МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Физические методы контроля»

ПРОГРАММИРУЕМЫЕ ЦИФРОВЫЕ УСТРОЙСТВА

Методические рекомендации к лабораторным работам для студентов специальности 1-54 01 02 «Методы и приборы контроля качества и диагностики состояния объектов» очной и заочной форм обучения

Могилев 2019

УДК 004.31
ББК 32.973-018
П 78

Рекомендовано к изданию
учебно-методическим отделом
Белорусско-Российского университета

Одобрено кафедрой «Физические методы контроля» «6» февраля 2019 г., протокол № 6

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

Рецензент канд. техн. наук, доц. С. В. Болотов

В методических рекомендациях кратко изложены изучаемые теоретические сведения, приведен порядок выполнения экспериментальных исследований, указана структура отчета о выполненной работе и дан список контрольных вопросов для самопроверки. Составлены в соответствии с учебной программой по дисциплине «Программируемые цифровые устройства» для студентов специальности 1-54 01 02 «Методы и приборы контроля качества и диагностики состояния объектов» очной и заочной форм обучения.

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 21 экз. Заказ №

Издатель и полиграфическое исполнение:
Межгосударственное образовательное учреждение высшего образования
«Белорусско-Российский университет».
Свидетельство о государственной регистрации издателя,
изготовителя, распространителя печатных изданий
№ 1/156 от 07.03.2019.
Пр-т Мира, 43, 212022, Могилев.

Содержание

1 Лабораторная работа № 1. Установка, изучение и начало работы в среде программирования CodeBlocks

Цель работы: овладеть навыками работы в среде программирования CodeBlocks.

1.1 Основные теоретические положения

1.1.1 Среда программирования CodeBlocks. В этой среде есть минимально необходимый комплект (редактор, компилятор и отладчик) для разработки программ. Скачать CodeBlocks можно из интернета. Далее надо:

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

– выбрать путь установки;

– процесс установки завершить, нажав Finish.

Запуск среды программирования CodeBlocks и создание проекта.

Запускается среда программирования CodeBlocks через Пуск –> Все программы –> CodeBlocks –> CodeBlocks или путем щелчка по соответствующему ярлыку на рабочем столе. Главное окно программы после запуска будет иметь такой вид (рисунок 1.1).

Рисунок 1.1 – Главное окно программы CodeBlocks после запуска

При первом запуске программа выдаст сообщение – вопрос: "хотите ли вы ассоциировать файлы исходного кода на C и C++ со средой CodeBlocks", – выбираем в появившемся окне третий вариант, т. е. ассоциировать. Впоследствии файлы с расширениями **.c** и **.cpp** будут автоматически открыты в этой среде.

Создание проекта. Проект – это набор файлов, генерируемых средой программирования, необходимых для последующей компиляции программы (исходные, заголовочные, объектные файлы), а также различные вспомогательные файлы (сохраняющие рабочее пространство и др.). 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++ (рисунок 1.2).

Рисунок 1.2 − Стартовая страница, на которой располагаются кнопки для создания проектов

Далее в следующем окошке жмем "Next", в следующем окне выбираем язык C++, жмем "Next",. Далее вводим название проекта и папку, в которой он будет находиться. Можно, например, для всех проектов создать папку в корне какого-либо диска на компьютере с названием "My project C++"

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

Структуру проекта можно посмотреть 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). Заметим, что 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. Все это можно сделать через меню программы, но удобнее делать через панель (она расположена сверху). 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.

Рассмотрим устройство программы подробнее.

Первая строка – это

```
#include <iostream>
```

#include – это директива препроцессора, она подключает заголовочный файл (файл с расширением **.h**) **iostream.h**, который содержит объявления функций и переменных для потокового ввода и вывода. Имя подключаемого модуля указывается в <> (когда заголовочный файл находится в каталоге \INCLUDE\ конкретной среды разработки), либо в " " (когда заголовочный файл находится в том же каталоге, где и включающий его модуль разрабатываемой программы с расширением **.cpp**. Заголовочные файлы с помощью директивы препроцессора будут подключаться всегда, в зависимости от того, что нужно в программе выполнить. В данном случае нам нужна была функция вывода на экран, которая будет рассматриваться ниже.

Так как язык С++ относится к языкам высокого уровня, то многие инструкции (команды) процессору скрыты, чтобы программисту не приходилось каждый раз программировать то, что уже придумано до него. Поэтому в заголовочных файлах хранятся объявления функций, а мы просто пользуемся этой функцией, не зная ее реализации. Так и в написанной программе мы пользуемся функцией "**cout**", которая управляет потоковым выводом данных (в нашем случае выводит строку "This is my first program!" на экран).

Вторая строка *using namespace std;* всегда должна быть в программе.

Далее следует описание единственной в программе функции **main()**. Абсолютно любая консольная программа на языке С++ обязательно включает в себя эту функцию, именно с нее и начинается выполнение программы. Ключевое слово **int**, расположенное перед именем функции, указывает на то, что по завершении своей работы функция **main** вернет операционной системе целочисленное значение. Это мы даже можем наблюдать в окне нашей программы: ниже строки, которую мы запрограммировали на вывод, у нас среда выводит строку, в которой мы видим returned 0, что означает нормальное завершение работы программы (функции **main**). Если это значение отличное от нуля, то значит, что в нашей программе есть ошибка.

Тело самой функции (оно всегда заключено между скобками { }) содержит строку

```
cout << "This is my first program! " << endl;
```

в которой **cout <<** – это оператор консольного вывода последовательности символов (в нашем случае это предложение "This is my first program!"). И затем оператор **<< endl**, который переводит курсор на следующую строку.

Любая инструкция в языке С++ заканчивается *точкой с запятой*.

return – это оператор возврата из функции. **return 0** – означает, что по окончании своей работы, функция main() вернет значение 0.

Выводы

1 Среда программирования – это программа, в которой программисты разрабатывают свои программы.

2 Основные компоненты среды программирования – это редактор, компилятор и отладчик.

3 В редакторе набирается текст программы. Редактор имеет подсветку синтаксиса конкретного языка программирования.

4 Компилятор переводит программу, набранную в редакторе, в машинный язык, непосредственно понятный компьютеру.

5 Отладчик служит для нахождения ошибок в программе. А ошибки в программах бывают даже у очень опытных программистов.

6 Не забывайте в своих программах с помощью директивы препроцессора подключать необходимые заголовочные файлы. В конце этой строки точка с запятой не ставится, т. к. это не оператор!

7 Абсолютно любая программа на языке C++ содержит в себе функцию **main()**. Именно с нее и начинается построчное выполнение программы.

8 Тело функции main() (как и любых иных функций и управляющих структур в языке C++) обязательно заключается в фигурные скобки {}.

9 Любая инструкция в языке C++ заканчивается точкой с запятой.

1.2 Порядок выполнения работы

1.2.1 Скачать программу CodeBlocks и выполнить её установку по описанной выше методике.

1.2.2 Записать и выполнить в CodeBlocks программу «PCULab.1.1», код которой приведен ниже. В коде программы надо дописать выводимый текст и комментарии к выполняемым операциям.

```
/* Программа «PCULab.1.1*/
#include <iostream>
using namespace std;
int main()
{
cout<< "My name is . I live in . I am student ." <<endl;
return 0;
}
```

1.2.3 Записать и выполнить в CodeBlocks программу «PCULab.1.2», код которой приведен ниже. Записать в коде программы комментарии к выполняемым операциям.

```
/* Программа «PCULab.1.2. Вывод чисел разными способами */
#include <iostream>
```

```
using namespace std;
int main()
{
cout<< "1 2 3 4" << endl; //первый способ;
cout<< "1   " << "2   " << "3   " << "4   " << endl; //второй способ;
cout<< "1" << endl;//третий способ;
cout<< "2" << endl;
cout<< "3" << endl;
cout<< "4" << endl;
return 0;
}
```

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, коды программ, блок-схемы алгоритмов программ, выводы по результатам их выполнения.

Контрольные вопросы

1 Что называют средой программирования?
2 Назовите основные компоненты среды программирования.
3 Где набирается текст программы?
4 Какую функцию выполняет компилятор?
5 Какую функцию выполняет отладчик?
6 С какой функции начинается построчное выполнение программы?
7 В какие скобки заключается тело функции main()?
8 Чем заканчивается любая инструкция в языке С++?

2 Лабораторная работа № 2. Исследование основных операторов и структур программ на языке С++

Цель работы: исследовать назначение и особенности использования основных операторов языка С++ для типовых структур программ.

2.1 Основные теоретические положения

Основные операторы и структуры программ на языке С++. Понятие «**переменная**» в языке С++. «Переменная» – это выделенная программистом при составлении программы область в памяти компьютера, в которой будут храниться её конкретные значения. Значения переменных могут быть числовыми, символьными, логическими.

Объявить переменную можно в любом месте программы до ее первого использования. Однако, если объявлять переменные в начале функции main(), читаемость программы будет лучше. При запуске программа резервирует в

оперативной памяти компьютера столько места, сколько необходимо для хранения объявленных переменных. Типы переменных в C++, их допустимые значения и объём занимаемой памяти в байтах представлены в таблице 2.1.

