МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «вычислительная техника»

**Исследование методов программно-аппаратной реализации**

**алгоритмов управления**

**11 вариант**

Выполнил:

студент группы ИВТАПбд-21

Кондратьев П.С.

Проверила:

Лылова А. В.

Ульяновск, 2017

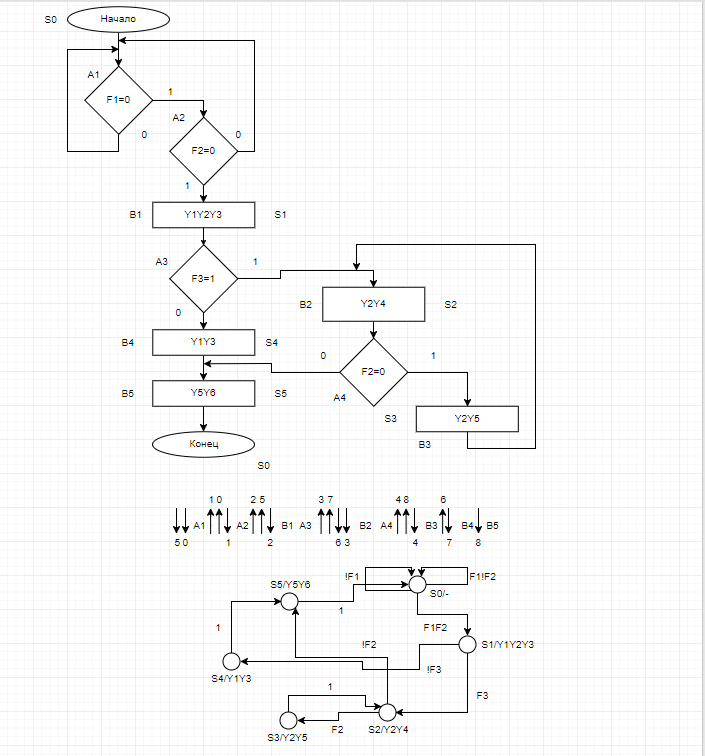
**Техническое Задание:**

1. Проанализировать исходный алгоритм функционирования устройства логического управления (УЛУ).
2. Построить блок-схему исходного алгоритма функционирования УЛУ.
3. Построить ЛСА по полученной блок-схеме работы УЛУ.
4. Построить отмеченную ГСА по полученной блок-схеме работы УЛУ.
5. Построить граф автомата Мура или Мили по отмеченной ГСА, полученной в предыдущем пункте.
6. Минимизировать логические функции формирования входных сигналов УЛУ.
7. Построить схему формирования входных сигналов для аппаратной реализации УЛУ по минимизированным логическим функциям.
8. Разработать аппаратную реализацию управляющего автомата УЛУ (используя канонический метод структурного синтеза).
9. Разработать программную модель УЛУ на любом языке программирования.

**1. Вариант 11:** Мура.

1. *Начало*
2. *Если F1 = 0, идти к 2*
3. *Если F2 = 0, идти к 2*
4. *Y1, Y2, Y3*
5. *Если F3 = 1, идти к 9*
6. *Y1,Y3*
7. *Y5,Y6*
8. *Идти к 13*
9. *Y2,Y4*
10. *Если F2 = 0, идти к 7*
11. *Y2, Y5*
12. *Идти к 9*
13. *Конец*

**2.** Построим блок-схему исходного алгоритма функционирования УЛУ, ЛСА по полученной блок-схеме работы УЛУ, отмеченную ГСА по полученной блок-схеме работы УЛУ, граф автомата Мура или Мили по отмеченной ГСА, полученной в предыдущем пункте.

