Федеральное агентство по рыболовству
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Астраханский государственный технический университет»

Институт информационных технологий и коммуникаций
Направление подготовки 09.03.04 Программная инженерия
Профиль «Разработка программно-информационных систем»
Кафедра «Автоматизированные системы обработки информации и управления»

КУРСОВОЙ ПРОЕКТ

Учебно-демонстрационная программа «Транзистор»

по дисциплине «Компьютерное моделирование»

Допущен к защите
«__» _____20__г.
Руководитель
_____

Оценка, полученная на защите
«_____»

Проект выполнен
обучающимся группы ДИПРб-21
Катуниным С.С.
_____

Руководитель
ст. преп. каф. АСОИУ Толасова В.В

Члены комиссии:

_____

_____

_____

Астрахань – 2020

ФЕДЕРАЛЬНОЕ АГЕНТСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ ПО РЫБОЛОВСТВУ

АСТРАХАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УТВЕРЖДАЮ
Заведующий кафедрой
д.т.н., доцент
Т.В. Хоменко_____
«____»_____202 __ г.

Кафедра
«Автоматизированные системы
обработки информации и управления»

ЗАДАНИЕ

на выполнение курсового проекта

Обучающийся ***Катунин Сергей Сергеевич***

Группа ***ДИПРб-21***

Дисциплина ***Компьютерное моделирование***

Тема ***Учебно-демонстрационная программа по теме «Транзистор»***

Дата получения задания «_____»_____202___г.

Срок представления обучающимся КП на кафедру «_____»_____202___г.

Руководитель ***ст. преп. каф. АСОИУ***_____ ***Толасова В.В.*** «_____»_____202___г.

должность, степень, звание подпись ФИО

Обучающийся _____ ***Катунин С.С.*** «_____»_____202___г.

подпись ФИО

Задачи

Разработка программного продукта, который

- предоставляет пользователю теоретический материал по теме «Транзистор»;
- демонстрирует процесс работы транзистора в режиме ключа и принцип работы транзистора в вычислительной технике на примере простого сумматора;
- предоставляет пользователю возможность тестирования по теме «Транзистор»;

Список рекомендуемой литературы

1. Айсберг Е. Транзистор?.. Это очень просто! Перевод с французского Ю. Л. Смирнова. М. – Л., издательство «ЭНЕРГИЯ», 1963. 112 стр. с ил. (Массовая радиобиблиотека. Вып. 480).

2. Р. Хаггарти. Дискретная математика для программистов. Москва: Техносфера, 2003. – 320 с.

3. Шлее М. Qt 5.10. Профессиональное программирование на C++. – СПб.: БХВ-Петербург, 2018. – 1072 с.: ил. – (В подлиннике)

УТВЕРЖДАЮ

Заведующий кафедрой

к.т.н., доцент

Т.В. Хоменко_____

«____»_____20 ___г.

К заданию на курсовой проект
по дисциплине
«Компьютерное моделирование»

КАЛЕНДАРНЫЙ ГРАФИК
курсового проектирования

№ п/п	Разделы, темы и их содержание, графический материал	Дата сдачи	Объем, %
1	Выбор темы	15.03.2020	1
2	Техническое задание	29.03.2020	3
3	Разработка модели, проектирование системы ▪ *введение,* ▪ *технический проект,* ▪ *программа и методика испытаний,* ▪ *литература*	29.04.2020	25
4	Программная реализация системы ▪ *работающая программа,* ▪ *рабочий проект* ▪ *скорректированное техническое задание (при необходимости)*	29.05.2020	40
5	Тестирование и отладка системы, эксперименты ▪ *работающая программа с внесёнными изменениями,* ▪ *окончательные тексты всех разделов*	05.06.2020	50
6	Компоновка текста Подготовка презентации и доклада ▪ *пояснительная записка* ▪ *презентация* ▪ *электронный носитель с текстом пояснительной записки, исходным кодом проекта, презентацией и готовым программным продуктом*	12.06.2020	59
7	Защита курсового проекта	17.06.2020 - 22.06.2020	60-100

С графиком ознакомлен «____» _____ 20 ___г.

Катунин С.С. _____, обучающийся группы ДИПРб-21
(фамилия, инициалы, подпись)

График курсового проектирования выполнен
без отклонений / с незначительными отклонениями / со значительными отклонениями

нужное подчеркнуть

Руководитель курсового проекта_____ ст. преп. каф. АСОИУ Толасова В.В.
подпись, ученая степень, звание, фамилия, инициалы

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Физика – это сложная наука, которая может занять множество часов у студента перед тем, как он усвоит тот или иной материал. Для более быстрого и глубокого понимания темы необходимо, помимо теоретических знаний, иметь наглядное представление о том, как протекают физические процессы. По этой причине в школьной программе присутствуют различные лабораторные опыты и эксперименты, которые должны помочь ученикам лучше понять изучаемый материал. Но некоторые опыты можно нагляднее и эффективнее смоделировать на компьютере, например, продемонстрировав движение электронов с помощью графической анимации, а некоторые опыты и вовсе невозможно провести в учебных организациях, в виду отсутствия необходимого лабораторного оборудования или специфики изучаемого материала.

Появление транзистора произвело революцию как в радиотехнике, так и в вычислительной технике. У студентов на прикладных специальностях могут возникать многочисленные трудности с усвоением данного материала: общий принцип работы транзистора и его назначение, физическая природа транзистора, а именно, полупроводники типов p и n, дырки, явление p-n перехода, и так далее. Помимо этих сложностей, студенту может быть тяжело усвоить роль транзистора в вычислительной технике, то есть как работает транзистор в режиме ключа, принцип устройства простейших сумматоров на основе транзисторов.

Целью создания учебно-демонстрационной программы «Транзистор» является автоматизация обучения и контроля знаний по теме «Транзистор».

Назначение программы – повышение качества знаний студентов, снижение нагрузки на преподавателя.

1 ТЕХНИЧЕСКИЙ ПРОЕКТ

1.1 Анализ предметной области

1.1.1 Транзистор в радиотехнике

Транзистор – радиоэлектронный компонент из полупроводникового материала, обычно с тремя выводами способный от небольшого входного сигнала управлять значительным током в выходной цепи, что позволяет использовать его для усиления, генерирования, коммутации и преобразования электрических сигналов. Благодаря своей способности работать при низких напряжениях и значительных токах в режиме электрического ключа, транзисторы позволили уменьшить потребность в электромагнитных реле и механических переключателях в оборудовании, а благодаря способности к миниатюризации и интеграции позволили создать интегральные схемы, заложив основы микроэлектроники. Транзисторы по структуре, принципу действия и параметрам делятся на два класса — биполярные и полевые (униполярные). Для простоты в данной курсовой работе будут рассматриваться только **биполярные** транзисторы.

Биполярный транзистор – трёхэлектродный полупроводниковый прибор, один из типов транзисторов. В полупроводниковой структуре сформированы два p-n перехода, перенос заряда через которые осуществляется носителями двух полярностей — электронами и дырками. Биполярный транзистор состоит из трёх полупроводниковых слоёв с чередующимся типом примесной проводимости: эмиттера, базы и коллектора. В зависимости от порядка чередования слоёв различают **n-p-n** (эмиттер — n-полупроводник, база — p-полупроводник, коллектор — n-полупроводник) и **p-n-p** транзисторы. Для простоты далее будут рассматриваться только **n-p-n** транзисторы.

Полупроводник – материал, по удельной проводимости занимающий промежуточное место между проводниками и диэлектриками, и отличающийся от проводников сильной зависимостью удельной проводимости от концентрации примесей, температуры и воздействия различных видов излучения. Как правило полупроводник четырехвалентен, то есть на внешней (валентной) оболочке атома имеет 4 электрона. По виду проводимости полупроводники делятся на электронные (**n-типа**) и дырочные (**p-типа**). Для изготовления полупроводника того, или иного типа, в кристаллическую структуру вносят примеси.

Примесь в полупроводнике – атом другого химического элемента в чистой кристаллической решётке (например, атом фосфора, бора и т. д. в кристалле кремния). В зависимости от того, отдаёт ли примесной атом электрон в кристалл полупроводника (в вышеприведённом примере – фосфор) или захватывает его (бор), примесные атомы называют донорными или акцепторными.

Дырка – во время разрыва связи между электроном и ядром появляется свободное место в электронной оболочке атома, соответственно, дыркой является незаполненная валентная связь в полупроводнике, которая проявляет себя как положительный заряд, численно равный заряду электрона. То есть это отсутствие электрона в почти полностью заполненной валентной зоне. Для создания заметной концентрации дырок в полупроводниках используется легирование полупроводника акцепторными примесями. Обычно подвижность дырок в полупроводнике ниже подвижности электронов.

Полупроводник n-типа – полупроводник, в котором основные носители заряда — электроны. Для того чтобы получить полупроводник n-типа, полупроводник легируют донорами. Обычно это атомы, которые имеют на валентной оболочке на один электрон больше, чем у атомов полупроводника, который легируется. На рисунке 1.1 видно проводимость в полупроводнике n-типа при подключении к нему источника тока. Свободные электроны (помечены знаком минус) отрываются от пятивалентных атомов, которые из-за этого становятся положительными ионами (помечены знаком плюс). Поскольку в данном случае перенос заряда осуществляется электроном, а не дыркой, то данный вид полупроводников проводит электрический ток подобно металлам.

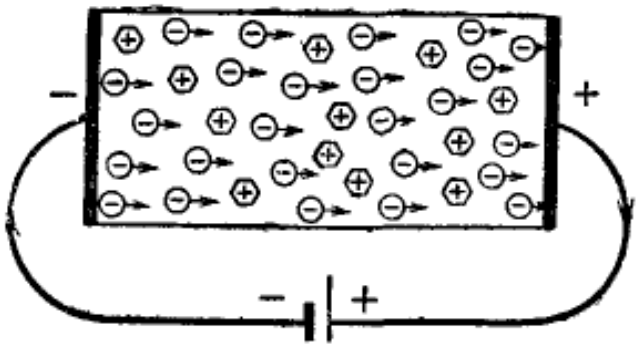

Рисунок 1.1 – Полупроводник n-типа

Полупроводник p-типа – полупроводник, в котором основными носителями заряда являются дырки. Полупроводники p-типа получают методом легирования собственных полупроводников акцепторами. В полупроводнике p-типа трехвалентный примесный атом захватывает электрон соседнего атома полупроводника, оставляя там дырку, которая в свою очередь заполняется электроном, оторвавшимся от соседнего атома, и т. д. Таким образом дырка, представляющая собой положительный заряд, перемещается от положительного полюса подключенного источника тока к отрицательному.

На рисунке 1.2 показаны последовательные фазы дырочной проводимости. В последней фазе электрон, поступивший из источника тока, заполняет ближайшую к отрицательному полюсу дырку, одновременно другой электрон покидает ближайший к положительному полюсу атом, на его месте возникает новая «дырка», и всё начинается сначала.

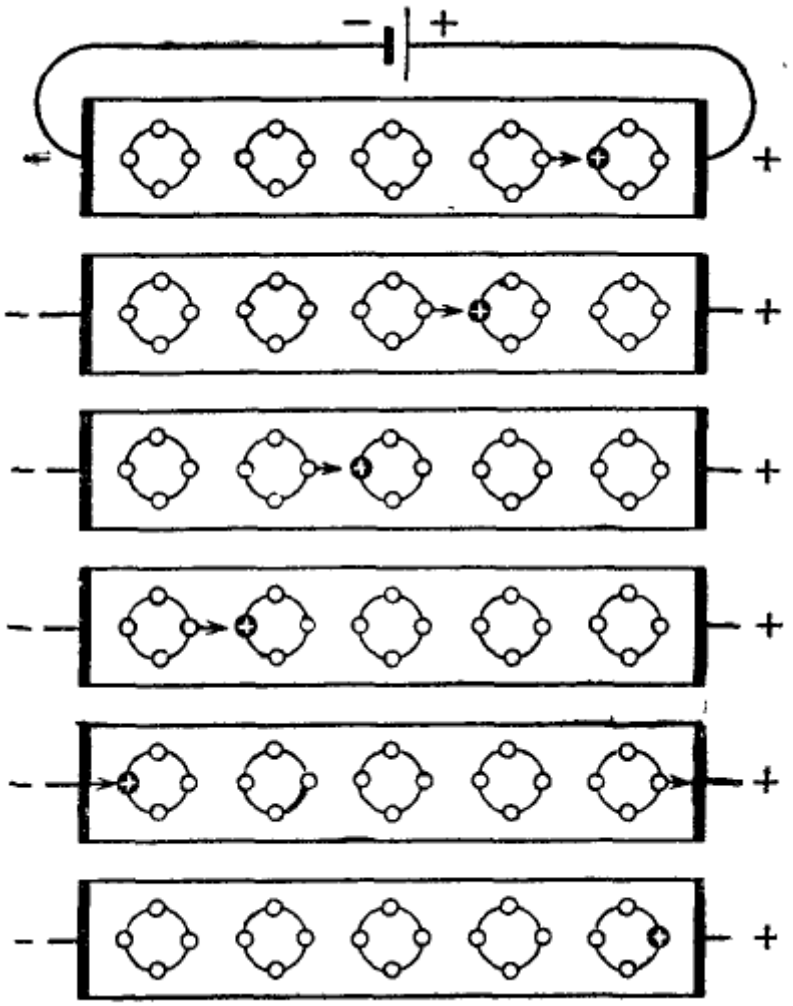

Рисунок 1.2 – Полупроводник p-типа

P-n переход – область соприкосновения двух полупроводников с разными типами проводимости — дырочной (p, от англ. positive — положительная) и электронной (n, от англ. negative — отрицательная). В полупроводнике p-типа, который получается посредством акцепторной примеси, концентрация дырок намного превышает концентрацию электронов. В полупроводнике n-типа, который получается посредством донорной примеси, концентрация электронов намного превышает концентрацию дырок. Если между двумя такими полупроводниками установить контакт, то возникнет диффузионный ток — основные носители заряда (электроны и дырки) хаотично перетекают из той области, где их больше, в ту область, где их меньше, и рекомбинируют друг с другом. Как следствие, вблизи границы между областями практически не будет свободных (подвижных) основных носителей заряда, но

останутся ионы примесей с некомпенсированными зарядами. Область в полупроводнике p-типа, которая примыкает к границе, получает при этом отрицательный заряд, приносимый электронами, а пограничная область в полупроводнике n-типа получает положительный заряд, приносимый дырками (точнее, теряет уносимый электронами отрицательный заряд).

На рисунке 1.3 изображен p-n переход. Дырки области p (изображены как черный круг со знаком плюс) отталкиваются от перехода, оставляя возле него отрицательные ионы акцепторной примести (треугольник со знаком минус). Точно так же свобоодные электроны области n (круг со знаком минус) отталкиваются от перехода, оставляя возле него положительные ионы донорной примеси (шестиугольник со знаком плюс).

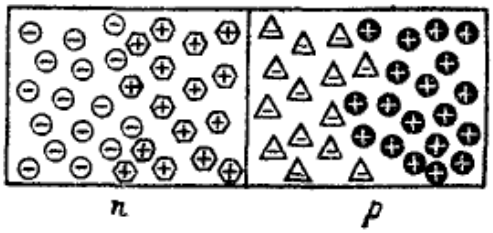

Рисунок 1.3 – Переход p-n.

Таким образом в полупроводниках типа p и типа n было вызвано перемещение подвижных зарядов в оба конца каждой области, тогда как в отсутствие p-n перехода заряды распределяются равномерно по всему кристаллу полупроводника.

Если положительный полюс источника тока соединен с областью p, а отрицательный полюс – с областью n (прикладываемое напряжение в **прямом** направлении), то положительный полюс источника будет притягивать к себе электрон каждый раз, когда другой электрон преодолеет переход, перепрыгнув из области n в область p. То есть возникнет ток, образуемый электронами и дырками, перемещающимися в противоположных направлениях. На рисунке 1.4 обозначены только носители зарядов: электроны (помечены знаком минус) и дырки (помечены знаком плюс).

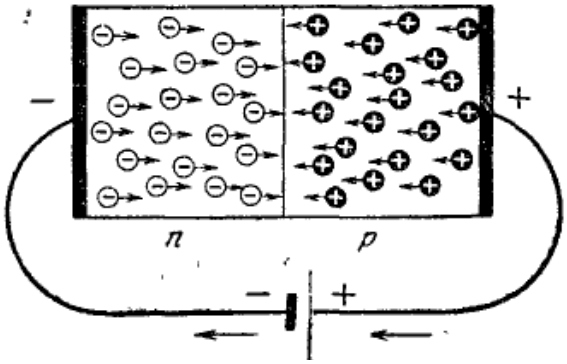

Рисунок 1.4 – Прохождение тока через p-n переход.

Однако, если приложить положительный полюс источника к области n, а отрицательный полюс – к области p (прикладываемое напряжение в **обратном** направлении), то результат будет иным. На рисунке 1.5 видно, как электроны отрицательного полюса источника притянули дырки области p к концу кристалла полупроводника. А к другому концу кристалла положительный потенциал источника притянул свободные электроны. При этом ни электроны, ни дырки не пересекают переход, а потенциальный барьер только увеличился, значит, никакого тока нет. Именно по этой причине p-n переход называют полупроводниковым диодом.

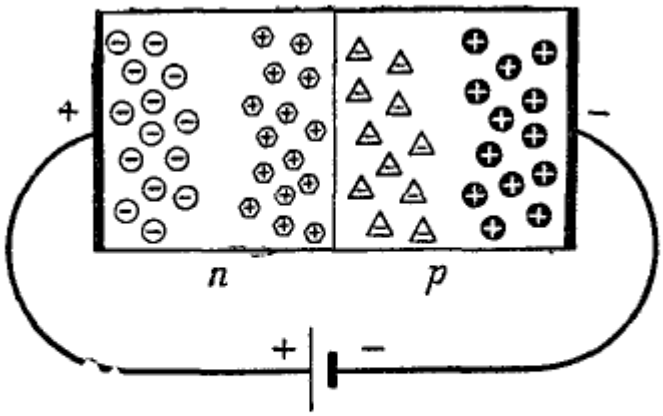

Рисунок 1.5 – Переход p-n с приложенным обратным напряжением.

Транзистор структуры n-p-n состоит из двух противоположно направленных p-n переходов. На рисунке 1.6 изображен **n-p-n** транзистор, у которого между эмиттером и коллектором приложено напряжение. Левая область называется эмиттером, серединная — базой, а правая – коллектором. При любой полярности один из переходов окажется в прямом, а другой в обратном направлении и будет препятствовать прохождению тока.

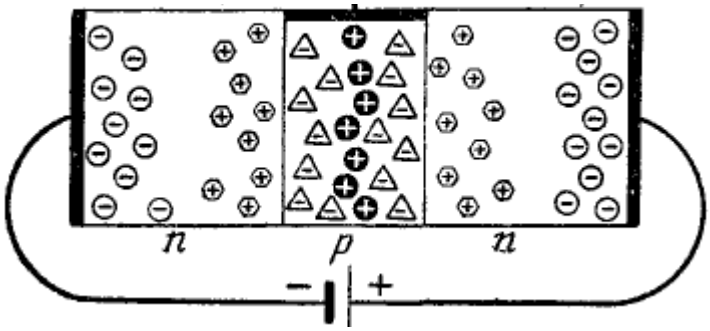

Рисунок 1.6 – Транистор n-p-n с приложенным напряжением между эмиттером и коллектором

Если приложить в прямом направлении небольшое напряжение между эмиттером и базой, то увидим, что ток внесет в базу (область p) свободные электроны из эмиттера, который состоит из полупроводника типа n. А так как база тонкая, то лишь небольшого количества этих электронов хватит для заполнения дырок, находящихся в области p. Через вывод базы будет выходить небольшой ток базы $I_б$. Но большинство проникших в базу электронов продолжит

свое движение и проникнет в коллектор, откуда они будут извлечены куда более высоким потенциалом источника напряжения $E_{к-э}$. Следовательно, они преодолеют потенциальный барьер второго перехода и, пройдя через коллектор и источник $E_{к-э}$, вернутся к эмиттеру. Данный процесс отображен на рисунке 1.7.

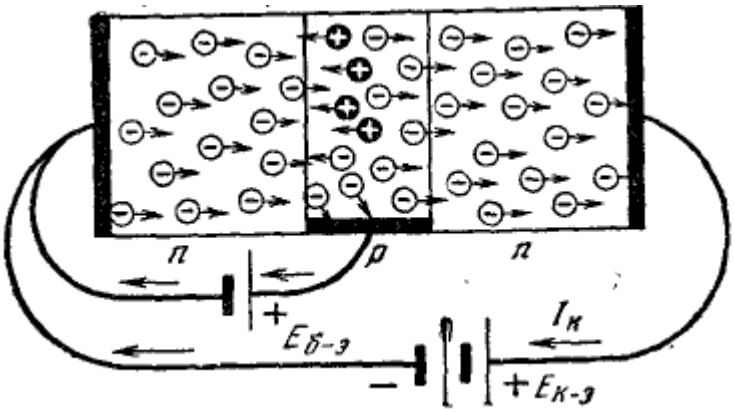

Рисунок 1.7 – Открытый n-p-n транзистор.

1.1.2 Транзистор в вычислительной технике

Транзисторная логика – у транзисторов есть три принципиальных способа соединения, которые соответствуют трём базовым логическим операциям. Основных логических операций всего три: И, ИЛИ, НЕ. Все остальные получаются из их комбинаций. Для примера представим, что через транзисторы мы хотим включить лампочку в комнате и у нас есть выключатель на стене.

