Федеральное агентство по образованию

Государственное образовательное учреждение
высшего профессионального образования

Владимирский государственный университет

Кафедра управления качеством и технического регулирования

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ЛАБОРАТОРНЫМ РАБОТАМ ПО ДИСЦИПЛИНАМ «СЕТИ, ЭВМ И СРЕДСТВА КОММУНИКАЦИЙ», «ВЫЧИСЛИТЕЛЬНАЯ И МИКРОПРОЦЕССОРНАЯ ТЕХНИКА»

Составитель
Д.Ю. ОРЛОВ

Владимир 2007

УДК 621.317 + 621.753.1 + 658.16 (075.8)
ББК 32.972я7
М54

Рецензент
Кандидат технических наук, доцент кафедры управления
качеством и технического регулирования
Владимирского государственного университета
З.В. Мищенко

Печатается по решению редакционного совета
Владимирского государственного университета

М54 **Методические** указания к лабораторным работам по дисциплинам «Сети, ЭВМ и средства коммуникаций», «Вычислительная и микропроцессорная техника» / Владим. гос. ун-т ; сост. Д. Ю. Орлов. – Владимир : Изд-во Владим. гос. ун-та, 2007. – 57 с.

Методические указания включают в себя цель выполнения лабораторных работ, краткую теорию, методику выполнения, описание работы программного обеспечения и виртуального моделирующего комплекса Electronics Workbench. Представлены вопросы, требования к оформлению отчета и библиографический список.

Предназначены для самостоятельного изучения студентами специальностей 200503 – стандартизация и сертификация, 220501 – управление качеством, 200501 – метрология и метрологическое обеспечение дневной формы обучения.

Табл. 19. Ил. 21. Библиогр.: 6 назв.

УДК 621.317 + 621.753.1 + 658.16 (075.8)
ББК 32.972я7

1. ОРГАНИЗАЦИЯ ВЫПОЛНЕНИЯ ЛАБОРАТОРНЫХ РАБОТ

Перед выполнением лабораторных работ каждый студент обязан изучить правила техники безопасности и пожарной безопасности при работе в компьютерном классе, изложенные в действующих инструкциях, а затем расписаться в журнале инструктажей.

Очередность выполнения работ указывается преподавателем.

Началу выполнения работы предшествует допуск, который включает в себя знание цели работы, теоретического материала, принципов работы с ЭВМ и методики выполнения работы.

Студенты, не подготовившиеся к занятию, до проведения работы не допускаются. Отработка лабораторных работ проводится вне расписания по согласованию с преподавателем. При наличии задолженностей по двум незащищенным работам к выполнению следующей работы студент не допускается.

После выполнения работы каждый студент должен подписать у преподавателя полученные результаты, выключить ЭВМ, контрольно-измерительные приборы и привести рабочее место в порядок.

Отчет о выполнении лабораторной работы представляется каждым студентом на листе бумаги форматом А4 в соответствии с СТП 71.5-04. Отчет должен содержать:

– титульный лист с указанием кафедры, учебной группы, фамилии, имени, отчества студента, номера и названия лабораторной работы, а также фамилии и должности преподавателя, принимающего работу;

– цель работы, теоретическое описание работы и, в зависимости от типа работы, логические схемы, алгоритмы работы программ таблицы смоделированных величин и другой требуемый по заданию графический и текстовый материал;

– краткие выводы по результатам проделанной работы.

2. СИСТЕМА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ ELECTRONICS WORKBENCH (САПР)

Система автоматизированного проектирования (САПР) Electronics Workbench предназначена для моделирования и анализа электрических и логических схем. Правильно говорить: система моделирования и анализа электрических и логических схем Electronics Workbench, но для краткости здесь и далее мы будем называть ее программой.

Программа Electronics Workbench позволяет моделировать аналоговые, цифровые и цифро-аналоговые схемы большой степени сложности. Имеющиеся в программе библиотеки включают в себя большой набор широко распространенных электронных компонентов. Есть возможность подключения и создания новых библиотек компонентов.

Параметры компонентов можно изменять в широком диапазоне значений. Простые компоненты описываются набором параметров, значения которых можно изменять непосредственно с клавиатуры, активные элементы – моделью, представляющей собой совокупность параметров и описывающей конкретный элемент или его идеальное представление. Модель выбирается из списка библиотек компонентов, параметры модели также могут быть изменены пользователем.

Широкий набор приборов позволяет проводить измерения различных величин, задавать входные воздействия, строить графики. Все приборы изображаются в виде, максимально приближенном к реальному, поэтому работать с ними просто и удобно.

Результаты моделирования можно вывести на принтер или импортировать в текстовый или графический редактор для их дальнейшей обработки.

Программа Electronics Workbench совместима с программой P-SPICE, то есть предоставляет возможность экспорта и импорта схем и результатов измерений в различные ее версии.

Основные достоинства программы

Экономия времени. Работа в реальной лаборатории требует больших временных затрат на подготовку эксперимента. Теперь, с появлением Electronics Workbench, электронная лаборатория всегда есть под рукой, что позволяет сделать изучение электрических и логических схем более доступным.

Достоверность измерений. В природе не существует двух совершенно одинаковых элементов, т. е. все реальные элементы имеют большой разброс значений, что приводит к погрешностям в ходе проведения эксперимента. В Electronics Workbench все элементы описываются строго установленными параметрами, и поэтому каждый раз в ходе эксперимента будет повторяться результат, определяемый только параметрами элементов и алгоритмом расчета.

Удобство проведения измерений. Учеба невозможна без ошибок, а ошибки в реальной лаборатории порой очень дорого обходятся экспериментатору. Работая с Electronics Workbench, экспериментатор застрахован от случайного поражения током, а приборы не выйдут из строя из-за неправильно собранной схемы. Благодаря этой программе в распоряжении пользователя имеется такой широкий набор приборов, который вряд ли будет доступен в реальной жизни. Таким образом, всегда имеется уникальная возможность для планирования и проведения широкого спектра исследований электронных схем при минимальных затратах времени.

Графические возможности. Сложные схемы занимают достаточно много места, изображение при этом стараются сделать более плотным, что часто приводит к ошибкам в подключении проводников к элементам цепи. Electronics Workbench позволяет разместить схему таким образом, чтобы были четко видны все соединения элементов и одновременно вся схема целиком. Возможность изменения цвета проводников позволяет сделать схему более удобной для восприятия. Можно отображать различными цветами и графики, что очень удобно при одновременном исследовании нескольких зависимостей.

Стандартный интерфейс Windows. Программа Electronics Workbench использует стандартный интерфейс Windows, что значительно облегчает ее использование. Интуитивность и простота интерфейса делают программу доступной любому, кто знаком с основами использования Windows.

Совместимость с программой P-SPICE. Программа Electronics Workbench базируется на стандартных элементах программы SPICE. Это позволяет экспортировать различные модели элементов и проводить обработку результатов, используя дополнительные возможности различных версий программы P-SPICE.

Для установки программы необходимы:

- IBM-совместимый компьютер с модификацией процессора не ниже Pentium 200;
- не менее 10MB свободного пространства на жестком диске;
- операционная система MS Windows 9x/2000/XP или более поздние версии;
- манипулятор типа «мышь».

Компоненты и проведение экспериментов

В библиотеки компонентов программы входят пассивные элементы, транзисторы, управляемые источники, управляемые ключи, гибридные элементы, индикаторы, логические элементы, триггерные устройства, цифровые и аналоговые элементы, специальные комбинационные и последовательные схемы. Активные элементы могут быть представлены моделями как идеальных, так и реальных элементов. Возможно также создание своих моделей и добавление их в библиотеки элементов.

В программе используется большой набор приборов для проведения измерений: амперметр, вольтметр, осциллограф, мультиметр, Боде-плоттер (графопостроитель частотных характеристик схем), функциональный генератор, генератор слов, логический анализатор и логический преобразователь.

Electronics Workbench может проводить анализ схем на постоянном и переменном токах. При анализе на постоянном токе определяется рабочая точка схемы в установившемся режиме работы. Результаты этого анализа не отражаются на приборах, они используются для дальнейшего анализа схемы. Анализ на переменном токе использует результаты анализа на постоянном токе для получения линеаризованных моделей нелинейных компонентов. Анализ схем в режиме АС может проводиться как во временной, так и в частотной областях. Программа также позволяет производить анализ цифро-аналоговых и цифровых схем.

В Electronics Workbench можно исследовать переходные процессы при воздействии на схемы входных сигналов различной формы.

Electronics Workbench позволяет строить схемы различной степени сложности при помощи определенных операций, среди которых:

- выбор элементов и приборов из библиотек;
- перемещение элементов и схем в любое место рабочего поля;
- поворот элементов и групп элементов на углы, кратные 90 градусам;
- копирование, вставка или удаление элементов, групп элементов, фрагментов схем и целых схем;
- изменение цвета проводников;
- выделение цветом контуров схем для более удобного восприятия;
- одновременное подключение нескольких измерительных приборов и наблюдение их показаний на экране монитора;
- присваивание элементу условного обозначения;
- изменение параметров элементов в широком диапазоне.

Все операции выполняются при помощи мыши и клавиатуры. Управление только с клавиатуры невозможно. Путем настройки приборов можно:

- изменять шкалы приборов в зависимости от диапазона измерений;
- задавать режим работы прибора;
- задавать вид входных воздействий на схему (постоянные и гармонические токи и напряжения, треугольные и прямоугольные импульсы).

Графические возможности программы позволяют:

- одновременно наблюдать несколько кривых на графике;
- отображать кривые на графиках различными цветами;
- измерять координаты точек на графике;
- импортировать данные в графический редактор, что позволяет осуществить необходимые преобразования рисунка и вывод его на принтер.

Electronics Workbench позволяет использовать результаты, полученные в программах P-SPICE, PCB, а также передавать результаты из Electronics Workbench в эти программы. Можно вставить схему или ее фрагмент в текстовый редактор и напечатать в нем пояснения или замечания по работе схемы.

В библиотеки элементов программы Electronics Workbench входят аналоговые, цифровые и цифро-аналоговые компоненты. Все компоненты можно условно разбить на следующие группы: базовые компоненты, источники, линейные компоненты, ключи, нелинейные компоненты, индикаторы, логические компоненты, узлы комбинационного типа, узлы последовательного типа, гибридные компоненты.

Лабораторный практикум в системе автоматизированного проектирования Electronics Workbench

Данный лабораторный практикум в Electronics Workbench выполняет роль своеобразного инструктора в области экспериментального исследования электронных схем на персональном компьютере при изучении дисциплин «Сети, ЭВМ и средства коммуникаций», «Вычислительная и микропроцессорная техника», «Информатика» и состоит из комплекта лабораторных работ по следующей тематике:

– логические схемы и функции алгебры логики;
– комбинационные схемы средней степени интеграции (исследование дешифраторов, мультиплексоров) и т. д.

При этом у студентов подразумевается наличие навыков работы с персональным компьютером (ПК) и достаточный для изучения дисциплины объем знаний о логических и арифметических основах работы цифровых устройств на основе курса школьной информатики.

Инструкции к каждой лабораторной работе включают следующие разделы: цель работы, приборы и элементы, краткие сведения из теории, порядок проведения экспериментов, результаты экспериментов, контрольные вопросы.

Проведение лабораторного практикума по Electronics Workbench принципиально позволяет решить три задачи:

1. Привести краткие теоретические сведения, с помощью которых можно исследовать представленные схемы аналитическими методами. После такого теоретического анализа студент должен представлять себе картину процессов в рассматриваемых схемах.

2. Раскрыть методику проведения экспериментальных исследований, в которой используются широкие потенциальные возможности всего многообразия инструментальных средств и библиотек элементов программы Electronics Workbench. Студент при этом получает возможность понять и представить весь арсенал методов планирования и проведения экспериментов в схемах, построенных на базе имеющихся в программе моделей идеальных и реальных элементов.

3. Представить схемы для самостоятельных исследований с уже отлаженными установками в качестве исходного материала для решения самых различных задач. При этом преследуются две цели: во-первых, показать широкий арсенал неочевидных приемов, которыми будущий специалист может воспользоваться на практике; во-вторых, путем подбора схем и параметров их элементов обеспечить получение результата за короткое время (как правило, не более пяти минут).

Для этого все приведенные в лабораторном практикуме схемы тщательно проверены, их моделирование дает результат, адекватный теоретическому анализу.

Обычно порядок работы при моделировании цифровых устройств с помощью Electronics Workbench выглядит следующим образом:

1. Обозначается цель и алгоритм работы цифрового устройства.

2. Разрабатывается логическая схема. При этом используются законы алгебры логики и способы минимизации логических функций.

3. Разработанная схема переносится в Electronics Workbench для анализа ее работы и, если это необходимо, вносится соответствующая корректировка в схему.

Работа с Electronics Workbench происходит в соответствии с концепцией, принятой в Windows, и после загрузки окно программы (рис. 1) состоит из рабочей области 1, предназначенной для непосредственного составления логической схемы; панелей логических элементов и дополнительных устройств 2; кнопки «Пуск» 3, включающей собранную схему в работу.