Таблица 2.1 – Типы переменных языка C++

Имя типа	Размер в байтах	Допустимое значение
char	1 байт	От –128 до 127
int	*Зависит от реализации*	
short int	2 байта	От -32 768 до 32 767
long int	4 байта	От –2 147 483 648 до 2 147 483 647
unsigned char	1 байт	От 0 до 255
unsigned int	*Зависит от реализации*	
unsigned short int	2 байта	От 0 до 65 535
unsigned long int	4 байта	От 0 до 4 294 967 295
float	4 байта	От ~3,4·e^{-38} до ~3,4·e^{+38}
doubl	8 байт	От ~1,7·e^{-308} до ~1,7·e^{+308}
long doubl	10 байт	От ~3,4·e^{-4932} до ~1,1·e^{+4932}

Операторы, используемые в C++. Математические функции библиотеки math.h языка программирования C++ приведены в таблице 2.2. Операторы символьных констант приведены в таблице 2.3.

Таблица 2.2 – Математические функции библиотеки **math.h** в C++

Функция	Описание работы функции
double **sqrt** (double **x**);	Данная математическая функция вычисляет и возвращает корень из положительного числа, принимая его в качестве аргумента **x**
double **pow** (double **x**, double **y**);	Возводит число, принимаемое в качестве аргумента **x**, в степень, принимаемую в качестве аргумента **y**
double **fabs** (double **x**);	Вычисляет абсолютное значение числа **x** (иными словами, его модуль)
double **fmod** (double **x**, double **y**);	Вычисляет остаток от деления **x** на **y**
double **ceil** (double **x**);	Вычисляет наименьшее целое, значение которого не будет меньшим, чем **x**. Например, функция **ceil (4.68)** вернет значение **5.00**
double **floor** (double **x**);	В отличии от предыдущей функции эта функция вычисляет наибольшее целое, по значению не превосходящее **x**. Например, функция **floor (4.68)** вернет значение **4.00**
double **modf** (double **value**, double***ptr**);	Разбивает значение аргумента **value** на целую и дробные части. Целую часть функция сохраняет в объекте, на который указывает указатель ***ptr**, а дробную возвращает
double **cos** (double **x**);	Вычисляет косинус аргумента **x**, который задается в **радианах**
double **sin** (double **x**);	То же, но синус

Окончание таблицы 2.2

Функция	Описание работы функции
double **tan** (double **x**);	То же, но тангенс
double **acos** (double **x**);	Вычисляет главное значение арккосинуса **x**. Аргумент **x** должен быть из интервала [–1; +1]. Функция возвращает значение в радианах из интервала [0; π]
double **asin** (double **x**);	Вычисляет главное значение арксинуса **x**. Аргумент **x** должен быть из интервала [–1; +1]. Функция возвращает значение в радианах из интервала [–π/2; + π/2]
double **atan** (double **x**);	Вычисляет главное значение арктангенса **x**
double **exp** (double **x**);	Вычисляет значение показательной функции аргумента **x**
double **log** (double **x**);	Вычисляет натуральный логарифм аргумента **x**
double **log10** (double **x**);	Вычисляет десятичный логарифм аргумента **x**

Таблица 2.3 – Операторы символьных констант, используемые в С++

Символьные константы	Обозначение действий
\n	Переход на новую строку
\t	Табуляция (на 6 символов)
\0	Нулевой символ
\\	Обратная косая черта
\v	Вертикальная табуляция (на 6 позиций вниз)
\f	На новой строке в том же месте
\a	Подача звукового сигнала

Для выполнения арифметических и логических действий над переменными в языке С++ используются операторы, приведенные в таблице 2.4.

Таблица 2.4 – Арифметические и логические операторы, используемые в языке С++

Название операции	Используемый в коде символ
Арифметическое сложение	+
Арифметическое вычитание	–
Умножение	*
Деление	/
Отрицание	!
Присваивание	=
Вычисление остатка	%
Логическое сложение	&&
Логическое умножение	||
Проверка на равенство	
Равно	= =
Не равно	! =

Окончание таблицы 2.4

Название операции	Используемый в коде символ
Проверка отношения	
Больше	>
Меньше	<
Больше или равно	> =
Меньше или равно	< =

Математические функции в языке C++. Все математические функции принимают в качестве аргументов и возвращают числа с плавающей точкой двойной точности **double**. Иными словами, принимают дробные числа, которые могут содержать после запятой 15…16 разрядов. Это не значит, что при использовании данных функций нужно обязательно пользоваться дробными числами – можно использовать любые числа. Программа сама выполнит необходимые приведения типов.

2.2 Порядок выполнения работы

2.2.1.Выполнить программу «PCULab.2.1», код которой приведен ниже. Записать в коде программы-комментарии к выполняемым операциям.

```
/* Программа «PCULab.2.1», которая просит от пользователя ввести два числа, а потом выводит на экран результаты вычислений: сумму, разность, произведение и частное */
#include <iostream>
using namespace std;
int main()
{
setlocale(LC_CTYPE,"Russian"); // вывод русского текста
int a, b;
cout<< "Введите число a: " <<endl;
cin>> a;
cout<< "Введите число b: " <<endl;
cin>> b;
cout<< "Сумма " << a << " и " << b << " равна " << a + b <<endl;
cout<< "Разность " <<a<< " и " <<b<< " равна " <<a - b<<endl;
cout<< "Произведение " <<a<< " и " <<b<< " равно " <<a * b<<endl;
cout<< "Частное " << a << " и " << b << " равно " << a / b <<endl;
return 0;
}
```

2.2.2 Составить программу «PCULab.2.2», которая просит от пользователя ввести два числа, а потом выводит на экран результаты логических операций над ними: И, ИЛИ, НЕ. Записать в коде программы комментарии к выполняемым операциям.

/* Программа «PCULab.2.2», которая просит от пользователя ввести два числа, а потом выводит на экран результаты логических операций над ними: И, ИЛИ, НЕ */

```
#include <iostream>
using namespace std;
int main()
{
setlocale(LC_CTYPE,"Russian"); // вывод русского текста
int a, b;
a=8;
b=4;
//cout<< "Введите  число a: " <<endl;
//cin>> a;
//cout<< "Введите число b: " <<endl;
//cin>> b;
unsignedint c=a&b;
unsignedint d=a|b;
unsignedint e=~a;
unsignedint g=~b;
cout<< "Логическое И над числами " <<a<< " и " <<b<< " равно " <<c<<endl;
cout<< "Логическое ИЛИ над числами " <<a<< " и " <<b<< " равно " <<d<<endl;
cout<< "Логическое НЕ над числом " <<a<< " равно " <<e<<endl;
cout<< "Логическое НЕ над числом " <<b<< " равно " <<g<<endl;
return 0;
}
```

2.2.3 В программе «PCULab.2.2» измените код, чтобы числа a и b вводились с клавиатуры.

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, измененный код программ «PCULab.2.1», «PCULab.2.2», результаты их выполнения, блок-схемы алгоритмов этих программ, выводы.

Контрольные вопросы

1 Опишите результат выполнения логической операции И над одноразрядными двоичными числами.

2 Опишите результат выполнения логической операции ИЛИ над одноразрядными двоичными числами.

3 Опишите результат выполнения логической операции НЕ над одноразрядным двоичным числом.

4 Опишите результат выполнения логической операции ИСКЛЮЧАЮЩЕЕ ИЛИ над одноразрядными двоичными числами.

3 Лабораторная работа № 3. Разработка и исследование программного обеспечения на С++ для цифрового прибора с линейной функцией преобразования

Цель работы: исследовать работу программного обеспечения цифрового прибора на микроконтроллере с линейной функцией преобразования.

3.1 Основные теоретические положения

Чтобы проектируемый 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). Это может быть сделано путем одновременного воздействия измеряемой величиной X на РСИ и ОСИ (рисунок 3.1).

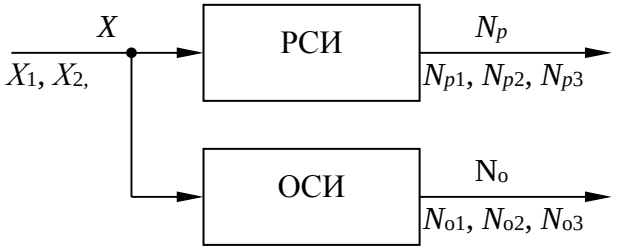

Рисунок 3.1 – Градуировка РСИ с использованием ОСИ

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 3.2).

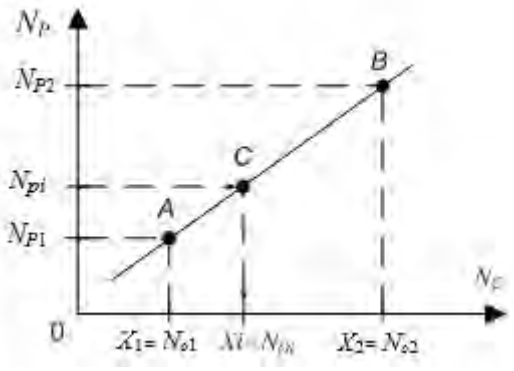

Рисунок 3.2 – Градуировочная характеристика РСИ

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:

$$X_i = \frac{(N_{o2} - N_{o1})(N_{pi} - N_{p1})}{N_{p2} - N_{p1}} + N_{o1}, \qquad (3.1)$$

где N_{pi} – числовое значение, получаемое с помощью РСИ при воздействии на него величиной X_i при её измерении.

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.

3.2 Порядок выполнения работы

3.2.1 Разработайте программу, которая будет обеспечивать:

– отображение в диалоговом окне выражения «Vvedite pokazanie pribora *Npi* =»;

– ввод с клавиатуры числа *Npi*;

– вычисление результата измерения с использованием выражения (3.1);

– отображение в диалоговом окне выражения «Rezultat izmereniy X =».

В качестве исходных данных используйте координаты точек A и B: N_{01} = 10, N_{P1} = 70, N_{02} = 74, N_{P2} = 83.

