МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РФ
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Вычислительная техника»

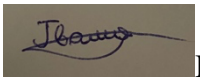
Курсовая работа
по дисциплине «Метод моделирования цифровых устройств»
на тему «Моделирование микропроцессорного управляющего устройства»

Специальность подготовки 09.05.01 «Применение и эксплуатация автоматизированных систем специального назначения»

Специализация №1: «Эксплуатация вычислительных машин, комплексов, систем и сетей специального назначения»

ПГУ 1.090501.03

Выполнил студент группы 18 ВО2

Гвалия Д.М.

Руководитель: д.т.н, профессор каф. ВТ

Зинкин С.А.

Работа защищена с оценкой_____

Преподаватели_____

_____

_____

Дата защиты_____

2020

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники

Кафедра “Вычислительная техника”

“УТВЕРЖДАЮ”

Зав. кафедрой ВТ
М.А. Митрохин

«14» сентября 2020 г.

ЗАДАНИЕ

на курсовую работу по дисциплине

Методы моделирования цифровых устройств

Студенту Гвалия Д.М. Группа 18ВО2

Тема_проекта: Моделирование микропроцессорного управляющего устройства. Вариант №3

Исходные данные (технические требования) на проектирование:

Промоделировать работу микропроцессорного управляющего устройства, реализующего логические функции и управляющий микропрограммный автомат. Ввод и вывод управляющих сигналов задавать при помощи регистров. Языки моделирования и программирования: C, C++, ассемблер MCS51.

Объем работы по курсу:

1. Расчетная часть

   1) Введение и постановка задачи.

   2) Описание моделирующей среды.

   3) Разработка алгоритма.

   4) Разработка программы.

   5) Руководство оператора.

   6) Контрольный просчет.

   Приложение А – алгоритмы.

Приложение Б – (рекомендуется листинги программ на языках С++ и на ассемблере микроконтроллера MCS51 разместить прямо в основном тексте).

Пример содержания пояснительной записки

Введение (описание микроконтроллера, схема, команды).

1.Моделирование комбинационной схемы на языке С (реализация функции алгебры логики на С).

2.Моделирование сложной комбинационной схемы на языке С
(реализация функции алгебры логики, заданной СДНФ).

3.Моделирование минимизированной сложной комбинационной схемы на языке С (реализация функции алгебры логики, заданной минимальной ДНФ).

4.Моделирование управляющего алгоритма на языке С.

5.Моделирование комбинационной схемы на языке Ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики на языке Ассемблера для микроконтроллера MCS51).

6.Моделирование сложной комбинационной схемы на на языке Ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики, заданной СДНФ на языке Ассемблера для микроконтроллера MCS51).

7.Моделирование минимизированной сложной комбинационной схемы на языке Ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики, заданной минимальной ДНФ на языке Ассемблера для микроконтроллера MCS51).

8.Моделирование управляющего алгоритма на языке Ассемблера для микроконтроллера MCS51.

Заключение.

Графическая часть:

Алгоритмы моделирования

Экспериментальная часть

Результаты отладки алгоритма программы с использованием электронного симулятора EdSim51.

Срок завершения выполнения работы по разделам

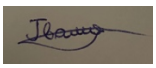
В соответствии с графиком защит работ на неделе, предшествующей зачетной сессии.

Дата выдачи задания “14” сентября 2020 г.

Дата защиты проекта: по графику

Руководитель_____/Зинкин С.А./

Задание получил “14” сентября 2020 г.

Студент_____  _____/ Гвалия Д.М./

Реферат

Пояснительная записка состоит из 42 листов, 24 рисунков, 9 таблиц, и 2 приложений.

Цель работы - моделирование работы микропроцессорного управляющего устройства, реализующего логические функции и управляющий микропрограммный автомат. Ввод и вывод управляющих сигналов задавать при помощи регистров.

В результате проделанной работы была выполнена модель микропроцессорного управляющего устройства, реализующего логические функции и управляющий алгоритм с использованием языка C++ и ассемблера MCS51.

Содержание

Введение..... 5
1 Постановка и реализация задачи..... 6-27
1.1 Моделирование комбинационных схем на языке С (реализация функций алгебры логики на С)..... 6-8
1.2 Моделирование сложной комбинационной схемы на языке С (реализация функции алгебры логики, заданной СДНФ)..... 9-13
1.3 Моделирование управляющего алгоритма на языке С..... 14-15
1.4 Моделирования комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики на языке ассемблера для микроконтроллера MCS51)..... 16-18
1.5 Моделирование сложной комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики, заданной СДНФ на языке ассемблера для микроконтроллера MCS51)..... 19-23
1.6 Моделирования управляющего алгоритма на языке ассемблера для микроконтроллера MCS51..... 24-27
2 Описание моделирующей среды..... 28-30
3 Руководство оператора..... 31-33
Заключение..... 34
Список использованных источников..... 35
Приложение А. Алгоритмы..... 36-38
Приложение Б. Система команд микроконтроллера MCS51..... 39-42

Введение

Диапазон применения микропроцессорной техники в настоящее время очень широк, к микропроцессорным системам предъявляются разнообразные требования. Ниже представлены основные типы микропроцессорных систем (МПС), различающихся мощностью, универсальностью, быстродействием и структурой:

- микроконтроллеры (МК) - наиболее простой тип микропроцессорных систем, в которых все или большинство узлов системы выполнены в виде одной микросхемы;

-контроллеры - управляющие микропроцессорные системы, выполненные в виде отдельных модулей;

- микрокомпьютеры - более мощные микропроцессорные системы с развитыми средствами сопряжения с внешними устройствами;

- компьютеры (в том числе персональные) - самые мощные и наиболее универсальные микропроцессорные системы.

В настоящее время при разработке новых МПС чаще выбирают путь использования микроконтроллеров (примерно в 80% случаев). При этом МК применяются самостоятельно (с дополнительной аппаратурой) или в составе сложных контроллеров с развитыми средствами ввода/вывода.

Классические МПС на базе микросхем процессоров и микропроцессорных комплектов выпускаются сейчас редко из-за сложности процесса разработки и отладки этих систем. Данный тип МПС выбирают тогда, когда микроконтроллеры не могут обеспечить требуемых характеристик.

1 Постановка задачи

1.1 Моделирование комбинационных схем на языке С (реализация функции алгебры логики на С).

Формулировка задания:

Построить таблицы истинности для данных формул. Смоделировать комбинационные схемы на ПЭВМ.

Ход работы:

Дана функция F1=¬(X1&(Y1v¬X1))→(X1&Y1). Построена ее таблица истинности (Таблица 1).

Таблица 1 - Таблица истинности функции F1

x	y	⌐x	yv(⌐x)	x&(yv(⌐x))	⌐(x&(yv(⌐x)))	x&y	(⌐(x&(yv(⌐x)))→(x&y)
0	0	1	1	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	0
1	1	0	1	1	0	1	1

Дана функция F2= X&Yv¬Z. Построена ее таблица истинности (Таблица2)

Таблица 2 - Таблица истинности функции F2

x	y	z	x&y	⌐z	(x&y)v(⌐z)
0	0	0	0	1	1
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	0	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	1	0	1

Программа F1 разработана в среде Microsoft Visual Studio 2010 и предназначена для вывода на экран таблиц истинности сложных комбинированных функций F1 и F2.

Код программы:

```
#include <Windows.h>
#include <conio.h>
#include <iostream>
using namespace std;

int main(){
   bool x1=0, y1=0, f;
   printf("Source: !(x1&(y1|!x1))<=(x1&y1)\n");
   printf("x,y,f\n");
   for (int i=0;i<2;i++){
      for (int j=0;j<2;j++){
         f=!(x1&&(y1||!x1))<=(x1&&y1);
         printf("%i %i %i \n",x1,y1,f);
         x1=!x1;
```

```
    }
    y1=!y1;
  }
  printf("\n\n\n");
   bool x=0, y=0,z=0, f1;
  printf("Source: (x&y)|(!z)\n");
  printf("x,y,z,f\n");
  for (int i=0;i<2;i++){
    for (int j=0;j<2;j++){
      for (int k = 0; k < 2; k++){
        f1=(x&&y)||(!z);
        printf("%i %i %i %i \n",x,y,z,f1);
        x=!x;}
      y=!y;}
    z=!z;}}
```

После запуска программы на экраны появляется консольное окно (Рисунок 1), а на ней результат работы программы.

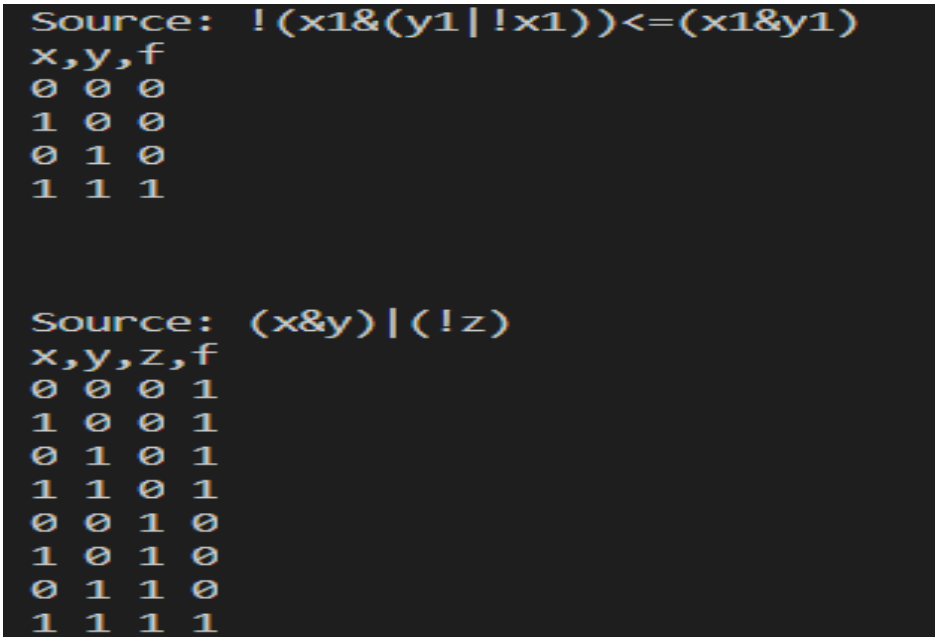


Рисунок 1 - Результат работы программы F1

1.2 Моделирование сложной комбинационной схемы на языке С (реализация функции алгебры логки, заданной СДНФ).

