

Министерство цифрового развития,
связи и массовых коммуникаций
Федеральное государственное
образовательное бюджетное учреждение
высшего профессионального образования
«Сибирский государственный университет
телекоммуникаций и информатики»
(СибГУТИ)

Л.Г. Рогулина
А.М. Сажнев

Электротехника, электроника и схемотехника
Часть 2. Цифровые устройства

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Новосибирск
2024

УДК 621.314.2

Дтн, доц. Л.Г. Рогулина, к.т.н., доц. А.М. Сажнев. Электротехника, электроника и схемотехника.

Часть 2. Цифровые устройства: Учебно-методическое пособие / ГОУ ВПО СибГУТИ. Новосибирск, 2024 г. – 121 стр.

В учебно-методическом пособии приводятся базовые теоретические сведения и описания лабораторных работ по основным разделам дисциплины: представление логических элементов в виде диодной или транзисторной сборки, коды, синтез комбинационных схем, дешифраторы, шифраторы, мультиплексоры и демультиплексоры, преобразователи кодов, триггеры, регистры, счетчики. Каждое из описаний имеет краткое теоретическое введение, задание на работу и контрольные вопросы. Все работы выполняются путем моделирования с использованием программы Electronics Workbench (EWB). Учебное пособие может быть использовано студентами всех форм обучения, а также полезно для дипломного проектирования.

Каф. ВС

Илл. 163, табл. 20, список лит. – 6 назв.

Рецензент: к.ф.-м.н. Ракшун Я.В.

Для студентов направления 09.03.01 «Информатика и вычислительная техника» изучающих дисциплину «Электротехника, электроника и схемотехника».

Утверждено редакционно-издательским советом ГОУ ВПО СибГУТИ
в качестве учебного пособия

© Сибирский государственный
университет телекоммуникаций
и информатики, 2024 г.

Оглавление

Введение	5
1 Лабораторная работа № 1. Преобразование аналогового сигнала в цифровой	6
1.1 Цель работы	6
1.2 Пояснения к работе	6
1.3 Описание моделей аналогового и цифрового сигналов	12
1.4 Порядок выполнения работы	14
1.5 Результаты работы	19
1.6 Контрольные вопросы	19
2 Лабораторная работа № 2. Схемная реализация логических элементов.	20
2.1 Цель работы	20
2.2 Пояснения к работе	20
2.3 Порядок выполнения работы	24
2.4 Результаты работы	28
2.5 Контрольные вопросы	28
3 Лабораторная работа №3. Синтез комбинационных цифровых устройств .	29
3.1 Цель работы	29
3.2 Пояснения к работе	29
3.3 Порядок выполнения работы	34
3.4 Результаты работы	38
3.5 Контрольные вопросы	38
4 Лабораторная работа № 4. Минимизация логических устройств	39
4.1 Цель работы	39
4.2 Пояснения к работе	39
4.3 Порядок выполнения работы	48
4.4 Результаты работы	51
4.5 Контрольные вопросы	51
5 Лабораторная работа № 5. Исследование дешифратора и шифратора . . .	52
5.1 Цель работы	52
5.2 Пояснения к работе	52
5.3 Описание модели дешифратора и шифратора	56
5.4 Порядок выполнения работы	58
5.5 Результаты работы	60
5.6 Контрольные вопросы	60
6 Лабораторная работа № 6. Исследование работы мультиплексора и демультиплексора	61
6.1 Цель работы	61
6.2 Пояснения к работе	61
6.3 Описание модели мультиплексора и демультиплексора	64
6.5 Порядок выполнения работы	65
6.4 Результаты работы	68
6.5 Контрольные вопросы	68
7 Лабораторная работа №7. Синтез и исследование преобразователей кодов.	69
7.1 Цель работы	69

