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 1.090501.11

Выполнил: студентка гр.18ВО2

Новинская А.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 Новинской А.М. Группа 18ВО2

Тема проекта: Моделирование микропроцессорного управляющего устройства. Вариант №11

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

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_____/ Новинская А.М./

Реферат

Пояснительная записка состоит из 47 листов, 26 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.

Содержание

Введение..... 6

1 Постановка и реализация задачи 7-33

1.1 Моделирование комбинационных схем на языке С (реализация функций алгебры логики на С) 7-9

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..... 16-19

1.4 Моделирование комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики на языке ассемблера для микроконтроллера MCS51)..... 20-23

1.5 Моделирование сложной комбинационной схемы на языке ассемблера для микроконтроллера MCS51 (реализация функции алгебры логики, заданной СДНФ на языке ассемблера для микроконтроллера MCS51) 24-28

1.6 Моделирование управляющего алгоритма на языке ассемблера для микроконтроллера MCS51..... 29-33

2 Описание моделирующей среды..... 34-36

3 Руководство оператора..... 37-39

Заключение..... 40

Список использованных источников..... 41

Приложение А. Алгоритмы..... 42-44

Приложение Б. Система команд микроконтроллера MCS51..... 45-47

Введение

Потребность считать возникала у людей вместе с появлением цивилизации. Им было необходимо осуществлять торговые сделки, проводить землемерные работы, управлять запасами урожая, следить за астрономическими циклами. Для этого издревле были изобретены различные инструменты, от счётных палочек и абака, в ходе развития науки и техники эволюционировавшие в калькуляторы и разнообразные вычислительные устройства, в том числе и персональные компьютеры. /1/

Структурное программирование — это технология создания программ, позволяющая путем соблюдения определенных правил уменьшить время разработки и количество ошибок, а также облегчить возможность модификации программы. Структурный подход охватывает все стадии разработки проекта: спецификацию, проектирование, собственно программирование и тестирование. Особенно важно придерживаться определенной дисциплины при программировании на C++. /2/

C++ является языком программирования общего назначения. Естественная для него область применения - системное программирование, понимаемое в широком смысле этого слова. Кроме того, C++ успешно используется во многих областях приложения, далеко выходящих за указанные рамки. Реализации C++ теперь есть на всех машинах, начиная с самых скромных микрокомпьютеров - до самых больших супер-ЭВМ, и практически для всех операционных систем. /4/

Микроконтроллеры семейства MCS-51 представляют собой функционально завершенные 8-разрядные однокристальные микроЭВМ гарвардской архитектуры, содержащие все необходимые узлы для работы в автономном режиме. На основе таких микросхем могут быть реализованы алгоритмы управления различных устройств. Микроконтроллеры данного семейства выпускаются в корпусах PLCC, DIP и QFP. /3/

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 F1= A∨ (B & ¬A)→ (A & ¬B)

Построена её таблица истинности (Таблица 1).

Таблица 1 — Таблица истинности функции F1

№	A	B	¬A	¬B	B &¬A (1)	A∨ 1	A&¬B	F1
1	0	0	1	1	0	0	0	1
2	0	1	1	0	1	1	0	0
3	1	0	0	1	0	1	1	1
4	1	1	0	0	0	1	0	0

Дана функция F2= ¬X&Yv¬Z

Построена её таблица истинности (Таблица 2).

Таблица 2 — Таблица истинности функции F2

№	X	Y	Z	¬X	¬Z	¬X &Y	F2
1	0	0	0	1	1	0	1
2	0	0	1	1	0	0	0
3	0	1	0	1	1	1	1
4	0	1	1	1	0	1	1
5	1	0	0	0	1	0	1
6	1	0	1	0	0	0	0
7	1	1	0	0	1	0	1
8	1	1	1	0	0	1	1

Программы lab1.1 и lab1.2 разработаны в среде Visual Studio 2017 и предназначены для вывода на экран таблиц истинности сложных комбинированных функций.

Код программы:

Код программы lab1.1 для функции F1:

```
#include "pch.h"
#include <iostream>
#include <stdio.h>
#include <conio.h>
#include <clocale>
#include <Windows.h>
int main() {
	bool a = 0, b = 0, f1;
	printf("Laba1:\n");
	printf("Zadanie2:\n");
	printf("Таблица истинности:\n");
	printf("a b  !a  !b  (b && !a)    (a || (b && !a))    (a && !b)     f1\n");
	for (int i = 0; i < 2; i++) {
		for (int j = 0; j < 2; j++) {
			f1 = a || (b && !a) > (a && !b);
			printf("%i %i   %i   %i     %i              %i              %i         %i \n", a,
```

```
b, !a, !b, (b && !a), (a || (b && !a)), (a && !b), f1);
                b = !b;
            }
            a = !a;
        }
        system("PAUSE");
        return 0;
    }
```

Код программы lab1.2 для функции F2:

```
#include <iostream>
#include <stdio.h>
#include <conio.h>
#include <clocale>
#include <Windows.h>
int main() {
    bool x = 0, y = 0, z = 0, f2;
    printf("Laba1:\n");
    printf("Zadanie2:\n");
    printf("Truth table:\n");
    printf("x y z  !x  !z  (!x && y)    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, !x, !z, (!x
&& y), f2);

                z = !z;
            }
            y = !y;
        }
        x = !x;
    }
    system("PAUSE");
    return 0;
}
```

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

Результаты:

