

Микропроцессоры серии К1814

Интегральные микросхемы серии К1814, выполненные по *p*-МДП-технологии, представляют собой однокристальные 4-разрядные мик-

ро-ЭВМ, предназначены для построения микропроцессорных систем управления. В серию входят:

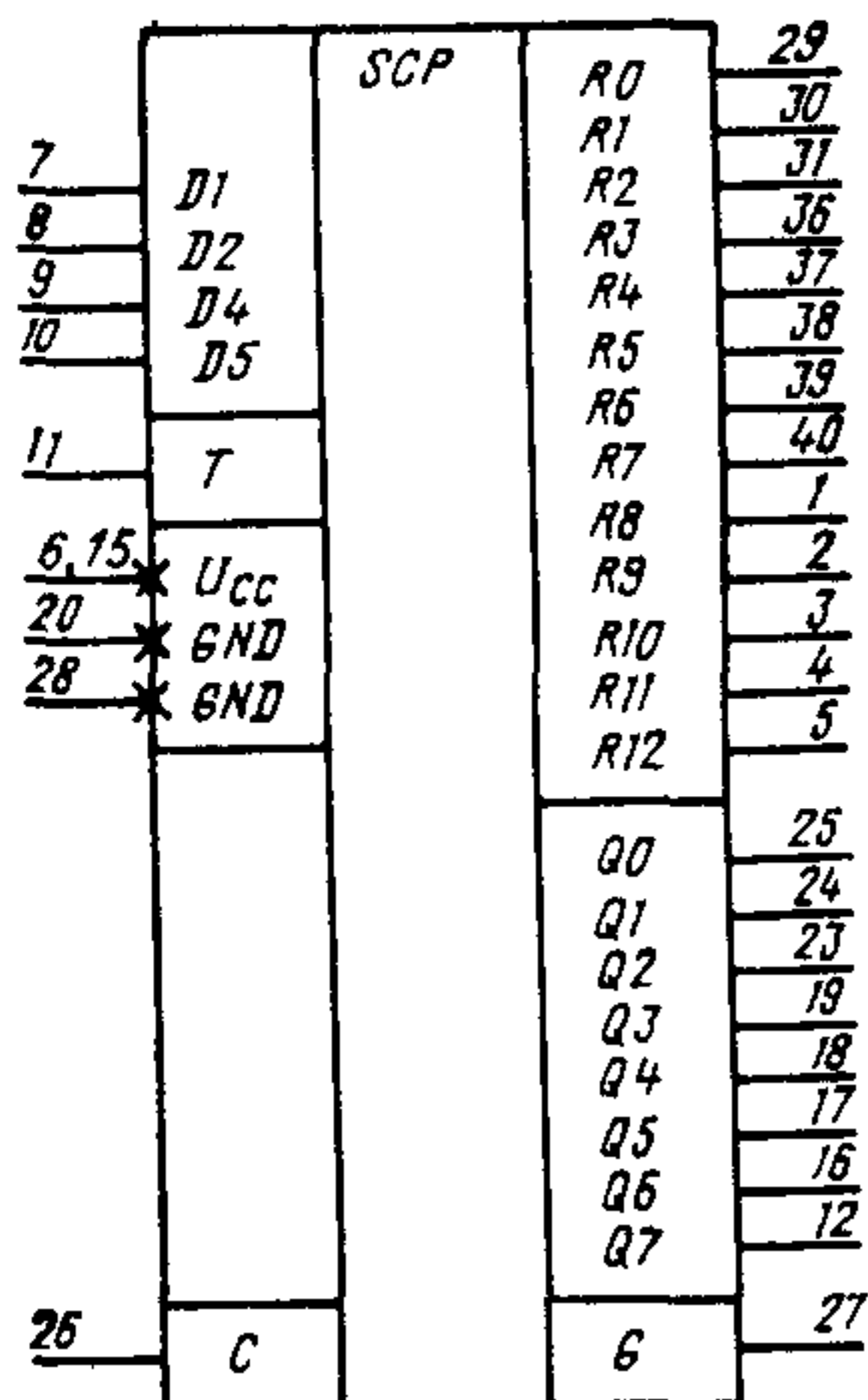


Рис. 1. Условное графическое обозначение однокристалльной микро-ЭВМ

Таблица 1

Параметр	Обозначение	Значения параметров	
		мин.	макс.
Напряжение питания, В	U_{CC}	-9,9	-8,1
Входное напряжение низкого уровня, В	U_{IL}	—	-4
Входное напряжение высокого уровня, В	U_{IH}	-0,75	—
Потребляемая мощность, мВт	P_{CC}	—	70
Тактовая частота, кГц	—	100	350
Рабочий диапазон температур, °С	T	-10	+70

Таблица 2

Вывод	Обозначение	Тип вывода	Функциональное назначение выводов
1—5, 29—31, 36—40	$R8—R12$, $R0—R2$, $R3—R7$	Выход	Порт R
7—10	$D1, D2, D4, D5$	Вход	Порт D
12, 16—19, 23—25	$Q7—Q0$	Выход	Порт Q
11	Γ	Вход	Сброс, тест
27	G	Выход	Тактовый сигнал
26	C	Вход	Тактовый сигнал
6, 15	U_{CC}	—	Напряжение питания
20, 28	GND	—	Общий