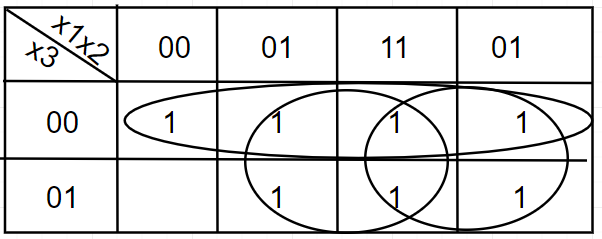


**3.** Минимизируем логические функции формирования входных сигналов УЛУ.

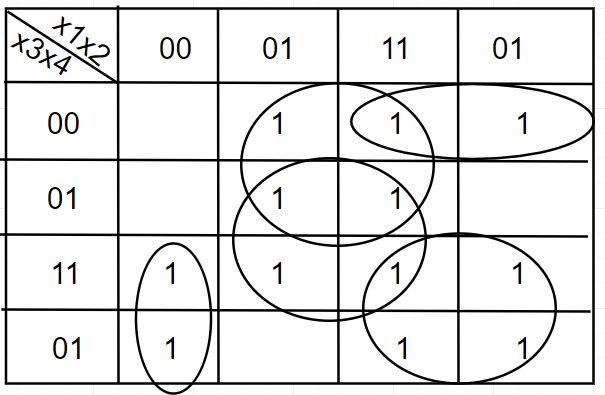
Логические функции формирования выходных сигналов:

* F1 = x1x2 v !x2!x3 v x1!x2x3 v !x1!x3 v !x1x2x3
* F2 = !x1!x2x3 v x1x3 v x2!x3 v x2x4 v x1!x3!x4
* F3 = !x2!x3 v x1x2x3 v x1!x2 v !x1!x2x3 v !x1x2

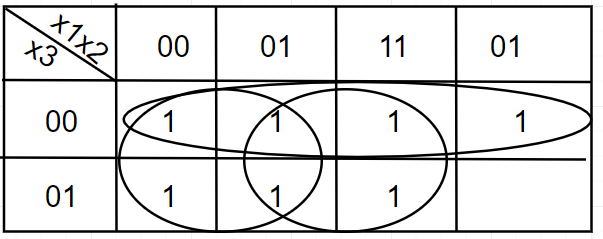
Минимизация производиться с помощью Карт Карно.