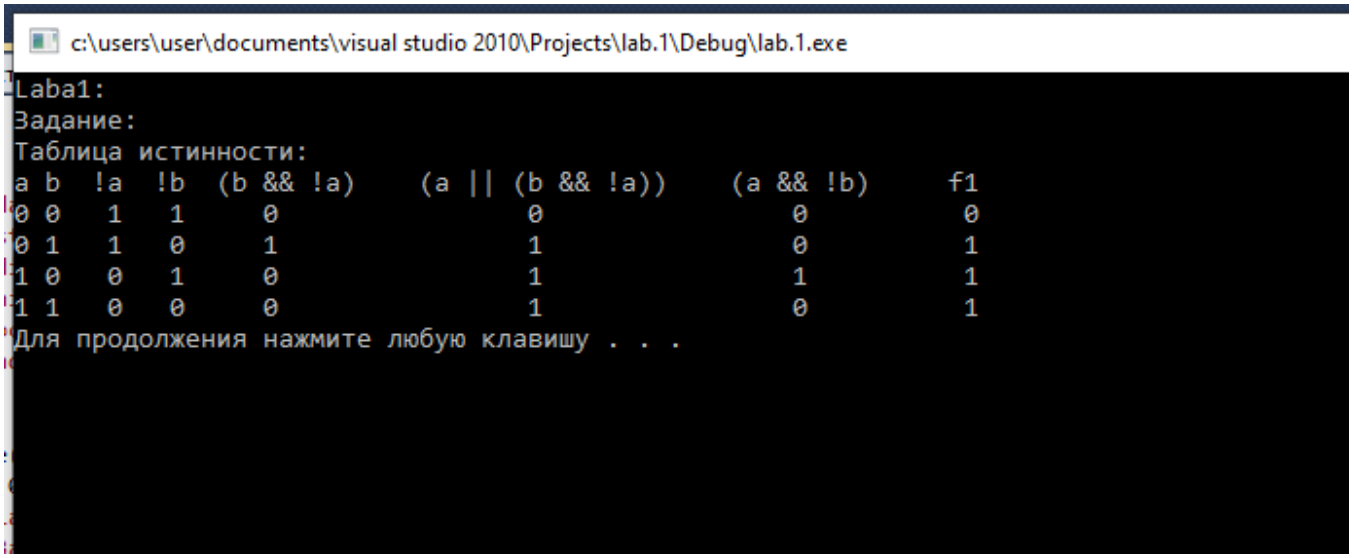


Рисунок 1 — Таблица истинности функции F1

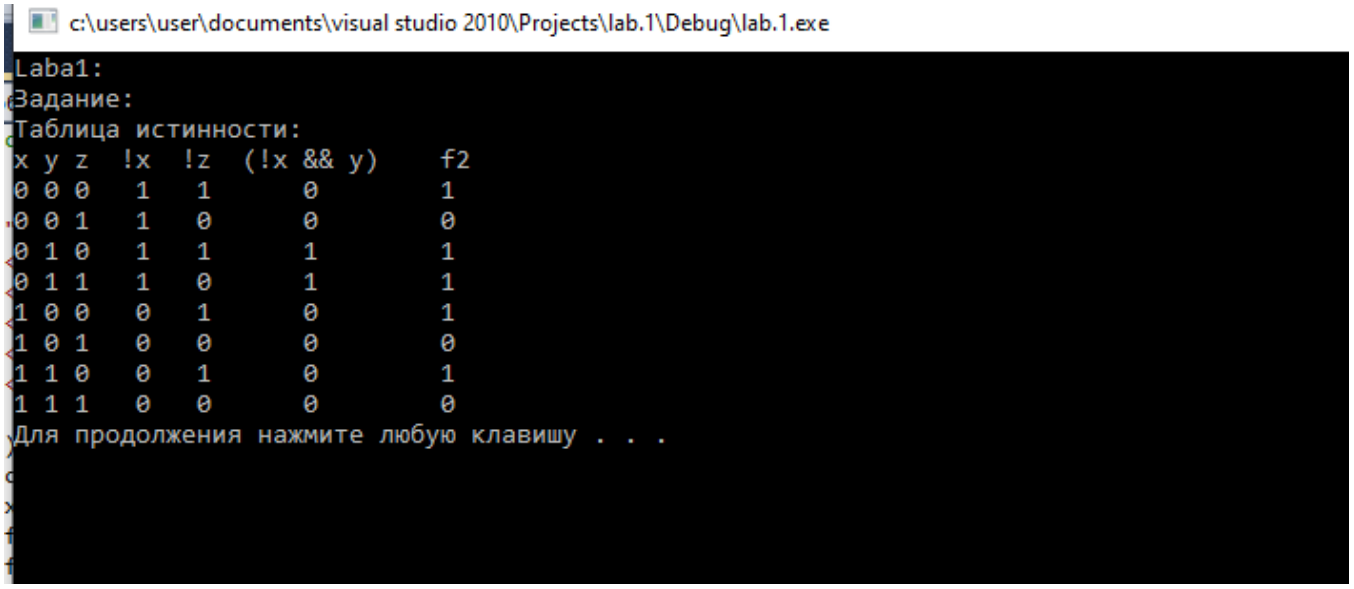


Рисунок 2 — Таблица истинности функции F2

1.2 Моделирование сложной комбинационной схемы на языке C

(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 конституент
11	0,1,4,6,7,8,9,15

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(x1,x2,x3,x4)
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
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
12	1	1	0	0	0
13	1	1	0	1	0

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.

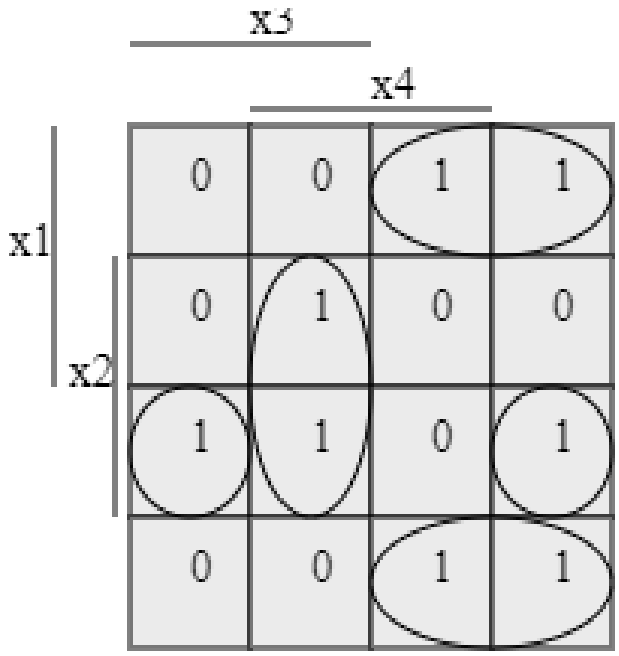
2. Если аргумент в наборе равен 1, он записывается без отрицания, а если равен 0, то с отрицанием.

3. Все конъюнкции соединяются знаком дизъюнкции.

Согласно правилам составления СДНФ построена логическая функция F1(x1,x2,x3,x4)=(¬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)

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)=(¬x2&¬x3)∨ (¬x1&x2&¬x4)∨ (x2&x3&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	1	1
2	0	0	1	0	0	0
3	0	0	1	1	0	0
4	0	1	0	0	1	1
5	0	1	0	1	0	0
6	0	1	1	0	1	1
7	0	1	1	1	1	1
8	1	0	0	0	1	1
9	1	0	0	1	0	0

10	1	0	1	0	0	0
11	1	0	1	1	0	0
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	1	1

Из анализа таблицы 5 видно, что значения функций F(x1,x2,x3,x4) и F2(x1,x2,x3,x4) совпадают на всех наборах значений аргументов x1,x2,x3,x4. Следовательно минимизация логической функции F1(x1,x2,x3,x4) выполнена верно.

Программа lab2 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 lab2:

```
#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("СДНФ:\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("\nМДНФ:\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) || (!x1 && x2 && !x4) || (x2 && x3 && 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:

Рисунок 3 — Результаты работы программы

1.3 Моделирование управляющего алгоритма на языке С

Формулировка задания:

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

Ход работы:

Алгоритм решения задачи в виде схемы программы представлен в приложении А на рис. А.1.

Программа lab3 разработаны в среде Visual Studio 2017 и предназначена для вывода на экран поэтапных результатов выполнения отдельных блоков схемы.

Код программы lab3:

```
#include "pch.h"
#include <iostream>
#include "stdio.h"
#include "iomanip"
#include "windows.h"
#include "conio.h"
using namespace std;
int main()
{
SetConsoleCP(1251);
SetConsoleOutputCP(1251);
bool x1, x2, x3, x4, x5;
```



```
while ("Конец\n")
{
cout << "Начало" << endl;
cout << "Введите x1" << endl;
cin >> x1;
```

```
if (x1) {
cout << "Введите x2" << endl;
cin >> x2;
if (x2) {
cout << "A1" << endl;
cout << "Введите x3" << endl;
cin >> x3;
if (x3) {
l1: cout << "A2" << endl;
goto l4;
}
else {
l2: cout << "Введите x5" << endl;
cin >> x5;
if (x5) {
goto l3;
}
else {
goto l1;
}
}
}
else {
cout << "A3" << endl;
goto l2;
}
}
```



```
else {
cout << "A4" << endl;
cout << "Введите x4" << endl;
cin >> x4;
if (x4){
```

```
l3: cout << "A5" << endl;
goto l4;
_getch();
return 0;
}
else {
goto l2;
}
}
l4: cout << "Конец\n" << endl;
}
}
```

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

Результаты:

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 (табл. 6) на микроконтроллере MCS-51.

Таблица 6 — Исходные данные

Номер варианта	Логическая функция	Уровни сигналов управления	
		разрешение	подтверждение
11	¬ X & Y ∨ ¬ Z	H	L

Ход работы:

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

Таблица 7 — Таблица истинности функции F(x1,x2,x3,x4)

№	X	Y	Z	¬ X	¬ Z	¬ X & Y	F(X, Y, Z)
0	0	0	0	1	1	0	1
1	0	0	1	1	0	0	0
2	0	1	0	1	1	1	1
3	0	1	1	1	0	1	1
4	1	0	0	0	1	0	1
5	1	0	1	0	0	0	0
6	1	1	0	0	1	0	1
7	1	1	1	0	0	0	0

Программа lab4 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 (рис. Б.1-Б.5).

Код программы lab4:

```
JMP BEGIN;
```

```
BEGIN:
CLR C;                      C=0
MOV P1, #00001001B;         P1=00001001
MOV C, P1.2;                C=X
CPL C;                      C=!X
ANL C, P1.1;                C=!X & Y
ORL C, /P1.0;               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.