Формулировка Задания:

Составить СДНФ функции по заданным значениям аргументов, на которых логическая функция принимает значение истинна (Таблица 3). Построить таблицы истинности для всех функций. Смоделировать полученную схему на ПЭВМ, минимизировать ее.

Таблица 3 - исходные данные

Номер варианта	Номер конституента
3	0, 2, 4, 5, 6, 7 ,9, 11

Ход работы:

Для логической функции, заданной списком единичных наборов была построена таблица истинности (Таблица 4)

Таблица 4 - таблица истинности функции F(x1,x2,x3,x4)

№	x1	x2	x3	x4	F
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1

Продолжение таблицы 4

10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

Составляем СДНФ:

F=(¬x1&¬x2&¬x3&¬x4)v(¬x1&¬x2&x3&¬x4)v(¬x1&x2&¬x3&¬x4)v(¬x1&x2&¬x3&x4)v(¬x1&x2&x3&¬x4)v(¬x1&x2&x3&x4)v(x1&¬x2&¬x3&x4)v(x1&¬x2&x3&x4)

С помощью диаграммы Вейча была минимизирована СДНФ (F1(x1,x2,x3,x4)).

Получена минимизированная функция

F2(x1,x2,x3,x4)=(¬x1&¬x4)v(¬x1&x2)v(x1&¬x2&x4).

Для проверка правильности полученной минимизированной функции была построена ее таблица истинности.

Таблица 5 - таблица истинности функции F2

№	X1	X2	X3	X4	F1	F2
0	0	0	0	0	1	1
1	0	0	0	1	0	0
2	0	0	1	0	1	1
3	0	0	1	1	0	0
4	0	1	0	0	1	1
5	0	1	0	1	1	1
6	0	1	1	0	1	1
7	0	1	1	1	1	1
8	1	0	0	0	0	0

Продолжение таблицы 5

9	1	0	0	1	1	1
10	1	0	1	0	0	0
11	1	0	1	1	1	1
12	1	1	0	0	0	0
13	1	1	0	1	0	0
14	1	1	1	0	0	0
15	1	1	1	1	0	0

Из анализа таблицы 5 видно, что значения функции F1 и F2 совпадают на всех наборах данных. Следовательно минимизация была выполнена успешно.

Программа lab_2 разработана в среде Microsoft Visual Studio 2010 и предназначена для вывода таблиц истинности СДНФ и минимизированной функции.

Код программы:

```
#include "stdafx.h"
#include <Windows.h>
#include <conio.h>
#include <stdio.h>
#include <iostream>
using namespace std;
int _tmain(int argc, _TCHAR* argv[])
{  setlocale(LC_ALL, "Russian");
      bool x1=0, x2=0,x3=0, x4=0, f;
      printf("СДНФ:
(!x1&&!x2&&!x3&&!x4)||(!x1&&!x2&&x3&&!x4)||(!x1&&x2&&!x3&&!x4)||(!x1&&x2&&!x3&&x4
)||(!x1&&x2&&x3&&!x4)||(!x1&&x2&&x3&&x4)||(x1&&!x2&&!x3&&x4)||(x1&&!x2&&x3&&x4)\n"
);
      printf("\n");
      printf("x1,x2,x3,x4,f1\n");
      for (int i=0;i<2;i++){
            for (int j=0;j<2;j++){
                  for (int k=0;k<2;k++){
                        for (int z=0;z<2;z++){
    f=(!x1&&!x2&&!x3&&!x4)||(!x1&&!x2&&x3&&!x4)||(!x1&&x2&&!x3&&!x4)||(!x1&&x2&&!x3
&&x4)||(!x1&&x2&&x3&&!x4)||(!x1&&x2&&x3&&x4)||(x1&&!x2&&!x3&&x4)||(x1&&!x2&&x3&&
x4);
printf("%i " " %i " " %i " " %i " " %i " " \n",x1,x2,x3,x4,f);
```

```
                x4=!x4;
                    }
            x3=!x3;
            }

        x2=!x2;
            }
            x1=!x1;

        }
        printf("\n");
        x1=0;
        x2=0;
        x3=0;
        x4=0;
        printf("МДНФ: (!x1&&!x4)||(!x1&&x2)||(x1&&!x2&&x4)\n");
        printf("x1,x2,x3,x4,f2\n");
        for (int i=0;i<2;i++){
            for (int j=0;j<2;j++){
                for (int k=0;k<2;k++){
                    for (int z=0;z<2;z++){
                f=(!x1&&!x4)||(!x1&&x2)||(x1&&!x2&&x4);
                printf("%i " " %i " " %i " " %i " " %i " " \n",x1,x2,x3,x4,f);
                x4=!x4;
                    }
            x3=!x3;
            }

        x2=!x2;
            }
            x1=!x1;

        }


        getch();
        return 0;
    }
```

После запуска программы, на экране появляется консольное окно, в котором выводится результат работы программы.

Результат:

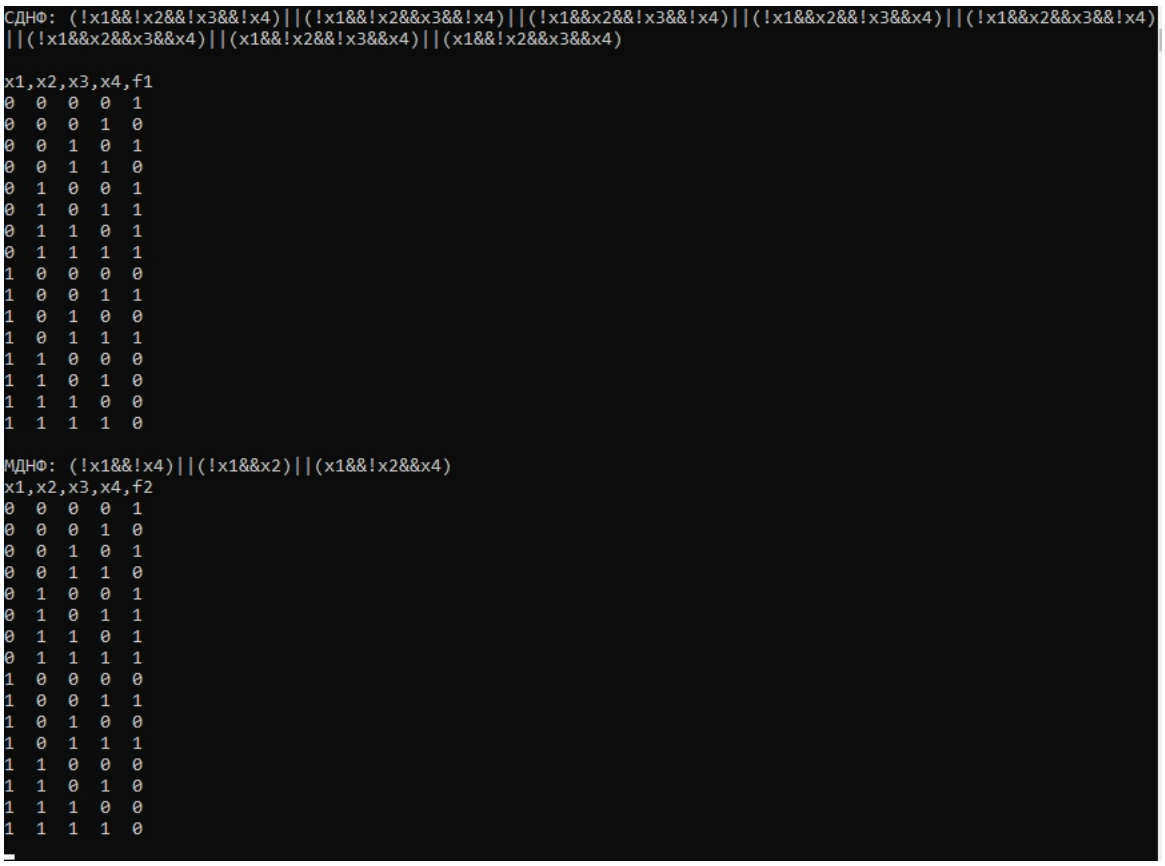


Рисунок 2 - результат работы программы lab_2

1.3 Моделирование управляющего алгоритма на языке С

Формулировка задания:

Реализовать сложную неструктурированную ГСА(Приложение А рисунок А.1) на языке С++ с использованием оператора goto.

Ход работы:

Алоритм решения задачи в виде схемы программы представлен в приложении А на рисунке А.2».

Программа lab3 разработана в среде Microsoft Visual Studio 2010 и предназначена для вывода на экран результат реализации сложной ГСА.

Код программы:

```
#include "stdafx.h"
#include "iomanip"
#include "windows.h"
#include "iostream"
#include "conio.h"

using namespace std;

int _tmain(int argc, _TCHAR* argv[]){
  bool x1, x2, x3, x4;

  cout«"vvod x1"«endl;
  cin»x1;

  if(!x1){
    cout«"a1"«endl;
    cout«"a2"«endl;

  }else {
    cout«"a2"«endl;
  }
  cout«"vvod x2"«endl;
  cin»x2;
  if(!x2){
    cout«"a3"«endl;
    l1:cout«"a4"«endl;
    cout«"a5"«endl;
    cout«"a6"«endl;
```

```
    goto le;

  }else{
    l2: cout«"vvod x3"«endl;
    cin»x3;
    if(x3){
      cout«"a1"«endl;
      cout«"a4"«endl;
      cout«"vvod x4"«endl;
      cin»x4;
      if(x4){
        goto l2;
      }else{
        goto le;
      }

    }else{
        goto l1;
      }


    }

  le: cout«"end"«endl;

  getch();

  return 0;
}
```

После запуска программы на выполнение появляется консольное окно, а в нем результат работы программы (Рисунок 3).

