ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Физические методы контроля»

ПРОГРАММИРУЕМЫЕ ЦИФРОВЫЕ УСТРОЙСТВА

Методические рекомендации к практическим занятиям для студентов направления подготовки 12.03.04 «Биотехнические системы и технологии» дневной формы обучения

Могилев 2018

УДК 621.377:004
ББК 32.85:32.973-018
П 78

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Физические методы контроля» «12» января 2018 г., протокол № 6

Составитель канд. техн. наук, доц. А. А. Афанасьев

Рецензент канд. техн. наук, доц. И. В. Лесковец

В методических рекомендациях рассмотрены вопросы проектирования и программирования цифровых электронных устройств, принципы их работы, приведены примеры решения типовых задач, задачи для самостоятельной работы. Составлены в соответствии с рабочей программой по дисциплине «Программируемые цифровые устройства» для студентов направления подготовки 12.03.04 «Биотехнические системы и технологии».

Учебно-методическое издание

ПРОГРАММИРУЕМЫЕ ЦИФРОВЫЕ УСТРОЙСТВА

Ответственный за выпуск	С. С. Сергеев
Технический редактор	С. Н. Красовская
Компьютерная верстка	Н. П. Полевничая

Подписано в печать . Формат 60×84/16. Бумага офсетная. Гарнитура Таймс.
Печать трафаретная. Усл. печ. л. . Уч.-изд. л. . Тираж 16 экз. Заказ №

Издатель и полиграфическое исполнение:
Государственное учреждение высшего профессионального образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 24.01.2014.
Пр. Мира, 43, 212000, Могилев.

Содержание

1 Электронные элементы аппаратной поддержки однокристальных микроЭВМ и схемы их подключения 4
2 Программирование микропроцессорных устройств на ассемблере…... 14
3 Разработка программного обеспечения на С++..... 16
3.1 Среда программирования CodeBlocks, её запуск и создание проекта.....16
3.2 Разработка программного обеспечения на С++..... 17
4 Разработка программного обеспечения цифровых измерительных приборов с встроенными микроконтроллерами на С++.....30
5 Разработка программируемых цифровых устройств на основе микроконтроллеров и их программного обеспечения..... 41
Список литературы.....46
Приложение А..... 47

1 Электронные элементы аппаратной поддержки однокристальных микроЭВМ и схемы их подключения

Задача 1. Разработать электрическую принципиальную схему подключения к микропроцессору (МП) серии MCS-51 (тип задан) указанного регистра для хранения младшего байта адреса. Исходные данные к задаче 1 приведены в таблице 1.1.

Таблица 1.1 – Исходные данные к задаче 1

Номер варианта	Тип микросхемы МП	Тип микросхемы регистра
0	КР1816ВЕ31	К555ИР22
1	8051АН	К531ИР22П
2	КР1816ВЕ51	1533ИР37
3	8031АН	1533ИР22
4	КР1830ВЕ31	К555ИР23
5	80С51ВН	К561ИР6
6	КР1830ВЕ51	КР580ИР82
7	80С31ВН	К1533ИР33
8	АТ89С51	1533ИР33
9	КМ1816ВЕ751	533ИР22

Пример решения задачи 1. Для временного хранения младшего байта адреса используем восьмиразрядный регистр К555ИР22, схема включения которого приведена на рисунке 1.1.

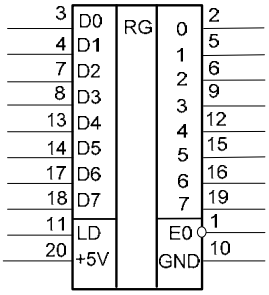

Рисунок 1.1 – Условное обозначение на схемах регистра К555ИР22

Через входы регистра D0…D7 из порта P0 МП в каждом машинном цикле записывается младший байт адреса. Его выдача из МП сопровождается формированием синхросигнала высокого уровня ALE на 30-м выводе МП. Этот сигнал надо подать на вход LD регистра, т. к. при появлении сигнала высокого уровня на этом входе будет разрешена операция записи в него числа. Записанное в регистр число (младший байт адреса) появляется на его выходах 0…7 при низком уровне сигнала на входе EO. Схема подключения регистра К555ИР22 к МП показана на рисунке 1.2.

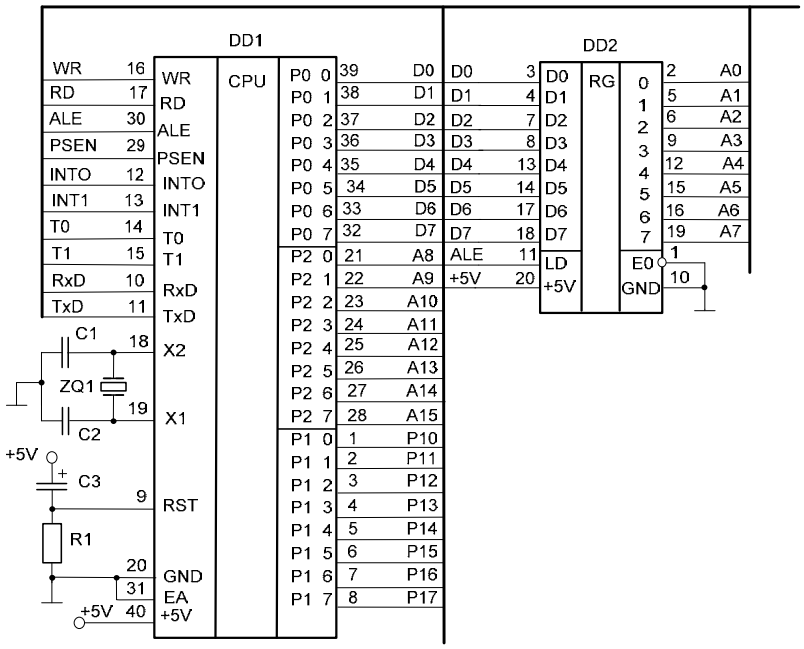

Рисунок 1.2 – Схема подключения регистра К555ИР22 к МП

Задача 2. Разработать электрическую принципиальную схему подключения к микропроцессору серии MCS-51 указанной микросхемы оперативного запоминающего устройства (ОЗУ). Составить таблицу адресов первой и последней ячеек подключенного к МП ОЗУ в двоичной и шестнадцатеричной формах. Исходные данные к задаче 2 приведены в таблице 1.2.

Таблица 1.2 – Исходные данные к задаче 2

Номер варианта	Тип микросхемы МП	Тип микросхемы ОЗУ
0	8051АН	КР537РУ10
1	КР1816ВЕ51	НМ64256
2	8031АН	КР537РУ17
3	КР1830ВЕ31	НМ64128
4	80С51ВН	КР537РУ10
5	КР1830ВЕ51	НМ64256
6	80С31ВН	КР537РУ17
7	АТ89С51	НМ64128
8	КМ1816ВЕ751	КР537РУ10
9	КР1816ВЕ31	НМ64256

Пример решения задачи 2. Рассмотрим подключение к МП микросхемы НМ62256, имеющей 32768 ячеек для записи в них восьмиразрядных двоичных чисел. На схемах её изображают следующим образом (рисунок 1.3).

Назначение её выводов:

– VCC – питание +5 В;

– GND – общий вывод;

– A0...A14 – адресные входы;

– D0…D7 – входы-выходы данных;
– OE – вход «разрешение чтения» (активный уровень – низкий);
– WE – вход «разрешение записи» (активный уровень – низкий);
– CE – вход «выбор микросхемы» (активный уровень – низкий).

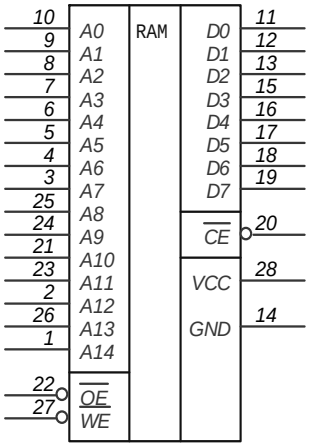

Рисунок 1.3 – Условное обозначение на схемах микросхемы ОЗУ НМ62256

Руководствуясь этими справочными данными, подключение делаем так, как показано на рисунке 1.4.

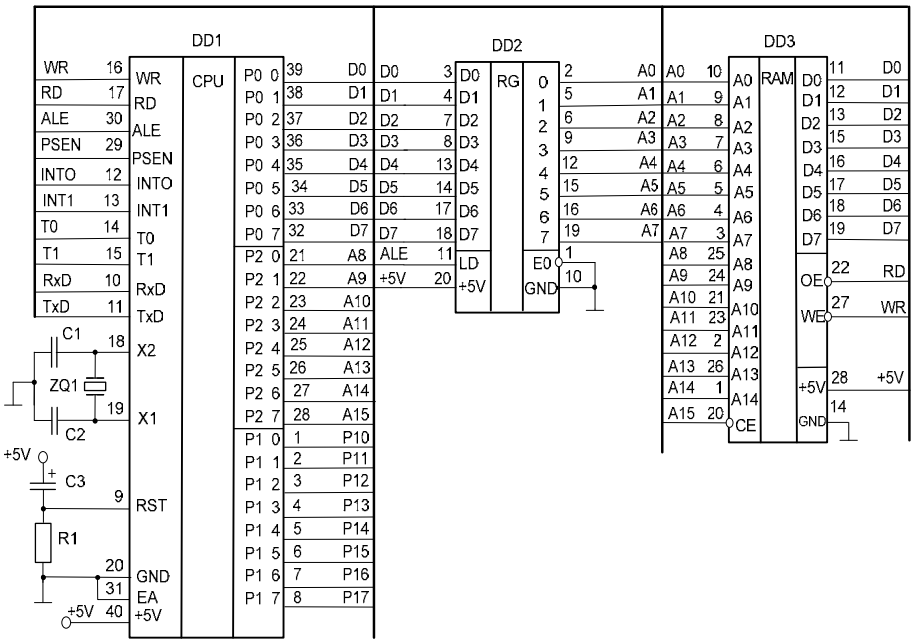

Рисунок 1.4 – Пример подключения к МП микросхемы ОЗУ НМ62256

Задача 3. Разработать электрическую принципиальную схему подключения к микропроцессору серии MCS-51 (тип задан) указанной микросхемы постоянного запоминающего устройства (ПЗУ, ППЗУ). Составить таблицу адресов первой и последней ячеек подключенного к МП ПЗУ в десятичной, двоичной и шестнадцатеричной формах. Исходные данные приведены в таблице 1.3.

Таблица 1.3 – Исходные данные к задаче 3

Номер варианта	Тип микросхемы МП	Тип микросхемы ПЗУ(ППЗУ)
0	КР1816ВЕ51	КР573РФ2
1	8031АН	К558РР2
2	КР1830ВЕ31	К573РФ6
3	80С51ВН	К558РР3
4	КР1830ВЕ51	К573РФ5
5	80С31ВН	К1609РР1
6	АТ89С51	27512
7	КМ1816ВЕ751	К573РФ4
8	КР1816ВЕ31	К1611РР1
9	8051АН	К573РФ7

Пример решения задачи 3. Рассмотрим подключение к МП микросхемы К573РФ4, имеющей 8192 ячейки для записи и хранения в них 8-разрядных двоичных чисел. На схемах её изображают следующим образом (рисунок 1.5). Назначение выводов приведено в таблице 1.4. Руководствуясь этими справочными данными, подключение делаем следующим образом (рисунок 1.6).

