ФЕДЕРАЛЬНОЕ АГЕНТСТВО ЖЕЛЕЗНОДОРОЖНОГО ТРАНСПОРТА
Федеральное Государственное Бюджетное Образовательное Учреждение Высшего Профессионального Образования
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ»
(МИИТ)

Кафедра: «Железнодорожная автоматика, телемеханика и связь»

ОСНОВЫ МИКРОПРОЦЕССОРНОЙ ТЕХНИКИ

Задание на контрольную работу №1 с методическими указаниями по дисциплине для студентов-специалистов 3 курса,
специальности: **«Системы обеспечения движения поездов»**

специализации: **«Электроснабжение железных дорог»**

Москва, 2013 г.

ОБЩИЕ УКАЗАНИЯ

В контрольной работе студент должен разработать структуру микропроцессорного устройства управления объектом, составить машинный алгоритм функционирования данного устройства и по нему написать в операторах языка Ассемблер программу. Полученную программу записать в машинных кодах с использованием шестнадцатеричной системы счисления и отладить с использованием программного эмулятора.

Для выполнения контрольной работы необходимо:

- рекомендуемую литературу;
- определить свой вариант задания;
- изучить заданный алгоритм работы микропроцессорного устройства управления объектом и дать его описание;
- разработать и привести структурную схему микропроцессорного устройства управления объектом, предполагая, что программа функционирования и исходные данные хранятся в постоянном запоминающем устройстве (ПЗУ), для хранения промежуточных и окончательных результатов используется оперативное запоминающее устройство (ОЗУ), каждый тип результата выводится через свой порт вывода;
- привести описание разработанной структурной схемы микропроцессорного устройства;
- составить машинный алгоритм работы микропроцессорного устройства;
- написать текст программы на языке Ассемблер с комментариями;
- представить текст программы в машинных кодах используемого микропроцессора в шестнадцатеричной системе счисления;
- определить адресное пространство программы, используемые адреса ячеек ПЗУ и ОЗУ, адреса портов вывода;
- отладить программу, используя программный эмулятор микропроцессора.

Вариант задания (прил.1) соответствует последней цифре шифра студента и определяет заданный алгоритм работы микропроцессорного устройства. Последние три цифры учебного шифра определяют адрес ячейки памяти в шестнадцатеричной системе счисления, с которой начинается программа. Две последние цифры учебного шифра определяют значения константы допуска DOP, используемой в алгоритме, также в шестнадцатеричной системе счисления. Параметры PARIN и PAROU выбираются произвольно так, чтобы при отладке программы на эмуляторе выполнялись все ветви алгоритма (задается несколько значений).

В контрольной работе должны быть выполнены все пункты задания. Пояснительная записка должна содержать исходные данные по варианту, схему микропроцессорного устройства и схемы алгоритмов. Каждый чертеж вставляется в пояснительную записку после той страницы, на которой имеется первая ссылка на него. Пояснения выполненной студентом работы должны быть краткими и разборчивыми для чтения. В контрольную работу вкладывается листинг программы, полученный после ее отладки. Система команд микропроцессора приведена в прил.2.

МЕТОДИЧЕКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ КОНТРОЛЬНОЙ РАБОТЫ

Структурная схема микропроцессорного устройства

Типовая структурная схема микропроцессорного устройства показана на рис.1. В микропроцессорное устройство входят следующие основные блоки: микропроцессор (МП),

системный контроллер, генератор тактовых импульсов, постоянное запоминающее устройство (ПЗУ), оперативное запоминающее устройство (ОЗУ), порты ввода и порты вывода.

Блоки, входящие в состав микропроцессорного устройства соединяются между собой посредством шин. В рассматриваемой трехшинной структуре используются: шина адреса (ША) для передачи 16-ти разрядного адреса, шина данных (ШД) для передачи 8-ми разрядного слова данных и шина управления (ШУ) для передачи сигналов управления отдельными блоками.

Функции обработки данных и управления работой блоков микропроцессорного устройства возложены на МП. Он обеспечивает выдачу адресов на ША, выдачу слова на ШД, прием слова с ШД и выдачу сигналов, из которых формируются сигналы управления, поступающие в ШУ.

Для управления работой ПЗУ, ОЗУ и портов ввода-вывода микропроцессор формирует управляющие сигналы, поступающие на системный контроллер. Системный контроллер в свою очередь вырабатывает необходимые управляющие сигналы и выдает их на ШУ. Данные от/к микропроцессора к остальным блокам устройства также поступают через системный контроллер. Таким образом, системный контроллер позволяет организовать двунаправленную передачу данных и их буферизацию.

Для синхронизации во времени процессов, происходящих в блоках микропроцессорного устройства, используется еще одна интегральная схема - генератор тактовых импульсов, который выдает в МП и системный контроллер последовательности синхроимпульсов.

Хранение программы, обеспечивающей функционирование микропроцессорного устройства по заданному алгоритму, исходных данных и результатов осуществляется в памяти устройства. Для хранения промежуточных данных и результатов используется ОЗУ. Слово поступает в ОЗУ по шине данных и записывается в ячейку памяти, которая указывается адресом, поступающим с ША. Режим записи или считывания слова задается сигналами с ШУ.

Для микропроцессорного устройства, управляющего определенным процессом по конкретному алгоритму, программу удобно хранить в ПЗУ, куда она записывается заранее на стадии подготовки микропроцессорного устройства к эксплуатации и не может быть стерта. ПЗУ является энергонезависимой памятью в отличие от ОЗУ, где хранимая информация разрушается при прерывании электропитания. Таким образом, ПЗУ работает только на считывание информации.

Исходные данные могут поступать в микропроцессорное устройство от устройств ввода по специальной команде МП через порты ввода. Необходимый порт ввода выбираются по адресу, поступающему от МП по ША, а по сигналу считывания, выдаваемого на ШУ, информация передается на ШД и в МП.

Результаты операции могут быть выведены на устройства вывода через порты вывода аналогичным образом.

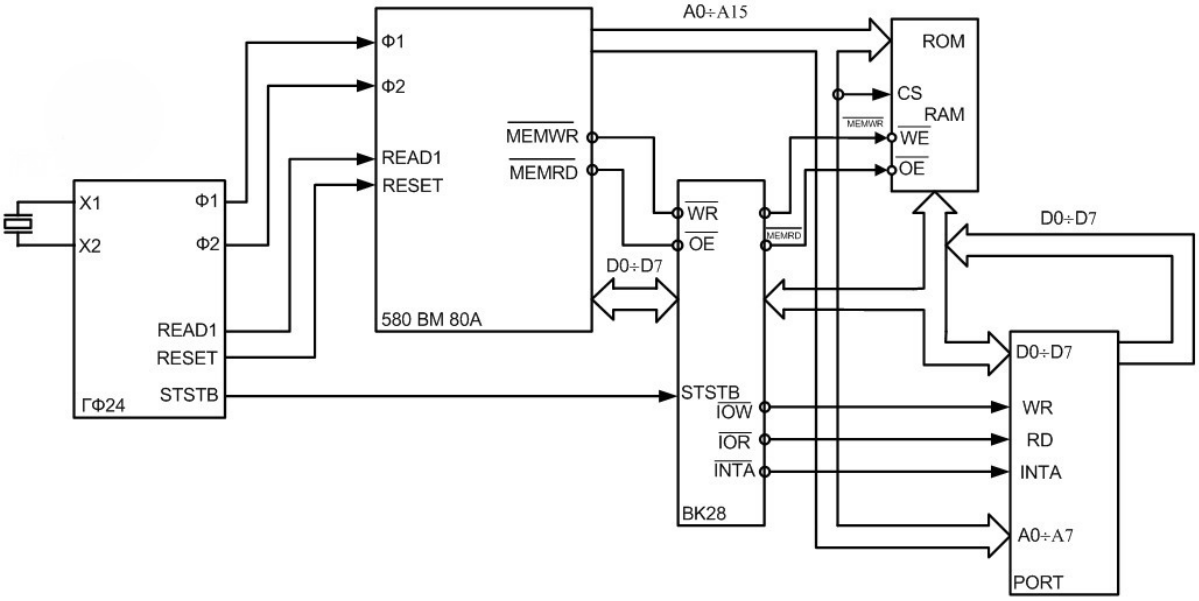


Рисунок 1. Структурная схема микропроцессорного устройства

Программирование микропроцессорного устройства

Любая программа состоит из упорядоченного набора команд. Каждая команда несет в себе информацию, определяющую некоторую элементарную последовательность действий микропроцессорного устройства.

Система команд микропроцессора К580ВМ80А, используемого в контрольной работе для построения микропроцессорной системы приведена в приложении 2.

По назначению они разбиты на отдельные группы:

– команды пересылки данных (пересылка, загрузка, обмен, запись);

– команды арифметических операций (сложение, вычитание, инкремент, декремент);

– команды логических операций («И», «ИЛИ», «НЕ», «Исключающее ИЛИ», сравнение, сдвиги);

– команды ветвлений и переходов;

– команды ввода-вывода и работы со стеком.

Команда может иметь длину один, два или три байта и соответственно занимать в памяти от одной до трех последовательно расположенных ячеек. Форматы команд показаны на рис.2.

Первый байт любой команды определяет код операции, т.е. действия МП. Далее, в зависимости от того требуется ли дополнительная информация для выполнения необходимой команды, могут записываться второй, а иногда и третий байты, представляющие собой данные. Для двухбайтных команд в качестве данных может записываться 8-ми разрядный операнд, с которым далее оперирует микропроцессор или адрес порта ввода-вывода. В качестве второго и третьего байтов для трехбайтовых команд могут использоваться как сами двухбайтовые операнды, так и адреса ячеек памяти, к которым обращается процессор.