Рисунок 3 - результат работы программы lab3

1.4 Моделирование комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики на языке ассемблера для микроконтроллера MCS51)

Формулировка задания:

Реализовать логическую функцию (x&y)v¬z на микроконтроллере MCS51.

Ход работы:

Для логической функции была построена таблица истинности

Таблица 6 - таблица истинности функции (x&y)v¬z

x	y	z	x&y	⌐z	(x&y)v(⌐z)
0	0	0	0	1	1
0	0	1	0	0	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	0	1	1
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	1	0	1

Текст программы:

```
JMP BEGIN;
BEGIN:
CLR C;                  C=0
MOV P1, #00001001B;     P1=00001001
MOV C, P1.0;            C=X
ANL C, P1.1;            C=X&Y
MOV 00, C;              Запись промежуточного результата
MOV C, P1.2;            C=Z
CPL C;                  C=¬Z
ORL C, 00;              C=x&y+¬Z
```

```
MOV P1.7, C;
SETB P1.4;                Конец вывода
MOV P1.5, C;              Выдача результата
JMP BEGIN;
```

Результат работы программы приведен на рисунках 4-6.

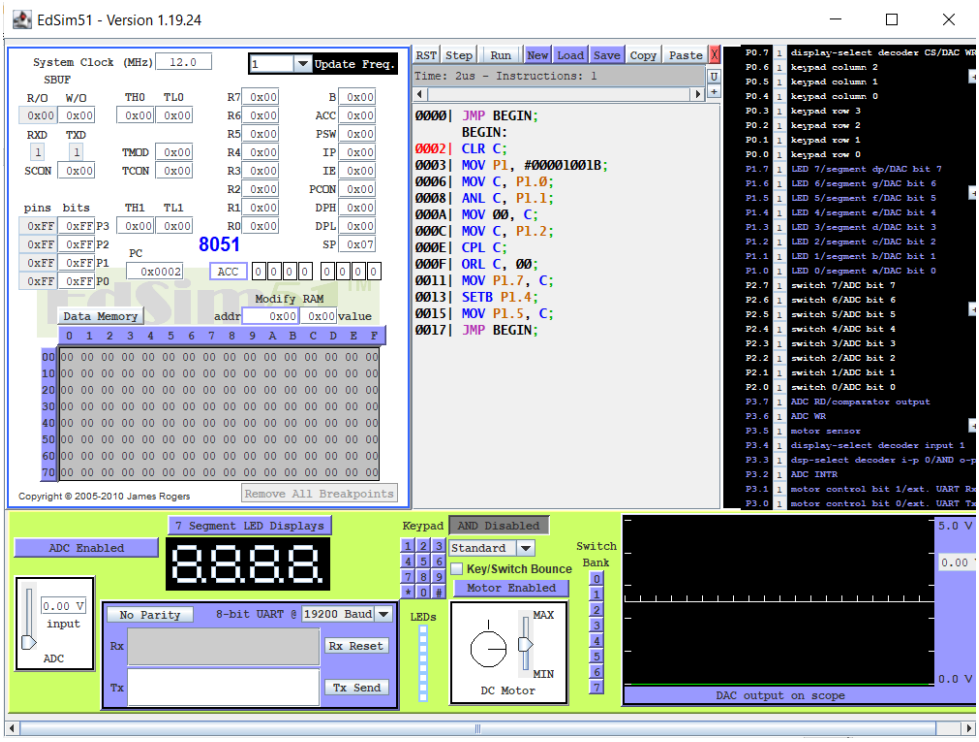


Рисунок 4 - Результат работы программы

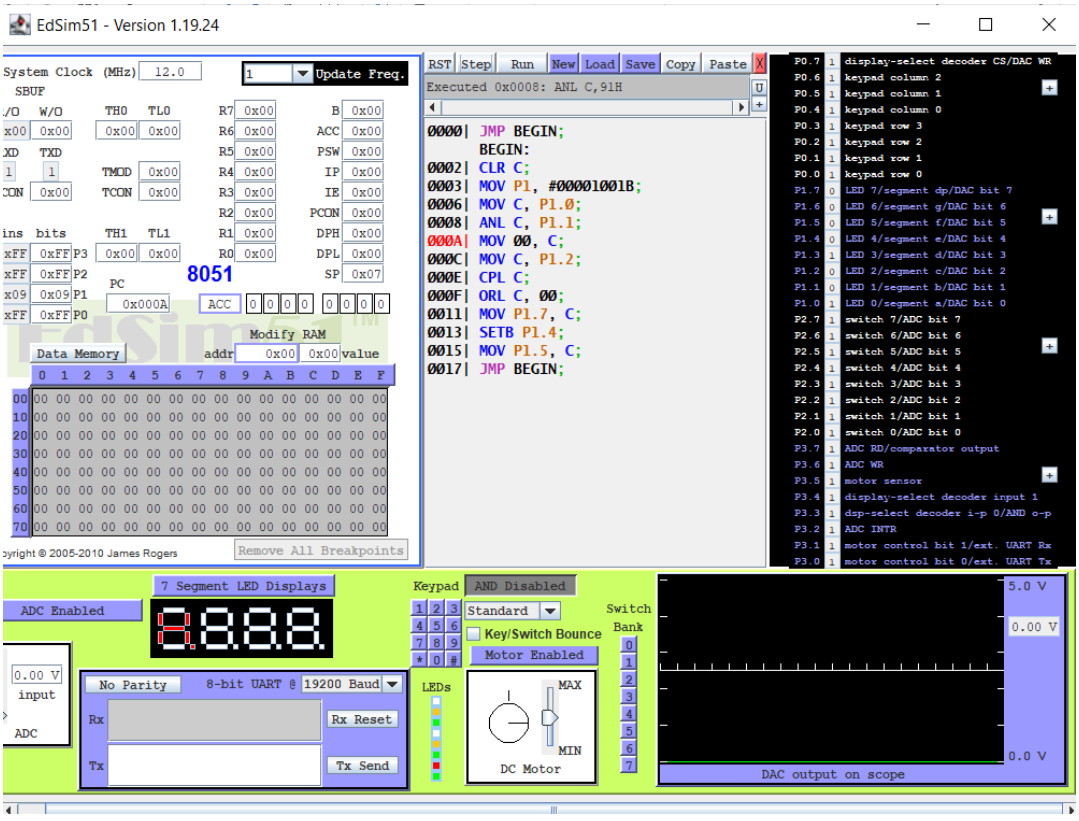


Рисунок 5 - Результат работы программы

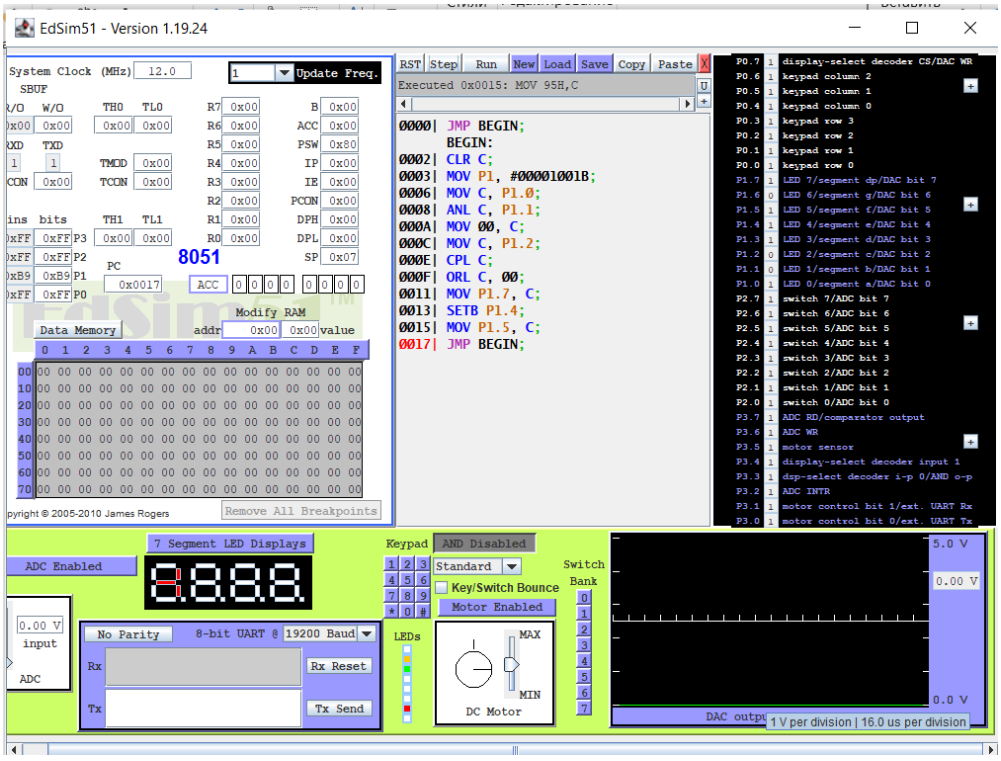


Рисунок 6 - результат работы программы

1.5 Моделирование сложной комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики, заданной СДНФ на языке ассемблера для микроконтроллера MCS51).

Формулировка задачи:

Составить СДНФ функции по заданным значениям аргументов, на которых логическая функция принимает значение, равное единице. Построить таблицу истинности функции. Реализовать логическую функцию на микроконтроллере MCS-51.