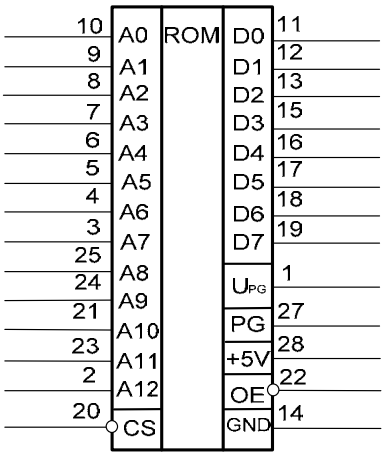

Рисунок 1.5 – Условное обозначение на схемах микросхемы ПЗУ К573РФ4

Таблица 1.4 − Назначение выводов микросхемы К573РФ4

Вывод	Назначение	Обозначение
2,3…10, 21, 23, 24, 25	Адресные входы	A12, A7…A0, A10, A11, A9, A8
11…13, 15…19	Входы-выходы данных	D0…D2, D3…D7
20	Выбор микросхемы	CS
22	Разрешение по выходу	OE
27	Сигнал программирования	PR
28	Напряжение питания	Ucc
1	Напряжение программирования	Upr
14	Общий	0 В

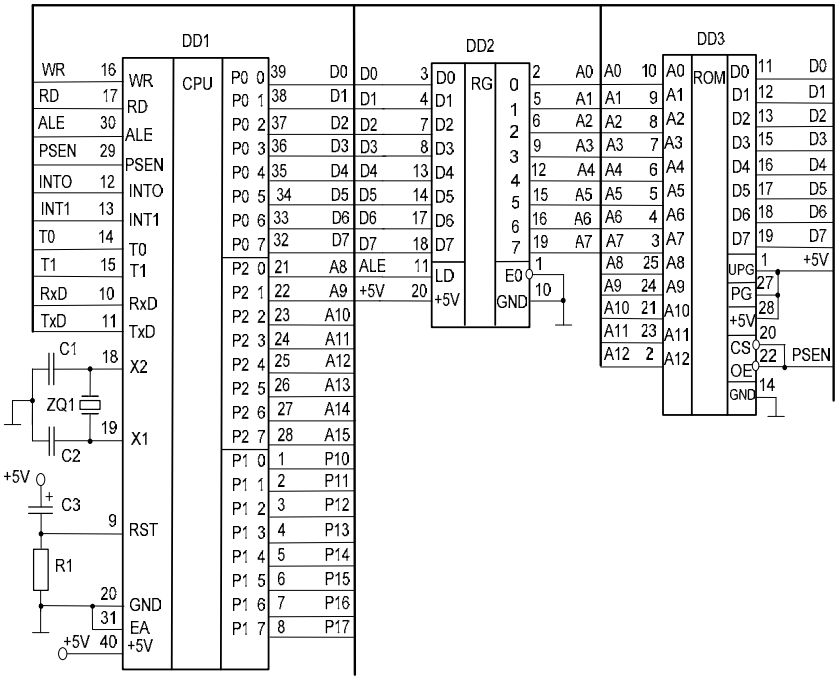

Рисунок 1.6 – Пример подключения к МП микросхемы ППЗУ К573РФ4

Задача 4. Разработать электрическую принципиальную схему подключения к микропроцессору серии MCS-51 (тип задан) указанной микросхемы аналогово-цифрового преобразователя (АЦП). Исходные данные приведены в таблице 1.5.

Таблица 1.5 – Исходные данные к задаче 4

Номер варианта	Тип микросхемы МП	Тип микросхемы АЦП
0	КР1830ВЕ31	AD7994
1	80С51ВН	К572ПВ4
2	КР1830ВЕ51	AD7417
3	80С31ВН	AD7994
4	AT89C51	К572ПВ4
5	КМ1816ВЕ751	AD7417
6	КР1816ВЕ31	AD7994
7	8051АН	К572ПВ4
8	КР1816ВЕ51	AD7417
9	8031АН	К572ПВ4

Пример решения задачи 4. Рассмотрим подключение к МП микросхемы К572ПВ4, имеющей восемь входов встроенного аналогового коммутатора, через который аналоговые сигналы от датчиков поочередно подаются на вход АЦП. Результаты преобразования записываются во встроенное ОЗУ, состоящее из восьми 8-разрядных ячеек. Циклическая работа коммутатора начинается сразу после подачи на микросхему питающего напряжения. Каждый раз, когда

заканчивается преобразование в очередном подключенном канале, обновляется информация в ячейке ОЗУ, соответствующей этому каналу.

На схемах микросхему К572ПВ4 изображают следующим образом (рисунок 1.7). Назначение выводов приведено в таблице 1.6.

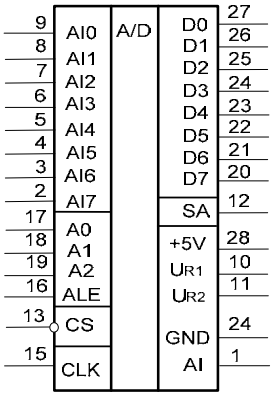

Рисунок 1.7 – Условное обозначение на схемах микросхемы АЦП К572ПВ4

АЦП может работать в следующих режимах:

− однополярном положительном ($U_{вх}$ = 0…2,5 В при U_{R1} = +2,5 В и U_{R2} = 0 В);

− однополярном отрицательном ($U_{вх}$ = −2,5…0 В при U_{R1} = 0 В и U_{R2} = −2,5 В);

− биполярном ($U_{вх}$ = −1,25…+1,25 В при U_{R1} = +1,25 В и U_{R2} = −1,25 В).

Для выбора режима необходимо только подать соответствующие опорные напряжения. Аналоговые сигналы могут подаваться одновременно по восьми входам AI0…AI7 или по одному входу AI.

В течение всего периода преобразования результаты хранятся в ячейках ОЗУ АЦП.

Таблица 1.6 − Назначение выводов микросхемы К572ПВ4

Вывод	Номер вывода	Назначение вывода
AI0…AI7, AI	1…9	Аналоговые входы
U_{R1}, U_{R2}	10, 11	Входы для подачи опорных напряжений
A0…A2	17…19	Адресные входы
C	15	Вход тактовых импульсов
ALE	16	Вход управления при обращении к ОЗУ АЦП
CS	13	Выбор микросхемы
D0…D7	20…27	Информационные выходы
SA	12	Выходной сигнал, информирующий о начале преобразования

После каждого цикла преобразования содержимое каждой ячейки ОЗУ обновляется. Считывание информации из ОЗУ может осуществляться в произвольном порядке. Адресные входы определяют внутренние двоичные адреса ячеек ОЗУ. Номер канала и номер ячейки ОЗУ, в которой хранится результат преобразования, совпадают. Для считывания сигналы подаются на адресные входы A0, A1, A2 при наличии сигнала высокого уровня на входе ALE. В момент перехода сигнала на входе ALE из единицы в ноль во внутреннем регистре адреса фиксируется номер выбранной ячейки ОЗУ и её содержимое

появляется на выходах D0…D7 при CS = 0. Номер выбранной ячейки сохраняется в течение времени, пока сигнал ALE = 0.

Руководствуясь этими справочными данными, подключение делаем так, как показано на рисунке 1.8.

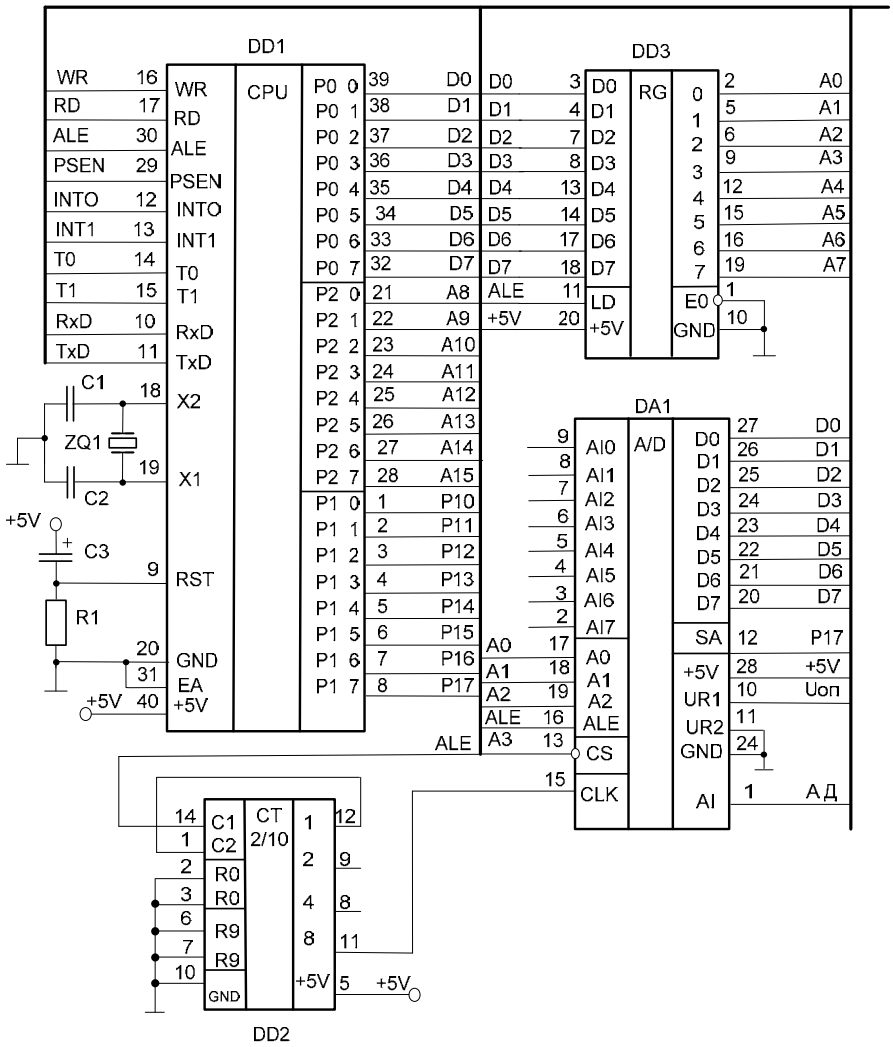

Рисунок 1.8 – Пример подключения к МП микросхемы АЦП КР572ПВ4

Задача 5. Разработать электрическую принципиальную схему подключения к микропроцессору (МП) серии MCS-51 (тип задан) цифровых индикаторов указанного типа. Исходные данные приведены в таблице 1.7.

Пример 1 решения задачи 5. Рассмотрим подключение к МП микросхемы жидкокристаллического индикатора (ЖКИ) WM–1602N. Он имеет две строки по 16 символов. Индикатор работает под управлением встроенного контроллера. Условное обозначение микросхемы приведено на рисунке 1.9.

Назначение выводов у этой микросхемы следующее:

– VCC – питание +5 В;

– GND – общий вывод;

– Contrast – вход для регулировки контрастности изображения символов на индикаторе;

– RS – вход выбора регистра: «0» – регистр команд; «1» – регистр данных;

– RW – выбор режима записи или чтения;
– E – вход разрешения чтения/записи;
– DB0…DB7 – выводы для передачи данных.

Таблица 1.7 – Исходные данные к задаче 5

Номер варианта	Тип микросхемы МП	Тип микросхемы индикатора	Число разрядов ЦОУ
0	80С51ВН	WM-C1602N	
1	КР1830ВЕ51	АЛС324Б	3
2	80С31ВН	СС56-21EWA	8
3	AT89C51	E40561	4
4	КМ1816ВЕ751	ТОТ-3361АН-1N	9
5	КР1816ВЕ31	WM-C1602N	
6	8051АН	E40561	8
7	КР1816ВЕ51	СС56-21EWA	8
8	8031АН	ТОТ-3361АН-1N	9
9	КР1830ВЕ31	WM-C1602N	

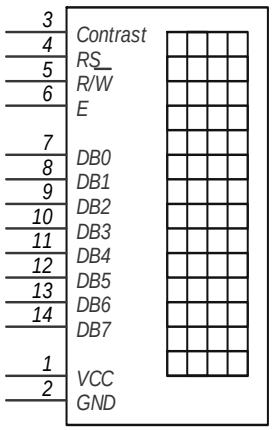

Рисунок 1.9 – Условное обозначение на схемах микросхемы ЖКИ WM-C1602N

Руководствуясь этими справочными данными, подключение делаем следующим образом (рисунок 1.10).

Пример 2 решения задачи 5. Рассмотрим подключение к МП цифробуквенных светодиодных семисегментных индикаторов, например, двух 4-разрядных микросхем E40561. На рисунке 1.11 показано расположение сегментов индикатора и их размеры. Электрическая схема микросхемы и номера выводов E40561 приведены на рисунке 1.12. Руководствуясь справочными данными, подключение делаем следующим образом (рисунок 1.13).

Задача 6. Определите в соответствии с рисунком 1.13 системный адрес регистра DD3 и запишите его в двоичной и шестнадцатеричной формах.

Задача 7. Составьте таблицу состояния выходов микросхемы DD6 (см. рисунок 1.13) в зависимости от сигналов, подаваемых на его входы.