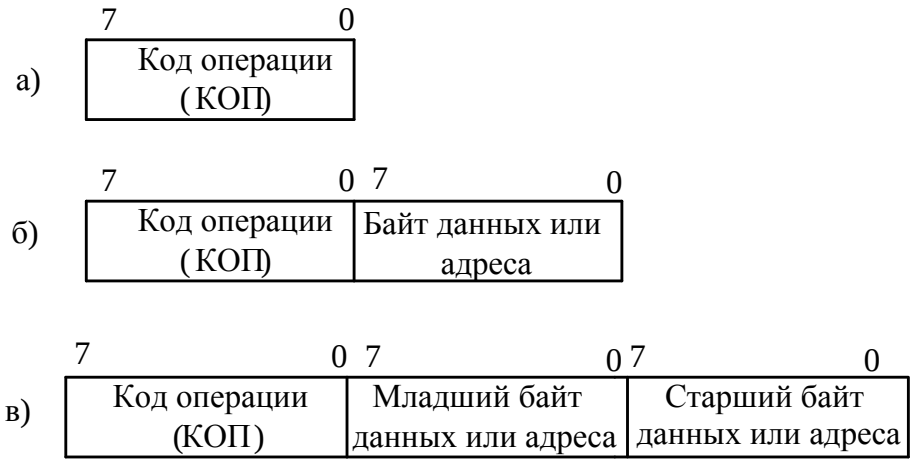


Рисунок 2. Форматы команд: а) однобайтовая; б) двухбайтовая; в) трехбайтовая

При записи команды в двоичной системе счисления (как показано в приложении 2) вероятность ошибки, как в самой команде, так и в адресе, по которому она хранится, увеличивается. Поэтому при написании программ пользуются представлением кодов команд и данных в шестнадцатеричной системе счисления.

Однако запись программы в шестнадцатеричной системе счисления (в машинных кодах) имеет недостаток – пользователь должен помнить машинные коды. Более удобным является программирование на языке Ассемблер.

Любая команда на языке Ассемблер записывается следующим образом:

Метка: Операция Данные; Комментарий

Метка используется для обозначения адреса ячейки памяти, в которой хранится данная команда. Она может состоять из шести символов, не должна включать знаков пунктуации и пробелов, причем первым символом должна быть буква. Метка всегда определяется двоеточием и является необязательным элементом команды. Она применяется только при необходимости.

Операция является обязательным элементом команды. Она представляет собой мнемоническую запись из двух – четырех букв, которые указывают на характер выполняемых действий, например:

HLT – мнемоническое обозначение команды останов МП.

Данные – часть команды, в которой может размещаться одно или два восьмиразрядных слова в зависимости от типа команды (адрес ячейки памяти, адрес порта ввода- вывода, непосредственные данные).

Комментарий отделяется от команды точкой с запятой. Комментарием является запись облегчающая понимание назначения команды. Комментарий является необязательной частью команды. Тем не менее рекомендуется снабжать команды программы комментариями, которые помогают определить роль команды в алгоритме решения задачи.

Пример выполнения контрольной работы

Алгоритм работы устройства управления объектом приведен на рис.3.

Изучение предложенного алгоритма позволяет представить микропроцессорное устройство управления (МУУ) следующими функциями.

МУУ считывает из памяти параметр PAROU (блок 1), Увеличивает значение переменной PAROU на единицу (блок 2) и сравнивает его с допуском DOP, значение которого задано (блок 3). Если PAROU < DOP, то алгоритмом функционирования предусмотрен возврат к блоку 2. Если PAROU > DOP, то значение параметра PAROU уменьшается на 25 (блок 4) и результат

выдается в порт PRT1 на объект управления (блок 5). После чего МУУ переходит к считыванию очередного значения параметра PAROU (блок 1).

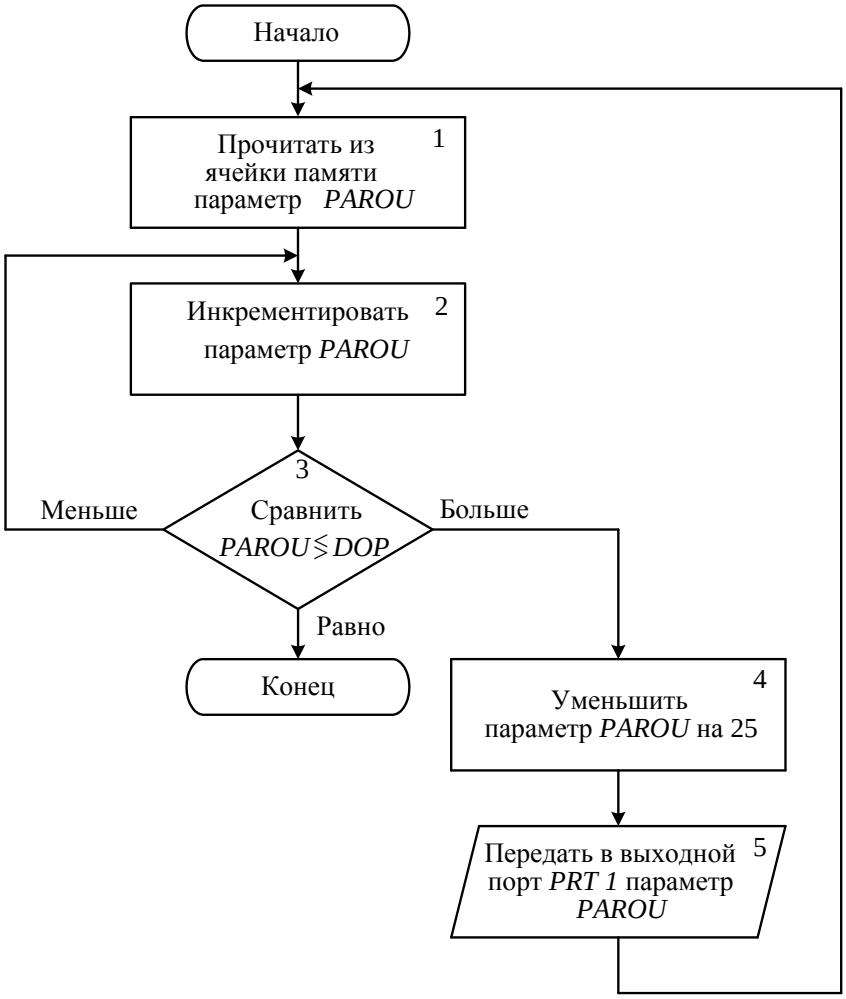


Рисунок 3. Алгоритм работы устройства управления

Процесс управления заканчивается при достижении ситуации, когда PAROU = DOP.

При разработке структурной схемы МУУ в качестве основы используют структурную схему, показанную на рис.1.

Далее студенту следует описать назначение каждого блока структурной схемы и на основании принципов функционирования МУУ и системы команд составить машинный алгоритм работы устройства управления.

Машинный алгоритм показан на рис.4.

Блоком 1 алгоритма осуществляется загрузка пары регистров HL адресом ячейки памяти, в которой хранится параметр PAROU. Блок 2 пересылает содержимое ячейки памяти (параметр PAROU), адрес которой указан в паре регистров HL в регистр аккумулятор. Блок 3 осуществляет инкремент содержимого регистра аккумулятора. Далее в регистровую пару HL заносится адрес ячейки памяти, в которой хранится параметр DOP (блок 4).

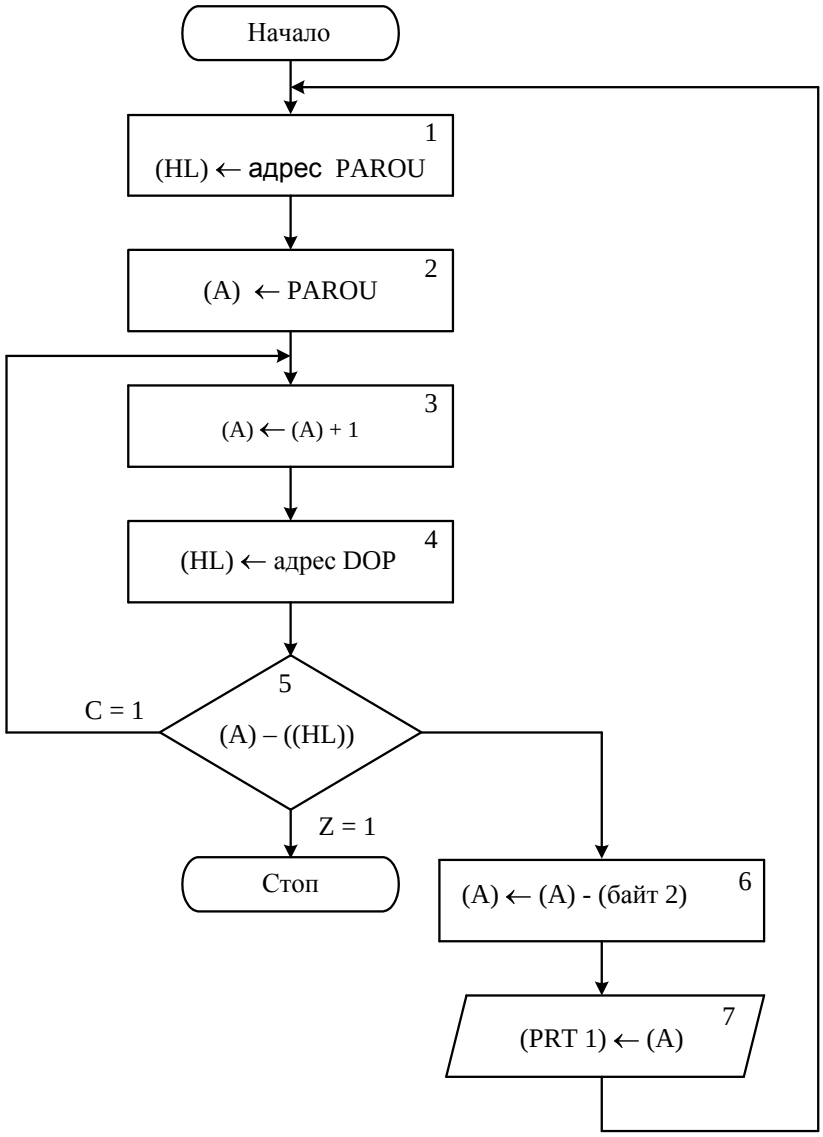


Рисунок 4. Машинный алгоритм функционирования МУУ

Блок 5 осуществляет сравнение содержимого регистра аккумулятора A и содержимого ячейки памяти M, адресуемой парой регистров HL. В ячейке памяти хранится параметр DOP, а в регистре аккумуляторе – параметр PAROU. В результате сравнения содержимого аккумулятора с содержимым ячейки памяти устанавливаются флаги C и Z. Если (A) < (M), то флаг переноса C = 1 и осуществляется переход к блоку 3. Если (A) = (M), то разность (A) – (M) = 0 и устанавливается флаг Z =1 и осуществляется переход к блоку СТОП. Если не выполняется ни одно из рассмотренных условий, осуществляется переход к блоку 6. В результате выполнения операции сравнения содержимое регистра аккумулятора, где хранится параметр PAROU, не изменяется.

