

**МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Кафедра «Вычислительная техника»

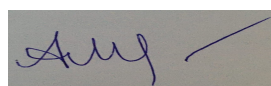
**Курсовая работа
по дисциплине «Методы моделирования цифровых устройств»
на тему «Моделирование микропроцессорного управляющего
устройства»**

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

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

ПГУ 1.090501.09

Выполнил: студент гр.18ВО2



Маслов А.Ю.

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

_____Зинкин С.А.

Работа защищена с оценкой _____

Преподаватели _____

Дата защиты _____

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 **Маслову А.Ю.** Группа **18BO2**

Тема_проекта: **Моделирование микропроцессорного управляющего**
устройства. Вариант № 9

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

2. Экспериментальная часть

Результаты отладки алгоритма программы с использованием электронного симулятора EdSim51.

Срок завершения выполнения работы по разделам

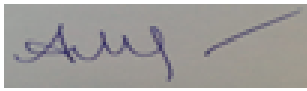
В соответствии с графиком защит работ на неделе, предшествующей зачетной сессии.

Дата выдачи задания “14” сентября 2020 г.

Дата защиты проекта: по графику

Руководитель _____ /Зинкин С.А./

Задание получил “14” сентября 2020 г.

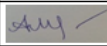
Студент  / Маслов А.Ю./

Реферат

Пояснительная записка состоит из 47 листов, 27 рисунков, 10 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 1.090501.09			
Изм.	Лист	№ докум.	Подпись	Дата	Моделирование микропроцессорного управляющего устройства Пояснительная записка	Лит.	Лист	Листов
Разраб.		Маслов А.Ю.						
Провер.		Зинкин С.А.					3	47
						Гр. 18BO2		
Н. Контр.								
Утверд.								

Содержание

Введение.....	5
1 Постановка и реализация задачи	6-32
1.1 Моделирование комбинационных схем на языке С (реализация функций алгебры логики на С)	6-9
1.2 Моделирование сложной комбинационной схемы на языке С (реализация функции алгебры логики, заданной СДНФ).....	10-15
1.3 Моделирование управляющего алгоритма на языке С.....	16-18
1.4 Моделирование комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики на языке ассемблера для микроконтроллера MCS51).....	19-22
1.5 Моделирование сложной комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики, заданной СДНФ на языке ассемблера для микроконтроллера MCS51)	23-27
1.6 Моделирование управляющего алгоритма на языке ассемблера для микроконтроллера MCS51.....	28-32
2 Описание моделирующей среды.....	33-35
3 Руководство оператора.....	36-38
Заключение.....	39
Список использованных источников.....	40
Приложение А. Алгоритмы.....	41-43
Приложение Б. Система команд микроконтроллера MCS51.....	44-47

Введение

Программирование — в обычном понимании, это процесс создания компьютерных программ. /5/

Профессиональное программирование — вполне прагматичная деятельность, направленная на получение реального программного продукта, которое требует высокой теоретической подготовленности не только в области знания языков программирования и принципов создания программ, но и в области математики, системного анализа, исследования операций, системотехники.

Программист — одна из самых востребованных специальностей в современном обществе. Он должен хорошо ориентироваться в уже имеющемся программном обеспечении вычислительной техники и автоматизированных систем, программной защите информации, стандартизации и лицензировании программных продуктов. /6/

VisualStudio 2017 - это среда быстрой разработки, в которой в качестве языка программирования используется язык C ++. /4/

Язык C++ в настоящее время считается господствующим языком, используемым для разработки коммерческих программных продуктов./3/

Эффективная автоматизация машин, приборов и технологического оборудования, которая является обязательным условием ускорения научно-технического прогресса, основана на широком применении средств микропроцессорной техники (МТ). Автоматизация с использованием микропроцессоров (МП) и микроконтроллеров (МК) быстро распространяется в самых разнообразных отраслях народного хозяйства, охватывает новые сферы./1/

Программные средства изучения и проектирования микроконтроллерных систем на основе МК 8051 (электронный симулятор EdSim51) предназначены для освоения программирования микроконтроллеров семейства МК 8051 и для программирования и отладки совместной работы микроконтроллера с основными компонентами микропроцессорных систем управления (СУ)./2/

1 Постановка задачи

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

Формулировка задания:

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

Ход работы:

Дана функция $X \& \neg ((\neg Y \& X) \rightarrow (X \vee Y))$

Построена её таблица истинности (Таблица 1).

Таблица 1 — Таблица истинности функции F1

№	X	Y	$\neg Y$	$\neg Y \& X$	$X \vee Y$	$(\neg Y \& X) \rightarrow (X \vee Y)$	$\neg ((\neg Y \& X) \rightarrow (X \vee Y))$	F1
1	0	0	1	0	0	1	0	0
2	0	1	0	0	1	1	0	0
3	1	0	1	1	1	1	0	0
4	1	1	0	0	1	1	0	0

Дана функция $F2 = \neg (X \vee Y \& \neg Z)$

Построена её таблица истинности (Таблица 2).

Таблица 2 — Таблица истинности функции F2

№	X	Y	Z	$\neg Z$	$Y \& \neg Z$	$X \vee Y \& \neg Z$	F2
1	0	0	0	1	0	0	1
2	0	0	1	0	0	0	1
3	0	1	0	1	1	1	0
4	0	1	1	0	0	0	1

Продолжение таблицы 2

№	X	Y	Z	$\neg Z$	$Y \& \neg Z$	$X \vee Y \& \neg Z$	F2
5	1	0	0	1	0	1	0
6	1	0	1	0	0	1	0
7	1	1	0	1	1	1	0
8	1	1	1	0	0	1	0

Программы l1_z1 и l1_z2 разработаны в среде Visual Studio 2017 и предназначены для вывода на экран таблиц истинности сложных комбинированных функций.

Код программы:

Код программы l1_z1 для функции F1:

```
#include "pch.h"
#include <iostream>
#include <stdio.h>
#include <conio.h>
#include <locale>
#include <Windows.h>
int main() {
    bool x = 0, y = 0, f1;
    printf("Laba1:\n");
    printf("Zadanie1:\n");
    printf("Truth table:\n");
    printf("x y !y (!y && x) (x || y) ((!y && x) <= (x || y)) (!(!y && x) <= (x || y))\n");
    f1\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            f1 = x && (!(!y && x) <= (x || y));
            printf("%i %i %i %i %i %i %i\n", x, y, !y, (!y && x), (x || y), ((!y && x) <= (x || y)), (!(!y && x) <= (x || y)), f1);
            y = !y;
        }
        x = !x;
    }
    system("PAUSE");
    return 0;
}
```

Код программы l1_z2 для функции F2:

```
#include "pch.h"
#include <iostream>
#include <stdio.h>
#include <conio.h>
#include <locale>
#include <Windows.h>
int main() {
    bool x = 0, y = 0, z = 0, f2;
    printf("Lab1:\n");
    printf("Zadanie2:\n");
    printf("Truth table:\n");
    printf("x y z !z (y && !z) (x || y && !z) f2\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            for (int k = 0; k < 2; k++) {
                f2 = !(x || y && !z);
                printf("%i %i %i %i %i %i %i\n", x, y,
z, !z, (y && !z), (x || y && !z), f2);
                z = !z;
            }
            y = !y;
        }
        x = !x;
    }
    system("PAUSE");
    return 0;
}
```

После запуска программ на выполнение появляется форма (Рисунок 1 и Рисунок 2), а на ней результаты работы программ.

Результаты:

```

Laba1:
Zadanie1:
Truth table:
x y !y (!y && x) (x || y) ((!y && x) <= (x || y)) !((!y && x) <= (x || y)) f1
0 0 1 0 0 1 0 0
0 1 0 0 1 1 0 0
1 0 1 1 1 1 0 0
1 1 0 0 1 1 0 0
Для продолжения нажмите любую клавишу . . .

```

Рисунок 1 — Таблица истинности функции F1

```

Laba1:
Zadanie2:
Truth table:
x y z !z (y && !z) (x || y && !z) f2
0 0 0 1 0 0 1
0 0 1 0 0 0 1
0 1 0 1 1 1 0
0 1 1 0 0 0 1
1 0 0 1 0 1 0
1 0 1 0 0 1 0
1 1 0 1 1 1 0
1 1 1 0 0 1 0
Для продолжения нажмите любую клавишу . . .

```

Рисунок 2 — Таблица истинности функции F2

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, равное единице (таблица 3). Построить таблицы истинности для всех функций. Смоделировать полученную комбинационную схему на ПЭВМ, минимизировать её.

Таблица 3 — Исходные данные

Номер варианта	Номера конститuent
9	0, 5, 7, 8, 9, 12, 13, 15

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	$F(x_1, x_2, x_3, x_4)$
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
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	1
10	1	0	1	0	0
11	1	0	1	1	0

Продолжение таблицы 4

№	x1	x2	x3	x4	F(x1,x2,x3,x4)
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

Составлена СДНФ.

Правило составления СДНФ логической функции:

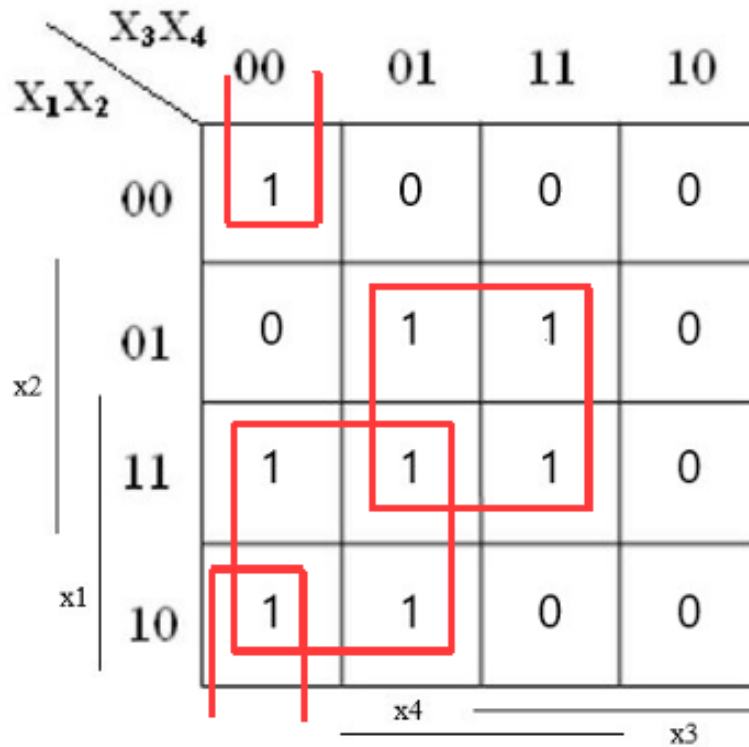
1. Составляются конъюнкции для тех наборов, на которых функция принимает значение 1.
2. Если аргумент в наборе равен 1, он записывается без отрицания, а если равен 0, то с отрицанием.
3. Все конъюнкции соединяются знаком дизъюнкции.