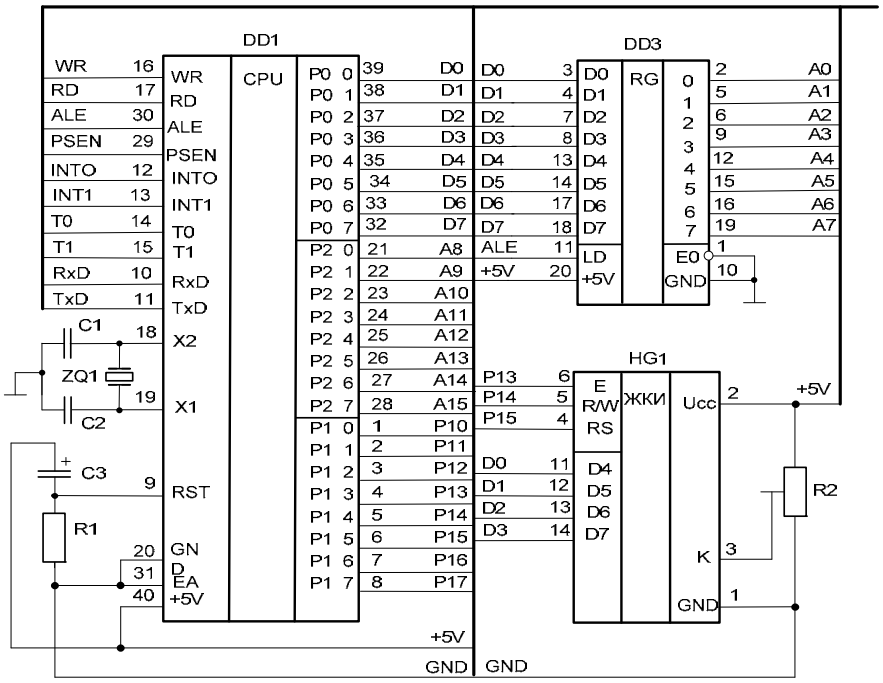

Рисунок 1.10 – Пример подключения к МП микросхемы ЖКИ WM-C1602N

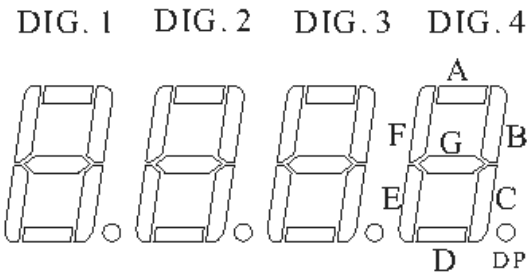

Рисунок 1.11 – Расположение сегментов микросхемы E40561

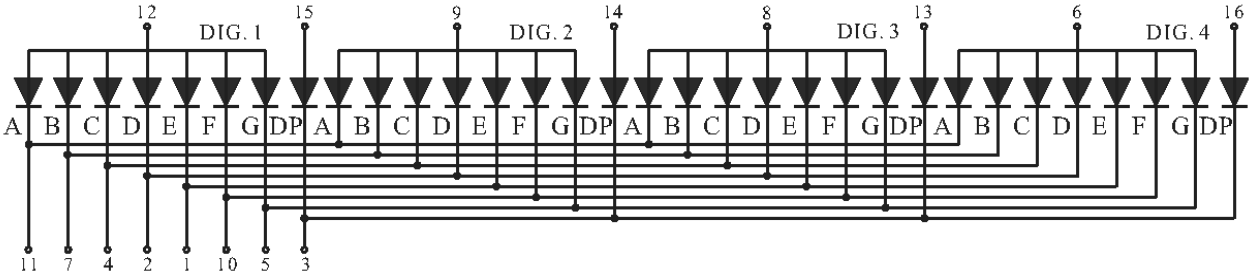

Рисунок 1.12 – Электрическая схема микросхемы E40561

Задача 8. Составьте таблицу выходных сигналов микросхемы DD3 (см. рисунок 1.13), обеспечивающих отображение символов от 0 до 9 на индикаторах HG1, HG2. Запишите полученные двоичные коды символов в шестнадцатеричной форме. Почему выходные сигналы микросхемы DD3 одновременно подаются на сегменты всех индикаторов? Как обеспечивается отображение символа, двоичный код которого записан в регистр DD3, в нужном разряде цифрового отсчетного устройства (например, в разряде ANODE3 HG1 или в разряде ANODE1 HG2)?

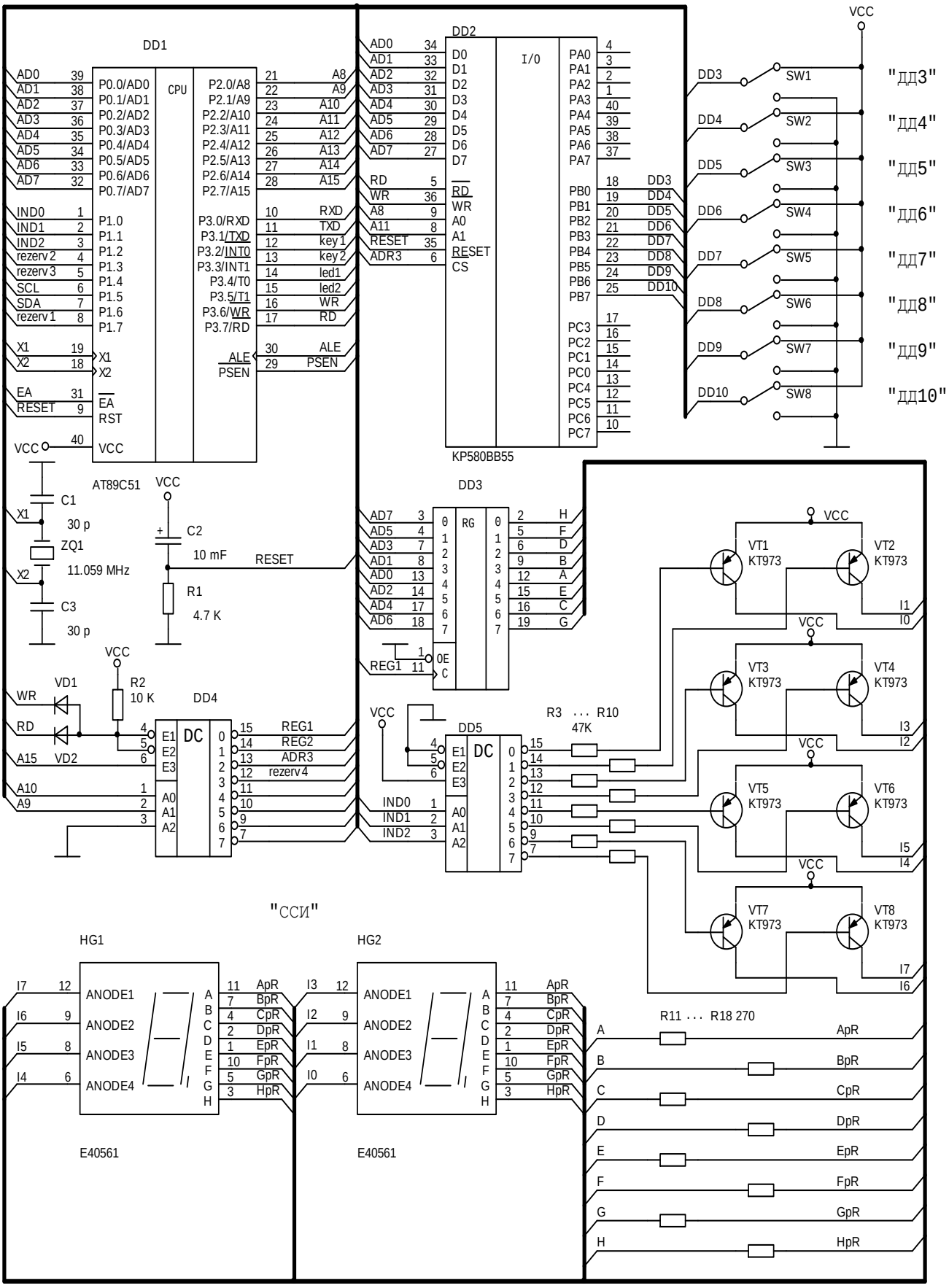

Рисунок 1.13 – Пример подключения к МП цифробуквенных светодиодных семисегментных индикаторов (HG1 и HG2)

Задача 9. Перечислите все элементы (включая источник питания) замкнутого электрического контура, в состав которого входит сегмент C (E) разряда ANODE2 (ANODE4) индикатора HG1 (HG2) (см. рисунок 1.13). Каково назначение резисторов R11…R18 в этих контурах?

Задача 10. Опишите функциональное назначение элементов R1 и C2 (см. рисунок1.13).

Задача 11. Опишите функциональное назначение элементов R3…R10 и VT1…VT8 (см. рисунок 1.13).

Задача 12. Опишите функциональное назначение элементов C1, C3 и ZQ1 (см. рисунок 1.13).

Задача 13. Опишите функциональное назначение элементов R2, VD1 и VD2 (см. рисунок 1.13).

2 Программирование микропроцессорных устройств на ассемблере

Задача 14. Запишите последовательность команд для вывода на цифробуквенный светодиодный семисегментный индикатор заданного символа в соответствии с рисунком 1.13. Номер индикатора и номер его разряда, на котором должен быть отображен символ, указаны в таблице 2.1.

Указания к решению задачи 14. Работой светодиодных сегментов управляют сигналы, формируемые на выходах регистра DD3 (если на выходе регистра «0» – сегмент светится, если «1» – нет). Поэтому надо по схеме рисунка 1.13 в соответствии с заданным символом определить, какое двоичное число следует записать в регистр DD3, затем определить системные адреса элементов DD3 и DD4, записать их в шестнадцатеричной форме и использовать в командах (таблица А.1).

Последовательность команд записать в виде таблицы (таблица 2.2). Проанализировать соответствие результатов определению выполняемых операций. Выделить команды, модифицирующие флаги слова состояния программы PSW и пояснить состояния флагов после выполнения этих команд.

Задача 15. Записать последовательность команд для вывода на заданный индикатор символа «0» или «1» в соответствии с рисунком 1.13 в зависимости от положения переключателя SW1…SW8, указанного в задании. Исходные данные к задаче 15 приведены в таблице 2.3.

Указание к решению задачи 15. При нижнем положении переключателя на индикаторе следует отобразить «0», при верхнем положении − «1». Решение задачи оформить, как показано в таблице 2.2.

Таблица 2.1 – Исходные данные к задаче 14

Номер варианта	Номер индикатора на рисунке 1.13	Номер разряда в индикаторе	Отображаемый на индикаторе символ
0	HG1	3	8
1	HG2	1	6
2	HG1	2	3
3	HG2	3	1
4	HG1	4	2
5	HG2	2	0
6	HG1	1	4
7	HG2	4	7
8	HG1	2	5
9	HG2	3	9

Таблица 2.2 − Пример записи последовательности выполняемых команд

Номер команды	Команда	Код	Выполняемая операция	Содержимое регистров и памяти до и после выполнения команды
1	MOV A,R0	E8	Пересылка байта данных из регистр R0 в аккумулятор	До: A(00H); R0(F2H). После: A(F2H); R0(F2H)
2	…	…	…	…

Таблица 2.3 – Исходные данные к задаче 15

Номер варианта	Номер индикатора на рисунке 1.13	Номер разряда в индикаторе	Переключатель на рисунке 1.13	Положение переключателя
0	HG1	1	SW1	Нижнее
1	HG2	1	SW5	Верхнее
2	HG1	2	SW2	Нижнее
3	HG2	2	SW6	Верхнее
4	HG1	3	SW3	Нижнее
5	HG2	3	SW7	Верхнее
6	HG1	4	SW4	Нижнее
7	HG2	4	SW8	Верхнее
8	HG1	3	SW1	Верхнее
9	HG2	4	SW2	Нижнее

3 Разработка программного обеспечения на C++

3.1 Среда программирования CodeBlocks, ее запуск и создание проекта

В среде программирования CodeBlocks есть минимально необходимый комплект (редактор, компилятор и отладчик) для разработки программ. Скачать CodeBlocks можно из интернета (бесплатно) или из ЭВМ дисплейного класса. Далее надо:

– распаковать скачанный архив и запустить инсталляционный файл. В окошке выбора компонентов для установки выбираем либо standart, либо full (принципиальной разницы нет);

– выбрать путь установки, либо оставить по умолчанию;

– процесс установки завершить, нажав Finish.

Запуск среды программирования CodeBlocks. Запускается среда программирования CodeBlocks через Пуск –> Все программы –> CodeBlocks –> CodeBlocks или путем щелчка по соответствующему ярлыку на рабочем столе.

Создание проекта. По умолчанию в CodeBlocks при запуске открывается стартовая страница, на которой располагаются кнопки для создания проектов, открытия через проводник, открытия последних проектов. Щелкаем по «Create a new project» (Создать новый проект). В появившемся окошке выбираем значок «Console application» и жмем GO. Таким образом мы будем создавать проекты для построения консольных приложений на языке программирования C++.

