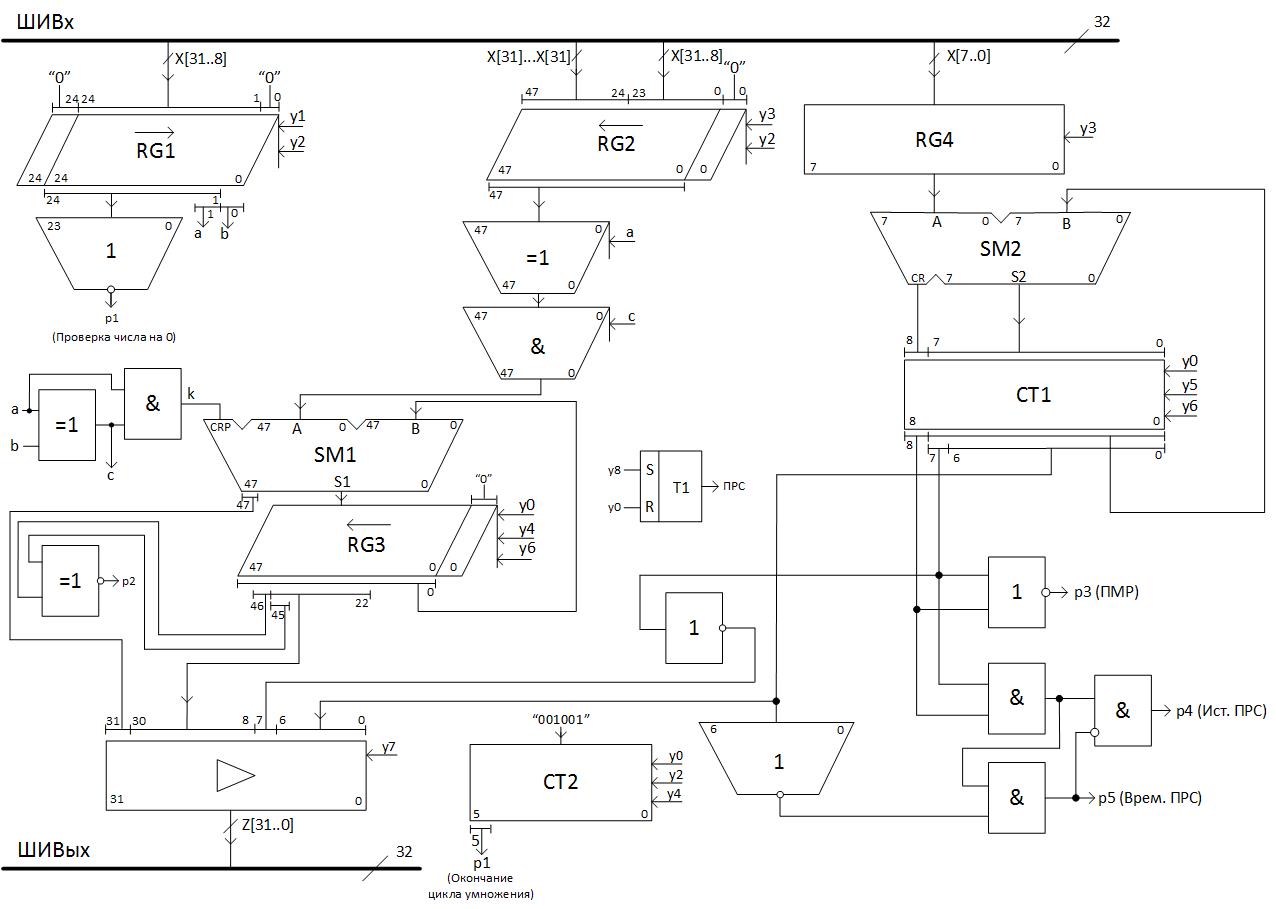


Рисунок 1 – Функциональная схема операционного автомата

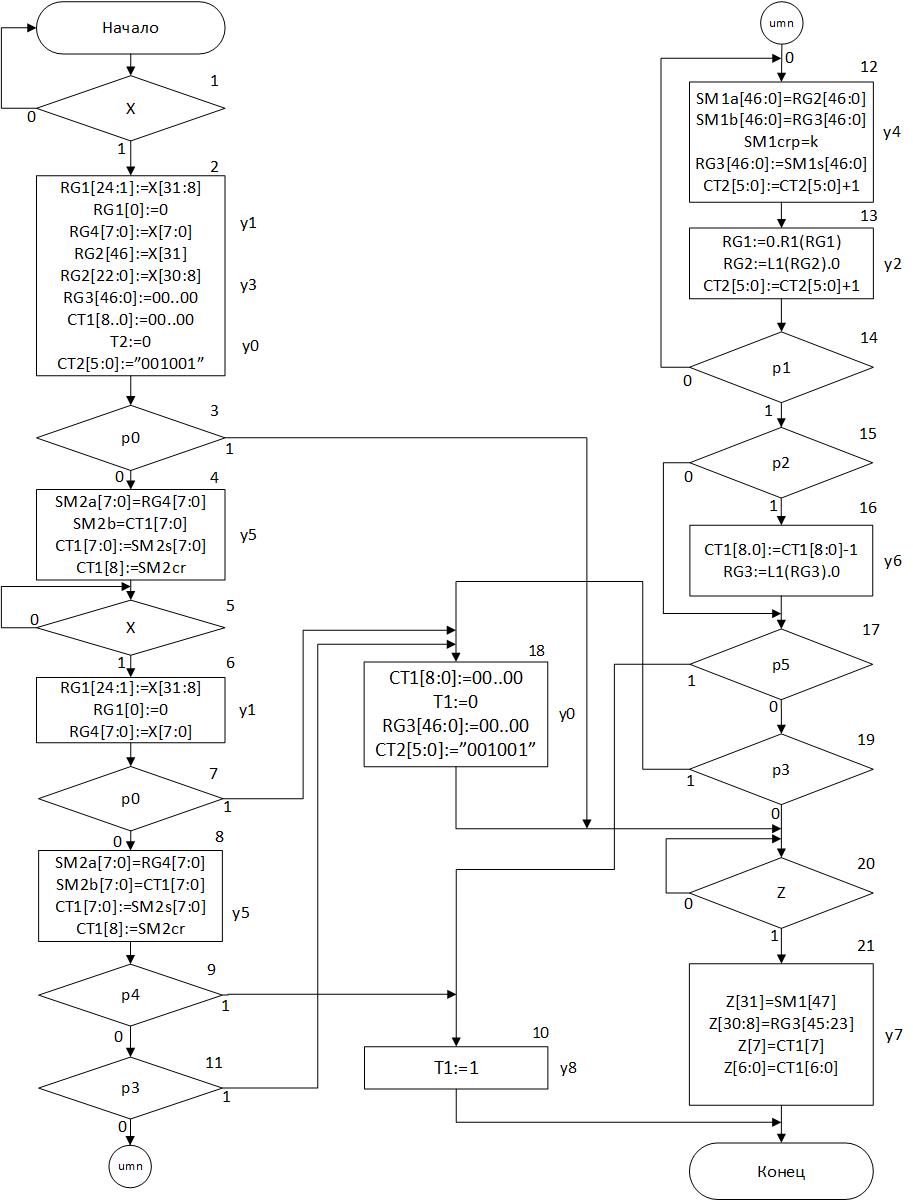


Рисунок 2 – Содержательная ГСА

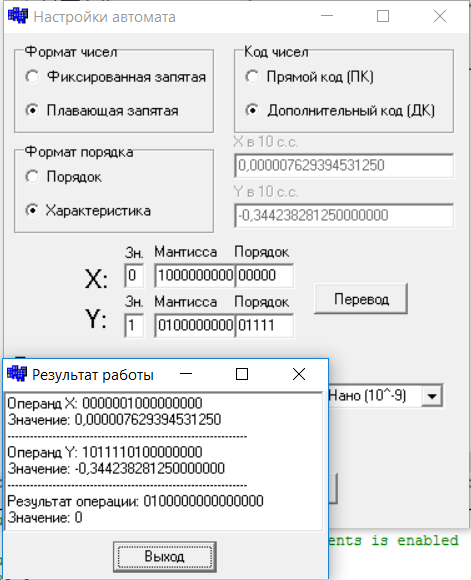


Рисунок 3 – Результат работы программы

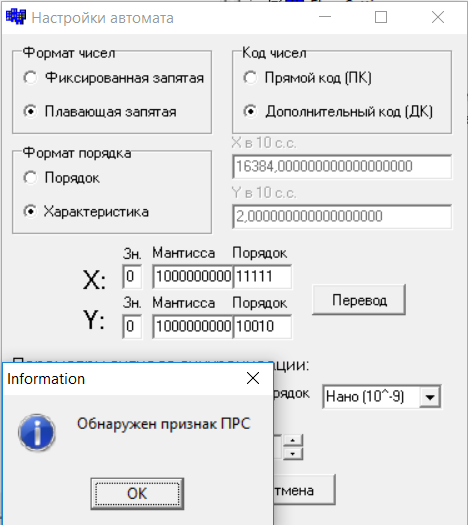


Рисунок 4 – Результат работы программы

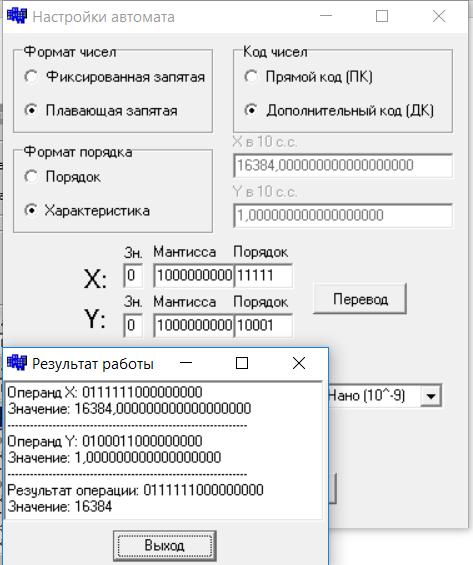


Рисунок 5 – Результат работы программы

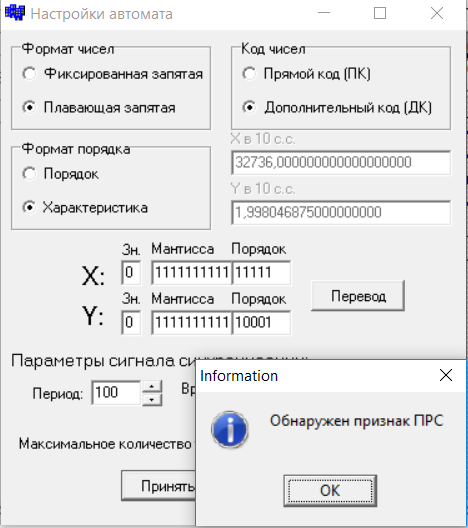


Рисунок 6 – Результат работы программы

Приложение А

(обязательное)

Листинг микропрограммы

// Generated by Quartus II Version 9.1 (Build Build 350 03/24/2010)

// Created on Tue Apr 18 15:19:54 2017