Операция «НЕ» – меняет значение на противоположное. В транзисторах на базовом уровне существуют только понятия «есть ток», то есть 1, и «нет тока», то есть 0. Следовательно: НЕ (1) = 0, и НЕ (0) = 1. В примере с лампочкой это звучит как: «Если выключатель выключен, то лампочка должна гореть». Примерная схема такого подключения представлена на рисунке 1.8.

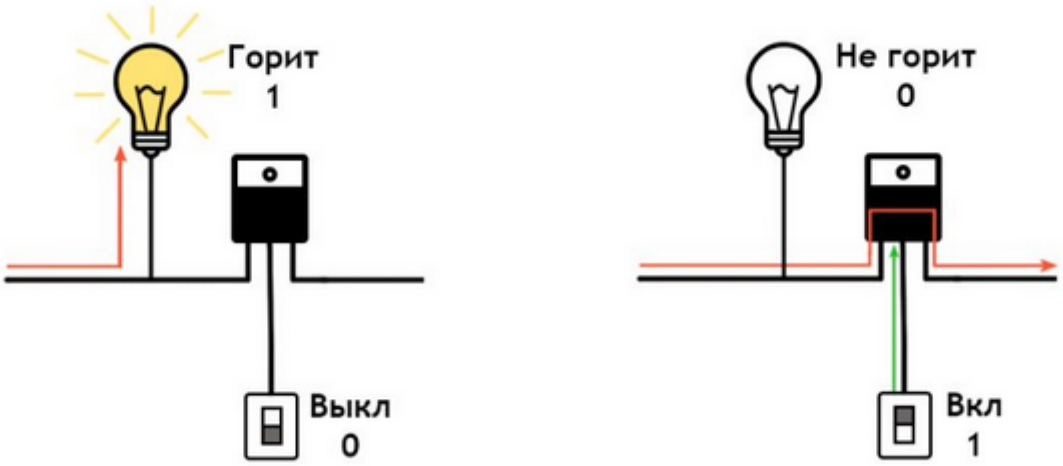

Рисунок 1.8 – Логическая операция «НЕ».

На схемах эта логическая операция обозначается фигурой, изображенной на рисунке 1.9.

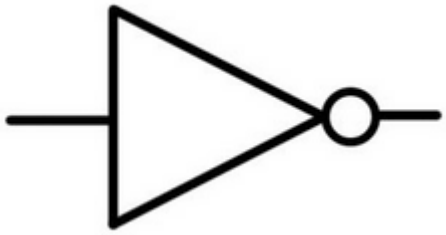

Рисунок 1.9 – Фигура, обозначающая логическую операцию «НЕ».

Операция «И» – в этой операции участвует два параметра, причем результат равен 1 только тогда, когда оба параметра – 1. Схема подключения в примере с лампочкой изображена на рисунке 1.10.

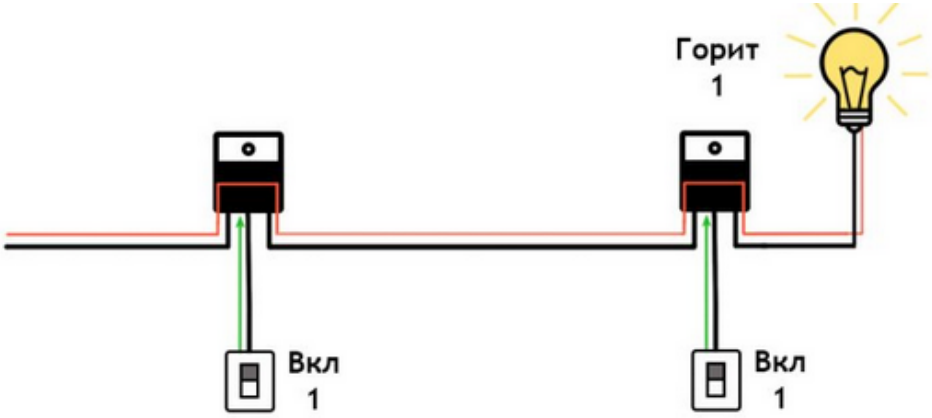

Рисунок 1.10 – Логическая операция «И».

На схемах эта логическая операция обозначается фигурой, изображенной на рисунке 1.11.

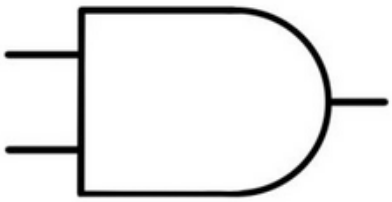

Рисунок 1.11 – Фигура, обозначающая логическую операцию «И».

Операция «ИЛИ» – в этой операции так же участвует два параметра, причем результат равен 0 только тогда, когда оба параметра – 0. А если хотя бы одна единица есть, результат тоже будет единицей. В нашем примере с лампочкой, для того, чтобы лампочка загорелась, должен быть включен хотя бы один выключатель.

Схема подключения в примере с лампочкой изображена на рисунке 1.12.

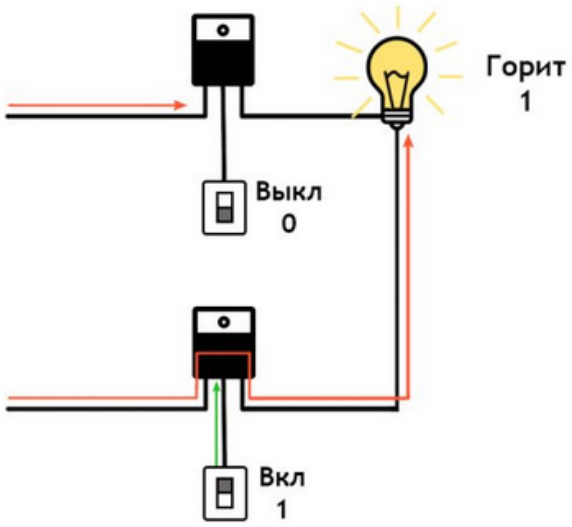

Рисунок 1.12 – Логическая операция «ИЛИ».

На схемах этот логический элемент обозначается фигурой, изображенной на рисунке 1.13.

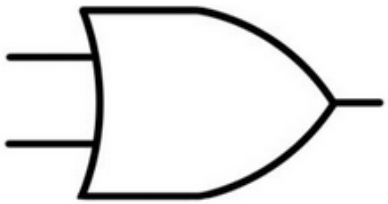

Рисунок 1.13 – Фигура, обозначающая логическую операцию «ИЛИ».

Полубитный сумматор – это устройство, которое вычисляет сумму двух двоичных чисел. Пусть *x* и *y* обозначают двоичные цифры, которые предстоит сложить, а *u* и *v* – двоичные цифры суммы, получающейся на выходе сумматора. $U = x$ И y, v = (НЕ x И y) ИЛИ (x И НЕ y). Таким образом, *u* – разряд переноса, а *v* – сложение по модулю 2. На рисунке 1.14 изображена функциональная схема, реализующая полубитный сумматор, где элемент 3 – *u*, а элемент 4 – *v*.

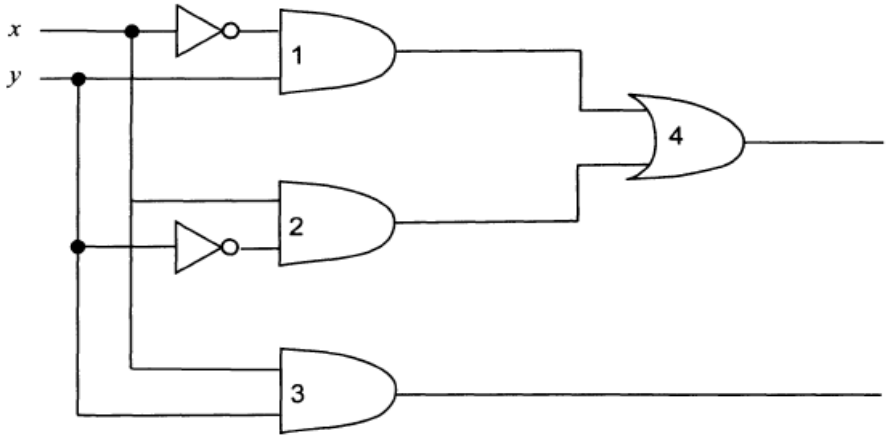

Рисунок 1.14 – Функциональная схема полубитного сумматора

2-битный сумматор – это устройство, которое вычисляет сумму двузначных двоичных чисел, выдавая в качестве ответа трехзначное двоичное число. Например, $10_2 + 11_2 = 101_2$. В схеме 2-битного сумматора используется полубитный сумматор. Обозначим через *a* и *b* цифры первого вводимого в сумматор числа, а через *c* и *d* – цифры второго. Пусть *e*, *f* и *g* – цифры вычисляемой суммы. На рисунке 1.15 представлена функциональная схема, реализующая 2-битный сумматор.

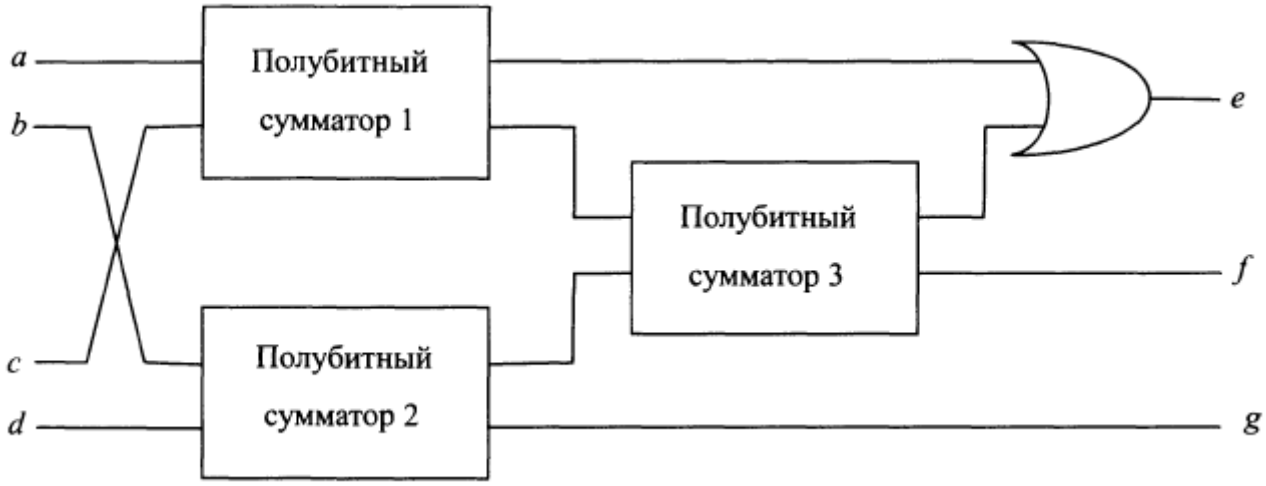

Рисунок 1.15 – Функциональная схема 2-битного сумматора

1.1.3 Демонстрация работы транзистора

Для понимания принципа работы транзистора в качестве ключа, необходимо продемонстрировать как движутся носители зарядов в полупроводниках n и p типа (электроны и дырки соответственно), что происходит при подачи прямого и обратного напряжения на p-n переход, то есть дать возможность пользователю выбирать полярность подключения источника напряжения, а также необходимо визуализировать процесс протекания тока через эмиттер и коллектор n-p-n транзистора при подаче напряжения на базу, в виде анимированного движения отрицательно заряженных частиц.

Чтобы продемонстрировать принцип работы сумматора, необходимо дать возможность ввода пользователю входных значений на полубитный и 2-битный сумматор, с последующим выводом на экран промежуточных (результаты логических элементов схемы) и выходных значений сумматора.

1.1.4 Проверка знаний

После того, как студент ознакомился с теоретическим материалом по теме «Транзистор», он может пройти тестирование, для того чтобы выяснить, насколько хорошо был усвоен полученный теоретический материал.

В случае, если вопросов будет слишком много, студент может утомиться в процессе тестирования, однако, если вопросов будет слишком мало, то по результату тестирования нельзя будет утверждать, что тестируемый усвоил материал. Следует предоставить студенту вопросы закрытого типа с выбором одного правильного ответа из четырех предложенных, а также открытые вопросы с кратким ответом, и подсчитать процент верных ответов (традиционно считается, что уровень знаний тестируемого достаточен, если количество правильных ответов составляет не менее 60%, на хорошем уровне при 75% правильных ответов, и на отличном уровне при 85%).

В базе будет содержаться не менее десяти вопросов, количество вопросов устанавливается перед тестированием, выбор количества варьируется от 2 до 10, по умолчанию количество вопросов равно пяти. Соответствующие вопросы будут выбраны случайным образом из базы вопросов и добавлены в тест, порядок вариантов ответов каждого вопроса при повторном прохождении теста будет меняться.

По завершении тестирования студент сможет ознакомиться со своим результатом. Каждый правильный ответ на вопрос будет стоить по одному баллу. Процент правильных ответов высчитывается по формуле:

$$P = R/N * 100\%,$$

где P – процент правильных ответов,

R – количество баллов,

N – общее количество вопросов.

1.2 Технология обработки информации

Анализ предметной области показал, что программа рассчитана на одного пользователя, который может ознакомиться с теоретическим материалом, перейти в режим демонстрации или пройти тестирование.

В ходе анализа предметной области была построена диаграмма вариантов использования. Она представлена в *Приложении 2.*

Вариант использования «Перейти в режим демонстрации» включает обязательные функции «Выбрать полярность подключения источника тока» и «Ввести входные данные в сумматор и получить выходные результаты».

Вариант использования «Пройти тестирование» расширен функциями «Посмотреть статистику» и «Выбрать количество вопросов» и включает обязательную функцию «Получить результаты».

1.2.1 Диаграмма классов

В ходе анализа предметной области была разработана диаграмма классов. Она представлена в *Приложении 3* в нотации UML с некоторыми нестандартными Qt обозначениями, такими как: **< –** для обозначения сигнала Qt (функция, которая ничего не делает и вызывается когда происходит определенное событие) и **$ –** для обозначения слота Qt (метод-функция, которая вызывается в ответ на определенный сигнал).

Главным классом данной программы является класс ***MainWindow*** (главное окно – контроллер), являющегося наследником класса Qt – ***QmainWindow***. ***MainWindow*** включает в себя:

- указатель на интерфейс Qt;
- указатель на объект класса Teoria (теоретическая подсистема программы);
- указатель на объект класса Demo (демонстрационная подсистема программы);
- указатель на объект класса Test (подсистема тестирования).

Методы:

- функция – слот обработки нажатия на кнопку перехода к теории;
- функция – слот обработки нажатия на кнопку перехода к демонстрации;
- функция – слот обработки нажатия на кнопку перехода к тестированию;
- функция, которая делает главное меню – активным окном приложения.

Класс ***Teoria*** (подсистема теории) является наследником класса Qt – ***QWidget*** и включает в себя указатель на интерфейс Qt.

Методы:

- функция – сигнал для возврата в главное меню;
- функция – слот обработки нажатия на кнопку перехода в главное меню;
- функция, которая переопределяет поведение программы при нажатии на ссылку HTML текста.

Класс ***Demo*** (подсистема демонстрации) является наследником класса Qt – ***QmainWindow*** и включает в себя:

- указатель на объект класса PictureBox, который реализует графическую часть демонстрации;
- указатель на интерфейс Qt.

Методы:

- функция, которая скрывает поля для ввода, предназначенных для ввода входных данных сумматора;
- функция – сигнал для возврата в главное меню;
- функци – слот обработки переключения на другую визуализацию;

- функция – слот обработки нажатия на кнопку смены полярности подключаемого источника тока в демонстрации p-n перехода;
- функция – слот обработки нажатия на кнопку перехода в главное меню;
- функция – слот обрабатывающая ввод в поле для ввода;

Класс ***PictureBox*** (графическая часть подсистемы демонстрации) является наследником класса Qt – ***QGraphicsView*** и включает в себя:

- указатель на объект класса Qt – QGraphicsScene (модель или данные, которые класс PictureBox будет отображать);
- указатель на объект класса P_semiconductor (демонстрация p-полупроводника);
- указатель на объект класса N_semiconductor (демонстрация n-полупроводника);
- указатель на объект класса Diod (демонстрация p-n перехода);
- указатель на объект класса Transistor (демонстрация n-p-n транзистора);
- указатель на объект класса HalfBitSummator (демонстрация полубитного сумматора);
- указатель на объект класса TwoBitSummator (демонстрация двубитного сумматора).

Методы:

- функция, которая обрабатывает событие изменения размеров окна;
- функция, которая ставит на паузу все возможные анимации, которые шли в данный момент;
- функция, отображающая демонстрацию p-полупроводника;
- функция, отображающая демонстрацию n-полупроводника;
- функция, отображающая демонстрацию p-n перехода;
- функция, отображающая демонстрацию n-p-n транзистора;
- функция, отображающая демонстрацию полубитного сумматора;
- функция, отображающая демонстрацию двубитного сумматора;
- функция, передающая входные данные для полубитного сумматора из полей для ввода;
- функция, передающая входные данные для двубитного сумматора из полей для ввода;
- функция, рисующая символ или набор символов (строку) по указанным координатам и с указанным размером шрифта;
- функция, обрабатывающая нажатие на кнопку смены полярности подключаемого источника тока;
- функция – слот, которая сбрасывает анимацию для демонстрации p-полупроводника в исходное состояние.

Класс ***P_semiconductor*** (демонстрация p-полупроводника) включает в себя:

- массив указателей на объект класса Particle (частица – дырка или электрон);
- массив указателей на объект класса Qt – QPropertyAnimation (анимации для частиц);
- указатель на объект класса Qt – QGraphicsItemGroup (группа отображаемых элементов);
- указатель на объект класса Qt – QSequentialAnimationGroup (группа последовательных анимаций);
- константная вещественная переменная – ширина полупроводника;
- константная вещественная переменная – высота полупроводника;
- константная вещественная переменная – половина высоты полупроводника;
- константная вещественная переменная – расстояние между атомами;
- константная вещественная переменная – размер атома;
- константная вещественная переменная – половина размера атома;
- константная вещественная переменная – размер частицы;
- константная вещественная переменная – половина размера частицы;
- целая переменная – длительность анимации;
- целая переменная – длительность паузы между анимациями;
- булевая переменная – активна ли демонстрация;
- булевая переменная – готова ли демонстрация к отрисовке.

Методы:

- функция, которая отрисовывает все элементы демонстрации;
- функция, отрисовывающая источник тока;
- функция, отрисовывающая один атом;
- функция, отрисовывающая все атомы;
- функция, отрисовывающая электроны;
- функция, отрисовывающая границы полупроводника, куда улетают частицы;
- функция, подготавливающая анимации;
- функция, подготавливающая первую и последнюю анимациию;
- функция, сбрасывающая анимации на исходное положение;
- функция, которая ставит анимацию на паузу;
- функция, которая снимает анимацию с паузы.

Класс **N_*semiconductor*** (демонстрация n-полупроводника) включает в себя:

- массив указателей на объект класса Particle (частица – дырка или электрон);
- массив указателей на объект класса Qt – QPropertyAnimation (анимации для частиц);
- указатель на объект класса Qt – QGraphicsItemGroup (группа отображаемых элементов);

- константная вещественная переменная – ширина полупроводника;
- константная вещественная переменная – высота полупроводника;
- константная вещественная переменная – половина высоты полупроводника;
- константная вещественная переменная – размер частицы;
- константная вещественная переменная – половина размера частицы;
- указатель на объект класса Qt – QTimer (таймер, необходимый для анимации движущегося потока частиц);
- целая переменная – длительность анимации;
- целая переменная – номер текущей анимации;
- булевая переменная – активна ли демонстрация;
- булевая переменная – готова ли демонстрация к отрисовке.

Методы:

- функция, которая отрисовывает все элементы демонстрации;
- функция, отрисовывающая источник тока;
- функция, отрисовывающая один пятивалентный ион;
- функция, отрисовывающая все ионы;
- функция, отрисовывающая электроны;
- функция, отрисовывающая границы полупроводника, куда улетают частицы;
- функция, подготавливающая анимации;
- функция, которая ставит анимацию на паузу;
- функция, которая снимает анимацию с паузы.

Класс **Diod** (демонстрация p-n перехода) включает в себя:

- массив указателей на объект класса Particle (частица – дырка или электрон);
- массив указателей на объект класса Qt – QPropertyAnimation (анимации для частиц);
- указатель на объект класса Qt – QGraphicsItemGroup (группа отображаемых элементов);
- константная вещественная переменная – ширина полупроводника;
- константная вещественная переменная – высота полупроводника;
- константная вещественная переменная – половина высоты полупроводника;
- константная вещественная переменная – размер частицы;
- константная вещественная переменная – половина размера частицы;
- указатель на объект класса Qt – QTimer (таймер, необходимый для анимации движущегося потока частиц);
- указатель на объект класса Qt – QGraphicsItemGroup (группа движущихся электронов при прямом подключении диода);

- указатель на объект класса Qt – QGraphicsItemGroup (группа недвижущихся статичных электронов при обратном подключении диода);
- указатель на объект класса Qt – QGraphicsItemGroup (группа с элементами источника тока, прямое подключение);
- указатель на объект класса Qt – QGraphicsItemGroup (группа с элементами источника тока, обратное подключение);
- целая переменная – длительность анимации;
- целая переменная – номер текущей анимации;
- булевая переменная – активна ли демонстрация;
- булевая переменная – готова ли демонстрация к отрисовке.