Далее в следующем окошке жмем просто «Next», далее в следующем окне выбираем язык C++, жмем Next. Далее нас попросят ввести название проекта и папку, в которой он у нас будет находиться. Можно, например, для всех проектов создать папку в корне какого-либо диска на вашем компьютере с названием «My project C++».

В следующем окошке выбора компилятора оставляем все как есть и жмем Finish.

Проект создан, все необходимые файлы добавлены. Структуру проекта можно посмотреть слева на панели «Management» (вкладка «Projects»). Здесь мы видим, что в проект (в папку «Sourses») средой автоматически добавлен файл «main.cpp» (файл, содержащий исходный код программы). В нем и пишется код разрабатываемой программы. Открываем его для редактирования в редакторе CodeBlocks (щелкаем по нему левой кнопкой мыши два раза). Файл не пуст: среда программирования сама в него добавила заготовку программы. Делается это для облегчения работы программисту. Набираем в редакторе CodeBlocks следующий текст программы:

```
#include <iostream>
using namespace std;
int main()
{
cout << "This is my first program!" << endl;
```

```
return 0;
}
```

Теперь программу нужно скомпилировать, построить, а затем запустить на выполнение. Находим кнопку, которая внешне напоминает шестеренку (при наведении курсора всплывает подсказка «build»), она служит для компиляции и построения проекта. Нажимаем и наблюдаем за процессом внизу на панели «Logs» (вкладка «build messages». Если панель не видна, то нажмите клавишу F2). Если ошибок нет, то значит компиляция (проверка на синтаксические ошибки) и построение (объединение всех нужных файлов в единый объектный модуль для дальнейшего запуска) прошли успешно.

Теперь можно запустить программу на выполнение. Для этого жмем кнопку рядом, в виде треугольника («Run»). Должно появиться окошко с результатом выполнения написанной программы. Она выводит на экран строку «This is my first program!».

Составьте другие фразы, введите их в код программы и запустите скорректированную программу на выполнение.

3.2 Разработка программного обеспечения на C++

Линейные программы. Программы, выполняющие различные операции последовательно, одну за другой называются линейными. Рассмотрим примеры программ, имеющих линейную структуру.

Задача 16. Разработайте программу, которая будет содержать просьбу о вводе пользователем с клавиатуры трёх чисел с последующим вычислением их суммы и произведения и выводом результатов на экран дисплея.

Решение

Текст программы приведен ниже.

```
//Программа, которая просит пользователя ввести три числа,
//и затем выводит их сумму и произведение
#include <iostream>
using namespace std;

int main()
{
int a, b, c;
cout << "Vvedite poshaluyasta tri chisla\n";
cin >> a >> b >> c;
cout << "Summa ravna: " << a + b + c << endl;
cout << "Proizvedeniye ravno: " << a * b * c << endl;
return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Задача 17. Доработайте программу, приведенную в задаче 16, чтобы в ней обеспечивался ввод пользователем пяти чисел с последующим вычислением их суммы и произведения и выводом результатов на экран дисплея. Выполните программу в среде CodeBlocks.

Задача 18. Разработайте программу, которая будет содержать просьбу о вводе пользователем с клавиатуры двух чисел с последующим вычислением их суммы, разности, произведения и частного и выводом результатов на экран дисплея.

Решение

Текст программы приведен ниже.

```
/*Программа, которая просит от пользователя ввести два числа, а потом выводит результаты вычислений, а именно: сумму, разность, произведение и частное*/
#include <iostream>
using namespace std;
int main()
{
int a, b;
cout << "Vvedite poshaluyasta dva chisla\n";
cin >> a >> b;
cout << "Summa " << a << " i " << b << " ravna " << a + b << endl;
cout << "Raznost " << a << " i " << b << " ravna " << a – b << endl;
cout << "Proizvedeniye " << a << " i " << b << " ravno " << a * b << endl;
cout << "Chastnoye " << a << " i " << b << " ravno " << a / b << endl;
return 0;
}
```


Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Задача 19. Разработайте программу, которая будет содержать просьбу о вводе пользователем с клавиатуры трёх чисел с последующим вычислением их квадратов и выводом результатов на экран дисплея. Выполните программу в среде CodeBlocks.

Задача 20. Разработайте программу, которая будет запрашивать у пользователя значения двух катетов прямоугольного треугольника, а выдавать значение его гипотенузы.

Решение

Текст программы приведен ниже.

```
//Программа находит значение гипотенузы прямоугольного/треугольника
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
float ab, bc; //объявляем переменные типа float, для хранения значений катетов
float ac;     //для хранения значения гипотенузы
cout << "Vvedite katet 1: " << endl; //печатаем на экране подсказку о/вводе
cin >> ab;                 //запоминаем введенное значение в переменную ab
cout << "Vvedite katet 2: " << endl; //печатаем на экране подсказку о вводе
cin >> bc;                 //запоминаем введенное значение в переменную bc
ac = sqrt((ab * ab) + (bc * bc));    //считаем гипотенузу по формуле
cout << "Gipotenuza ravna: " << ac << endl; //выводим результат на экран
return 0;
}
```

Выполните программу в среде CodeBlocks.

Задача 21. Изменить программу, приведенную в задаче 19, которая будет запрашивать у пользователя значения трёх сторон треугольника, а выдавать значение его площади. Выполните программу в среде CodeBlocks.

Задача 22. Разработайте программу, которая будет запрашивать у пользователя радиус круга, определять диаметр круга, длину окружности и площадь круга с выводом результатов на экран дисплея.

Решение

Текст программы приведен ниже.

```
/*Программа считывает радиус круга и выводит на дисплей диаметр круга, длину окружности и площадь круга*/
#include <iostream>
using namespace std;

int main()
{
    const float p = 3.14159;
    float radius;
    cout << "Vvedite radius kruga: \n";
    cin >> radius;
    cout << "Diametr kruga raven " << 2 * radius << endl;
    cout << "Dlina okrushnosti ravna " << 2 * p * radius << endl;
    cout << "Ploshad' kruga ravna " << p * radius * radius << endl;
    return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Программы с ветвлением. В задачах, когда в зависимости от ситуации надо принимать различные решения, используются условные операторы, которые способны изменить последовательность выполнения операций.

Задача 23. Разработайте программу, которая будет запрашивать у пользователя число и если это число положительное, то программа будет выводить сообщение о том, что число положительное, если введенное число отрицательное, то программа будет выводить сообщение о том, что число отрицательное, и если введен ноль, то программа будет выводить сообщение о том, что введен ноль.

Решение

Текст программы приведен ниже.

```
//Используется условный оператор if – else – if
#include <iostream>
using namespace std;
int main()
{
    int value;
    cout << "Enter number: " << endl;
    cin >> value;
    if (value > 0)
    {
       cout << "Number positive" << endl;
       cout << "You have entered number: " << value << endl;
    }
    else if (value == 0)
    {
       cout << "The number is not neither positive, not negative" << endl;
       cout << "You have entered number: " << value << endl;
    }
    else
    {
       cout << "Number negative" << endl;
       cout << "You have entered number: " << value << endl;
    }
    return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Задача 24. Разработайте программу, которая будет содержать просьбу о вводе пользователем с клавиатуры положительного числа, и в зависимости от

его значения, программа будет выводить сообщение о том, что введенное число меньше или равно 10, или введенное число лежит в диапазоне больше 10, но меньше или равно 100, или введенное число больше 100. Выполните программу в среде CodeBlocks.

Задача 25. Разработайте программу, которая после ввода числа с клавиатуры будет не только сообщать о том, что число положительное или отрицательное, но и будет сообщать – это число больше 100 или меньше (соответственно и с отрицательным числом).

Решение

Текст программы приведен ниже.

```
//Используется условный оператор if, вложенный в оператор if
#include <iostream>
using namespace std;

int main()
{
    int value;
    cout << "Enter number: " << endl;
    cin >> value;
    if (value > 0)
    {
       cout << "Number positive" << endl;
       cout << "You have entered number: " << value << endl;
       if (value >= 100)
          cout << "This number is more or equally 100" << endl;
       else
          cout << "This number is less 100" << endl;
    }
    else if (value == 0)
    {
       cout << "The number is not neither positive, nor negative" << endl;
       cout << "You have entered number: " << value << endl;
    }
    else
    {
       cout << "Number negative" << endl;
       cout << "You have entered number: " << value << endl;
       if (value >= -100)
          cout << "This number is more or equally -100" << endl;
       else
          cout << "This number is less -100" << endl;
```

```
    }
    return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Задача 26. Разработайте программу, в которой пользователь вводит число, а программа идентифицирует его (если 0, то программа сообщает, что введен ноль и т. д.). Если число не идентифицировано, то выводится соответствующее сообщение.

Решение

Текст программы приведен ниже.

```
//Используется оператор выбор switch case
#include
using namespace std;

int main()
{
    int value;
    cout << "Enter number: " << endl;
    cin >> value;
    switch (value)
    {
       case 0:
          cout << "You have entered number 0" << endl;
          break;
       case 5:
          cout << "You have entered number 5 " << endl;
          break;
       case 21:
          cout << "You have entered number 21 " << endl;
          break;
       default:
          cout << "The number is not identified" << endl;
          break;
    }
    return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Циклические программы.

Задача 27. Разработайте программу, в которой надо вывести на экран дисплея свою фамилию ровно 7 раз.

Решение

Текст программы приведен ниже.

```
//Используется оператор циклов while с предусловием
#include <iostream>
using namespace std;

int main()
{
int i;
i = 1;
while (i <= 7)
{
    cout << "My family is   " << endl;
    i = i + 1;
 }
 return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям.

Задача 28. Разработайте программу, в которой надо вывести на экран дисплея свою фамилию ровно 7 раз.

Решение

Текст программы приведен ниже.

```
/*Используется оператор циклов while с предусловием с оператором break*/
#include <iostream>
using namespace std;

int main()
{
    int i;
    i = 1;
    while (i <= 7)
    {
       cout << "Your name" << endl;
          if (i == 5)
```

```
            break;
        i = i + 1;
    }
    return 0;
}
```

Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям. В чем отличие этой программы от предыдущей.

Задача 29. Разработайте программу, в которой надо вывести на экран дисплея аббревиатуру факультета и университета ровно 5 раз.

Решение

Текст программы приведен ниже.

```
//Используется оператор циклов do while с постусловием
#include <iostream>
using namespace std;

int main()
{
//инициализируем переменную значением 1 при объявлении
int i = 1;
do
{
        cout << "ETF BRU" << endl;
        i++;
}
while (i <= 5);
return 0;
}
```


Выполните программу в среде CodeBlocks и составьте комментарии к выполняемым операциям. В чем отличие этой программы от предыдущей.

Задача 30. Разработайте программу, которая запрашивает у пользователя целые числа, подсчитывает количество четных и в конце выводит результат на экран дисплея.

Решение

Текст программы приведен ниже.

```
/*Программа запрашивает у пользователя целые числа, подсчитывает количество
четных и в конце выводит результат. Используется оператор циклов do while с
постусловием*/
#include <iostream>
using namespace std;
```

```
//главная функция программы, с нее начинается выполнение
int main()
{
//объявляем необходимые переменные
    int var, counter = 0;
//начинаем цикл ввода чисел и подсчета четных
//условие выхода из цикла – ввод пользователем -1
    do
    {
//запрашиваем число и сохраняем в переменную var
        cout << "Vvedite celoye chislo (vihod -1): ";
        cin >> var;
//если число нечетное, то возвращаемся на начало цикла, не дойдя до конца
        if (var % 2 != 0)
            continue;
/*если число было четным, то выполнение программы доходит до этого оператора и счетчик количества четных чисел увеличивается на единицу*/
        counter++;
    }
    while (var != -1);
//выводим результат подсчета четных чисел на экран
    cout << endl << "Vvedeno chetnih chisel: " << counter << endl;
//если программа отработала без ошибок, то функция main()
//возвращает число нуль
    return 0;
}
```

Выполните программу в среде CodeBlocks.

Задача 31. Разработайте программу, в которой надо ввести известные рейтинги 10 студентов (целые числа в диапазоне 0…60) и определить средний рейтинговый балл группы с выводом результата на экран дисплея.