Примечание Выводы 13—15, 21, 22, 32—35 микросхемы не задействованы

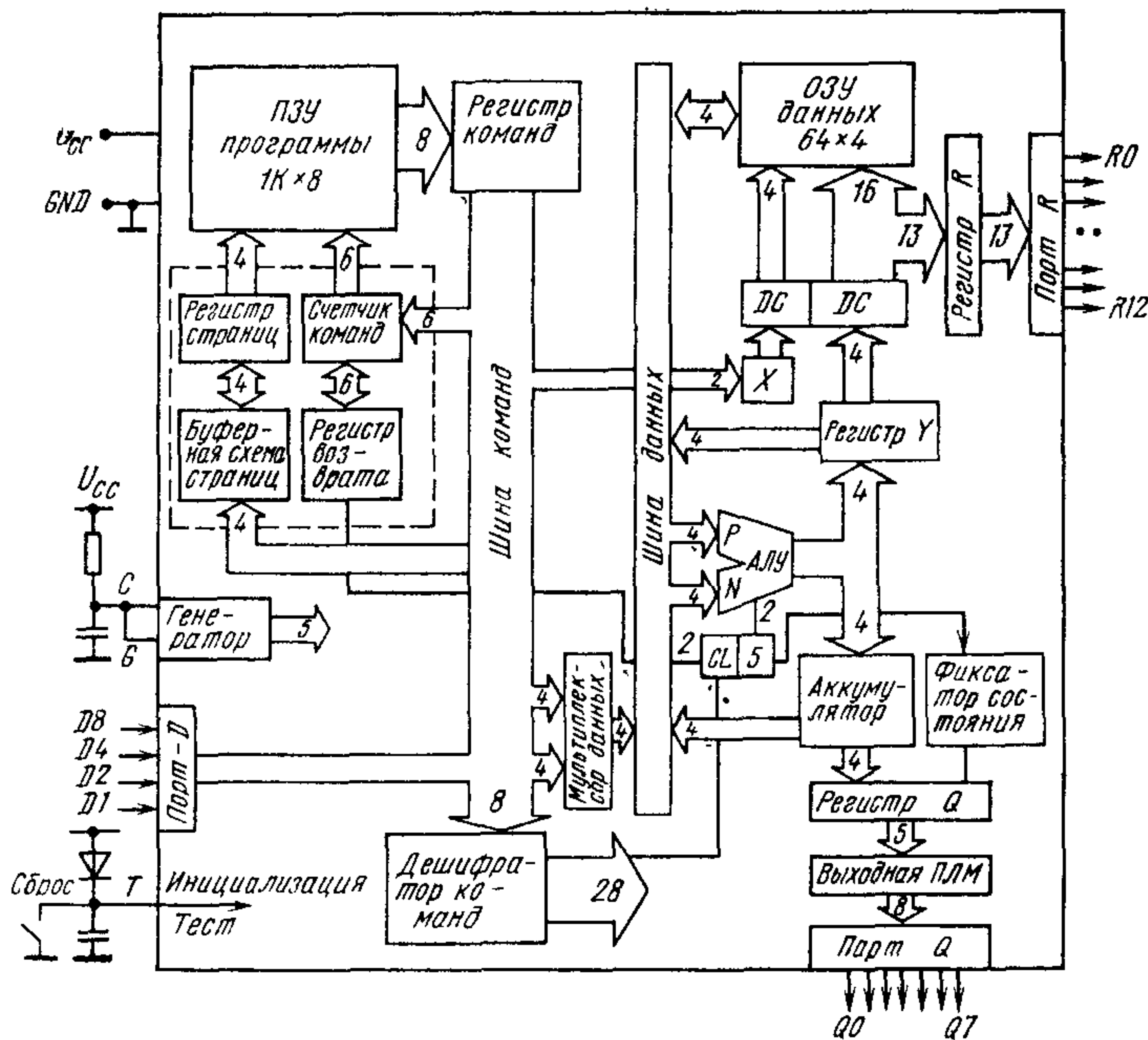


Рис. 3. Архитектура однокристалльной микро-ЭВМ

однокристалльная микро-ЭВМ (ОМ ЭВМ), которая содержит 4-разрядный процессор, ПЗУ программы емкостью 1К 8-разрядных команд, ОЗУ данных емкостью 64 4-разрядных слова, схемы ввода/вывода и встроенный тактовый генератор (КР1814ВЕ4);

Однокристалльные микро-ЭВМ выполняют функции специализированных микроконтроллеров в соответствии с программой, хранящейся в масочно-программируемом ПЗУ, и могут быть применены в периферийных устройствах вычислительной техники, различной контрольно-измерительной, медицинской и бытовой аппаратуре.

Условные графические обозначения микросхем ОМ ЭВМ приведены на рис. 1, функциональное назначение выводов микросхем ОМ ЭВМ показано в табл. 2.

Электрические параметры и режимы эксплуатации микросхем серии К1814 приведены в табл. 1.