3.2.2 Запустите среду программирования CodeBlocks.

3.2.3 Создайте новый проект и присвойте ему имя «PCULab.3.1».

3.2.4 Вставьте в созданный проект разработанный код программы.

3.2.5 Запустите процесс компиляции программы.

3.2.6 Запустите процесс выполнения программы, выбрав курсором кнопку в виде треугольника ("Run").

3.2.7 С помощью клавиатуры сделайте ввод числа, представляющего собой виртуальную реакцию РСИ на воздействие на него измеряемой величины X. Далее нажмите клавишу "Enter". В окошке должен появиться результат выполнения написанной программы.

3.2.8 Проделайте 3…5 раз действия, прописанные в п. 3.2.6, с разными числами, вводимыми с помощью клавиатуры.

3.2.9 Составьте блок-схему алгоритма программы «PCULab.3.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.3.1», блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Какое значение имеет запись в программе «#include<math.h»?
2 Какие типы данных используются в исследуемой программе?
3 Какие действия выполняются по команде «\n"»?
4 Какие действия выполняются по команде «cin >> *Npi*;»?
5 Какие действия выполняются по команде «X = (((*N*02 – *N*01) * (*Npi* –*N*P1)) / /(*N*P2 – *N*P1)) + *N*01;»?

4 Лабораторная работа № 4. Разработка и исследование программного обеспечения для РСИ с линейной функцией преобразования с использованием массивов данных

Цель работы: разработать и исследовать работу программного обеспечения цифрового прибора на микроконтроллере с линейной функцией преобразования с использованием массивов данных.

4.1 Основные теоретические положения

Программа составляется для линейной функции преобразования РСИ.
Алгоритм вычисления величины X при линейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 3.

4.2 Порядок выполнения работы

4.2.1 Запустите среду программирования CodeBlocks.
4.2.2 Создайте новый проект и присвойте ему имя «PCULab.4.1».
4.2.3 Разработайте программу на C++, с помощью которой можно:
1) выполнить ввод числа точек N на градуировочной характеристике с проверкой правильности ввода;
2) выполнить ввод числа результатов наблюдений n из диапазона (5…7) с проверкой правильности ввода;
3) сформировать массив N_O[ ] показаний образцового прибора путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;
4) сформировать массив N_{PG}[ ] показаний рабочего средства измерения путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;
5) сформировать массив N_P[ ] результатов наблюдений с помощью рабочего средства измерения n = (5…7) путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;
6) найти сумму элементов массива N_P[ ] sumN_P[ ];

7) найти среднее арифметическое Nps элементов массива N_P[ ] по формуле

Nps = sumNp / n

и отобразить Nps на экране;

8) рассчитать результат измерения по формуле (3.1), используя выражение

RezIzmer = ((massivNo[1]-massivNo[0])*(Nps-massivNpg[0]))/
/(massivNpg[1]-massivNpg[0])+massivNo[0],

и отобразить RezIzmer на экране.

4.2.4. Запустить процесс компиляции программы.

4.2.5 Запустить процесс выполнения программы, выбрав курсором кнопку в виде треугольника ("Run").

4.2.6 С помощью клавиатуры сделать ввод числа точек *N*, ввод числа результатов наблюдений *n*, ввод элементов массива N_O[ ], ввод элементов массива N_{PG}[ ], ввод элементов массива N_P[ ].

Далее нажмите клавишу «Enter». В окошке должен появиться результат выполнения написанной программы (рисунок 4.1).

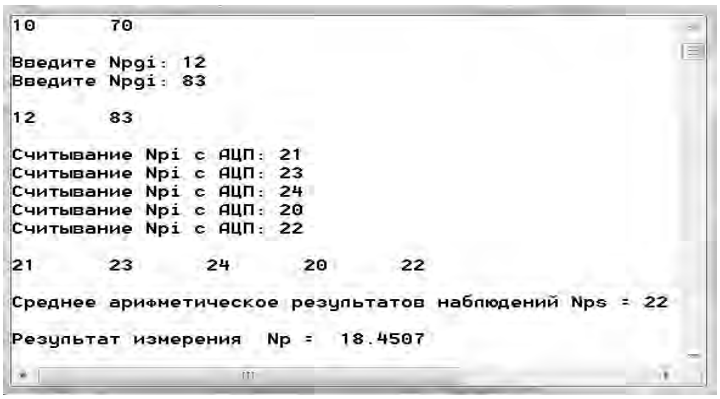

Рисунок 4.1 − Окно с результатом выполненной программы «PCULab.4»

4.2.7 Проделайте 3…5 раз действия, прописанные в п. 4.2.6, с разными числами, вводимыми с помощью клавиатуры.

4.2.8 Составьте блок-схему алгоритма программы «PCULab.4».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.4», блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Какие типы данных используются в исследуемой программе?

2 Как отделяются от кода программы комментарии?

5 Лабораторная работа № 5. Разработка и исследование программного обеспечения на C++ для цифрового прибора с нелинейной функцией преобразования

Цель работы: разработать и исследовать работу программного обеспечения цифрового прибора на микроконтроллере с нелинейной функцией преобразования.

5.1 Основные теоретические положения

В большинстве случаев функция преобразования РСИ является нелинейной (рисунок 5.1).

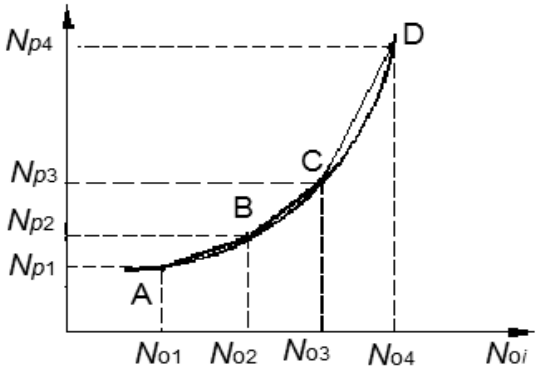

Рисунок 5.1 – Градуировочная характеристика РСИ с нелинейной функцией преобразования

Поэтому определение координат двух точек при его градуировке оказывается недостаточным. 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 (3.1) (см. рисунок 5.1).

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

Для отрезка АВ с учетом (3.1) уравнение прямой запишется в виде:

$$X = \frac{(N_{o2} - N_{o1})(N_p - N_{p1})}{N_{p2} - N_{p1}} + N_{o1}. \qquad (5.1)$$

Для отрезка ВС

$$X = \frac{(N_{o3} - N_{o2})(N_p - N_{p2})}{N_{p3} - N_{p2}} + N_{o2}. \qquad (5.2)$$

Для отрезка CD

$$X = \frac{(N_{o4} - N_{o3})(N_p - N_{p3})}{N_{p4} - N_{p3}} + N_{o3}. \quad (5.3)$$

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_{Pi} 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

$$X = \frac{N_{O(i+1)} - N_{Oi}}{N_{P(i+1)} - N_{Pi}}(N_P - N_{pi}) + N_{oi}. \quad (5.4)$$

5.2 Порядок выполнения работы

5.2.1 Создайте в CodeBlocks новый проект и присвойте ему имя «PCULab.5.1».

5.2.2. Вставьте в проект «PCULab.5.1» приведенный ниже код. Запишите в коде программы комментарии к выполняемым операциям.

```
// Программа «PCULab.5.1»
// Вычисление измеряемой величины в приборе с
// нелинейной функцией преобразования (кривая аппроксимирована
// тремя прямолинейными отрезками;
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
setlocale(LC_CTYPE,"Russian"); // Записать комментарий;
// Записать комментарий;
   float No1,No2,Np1,Np2,No3,Np3,No4, Np4,RezIzmer;
   int Np;
// Записать комментарий;
   No1 = 20;
   No2 = 40;
   No3 = 80;
   No4 = 110;
   Np1 = 30;
   Np2 = 50;
   Np3 = 90;
   Np4 = 130;
// Записать комментарий;
  cout << "Считывание данных с АЦП: \n";
```

```
  cout << endl; // Записать комментарий;
  cin >> Np;
// Записать комментарий;
   if (Np1<=Np && Np<=Np2)
   {
      RezIzmer=((No2-No1)*(Np-Np1))/(Np2-Np1)+No1;
   }
 // Записать комментарий;
   else
      if (Np2<Np && Np<=Np3)
         {
            RezIzmer=((No3-No2)*(Np-Np2))/(Np3-Np2)+No2;
         }
      else
      // Записать комментарий;
            RezIzmer=((No4-No3)*(Np-Np3))/(Np4-Np3)+No3;
// Записать комментарий;
  cout << endl; // Записать комментарий;
  cout << "Результат измерения  =  " << RezIzmer  << endl;
  return 0;
}
```

5.2.3 Выполните компиляцию программы и после её завершения запустите процесс выполнения программы, выбрав "Run". Должно появиться окошко программы (рисунок 5.2).

Рисунок 5.2 – Окно исполняемой программы «PCULab.5.1»

5.2.4 С помощью клавиатуры сделайте ввод числа, представляющего собой виртуальную реакцию РСИ на воздействие на него измеряемой величины X. Далее нажмите клавишу «Enter». В окошке должен появиться результат выполнения написанной программы, как показано на рисунке 5.3.