Решение

Текст программы приведен ниже.

```
/*Известны рейтинги 10 студентов (целые числа в диапазоне 0–60). Нужно определить средний рейтинговый балл группы*/
#include <iostream>
using namespace std;

int main()
{
//объявляем необходимые переменные
    float average;
    int ball, counter, total;
//задаем начальные значения
```

```
    total = 0, counter = 0;
//запрашиваем балл студента и запоминаем
    cout << "Vvedite ball, dlya vihoda vvedite -1\n";
    cin >> ball;
//начинаем цикл, если не было введено -1
    while (ball != -1)
    {
//накапливаем общее количество баллов и засчитываем ввод
        total += ball;
        counter++;
          if (counter == 10)
              break;
        cout << "Vvedite ocenku, dlya vihoda vvedite -1\n";
        cin >> ball;
    }
//проверка на ввод, если было введено хотя бы одно значение
    if (counter != 0)
    {
//рассчитываем средний балл
        average = (float) total / counter;
        cout << "Srednee znacheniye: " << average << endl;
    }
    else
//если не было введено ни одно значение
        cout << "balli ne vvedeni\n";
    return 0;
}
```

Выполните программу в среде CodeBlocks.

Задача 32. Разработайте программу, которая последовательно запрашивает у пользователя десять чисел, находит максимальное число и выводит результат на экран дисплея.

Решение

Текст программы приведен ниже.

```
// Используется оператор циклов for
using namespace std;

int main()
{
int counter;
float number, largest;
//запрашиваем первое число, запоминаем его
cout << "Inter: ";
```

```
cin >> largest;
//запрашиваем остальные девять чисел в цикле for
for (counter = 1; counter <= 9; counter++)
{
        cout << "Inter: ";
        cin >> number;
//если вновь введенное число больше, то перезаписываем max
        if (number > largest)
                largest = number;
}
cout << endl << "Max: " << largest << endl;
return 0;
}
```

Выполните программу в среде CodeBlocks.

Функции. Функция представляет собой самостоятельный элемент кода программы или фрагмент кода программы, который выполняет свою отдельную небольшую задачу. В связи с этим код программы значительно упрощается.

Задача 33. Разработайте программу, которая округляет число, введенное пользователем с клавиатуры, до целого значения, до одной десятой, до одной сотой, до одной тысячной и выводит результаты на экран дисплея.

Решение

Текст программы приведен ниже.

```
//Программа, определяющая четыре функции для округления X различными способами
#include <iostream>
#include <iomanip>
#include <math.h>
using namespace std;
//описываем прототипы четыре функции округления
double roundToInteger(double integer)
{
    return floor(integer * 1 + .5) / 1;
}
double roundToTenths(double tenths)
{
    return floor(tenths * 10 + .5) / 10;
}
double roundToHundredths(double hundredths)
{
    return floor(hundredths * 100 + .5) / 100;
}
```

```
double roundToThousandths(double thousandths)
{
    return floor(thousandths * 1000 + .5) / 1000;
}
//главная функция программы
int main()
{
    double number;
    cout << "Enter number: ";
    cin >> number;
//выводим результаты работы в табулированном формате
    cout << number << setw(10) << roundToInteger(number) << setw(10) << setprecision(1)
        << setiosflags (ios::fixed | ios::showpoint) << roundToTenths(number)
        << setw(10) << setprecision(2) << setiosflags (ios::fixed | ios::showpoint)
        << roundToHundredths(number) << setw(10) << setprecision(3)
        << setiosflags (ios::fixed | ios::showpoint) << roundToThousandths(number) << endl;
    return 0;
}
//описываем четыре функции округления
    cout << number << setw(10) << roundToInteger(number) << setw(10) << setprecision(1)
        << setiosflags (ios::fixed | ios::showpoint) << roundToTenths(number)
        << setw(10) << setprecision(2) << setiosflags (ios::fixed | ios::showpoint)
        << roundToHundredths(number) << setw(10) << setprecision(3)
        << setiosflags (ios::fixed | ios::showpoint) << roundToThousandths(number) << endl;
//выводим результаты работы
    cout << number << "    " << roundToInteger(number) << "    " << roundToTenths(number)
        << "    " << roundToHundredths(number) << "    " << roundToThousandths(number) << endl;
```

Выполните программу в среде CodeBlocks.

Массивы. Массив – это совокупность переменных, содержащих данные одного типа (например, int), объединенных одним общим именем. Каждая отдельно взятая переменная называется элементом (ячейкой) массива. В памяти элементы массива всегда располагаются строго последовательно, благодаря чему повышается скорость доступа к данным. Доступ к элементам массива осуществляется через индексы , которые указываются после имени в квадратных скобках и обязательно нумеруются, начиная с нуля.

Задача 34. Надо создать одномерный массив, содержащий 15 элементов, наполнить его случайными значениями в интервале от 1 до 200. Посчитать и

вывести результаты на экран дисплея: сумму всех четных элементов массива; сумму всех нечетных элементов массива; сумму всех элементов массива; произведение всех элементов массива, значения которых меньше 50; произведение элементов массива с индексами от 2 и 7.

Решение

Текст программы приведен ниже.

```
//Программа с использование одномерного массива
#include <iostream>
#include <stdlib.h>
#include <time.h>

//прототип функции для печати массива
void printArray(int[], const int);
using namespace std;

int main()
{
//объявляем необходимые переменные
const int size = 15;
int massiv[size];
int sumChet = 0, sumNechet = 0, sum = 0, proizv1 = 1, proizv2 = 1;
//задаем начало отчета для рандомизатора rand()
srand(time(NULL));
//наполняем массив случайными величинами
for(int i = 0; i < size; i++)
        massiv[i] = 1 + rand() % 200;
//выводим содержимое массива на экран
printArray(massiv, size);
//выполняем необходимые подсчеты в цикле
for(int i = 0; i < size; i++)
{
        if(massiv[i] % 2 == 0)
                sumChet += massiv[i];
        if(massiv[i] % 2 != 0)
                sumNechet += massiv[i];
        sum += massiv[i];
        if(massiv[i] < 50)
                proizv1 *= massiv[i];
        if(i == 2 || i == 7)
                proizv2 *= massiv[i];
}
//выводим результаты работы программы
cout << "\nSumma chetnih: " << sumChet << endl;
cout << "Summa nechetnih: " << sumNechet << endl;
```

```
cout << "Summa vseh: " << sum << endl;
cout << "Proizvedeniye elementov do 50: " << proizv1 << endl;
cout << "Proizvedeniye elementov ot 0 do 4: " << proizv2 << endl;
return 0;
}
//функция вывода массива на экран
void printArray(int a[], const int s)
{
  for(int k = 0; k < s; k++)
        cout << a[k] << " ";
        cout << endl;
}
```

Результат работы программы приведен на рисунке 3.1.

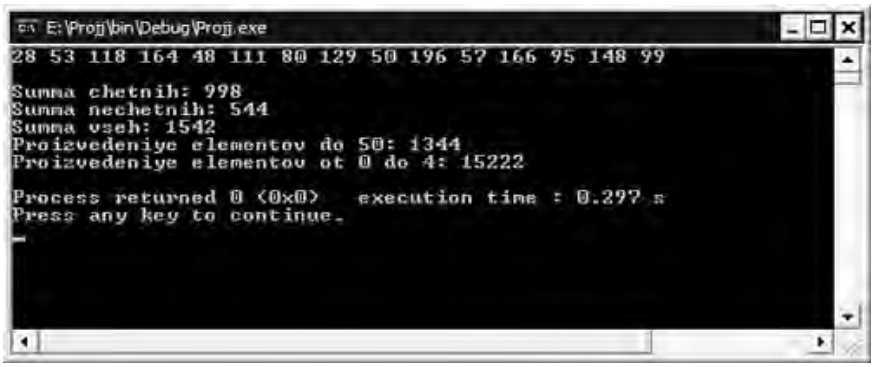

Рисунок 3.1 – Результат выполнения программы задачи 34

4 Разработка программного обеспечения цифровых измерительных приборов с встроенными микроконтроллерами на С++

Чтобы проектируемый цифровой прибор – рабочее средство измерения (РСИ) – стал измерительным, необходимо предусмотреть разработку специального программного обеспечения, обеспечивающего выполнение градуировочных операций. Сущность их заключается в передаче РСИ единицы измеряемой физической величины от образцового средства измерения (ОСИ) или меры. Это может быть сделано путем одновременного воздействия измеряемой величиной *X* на РСИ и ОСИ (рисунок 4.1).

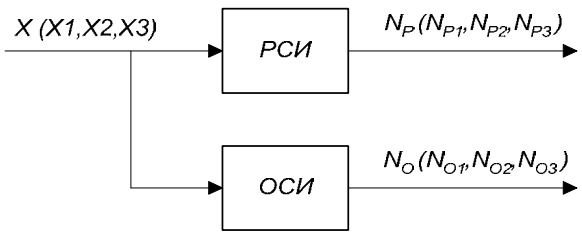

Рисунок 4.1 – Градуировка РСИ с использованием ОСИ

При этом на их ЦОУ появятся числа N_p и N_o соответственно, причем N_o будет действительным значением X. Выполнив несколько таких экспериментов, по полученным числовым значениям N_{p1}, N_{p2}, $N_{o1} = X_1$ и $N_{o2} = X_2$ может быть построена графическая зависимость $N_p = f(X)$, которая и будет являться градуировочной характеристикой РСИ (рисунок 4.2).

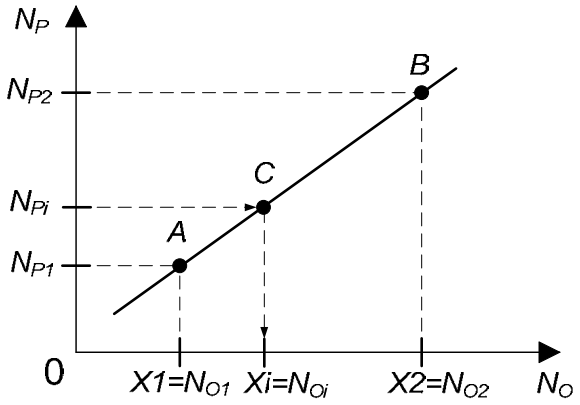

Рисунок 4.2 – Градуировочная характеристика РСИ

Если зависимость $N_p = f(X)$ линейная, то можно записать математическое выражение для градуировочной характеристики, которое будет представлять собой уравнение прямой, проходящей через две точки A и B с известными координатами.

Тогда нахождение неизвестной измеряемой величины X_i может быть осуществлено путем решения уравнения:

$$X_i = \frac{(N_{o2} - N_{o1})(N_{pi} - N_{p1})}{N_{p2} - N_{p1}} + N_{o1}, \qquad (4.1)$$

где N_{pi} – числовое значение, полученное с помощью РСИ при воздействии на него измеряемой величиной X_i.

Координаты точек $A(N_{O1}, N_{P1})$ и $B(N_{O2}, N_{P2})$ – величины известные, которые определяются при градуировке РСИ и записываются в его ППЗУ.

Задача 35. Разработайте программу, которая вычисляет измеряемые величины в приборе с линейной функцией преобразования по двум координатам точек градуировочной характеристики и выводит результат на экран дисплея.

Решение

Текст программы приведен ниже.

```
/*Вычисление измеряемой величины в приборе с линейной функцией преобразования (используются координаты двух точек)*/
#include <iostream>
#include <math.h>
```

```
using namespace std;

int main()
{
//Описание переменных и их типов;
    float No1,No2,Np1,Np2,RezIzmer;
    int Np;

//Константы, записываемые в память прибора при градуировке;
    No1 = 12,48;
    No2 = 80,43;
    Np1 = 18,65;
    Np2 = 106,32;
/*Считывание текущего значения Np с АЦП в режиме измерения (здесь – с клавиатуры)*/
    cout << "Chit data c ADC: \n";
    cin >> Np;
//Вычисление результата измерения;
    RezIzmer=(((No2–No1)*(Np–Np1))/(Np2–Np1))+No1;
//Вывод результата измерения на дисплей;
    cout << "Rezultat Izmereniy  =  " << RezIzmer  << endl;
    return 0;
}
```

Выполните программу в среде CodeBlocks.

Задача 36. Измените код программы в задаче 35 так, чтобы переменные N_{o1}, N_{o2}, N_{p1}, N_{p2} вводились с клавиатуры пользователем. Выполните программу в среде CodeBlocks.

При градуировке РСИ с нелинейной функцией преобразования необходимо определить координаты возможно большего числа точек в диапазоне измерения величины X и записать их значения в его ППЗУ.

Нелинейная функция преобразования может быть аппроксимирована набором прямолинейных отрезков, каждый из которых описывается выражением, аналогичным (4.1) (рисунок 4.3).