F1 = x2 v !x3 v x1

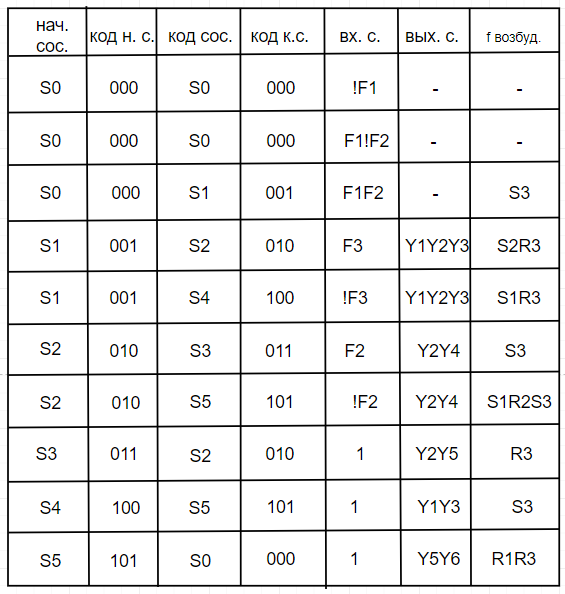


F2 = x1!x3!x4 v x2!x3 v x2x4 x1x3 v !x2x3



F3 = x2 v !x1 v !x3

**4.** Заполним аппаратную реализацию управляющего автомата УЛУ (используя канонический метод структурного синтеза).



Y1 = !α1!α2α3 v α1!α2!α3 Y2 = !α1α2!α3 v !α1α3 Y3 = !α1!α2α3 v α1!α2! α3

Y4 = !α1α2!α3 v α1!α2!α3 Y5 = !α1α2α3 v α1!α2α3 Y6 = α1!α2α3

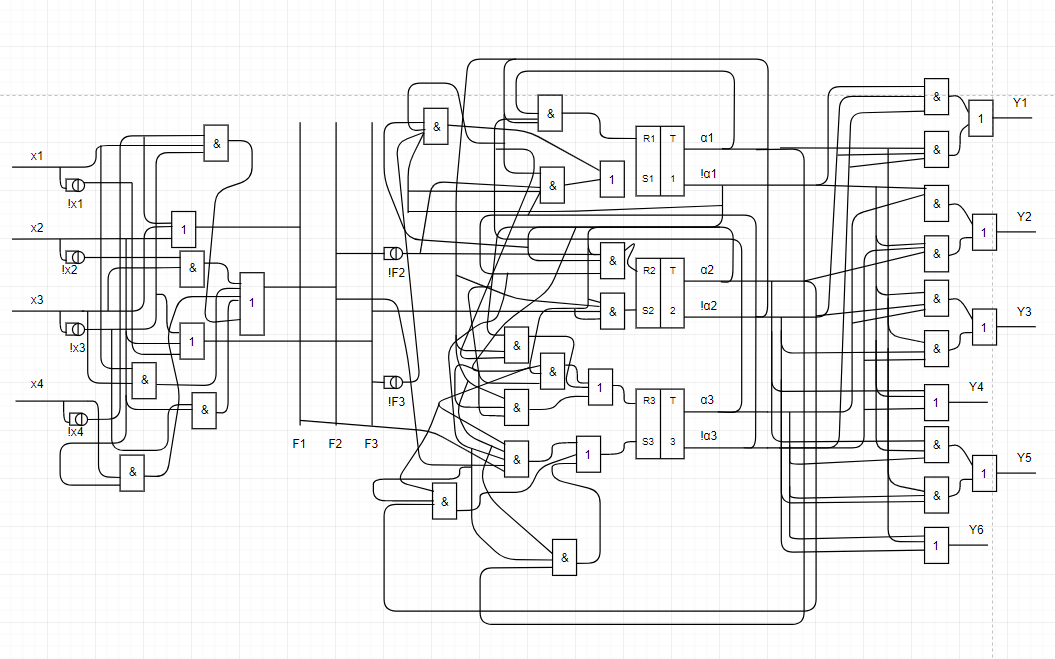
R1 = α1!α2α3 R2 = !α1α2!α3!F2 R3 = !α1!α2α3 v !α1α2α3 v α1!α2α3

S1 = !α1!α2α3!F3 v ! α1α2!α3!F2

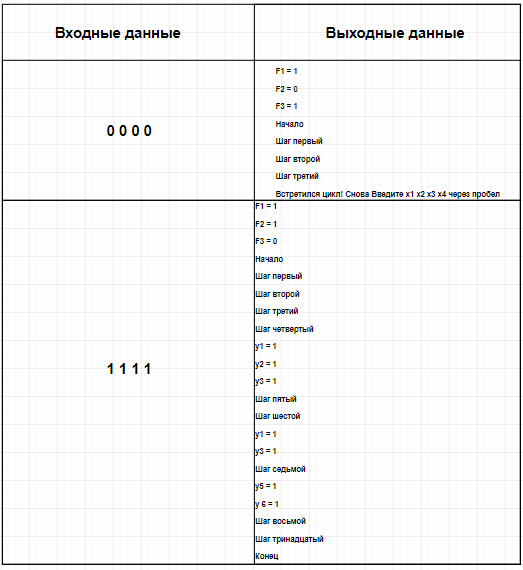
S2 = !α1!α2α3F3

S3 = !α1!α2!α3F1F2 v !α1 α2!α3 v α1!α2!α3

**5.** Разработаем аппаратную реализацию управляющего автомата УЛУ (используя канонический метод структурного синтеза).



**6.** Разработал программную модель УЛУ на языке с++



Исходный код

#include <iostream>

#include <algorithm>

#include <vector>

#include <cstdio>

using namespace std;

vector<bool> F(3);

vector<bool> state(4);

int step = 1;

void get\_input() {

int x1, x2, x3, x4;

cout << "Введите x1 x2 x3 x4 через пробел" << endl;

cin >> x1 >> x2 >> x3 >> x4;