Блок 6 осуществляет операцию уменьшения содержимого регистра аккумулятора, т.е. уменьшение значения PAROU на десятичную константу 25 (десятичное число 25 в шестнадцатеричной системе счисления 19H) и результат заносится в аккумулятор. Далее блоком 7 содержимое аккумулятора передается в выходной порт PRT1, и управление передается на блок 1 алгоритма.

В соответствии с рассмотренным алгоритмом составим программу для МУУ (табл.1).

Таблица 1. Программа функционирования МУУ

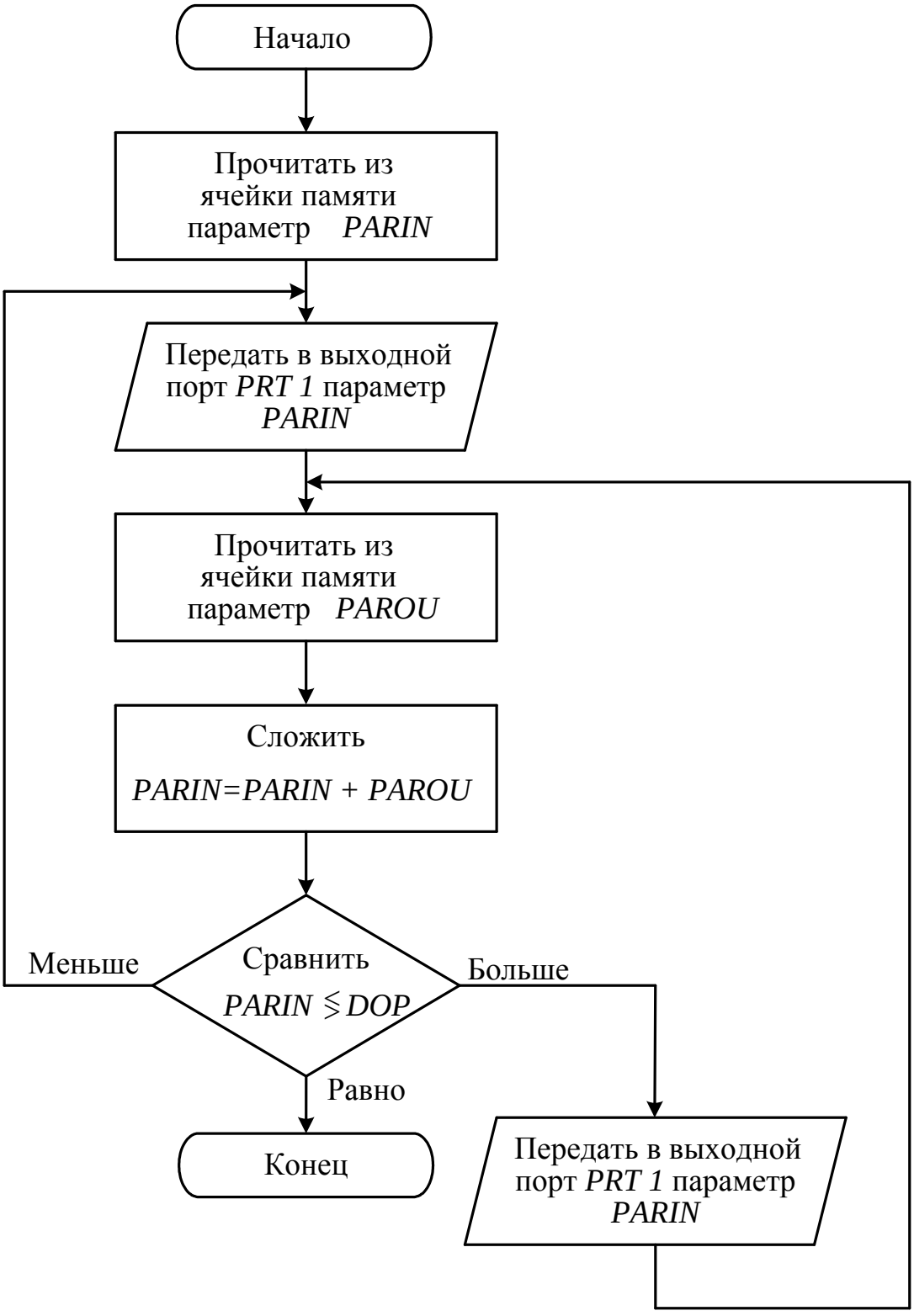
Машинные	**Ассемблер**

коды				
Адрес	**Код**	**Метка**	**Команда**	**Комментарий**
0131	21	BEGIN	LXI H, 0148H	;загрузить в пару регистров HL адрес ячейки памяти, в которой хранится параметр PAROU
0132	48			
0133	01			
0134	7E		MOV A, M	;переслать содержимое ячейки памяти, адрес которой находится в паре регистров HL в регистр аккумулятор
0135	3C	LET1	INR A	;увеличение PAROU на 1
0136	21		LXI 0149H	;загрузка в пару регистров HL адреса ячейки памяти, в которой хранится значение допуска DOP
0137	49			
0138	01			
0139	BE		CMP M	;сравнение содержимого регистра аккумулятора с содержимым ячейки памяти, адрес которой указан в паре регистров HL
013A	CA		JZ STOP	;переход по адресу STOP при равенстве сравниваемых величин
013B	47			
013C	01			
013D	DA		JC LET1	;переход по адресу LET1 при отрицательном результате сравнения
013E	35			
013F	01			
0140	D6		SUI 19H	; уменьшение параметра PAROU на 25
0141	19H			
0142	D3		OUT PORT1	;вывод параметра PAROU в порт PRT1
0143	F0			
0144	C3		JMP BEGIN	;передача управления команде по адресу BEGIN
0145	31			
0146	01			
0147	76	STOP	HLT	;останов
0148	59	PAROU	EQU 59H	;параметр PAROU
0149	54	DOP	EQU 54H	;параметр DOP

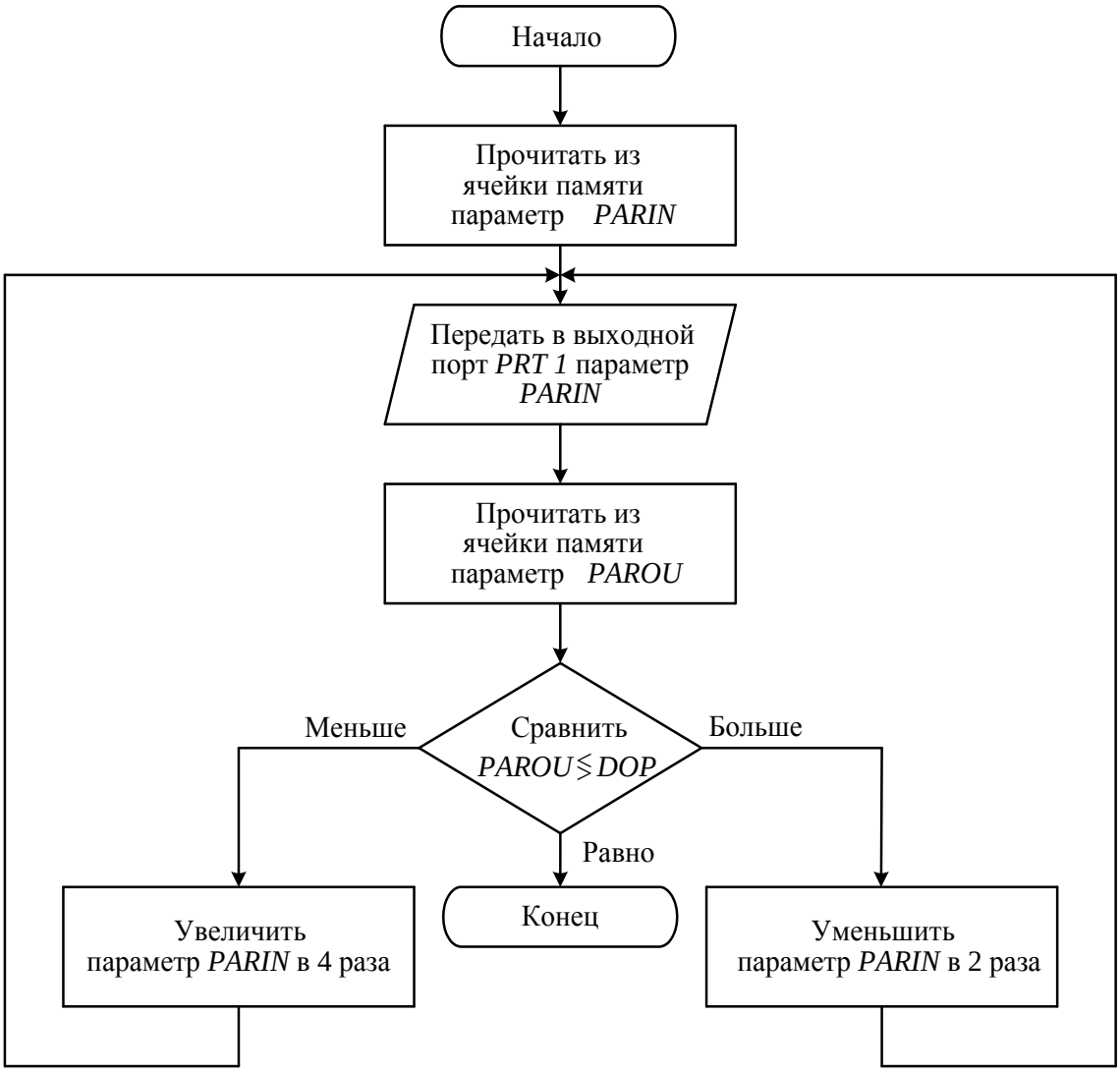
Таким образом, программа функционирования МУУ размещается в 25 ячейках памяти. Ячейка памяти с адресом 0148H используется для хранения параметра PAROU, равного 59H, ячейка с адресом 0149H – для хранения допуска DOP, равного 54H. Адресное пространство памяти, занимаемое программой, определяется адресами 0131H ÷ 0149H. Порту PRT1 присвоен адрес F0H.