Согласно правилам составления СДНФ построена логическая функция

$$F1(x1,x2,x3,x4)=(\neg x1 \& \neg x2 \& \neg x3 \& \neg x4) \vee (\neg x1 \& x2 \& \neg x3 \& x4) \vee (\neg x1 \& x2 \& x3 \& x4) \vee (x1 \& \neg x2 \& \neg x3 \& \neg x4) \vee (x1 \& \neg x2 \& \neg x3 \& x4) \vee (x1 \& x2 \& \neg x3 \& \neg x4) \vee (x1 \& x2 \& \neg x3 \& x4) \vee (\neg x1 \& \neg x2 \& x3 \& \neg x4)$$

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

Диаграмма Вейча (карта Карно) для СДНФ.



Получена минимизированная ДНФ функция

$$F2(x1, x2, x3, x4) = (\neg x2 \& \neg x3 \& \neg x4) \vee (x1 \& \neg x3) \vee (x2 \& x4)$$

Для проверки полученной минимизированной функции $F2(x1, x2, x3, x4)$ построена её таблица истинности (таблица 5).

Таблица 5 — Таблица истинности минимизированной функции $F2(x1, x2, x3, x4)$

№	x1	x2	x3	x4	F(x1,x2,x3,x4)	F2(x1,x2,x3,x4)
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	0	0
4	0	1	0	0	0	0
5	0	1	0	1	1	1
6	0	1	1	0	0	0
7	0	1	1	1	1	1

Продолжение таблицы 5

№	x1	x2	x3	x4	F(x1,x2,x3,x4)	F2(x1,x2,x3,x4)
8	1	0	0	0	1	1
9	1	0	0	1	1	1
10	1	0	1	0	0	0
11	1	0	1	1	0	0
12	1	1	0	0	1	1
13	1	1	0	1	1	1
14	1	1	1	0	0	0
15	1	1	1	1	1	1

Из анализа таблицы 5 видно, что значения функций $F(x_1, x_2, x_3, x_4)$ и $F_2(x_1, x_2, x_3, x_4)$ совпадают на всех наборах значений аргументов x_1, x_2, x_3, x_4 . Следовательно минимизация логической функции $F_1(x_1, x_2, x_3, x_4)$ выполнена верно.

Программа 12 разработаны в среде Visual Studio 2017 и предназначена для вывода на экран таблиц истинности исходной комбинированной функции и минимизированной функции, а также для проверки правильности минимизации.

Код программы 12:

```
#include "pch.h"
#include <iostream>
#include <stdio.h>
#include <Windows.h>
using namespace std;

int main() {

    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
```

```

bool x1 = 0, x2 = 0, x3 = 0, x4 = 0, f, f2;
printf("CДНФ:\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 l = 0; l < 2; l++) {
                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;
}
x1 = 0;
x2 = 0;
x3 = 0;
x4 = 0;
printf("\nMДНФ:\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 l = 0; l < 2; l++) {
                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);
                f2 = (!x2 && !x3 && !x4) || (x1 && !x3) || (x2
&& x4);

                if (f ^ f2) {
                    printf("Error!\n");
                    break;
                }
                printf("%i %i %i %i %i\n", x1, x2, x3, x4, f2);
                x4 = !x4;
            }
            x3 = !x3;
        }
    }
}

```

```

    }
    x2 = !x2;
}
x1 = !x1;
}

system("PAUSE");
return 0;
}

```

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

Результаты:

```

СДНФ:
x1 x2 x3 x4 f
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
0 0 1 1 0
0 1 0 0 0
0 1 0 1 1
0 1 1 0 0
0 1 1 1 1
1 0 0 0 1
1 0 0 1 1
1 0 1 0 0
1 0 1 1 0
1 1 0 0 1
1 1 0 1 1
1 1 1 0 0
1 1 1 1 1

МДНФ:
x1 x2 x3 x4 f2
0 0 0 0 1
0 0 0 1 0
0 0 1 0 0
0 0 1 1 0
0 1 0 0 0
0 1 0 1 1
0 1 1 0 0
0 1 1 1 1
1 0 0 0 1
1 0 0 1 1
1 0 1 0 0
1 0 1 1 0
1 1 0 0 1
1 1 0 1 1
1 1 1 0 0
1 1 1 1 1

Для продолжения нажмите любую клавишу . . .

```

Рисунок 3 — Результаты работы программы

1.3 Моделирование управляющего алгоритма на языке C

Формулировка задания:

Реализовать сложную неструктурированную граф-схему алгоритма (ГСА) (см. Приложение А рисунок А.1) на языке C/C++ с использованием оператора goto.

Ход работы:

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

Программа 13 разработаны в среде Visual Studio 2017 и предназначена для вывода на экран поэтапных результатов выполнения отдельных блоков схемы.

Код программы 13:

```
#include "pch.h"
#include <iostream>
#include "stdio.h"
#include "io manip"
#include "windows.h"
#include "conio.h"
using namespace std;
int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    bool x1, x2, x3, x4, x5, x6, x7;
    while ("Конец\n"){
        cout << "Начало" << endl;
        cout << "A1" << endl;
11:    cout << "Введите x1" << endl;
        cin >> x1;
        if (x1) {
```

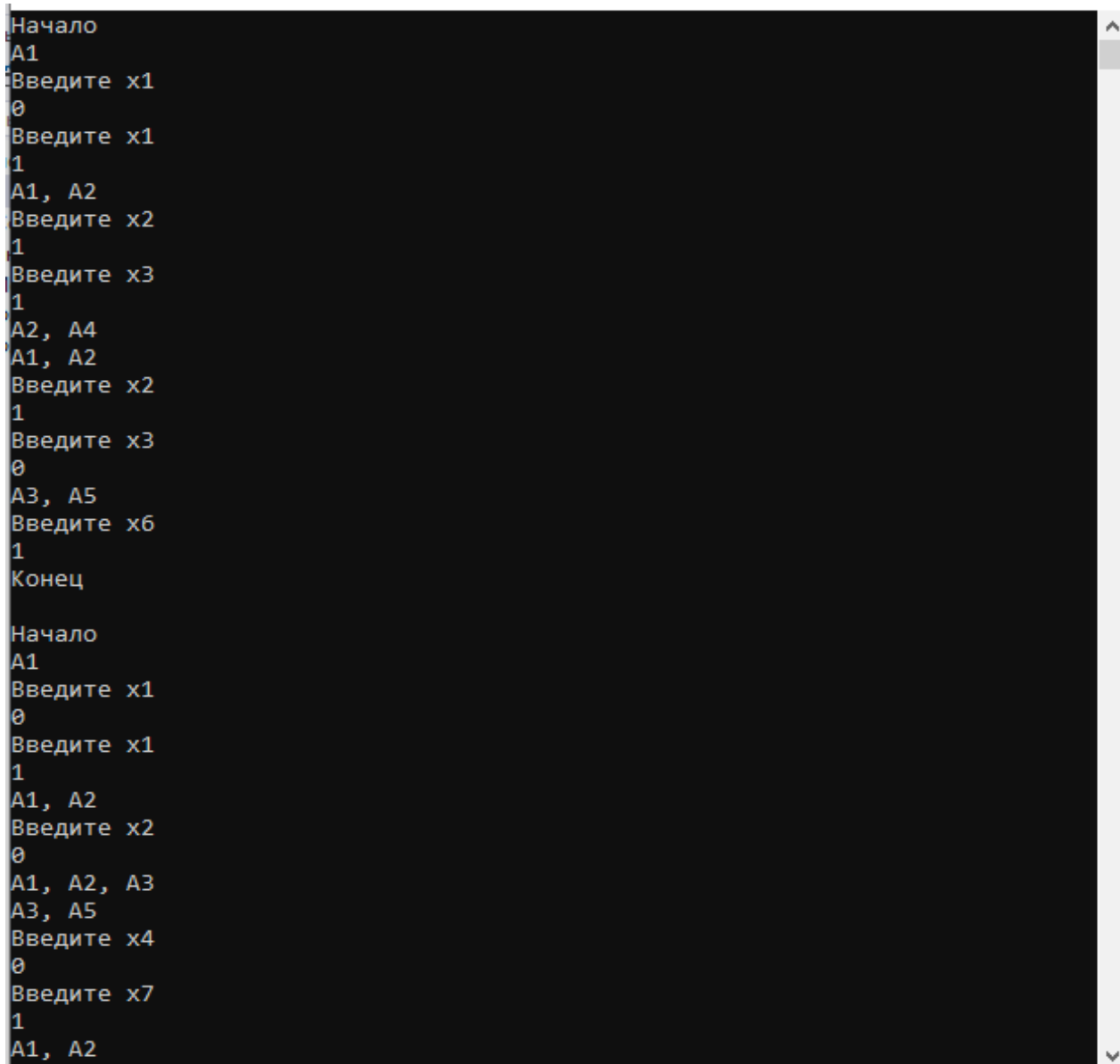
```

12:      cout << "A1, A2" << endl;
13:      cout << "Введите x2" << endl;
        cin >> x2;
        if (x2) {
            cout << "Введите x3" << endl;
            cin >> x3;
            if (x3) {
14:                cout << "A2, A4" << endl;
                    goto l2;}
            else {
                goto l6;}}
        else {
            cout << "A1, A2, A3" << endl;
            cout << "A3, A5" << endl;
            cout << "Введите x4" << endl;
            cin >> x4;
            if (x4) {
                cout << "A1, A3" << endl;
                cout << "Введите x5" << endl;
                cin >> x5;
                if (x5) {
16:                    cout << "A3, A5" << endl;
                        cout << "Введите x6" << endl;
                        cin >> x6;
                        if (x6) {
                            goto l7;
                            _getch();
                            return 0;}
                        else {
                            goto l3;}
                        }
                    else {
                        goto l4;}}
                else {
                    cout << "Введите x7" << endl;
                    cin >> x7;
                    if (x7) {
                        goto l2;}
                    else {
                        goto l4;}}
                }}
            else {
                goto l1;}
17:      cout << "Конец\n" << endl; }}

```

После запуска программ на выполнение появляется консольное окно (Рисунок 4), а в нем результаты работы программ.