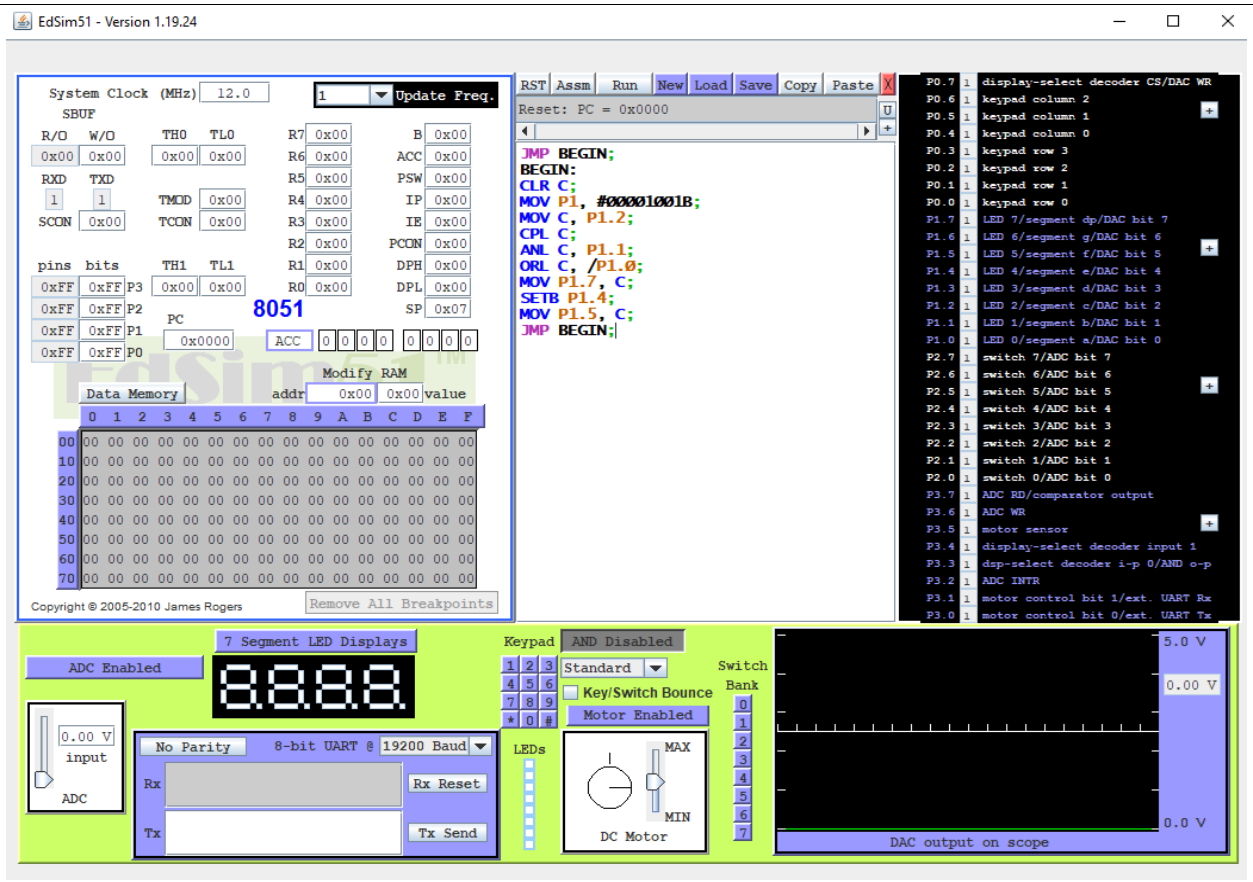


Рисунок 5 — Результаты работы программы

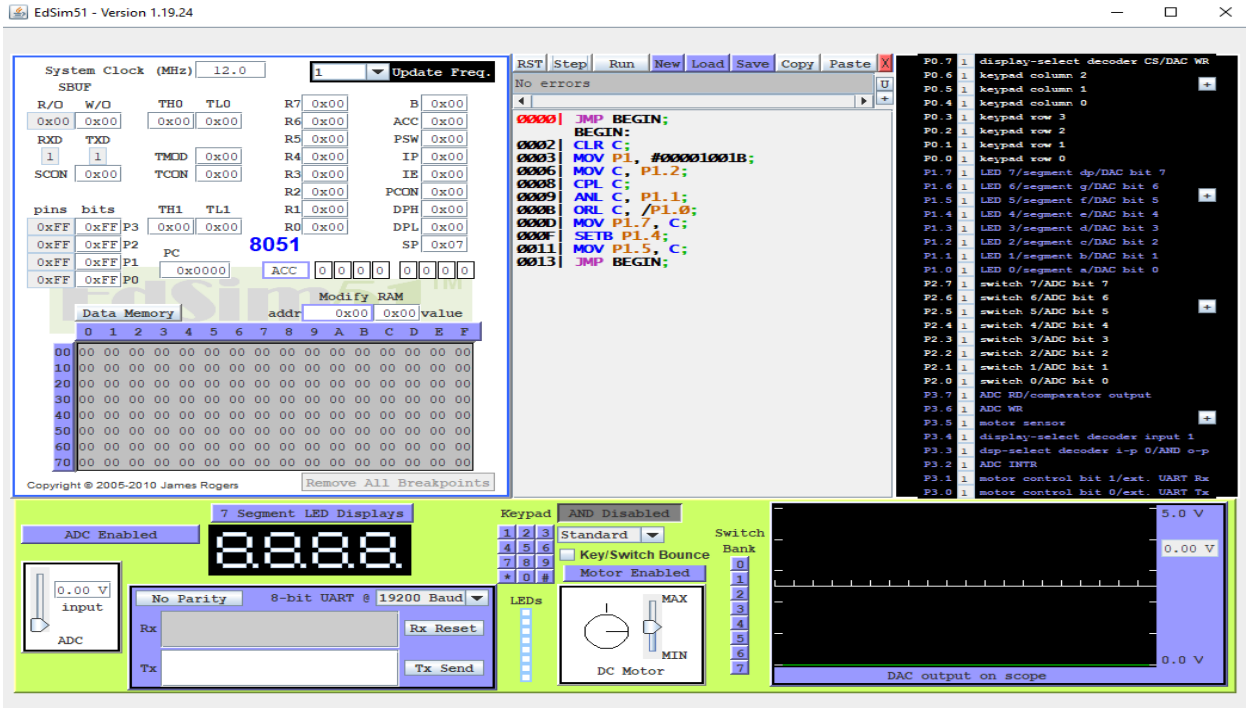


Рисунок 6 — Результаты работы программы

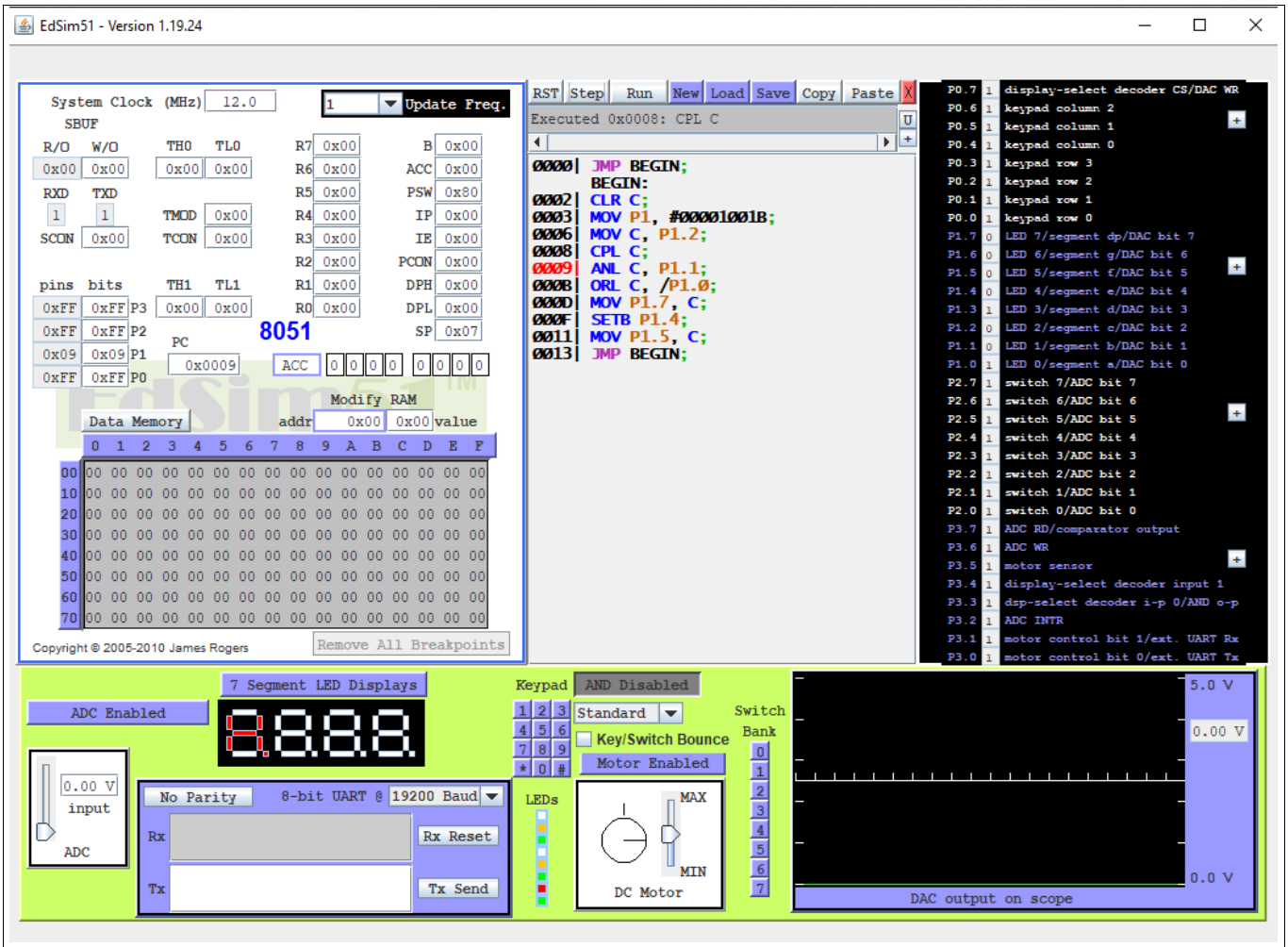


Рисунок 7 — Результаты работы программы

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 конституент
11	0, 1, 4, 6, 7, 8, 9, 15

Ход работы:

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

Таблица 9 — Таблица истинности функции F(x1,x2,x3,x4)

№	x1	x2	x3	x4	F(x1,x2,x3,x4)
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	1
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
12	1	1	0	0	0
13	1	1	0	1	0
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 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) ∨

∨ (x1&x2&x3&x4)

Программа lab5 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 C.