7.2 Пояснения к работе	69
7.3 Описание схемы подключения для исследования преобразователя кода	70
7.4 Порядок выполнения работы	71
7.5 Результаты работы	75
7.6 Контрольные вопросы	75
8 Лабораторная работа № 8. Матричная реализация логических функций....	76
8.1 Цель работы	76
8.2 Пояснения к работе	76
8.3 Порядок выполнения работы	79
8.4 Результаты работы	81
8.5 Контрольные вопросы	81
9 Лабораторная работа № 9. Арифметические сумматоры.....	82
9.1 Цель работы	82
9.2 Пояснения к работе	82
9.3 Порядок выполнения работы	86
9.4 Результаты работы	87
9.5 Контрольные вопросы	87
10 Лабораторная работа № 10. Исследование триггеров	88
10.1 Цель работы	88
10.2 Пояснения к работе	88
10.2.1 Асинхронные триггеры	89
10.2.2 Синхронные триггеры	95
10.2.3 Способы управления триггерами	99
10.2.4. Модели триггеров в программе EWB	101
10.3 Порядок выполнения работы	101
10.4 Результаты работы	103
10.5 Контрольные вопросы	104
11 Лабораторная работа № 11. Регистры.....	105
11.1 Цель работы	105
11.2 Пояснения к работе	105
11.3 Исследование работы параллельного регистра на D-триггерах	115
11.4 Порядок выполнения работы	116
11.5 Результаты работы	118
11.6 Контрольные вопросы	118
12 Литература.	118

Введение

Разработка программного обеспечения для проектирования электронных устройств связана с созданием адекватных моделей, позволяющих исследовать физические процессы в радиоаппаратуре. Физическое моделирование связано с большими материальными затратами, а часто просто невозможно из-за чрезвычайной сложности устройств. Поэтому при разработке таких устройств прибегают к математическому моделированию с использованием средств и методов вычислительной техники. Примером такой моделирующей программы является Electronics Workbench (EWB). Она позволяет создавать на экране монитора принципиальные электрические схемы устройств, подключать контрольно-измерительные приборы, которые по характеристикам и внешнему виду близки к их промышленным аналогам, заносить результаты в текстовый файл. Моделирование начинается щелчком обычного выключателя. Созданная сравнительно недавно пятая версия программы EWB работает под управлением оболочки Windows 7,8 и 10, легко осваивается, удобна в пользовании, занимает на жестком диске около 16 Мбайт, обладает преемственностью, т.е. все схемы, созданные в версиях 3.0 и 4.1. могут быть промоделированы и в пятой версии, обладает совместимостью с программой P-Spice.

Особенностью программы EWB является наличие контрольно-измерительных приборов, по внешнему виду, органам управления и характеристикам максимально приближенных к их промышленным аналогам, что способствует приобретению практических навыков работы с наиболее распространенными измерительными приборами: мультиметром, осциллографом, генератором сигналов, а также достаточно обширной библиотекой логических элементов и цифровых устройств. Работа программы осуществляется в виде характерных для Windows окон, управление можно осуществлять с клавиатуры и с помощью мышки, что определяет высокие эргономические качества программы и не требует высокой квалификации у пользователя.

В лабораторных работах предлагаются уже составленные, готовые схемы с подключенными контрольно-измерительными приборами. Работа над устройством заключается в изучении, протекающих в нём процессов, их количественном и качественном анализе.

Лабораторный цикл по курсу “Электротехника, электроника и схемотехника” предполагает знание студентами основ компьютерного моделирования и навыков работы с программой EWB.

По каждой лабораторной работе оформляется отчет, который должен содержать: титульный лист, цель работы, задание на работу, схему устройства, таблицы истинности и данные выполненных измерений, расчёты с формулами и подставленными численными данными, графические результаты временных зависимостей сигналов и выводы по работе.

1 Лабораторная работа № 1

Преобразование аналогового сигнала в цифровой

1.1 Цель работы

Изучение преобразования аналогового сигнала в цифровой посредством дискретизации и квантования заданного входного сигнала и представления его в виде двоичного кода.

1.2 Пояснения к работе

Цифровые сигналы широко используются при решении многих задач, встречающихся практически во всех областях жизни. Это объясняется как развитием радиоэлектроники вообще, так и рядом преимуществ цифровой техники по сравнению с традиционной аналоговой. Можно выделить два главных направления приложения цифровой техники: *автоматизированное управление* технологическими процессами, включая автоматизированные контроль и диагностику технических средств и *использование вычислительных технологий* для передачи и обработки сигналов, автоматизации проектирования, решения задач административно-организационного управления.