Ход работы:

Таблица 7 - номера набора:

0	2	4	5	6	7	9	11

Построим таблицу истинности по заданным наборам:

Таблица 8 - таблица истинности

№	x1	x2	x3	x4	F
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

Составляем СДНФ:

F=(¬x1&¬x2&¬x3&¬x4)v(¬x1&¬x2&x3&¬x4)v(¬x1&x2&¬x3&¬x4)v(¬x1&x2&¬x3&x4)v(¬x1&x2&x3&¬x4)v(¬x1&x2&x3&x4)v(x1&¬x2&¬x3&x4)v(x1&¬x2&x3&x4)

Программа lab5 разработаны на микроконтроллере MCS-51 ипредназначена для реализации сложной логической функции, заданной в формеСДНФ.

Система команд микроконтроллера MCS-51 представлена в ПриложенииБ рисунки Б.1-Б.5./2/

Текст программы:

```
JMP BEGIN;
BEGIN:
CLR C; C=0
MOV P1,#00000100B; P1=00000100
MOV C,P1.1; C=X1
CPL C; C=/X1
ANL C,/P1.2; C=/X1&/X2
ANL C,/P1.3; C=/X1&/X2&/X3
ANL C,/P1.4; C=/X1&/X2&/X3&/X4
JC KONEC;
MOV C,P1.1; C=X1
CPL C; C=/X1
ANL C,/P1.2; C=/X1&/X2
ANL C,P1.3; C=/X1&/X2&X3
ANL C,/P1.4; C=/X1&/X2&X3&/X4
JC KONEC;
MOV C,P1.1; C=X1
CPL C; C=/X1
ANL C,P1.2; C=/X1&X2
ANL C,/P1.3; C=/X1&X2&/X3
ANL C,/P1.4; C=/X1&X2&/X3&/X4
JC KONEC;
MOV C,P1.1; C=X1
CPL C; C=/X1
ANL C,P1.2; C=/X1&X2
ANL C,/P1.3; C=/X1&X2&/X3
ANL C,P1.4; C=/X1&X2&/X3&X4
JC KONEC;
MOV C,P1.1; C=X1
CPL C; C=/X1
ANL C,P1.2; C=/X1&X2
ANL C,P1.3; C=/X1&X2&X3
ANL C,/P1.4; C=/X1&X2&X3&/X4
JC KONEC;
```

```
MOV C,P1.1; C=X1
CPL C; C=/X1
ANL C,/P1.2; C=/X1&X2
ANL C,P1.3; C=/X1&X2&X3
ANL C,P1.4; C=/X1&X2&X3&X4
JC KONEC; 7777
MOV C,P1.1; C=X1
ANL C,/P1.2; C=X1&/X2
ANL C,/P1.3; C=X1&/X2&/X3
ANL C,P1.4; C=X1&/X2&/X3&X4
JC KONEC; 8888
MOV C,P1.1; C=X1
ANL C,/P1.2; C=X1&/X2
ANL C,P1.3; C=X1&/X2&X3
ANL C,P1.4; C=X1&/X2&X3&X4
KONEC: MOV P1.0,C;
JMP BEGIN;
```

Результат работы программы можно увидеть на рисунках 7-8

Результаты работы программы:

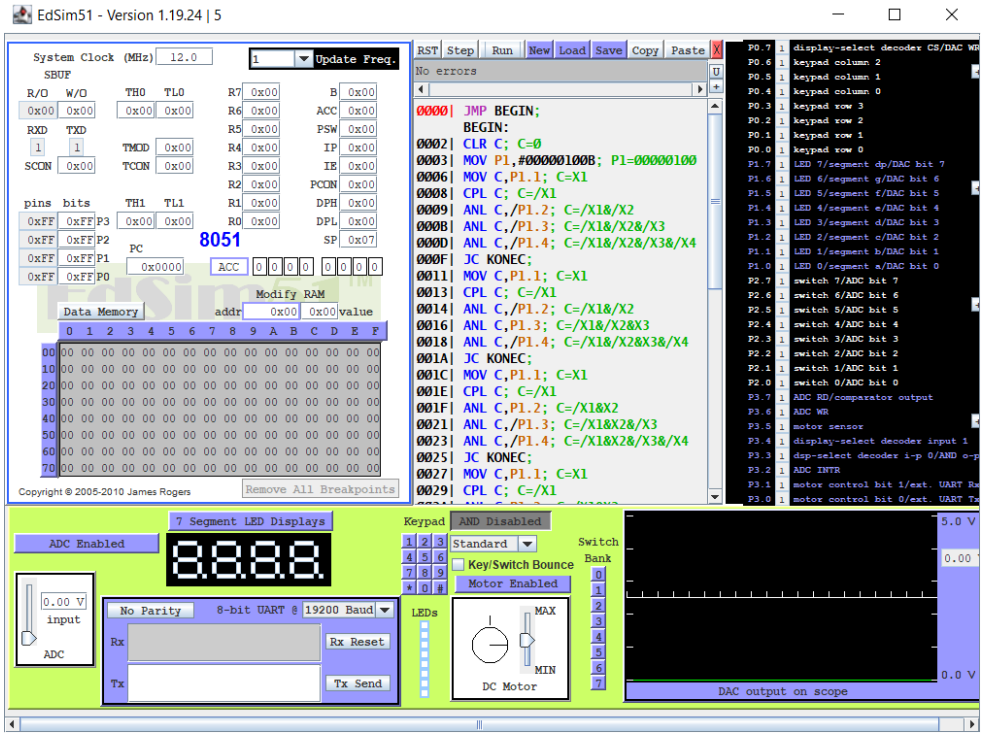


Рисунок 7 — Результаты работы программы

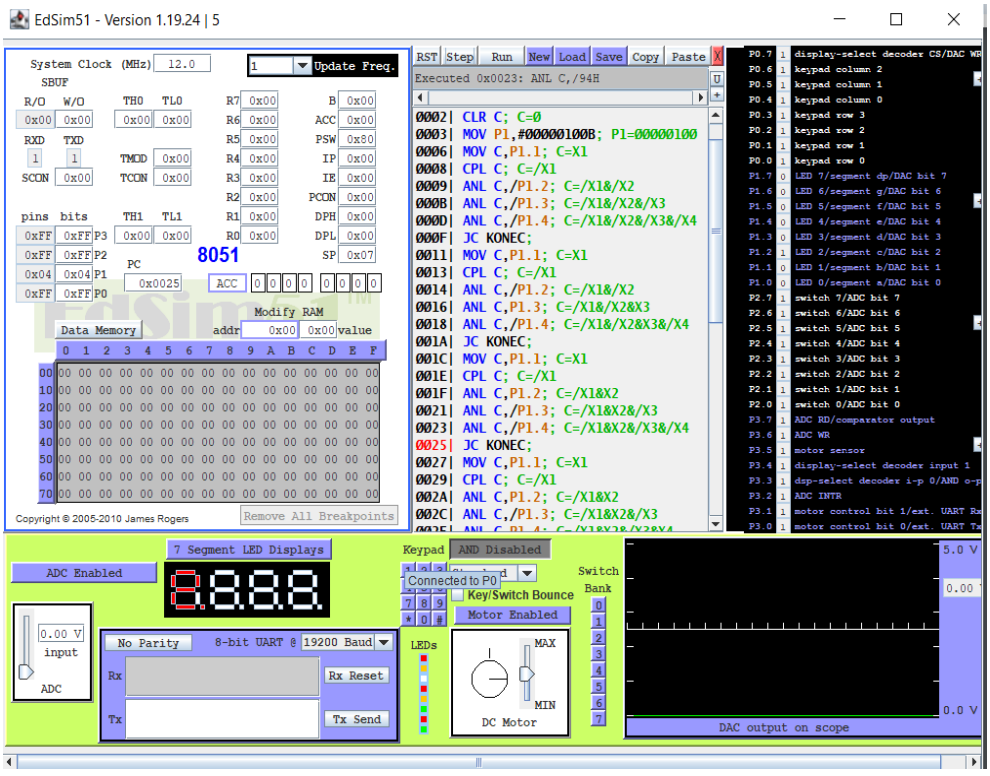


Рисунок 8 — Результаты работы программы

1.6 Моделирование управляющего алгоритма на языке ассемблера MCS51

Формулировка задачи

Реализовать сложную неструктурированную граф-схему алгоритма (ГСА) на микроконтроллере MCS-51.

Ход работы:

Алгоритм решения задачи в виде схемы программы представлен в приложении А на рисунке А.2

Программа lab6 разработана на микроконтроллере MCS51 и предназначена для реализации сложной неструктурированной ГСА.

Текст программы

Таблица 9 - текст программы lab6

	Псевдокод	**Ассемблер MCS-51**	**Комментарий**

BEGIN	AN	JMP BEGIN;
		BEGIN: CLR C;
		MOV P0, #10110B;
		SETB P1.0;
		CLR P1.0;
		MOV P2, #00000000B;
			Ввод значений
			
			Ввод x1 - 3
			Ввод x4=1
			Ввод x4=0
			Начало AN

AX1:	AX1	MOV C, P0.1;	Ввод X1
	IF X1=1 GOTO A2	JC A2;	
	A1	MOV P2, #00000001B;	Исп. X1
	A2	MOV P2, #00000010B;	Исп. X2
	GOTO AX2	JMP AX2;	
A2:	A2	A2:MOV P2, #00000010B;	Исп. X2
	GOTO AX2	JMP AX2;	
AX2:	AX2	AX2:MOV C, P0.2;	Ввод X2
	IF X2=1 GOTO AX3	JC AX3;	
	A3	MOV P2, #00000011B;	Исп. X3
	A4	A4:MOV P2, #00000100B;	Исп. X4
	A5	MOV P2, #00000101B;	Исп. X5
	A6	MOV P2, #00000110B;	Исп. X6
	GOTO AK	JMP AK;	
AX3:	AX3	AX3:MOV C, P0.3;	Ввод X3
	IF X3=0 GOTO A4	JNC A4;	
	A1	MOV P2, #00000001B;	Исп. X1
	A4	MOV P2, #00000100B;	Исп. X4
	AX4	MOV C, P0.4;	Ввод X4
	IF X3=1 GOTO AX3	JC AX3;	
	GOTO AK	JMP AK;	
AK:	AK	AK:MOV P2, #00000110B;	Конец АК
		JMP BEGIN;	

Результат работы программы приведён на рисунках 9 -11.