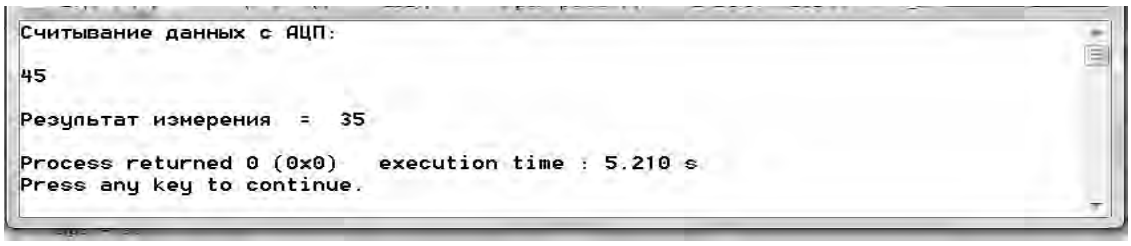

Рисунок 5.3 – Окно с результатом выполненной программы «PCULab.5.1»

5.2.5 Проделайте 3…5 раз действия, прописанные в п. 5.2.4, с разными числами, вводимыми с помощью клавиатуры.

5.2.6 Составьте блок-схему алгоритма программы «PCULab.5.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.5.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Каким количеством прямолинейных отрезков аппроксимирована кривая в программе «PCULab.5.1»?
2 Какие типы данных используются в исследуемой программе?
3 Какие действия выполняются по команде«if (Np1 <= Np&&Np <= Np2);»?
4 Какие действия выполняются по команде «cout << endl << '\t';»?

6 Лабораторная работа № 6. Разработка и исследование программного обеспечения для РСИ с нелинейной функцией преобразования с использованием массивов измерительной информации

Цель работы: разработать и исследовать работу программного обеспечения цифрового прибора на микроконтроллере с нелинейной функцией преобразования с использованием массивов измерительной информации.

6.1 Основные теоретические положения

Программа составляется для РСИ с нелинейной функцией преобразования.
Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 5.

6.2 Порядок выполнения работы

6.2.1 Создайте в CodeBlocks новый проект и присвойте ему имя «PCULab.6.1».
6.2.2 Разработайте программу на C++, с помощью которой можно:
– выполнить ввод числа точек N (N = 3…5) на градуировочной характеристике;
– выполнить ввод числа результатов наблюдений n из диапазона (5…7);
– сформировать массив N_O[ ] показаний образцового прибора путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;
– сформировать массив N_{PG}[ ] показаний рабочего средства измерения путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– сформировать массив N_P[ ] результатов наблюдений с помощью рабочего средства измерения n = (5…7) путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– найти сумму элементов массива N_P[ ] sumN_P[ ];

– найти среднее арифметическое Nps элементов массива N_P[ ]

$$Nps = sumNp / n$$

и отобразить Nps на экране;

– выполнить проверку попадания Nps в диапазон Npi...Np(i + 1) и при выполнении условия вычислить результат измерения по формуле (5.4), используя выражение

$$RezIzmer = ((massivNo[i+1] – massivNo[i])*(Nps – massivNpg[i]))/(massivNpg[i+1] – massivNpg[i]) + massivNo[i],$$

и отобразить RezIzmer на экране.

6.2.3 Запустите процесс компиляции программы.

6.2.4 Запустите процесс выполнения программы, выбрав курсором кнопку "Run".

6.2.5 С помощью клавиатуры сделайте ввод числа точек N, ввод числа результатов наблюдений *n*, ввод элементов массива N_O[ ], ввод элементов массива N_{PG}[ ], ввод элементов массива N_P[ ].

Результат выполнения написанной программы (как вариант) показан на рисунке 6.1.

6.2.6 Проделайте 3…5 раз действия, прописанные в п. 6.2.5, с разными числами, вводимыми с помощью клавиатуры.

6.2.7 Составьте блок-схему алгоритма программы «PCULab.6.1».

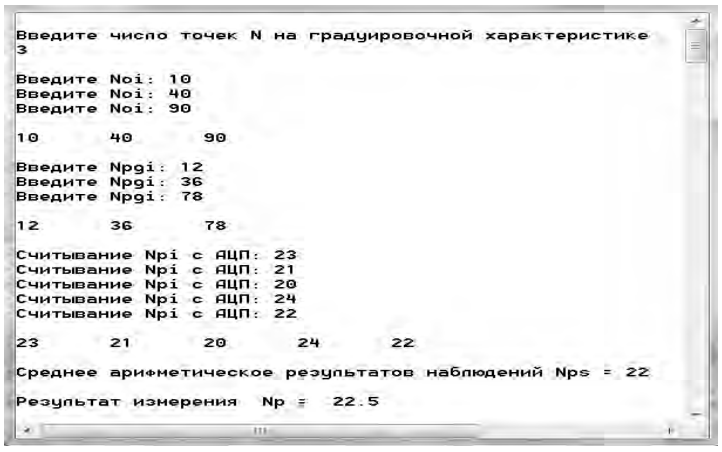

Рисунок 6.1 − Окно с результатом выполненной программы «PCULab.6.1»

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.6.1», блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Каким количеством прямолинейных отрезков аппроксимирована кривая в программе «PCULab.6.1»?

2 Какие типы данных используются в исследуемой программе?

3 Какие действия выполняются по команде «if (Np1 <= Np&&Np <= Np2);»?

4 Какие действия выполняются по команде «cout << endl << '\t';»?

7 Лабораторная работа № 7. Разработка и исследование программного обеспечения для цифрового прибора с нелинейной функцией преобразования, аппроксимируемой N–1 прямолинейными отрезками

Цель работы: разработать и исследовать работу программного обеспечения для цифрового прибора на микроконтроллере с нелинейной функцией преобразования, аппроксимируемой N – 1 прямолинейными отрезками.

7.1 Основные теоретические положения

Программа составляется для РСИ с нелинейной функцией преобразования.

Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 5.

Программа также выполняет вычисление среднего арифметического результатов наблюдений, что позволяет уменьшить случайную составляющую погрешности результата измерения.

Измеряемая величина X рассчитывается с помощью выражения

$$X = ((massivNo[i + 1] – massivNo[i])*(Nps – massivNpg[i]))/(massivNpg[i + 1] – – massivNpg[i]) + massivNo[i];, \quad (7.1)$$

где Nps – среднее арифметическое результатов наблюдений; Nps = sumNp / n;

sumNp – сумма результатов наблюдений;

n – число наблюдений измеряемой величины;

massivNo[i], massivNpg[i] – массивы показаний образцового и рабочего средств измерений, формируемые при градуировке РСИ и записываемые в его ППЗУ.

7.2 Порядок выполнения работы

7.2.1 Создайте новый проект и присвойте ему имя «PCULab.7.1».

7.2.2 Вставьте в созданный проект нижеприведенный код программы, запишите к ней комментарии.

```
/* PCULab.7.1. Вычисление измеряемой величины в приборе с
 нелинейной функцией преобразования (кривая аппроксимирована
 (N–1) прямолинейными отрезками (N – число точек на кривой);*/
#include <iostream>
//#include <math.h>
using namespace std;
int main()
{
 // Записать комментарий,
int i; // Записать комментарий;
int N; // Записать комментарий;
cout<<endl;
cout<< "Vveditechislotocheknakrivoy N = ";
cin>> N;
cout<<endl; // Записать комментарий;
unsigned short intarraySize = N;
floatmassivNo[arraySize];
for(inti = 0; i<arraySize; i++)
   {
            cout<< "VvediteNoi: ";
            cin>>massivNo[i];
  }
cout<<endl; // Записать комментарий;
for(inti = 0; i<arraySize; i++)
              {
cout<<massivNo[i] << "\t";
              }
cout<<endl; // Записать комментарий;
floatmassivNpg[arraySize];
for(inti = 0; i<arraySize; i++)
   {
            cout<< "VvediteNpgi: ";
            cin>>massivNpg[i];
  }
cout<<endl; // Записать комментарий;
for(inti = 0; i<arraySize; i++)
{
cout<<massivNpg[i] << "\t";
              }
cout<<endl; // Записать комментарий;
// Записать комментарий;
int n = 5;
floatsumNp;
floatRezIzmer;
floatmassivNp[n];
for(inti = 0; i< n; i++)
   {
cout<< "Chituvanie c ADC Npi: ";
  cin>>massivNp[i];
   }
cout<<endl; // Записать комментарий;
for(inti = 0; i< n; i++)
```

```
            {
        cout<<massivNp[i] << "\t";
        sumNp += massivNp[i];
            }
        cout<<endl; // Записать комментарий;
        floatNps = sumNp / n;
          cout<< "Credneearifmet. Nps = " <<Nps;
        // Записать комментарий;
        for(inti = 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; // Записать комментарий;
        // Записать комментарий;
        cout<< "RezultatIzmereniy  =  " <<RezIzmer<<endl;
        return 0;
        }
```

7.2.3 Запустите процесс компиляции программы, затем её выполнение.

7.2.4 Введите в программу команду для вывода на дисплей текста на русском языке. Замените в программе выводимый текст на русский.

7.2.5 Составьте блок-схему алгоритма программы «PCULab.7.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.7.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Каким количеством прямолинейных отрезков аппроксимирована кривая в программе «PCULab.7.1»?

2 Какие типы данных используются в исследуемой программе?

3 Как задается количество наблюдений измеряемой величины в программе «PCULab.7.1»?

4 Чем обеспечивается в программе вывод на дисплей текста на русском языке?

8 Лабораторная работа № 8. Разработка и исследование программного обеспечения для РСИ с проверкой вводимых данных в режиме градуировки

Цель работы: разработать и исследовать работу программного обеспечения для РСИ на микроконтроллере с нелинейной функцией преобразования, имеющей дополнительные функции при его градуировке.