Методы:

- функция, которая отрисовывает все элементы демонстрации;
- функция, отрисовывающая источник тока в прямом подключении;
- функция, отрисовывающая источник тока в обратном подключении;
- функция, отрисовывающая движущиеся частицы;
- функция, отрисовывающая статичные недвижущиеся частицы;
- функция, отрисовывающая границы полупроводника, куда улетают частицы;
- функция, подготавливающая анимации;
- функция, которая ставит анимацию на паузу;
- функция, которая снимает анимацию с паузы.
- функция – слот, которая обрабатывает нажатие на кнопку смены полярности подключаемого источника тока.

Класс **Transistor** (демонстрация n-p-n транзистора) включает в себя:

- массив указателей на объект класса Particle (частица – дырка или электрон);
- массив указателей на объект класса Qt – QPropertyAnimation (анимации для частиц);
- указатель на объект класса Qt – QGraphicsItemGroup (группа отображаемых элементов);
- константная вещественная переменная – ширина полупроводника;
- константная вещественная переменная – высота полупроводника;
- константная вещественная переменная – половина высоты полупроводника;
- константная вещественная переменная – размер частицы;
- константная вещественная переменная – половина размера частицы;
- указатель на объект класса Qt – QTimer (таймер, необходимый для анимации движущегося потока частиц);

- указатель на объект класса Qt – QGraphicsItemGroup (группа движущихся электронов при прямом подключении диода);
- указатель на объект класса Qt – QGraphicsItemGroup (группа с элементами источника тока);
- целая переменная – длительность анимации;
- целая переменная – номер текущей анимации;
- булевая переменная – активна ли демонстрация;
- булевая переменная – готова ли демонстрация к отрисовке.

Методы:

- функция, которая отрисовывает все элементы демонстрации;
- функция, отрисовывающая источник тока;
- функция, отрисовывающая движущиеся частицы;
- функция, отрисовывающая границы полупроводника, куда улетают частицы;
- функция, которая отрисовывает стрелку (направление движения электронов);
- функция, подготавливающая анимации;
- функция, которая ставит анимацию на паузу;
- функция, которая снимает анимацию с паузы;
- функция – слот, которая подготавливает отдельную анимацию для электронов, идущих на базу.

Класс **Particle** (частица – дырка или электрон), который является частью классов ***P_semiconductor***, ***N_semiconductor***, ***Transistor***, ***Diod***, включает в себя:

- перечисление, представляющее собой тип частицы, где 0 – электрон, а 1 – дырка;
- объект класса Qt – QRectF (область экрана, занимаемая частицой в координатах левого верхнего угла и значение ширины и высоты в пикселях);
- тип частицы (перечисление, описанное выше);
- объект класса Qt – QColor (цвет частицы)

Методы:

- функция, которая возвращает занимаемую частицей область экрана;
- функция, отрисовывающая частицу;
- функция, позволяющая редактировать область, занимаемую частицой (т.е можно задавать размер и координаты местоположения).

Класс **HalfBitSummator** (демонстрация полубитного сумматора), который является частью класса ***PictureBox*** включает в себя:

- массив указателей на объект класса Qt – QGraphicsTextItem (входные, промежуточные и выходные данные сумматора);

- объект класса SummatorLogic;
- указатель на объект класса QGraphicsItemGroup (группа отрисовываемых элементов).

Методы:

- функция, отрисовывающая функциональную схему сумматораж
- функция, отрисовывающая элемент операции «НЕ»;
- функция, отрисовывающая элемент операции «ИЛИ»;
- функция, отрисовывающая элемент операции «И»;
- функция, подготавливающая начальные позиции для текстовых элементов выводимых данных;
- функция, вызывающая соответствующую функцию в объекте класса SummatorLogic и выводящая результаты на экран (входные, промежуточные и выходные данные);
- функция, переводящая значение булевой переменную в символ 0 или 1;

Класс **Summator** (функциональная часть сумматора), который является частью класса ***HalfBitSummator*** включает в себя:

- вложенный класс HalfBitRes, в который входят выходные данные u и v (булевы переменные), а также промежуточные значения oper1 и oper2;
- вложенный класс TwoBitRes, в который входят выходные данные e, f и g, а также пары элементов sum1, sum2, sum3, где каждая сумма представляет собой выходные данные выполнения функции вычисления полубитного сумматора, исключая промежуточные.

Методы:

- функция, реализующая функциональную часть полубитного сумматора, принимая два входных параметра, возвращает результат в виде класса HalfBitRes;
- функция, реализующая функциональную часть полубитного сумматора, принимая два входных параметра, возвращает результат в виде пары элементов (выходные данные, исключая промежуточные);
- функция, реализующая функциональную часть двубитного сумматора, принимая четыре входных параметра, возвращает результат в виде класса TwoBitRes.

Класс **TwoBitSummator** (функциональная часть двубитного сумматора), который является наследником класса ***HalfBitSummator*** и частью класса ***PictureBox***, включает в себя:

- указатели на объекты класса QpointF (координаты для входных данных a, b, c, d; для выходных e, f, g; и промежуточных; sum1, sum3).

Методы:

- функция, отрисовывающая функциональную схему двубитного сумматора;

- функция, подготавливающая начальные позиции для текстовых элементов выводимых данных;
- функция, вызывающая соответствующую функцию в объекте класса SummatorLogic и выводящая результаты на экран (входные, промежуточные и выходные данные);
- функция, отрисовывающая прямоугольник по заданным координатам (под прямоугольником понимается полубитный сумматор на данной схеме).

Класс ***Test*** (подсистема тестирования) является наследником класса Qt – ***Qwidget*** и включает в себя:

- строка, содержащая весь текст из файла с базой вопросов;
- массив объектов класса ***Question***, состоящего из строки с текстом вопроса, целого числа – типа вопроса, и массива из четырех объектов класса ***Answer***, который в свою очередь состоит из строки – текста ответа и булевы переменной – правильный ли ответ.
- объект класса ***user_answers***, состоящего из массива целых чисел – ответов пользователя на вопросы закрытого типа и массива строк – ответов пользователя на вопросы открытого типа с кратким ответом;
- количество вопросов N;
- индекс текущего вопроса i_step;
- указатель на интерфейс Qt.

Методы:

- функция – сигнал для возврата в главное меню;
- функции обработки нажатий на кнопки перехода к следующему и предыдущему вопросам;
- функции – слоты, обрабатывающие выбор пользователем флаговых кнопок (чекбоксы);
- функция – слот обработки нажатия на кнопку завершения тестирования;
- функция – слот обработки ввода пользователем ответа на вопрос открытого типа;
- функция – слот обработки нажатия на кнопку начала тестирования;
- функция – слот обработки нажатия на кнопку перехода в главное меню;
- функция – слот обработки нажатия на кнопку просмотра статистики тестирования;
- функция, которая начинает процесс тестирования;
- функция формирования вопросов;
- функция вывода текущего вопроса (как закрытого, так и открытого типа) на экран;
- функция, проверяющая ответы пользователя;
- функция, которая выводит результаты тестирования;

- функция, которая сохраняет результаты в файл со статистикой;
- функция, которая читает и выводит статистику на экран.

1.2.2 Форматы данных для подсистемы теории

Теоретический материал, база вопросов и статистика результатов тестирования хранятся в текстовых файлах с названиями «text.txt», «quest.txt» и «stats.txt» соответственно.

Теоретический материал представляет собой текстовый файл, содержащий гипертекст (HTML). Текст в файле должен быть отформатирован вручную с использованием стандартных HTML-тегов. Для того, чтобы корректно работало содержание, в тексте файла с теорией все необходимые заголовки или оглавления должны быть выделены тегом ****** ******. На месте многоточия следует указать число – каким по порядку элементом в таблице с содержанием будет данный заголовок.

1.2.3 Форматы данных для подсистемы тестирования

База вопросов представляет собой текстовый файл, который имеет следующую структуру: каждый вопрос начинается с символа «**?**», после которого указывается число — тип вопроса. Если указана цифра «**1**», значит описывается вопрос закрытого типа, в случае, если «**2**», то вопрос – открытый с кратким ответом. Вопрос с закрытым типом занимает пять строк – на первой строке (после символов «**?1**») находится текст вопроса, на оставшихся четырех — дистракторы, причем первый вариант ответа является правильным. Вопрос с открытым типом занимает две строки – на первой строке (после символов «**?2**») находится текст вопроса, на второй строке – текст правильного ответа. В программе вопрос будет представлен в виде структуры, в которой хранятся строка с вопросом, тип вопроса и четыре дистрактора. В случае вопроса открытого типа, используется только первый дистрактор из четырех в качестве правильного краткого ответа. Дистрактор, в свою очередь, тоже является структурой и состоит из строки (вариата ответа), и переменной логического типа, которая определяет правильность данного ответа.

1.2.4 Алгоритм вывода теоретического материала

Дано: ***text.txt*** **–** текстовый файл с теорией, ***window*** – окно, в которое выводится текст теории, ***contentTable*** – вспомогательное окно, в котором находится таблица с содержанием

пока пользователь не дал команду выйти в главное меню

| вывести содержимое файла text.txt в окно window

| найти все выделенные тегом заголовки и вывести их в contentTable

конец цикла

1.2.5 Алгоритм реализующий полубитный сумматор

Дано: ***x*** и ***y*** – входные двоичные цифры, которые предстоит сложить, ***u*** и ***v*** – выходные двоичные цифры суммы, ***oper1*** и ***oper2*** – промежуточные операции, используемые для реализации сложения по модулю два (***x*** ^ ***y***).

u = ***x*** & ***y***

oper1 = ~***x*** & ***y***

oper2 = ***x*** & ~***y***

v = ***oper1*** | ***oper2***

вернуть результат{***u***, ***v***, ***oper1***, ***oper2***}

1.2.6 Алгоритм демонстрирующий работу полубитного сумматора

Дано: ***x*** и ***y*** – входные двоичные цифры, которые предстоит сложить, ***u*** и ***v*** – выходные двоичные цифры суммы, ***oper1*** и ***oper2*** – промежуточные операции, используемые для реализации сложения по модулю два (***x*** ^ ***y***).

ввод пользователя: ***x*** и ***y***

выполнить алгоритм 1.2.5 с параметрами ***x*** и ***y***

вывод на экран функциональной схемы, с отмеченными на ней промежуточными значениями ***oper1*** и ***oper2*** и выходными значениями ***u*** и ***v***

1.2.7 Алгоритм реализующий 2-битный сумматор

Дано: ***a*** и ***b*** – входные двоичные цифры первого вводимого в сумматор числа, ***c*** и ***d*** – цифры второго; ***e***, ***f*** и ***g*** – цифры вычисляемой суммы, ***sum1***, ***sum2***, ***sum3*** – промежуточные операции (полубитные сумматоры).

sum1 **=** число с цифрами ***sum1_u*** и ***sum1_v*** из алгоритма 1.2.5 с параметрами ***a*** и ***c***

sum2 = число с цифрами ***sum2_u*** и ***sum2_v*** из алгоритма 1.2.5 с параметрами ***b*** и ***d***

sum3 = число с цифрами ***sum3_u*** и ***sum3_v*** из алгоритма 1.2.5 с параметрами ***sum1_v*** и ***sum2_u***

e = ***sum1_u*** | ***sum3_u***

f = ***sum3_v***

g = ***sum2_v***

вернуть результат{***e, f, g, sum1, sum2, sum3***}

1.2.8 Алгоритм демонстрирующий работу 2-битного сумматора

Дано: ***a*** и ***b*** – входные двоичные цифры первого вводимого в сумматор числа, ***c*** и ***d*** – цифры второго; ***e***, ***f*** и ***g*** – цифры вычисляемой суммы, ***sum1***, ***sum2***, ***sum3*** – промежуточные операции (полубитные сумматоры).

ввод пользователя: ***a***, ***b***, ***c***, ***d***

выполнить алгоритм 1.2.7 с параметрами ***a***, ***b***, ***c***, ***d***

вывод на экран функциональной схемы, с отмеченными на ней промежуточными значениями ***sum1***, ***sum2***, ***sum3*** и выходными значениями ***e***, ***f*** и ***g***

1.2.9 Алгоритм реализующий режим демонстрации

пока пользователь не дал команду выйти в главное меню
- если пользователь выбрал демонстрацию p-полупроводника, то вывести анимацию на экран, изображающую движение дырок в p-полупроводнике
- если пользователь выбрал демонстрацию p-полупроводника, то вывести анимацию на экран, изображающую движение электронов в p-полупроводнике
- если пользователь выбрал демонстрацию p-n перехода, то
  - определить ***полярность*** подключаемого источника тока
  - если ***полярность*** соответствует прямому направлению прикладываемого напряжения то вывести анимацию на экран, изображающую протекание тока в полупроводнике через p-n переход
  - иначе вывести изображение на экран, изображающее состояние p-n перехода при обратном направлении приложенного напряжения (отсутствие тока)
- конец ветвления
- если пользователь выбрал демонстрацию транзистора, то вывести анимацию на экран, изображающую протекание тока в транзисторе, при подаче тока на базу
- если пользователь выбрал демонстрацию полубитного сумматора, то выполнить алгоритм 1.2.6
- если пользователь выбрал демонстрацию 2-битного сумматора, то выполнить алгоритм 1.2.8

конец цикла

1.2.10 Алгоритм чтения базы вопросов из файла

Дано: ***Answer*** — структура с полями ***text*** (строковая константа, текст варианта ответа) и переменной логического типа ***right*** (определяет правильность ответа), ***Question*** — структура с полями ***text*** (строковая константа, текст вопроса), ***mode*** - тип вопроса (1 - закрытый или 2 - открытый) и ***arr[4]*** – массив с элементами типа ***Answer***. ***Questions[N]*** – массив с элементами типа ***Question***, ***N*** – количество вопросов в базе вопросов, ***quest.txt*** – текстовый файл с базой вопросов.

пока (в файле есть строки с текстом)

| прочитать строку ***temp_string***
| объявить переменную ***temp_question*** типа ***Question***
| прочитать тип вопроса ***mode*** для ***temp_question*** из строки ***temp_string***
| прочитать текст вопроса ***text*** для temp_question из строки ***temp_string***
| если mode = 1 (закрытый тип вопроса) то
| | отметить первый вариант ответа ***temp_question.arr[0]*** правильным
| | пока i < 4
| | | прочитать следующую строку из файла в ***temp_string***
| | | прочитать текст ответа ***temp_question.arr[i].text*** из ***temp_string***
| | конец цикла
| а если mode = 2 (открытый тип вопроса) то
| | прочитать следующую строку из файла в ***temp_string***
| | прочитать текст ответа ***temp_question.arr[0].text*** из ***temp_string***
| конец ветвления
| поместить ***temp_question*** в массив ***Questions[N]***
конец цикла

1.2.11 Алгоритм тестирования

Дано: ***Answer*** — структура с полями ***text*** (строковая константа, текст варианта ответа) и переменной логического типа ***right*** (определяет правильность ответа), ***Question*** — структура с полями ***text*** (строковая константа, текст вопроса), ***mode*** - тип вопроса (1 - закрытый или 2 - открытый) и ***arr[4]*** – массив с элементами типа ***Answer***. ***Questions[N]*** – массив с элементами типа ***Question***, ***N*** – количество вопросов в базе вопросов, ***size*** – количество вопросов в тесте.

перемешать вопросы в массиве вопросов
перемешать варианты ответов в вопросах закрытого типа
цел ***i*** = 0
ch[size] – массив целых чисел, заполненный значением -1 (ответы пользователя на вопросы закрытого типа)
ch_str[size] – массив пустых строк (ответы пользователя на вопросы открытого типа)
пока пользователь не дал команду завершить тестирование
| вывести на экран текст вопроса
| если ***Questions[i].mode*** = 1 (вопрос закрытого типа) то
| | вывести на экран четыре варианта ответа
| | если ***ch[i]*** = -1 то записать ответ пользователя в ***ch[i]***

| а если ***Questions[i].mode*** = 2 (вопрос открытого типа)
| | то предоставить пользователю поле для ввода краткого ответа
| | если ***ch_str[i]*** — пуста строка то записать краткий ответ пользователя в ***ch_str[i]***
| конец ветвления
| если пользователь запросил следующий вопрос и ***i*** < ***size*** -1 то увеличить ***i*** на 1
| если пользователь запросил предыдущий вопрос и ***i*** > 0 то уменьшить ***i*** на 1

конец цикла

подсчитать количество правильных ответов и процент правильных ответов

вывести результаты тестирования

1.2.12 Основной алгоритм

вывести условия меню

пока пользователь не дал команду завершить программу

| если пользователь дал команду «перейти в режим теории» то вызвать алгоритм вывода теоретического материала 1.2.4
| если пользователь дал команду «перейти в режим демонстрации» то вызвать алгоритм реализующий режим демонстрации 1.2.9
| если пользователь дал команду «перейти в режим тестирования» то вызвать алгоритм тестирования 1.2.11

конец цикла

1.3 Входные и выходные данные

Входные данные:

- Выбор пунктов меню.
- Полярность источника тока, для демонстрации p-n перехода;
- Входные данные сумматора, для демонстрации работы транзистора в вычислительной технике;
- Ответ пользователя на вопрос тестирования;

Выходные данные:

- Теоретический материал, предоставляемый пользователю для изучения.
- Демонстрация движения электронов в n-полупроводнике;
- Демонстрация движения дырок в p-полупроводнике;
- Демонстрация процесса p-n перехода;
- Демонстрация процесса работы n-p-n транзистора;

- Демонстрация принципа работы полубитного сумматора, включая результаты вычислений;
- Демонстрация принципа работы 2-битного сумматора, включая результаты вычислений;
- Тестовые вопросы;
- Оценка, полученная в результате тестирования.

1.4 Системные требования

Для реализации поставленной задачи была выбрана графическая библиотека Qt, поскольку она является бесплатным кроссплатформенным фреймворком с открытым исходным кодом для разработки программного обеспечения на языке программирования C++. Возможности библиотеки Qt позволяют быстро разработать приложение с оконным интерфейсом и с необходимыми для демонстрации графическими элементами (в Qt можно оперировать графическими примитивами), поскольку вместе с библиотекой поставляется интегрированная среда Qt Creator и визуальная среда разработки графического интерфейса Qt Designer, позволяющей создавать диалоги и формы в режиме WYSIWYG («что видишь, то и получишь»). Таким образом, в качестве среды разработки была выбрана интегрированная среда Qt Creator (версия Qt 5.4) и язык C++ (стандарт C++ 11 и более поздние).

Рекомендуемая конфигурация:

- Intel-совместимый процессор с частотой не менее 1,6 ГГц;
- не менее 512 МБ ОЗУ;
- не менее 50 МБ свободного места на диске;
- дисковод CD-ROM/DVD-ROM

Операционная система: Windows XP (x86) с пакетом обновления 3 (SP3) или более поздние.

2 РАБОЧИЙ ПРОЕКТ

2.1 Общие сведения о работе системы

Программный продукт разработан в интегрированной среде Qt Creator 4.12.3 (версия фреймворка Qt 5.6) на языке C++ (стандарт C++ 11 и более поздние), с использованием компилятора MinGW GCC (версия 4.9.2). Программа работает под управлением операционной системы Windows XP (x86) Professional (SP3) и более поздними.

2.2 Функциональное назначение программного продукта

Разработанный программный продукт предназначен для повышения качества знаний пользователей по теме «Транзистор», а также для снижения нагрузки на преподавателя. Программа имеет следующие функциональные возможности:

- предоставление пользователю теоретического материала по теме «Транзистор»;
- демонстрирация процесса работы транзистора в режиме ключа и принципа работы транзистора в вычислительной технике на примере простого сумматора;
- тестирование пользователя по теме «Транзистор»;

Программа имеет следующие функциональные ограничения:

- программа не предоставляет возможность изменять схему электрической цепи или схему сумматора;
- программа не предоставляет возможность шифровать / расшифровывать текстовые файлы с базой вопросов и с теорией;
- программа не должна обеспечивать редактирование статистики результатов тестирования.

2.3 Инсталляция и выполнение программного продукта

Программу можно установить на компьютер с помощью инсталляционного пакета, а после запустить с помощью соответствующего ярлыка на рабочем столе, который имеет название «Учебно-демонстрационная программа (Транзистор)».

Для выполнения программы необходимо:

1. Скопировать на жесткий диск компьютера файл с инсталляционным пакетом – setup.exe.
2. Запустить setup.exe и следовать инструкциям инсталлятора.
3. Запустить ярлык на рабочем столе с названием «Учебно-демонстрационная программа (Транзистор)».

2.4 Описание программы

Программа состоит из 19 классов – 12 классов для модуля демонстрации, 5 классов – для модуля тестирования, 1 класс – для модуля теории, 1 класс – класс «интерфейс-контроллер».

В таблице 2.1 приведено описание класса Demo, реализующего модуль демонстрации и используемое в классе-контроллере MainWindow (заголовочный файл demo.h).