Результаты:



```
Начало
A1
Введите x1
0
Введите x1
1
A1, A2
Введите x2
1
Введите x3
1
A2, A4
A1, A2
Введите x2
1
Введите x3
0
A3, A5
Введите x6
1
Конец

Начало
A1
Введите x1
0
Введите x1
1
A1, A2
Введите x2
0
A1, A2, A3
A3, A5
Введите x4
0
Введите x7
1
A1, A2
```

Рисунок 4 — Результаты работы программы

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 (таблица 6) на микроконтроллере MCS-51.

Таблица 6 — Исходные данные

Номер варианта	Логическая функция	Уровни сигналов управления	
		разрешение	подтверждение
9	$\neg (X \vee Y \& \neg Z)$	L	L

Ход работы:

Для логической функции была построена таблица истинности (таблица 7).

Таблица 7 — Таблица истинности функции $F(X, Y, Z)$

№	X	Y	Z	$\neg Z$	$Y \& \neg Z$	$X \vee Y \& \neg Z$	$F(X, Y, Z)$
0	0	0	0	1	0	0	1
1	0	0	1	0	0	0	1
2	0	1	0	1	1	1	0
3	0	1	1	0	0	0	1
4	1	0	0	1	0	1	0
5	1	0	1	0	0	1	0
6	1	1	0	1	1	1	0
7	1	1	1	0	0	1	0

Программа 14 разработаны на микроконтроллере MCS-51 и предназначена для реализации логической функции.

Для выполнения побитовых операций используются выходы P1.0-P1.7 микроконтроллера и битовый аккумулятор C.

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

Код программы l4:

JMP BEGIN;	
BEGIN:	
CLR C;	C=0
MOV P1, #00001001B;	P1=00001001
MOV C, P1.1;	C=Y
ANL C, /P1.0;	C=Y & !Z
MOV 00, C;	запись промежуточного результата
MOV C, P1.2;	C=X
ORL C, 00;	C=X + Y & !Z
CPL C;	C=!(X + Y & !Z)
MOV P1.7, C;	
SETB P1.4;	конец вывода
MOV P1.5, C;	выдача результата
JMP BEGIN;	

1. Запустить программу EdSim51;
2. Нажать RST;
3. Записать программу;
4. Нажимая STEP, пошагово выполнять программу следя за портом P1.

Результаты работы программ приведены на рисунках 5-7.