ПРИЛОЖЕНИЕ 1
ВАРИАНТЫ ЗАДАНИЯ АЛГОРИТМОВ ФУНКЦИОНИРОВАНИЯ

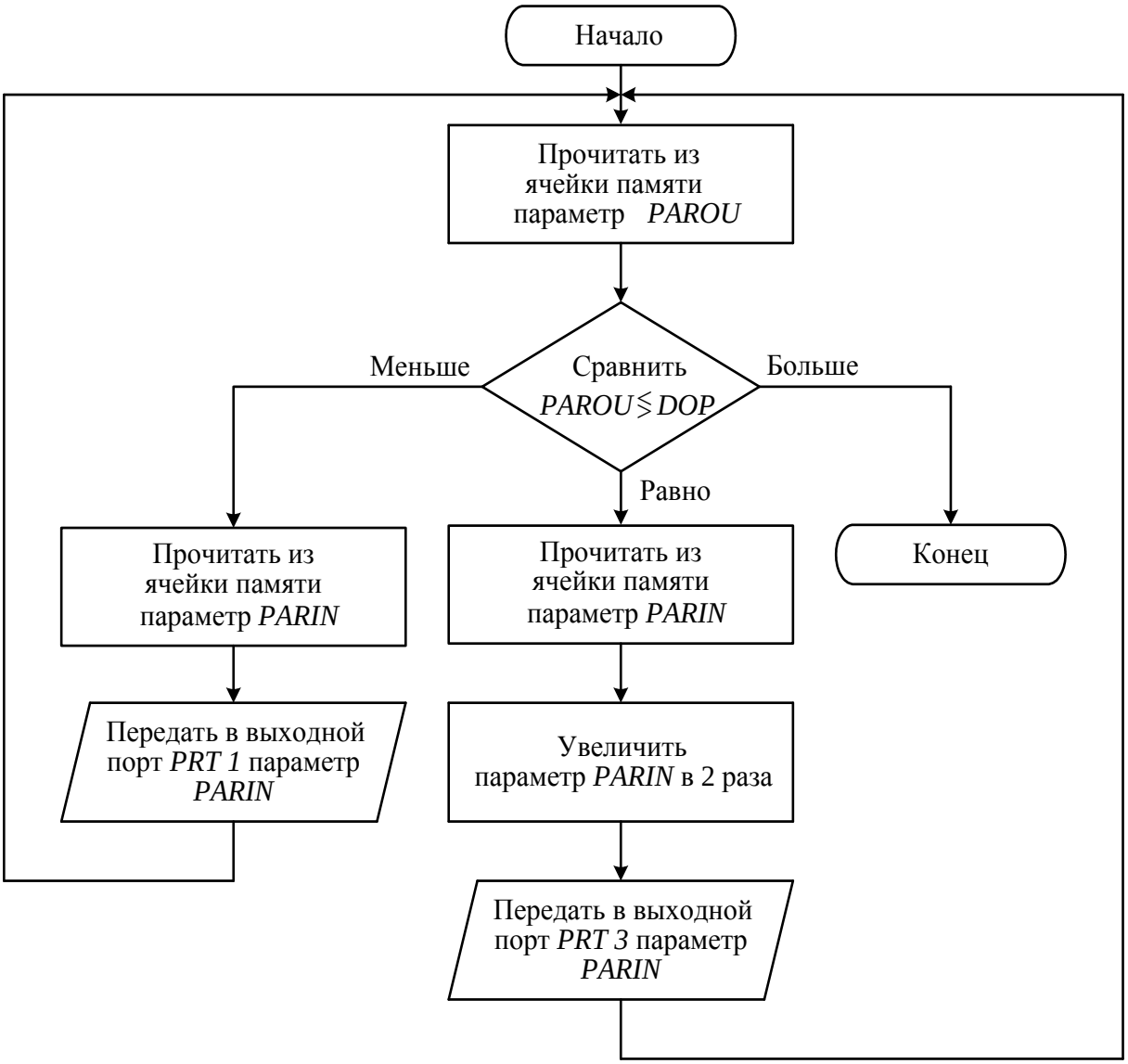
Вариант 0



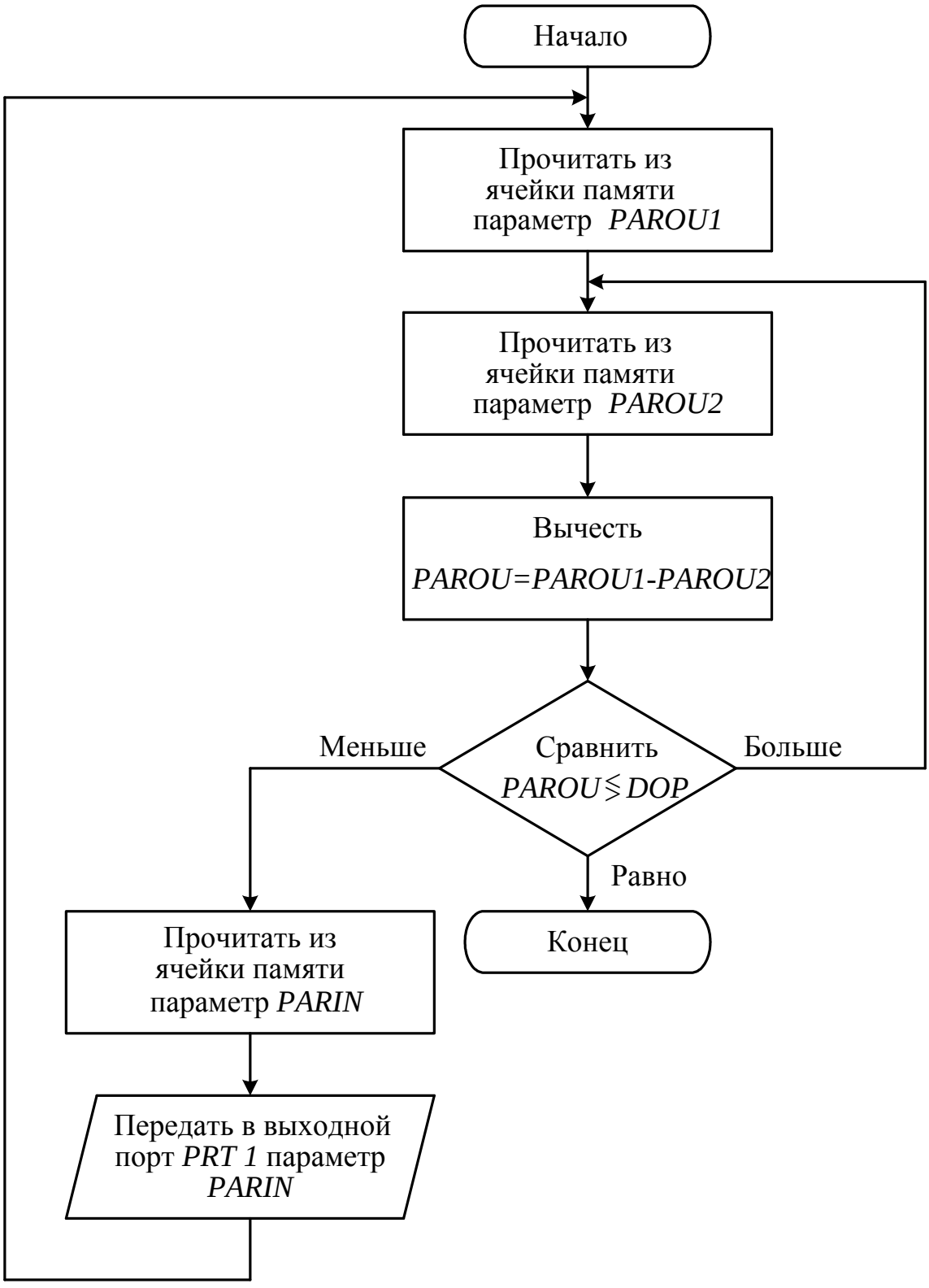
Вариант 1


Начало
Прочитать из ячейки памяти параметр PARIN
Передать в выходной порт PRT 1 параметр PARIN
Прочитать из ячейки памяти параметр PAROU
Сравнить PAROU ⩹ DOP
Меньше
Больше
Равно
Увеличить параметр PARIN в 4 раза
Конец
Уменьшить параметр PARIN в 2 раза