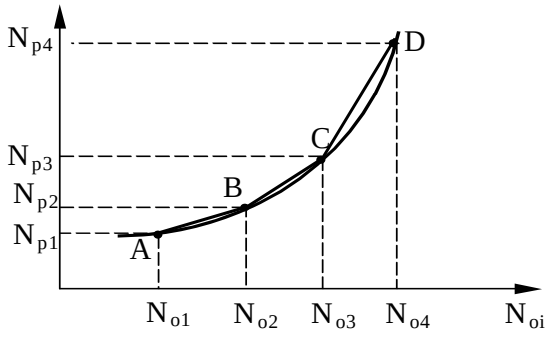

Рисунок 4.3 – Градуировочная характеристика РСИ с нелинейной функцией преобразования

Координаты двух соседних точек позволяют составить уравнение отрезка, соединяющего эти точки.

Для отрезка *AB* с учетом (4.1) уравнение прямой запишется в виде:

$$X = \frac{(N_{o2} - N_{o1})(N_p - N_{p1})}{N_{p2} - N_{p1}} + N_{o1}. \quad (4.2)$$

Для отрезка *BC*

$$X = \frac{(N_{o3} - N_{o2})(N_p - N_{p2})}{N_{p3} - N_{p2}} + N_{o2}. \quad (4.3)$$

Для отрезка *CD*

$$X = \frac{(N_{o4} - N_{o3})(N_p - N_{p3})}{N_{p4} - N_{p3}} + N_{o3}. \quad (4.4)$$

Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ будет отличаться выполнением дополнительных операций, связанных с определением координат двух соседних точек, между числовыми значениями которых окажется измеряемая величина. Определение двух ближайших координат N_p и $N_{P(i+1)}$ осуществляется путем сравнения полученного РСИ числа N_P и значений, зафиксированных в его ППЗУ при градуировке. Для числовых значений N_{Pi} и $N_{P(i+1)}$ в ППЗУ хранятся также соответствующие им числовые значения N_{Oi} и $N_{O(i+1)}$. Тогда измеряемая величина X может быть определена с помощью выражения

$$X = \frac{N_{O(i+1)} - N_{Oi}}{N_{P(i+1)} - N_{Pi}}(N_P - N_{pi}) + N_{oi}. \quad (4.5)$$

Задача 37. Разработайте программу, которая вычисляет измеряемую величины в приборе с нелинейной функцией преобразования по четырём координатам точек градуировочной характеристики и выводит результат на экран дисплея.

Решение

Текст программы приведен ниже.

```
/*Вычисление измеряемой величины в приборе с нелинейной функцией преобразования (кривая аппроксимирована тремя прямолинейными отрезками)*/
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
```

```
//Описание переменных и их типов;
    float No1,No2,Np1,Np2,No3,Np3,No4, Np4,RezIzmer;
    int Np;
//Константы, записываемые в память прибора при градуировке;
   No1 = 20; No2 = 40; No3 = 80; No4 = 110; Np1 = 30; Np2 = 50;
   Np3 = 90; Np4 = 130;
/*Считывание текущего значения Np с АЦП в режиме измерения (здесь – с клавиатуры)*/
   cout << "Chit data c ADC: \n";
   cout << endl << '\t'; //пустая строка плюс отступ при выводе Np;
   cin >> Np;
/*Проверка попадания Np в диапазон (Np1...Np2) и, при выполнении условия, вычисление результата измерения*/
    if (Np1<=Np && Np<=Np2)
    {
       RezIzmer=((No2–No1)*(Np–Np1))/(Np2–Np1)+No1;
    }
/*Иначе, проверка попадания Np в диапазон (Np2...Np3) и, при выполнении условия, вычисление результата измерения*/
    else
       if (Np2<Np && Np<=Np3)
          {
             RezIzmer=((No3–No2)*(Np–Np2))/(Np3–Np2)+No2;
          }
       else
/*Иначе, проверка попадания Np в диапазон (Np3...Np4) и, при выполнении условия, вычисление результата измерения*/
             RezIzmer=((No4–No3)*(Np–Np3))/(Np4–Np2)+No3;
//Вывод результата измерения на дисплей;
   cout << endl; // пустая строка;
   cout << "Rezultat Izmereniy  =  " << RezIzmer  << endl;
   return 0;
}
```

Выполните программу в среде CodeBlocks.

Задача 38. Измените код программы в задаче 37 так, чтобы переменные *No*1, *No*2, *Np*1, *Np*2, *No*3, *Np*3, *No*4, *Np*4 вводились с клавиатуры пользователем. Выполните программу в среде CodeBlocks.

Задача 39. Разработайте программу, которая вычисляет измеряемую величину в приборе с нелинейной функцией преобразования (кривая аппроксимирована $N - 1$ прямолинейными отрезками) с вычислением среднего арифметического результатов наблюдений и выводит результат на экран дисплея. Число точек на кривой функции преобразования и число наблюдений измеряемой величины вводится с клавиатуры при градуировке прибора.

Решение

Текст программы приведен ниже.

```
/*Вычисление измеряемой величины в приборе с нелинейной функцией преобразования (кривая аппроксимирована (N – 1) прямолинейными отрезками). Число точек на кривой функции преобразования и число наблюдений измеряемой величины вводится с клавиатуры при градуировке прибора*/
#include <iostream>
//#include <math.h>
using namespace std;

int main()
{
//Описание переменных и их типов;
    unsigned short int n;   //Число наблюдений измеряемой величины;
    unsigned short int N; /*Число точек на кривой функции преобразования, вводимых с клавиатуры при градуировке прибора*/
    cout << endl;    //пустая строка;
    cout << "Vvedite chislo tochek na krivoy N = ";
        cin >> N;
        unsigned short int arraySize = N;
        cout << endl;  // пустая строка;
    cout << "Vvedite chislo rezultatov nabludeniy n = ";
        cin >> n;
        cout << endl;  // пустая строка;
float massivNo[arraySize];
    for(int i = 0; i < arraySize; i++)
        {
cout << "Vvedite Noi: ";
cin >> massivNo[i];
}
        cout << endl;    // пустая строка;
            for(int i = 0; i < arraySize; i++)
            {
             cout << massivNo[i] << "\t";
            }
        cout << endl;     // пустая строка;
        cout << endl;     // пустая строка;
float massivNpg[arraySize];
    for(int i = 0; i < arraySize; i++)
        {
cout << "Vvedite Npgi: ";
cin >> massivNpg[i];
}
        cout << endl;     // пустая строка;
```

```
                for(int i = 0; i < arraySize; i++)
                {
                 cout << massivNpg[i] << "\t";
                }
            cout << endl;     // пустая строка;
            cout << endl;     // пустая строка;
```

*/*Считывание текущего значения Np с АЦП в режиме измерения (здесь – с клавиатуры)*/*

```
            float sumNp;
            float RezIzmer;
        float massivNp[n];
            for(int i = 0; i < n; i++)
                {
        cout << "Chituvanie c ADC Npi: ";
        cin >> massivNp[i];
        }
            cout << endl;       // пустая строка;
               for(int i = 0; i < n; i++)
               {
                 cout << massivNp[i] << "\t";
                 sumNp += massivNp[i];
               }
            cout << endl;       // пустая строка;
            cout << endl;       // пустая строка;
            float Nps = sumNp / n;
      cout << "Crednee arifmet. Nps = " << Nps;
```

*/*Проверка попадания Np в диапазон (Np1...Np2) и, при выполнении условия, вычисление результата измерения*/*

```
            for(int i = 0; i < arraySize; i++)
            {
        if (massivNpg[i]<= Nps && Nps <= massivNpg[i+1])
            {
            RezIzmer=((massivNo[i+1]–massivNo[i])*(Nps–
      massivNpg[i]))/(massivNpg[i+1]–massivNpg[i])+massivNo[i];
            }
       }
            cout << endl;       //пустая строка;
```

//Вывод результата измерения на дисплей;

```
        cout << endl; // пустая строка;
        cout << "Rezultat Izmereniy  =  " << RezIzmer  << endl;
        return 0;
}
```

Результат работы программы приведен на рисунке 4.4.

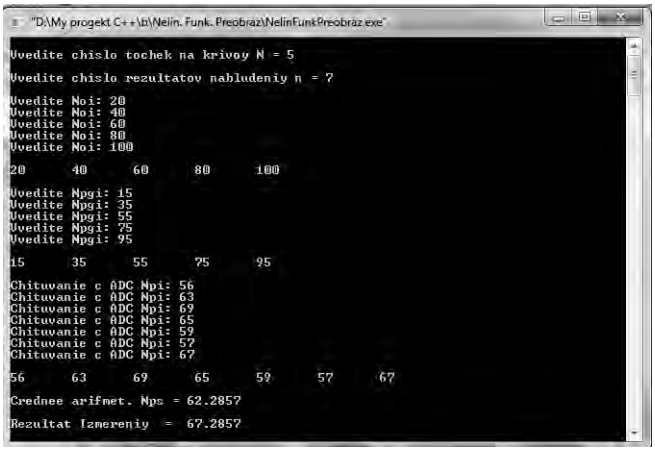

Рисунок 4.4 – Результат выполнения программы в среде программирования CodeBlocks при $N = 5$ и $n = 7$

Задача 40. Разработайте программу, которая вычисляет измеряемую величины в приборе с нелинейной функцией преобразования (кривая аппроксимирована $N - 1$ прямолинейными отрезками). Число точек на кривой функции преобразования и число наблюдений измеряемой величины вводятся с клавиатуры при градуировке прибора, указаны диапазоны их значений и после ввода выполняется проверка правильности ввода; если вводимое значение не попадает в рекомендуемый диапазон, предлагается повторить ввод. При вычислении среднего арифметического минимальное и максимальное значения результатов наблюдений отбрасываются. Автоматически устанавливается количество разрядов после запятой в полученном результате в зависимости от его попадания в программно-заданные диапазоны измерения.

Решение

Текст программы приведен ниже.

```
#include <iostream>
#include <math.h>
using namespace std;
//прототип функции округления с именем "F_Okruglenie" и аргументом "chislo"; это
//стандартная функция, взывается из библиотеки <math.h>;
float F_Okruglenie(float chislo,unsigned short k);
/*прототип функции статистической обработки с именем "F_StatObrabotka"
и аргументами "sumNp","minNp","maxNp" и "n"; эта функция нестандартная, создана
программистом самостоятельно */;
float F_StatObrabotka(float sumNp,float minNp,float maxNp,int n);
int main()
{
    //Описание переменных и их типов;
  unsigned short n;   //Число наблюдений измеряемой величины;
  unsigned short N; /*Число точек на кривой функции преобразования, вводимых с
клавиатуры при градуировке прибора*/
```

```
   float minNp,maxNp;   /*минимальное и максимальное значения результатов наблюдений измеряемой величины*/
   unsigned short k;
   cout << endl;              //пустая строка;
do{
   cout << "Vvedite chislo tochek na krivoy(N=2...10) N = ";
         cin >> N;
} while (N > 10 || N < 2 );
         unsigned short arraySize = N;
         cout << endl;        // пустая строка;
do{
   cout << "Vvedite chislo rezultatov nabludeniy (n=3...7) n = ";
         cin >> n;

} while (n <3||n > 7 );
         cout << endl;        // пустая строка;
float massivNo[arraySize];
   for(int i = 0; i < arraySize; i++)
   {
cout << "Vvedite Noi: ";
cin >> massivNo[i];
    }
        cout << endl;         //пустая строка;
             for(int i = 0; i < arraySize; i++)
             {
              cout << massivNo[i] << "\t";
             }
        cout << endl;         // пустая строка;
        cout << endl;         //пустая строка;
float massivNpg[arraySize];
   for(int i = 0; i < arraySize; i++)
   {
cout << "Vvedite Npgi: ";
cin >> massivNpg[i];
   }
        cout << endl;         //пустая строка;
             for(int i = 0; i < arraySize; i++)
             {
              cout << massivNpg[i] << "\t";
             }
         cout << endl;        //пустая строка;
         cout << endl;        //пустая строка;
/*Считывание текущего значения Np с АЦП в режиме измерения (здесь – с клавиатуры)*/
         float sumNp;
         float RezIzmer;
```

```
            float massivNp[n];
            maxNp=0;
        for(int i = 0; i < n; i++)
        {
    cout << "Chituvanie c ADC Npi: ";
    cin >> massivNp[i];
        }
            cout << endl;          //пустая строка;
              for(int i = 0; i < n; i++)
              {
               cout << massivNp[i] << "\t";
               sumNp += massivNp[i];
              if (massivNp[i] < minNp) minNp = massivNp[i];    //минимум;
              if (massivNp[i] > maxNp) maxNp = massivNp[i];  //максимум;
              }
            cout << endl;          //пустая строка;
            cout << endl;          //пустая строка;
    float Nps = F_StatObrabotka(sumNp,minNp,maxNp,n);
    /*Проверка попадания Np в диапазон (Npi...Np(i+1)) и, при выполнении условия,
    вычисление результата измерения*/
        for(int i = 0; i < arraySize; i++)
        {
             if (massivNpg[i]<= Nps && Nps <= massivNpg[i+1])
            {
            RezIzmer=((massivNo[i+1]–massivNo[i])*(Nps–
    massivNpg[i]))/(massivNpg[i+1]–massivNpg[i])+ massivNo[i];
            }
    }
            cout << endl;          //пустая строка;
/* автоматический выбор количества разрядов после запятой в полученном результате
измерения (RezIzmer) в зависимости от его попадания в программно - заданный поддиапазон
измерения */;
            if (RezIzmer >0 && RezIzmer <= 10)
              {
              k=2;
              }
            else k=1;
      //Вывод результата измерения с округлением на дисплей;
            cout << endl;                  //пустая строка;
            cout << "Rezultat Izmereniy  =  " << F_Okruglenie(RezIzmer, k) << endl;
//вызов функции округления результата измерения;
            return 0;
          }
      //Описание функции "F_Okruglenie";
            float F_Okruglenie(float RezIzmer,unsigned short k)
            {
```

```
/*внутри функции округления используется стандартная математическая функция возведения в //степень k  числа 10*/
        return floor(RezIzmer * pow(10,k) + .5) / pow(10,k);
    }
//Описание функции "F_StatObrabotka";
    float F_StatObrabotka(float sumNp,float minNp,float maxNp,int n)

    {
       float Nps;
       cout << "Minimalnoe znachenie minNp = "<< minNp << endl; //вывод минимального значения;
        cout << "Maximalnoe znachenie maxNp = "<< maxNp << endl; //вывод максимального значения;
  Nps=(sumNp – minNp – maxNp) / (n–2); //вычисление среднего значения;
       cout << endl;          //пустая строка;
       cout << "Crednee arifmet. Nps = " << Nps;
       return Nps;
    }
```