Результат работы программы

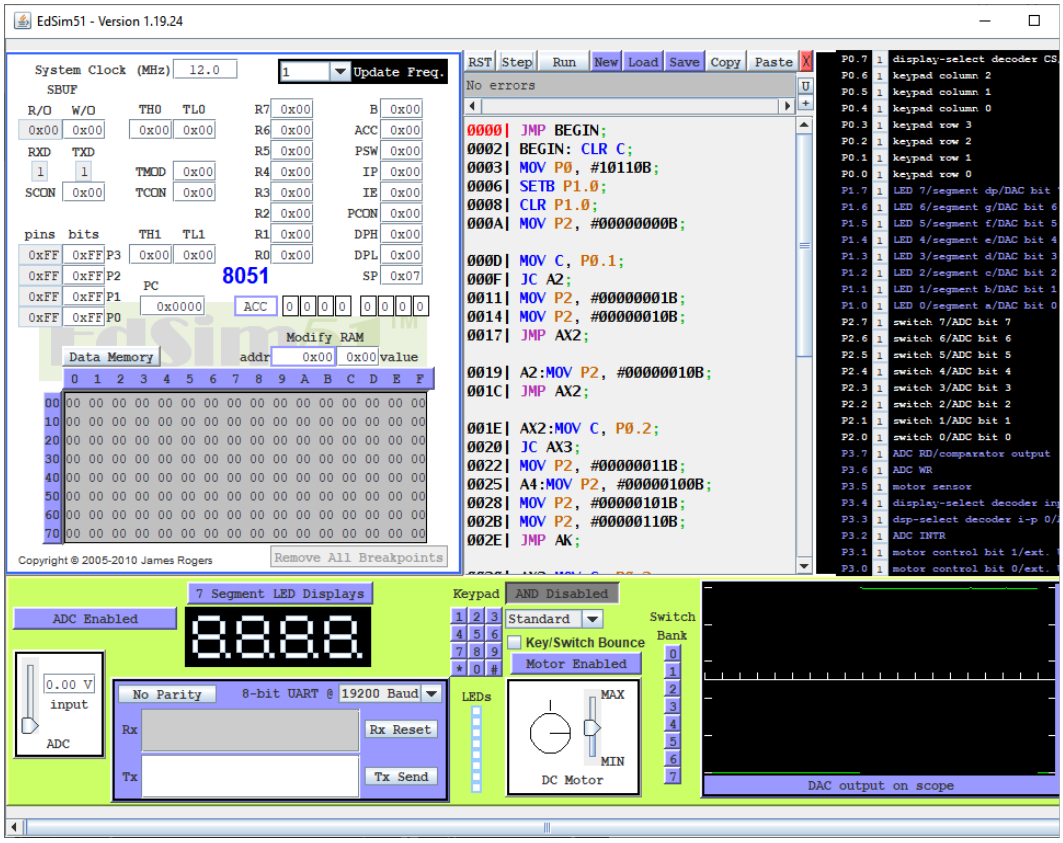


Рисунок 9 - Результат работы программы

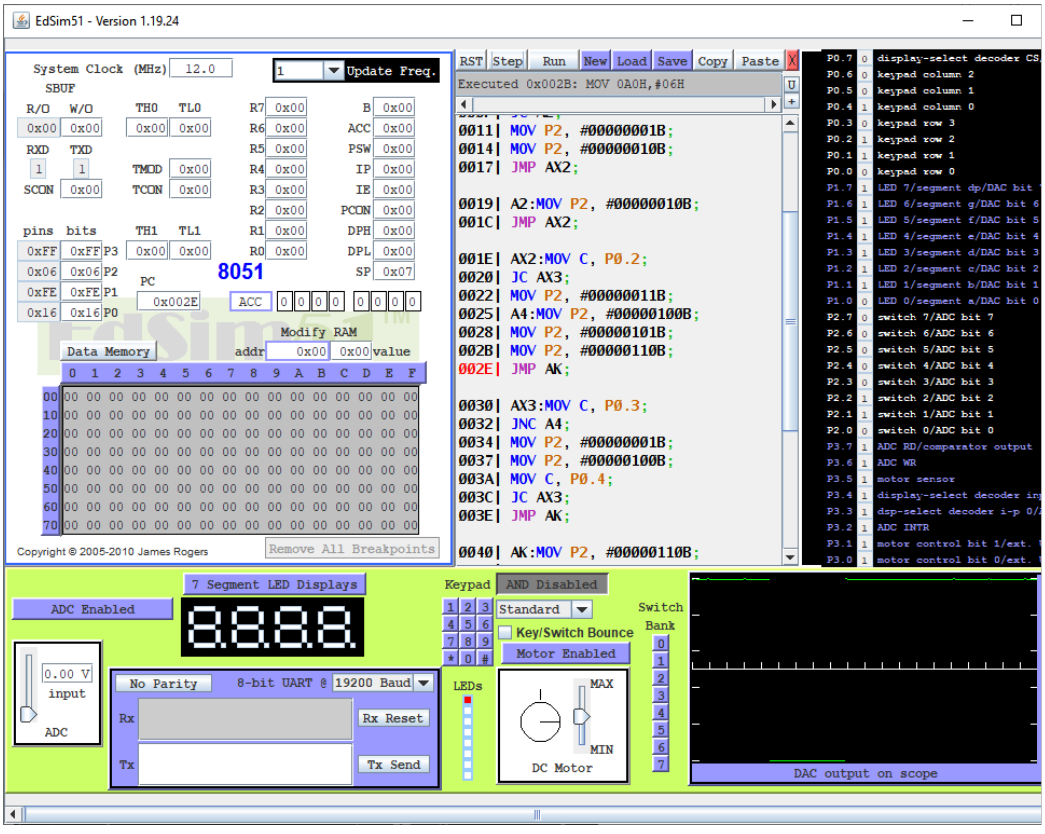


Рисунок 10 - Результат работы программы

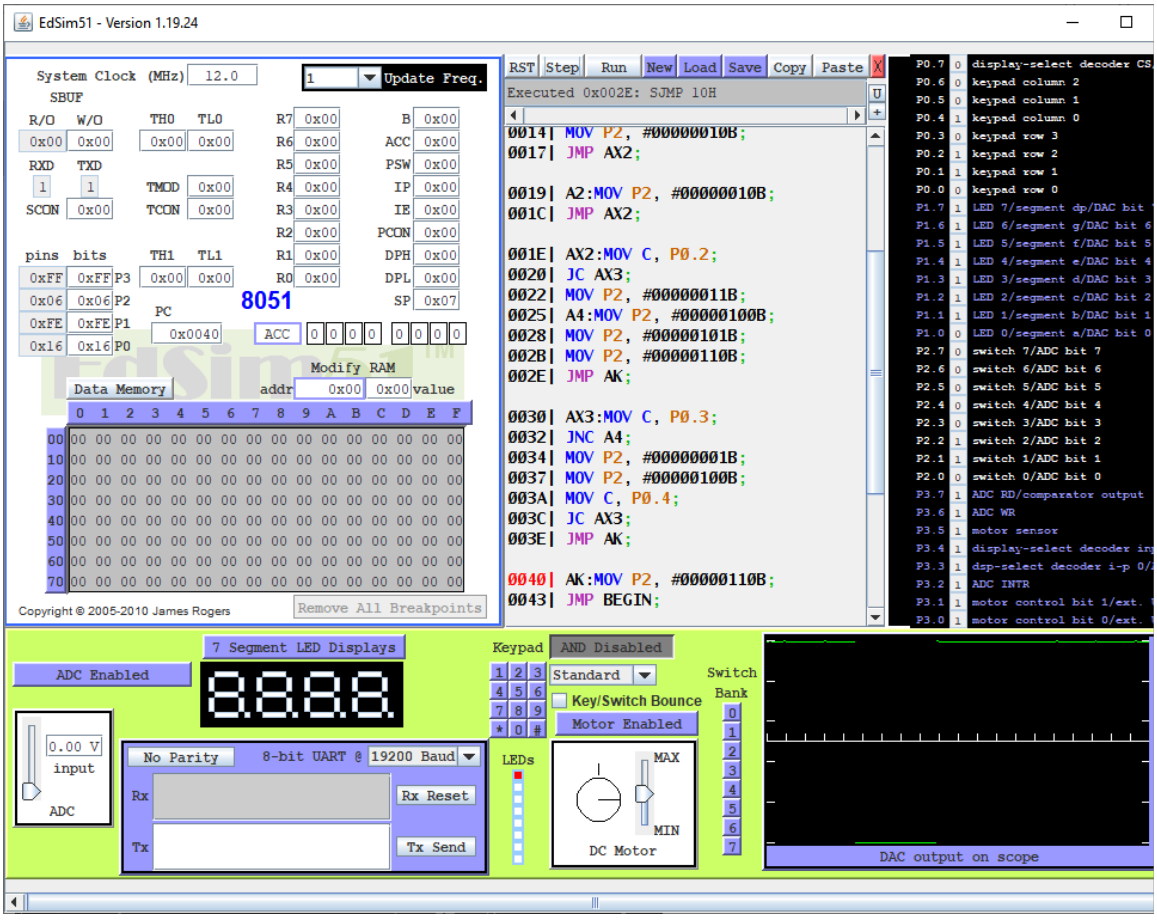


Рисунок 11 - Результат работы программы

2. Описание моделирующей среды

Программные средства изучения и проектирования микроконтроллерных систем на основе МК 8051 предназначены для освоения программирования микроконтроллеров семейства МК 8051 и для программирования и отладки совместной работы микроконтроллера с основными компонентами микропроцессорных систем управления (СУ). EdSim51 - это самый популярный и простой симулятор, доступный для микроконтроллера 8051. EdSim - это фактически виртуальный 8051 с клавиатурой, АЦП, ЦАП, 7-сегментным дисплеем и другими внешними периферийными устройствами. EdSim специально создан для студентов и их образовательных целей. Изюминкой этого симулятора является "простота и удобство использования". Любой новичок может изучить это программное обеспечение довольно быстро. Стенд включает в себя следующие основные компоненты:

- микроконтроллер 8051;
- схему подключения блока светодиодов, цифроаналогового преобразователя и дисплея;
- схему подключения блока светодиодов, ЦАП и жидкокристаллического индикатора;
- схему подключения блока механических переключателей и аналого-цифрового преобразователя;
- схему подключения компаратора и ЦАП;
- схему подключения электродвигателя и универсального асинхронного приёмопередатчика;
- схему подключения клавиатуры.

