

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
И УПРАВЛЕНИЯ В ТЕХНИЧЕСКИХ СИСТЕМАХ

Кафедра информационных технологий и компьютерных систем

ИССЛЕДОВАНИЕ ПРОЦЕДУР СИНТЕЗА УПРАВЛЯЮЩИХ АВТОМАТОВ С ПРОГРАММИРУЕМОЙ ЛОГИКОЙ

Методические указания
к циклу лабораторных работ
по дисциплине «Архитектура ЭВМ. Специальный раздел»
для студентов дневной и заочной форм обучения направления
09.03.01 – Информатика и вычислительная техника и
09.03.04 – Программная инженерия
дневной формы обучения

Севастополь
СевГУ
2020

УДК 004.312.466:519.713(076.5)
ББК 32.965.7я7
И889

Р е ц е н з е н т:

В.С. Чернега - канд. техн. наук, доцент,
доцент кафедры информационных систем

Составители: Ю.К. Апраксин, Т.В. Волкова

И889 Исследование процедур синтеза управляющих автоматов с программируемой логикой : методические указания к циклу лабораторных работ по дисциплине «Архитектура ЭВМ. Специальный раздел» для студентов направлений 09.03.01 – Информатика и вычислительная техника и 09.03.04 – Программная инженерия дневной формы обучения / Сост. Ю.К. Апраксин, Т.В. Волкова ; Министерство образования и науки Российской Федерации, Севастопольский государственный университет, Институт информационных технологий и управления в технических системах, кафедра информационных технологий и компьютерных систем. – Севастополь : СевГУ, 2020. – 44 с. – Текст : электронный.

Методические указания предназначены для организации эффективной работы студентов в процессе подготовки и выполнения лабораторных работ, включенных в цикл «Исследование процедур синтеза управляющих автоматов с программируемой логикой». Основой каждой лабораторной работы является разработка микропрограммы по заданному алгоритму функционирования управляющего автомата и ее исследование. Методические указания к каждой лабораторной работе содержат четкую постановку задачи, варианты заданий, методические рекомендации к домашней подготовке к лабораторной работе, последовательность ее выполнения и содержание отчета. В приложении приводится описание программных средств, используемых при выполнении лабораторных работ.

УДК 004.312.466:519.713(076.5)
ББК 32.965.7я7

Методические указания рассмотрены и утверждены на заседании кафедры информационных технологий и компьютерных систем, протокол № 3 от 6 ноября 2019 г.

Допущено учебно-методическим центром СевГУ в качестве методических указаний.

Изд. № 80/2020. Объем 2,75 п.л.
РИИЦМ ФГАОУ ВО «Севастопольский государственный университет»

© ФГАОУ ВО «Севастопольский
государственный университет», 2020

СОДЕРЖАНИЕ

1. Общие сведения об управляющих автоматах (УА), построенных на основе принципа программируемой логики.....	4
2. Цель работ	8
3. Содержание работ	8
4. Лабораторная работа № 1. Разработка микропрограммы для УА использующего принцип естественной адресации микрокоманд	10
5. Лабораторная работа № 2. Разработка микропрограммы для УА использующего принцип принудительной адресации микрокоманд	24
Библиографический список.....	34
Приложение А.....	35

1. ОБЩИЕ СВЕДЕНИЯ ОБ УПРАВЛЯЮЩИХ АВТОМАТАХ (УА), ПОСТРОЕННЫХ НА ОСНОВЕ ПРИНЦИПА ПРОГРАММИРУЕМОЙ ЛОГИКИ

В основе идеи микропрограммирования (использования принципа «программируемой» логики) лежит тот факт, что для инициирования любой микрооперации (МО) или их совокупности достаточно сформировать управляющее двоичное слово, в котором каждый бит соответствует одному управляющему сигналу, иницирующему конкретную МО. Такое управляющее слово называют микрокомандой (МК). Последовательность МК, реализующих определенный алгоритм функционирования управляющего автомата (УА), образует микропрограмму (МП).

Характерной особенностью УА с программируемой логикой является хранение МП в специализированном постоянном запоминающем устройстве (ПЗУ), называемом памятью микропрограмм (ПМП). Обобщенная структура УА с хранимой в памяти логикой изображена на рисунке 1.

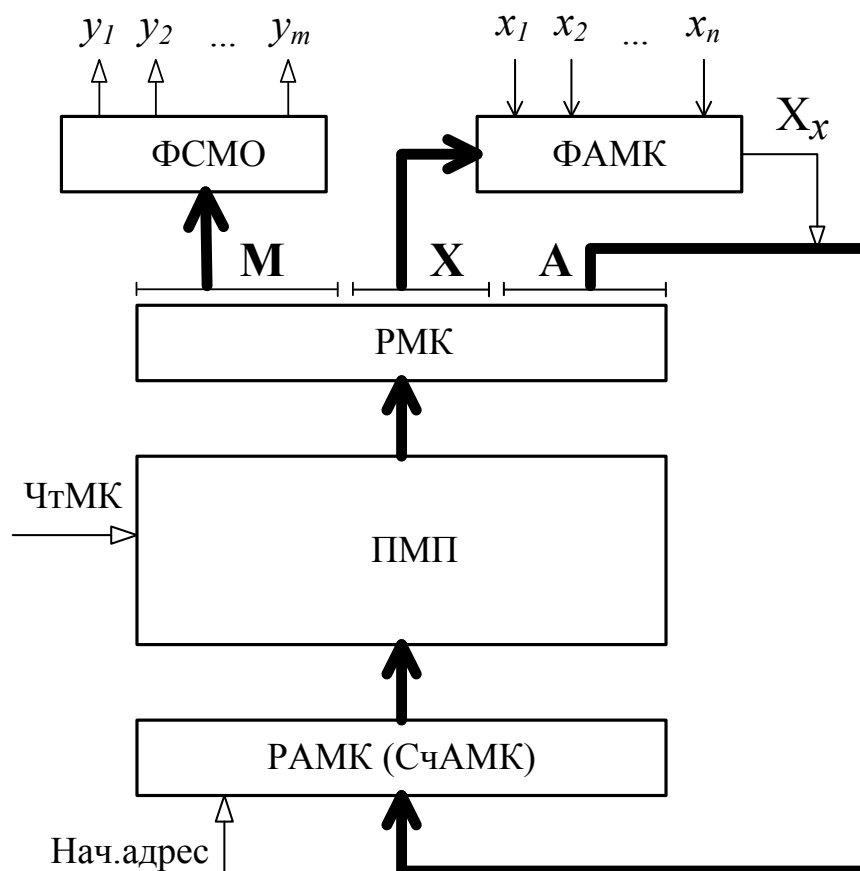


Рисунок 1 – Обобщенная структура УА с программируемой логикой

В состав устройства кроме ПМП входят:

- регистр микрокоманды (РМК);
- регистр адреса микрокоманды (РАМК);
- формирователь сигналов микроопераций (ФСМО);
- формирователь адреса микрокоманды (ФАМК).

Запуск микропрограммы выполнения того или иного вычислительного процесса осуществляется в результате подачи на РАМК «начального адреса» МП (адреса первой микрокоманды конкретной МП) и последующего выбора текущей МК по сигналу ЧтМК (чтение МК). Выбранная МК попадает на РМК.

Каждая МК в общем случае должна содержать операционную (М) и адресную (Х . А) части. Код операционной части поступает на ФСМО, на выходе которого формируются управляющие сигналы $y_1, y_2, \dots, y_m$, инициирующие выполнение МО в обрабатывающем устройстве. Код адресной части МК состоит из двух полей. Первая часть адресного кода (Х), задающая номер проверяемого логического условия, подается на ФАМК. Формирователь адреса среди поступивших на его информационные входы осведомительных сигналов $x_1, x_2, \dots, x_n$ выбирает x_x (сигнал с номером, заданным полем Х МК) и в зависимости от его значения (0 или 1) формирует адрес следующей исполняемой МК, который фиксируется в РАМК.

В данном цикле лабораторных работ рассматриваются все основные подходы к проектированию МП для УА с программируемой логикой. Каждый подход характеризуется в первую очередь способом кодирования операционной части МК. На рисунке 2 изображены структуры операционной части МК и схемы расшифровывания кодов микроопераций, зафиксированных в поле М.

При горизонтальном способе кодирования МО (рисунок 2, а) под каждый управляющий сигнал y_i ($i=1, 2, \dots, m$) в операционной (М) части МК выделяется один (свой) двоичный разряд, что позволяет в рамках одной МК формировать любые сочетания управляющих сигналов. ФСМО для такого способа кодирования обладает наименьшей стоимостью и минимальным временем срабатывания.

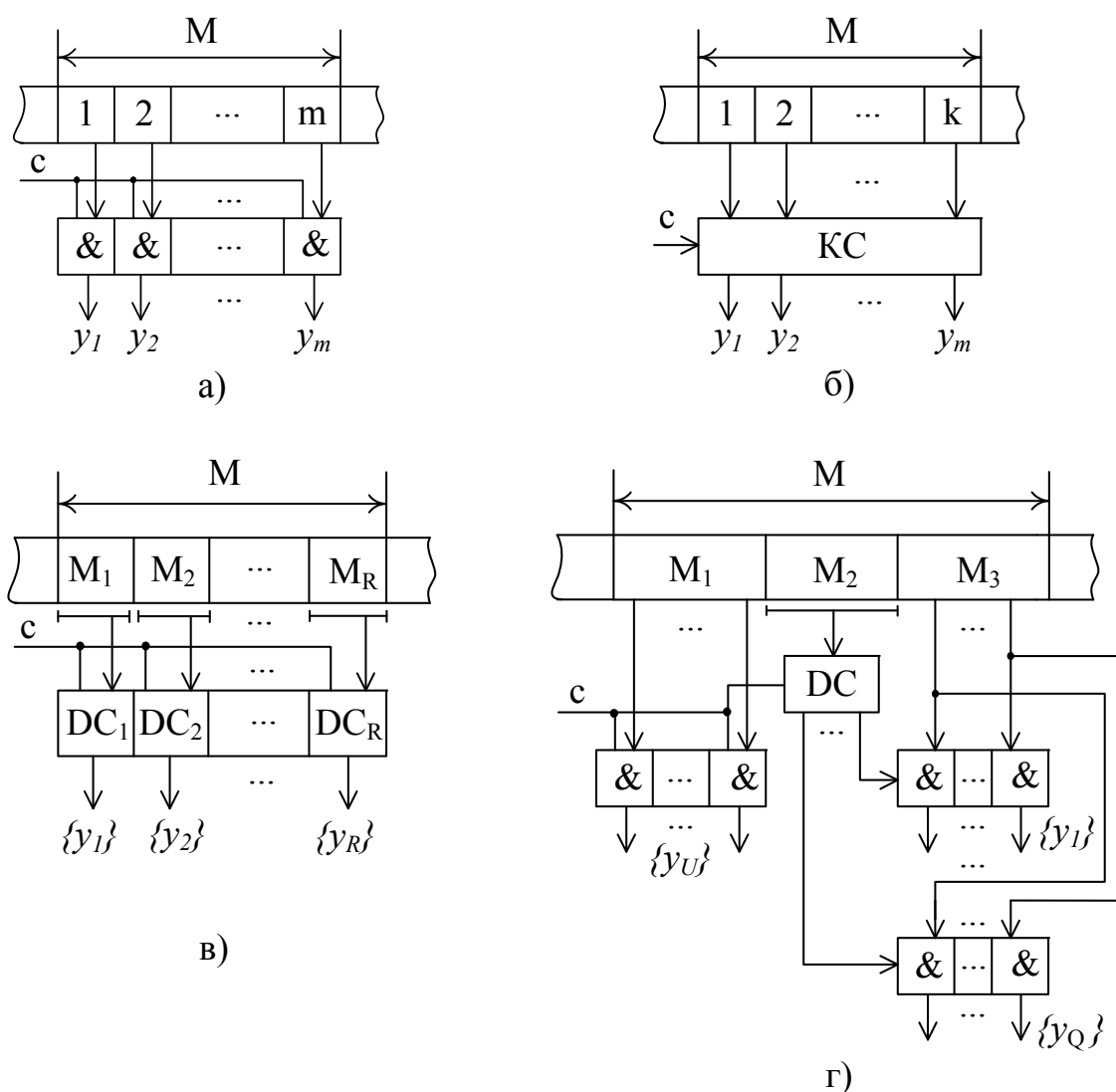


Рисунок 2 – Форматы операционной части МК и структуры ФСМО

при различных способах кодирования МО:

- а) горизонтальный способ;
- б) вертикальный способ;
- в) горизонтально-вертикальный способ;
- г) вертикально-горизонтальный способ

Недостаток – линейный рост длины поля операционной части при увеличении общего количества выполняемых МО.

При вертикальном способе кодирования (рисунок 2, б) каждой различимой совокупности МО, включенной в ту или иную МК, присваивается позиционный код минимальной длины $k = \lceil \log_2 \tilde{m} \rceil$, где $\tilde{m}$ – количество различимых по операционной части МК. ФСМО в этом случае представляет

собой комбинационную схему (КС), реализующую m булевых функций от k переменных.

При горизонтально-вертикальном способе кодирования МО (рисунок 2, в) все множество МО разбивается на подмножества, в каждое из которых включаются только несовместимые по времени исполнения МО. Внутри каждого подмножества сигналы управления кодируются вертикальным способом. Подмножества в операционной части МК располагаются по горизонтальному принципу. Другое название этого способа – кодирование раздельными полями. Расшифровка кодов МО осуществляется ФСМО, представляющим собой R дешифраторов (по одному на каждое выделенное подмножество МО).

При вертикально-горизонтальном способе кодирования (рисунок 2, г) все множество МО также делится на подмножества, однако в каждое подмножество включаются только те МО, которые связаны между собой отношением совместимости по времени исполнения (встречаются вместе хотя бы в одной МК). Для всех этих подмножеств выделяется в операционной части МК одно поле M_3 , длина которого определяется максимальным количеством МО в подмножествах. Принцип кодирования МО в поле M_3 – горизонтальный. Идентифицирующее поле M_2 заполняется вертикальным кодом номера подмножества, зафиксированного в поле M_3 . Отличительной особенностью вертикально-горизонтального способа кодирования является требование несовместимости выделенных подмножеств МО между собой. Удовлетворить этому требованию можно, выделив наиболее часто встречающиеся в МК микрооперации в отдельное подмножество (универсальную группу). Кодирование МО универсальной группы – горизонтальное. Код универсальной группы помещается в поле M_1 операционной части МК. Другое название этого способа – кодирование несовместимыми подмножествами МО.