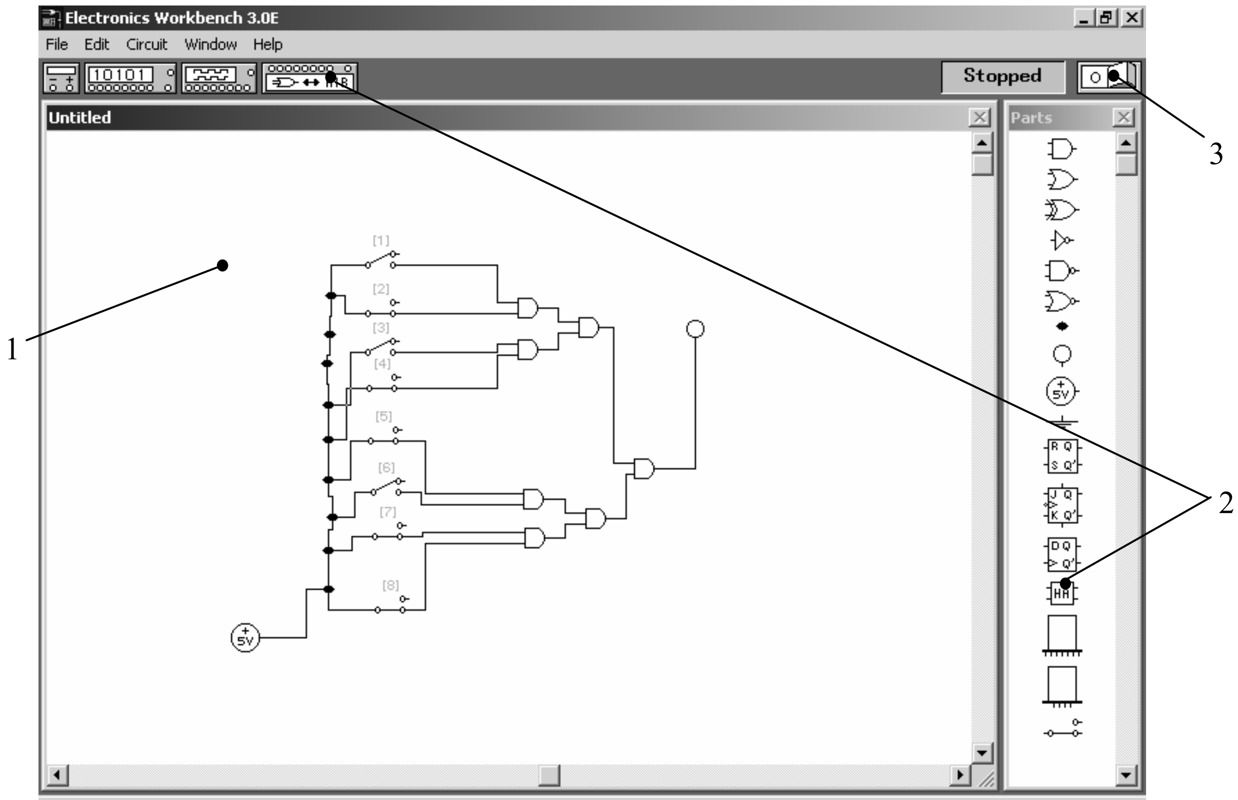


Рис. 1. Рабочая область Electronics Workbench

Составление схемы происходит обычным в Windows методом переноса (drag & drop) логических элементов из панели элементов в рабочую область с последующим их соединением в логическую схему.

3. ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа № 1
ИЗУЧЕНИЕ ОСНОВ РАБОТЫ ЦИФРОВЫХ УСТРОЙСТВ

***Цель работы**:* изучение основ работы цифровых устройств, законов алгебры логики, знакомство с системой автоматизированного проектирования электронных устройств Electronics Workbench.

1. Объект и средства исследования

Объектом исследования являются: среда автоматизированного проектирования Electronics Workbench, цифровые устройства, законы алгебры логики, формы представления логических функций, логические элементы, таблицы состояний логических элементов и логические схемы.

2. Теоретическая часть

Основные понятия *булевой алгебры* – это понятия логической переменной и логической функции.

Логической переменной называется величина, которая может принимать одно из двух возможных состояний (значений), одно из которых обозначается символом «0», другое – «1» (для обозначения состояний возможно применение и других символов, например, «Да» и «Нет», «Включено» и «Выключено» и др.). Сами двоичные переменные чаще обозначают символами x_1, x_2, … . В силу определения логические переменные можно называть также двоичными переменными.

Логической (булевой) функцией (обычное обозначение – y) называется функция двоичных переменных (аргументов), которая также может принимать одно из двух возможных состояний (значений): «0» или «1». Значение некоторой логической функции n переменных определяется или задается для каждого набора (сочетания) двоичных переменных. Количество возможных различных наборов, которые могут быть составлены из n аргументов, очевидно, равно 2^n. При этом, поскольку сама функция на каждом наборе может принимать значение «0» или «1», то общее число возможных функций от n переменных равно 2^{2^n}.

Таким образом, множество состояний (значений), которые могут принимать как аргументы, так и функции, равно двум. Для этих состояний в булевой алгебре определяются отношение эквивалентности, обозначаемое символом равенства (=) и три операции: а) логического сложения (дизъюнкции); б) логического умножения (конъюнкции); в) логического отрицания (инверсии), обозначаемые соответственно символами:

$+$ или $\vee$ – операция дизъюнкции;

$\bullet$ или $\wedge$, или & – операция конъюнкции;

$\bar{*}$ – операция инверсии (* – символ аргумента или функции).

Известны следующие соотношения (законы) алгебры логики:

1. Законы одинарных элементов (а – универсального множества; б – нулевого множества, в – тавтологии):

   а) $x + 1 = 1$, $x \cdot 1 = x$; б) $x + 0 = x$, $x \cdot 0 = 0$; в) $x + x = x$, $x \cdot x = x$.

2. Законы отрицания (а – двойного отрицания; б – дополнительности, в – двойственности):

   а) $\bar{\bar{x}} = x$; б) $x + \bar{x} = 1$, $x \cdot \bar{x} = 0$; в) $\overline{x_1 + x_2} = \bar{x}_1 \cdot \bar{x}_2$, $\overline{x_1 \cdot x_2} = \bar{x}_1 + \bar{x}_2$.

3. Законы абсорбции или поглощения (а) и склеивания (б):

   а) $x_1 + x_1 \cdot x_2 = x_1$, $x_1 \cdot (x_1 + x_2) = x_1$; б) $x_1 \cdot x_2 + x_1 \cdot \bar{x}_2 = x_1$, $(x_1 + x_2) \cdot (x_1 + \bar{x}_2) = x_1$.

Законы двойственности (2, *в*), называемые также законами де Моргана, были обобщены К. Шенноном на случай произвольного (n) числа аргументов.

Кроме законов, перечисленных выше и не имеющих аналогов в обычной алгебре (алгебре чисел), для алгебры логики справедливы законы обычной алгебры: коммутативные или переместительные, дистрибутивные или распределительные, ассоциативные или сочетательные, например:

Сочетательный закон $\begin{cases} x_1 \cdot (x_2 \cdot x_3) = (x_1 \cdot x_2) \cdot x_3 \\ x_1 + (x_2 + x_3) = (x_1 + x_2) + x_3 \end{cases}$

Переместительный закон $\begin{cases} x_1 \cdot x_2 = x_2 \cdot x_1 \\ x_1 + x_2 = x_2 + x_1 \end{cases}$

Распределительный закон $\begin{cases} x_1 \cdot (x_2 + x_3) = x_1 \cdot x_2 + x_1 \cdot x_3 \\ x_1 + (x_2 \cdot x_3) = (x_1 + x_2) \cdot (x_1 + x_3) \end{cases}$

Для сложного логического выражения установлен определенный порядок выполнения операций: 1) операции инверсии; 2) операции конъюнкции; 3) операции дизъюнкции. Например, запись логического выражения $x_1 \vee x_2 \cdot \overline{x}_3 \vee \overline{x}_4 \cdot x_2$ предполагает, что при вычислении выражения вначале выполняют операции инверсии $\overline{x}_3$ и $\overline{x}_4$, затем операции конъюнкции $x_2 \cdot \overline{x_3}$ и $\overline{x_4} \cdot x_2$ и в конце – операции дизъюнкции. Если требуется нарушить это правило, используют скобки. Например $(x_1 \vee x_2) \cdot (\overline{x_3} \cdot x_4)$. В этом случае сначала выполняют операции в скобках (а если одни скобки вложены в другие, то в первую очередь выполняют операции в самых внутренних скобках).

Любая логическая функция y n двоичных переменных $x_1, x_2, ..., x_n$ может быть задана таблично. Такие таблицы, получившие название *таблиц истинности*, содержат 2^n строк, в которые записываются все возможные двоичные наборы значений аргументов, а также соответствующее каждому из этих наборов значение функции.

Пример 1

Задание. Составить таблицу истинности логической функции y равнозначности (эквивалентности) трех двоичных переменных x_1, x_2 и x_3, т. е. функции, которая принимает единичное значение только при совпадении всех трех аргументов, ее образующих.

Решение. Сначала выпишем все возможные наборы (комбинации) трех переменных x_1, x_2 и x_3. Таких наборов, очевидно, 8.

Чтобы не ошибиться при перечислении наборов аргументов, нужно сразу приучиться перечислять их единообразно – в виде возрастающей последовательности чисел, представленных в двоичной системе счисления. Для рассматриваемого примера наборы трех переменных нужно перечислить в следующем порядке: 000, 001, 010, 011, 100, 101, 110, 111 – итого восемь двоичных чисел – от 0 до 7.

Далее для каждого набора двоичных переменных определим, исходя из смысла ситуации, соответствующее значение функции. В результате получаем таблицу истинности логической функции «равнозначность трех двоичных переменных» (табл. 1).

Таблица 1

Номер набора	Двоичные переменные			Логические функции
	x_1	x_2	x_3	y
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Задание логической функции таблицей истинности не всегда удобно. При большом числе двоичных переменных ($n \geq 6$) табличный способ задания функции становится громоздким и теряет наглядность. Возможен и аналитический способ задания логических функций, который предусматривает запись функции в форме логического выражения, устанавливающего, какие логические операции над аргументами функции должны выполняться и в какой последовательности.

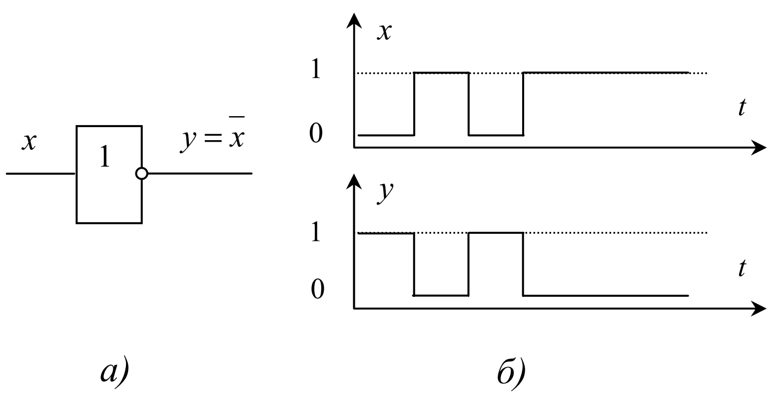
Алгебра логики предполагает возможность образования сложных функций, т. е. функций, аргументы которых являются функциями других двоичных аргументов. Например, если $y = f(z_1, z_2)$, а $z_1 = f'(x_1, x_2)$ и $z_2 = f''(x_3, x_4)$, очевидно, что $y = f(x_1, x_2, x_3, x_4)$. Операция замены аргументов одной функции другими функциями называется *суперпозицией* функций. Эта операция дает возможность выразить сложную логическую функцию через более простые (элементарные).

Приведем описание некоторых, имеющих большое значение в цифровой технике, элементарных логических функций и ЛЭ, реализующих эти функции.

Функция «отрицание» – это функция одного аргумента (другие названия функции: *инверсия, логическая связь НЕ*). Аналитическая форма задания этой функции: $y = \overline{x}$, где y – логическая функция, x – аргумент.

Электронный ЛЭ, реализующий функцию «Отрицание» в виде определенных уровней электрических сигналов, называют *инвертором,* или ЛЭ «НЕ». Инвертор на схемах изображается, как показано на рис. 2, *а*. Вход ЛЭ слева, выход – справа. На выходной линии, в месте соединения ее с прямоугольником, изображается кружок – *символ инверсии*. На языке цифровой техники инверсия означает, что выходной сигнал (y) противоположен входному (x).

Сказанное иллюстрирует рис. 2, *б*, на котором приведены временные диаграммы инвертора.



x	y
0	1
1	0

в)

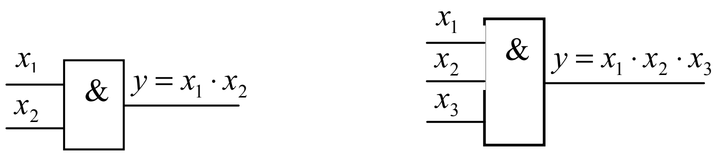
Рис. 2. Инвертор: а – условное изображение; б – временные диаграммы; в – таблица истинности

Функция «конъюнкция» – это функция двух или большего числа аргументов (другие названия функции: *логическое умножение, логическая связь И*). Аналитическая форма задания функции двух аргумент x_1 и x_2:

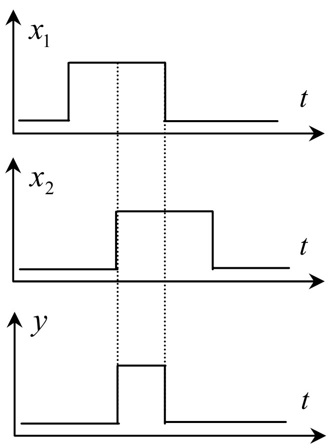
$$y = x_1 \cdot x_2 \text{ или } y = x_1 \wedge x_2 \text{ или } y = x_1 \,\&\, x_2 .$$

Функция «конъюнкция» равна тогда и только тогда, когда все ее аргументы равны 1.

ЛЭ, реализующий функцию «Конъюнкция» называют конъюнктором или ЛЭ «И». На рис. 3 приведены: условное графическое изображение двухвходового (а) и трехвходового (б) конъюнкторов; временные диаграммы (в) и таблица истинности (г) двухвходового конъюнктора.