Структурная схема ОМ ЭВМ приведены на

рис. 3. Микро-ЭВМ построена с разделением шин команд и данных, что позволяет использовать данные и команды независимой разрядности, а также реализовать конвейерный принцип выполнения операции. Четырехразрядная организация достаточна для выполнения большинства контроллерных функций и особенно удобна при использовании двоично-десятичной арифметики.

Разделение шин команд и данных обуславливает аппаратное и программное разделение адресации памяти и ОЗУ данных.

Адресное поле 1К разбито на 16 страниц по 64 байта. Адрес страницы задается 4-разрядным регистром адреса страницы (РА), а 6-разрядный счетчик команд (РС) адресует 64 команды внутри каждой страницы. РС не является последовательным счетчиком. Счет осуществляется по псевдослучайному закону, приведенному в табл. 4. Изменение порядка выборки команд программы осуществляется командами условного перехода *BR*, возврата из подпрограммы *RETN*. По командам *BR* и *CALL* осуществляется занесение РС 6-разрядного адреса перехода из адресного поля *W* этих команд. Одновременно в РА загружается 4-разрядный адрес страницы перехода из буферного регистра адреса страницы РВ.

Для реализации длинных ветвлений в РВ перед командами *BR* или *CALL* загружают страницы перехода из поля команды *LDP*. Если содержимое РА и РВ одинаково, то происходит короткое ветвление программы (переход на той же странице), если в РА из РВ загружается содержимое новой страницы, то

Номер команды	Восьмеричное значение РС	Номер команды	Восьмеричное значение РС	Номер команды	Восьмеричное значение РС	Номер команды	Восьмеричное значение РС
0	00	16	47	32	34	48	42
1	01	17	16	33	70	49	04
2	03	18	35	34	61	50	11
3	07	19	72	35	43	51	23
4	17	20	65	36	06	52	46
5	37	21	53	37	15	53	14
6	77	22	26	38	33	54	31
7	76	23	54	39	66	55	62
8	75	24	30	40	55	56	45
9	73	25	60	41	32	57	12
10	67	26	41	42	64	58	25
11	57	27	02	43	51	59	52
12	36	28	05	44	22	60	24
13	74	29	13	45	44	61	50
14	71	30	27	46	10	62	20
15	63	31	56	47	21	63	40

происходит длинное ветвление программы с переходом на другую страницу памяти. В случае команды *CALL* в регистре возврата из подпрограммы *SR* запоминается адрес возврата в основную программу, а в *PВ* запоминается адрес страницы основной программы.

По команде *RETN* происходит возврат в исходную программу; при этом содержимое *PВ* пересылается в *РА*, а содержимое *SR* — в *РС*. Команды *BR* и *CALL* условны по состоянию триггера состояния *S* и выполняются, если $S=1$. В противном случае ветвления не происходит и в следующем цикле *РС* адресует следующую команду программы. Такая структура адресации памяти программ проста в программировании, но не позволяет осуществлять вложение подпрограмм и длинные ветвления в подпрограмме.

Структура адресации *ОЗУ* данных также страничного типа. *ОЗУ* емкостью 64×4 бит разбито на четыре страницы по 16 4-разрядных слова. Адрес страницы задается 2-разрядным регистром *X*, адрес слова внутри каждой страницы задается регистром *Y*. В отличие от регистра *X* регистр *Y* участвует в операциях *АЛУ* и является регистром общего назначения (*РОН*). Изменение содержимого регистра *X* осуществляется по командам «Загрузить регистр *X*» (*LDX*) и «Образовать дополнение содержимого регистра *X*» (*COMX*).

Четырехразрядное *АЛУ* выполняет операции сложения, вычитания, арифметического и логического сравнения над операциями, поступающими с шины данных через входные мультиплексоры. Результат арифметической операции через выходной селектор в соответствии с микрокомандами отсылается в один из *РОН*, аккумулятор или регистр *Y*. Одновременно с арифметическими операциями *АЛУ* производит

сравнение входных величин и выработку признака равенства или неравенства операндов. Результат сравнения (*NE*) или содержимое разряда переноса (*CR*) соответствующими микрокомандами могут пересылаться в триггер состояния.

Входными операндами *АЛУ* могут быть выходные коды регистра *Y*, *ОЗУ*, мультиплексора данных *СДВ*, прямой или инверсный код аккумулятора, константа 15.

Мультиплексор данных *СДВ* представляет собой логическую схему, предназначенную для выбора и передачи на шину данных входной информации из порта *D*, константы *C* с шины команд или битовой маски для маскирования разрядов при выполнении операций над битами.

Пересылки данных в *РОН* осуществляются через *АЛУ* путем сложения их с нулями.