Таблица 2.1 – Описание класса Demo

Поле	**Тип**	**Назначение**
ui	Ui::Demo*	Поле, необходимое для работы ui форм в Qt
graphic	PictureBox*	Класс, реализующий графическую часть модуля демонстрации
Метод		**Описание**
Demo(QWidget *parent = nullptr)		Конструктор класса
~Demo()		Деструктор класса
void hideInputs()		Скрывает поля для ввода, которые предназначены для сумматоров, в случае если данный момент визуализируется не сумматор
void return_to_menu()		Сигнал о возврате из модуля теории в главное меню
void on_tabWidget_currentChanged(int index)		Метод, обрабатывающий переключение на другую визуализацию
void on_pushButton_changePolarity_clicked()		Метод, обрабатывающий изменения переключения полярности подключения
void on_pushButton_toMenu_clicked()		Нажатие на кнопку «Вернуться в главное меню» (вызывает сигнал return_to_menu)
void on_input1_textEdited(const QString &arg1)		Метод, обрабатывающий ввод в текстовое поле 1
void on_input2_textEdited(const QString &arg1)		Метод, обрабатывающий ввод в текстовое поле 2

В таблице 2.2 приведено описание класса PictureBox, реализующего графическую часть модуля демонстрации и используемое в классе Demo (заголовочный файл picturebox.h).

Таблица 2.2 – Описание класса PictureBox

Поле	**Тип**	**Назначение**
scene	QGraphicsScene*	Графическая сцена, на которой происходят все отрисовки
p_semi	P_semiconductor*	Класс, визуализирующий p-полупроводник
n_semi	N_semiconductor*	Класс, визуализирующий n-полупроводник
perehod	Diod*	Класс, визуализирующий p-n переход

Продолжение таблицы 2.2

Поле	**Тип**	**Назначение**
trans	Transistor*	Класс, визуализирующий n-p-n транзистор
halfbit	HalfBitSummator*	Класс, визуализирующий полубитный сумматор
twobit	TwoBitSummator*	Класс, визуализирующий двубитный сумматор
Метод		**Описание**
PictureBox(QWidget *parent = nullptr)		Конструктор класса
~PictureBox()		Деструктор класса
void resizeEvent(QResizeEvent *event)		Метод, обрабатывающий изменения размеров окна с масштабированием содержимого сцены
void pauseAll()		Остановить все активные анимации и скрыть все активные визуализации со сцены
void show_p_semi()		Вывести на сцену группу элементов p_semi
void show_n_semi()		Вывести на сцену группу элементов n_semi
void show_diod()		Вывести на сцену группу элементов perehod
void show_transistor()		Вывести на сцену группу элементов trans
void show_halfbitsum()		Вывести на сцену группу элементов halfbit
void show_twobitsum()		Вывести на сцену группу элементов twobit
void callHalfBitSum(bool x, bool y)		Переадресовываем данные полученные из полей ввода в функциональную часть сумматора
void callTwoBitSum(bool a, bool b, bool c, bool d)		Переадресовываем данные полученные из полей ввода в функциональную часть сумматора
QGraphicsTextItem *drawSigns(const std::string &c, const QPointF &coords, QGraphicsItemGroup *group, int FontSize = 28)		Отрисовка символа или строки символов, по указанным координатам, и с указателем на родительскую группу элементов
void changePolarity()		Вызываем соотвествующий метод в perehod, меняющий полярность подключения источника тока
void reset_Animations_for_p_semi()		Сбрасываем анимацию в исходное положение в p_semi

В таблице 2.3 приведено описание класса P_semiconductor, визуализирущий p-полупроводник и используемый в класе PictureBox (заголовочный файл p_semiconductor.h).

Таблица 2.3 – Описание класса P_semiconductor

Поле	**Тип**	**Назначение**
particleArr	std::vector<Particle*>	Массив частиц
animationArr	std::vector<QProperty Animation*>	Массив анимаций для частиц
group	QGraphicsItemGroup*	Группа с элементами сцены
animation_group	QSequentialAnimation Group*	Последовательная группа для анимаций
w	const double	Ширина полупроводника в пикселях
h	const double	Высота полупроводника
halfH	const double	Половина высоты полупроводника
betweenAtoms	const double	Расстояние между атомами
atomSize	const double	Размер атома
halfAtomSize	const double	Половина размера атома
particleSize	const double	Размер частицы
halfParticleSize	const double	Половина размера частицы
animDuration	int	Длительность анимации частицы
pauseDuration	int	Длительность паузы между анимациями
active	bool	Активна ли анимация в данный момент
readyToShow	bool	Готова ли визуализация к отображению
Метод		**Описание**
P_semiconductor(QWidget *parent = nullptr)		Конструктор класса
~P_semiconductor()		Деструктор класса
void draw()		Функция, которая отрисовывает процесс движения электронов и дырки в p-полупроводнике
void drawPowerSource()		Отрисовывает схему источника питания
void drawAtom(QGraphicsItem* parent, const QPointF &coords)		Отрисовывает один атом по заданной координате
void drawAtoms(QGraphicsItem* parent, const QPointF &coords)		Отрисовывает все атомы по начальной заданной координате
void drawElectrons(const QPointF &coords)		Отрисовывает электроны

Продолжение таблицы 2.3

Метод	**Описание**
void drawBounds()	Отрисовывает границы полупроводника, куда улетают частицы
void prepareAnim(const QPointF &coords)	Подготавливает анимации (инициализация начальными и конечными координатами)
void prepareAnimFirstAndLast(const QPointF &coords)	Подготавливает первую и последнюю анимацию
void resetAnimations()	Сброс анимаций на исходное положение
void pause()	Ставит анимацию на паузу
void unpause()	Снимает анимацию с паузы

В таблице 2.4 приведено описание класса N_semiconductor, визуализирущий n-полупроводник и используемый в класе PictureBox (заголовочный файл n_semiconductor.h).

Таблица 2.4 – Описание класса N_semiconductor

Поле	**Тип**	**Назначение**
particleArr	std::vector<Particle*>	Массив частиц
animationArr	std::vector<QProperty Animation*>	Массив анимаций для частиц
group	QGraphicsItemGroup*	Группа с элементами сцены
w	const double	Ширина полупроводника в пикселях
h	const double	Высота полупроводника
halfH	const double	Половина высоты полупроводника
particleSize	const double	Размер частицы
halfParticleSize	const double	Половина размера частицы
animTimer	QTimer*	Таймер, выпускающий частицы с определенной переодичностью для эффекта потока частиц
animDuration	int	Длительность анимации частицы
i	int	Индекс текущей выпускаемой частицы
active	bool	Активна ли анимация в данный момент
readyToShow	bool	Готова ли визуализация к отображению
Метод		**Описание**
N_semiconductor(QWidget *parent = nullptr)		Конструктор класса

Продолжение таблицы 2.4

Метод	Описание
~N_semiconductor()	Деструктор класса
void draw()	Функция, которая отрисовывает процесс движения электронов в n-полупроводнике
void drawPowerSource()	Отрисовывает схему источника питания
void drawIon(const QPointF &coords)	Отрисовывает один пятивалентный ион по заданной координате
void drawIons()	Отрисовывает все ионы
void drawElectrons()	Отрисовывает электроны
void drawBounds()	Отрисовывает границы полупроводника, куда улетают частицы
void prepareAnim()	Подготавливает анимации (инициализация начальными и конечными координатами)
void pause()	Ставит всю анимацию (все электроны) на паузу
void unpause()	Снимает анимацию с паузы
void resumeAnim()	Метод, снимающий паузу с анимации под индексом i. Этот метод вызывает таймер.

В таблице 2.5 приведено описание класса Diod, визуализирущий p-n переход и используемый в класе PictureBox (заголовочный файл diod.h).

Таблица 2.5 – Описание класса Diod

Поле	Тип	Назначение
particleArr	std::vector<Particle*>	Массив частиц
animationArr	std::vector<QProperty Animation*>	Массив анимаций для частиц
group	QGraphicsItemGroup*	Группа с элементами сцены
w	const double	Ширина полупроводника в пикселях
h	const double	Высота полупроводника
halfH	const double	Половина высоты полупроводника
particleSize	const double	Размер частицы
halfParticleSize	const double	Половина размера частицы

Продолжение таблицы 2.5

Поле	**Тип**	**Назначение**
animTimer	QTimer*	Таймер, выпускающий частицы с определенной переодичностью для эффекта потока частиц
movingParticles	QGraphicsItemGroup*	Группа с элементами движущихся частиц
idleParticles	QGraphicsItemGroup*	Группа с элементами статичных недвижущихся частиц
powerSourceOn	QGraphicsItemGroup*	Группа с элементами источника тока (прямое подключение)
powerSourceOff	QGraphicsItemGroup*	Группа с элементами источника тока (обратное подключение)
animDuration	int	Длительность анимации частицы
i	int	Индекс текущей выпускаемой частицы
active	bool	Активна ли анимация в данный момент
readyToShow	bool	Готова ли визуализация к отображению
Метод		**Описание**
Diod(QWidget *parent = nullptr)		Конструктор класса
~Diod()		Деструктор класса
void draw()		Функция, которая отрисовывает процесс движения электронов и дырок в p-n переходе
void drawPowerSourceOn()		Отрисовывает схему источника питания при прямом подключении
void drawPowerSourceOff()		Отрисовывает схему источника питания при обратном подключении
void drawParticles()		Отрисовывает движущиеся электроны и дырки
void drawIdleParticles()		Отрисовывает статичные электроны и дырки (при обратном подключении источника тока)
void drawBounds()		Отрисовывает границы полупроводника, куда улетают частицы
void prepareAnim()		Подготавливает анимации (инициализация начальными и конечными координатами)
void pause()		Ставит всю анимацию (все электроны) на паузу

Продолжение таблицы 2.5

Метод	**Описание**
void unpause()	Снимает анимацию с паузы
void resumeAnim()	Метод, снимающий паузу с анимации под индексом i. Этот метод вызывает таймер.
void changePolarity()	Перерисовывает схему источника тока с прямого подключения на обратное, и наоборот

В таблице 2.6 приведено описание класса Transistor, визуализирущий n-p-n транзистор и используемый в класе PictureBox (заголовочный файл transistor.h).

Таблица 2.6 – Описание класса Transistor

Поле	**Тип**	**Назначение**
particleArr	std::vector<Particle*>	Массив частиц
animationArr	std::vector<QProperty Animation*>	Массив анимаций для частиц
group	QGraphicsItemGroup*	Группа с элементами сцены
w	const double	Ширина полупроводника в пикселях
h	const double	Высота полупроводника
halfH	const double	Половина высоты полупроводника
particleSize	const double	Размер частицы
halfParticleSize	const double	Половина размера частицы
animTimer	QTimer*	Таймер, выпускающий частицы с определенной переодичностью для эффекта потока частиц
movingParticles	QGraphicsItemGroup*	Группа с движущимися частицами
powerSource	QGraphicsItemGroup*	Группа с элементами схемы источника питания
animDuration	int	Длительность анимации частицы
i	int	Индекс текущей выпускаемой частицы
active	bool	Активна ли анимация в данный момент
readyToShow	bool	Готова ли визуализация к отображению
Метод		**Описание**
N_semiconductor(QWidget *parent = nullptr)		Конструктор класса

Продолжение таблицы 2.6

Метод	**Описание**
~N_semiconductor()	Деструктор класса
void draw()	Функция, которая отрисовывает процесс движения электронов в n-p-n транзисторе
void drawPowerSource()	Отрисовывает схему источника питания
void drawParticles()	Отрисовывает движущиеся электроны
void drawBounds()	Отрисовывает границы полупроводника, куда улетают частицы
void prepareAnim()	Подготавливает анимации (инициализация начальными и конечными координатами)
void drawArrow(const QPointF &coords)	Нарисовать стрелку, указывающую направление движения электронов
void pause()	Ставит всю анимацию (все электроны) на паузу
void unpause()	Снимает анимацию с паузы
void resumeAnim()	Метод, снимающий паузу с анимации под индексом i. Этот метод вызывает таймер

В таблице 2.7 приведено описание класса Particle, отрисовывающий анимированную частицу (дырку или электрон) и используемый в классах P_semiconductor, N_semiconductor, Diod и Transistor (заголовочный файл particle.h).

Таблица 2.7 – Описание класса Particle

Поле	**Тип**	**Назначение**
rect	QRectF	Область экрана в пикселях, занимаемая частицой
type	ParticleType	Тип частицы
color	QColor	Цвет частицы
Метод		**Описание**
Particle()		Конструктор класса
Particle(ParticleType _type)		Конструктор класса с типом частицы
~Particle()		Деструктор класса
virtual QRectF boundingRect()		Функция, возвращая занимаемую область

Продолжение таблицы 2.7

Метод	Описание
virtual void paint(QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget)	Отрисовывает частицу
const QRectF geometry()	Выполняет аналогичную функцию, как и функция boundingRect()
void setGeometry(const QRectF &value)	Устанавливает область, то есть задает координаты и размеры частицы

В таблице 2.8 приведено описание вложенного перечисления ParticleType (в классе Particle), в котором перечислены типы частиц (заголовочный файл particle.h).

Таблица 2.8 – Описание перечисления ParticleType

Название	Номер	Назначение
electron	0	Электрон
hole	1	Дырка

В таблице 2.9 приведено описание класса HalfBitSummator, демонстрирующий полубитный сумматор и используемый в классе PictureBox (заголовочный файл halfbitsummator.h).

Таблица 2.9 – Описание класса HalfBitSummator

Поле	Тип	Назначение
logic	SummatorLogic	Класс, отвечающий за функциональную часть сумматора
group	QGraphicsItemGroup*	Группа элементов полубитного сумматора
textItems_array	std::vector<QGraphicsTextItem*>	Массив с текстовыми элементами для вывода данных
Метод		**Описание**
HalfBitSummator()		Конструктор класса
~HalfBitSummator()		Деструктор класса
virtual void draw()		Функция, отрисовывающая схему полубитного сумматора
void drawInversion(const QPointF &coords)		Отрисовывает элемент операции «НЕ»
void drawAnd(const QPointF &coords)		Отрисовывает элемент операции «И»
void drawOr(const QPointF &coords)		Отрисовывает элемент операции «ИЛИ»

Продолжение таблицы 2.9

Метод	Описание
virtual void PrepareInputs()	Подготовка текстовых элементов (инициализация координат)
void calculateAndDraw(bool x, bool y)	Функция, отрисовывающая входные, промежуточные и выходные данные
char boolToChar(bool x)	Функция, принимающая в качестве параметра булеву функцию и возвращая символ 0 или 1 в зависимости от параметра.

В таблице 2.10 приведено описание класса SummatorLogic, реализующего функциональную часть сумматора и используемого в классе HalfBitSummator (заголовочный файл summatorlogic.h).

Таблица 2.10 – Описание класса SummatorLogic

Метод	Описание
SummatorLogic()	Конструктор класса
~SummatorLogic()	Деструктор класса
HalfBitRes calculateHalfBitFull(bool x, bool y)	Функция, возвращая результат вычисления полубитного сумматора в виде структуры HalfBitRes (выходные и промежуточные данные)
std::pair<bool,bool> calculateHalfBit(bool x, bool y)	Функция, возвращающая результат вычисления полубитного сумматора в виде пары булевых переменных (выходные данные)
TwoBitRes calculateTwoBit(bool a, bool b, bool c, bool d)	Функция, возвращающая результат вычисления двубитного сумматора в виде структуры TwoBitRes (выходные и промежуточные данные)

В таблице 2.11 приведено описание вложенной структуры HalfBitRes в классе SummatorLogic, которое является результатом вычислений полубитного сумматора.

Таблица 2.11 – Описание структуры HalfBitRes

Поле	**Тип**	**Назначение**
u	bool	Первая цифра выходного двоичного двузначного числа
v	bool	Вторая цифра выходного двоичного двузначного числа
oper1	bool	Результат промежуточной операции oper1
oper2	bool	Результат промежуточной операции oper2

В таблице 2.12 приведено описание вложенной структуры TwoBitRes в классе SummatorLogic, которое является результатом вычислений двубитного сумматора.

Таблица 2.12 – Описание структуры TwoBitRes

Поле	**Тип**	**Назначение**
e	bool	Первая цифра выходного двоичного трехзначного числа
f	bool	Вторая цифра выходного двоичного трехзначного числа
g	bool	Третья цифра выходного двоичного трехзначного числа
sum1	std::pair<bool,bool>	Промежуточный результат вычисления 1-ого полубитного сумматора в схеме двубитного сумматора
sum2	std::pair<bool,bool>	Промежуточный результат вычисления 2-ого полубитного сумматора
sum3	std::pair<bool,bool>	Промежуточный результат вычисления 3-ого полубитного сумматора

В таблице 2.13 приведено описание класса TwoBitSummator, который является наследником класса HalfBitSummator, демонстрирующий двубитный сумматор и используемый в классе PictureBox (заголовочный файл twobitsummator.h).

Таблица 2.13 – Описание класса TwoBitSummator

Поле	**Тип**	**Назначение**
point_a	QPointF*	Координаты для вывода входного значения a

Продолжение таблицы 2.13

Поле	**Тип**	**Назначение**
point_b	QPointF*	Координаты для вывода входного значения b
point_c	QPointF*	Координаты для вывода входного значения c
point_d	QPointF*	Координаты для вывода входного значения d
point_e	QPointF*	Координаты для выходного значения e
point_f	QPointF*	Координаты для выходного значения f
point_g	QPointF*	Координаты для выходного значения g
point_sum1_centre	QPointF*	Координаты для промежуточного значения 1-ого полубитного сумматора
point_sum3_centre	QPointF*	Координаты для промежуточного значения 3-ого полубитного сумматора
Метод		**Описание**
TwoBitSummator()		Конструктор класса
~TwoBitSummator()		Деструктор класса
virtual void draw()		Функция, отрисовывающая схему двубитного сумматора
virtual void PrepareInputs()		Подготовка текстовых элементов (инициализация координат)
void calculateAndDraw(bool x, bool y)		Функция, отрисовывающая входные, промежуточные и выходные данные
void drawBox(const QPointF& coords)		Функция, отрисовывающая прямоугольник (который в схеме обозначает полубитный сумматор) по заданной координате

В таблице 2.14 приведено описание класса Teoria, реализующего модуль теории и используемое в классе-контроллере MainWindow (заголовочный файл teoria.h).

Таблица 2.14 – Описание класса Teoria

Поле	**Тип**	**Назначение**
ui	Ui::Teoria*	Поле, необходимое для работы ui форм в Qt
Метод		**Описание**
Teoria(QWidget *parent = nullptr)		Конструктор класса
~Teoria()		Деструктор класса
void return_to_menu()		Сигнал о возврате из модуля теории в главное меню

Продолжение таблицы 2.14

Метод	Описание
void on_textBrowser_anchorClicked(const QUrl &arg1)	Метод, обрабатывающий нажатия на ссылки в основном тексте, для перемещения текущего отображаемого раздела в части окна со словарём
void on_pushButton_clicked()	Нажатие на кнопку «Вернуться в главное меню» (вызывает сигнал return_to_menu)

В таблице 2.15 приведено описание класса loader, реализующего загрузку текста из файла. Статитеческий метод данного класса используется в классе Test.

Таблица 2.15 – Описание класса loader

Поле	Тип	Назначение
-	-	-
Метод		**Описание**
loader()		Конструктор класса
static std::string getText_from_file(const std::string &filename)		Получить текст в виде строки из файла с названием filename, если файл не существует, то вернуть пустую строку

В таблице 2.16 приведено описание вложенного перечисления quest_mode (в классе Test), в котором перечислены все типы вопросов (заголовочный файл test.h).

Таблица 2.16 – Описание перечисления quest_mode

Название	Номер	Назначение
closed_answer	0	Закрытый тип вопроса
open_answer	1	Открытый тип вопроса с кратким ответом

В таблице 2.17 приведено описание вложенной структуры user_answers (в классе Test), содержащей ответы пользователя (заголовочный файл test.h).

Таблица 2.17 – Описание структуры user_answers

Поле	Тип	Назначение
num	std::vector<int>	Ответы пользователя на вопросы закрытого типа в виде чисел
str	std::vector<QString>	Ответы пользователя на вопросы открытого типа с кратким ответом в виде строк

В таблице 2.18 приведено описание вложенной структуры Answer (в классе Test), являющейся вариантом ответа на вопрос (заголовочный файл test.h).

Таблица 2.18 – Описание структуры Answer

Поле	**Тип**	**Назначение**
text	QString	Текст дистрактора
right	bool	Правильный ли ответ

В таблице 2.19 приведено описание вложенной структуры Answer (в классе Test), являющейся дистрактором вопроса (заголовочный файл test.h).

Таблица 2.19 – Описание структуры Question

Поле	**Тип**	**Назначение**
text	QString	Текст дистрактора
mode	int	Тип вопроса
arr	std::array<Answer, 4>	4 варианта ответа

В таблице 2.20 приведено описание класса Test, реализующего модуль тестирования и используемое в классе-контроллере MainWindow (композиция) (заголовочный файл test.h).