2. ЦЕЛЬ РАБОТ

Выполнение лабораторных работ преследует следующие цели: изучение принципов построения управляющих автоматов с программируемой логикой, методов адресации МК и способов кодирования МО, приобретение навыков разработки, отладки и исследования микропрограмм с помощью программного эмулятора УА [1, стр. 283 – 307].

3. СОДЕРЖАНИЕ РАБОТ

Выполнение каждой лабораторной работы предусматривает самостоятельную подготовку, выполнение задания в лаборатории и оформление отчета. Для облегчения подготовки к лабораторной работе методические указания содержат ссылки на учебно-техническую литературу с указанием разделов, которые необходимо изучить перед выполнением лабораторной работы.

3.1. Самостоятельная подготовка к выполнению лабораторной работы

В ходе подготовки к выполнению лабораторной работы необходимо:

- разработать в соответствии с вариантом задания микропрограмму УА в условных обозначениях;
- на основе минимального множества тестов провести тестирование МП;
- провести кодирование операционной части МК;
- изучить руководство пользователя эмулятора УА.

Тест представляет собой набор значений осведомительных сигналов $x_1^{\sigma_1} x_2^{\sigma_2} \dots x_n^{\sigma_n}$ и последовательность управляющих сигналов, определяемых по ГСА функционирования УА в соответствии с заданным набором.

Универсальное множество тестов состоит из 2^n наборов осведомительных сигналов (от 00...0 до 11...1) с соответствующими им последовательностями управляющих сигналов.

Минимальное множество тестов есть минимально возможная совокупность тестов, покрывающая все существующие дуги ГСА функционирования УА.

Под тестированием МП понимается моделирование поведения УА по разработанной МП для всех тестовых наборов из минимального множества тестов.

3.2. Выполнение работ в лаборатории

В процессе выполнения работы необходимо:

- разработать форматы МК;
- представить разработанную и оттестированную МП в естественных адресах;
- с помощью клавиатуры, используя программную среду эмулятора, ввести МП в память компьютера;
- отладить, пользуясь подготовленными тестами, составленную МП;
- представить итоги в виде протокола трассировки МП.

Особенности выполнения каждой работы, варианты заданий, примеры составления МП для УА, способы тестирования и контрольные вопросы изложены в разделах описания соответствующих лабораторных работ.

3.3. Содержание отчета

Отчет по лабораторной работе должен содержать следующие разделы:

- формулировку цели работы;
- постановку задачи (в соответствии с вариантом);
- тексты разработанных МП в условных обозначениях;
- результаты тестирования МП;
- разработанные форматы МК и результаты кодирования МО;
- микропрограмма в естественных адресах;
- протоколы трассировки МП;
- схему УА с заданным типом адресации МК для разработанной МП;
- схемы ФСМО для способов кодирования МО, заданных вариантом;
- выводы, указывающие на трудности, возникшие при выполнении работы, и способы их преодоления.

В процессе защиты отчета студент должен продемонстрировать знание теоретического материала, умение составлять МП для УА и навыки работы с эмулятором. Отчет по лабораторной работе должен быть представлен преподавателю на следующем после выполнения лабораторной работы занятии.

4. ЛАБОРАТОРНАЯ РАБОТА № 1. РАЗРАБОТКА МП ДЛЯ УА, ИСПОЛЬЗУЮЩЕГО ПРИНЦИП ЕСТЕСТВЕННОЙ АДРЕСАЦИИ МК

4.1. Варианты заданий

Варианты заданий приведены в таблице 1 и отличаются друг от друга способом кодирования МО и ГСА функционирования УА (рисунок 3).

Таблица 1 – Варианты заданий на лабораторную работу № 1

Номер варианта	Способ кодирования МО	Номер ГСА
1	Г, Г-В	1
2	Г, В-Г	2
3	В, Г-В	3
4	В, В-Г	4
5	В-Г, Г-В	5
6	В, В-Г	1
7	В, Г-В	2
8	Г, В-Г	3
9	Г, Г-В	4
10	Г, В	5

В таблице 1 использованы следующие обозначения способов кодирования МО: Г – горизонтальный, В – вертикальный, Г-В – горизонтально-вертикальный (раздельными полями), В-Г – вертикально-горизонтальный (несовместимыми подмножествами).

4.2. Теоретическая подготовка

При подготовке к лабораторной работе необходимо изучить [2, стр.254 – 276].

Главной особенностью всех вариантов задания на лабораторную работу №1 является использование принципа естественной адресации МК. Такой подход к проектированию МП предполагает использование двух различных форматов МК: **В.М** – для операционных МК и **В.Х.А** – для управляющих МК. Здесь В, М, Х, А – поля микрокоманд: В – одноразрядное поле бит-маркера (В=0 для операционных МК и В=1 для управляющих МК), М – поле для представления кода МО, включенных в МК, Х – поле кода (номера) проверяемого логического условия, А – поле адреса МК, исполнение которой осуществляется в случае истинности проверяемого логического условия.

Исполнительный адрес МК вычисляется по следующему правилу:

$$A_{исп} = \begin{cases} A, & \text{если } X = i, \\ CчАМК + 1, & \text{если } X \neq i. \end{cases}$$

Здесь i – индекс осведомительного сигнала x_i , значение которого равно 1.

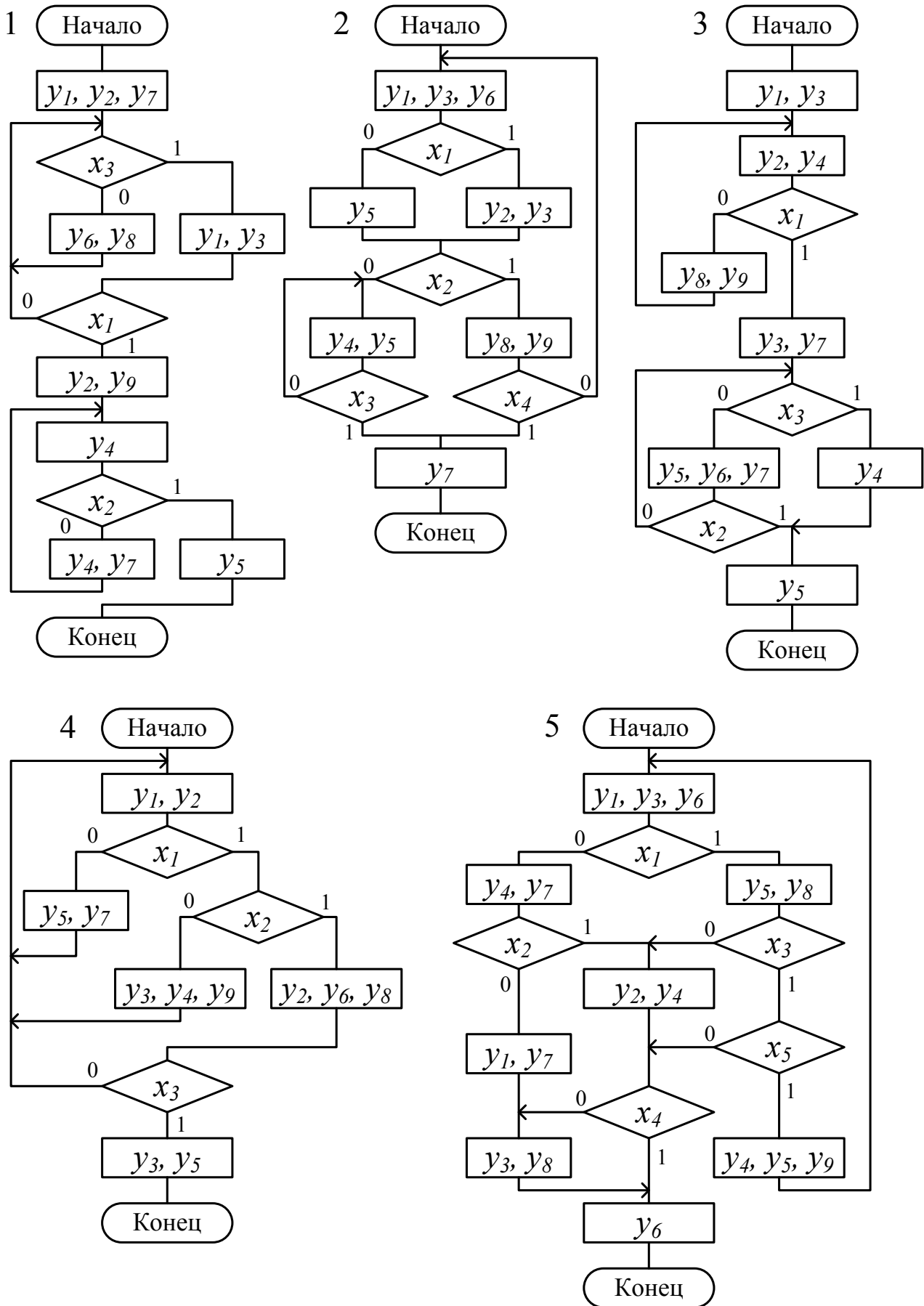


Рисунок 3 – ГСА функционирования УА

4.3. Пример разработки МП для УА

На рисунке 4, а изображена ГСА функционирования УА с программируемой логикой. Порядок этапов разработки МП можно определить следующим образом:

- а) определение минимального множества тестов;
- б) расстановка адресов МК;
- в) составление МП в условных обозначениях;
- г) тестирование МП в условных обозначениях;
- д) кодирование операционной части МК;
- е) разработка форматов МК;
- ж) составление двоичного кода МП.

Выполнение пункта а) предполагает определение минимального множества путей из вершины «Начало» в вершину «Конец» на ГСА, причем выбранная совокупность путей при фиксированном наборе значений осведомительных сигналов для каждого пути должна покрывать все дуги ГСА.

Пути на рисунке 4, а изображены пунктирными линиями и обозначены, соответственно, P_1 , P_2 , P_3 . Соответствующие выделенным путям тесты изображены на рисунке 5.

Выполнение пункта б) технологического процесса разработки МП предполагает расстановку адресов, начиная с базового адреса $A+0$, для каждой вершины ГСА по пути невыполнения проверяемых логических условий. Правило вычисления каждого следующего адреса – инкрементирование предыдущего (естественная адресация). Если очередная дуга приводит к вершине с уже проставленным адресом, то на этой дуге отмечается дополнительная МК безусловного перехода (на дуге между вершинами, отмеченными адресами $A+5$ и $A+2$, вставляется МК с адресом $A+6$, на дуге между вершинами $A+3$ и $A+1$ вставляется МК с адресом $A+4$).

Результатом выполнения пункта в) будет МП в условных обозначениях, приведенная в таблице 2.

Таблица 2 – МП в условных обозначениях

Адрес МК	Микрокоманда			Примечание
	В	М		Операционная МК
	В	Х	А	Управляющая МК
A+0	0	y_1, y_2		операционная МК
A+1	1	x_1	A+5	условный переход
A+2	0	y_3, y_5		
A+3	1	x_2	A+7	
A+4	1	$x_{\bar{0}n}$	A+1	безусловный переход
A+5	0	y_1, y_3		
A+6	1	$x_{\bar{0}n}$	A+2	
A+7	1	x_3	A+9	
A+8	0	y_4		
A+9	0	y_2, y_6		
A+10	1	$x_{\bar{0}n}$	A+0	

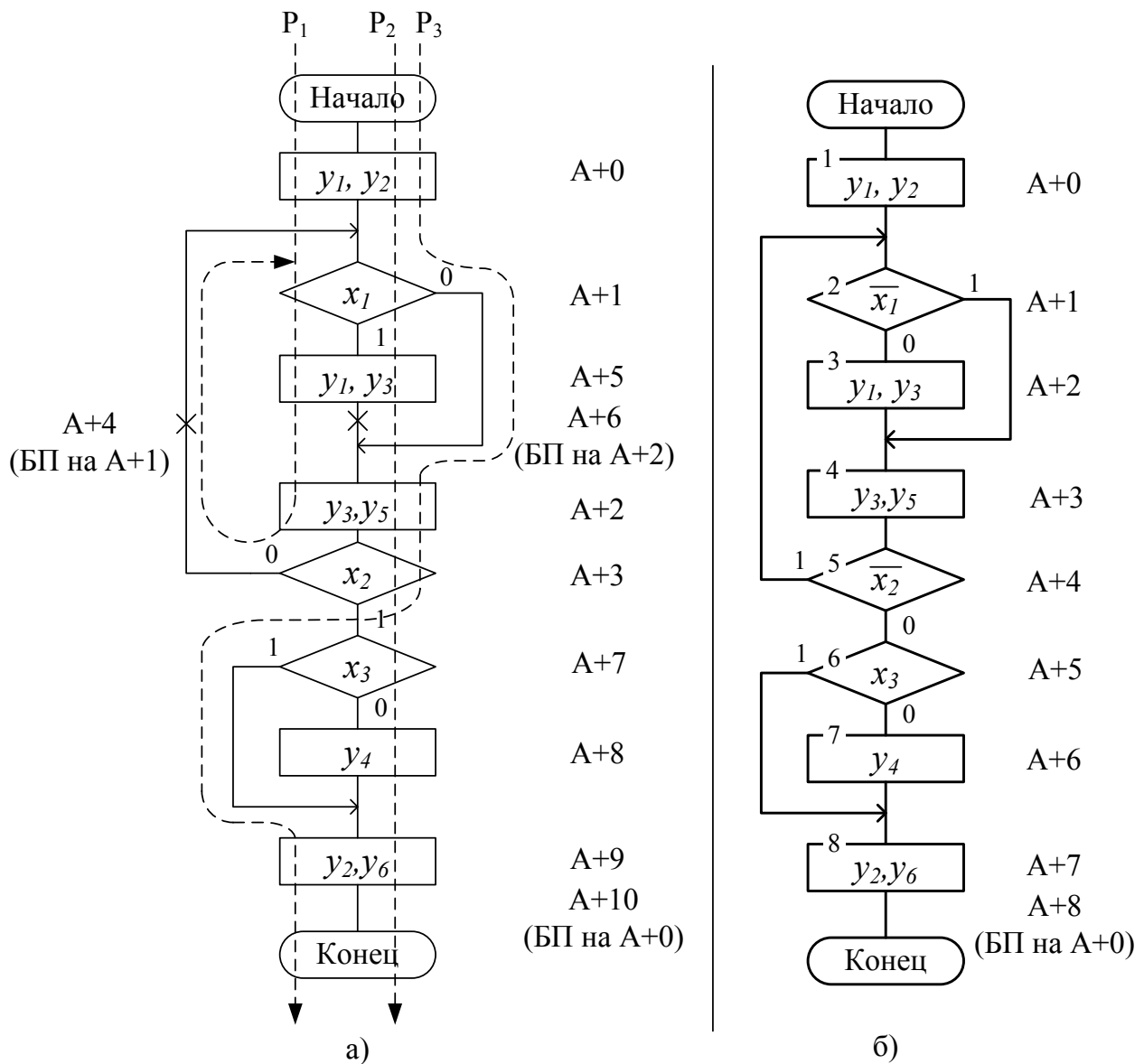


Рисунок 4 – ГСА функционирования УА (пример): а) первый вариант расстановки адресов; б) второй вариант расстановки адресов