Базовая система команд (табл. 5) содержит 43 команды. Выполняемые операции: пересылка, арифметические, арифметическое и логическое сравнение, поразрядная обработка слов памяти, загрузка констант, ввода/вывода, адресации. Все команды однобайтовые и имеют один из четырех форматов согласно табл. 6. Декодирование кодов команд и формирование сигналов управления осуществляются дешифратором команд. На выходе дешифратора формируется горизонтальный микрокод, разряды которого инициируют выполнение микроопераций. Часть дешифратора выполнена на базе *ПЛМ* мощностью в 30 произведений, причем матрица «И» *ПЛМ* осуществляет непосредственное декодирование кода команды, а матрица «ИЛИ» выполняет функции памяти микропрограмм. Управляющие сигналы, возбуждаемые на выходе *ПЛМ*, инициируют выполнение комбинаций из 16 программируемых микроопераций. Для обеспечения возможности формирования сложных команд типа «Чтение—операция—запись» выбрана организация выполнения микроопераций, обычная для горизонтального микропрограммирования.

Для некоторых операций, связанных с адресацией, вводом и модификацией разрядов, достаточно произвести одну микрооперацию, причем эти микрооперации не используются в совокупности с другими микрооперациями в одном командном цикле. Поэтому с целью экономии микропрограммной *ПЛМ* формирование сигналов управления для данных операций осуществляется непосредственно на выходе дешифратора (матрица «И» *ПЛМ*).

Слово состояния программы определяется содержанием триггера признака подпрограммы *CL* и триггера состояния *S*. Триггер признака подпрограммы устанавливается командой *CALL* и определяет занесение адреса возврата в регистр возврата и буфер страницы и блокировку длинных переходов в режиме вызова. Таким образом, длина подпрограммы не может превышать 64 команды. Сброс триггера производится при возвращении в основную программу по команде *RETN*.

Таблица 5

Тип команд	Мнемоника команд	Код операции (двоичный)	Состояние PC	Описание команды	
Пересылки	<i>TAY</i>	00100100	1	Переслать содержимое аккумулятора в регистр Y	
	<i>TYA</i>	00100011	1	Переслать содержимое регистра Y в аккумулятор	
	<i>TAM</i>	00000011	1	Переслать содержимое аккумулятора в память	
	<i>TAMIY</i>	00100000	1	Переслать содержимое аккумулятора, инкремент регистра Y	
	<i>TAMZA</i>	00000100	1	Переслать содержимое аккумулятора в память, очистить аккумулятор	
	<i>TMY</i>	00100010	1	Переслать содержимое памяти в регистр Y	
	<i>TMA</i>	00100001	1	Переслать содержимое памяти в аккумулятор	
	<i>XMA</i>	00101110	1	Поменять содержимое памяти и аккумулятора	
Засылки констант	<i>CLA</i>	00101111	1	Очистить аккумулятор	
	<i>TCY</i>	0010(C)	1	Загрузить константу в регистр Y	
	<i>TSMIY</i>	0110(C)	1	Загрузить константу в память, увеличить регистр Y	
Арифметические операции	<i>AMMAC</i>	00100101	CR	Сложить содержимое аккумулятора и памяти, результат переслать в аккумулятор	
	<i>SAMAN</i>	00100111	CR	Вычесть содержимое аккумулятора из памяти, результат переслать в аккумулятор	
	<i>IMAC</i>	00101000	CR	Икремент содержимого памяти и загрузить в аккумулятор	
	<i>DMAN</i>	00101010	CR	Декремент содержимого памяти и загрузить в аккумулятор	
	<i>IA</i>	00001110	1	Икремент содержимого аккумулятора	
	<i>DAN</i>	00000111	CR	Декремент содержимого аккумулятора	
	<i>IYC</i>	00101100	CR	Икремент содержимого регистра Y	
	<i>DYN</i>	00101100	CR	Декремент содержимого регистра Y	
	<i>A6AAC</i>	00000110	CR	Увеличить содержимое аккумулятора на 6	
	<i>A8AAC</i>	00000001	CR	Увеличить содержимое аккумулятора на 8	
	<i>A10AAC</i>	00000101	CR	Увеличить содержимое аккумулятора на 10	
	<i>CRAIZ</i>	00101101	CR	Образовать дополнение аккумулятора до 2, установить триггер состояния	
	Сравнения	<i>ALEM</i>	00101001	CR	Если аккумулятор меньше или равен памяти, то установить триггер состояния
		<i>ALEC</i>	0111(C)	CR	Если аккумулятор меньше или равен константе, то установить триггер состояния
<i>MNEZ</i>		C0100110	NE	Если содержимое памяти не равно 0, то установить триггер состояния	
<i>YNEA</i>		00000010	NE	Если содержимое регистра Y не равно аккумулятору, то установить триггер состояния, бит состояния переслать в фиксатор состояния	
<i>YNEC</i>		0101(C)	NE	Если содержимое регистра Y не равно константе, то установить триггер состояния	
Побитовой обработки памяти	<i>SBIT</i>	001100B	1	Установить бит памяти	
	<i>RBIT</i>	001101B	1	Сбросить бит памяти	
	<i>TBITI</i>	001110B	NE	Проверить бит памяти, и если 1, то установить триггер состояния	
Адресации страниц ОЗУ	<i>LDX</i>	001111B	1	Загрузить константу в регистр X	
	<i>COMX</i>	00000000	1	Ивертировать содержимое регистра X	
	<i>LDP</i>	0001(C)	1	Загрузить константу в буфер адреса страницы	
	<i>BR</i>	10(W)	1	Переход, условно по состоянию	
	<i>CALL</i>	11(W)	1	Обращение к подпрограмме, условно по состоянию	