Система команд микроконтроллера MCS-51 представлена в Приложении Б (рис. Б.1-Б.5).

Код программы lab5:

```
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
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
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 рис. 8-10.

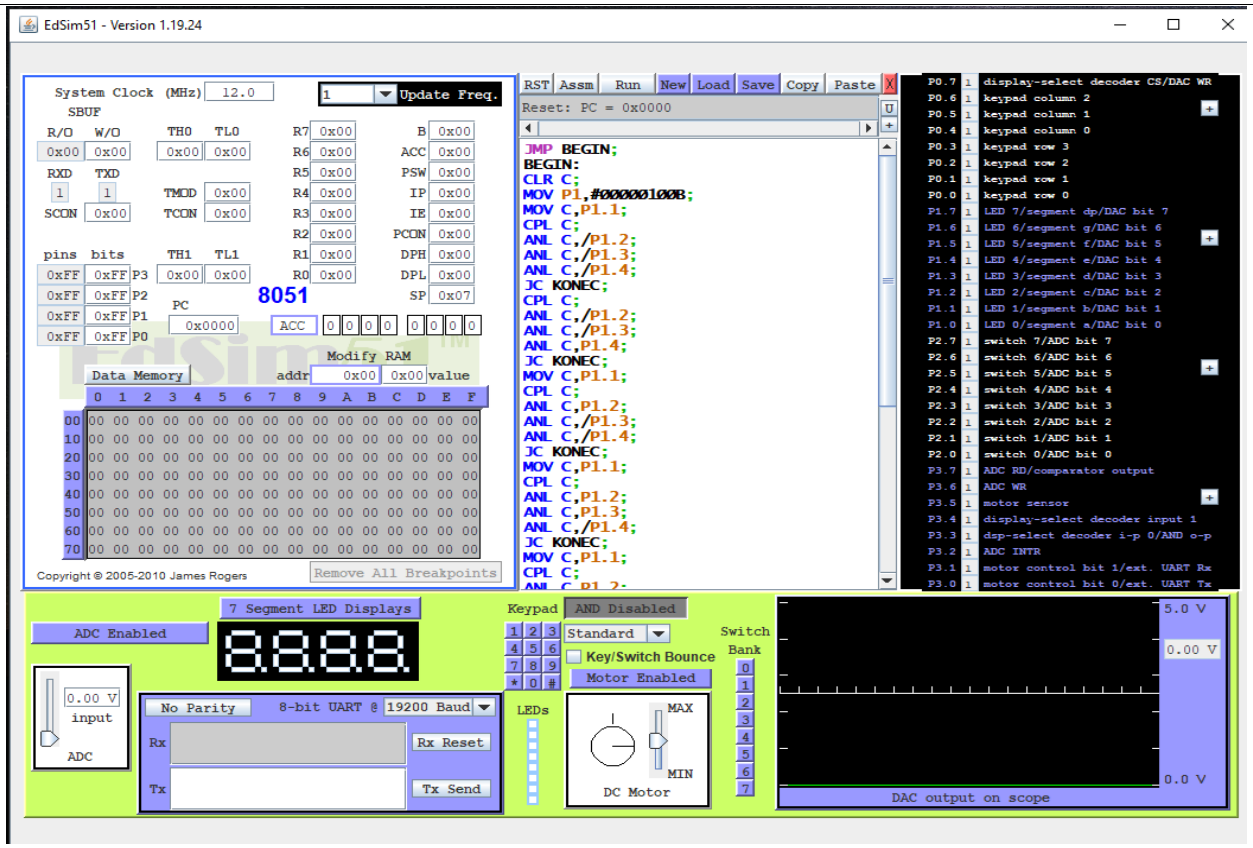


Рисунок 8 — Результаты работы программы

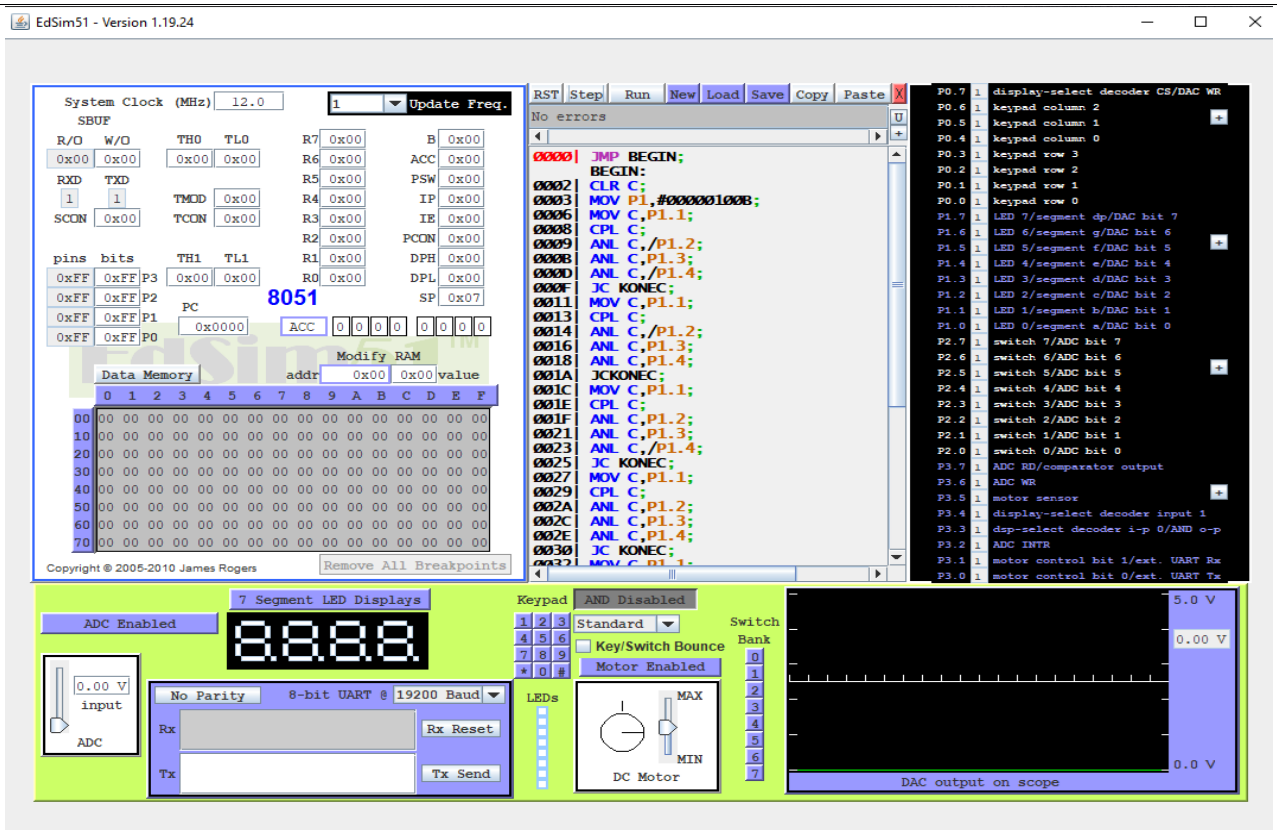


Рисунок 9 — Результаты работы программы

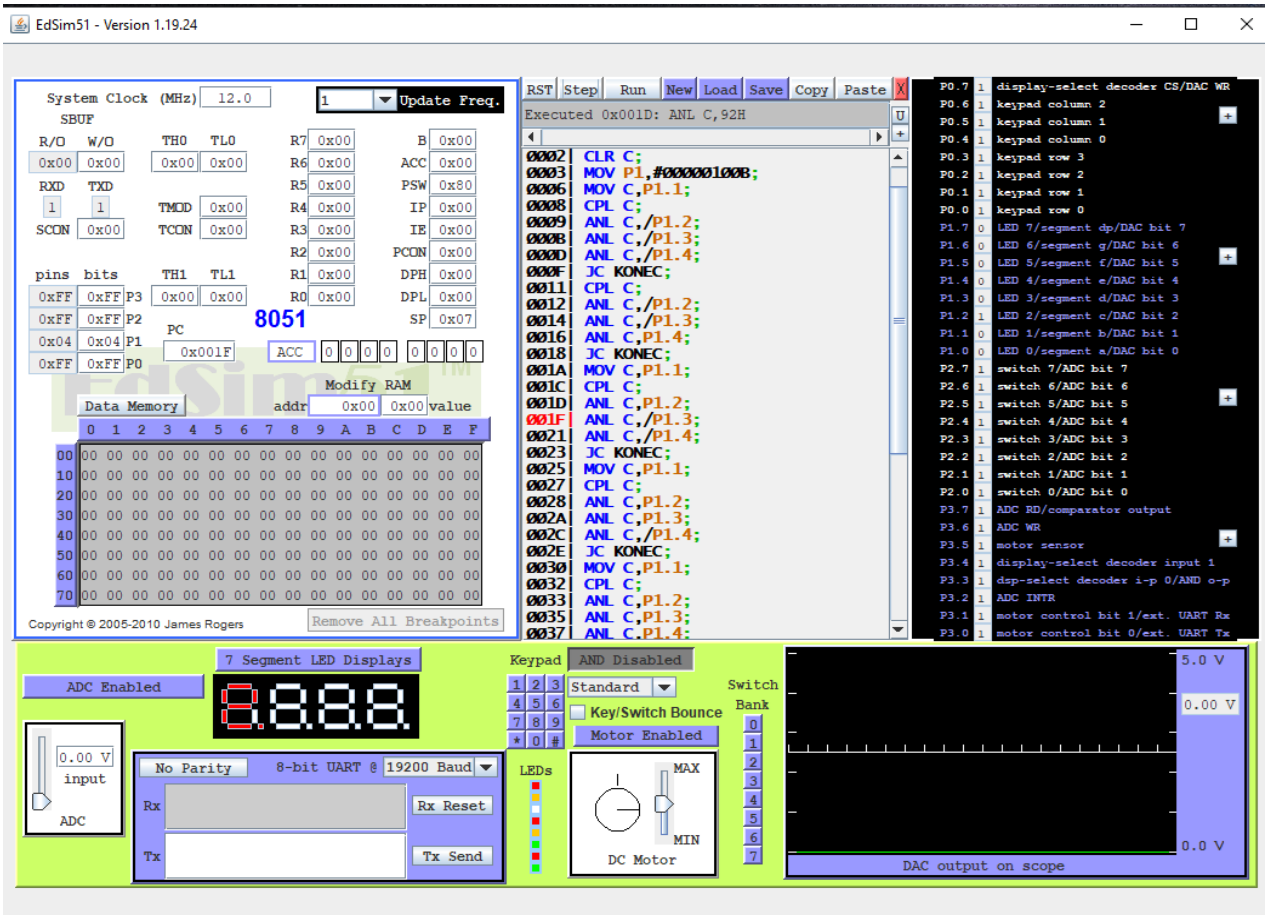


Рисунок 10 — Результаты работы программы

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.1.

Программа lab6 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 (рис. Б.1-Б.5).

