

**МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РФ
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Кафедра «Вычислительная техника»

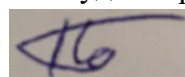
**Курсовая работа
по дисциплине «Метод моделирования цифровых устройств»
на тему «Моделирование микропроцессорного управляющего
устройства»**

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

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

ПГУ 1.090501.12

Выполнил: студент гр.18ВО2



Паняев А.В.

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 Паняев А.В. Группа 18BO2

Тема_проекта: Моделирование микропроцессорного управляющего
устройства. Вариант №12

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: С, С++, 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 C++ 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 C (реализация функции алгебры логики на C).

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

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

4. Моделирование управляющего алгоритма на языке C.

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 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.

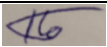
Студент _____ /Паняев А.В./

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.

					ПГУ 1.090501.12			
Изм.	Лист	№ докум	Подпись	Дата				
Разраб.		Паняев А.В			Моделирование микропроцессорного управляющего устройства Пояснительная записка	Лит.	Лист	Листов
Провер.		Зинкин С.А.					3	42
Н. Контр.						Гр. 18BO2		
Утверд.								

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

Микроконтроллеры используются во всех сферах жизнедеятельности человека, устройствах, которые окружают его. Простота подключения и большие функциональные возможности. С помощью программирования микроконтроллера можно решить многие практические задачи аппаратной техники.

В современном мире трудно найти область техники, где не применялись бы микропроцессоры.

Применение МК можно разделить на два этапа: первый - программирование, когда пользователь разрабатывает программу и прошивает ее непосредственно в кристалл, и второй - согласование спроектированных исполнительных устройств с запрограммируемым МК. Значительно облегчают отладку программы на первом этапе - симулятор, который наглядно моделирует работу микропроцессора. На втором этапе для отладки используется внутрисхемный эмулятор, который является сложным и дорогим устройством, зачастую недоступным рядовому пользователю.

В тоже время в литературе мало уделено внимания вопросам обучения программированию некоторых недорогих МК, в сочетании с реальными исполнительными устройствами.

Разработка макета программатора, отличающегося простотой, наглядностью и низкой себестоимостью, становится необходимой как для самого программирования кристаллов, так и для наглядного обучения широкого круга пользователей основам программирования МК.

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 $F1 = \neg(A \vee B) = (\neg A \& \neg B)$. Построена ее таблица истинности (Таблица 1).

Таблица 1 - Таблица истинности функции F1

a	b	$a \vee b$	$\neg(a \vee b)$	$\neg a$	$\neg b$	$\neg a \& \neg b$	$\neg(a \vee b) = (\neg a \& \neg b)$
0	0	0	1	1	1	1	1
0	1	1	0	1	0	0	1
1	0	1	0	0	1	0	1
1	1	1	0	0	0	0	1

Дана функция $F2 = \neg X \& \neg Y \vee \neg Z$. Построена ее таблица истинности (Таблица 2)

Таблица 2 - Таблица истинности функции F2

x	y	z	$\neg x$	$\neg y$	$\neg z$	$\neg x \& \neg y$	$\neg x \& \neg y \vee \neg z$
0	0	0	1	1	1	1	1
0	0	1	1	1	0	1	1
0	1	0	1	0	1	0	1
0	1	1	1	0	0	0	0
1	0	0	0	1	1	0	1
1	0	1	0	1	0	0	0
1	1	0	0	0	1	0	1
1	1	1	0	0	0	0	0

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;

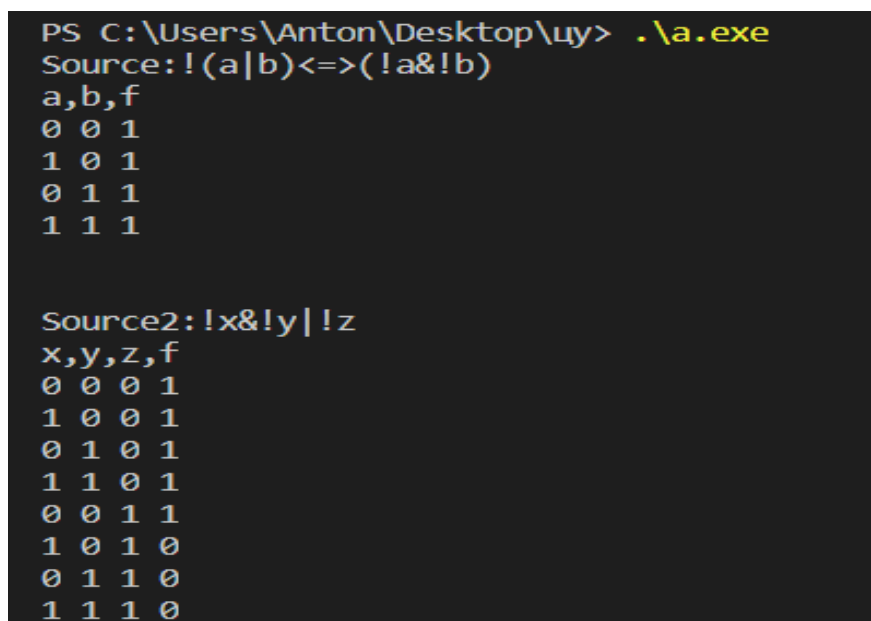
int main(){
    bool a=0, b=0, f;
    printf("Source:\n");
    printf("a,b,f\n");
    for (int i=0;i<2;i++){
        for (int j=0;j<2;j++){
            f=!(a||b)==(!a&&!b);
            printf("%i %i %i \n",a,b,f);
            a=!a;        }
    }
```

```

    b=!b;  }
printf("\n\n\n");
    bool x=0, y=0,z=0, f1;
printf("Source:\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.



```

PS C:\Users\Anton\Desktop\цц> .\a.exe
Source:!(a|b)<=>(!a&!b)
a,b,f
0 0 1
1 0 1
0 1 1
1 1 1

Source2:!x&!y|!z
x,y,z,f
0 0 0 1
1 0 0 1
0 1 0 1
1 1 0 1
0 0 1 1
1 0 1 0
0 1 1 0
1 1 1 0

```

Рисунок 1 - Результат работы программы F1

1.2 Моделирование сложной комбинационной схемы на языке С (реализация функции алгебры логики, 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
12	0, 3, 4, 5, 7, 8, 10, 11

Ход работы:

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

Таблица 4 - таблица истинности функции $F(x_1, x_2, x_3, x_4)$

№	x_1	x_2	x_3	x_4	F
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0

Продолжение таблицы 4

10	1	0	1	0	1
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:

$$F = (\neg x_1 \& \neg x_2 \& \neg x_3 \& \neg x_4) \vee (\neg x_1 \& \neg x_2 \& x_3 \& x_4) \vee (\neg x_1 \& x_2 \& \neg x_3 \& \neg x_4) \vee (\neg x_1 \& x_2 \& \neg x_3 \& x_4) \vee (\neg x_1 \& x_2 \& x_3 \& x_4) \vee (x_1 \& \neg x_2 \& \neg x_3 \& \neg x_4) \vee (x_1 \& \neg x_2 \& x_3 \& \neg x_4) \vee (x_1 \& \neg x_2 \& x_3 \& x_4)$$

С помощью диаграммы Вейча была минимизирована СДНФ ($F_1(x_1, x_2, x_3, x_4)$).