а) *б)*



x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

в) *г)*

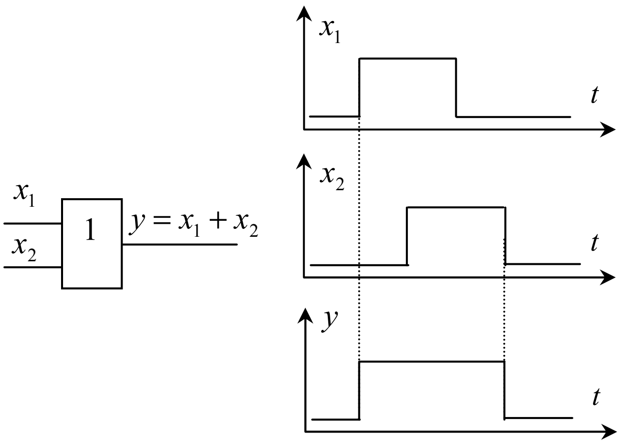
Рис. 3. Конъюнктор

ЛЭ «И» часто используют для управления потоком информации. При этом на один из его входов поступают сигналы, несущие некоторую информацию, а на другой – управляющий сигнал: пропустить информацию – 1, не пропустить – 0. ЛЭ «И», используемый таким образом, называют *вентиль*.

Функция «дизъюнкция» – это функция двух или большего числа аргументов (другие названия функции: *логическое сложение, логическая связь ИЛИ*). Функция равна 1, если хотя бы один из ее аргументов равен 1 (рис. 4, *в*). Обозначение функции «Дизъюнкция»:

$$y = x_1 + x_2 \text{ или } y = x_1 \vee x_2.$$

ЛЭ, реализующий функцию «дизъюнкция», называют дизъюнктором или ЛЭ «ИЛИ». Условное изображение и временные диаграммы ЛЭ «ИЛИ» приведены на рис. 4.



x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

а) *б)* *в)*

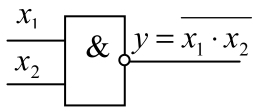
Рис. 4. Дизъюнктор: а – условное изображение; б – временные диаграммы; в – таблица истинности

Функция «штрих Шеффера» (другое название функции – *логическая связь «И-НЕ»*) – это функция двух или большего числа аргументов. Таблица истинности функции «И-НЕ» представлена на рис. 5, *б*.

Легко видеть, что это инверсия функции «И», т. е. отрицание конъюнкции. Функция равна 1, если равен 0 хотя бы один из ее аргументов, функция равна 0 при равенстве всех аргументов 1.

Обозначение функции «И-НЕ»: $y = \overline{x_1 \cdot x_2}$.

Условное изображение ЛЭ, реализующего функцию «штрих Шеффера», приведено на рис. 5, *а*.



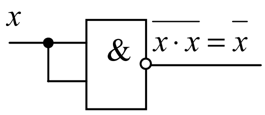
а)

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

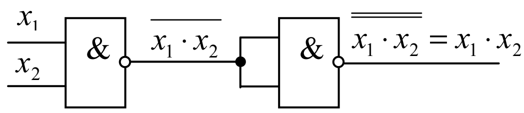
б)

Рис. 5. ЛЭ «И-НЕ»:
а – условное изображение;
б – таблица истинности

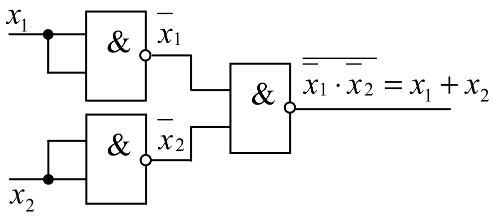
Используя только ЛЭ «И-НЕ», можно реализовать любую из вышерассмотренных логических функций (НЕ, И, ИЛИ), как показано на рис. 6.



а)



б)



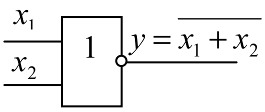
в)

Рис. 6. Реализация с помощью ЛЭ «И-НЕ» функций: а – НЕ; б – И; в – ИЛИ

Функция «стрелка Пирса» – это функция двух или большего числа аргументов (другое название функции – *логическая связь «ИЛИ-НЕ»*).

Данная функция является инверсией функции «ИЛИ», значения функции представлены на рис. 7, *б*, в формулах обозначается как $y = \overline{x_1 + x_2}$.

Условное изображение ЛЭ, реализующего функцию «ИЛИ-НЕ», приведено на рис. 7, *а*.

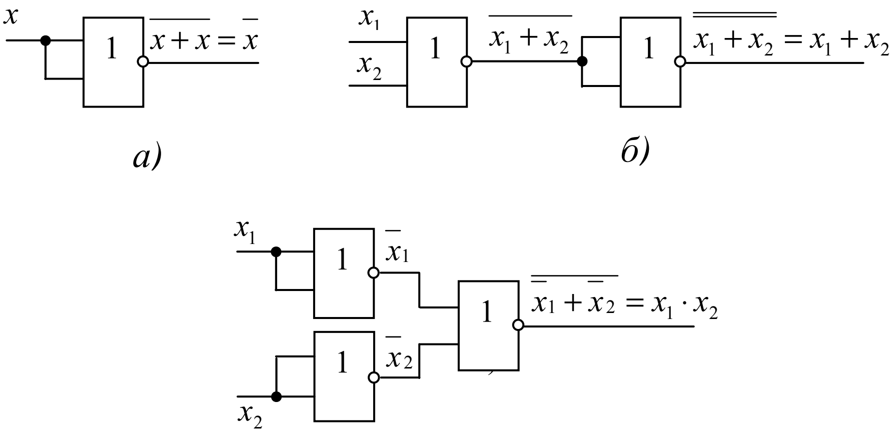


x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0

а) *б)*

Рис. 7. ЛЭ «ИЛИ-НЕ»:
а – условное изображение;
б – таблица истинности

ЛЭ «ИЛИ-НЕ», также как и ЛЭ «И-НЕ», позволяет реализовывать логические функции НЕ, ИЛИ, И. Отмеченное иллюстрирует рис. 8.



а) *б)*

в)

Рис. 8. Реализация с помощью ЛЭ «ИЛИ-НЕ»
функций: а – НЕ; б – ИЛИ; в – И

Функция «сумма по модулю 2»(М2) – это функция двух или большего числа аргументов. Обозначение в формулах: $y = x_1 \oplus x_2$ (в случае функции двух аргументов x_1 и x_2).

Таблица истинности функции представлена на рис. 9, *а*. На рис. 9, *б* приведено условное графическое изображение двухвходового ЛЭ, реализующего эту функцию. Название функции связано с тем, что $x_1 \oplus x_2$ есть арифметическая сумма двоичных чисел x_1 и x_2 в пределах одного разряда: 0 + 0 = 0; 0 + 1 = 1;

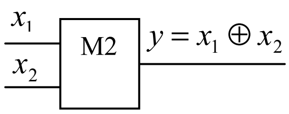
1 + 0 = 1; 1 + 1 = 10. В последнем случае возникает единица переноса в соседний старший разряд, а в разряде самих слагаемых получается нуль. Отсюда широкое применение этого ЛЭ при построении суммирующих устройств.

Функция М2 обладает интересным свойством, которое полезно запомнить: при инвертировании одного из аргументов вся функция инвертируется, т. е. $\overline{x_1} \oplus x_2 = x_1 \oplus \overline{x_2} = \overline{x_1 \oplus x_2}$.

Инверсия суммы по модулю 2 для двух аргументов имеет и собственный смысл: это функция *равнозначности* $x_1 \equiv x_2$; она равна единице, если $x_1 = x_2$. Следовательно, для построения схем сравнения одноразрядных чисел достаточно проинвертировать один из аргументов или результат.

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

а)



б)

Рис. 9. Функция М2: а – таблица истинности; б – изображение ЛЭ, реализующего М2

Полезно запомнить также следующие очевидные соотношения:

$$x \oplus 0 = x; \quad x \oplus 1 = \overline{x};$$

$$x \oplus x = 0; \quad x \oplus \overline{x} = 1.$$

Первые два равенства позволяют применять ЛЭ М2 в качестве *управляемого инвертора*. Если использовать один из входов М2 как управляющий и подавать на него уровень логического 0 или 1, то информация, поступающая по второму входу, будет пропускаться на выход без изменения или инвертироваться.

В случае двух аргументов функцию М2 называют также функцией *неравнозначности, исключающим ИЛИ*, поскольку полностью совпадают таблицы истинности этих функций. Если же функция М2 трех или большего числа аргументов, то применение названий «неравнозначность», «исключающее ИЛИ» не правомерно. Последнее следует из сопоставления таблиц истинности этих функций (табл. 2), из которой ясно, что это совершенно различные функции.

Таблица 2

Аргументы			Функции		
x_1	x_2	x_3	$M2 = x_1 \oplus x_2 \oplus x_3$	Неравно-значность	Исключающее ИЛИ (один и только один)
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	0	1	0
1	1	1	1	0	0

Стандартные ИС ЛЭ И, ИЛИ, И-НЕ, ИЛИ-НЕ имеют 2, 3, 4 или 8 входов. Число аргументов, входящих в конъюнкцию (дизъюнкцию) или ее инверсию, может отличаться от числа входов ЛЭ. Типовыми ситуациями являются наличие у имеющегося ЛЭ «лишних» (неиспользуемых) в данном случае входов или, напротив, нехватка у имеющегося ЛЭ необходимого числа входов. Например, нужно получить конъюнкцию (дизъюнкцию) или ее инверсию пяти переменных. В сериях ИС нет ЛЭ с пятью входами и придется взять элемент с восемью входами, у которого окажется три «лишних» входа (рис. 10, *а*). Принципиально возможно поступить следующим образом: «лишние» входы подсоединить к задействованным (рис. 10, *б*) или подать на них некоторые константы (логические «1» или «0»), не изменяющие логику работы ЛЭ (рис. 10, *в*).

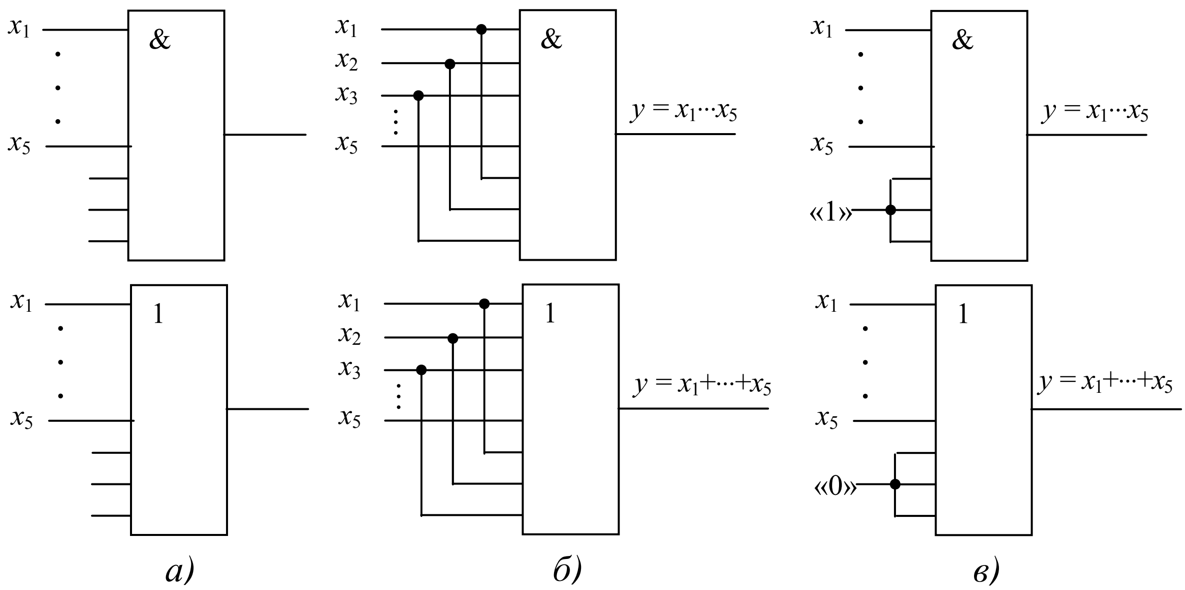


Рис. 10. Подключение неиспользуемых входов

В других случаях число входов ЛЭ меньше числа аргументов конъюнкции (дизъюнкции) или ее инверсии. Для ЛЭ И и ИЛИ решение задачи не представляет трудностей – для получения нужного числа входов берется несколько ЛЭ, выходы которых объединяются далее элементом того же типа (рис. 11). На этом рисунке звездочка обозначает операцию конъюнкцию или дизъюнкцию.

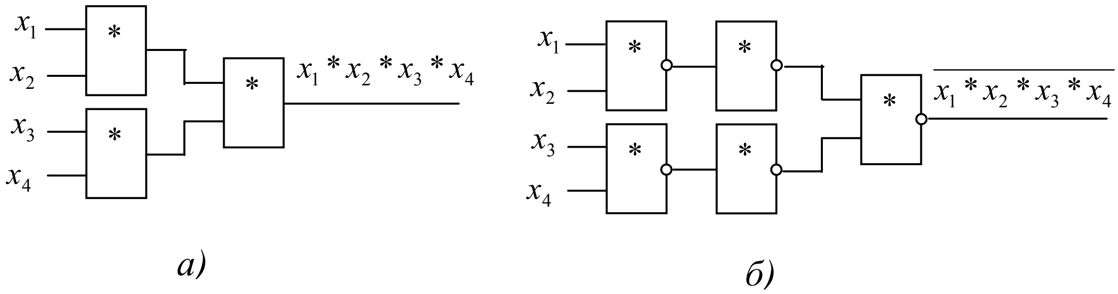


Рис. 11. Схема наращивания числа входов ЛЭ

Таблица истинности, название и обозначение некоторых логических функций по системе ГОСТ и ISO рассмотрены в прил. 1.