Кроме того, стенд включает источник постоянного напряжения и осциллограф.

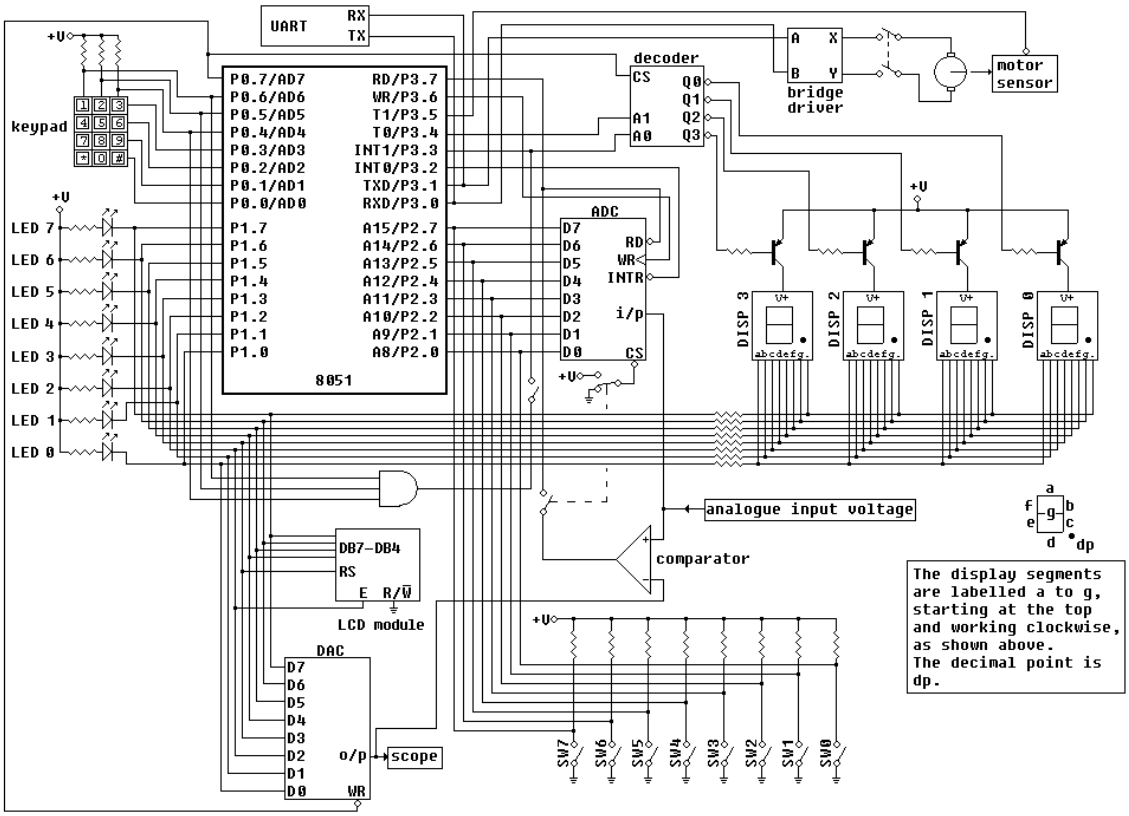


Рисунок 12 - Функциональная схема модели лабораторного стенда EdSim51

На рисунке 13 приведен внешний вид EdSim51.

Рассмотрим основные компоненты пользовательского интерфейса, их функции и связь с элементами функциональной схемы EdSim51.

На рисунке 8 можно выделить четыре основных панели:

- панель микроконтроллера;
- панель ассемблерного кода;
- панель периферийного оборудования;
- Панель отображения состояния линий портов ввода/вывода.

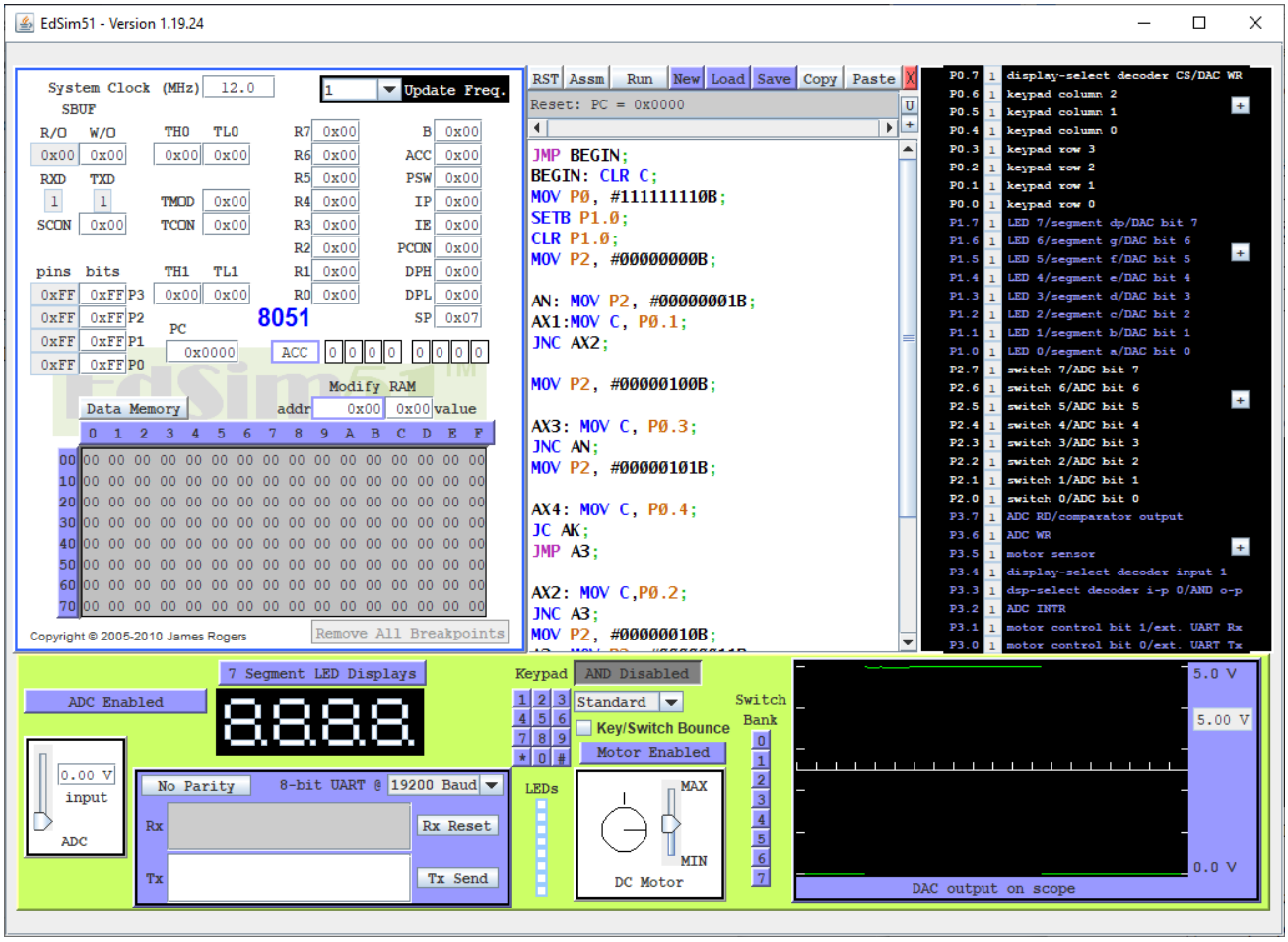


Рисунок 13 - Внешний вид модели лабораторного стенда EdSim51

3. Руководство оператора

Ввод текста программы происходит в панель ассемблерного кода (рисунок 14).

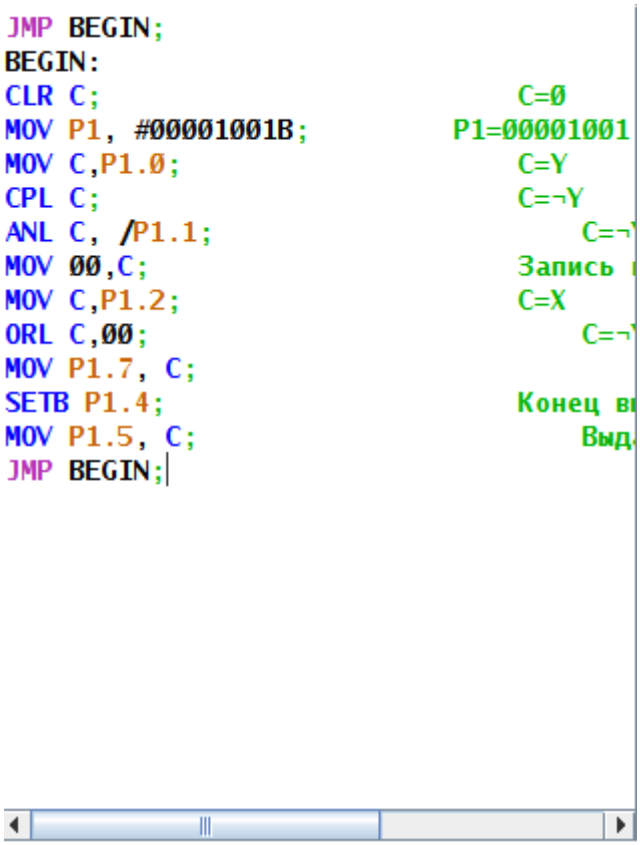


Рисунок 14 - Панель ассемблерного кода

Для запуска программы необходимо нажать на кнопку Run, для ручной отладки необходимо нажать на кнопку Step. Для редактирования кода после запуска программы необходимо нажать на кнопку Reset (Рисунок 15).