Получена минимизированная функция

$$F_2(x_1, x_2, x_3, x_4) = (x_1 \& \neg x_2 \& \neg x_4) \vee (x_1 \& \neg x_2 \& x_3) \vee (\neg x_1 \& \neg x_3 \& \neg x_4) \vee (\neg x_1 \& x_2 \& \neg x_3) \vee (\neg x_1 \& x_3 \& x_4).$$

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

Таблица 5 - таблица истинности функции F_2

№	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	0	0
3	0	0	1	1	1	1
4	0	1	0	0	1	1
5	0	1	0	1	1	1
6	0	1	1	0	0	0
7	0	1	1	1	1	1
8	1	0	0	0	1	1

Продолжение таблицы 5

9	1	0	0	1	0	0
10	1	0	1	0	1	1
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,f\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 f
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&&!x2&&!x4)||((x1&&!x2&&x3)||(!x1&&!x3&&!x4)||(!x1&&x2&&!x3)||(!x1&&x3&&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&&!x2&&!x4)||((x1&&!x2&&x3)||(!x1&&!x3&&!x4)||(!x1&&x2&&!x3)||(!x1&&x3&&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:

```

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,f1
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
0 0 1 1 1
0 1 0 0 1
0 1 0 1 1
0 1 1 0 0
0 1 1 1 1
1 0 0 0 1
1 0 0 1 0
1 0 1 0 1
1 0 1 1 1
1 1 0 0 0
1 1 0 1 0
1 1 1 0 0
1 1 1 1 0
МДНФ: (x1&&!x2&&!x4)|| (x1&&!x2&&x3)|| (!x1&&!x3&&!x4)|| (!x1&&x2&&!x3)|| (!x1&&x3&&x4)
x1,x2,x3,x4,f2
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
0 0 1 1 1
0 1 0 0 1
0 1 0 1 1
0 1 1 0 0
0 1 1 1 1
1 0 0 0 1
1 0 0 1 0
1 0 1 0 1
1 0 1 1 1
1 1 0 0 0
1 1 0 1 0
1 1 1 0 0
1 1 1 1 0

```

1.3 Моделирование управляющего алгоритма на языке C

Формулировка задания:

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

Ход работы:

Алгоритм 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, x5, x6;

    cout<<"a1"<<endl;
    l1: cout<<"a2"<<endl;
    cout<<"vvod x1"<<endl;
    cin>>x1;

    if(x1){
        cout<<"vvod x2"<<endl;
        cin>>x2;
    }else {goto l1;}
    if (x2){
        cout<<"vvod x4"<<endl;
        cin>>x4;
    }else{cout<<"a3"<<endl;
        cout<<"vvod x3"<<endl;
        cin>>x3;
        if(x3){cout<<"a4"<<endl;
            cout<<"vvod x1"<<endl;
            cin>>x1;
            goto l7;
        }else{
            cout<<"a5"<<endl;
```

```

        goto le;
    }

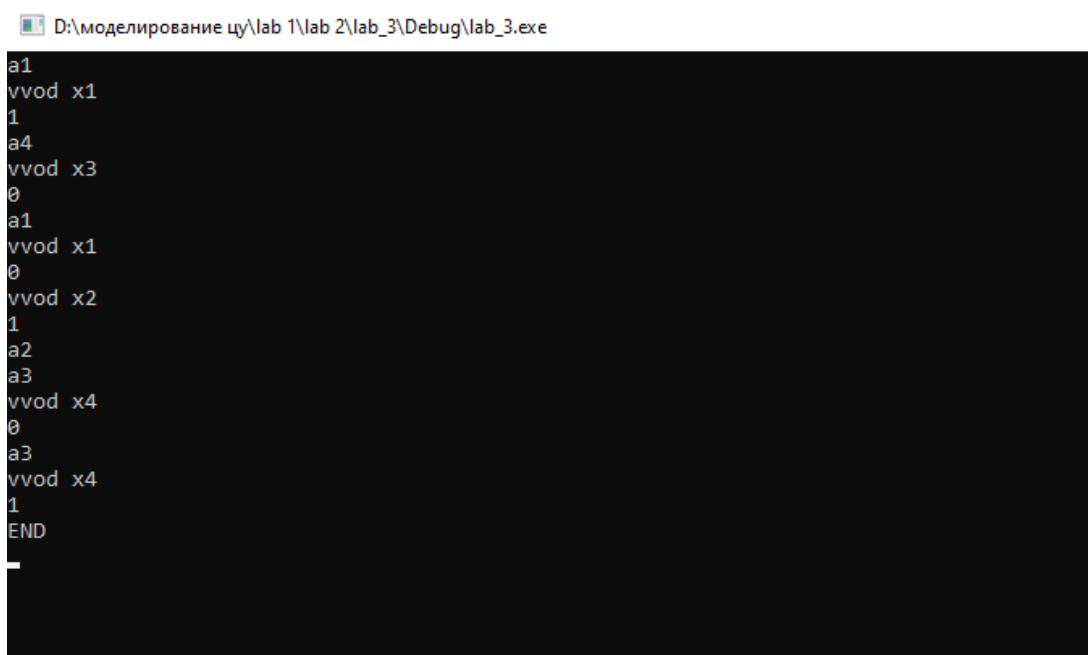
}

if (!x4){
    l3: cout<<"a6"<<endl;
    cout<<"vvod x6"<<endl;
    cin>>x6;
}else{goto l1;}

if(!x6){
    cout<<"a8"<<endl;
l4: cout<<"vvod x5"<<endl;
    cin>>x5;
}else{
    cout<<"a7"<<endl;
    cout<<"a1"<<endl;
    goto l4;
}
if(!x5){goto l5;}
else{goto l3;}
l7: if(x1){
    l5: cout<<"a3"<<endl;
    cout<<"vvod x3"<<endl;
    if(x3){
        cout<<"a5"<<endl;
        goto le;
    }else{goto l6}
}else{ cout<<"a3"<<endl;
    l6: cout<<"a4"<<endl;
    goto le;
}
le: cout<<"end"<<endl;
getch();
return 0;}

```

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