3. Порядок выполнения работы

1. Для ЛЭ, соответствующих вашему варианту (табл. 3), снять таблицу истинности; записать логические выражения, реализуемые ЛЭ; изобразить временные диаграммы, характеризующие работу ЛЭ.

Таблица 3

Номер бригады (варианта)	Исследуемые логические элементы							
	НЕ	2И	2И-НЕ	2ИЛИ	2М2	3И	3И-НЕ	3ИЛИ
1	+			+	+	+	+	
2	+	+	+		+			+
3	+	+		+	+		+	
4	+		+		+	+		+
5	+	+		+	+		+	

2. Реализовать логическую функцию, соответствующую вашему варианту, используя заданный тип ЛЭ (табл. 4), снять таблицу истинности ЛЭ или соединения ЛЭ (схемы), реализующих требуемую функцию.

Таблица 4

Номер бригады (варианта)	Функция, подлежащая реализации	Тип используемых ЛЭ
1	а) $y=\overline{x_1 \cdot x_2 \cdot x_3}$ б) $y=x_1 \oplus x_2 \oplus x_3$	4И-НЕ 2М2
2	а) $y=\overline{x_1 \cdot x_2}$ б) $y=x_1 \oplus x_2 \oplus x_3 \oplus x_4$	4И-НЕ 2М2
3	а) $y=\overline{x_1 \cdot x_2 \cdot x_3}$ б) $y=x_1 \oplus x_2 \oplus x_3$	2И-НЕ 2М2
4	а) $y=x_1 \cdot x_2 \cdot x_3$ б) $y=x_1 \oplus x_2 \oplus x_3 \oplus x_4$	2И 2М2
5	а) $y=x_1 + x_2 + x_3$ б) $y=x_1 \oplus x_2 \oplus x_3$	4ИЛИ 2М2

4. Контрольные вопросы

1. Какие операции можно выполнять с ЛЭ?
2. Что такое таблица истинности ЛЭ или устройства, осуществляющего некоторое логическое преобразование?
3. Укажите размерность таблицы истинности (число строк и число столбцов) ЛЭ: 4И и 2 ИЛИ.
4. Объясните, почему неиспользуемые входы ЛЭ «ИЛИ», «ИЛИ-НЕ» соединяют с корпусом (уровнем логического «0»), а на неиспользуемые входы ЛЭ «И», «И-НЕ» подается напряжение уровня логической «1».

5. ЛЭ каких типов соответствуют приведенным таблицам истинности?

x_1	0	0	1	1
x_2	0	1	0	1
y	0	1	1	0

а)

x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
y	1	1	1	1	1	1	1	0

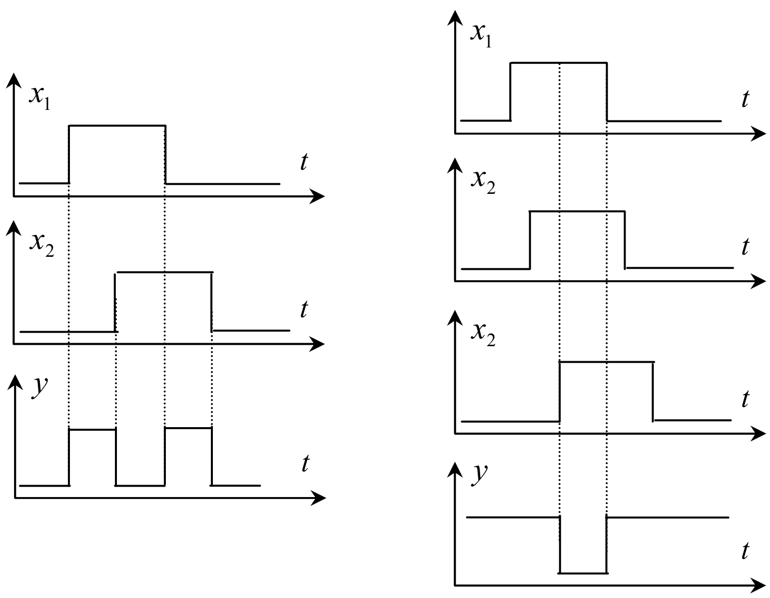
б)

6. Используя ЛЭ наборного поля, получите три различных варианта схем, реализующих логическую функцию «5И-НЕ». Который из них является наиболее оптимальным (рациональным)?

7. Какую логическую функцию реализует цепочка из К последовательно соединенных инверторов, если К – нечетное число, К – четное? Чему эквивалентны такие цепочки?

8. Изобразите временные диаграммы, характеризующие функционирование ЛЭ: НЕ, 3И, 3ИЛИ, 3И-НЕ, 3М2.

9. Запишите логические выражения и составьте таблицы истинности ЛЭ, которым соответствуют приведенные временные диаграммы:



а) *б)*

Лабораторная работа № 2

СИНТЕЗ КОМБИНАЦИОННЫХ УСТРОЙСТВ

Цель работы: синтез логического устройства с использованием метода минимизации логических функций типа «карт Карно».

1. Объект и средства исследования

Объектом исследования являются метод синтеза логических устройств (карты Карно), дискретные элементы и полученное комбинационное устройство, смоделированное в системе Electronics Workbench.

2. Теоретическая часть

Комбинационным цифровым устройством (КЦУ) называется устройство, выходные сигналы которого в некоторый момент времени работы однозначно определяются лишь сигналами, действующими в тот же момент времени на его входах. В КЦУ отсутствуют элементы памяти, поэтому выходные сигналы таких устройств формируются и сохраняются только в период действия входных.

КЦУ применяются для выполнения целого ряда логических и арифметических преобразований над входными сигналами и используются в качестве шифраторов, дешифраторов, сумматоров, мультиплексоров и других функциональных узлов.

В общем случае проектируемое КЦУ может быть представлено в виде черного ящика (ЧЯ), имеющего n входов и m выходов. Единственно, что изначально известно об этом ЧЯ – это требуемый алгоритм его функционирования, т. е. характер связи между входными воздействиями и выходными сигналами (реакциями). Проектирование сводится к определению оптимальной (в некотором смысле) структуры (схемы) КЦУ (ЧЯ), реализуемой в заданном базисе ЛЭ. Другими словами, проектирование КЦУ сводится к нахождению схемы КЦУ, удовлетворяющей требуемому алгоритму функционирования при двух следующих ограничениях: во-первых, схема КЦУ должна быть реализована с помощью ЛЭ заданного функционального полного набора; во-вторых, поскольку требуемый алгоритм функционирования в общем случае может быть реализован с помощью различных схем, то должна быть определена (выбрана) некоторая, в определенном смысле, наилучшая (оптимальная) схема, например схема, отличающаяся минимумом аппаратурных затрат, т. е. минимальным числом ЛЭ или ИС.

Процесс проектирования КЦУ в общем случае включает следующие этапы:

1. Словесное описание алгоритма функционирования КЦУ, т. е. описание работы устройства в понятийной форме (на обычном языке).
2. Оценка размерности задачи и решение вопроса о проектировании КЦУ в целом или по частям, чему предшествует разделение (условное) КЦУ на составные части. В отдельных случаях для снижения трудоемкости и громоздкости задачи проектирования КЦУ разбивается на ряд более простых устройств (узлов), в совокупности реализующих требуемый алгоритм функционирования, проектирование которых не составляет особых сложностей.
3. Переход от словесного к формализованному заданию алгоритма функционирования КЦУ с помощью логических (булевых) функций.
4. Минимизация логических функций.
5. Преобразование минимальных форм логических функций к виду, реализуемому ЛЭ заданного функционально полного набора для получения наименьшего количества электронного оборудования и рационального построения функциональной схемы устройства.
6. Построение схемы КЦУ по полученным логическим функциям (этапы 1 – 5).

На практике чаще всего используются следующие две исходные канонические формы представления функций: совершенная дизъюнктивная нормальная форма (СДНФ) и совершенная конъюнктивная нормальная форма (СКНФ). Необходимо также ввести следующие понятия:

– *минтерм* – элементарная конъюнкция, образуется логическим умножением переменных и их отрицаний:

$$f(x_1, x_2, x_3) = x_1 \cdot x_2 \cdot \overline{x_3}, \; f(z_1, z_2, z_3) = \overline{z_1} \cdot z_2 \cdot \overline{z_3};$$

– *макстерм* – элементарная дизъюнкция, образуется логическим сложением переменных и их отрицаний:

$$f(x_1, x_2, x_3) = x_1 \vee x_2 \vee \overline{x_3}, \; f(z_1, z_2, z_3) = \overline{z_1} \vee z_2 \vee \overline{z_3};$$

– *ранг* – число переменных образующих минтерм или макстерм:

$f(x_1, x_2, x_3) = x_1 \cdot x_2 \cdot \overline{x_3}$ – минтерм III ранга;

$f(z_1, z_2, z_3, z_4) = \overline{z_1} \vee z_2 \vee \overline{z_3} \vee z_4$ – макстерм IV ранга.

Если в состав логической формулы входят наборы аргументов одинакового ранга, то такая форма называется *совершенной*.

Совершенная дизъюнктивная нормальная форма (СДНФ)

Дизъюнктивной нормальной формой (ДНФ) называется такая форма представления функции, при которой логическое выражение функции строится в виде дизъюнкции ряда членов (минтермов), каждый из которых является простой конъюнкцией аргументов или их инверсий. Примером ДНФ может служить выражение

$$f(x_1, x_2, x_3) = x_1 \vee x_1 \cdot \overline{x_3} \vee \overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee x_2 \cdot x_3 \qquad (1)$$

Рассмотрим форму представления функции, не являющуюся ДНФ. Например, функция

$$f(x_1, x_2, x_3) = x_1 \vee x_1 \cdot \overline{x_3} \vee \overline{x_1} \cdot \overline{x_2} \cdot x_3 \vee \overline{x_2 \cdot x_3}$$

представлена не в ДНФ, так как последний член не является простой конъюнкцией аргументов.

Также не является ДНФ следующая форма представления функции:

$$f(x_1, x_2, x_3) = x_1 \cdot (x_2 \cdot \overline{x_3} \vee \overline{x_2} \cdot x_3) \vee x_2 \cdot x_3 .$$

Если в каждом члене ДНФ содержатся аргументы одинакового ранга, то такая форма называется *совершенной ДНФ* (СДНФ). Выражение (1) не является СДНФ, так как в нем лишь третий член содержит все аргументы функции.

Для перехода от ДНФ к СДНФ необходимо в каждый из членов, в которых представлены не все аргументы, ввести выражение вида $x_i \vee \overline{x_i}$, где x_i – отсутствующий в члене аргумент. Так как $x_i \vee \overline{x_i} = 1$, такая операция не может изменить значений функции. Рассмотрим переход от ДНФ к СДНФ на примере следующего выражения:

$$f(x_1, x_2, x_3) = x_1 \vee x_2 \cdot \overline{x_3}.$$

Добавление в члены выражений вида $x_i \vee \overline{x_i}$ приведет к функции

$$f(x_1, x_2, x_3) = x_1 \cdot (x_2 \vee \overline{x_2}) \cdot (x_3 \vee \overline{x_3}) \vee x_2 \cdot \overline{x_3} \cdot (x_1 \vee \overline{x_1}) =$$
$$= x_1 \cdot x_2 \cdot x_3 \vee x_1 \cdot x_2 \cdot \overline{x_3} \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot \overline{x_3} \vee x_1 \cdot x_2 \cdot \overline{x_3} \vee \overline{x_1} \cdot x_2 \cdot \overline{x_3}.$$

На основании свойств операции конъюнкции и дизъюнкции получим

$$x_1 \cdot x_2 \cdot \overline{x_3} \vee x_1 \cdot x_2 \cdot \overline{x_3} = x_1 \cdot x_2 \cdot \overline{x_3}.$$

Отсюда после приведения подобных членов

$$f(x_1, x_2, x_3) = x_1 \cdot x_2 \cdot x_3 \vee x_1 \cdot x_2 \cdot \overline{x_3} \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot \overline{x_3} \vee \overline{x_1} \cdot x_2 \cdot \overline{x_3},$$

т. е. имеем СДНФ.

Совершенная конъюнктивная нормальная форма (СКНФ)

Конъюнктивной нормальной формой (КНФ) называется форма представления функции в виде конъюнкции ряда членов (макстермов), каждый из которых является простой дизъюнкцией аргументов (или их инверсий).

Примером КНФ может служить следующая форма представления функции:

$$f(x_1, x_2, x_3) = x_1 \cdot (x_2 \vee \overline{x_3}) \cdot (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \cdot (x_2 \vee x_3).$$

Рассмотрим формы представления функций, не являющиеся КНФ:

$$f(x_1, x_2, x_3) = x_1 \cdot (x_2 \vee \overline{x_3}) \cdot (\overline{\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}}) \cdot (x_2 \vee x_3)$$

(здесь третий член не является простой дизъюнкцией аргументов или их инверсий);

$$f(x_1, x_2, x_3) = x_1 \vee (x_2 \vee \overline{x_3}) \cdot (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \cdot (x_2 \vee x_3)$$

(эта форма также не является КНФ, так как в ней первый член не связан с остальными операциями конъюнкции).

В СКНФ в каждом члене КНФ должны быть представлены все аргументы. Для перехода от КНФ к СКНФ необходимо добавить к каждому члену, не содержащему всех аргументов, члены вида, где x_i – аргумент, не представленный в члене. Так как $x_i \cdot \overline{x_i} = 0$, то такая операция не может повлиять на значение функции. Дальнейшие преобразования происходят аналогично СДНФ.

Табличная форма задания функции

Если исходная функция задана в табличной форме, то, например, СДНФ может быть получена непосредственно.

