микрокоманд БИС КІ804ВУ4 (см. табл. П10). Микропрограмма начинается с адреса 50_{10} . Фрагмент размещения микрокоманд в управляющей памяти представлен на рис. 4,в.

РАБОТА 3. ОРГАНИЗАЦИЯ ПОДПРОГРАММ И ЦИКЛОВ

Цель работы: приобретение навыков практической работы со стеком при организации в микропрограмме подпрограмм и циклов

Индивидуальное задание

1. Оформить фрагменты микроалгоритмов (см. рис. 2) в виде основной программы и трех подпрограмм, соединяя их в последовательности, указанной в табл. 5. Основная программа должна осуществлять вызов первой подпрограммы, та в свою очередь - второй, а вторая - третьей подпрограммы. Разработать микропрограмму функционирования БМУ, разместить микрокоманды в управляющей памяти и обозначить содержимое стека при выполнении стековых операций.

2. Для фрагментов микроалгоритмов 1 и 4 в микропрограмме организовать циклы с использованием стека.

Методические указания

Использование подпрограмм позволяет обращаться в различных местах основной программы к ее повторяющемуся фрагменту. При этом уменьшается количество ячеек памяти, занимаемой программой.

Во многих ЭВМ одна подпрограмма может вызвать другую, которая в свою очередь может вызвать третью подпрограмму и т.д. Для этого необходимо организовать "вложение" адресов возврата, служащих для связывания частей программы. Пример вложения подпрограмм показан на рис. 5. Обеспечить выборку адресов возврата в порядке, обратном их поступлению (LIFO), обеспечивает стековая память, имеющаяся в составе БИС КІ804ВУ4 /1, 2/.

Для обращения к подпрограмме можно использовать команду CJS (условный переход к подпрограмме), а для возврата из подпрограммы - команду CRTN (условный возврат из подпрограммы). Будем полагать, что сигнал на входе $\overline{CCE}$ БМУ имеет высокий уровень. Это приведет к отмене проверки условия $\overline{CC}$ и принудительному выполнению названных операций.