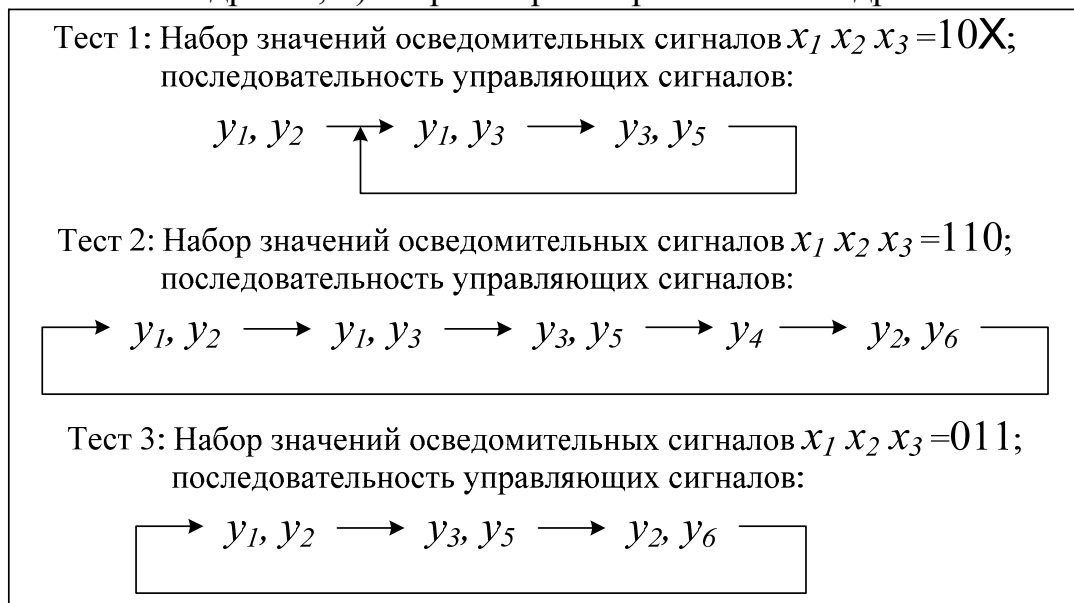


Рисунок 5 – Тесты для проверки МП

Результатом выполнения пункта г) процесса разработки МП является последовательность адресов исполняемых МК микропрограммы с указанием генерируемых этими МК управляющих сигналов.

Тестирование производится для всех разработанных при исполнении пункта а) тестов. Генерация адресов прекращается при повторении какого-либо адреса.

Тест 1: $x_1 x_2 x_3 = 10X$

A+0	y_1, y_2
A+1	— $x_1=1$
A+5	y_1, y_3
A+6	— $x_{\bar{o}n} \equiv 1$
A+2	y_3, y_5
A+3	— $x_2=0$
A+4	— $x_{\bar{o}n} \equiv 1$
A+1	

Тест 3: $x_1 x_2 x_3 = 011$

A+0	y_1, y_2
A+1	— $x_1=0$
A+2	y_3, y_5
A+3	— $x_2=1$
A+7	— $x_3=1$
A+9	y_6
A+10	— $x_{\bar{o}n} \equiv 1$
A+0	

Тест 2: $x_1 x_2 x_3 = 110$

A+0	y_1, y_2
A+1	— $x_1=1$
A+5	y_1, y_3
A+6	— $x_{\bar{o}n} \equiv 1$
A+2	y_3, y_5
A+3	— $x_2=1$
A+7	— $x_3=0$
A+8	y_4
A+9	y_6
A+10	— $x_{\bar{o}n} \equiv 1$
A+0	

Сопоставление последовательностей вырабатываемых МП управляющих сигналов с соответствующими тестовыми последовательностями позволяет сделать вывод о логически правильной МП (если эти последовательности совпадают для каждого теста). В случае какого-либо несовпадения следует проанализировать соответствующую тестовую ситуацию и внести коррективы в МП.

В рассматриваемом примере произошло полное совпадение последовательностей, предусмотренных тестами, с последовательностями управляющих сигналов, вырабатываемыми МП. Это обстоятельство позволяет перейти к реализации пункта д), т.е. кодированию операционной части МК.

Примечание. – Сокращение количества адресов в МП возможно, если применить подход, проиллюстрированный на рисунке 4, б: в вершинах с

номера 2 и 5 можно проверять не прямые, а инверсные значения осведомительных сигналов x_1 и x_2 соответственно. Однако, в некоторых случаях этот подход может увеличить количество проверяемых осведомительных сигналов (если в одной вершине ГСА будет проверяться прямое значение осведомительного сигнала, а в другой вершине – инверсное значение того же самого осведомительного сигнала). Далее рассматривается МП для случая, рассмотренного на рисунке 4, а (таблица 2).

При использовании горизонтального принципа кодирования МО составляется таблица МК, правая часть которой представляет двоичные коды МК (таблица 3).

Таблица 3 – Горизонтальное кодирование МО в МК

МК в условных обозначениях	Двоичный код МК					
	Разряды поля М					
	1	2	3	4	5	6
y_1, y_2	1	1	0	0	0	0
y_1, y_3	1	0	1	0	0	0
y_3, y_5	0	0	1	0	1	0
y_4	0	0	0	1	0	0
y_2, y_6	0	1	0	0	0	1

Вертикальный принцип кодирования МО предполагает, что каждая МК кодируется кодом минимальной длины, а для расшифровки используемых кодов необходимо разработать комбинационную схему, реализующую систему булевых функций. Для рассматриваемого примера можно предложить следующее кодирование (таблица 4). Принцип соседнего кодирования используется для упрощения булевых функций, определяющих конкретные МО.

Таблица 4 – Вертикальное кодирование МО в МК

МК в условных обозначениях	Двоичный код МК		
	Разряды поля М		
	1	2	3
y_1, y_2	1	1	1
y_1, y_3	1	1	0
y_3, y_5	1	0	0
y_4	1	0	1
y_2, y_6	0	1	1

Система булевых функций:

$$\begin{aligned}
 y_1 &= M_1 M_2 & y_4 &= M_1 \bar{M}_2 M_3 \\
 y_2 &= M_2 M_3 & y_5 &= M_1 \bar{M}_2 \bar{M}_3 \\
 y_3 &= M_1 \bar{M}_3 & y_6 &= \bar{M}_1 M_2 M_3
 \end{aligned}$$

Для осуществления горизонтально-вертикального принципа кодирования МО необходимо разбить все множество МО на подмножества несовместимых МО. Для этого удобно построить матрицу несовместимости МО (рисунок 6). В этой матрице на пересечении строк и столбцов, соответствующих микрооперациям, которые упомянуты в одной микрокоманде (операторной вершине ГСА на рисунке 4, а) ставятся нули, затем остальные клетки заполняются единицами. Таким образом единичное значение элемента матрицы задает отношение несовместимости соответствующих МО. Матрица симметрична относительно главной диагонали, поэтому можно ограничиться построением только выделенной на рисунке 6 части матрицы.

	y_1	y_2	y_3	y_4	y_5	y_6
y_1	-	0	0	1	1	1
y_2	0	-	1	1	1	0
y_3	0	1	-	1	0	1
y_4	1	1	1	-	1	1
y_5	1	1	0	1	-	1
y_6	1	0	1	1	1	-
	y_1	y_2	y_3	y_4	y_5	y_6

Рисунок 6 – Матрица несовместимости МО

Составляются подмножества МО, в каждое из которых входят только несовместимые между собой МО. Учитывая, что расшифровывать коды будут дешифраторы, мощность каждого подмножества должна стремиться к величине $2^k - 1$, где $k=1, 2, \dots$. Для получения подмножеств можно использовать метод прямого включения микроопераций. Ниже этот метод проиллюстрирован на рассматриваемом примере.

Шаг 1. Открываем первое множество. Добавляем в него микрооперацию y_1 .

Шаг 2. Пытаемся добавить в первое подмножество y_2 . Это невозможно, так как y_2 совместима с y_1 (эта информация получена из матрицы несовместимости). Открываем второе подмножество. Добавляем в него y_2 .

Шаг 3. Пытаемся добавить y_3 в первое подмножество. Это невозможно, так как y_3 совместима с y_1 . Добавляем y_3 во второе подмножество. Это возможно, так как y_3 несовместима с y_2 .

Шаг 4. Добавляем y_4 в первое подмножество (несовместима с y_1). Добавляем также y_4 во второе подмножество (несовместима с y_2, y_3).

Подмножества после четвертого шага выглядят следующим образом:

- 1) y_1, y_4 ;
- 2) y_2, y_3, y_4 .

Шаг 5. Добавляем y_5 в первое подмножество. Во второе подмножество y_5 добавить нельзя (совместима с y_3).

Подмножества после пятого шага выглядят следующим образом:

1) y_1, y_4, y_5 ;

2) y_2, y_3, y_4 .

Шаг 6. Добавляем y_6 в первое подмножество. Во второе подмножество y_6 добавить нельзя (совместима с y_2).

Подмножества после шестого шага выглядят следующим образом:

1) y_1, y_4, y_5, y_6 ;

2) y_2, y_3, y_4 .

Шаг 7. Убираем y_4 из первого подмножества. Окончательный результат:

1) y_1, y_5, y_6 ;

2) y_2, y_3, y_4 .

Отметим, что мощность полученных подмножеств равна $3 = 2^2 - 1$. Каждое подмножество будет представлено двумя двоичными разрядами. Комбинация 00 в этих разрядах будет означать, что в МК не присутствуют МО из данного подмножества, остальные три комбинации будут сопоставлены единичным значениям соответствующих управляющих сигналов, входящих в данное подмножество.

Отметим, также, что метод прямого включения является приближенным и в общем случае дает не единственное решение. Результат разбиения на подмножества в более сложных, чем наш пример, случаях зависит от порядка рассмотрения управляющих сигналов. Из нескольких вариантов разбиения на подмножества разработчик может выбрать лучший по критерию минимальной длины операционной части МК или по критерию упрощения структуры ФСМО.

МО, включенную в несколько подмножеств, необходимо оставить только в одном из них. На решение разработчика в этом случае могут повлиять следующие соображения:

1) возможность сократить длину операционной части МК (по этому критерию в нашем примере исключена из первого подмножества и оставлена во втором подмножестве y_4 , что сократило число двоичных разрядов, необходимых для кодирования первого подмножества и не увеличило число разрядов, кодирующих второе подмножество),

2) возможность упростить структуру ФСМО (когда присутствие некоторой МО в одном подмножестве не увеличивает число двоичных разрядов, кодирующих данное подмножество, а исключение той же МО из другого подмножества приводит к горизонтальному кодированию в этом подмножестве – число разрядов, кодирующих подмножество, равно числу микроопераций в подмножестве);

3) при исключении некоторой МО из подмножества (при условии, что она останется в каком-либо другом подмножестве), в данное подмножество можно добавить другую МО, совместимую с исключенной, но не совместимую с оставшимися в подмножестве МО.

Кодирование МО в полученных подмножествах показано в таблице 5. Двоичное представление операционных МК приведено в таблице 6. Таким образом, по сравнению с горизонтальным кодированием удалось сократить длину операционной части МК на 2 разряда.

Таблица 5 – Кодирование МО в подмножествах

Номер подмножества	МО	Двоичный код
M_1	y_1	01
	y_5	10
	y_6	11
M_2	y_2	01
	y_3	10
	y_4	11

Таблица 6 – Горизонтально-вертикальное кодирование МО в МК

МК в условных обозначениях	Двоичный код МК	
	M_1	M_2
y_1, y_2	0 1	0 1
y_1, y_3	0 1	1 0
y_3, y_5	1 0	1 0
y_4	0 0	1 1
y_2, y_6	1 1	0 1

При использовании вертикально-горизонтального принципа кодирования МО необходимо выделить подмножества совместимых МО и обеспечить при этом несовместимость сформированных подмножеств (ни одна МО одного подмножества не должна быть совместима с какой-либо МО любого другого подмножества). Строим граф совместимости МО (рисунок 7, а). На этом графе МО, которые встречаются в одной МК, соединяются ребром.



Рисунок 7 – Граф совместимости МО:

а) исходный; б) после удаления y_1 и y_3 в универсальную группу

Для формирования несовместимых подмножеств с приблизительно одинаковой мощностью выделяют в универсальную группу МО, имеющие наибольшее количество связей с другими МО. В примере этими МО могут быть y_1 и y_3 . Если обе эти МО выделить в универсальную группу (поле M_1), то на графе совместимости останутся четыре МО (рисунок 7, б), которые можно разбить на два подмножества: в первое подмножество включаем y_2, y_6 , во второе подмножество – y_4 и y_5 . Подмножества будут фиксироваться в поле M_3 . Способ кодирования МО в полях M_1 и M_3 – горизонтальный. Идентифицирующее поле M_2 будет содержать двоичный позиционный код

номера подмножества. Для полученного разбиения это поле будет одноразрядным. Кодирование для этого случая приведено в таблице 7.

Отметим, что в результате такого кодирования МО удалось сократить длину операционной части МК по сравнению с горизонтальным кодированием на один разряд. В более сложных микропрограммах с большим количеством микроопераций можно достичь более существенного сокращения.