Тип команд	Мнемоника команд	Код операции (двоичный)	Состояние PC	Описание команды
Ввод/вывод	<i>RETN</i>	00001111	1	Возврат из подпрограммы
	<i>DNEZ</i>	00001001	<i>NE</i>	Если входные данные не равны 0, то установить триггер состояния
	<i>TDA</i>	00001000	1	Переслать входные данные в аккумулятор
	<i>SETR</i>	00001101	1	Установить разряд регистра <i>R</i>
	<i>TIQ</i>	00001010	1	Переслать содержание аккумулятора и фиксатора состояния в регистр <i>Q</i>
	<i>CLQ</i> <i>PSTR</i>	00001011 00001100	1 1	Очистить регистр <i>Q</i> Сбросить разряды регистра <i>R</i>

Примечание. *CR* — перенос в результате арифметической операции, перенос-заем; *NE* — результат компарации (*NE=1*, если операнды не равны); (*C*) — код константы; (*W*) — 6-разрядное поле адреса перехода.

Триггер состояния определяет разрешение передач управления. Содержимое триггера определяется командой, выполненной в предыдущем цикле. В зависимости от команды (см. табл. 5) триггер состояния принимает безусловное значение 1, значение разряда переноса сумматора, либо результата сравнения операндов компаратором. Содержимое триггера состояния через выходной фиксатор состояния *SL* может быть выведено в выходной регистр *Q*. Фиксатор состояния может быть модифицирован только командой *YNEA*.

Микро-ЭВМ имеет отдельные схемы ввода и вывода, позволяющие параллельно вводить входные данные и выводить содержимое двух выходных регистров *Q* и *R* на отдельные порты вывода.

Входные данные вводятся в схему через 4-разрядный входной порт *D* и далее через мультиплексор данных поступают на шину данных. Входная информация из порта *D* может пересылаться в аккумулятор по команде *TDA* или в АЛУ для проверки на нуль по

команде *DNEZ*. Выходной регистр *Q* используется для параллельного вывода данных. Запись в регистр *Q* производится по команде вывода *TIQ*; при этом содержимое аккумулятора и фиксатора состояния *SL* пересылается в регистр *Q*.

Для удобства представления выходной информации порта *Q*, например на 7-сегментном индикаторе или другом устройстве отображения информации, в схеме имеется выходной шифратор информации, выполненный на базе ПЛМ мощностью 20 произведений. Матрица «ИЛИ» ПЛМ имеет восемь выходных линий в соответствии с разрядностью порта *Q*. Программирование выходной ПЛМ производится маской на стадии изготовления в соответствии с требованиями пользователя. В универсальной микро-ЭВМ выходная ПЛМ кодируется как повторитель.

Выходной регистр *R* имеет 13 индивидуально программируемых ячеек памяти. В одном командном цикле может быть установлена (командой *SETR*) или сброшена (командой *RSTR*) только одна ячейка регистра *R*, адресуемая текущим содержимым регистра *Y*.

Все ячейки регистра *R* имеют параллельный выход на выводы порта *R*. Индивидуальное программирование выходных линий порта *R* позволяет эффективно использовать их для непосредственного управления исполнительными устройствами, стробирования входных и выходных данных, сканирования клавиатуры, дисплея и других применений.

Разделение шин команд и данных позволяет организовать двухуровневый конвейер операций, совместив цикл выборки и исполнительный цикл соседних команд. При такой организации время выполнения операций определяется длительностью машинного цикла микро-ЭВМ, а не полного цикла команды. Длительность машинного цикла составляет 6 тактов генератора или 20 мкс на частоте 300 кГц и одинакова для всех команд, что очень удобно при программировании задач реального времени. Синхронизация схемы в пределах цикла

Таблица 6

Формат команд	Разряд код команд							
	<i>K8</i>	<i>K7</i>	<i>K6</i>	<i>K5</i>	<i>K4</i>	<i>K3</i>	<i>K2</i>	<i>K1</i>
I	КОП		MSB			(W)	LSB	
II	КОП		LSB (C)			MSB		
III	КОП		LSB (B)			MSB		
IV	КОП							

Примечание. *K8-K1* — разряды кода команды, (*W*) — 6-разрядное поле адреса перехода; (*C*) — 4-разрядное поле константы или адреса страницы программной памяти; (*B*) — 2-разрядное поле бита ячейки ОЗУ.