Код программы lab6:

Код программы lab6 представлен в таблицы 10:

Таблица 10 — Код программы lab6

	Псевдокод	**Ассемблер MCS51**	**Комментарий**

BEGIN:	AN	JMP BEGIN;
		BEGIN: CLR C;
		MOV P0, #111111110B;
		MOV P2, #00000000B;
			Ввод значений условий
			Ввод X1-X5
			Начало AN

	AX1	MOV C, P0.1;	Ввод X1
	IF X1=0 GOTO A4	JNC A4;	
	AX2	MOV C, P0.2;	Ввод X2

	IF X2=0 GOTO A3	JNC A3;	
	A1	MOVP2, #00000001B;	Исп. A1
	AX3	MOV C, P0.3;	Ввод X3
	IF X3=0 GOTO AX5	JNC AX5;	
A2:	A2	A2: MOV P2, #00000010B;	Исп. A2
	GOTO AK	JMP AK;	
A4:	A4	A4: MOV P2, #00000100B;	Исп. A4
	AX4	MOV C, P0.4;	Ввод X4
	IF X4=0 GOTO AX5	JNC AX5;	
A5:	A5	A5: MOV P2, #00000101B;	Исп. A5
	GOTO AK	JMP AK;	
A3:	A3	A3: MOV P2, #00000011B;	Исп. A3
AX5:	AX5	AX5: MOV C, P0.5;	Ввод X5
	IF X5=0 GOTO A2	JNC A2;	
	GOTO A5	JMP AK;	
AK:	AK	AK: MOV P2, #00001100B; 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.

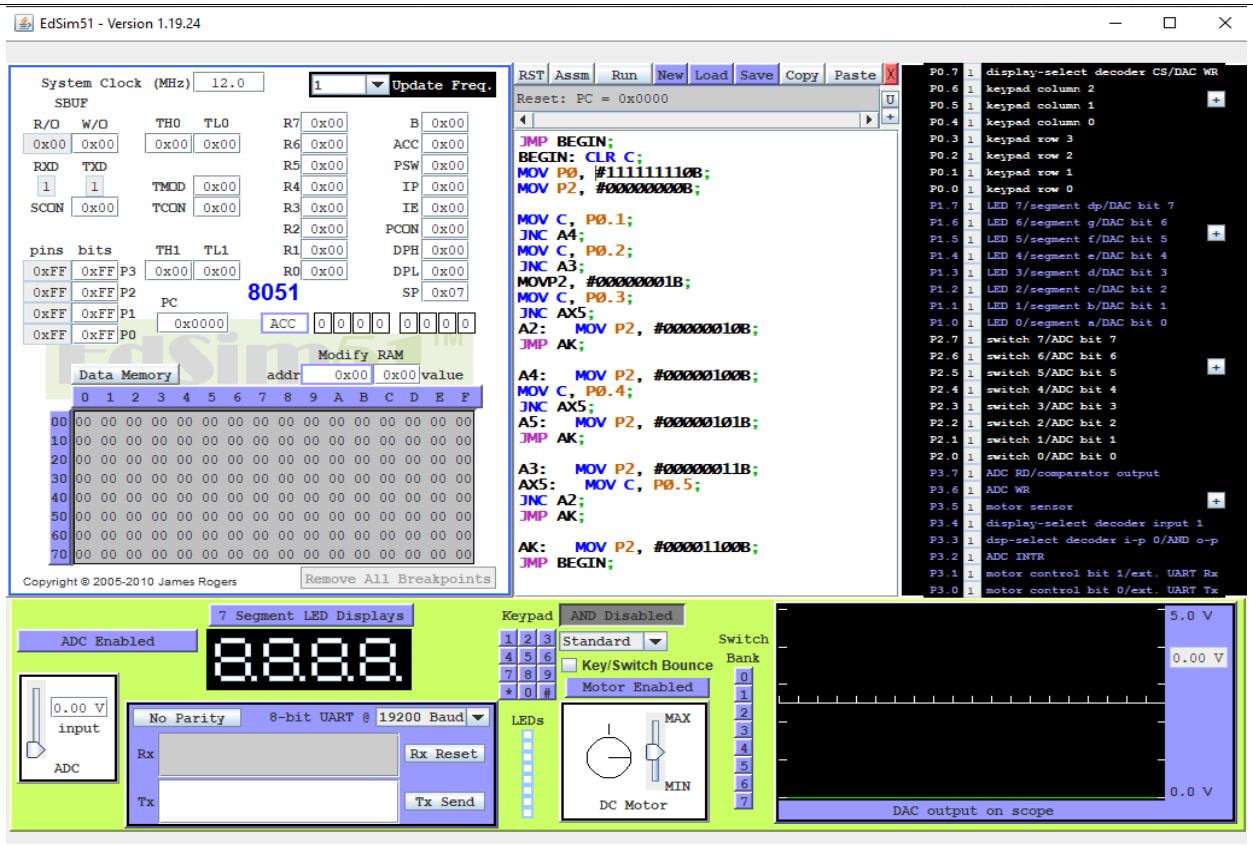


Рисунок 11 — Результаты работы программы

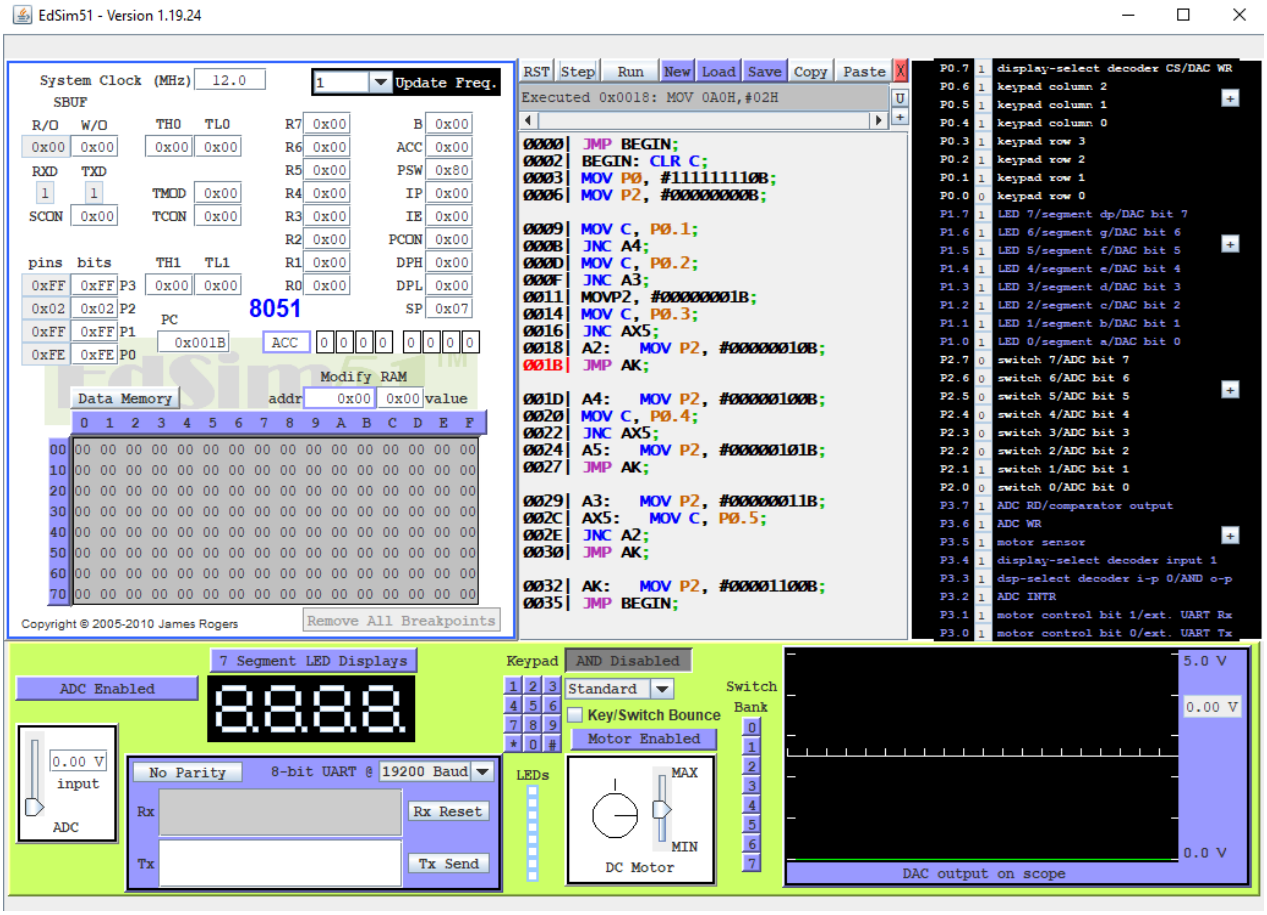


Рисунок 12 — Результаты работы программы

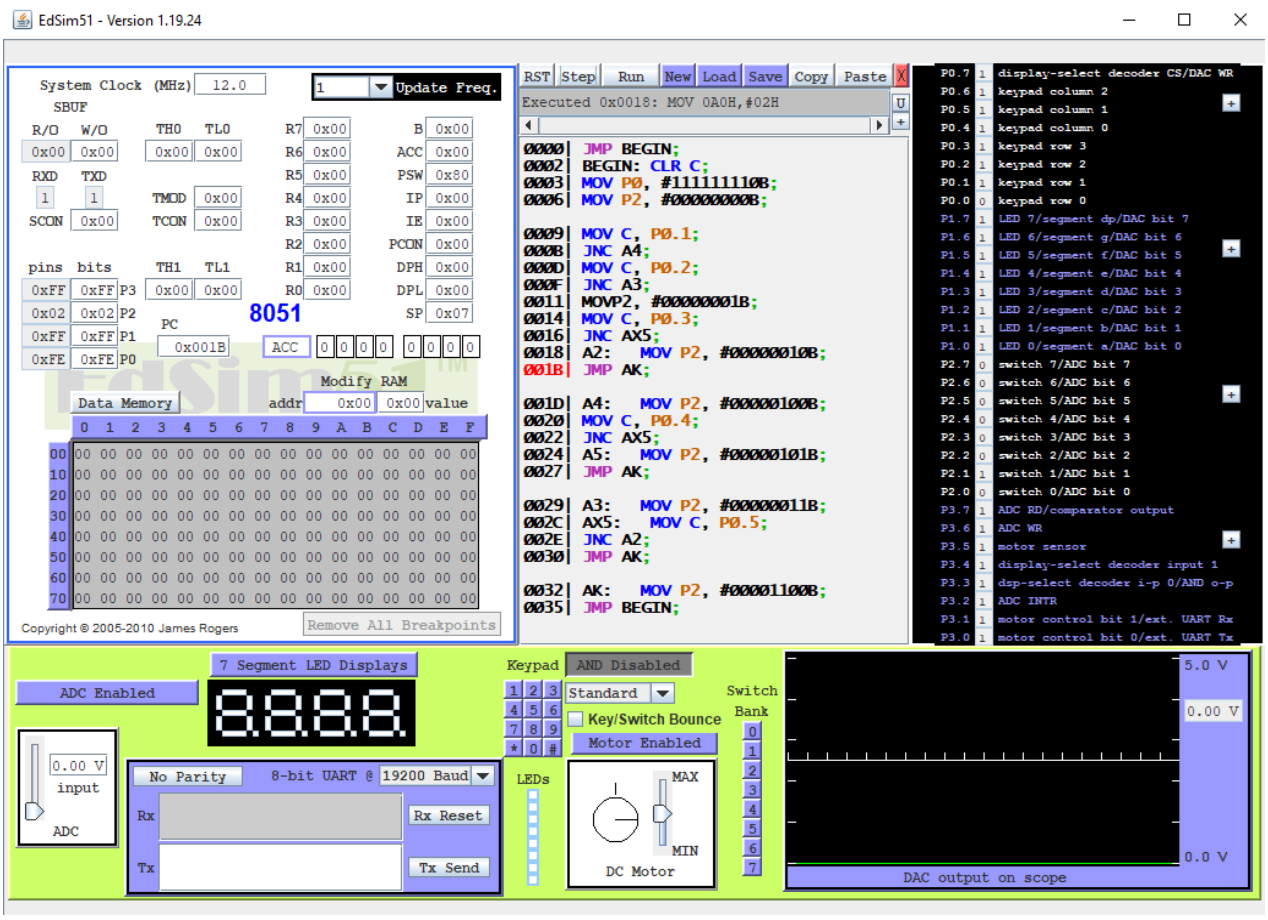


Рисунок 13 — Результаты работы программы

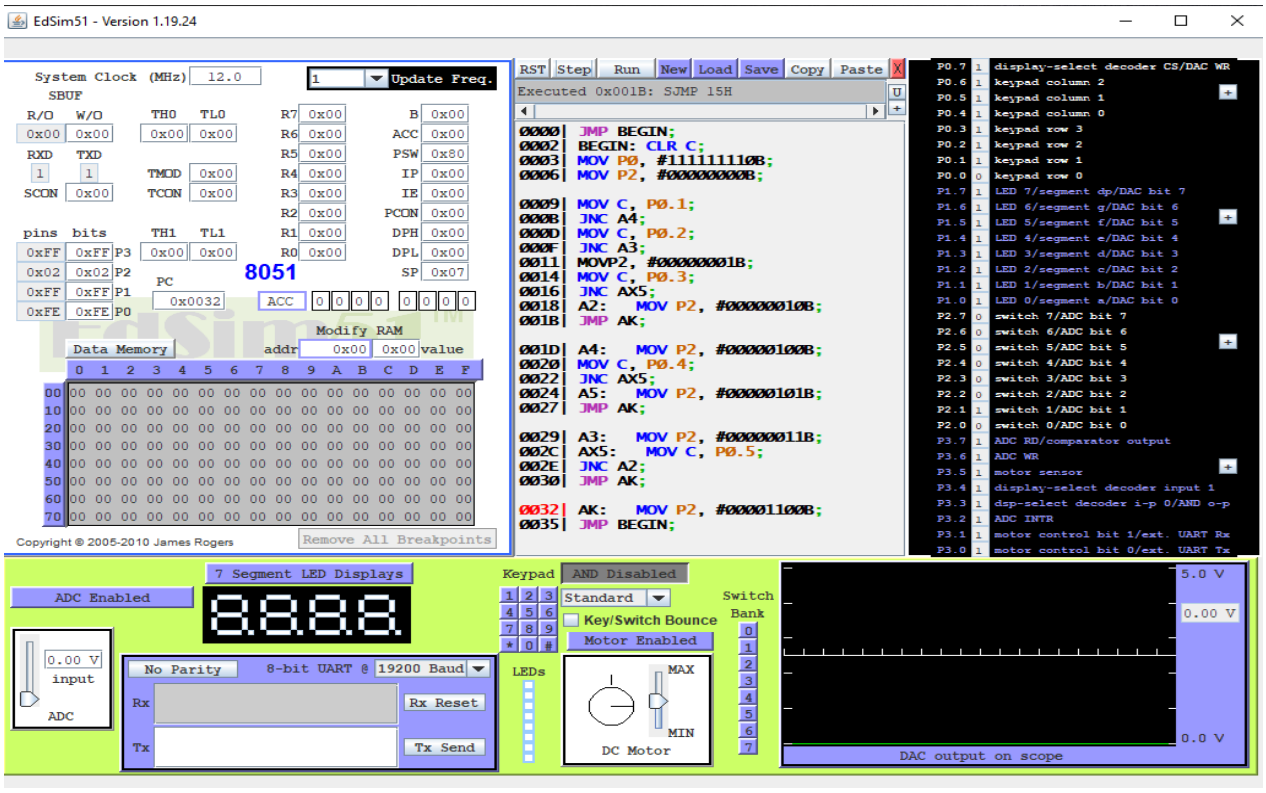


Рисунок 14 — Результаты работы программы

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)./3/

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. /2/

Кроме того, стенд включает источник постоянного напряжения и