Таблица 5

Значение функции	Номер набора							
	1	2	3	4	5	6	7	8
x_1	0	0	0	0	1	1	1	1
x_2	0	0	1	1	0	0	1	1
x_3	0	1	0	1	0	1	0	1
$f(x_1, x_2, x_3)$	0	0	1	1	0	1	0	1

Пусть задана функция в форме табл. 5. Для этой функции СДНФ имеет вид

$$f(x_1, x_2, x_3) = \overline{x_1} \cdot x_2 \cdot \overline{x_3} \vee \overline{x_1} \cdot x_2 \cdot x_3 \vee x_1 \cdot \overline{x_2} \cdot x_3 \vee x_1 \cdot x_2 \cdot x_3. \quad (2)$$

Каждый член в (2) соответствует некоторому набору значений аргументов, при котором функция $f(x_1, x_2, x_3)$ равна 1. Каждый из наборов аргументов, при которых функция $f(x_1, x_2, x_3)$ равна 1 (3, 4, 6, 8-й столбцы наборов), обращает в единицу соответствующий член выражения (2), вследствие чего и вся функция оказывается равной единице.

Можно сформулировать следующее правило записи СДНФ функции, заданной таблицей истинности. Необходимо записать столько членов в виде конъюнкций всех аргументов, сколько единиц содержит функция в таблице. Каждая конъюнкция должна соответствовать определенному набору значений аргументов, обращающему функцию в единицу, и если в этом наборе значение аргумента равно нулю, то в конъюнкцию входит инверсия данного аргумента.

Следует отметить, что любая функция имеет единственную СДНФ.

Структурная схема логического устройства может быть построена непосредственно по канонической форме (СДНФ или СКНФ) реализуемой функции.

Получающаяся при этом логическая схема функции (2) показана на рис. 12.

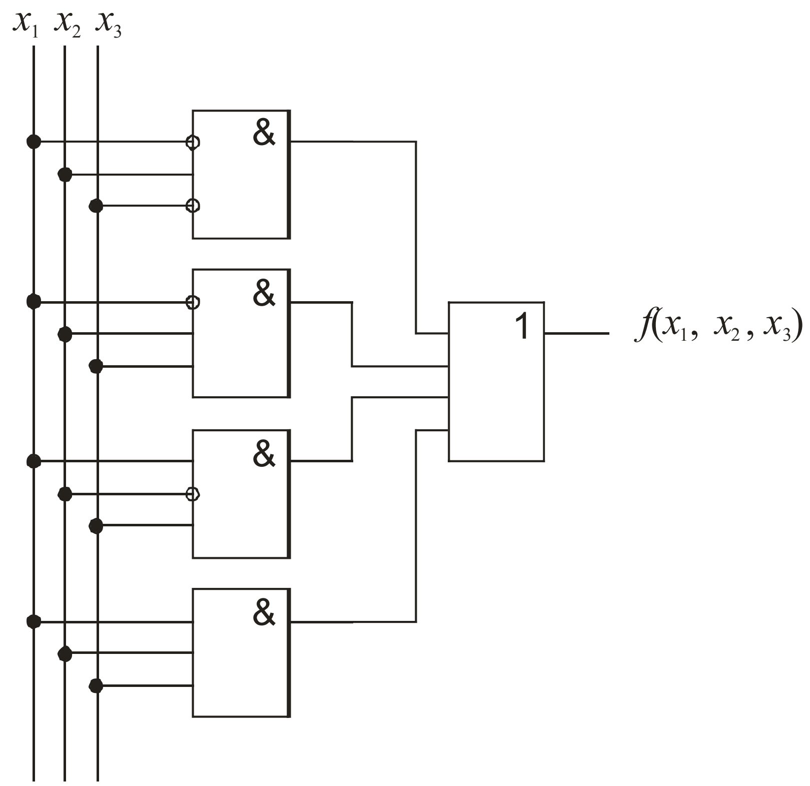


Рис. 12. Логическая схема комбинационного устройства

Недостаток такого метода построения структурных схем, обеспечивающего в общем правильное функционирование устройства, состоит в том, что получающиеся схемы чаще всего неоправданно сложные, требуют использования большого числа логических элементов, имеют низкие экономичность и надежность. Во многих случаях удается так упростить логическое выражение, не изменив функции, что соответствующая структурная схема оказывается существенно более простой. Методы такого упрощения функции называются методами *минимизации функций.*

Компактная форма таблицы задания функции

В табл. 6 представлена одна из форм таблицы истинности некоторой сложной функции четырех аргументов. При n аргументах число наборов их значений составляет 2^n, и с ростом n быстро увеличивается число столбцов в таблице. При больших n таблица становится весьма громоздкой и неудобной для использования.

Таблица 6

Значение функции	Номер подбора															
x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
x_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
x_1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x_1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$f(x_1, x_2, x_3, x_4)$	1	0	0	1	0	1	1	0	0	0	0	1	1	0	1	1

Для обеспечения большей компактности часто отдают предпочтение другой форме таблицы истинности (показана в табл. 7 для функции четырех аргументов). Таблица строится следующим образом. Все аргументы функции делятся на две группы. Столбцам и строкам таблицы приписывают комбинации значений аргументов одной и другой группы. В клетках, расположенных на пересечении столбцов и строк, записываются соответствующие значения функции.

Таблица 7

$x_1 \cdot x_2$ / $x_3 \cdot x_4$	00	01	11	10
00	1	0	0	1
01	0	1	0	0
11	0	1	0	1
10	1	0	1	1

Последовательности комбинаций значений аргументов, приписываемых столбцам и строкам таблицы, соответствуют последовательности чисел в так называемом *коде Грея*. Числа в коде Грея можно получить из двоичных чисел путем их сложения по модулю 2 (mod 2) с теми же числами, сдвинутыми на один разряд вправо. Например, представление двоичного числа 1101 в коде Грея получается следующим образом:

	1 1 0 1	двоичное число
$\oplus$	1 1 0 $\rightarrow$ 1	сдвинутое вправо число
	1 0 1 1	число в коде Грея

Минимизация функций с использованием карт Карно

Аргументы функции делятся на две группы, комбинации значений аргументов одной группы приписываются столбцам таблицы, а другой – строкам таблицы. Столбцы и строки обозначаются комбинациями, соответствующими последовательности чисел в коде Грея (это сделано для того, чтобы склеивающиеся клетки находились рядом). Обозначения столбца и строки, на пересечении которых находится клетка таблицы, образуют набор, значение функции на этом наборе записывается в клетку.

Для получения МДНФ (минимальной ДНФ) функции охватываются областями (контуром) клетки таблицы, содержащие 1. Контуры должны быть прямоугольной формы и содержать 2^k клеток (при целочисленном значении k). Для каждого контура составляется набор из двух комбинаций: приписанных столбцам и приписанных строкам, на пересечении которых расположена область. При этом если контуру соответствуют несколько комбинаций кода Грея, приписанных столбцам или строкам, то при составлении набора области записывается общая часть этих комбинаций, а на месте различающихся разрядов комбинаций ставятся звездочки. Например, для функции, представленной табл. 8, контуру I будет соответствовать набор 1*00 или член МДНФ $K_{\mathrm{I}} = x_1 \cdot \overline{x_3} \cdot \overline{x_4}$, области II – набор 0**1 или член МДНФ $K_{\mathrm{II}} = \overline{x}_1 \cdot x_4$.Таким образом, для этой функции МДНФ

$$K = K_{\mathrm{I}} + K_{\mathrm{II}} \text{ или } f(x_1, x_2, x_3, x_4) = x_1 \cdot \overline{x_3} \cdot \overline{x_4} \vee \overline{x_1} \cdot x_4 .$$

Для получения МКНФ областями охватываются клетки, содержащие 0, и члены МКНФ записываются через инверсии цифр, получаемых для наборов отдельных областей. Так, для функции, представленной в табл. 9, области I соответствует набор * 100 и член МКНФ $\overline{x_2} \vee x_3 \vee x_4$, области II – набор 0* 1 * и член $x_1 \vee \overline{x_3}$. Таким образом, МКНФ функции

$$f(x_1, x_2, x_3, x_4) = (\overline{x_2} \vee x_3 \vee x_4) \cdot (x_1 \vee \overline{x_3}).$$

Таблица 8

$x_1 \cdot x_2$ / $x_3 \cdot x_4$	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	0	0

II I

Таблица 9

$x_1 \cdot x_2$ / $x_3 \cdot x_4$	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	0	0	1	1
10	0	0	1	1

II I

3. Порядок выполнения работы

1. Преобразовать таблицу в соответствии с вариантом бригады (табл. 10 – 14) к компактному виду.

Таблица 10

Бригада 1

x_1	1	0	0	1	1	1	1	0	1	0	0
x_2	0	0	1	1	0	0	1	1	0	0	0
x_3	1	0	0	0	0	1	1	1	0	0	0
x_4	1	1	1	1	0	1	1	1	0	0	1
x_5	1	0	1	0	0	1	1	0	0	0	0
x_6	0	1	1	1	0	0	1	1	0	0	1

Таблица 11

Бригада 2

x_1	0	0	1	0	0	1	0	0	1	1	1
x_2	1	1	1	0	1	1	0	1	1	0	1
x_3	0	0	0	1	0	0	0	1	1	1	0
x_4	0	1	0	1	1	0	0	1	0	0	1
x_5	1	1	0	1	0	1	0	0	1	0	1
x_6	1	1	1	0	0	0	1	1	0	1	1

Таблица 12

Бригада 3

x_1	1	0	0	0	0	0	1	0	1	0	0
x_2	1	0	1	0	0	1	0	0	0	1	1
x_3	0	1	1	1	1	0	1	1	1	1	0
x_4	1	0	1	1	1	0	1	1	1	1	1
x_5	1	1	1	1	1	0	0	1	1	0	0
x_6	1	1	0	0	1	0	0	0	1	0	0

Таблица 13

Бригада 4

x_1	1	1	0	0	0	0	0	0	0	1	1
x_2	0	1	0	1	1	0	1	1	0	1	1
x_3	0	0	0	0	1	1	1	0	1	1	1
x_4	1	0	1	1	0	0	0	1	1	1	1
x_5	0	1	1	1	0	0	0	1	1	1	1
x_6	0	0	1	0	0	0	1	0	0	0	0

Таблица 14

Бригада 5

x_1	0	1	0	1	0	0	0	1	0	1	0
x_2	1	1	0	0	0	1	0	0	1	1	0
x_3	0	0	0	0	1	0	1	1	0	0	0
x_4	0	0	1	1	0	0	0	1	0	1	1
x_5	0	1	0	1	0	1	0	0	1	1	0
x_6	1	0	0	0	1	0	1	0	1	1	1

Примечание. Для всех вариантов значение функции выхода $f(x_1, x_2, x_3, x_4, x_5, x_6) = 1$.

2. Для получения МДНФ минимизировать логическую функцию методом карт Карно.

3. По полученному логическому выражению составить логическую схему и построить ее в системе Electronics Workbench.

4. Проверить правильность работы схемы основываясь на исходных данных табл. 10.

5. Оформить отчет, содержащий помимо стандартных требований табл. 10 (для варианта бригады), компактную таблицу истинности, описание процесса минимизации логической функции, минимальную дизъюнктивную нормальную форму, логическую схему синтезированного комбинационного устройства.

4. Контрольные вопросы

1. Что такое минтерм, макстерм, КНФ, ДНФ?
2. Правила преобразования нормальных форм функций в совершенные.
3. Правила образования компактных форм таблиц истинности.
4. Правила перевода чисел в различные системы счисления, в коды Грея.

Лабораторная работа № 3

ИССЛЕДОВАНИЕ РАБОТЫ СЧЕТЧИКА И ДЕШИФРАТОРА

Цель работы: изучение принципов работы реверсивного счетчика и дешифратора на примере автомата эффектов типа «бегущий огонь».