Цифровое устройство это устройство, осуществляющее прием, хранение и преобразование дискретной информации по некоторому алгоритму, т.е. вырабатывающее сигнал, определенным образом связанный с совокупностью входных сигналов. Обычно эта связь оказывается достаточно сложной и реализуется набором более простых преобразований сигналов. При этом реализация множества преобразований сигналов в устройстве оказывается возможной как аппаратным путем, т.е. с использованием отдельных узлов для соответствующих преобразований, так и с использованием универсального узла – микропроцессорной системы, где осуществляются эти преобразования. По этому принципу различают устройства с *жесткой логикой и программируемой логикой*. Выбор оптимального принципа построения конкретного устройства определяется требованиями, предъявляемыми к данному устройству. В любом случае принцип действия этих устройств базируется на использовании законов булевой алгебры, использующей двоичное представление сигналов.

Сигнал – это изменение физической величины, используемой для передачи данных. Сигнал образуется на основе некоторой физической величины (электромагнитные или акустические колебания, электрическое напряжение и др.), традиционно называемой энергетическим носителем, путем изменения одного или нескольких ее параметров (амплитуды, частоты, фазы, длительности и др.) по закону передаваемой информации. Считают, что сигнал – это материально-энергетическое воплощение сообщения. Посредством совокупности сигналов можно представить любое сложное сообщение. Сигнал может преобразовываться без изменения смысла информации из одной физической величины в другую, более удобную для передачи по каналу связи и

обработки в компьютере. Изменение параметров физической величины по закону передаваемого сообщения называют модуляцией, а изменяемые параметры – информативными.

Сигналы классифицируют по ряду признаков, например, степени определенности ожидаемых значений – случайные и детерминированные; по структуре временного изменения – непрерывные и дискретные; по особенностям спектрального представления – низкочастотные и высокочастотные, узкополосные и широкополосные; по способу преобразования – кодированные, декодированные, усиленные, дискретизированные и т.д.; по принадлежности к виду связи – вещательные, телеграфные, телефонные, радиолокационные и пр.

В процессе передачи сигналов от источника к приемнику физические величины и способы их модуляции могут многократно изменяться, но содержание сообщение остается неизменным, поскольку оно определяется только законом модуляции.

В общем случае, способ формализованного описания различных сигналов и соответственно сообщений называется *представлением информации*. В теории информации рассматривают не физическое, а математическое представление сигналов, то есть их описание с помощью различных функций, формул, графиков, законов распределения вероятностей и т.д.

Наиболее распространенными способами представления сигналов являются временной, спектральный и статистический. Во многих случаях информация о протекании некоторого физического процесса поступает от соответствующих датчиков в виде электрических сигналов, которые непрерывно изменяются во времени. Однако в цифровой технике, изначально предполагающей использование нелинейных преобразований параметров сигналов, преобладает временное их описание, отражающее закон изменения одного из параметров (чаще всего напряжения) во времени. Таким образом, сигналы представляются одномерными функциями времени. Для наглядности при их анализе часто используются временные диаграммы, графически отображающие изменения сигналов на временной оси.

По характеру поведения сигнала на временной оси $y(t)$ различают следующие разновидности сигналов:

- 1) Непрерывная функция непрерывного времени в интервале $0 \leq t \leq T$ (рис. 1.1 а);
- 2) Непрерывная функция дискретного времени (рис. 1.1, б). Значения функции определяются только на дискретном множестве $t_j, j=1,2,\dots, J$. Функция $y(t)$ может принимать любые значения в заданном диапазоне от y_{\min} до y_{\max} . Преобразование функции $y(t)$ непрерывного времени t в функцию $y(t_i)$ дискретного времени t_i называется *дискретизацией (квантованием, dT- шаг квантования) по времени*.
- 3) Дискретная функция дискретного времени (рис.1.1, в). Здесь дискретизированы и время и значения, принимаемые функцией (dY - шаг квантования по амплитуде). Это т.н. цифровой сигнал.