8.1 Основные теоретические положения

Программа составляется для РСИ с нелинейной функцией преобразования.

Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 7.

Число точек на кривой функции преобразования прибора и число наблюдений измеряемой величины вводится с клавиатуры при его градуировке; указаны диапазоны их значений и после ввода проверяется проверка правильности ввода; если вводимое значение не попадает в рекомендуемый диапазон, предлагается повторить ввод.

8.2 Порядок выполнения работы

8.2.1 Создайте новый проект и присвойте ему имя «PCULab.8.1».

8.2.2 Вставьте в созданный проект нижеприведенный код программы, запишите к ней комментарии.

```
/* PCULab.8.1 */
 #include <iostream>
 using namespace std;
 intmain()
 {
  // Записать комментарий;
 unsignedshort n; // Записать комментарий;
 unsignedshort N; /* Записать комментарий;*/
 cout<<endl; // пустая строка;
 do{
 cout<< "Vveditechislotocheknakrivoy(N=2...10) N = ";
 cin>> N;
 } while (N > 10||N <2 );
 unsigned short arraySize = N;
 cout<<endl; // пустая строка;
 do{
 cout<< "Vveditechislorezultatovnabludeniy (n=3...5) n = ";
 cin>> n;
 } while (n <3||n >5 );
 cout<<endl; // пустая строка;
 floatmassivNo[arraySize];
 for(inti = 0; i<arraySize; i++)
    {
```

```
            cout<< "VvediteNoi: ";
            cin>>massivNo[i];
        }
        cout<<endl; // пустая строка;
        for(inti = 0; i<arraySize; i++)
                {
        cout<<massivNo[i] << "\t";
                }
        cout<<endl; // пустая строка;
        floatmassivNpg[arraySize];
        for(inti = 0; i<arraySize; i++)
          {
            cout<< "VvediteNpgi: ";
            cin>>massivNpg[i];
        }
        cout<<endl; // пустая строка;
        for(inti = 0; i<arraySize; i++)
                {
        cout<<massivNpg[i] << "\t";
                }
        cout<<endl; // пустая строка;
        // Записать комментарий;
        floatsumNp;
        floatRezIzmer;
        floatmassivNp[n];
        for(inti = 0; i< n; i++)
          {
            cout<< "Chituvanie c ADC Npi: ";
            cin>>massivNp[i];
        }
        cout<<endl; // пустая строка;
        for(inti = 0; i< n; i++)
                {
        cout<<massivNp[i] << "\t";
        sumNp += massivNp[i];
        }
        cout<<endl; // пустая строка;
        cout<<endl; // пустая строка;
        floatNps = sumNp / n;
        cout<< "Credneearifmet. Nps = " <<Nps;
        // Записать комментарий;
        for(inti = 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;
        // Записать комментарий;
        cout<<endl; // пустая строка;
```

```
cout<< "RezultatIzmereniy  =  " <<RezIzmer<<endl;
return 0;
}
```

8.2.3 Запустите процесс компиляции программы, затем её выполнение.

8.2.4 Введите в программу команду для вывода на дисплей текста на русском языке. Замените в программе выводимый текст на русский.

8.2.5 Составьте блок-схему алгоритма программы «PCULab.8.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.8.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Какие действия выполняются по команде

«do{
cout<< "Vveditechislotocheknakrivoy(N=2...10) N = ";
cin>> N;
} while (N > 10||N <2);»?

2 Какие действия выполняются по команде

«do{
cout<< "Vveditechislorezultatovnabludeniy (n=3...5) n = ";
cin>> n;
} while (n <3||n >5);»?

9 Лабораторная работа № 9. Разработка и исследование программного обеспечения на C++ для РСИ с элементами статистической обработки данных

Цель работы: разработать и исследовать работу программного обеспечения для РСИ с нелинейной функцией преобразования, имеющего дополнительные функции по обработке данных.

9.1 Основные теоретические положения

Программа составляется для РСИ с нелинейной функцией преобразования.

Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 8.

Число точек на кривой функции преобразования прибора и число наблюдений измеряемой величины вводится с клавиатуры при его градуировке; указаны диапазоны их значений и после ввода проверяется 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.

9.2 Порядок выполнения работы

9.2.1 Создайте новый проект и присвойте ему имя «PCULab.9.1».

9.2.2 Вставьте в созданный проект нижеприведенный код программы, запишите к ней комментарии.

```
/* PCULab.9.1 */
  #include <iostream>
  using namespace std;
  intmain()
  {
  unsignedshort n; // Записать комментарий;
  unsignedshort N; // Записать комментарий;
  floatminNp,maxNp; // Записать комментарий;
  cout<<endl; // пустая строка;
  do{
  cout<< "Vveditechislotocheknakrivoy(N=2...10) N = ";
  cin>> N;
  } while (N > 10 || N <2 );
  unsigned short arraySize = N;
  cout<<endl; // пустая строка;
  do{
  cout<< "Vveditechislorezultatovnabludeniy (n=3...7) n = ";
  cin>> n;
  } while (n <3||n >7 );
  cout<<endl; // пустая строка;
  floatmassivNo[arraySize];
  for(inti = 0; i<arraySize; i++)
     {
  cout<< "VvediteNoi: ";
  cin>>massivNo[i];
      }
  for(inti = 0; i<arraySize; i++)
                {
  cout<<massivNo[i] << "\t";
                }
  cout<<endl; // пустая строка;
  floatmassivNpg[arraySize];
  for(inti = 0; i<arraySize; i++)
     {
           cout<< "VvediteNpgi: ";
           cin>>massivNpg[i];
     }
  cout<<endl; // пустая строка;
  for(inti = 0; i<arraySize; i++)
                {
```

```
cout<<massivNpg[i] << "\t";
            }
cout<<endl; // пустая строка;
// Записать комментарий;
floatsumNp;
floatRezIzmer;
floatmassivNp[n];
maxNp=0;
for(inti = 0; i< n; i++)
   {
cout<< "Chituvanie c ADC Npi: ";
cin>>massivNp[i];
   }
cout<<endl; // пустая строка;
for(inti = 0; i< n; i++)
         {
cout<<massivNp[i] << "\t";
sumNp += massivNp[i];
if (massivNp[i] <minNp) minNp = massivNp[i]; // Записать комментарий;
if (massivNp[i] >maxNp) maxNp = massivNp[i]; // Записать комментарий;
}
cout<<endl; // пустая строка;
cout<<endl; // пустая строка;
cout<< "MinimalnoeznacheninNp = "<<minNp<<endl; // Записать комментарий;
cout<< "MaximalnoeznacheniemaxNp = "<<maxNp<<endl; // Записать комментарий;
cout<<endl; // пустая строка;
floatNps = (sumNp – minNp – maxNp) / (n–2); // Записать комментарий;
cout<< "Credneearifmet. Nps = " <<Nps;
// Записать комментарий;
for(inti = 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;
// Записать комментарий;
cout<<endl; // пустая строка;
cout<< "RezultatIzmereniy  =  " <<RezIzmer<<endl;
return 0;
}
```

9.2.3 Запустите процесс компиляции программы, затем её выполнение.

9.2.4 Введите в программу команду для вывода на дисплей текста на русском языке. Замените в программе выводимый текст на русский.

9.2.5 Составьте блок-схему алгоритма программы «PCULab.9.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.9.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Какие действия выполняются по команде «if (massivNp[i] <minNp) minNp = massivNp[i];»?

2 Какие действия выполняются по команде «if (massivNp[i] >maxNp) maxNp = massivNp[i];»?

3 Какие действия выполняются по команде «cout<< "MinimalnoeznacheniеminNp = "<<minNp<<endl;»?

4 Какие действия выполняются по команде « cout<< "MaximalnoeznacheniemaxNp = "<<maxNp<<endl;»?

10 Лабораторная работа № 10. Разработка и исследование программного обеспечения для РСИ с округлением результата измерения

Цель работы: разработать и исследовать работу программного обеспечения для РСИ на микроконтроллере с использованием функции округления результата измерения.

10.1 Основные теоретические положения

Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 9.

Число точек на кривой функции преобразования РСИ и число наблюдений измеряемой величины вводится с клавиатуры при его градуировке; указаны диапазоны их значений и после ввода делается 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. Результат измерения округляется до заданного в программе числа разрядов после запятой.

10.2 Порядок выполнения работы

10.2.1 Запустите среду программирования CodeBlocks.

10.2.2 Создайте новый проект и присвойте ему имя «PCULab.10.1».

10.2.3 Разработайте программу на C++, с помощью которой надо:

– выполнить ввод числа точек N на кривой нелинейной функции преобразования для ее аппроксимации из диапазона 3…10 с проверкой правильности ввода;

– выполнить ввод числа результатов наблюдений n из диапазона 5…7 с проверкой правильности ввода;

– сформировать массив $N_O[\ ]$ показаний образцового прибора N_O = 3…10 путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– сформировать массив $N_{PG}[\ ]$ показаний рабочего средства измерения N_{PG} = 3…10 путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– сформировать массив $N_P[\ ]$ результатов наблюдений с помощью рабочего средства измерения n = 5…7 путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– найти сумму элементов массива $N_P[\ ]$ $sumN_P[\ ]$;

– в массиве $N_P[\ ]$ найти $minN_P$ и $maxN_P$ и отобразить их на экране;

– найти среднее арифметическое Nps элементов массива $N_P[\ ]$, отбросив минимальное и максимальное значения $minN_P$ и $maxN_P$, по формуле

$$Nps = (sumNp - minNp - maxNp) / (n - 2),$$

и отобразить Nps на экране;

– рассчитать результат измерения по формуле

$$RezIzmer = ((massivNo[i+1] - massivNo[i]) \cdot (Nps - massivNpg[i]))/(massivNpg[i+1] - massivNpg[i]) + massivNo[i]$$

и отобразить RezIzmer на экране;

– используя функцию округление, округлить результат измерения до заданного числа разрядов после запятой и отобразить скорректированный RezIzmer на экране.