Результаты работы программы при попадании результата измерения в разные диапазоны приведены на рисунках 4.5 и 4.6.

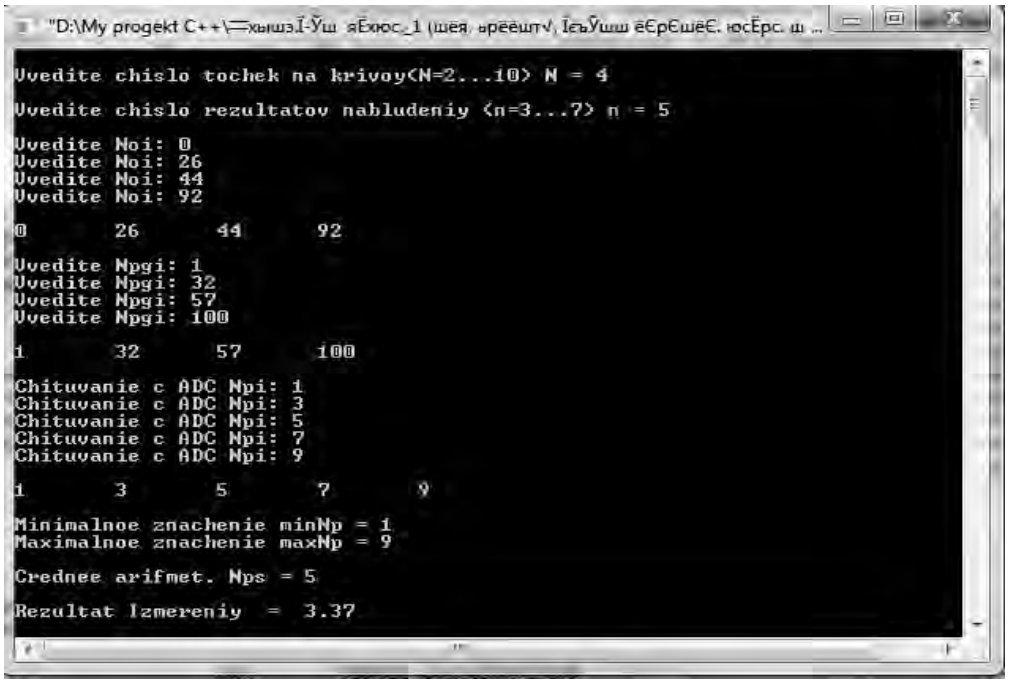

Рисунок 4.5 – Результат выполнения программы в среде программирования CodeBlocks при $N = 4$ и $n = 5$ с элементами статистической обработки и округления результата измерения до двух значащих цифр после запятой

Задача 41. Доработайте программу задачи 40 так, чтобы в заданном диапазоне измерений, разбитом на три поддиапазона, количество разрядов после запятой в полученном результате автоматически округлялось следующим образом: в первом поддиапазоне – до двух разрядов после запятой; во втором

поддиапазоне – до одного разряда после запятой; в третьем поддиапазоне – до целого значения.

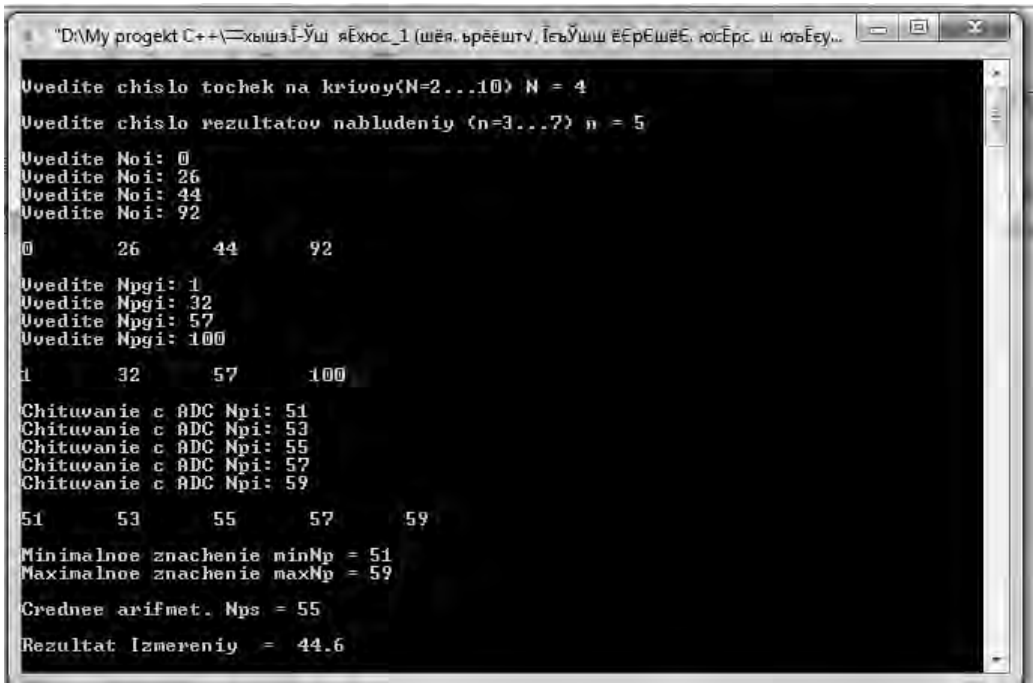

Рисунок 4.6 – Результат выполнения программы в среде программирования CodeBlocks при $N = 4$ и $n = 5$ с элементами статистической обработки и округления результата измерения до одной значащей цифры после запятой

5 Разработка программируемых цифровых устройств на основе микроконтроллеров и их программного обеспечения

Задача 42. Разработайте схему подключения к контроллеру Arduino UNO сенсорного модуля с четырьмя клавишами на базе микросхемы TTP-224 и программу для индикации нажатого состояния каждой клавиши с помощью светодиода. Номер активной клавиши должен также отображаться на мониторе ЭВМ.

Решение

Внешний вид сенсорного модуля на базе микросхемы TTP-224 приведен на рисунке 5.1.

Монтажная схема подключения сенсорного модуля и светодиода к контроллеру Arduino UNO приведена на рисунке 5.2.

Сенсорная панель имеет четыре пронумерованных сенсорных контакта. Сенсорные контакты с номерами от 1 до 4 имеют прямые выходы с логическим уровнем. Во время прикосновения к сенсорному контакту на соответствующем выходе появляется логическая единица.

Обозначение и назначение контактов:

– VCC – контакт для подключения напряжения питания от 3 до 5 В;

− GND − контакт для подключения общего провода;
− OUT1 − цифровой выход от первого сенсорного контакта;
− OUT2 − цифровой выход от второго сенсорного контакта;
− OUT3 − цифровой выход от третьего сенсорного контакта;
− OUT4 − цифровой выход от четвертого сенсорного контакта.

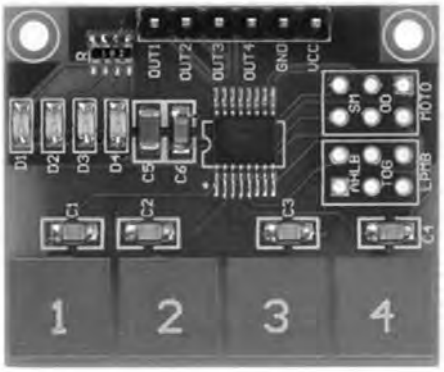

Рисунок 5.1 − Сенсорный модуль на базе микросхемы TTP-224

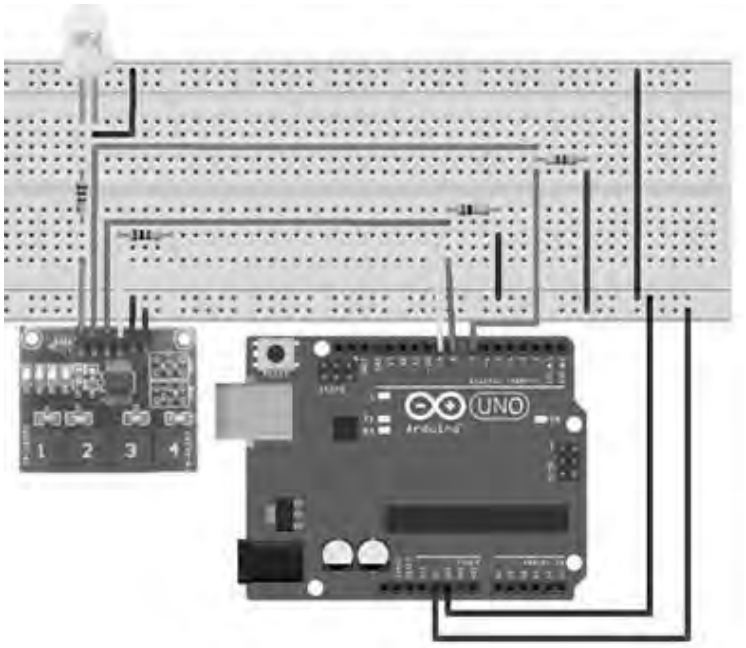

Рисунок 5.2 − Монтажная схема подключения сенсорного модуля и светодиода к контроллеру Arduino UNO

Для индикации нажатого состояния каждой клавиши будем использовать светодиод. Электрическая схема подключения к контроллеру Arduino UNO сенсорного модуля и светодиода приведена на рисунке 5.3.