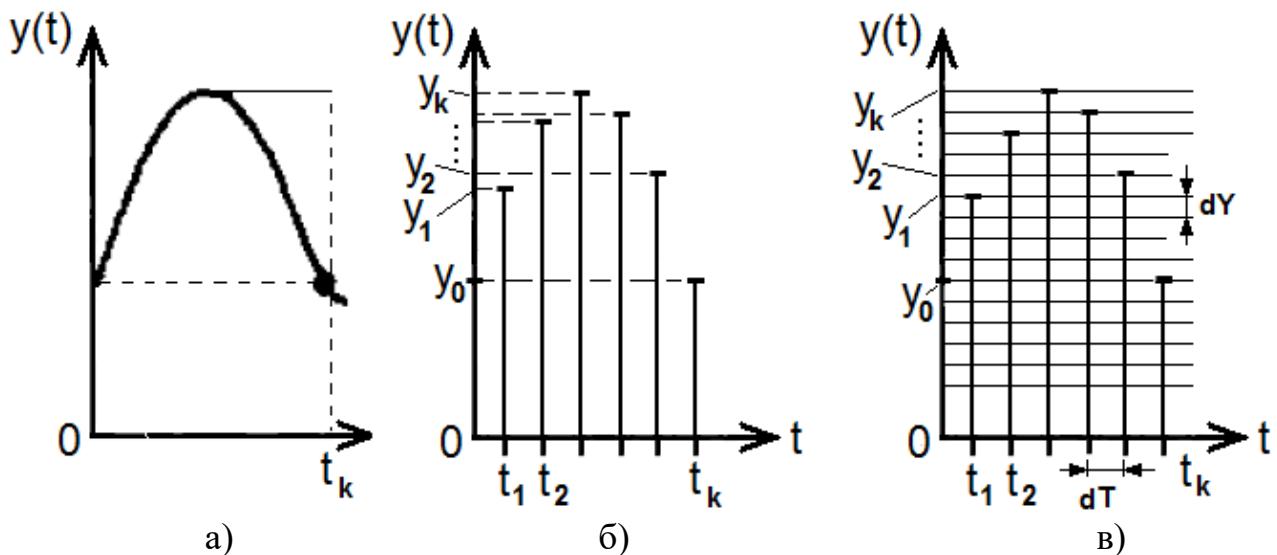


Рисунок 1.1 – Виды сигналов

В цифровой электронике используются цифровые сигналы, число возможных дискретных значений для которых ограничено до минимума, необходимого для передачи информации (одного бита). В этом случае сигналы соотносят к одному из лишь двух возможных значений (0 или 1). При этом соответствующее значение бита информации может быть передано как уровнем сигнала (потенциальные сигналы), так и в виде перехода из одного состояния в другое (импульсные сигналы). В свою очередь, в зависимости от соотношения уровней для потенциальных сигналов или направления изменения уровня сигнала для импульсных различают положительную логику ($U^0 < U^1$ для потенциальных сигналов, переход $0 \rightarrow 1$ принимается за 1 для импульсных сигналов) и отрицательную логику в противном случае.

Такое неполное использование возможностей электрических сигналов для передачи информации приводит к усложнению алгоритмов получения тех же результатов, что и в аналоговых устройствах. К тому же аналоговые схемы обладают потенциально более высоким быстродействием. Но современное состояние теории и практики цифровой обработки сигналов позволяет часто несколько более простых и более точных решений многих задач. Одним из достоинств цифровой электроники является хорошо развитая элементная база, что позволяет обычно свести проектирование цифровых устройств к отысканию наиболее подходящего алгоритма и его оптимальной реализации.

Согласно теоремы Котельникова сигнал, описываемый функцией с ограниченным спектром, определяется своими дискретными значениями, которые отсчитываются через интервалы времени $t = 1/2 \cdot F_C$, где F_C – ширина спектра. Это означает, что сигнал $y(t)$ можно представить отдельными мгновенными значениями, которые отсчитываются через конечный интервал времени. По этим значениям оказывается возможным полностью восстановить исходный непрерывный сигнал. Таким образом, осуществляя дискретизацию с необходимой частотой и квантование с достаточной точностью, оказывается возможным реализовать достоинства цифровых методов обработки и для

аналоговых сигналов. При этом часто используются и самые разные способы представления сигналов. В частности, к дискретно-непрерывным сигналам относят также время – импульсное представление первичного сигнала $y(t)$ прямоугольными импульсами с непрерывным информативным параметром t_i/T , где t_i – длительность импульсов, пропорциональная значению сигнала; T – период повторения импульсов (рис. 1.2 а). При число-импульсном представлении (рис. 1.2 б) информативным параметром является количество импульсов за период T .