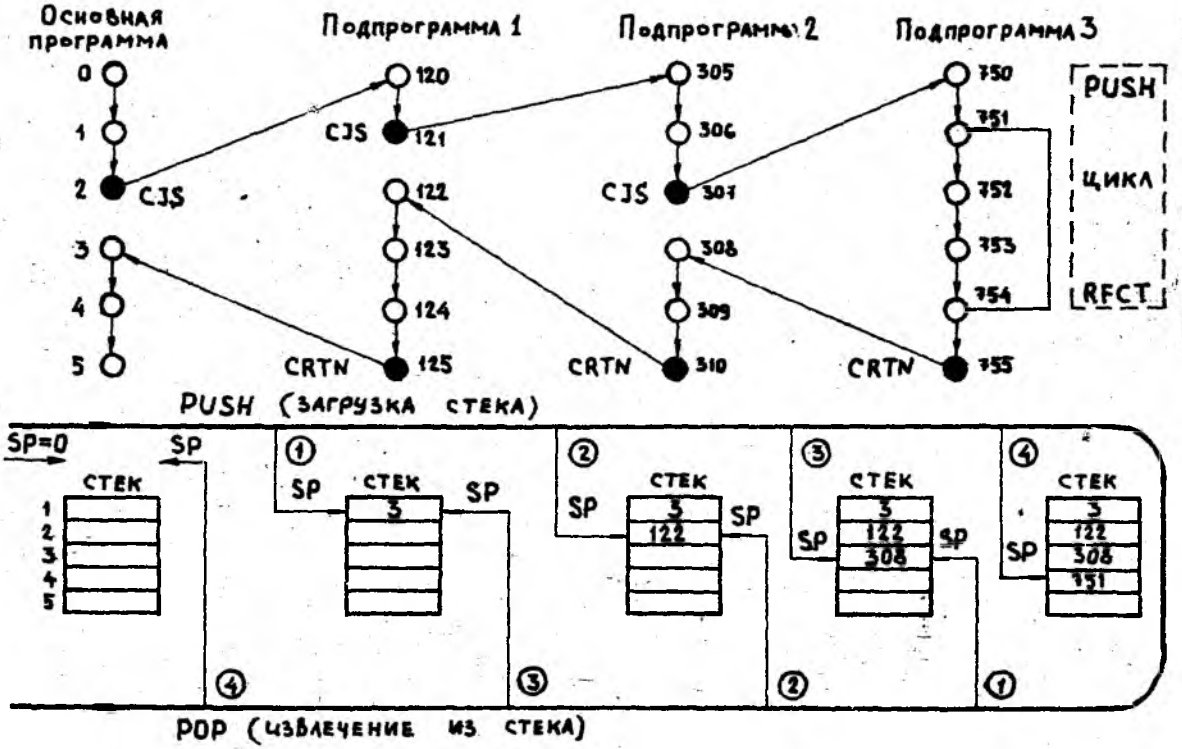

Рис. 5. Пример организации подпрограмм

В основной программе перед обращением к подпрограмме по команде JZ производится начальная установка указателя стека (SP:=D). При выполнении команды CJS в основной программе происходит передача управления по адресу 120 первой подпрограмме, а в первую ячейку стека записывается адрес 3 возврата в основную программу. При выполнении команды CRTN в подпрограмме адрес возврата извлекается из стека и управление передается основной программе.

Для организации в микропрограмме циклов обычно используются команды PUSH и RPCT . На рис. 5 цикл показан при выполнении третьей подпрограммы. По команде PUSH , находящейся по адресу 750, в стек загружается адрес 751, а в регист-счетчик СТ с регистра микрокоманд - величина, на единицу меньшая требуемого числа повторений (не более 4095). При выполнении команды RPCT проверяется, отлично ли содержимое СТ от нуля. Если это так, то происходит отрицательное приращение содержимого СТ, и адрес следующей микрокоманды извлекается из верхней части стека (это адрес 751). Если СТ = 0, то это рассматривается как условие выхода из цикла и управление передается следующей по порядку микрокоманде с адресом 754. Кроме того, выполняется отрицательное приращение SP, однако находящийся в вершине стека адрес не используется.

РАБОТА 4. МИКРОПРОГРАММЫ ВЫПОЛНЕНИЯ ОПЕРАЦИЙ УМНОЖЕНИЯ

Цель работы: приобретение практических навыков составления микропрограмм при выполнении операций умножения.

Индивидуальное задание

Разработать микропрограмму выполнения операции умножения 8-разрядных двоичных целых чисел по алгоритму, заданному в табл.9. Первоначально операнды расположены в регистрах общего назначения центрального процессора (табл. 10). Предусмотреть выполнение контрольного примера над операндами, заданными в табл. 10. При выполнении задания использовать структурную схемы БМУ, приведенную на рис. 3,а

Разрядность процессора принять равной разрядности исходных операндов.

Таблица 9

d_1	d_2	Алгоритм умножения
0	0	А [R1 (Σ), младш. МТ]
0	1	Б [L1 (МН), младш. МТ]
1	0	В [L1 (Σ), старш. МТ]
1	1	Г [R1 (МН), старш. МТ]
d_3		Представление операндов
0		прямой код
1		дополнительный код
d_4		Операционная часть
0		К1804 ВС1
1		К1804 ВС2

Таблица 10

d_5	d_6	МН	МТ	d_7	d_8	МН	МТ
0	0	R1	RQ	0	0	-5	+7
0	1	R10	R6	0	1	+6	-4
1	0	RQ	R7	1	0	-10	-23
1	1	R8	R13	1	1	+4	-7

В качестве примера рассмотрим реализацию алгоритма умножения по А в дополнительном коде на процессорных элементах КІ804ВС2.