10.2.4 Запустите процесс компиляции программы.

10.2.5 Запустите процесс выполнения программы.

10.2.6 С помощью клавиатуры сделайте ввод числа точек *N*, ввод числа результатов наблюдений *n*, ввод элементов массива $N_O[\ ]$, ввод элементов массива $N_{PG}[\ ]$, ввод элементов массива $N_P[\ ]$.

Результат выполнения написанной программы (как вариант) показан на рисунке 10.1.

10.2.7 Проделайте 2…3 раза действия, прописанные в п. 10.2.6, с разными числами, вводимыми с помощью клавиатуры.

10.2.8 Составьте блок-схему алгоритма программы «PCULab.10.1».

```
Uvedite chislo tochek na krivoy(N=2...10) N = 3
Uvedite chislo rezultatov nabludeniy (n=3...7) n = 7
Uvedite Noi: 10
Uvedite Noi: 45
Uvedite Noi: 110
10      45      110
Uvedite Npgi: 15
Uvedite Npgi: 58
Uvedite Npgi: 125
15      58      125
Chituvanie c ADC Npi: 43
Chituvanie c ADC Npi: 42
Chituvanie c ADC Npi: 47
Chituvanie c ADC Npi: 44
Chituvanie c ADC Npi: 41
Chituvanie c ADC Npi: 45
Chituvanie c ADC Npi: 49
43      42      47      44      41      45      49
Minimalnoe znachenie minNp = 41
Maximalnoe znachenie maxNp = 49
Crednee arifmet. Nps = 44.2

Rezultat Izmereniy  =  33.7674

RezultatIzmereniy c okrugleniem =  33.8
```

Рисунок 10.1 – Окно с результатами выполненной программы «PCULab.10.1»

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.10.1», блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Как задается количество наблюдений измеряемой величины в программе «PCULab.10.1» ?

2 Как задается количество точек на кривой функции преобразования в программе «PCULab.10.1»?

3 Каким количеством прямолинейных отрезков аппроксимируется кривая функции преобразования в программе «PCULab.10.1» ?

4 Как задается число разрядов после запятой в программе «PCULab.10.1» ?

11 Лабораторная работа № 11. Разработка и исследование программного обеспечения на C++ для обработки данных в РСИ с использованием прототипов функций

Цель работы: исследовать работу программного обеспечения для цифрового прибора на микроконтроллере с использованием прототипов функций обработки данных.

11.1 Основные теоретические положения

Алгоритм вычисления величины X при нелинейной зависимости $N_p = f(X)$ аналогичен рассмотренному в лабораторной работе № 10.

Нелинейная функция преобразования аппроксимируется *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 проверяется 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.

Результат измерения округляется до двух значащих цифр после запятой. В этой программе до функции «intmain()» описан прототип функция «F_Okruglenie», после функции «intmain()» описана сама функция «F_Okruglenie».

11.2 Порядок выполнения работы

11.2.1 Запустите среду программирования CodeBlocks.

11.2.2 Создайте новый проект и присвойте ему имя «PCULab.11.1».

11.2.3 Вставьте в созданный проект нижеприведенный код программы, запишите к ней комментарии.

```
/* PCULab.11.1 */
#include <iostream>
 #include <math.h>
 usingnamespacestd;
 //определяем и описываем функцию округления с именем "F_Okruglenie" и
 аргументом "chislo";
 floatF_Okruglenie(float chislo);
 intmain()
 {
 unsignedshort n;   // Записать комментарий;
 unsignedshort N;   // Записать комментарий;
 floatminNp,maxNp;  // Записать комментарий;
 cout<<endl; // пустая строка;
 do{
 cout<< "Vveditechislotocheknakrivoy(N=2...10) N = ";
 cin>> N;
 } while (N > 10 || N <2 );
 unsigned short arraySize = N;
 do{
 cout<< "Vveditechislorezultatovnabludeniy (n=3...7) n = ";
 cin>> n;
 } while (n <3||n >7 );
 floatmassivNo[arraySize];
 for(inti = 0; i<arraySize; i++)
    {
 cout<< "VvediteNoi: ";
 cin>>massivNo[i];
     }
 for(inti = 0; i<arraySize; i++)
```

```
            {
cout<<massivNo[i] << "\t";
            }
cout<<endl; // пустая строка;
floatmassivNpg[arraySize];
for(inti = 0; i<arraySize; i++)
   {
cout<< "VveditеNpgi: ";
cin>>massivNpg[i];
   }
cout<<endl; // пустая строка;
for(inti = 0; i<arraySize; i++)
            {
cout<<massivNpg[i] << "\t";
            }
cout<<endl; // пустая строка;
// Записать комментарий;
floatsumNp;
floatRezIzmer;
floatmassivNp[n];
maxNp=0;
for(inti = 0; i< n; i++)
   {
cout<< "Chituvanie c ADC Npi: ";
cin>>massivNp[i];
   }
cout<<endl; // пустая строка;
for(inti = 0; i< n; i++)
         {
cout<<massivNp[i] << "\t";
sumNp += massivNp[i];
if (massivNp[i] <minNp) minNp = massivNp[i]; // Записать комментарий;
if (massivNp[i] >maxNp) maxNp = massivNp[i]; // Записать комментарий;
}
cout<<endl; // пустая строка;
cout<< "MinimalnoeznacheniеminNp = "<<minNp<<endl;
cout<< "MaximalnoeznacheniemaxNp = "<<maxNp<<endl;
cout<<endl; // пустая строка;
floatNps = (sumNp – minNp – maxNp) / (n–2); // Записать комментарий;
cout<< "Credneearifmet. Nps = " <<Nps;
// Записать комментарий;
for(inti = 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];
}
}
  // Записать комментарий;
cout<< "RezultatIzmereniy  =  " <<F_Okruglenie(RezIzmer) <<endl;
return 0;
```

```
}
floatF_Okruglenie(float RezIzmer)
{
returnfloor(RezIzmer * 10 + .5) / 10;
}
```

11.2.3 Запустите процесс компиляции программы, затем её выполнение.

11.2.4 Введите в программу команду для вывода на дисплей текста на русском языке. Замените в программе выводимый текст на русский.

11.2.5 Составьте блок-схему алгоритма программы «PCULab.11.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.11.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Что означает запись «floatF_Okruglenie(floatchislo);» в начале программы?

2 Как в программе изменить число значащих цифр после запятой в результате измерения?

3 Какие действия выполняются по команде «floatF_Okruglenie(floatchislo)
{
returnfloor(chislo * 10 + .5) / 10;
}»?

12 Лабораторная работа № 12. Разработка и исследование программного обеспечения для РСИ с двумя поддиапазонами измерений

Цель работы: исследовать работу программного обеспечения для цифрового прибора на микроконтроллере с двумя поддиапазонами измерений и автоматической корректировкой функции округления результата измерения.

12.1 Основные теоретические положения

Автоматически устанавливается количество разрядов после запятой в полученном результате в зависимости от его попадания в программно-заданные поддиапазоны измерения. Первый поддиапазон – от 0 до 10, второй – от 10 до 100. В программе используются две функции: округления и статистической обработки. Они имеют следующие прототипы:

«floatF_Okruglenie(floatchislo,unsignedshortk);»;
«floatF_StatObrabotka(floatsumNp,floatminNp,floatmaxNp,int n);».

12.2 Порядок выполнения работы

12.2.1 Запустите среду программирования CodeBlocks.
12.2.2 Создайте новый проект и присвойте ему имя «PCULab.12.1».
12.2.3 Вставьте в созданный проект нижеприведенный код программы.

```
/* PCULab.12.1 */
  #include <iostream>
  #include <math.h>
  usingnamespacestd;
/*прототип функции округления с именем "F_Okruglenie" и аргументом "chislo"; это стандартная функция, вызывается из библиотеки <math.h> */;
  floatF_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 */;
  floatF_StatObrabotka(float sumNp,floatminNp,floatmaxNp,int n);
  intmain()
  {
  unsignedshort n;   // Число наблюдений измеряемой величины;
  unsignedshort N; /* Число точек на кривой функции преобразования, вводимых с клавиатуры при градуировке прибора*/;
  floatminNp,maxNp;      /* минимальное и максимальное значения результатов наблюдений измеряемой величины*/;
  unsigned short k;
     cout << endl; // пустая строка;
  do{
  cout<< "Vveditechislotocheknakrivoy(N=2...10) N = ";
  cin>> N;
  } while (N > 10 || N <2 );
  unsigned short arraySize = N;
  cout<<endl; // пустая строка;
  do{
  cout<< "Vveditechislorezultatovnabludeniy (n=3...7) n = ";
  cin>> n;
  } while (n <3||n >7 );
  cout<<endl; // пустая строка;
  floatmassivNo[arraySize];
  for(inti = 0; i<arraySize; i++)
     {
  cout<< "VvediteNoi: ";
  cin>>massivNo[i];
      }
  cout<<endl; // пустая строка;
  for(inti = 0; i<arraySize; i++)
              {
  cout<<massivNo[i] << "\t";
              }
  cout<<endl; // пустая строка;
  floatmassivNpg[arraySize];
  for(inti = 0; i<arraySize; i++)
```

```
    {
cout<< "VvediteNpgi: ";
cin>>massivNpg[i];
    }
cout<<endl; // пустая строка;
for(inti = 0; i<arraySize; i++)
        {
cout<<massivNpg[i] << "\t";
        }
cout<<endl; // пустая строка;
/* Считывание текущего значения Np с АЦП в режиме измерения (здесь – с клавиатуры) */;
floatsumNp;
floatRezIzmer;
floatmassivNp[n];
maxNp=0;
for(inti = 0; i< n; i++)
    {
cout<< "Chituvanie c ADC Npi: ";
cin>>massivNp[i];
    }
cout<<endl; // пустая строка;
for(inti = 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;
floatNps = F_StatObrabotka(sumNp,minNp,maxNp,n);
/* Проверка попадания Np в диапазон (Npi...Np(i+1)) и, при выполнении условия,
вычисление результата измерения */;
for(inti = 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<< "RezultatIzmereniy  =  " <<F_Okruglenie(RezIzmer, k) <<endl;
```


```
// вызов функции округления результата измерения;
        return 0;
        }
//Описание функции "F_Okruglenie";
        floatF_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";
        floatF_StatObrabotka(float sumNp,floatminNp,floatmaxNp,int n)
          {
        floatNps;
        cout<< "MinimalnoeznachenieminNp = "<<minNp<<endl; /*вывод минимального значения */;
cout<< "MaximalnoeznacheniemaxNp = "<<maxNp<<endl; /*вывод максимального значения */;
        Nps=(sumNp – minNp – maxNp) / (n–2); /* вычисление среднего значения */;
        cout<<endl; // пустая строка;
        cout<< "Credneearifmet. Nps = " <<Nps;
        returnNps;
          }
```