// Module Declaration

module block\_name

(

// {{ALTERA\_ARGS\_BEGIN}} DO NOT REMOVE THIS LINE!

clk, clkout, Z, y, p

// {{ALTERA\_ARGS\_END}} DO NOT REMOVE THIS LINE!

);

// Port Declaration

// {{ALTERA\_IO\_BEGIN}} DO NOT REMOVE THIS LINE!

input clk;

input [5:0] p;

output clkout;

output Z;

output [8:0] y;

// {{ALTERA\_IO\_END}} DO NOT REMOVE THIS LINE!

integer pc=1;//счетчик тактов

reg [8:0] y;

wire clkout;

reg Z=0;

assign clkout=!clk;

always @(posedge clk)

begin

y=9'b000000000;

case(pc)

1:begin

y[1]=1; y[3]=1; y[0]=1;

pc=pc+1;

end

2:begin

if (p[0]==1)

begin

y[0]=1;

Z=1;

pc=50;

end

else

begin

y[5]=1;

pc=pc+1;

end

end

3:begin

y[1]=1;

pc=pc+1;

end

4:begin

if (p[0]==1)

begin

y[0]=1;

Z=1;

pc=50;

end

else

begin

y[5]=1;

pc=pc+1;

end

end

5:begin

if (!p[4]&!p[3])

begin

y[4]=1;

pc=pc+1;

end

else if (!p[4]&p[3])

begin

y[0]=1;

Z=1;

pc=50;

end

else if (p[4]==1)

begin

y[8]=1;

pc=50;

end

end

6:begin

y[2]=1;

pc=pc+1;

end

7:begin

if (p[1]==0)

begin

y[4]=1;

pc=6;

end

else if (p[1]&p[2])

begin

y[6]=1;

pc=pc+1;

end

else if (p[1]&!p[2])

begin

pc=pc+1;

end

end

8:begin

if (p[5]==1)

begin

y[8]=1;

pc=50;

end

else if (!p[5]&p[3])

begin

y[0]=1;

Z=1;

pc=50;

end

else if (!p[5]&!p[3])

begin

Z=1;

y[7]=1;

pc=50;

end

end

endcase;

end

endmodule