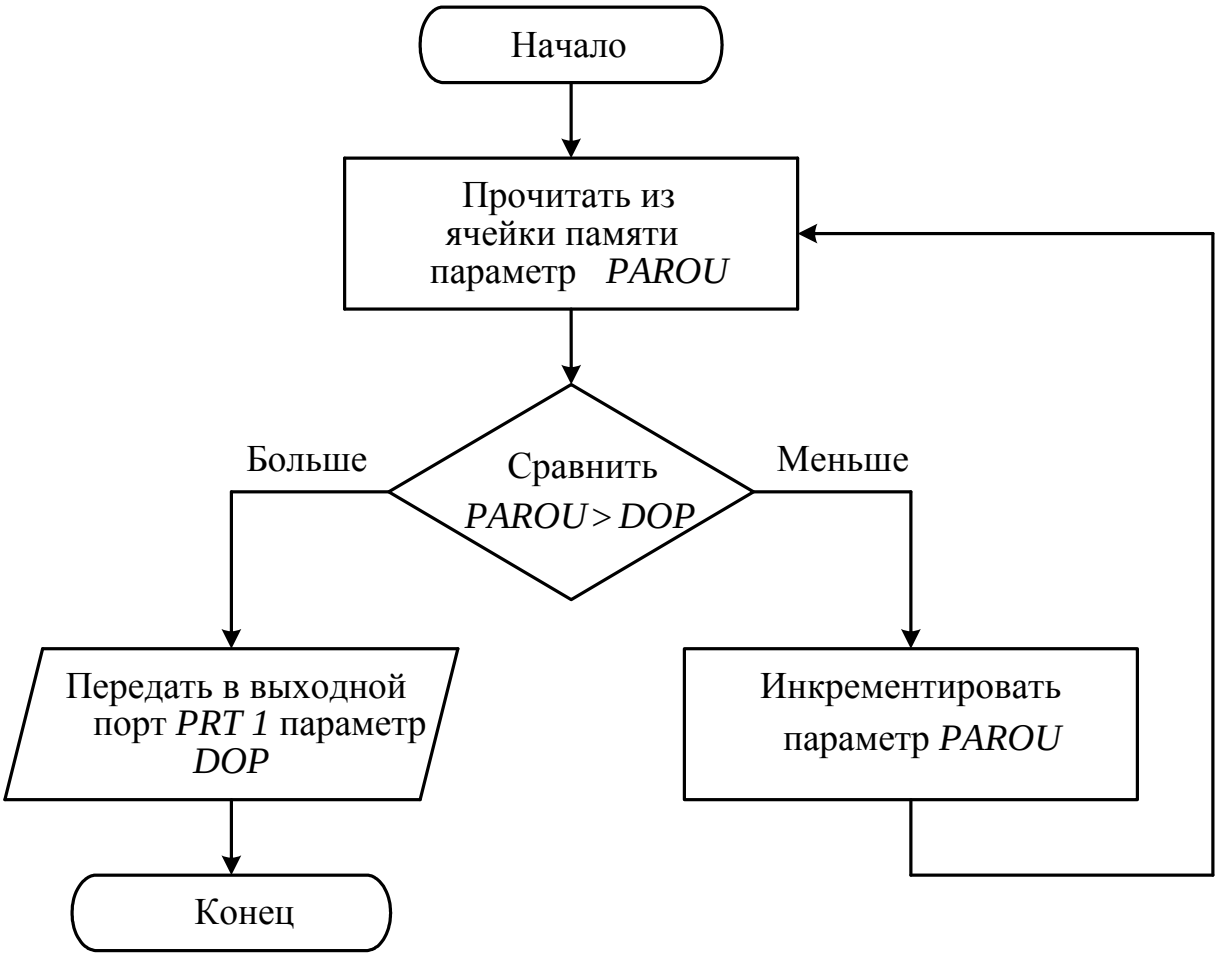
Вариант 2


Начало
Прочитать из ячейки памяти параметр PAROU
Сравнить PAROU ≶ DOP
Меньше
Больше
Равно
Прочитать из ячейки памяти параметр PARIN
Прочитать из ячейки памяти параметр PARIN
Конец
Передать в выходной порт PRT 1 параметр PARIN
Увеличить параметр PARIN в 2 раза
Передать в выходной порт PRT 3 параметр PARIN

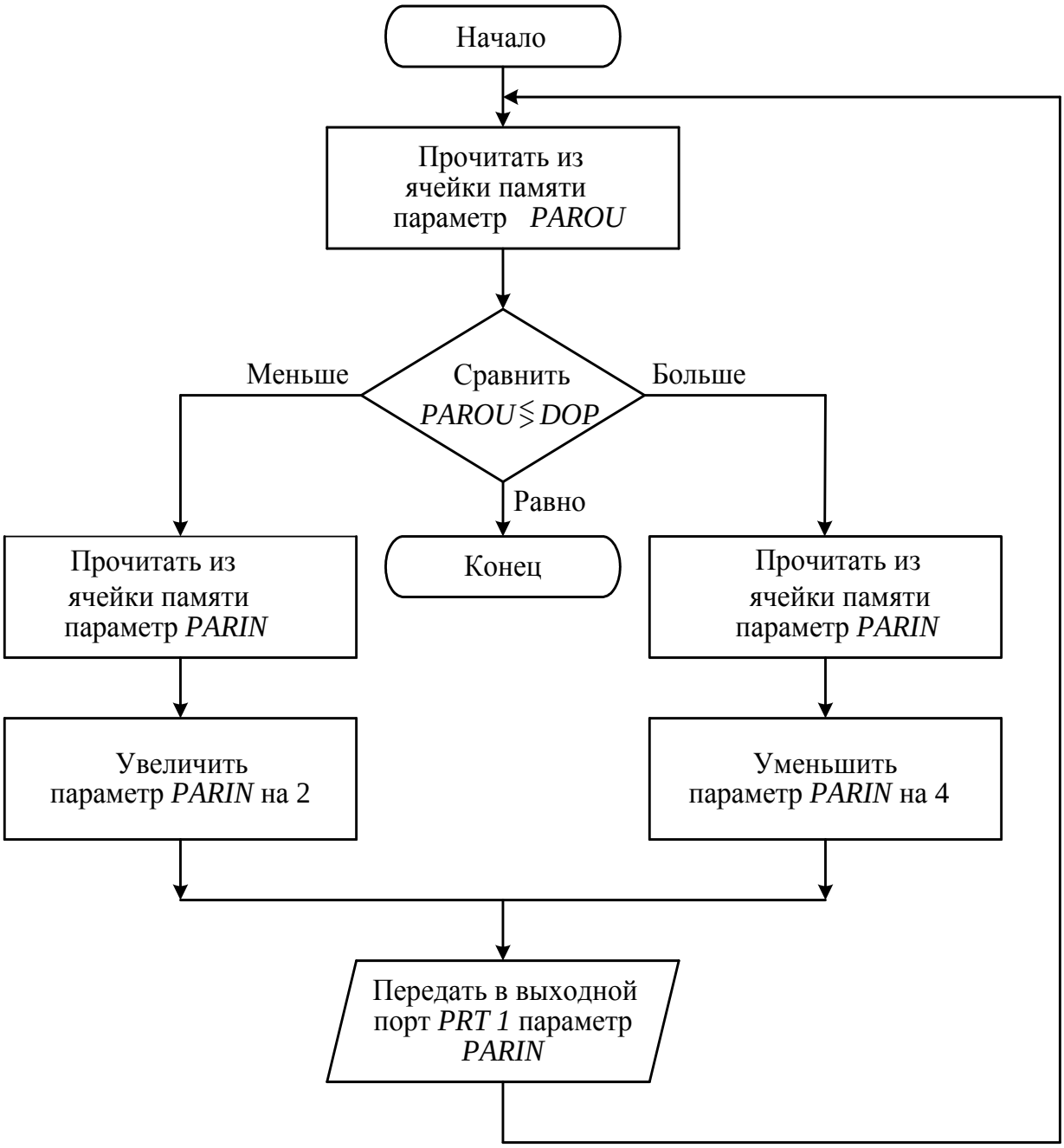
Вариант 3


Начало
Прочитать из ячейки памяти параметр PAROU1
Прочитать из ячейки памяти параметр PAROU2
Вычесть PAROU=PAROU1-PAROU2
Сравнить PAROU ≶ DOP
Меньше
Больше
Равно
Прочитать из ячейки памяти параметр PARIN
Конец
Передать в выходной порт PRT 1 параметр PARIN