Таблица 7 – Вертикально-горизонтальное кодирование МО в МК

МК в условных обозначениях	Двоичный код МК		
	M_1	M_2	M_3
y_1, y_2	1 0	0	1 0
y_1, y_3	1 1	0	0 0
y_3, y_5	0 1	1	0 1
y_4	0 0	1	1 0
y_2, y_6	0 0	0	1 1

Таблица 8 – Альтернативный вариант вертикально-горизонтального кодирования МО в МК

МК в условных обозначениях	Двоичный код МК		
	M_1	M_2	M_3
y_1, y_2	1 1 0	0 0	0
y_1, y_3	1 0 1	0 0	0
y_3, y_5	0 0 1	0 1	1
y_4	0 0 0	0 0	1
y_2, y_6	0 1 0	1 0	1

Как и в случае горизонтально-вертикального кодирования при вертикально-горизонтальном кодировании можно получить несколько альтернатив разбиения на подмножества и оценить эти альтернативы. Например, если в универсальную группу выделить три наиболее связанные с другими МО: y_1, y_2 и y_3 , то из оставшихся МО можно сформировать три подмножества: 1) y_4 , 2) y_5 , 3) y_6 . Вариант кодирования для такого разбиения представлен в таблице 8. Эта альтернатива заведомо хуже предыдущей, так как не дает сокращения длины операционной части по сравнению с горизонтальным кодированием, но существенно усложняет ФСМО. Поэтому выбираем первую альтернативу (таблица 7).

По разработанным таблицам кодирования МК можно представить форматы операционных МК для любого принципа кодирования МО (рисунок 8).

Формат управляющей микрокоманды **X.A** определяется количеством проверяемых логических условий (осведомительных сигналов) и количеством микрокоманд (адресов) в микропрограмме. Длина поля **X** равна $\lceil \log_2 |X| \rceil$, где $X = \{x_1, x_2, \dots, x_n, x_{\bar{0}n}\}$. Для рассматриваемого примера $X = \{x_1, x_2, x_3, x_{\bar{0}n}\}$. Соответственно, мощность множества осведомительных сигналов $|X| = 4$, длина поля **X** равна двум двоичным разрядам.

Длина адресного поля **A** вычисляется по аналогичной формуле $\lceil \log_2 |MK| \rceil$, где $|MK|$ – общее количество МК в МП. Эта схема вычислений справедлива при условии, что МП в памяти располагается с нулевого адреса. Для рассматриваемого примера формат управляющей МК изображен на рисунке 9.

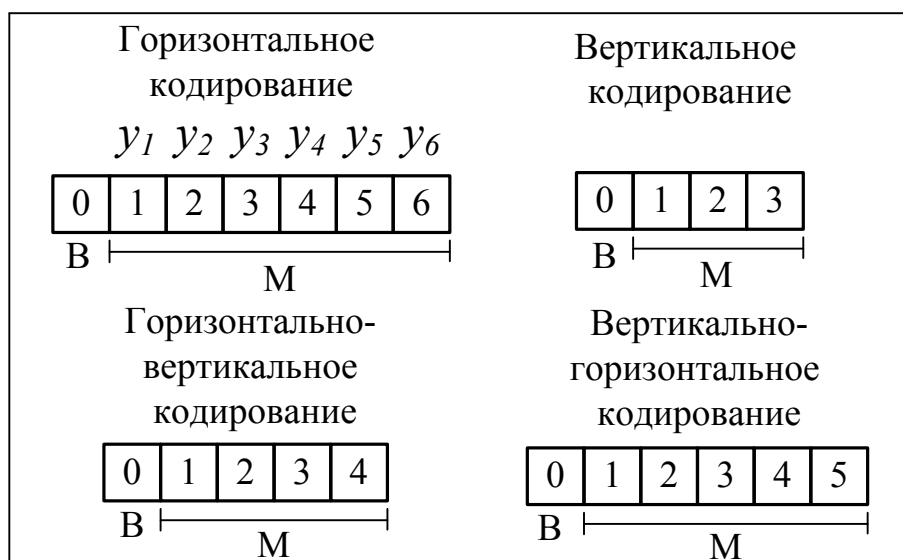


Рисунок 8 – Форматы операционных МК при различных способах кодирования МО

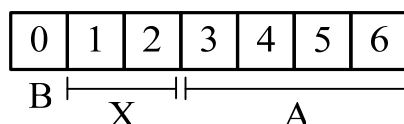


Рисунок 9 – Формат управляющей МК

Каждый $x_i \in \{X\}$ кодируется одним из множества двухразрядных кодов $\{00, 01, 10, 11\}$.

Замечание. Длины форматов для операционных и управляющих МК должны быть одинаковыми. Поэтому более короткий формат расширяется за счет добавления необходимого количества неиспользуемых разрядов (эти разряды, как правило, доопределяются нулями).

Для рассматриваемого примера в случае горизонтального кодирования МО длины операционных и управляющих микрокоманд совпадают. Но при использовании вертикального, горизонтально-вертикального и вертикально-горизонтального способа кодирования МО длины форматов операционных МК следует увеличить соответственно на 3, 2 и 1 разряд. Можно сделать вывод о том, что для рассмотренной простой МП самый эффективный способ кодирования – горизонтальный (обеспечивает самую простую структуру ФСМО; сокращать длину операционной МК нет смысла, т.к. длина управляющей МК в битах равна общему числу МО). Однако, в более сложных МП с большим количеством МО и МК такая ситуация, как правило, не возникает, и длину операционной МК следует сокращать путем применения описанных выше способов кодирования МО.

Итогом разработки форматов МК, например, для определенного вариантом задания на лабораторную работу горизонтально-вертикального способа кодирования МО, будут форматы, изображенные на рисунке 10.

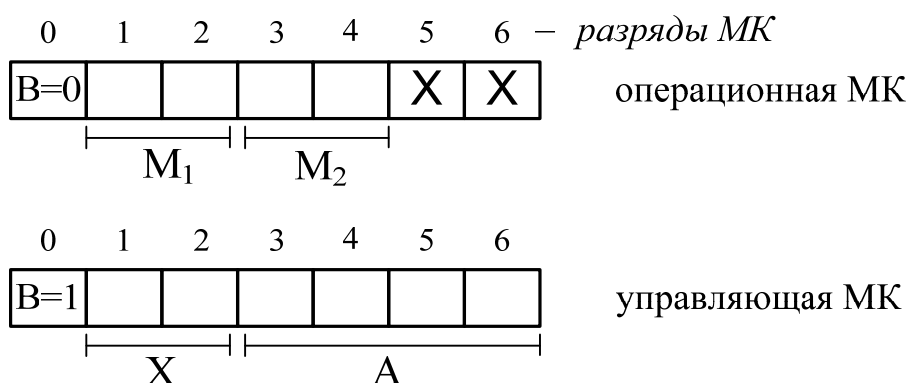


Рисунок 10 – Итоговые форматы МК при использовании горизонтально – вертикального способа кодирования МО

Таблица 9 – Двоичный код МП для УА с естественной адресацией

Адрес МК	Двоичный код МК						
	0	1	2	3	4	5	6
0000	0	0	1	0	1	X	X
0001	1	0	1	0	1	0	1
0010	0	1	0	1	0	X	X
0011	1	1	0	0	1	1	1
0100	1	0	0	0	0	0	1
0101	0	0	1	1	0	X	X
0110	1	0	0	0	0	1	0
0111	1	1	1	1	0	0	1
1000	0	0	0	1	1	X	X
1001	0	1	1	0	1	X	X
1010	1	0	0	0	0	0	0

На основании разработанных форматов МК, таблиц кодирования операционной части МК и МП в условных обозначениях выполняется окончательное представление МП в двоичном виде.

Двоичный код МП для рассматриваемого примера в случае горизонтально-вертикального способа кодирования МО приведен в таблице 9.

4.4. Особенности выполнения работы

Работа выполняется на персональном компьютере с использованием программного эмулятора УА с программируемой логикой. Внешний интерфейс эмулятора и правила работы с ним приведены в приложении.

Замечания.

Результатом работы над заданием должны быть две таблицы, представляющие двоичные коды микропрограмм для двух заданных вариантов способов кодирования МО. Каждая таблица должна сопровождаться структурной схемой ФСМО. Студент также должен привести структурную схему УА с естественной адресацией для своего варианта задания. Указанные схемы для рассмотренного примера МП (таблица 9) приведены на рисунках 11, 12.

синхросерии $C1$ рассчитываются исходя из логики работы схемы и характеристик составляющих ее элементов.

Отметим, что если заданием предусмотрено вертикальное кодирование, то ФСМО можно построить двумя способами. Первый способ состоит в реализации системы булевых функций, полученных по таблице 4, на логических элементах заданного базиса. Второй способ основан на тривиальном кодировании микрокоманд (таблица 10). Соответствующий ФСМО приведен на рисунке 13.

Таблица 10 – Тривиальное кодирование операционных МК

МК	МО, входящие в МК	Двоичный код МК		
		Разряды поля М		
		1	2	3
Y_1	y_1, y_2	0	0	0
Y_2	y_1, y_3	0	0	1
Y_3	y_3, y_5	0	1	0
Y_4	y_4	0	1	1
Y_5	y_2, y_6	1	0	0

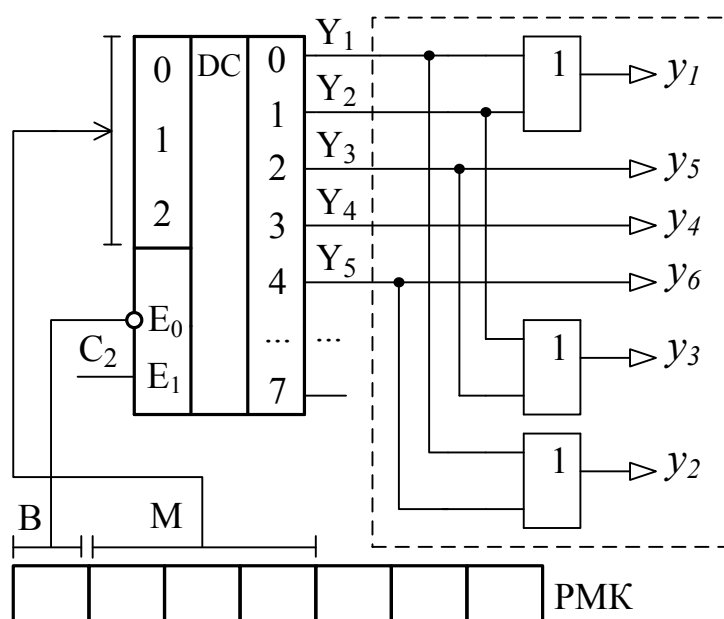


Рисунок 13 – Одна из возможных структур ФСМО при вертикальном кодировании МО (реализована в эмуляторе)

Примечание. – Необходимо учитывать, что в эмуляторе УА с программируемой логикой, используемом при выполнении лабораторных работ, часть ФСМО, обведенная на рисунке 13 пунктиром, не моделируется, т.е. работа формирователя при вертикальном кодировании будет ограничиваться только расшифровкой микрокоманд.

4.5. Контрольные вопросы

4.5.1. Определите понятие «способ адресации» МК.

4.5.2. Как осуществляется расстановка адресов на ГСА функционирования УА с естественной адресацией МК

4.5.3. Поясните необходимость двух типов форматов МК при использовании естественного способа адресации МК.

4.5.4. Охарактеризуйте структуру УА структуру УА с программируемой логикой, ориентированной на использование естественной адресации МК.

4.5.5. Почему устройство, задающее адрес МК в памяти МП, называется счетчиком?

4.5.6. Определите положительные и отрицательные стороны заданных вариантов задания способов кодирования МО.

4.5.7. Какие режимы работы программного эмулятора предусмотрены?

5. ЛАБОРАТОРНАЯ РАБОТА № 2. РАЗРАБОТКА МП ДЛЯ УА, ИСПОЛЬЗУЮЩЕГО ПРИНЦИП ПРИНУДИТЕЛЬНОЙ АДРЕСАЦИИ МК

5.1. Варианты заданий

Варианты заданий приведены в таблице 11. Обозначенные в таблице 11 ГСА функционирования УА приведены на рисунке 3.

Микропрограммы для УА должны быть разработаны с использованием принципа принудительной адресации МК.

Таблица 11 – Варианты заданий

Номер варианта	Способ кодирования МО	Номер ГСА
1	В, В-Г	1
2	В, Г-В	2
3	Г, В-Г	3
4	Г, Г-В	4
5	Г, В	5
6	Г, Г-В	1
7	Г, В-Г	2
8	В, Г-В	3
9	В, В-Г	4
10	В-Г, Г-В	5

5.2. Теоретическая подготовка

При подготовке к лабораторной работе № 2 необходимо в первую очередь обратить внимание на особенности используемого принципа адресации МК [3, стр. 293 – 329], а именно, на процедуру формирования единого формата МК, на способ расстановки адресов МК в ГСА функционирования УА и на преобразование исходной ГСА, если в этом есть необходимость.

Принцип принудительной адресации МК предполагает единый формат МК типа **М.Х.А**. Причем исполнительный адрес следующей МК определяется в предыдущей МК по правилу

$$A_{исп} = \begin{cases} A.1, \text{ если } X = i \ (x_i = 1), \\ A.0, \text{ в противном случае.} \end{cases}$$

5.3. Пример разработки МП

Формат МК **М.Х.А** предполагает, что каждой МК должен соответствовать участок ГСА функционирования УА, изображенный на рисунке 14,а. Эта микрокоманда инициирует в ОА микрооперации, заданные в операционной части (поле **М**). В процессе выполнения этих МО может измениться значение осведомительного сигнала x_i , номер которого i задан в поле **Х** МК. По окончании выполнения микроопераций и выработки осведомительных сигналов в ОА, рассматриваемая МК инициирует переход к МК, расположенной в ПМК по четному адресу **А.0**, если $x_i = 0$. Если $x_i = 1$, то переход осуществляется к микрокоманде, расположенной по нечетному адресу **А.1**.

Если фрагмент ГСА не содержит операторной вершины между двумя условными (рисунок 14,б), то МК, соответствующая второй условной вершине трактуется как МК с пустой (нулевой) операционной частью (поле **М**). Такая МК осуществляет переход к следующей МК без инициирования микроопераций в операционном автомате (правила перехода такие же как и в описанном выше случае).

Если участок ГСА не содержит условной вершины между двумя операторными (рисунок 14, в, г), то МК, соответствующая первой операторной вершине, в условной части (**Х**) должна содержать номер осведомительного сигнала, тождественно равного 0 (на рисунке 14,в $x_0 \equiv 0$), если микрокоманде, соответствующей второй операторной вершине, ранее был сопоставлен четный адрес. Аналогично, МК, соответствующая первой операторной вершине, в условной части (**Х**) должна содержать номер осведомительного сигнала, тождественно равного 1 (на рисунке 14, г $x_n \equiv 1$), если МК, соответствующей второй операторной вершине, ранее был сопоставлен нечетный адрес. На рисунках 14, б, в, г изображены исходные и преобразованные путем добавления недостающей части микрокоманды фрагменты ГСА (такое преобразование лучше делать в уме ;-).