1. Объект и средства исследования

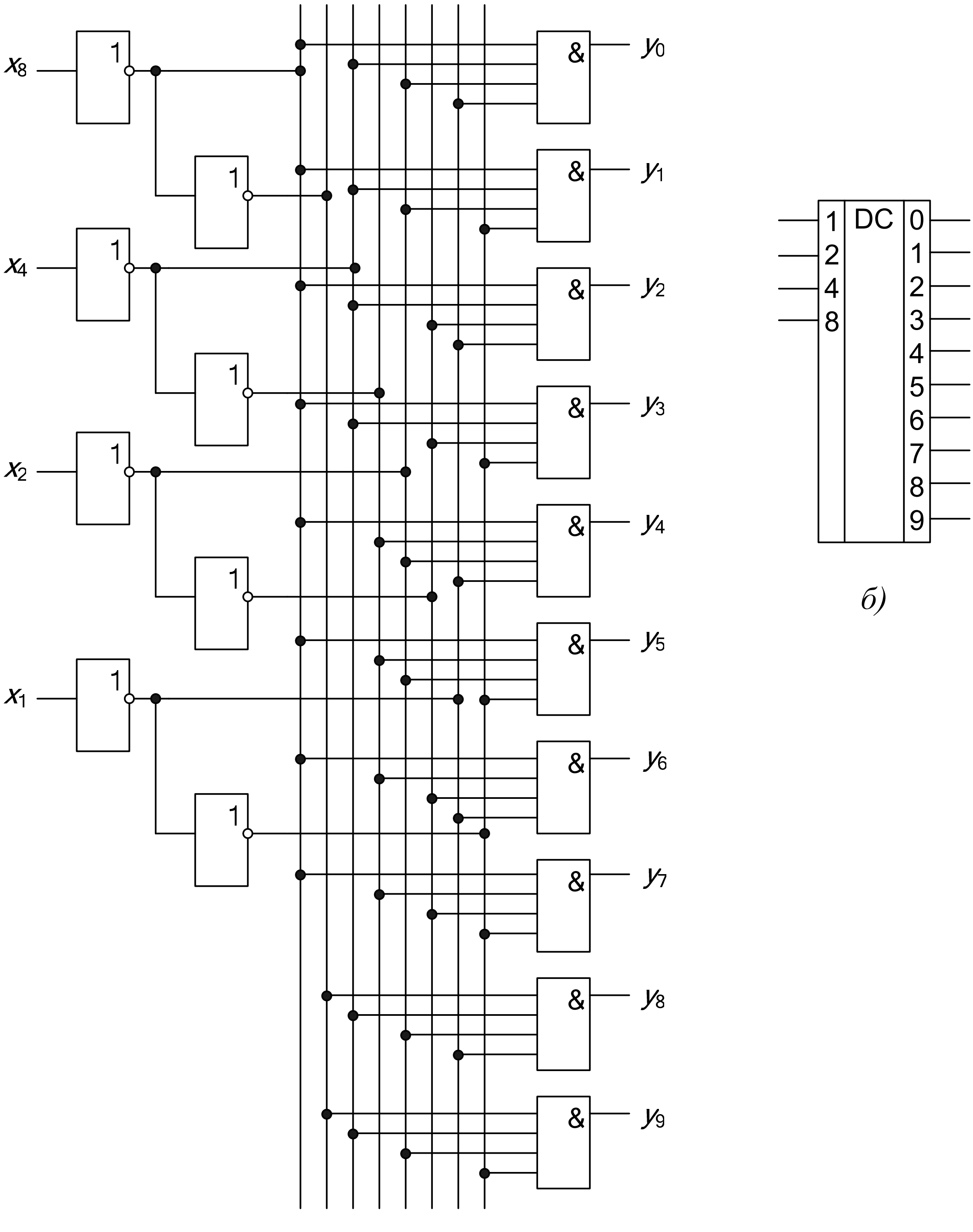
Объектом исследования являются четырехразрядный реверсивный счетчик, дешифратор, генератор.

2. Теоретическая часть

Для преобразования двоичных чисел в небольшие по значению десятичные числа используются *дешифраторы* (называемые также декодерами). Входы дешифратора предназначаются для подачи двоичных чисел, выходы последовательно нумеруются десятичными числами. При подаче на входы двоичного числа появляется сигнал на определенном выходе, номер которого соответствует входному числу.

На рис. 13, *а* показан дешифратор в развернутом виде, состоящий из логических элементов «И». Символическое изображение дешифратора представлено на рис. 13, *б*. Символ DC образован из букв английского слова Decoder. Слева показаны входы, на которых отмечены весовые коэффициенты двоичного кода, справа – выходы, пронумерованные десятичными числами, соответствующие отдельным комбинациям входного двоичного кода. На каждом выходе образуется уровень лог. 1 при строго определенной комбинации входного кода.

По способу построения различают *линейные* и *прямоугольные* дешифраторы.



а) *б)*

Рис. 13. Дешифратор: а – в развернутом виде; б – условное обозначение

В данной работе используется линейный дешифратор осуществляющий преобразование, заданное табл. 15.

Таблица 15

Выходной код 8421				Номер выхода (в десятичной системе)
x_8	x_4	x_2	x_1	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Значения выходных переменных определяются следующими логическими выражениями:

В линейном дешифраторе выходные переменные формируются по (3):

$$\begin{aligned}
y_0 &= \overline{x_8}\cdot\overline{x_4}\cdot\overline{x_2}\cdot\overline{x_1}, & y_5 &= \overline{x_8}\cdot x_4\cdot\overline{x_2}\cdot x_1,\\
y_1 &= \overline{x_8}\cdot\overline{x_4}\cdot\overline{x_2}\cdot x_1, & y_6 &= \overline{x_8}\cdot x_4\cdot x_2\cdot\overline{x_1},\\
y_2 &= \overline{x_8}\cdot\overline{x_4}\cdot x_2\cdot\overline{x_1}, & y_7 &= \overline{x_8}\cdot x_4\cdot x_2\cdot x_1,\\
y_3 &= \overline{x_8}\cdot\overline{x_4}\cdot x_2\cdot x_1, & y_8 &= x_8\cdot\overline{x_4}\cdot\overline{x_2}\cdot\overline{x_1},\\
y_4 &= \overline{x_8}\cdot x_4\cdot\overline{x_2}\cdot\overline{x_1}, & y_9 &= x_8\cdot\overline{x_4}\cdot\overline{x_2}\cdot x_1.
\end{aligned} \tag{3}$$

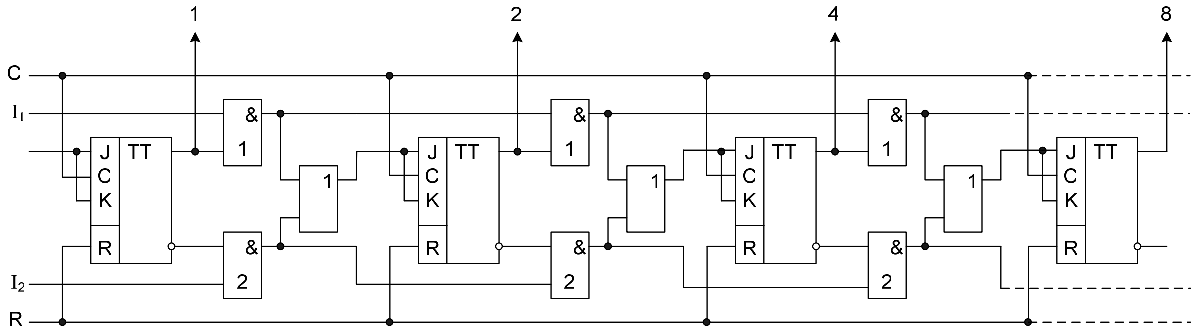
Структура дешифратора имеет особенности, характерные для дешифраторов в интегральном исполнении:

– для уменьшения числа входов формирование инверсий входных переменных осуществляется в самом дешифраторе;

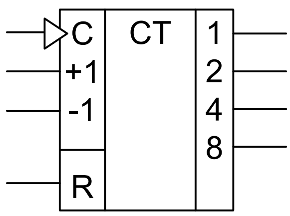
– подключенные непосредственно к входам дополнительные инверторы уменьшают нагрузку со стороны дешифратора на его входные цепи.

Счетчик – это цифровое устройство, определяющее, сколько раз на его входе появился некоторый определенный логический уровень. В дальнейшем во всех случаях, когда это не оговаривается специально, будем полагать, что счетчик подсчитывает содержащиеся во входном сигнале переходы с уровня лог. 0 к уровню лог. 1. При входном сигнале, имеющем форму последовательности импульсов, счетчик ведет счет поступающих на вход импульсов. Числа в счетчике представляются некоторыми комбинациями состояний триггеров. При поступлении на вход очередного уровня лог. 1 в счетчике устанавливается новая комбинация состояний триггеров, соответствующая числу, на единицу большему предыдущего числа. Таким образом, счетчик представляет собой логическое устройство последовательностного типа, в котором новое состояние определяется предыдущим состоянием и значением логической переменной на входе.

Реверсивный счетчик (рис. 14) допускает в процессе работы переключение из режима суммирования в режим вычитания и наоборот.



а)



б)

Рис. 14. Реверсивный счетчик:
а – в развернутом виде; б – условное обозначение

В схеме такого счетчика предусмотрены две цепи передачи переносов, одна из которых соответствует схеме суммирующего счетчика, другая – схеме вычитающего счетчика. Управляющие сигналы I_1 и I_2 включают в работу одну или другую цепь.

При $I_1 = 1$ и $I_2 = 0$ оказываются закрытыми элементы лог. «И» № 2 и, следовательно, отключена цепь передачи переносов режима вычитания. Счетчик работает в режиме суммирования. При $I_1 = 0$ и $I_2 = 1$ закрыты элементы лог. «И» № 1 и отключена, таким образом, цепь передачи переносов режима суммирования, счетчик работает в режиме вычитания.

Автомат эффектов «бегущий огонь»

Существует несколько вариантов реализации автомата подобного типа. В данной работе он выполнен на основе реверсивного счетчика и дешифратора (рис. 15).

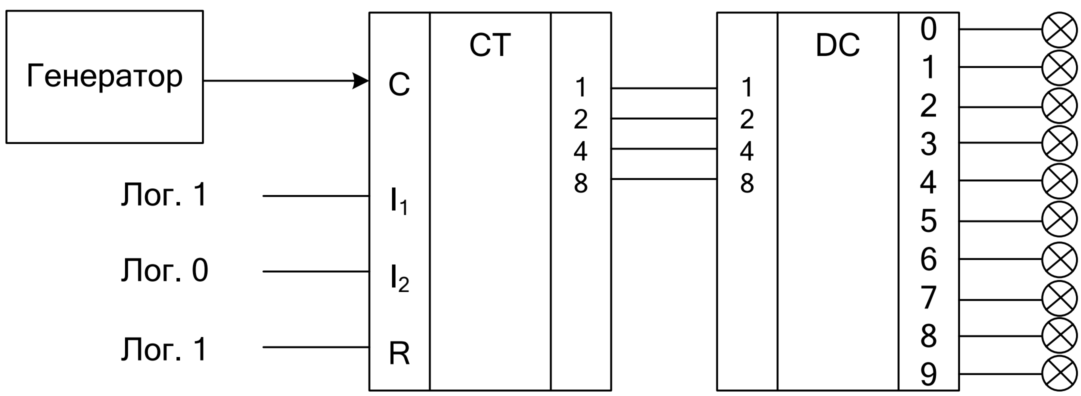


Рис. 15. Схема автомата эффектов «бегущий огонь»

Задающий генератор вырабатывает прямоугольные импульсы, количество которых подсчитывает счетчик. Дешифратор преобразует двоичный код на выходе счетчика в десятичный. Лампы загораются последовательно одна за одной, что создает эффект «бегущего огня».

3. Порядок выполнения работы

1. Построить схему реверсивного счетчика импульсов в системе Electronics Workbench (см. рис. 14) и исследовать его работу.

2. Не удаляя предыдущей схемы, построить в рабочей области Electronics Workbench дешифратор (см. рис. 13, *а*). Исследовать его работу.

3. Соединить дешифратор со счетчиком по следующей схеме типа «Автомат эффектов “бегущий огонь”» (см. рис. 15). Исследовать его работу.

4. Построить «Автомат эффектов “бегущий огонь”», обнуляемый на 8-м импульсе счета.

5. Построить «Автомат эффектов “бегущий огонь”» с реверсом направления счета.

6. Оформить отчет, содержащий пять схем: счетчик импульсов (в развернутом виде), дешифратор (в развернутом виде) и «Автомат эффектов “бегущий огонь”», «Автомат эффектов “бегущий огонь”» с обратной связью, «Автомат эффектов “бегущий огонь”» с реверсом счета (в виде условных обозначений элементов).

4. Контрольные вопросы

1. Типы триггеров, принципы их работы.

2. Триггеры с двухступенчатым запоминанием информации.

3. Шифраторы и дешифраторы, основные типы и принципы их работы.

4. Счетчики. Складывающие и вычитающие счетчики. Реверсивные счетчики.

5. Преобразователи кодов.

Лабораторная работа № 4

ИССЛЕДОВАНИЕ РАБОТЫ ДВОИЧНОГО СУММАТОРА

Цель работы: изучение принципов работы двоичного сумматора, шифратора, буферного регистра и семисегментного индикатора.

1. Объект и средства исследования

Объектом исследования являются двоичный сумматор, шифратор, буферный регистр и семисегментный индикатор.

2. Теоретическая часть

Правила выполнения арифметических операций

Арифметические действия (операции) относятся к числу наиболее распространенных операций, выполняемых цифровыми устройствами (ЦУ).

Правила выполнения арифметических операций над двоичными числами аналогичны соответствующим правилам десятичной арифметики и сведены в табл. 16.

Таблица 16

Двоичное сложение			
Слагаемые k-го разряда	Сумма k -го разряда	Перенос в k +1-й разряд	Пример
0 + 0 =	0	0	1100 – перенос
0 + 1 =	1	0	+ 1101 – 1-е слагаемое
1 + 0 =	1	0	1100 – 2-е слагаемое
1 + 1 =	0	1	11001 – сумма

Окончание табл. 16

Двоичное вычитание							
Уменьшаемое k-го разряда		Вычитаемое k-го разряда		Разность k-го разряда	Заем из в k +1-й разряда		Пример
0	–	0	=	0	0		010 – заем
0	–	1	=	1	1	–	1101 – уменьшаемое
1	–	0	=	1	0		1010 – вычитаемое
1	–	1	=	0	0		0011 – разность

Двоичное умножение						
Множимое k-го разряда		Множитель k-го разряда		Произведение k-го разряда		Пример
0	×	0	=	0	×	1010 – множимое 101 – множитель
0	×	1	=	0		1010
1	×	0	=	0	+	0000
1	×	1	=	1	+	1010
						110010 – произведение

Двоичное деление					
Делимое k-го разряда		Делитель k-го разряда		Частное k-го разряда	Пример
0	:	0	=	?	–1111 | 10 10 111 – частное
0	:	1	=	0	–11 10
1	:	0	=	?	–11 10
1	:	1	=	1	1 – остаток

Для выполнения арифметических операций над двоичными числами со знаком вводят дополнительный (знаковый) разряд, который указывает, является ли число положительным или отрицательным. Если число положительное, в знаковый разряд проставляется символ 0, если отрицательное, то в знаковый разряд проставляется символ 1. Например, число (+ 5) с учетом знакового разряда (отделяется точкой) запишется как 0.101, а число (– 3) – как 1.011.