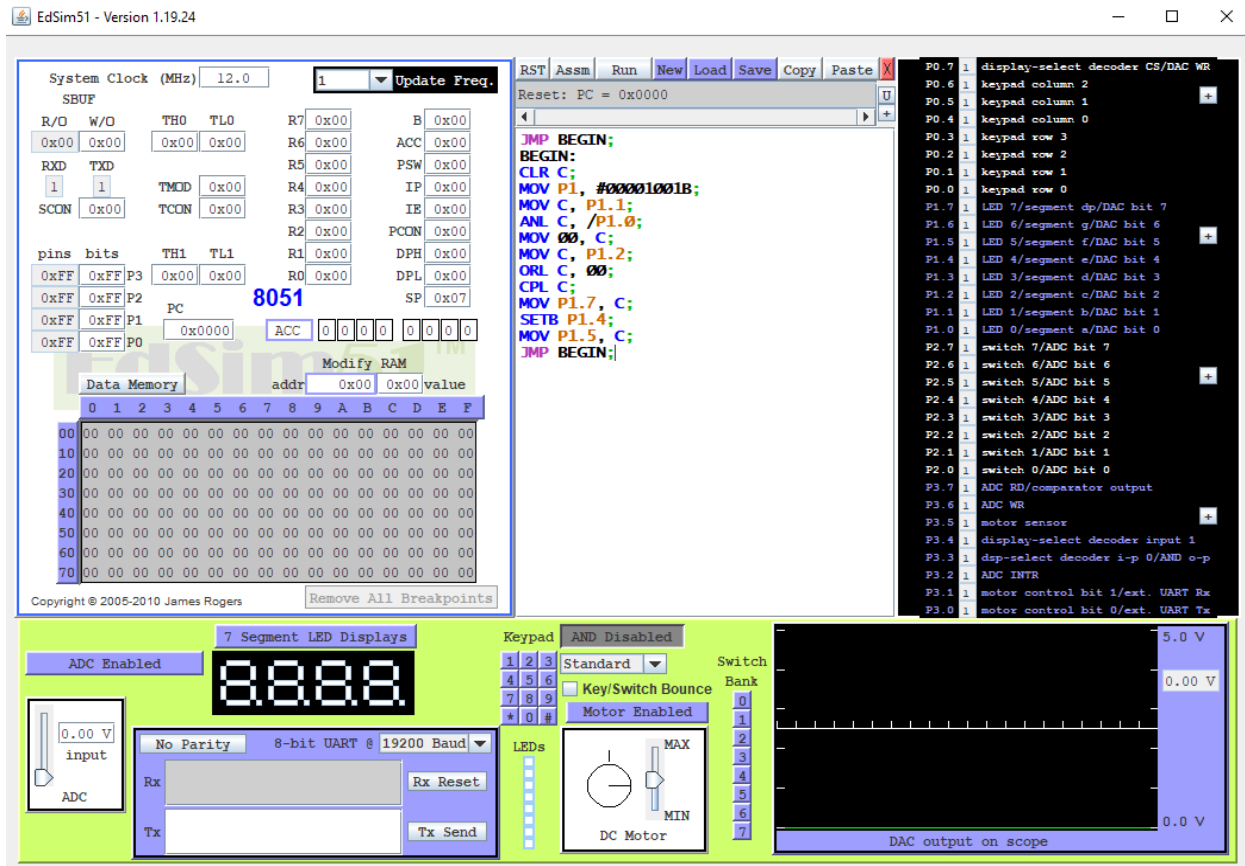


Рисунок 5 — Результаты работы программы

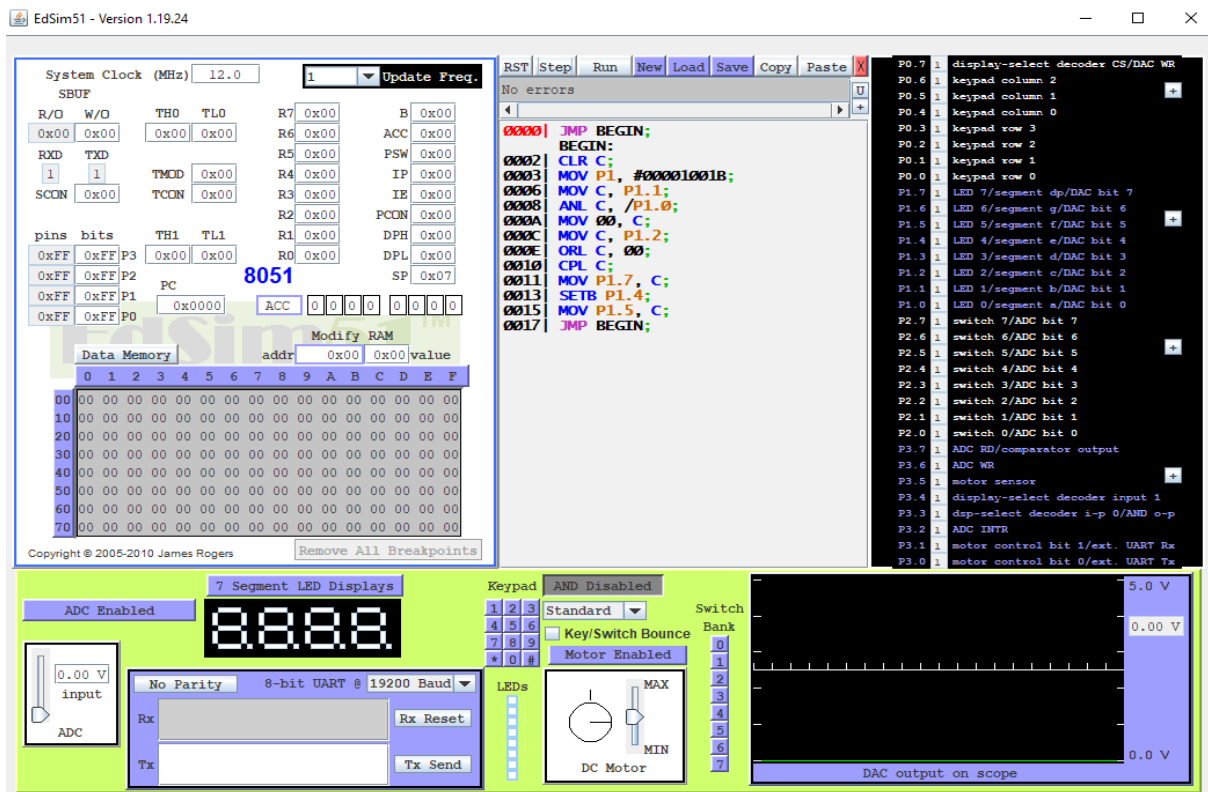


Рисунок 6 — Результаты работы программы

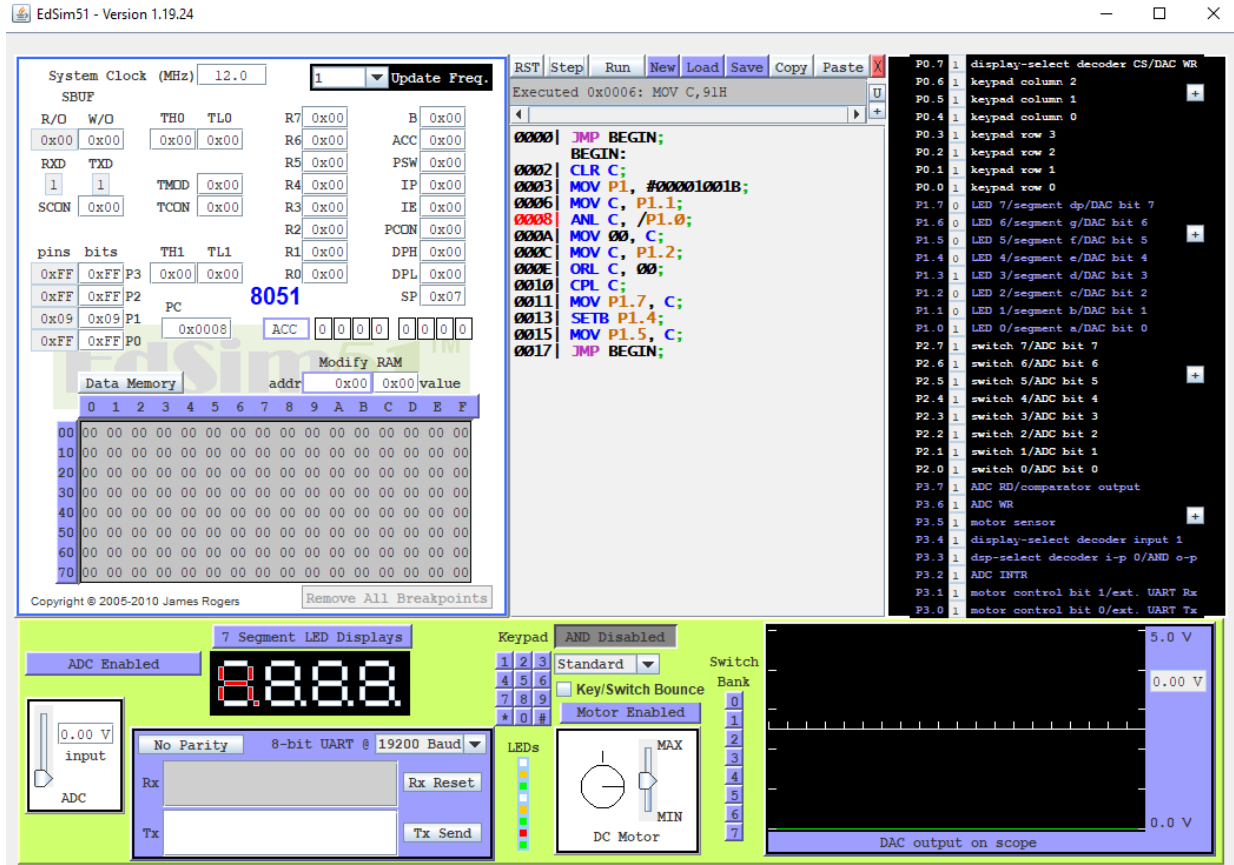


Рисунок 7 — Результаты работы программы

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 задания:

Составить СДНФ функции по заданным значениям аргументов (таблица 8), на которых логическая функция принимает значение, равное единице. Построить таблицу истинности функции. Реализовать логическую функцию на микроконтроллере MCS-51.

Таблица 8 — Исходные данные

Номер варианты	Номера конститuent
9	0, 5, 7, 8, 9, 12, 13, 15

Ход работы:

Для логической функции, заданной списком единичных наборов была построена таблица истинности (таблица 9).

Таблица 9 — Таблица истинности функции $F(x_1, x_2, x_3, x_4)$

№	x_1	x_2	x_3	x_4	$F(x_1, x_2, x_3, x_4)$
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
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	1
10	1	0	1	0	0

Продолжение таблицы 9

№	x1	x2	x3	x4	F(x1,x2,x3,x4)
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	1

Составлена СДНФ.

Правило составления СДНФ логической функции:

1. Составляются конъюнкции для тех наборов, на которых функция принимает значение 1.
2. Если аргумент в наборе равен 1, он записывается без отрицания, а если равен 0, то с отрицанием.
3. Все конъюнкции соединяются знаком дизъюнкции.

Согласно правилам составления СДНФ построена логическая функция

$$F1(x1,x2,x3,x4)=(\neg x1 \& \neg x2 \& \neg x3 \& \neg x4) \vee (\neg x1 \& x2 \& \neg x3 \& x4) \vee (\neg x1 \& x2 \& x3 \& x4) \vee \\ \vee (x1 \& \neg x2 \& \neg x3 \& \neg x4) \vee (x1 \& \neg x2 \& \neg x3 \& x4) \vee (x1 \& x2 \& \neg x3 \& \neg x4) \vee (x1 \& x2 \& \neg x3 \& x4) \vee \\ \vee (\neg x1 \& \neg x2 \& x3 \& \neg x4)$$

Программа 15 разработаны на микроконтроллере MCS-51 и предназначена для реализации сложной логической функции, заданной в форме СДНФ.

Для выполнения побитовых операций используются выходы P1.0-P1.7 микроконтроллера и битовый аккумулятор С.

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

Код программы l5:

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;	переходна KONEC, если C=1
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;	переходна KONEC, если C=1
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;	переходна KONEC, если C=1
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;	переходна KONEC, если C=1
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;	переходна KONEC, если C=1
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;	переходна KONEC, если C=1
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;	переходна KONEC, если C=1
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;	P1.0=результат (P1.0=f)
JMP BEGIN;	

1. Запустить программу EdSim51;
2. Нажать RST;
3. Записать программу;
4. Нажимая STEP, пошагово выполнять программу следя за портом P1.

Результаты работы программ приведены на рисунках 8-10.