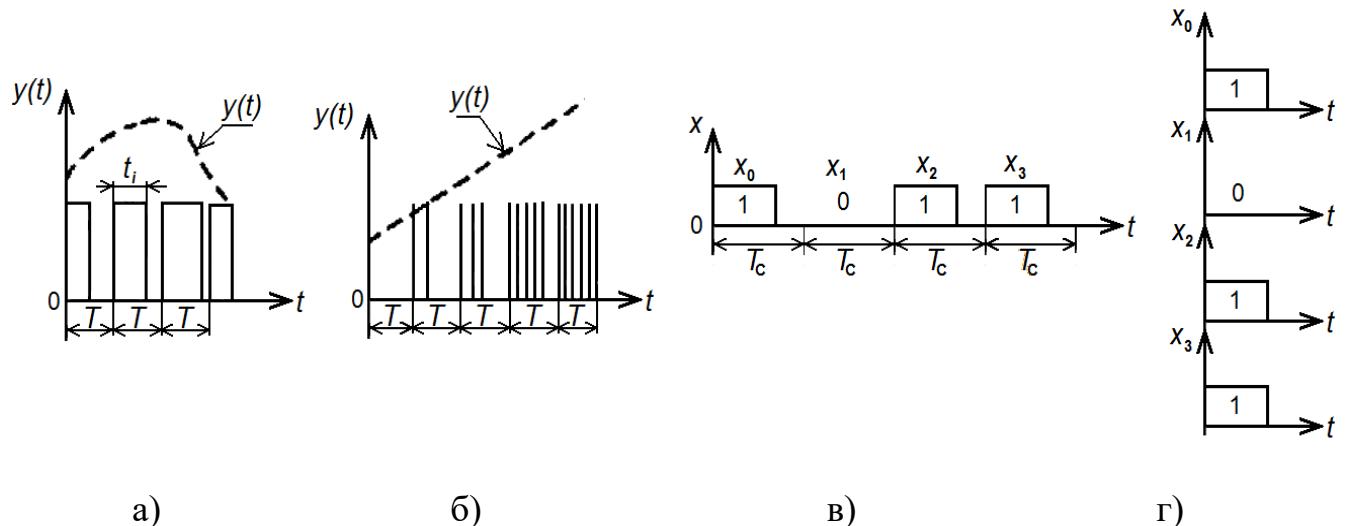


Рисунок 1.2 – Способы представления сигналов

В цифровой схемотехнике используют разрядно-цифровое кодирование, в котором исходный сигнал представляется группой символов, отображающих значения цифр 0 и 1 двоичной системы счисления уровнями электрических сигналов. Передача цифрового сигнала может быть осуществлена как последовательной передачей его символов во времени (последовательный код) с помощью единственного канала передачи (рис. 1.2 в), так и одновременной их передачей с использованием многоканальной связи (параллельный код) (рис. 1.2 г).

На практике последовательный код используют при передаче информации на большие расстояния (например, между компьютерами), а параллельный код – при передаче информации на малые расстояния (например, между блоками компьютера). Параллельный код обеспечивает, очевидно, наибольшую скорость передачи, но требует существенно большего числа независимых линий связи.

Абсолютные значения уровней напряжений, соответствующих 0 и 1, определяются конкретной элементной базой, используемой при построении цифровых устройств. В частности, в наиболее распространенных ТТЛШ сериях эти напряжения соответствуют 0,4 В и 2,4 В. Между этими значениями существует некоторая защитная переходная зона с тем, чтобы свести до минимума влияние внешних условий и помех на работу устройств. Менее

строго уровни напряжений, соответствующие 0 и 1 определены в КМОП – сериях, допускающих большой разброс возможных напряжений питания (3,...,15) В, поэтому лог.0 соответствует сигнал $< 0,3U_{пит}$, а лог. 1 сигнал $> 0,7U_{пит}$.