При сложении чисел с одинаковыми знаками числа складываются и сумме присваивается код знака слагаемых, например:

$$+\begin{array}{r}3\\2\\\hline 5_{10}\end{array} \Rightarrow +\begin{array}{r}0.011\\0.010\\\hline 0.101_2\end{array} \qquad\qquad +\begin{array}{r}-3\\-2\\\hline -5_{10}\end{array} \Rightarrow +\begin{array}{r}0.011\\0.010\\\hline 0.101_2\end{array}$$

Несколько усложняется операция сложения чисел с разными знаками (алгебраическое сложение), что равносильно вычитанию чисел. В этом случае необходимо определить большее по модулю число, произвести вычитание и присвоить разности знак большего (по модулю) числа.

Для упрощения выполнения этой операции слагаемые представляются в обратном или дополнительном кодах, поскольку известно, что операция вычитания (алгебраического сложения) сводится к операции простого арифметического сложения двоичных чисел, представленных в обратном или дополнительном кодах. Положительные числа в прямом, обратном и дополнительном кодах имеют один и тот же вид, а отрицательные – различный.

Чтобы представить отрицательное двоичное число в обратном коде, надо поставить в знаковый разряд 1, а во всех остальных разрядах прямого кода заменить единицы нулями, а нули – единицами, т. е. проинвертировать число.

При записи отрицательного двоичного числа в дополнительном коде, надо поставить 1 в знаковый разряд, а остальные разряды получить из обратного кода числа, прибавлением 1 к младшему разряду.

Приведем примеры записи двоичных чисел со знаками в прямом, обратном и дополнительном кодах.

Число	Прямой код	Обратный код	Дополнительный код
+6	0.110	0.110	0.110
-5	1.101	1.010	1.011
-11	1.1011	1.0100	1.0101

Поясним процедуру вычитания чисел 5 и 3 и 3 и 5. Последовательность и взаимосвязь операций представлена в табл. 17.

Таблица 17

Операция	Обратный код	Дополнительный код
5 - 3 = 5 + (- 3) = 2	$x_1 = 0.101$ $x_2 = 1.100$ $+\begin{array}{r}0.101\\ \underline{1.100}\\ 10.001\end{array}$ 0.010 Перенос в младший разряд. Сумма положительна	$x_1 = 0.101$ $x_2 = 1.101$ $+\begin{array}{r}0.101\\ \underline{1.101}\\ 10.010\end{array}$ 0.010 Единица переноса в младший разряд игнорируется. Сумма положительна
3 - 5 = 3 + (- 5) = - 2	$x_1 = 0.011$ $x_2 = 1.010$ $+\begin{array}{r}0.011\\ \underline{1.010}\\ 1.101\end{array}$ Перенос в младший разряд отсутствует. Сумма отрицательна и представлена в обратном коде.	$x_1 = 0.011$ $x_2 = 1.011$ $+\begin{array}{r}0.011\\ \underline{1.011}\\ 1.110\end{array}$ Сумма отрицательна и представлена в дополнительном коде.

Из приведенных примеров следует, что при использовании обратного кода в устройстве, обеспечивающем суммирование многоразрядных двоичных чисел – двоичном сумматоре, необходимо предусмотреть цепь циклического переноса. В случае использования дополнительного кода эта цепь отсутствует.

Из приведенного выше можно сделать следующее заключение. В ЦУ (в компьютере, в частности) нет надобности использовать два специализированных вычислительных устройства, одно из которых – двоичный сумматор, а другое – двоичный вычитатель. Оказывается, что применение простого математического «трюка» (представление двоичных чисел в обратном или дополнительном коде) позволяет приспособить двоичный сумматор для выполнения как операций сложения двоичных чисел, так и операций их вычитания.

Более того, с помощью двоичного сумматора можно обеспечить также выполнение и операций умножения и деления двоичных чисел (т. е. всех четырех арифметических действий), поскольку умножение представляет собой последовательное сложение, а деление – последовательное вычитание. Примеры выполнения этих операций приведены в табл. 18.

Таблица 18

Операция	Процедура	Последовательность основных микроопераций
5 × 3 = 15	101 011 101 1010 00000 01111	Суммирование Сдвиг суммы Сдвиг
15 : 3 = 5	0.1111 | 11 1.01 | 101 0.00 0.01 1.01 1.10 0.01 0.11 1.01 0.00	Суммирование в дополнительном коде Сумма положительна – запись единицы Сдвиг остатка Сумма отрицательна – запись нуля Восстановление остатка Сдвиг остатка Запись единицы, конец вычислений

Двоичные сумматоры

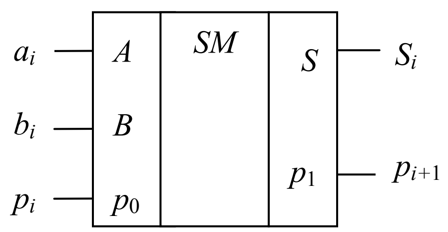
Суммирование многоразрядных двоичных чисел $A = a_n a_{n\text{-}1} \ldots a_0$ и $B = b_n b_{n\text{-}1} \ldots b_0$ производится путем их поразрядного сложения с переносом между разрядами. Поэтому основным узлом многоразрядных сумматоров является комбинационный одноразрядный сумматор, который выполняет арифметическое сложение трех одноразрядных чисел (цифр): цифры данного разряда первого слагаемого (a_i), цифры данного разряда второго слагаемого (b_i) и цифры (1 или 0) переноса из соседнего младшего разряда (p_i). В результате сложения для каждого разряда получаются две цифры – сумма для этого разряда (S_i) и перенос в следующий старший разряд (p_{i+1}).

Условное графическое изображение одноразрядного сумматора и его таблица истинности (функционирования) приведены на рис. 16.

Для синтеза схемы одноразрядного сумматора запишем выражения для S_i и p_{i+1} (выходов сумматора):

$$S_i = a_i\bar{b}_i\bar{p}_i + \bar{a}_i b_i\bar{p}_i + \bar{a}_i\bar{b}_i p_i + a_i b_i p_i = (a_i\bar{b}_i + \bar{a}_i b_i)\bar{p}_i + (\bar{a}_i\bar{b}_i + a_i b_i)p_i = \\ = (a_i \oplus b_i)\bar{p}_i + \overline{(a_i \oplus b_i)}p_i = (a_i \oplus b_i) \oplus p_i; \quad (4)$$

$$p_{i+1} = a_i b_i\bar{p}_i + a_i\bar{b}_i p_i + \bar{a}_i b_i p_i + a_i b_i p_i = a_i b_i(\bar{p}_i + p_i) + a_i\bar{b}_i p_i + \bar{a}_i b_i p_i = a_i b_i + p_i(a_i \oplus b_i). \quad (5)$$



а)

a_i	b_i	p_i	S_i	p_{i+1}
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
1	1	0	0	1
0	0	1	1	0
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

б)

Рис. 16. Одноразрядный сумматор: а – условное обозначение; б – таблица истинности

Схема одноразрядного сумматора, построенная в соответствии с выражениями (4) и (5), приведена на рис. 17.

Многоразрядный параллельный сумматор может быть составлен из одноразрядных сумматоров, число которых равно числу разрядов слагаемых, путем соединения выхода, на котором формируется сигнал переноса данного разряда, с входом для сигнала переноса соседнего старшего разряда. Такой способ организации переноса называется последовательным.

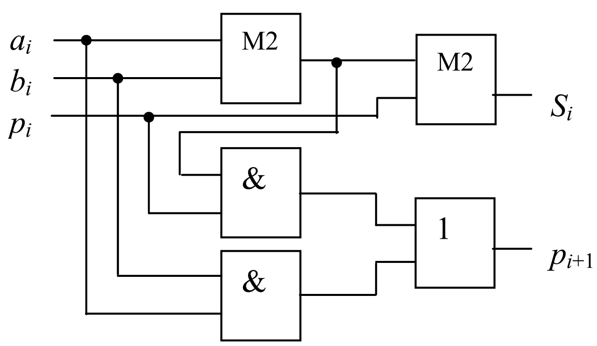


Рис. 17. Схема одноразрядного сумматора

Пример построения трехразрядного параллельного сумматора демонстрирует рис. 18. В сумматорах этого типа перенос распространяется последовательно от разряда к разряду по мере образования суммы в каждом разряде. При наиболее неблагоприятных условиях переноса, например при сложении чисел 11…11 и 00…01, будет иметь место «пробег» единицы переноса через весь сумматор от самого младшего к самому старшему разряду. Поэтому в наихудшем случае время распространения переноса

$$T_{\text{зд.р.пер}} = nt_{\text{зд.р.пер}},$$

где $t_{\text{зд.р.пер}}$ – время задержки распространения переноса в одном разряде;

n – число разрядов сумматора.

Данный тип сумматора наиболее прост с точки зрения схемы цепей распространения переноса, но имеет сравнительно низкое быстродействие.

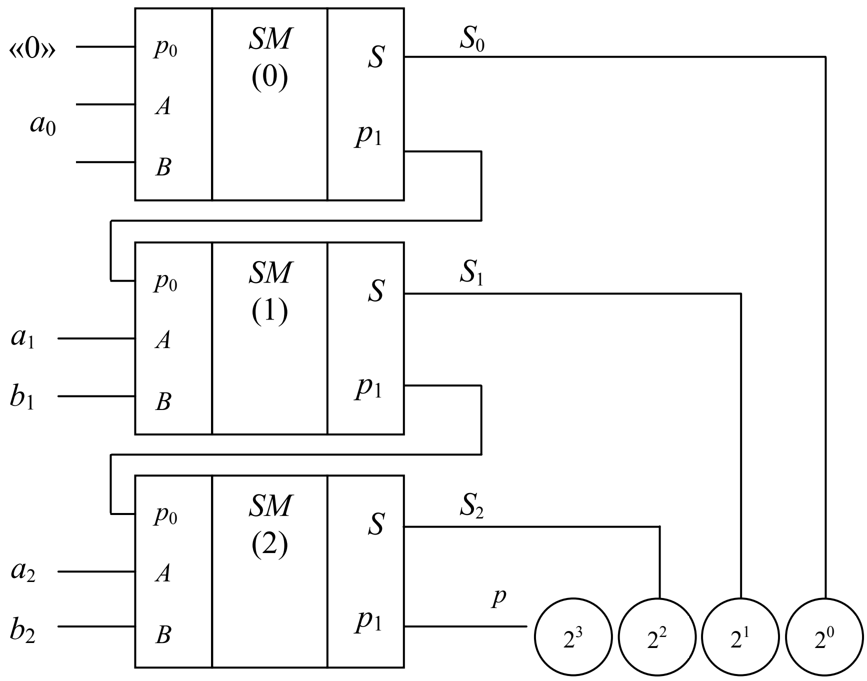


Рис. 18. Функциональная схема трехразрядного параллельного сумматора с последовательным переносом

Более высоким быстродействием обладают сумматоры с параллельным переносом, в которых сигналы переноса формируются во всех разрядах одновременно. Этой цели служат специальные схемы ускоренного переноса.

Шифратор

Шифратор (называемый также кодером) осуществляет преобразование десятичных чисел в двоичную систему счисления.

Шифраторы широко используются в разнообразных устройствах ввода информации в цифровые системы. Такие устройства могут снабжаться клавиатурой, каждая клавиша которой связана с определенным входом шифратора. При нажатии выбранной клавиши подается сигнал на соответствующий вход шифратора и на его выходе возникает двоичное число, соответствующее выгравированному на клавише символу.

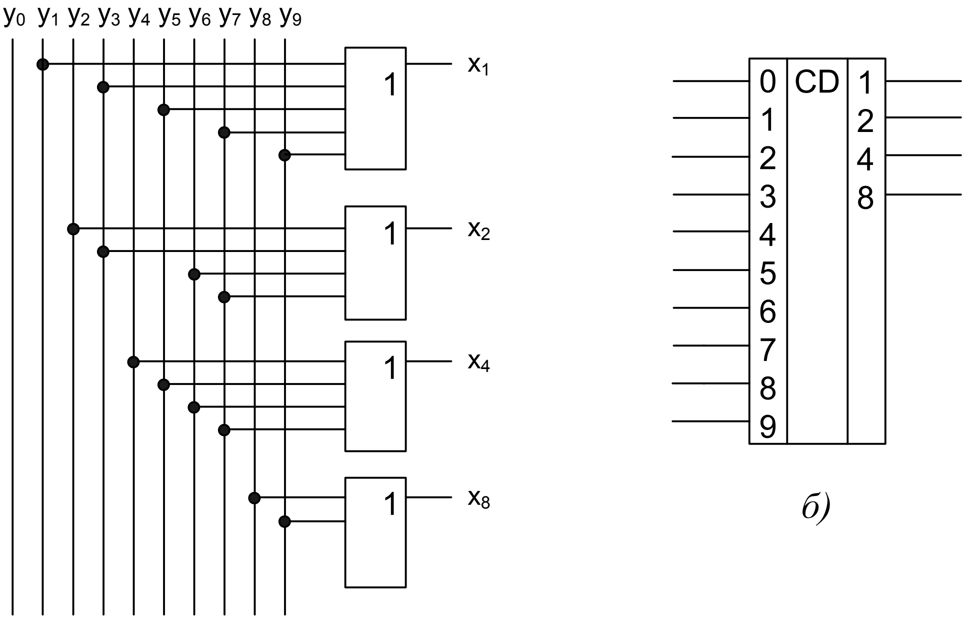
На рис. 19 приведен шифратор, преобразующий десятичные числа 0, 1, 2 ... 9 в двоичное представление в коде 8421. Символ CD образован из букв, входящих в английское слово Coder. Слева показаны 10 входов, обозначенных десятичными цифрами 0, 1, 2 ... 9, справа – выходы шифратора; цифрами 1, 2, 4, 8 обозначены весовые коэффициенты двоичных разрядов, соответствующих отдельным выходам.