Умножение по алгоритму А выполняется с младших разрядов множителя со сдвигом частичных сумм вправо на один разряд. Множимое в этом алгоритме неподвижно. В процессе перемножения чисел в дополнительном коде в результате получается одновременно знаковая и цифровая части произведения. Результат произведения дополнительных кодов сомножителей равен дополнительному коду результата в случае положительного множителя. Если множитель отрицательный, то произведение чисел в дополнительном коде получается прибавлением поправки (-МН) к произведению дополнительных кодов сомножителей. Пример выполнения операции умножения приведен на рис. 6.

Начальные условия выполнения алгоритма умножения: регистр R0 установлен в нулевое состояние, множимое находится в регистре R1, множитель - в R2 . В результате выполнения умножения двух n разрядных сомножителей получается $2n$ - разрядное произведение, n старших разрядов которого находятся в регистре R0 , а n младших разрядов - в регистре RQ .

Соединение шин при выполнении умножения в дополнительном коде показано на рис. 7. На входе CO должен быть "0" до последнего такта, когда он должен принимать значение сигнала на шине Z . Поэтому перед входом CO ставится вентиль, на который подается сигнал Z и разрешающий сигнал $\overline{OEZ}$, принимающий единичное значение только на последнем такте.

На рис. 8 представлена блок-схема, а в табл. II - микропрограмма умножения чисел по алгоритму А в дополнительном коде. После начальной установки множитель из регистра R2 высылается в регистр RQ . В счетчик тактов СТ БМУ устанавливается код I4, а в стек загружается адрес II3 для организации цикла умножения.

Цикл умножения чисел в дополнительном коде реализуется второй микрокомандой. Здесь имеется в виду, что признак Z совпадает с разрядом RQ[0] младшей процессорной секции. По каждому положительному фронту сигнала CLK выходы АЛУ сдвигаются в сторону младших разрядов и полученное частичное произведение записывается в регистр R0 . При этом в старший разряд RF3 поступает сигнал F3 ⊕ OVR , чем обеспечивается передача "I" при возникновении переноса при сложении и в случае отрицательного результата. Младший разряд сдвинутого частичного произведения передается

Мн=А=+5 , Мт=В=-3

	0 1 0 1 ,		A
	1 1 0 1 ,		$B_{дк}$
+	0 0 0 0	0 0 0 0	Σ0
	0 1 0 1		A
	0 1 0 1	0 0 0 0	Σ1
+	0 0 1 0	1 0 0 0	R1(Σ1)
	0 0 0 0		0
	0 0 1 0	1 0 0 0	Σ2
+	0 0 0 1	0 1 0 0	R1(Σ2)
	0 1 0 1		A
	0 1 1 0	0 1 0 0	Σ3
+	0 0 1 1	0 0 1 1	R1(Σ3)
	1 0 1 1		$A_{дк}$ (кор)
	1 1 1 0	0 0 1 0	Σ4
	1 1 1 1	0 0 0 1	, $П_{дк}$