Таблица 2.20 – Описание класса Test

Поле	**Тип**	**Назначение**
all_text	std::string	Содержимое текстового файла с базой вопросов, полученное с помощью статического метода класса loader
Questions	std::vector<Question>	Сформированные вопросы
user_choices	user_answers	Ответы пользователя
N	size_t	Количество вопросов в тесте
i_step	size_t	Номер текущего отображаемого вопроса
Метод		**Описание**
Test(QWidget *parent = nullptr)		Конструктор класса
~Test()		Деструктор класса
void return_to_menu()		Сигнал о возврате из модуля тестирования в главное меню
void on_next_clicked()		Нажатие на кнопку «>» (переход к следующему вопросу)
void on_prev_clicked()		Нажатие на кнопку «<» (переход к предыдущему вопросу)

Продолжение таблицы 2.20

Метод	**Описание**
void on_Answer1_checkBox_clicked()	Выбор 1 варианта ответа
void on_Answer2_checkBox_clicked()	Выбор 2 варианта ответа
void on_Answer3_checkBox_clicked()	Выбор 3 варианта ответа
void on_Answer4_checkBox_clicked()	Выбор 4 варианта ответа
void on_finish_clicked();	Нажатие на кнопку «Завершить»
void on_answer_textEdited(const QString &arg1)	Обработка ввода краткого ответа
void on_spinBox_valueChanged(int arg1)	Ввод количества вопросов в тесте
void on_pushButton_start_clicked()	Нажатие на кнопку «Начать тестирования»
void on_return_to_menu_button_clicked()	Нажатие на кнопку «Вернуться в главное меню» (вызывает сигнал return_to_menu)
void on_pushButton_results_clicked()	Нажатие на кнопку «Посмотреть статистику»

В таблице 2.21 приведено описание класса MainWindow, главного интерфейсного класса-контроллера (заголовочный файл mainwindow.h). Классы Demo, Test и Teoria являются частью MainWindow (композиция), что не отображено в таблице 2.15, поскольку они были привязаны к классу MainWindow в дизайнере форм Qt, который автоматически генерирует файл ui_mainwindow.h, в котором уже и содержатся указатели на классы Demo, Test и Teoria.

Таблица 2.21 – Описание класса MainWindow

Поле	**Тип**	**Назначение**
ui	Ui::MainWindow*	Поле, необходимое для работы ui форм в Qt
Метод		**Описание**
MainWindow(QWidget *parent = nullptr)		Конструктор класса
~MainWindow()		Деструктор класса
void on_pushButton_teoria_clicked()		Скрывает все виджеты и делает текущим отображаемым виджетом — виджет с теорией
void on_pushButton_test_clicked()		Скрывает все виджеты и делает текущим отображаемым виджетом — виджет с тестированием

Продолжение таблицы 2.21

Метод	Описание
void on_pushButton_demo_clicked()	Скрывает все виджеты и делает текущим отображаемым виджетом — виджет с демонстрацией
void return_to_menu()	Скрывает все виджеты и делает текущим отображаемым виджетом — виджет с главным меню

2.5 Разработанные меню и интерфейсы

После запуска программы на выполнение появится окно, содержащее главное меню (рис. 2.1), который помимо самого меню содержит информацию о теме курсового проекта и об авторе программы.

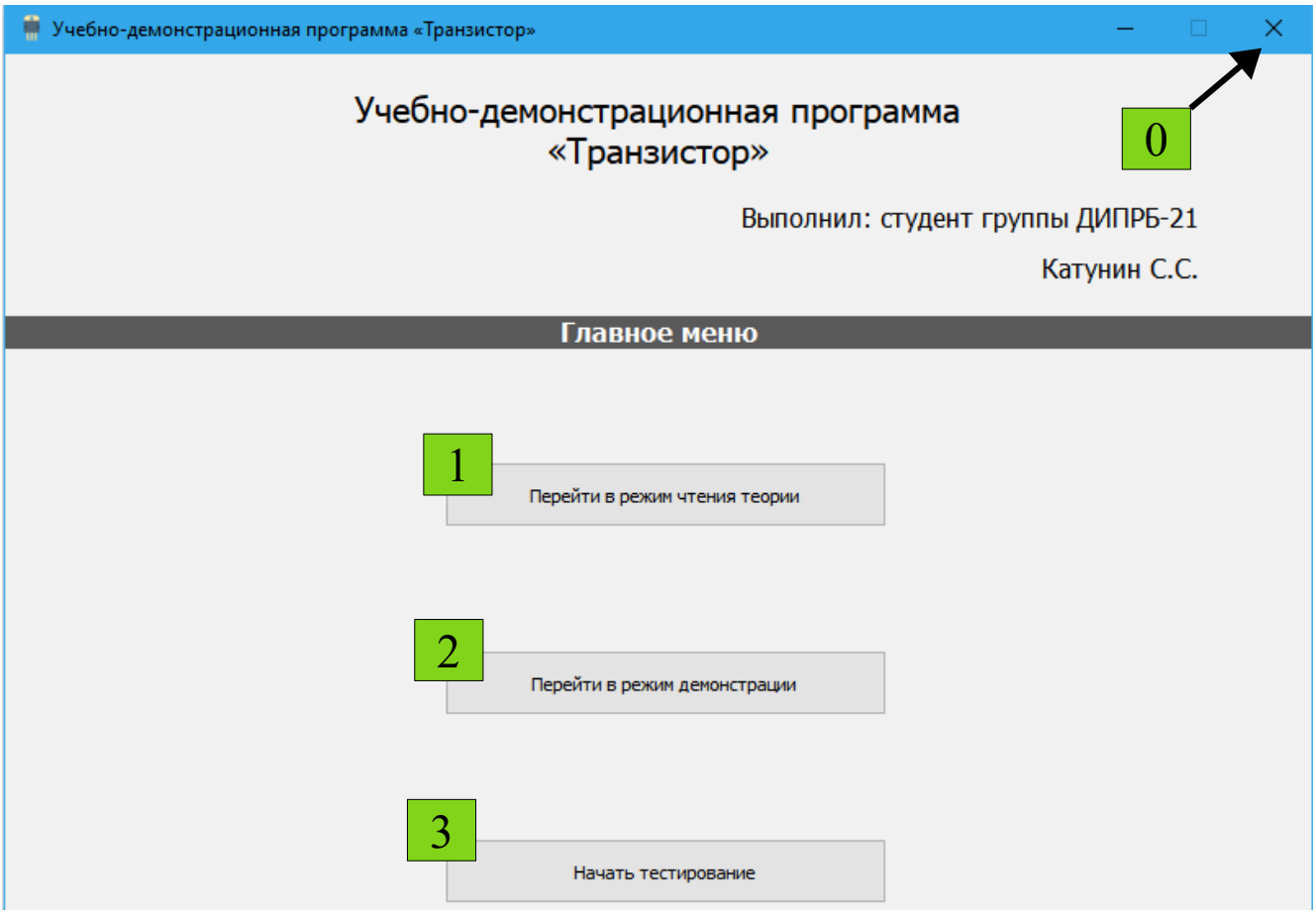

Рисунок 2.1 – Окно программы с главным меню

Кнопка «1» позволяет перейти к режиму чтения теории, кнопка «2» – к режиму демонстрации, кнопка «3» – к режиму тестирования, а кнопка «0» - завершает работу программы.

При нажатии на кнопку «1» главное меню сменится на режим чтения теории (рис. 2.2). В правой части окна можно наблюдать теоретический материал, который может включать рисунки, таблицы, различные стили для текста. В левой части окна представлено содержание, в котором вынесены разделы, которые соответствующим образом выделены в теории. При нажатии на элемент таблицы с содержанием страница автоматически прокрутится до места начала раздела, чтобы пользователь мог ознакомиться с ним подробнее.

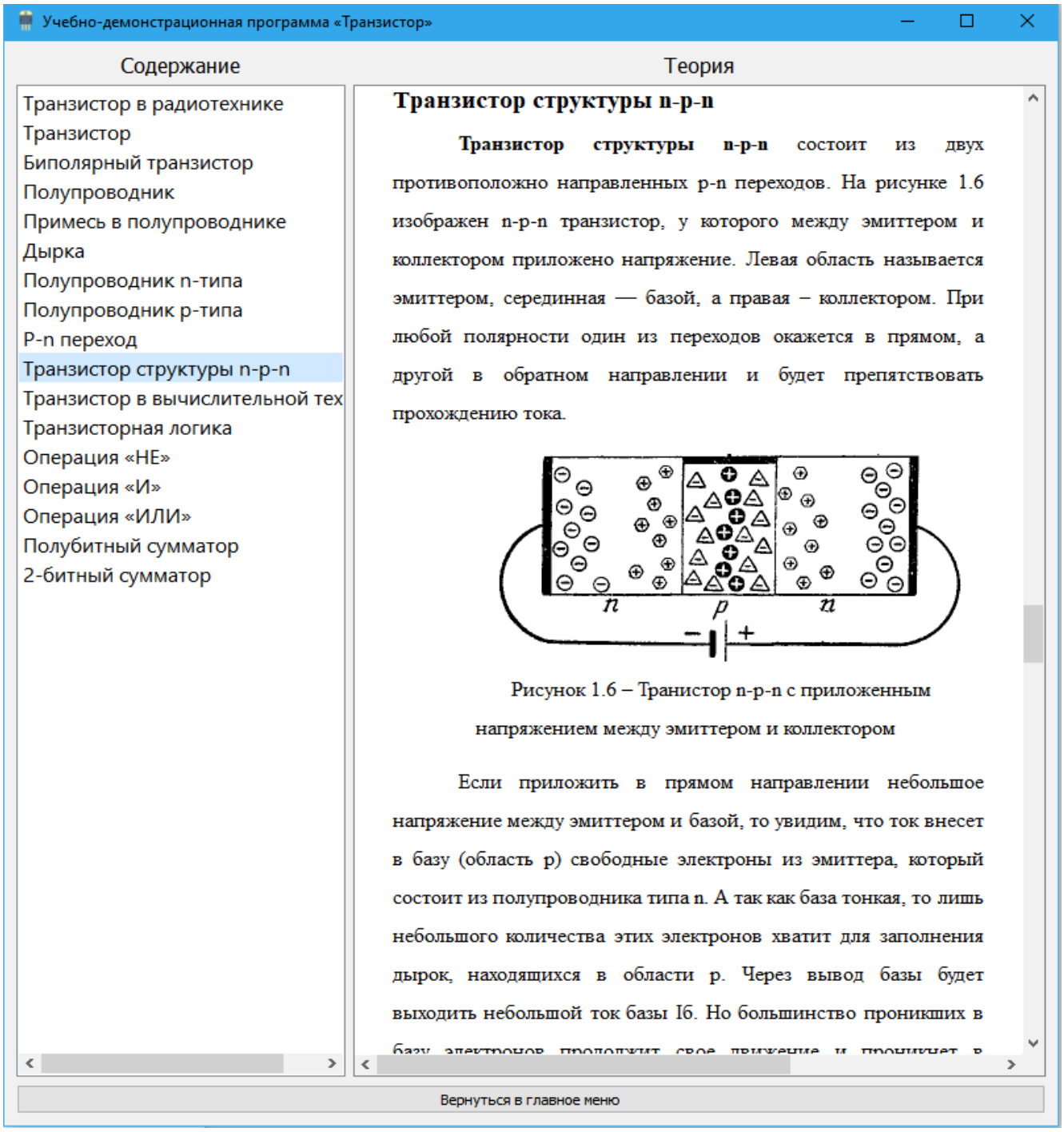

Рисунок 2.2 – Окно с выбранным режимом чтения теории

После нажатия кнопки «4» происходит возврат в главное меню (см. рис. 2.1).

После нажатия кнопки «2» в главном меню осуществляется переход в режим демонстрации (рис. 2.3).

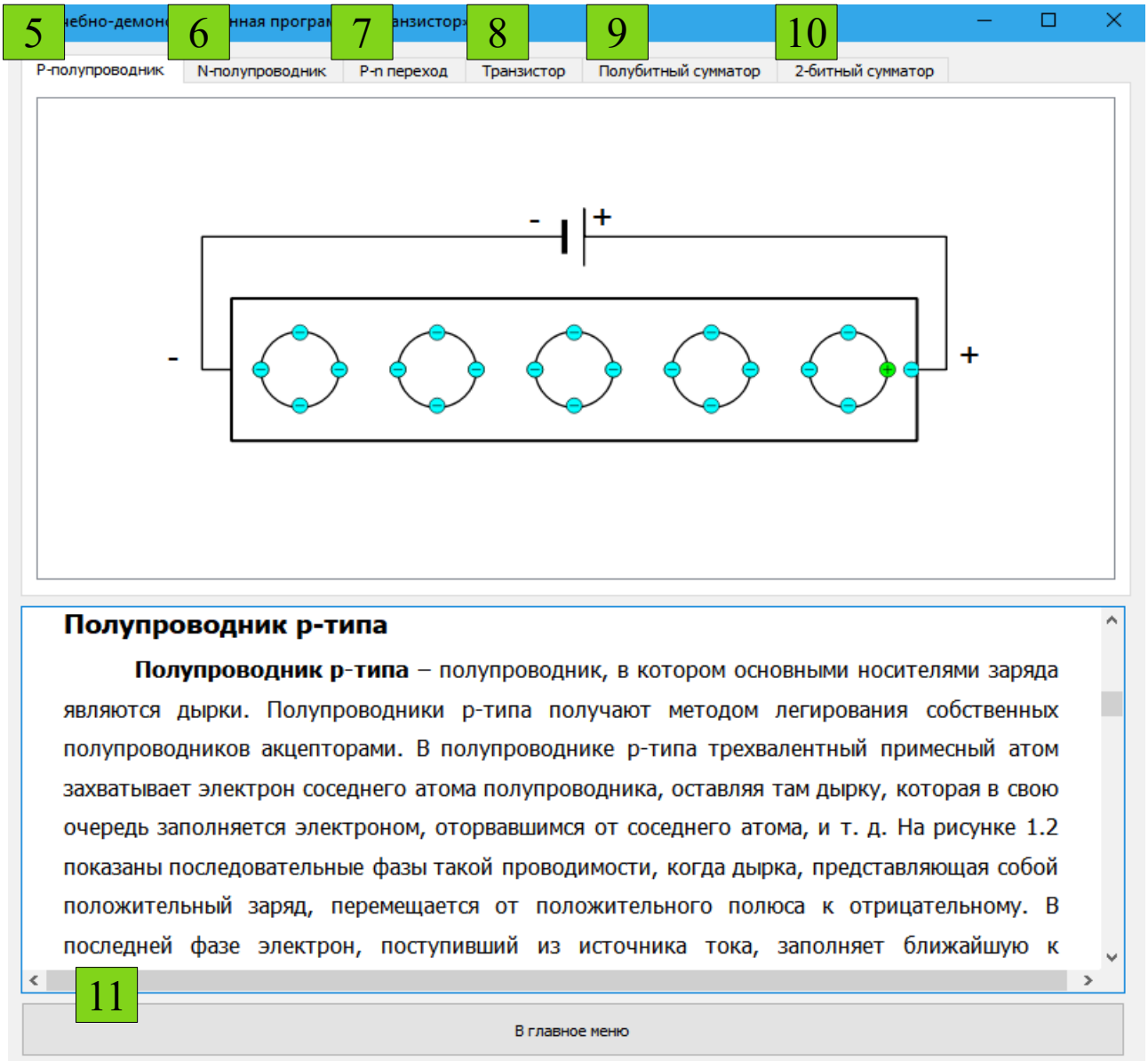

Рисунок 2.3 – Вкладка «Р-полупроводник» в режиме демонстрации

В верхней части программы находится область, в которой визуализируются анимированные физические процессы, а в нижней – располагается теория, с соответствующим теме визуализации разделом. По умолчанию программа встречает пользователя с демонстрацией р-полупроводника. Вкладки «5», «6», «7», «8», «9» и «10» используются для перехода к другим визуализациям при помощи нажатия на вкладку, при этом, в нижней части окна, теоретический материал обновится и станет соответствовать выбранной теме. Поскольку графика является векторной, размеры окна можно изменять (также присутствует возможность развернуть окно на весь экран) с масштабированием содержимого без ухудшения качества визуализации. В нижней части окна представлена кнопка возврата в главное меню программы. (см. рис. 2.1).

При нажатии на вкладку «6» осуществляется переход к визуализации n-полупроводника. (рис. 2.4).

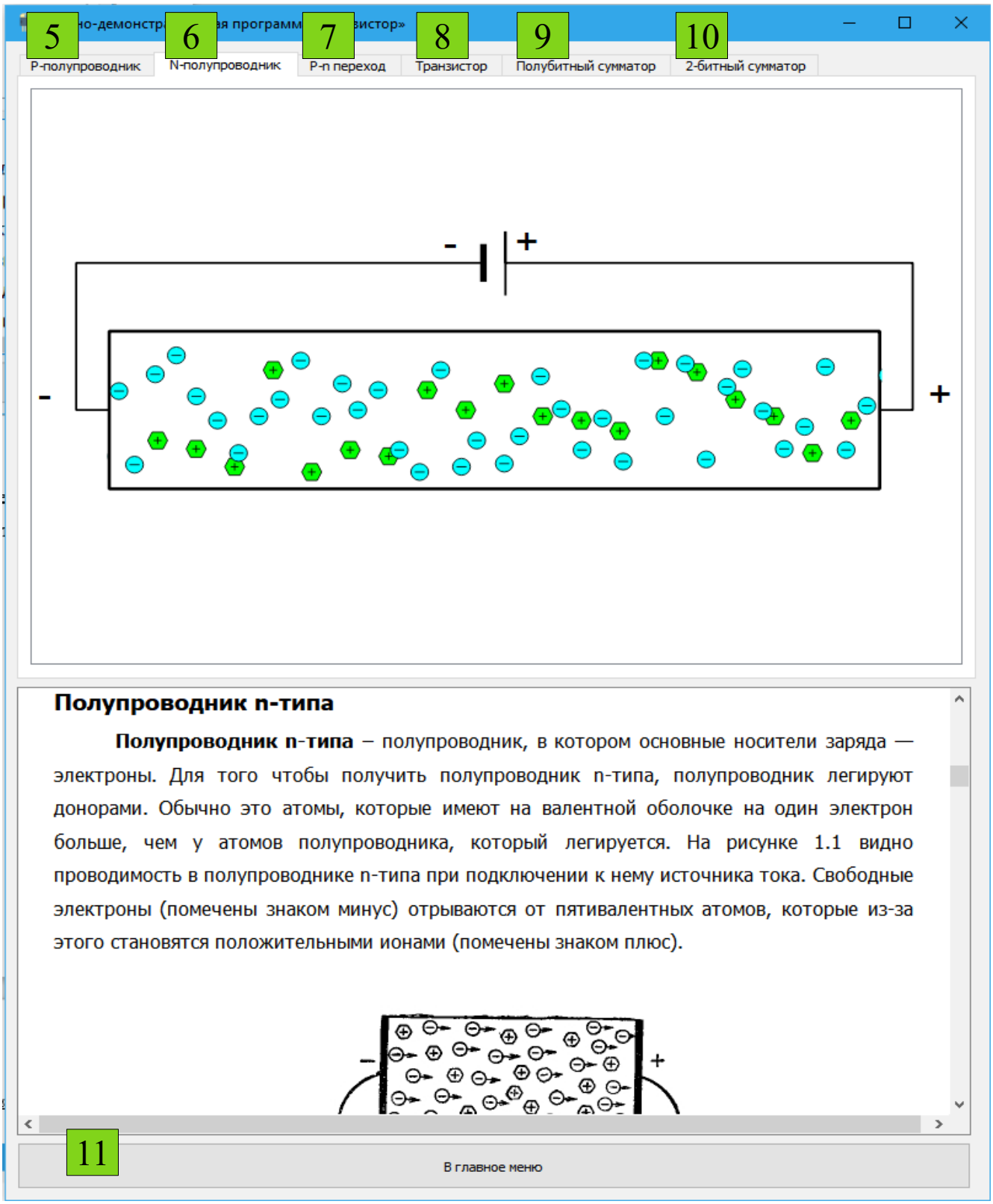

Рисунок 2.4 – Вкладка «N-полупроводник» в режиме демонстрации

При нажатии на вкладку «7» осуществляется переход к визуализации p-n перехода (рис. 2.5).

Рисунок 2.5 – Вкладка «P-n переход» в режиме демонстрации

После нажатия на кнопку «12» происходит обновление визуализации в верхней части окна и прямое подключение источника сменяется обратным (рис. 2.6).

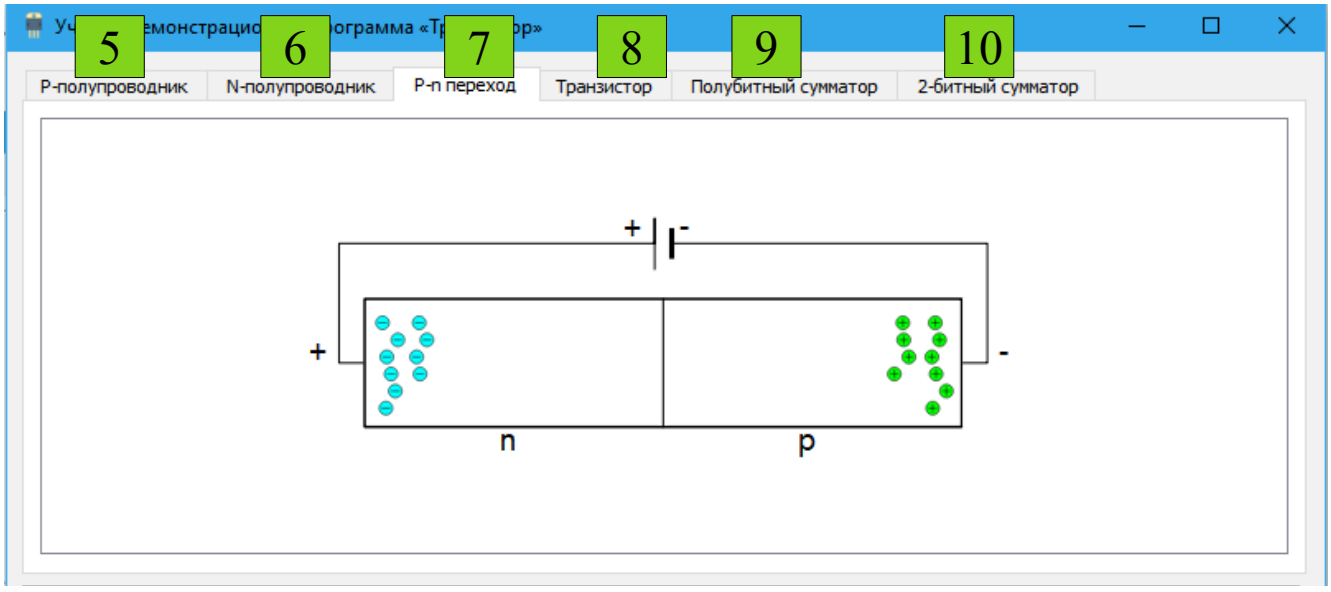

Рисунок 2.6 – p-n переход с обратным подключением

Нажав вкладку «8» осуществляется переход к визуализации n-p-n транзистора (рис. 2.7).