Одной из главных характеристик цифровых устройств является их *быстродействие*, определяющее предельную скорость изменения входных сигналов, при котором устройство нормально функционирует. Для оценки быстродействия обычно используются временные параметры тестовых сигналов, в качестве которых чаще всего используют прямоугольные импульсы. Под импульсом понимают кратковременное изменение напряжения или тока в электрической цепи. В цифровой технике используются видеоимпульсы, не имеющие в отличие от радиоимпульсов высокочастотного заполнения.

Основными характеристиками и параметрами импульсов (рис. 1.3) для характеристики быстродействия устройств являются:

- амплитуда импульса $U_m=A$;
- активная длительность импульса t_i (измеряется на уровне 0,5 А);
- длительность фронта t_f (определяется как время, за которое выходной сигнал изменяется от 0,1 А до 0,9 А);
- длительность среза t_{CP} (определяется как время, за которое выходной сигнал изменяется от 0,9 А до 0,1 А);
- время задержки $t_{зд}$ (определяется как время между началами активных длительностей импульсов на входе и выходе устройства).

При оценке быстродействия цифровых устройств с помощью периодической последовательности импульсов, вводится дополнительные параметры – период следования импульсов T и скважность $Q = T/t_i$.

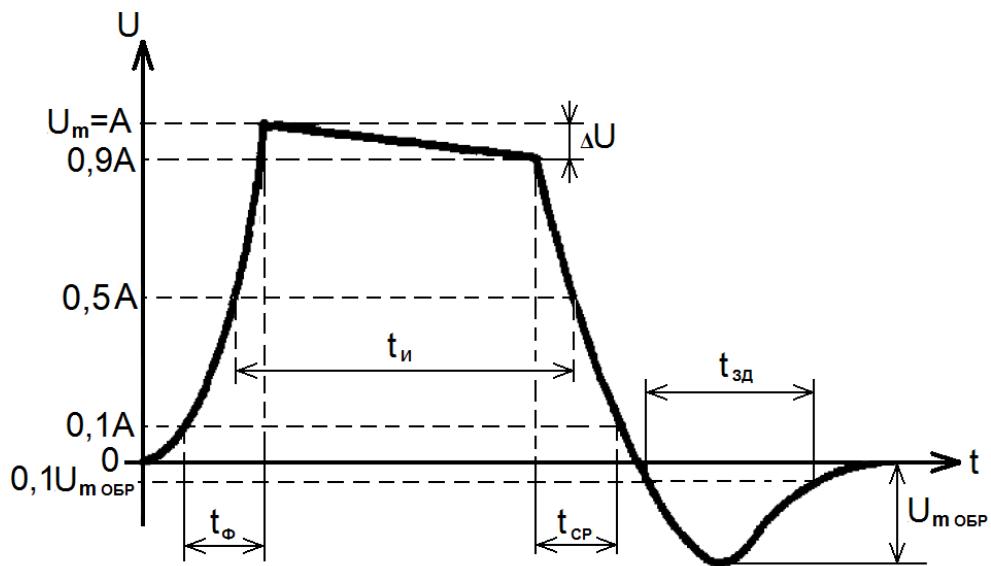


Рисунок 1.3 – Параметры импульса

В цифровых устройствах используются различные системы счисления (способ представления чисел с помощью графических символов). Системы счисления делят на позиционные и непозиционные. В позиционных системах значение одного и того же символа, изображающего число, изменяется с учётом положения этого символа относительно крайнего правого разряда или запятой. Например, в трехзначном числе 222 цифра два в зависимости от размещения в первом, втором или третьем разряде обозначает соответственно два, двадцать и двести. Таким образом, вводится понятие разряда.

Разряд – это положение цифры в записи числа. Значение разряда определяется его весом – степенью, в которую необходимо возвести основание системы счисления. Например, в десятичной системе 10^0 – единицы, 10^1 – десятки, 10^2 – сотни.