12.2.3 Запустите процесс компиляции программы, затем её выполнение.

12.2.4 Введите в программу команду для вывода на дисплей текста на русском языке. Замените в программе выводимый текст на русский.

12.2.5 Выполните программу дважды. Вначале сделайте ввод таких результатов наблюдений, чтобы после их обработки результат измерения оказался в поддиапазоне (0 <=RezIzmer<= 10). Затем сделайте ввод таких результатов наблюдений, чтобы после их обработки результат измерения оказался в поддиапазоне (10 <RezIzmer<= 100).

12.2.6 Составьте блок-схему алгоритма программы «PCULab.12.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.12.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Что означает запись «floatF_Okruglenie(float chislo,unsigned short k)»; в начале программы?

2 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего в первый поддиапазон?

3 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего во второй поддиапазон?

4 Как удалить из программы вывод на дисплей среднего арифметического результатов наблюдений?

13 Лабораторная работа № 13. Разработка и исследование программного обеспечения для многодиапазонного РСИ с автоматическим выбором диапазона измерения

Цель работы: исследовать работу программного обеспечения для цифрового прибора на микроконтроллере с тремя поддиапазонами измерений и автоматической корректировкой функции округления результата измерения для каждого поддиапазона.

13.1 Основные теоретические положения

Автоматически устанавливается количество разрядов после запятой в полученном результате в зависимости от его попадания в программно - заданные поддиапазоны измерения.

В программе используются две функции: округления и статистической обработки. Они имеют следующие прототипы:

«floatF_Okruglenie(floatchislo,unsignedshortk);»;
«floatF_StatObrabotka(floatsumNp,floatminNp,floatmaxNp,int n);».

13.2 Порядок выполнения работы

13.2.1 Запустите среду программирования CodeBlocks.

13.2.2 Создайте новый проект и присвойте ему имя «PCULab.13.1».

13.2.3 Разработайте программу на C++, с помощью которой можно:

– выполнить ввод числа точек N на кривой нелинейной функции преобразования для ее аппроксимации из диапазона 3…10 с проверкой правильности ввода;

– выполнить ввод числа результатов наблюдений n из диапазона 5…7 с проверкой правильности ввода;

– сформировать массив N_O[ ] показаний образцового прибора N_O = 3…10 путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– сформировать массив N_{PG}[ ] показаний рабочего средства измерения N_{PG} = 3…10 путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– сформировать массив N_P[ ] результатов наблюдений с помощью рабочего средства измерения n = 5…7 путем ввода элементов массива с помощью клавиатуры и отображения вводимых данных на экране;

– найти сумму элементов массива N_P[ ] sumN_P[ ];

– в массиве N_P[ ] найти minN_P и maxN_P и отобразить их на экране;

– найти среднее арифметическое Nps элементов массива N_P[ ], отбросив минимальное и максимальное значения $minN_P$ и $maxN_P$, по формуле

Nps = (sumNp - minNp - maxNp) / (n – 2),

и отобразить Nps на экране;

– рассчитать результат измерения по формуле

RezIzmer = ((massivNo[i+1] - massivNo[i]) · (Nps - massivNpg[i])) / (massivNpg[i+1] - massivNpg[i]) + massivNo[i]

и отобразить RezIzmer на экране;

– весь диапазон измерения разбить на три поддиапазона, например, 0…3, 3…30, 30…100;

– дополните программу функцией

setlocale(LC_CTYPE,"Russian"); // вывод русского текста.

В поддиапазоне 0…3 надо предусмотреть округление результата измерения до двух цифр после запятой, в поддиапазоне 3…30 − округление результата измерения до одной цифры после запятой, в поддиапазоне 30…100 – округление результата измерения до целого значения.

13.2.5. Запустите процесс компиляции программы.

13.2.6 Запустите процесс выполнения программы.

13.2.7 Выполните программу трижды с попаданием результата измерения поочередно во все поддиапазоны.

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.13.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего в первый поддиапазон?

2 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего во второй поддиапазон?

3 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего в третий поддиапазон?

14 Лабораторная работа № 14. Разработка и исследование программного обеспечения для градуировки РСИ в диалоговом режиме

Цель работы: разработать и исследовать работу программного обеспечения для РСИ на микроконтроллере с задаваемыми в диалоговом режиме количеством поддиапазонов измерений, их значениями, параметрами функции округления результата измерения для каждого поддиапазона.

14.1 Основные теоретические положения

При разработке этой программы за основу следует взять программу «PCULab.13.1» из лабораторной работы № 13.

В диалоговом окне следует предусмотреть вывод сообщений:

− «Введите количество поддиапазонов Npp =»;

− «Введите конечные значения каждого поддиапазона Dki =»;

− «Введите коэффициенты, указывающие на количество выводимых цифр до и после запятой, для каждого поддиапазона ki =».

Ввод данных должен осуществляться с помощью клавиатуры.

14.2 Порядок выполнения работы

14.2.1 Доработайте программу «PCULab.13.1» с учетом требований, изложенных в п. 14.1.

14.2.2 Запустите среду программирования CodeBlocks.

14.2.3 Создайте новый проект и присвойте ему имя «PCULab.14.1».

14.2.4 Вставьте в созданный проект разработанный код программы.

14.2.5 Запустите процесс компиляции программы.

14.2.6 Если ошибок нет, запустите процесс выполнения программы.

14.2.7 Результат выполнения программы протестируйте для каждого поддиапазона и продемонстрируйте преподавателю.

14.2.8 Составьте блок-схему алгоритма программы «PCULab.14.1».

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, код программы «PCULab.14.1», результаты её выполнения, блок-схему алгоритма этой программы, выводы по результатам её выполнения.

Контрольные вопросы

1 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего в первый поддиапазон?

2 Как в программе изменить число значащих цифр после запятой в результате измерения, попадающего во второй поддиапазон?

3 Какие дополнительные возможности появляются у цифрового прибора при использовании в нем разработанной программы в сравнении с программой «PCULab.13.1»?

15 Лабораторная работа № 15. Разработка и исследование программного обеспечения для управления сканирующими устройствами приборов неразрушающего контроля

Цель работы: разработать и исследовать работу программного обеспечения для управления сканирующим устройством прибора неразрушающего контроля.

15.1 Основные теоретические положения

Для создания одно- или многокоординатных систем сканирования в приборах неразрушающего контроля используются специальные электроприводы. Если от привода требуется повышенная точность позиционирования вала и высокий момент удержания при нулевой скорости, и не требуется повышенная максимальная скорость вращения**,** то целесообразно использовать шаговый привод вращения. Параметры блока управления определяются числом обмоток, потребляемым током и порядком коммутации обмоток шагового двигателя (ШД). Технический уровень блока управления определяется, в основном, примененной в них элементной базой.

Частота вращения и суммарный угол поворота вала ШД пропорциональны соответственно частоте и числу поданных импульсов управления. При отсутствии управляющих импульсов ШД находится в режиме фиксированной стоянки и сохраняет конечные результаты предыдущих перемещений. Привод с ШД сочетает возможности глубокого регулирования частоты вращения с возможностью числового задания и надежной фиксации конечных координат.

Помимо самого ШД необходимо также специальное электронное устройство для управления его работой, которое обеспечивает преобразование унитарной последовательности импульсов в m-фазную систему напряжений, питающих обмотки ШД через усилители мощности. Такие устройства управления разрабатываются на основе микроконтроллеров. Подключение ШД к микроконтроллерной системе делают следующим образом (рисунок 15.1).

В качестве электроприводов в системах сканирования в приборах неразрушающего контроля используются также двигатели постоянного тока (ДПТ). Для управления их работой вместе с микроконтроллером используются электромагнитные реле. На рисунке 15.2 показан внешний вид реле SRD-05VDC.

Данное реле управляется напряжением 5 В и способно коммутировать до 10 А 30 В DC и 10 А 250 В AC.