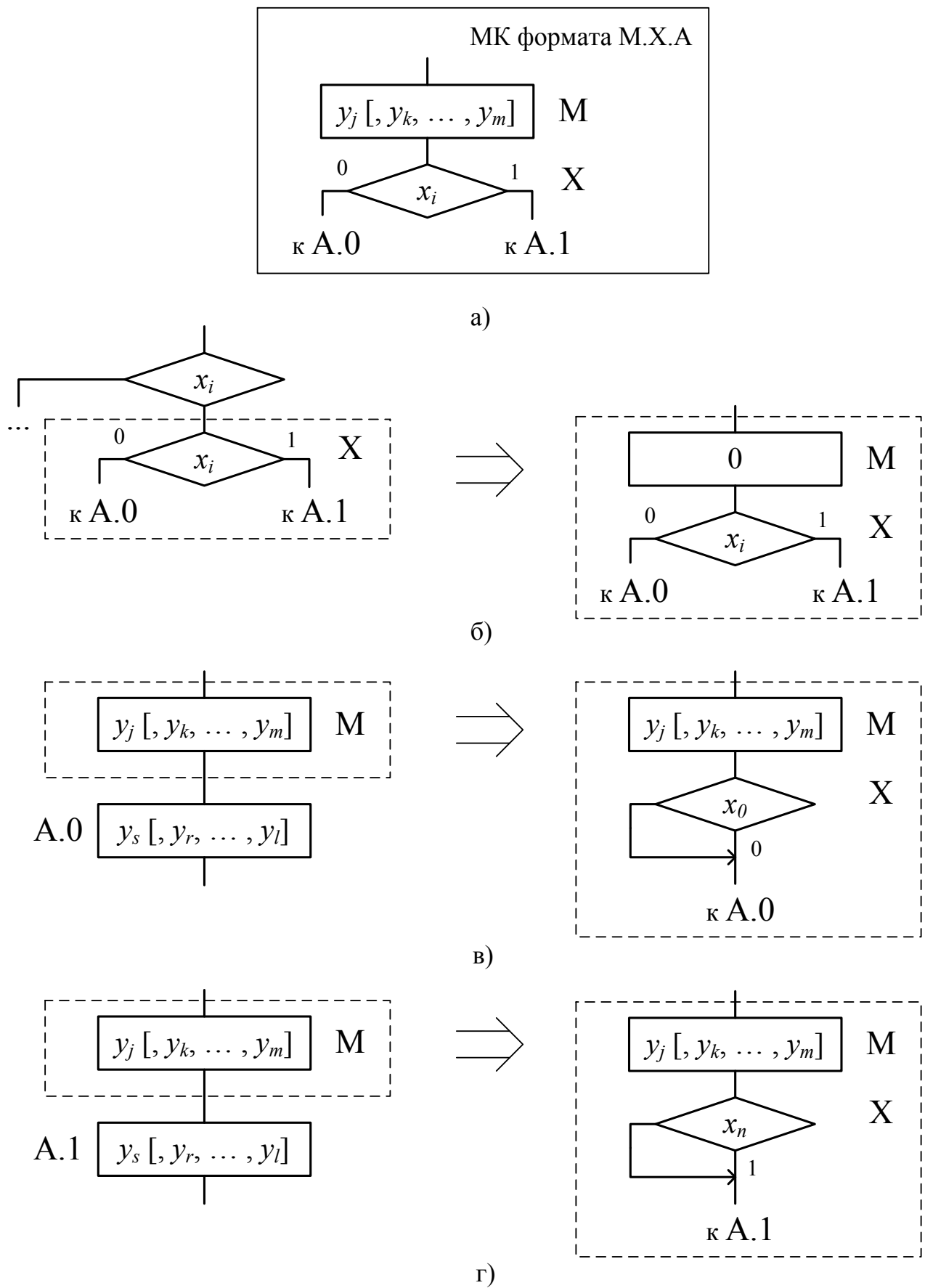


Рисунок 14 – Фрагменты ГСА, соответствующие МК, и их преобразование

Как и в предыдущей лабораторной работе для примера рассмотрим ГСА функционирования УА, приведенную на рисунке 4. В связи с этим этапы «определение минимального множества тестов» и «кодирование операционной части МК» общей процедуры построения МП выполнять не будем, а воспользуемся результатами этих этапов, полученных при выполнении лабораторной работы № 1.

На рисунке 15 приведены исходная и преобразованная ГСА, а также показана последовательность расстановки адресов. В таблице 12 приведена соответствующая микропрограмма в условных обозначениях.

Расстановка адресов начинается с базового адреса $A0.0$, который сопоставляется группе вершин, состоящей из операторной вершины номер 1 и условной вершины номер 2 на рисунке 15, а. По адресу $A0.0$ в ПМК (первая строка таблицы 12) будет находиться МК, иницилирующая выполнение микроопераций y_1 и y_2 и проверяющая после их выполнения значение осведомительного сигнала x_1 . В поле **A** этой МК следует записать следующий свободный адрес $A1$ (в данном случае важно, чтобы свободными, т.е. не сопоставленными вершинам ГСА, были обе позиции адреса – и $A1.0$ и $A1.1$). Если $x_1 = 0$, то следующей должна выполняться МК, находящаяся по адресу $A1.0$. Поэтому адресом $A1.0$ отмечена вершина номер 4. Если $x_1 = 1$, то следующей должна выполняться МК, расположенная по адресу $A1.1$. Поэтому адресом $A1.1$ отмечена вершина номер 3.

Аналогичные рассуждения применимы к МК, расположенной в ПМК по адресу $A1.0$ (включает в себя операторную вершину номер 4 и условную вершину номер 5). МК, находящиеся в ПМК по адресам $A0.0$ и $A1.0$ соответствуют случаю, изображенному на рисунке 14, а.

Таблица 12 – МП в условных обозначениях для ГСА, изображенной на рисунке 7

Адрес МК	МК			Примечания
	М	Х	А	
$A0.0$	y_1, y_2	x_1	$A1$	
$A0.1$	—	—	—	нет МК
$A1.0$	y_3, y_5	x_2	$A2$	
$A1.1$	y_1, y_3	x_0	$A1$	$x_0 \equiv 0$
$A2.0$	—	x_1	$A1$	
$A2.1$	—	x_3	$A3$	
$A3.0$	y_4	x_4	$A3$	$x_4 \equiv 1$
$A3.1$	y_2, y_6	x_0	$A0$	

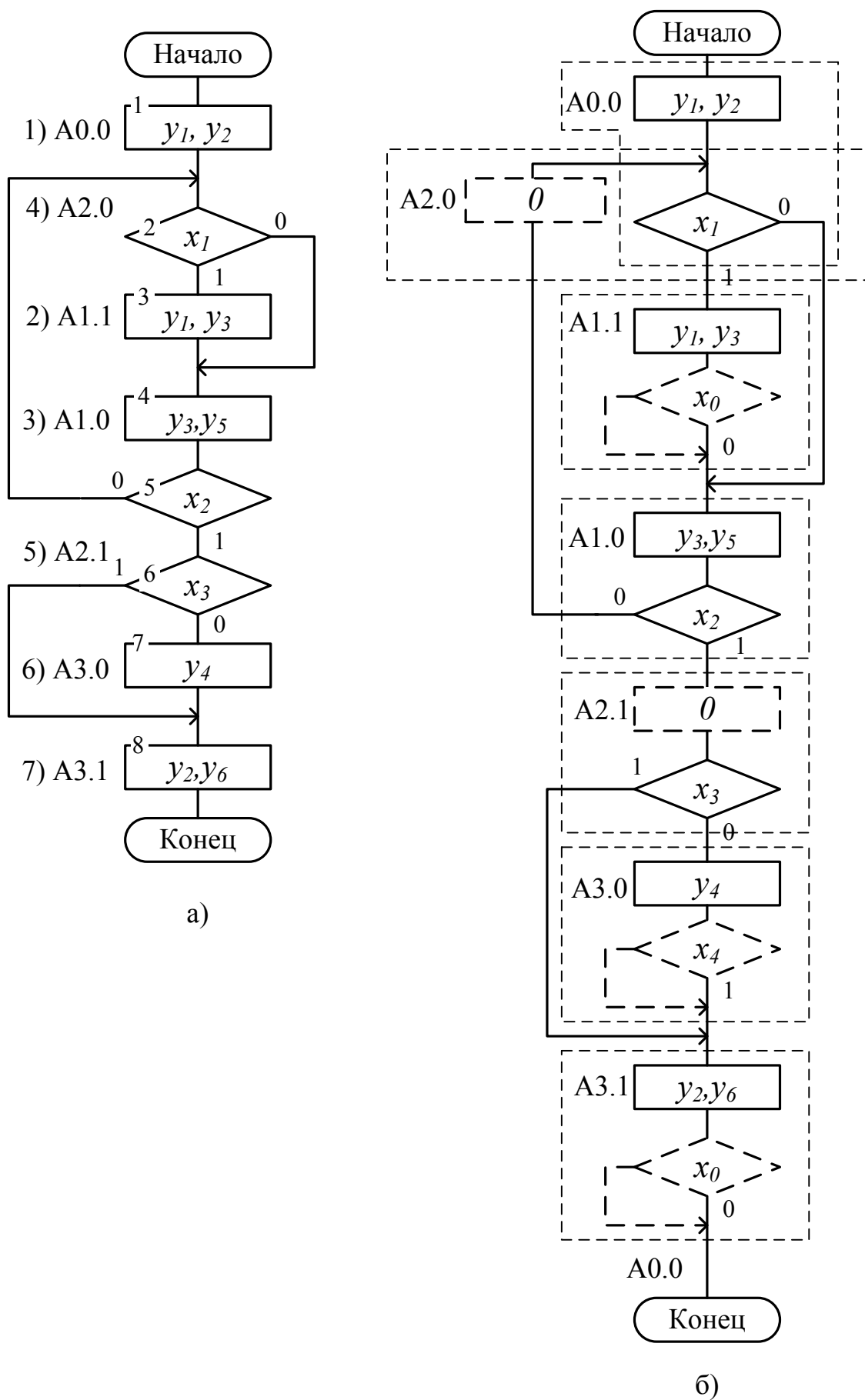


Рисунок 15 – Расстановка адресов на ГСА для случая принудительной адресации МК: а) исходная ГСА; б) преобразованная ГСА

Микрокоманда, которой сопоставлен адрес A1.1 соответствует случаю, изображенному на рисунке 14, в. Так как вершине номер 4 ранее был сопоставлен адрес A1.0, в обсуждаемой МК используется осведомительный сигнал $x_0 \equiv 0$.

Микрокоманда, которой сопоставлен адрес A2.0, соответствует случаю, изображенному на рисунке 14, б. У этой МК пустая операционная часть (поле М), условная часть представлена вершиной номер 2. Тому же случаю соответствует МК, которой сопоставлен адрес A2.1.

Микрокоманда, которой сопоставлен адрес A3.0 соответствует случаю, изображенному на рисунке 14, г. Так как вершине номер восемь ранее был сопоставлен адрес A3.1, в обсуждаемой МК используется осведомительный сигнал $x_4 \equiv 1$.

Микрокоманда, расположенная в ПМК по адресу A3.1, соответствует случаю, изображенному на рисунке 14, в. После выполнения МО y_2 , y_6 данная МК осуществляет возврат к МК, расположенной по адресу A0.0.

Как видно из приведенных выше рассуждений адрес A0.1 в рассмотренной МП не используется.

Примечание 1. – В микропрограмме, приведенной в таблице 12, используется 5 осведомительных сигналов. Следовательно, для кодирования номера осведомительного сигнала в поле X МК, потребуется 3 разряда. Это число разрядов можно сократить, если вместо значения осведомительного сигнала x_3 проверять его инверсию. Измененная ГСА изображена на рисунке 16. Анализ этой ГСА позволяет сделать заключение, что осведомительный сигнал $x_4 \equiv 1$ при составлении соответствующей МП не потребуется (объясните почему). Эту МП студенту предлагается построить самостоятельно.

Примечание 2. – Интересным является случай ГСА, изображенный на рисунке 17. По адресу A3.0 в МП, соответствующей этой ГСА, может стоять команда безусловного перехода на адрес A2.1 (— , x_4 , A2). Более предпочтительной является стратегия дублирования по адресу A3.0 микрокоманды, находящейся по адресу A2.1. Тогда в словах ПМП с адресами A2.1 и A3.0 будет находиться одна и та же информация: y_3 , x_4 , A0 ($x_4 \equiv 1$). Стратегия дублирования экономит такт, который потребовался бы для выполнения команды безусловного перехода, если бы применялась первая стратегия. Отметим также, что в рассматриваемой ГСА (рисунок 17) в отличие от ГСА, изображенной на рисунке 15, используется адрес A0.1. МП по этой ГСА студенту предлагается построить самостоятельно.