В непозиционных системах значение символа не определяется его положением в записи. Довольно часто для обозначения каких-то ключевых цифр используются уникальные символы. В цифровых устройствах и микропроцессорах чаще всего применяются следующие позиционные системы счисления: 1) десятичная; 2) двоичная; 3) восьмеричная; 4) шестнадцатеричная. Число в любой позиционной системе счисления можно представить в виде последовательности цифр:

$$A = a_n a_{n-1} \dots a_1 a_0, b_{-1} b_{-2} \dots b_{-k}, \quad (1.1)$$

где a_i, b_i – цифры данной системы счисления.

Или в виде формулы разложения:

$$A = a_n p^n + a_{n-1} p^{n-1} + \dots + a_2 p^2 + a_1 p^1 + a_0 p^0 + b_{-1} p^{-1} + b_{-2} p^{-2} + \dots + b_{-k} p^{-k}, \quad (1.2)$$

где p – основание системы счисления (количество различных цифр в системе счисления);

p^i – вес единицы данного разряда.

Представим натуральное число, например 216, в десятичной системе счисления. Это число имеет 3 разряда (2-1-6). Для представления числа воспользуемся данным выше определением разряда и расставим единицы, десятки и сотни, начиная со старшего, самого большого разряда:

$$\begin{aligned} & 2 \cdot 10^2 + 1 \cdot 10^1 + 6 \cdot 10^0, \\ & 2 \cdot 100 + 1 \cdot 10 + 6 \cdot 10^0 = 216. \end{aligned}$$

Т. е. представление числа в позиционной системе счисления можно описать следующим образом:

$$\sum_{k=0}^{n-1} a_k p^k,$$

где n – разрядность числа.

Разрядность – это количество разрядов, необходимых для записи конкретного числа в выбранной системе счисления. Имея фиксированную разрядность, невозможно записать число, количество разрядов в котором превышает заданную разрядность.

В десятичной системе счисления $p = 10$, разрешенные цифры (0, 1, 2, 3, 4, 5, 6, 7, 8, 9). Число можно представить так:

$$A_{10} = 247,56_{10} = 2 \cdot 10^2 + 4 \cdot 10^1 + 7 \cdot 10^0 + 5 \cdot 10^{-1} + 6 \cdot 10^{-2}.$$

Веса соседних разрядов влево и вправо от запятой различаются в десять раз ($p = 10$):

$$\dots 1000 \ 100 \ 10 \ 1, \ 1/10 \ 1/100 \ 1/1000 \ \dots$$

В двоичной системе счисления $p = 2$, разрешенные цифры (0, 1). Число представляется так:

$$A_2 = 101110,101_2 = 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + \\ + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 46,625_{10}.$$

Веса соседних разрядов влево и вправо от запятой различаются в два раза ($p = 2$): $\dots 32 \ 16 \ 8 \ 4 \ 2 \ 1, \ 1/2 \ 1/4 \ 1/8 \ \dots$

В восьмеричной системе счисления $p = 8$, разрешенные цифры (0, 1, 2, 3, 4, 5, 6, 7). Число представляется так:

$$A_8 = 125,46_8 = 1 \cdot 8^2 + 2 \cdot 8^1 + 5 \cdot 8^0 + 4 \cdot 8^{-1} + 6 \cdot 8^{-2} = 64 + 16 + 5 + 0,5 + \frac{6}{64} \cong 85,6_{10}$$

Веса соседних разрядов влево и вправо от запятой различаются в восемь раз ($p = 8$): $\dots 4096 \ 512 \ 64 \ 8 \ 1, \ 1/8 \ 1/64 \ 1/512 \ \dots$

В шестнадцатеричной системе счисления $p=16$, разрешенные цифры (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F). Число представляется так:

$$2AF,C4_{16} = 2 \cdot 16^2 + 10 \cdot 16^1 + 15 \cdot 16^0 + 12 \cdot 16^{-1} + 4 \cdot 16^{-2} = \\ = 512 + 160 + 15 + \frac{12}{16} + \frac{4}{256} \cong 687,76_{10}$$

Веса соседних разрядов влево и вправо от запятой различаются в шестнадцать раз ($p = 16$): $\dots 4096 \ 256 \ 16 \ 1, \ 1/16 \ 1/256 \ 1/4096 \ \dots$

Вполне очевидно, что для записи одного и того же числа в разных системах счисления требуется разное количество разрядов. Основание системы счисления во всех системах счисления записывается одинаково: 10.