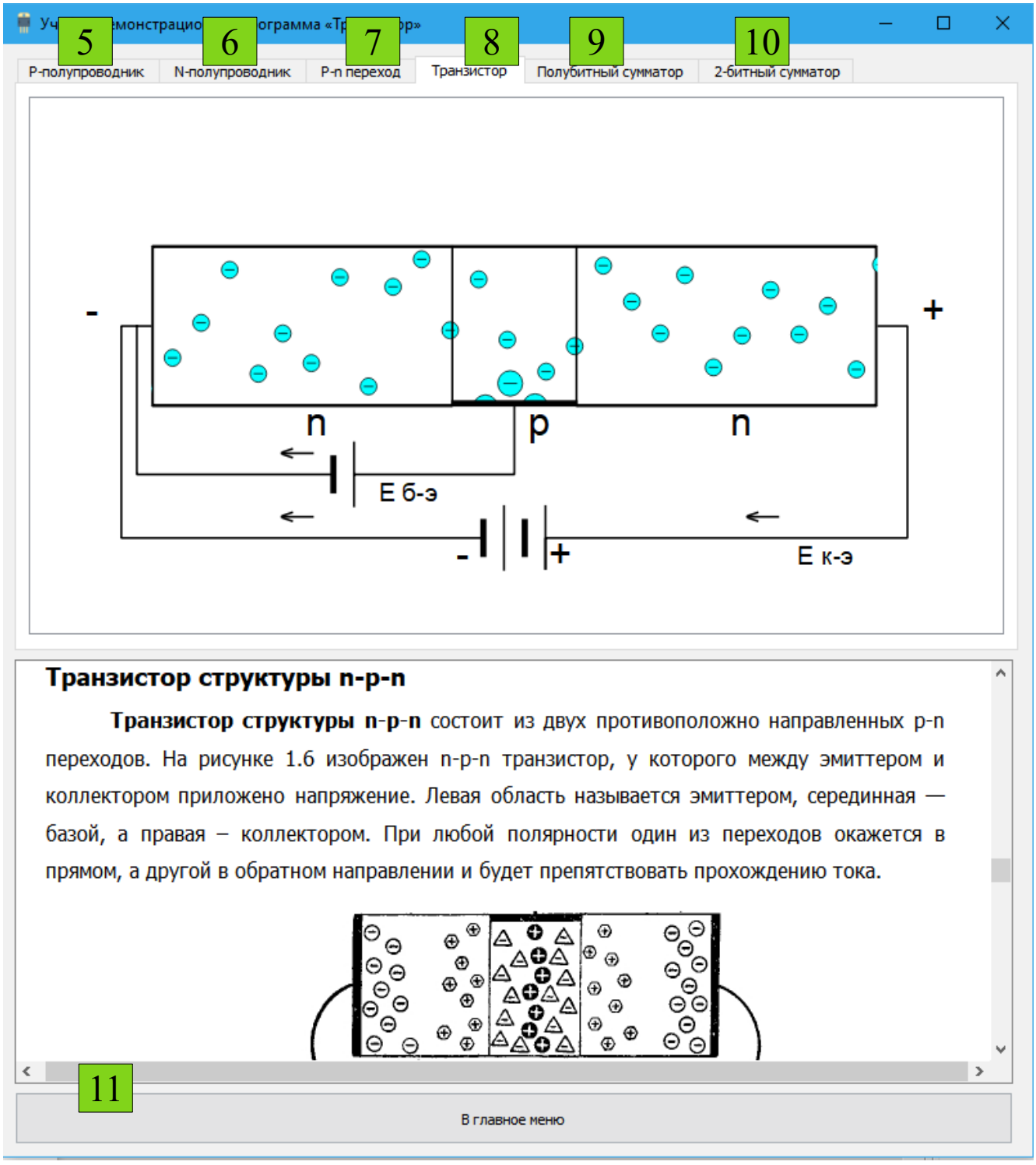

Рисунок 2.7 – Вкладка «Транзистор» в режиме демонстрации

При нажатии на вкладку «9» демонстрируется схема полубитного сумматора, а так же в серединной части окна появляются поля для ввода данных в сумматор «13» и «14» (рис. 2.8).

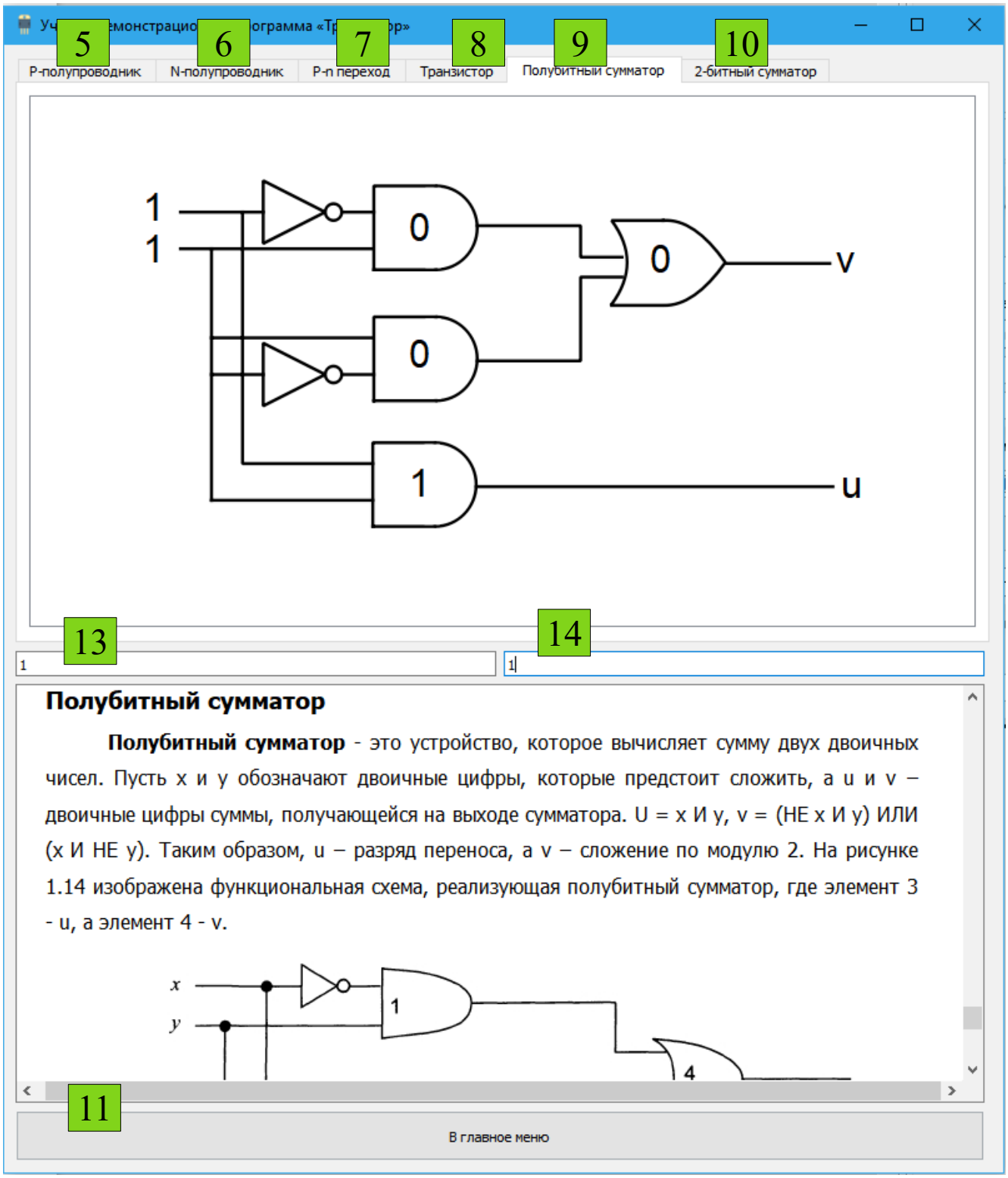

Рисунок 2.8 – Вкладка «Полубитный сумматор» в режиме демонстрации

При нажатии на вкладку «10» пользователю показывается схема двубитного сумматора, в серединной части окна располагаются поля для ввода данных в сумматор «13» и «14» (рис. 2.9).

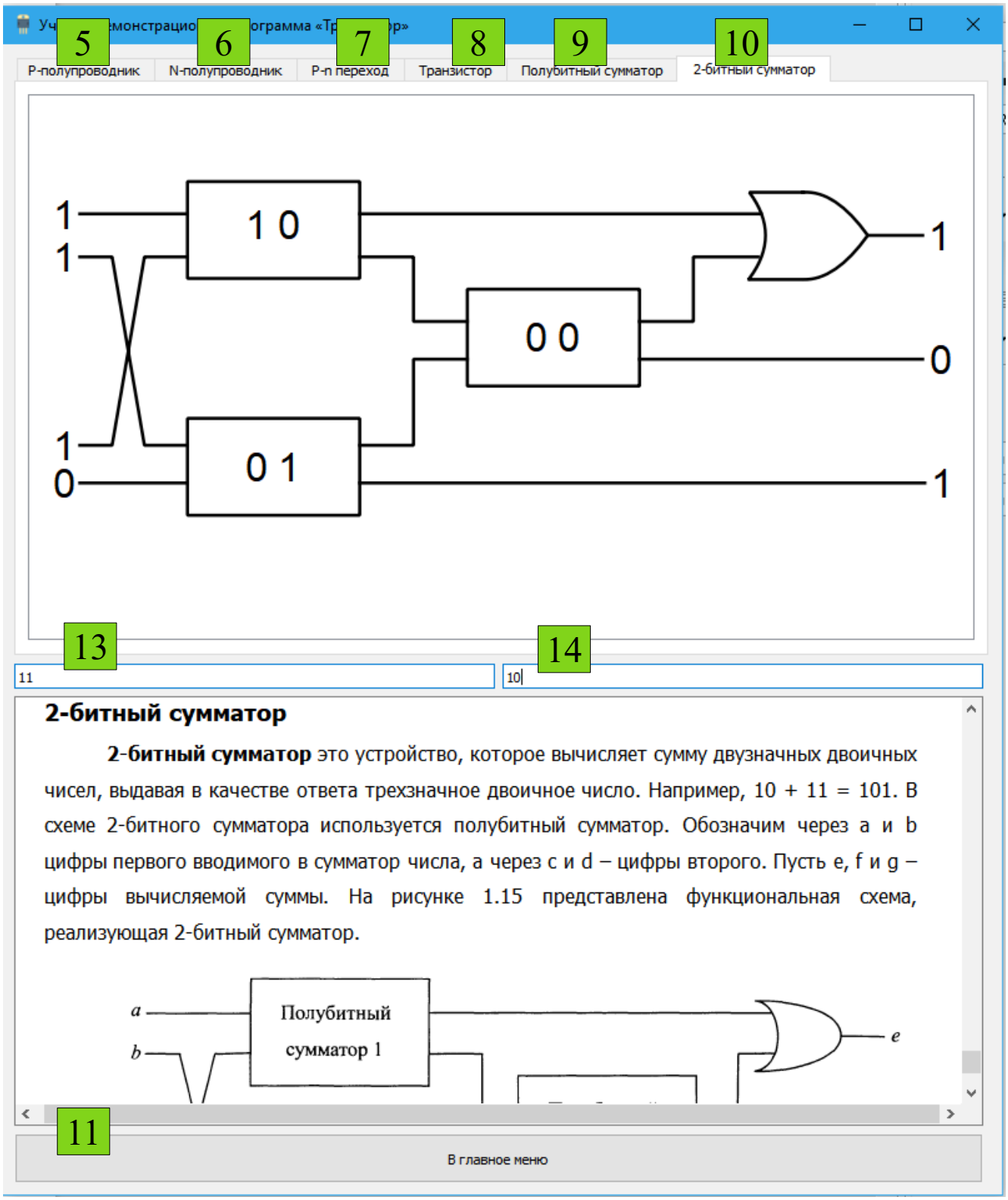

Рисунок 2.9 – Вкладка «Двубитный сумматор» в режиме демонстрации

При нажатии на вкладку «3» в главном меню перед пользователем предстает меню тестирования (рис. 2.10). Числовое поле «15» используется для задания количества вопросов в тесте, текстовое поле «16» – для задания имени тестируемого. Кнопка «17» начинает процесс тестирования, нажава на кнопку «18» можно ознакомиться со статистикой, и с помощью кнопки «19» можно вернуться в главное меню.

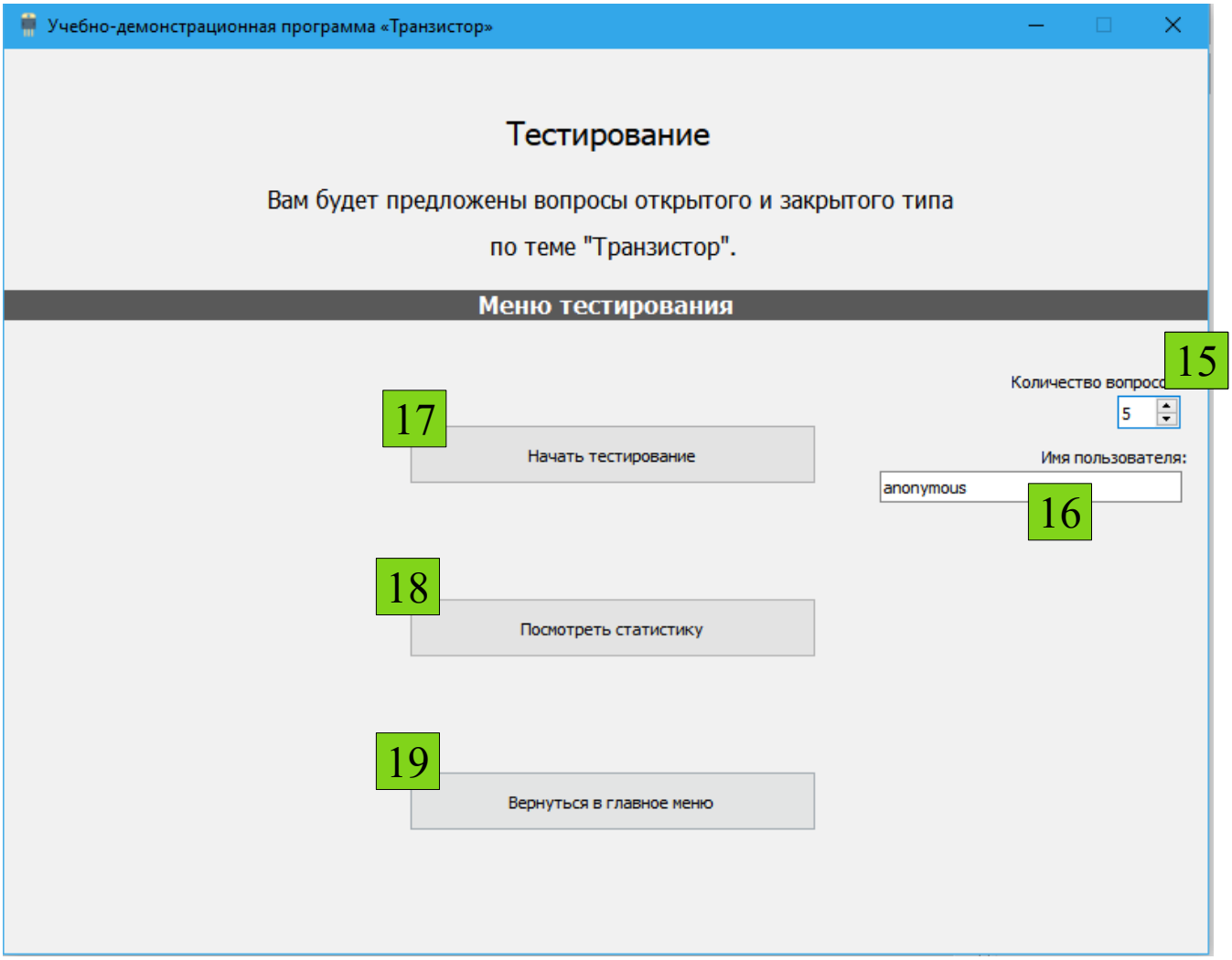

Рисунок 2.10 – Меню тестирования

После нажатия кнопки «17» начинается процесс тестирования, который может включать вопросы закрытого типа с 4-мя вариантами ответа и открытого типа с кратким ответом, который пользователь вводит с клавиатуры.

На рисунке 2.11 отображен вопрос закрытого типа.

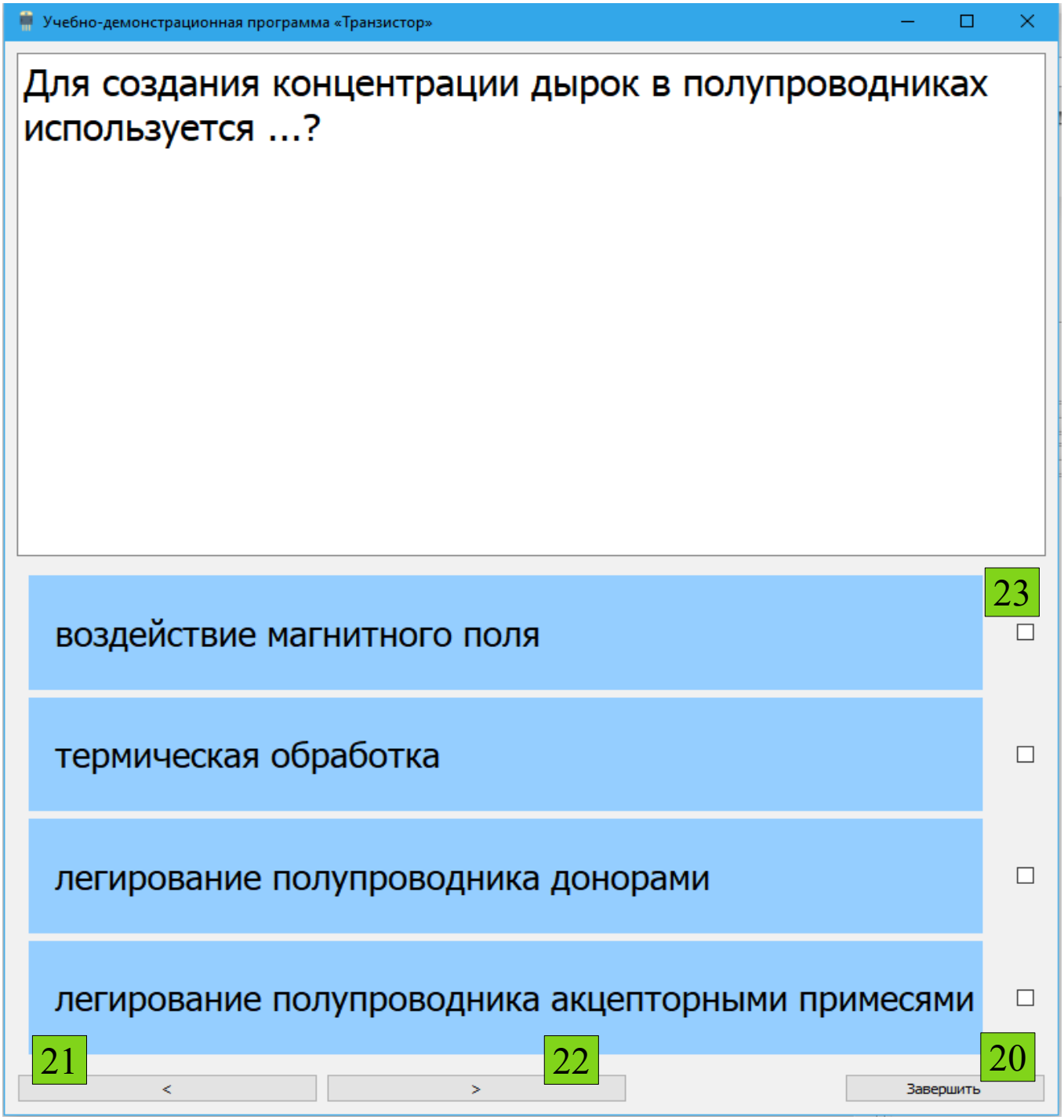

Рисунок 2.11 – Вопрос закрытого типа

С помощью кнопки «20» можно завершить тестирование. Кнопки «21» и «22» используются для навигации, а чекбоксы «23» используются для того, чтобы пользователь мог дать ответ на вопрос.

На рисунке 2.12 представлен вопрос открытого типа.