```

D:\моделирование цу\lab 1\lab 2\lab_3\Debug\lab_3.exe
a1
vvod x1
1
a4
vvod x3
0
a1
vvod x1
0
vvod x2
1
a2
a3
vvod x4
0
a3
vvod x4
1
END

```

Рисунок 3 - результат работы программы lab3

CPL C; $C = \neg Z$
 ORL C, 00; $C = \neg X \& \neg Y + \neg Z$
 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.

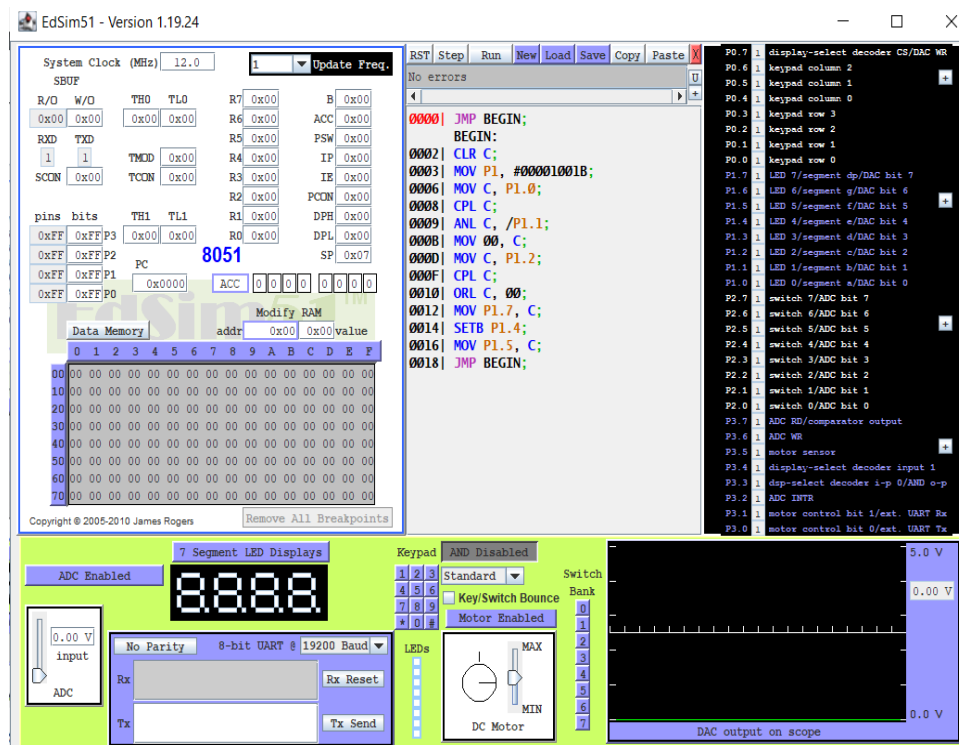


Рисунок 4 - Результат работы программы

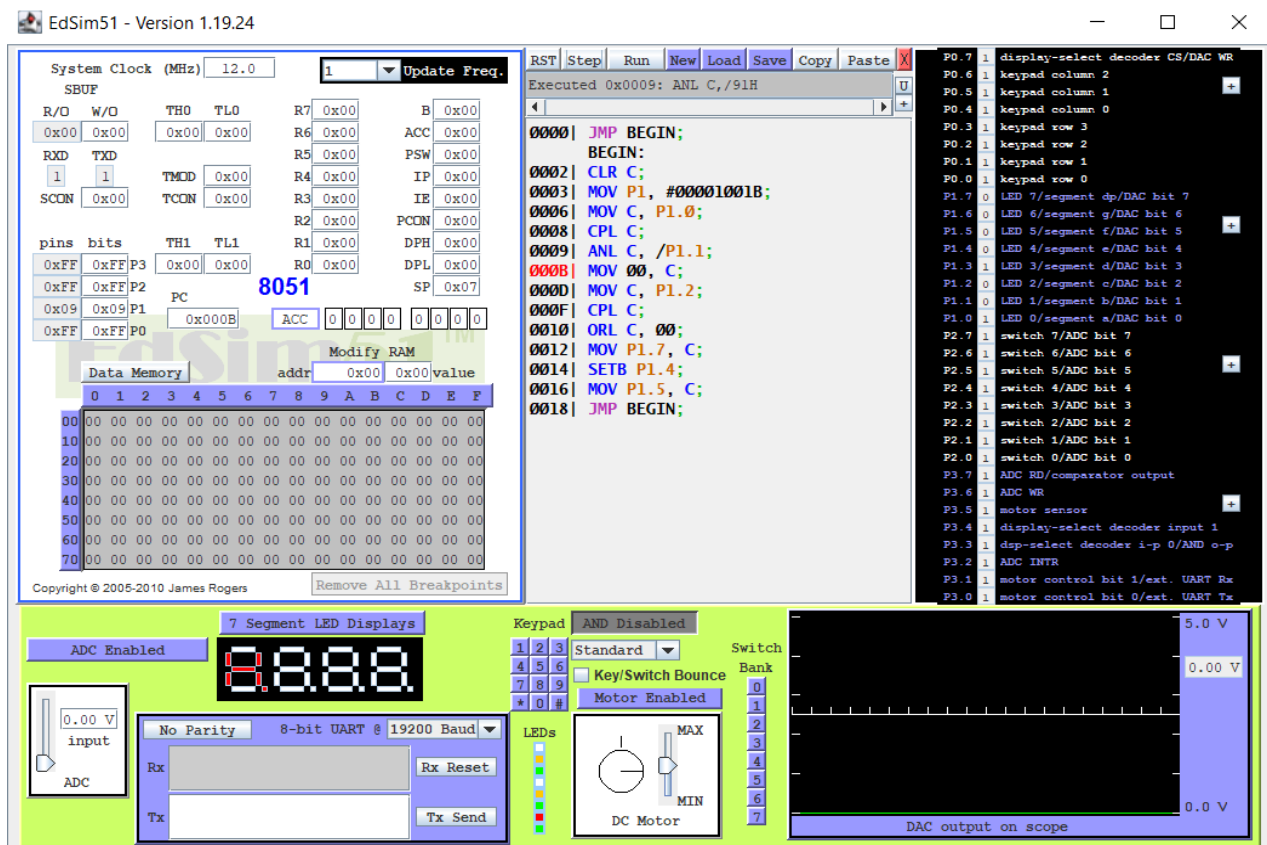


Рисунок 5 - Результат работы программы

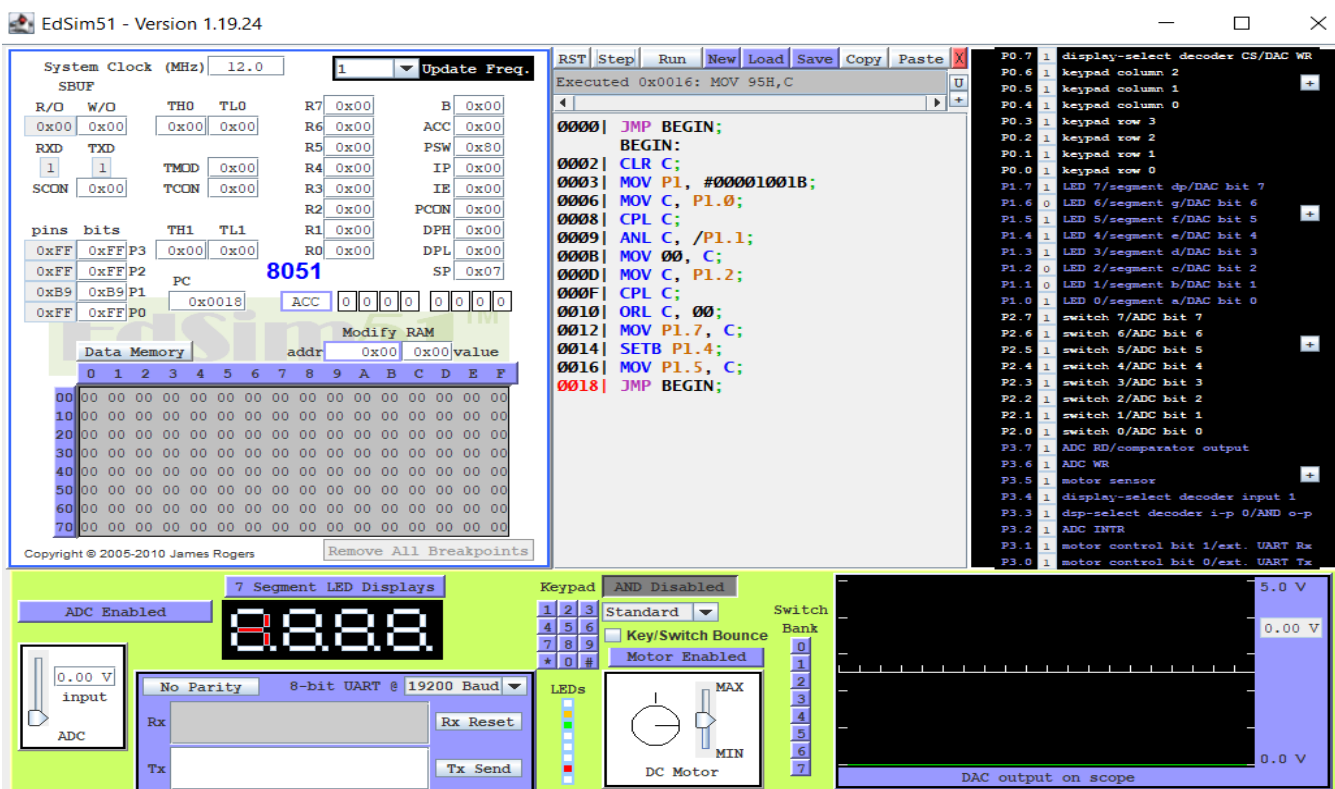


Рисунок 6 - результат работы программы

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	3	4	5	7	8	10	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	0
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