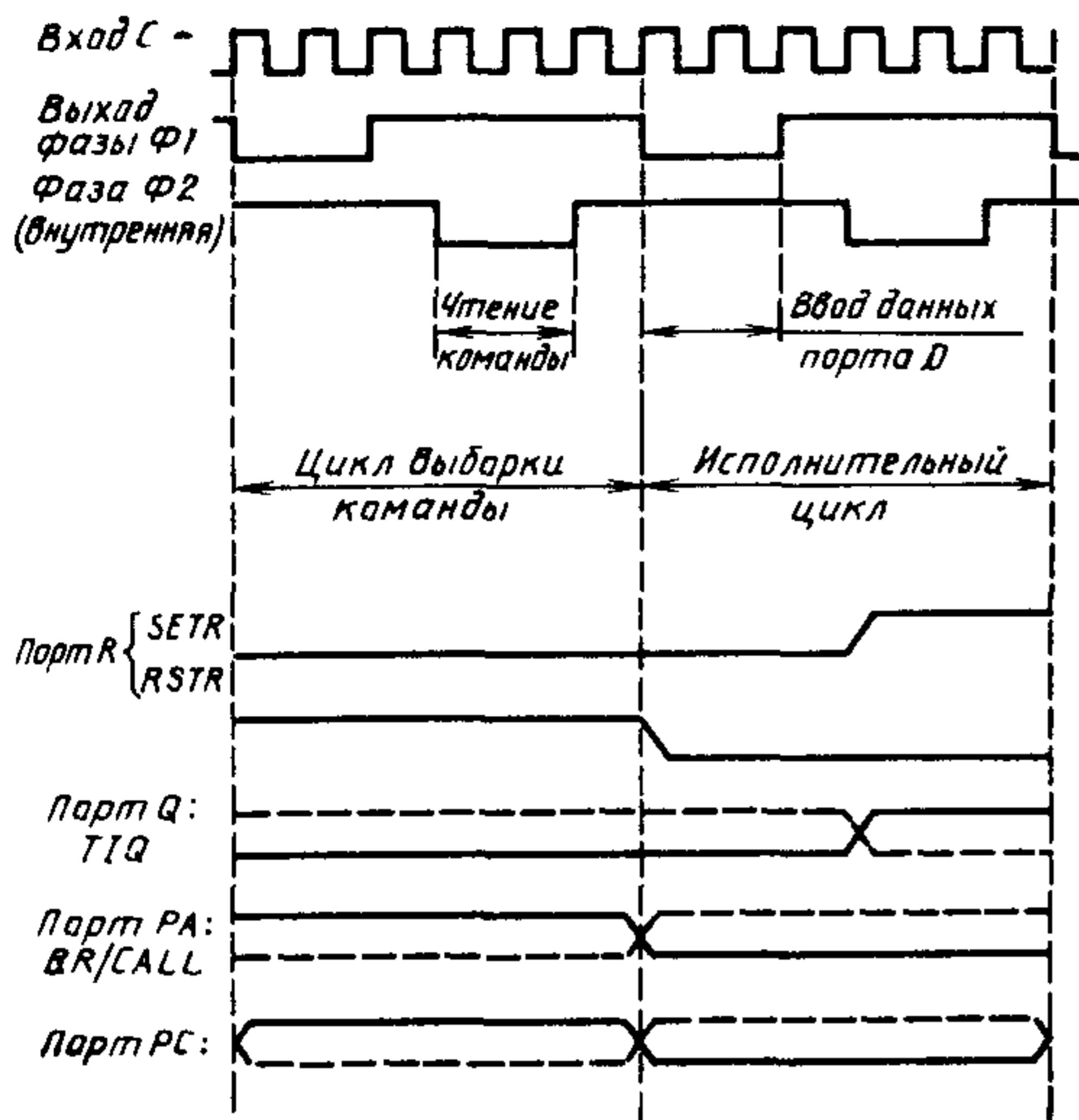


Рис. 4. Временные диаграммы работы микро-ЭВМ серии K1814

осуществляется пятью фазами, часть которых приведена на рис. 4. Здесь же указаны временные интервалы выполнения различных микроопераций, для команд вывода приведены временные диаграммы.

При включении питания специальной схемой производится начальная установка микросхемы в исходное состояние: $PA=PB=15$, $PC=0$, $Q=0$, $R=0$, $CL=0$. Поддержка внутренней схемы инициализации при работе с источниками питания, не обеспечивающими крутой фронт нарастания напряжения в микросхеме, обеспечивается по выводу T путем созда-

ния задержки установки уровня нуля на данном выводе внешней емкостью. Разряд емкости при отключении питания производится внешним диодом. Сброс микро-ЭВМ в начальное состояние осуществляется путем установки на выводе T напряжения высокого уровня на время не менее шести машинных циклов при нулевой информации на входах порта D . Состояние схемы после сброса аналогично состоянию после начальной установки.

Микро-ЭВМ может работать с внешней и внутренней синхронизацией. В первом случае тактовые импульсы подаются на вход C с внешнего генератора, во втором при подключении RC -цепи к замкнутым выводам C и G внутренний релаксационный генератор формирует тактовые импульсы с частотой, определяемой параметрами RC -цепи. Типовое значение частоты составляет 300 кГц.

На рис. 5 приведена схема включения однокристалльной микро-ЭВМ для реализации простого секундомера. При нажатии на кнопку 1 начинается отсчет времени с отображением на 4-разрядном индикаторе десятых долей секунды, секунд, десятков секунд и минут. При нажатии на кнопки 2 или 3 происходит остановка счета времени или сброс секундомера соответственно. Для отсчета времени на вход $D8$ микро-ЭВМ подаются импульсы частотой 50 кГц.