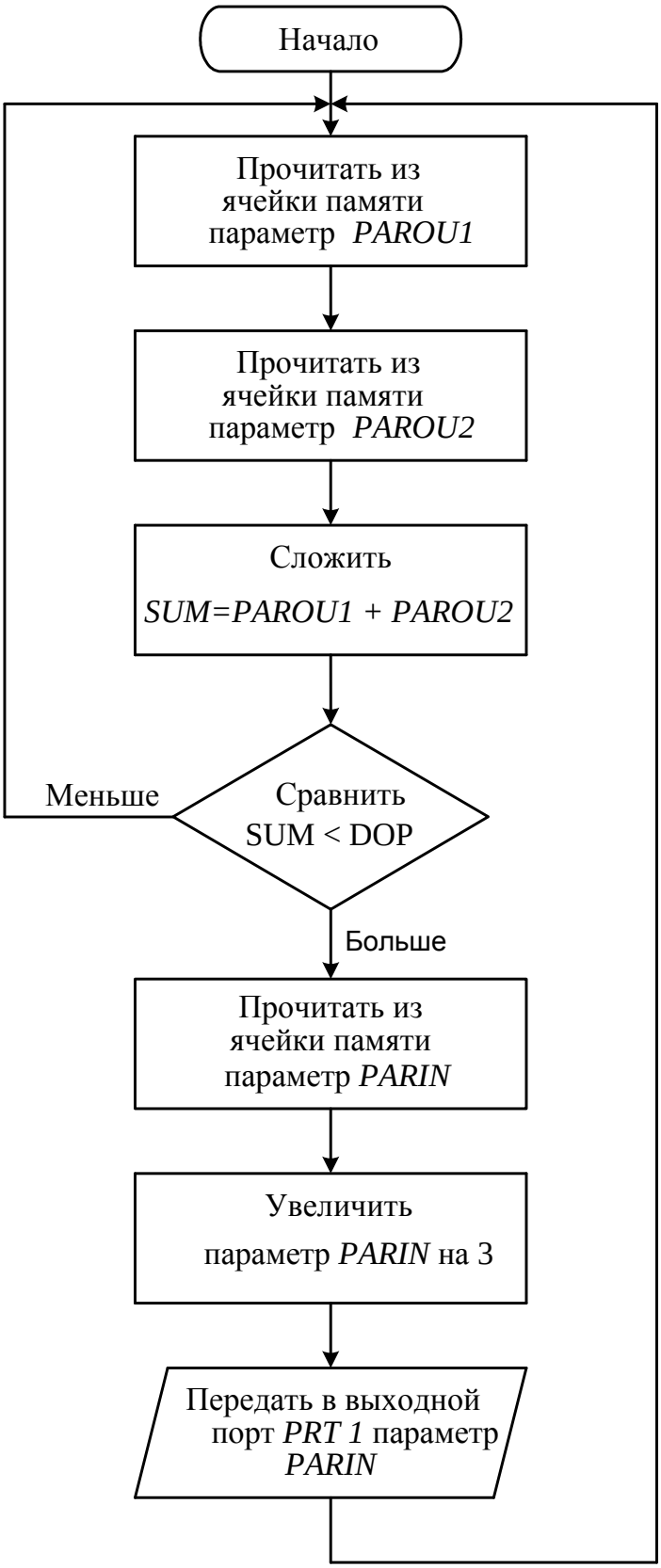
Вариант 4


Начало
Прочитать из ячейки памяти параметр PAROU
Сравнить PAROU > DOP
Больше
Меньше
Передать в выходной порт PRT 1 параметр DOP
Инкрементировать параметр PAROU
Конец

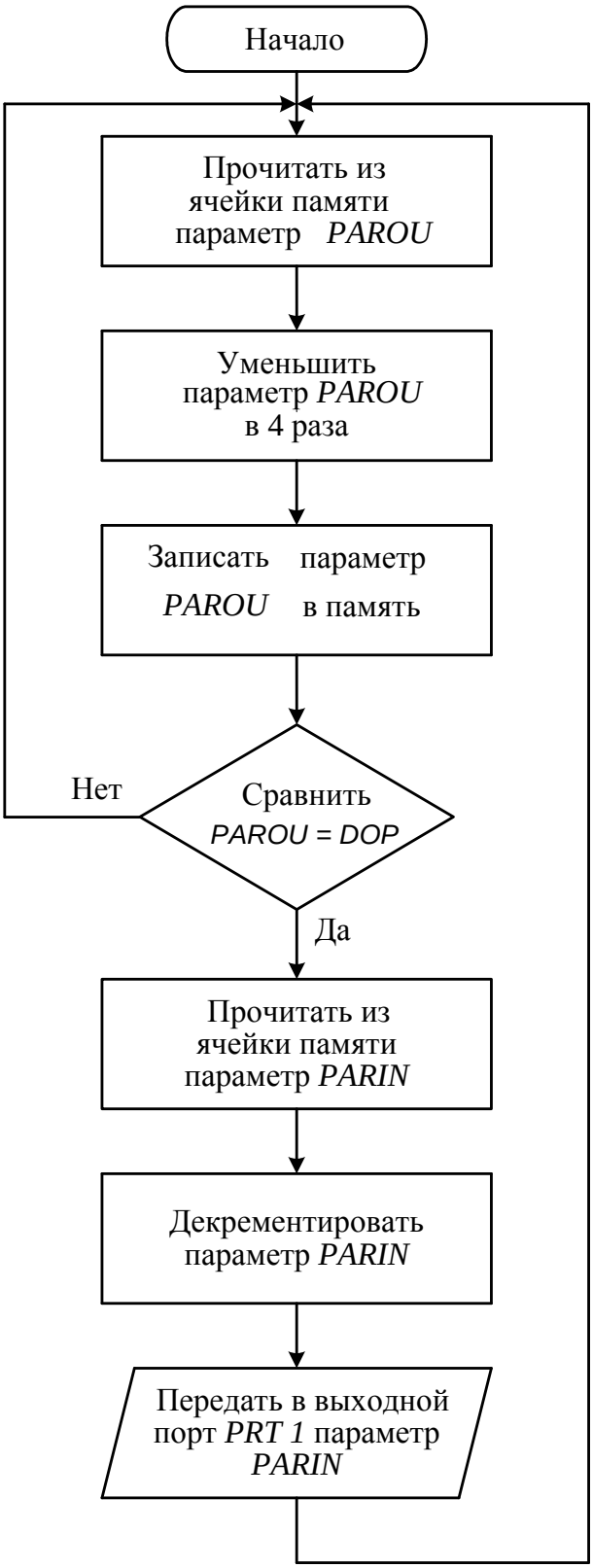
Вариант 5


Начало
Прочитать из ячейки памяти параметр PAROU
Сравнить PAROU ⩽ DOP
Меньше
Равно
Больше
Прочитать из ячейки памяти параметр PARIN
Конец
Прочитать из ячейки памяти параметр PARIN
Увеличить параметр PARIN на 2
Уменьшить параметр PARIN на 4
Передать в выходной порт PRT 1 параметр PARIN

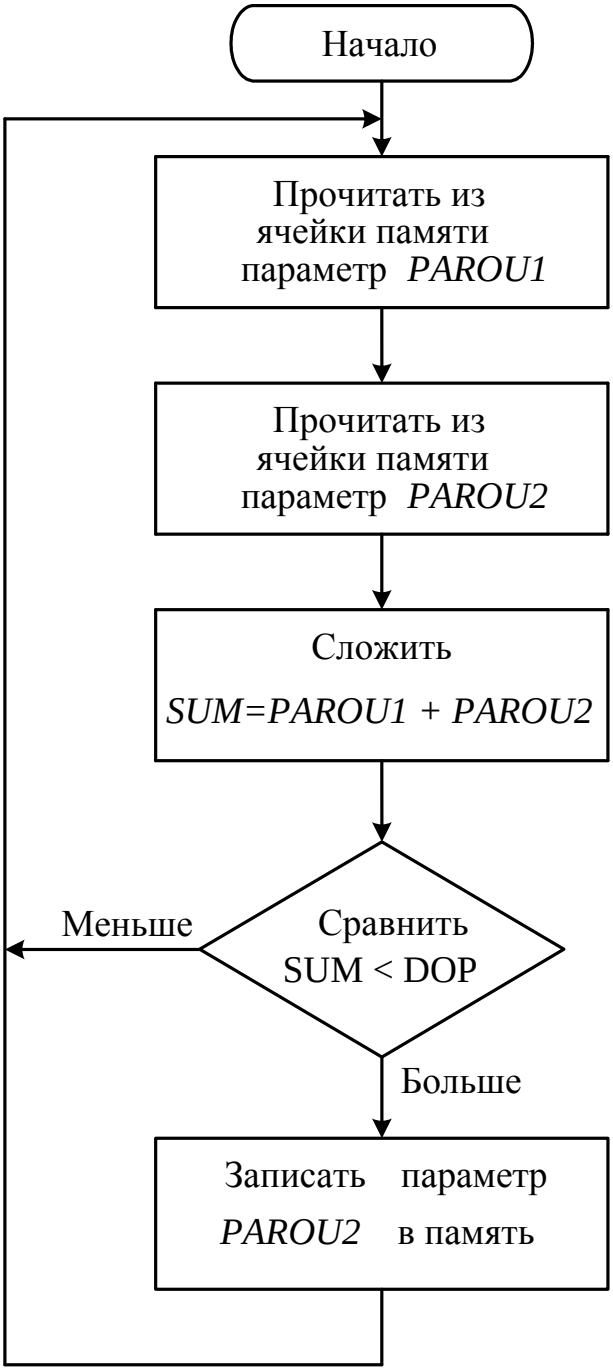
Вариант 6


Начало
Прочитать из ячейки памяти параметр PAROU1
Прочитать из ячейки памяти параметр PAROU2
Сложить
SUM=PAROU1 + PAROU2
Сравнить
SUM < DOP
Меньше
Больше
Прочитать из ячейки памяти параметр PARIN
Увеличить
параметр PARIN на 3
Передать в выходной порт PRT 1 параметр PARIN