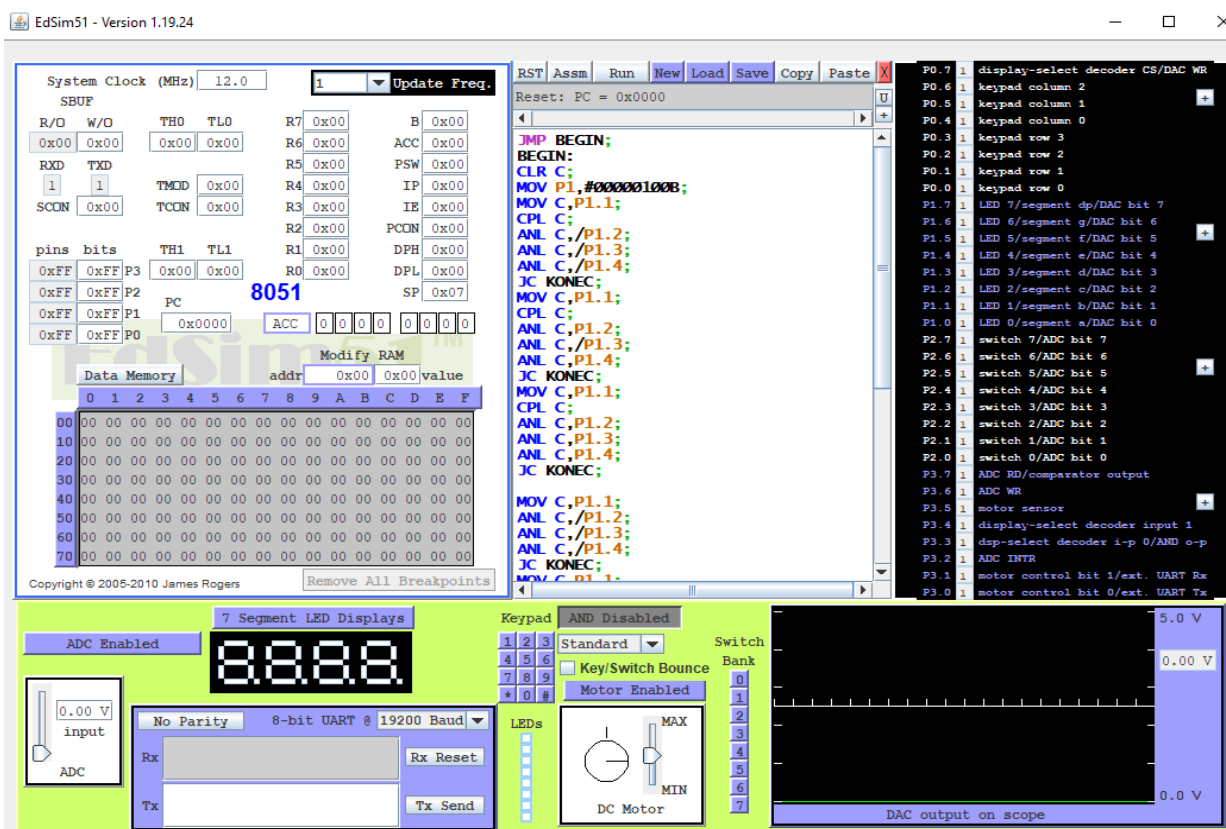


Рисунок 8 — Результаты работы программы

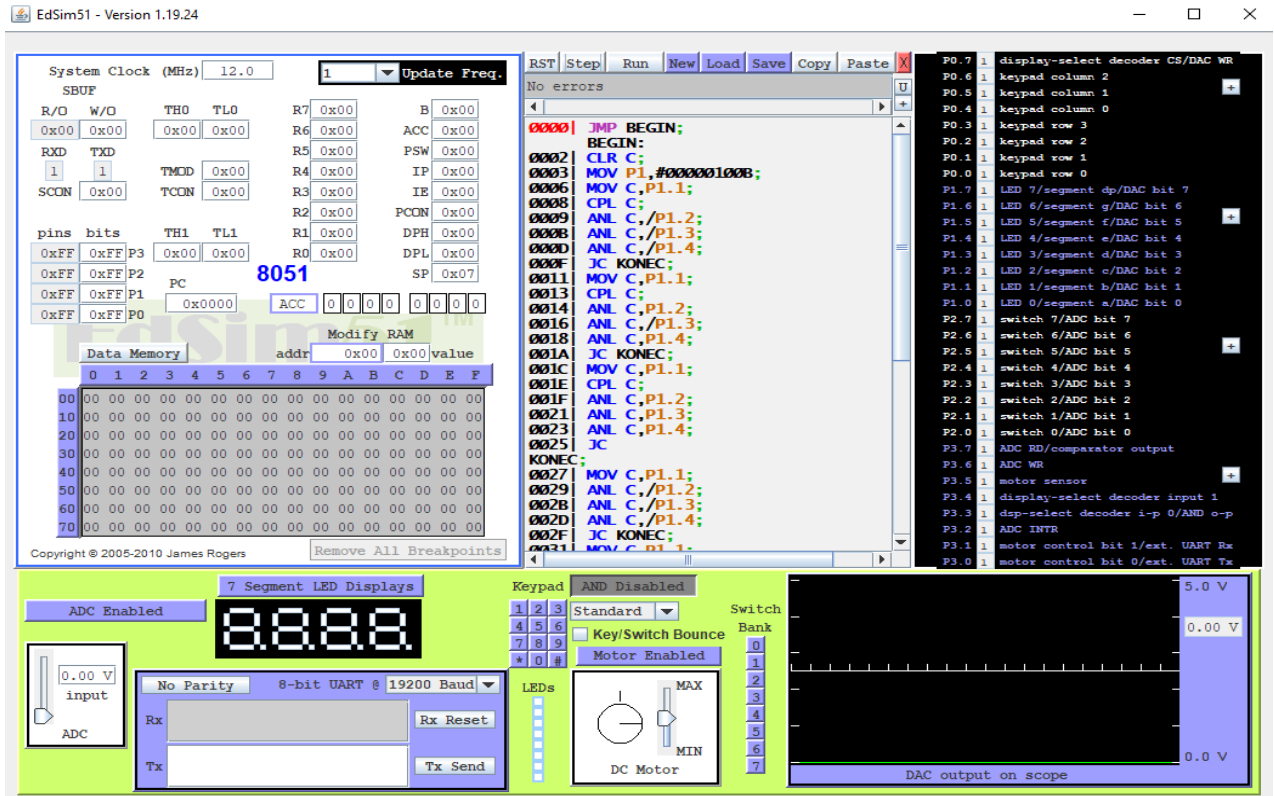


Рисунок 9 — Результаты работы программы

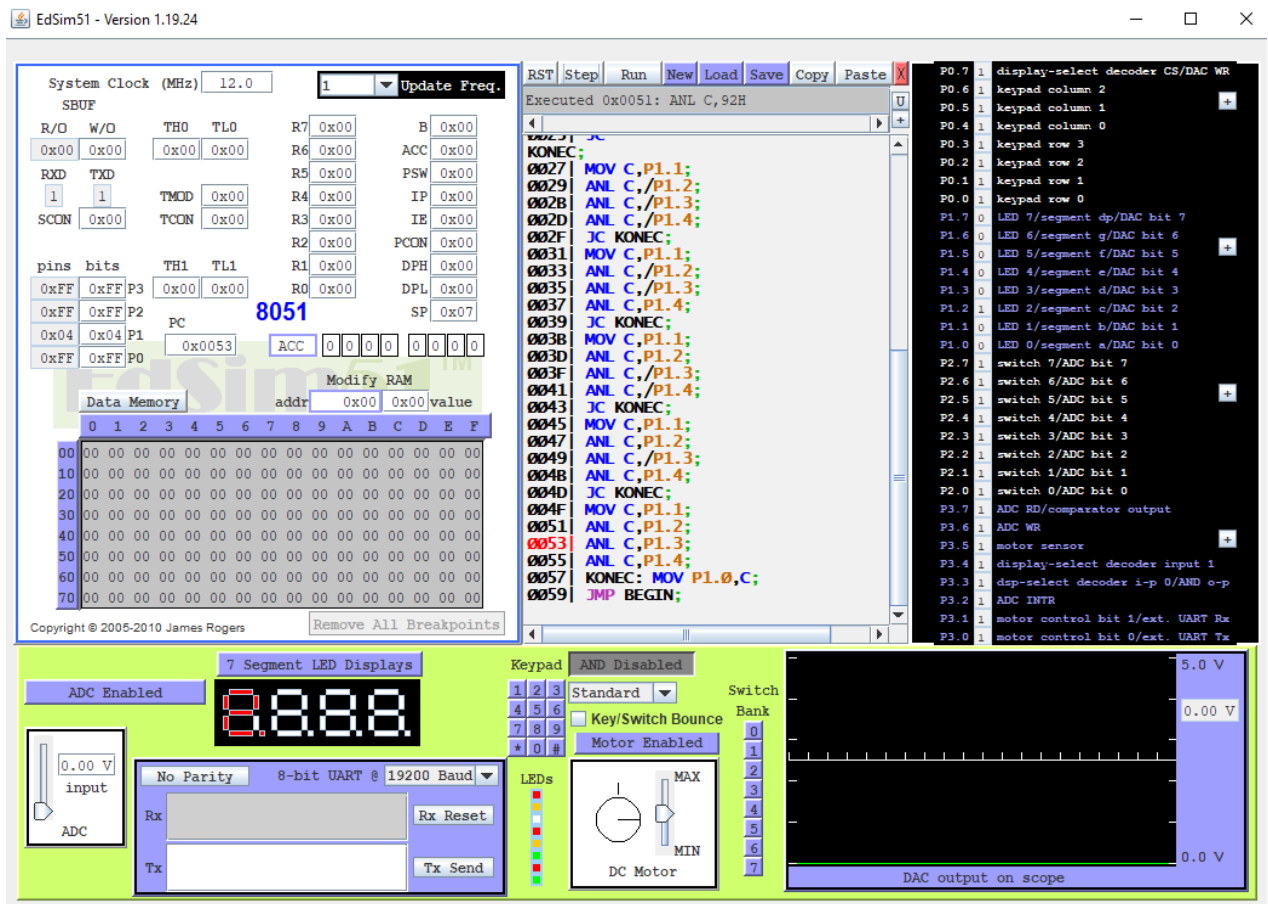


Рисунок 10 — Результаты работы программы

1.6 Моделирование управляющего алгоритма на языке ассемблера для микроконтроллера MCS51

Формулировка задания:

Реализовать сложную неструктурированную граф-схему алгоритма (ГСА) (см. Приложение А рисунок А.1) на микроконтроллере MCS-51.

Ход работы:

Алгоритм решения задачи в виде схемы программы представлен в Приложение А рисунок А.2.

Программа 16 разработаны на микроконтроллере MCS-51 и предназначена для реализации сложной неструктурированной граф-схемы алгоритма (ГСА) на микроконтроллере MCS-51.

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

Код программы 16:

Код программы 16 представлен в таблицы 10:

Таблица 10 — Код программы 16

	Псевдокод	Ассемблер MCS51	Комментарий
BEGIN:	AN	JMP BEGIN; BEGIN: CLR C; MOV P0, #11111110B; MOV P2, #00000000B;	Ввод значений условий Ввод X1-7 Начало AN
	A1	MOV P2, #00000001B;	Исп. A1

Продолжение таблицы 10

	Псевдокод	Ассемблер MCS51	Комментарий
AX1:	AX1	AX1: MOV C, P0.1;	Ввод X1
	IF X1=0 GOTO AX1	JNC AX1;	
A1, A2:	A1, A2	A1, A2: MOV P2, #00000001B; MOV P2, #00000010B;	Исп. A1 Исп. A2
AX2:	AX2	AX2: MOV C, P0.2;	Ввод X2
	IF X2=0 GOTO A1, A2, A3	JNC A1, A2, A3;	
	AX3	MOV C, P0.3;	Ввод X3
	IF X3=0 GOTO A3, A5	JNC A3, A5;	
A2, A4:	A2, A4	A2, A4: MOV P2, #00000010B; MOV P2, #00000100B;	Исп. A2 Исп. A4
	GOTO A1, A2	JMP A1, A2;	
A1, A2, A3:	A1, A2, A3	A1, A2, A3: MOV P2, #00000001B; MOV P2, #00000010B; MOV P2, #00000011B;	Исп. A1 Исп. A2 Исп. A3
	A3, A5	MOV P2, #00000011B; MOV P2, #00000101B;	Исп. A3 Исп. A5
	AX4	MOV C, P0.4;	Ввод X4
	IF X4=0 GOTO AX7	JNC AX7;	
	A1, A3	MOV P2, #00000001B; MOV P2, #00000011B;	Исп. A1 Исп. A3
	AX5	MOV C, P0.5;	Ввод X5
	IF X5=0 GOTO A2, A4	JNC A2, A4;	
A3, A5:	A3, A5	A3, A5: MOV P2, #00000011B; MOV P2, #00000101B;	Исп. A3 Исп. A5
	AX6	MOV C, P0.6;	Ввод X6
	IF X6=0 GOTO AX2	JNC AX2;	
	GOTO AK	JMP AK;	
AX7:	AX7	AX7: MOV C, P0.7;	Ввод X7
	IF X7=0 GOTO A2, A4	JNC A2, A4;	
	GOTO A1, A2	JMP A1, A2;	
AK:	AK	AK: MOV P2, #00000110B; JMP BEGIN;	Конец AK

1. Запустить программу EdSim51;
2. Нажать RST;
3. Записать программу;
4. Нажимая STEP, пошагово выполнять программу следя за портом P1.

Результаты работы программ приведены на рисунках 11-14.