осциллограф.

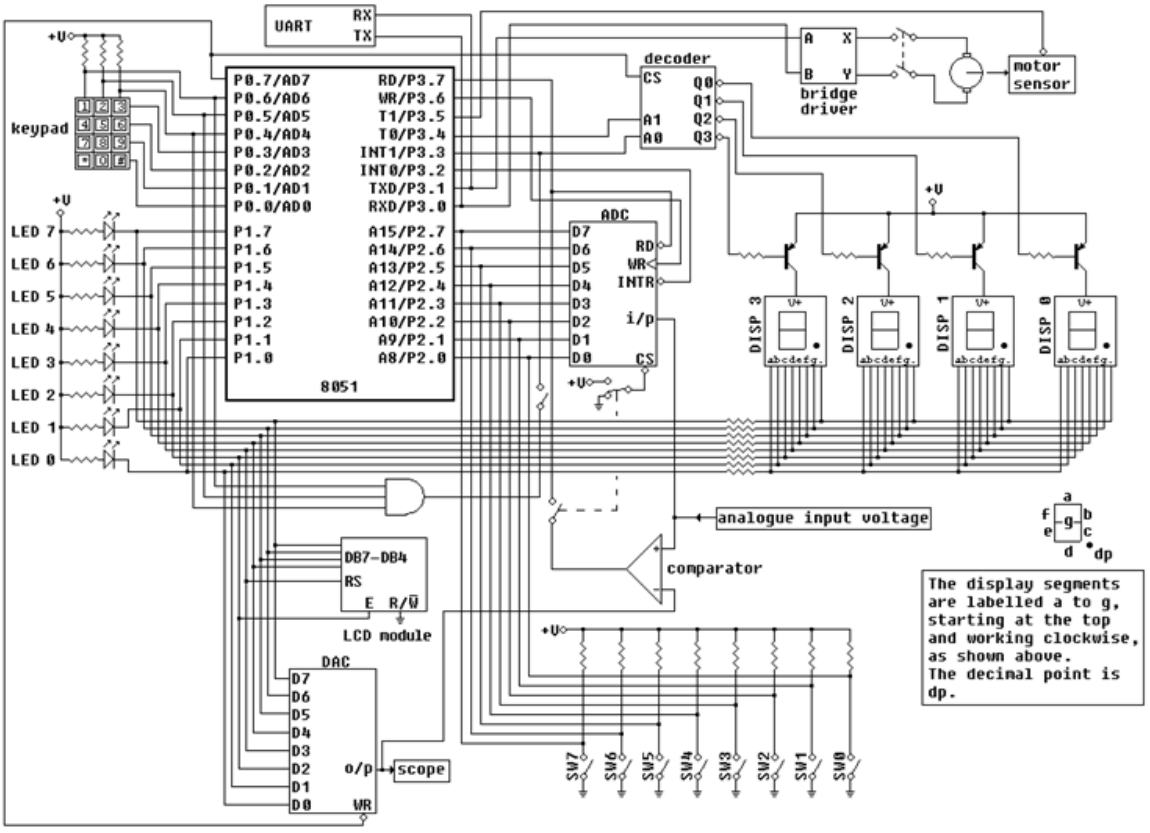


Рисунок 15 - Функциональная схема модели лабораторного стенда EdSim51

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.

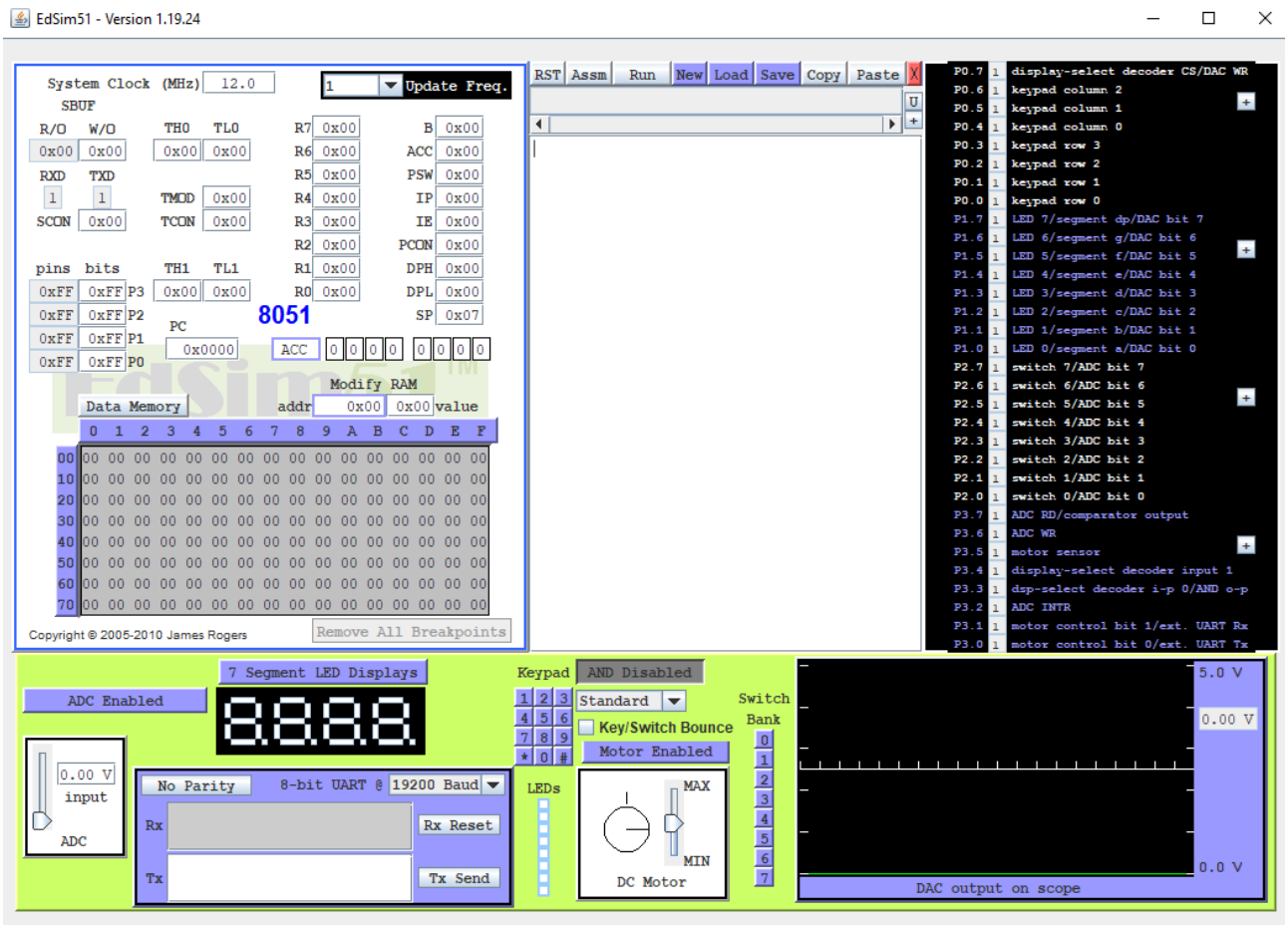


Рисунок 16 - Внешний вид модели лабораторного стенда EdSim51.

3 Руководство оператора

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

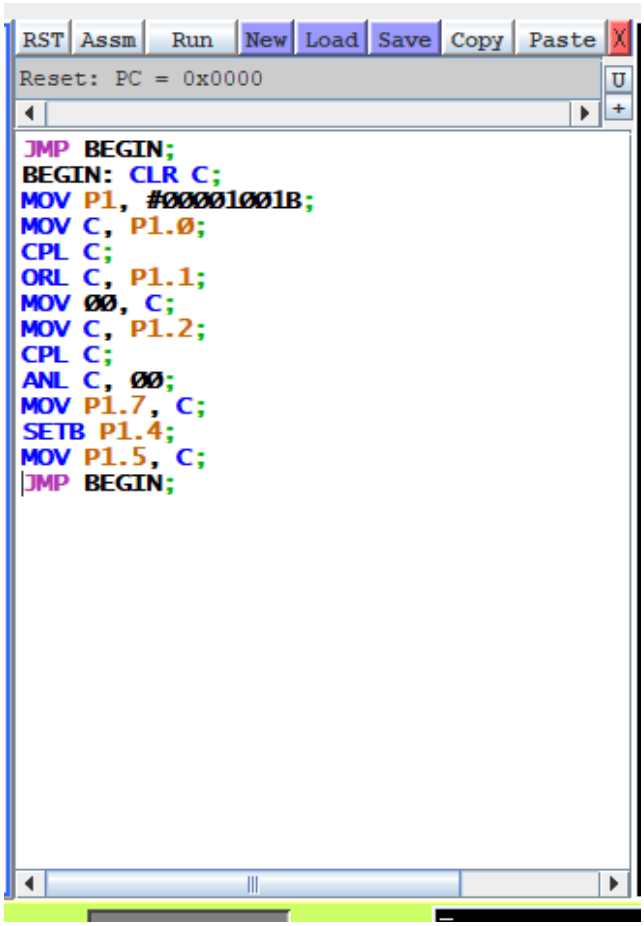


Рисунок 17 – Панель ассемблерного кода

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 (рис. 18).

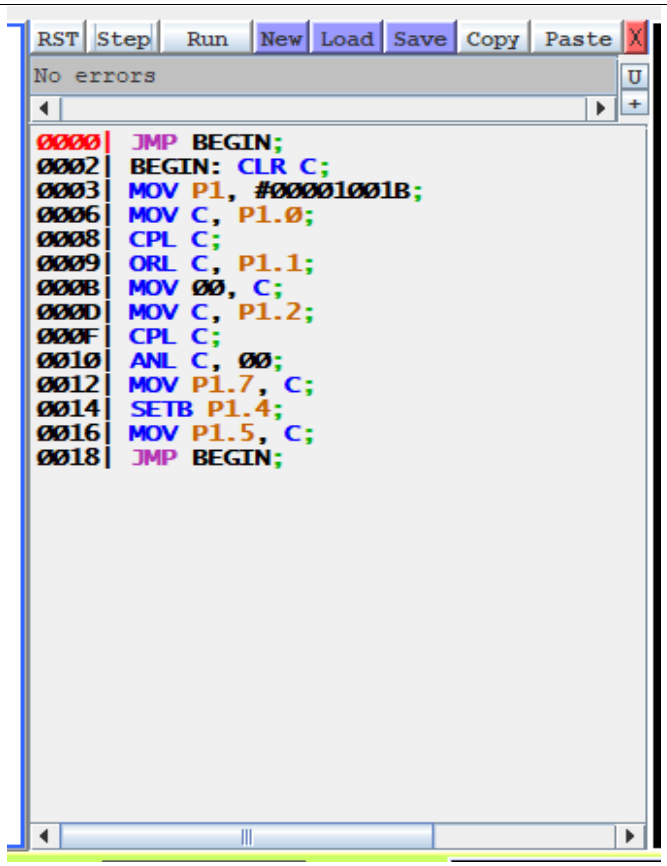


Рисунок 18 – Панель ассемблерного кода

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