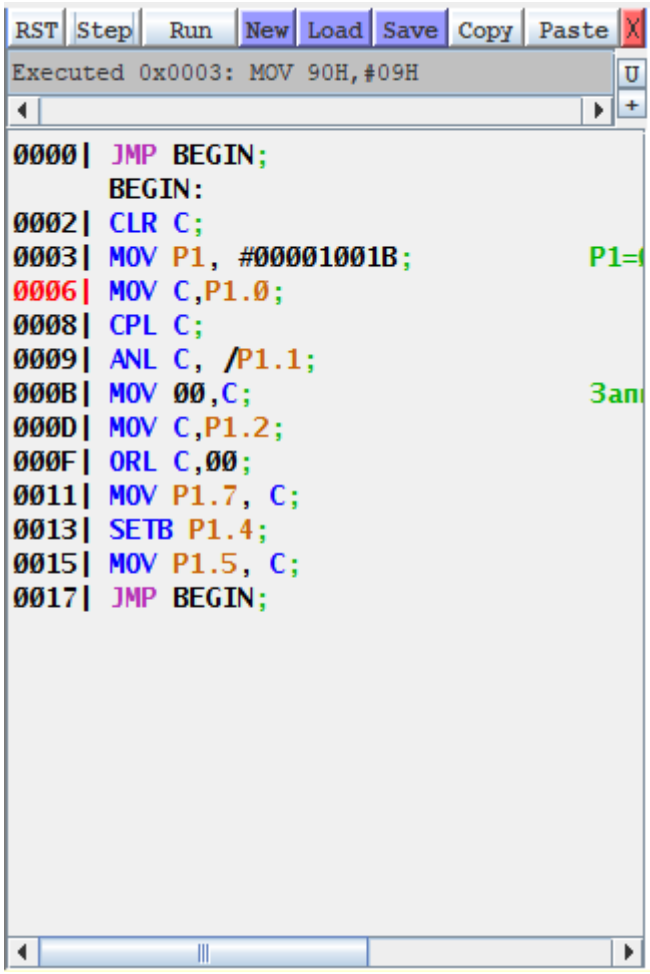


Рисунок 15 - Панель ассемблерного кода

При нажатии на одну из кнопок, отмеченных знаком “+” (рисунок 16), можно увидеть соединения портов в отдельном окне (рисунок 17).

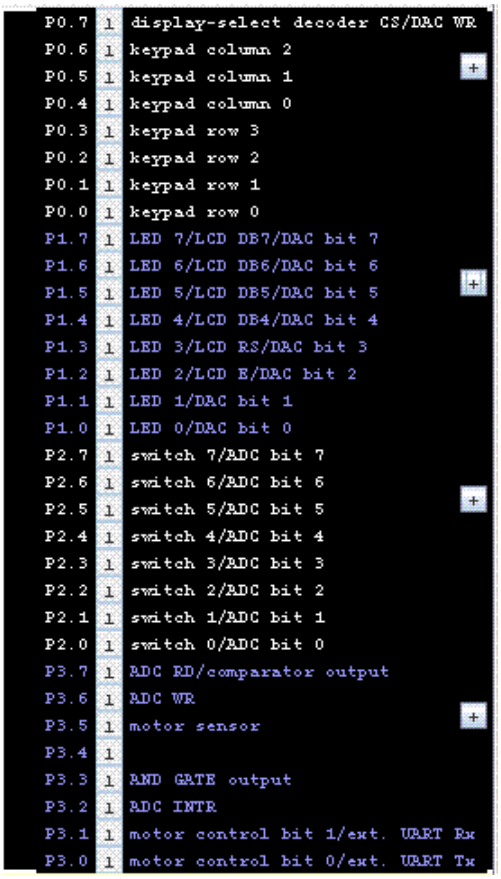


Рисунок 16 - Список регистров

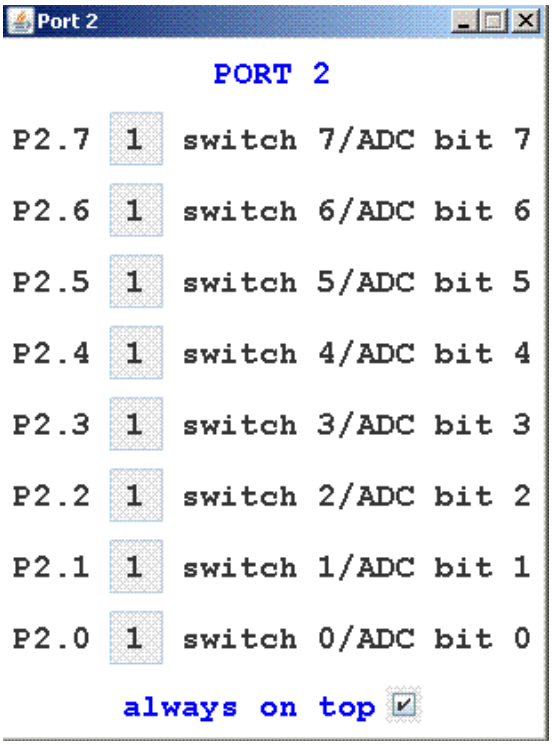


Рисунок 17 - Коммутация выводов P2.0-P2.7 и их состояния второго порта микроконтроллера

Заключение

В ходе выполнения курсовой работы были написаны программы реализующие логические функции и управляющий алгоритм с оператором goto, на языках C++ и с использованием ассемблера MCS51

Проведенные испытания показали, что программа удовлетворяет требованиям технического задания.

Оформлена документация, содержащая в себе описание требований к программе, методы тестирования программы, алгоритм решения задачи,описание и результаты тестирования программы, описание работы с программой, текст программы, выводы по проведённой работе, приложения к документации.

Список используемых источников

1) https://www.edsim51.com/introduction.html - официальный сайт

2) https://pue8.ru/protsessory/708-mikrokontrollery-mcs-51-programmnaya-model-struktura-komand-chast-5.html - пособие по микроконтроллерам MCS-51

3) https://www.circuitstoday.com/8051-simulator

Пособие по микроконтроллерам 8051

АЛГОРИТМЫ

Приложение А

(обязательное)