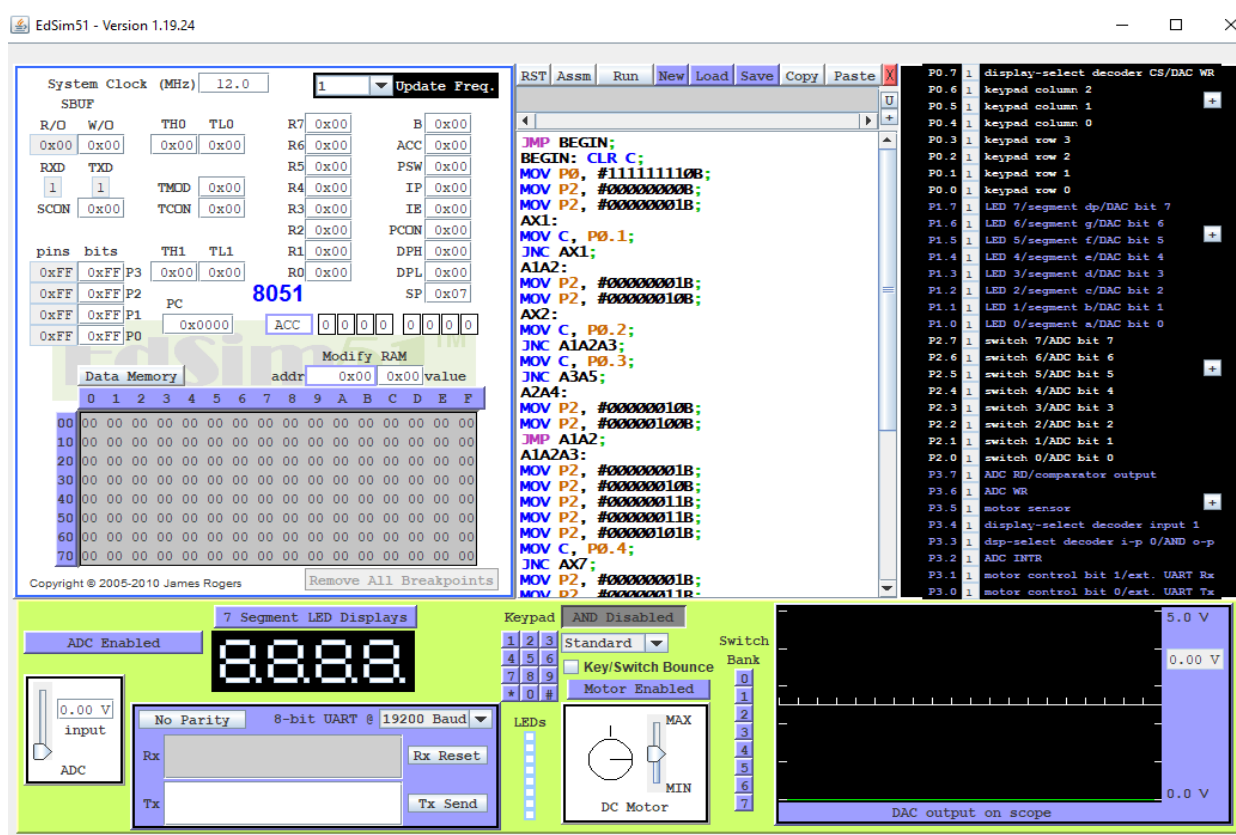


Рисунок 11 — Результаты работы программы

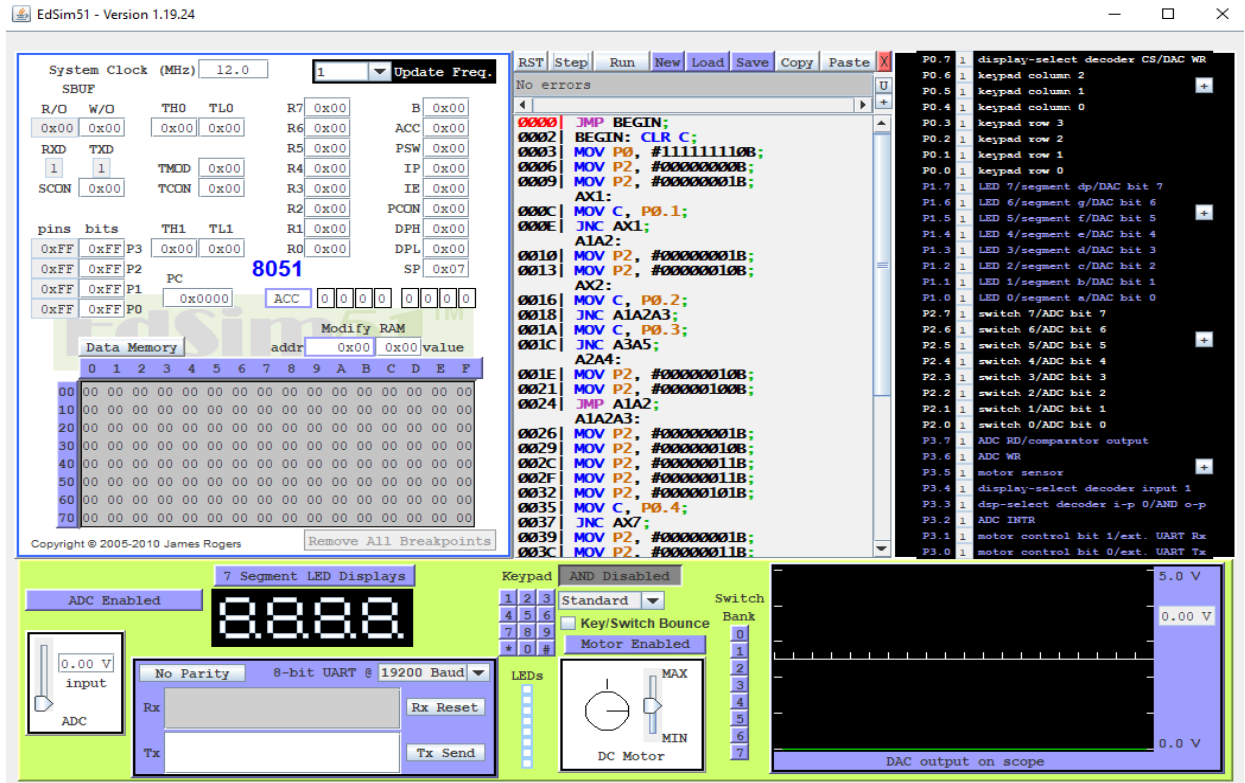


Рисунок 12 — Результаты работы программы

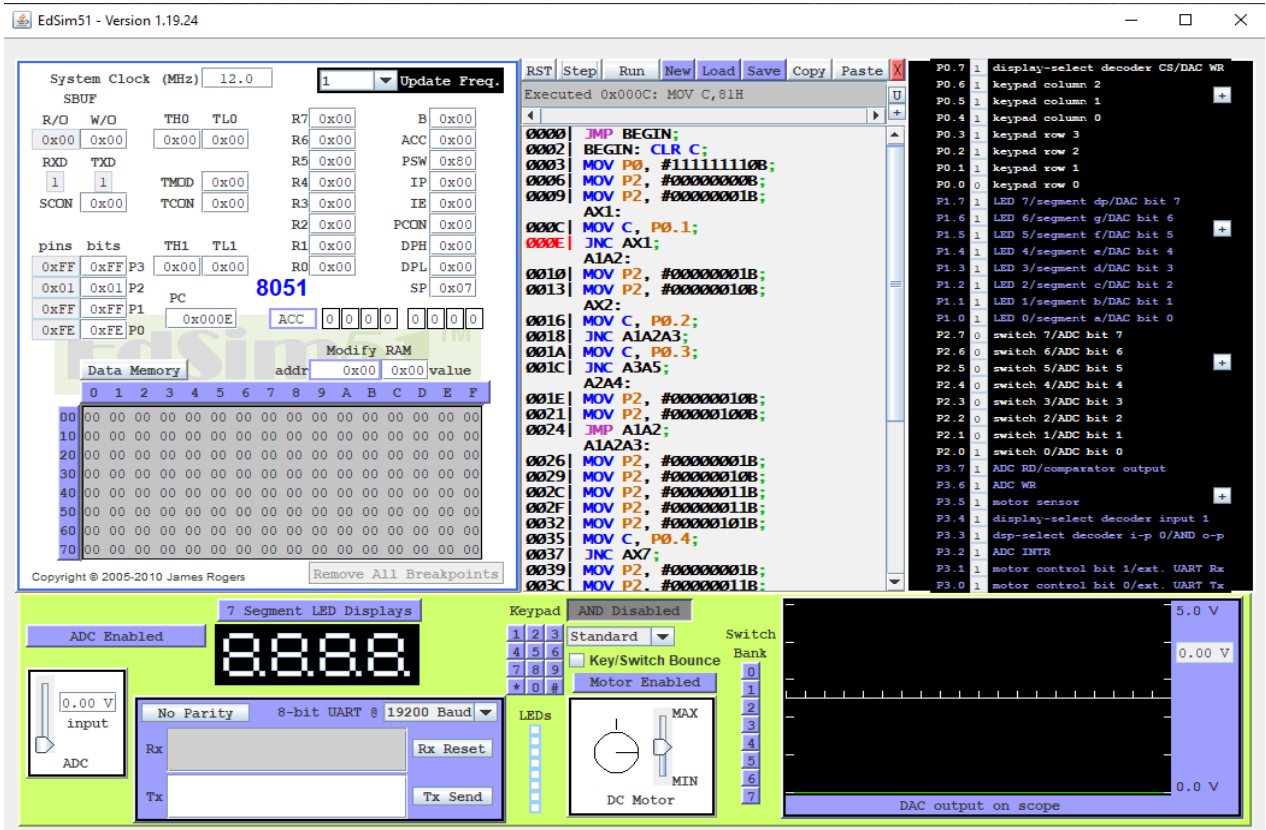


Рисунок 13 — Результаты работы программы

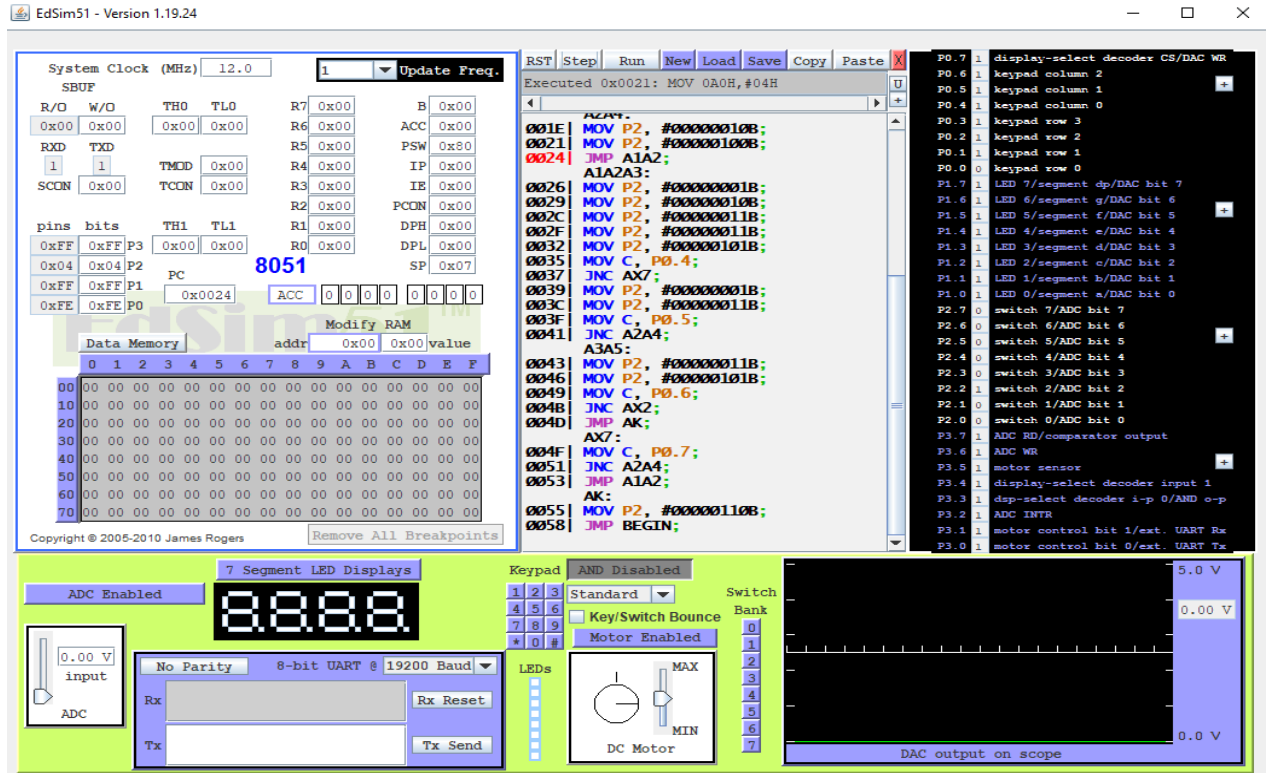


Рисунок 14 — Результаты работы программы

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)./2/

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 (рисунок 15):

- 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. /1/

Кроме того, стенд включает источник постоянного напряжения и осциллограф.