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:

$$F = (\neg x_1 \& \neg x_2 \& \neg x_3 \& \neg x_4) \vee (\neg x_1 \& \neg x_2 \& x_3 \& x_4) \vee (\neg x_1 \& x_2 \& \neg x_3 \& \neg x_4) \vee (\neg x_1 \& x_2 \& \neg x_3 \& x_4) \vee (\neg x_1 \& x_2 \& x_3 \& x_4) \vee (x_1 \& \neg x_2 \& \neg x_3 \& \neg x_4) \vee (x_1 \& \neg x_2 \& x_3 \& \neg x_4) \vee (x_1 \& \neg x_2 \& x_3 \& x_4)$$

Программа lab5 разработаны на микроконтроллере MCS-51 и предназначена для реализации сложной логической функции, заданной в форме СДНФ.

Система команд микроконтроллера MCS-51 представлена в Приложении Б 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
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
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
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:

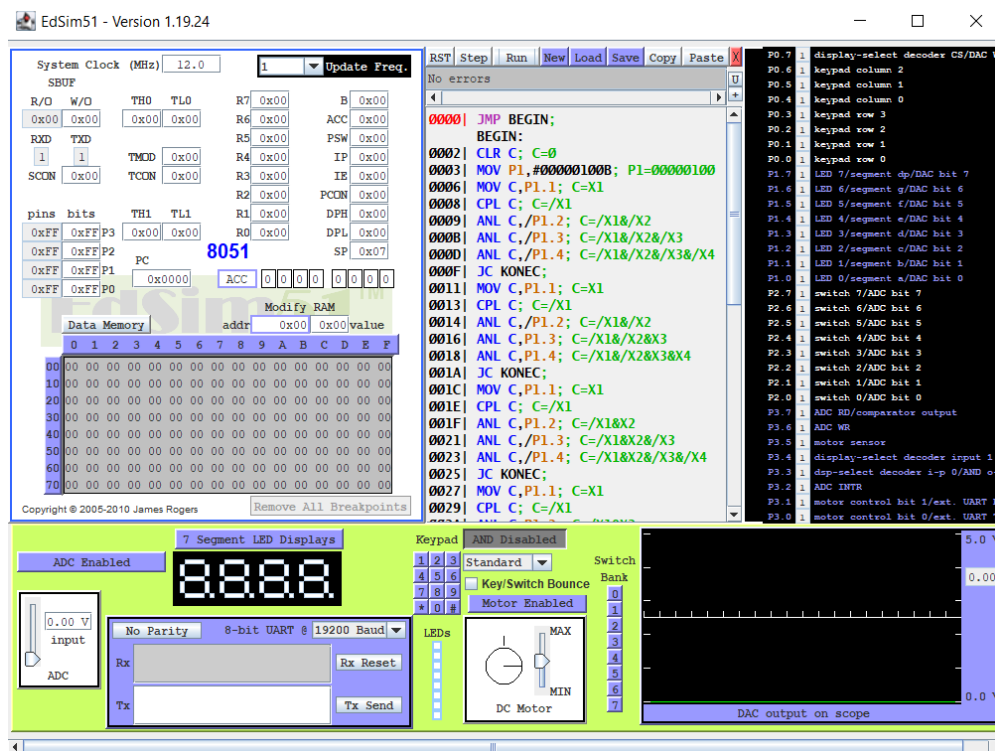


Рисунок 7 — Результаты работы программы

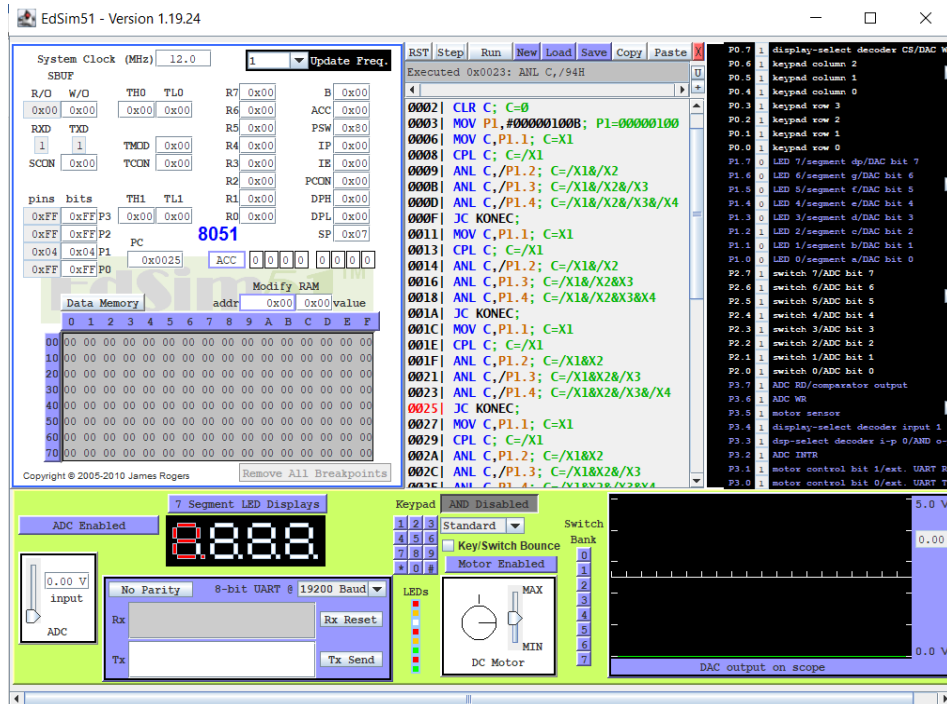


Рисунок 8 — Результаты работы программы

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, #1110B; SETB P1.0; CLR P1.0; MOV P2, #00000000B;	Ввод значений Ввод x1 - 3 Ввод x4=1 Ввод x4=0 Начало AN
	A1	MOV P2, #00000001B;	Исп. A1
	A2	A2:MOV P2, #00000010B;	Исп. A2
AX1:	AX1	MOV C, P0.1;	Ввод X1
	IF AX1=0 GOTO A2	JNC A2;	
	AX2	MOV C, P0.2;	Ввод X2
	IF AX2=0 GOTO A3	JNC A3;	
	AX4	MOV C, P0.4;	Ввод X4
	IF AX4=1 GOTO A2	JC A2;	

Продолжение таблицы 9

	A6	A6: MOV P2, #00000110B;	Исп. A6
	AX6	MOV C, P0.6;	Ввод X6
	IF AX6=0 GOTO A8	JNC A8;	