Разряды порта R $R0-R3$ используются для сканирования клавиатуры и индикатора, причем сигнал высокого уровня на выходах R соответствует разрешению свечения соответствующего разряда индикатора. Предполагается, что выходная ПЛМ закодирована для преобразования содержимого регистра Q в коды семисегментного индикатора так, что нулевому значению регистра Q соответствует 0 на инди-

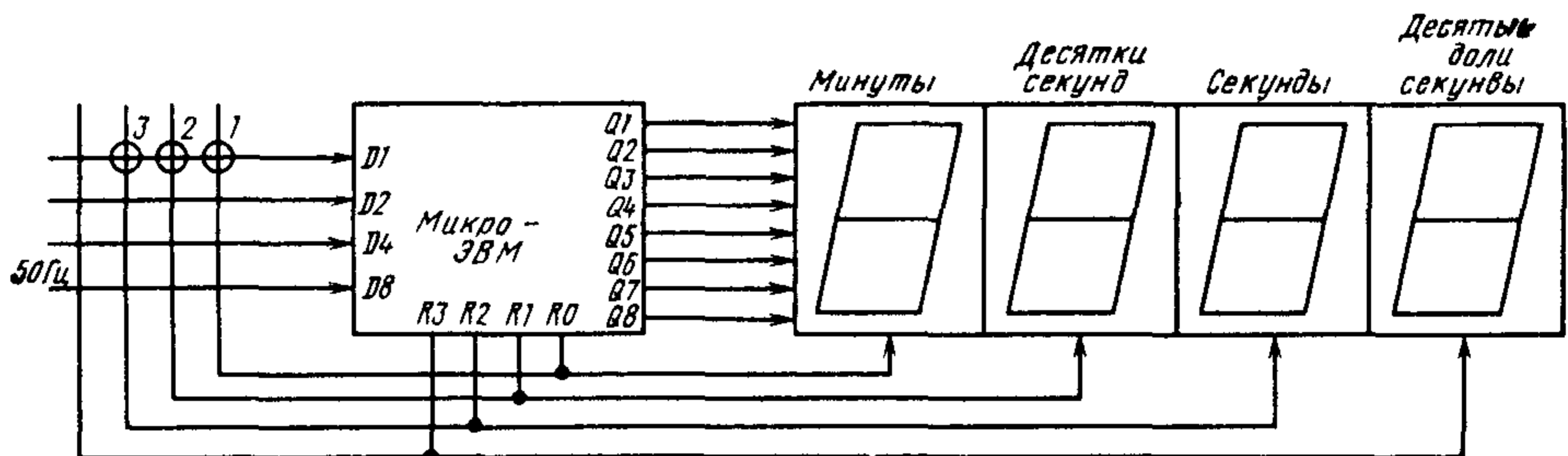


Рис. 5. Схема включения микро-ЭВМ при реализации секундомера: 1 — кнопка «Пуск»; 2 — кнопка «Стоп»; 3 — кнопка «Сброс»