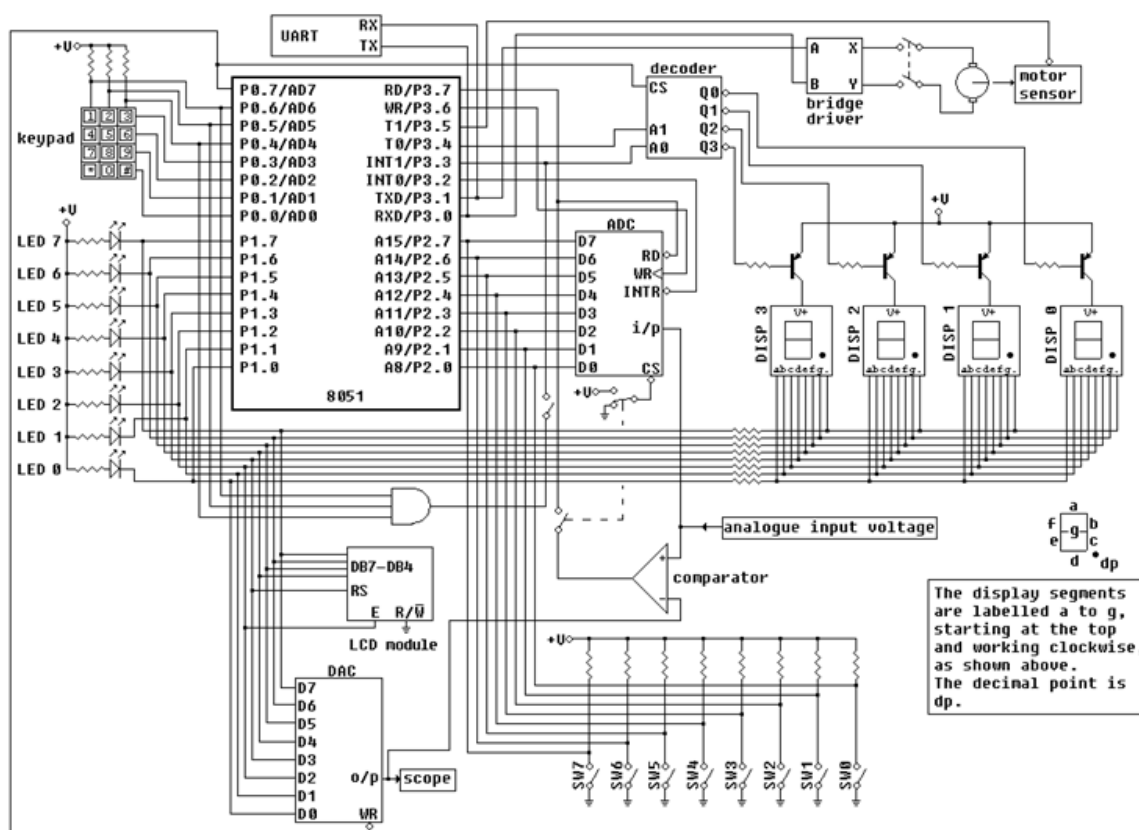


Рисунок 15 - Функциональная схема модели лабораторного стенда EdSim51

На рисунке 16 приведен внешний вид EdSim51.

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

На рисунке 16 можно выделить четыре основных панели:

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

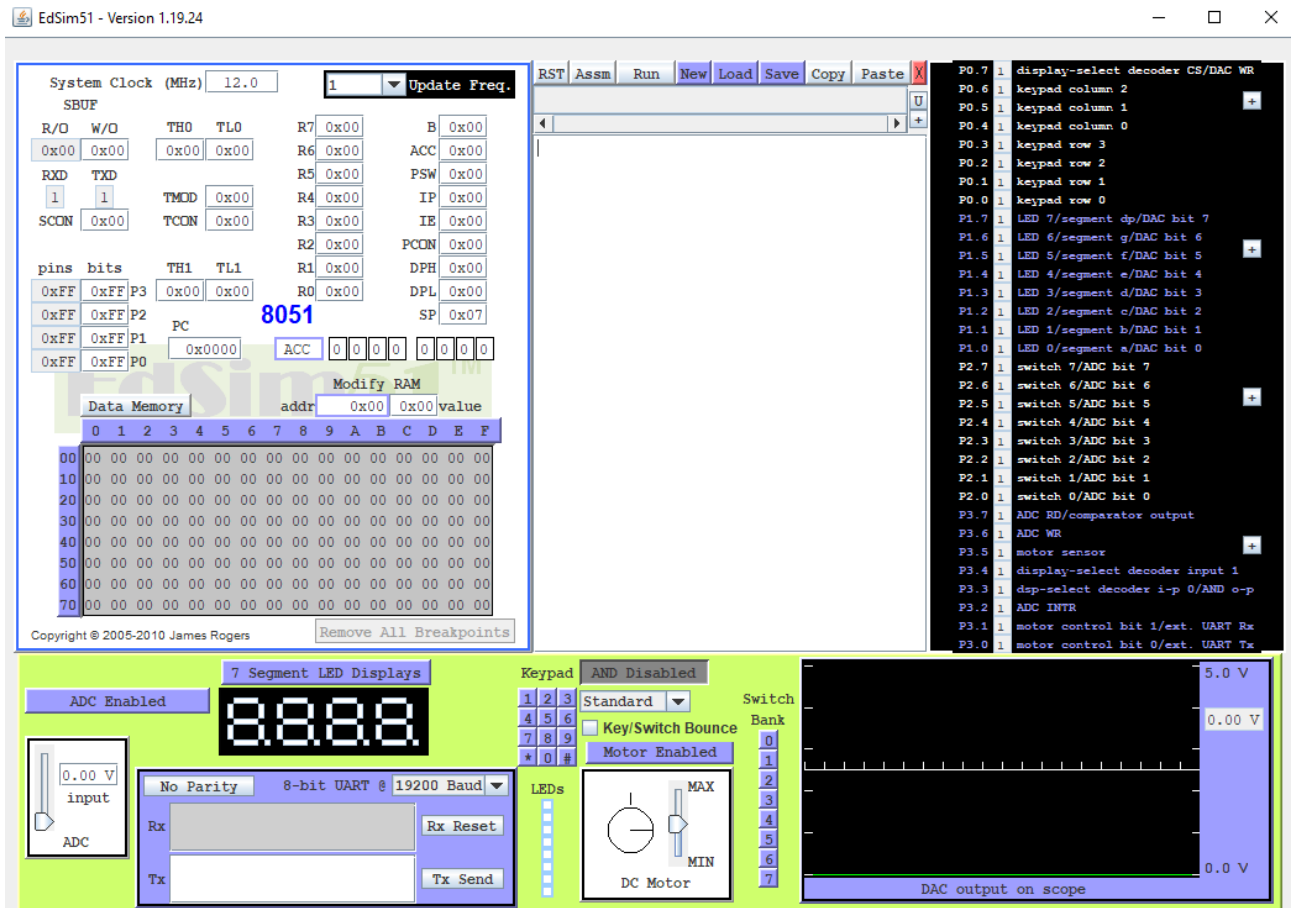


Рисунок 16 - Внешний вид модели лабораторного стенда EdSim51.

3 Руководство оператора

Ввод текста программы происходит в панель ассемблерного кода (рисунок 17).

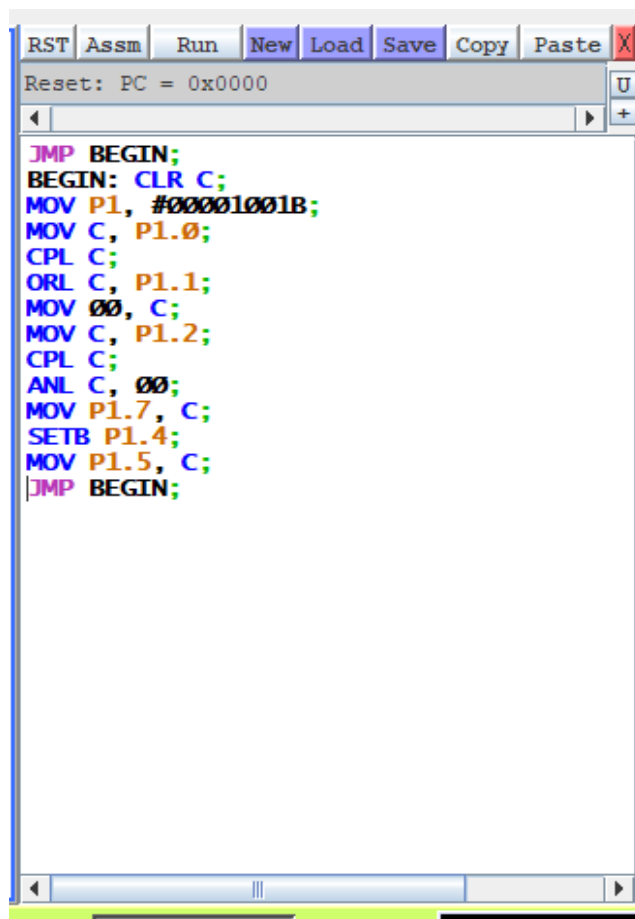
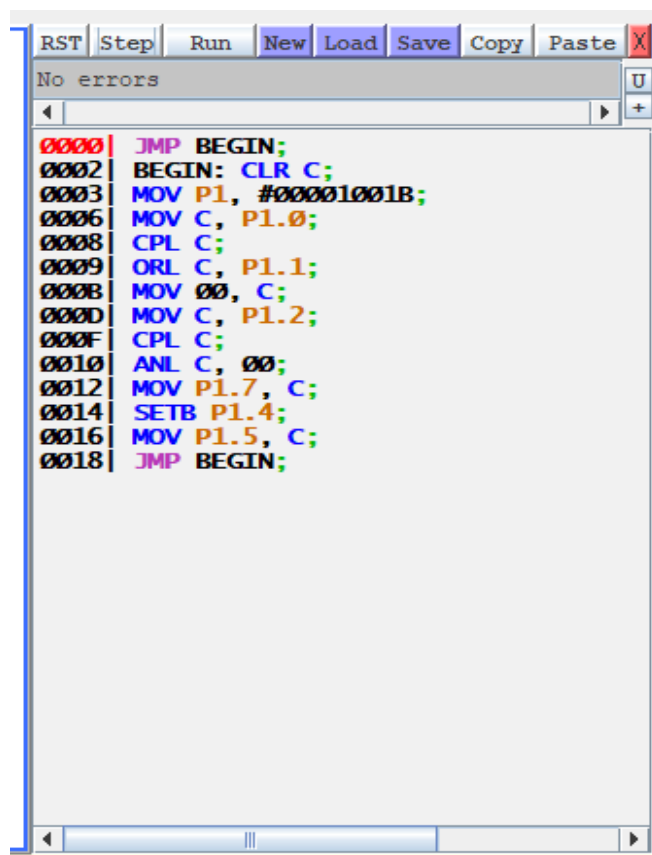


Рисунок 17 – Панель ассемблерного кода

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 18).



```
RST Step Run New Load Save Copy Paste X
No errors
U
+
0000 JMP BEGIN;
0002 BEGIN: CLR C;
0003 MOV P1, #00001001B;
0006 MOV C, P1.0;
0008 CPL C;
0009 ORL C, P1.1;
000B MOV 00, C;
000D MOV C, P1.2;
000F CPL C;
0010 ANL C, 00;
0012 MOV P1.7, C;
0014 SETB P1.4;
0016 MOV P1.5, C;
0018 JMP BEGIN;
```

Рисунок 18 – Панель ассемблерного кода

При нажатии на одну из кнопок, отмеченных знаком “+” (рисунок 19), можно увидеть соединения портов в отдельном окне (рисунок 20).