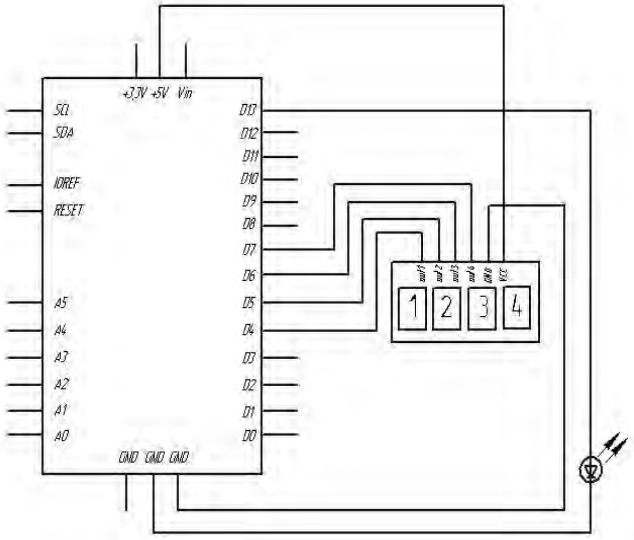

Рисунок 5.3 – Схема подключения сенсорной клавиатуры и светодиода к Arduino

Программа для индикации нажатого состояния каждой клавиши имеет вид:

```
/*Считывает значения напряжения с клавиатуры, если напряжение высокое (произошло касание клавиши), то светодиод загорается*/
//Задаем номера входов
int i;
// К контакту D13 подключен светодиод. Присвоим ему имя:
int led = 13;

void setup()
{
/*инициализируем последовательную передачу данных со скоростью 9600 бит в секунду*/
Serial.begin(9600);
for(i=4; i<=7;i++)
{
pinMode(i, INPUT);    //инициализируем цифровые выводы D4-D7 как входы;
pinMode(led, OUTPUT);  // инициализируем цифровой вывод D13 как выход;
}
}
//основной цикл:
void loop()
{
for(i=4; i<=7;i++)
  {
    if(digitalRead(4) == HIGH)
    {
digitalWrite(led, HIGH);      //команда на включение светодиода;
}
    else if(digitalRead(4) == LOW)
```

```
   {
digitalWrite(led, LOW);       //команда на выключение светодиода;
}
}
 delay(100);   // задержка в промежутке между считываниями для стабильности;
}
```

Выполните программу в среде arduino-1.6.7.

Задача 43. Разработайте схему цифрового прибора на контроллере Arduino UNO для измерения угла поворота и программу для вывода результата измерения на жидкокристаллический индикатор WM-C1602N с клавиатурой.

Решение

Внешний вид жидкокристаллического индикатора (ЖКИ) WM-C1602N с клавиатурой приведен на рисунке 5.4.

Рисунок 5.4 – Внешний вид жидкокристаллического индикатора (ЖКИ) WM-C1602N с клавиатурой

Сочетание в одном модуле жидкокристаллического индикатора, принимающего отображаемую информацию по шине 4 бит, и клавиатуры упрощает разрабатываемый прибор. ЖКИ дисплей имеет регулировку подсветки. Предусмотрены отверстия для установки на переднюю панель прибора. Подключение производится путем стыковки дисплея с платой Arduino UNO.

В качестве датчика угла поворота будем использовать потенциометр с сопротивлением 10 кОм. Если его подключить к источнику питания, то он будет выполнять роль делителя напряжения и с его помощью можно плавно изменять напряжение от 0 до + 5 В, которое подается на аналоговый вход Arduino (рисунок 5.5).

В программе надо использовать библиотеку LiquidCrystal. Она преобразовывает типы данных (из числовых в символьные) перед их выводом на экран.

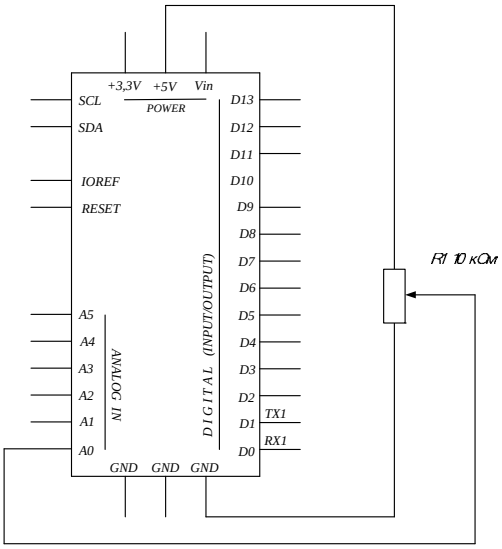

Рисунок 5.5 – Схема подключения датчика угла поворота к контроллеру Arduino

```
//Угол поворота
#include <LiquidCrystal.h>
LiquidCrystal lcd(8, 9, 4, 5, 6, 7);
int inputPin = A1;
float No1 = 0.0;
float No2 = 90.0;
float Nrj1 = 0.0;
float Nrj2 = 344.00;
float x;
int n;
float Nr[15];
float S;
void setup()
{
  Serial.begin(9600);
  lcd.begin(16, 2);
}
void loop()
{
{
  for (n = 0; n < 15; n++)
    Nr[n] = analogRead(inputPin);
}
{
for (n=0;n<15;n++)
    Serial.println(Nr[n]);
}
  S=Nr[0]*15.0/15.0;
   Serial.print("Srednee arifmeticheskoe znachenie - ");Serial.println(S);
    x=(((No2-No1)*(S-Nrj1))/(Nrj2-Nrj1))+No1;
```

```
    Serial.print("Resultat izmerenia - "); Serial.println(x);
    lcd.setCursor(0, 0);
    lcd.print("Ugol povorota =");
    lcd.setCursor(0, 1);
    lcd.print(x);
    lcd.setCursor(6,1);
    lcd.print(" grad");
delay(100);
}
```

Выполните программу в среде Arduino-1.6.7.

Список литературы

1 **Иванов, В. Н.** Электроника и микропроцессорная техника : учебник / В. Н. Иванов, И. О. Мартынова. – Москва : Академия, 2016. – 288 с.

2 **Титов, В. С.** Проектирование аналоговых и цифровых устройств : учебное пособие / В. С. Титов, В. И. Иванов, М. В. Бобырь. – Москва : ИНФРА-М, 2016. – 143 с.

3 **Horowitz, P.** The art of electronics. Third Edition / P. Horowitz, W. Hill. – New York : Cambridge: University Press, 2015. – 1192 с.

Приложение А
(справочное)

Таблица А.1 − Команды и их шестнадцатеричные коды МП MCS-51 в алфавитном порядке

Мнемокод	КОП	Мнемокод	КОП	Мнемокод	КОП
ACALL 0xxH	11	ANL A, R4	5C	DJNZ R1, rel	D9
ACALL 1xxH	31	ANL A, R5	5D	DJNZ R2, rel	DA
ACALL 2xxH	51	ANL A, R6	5E	DJNZ R3, rel	DB
ACALL 3xxH	71	ANL A, R7	5F	DJNZ R4, rel	DC
ACALL 4xxH	91	ANL A, @R0	56	DJNZ R5, rel	DD
ACALL 5xxH	B1	ANL A, @R1	57	DJNZ R6, rel	DE
ACALL 6xxH	D1	ANL A, #d	54	DJNZ R7, rel	DF
ACALL 7xxH	F1	ANL ad, A	52	INC A	04
ADD A, ad	25	ANL ad, #d	53	INC ad	05
ADD A, R0	28	ANL C, bit	82	INC DPTR	A3
ADD A, R1	29	ANL C, /bit	B0	INC R0	08
ADD A, R2	2A	CJNE A, ad, rel	B5	INC R1	09
ADD A, R3	2B	CJNE A, #d, rel	B4	INC R2	0A
ADD A, R4	2C	CJNE R0, #d, rel	B8	INC R3	0B
ADD A, R5	2D	CJNE R1, #d, rel	B9	INC R4	0C
ADD A, R6	2E	CJNE R2, #d, rel	BA	INC R5	0D
ADD A, R7	2F	CJNE R3, #d, rel	BB	INC R6	0E
ADD A, @R0	26	CJNE R4, #d, rel	BC	INC R7	0F
ADD A, @R1	27	CJNE R5, #d, rel	BD	INC @R0	06
ADD A, #d	24	CJNE R6, #d, rel	BE	INC @R1	07
ADDC A, ad	35	CJNE R7, #d, rel	BF	JB bit, rel	20
ADDC A, R0	38	CJNE @R0, #d, rel	B6	JBC bit, rel	10
ADDC A, R1	39	CJNE @R1, #d, rel	B7	JC rel	40
ADDC A, R2	3A	CLR A	E4	JMP @A + DPTR	73
ADDC A, R3	3B	CLR bit	C2	JNB bit, rel	30
ADDC A, R4	3C	CLR C	C3	JNC rel	50
ADDC A, R5	3D	CPL A	F4	JNZ rel	70
ADDC A, R6	3E	CPL bit	B2	JZ rel	60
ADDC A, R7	3F	CPL C	B3	LCALL ad16	12
ADDC A, @R0	36	DA A	D4	LJMP ad16	02
ADDC A, @R1	37	DEC A	14	MOV A, ad	E5
ADDC A, #d	34	DEC ad	15	MOV A, R0	E8
AJMP 0XXH	01	DEC R0	18	MOV A, R1	E9
AJMP 1XXH	21	DEC R1	19	MOV A, R2	EA
AJMP 2XXH	41	DEC R2	1A	MOV A, R3	EB
AJMP 3XXH	61	DEC R3	1B	MOV A, R4	EC
AJMP 4XXH	81	DEC R4	1C	MOV A, R5	ED
AJMP 5XXH	A1	DEC R5	1D	MOV A, R6	EE
AJMP 6XXH	C1	DEC R6	1E	MOV A, R7	EF
AJMP 7XXH	E1	DEC R7	1F	MOV A, @R0	E6
ANL A, ad	55	DEC @R0	16	MOV A, @R1	E7
ANL A, R0	58	DEC @R1	17	MOV A, #d	74
ANL A, R1	59	DIV AB	84	MOV ad, A	F5

Окончание таблицы А.1

Мнемокод	КОП	Мнемокод	КОП	Мнемокод	КОП
ANL A, R2	5A	DJNZ ad, rel	D5	MOV ad, R0	88
ANL A, R3	5B	DJNZ R0, rel	D8	MOV ad, R1	89
MOV ad, R2	8A	MOV @R1, A	F7	SUBB A, ad	95
MOV ad, R3	8B	MOV @R1, ad	A7	SUBB A, R0	98
MOV ad, R4	8C	MOV @R1, #d	77	SUBB A, R1	99
MOV ad, R5	8D	MOVC A, @ + DPTR	93	SUBB A, R2	9A
MOV ad, R6	8E	MOVC A, @ + PC	83	SUBB A, R3	9B
MOV ad, R7	8F	MOVX A, @DPTR	E0	SUBB A, R4	9C
MOV ad, @R0	86	MOVX A, @R0	E2	SUBB A, R5	9D
MOV ad, @R1	87	MOVX A, @R1	E3	SUBB A, R6	9E
MOV ad, #d	75	MOVX @DPTR, A	F0	SUBB A, R7	9F
MOV add, ads	85	MOVX @R0, A	F2	SUBB A, @R0	96
MOV bit, C	92	MOVX @R1, A	F3	SUBB A, @R1	97
MOV C, bit	A2	MUL AB	A4	SUBB A, #d	94
MOV DPTR, #16	90	NOP	00	SWAP A	C4
MOV R0, A	F8	ORL A, ad	45	XCH A, ad	C5
MOV R0, ad	A8	ORL A, R0	48	XCH A, R0	C8
MOV R0, #d	78	ORL A, R1	49	XCH A, R1	C9
MOV R1, A	F9	ORL A, R2	4A	XCH A, R2	CA
MOV R1, ad	A9	ORL A, R3	4B	XCH A, R3	CB
MOV R1, #d	79	ORL A, R4	4C	XCH A, R4	CC
MOV R2, A	FA	ORL A, R5	4D	XCH A, R5	CD
MOV R2, ad	AA	ORL A, R6	4E	XCH A, R6	CE
MOV R2, #d	7A	ORL A, R7	4F	XCH A, R7	CF
MOV R3, A	FB	ORL A, @R0	46	XCH A, @R0	C6
MOV R3, ad	AB	ORL A, @R1	47	XCH A, @R1	C7
MOV R3, #d	7B	ORL A, #d	44	XCHD A, @R0	D6
MOV R4, A	FC	ORL ad, A	42	XCHD A, @R1	D7
MOV R4, ad	AC	ORL ad, #d	43	XRL A, ad	65
MOV R4, #d	7C	ORL C, bit	72	XRL A, R0	68
MOV R5, A	FD	ORL C, /bit	A0	XRL A, R1	69
MOV R5, ad	AD	POP ad	D0	XRL A, R2	6A
MOV R5, #d	7D	PUSH ad	C0	XRL A, R3	6B
MOV R6, A	FE	RET	22	XRL A, R4	6C
MOV R6, ad	AE	RETI	32	XRL A, R5	6D
MOV R6, #d	7E	RL A	23	XRL A, R6	6E
MOV R7, A	FF	RLC A	33	XRL A, R7	6F
MOV R7, ad	AF	RR A	03	XRL A, @R0	66
MOV R7, #d	7F	RRC A	13	XRL A, @R1	67
MOV @R0, A	F6	SETB bit	D2	XRL A, #d	64
MOV @R0, ad	A6	SETB C	D3	XRL ad, A	62
MOV @R0, #d	76	SJMP rel	80	XRL ad, #d	63