	A7	MOV P2, #00000111B;	Исп. A7
	A1	MOV P2, #00000001B;	Исп. A1
	GOTO AX5	JMP AX5;	
	A3	A3:MOV P2, #00000011B;	Исп. A3
AX3:	AX3	MOV C, P0.3;	Ввод X3
	IF AX3=0 GOTO A5	JNC A5;	
	A4	MOV P2, #00000100B;	Исп. A4
	AX1	MOV C, P0.1;	Ввод X1
	IF AX1=0 GOTO A32	JNC A32;	
	A3	A33:MOV P2, #00000011B;	Исп. A3
	AX3	MOV C, P0.3;	Ввод X3
	IF AX3=0 GOTO A42	JNC A42;	
	A5	MOV P2, #00000101B;	Ввод A5
	GOTO AK	JMP AK;	
	A3	A32:MOV P2, #00000011B;	Исп. A3
	A4	A42:MOV P2, #00000100B;	Исп. A4
	GOTO AK	JMP AK;	
	A8	A8:MOV P2, #00001000B;	Исп. A8
AX5:	AX5	AX5: MOV C, P0.5;	Ввод X5
	IF AX5=1 GOTO A6	JC A6;	
	GOTO A33	JMP A33;	
	A5	A5:MOV P2, #00000101B;	Исп. A5
	AK	AK:MOV P2, #00001001B;	
		JMP BEGIN;	

Результат работы программы приведён на рисунках 9 -11.