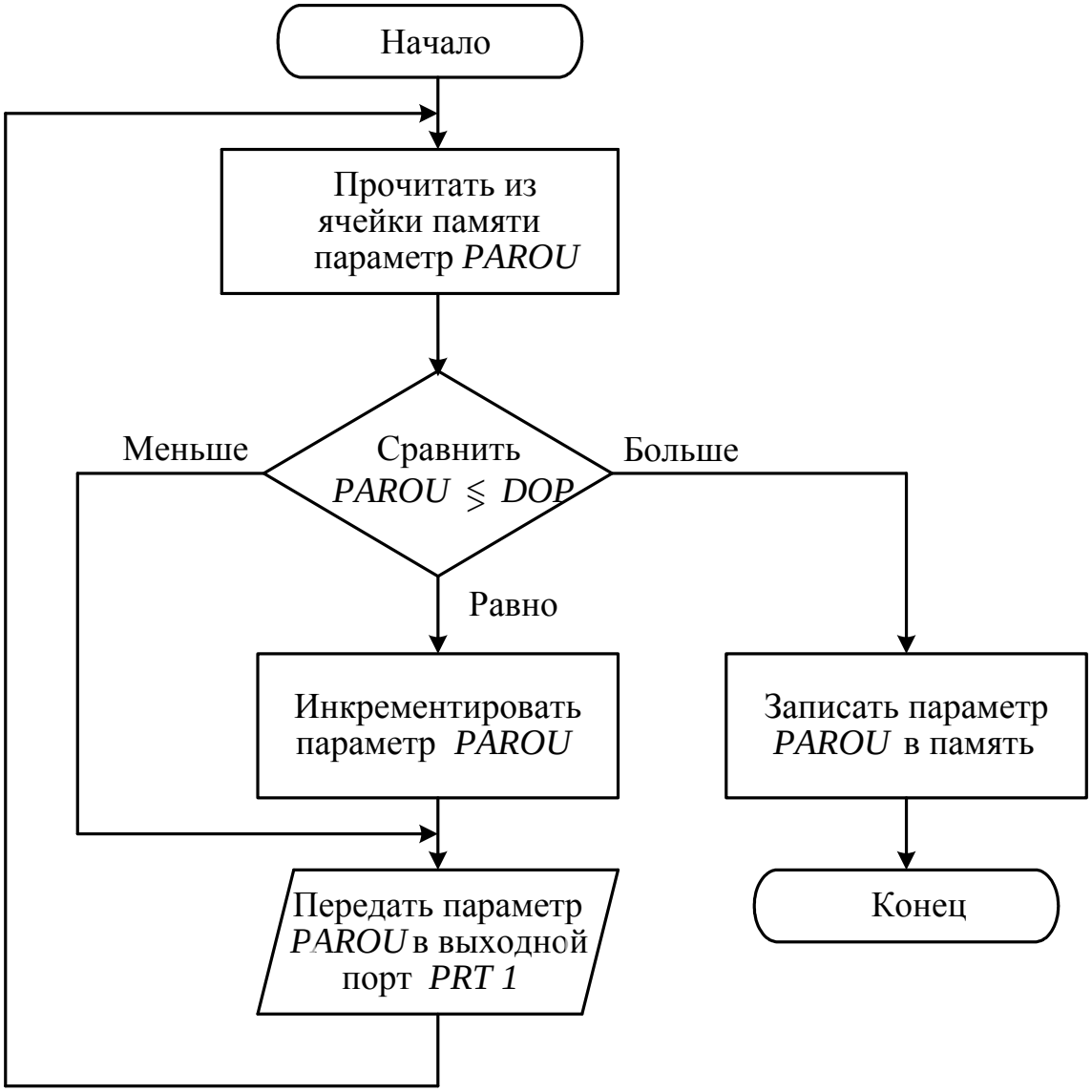
Вариант 7


Начало
Прочитать из ячейки памяти параметр PAROU
Уменьшить параметр PAROU в 4 раза
Записать параметр PAROU в память
Сравнить PAROU = DOP
Нет
Да
Прочитать из ячейки памяти параметр PARIN
Декрементировать параметр PARIN
Передать в выходной порт PRT 1 параметр PARIN

Вариант 8


Начало
Прочитать из ячейки памяти параметр PAROU1
Прочитать из ячейки памяти параметр PAROU2
Сложить
SUM=PAROU1 + PAROU2
Сравнить
SUM < DOP
Меньше
Больше
Записать параметр
PAROU2 в память

Вариант 9


Начало
Прочитать из ячейки памяти параметр *PAROU*
Сравнить *PAROU* ⩽ *DOP*
Меньше
Больше
Равно
Инкрементировать параметр *PAROU*
Записать параметр *PAROU* в память
Передать параметр *PAROU* в выходной порт *PRT 1*
Конец

ПРИЛОЖЕНИЕ 2

СИСТЕМА КОМАНД МИКРОПРОЦЕССОРА К580 ВМ80А

Наименование	Число тактов/циклов	Двоичный код	Название и описание
Команды передачи данных			
MOV R1, R2	5/1	01DDDSSS	Передача между регистрами. (R1) ← (R2). Содержимое регистра R2 передается в регистр R1. Содержимое регистра R2 не меняется.
MOV R, M	7/1	01DDD110	Передача из памяти. (R) ← (M). Содержимое ячейки памяти, адрес которой находится в паре регистров HL, передается в регистр R.
MOV M, R	7/1	01110SSS	Передача в память. (M) ← (R). Содержимое регистра R передается в ячейку памяти, адрес которой находится в паре регистров HL.
MVI R, data	7/2	00DDD110	Непосредственная передача в регистр. (R) ← (байт2). Содержимое байта 2 команды передается в регистр R.
MVI M, data	10/2	00110110	Непосредственная передача в память. (M) ← (байт2). Содержимое байта 2 команды передается в ячейку памяти, адрес которой указан в паре регистров HL.
Загрузка			
LXI RP, data	10/3	00RP0001	Непосредственная загрузка пары регистров. (RH) ← (байт3);(RL) ← (байт2). Байт 3 команды передается в старший регистр (RH) регистровой пары RP, а байт 2 в младший регистр (RL) этой же пары.
LDA addr	13/3	00111010	Прямая загрузка аккумулятора. (A) ← ((байт3)(байт2)). Содержимое ячейки памяти, адрес которой определен байтами 2 и 3 команды, передается в аккумулятор.
LHLD addr	16/3	00101010	Прямая загрузка пары регистров. L ← ((байт3)(байт2)); H ← ((байт3)(байт2)+1). Содержимое ячейки памяти, адрес которой определяется байтами 2 и 3 команды передается в регистр L. Содержимое следующей ячейки памяти передается в регистр H.
LDAX RP	7/1	00RP1010	Косвенная загрузка аккумулятора. (A) ← (M). Содержимое ячейки памяти, адрес которой определяется парой регистров RP, передается в аккумулятор. Могут использоваться только пары RP=B (регистры B и C) или RP=D (регистры D и E).
XCHG	4/1	11101011	Обмен. (H) ↔ (D); (L) ↔ (E). Содержимое регистров H и L обмениваются с содержимым регистров D и E.
STA addr	13/3	00110010	Прямое размещение содержимого аккумулятора. ((байт3)(байт2)) ← (A). Содержимое аккумулятора передается в ячейку памяти, адрес которой указан байтами 2 и 3 команды.
SHLD addr	16/3	00110010	Прямое размещение содержимого регистров. ((байт3)(байт2)) ← (L); ((байт3)(байт2)+1) ← (H). Содержимое регистра L передается в ячейку памяти, адрес которой определяется байтами 2 и 3 команды, одержимое регистра H передается в следующую ячейку памяти.

Наименование	Число тактов/циклов	Двоичный код	Название и описание
STAX RP	7/1	00RP0010	Косвенная загрузка аккумулятора. (M) ← (A). Содержимое аккумулятора передается в ячейку памяти, адрес которой содержится в паре регистров RP. Могут использоваться только пары RP=B (регистры B и C) или RP=D (регистры D и E).
Арифметические команды			
ADD R	4/1	10000SSS	Сложение содержимого регистра. (A) ← (A)+(R). Содержимое регистра R складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
ADD M	7/2	10000110	Сложение содержимого ячейки памяти. (A) ← (A)+((H)(L)). Содержимое ячейки памяти, адрес которой указан в паре регистров HL, складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
ADI data	7/2	11000110	Непосредственное сложение. (A) ← (A)+(байт2). Содержимое байта 2 команды складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
ADC R	4/1	10001SSS	Сложение содержимого регистра и переноса. (A) ← (A)+(R)+(C). Содержимое регистра R и флага переноса C (бит переполнения) складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
ADC M	7/2	10001110	Сложение содержимого ячейки памяти и переноса. (A) ← (A)+((H)(L))+(C). Содержимое ячейки памяти, адрес которой указан в паре регистров HL, и флага переноса C складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
ACI data	7/2	11001110	Непосредственное сложение с учетом переноса. (A) ← (A)+(байт2)+(C). Содержимое байта 2 команды и флага переноса C складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
DAD RP	10/3	00RP1101	Сложение содержимого пары регистров с содержимым пары регистров HL. (H)(L) ← (H)(L)+(RH)(RL). Содержимое пары регистров RP складывается с содержимым пары регистров HL. Устанавливается флаг C. C=1, если есть перенос при сложении с удвоенной точностью, C=0, если нет переноса.
DAA	4/1	00100111	Десятичная коррекция аккумулятора: 1. Если $(A_3 \ldots A_0) > 9$ или AC = 1, то (A)+6; 2. Если $(A_7 \ldots A_4) > 9$ или C = 1, то $(A_7 \ldots A_4)+6$
SUB R	4/1	10010SSS	Вычитание содержимого регистра (A) ← (A)-(R). Содержимое регистра R вычитается из содержимого аккумулятора. Результат помещается в аккумулятор.

Наименование	Число тактов/циклов	Двоичный код	Название и описание
			Устанавливаются флаги – Z, S, C, P, AC.
SUB M	7/2	10010110	Вычитание содержимого ячейки памяти. (A) ← (A)-(M). Содержимое ячейки памяти, адрес которой указан в паре регистров HL, вычитается из содержимого аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
SUI data	7/2	11010110	Непосредственное вычитание. (A) ← (A)-(байт2). Содержимое байта 2 команды вычитается из содержимого аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
SBB R	4/1	10011SSS	Вычитание содержимого регистра и переноса (A) ← (A)-(R)-(C). Содержимое регистра R и флага переноса C вычитается из содержимого аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги; Z, S, C, P, AC.
SBB M	7/2	10011110	Вычитание содержимого ячейки памяти и переноса. (A) ← (A)-(M)-(C). Содержимое ячейки памяти, адрес которой указан в паре регистров HL, и флага переноса C вычитается из содержимого аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC
SBI data	7/2	11011110	Непосредственное вычитание данных и переноса. (A) ← (A)-(байт2)-(C). Содержимое байта 2 команды и индикатора переноса C вычитается из содержимого аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, C, P, AC.
INR R	5/1	00DDD100	Инкремент содержимого регистра. (R) ← (R)+1. Содержимое регистра увеличивается на 1. Устанавливаются флаги – Z, S, P, AC.
INR M	10/3	00110100	Инкремент содержимого ячейки памяти. (M) ← ((H)(L))+1. Содержимое ячейки памяти, адрес которой указан в паре регистров HL, увеличивается на 1. Устанавливаются флаги – Z, S, P, AC
INX RP	5/1	00RP0011	Инкремент содержимого пары регистров. (RH)(RL) ← (RH)(RL)+1. Содержимое пары регистров RP увеличивается на 1
DCR R	5/1	00DDD101	Декремент содержимого регистра. (R) ← (R)-1. Содержимое регистра уменьшается на 1. Устанавливаются флаги – Z, S, P, AC.
DCR M	10/3	00110101	Декремент содержимого ячейки памяти (M) ← (M)-1. Содержимое ячейки памяти, адрес которой указан в паре регистров HL, уменьшается на 1. Устанавливаются флаги – Z, S, P, AC.
DCX RP	5/1	00RP1011	Декремент содержимого пары регистров (RH)(RL) ← (RH)(RL)-1. Содержимое пары регистров RP уменьшается на 1.
Логические команды			