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/segment dp/DAC bit 7
P1.6	1	LED 6/segment g/DAC bit 6
P1.5	1	LED 5/segment f/DAC bit 5
P1.4	1	LED 4/segment e/DAC bit 4
P1.3	1	LED 3/segment d/DAC bit 3
P1.2	1	LED 2/segment c/DAC bit 2
P1.1	1	LED 1/segment b/DAC bit 1
P1.0	1	LED 0/segment a/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	display-select decoder input 1
P3.3	1	dsp-select decoder i-p 0/AND o-p
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 19 – Список регистров

Port 2			—	☐	×
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 20 - Коммутация выводов P2.0-P2.7 и их состояния второго порта микроконтроллера

Заключение

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

Были проведены испытания программы, которые позволили проверить соответствие работы программы всем требованиям технического задания.

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.

За время работы над курсовым проектом был повышен уровень практических навыков программирования на языке C++ и на ассемблере MCS51.

Список использованных источников

1. Edsim51. Официальный сайт. - Режим доступа: <https://www.edsim51.com/introduction.html>
2. Пособие по микроконтроллерам 8051. - Режим доступа: <https://www.circuitstoday.com/8051-simulator>
3. Стивен Прата. Язык программирования C++ (C++11). Лекции и упражнения, 6-е издание — М.:Вильямс, 2012. - 1248 с.
4. Александров, Э.Э. Программирование на языке С в Microsoft Visual Studio 2010: учеб. Пособие / Э.Э. Александров, В.В. Афонин. - Саранск: Изд-во Мордов. Ун-та, 2010. - 428 с.
5. Компьютерное программирование. - Режим доступа: https://wreferat.baza-referat.ru/Компьютерное_программирование
6. Программирование и достижения компьютерной техники. - Режим доступа: <https://www.bestreferat.ru/referat-141666.html>

АЛГОРИТМЫ

Приложение А

(обязательное)

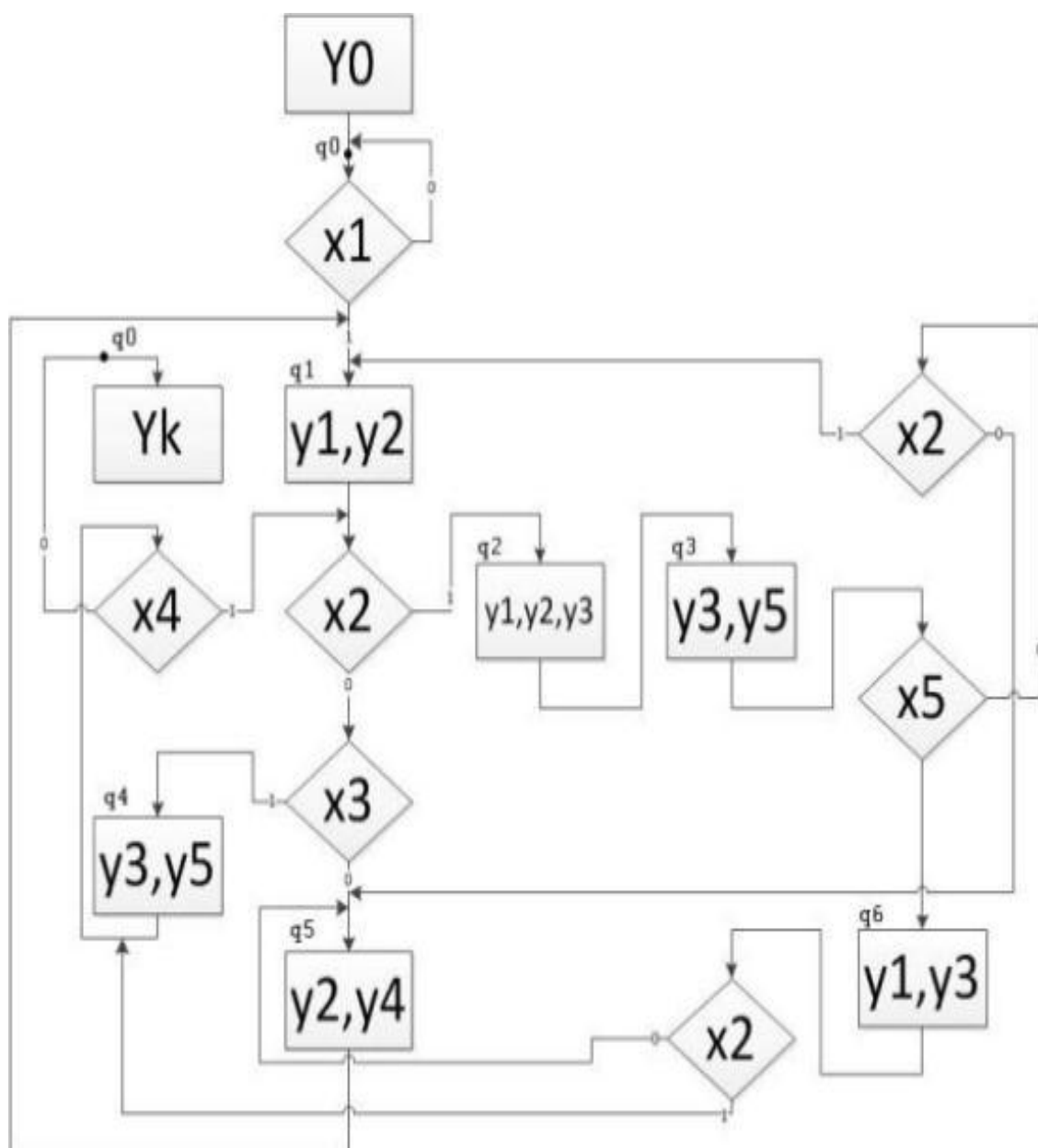


Рисунок А.1 – ГСА

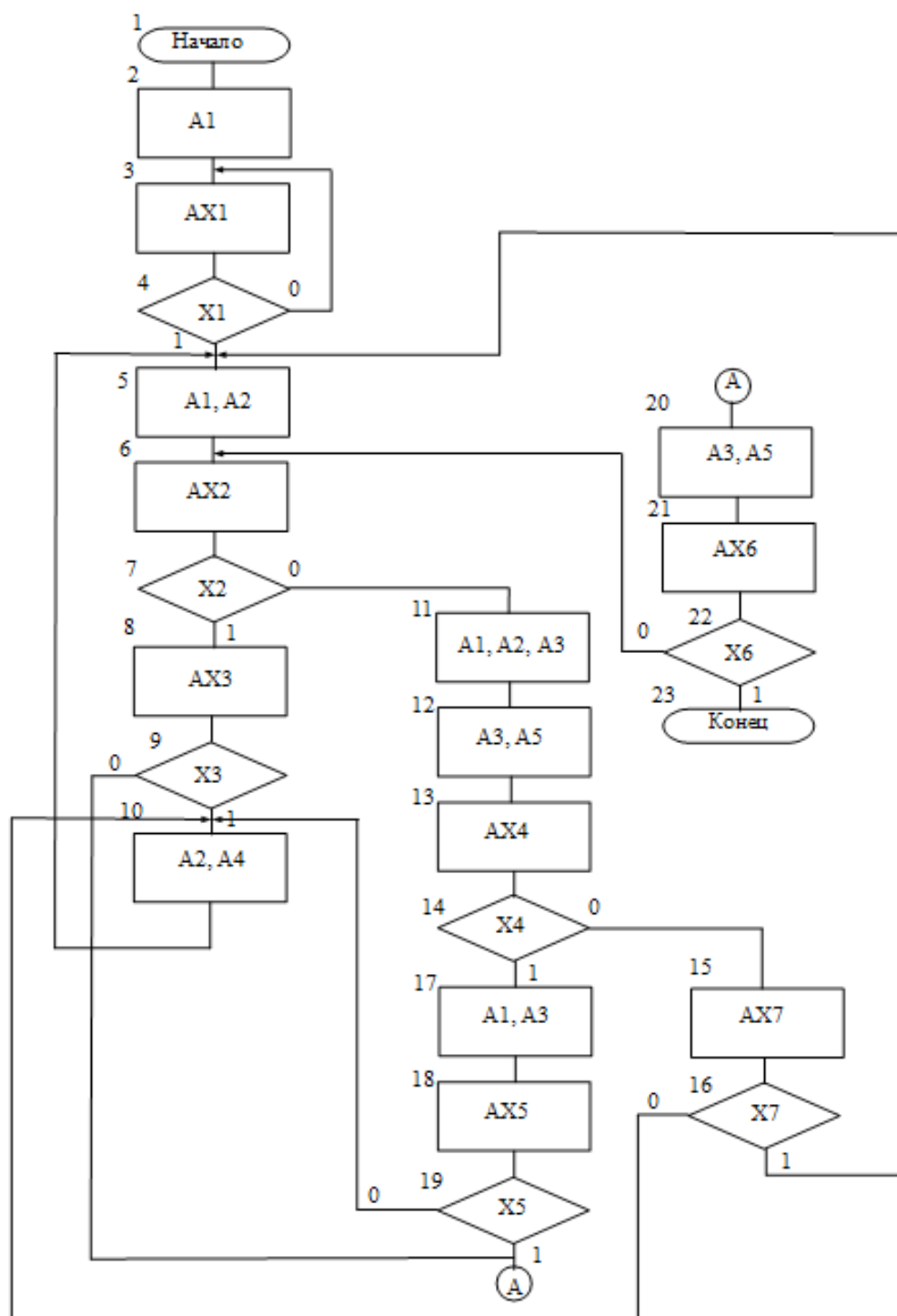


Рисунок А.2 – Схема программы

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=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 $A \neq \text{byte}$
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