Рис. 6. Алгоритм умножения

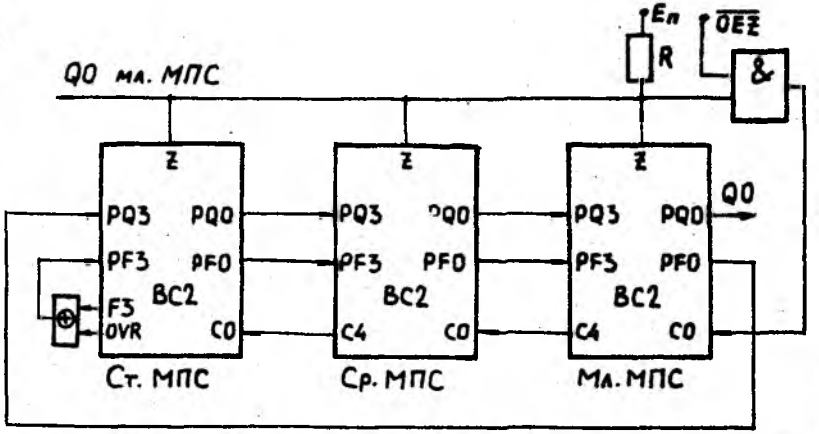

Рис. 7. Структурная схема алгоритма умножения

Таблица 11

Адрес МК	Микрокоманда: Управление К1804ВС2 I 8 7 6 5 4 3 2 1 0	$\overline{OEB}$	$\overline{OEY}$	AA	AB	Управление К1804ВУ4 $\overline{CCE}$	$\overline{OEZ}$	$\overline{RLD}$	I_{10}	D_{10}	Примечание: ФАМ	Операция ВС2
112	0 1 1 0 0 1 1 0 1	X	X	R2	X	1	0	0	4	14	PUSH	R2→RQ , 14→CT
113	0 0 1 0 0 0 0 0 0	0	0	R1	RQ	X	0	1	8	X	RFCT	Умножение в ДК
114	0 1 1 0 0 0 0 0 0	0	0	R1	RQ	X	1	1	14	X	CONT	Коррекция

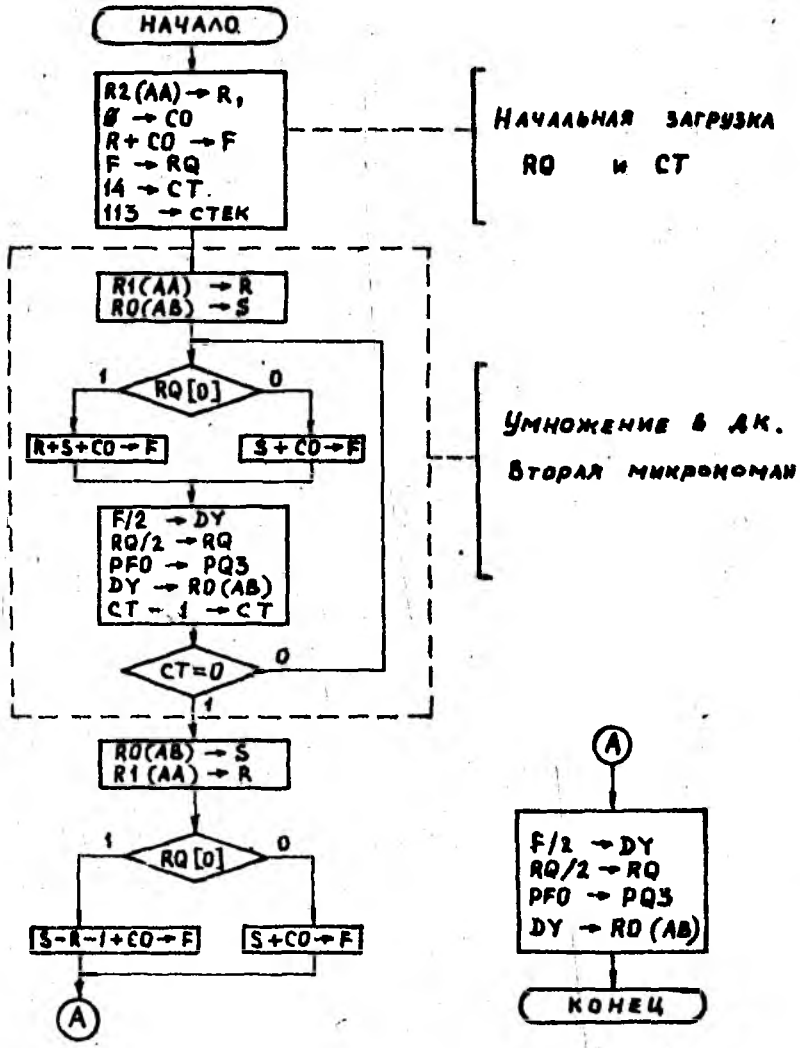

Рис. 8. Блок-схема алгоритма умножения