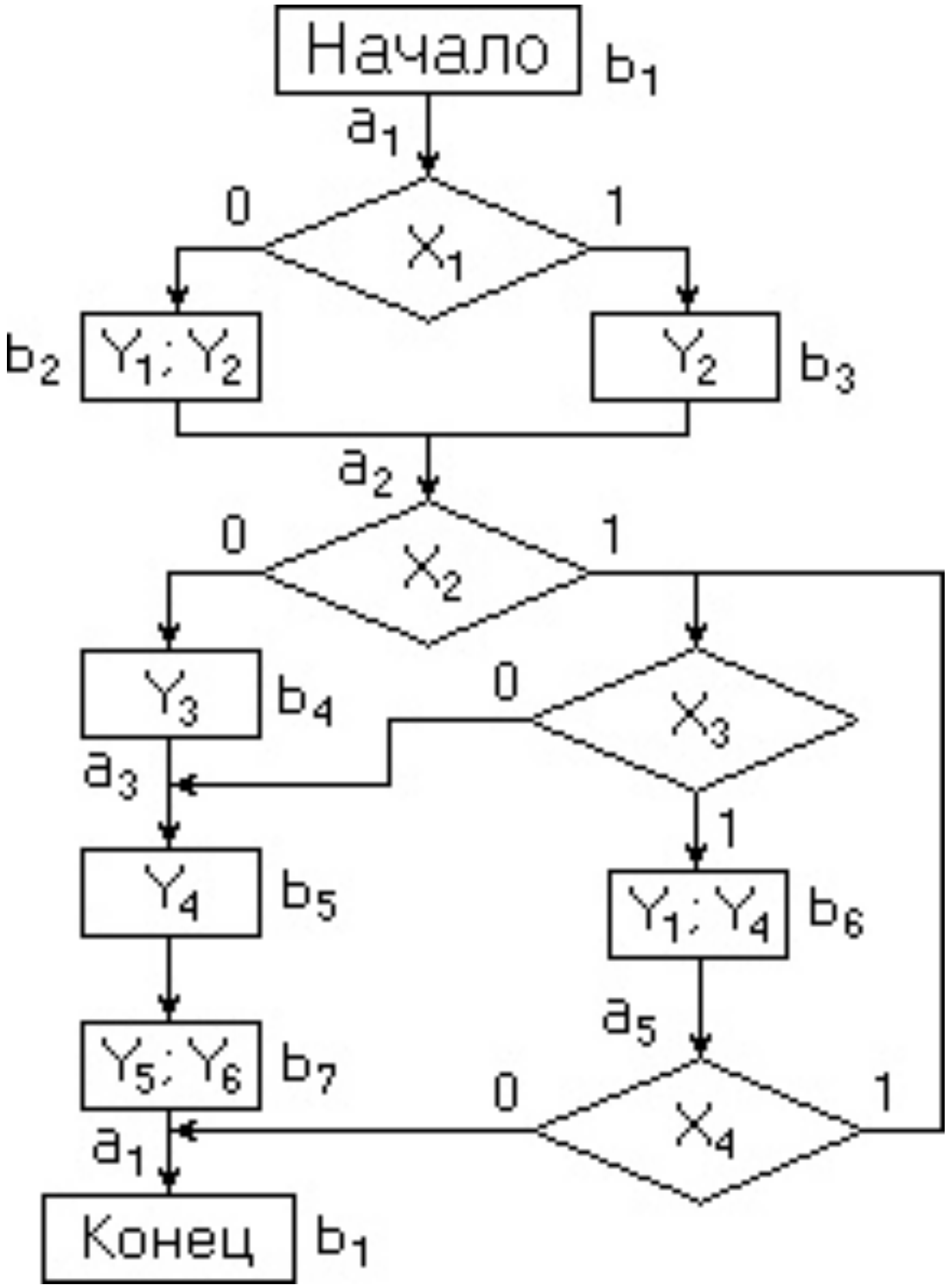


Рисунок А.1 - ГСА

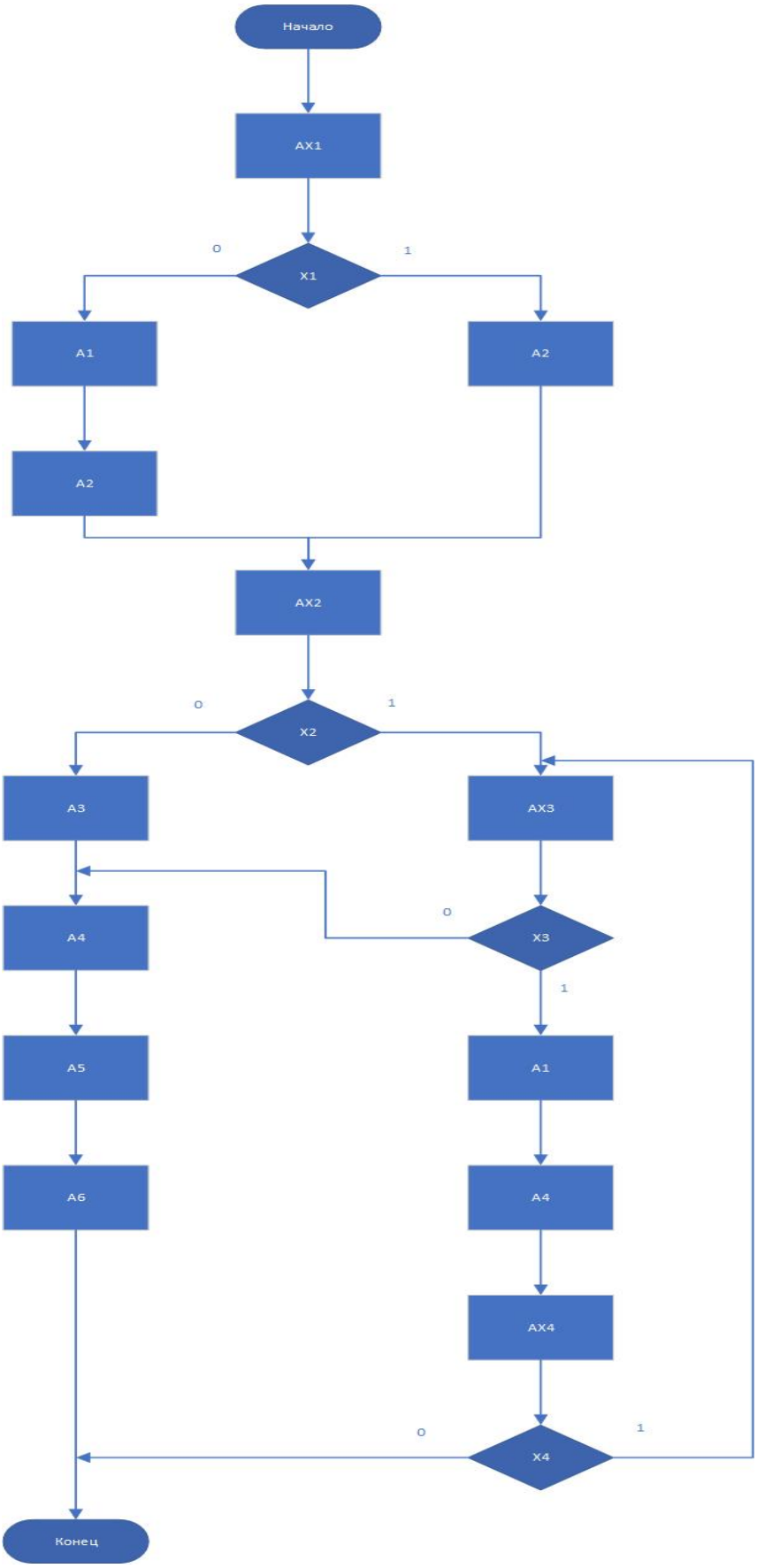


Рисунок А.2 - Схема программы

СИСТЕМА КОМАНД МИКРОКОНТРОЛЛЕРА MCS-51

Приложение Б

(обязательное)

Мнемоника	Режимы адресации	Время, такт	Действие
Арифметические инструкции			
ADD,A,byte	Dir,Ind,Reg,Imm	1	A=A+byte
ADDC A,byte	Dir,Ind,Reg,Imm	1	A=A+byte+C
DA A	Аккумулятор	1	Десятичня коррекция
DEC A	Аккумулятор	1	A=A-1
DEC byte	Dir,Ind,Reg	1	byte=byte-1
DIV AB	Аккумулятор и B	4	A=[A/B],B=mod(A/B)
INC A	Аккумулятор	1	A=A+1
INC byte	Dir,Ind,Reg	1	byte=byte+1
INC DPTR	DPTR	2	DPTR=DPTR+!
MUL AB	Аккумулятор и B	4	B,A=B*A
SUBB A,byte	Dir,Ind,Reg,Imm	1	A=A-byte-C
Логические инструкции			
ANL A,byte	Dir,Ind,Reg,Imm	1	A=A&byte
ANL byte,A	Dir	1	byte= byte&A
ANL byte,#data	Dir	2	byte= byte&#data
CLR A	Аккумулятор	1	A=00h
CPL A	Аккумулятор	1	A=^(A)
ORL A,byte	Dir,Ind,Reg,Imm	1	A=A v byte

Рисунок Б.1 – Система команд микроконтроллера MCS-51

Мнемоника	Режимы адресации	Время, такт	Действие
ORL byte,A	Dir	1	byte= byte v A
ORL byte,#data	Dir	2	byte= byte v #data
RL A	Аккумулятор	1	A=(A)ЦСЛ на 1бит
RLC A	Аккумулятор	1	A=(A)ЦСЛ на 1бит через C
RR A	Аккумулятор	1	A=(A)ЦСП на 1бит
RRC A	Аккумулятор	1	A=(A)ЦСП на 1бит через C
SWAP A	Аккумулятор	1	A(7...4) <=> A(3...0)
XRL A,byte	Dir,Ind,Reg,Imm	1	A=A ⊕ byte
XRL byte,A	Dir	1	byte= byte ⊕ A
XRL byte,#data	Dir	2	byte= byte ⊕ #data
Инструкции передачи данных во внутренней памяти данных			
MOV A,src	Dir,Ind,Reg,Imm	1	A=src
MOV dest, A	Dir,Ind,Reg	1	dest=A
MOV dest, src	Dir,Ind,Reg,Imm	1	dest=src
MOV DPTR,#data	Imm	2	DPTR=data
PUSH src	Dir	2	SP=SP+1; @SP=src
POP dest	Dir	2	dest=@SP; SP=SP-1
XCH A,byte	Dir,Ind,Reg	1	A <=> byte
XCHD A,@Ri	Ind	1	A <=> @Ri

Рисунок Б.2 – Система команд микроконтроллера MCS-51

Инструкции передачи данных, использующие внешнюю память данных			
MOVX A,@Ri		2	A=@Ri
MOVX @Ri,A		2	@Ri=A
MOVX A,@DPTR		2	A=@DPTR
MOVX @DPTR,A		2	@DPTR=A
Инструкции передачи данных, использующие память программ			
MOVC A,@A+DPTR		2	A=@(A+DPTR)
MOVC A,@A+PC		2	A=@(A+PC)
Булевы инструкции			
ANL C,bit	bit	2	C=C & bit
ANL C,/bit	bit	2	C=C & ^bit
CLR C	Битовый аккумулятор	1	C=0
CLR bit	bit	1	bit=0

Рисунок Б.3 – Система команд микроконтроллера MCS-51

Мнемоника	Режимы адресации	Время, такт	Действие
CPL C	Битовый аккумулятор	1	C=^C
CPL bit	bit	1	bit=^bit
JC rel		2	Переход, если C=1
JNC rel		2	Переход, если C=0
JB bit,rel	bit	2	Переход, если bit=1
JNB bit,rel	bit	2	Переход, если bit=0
JBC bit,rel	bit	2	Переход, если bit=1; bit=0
MOV C,bit	bit	2	C=bit
MOV bit,C	bit	2	bit=C
ORL C,bit	bit	2	C=C v bit
ORL C,/bit	bit	2	C=C v ^bit
SETB C	Битовый аккумулятор	1	C=1
SETB bit	bit	1	bit=1

Рисунок Б.4 – Система команд микроконтроллера MCS-51

Инструкции безусловных переходов			
CALL adr ACALL adr LCALL adr		2	Вызов подпрограммы переход на $\pm 2^{11}$ байт переход на $\pm 2^{16}$ байт
JMP adr AJMP adr LJMP adr SJMP adr		2	PC=adr переход на $\pm 2^{11}$ байт переход на $\pm 2^{16}$ байт переход на $\pm 2^{8}$ байт
JMP @A+DPTR		2	PC=A+DPTR
NOP		1	Пустая команда
RET		2	Возврат из подпрограммы
RETI		2	Возврат из обработчика прерывания
Инструкции условных переходов			
CJNE A,byte,rel	Dir,Imm	2	Переход, если A≠byte
CJNE byte,#data,rel	Ind,Reg	2	Переход, если byte≠#data
DJNZ byte,rel	Dir,Reg	2	byte=byte-1; переход, если byte≠0
JZ rel	Аккумулятор	2	Переход, если A=0
JNZ rel	Аккумулятор	2	Переход, если A≠0

Рисунок Б.5 – Система команд микроконтроллера MCS-51