Реле имеет две раздельных цепи: цепь управления, представленная

контактами A1, A2; и управляемая цепь, представленная контактами 1, 2, 3. Цепи никак не связаны между собой (рисунок 15.3).

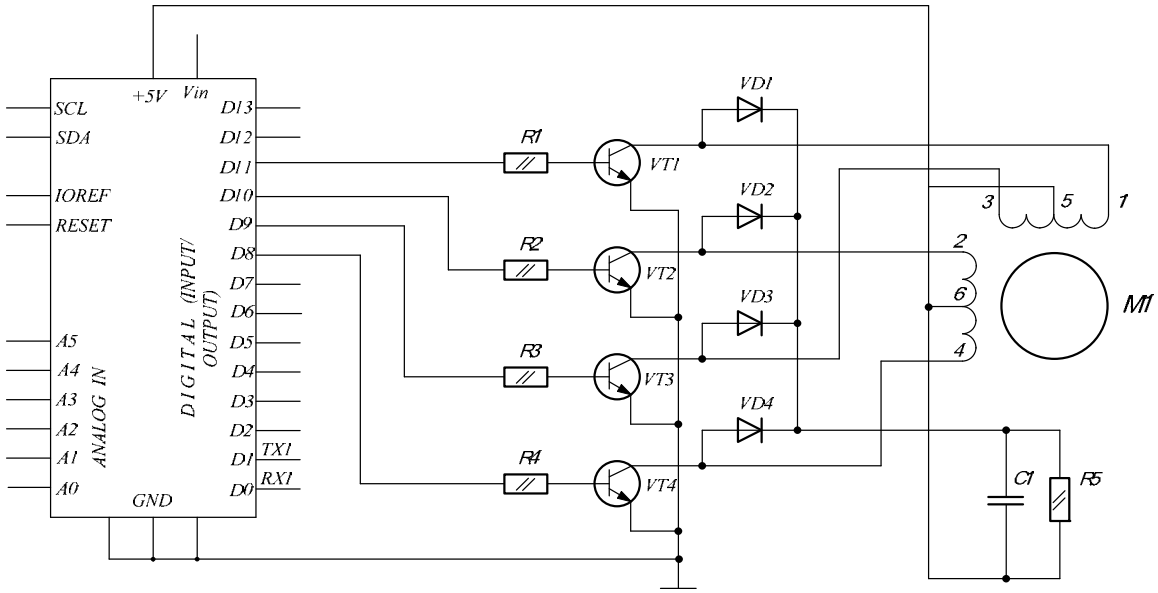

Рисунок 15.1 – Схема подключения к микроконтроллеру шагового двигателя

Рисунок 15.2 – Внешний вид реле SRD-05VDC

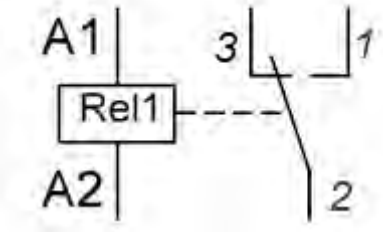

Рисунок 15.3 – Коммутационная схема реле SRD-05VDC

Релейный модуль имеет три вывода:

1) VCC: "+" питания;
2) GND: "–" питания;
3) IN: вывод входного сигнала управления.

Схема подключения реле к контроллеру Arduino приведена на рисунке 15.4.

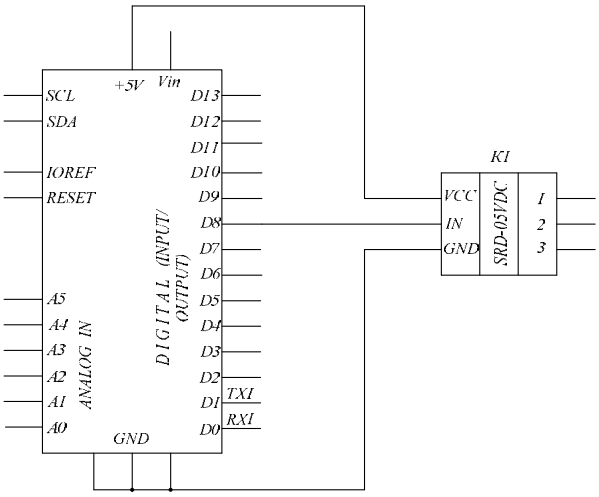

Рисунок 15.4 – Схема подключения реле к контроллеру Arduino

Приведем код программы «Rele1» включения и выключения реле с интервалом в 2 с.

```
// Реле модуль подключен к цифровому выводу D8
  int Relay = 8;
  void setup()
  {
  pinMode(Relay, OUTPUT);
  }
  void loop()
  {
  digitalWrite(Relay, LOW);   // реле включено
  delay(2000);
  digitalWrite(Relay, HIGH);  // реле выключено
  delay(2000);
  }
```

Напрямую к цифровым контактам контроллера Arduino можно подключать только устройства, потребляющие небольшую мощность, например, светодиод или динамик небольшой мощности, т. к. допустимый ток через цифровой вывод Arduino не должен превышать 40 мА. Поэтому при подключении электродвигателя непосредственно к контакту Arduino контроллер может выйти из строя. Чтобы этого не случилось, напряжение на электродвигатель надо подавать от внешнего источника через реле.

Схема подключения электродвигателя к контроллеру Arduino через реле приведена на рисунке 15.5.

Код программы, обеспечивающий периодическое включение электродвигателя на 8 с с интервалом в 4 с, представлен ниже.

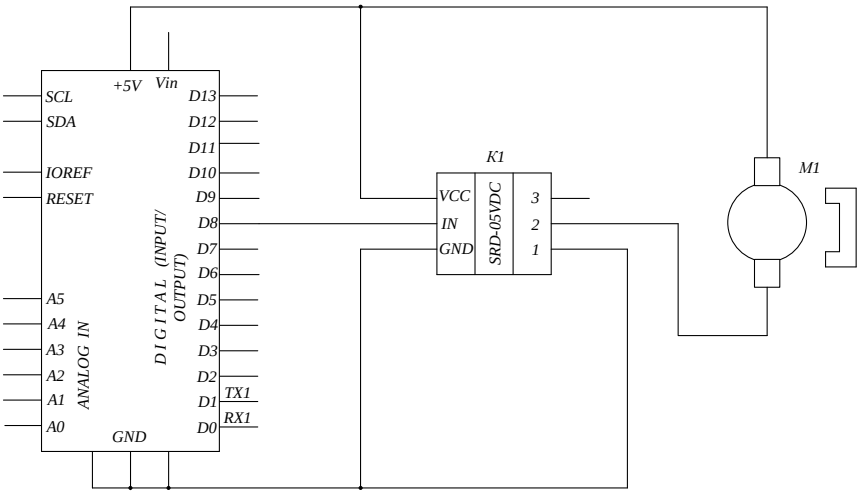

Рисунок 15.5 – Схема подключения электродвигателя к контроллеру Arduino через реле

Код программы «Motor1».

```
int Relay1 = 8;
void setup()
{
pinMode(Relay1, OUTPUT);
}
void loop()
{
digitalWrite(Relay1, HIGH);  // реле включено (двигатель работает)
delay(8000);
digitalWrite(Relay1, LOW);   // реле выключено (двигатель не работает)
delay(4000);
}
```

Также можно реализовать работу электродвигателя в реверсивном режиме (обеспечить вращение в обе стороны). Для этого нужно использовать два реле.

Код программы, обеспечивающий реверсивный режим работы электродвигателя, представлен ниже.

Код программы «Motor2».

```
int Relay3 = 8;
int Relay4 = 9;
void setup()
{
pinMode(Relay3, OUTPUT);
pinMode(Relay2, OUTPUT);
}
void loop()
{
digitalWrite(Relay3, LOW);   // реле включено
delay(1000);
digitalWrite(Relay3, HIGH);  // реле выключено
delay(1000);
```

```
digitalWrite(Relay4, LOW);   // реле включено
delay(1000);
digitalWrite(Relay4, HIGH);  // реле выключено
delay(1000);
}
```

Порядок выполнения работы

15.2.1 Подключить реле SRD-05VDC к контроллеру Arduino, контроллер Arduino подключить к ПК.

15.2.2 Загрузить в контроллер Arduino программу «Rele1», исследовать её выполнение.

15.2.3 Составить и загрузить в контроллер Arduino программу «Motor1», обеспечивающую работу электродвигателя в обычном режиме, и исследовать её выполнение.

15.2.4 Загрузить в контроллер Arduino программу «Motor2», обеспечивающую работу электродвигателя в реверсивном режиме, и исследовать её выполнение.

15.2.5 Нарисовать схему подключения электродвигателя к контроллеру Arduino, обеспечивающую его работу в реверсивном режиме.

Содержание отчета

Отчет о проделанной работе должен содержать: название работы, цель работы, схемы подключения к Arduino реле, электродвигателя в обычном и реверсивном режимах, программы и результаты экспериментальных исследований, выводы.

Контрольные вопросы

1 Опишите устройство и принцип работы реле.

2 Опишите принцип управления работой электродвигателя в обычном режиме.

3 Опишите принцип управления работой электродвигателя в реверсивном режиме.

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 **Миленина, С. А.** Электротехника, электроника и схемотехника: учебник и практикум для академ. бакалавриата / С. А. Миленина; под ред. Н. К. Миленина. – Москва : Юрайт, 2015. – 399 с.

4 **Бладыко, Ю. В.** Электроника. Практикум: учебное пособие / Ю. В. Бладыко. – Минск: ИВЦ Минфина, 2016. – 190 с.: ил.

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