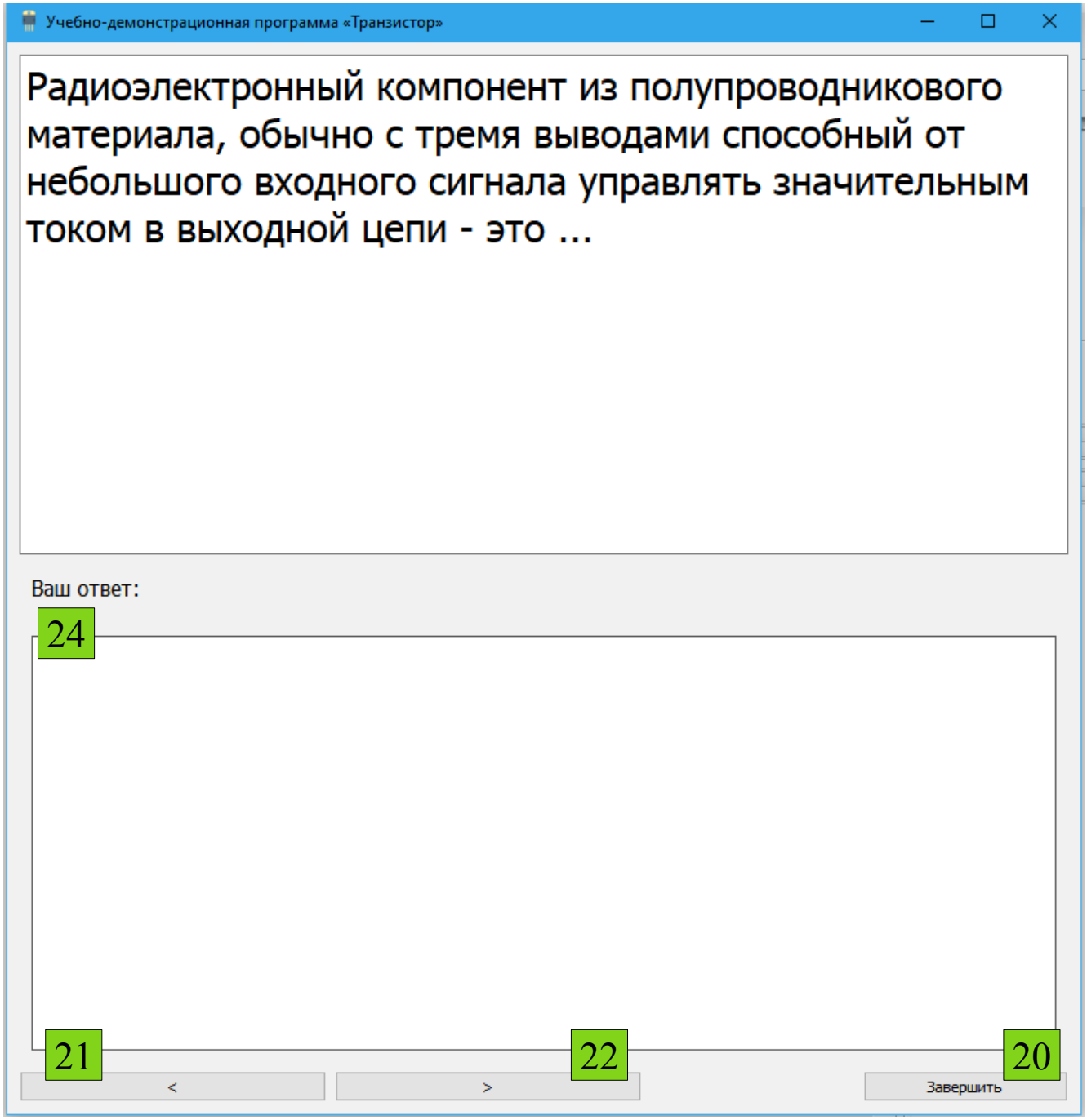

Рисунок 2.12 – Вопрос открытого типа

Текстовое поле «24» используется для того, чтобы пользователь мог дать краткий ответ на вопрос.

При нажатии на кнопку «20» пользователю становятся доступны результаты тестирования (рис. 2.13).

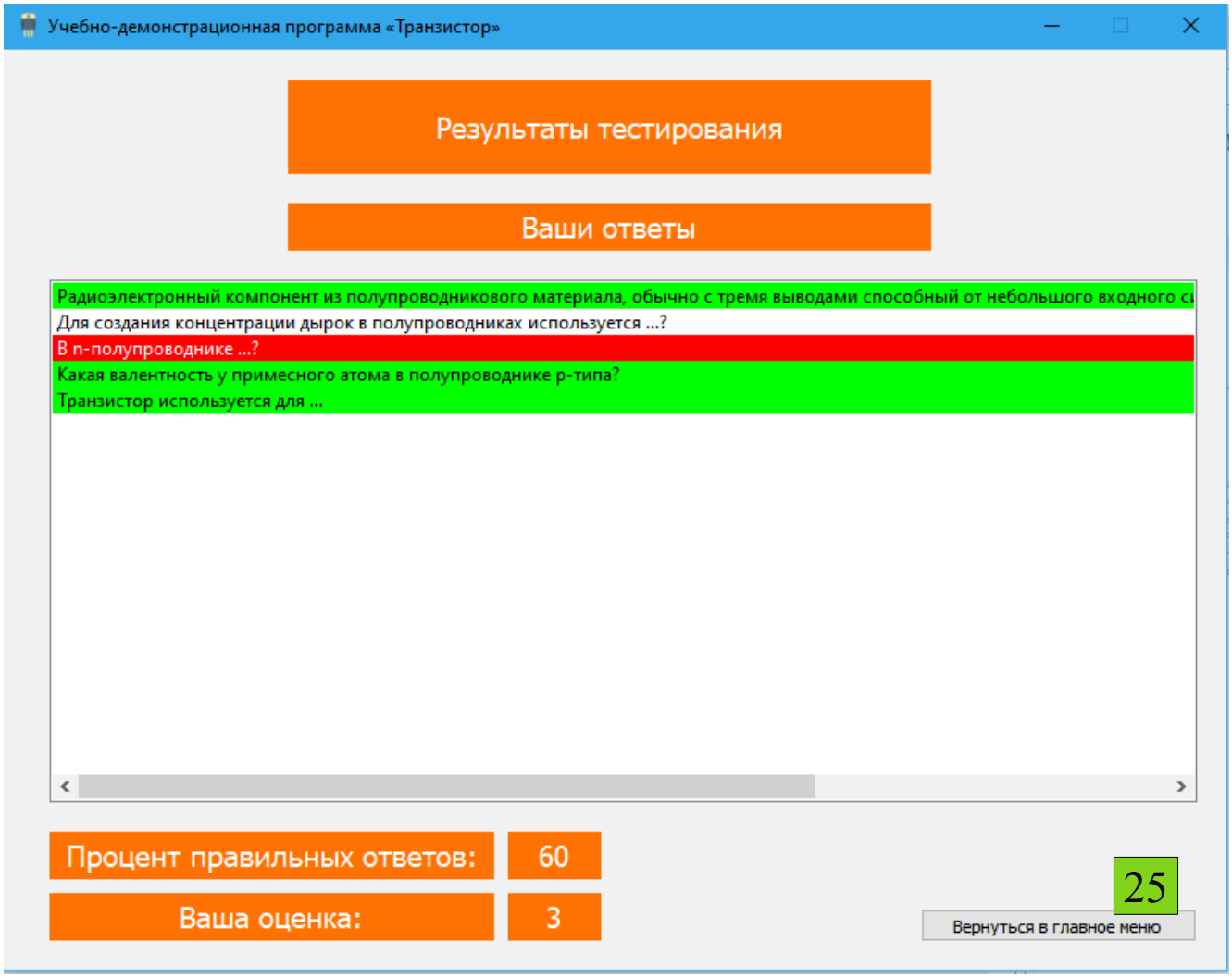

Рисунок 2.13 – Результаты тестирования

Зеленым цветом выделяются те вопросы, на которые был дан правильный ответ, красным выделяются те, на которые был дан неправильный ответ, а в случае, если вопрос был пропущен, то он не выделяется. В данном окне пользователь может ознакомиться с процентом правильных ответов и с оценкой. После нажатия на кнопку «25» происходит возврат в главное меню программы (см. рис. 2.1).

Если пользователь нажмет на кнопку «18» в меню тестирования, ему станет доступен просмотр статистики (рис. 2.14).

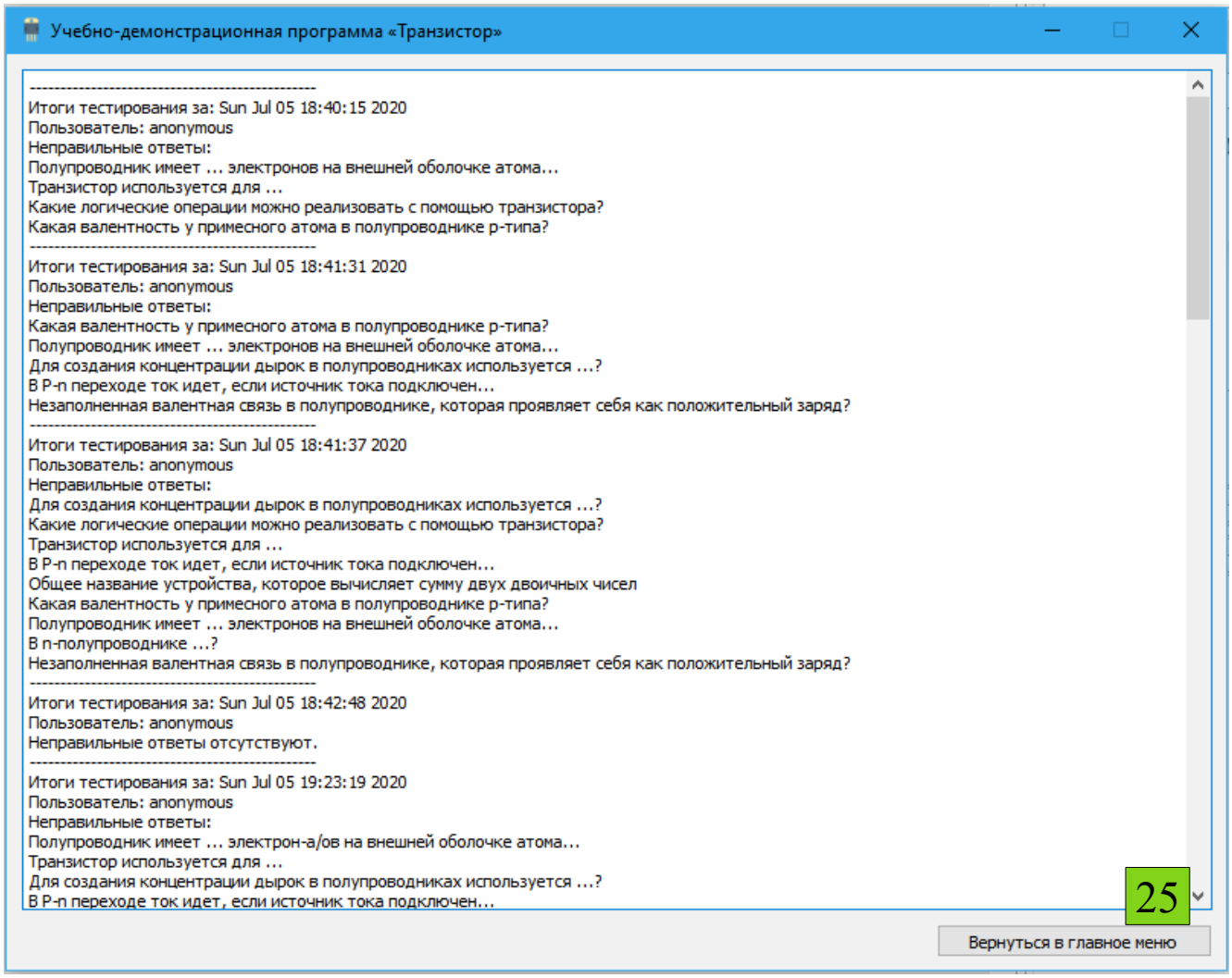

Рисунок 2.14 – Просмотр статистики тестирования

2.6 Сообщения системы

Программа сообщает об ошибках с помощью месседжбоксов. На рисунке 2.15 представлено сообщение об ошибке, возникающее в случае, если программа не может открыть файл с теорией.

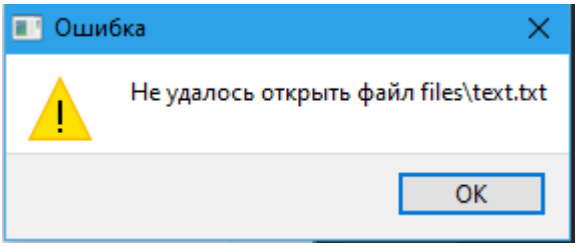

Рисунок 2.15 – Ошибка: не удалось открыть файл с теорией

На рисунке 2.16 представлена сообщение об ошибке, которое возникает, если программа не может открыть файл с базой вопросов.

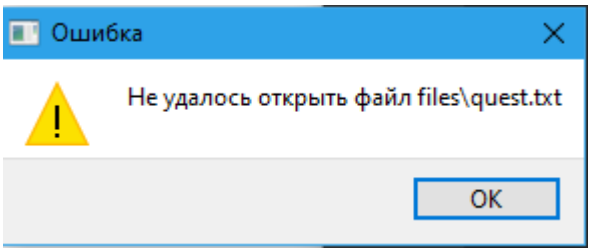

Рисунок 2.16 – Ошибка: не удалось открыть файл с базой вопросов

В случае появления других сообщений следует обратиться к разработчику.

3 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

3.1 Проверка работоспособности режима чтения теории

1. Запустить программу на выполнение. Появится главное меню. (см. рис. 2.1).
2. Нажать кнопку «Перейти в режим чтения теории», убедиться, что программа перешла в режим чтения теории (см. рис.2.2).
3. Нажать кнопку «Вернуться в главное меню», убедиться, что произошел возврат в главное меню (см. рис. 2.1).
4. Нажать кнопку «X (закрыть окно)» в правом верхнем углу окна, убедиться, что программа завершила свою работу.

3.2 Проверка работоспособности режима демонстрации

1. Запустить программу на выполнение. Появится главное меню. (см. рис. 2.1).
2. Нажать кнопку «Перейти в режим демонстрации», убедиться, что осуществился переход в режим демонстрации (см. рис. 2.3).
3. Удостовериться, что анимация работает корректно и соответствует теме указанной во вкладке.
4. Удостовериться, что теоретический материал в нижней половине экрана соответствует теме указанной во вкладке.
5. Нажать на вкладку «N-полупроводник», убедиться, что произошел переход к визуализации «N-полупроводник» (см. рис. 2.4).
6. Повторить пункты 3 и 4.
7. Нажать на вкладку «P-n переход», убедиться, что произошел переход к визуализации «P-n переход» (см. рис. 2.5).
8. Повторить пункты 3 и 4.
9. Нажать на кнопку «Изменить полярность подключения», убедиться, что визуализация обновилась и прямое подключение источника тока сменилось обратным (см. рис. 2.6).
10. Нажать на вкладку «Транзистор», убедиться, что произошел переход к визуализации «Транзистор» (см. рис. 2.7).
11. Повторить пункты 3 и 4.
12. Нажать на вкладку «Полубитный сумматор», убедиться, что произошел переход к визуализации «Полубитный сумматор» (см. рис. 2.8).
13. Повторить пункт 4.

14. Осуществить ввод в текстовые поля «13» и «14» и удостовериться, что введенные данные в поля отобразились на экране, а также что промежуточные и выходные данные правильные.
15. Попробовать ввести в поля символы отличные от двоичных цифр «0» и «1» и убедиться, что в поле невозможно ввести другие символы.
16. Нажать на вкладку «2-битный сумматор», убедиться, что произошел переход к визуализации «Двубитный сумматор» (см. рис. 2.9).
17. Повторить пункт 4, 14 и 15.
18. Нажать кнопку «В главное меню», убедиться, что произошел возврат в главное меню (см. рис. 2.1).
19. Нажать кнопку «X (закрыть окно)», убедиться, что программа завершила свою работу.

3.3 Проверка работоспособности режима тестирования

1. Запустить программу на выполнение. Появится главное меню. (см. рис. 2.1).
2. Нажать кнопку «Начать тестирование», убедиться, что на экране появилось меню тестирования (см. рис. 2.10).
3. Нажать кнопку «Начать тестирование», убедиться, что началось тестирование (см. рис. 2.11 или рис. 2.12)
4. Нажать кнопку «>», убедиться, что был совершен переход к следующему вопросу (см. рис. 2.11 или рис. 2.12)
5. Нажать кнопку «<», убедиться, что был совершен переход к предыдущему вопросу (см. рис. 2.11 или рис. 2.12)
6. Отметить чекбокс «23» в случае вопроса закрытого типа или ввести ответ в поле «24» (см. рис. 2.11 или рис. 2.12).
7. Нажать кнопку «Завершить», убедиться, что тестирование было завершено и был совершен переход к результатам тестирования, которые соответствуют выбранным ответам на вопросы (см. рис. 2.13).
8. Нажать кнопку «Вернуться в главное меню», убедиться, что был совершен возврат в главное меню программы (см. рис. 2.1).
9. Повторять пункты 2, 3, 4, 5, 6, 7 несколько раз, таким образом, чтобы получить оценки 2, 3, 4 и 5 в результатах тестирования (см. рис. 2.13).
10. Повторить пункт 2, после чего нажать кнопку «Посмотреть статистику», убедиться, что была открыта статистика тестирования.

11. Нажать кнопку «Вернуться в главное меню», убедиться, что был совершен возврат в главное меню программы (см. рис. 2.1).

12. Нажать кнопку «X (закрыть окно)», убедиться, что программа завершила свою работу.

ЗАКЛЮЧЕНИЕ

В результате курсового проектирования разработана учебно-демонстрационная программа «Транзистор». Программа предоставляет теоретический материал по теме «Транзистор», демонстрирует принцип работы транзистора в радио- и вычислительной технике, позволяет пройти тестирование, по окончании которого выводятся результаты тестирования.

Программа отвечает поставленным требованиям и может быть использована для обучения студентов и школьников старших классов, обучающихся физике.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Вирт. Н. Алгоритмы и структуры данных. Новая версия для Оберона + CD / Пер. с англ. Ткачев Ф. В. - М.: ДМК Пресс, 2010. – 272 с.: ил.
2. Кормен, Томас Х. и др. Алгоритмы: построение и анализ, 3-е изд. : Пер. с англ. – М.: ООО „И. Д. Вильямс“, 2013. – 1328 с. : ил. – Парал. тит. Англ.
3. Шлее М. Qt 5.10. Профессиональное программирование на C++. – СПб.: БХВ-Петербург, 2018. – 1072 с.: ил. – (В подлиннике)
4. Саттер Г., Александреску А. Стандарты программирования на C++.: Пер. с англ. — М.: Издательский дом "Вильямс", 2005.— 224 с. :ил.— Парал. тит. англ.
5. Лаптев В. В. C++. Объектно-ориентированное программирование: Учебное пособие. – СПб.: Питер, 2008. – 464 с.: ил. – (Серия «Учебное пособие»).
6. Qt Documentation – [Электронный ресурс] режим доступа: https://doc.qt.io (27.10.2019)

ПРИЛОЖЕНИЕ 1

ТЕХНИЧЕСКОЕ ЗАДАНИЕ
на выполнение курсового проекта

по дисциплине «Компьютерное моделирование»

Направление 090304 – Программная инженерия

Исполнитель: обучающийся гр. ДИПРб21 **Катунин С.С.**

Тема: Учебно-демонстрационная программа «Транзистор»

1 Назначение, цели и задачи разработки

Цель разработки – автоматизация обучения и контроля знаний по теме «Транзистор».

Назначение разработки:

- повышение качества знаний студентов, изучающих физику;
- снижение нагрузки на преподавателя.

Основные задачи, решаемые разработчиком в процессе выполнения курсового проекта:

- анализ предметной области;
- разработка программного продукта в соответствии с требованиями;
- документирование проекта в соответствии с установленными требованиями.

2 Характер разработки: прикладная квалификационная работа.

3 Основания для разработки

- Учебный план направления 09.03.04 «Программная инженерия» 2018 года набора.
- Рабочая программа дисциплины «Компьютерное моделирование».
- Распоряжение по кафедре АСОИУ №____ от «___»_____ 201_г.

4 Плановые сроки выполнения – осенний семестр 2019/20 учебного года:

Начало «_15_» _марта_ 2020 г.

Окончание «_22_» _июня_ 2020 г.

5 Требования к проектируемой системе

5.1 Требования к функциональным характеристикам

Проектируемая система представляет собой графическое приложение и должна обеспечивать выполнение следующих основных функций:

- Предоставление пользователю теоретического материала по теме «Транзистор»
  - ✓ теоретический материал организован в виде гипертекста (HTML);
  - ✓ теоретический материал может включать: текст, таблицы, рисунки;
  - ✓ теоретический материал должен сопровождаться содержанием для быстрого перехода к разделам;
- Демонстрация принципа работы транзистора:

- ✓ демонстрация процесса p-n перехода с возможностью изменять полярность подключаемого источника тока;
- ✓ демонстрация движения электронов в n-полупроводнике;
- ✓ демонстрация движения электронов и дырок в p-полупроводнике;
- ✓ демонстрация принципа работы транзистора в режиме ключа;
- ✓ демонстрация принципа работы транзистора в вычислительной технике, а именно демонстрация работы полубитного и 2-битного сумматора на основе транзистора и транзисторной логики (логические операции И, ИЛИ, НЕ) с возможностью ввода входных данных и с выводом промежуточных вычислений логических операций.