Результат работы программы

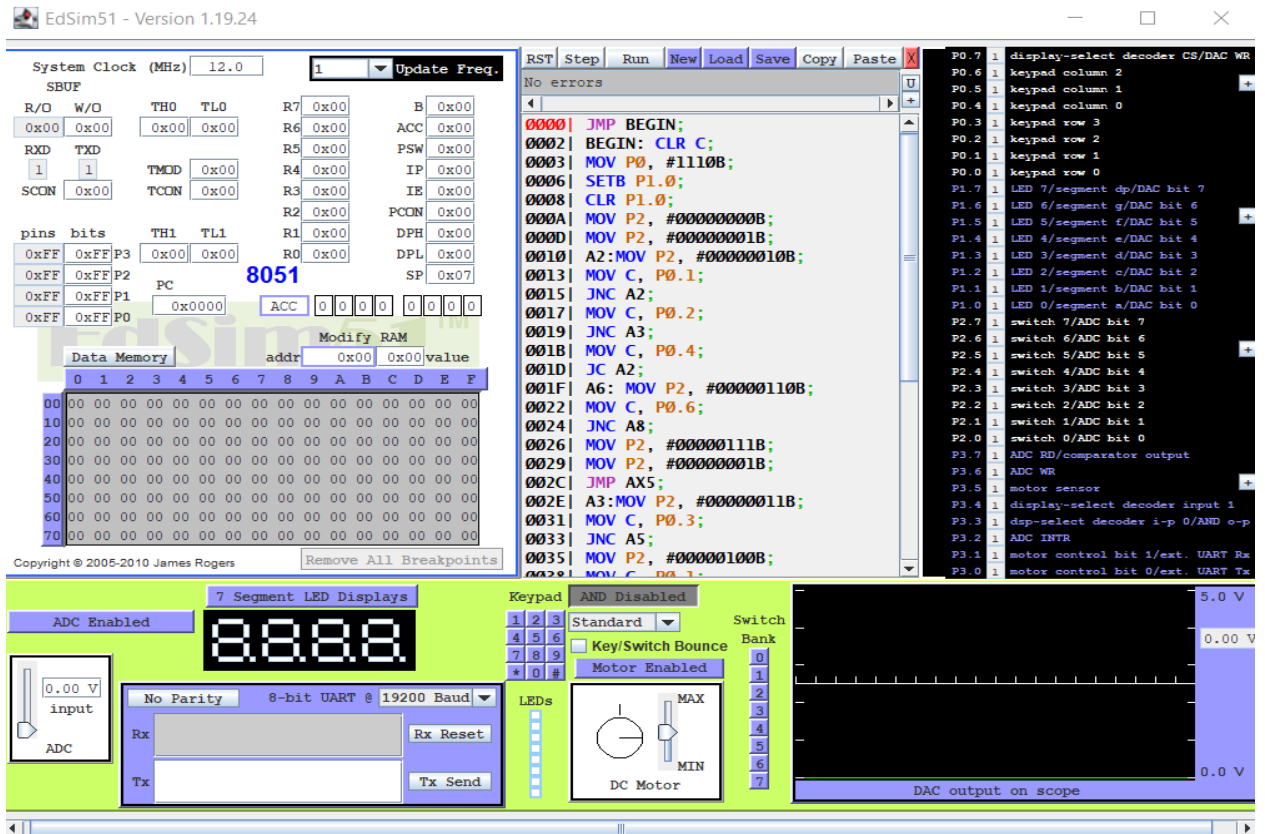


Рисунок 9 - Результат работы программы

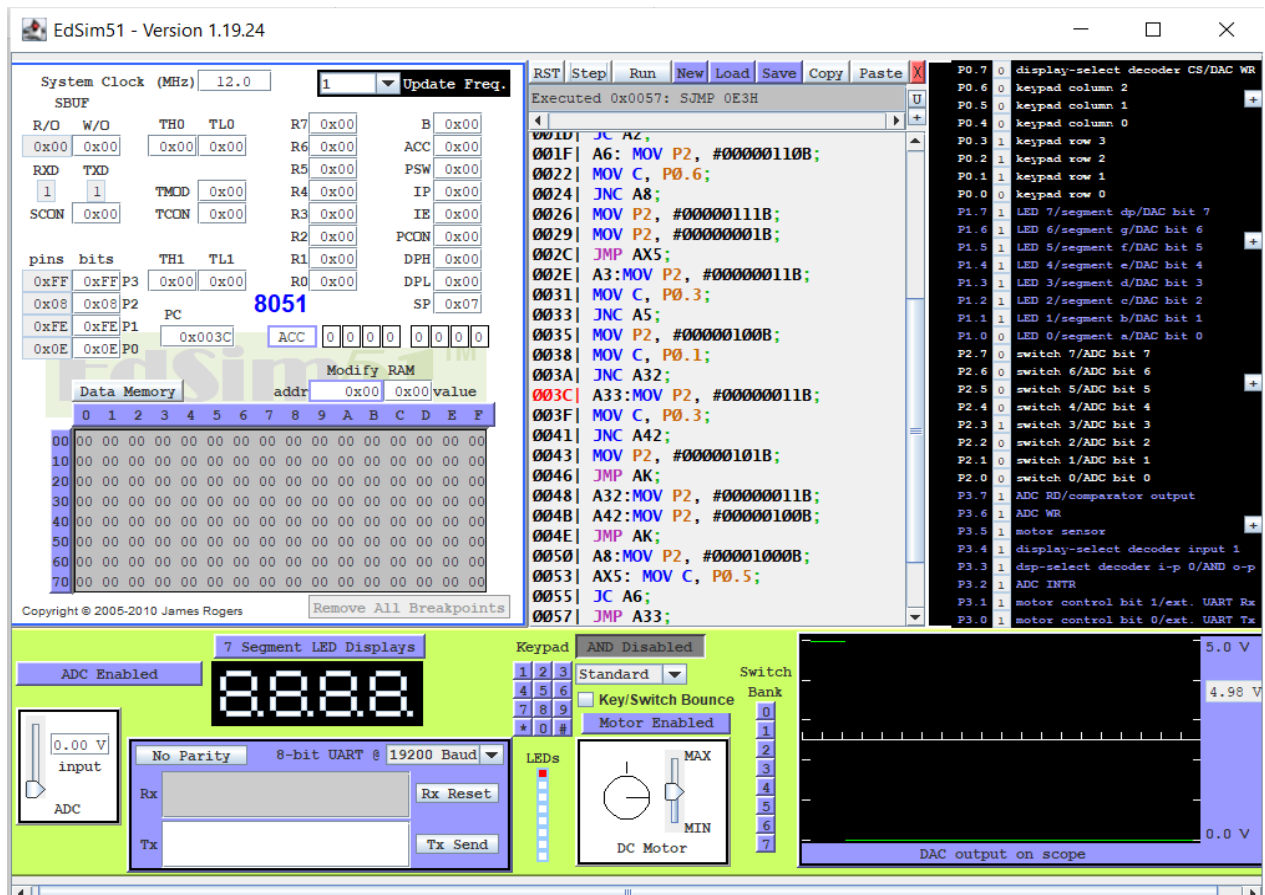


Рисунок 10 - Результат работы программы

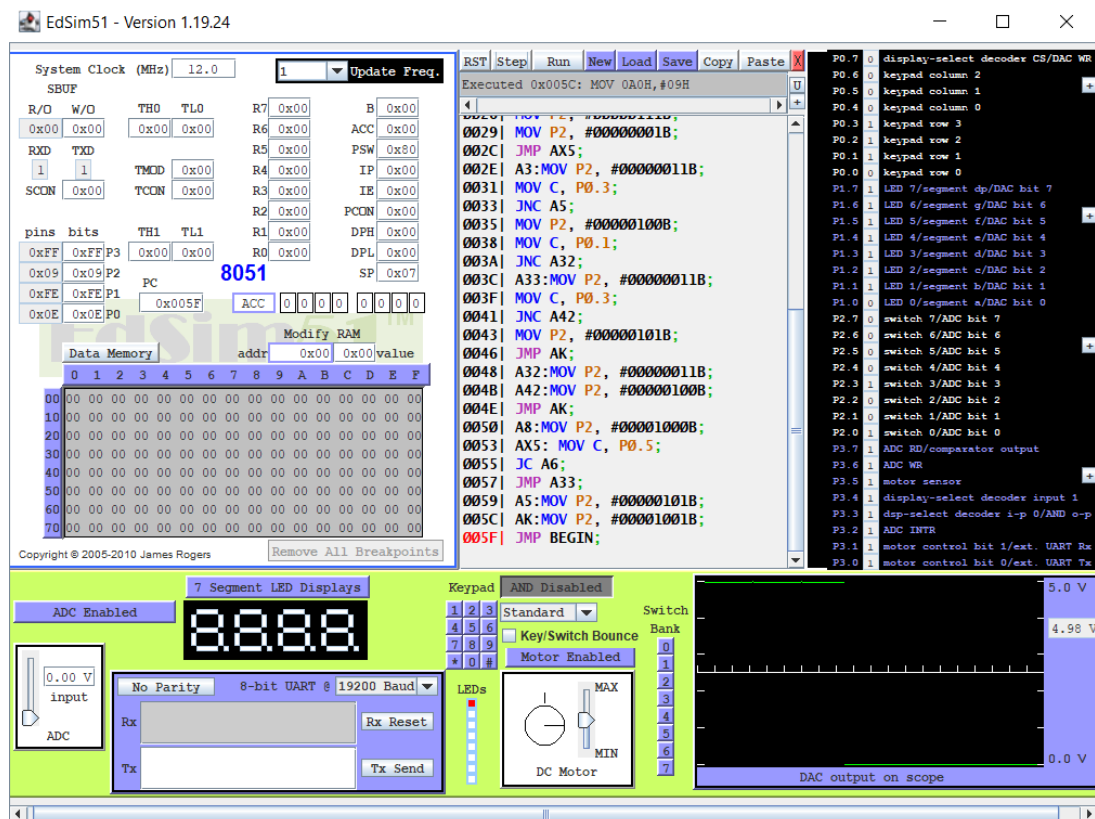


Рисунок 11 - Результат работы программы

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.

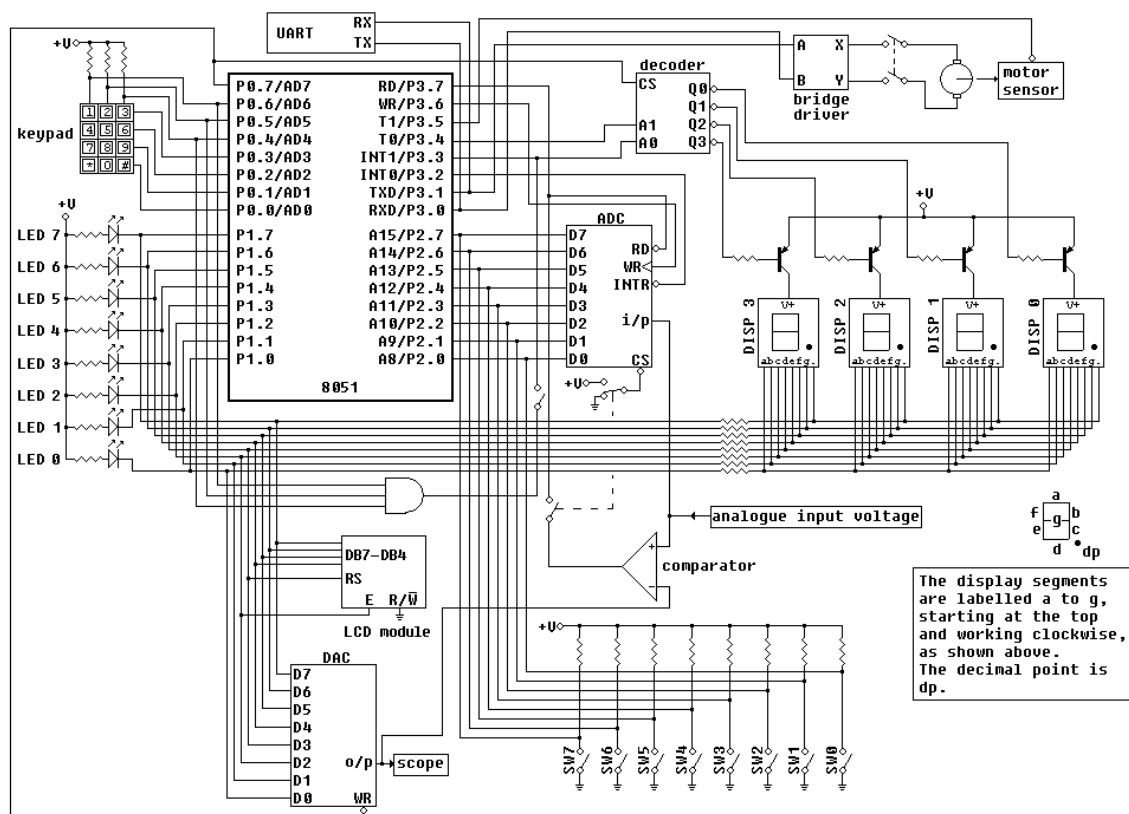


Рисунок 12 - Функциональная схема модели лабораторного стенда EdSim51

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.

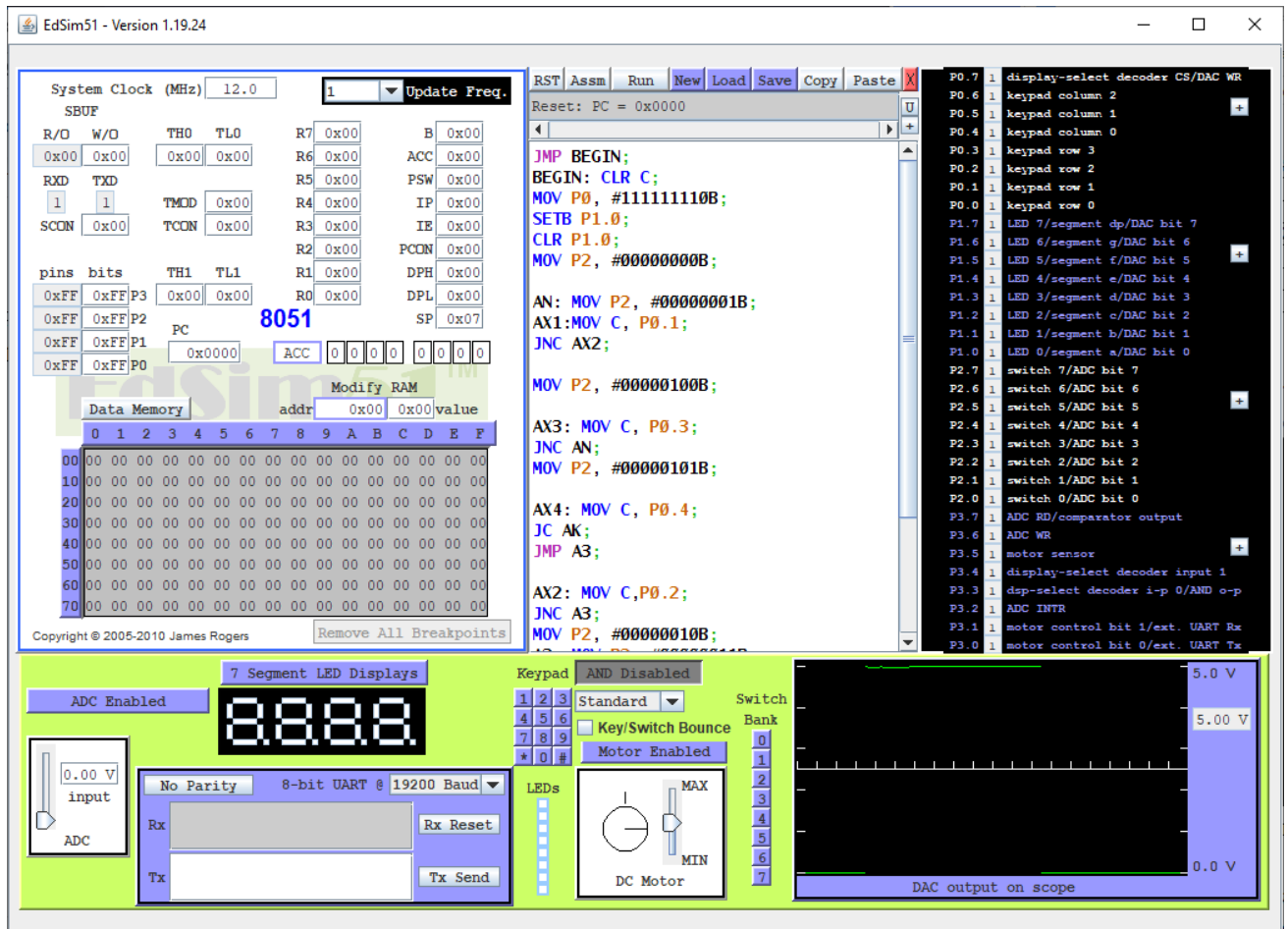
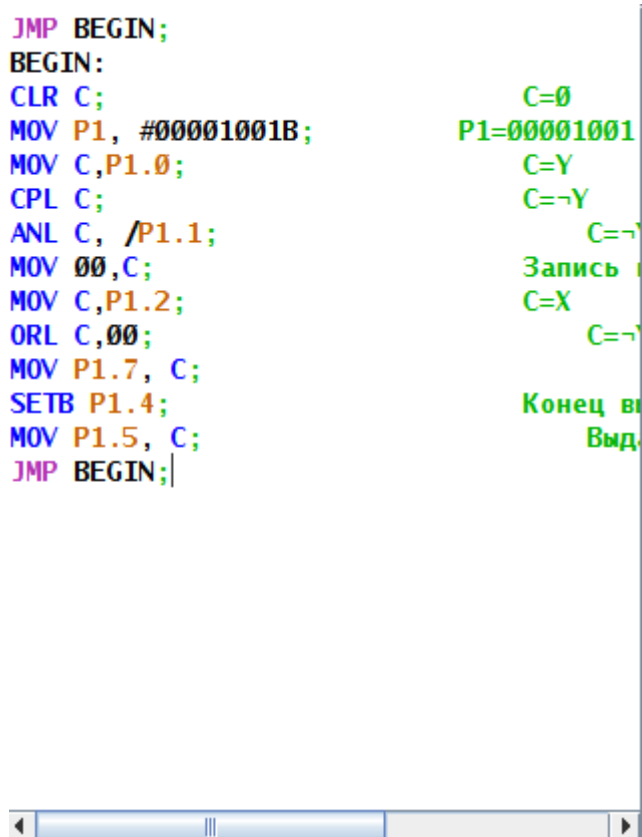


Рисунок 13 - Внешний вид модели лабораторного стенда EdSim51

3. Руководство оператора

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