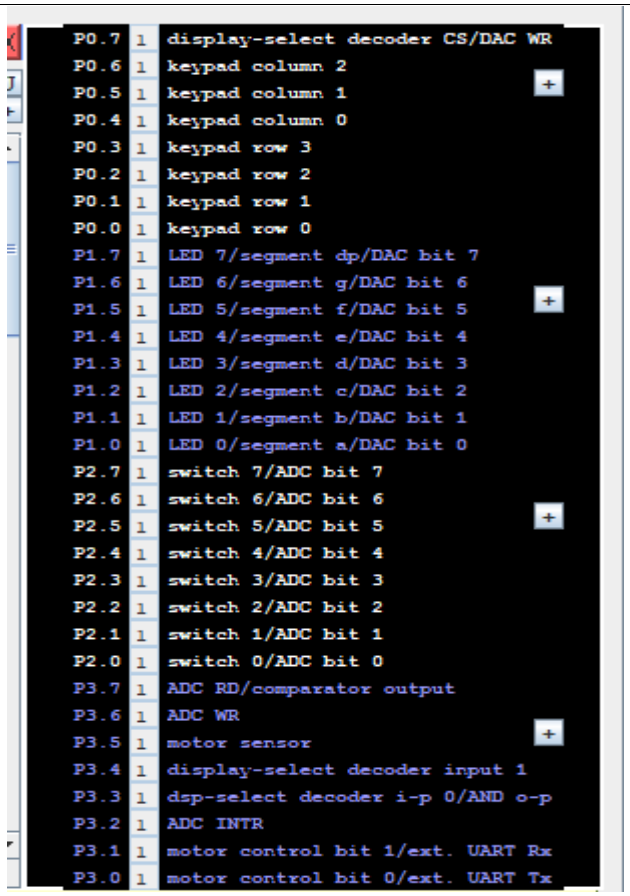


Рисунок 19 – Список регистров

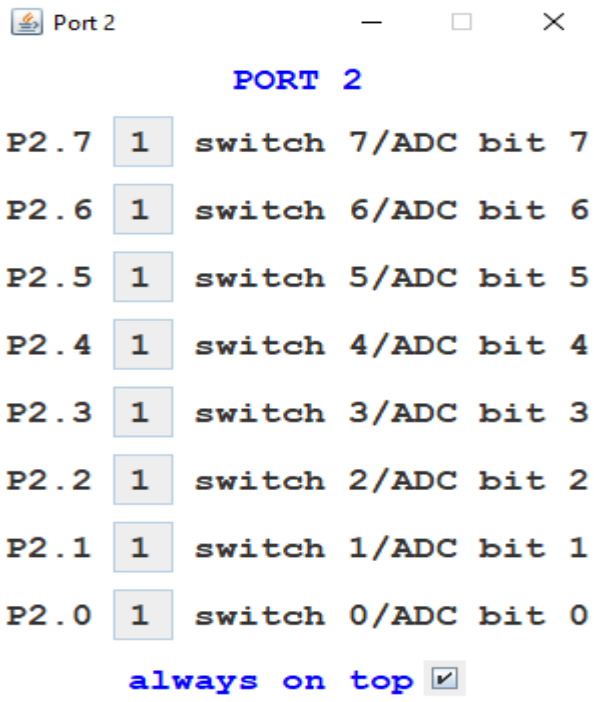


Рисунок 20 - Коммутация выводов P2.0-P2.7 и их состояния второго порта микроконтроллера

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. https://ru.wikipedia.org/wiki/История_персональных_компьютеров
2. Т.А. Павловская. С/С++. Программирование на языке высокого уровня.
3. Е.В. Моисейкин. Микроконтроллеры семейства MCS-51 Теория и практика.
4. Бьерн Страуструп. Язык программирования С++ .

АЛГОРИТМЫ

Приложение А

(обязательное)

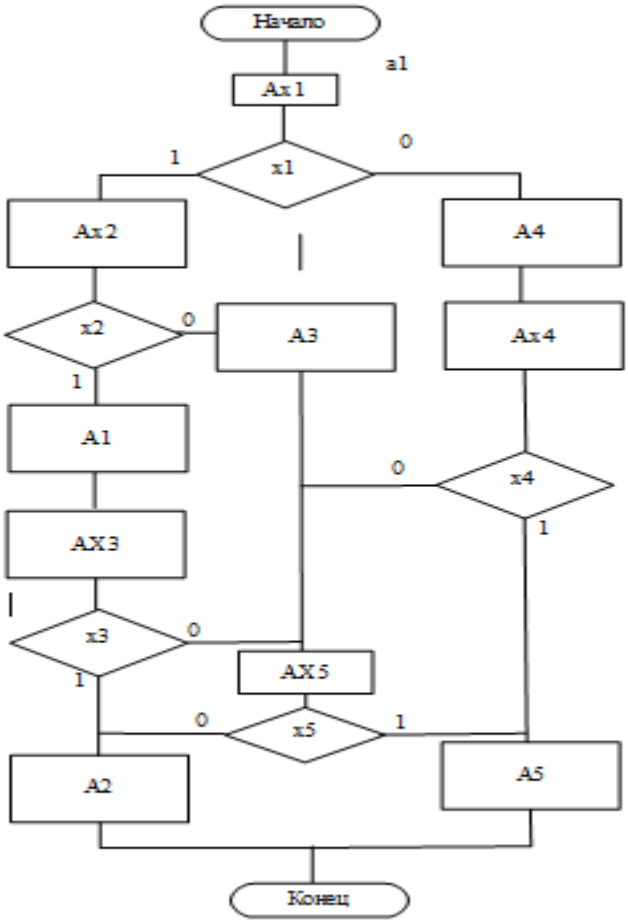


Рисунок А.1 – Схема программы

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