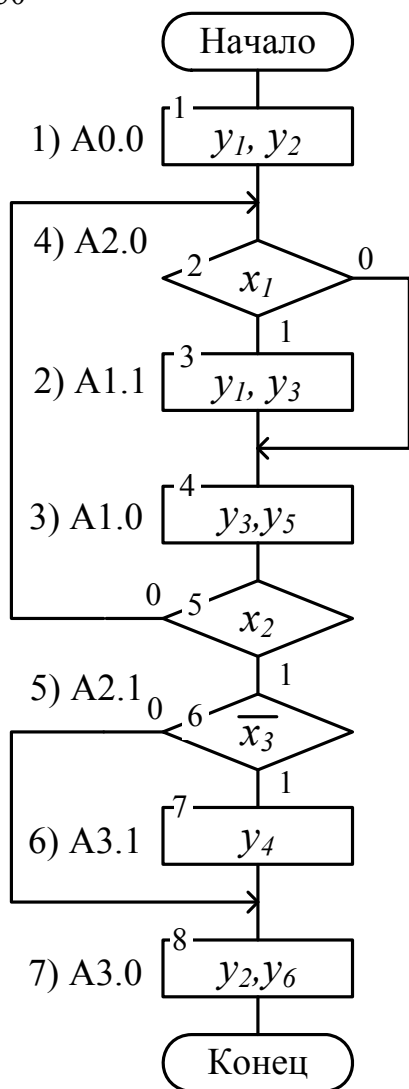


Рисунок 16 – ГСА с проверкой инверсного значения x_3

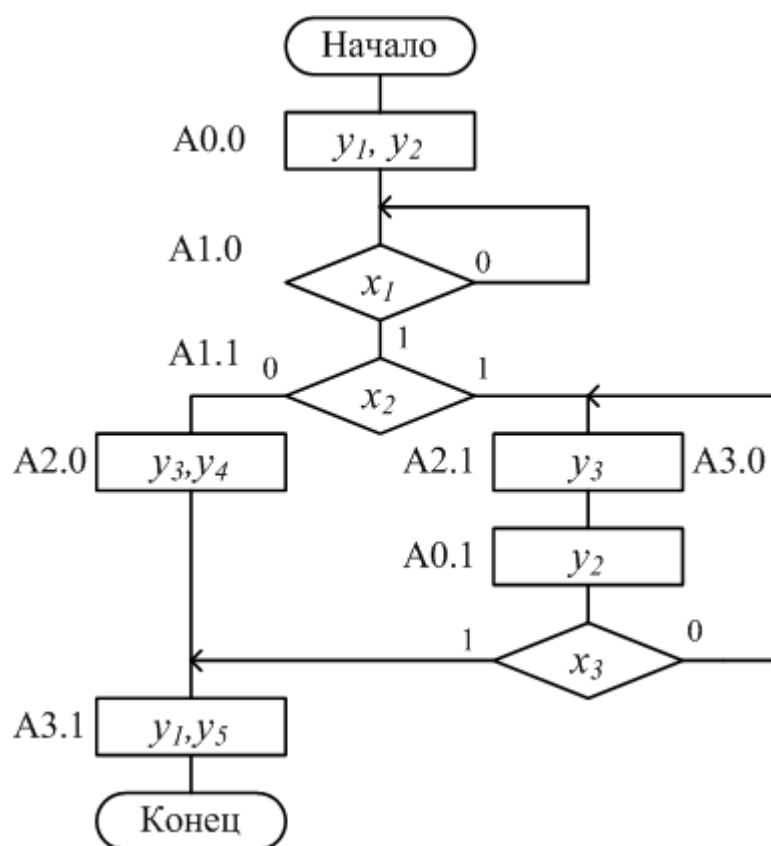


Рисунок 17 – ГСА с возможностью дублирования МК

Тестирование МП, приведенной в таблице 12, проводится на тестовых наборах, полученных для исходной ГСА (рисунок 5).

Тест 1: $x_1 x_2 x_3 = 10X$

A0.0	y_1, y_2	$x_1 = 1$
A1.1	y_1, y_3	$x_0 \equiv 0$
A1.0	y_3, y_5	$x_2 = 0$
A2.0	—	$x_1 = 1$
A1.1		

Тест 2: $x_1 x_2 x_3 = 110$

A0.0	y_1, y_2	$x_1 = 1$
A1.1	y_1, y_3	$x_0 \equiv 0$
A1.0	y_3, y_5	$x_2 = 1$
A2.1	—	$x_3 = 0$
A3.0	y_4	$x_4 \equiv 1$
A3.1	y_2, y_6	$x_0 \equiv 0$
A0.0		

Тест 3: $x_1 x_2 x_3 = 011$

A0.0	y_1, y_2	$x_1 = 0$
A1.0	y_3, y_5	$x_2 = 1$
A2.1	—	$x_3 = 1$
A3.1	y_2, y_6	$x_0 \equiv 0$
A0.0		

Тестирование прошло успешно, т.к. вырабатываемые МП последовательности управляющих сигналов полностью совпадают с необходимыми тестовыми последовательностями (рисунок 5). Сравните по времени выполнения тесты для УА с естественной адресацией и для УА с принудительной адресацией.

Для проработки формата МК необходимо в соответствии с заданием определиться со способом кодирования МО. Способы кодирования операционной части МК и их реализация для рассматриваемого примера обсуждались в разделе 4.3. Рассмотрим случай вертикально-горизонтального кодирования. Поле **М** в МК представляется пятиразрядным кодом **М(1:5)**: $M_1 = M(0:1)$ – двухразрядное поле для представления МО универсальной группы, $M_2 = M(2)$ – одноразрядное поле для представления кода номера группы (подмножества МО), $M(3:4)$ – двухразрядное поле для представления МО группы. Бит-маркер при принудительной адресации отсутствует, т.к. нет альтернативного формата МК.

Необходимо заметить, что поле **А** в МК содержит неполный (без младшего разряда) адрес следующей исполняемой МК. Младший разряд адреса вычисляется ФАМК и приписывается к старшим разрядам в процессе формирования исполнительного адреса $A_{\text{исп}}$.

На рисунке 18 приведен формат МК для рассматриваемого примера.

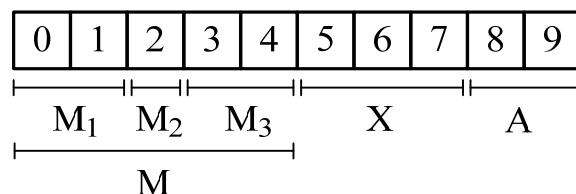


Рисунок 18 – Формат МК для рассматриваемого примера в случае принудительной адресации МК

Используя разработанный формат МК, составляем двоичный код МП (таблица 13). Применяем тривиальное кодирование для номера проверяемого логического условия и адреса следующей МК.

Таблица 13 – Двоичный код МК для УА с принудительной адресацией

Адрес МК	Двоичный код МК									
	М					Х			А	
	0	1	2	3	4	5	6	7	8	9
000	1	0	0	1	0	0	0	1	0	1
001	X	X	X	X	X	X	X	X	X	X
010	0	1	1	0	1	0	1	0	1	0
011	1	1	0	0	0	0	0	0	0	1
100	0	0	0	0	0	0	0	1	0	1
101	0	0	0	0	0	0	1	1	1	1
110	0	0	1	1	0	1	0	0	1	1
111	0	0	0	1	1	0	0	0	0	0

На рисунке 19 изображена структурная схема УА с принудительной адресацией для рассматриваемого примера. На рисунке 20 раскрыта структура ФСМО.

5.4. Особенности выполнения работы

Работа выполняется на персональном компьютере с использованием программного эмулятора УА с программируемой логикой. Внешний интерфейс эмулятора и правила работы с ним приведены в приложении.

5.5. Контрольные вопросы

5.5.1. В чем состоит отличительная особенность принудительной адресации по отношению к естественной?

5.5.2. Охарактеризуйте формат МК при использовании принудительного способа адресации.

5.5.3. Приведите фрагмент ГСА функционирования УА, соответствующий одной (заданной преподавателем) МК.

5.5.4. Объясните сущность определенного вариантом способа кодирования МО.

5.5.5. Какая информация используется при формировании адреса следующей исполняемой МК в случае принудительной адресации?

5.5.6. Какой способ кодирования МО в данном эмуляторе реализован не полностью (не позволяет расшифровывать МО)?

5.5.7. Сравните естественный и принудительный способы адресации по параметру «используемого информационного пространства» (объем ПМП).

5.5.8. Сравните естественный и принудительный способы адресации по параметру «число тактов исполнения одноименных тестовых последовательностей».

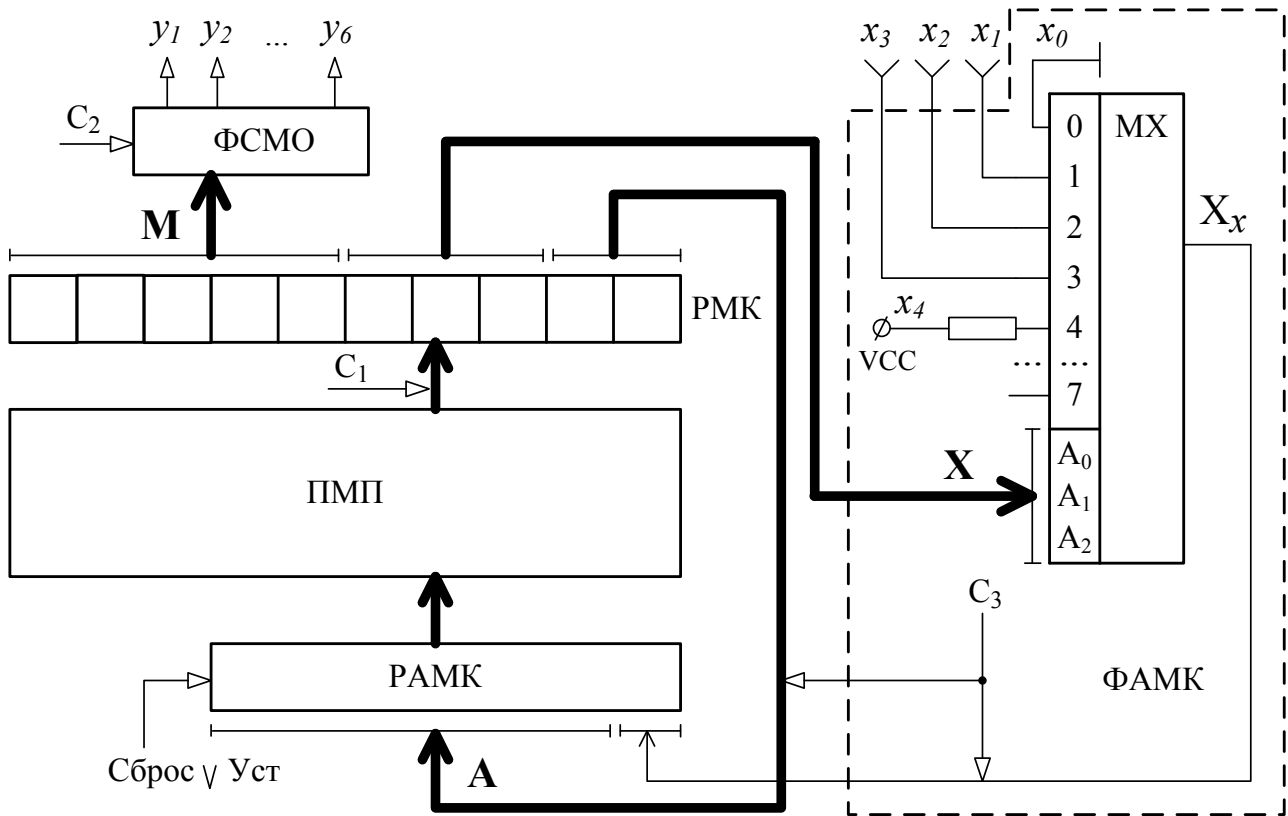


Рисунок 19 – Структура УА с принудительной адресацией для рассмотренного примера МП

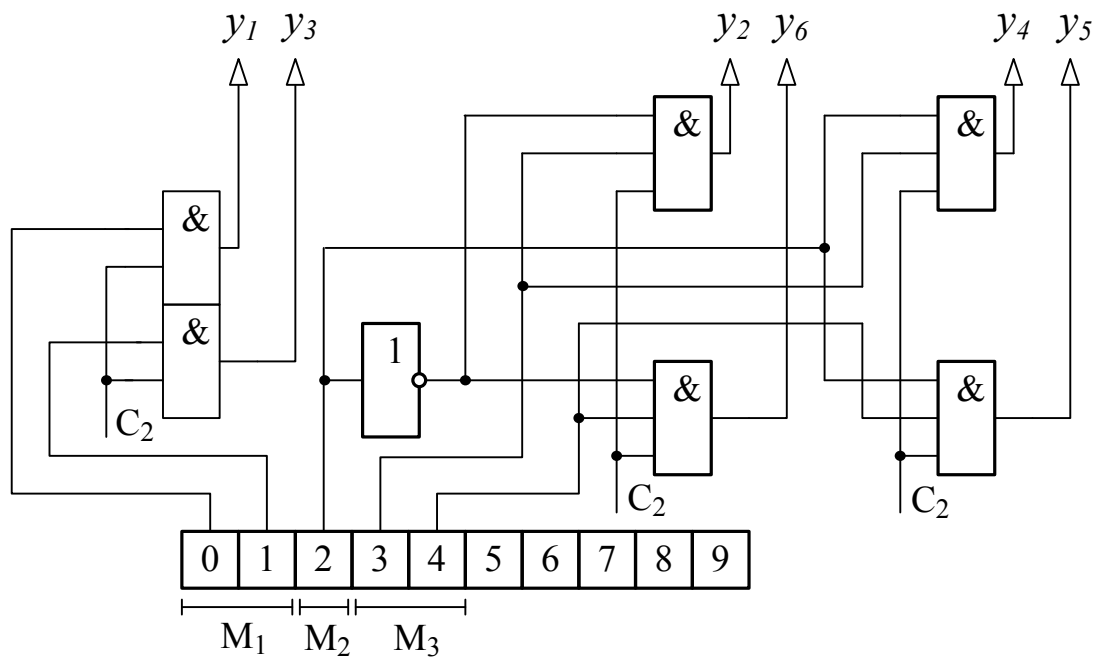


Рисунок 20 – Структура ΦСМО при вертикально-горизонтальном кодировании МО для рассмотренного примера МП

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Апраксин Ю.К. Основы теории и проектирования цифровых автоматов: учеб.пособие/ Ю.К. Апраксин. – Севастополь: Изд-во СевГТУ, 2001 – 345с.
2. Бабич Н.П. Компьютерная схемотехника. Методы построения и проектирования: учеб.пособие/ Н.П. Бабич, И.А. Жуков. – К.: МК–Пресс, 2004. – 576 с.
3. Цилькер Б.Я. Организация ЭВМ и систем/ Б.Я. Цилькер, С.А.Орлов. – СПб.: Питер, 2004. – 668 с.

ПРИЛОЖЕНИЕ А

Эмулятор управляющих автоматов с программируемой логикой. Руководство пользователя

1. Возможности эмулятора

Эмулятор управляющих автоматов с программируемой логикой, именуемый далее просто «эмулятор» представляет собой автономную программную систему, работающую под управлением ОС Windows 9X.

Эмулятор позволяет реализовать

2 способа адресации микрокоманд:

- принудительная адресация;
- естественная адресация;

4 стратегии кодирования операционной части микрокоманды:

- горизонтальное кодирование;
- вертикальное кодирование;
- кодирование несовместимыми подмножествами (вертикально-горизонтальное кодирование);
- кодирование отдельными полями (горизонтально-вертикальное кодирование).