Наименование	Число тактов/циклов	Двоичный код	Название и описание
ANA R	4/1	10100SSS	«И» с содержимым регистра. (A) ← (A)∧(R). Содержимое регистра R и содержимое аккумулятора логически умножается. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P, AC. C=0
ANA M	7/2	10100110	«И» с содержимым ячейки памяти. (A) ← (A)∧(M). Содержимое ячейки памяти, адрес которой указан в паре регистров HL и содержимое аккумулятора логически умножается. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P, AC. C = 0
ANI data	7/2	11100110	«И» непосредственно с данными. (A) ← (A)∧(байт2). Содержимое байта 2 команды и содержимое аккумулятора логически умножается. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P, AC. C = 0.
XRA R	4/1	10101SSS	«Исключающее ИЛИ» с содержимым регистра (A) ← (A)⊕(R). Исключающее ИЛИ выполняется с содержимым регистра R и содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P. AC=0, C=0.
XRA M	7/2	10101110	«Исключающее ИЛИ» с содержимым ячейки памяти. (A) ← (A)⊕(M). Исключающее ИЛИ выполняется с содержимым ячейки памяти, адрес которой указан в паре регистров HL и содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P. AC=0, C=0.
XRI data	7/2	11101110	«Исключающее ИЛИ» непосредственно с данными. (A) ← (A)⊕(байт2). Исключающее ИЛИ выполняется с содержимым байта 2 команды и содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P. AC=0, C=0.
ORA R	4/1	10110SS	«ИЛИ» с содержимым регистра. (A) ← (A)∨(R). Содержимое регистра R логически складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P. AC=0, C=0.
ORA M	7/2	10110110	«ИЛИ» с содержимым ячейки памяти. (A) ← (A)∨(M). Содержимое ячейки памяти, адрес которой указан в паре регистров HL логически складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P. AC=0, C=0.
ORI data	7/2	11110110	«ИЛИ» непосредственно с данными. (A) ← (A)∨(байт2). Содержимое байта 2 команды логически складывается с содержимым аккумулятора. Результат помещается в аккумулятор. Устанавливаются флаги – Z, S, P. AC=0, C=0.
CMP R	4/1	10111SSS	Сравнить содержимое регистра. (A)-(R). Содержимое

Наименование	Число тактов/циклов	Двоичный код	Название и описание
			регистра R вычитается из содержимого аккумулятора. Содержимое аккумулятора не изменяется. Флаги – Z=1, если (A)=(R), C=1, если (A)<(R).
CMP M	7/2	10111110	Сравнить содержимое ячейки памяти. (A)-(M). Содержимое ячейки памяти, адрес которой указан парой регистров HL, вычитается из содержимого аккумулятора. Содержимое аккумулятора не изменяется. Флаги – Z=1, если (A)=(M), C=1, если (A)<(M).
CPI data	7/2	11111110	Непосредственно сравнить данные. (A)-(байт2). Содержимое байта 2 команды вычитается из содержимого аккумулятора. Содержимое аккумулятора не изменяется. Флаги – Z=1, если (A)=(байт2), C=1, если (A)<(байт2).
RLC	4/1	00000111	Сдвиг влево. (A) ← (2A). Содержимое аккумулятора сдвигается на один разряд влево ($A_0 \leftarrow A_7$, $C \leftarrow A_7$, $A_{n+1} \leftarrow A_n$). Устанавливается флаг C.
RRC	4/1	00001111	Сдвиг вправо. (A) ← (A/2). Содержимое аккумулятора сдвигается на один разряд вправо ($A_7 \leftarrow A_0$, $C \leftarrow A_0$, $A_n \leftarrow A_{n+1}$). Устанавливается флаг C.
RAL	4/1	00010111	Циклический сдвиг влево. Содержимое аккумулятора сдвигается влево вместе с C ($C \leftarrow A_7$, $A_0 \leftarrow C$ $A_{n+1} \leftarrow A_n$). Устанавливается флаг C.
RAR	4/1	00011111	Циклический сдвиг вправо Содержимое аккумулятора сдвигается вправо вместе с C ($A_7 \leftarrow C$, $C \leftarrow A_0$, $A_n \leftarrow A_{n+1}$). Устанавливается флаг C.
CMA	4/1	00101111	Инвертировать содержимое аккумулятора. $(A) \leftarrow (\overline{A})$.Содержимое аккумулятора инвертируется.
CMC	4/1	00111111	Инвертировать флаг переноса. $(C) \leftarrow (\overline{C})$. Инвертируется флаг переноса.
STC	4/1	00110111	Установить перенос. (C) ← 1. Флаг переноса устанавливается в 1.
Команды ветвлений и переходов			
JMP addr	10/3	11000011	Ветвление. (PC) ← (байт3)(байт2). Управление передается команде, адрес которой указан в байтах 3 и 2 текущей команды.
Jcnd addr	10/3	11cnd010	Условное ветвление. Если (cnd), то (PC) ← ((байт3)(байт2). Если условие выполняется, то управление передается команде, адрес которой указан в байтах 2 и 3 текущей команды, иначе – выполняется следующая команда программы.
CALL addr	17/5	11001101	Вызов подпрограммы. (M_0) ← (PCH), (адрес (M_0) содержится в (SP)-1); (M_1) ← (PCL), (адрес (M_1) содержится в (SP)-2); (SP) ← ((SP)-2); (PC) ← (байт3)байт2).
Ccnd addr	17/5	11cnd100	Условный вызов подпрограммы. Если условие выполняется, то действия те же, что и в команде CALL,

Наименование	Число тактов/циклов	Двоичный код	Название и описание
			иначе – выполняется следующая команда программы.
RET	10/3	11001001	Возврат из полпрограммы. (M_0) ← (PCL) (адрес (M_0) содержится в (SP)); (M_1) ← (PCH) (адрес (M_1) содержится в (SP)+1); (SP) ← (SP)+2.
Rcnd	11/3	11cnd000	Условный возврат из подпрограммы. Если условие выполняется, то действия те же, что и в RET, иначе – выполняется следующая команда программы
RST N	11/3	11NNN111	Рестарт. (M_0) ← (PCH), (адрес (M_0) содержится в (SP)-1); (M_1) ← (PCL), (адрес (M_1) содержится в (SP)-2); (SP) ← ((SP)-2); (PC) ← (NNN×8).
PCHL	5/1	11101001	Косвенный переход по адресу, указанному в паре регистров HL. (PCH) ← (H); (PCL) ← (L).
Команды ввода/вывода, управления и работы со стеком			
IN port	10/3	11011011	Ввести данные. (A) ← (port). Данные из порта, адрес которого указан в байте 2 команды записываются в аккумулятор.
OUT port	10/3	11010011	Вывести данные. (port) ← (A). Данные из аккумулятора записываются в порт, адрес которого указан в байте 2 команды.
PUSH RP	11/3	11RP0101	Загрузить в стек содержимое пары регистров. (M_0) ← (RH), (адрес (M_0) содержится в (SP)-1); (M_1) ← (RL), (адрес (M_1) содержится в (SP)-2)); (SP) ← ((SP)-2).
PUSH PSW	11/3	11110101	Загрузить в стек содержимое регистра флагов. (M_0) ← (A); (адрес (M_0) содержится в (SP)-1); (M_1) ← PSW; (адрес (M_1) содержится в (SP)-2)); (SP) ← ((SP)-2).
POP RP	10/3	11RP0001	Считать из стека содержимое пары регистров. (RL) ← (M_0); (адрес (M_0) содержится в (SP)); (RH) ← (M_1); (адрес (M_1) содержится в (SP)+1)); (SP) ← ((SP)+2).
POP PSW	10/3	11110001	Считать из стека содержимое регистра флагов. (PSW) ← (M_0); (адрес (M_0) содержится в (SP)); (A) ← (M_1); (адрес (M_1) содержится в (SP)+1)); (SP) ← ((SP)+2).
XTHL	18/5	11100011	Обмен содержимым верхушки стека и пары регистров HL. (L) ↔ (M_0); (адрес (M_0) содержится в (SP)); (H) ↔ (M_1); (адрес (M_1) содержится в (SP)+1));
SPHL	5/1	11111001	Пересылка содержимого регистров HL в указатель стека. (SP) ← (HL).
EI	4/1	11111011	Разрешение прерываний после выполнения следующей команды.
DI	4/1	11110011	Запрещение прерываний после выполнения следующей команды.
HLT	7/1	01110110	Останов. Процессор останавливается.
NOP	4/1	00000000	Нет операций. Не выполняется никаких операций.

АДРЕСА РЕГИСТРОВ

Адрес (R)	Регистр
111	A
000	B
001	C
010	D
011	E
100	H
101	L

АДРЕСА ПАРЫ РЕГИСТРОВ

Адрес (RP)	Пара регистров
00	BC
01	DE
10	HL
11	SP

КОДЫ УСЛОВИЙ (cnd), ИСПОЛЬЗУЕМЫХ В КОМАНДАХ УСЛОВНОГО ПЕРЕХОДА, ВЫЗОВА ПОДПРОГРАММ И ВОЗВРАТА ИЗ НИХ

Коды	Мнемоника	Условия
000	NZ	Не ноль (Z = 0)
001	Z	Ноль (Z = 1)
010	NC	Нет переноса (C = 0)
011	C	Есть перенос (C = 1)
100	P0	Нечетный результат (P = 0)
101	PE	Четный результат (P = 1)
110	P	Результат положительный (S = 0)
111	M	Результат отрицательный (S = 1)