Работа шифратора может быть описана аналитически (6) или таблично (табл. 19).

$$\begin{aligned} x_1 &= \overline{y_1} \vee \overline{y_3} \vee \overline{y_5} \vee \overline{y_7} \vee \overline{y_9}, \\ x_2 &= \overline{y_2} \vee \overline{y_3} \vee \overline{y_6} \vee \overline{y_7}, \\ x_4 &= \overline{y_4} \vee \overline{y_5} \vee \overline{y_6} \vee \overline{y_7}, \\ x_8 &= \overline{y_8} \vee \overline{y_9}. \end{aligned} \qquad (6)$$

Таблица 19

Выходной код 8421				Номер входа (в десятичной системе)
x_8	x_4	x_2	x_1	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9



б)

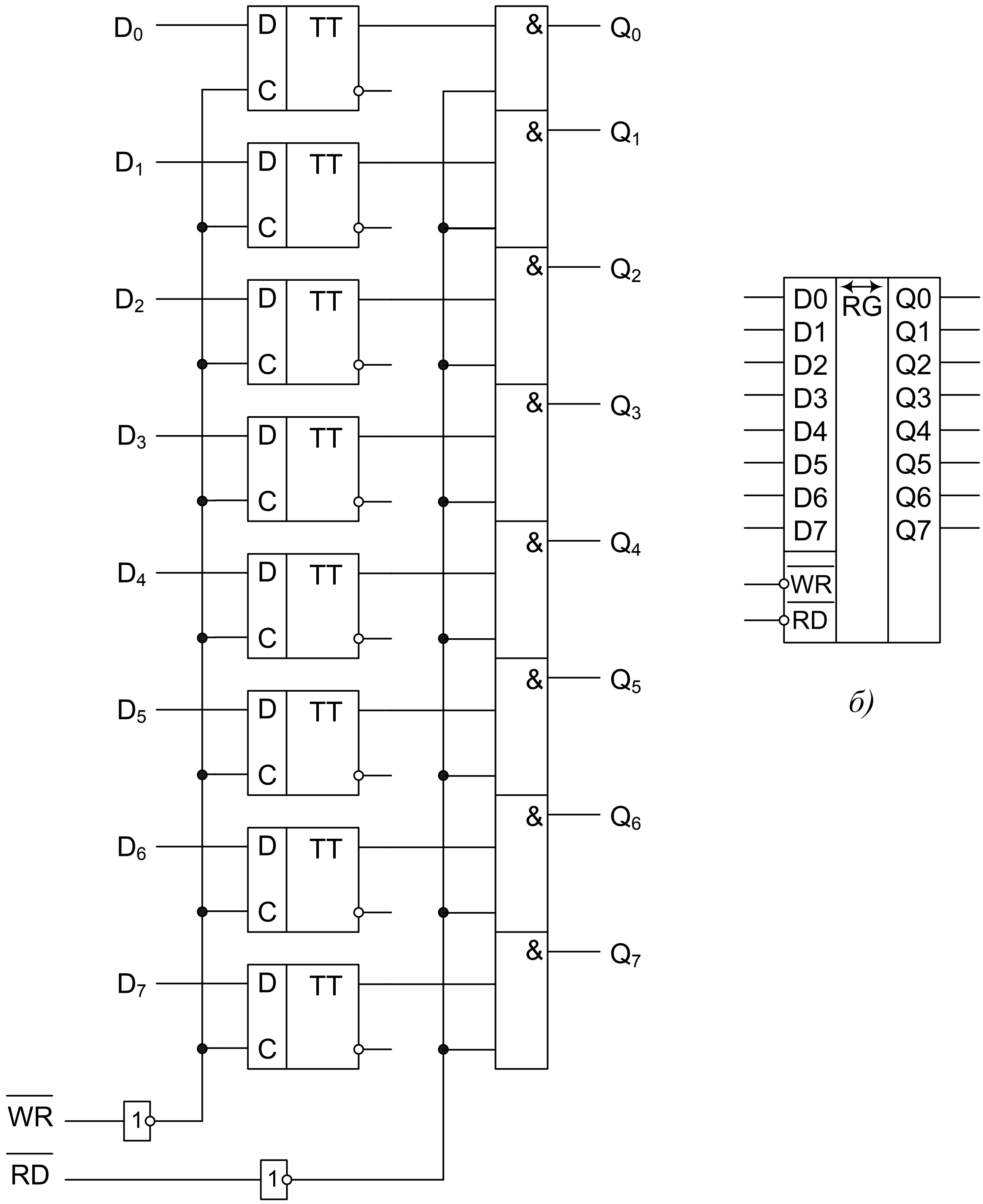
а)

Рис. 19. Шифратор: а – в развернутом виде; б – условное обозначение

Регистр

Регистр предназначен для временного хранения информации и может быть использован как ячейка оперативной памяти. Пример построения регистра на основе D-триггеров показан на рис. 20.

Логические элементы «И» играют роль вентилей и предназначены для формирования сигнала «Разрешение чтения».



а) *б)*

Рис. 20. Регистр: а – в развернутом виде; б – условное обозначение

3. Порядок выполнения работы

1. Собрать схему «калькулятор» (рис. 21).

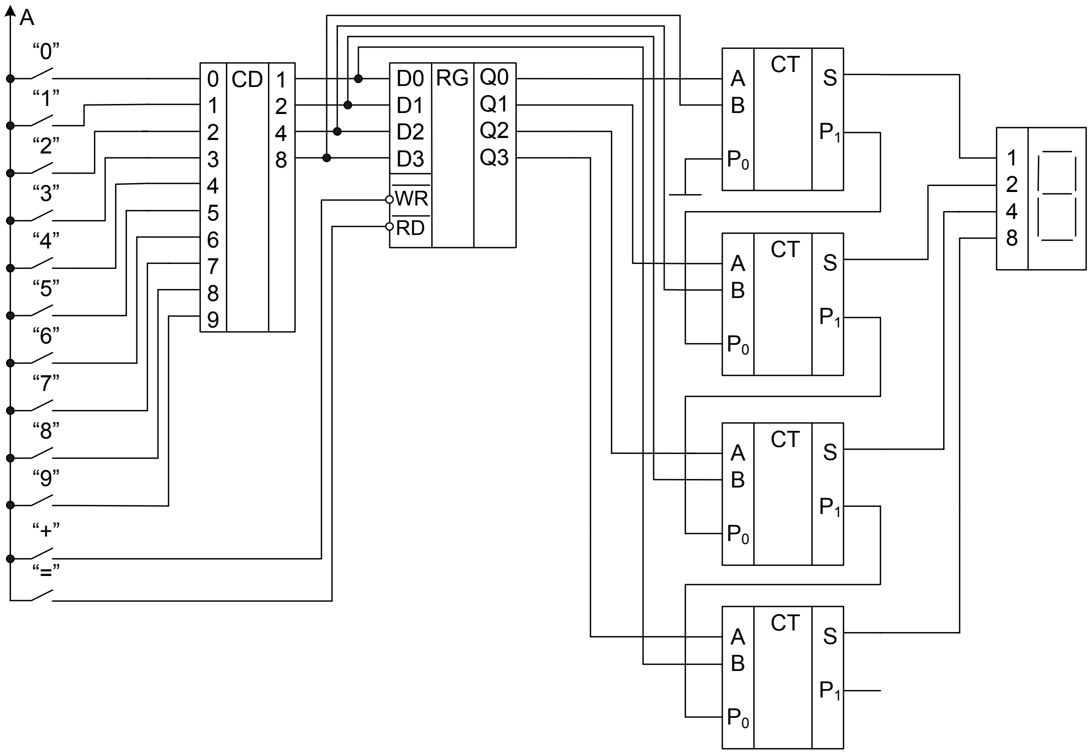


Рис. 21. Калькулятор

2. Исследовать работу схемы.
3. Предложить схему увеличения разрядности и возможности операций с дробными числами.

4. Контрольные вопросы

1. Представьте операнды (слагаемые – при сложении; уменьшаемое и вычитаемое – при вычитании) в двоичном обратном коде и выполните следующие операции:

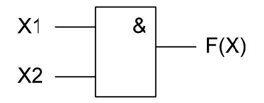
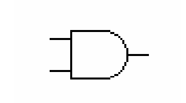
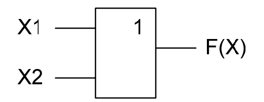
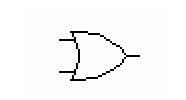
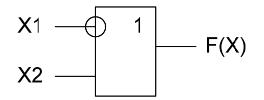
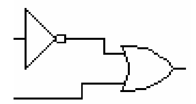
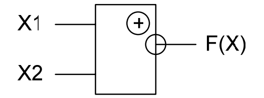
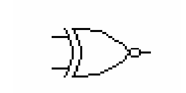
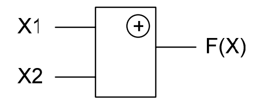
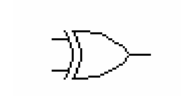
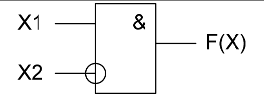
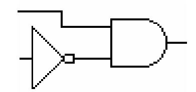
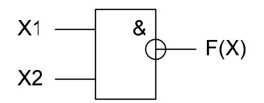
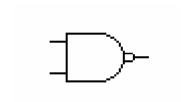
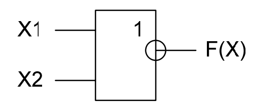
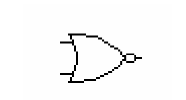
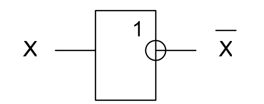
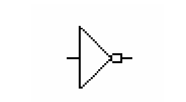
а) (+7)
 <u>(+1)</u> б) (+8)
 <u>(–5)</u> в) (+3)
 – <u>(+8)</u> г) (+13)
 – <u>(+10)</u>

Представьте операнды в двоичном дополнительном коде и выполните те же операции, что и в п. 1.

2. Дайте определение одноразрядного сумматора и спроектируйте его схему в ОФПН логических элементов
3. Укажите достоинства и недостатки двоичных сумматоров с последовательным переносом.

ПРИЛОЖЕНИЕ

Таблица истинности, название и обозначение некоторых логических функций

Обозначение логических операций (основное и дополнительное)	Таблица истинности					Как читается	Название операции	Обозначение логического элемента по ГОСТу	Обозначение логического элемента по ISO
	x_1	0	0	1	1				
	x_2	0	1	0	1				
$x_1 \cdot x_2$ $x_1 \wedge x_2$		0	0	0	1	x_1 и x_2	Конъюнкция; логическое И; логическое произведение		
$x_1 + x_2$ $x_1 \vee x_2$		0	1	1	1	x_1 или x_2	Дизъюнкция; логическое ИЛИ; логическая сумма		
$x_1 \to x_2$ $x_1 \supset x_2$		1	1	0	1	x_1 влечёт x_2; x_1 имплицирует x_2	Импликация		
$x_1 \infty x_2$ $x_1 \leftrightarrow x_2$		1	0	0	1	x_1 эквивалентно x_2	Эквивалентность; равнозначность		
$x_1 \oplus x_2$		0	1	1	0	x_1 неэквивалентно x_2	Сумма по модулю 2; неравнозначность; исключающее ИЛИ		
$x_1 \Delta x_2$		0	0	1	0	x_1 запрет по x_2	Запрет; отрицание импликации		
$x_1 \mid x_2$		1	1	1	0	x_1 и x_2 несовместны	Элемент (штрих) Шеффера; логическое И-НЕ; отрицание конъюнкции		
$x_1 \downarrow x_2$		1	0	0	0	ни x_1, ни x_2	Стрелка Пирса; логическое ИЛИ-НЕ; функция Вебба; отрицание дизъюнкции		
$\overline{x}$	x $\overline{x}$			0 1	1 0	не x	Инверсия; логическое НЕ; логическое отрицание		

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Опадчий, Ю. Ф.* Аналоговая и цифровая электроника (Полный курс) : учеб. для вузов / Ю. Ф. Опадчий, О. П. Глудкин, А. И. Гуров; под ред. О. П. Глудкина. – М. : Горячая линия-Телеком, 2002. – 768 с.

2. *Бройдо, В. Л.* Основы информатики: учеб. для вузов / В. Л. Бройдо. – СПб. : СПб ГИЭА, 1999. – 104 с.

3. *Гук, М.* Аппаратные средства локальных сетей / М. Гук. – СПб. : Питер, 2000. – 576 с.

4. *Каган, Б. М.* Электронные вычислительные машины и системы / Б. М. Каган. – М. : Энергоатомиздат, 1991. – 592 с.

5. *Пилгрим, А.* Персональные компьютеры. Кн. 1: Аппаратно-программная организация / А. Пилгрим. – СПб. : БХВ, 1999. – 848 с.

6. *Олифер, В. Г.* Компьютерные сети / В. Г. Олифер, Н. А. Олифер. – СПб. : Питер, 2000. – 672 с.

ОГЛАВЛЕНИЕ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
К ЛАБОРАТОРНЫМ РАБОТАМ
ПО ДИСЦИПЛИНАМ «СЕТИ, ЭВМ И СРЕДСТВА КОММУНИКАЦИЙ»,
«ВЫЧИСЛИТЕЛЬНАЯ И МИКРОПРОЦЕССОРНАЯ ТЕХНИКА»

Составитель
ОРЛОВ Дмитрий Юрьевич

Ответственный за выпуск – зав. кафедрой доцент Ю.А. Орлов

Подписано в печать 14.08.07.
Формат 60x84/16. Усл. печ. л. 3,25. Тираж 100 экз.
Заказ
Издательство
Владимирского государственного университета.
600000, Владимир, ул. Горького, 87.