Функциональные возможности эмулятора:

- 1) изменяемые длины полей микрокоманды;
- 2) сопоставление любому выходному сигналу Y и любому входному сигналу-условию X любой допустимой битовой комбинации;
- 3) ввод и редактирование микропрограммы;
- 4) трассировка микропрограммы;
- 5) изменение значений сигналов-условий X на каждом шагу трассировки;
- 6) непрерывное выполнение микропрограммы с выдачей протокола;
- 7) точки останова;
- 8) «визуальная трассировка», т.е. трассировка по структурной схеме УА;
- 9) сохранение формата микрокоманды и текста микропрограммы на внешнем носителе информации.

Ограничения:

- 1) в случае вертикального кодирования операционной части МК ФСМО декодирует микрокоманды, а не микрооперации;
- 2) максимальная длина операционной, условной и адресной части – 32 бита;
- 3) максимальное количество выходных сигналов Y и сигналов-условий X – 100;
- 4) максимальная длина микропрограммы – 65536 микрокоманд.

2. Работа с эмулятором

2.1. Запуск среды

Для запуска эмулятора достаточно выбрать команду «Выполнить» в меню кнопки «Пуск» и, указав путь и имя запускаемого файла EMULATE.EXE, нажать кнопку «ОК». При наличии на «рабочем столе» Windows ярлыка эмулятора воспользуйтесь им.

2.2. Полоса меню

После запуска эмулятор будет ожидать выбора вами команд меню. Вам будут доступны следующие меню: «Файлы» – команды «Создать», «Открыть» и «Выход»; «Настройка формата» – команда «Установка основных параметров»; «Справка» – команды «Помощь» и «О программе». Команды «Создать» и «Установка основных параметров» делают одно и то же: начинают процесс ввода формата микрокоманды. Команда «Открыть» считывает формат микрокоманды и микропрограмму из внешнего файла с расширением “EUA”, который был предварительно создан командой «Сохранить» или при выходе из эмулятора.

2.3. Ввод формата микрокоманды

Чтобы начать процесс ввода формата микрокоманды выберите команду меню «Файлы» «Создать». Перед вами откроется окно установки основных параметров микрокоманды (рисунок А.1).



Рисунок А.1 – Установка основных параметров

Здесь вы выбираете способ адресации микрокоманд, тип кодирования операционной части микрокоманды, а также длину каждой из трех частей микрокоманды – операционной, условной и адресной. На синхронно создаваемом внизу окна рисунке вы сможете видеть структуру создаваемой микрокоманды в виде групп прямоугольников. Каждый прямоугольник означает бит микрокоманды. Группы выделены цветами. Операционная часть имеет синий цвет, условная – красный и адресная – зелёный. Для естественной адресации первый бит микрокоманды определяет её вид и имеет чёрный цвет. Если число бит в какой-либо из групп больше 4, то количество бит отображается над этой группой.

Здесь и далее окна диалога имеют внизу 2 кнопки: «Дальше» и «Назад». В первом окне ввода формата вместо «Назад» стоит «Отмена» и при её нажатии процесс прекращается, а в последнем – вместо «Дальше» стоит «Готово» и при её нажатии процесс успешно завершается. При нажатии «Далее» программа выводит на экран следующее окно, при нажатии «Назад» – предыдущее.

В зависимости от выбранного типа кодирования операционной части появляется одно из четырёх окон, которые мы рассмотрим ниже.

Для **горизонтального кодирования** появляется окно, изображённое на рисунке А.2.

Номера Y	1	2	3	4	5	6	7	8	9	10	11
	1	2	3	4	5	6	10	50	75	90	99

Дальше

Назад

Рисунок А.2 – Горизонтальное кодирование

Здесь вы сможете для любого бита (номер бита в операционной части – на сером фоне) ввести соответствующий ему номер выходного сигнала Y, управляя с помощью клавиш курсора, клавиш ENTER, TAB или мыши. (Здесь и далее стандартные приёмы управления, такие, как переключение между элементами окна с помощью клавиши TAB, ввод с помощью ENTER, перемещение с помощью клавиш курсора не рассматриваются, а предпочтение уделено особым, нестандартным приёмам, характерным для этой системы.) Вы не сможете ввести для разных битов одинаковые номера Y, – сработает защита.

Для **вертикального кодирования** (рисунок А.3) вы с помощью «ползунка» выбираете количество Y, при этом эмулятор подставляет начальные значения битовых комбинаций, как двоичное представление номера – 1. После этого в таблице вы можете производить редактирование.

Для того, чтобы изменить номер Y в крайнем левом столбце таблицы – необходимо на нём мышью произвести double-click, т. е. «двойное нажатие левой кнопкой» (далее – double-click).

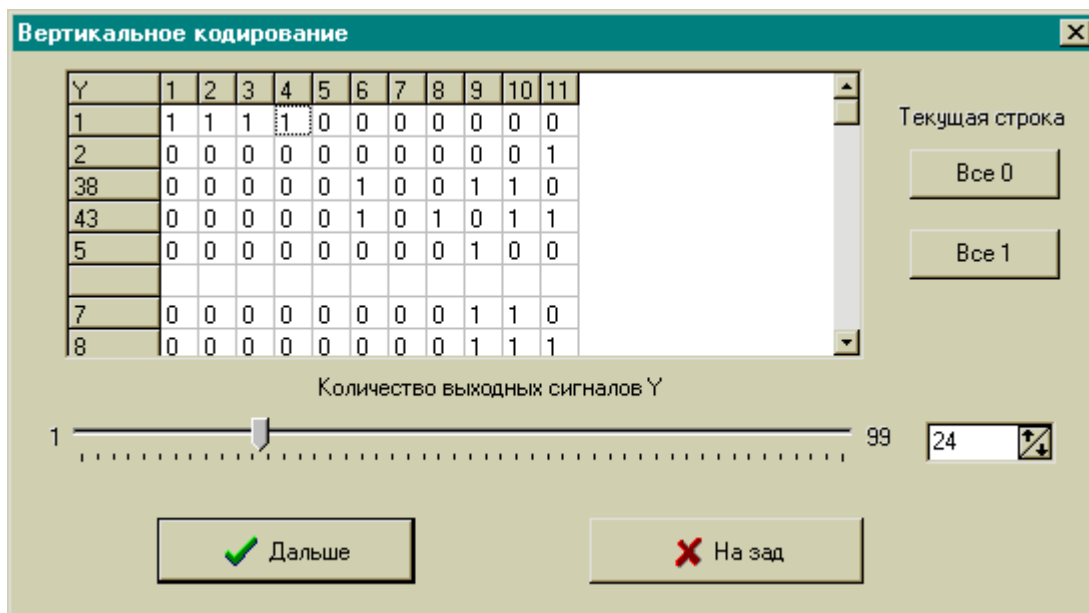


Рисунок А.3 – Вертикальное кодирование

Чтобы инвертировать бит в таблице, наведя на него курсор, нажмите «Пробел» или желаемую цифру «0» или «1», также можно воспользоваться мышью, произведя double-click над нужной ячейкой. Для заполнения текущей строки нулями или единицами – нажмите одну из кнопок справа. Чтобы убрать какую либо строку вообще – войдите в редактирование её номера и сотрите его.

При вводе номера сигнала срабатывает блок проверки: если такой номер уже существует, то старая строка стирается, также вы не сможете ввести в строку ввода ничего, кроме двух цифр, во избежание ошибок ввода.

По нажатию кнопки «Дальше» также выполняется проверка на повтор комбинаций. Если повторяющиеся комбинации найдены, то выдаётся соответствующее сообщение. Если оно будет проигнорировано, то возможна некорректная работа эмулятора.

Для **вертикально-горизонтального кодирования** (несовместимыми подмножествами МО) открывается окно, представленное на рисунке А.4.

Прежде всего, вы должны установить необходимые величины: количество подмножеств (рекомендуется начать с него), количество бит в универсальной группе (УГ), количество бит в подмножествах. Эмулятор автоматически определяет необходимое количество бит для кодировки номера подмножества и всегда регулирует размеры полей таким образом, что общая длина операционной части остаётся постоянной.

Далее необходимо ввести номера Y в УГ и подмножества (по аналогии с горизонтальным кодированием). Вы не сможете ввести одинаковый номер сигнала для двух бит в одной строке – это лишено смысла, и, поэтому, сработает защита. Однако, в разные строки возможно ввести одинаковые

номера. Но помните, что в классическом методе кодирования каждый сигнал должен быть представлен лишь один раз. Не ошибитесь!

Кодирование несовместимыми подмножествами

Длина универсальной группы 0 9 4

Количество подмножеств 1 32 4

Длина подмножеств 1 10 5

(Длина операционной части = 11)

Номера Y в универсальной группе

1	2	3	4
1	2	12	16

Номера Y в подмножествах

	7	8	9	10	11
00	3	4	5	6	7
01	9	10	11	14	15
10	8	18	17		
11	19	20			

Дальше Назад

Рисунок А.4 – Кодирование несовместимыми подмножествами

В правом нижнем углу окна будет отражена схема операционной части, так что вы всегда можете увидеть, какое место занимает каждое поле.

Для **горизонтально-вертикального кодирования** (раздельными полями) в открывшемся окне, изображенном на рисунке А.5, вначале необходимо установить *количество групп*. Это делается с помощью стрелок вверх и вниз, расположенных рядом с числом групп. После этого необходимо установить размеры каждой группы. Для этого обратите внимание на панель в верхней части окна. Она отражает разбиение операционной части на поля. Чтобы установить необходимый размер поля подведите мышь к разделителю между полями на панели (при этом форма курсора видоизменится) и, нажав правую кнопку, перемещайте до тех пор, пока курсор не достигнет необходимой позиции (ориентируйтесь по числам под панелью). Результаты вы можете видеть слева в элементе «Размеры групп», а также в матрицах.

После этого необходимо заполнить матрицы информацией о сигналах. При этом действия будут подобны действиям в случае вертикального кодирования. Для ввода каждого сигнала следует произвести double-click над одной из серых ячеек слева в матрице и, введя номер, нажать ENTER. В результате станет возможным редактирование двоичной комбинации, соответствующей введенному номеру.

Кодирование раздельными полями

1 3|4 6|7 8|9 11

1..3 4..6 7..8 9..11

Количество групп: 4

Размеры групп:

- 1: 1..3 (3)
- 2: 4..6 (3)
- 3: 7..8 (2)
- 4: 9..11 (3)

Далее Назад

Рисунок А.5 – Кодирование раздельными полями

При нажатии «Далее» выполнится проверка на нахождение одинаковых комбинаций в пределах каждой матрицы. Если таковые будут найдены, эмулятор выдаст соответствующее предупреждение.

Помните, что в классическом варианте кодирования каждый номер сигнала в кодировке должен встречаться не более одного раза.

После ввода сигналов операционной части микрокоманды система предлагает ввести коды условной части. Так как для кодирования условной части в данной версии используется вертикальный принцип, то ввод значений условий X будет аналогичен вводу значений сигналов Y для вертикального кодирования операционной части (рисунок А.3).

В завершении эмулятор предложит ввести дополнительные параметры (рисунок А.6).

Дополнительные параметры

Номер Y конечного: Нет

Номер X, который всегда 0: Нет

Номер X, который всегда 1: Нет

Готово Назад

Нет X1 X2 X3

Рисунок А.6 – Дополнительные параметры

Это Y конечный т. е. тот выходной сигнал, получив который автомат прекращает работу, X , значение которого всегда 0 и X , значение которого всегда 1. (Не пытайтесь задать двум последним одинаковое значение). Вы можете выбирать значения из раскрывающихся списков.

Нажав на кнопку «Готово», эмулятор заканчивает ввод формата микрокоманды и переходит в режим редактирования и трассировки микропрограммы.

2.4. Редактирование и отладка микропрограммы

Эмулятор переходит в этот режим либо после ввода формата микрокоманды, либо после успешного выполнения команды меню «Открыть». Активным становится окно «визуальной трассировки», которое имеет вид такой, как один из представленных ниже (рисунок А.7 и рисунок А.8), в зависимости от выбранного способа адресации.

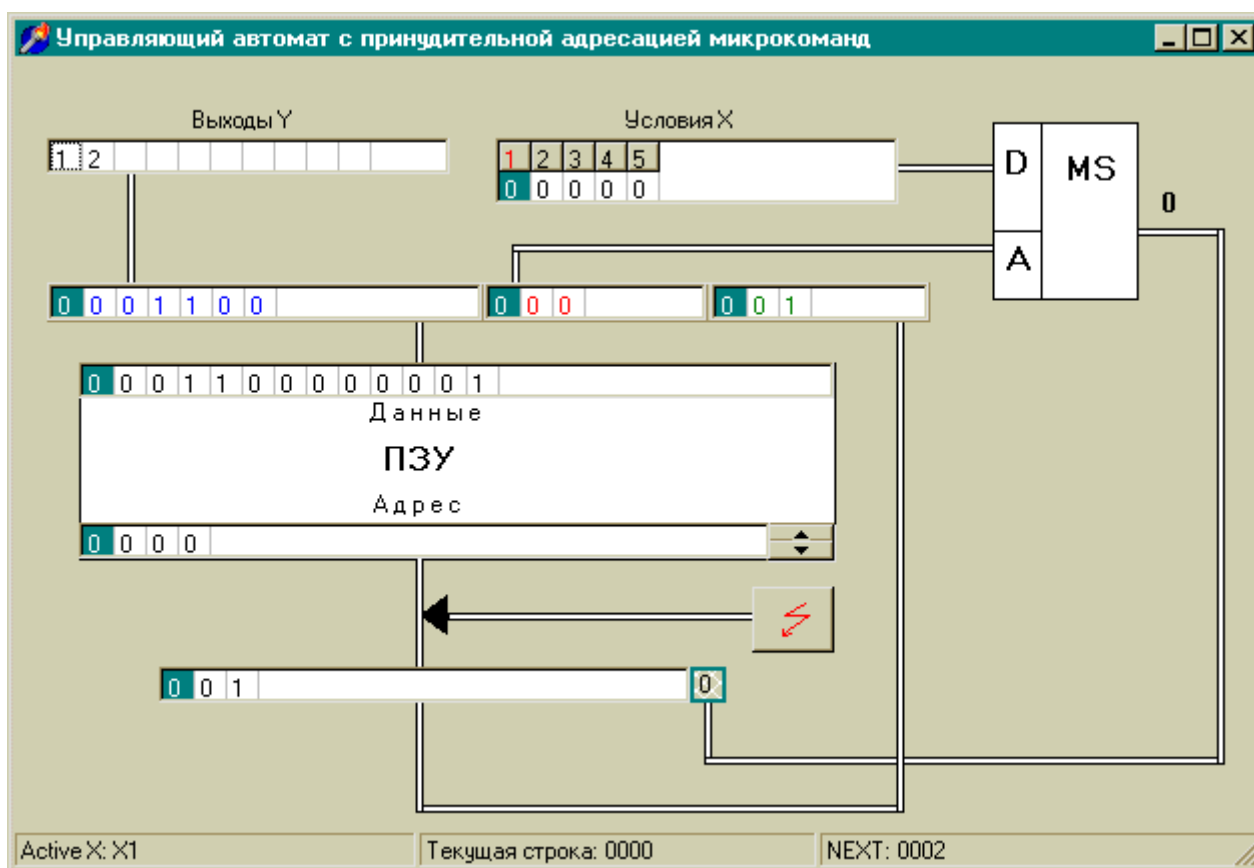


Рисунок А.7 – Окно «визуальной трассировки» для принудительной адресации

Здесь представлена структурная схема УА, её основные элементы: ПЗУ, регистры, мультиплексор. Кнопка с «молнией» используется, чтобы подать тактирующий импульс на УА. При этом выполняется одна микрокоманда. При необходимости можно изменить текущий номер микрокоманды с помощью кнопок, расположенных в поле адреса ПЗУ. Строка статуса окна содержит