- Тестирование пользователя по теме «Транзистор»:
  - ✓ вопросы закрытого типа с выбором одного правильного ответа, количество предлагаемых вариантов ответа – 4, при показе пользователю вопроса варианты ответа должны быть перемешаны;
  - ✓ вопросы открытого типа с кратким ответом;
  - ✓ при тестировании пользователю предлагается то количество вопросов, которое было указано перед тестированием, общее количество вопросов в базе не менее 10, база представляет собой текстовый файл;
  - ✓ формат файла и вопросы разрабатываются исполнителем самостоятельно;
  - ✓ перед тестированием пользователю должна быть предоставлена инструкция о проведении теста;
  - ✓ программа должна подсчитывать процент правильных ответов;
  - ✓ если ответ правильный, то к количеству правильных ответов добавляется 1 балл;
  - ✓ пользователь может пропустить вопрос с возможностью вернуться к нему;
  - ✓ по завершении тестирования общий итог выводится как на экран, так и в текстовый файл со статистикой (с указанием данных о тестируемом, даты и времени прохождении теста, текста вопросов, в которых была допущена ошибка);
  - ✓ тестирование считается успешным, если процент правильных ответов не менее 60%.
- **Интерфейс программы:** текст русский, шрифты кириллический и латинский, заголовки, термины и другая важная информация выделены цветом.

Система имеет функциональные ограничения:

- программа не предоставляет возможность изменять схему электрической цепи или схему сумматора;
- программа не предоставляет возможность шифровать / расшифровывать текстовые файлы с базой вопросов и с теорией;

- программа не должна обеспечивать редактирование статистики результатов тестирования.

5.2 Требования к эксплуатационным характеристикам

Программа не должна аварийно завершаться при любых действиях пользователя

Время реакции программы на действия пользователя не должно превышать 10 секунд.

5.3 Требования к программному обеспечению:

Средства разработки: интегрированная среда Qt Creator (версия Qt 5.6), язык C++ (стандарт C++ 11 и более поздние).

Операционная система: Windows XP (x86) с пакетом обновления 3 (SP3) или более поздние.

5.4 Требования к аппаратному обеспечению:

Рекомендуемая конфигурация:

- Intel-совместимый процессор с частотой не менее 1,6 ГГц;
- не менее 512 МБ ОЗУ;
- не менее 50 МБ свободного места на диске;
- Дисковод CD-ROM/DVD-ROM.

6 Стадии и этапы разработки

6.1 Эскизный проект (ЭП)

- Анализ предметной области.
- Подготовка проектной документации.

6.2 Технический проект (ТП)

- Разработка структур и форм представления данных.
- Разработка структуры программного комплекса.
- Подготовка пояснительной записки.

6.3 Рабочий проект (РП)

- Программная реализация.
- Тестирование и отладка программы.
- Подготовка программной и эксплуатационной документации.

6.4 Эксплуатация (Э)

Описание и анализ результатов проведенного исследования.

7 Требования к документированию проекта

К защите курсового проекта должны быть представлены следующие документы:

- Пояснительная записка к курсовому проекту:
- Презентация доклада.

- Программа, презентация и пояснительная записка к курсовому проекту на оптическом носителе.

Требования к структуре документов определены соответствующими стандартами ЕСПД.

Требования к оформлению определены соответствующими методическими указаниями.

8 Порядок контроля и приемки

Контроль выполнения курсового проекта проводится руководителем поэтапно в соответствии с утвержденным графиком выполнения проекта.

На завершающем этапе руководитель осуществляет нормоконтроль представленной исполнителем документации и принимает решение о допуске (недопуске) проекта к защите.

Защита курсового проекта проводится комиссией в составе не менее двух человек, включая руководителя проекта.

В процессе защиты проекта исполнитель представляет документацию, делает краткое сообщение по теме разработки и демонстрирует ее программную реализацию.

При выставлении оценки учитывается:

- степень соответствия представленной разработки требованиям технического задания;
- качество программной реализации, документации и доклада по теме проекта;
- соблюдение исполнителем графика выполнения курсового проекта.

9 Литература

1. Айсберг Е. Транзистор?.. Это очень просто! Перевод с французского Ю. Л. Смирнова. М. – Л., издательство «ЭНЕРГИЯ», 1963. 112 стр. с ил. (Массовая радиобиблиотека. Вып. 480).
2. Р. Хаггарти. Дискретная математика для программистов. Москва: Техносфера, 2003. – 320 с.
3. Шлее М. Qt 5.10. Профессиональное программирование на C++. – СПб.: БХВ-Петербург, 2018. – 1072 с.: ил. – (В подлиннике)

ПРИЛОЖЕНИЕ 2

ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ

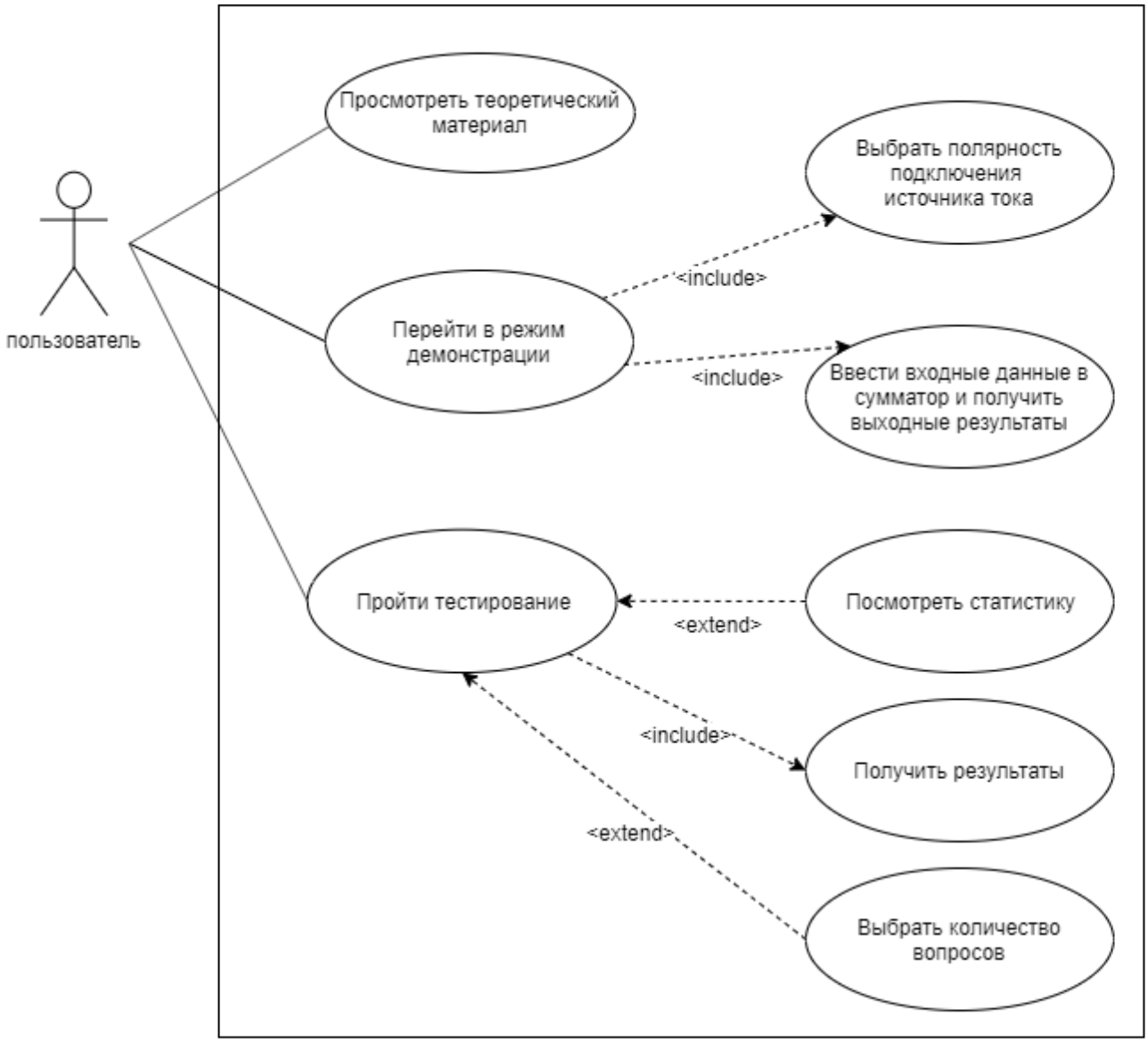

Рисунок П.2.1 — Диаграмма вариантов использования

ПРИЛОЖЕНИЕ 3

ДИАГРАММА КЛАССОВ

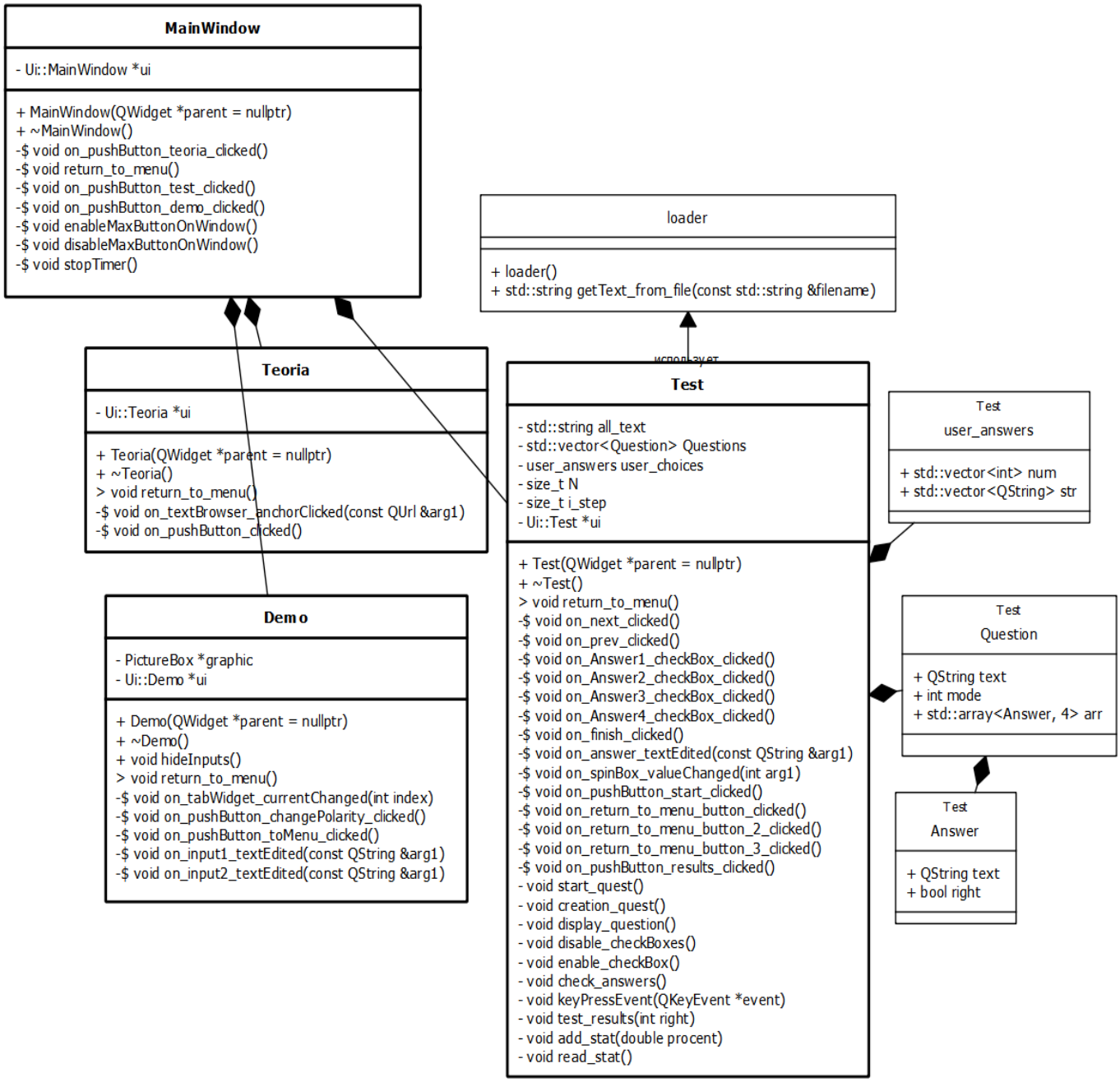

Рисунок П.3.1 — Диаграмма классов (1 часть)

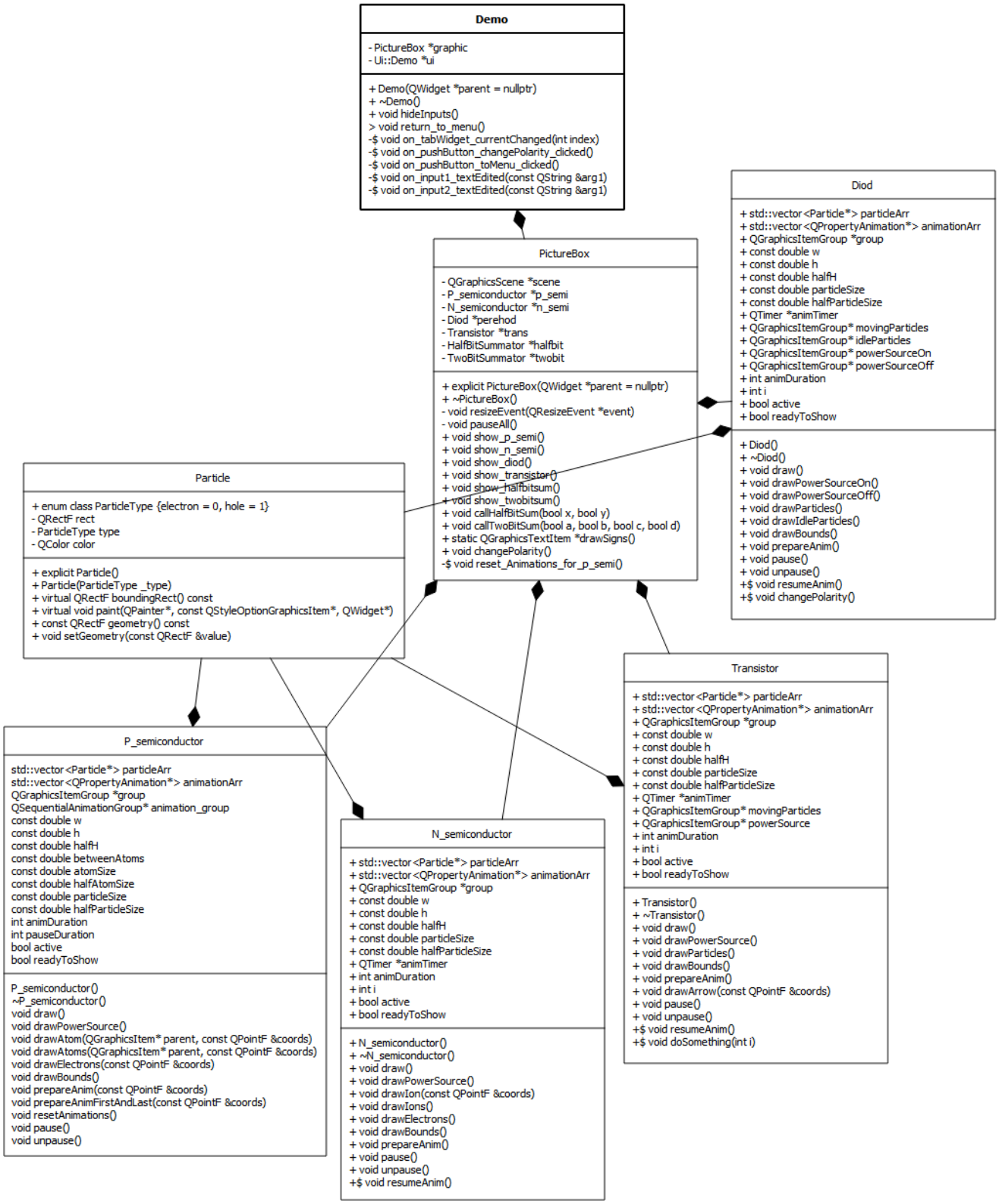

Рисунок П.3.2 — Диаграмма классов (2 часть)

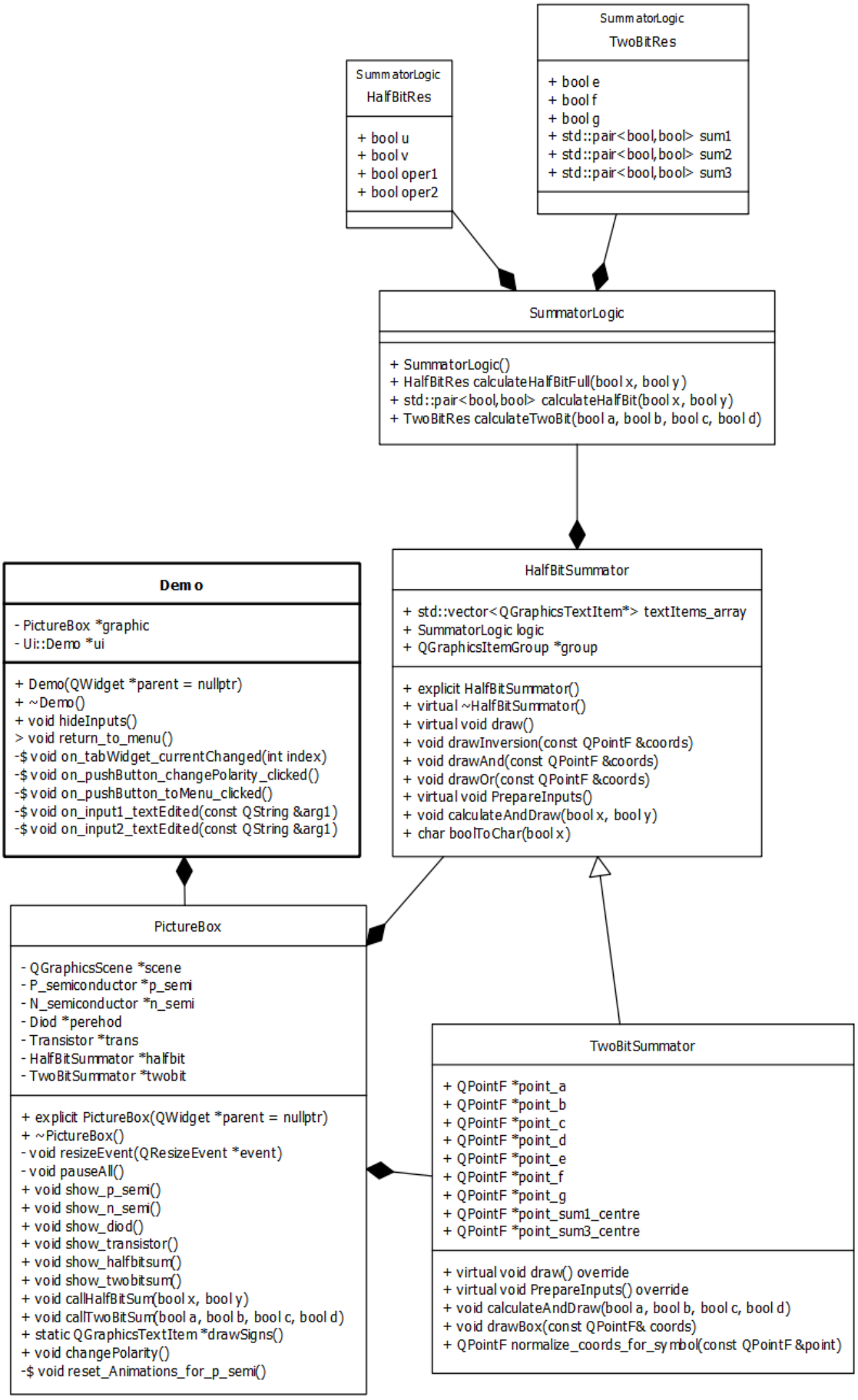

Рисунок П.3.3 — Диаграмма классов (3 часть)

ПРИЛОЖЕНИЕ 4

БАЗА ТЕКСТОВЫХ ВОПРОСОВ

**(содержание файла quest.txt,
первый ответ является правильным)**

?1Полупроводник имеет ... электрон-а/ов на внешней оболочке атома...

4

3

2

5

?1Для создания концентрации дырок в полупроводниках используется ...?

легирование полупроводника акцепторными примесями

легирование полупроводника донорами

термическая обработка

воздействие магнитного поля

?2Незаполненная валентная связь в полупроводнике, которая проявляет себя как положительный заряд?

дырка

?1В P-n переходе ток идет, если источник тока подключен...

в прямом направлении

в обратном направлении

в любом направлении

ток не идет при любом подключении

?1В n-полупроводнике ...?

избыток электронов

избыток дырок

недостаток электронов

ничего из перечисленного

?1Транзистор используется для ...

все перечисленное

генерирования сигналов

используется как "электрический ключ"

усиления сигнала

?1Какие логические операции можно реализовать с помощью транзистора?

все перечисленные операции

операция НЕ

операция ИЛИ

операция И

?1Какая валентность у примесного атома в полупроводнике р-типа?

3

2

4

5

?2Радиоэлектронный компонент из полупроводникового материала, обычно с тремя выводами способный от небольшого входного сигнала управлять значительным током в выходной цепи - это ...

транзистор

?2Общее название устройства, которое вычисляет сумму двух двоичных чисел

сумматор