F[0] = x2 | !x3 | x1;

F[1] = ((x1 & !x3 & !x4) | (x2 & !x3) | (x2 & x4) | (x1 & x3) | (!x2 & x3));

F[2] = x2 | !x1 | !x3;

}

int main() {

setlocale(LC\_ALL, "Russian");

get\_input();

cout << "F1 = " << F[0] << endl << "F2 = " << F[1] << endl << "F3 = " << F[2] << endl;

int cycle = 0;

while (1) {

switch (step)

{

case 1:

cout << "Начало" << endl;

cout << "Шаг первый" << endl;

step = 2;

break;

case 2:

cout << "Шаг второй" << endl;

if (F[0] == 0) {

step = 2;

cout << "Снова ";

get\_input();

} else {

step = 3;

}

break;

case 3:

cout << "Шаг третий" << endl;

if (F[1] == 0) {

step = 2;

cout << "Встретился цикл! Снова ";

get\_input();

} else {

step = 4;

}

break;

case 4:

cout << "Шаг четвертый" << endl;

state[0] = 0;

state[1] = 0;

state[2] = 1;

cout << "y1 = " << ((!state[0] & !state[1] & state[2]) | (state[0] & !state[1] & !state[2])) << endl;

cout << "y2 = " << ((!state[0] & state[2]) | (!state[0] & state[1] & !state[2])) << endl;

cout << "y3 = " << ((!state[0] & !state[1] & state[2]) | (state[0]) & !state[1] & !state[2]) << endl;

step = 5;

break;

case 5:

cout << "Шаг пятый" << endl;

if (F[2] == 1) {

step = 9;

}

else {

step = 6;

}

break;

case 6:

cout << "Шаг шестой" << endl;

state[0] = 1;

state[1] = 0;

state[2] = 0;

cout << "y1 = " << ((!state[0] & !state[1] & state[2]) | (state[0] & !state[1] & !state[2])) << endl;

cout << "y3 = " << ((!state[0] & !state[1] & state[2]) | (state[0]) & !state[1] & !state[2]) << endl;

step = 7;

break;

case 7:

cout << "Шаг седьмой" << endl;

state[0] = 1;

state[1] = 0;

state[2] = 1;

cout << "y5 = " << ((!state[0] & state[1] & state[2]) | (state[0] & !state[1] & state[2])) << endl;

cout << "y6 = " << ((state[0] & !state[1] & state[2])) << endl;

step = 8;

break;

case 8:

cout << "Шаг восьмой" << endl;

step = 13;

break;

case 9:

cout << "Шаг девятый" << endl;

state[0] = 0;

state[1] = 1;

state[2] = 0;

cout << "y2 = " << ((!state[0] & state[2]) | (!state[0] & state[1] & !state[2])) << endl;

cout << "y4 = " << ((!state[0] & state[1] & !state[2])) << endl;

step = 10;

break;

case 10:

cout << "Шаг десятый" << endl;

if (F[1] == 0) {

step = 7;

}

else {

step = 11;

cycle++;

}

break;

case 11:

cout << "Шаг одинадцатый" << endl;

state[0] = 0;

state[1] = 1;

state[2] = 1;

cout << "y2 = " << ((!state[0] & state[2]) | (!state[0] & state[1] & !state[2])) << endl;

cout << "y5 = " << ((!state[0] & state[1] & state[2]) | (state[0] & !state[1] & state[2])) << endl;

step = 12;

break;

case 12:

cout << "Шаг двенадцатый" << endl;

step = 9;

break;

case 13:

cout << "Шаг тринадцатый" << endl;

cout << "Конец" << endl;

step = 14;

break;

default:

break;

}

if (cycle > 1) {

cout << "!!!!Бесконечный цикл!!!!" << endl;

break;

}

}

return 0;

}