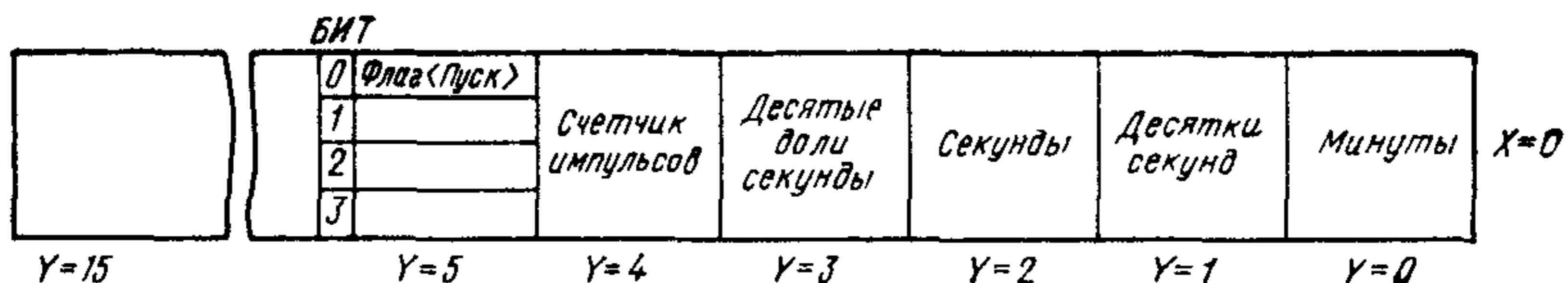


Рис. 6. Распределение памяти данных при реализации секундомера

Адрес строки ЗУ программы	Адрес команды	Код команды	Метка перехода	Мнемоника команды	Операнд	Примечание
15	0	00101111	<i>STA</i>	<i>CLA</i>		Старт программы 0→M (X=0, Y=0-5)
15	1	00111100		<i>LDX</i>	0	
15	3	01001010		<i>TCY</i>	5	
15	7	00000011	<i>CL</i>	<i>TAM</i>		
15	15	00101100		<i>DYN</i>		
15	31	10000111		<i>BR</i>	<i>CL</i>	Программа сканирования клавиатуры и дисплея
15	63	01000010	<i>DIS1</i>	<i>TCY</i>	4	
15	62	00101100	<i>DIS2</i>	<i>DYN</i>		
15	61	00100001		<i>TMA</i>		
15	59	00101011		<i>IYC</i>		
15	55	00001100		<i>RSTR</i>		
15	47	00101100		<i>DYN</i>		
15	30	00001010		<i>TIQ</i>		
15	60	00001101		<i>SETR</i>		
15	57	00001001		<i>DNEZ</i>		
15	51	10111010		<i>BR</i>	<i>IMP</i>	
15	39	01011111	<i>DIS3</i>	<i>YNEC</i>	15	
15	14	10111110		<i>BR</i>	<i>DIS2</i>	
15	29	10111111		<i>BR</i>	<i>DIS1</i>	
15	58	00001000	<i>IMP</i>	<i>TDA</i>		
15	53	01111000		<i>ALEC</i>	1	
15	43	10001010		<i>BR</i>	<i>F</i>	
15	22	00100011		<i>TYA</i>		
15	44	01001010		<i>TCY</i>	5	
15	24	00111000		<i>TBIT1</i>	0	Пуск — > на счет времени <Пуск> не установлен Счет времени
15	48	10000101		<i>BR</i>	<i>CLC</i>	
15	33	00100100		<i>TAY</i>		
15	2	10100111		<i>BR</i>	<i>DIS3</i>	
15	5	00100100	<i>CLC</i>	<i>TAY</i>		
15	11	00001100		<i>RSTR</i>		
15	23	01000010		<i>TCY</i>	4	
15	46	00101000		<i>IMAC</i>		
15	28	00000011		<i>TAM</i>		
15	36	01110010		<i>ALEC</i>	4	
15	49	10011001		<i>BR</i>	<i>WAIT</i>	
15	35	00101111	<i>INC</i>	<i>CLA</i>		
15	6	00000011		<i>TAM</i>		
15	13	00101100		<i>DYN</i>		
15	27	00101000		<i>IMAC</i>		
15	54	00000011		<i>TAM</i>		
15	45	00001111		<i>RETN</i>		
15	26	01111001		<i>ALEC</i>	9	
15	52	10011001		<i>BR</i>	<i>WAIT</i>	
15	41	11100011		<i>CALL</i>	<i>INC</i>	
15	18	01111001		<i>ALEC</i>	9	
15	36	10011001		<i>BR</i>	<i>WAIT</i>	
15	8	11100011		<i>CALL</i>	<i>INC</i>	
15	17	01111010		<i>ALEC</i>	5	
15	34	10011001		<i>BR</i>	<i>WAIT</i>	
15	4	11100011		<i>CALL</i>	<i>INC</i>	
15	9	01111001		<i>ALEC</i>	9	
15	19	10011001		<i>BR</i>	<i>WAIT</i>	
15	38	00101111		<i>CLA</i>		
15	12	00000011		<i>TAM</i>		
15	25	00001001	<i>WAIT</i>	<i>DNEZ</i>		Задержка до конца ИМП—СА
15	50	10011001		<i>BR</i>	<i>WAIT</i>	
15	37	10111111		<i>BR</i>	<i>DIS1</i>	

Адрес страницы ЗУ программы	Адрес команды	Код команды	Метка перехода	Мнемоника команды	Операнд	Примечание	
15	10	00010000	<i>F</i>	<i>LDP</i>	0		
15	21	10000000		<i>BR</i>	<i>FILT</i>		
0	0	11000000		<i>MEMORY</i>	00 000		
0	0	00100011	<i>FILT</i>	<i>TYA</i>		Определение кнопки	
0	1	01110000		<i>ALEC</i>	0		
0	3	10111110		<i>BR</i>	<i>SETFL</i>		
0	7	01111000		<i>ALEC</i>	1		
0	15	10101111		<i>BR</i>	<i>RSTFL</i>		
0	31	00011111		<i>LDP</i>	15		
0	63	10000000		<i>BR</i>	<i>STA</i>		
0	62	01001010	<i>SETFL</i>	<i>TCY</i>	5		Установка флага <пуск>
0	61	00110000		<i>SBIT</i>	0		
0	59	00100100		<i>TAY</i>			
0	55	10111001		<i>BR</i>	<i>DIS</i>		
0	47	01001010	<i>RSTFL</i>	<i>TCY</i>	5	Сброс <флага> <пуск>	
0	30	00110100		<i>RBIT</i>	0		
0	60	00100100		<i>TAY</i>			
0	57	00011111	<i>DIS</i>	<i>LDP</i>	15		
0	51	10100111		<i>BR</i>	<i>DIS3</i>		
0	0	00000000		<i>END</i>			

каторе, единице соответствует 1 на индикаторе, и так до девяти. Программа микро-ЭВМ, реализующая секундомер, приведена ниже

(табл. 7). В программе используются ячейки $Y=0-5$ нулевой страницы ОЗУ данных. Назначение ячеек ОЗУ приведено на рис. 6.