текущие данные о состоянии эмулятора. Номер активного условия X, если оно есть, подсвечивается красным цветом в шине, идущей к мультиплексору. Вы можете редактировать текущую строку микрокоманды (по аналогии с действиями при вводе формата – клавиши «1», «0», «Пробел», нажатие double-click мыши), изменять видимые размеры полей в регистре над ПЗУ, перетаскивая границы полей мышью (если поле полностью не вмещается в отведённые ему границы, подведите курсор мыши на это поле и, подождя, увидите его полностью). Вы можете просматривать номер и значение активного условия, текущий и следующий адрес, активные выходные сигналы, линии, значение сигнала в которых влияет на выбор следующего адреса.

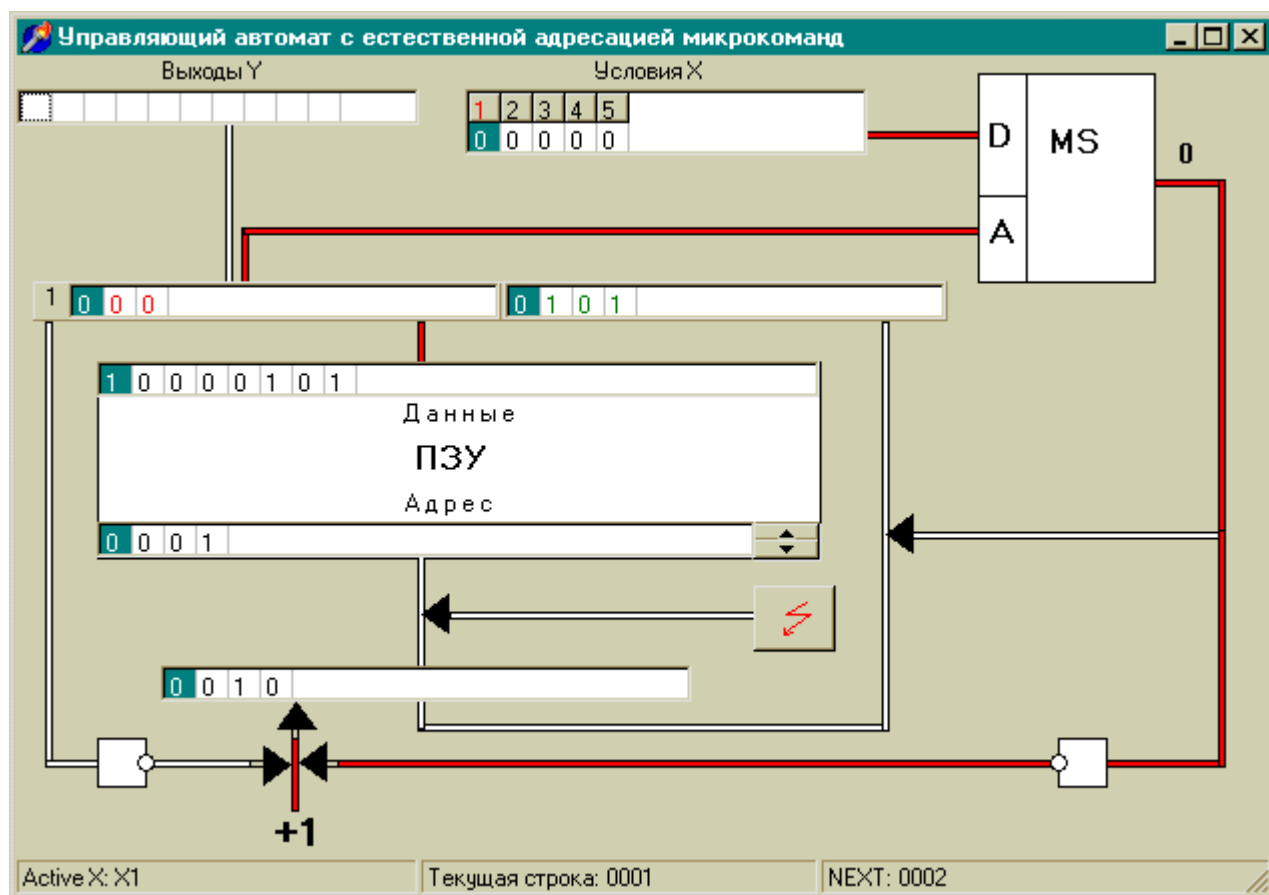


Рисунок А.8 – Окно «визуальной трассировки» для естественной адресации

Вы можете даже набрать в этом окне микропрограмму, но это гораздо удобнее делать в окне редактирования микропрограммы, перейти в которое можно через double-click в области ПЗУ, а также убрав текущее окно с экрана любым известным вам способом.

Таким образом, эмулятор делает активным окно редактирования микропрограммы, изображение которой представлено на рисунке А.9.

Если размер окна вас не устраивает, измените его сами «вручную», а затем выберите из меню «Справка» «Перерисовать». Эмулятор сам расположит внутренние окна удобным образом. Здесь главное «родительское» окно имеет 3

«дочерних». Это окно микропрограммы, окно текущих значений сигналов-условий X и окно активных выходов Y для текущей микрокоманды.

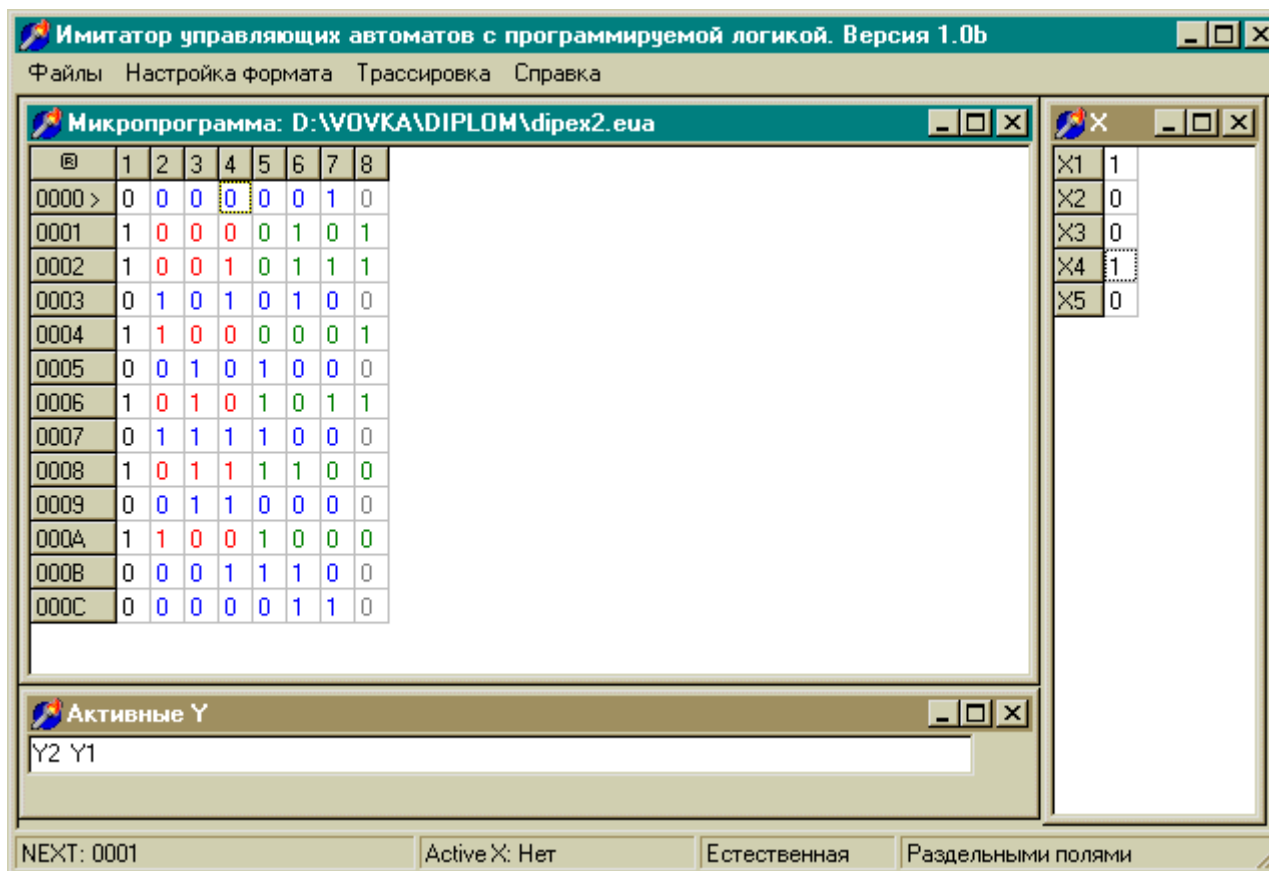


Рисунок А.9 – Окно редактирования микропрограммы

Вы можете редактировать микропрограмму как это уже описано прежде. Поля микрокоманд подсвечены разными цветами для облегчения понимания. Для ввода новой микрокоманды достаточно, находясь в последней строке, нажать стрелку «вниз».

Нумерация микрокоманд ведётся в шестнадцатеричной системе счисления. Текущая микрокоманда выделена в ячейке с номером символом '>'. Для пошаговой трассировки выберите из меню «Трассировка» «Шаг», или нажмите «F7». Строка статуса показывает данные о текущей имитации.

Для возврата в режим «визуальной трассировки» совершите double-click, в левой верхней ячейке окна микропрограммы.

2.5. Трассировка в непрерывном режиме

Эмулятор имеет возможность проводить непрерывную трассировку микропрограммы, создавая протокол трассировки. При этом могут быть использованы точки останова.

Для определения точки останова совершите double-click на ячейке с номером необходимой микрокоманды в окне микропрограммы. При этом она станет красной. Для снятия точки останова повторите эту процедуру.

Непрерывная трассировка начинается с текущего положения в микропрограмме и продолжается до конца программы, либо до Y конечного, либо до «заикливания», либо до точки останова. Для совершения непрерывной трассировки выберите из меню «Трассировка» «Запуск», либо нажмите «F9». При этом откроется окно следующего вида (рисунок А.10)

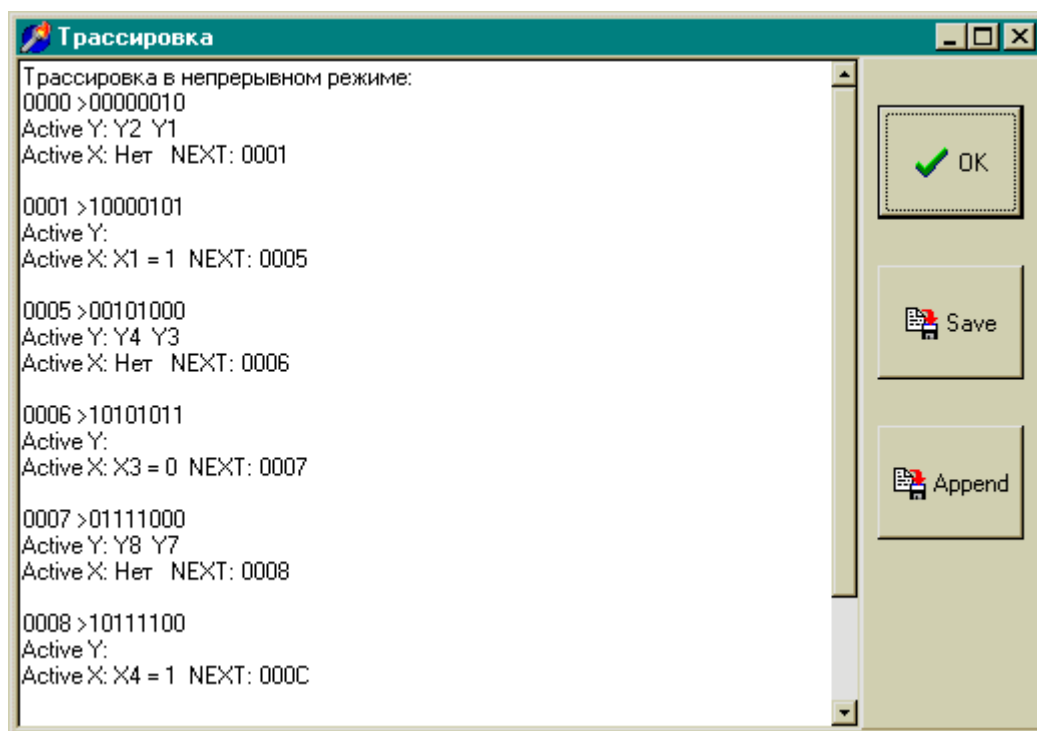


Рис. А.10 – Трассировка в непрерывном режиме

После вывода протокола на экран можно сохранить протокол в файле или добавить к уже существующему файлу. Для этого используйте кнопки справа. Текст в файле сохраняется в кодировке Windows и может быть прочитан стандартной программой «Блокнот».