```

    JMP BEGIN;
    BEGIN:
    CLR C;                C=0
    MOV P1, #00001001B;   P1=00001001
    MOV C,P1.0;           C=Y
    CPL C;               C=¬Y
    ANL C, /P1.1;         C=¬Y
    MOV 00,C;             Запись 1
    MOV C,P1.2;           C=X
    ORL C,00;             C=¬Y
    MOV P1.7, C;
    SETB P1.4;           Конец в
    MOV P1.5, C;         Выд
    JMP BEGIN;
  
```

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).

```

RST Step Run New Load Save Copy Paste X
Executed 0x0003: MOV 90H, #09H
0000| JMP BEGIN;
      BEGIN:
0002| CLR C;
0003| MOV P1, #00001001B;      P1=
0006| MOV C, P1.0;
0008| CPL C;
0009| ANL C, /P1.1;
000B| MOV 00, C;              3am
000D| MOV C, P1.2;
000F| ORL C, 00;
0011| MOV P1.7, C;
0013| SETB P1.4;
0015| MOV P1.5, C;
0017| JMP BEGIN;

```

Рисунок 15 - Панель ассемблерного кода

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

P0.7	1	display-select decoder CS/DAC WR	
P0.6	1	keypad column 2	
P0.5	1	keypad column 1	+
P0.4	1	keypad column 0	
P0.3	1	keypad row 3	
P0.2	1	keypad row 2	
P0.1	1	keypad row 1	
P0.0	1	keypad row 0	
P1.7	1	LED 7/LCD DB7/DAC bit 7	
P1.6	1	LED 6/LCD DB6/DAC bit 6	
P1.5	1	LED 5/LCD DB5/DAC bit 5	+
P1.4	1	LED 4/LCD DB4/DAC bit 4	
P1.3	1	LED 3/LCD RS/DAC bit 3	
P1.2	1	LED 2/LCD E/DAC bit 2	
P1.1	1	LED 1/DAC bit 1	
P1.0	1	LED 0/DAC bit 0	
P2.7	1	switch 7/ADC bit 7	
P2.6	1	switch 6/ADC bit 6	
P2.5	1	switch 5/ADC bit 5	+
P2.4	1	switch 4/ADC bit 4	
P2.3	1	switch 3/ADC bit 3	
P2.2	1	switch 2/ADC bit 2	
P2.1	1	switch 1/ADC bit 1	
P2.0	1	switch 0/ADC bit 0	
P3.7	1	ADC RD/comparator output	
P3.6	1	ADC WR	
P3.5	1	motor sensor	+
P3.4	1		
P3.3	1	RND GATE output	
P3.2	1	ADC INTR	
P3.1	1	motor control bit 1/ext. UART Rx	
P3.0	1	motor control bit 0/ext. UART Tx	

Рисунок 16 - Список регистров

PORT 2			
P2.7	1	switch 7/ADC bit 7	
P2.6	1	switch 6/ADC bit 6	
P2.5	1	switch 5/ADC bit 5	
P2.4	1	switch 4/ADC bit 4	
P2.3	1	switch 3/ADC bit 3	
P2.2	1	switch 2/ADC bit 2	
P2.1	1	switch 1/ADC bit 1	
P2.0	1	switch 0/ADC bit 0	
always on top			☒

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)

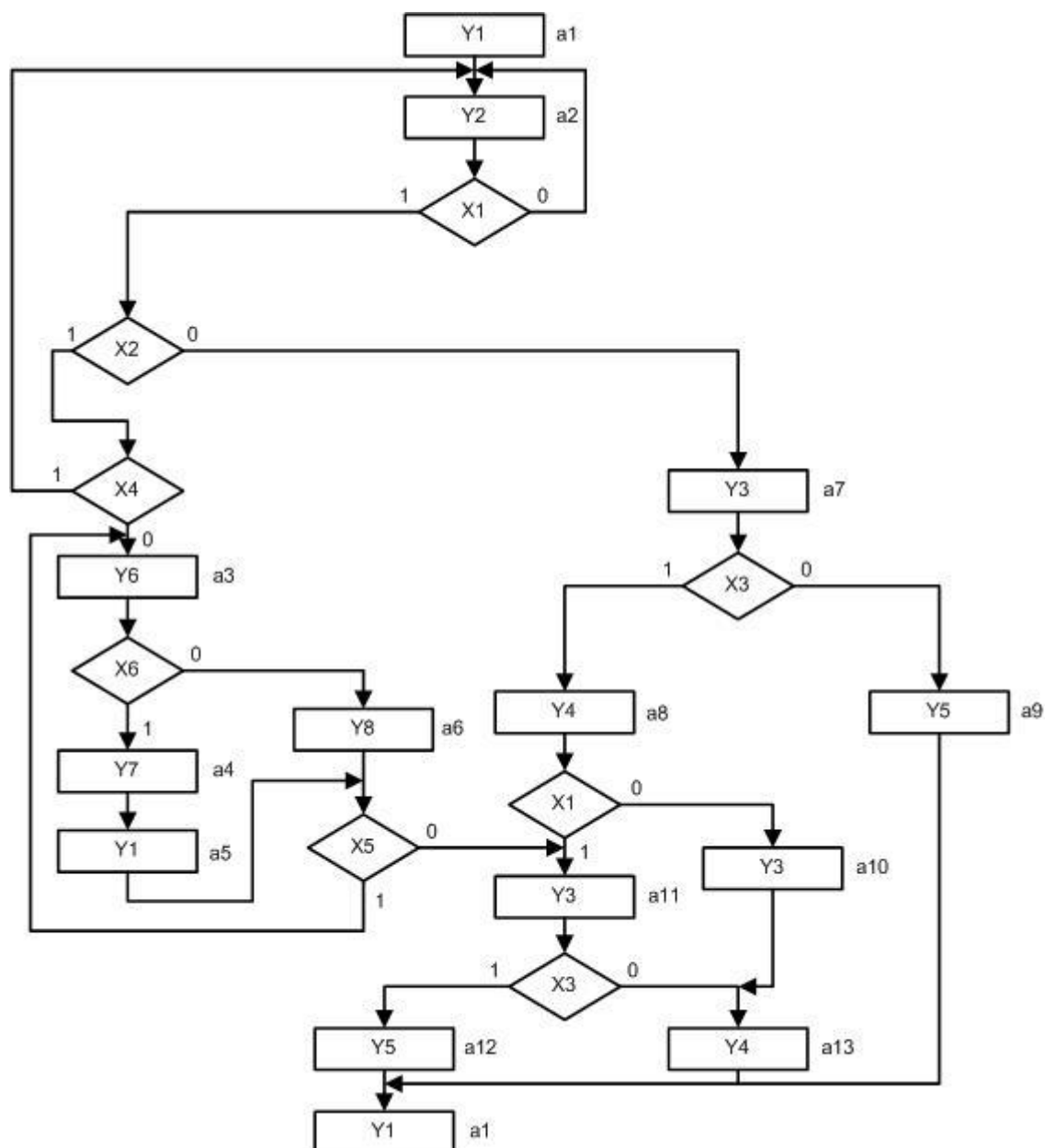


Рисунок А.1 - ГСА

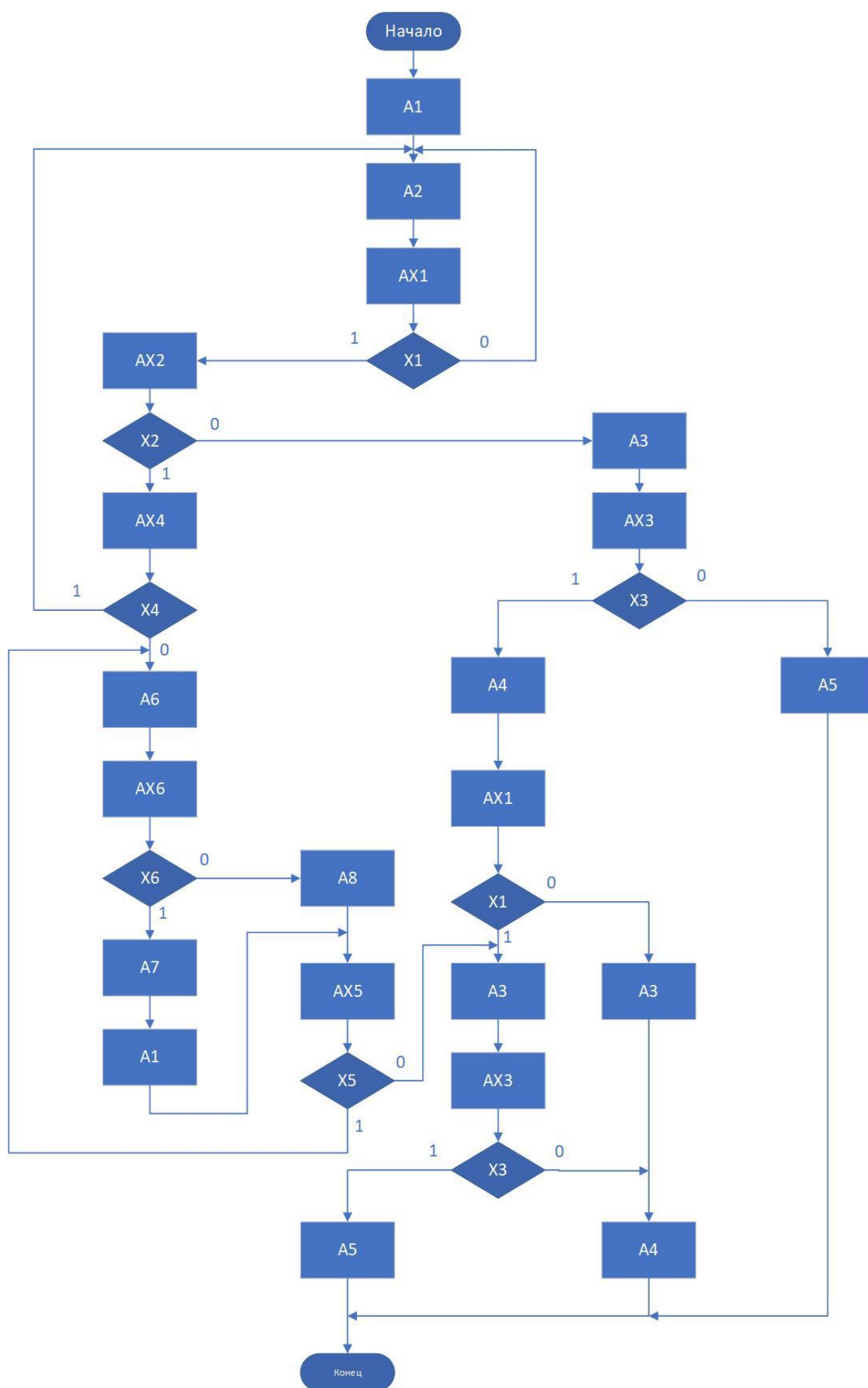


Рисунок А.2 - Схема программы

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	Десятичная коррекция
DEC A	Аккумулятор	1	$A=A-1$
DEC byte	Dir,Ind,Reg	1	$byte=byte-1$
DIV AB	Аккумулятор и B	4	$A=[A/B], B=\text{mod}(A/B)$
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	$A=\sim(A)$
ORL A,byte	Dir,Ind,Reg,Imm	1	$A=A\vee byte$

Рисунок Б.1 – Система команд микроконтроллера MCS-51

Мнемоника	Режимы адресации	Время, такт	Действие
ORL byte,A	Dir	1	$byte=byte\vee A$
ORL byte,#data	Dir	2	$byte=byte\vee \#data$
RL A	Аккумулятор	1	$A=(A)\text{ЦСЛ на 1бит}$
RLC A	Аккумулятор	1	$A=(A)\text{ЦСЛ на 1бит через C}$
RR A	Аккумулятор	1	$A=(A)\text{ЦСП на 1бит}$
RRC A	Аккумулятор	1	$A=(A)\text{ЦСП на 1бит через C}$
SWAP A	Аккумулятор	1	$A(7...4) \Leftrightarrow A(3...0)$
XRL A,byte	Dir,Ind,Reg,Imm	1	$A=A\oplus byte$
XRL byte,A	Dir	1	$byte=byte\oplus A$
XRL byte,#data	Dir	2	$byte=byte\oplus \#data$
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	$A \Leftrightarrow byte$
XCHD A,@Ri	Ind	1	$A \Leftrightarrow @Ri$

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 $A \neq \text{byte}$
CJNE byte,#data,rel	Ind,Reg	2	Переход, если $\text{byte} \neq \#data$
DJNZ byte,rel	Dir,Reg	2	$\text{byte} = \text{byte} - 1$; переход, если $\text{byte} \neq 0$
JZ rel	Аккумулятор	2	Переход, если $A = 0$
JNZ rel	Аккумулятор	2	Переход, если $A \neq 0$

Рисунок Б.5 – Система команд микроконтроллера MCS-51