1.3 Описание моделей аналогового и цифрового сигналов

Модель задания параметров аналогового сигнала показана на рисунке 1.4. Модель содержит:

- источник постоянного напряжения U1;

- источник постоянного напряжения U_2 ;
- нагрузочный резистор R_1 ;
- осциллограф предназначен для измерения уровней отсчетов аналогового сигнала.

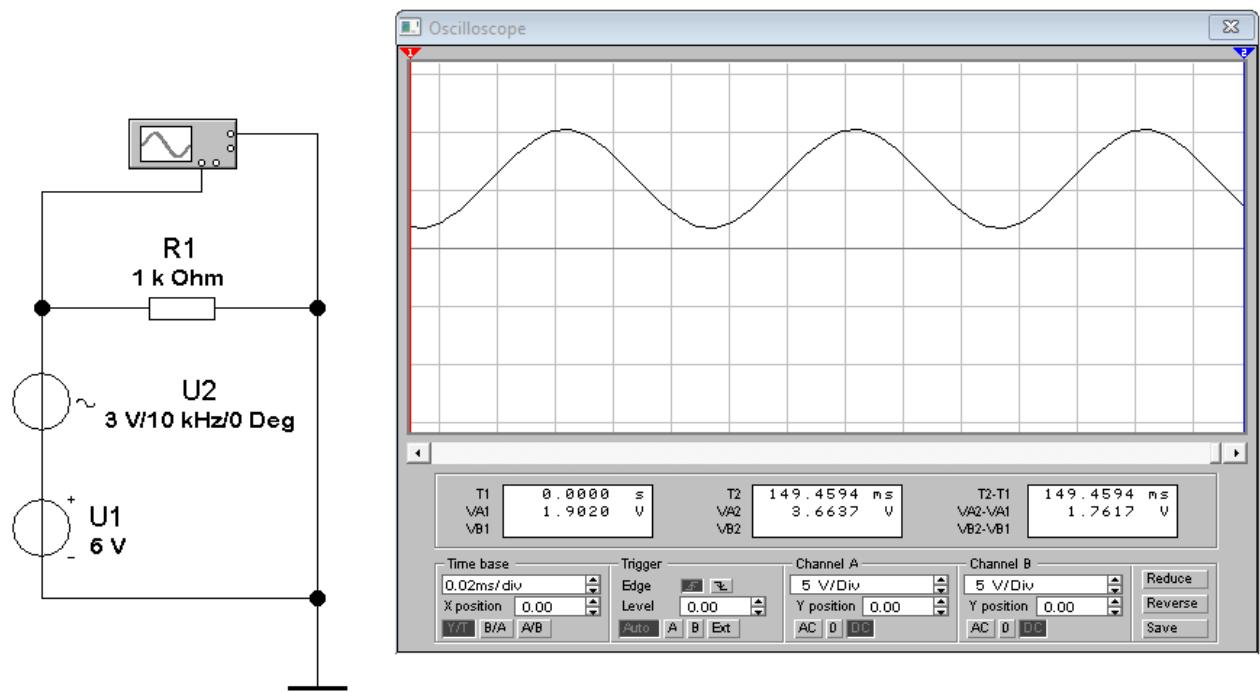


Рисунок 1.4 – Модель определения параметров аналогового сигнала (файл LAB1)

Модель для проверки кодирования результатов квантования аналогового сигнала представлена на рисунке 1.5 и включает в себя:

- цифровой генератор (Word Generator) , предназначенный для генерации 16-ти разрядных двоичных слов, которые набираются на экране, расположенным в левой части лицевой панели;
- семисегментный индикатор с преобразователем четырехразрядного

двоичного числа в шестнадцатеричную цифру ;

- четыре логических пробника ;

- логический анализатор предназначен для просмотра последовательности цифровых сигналов.

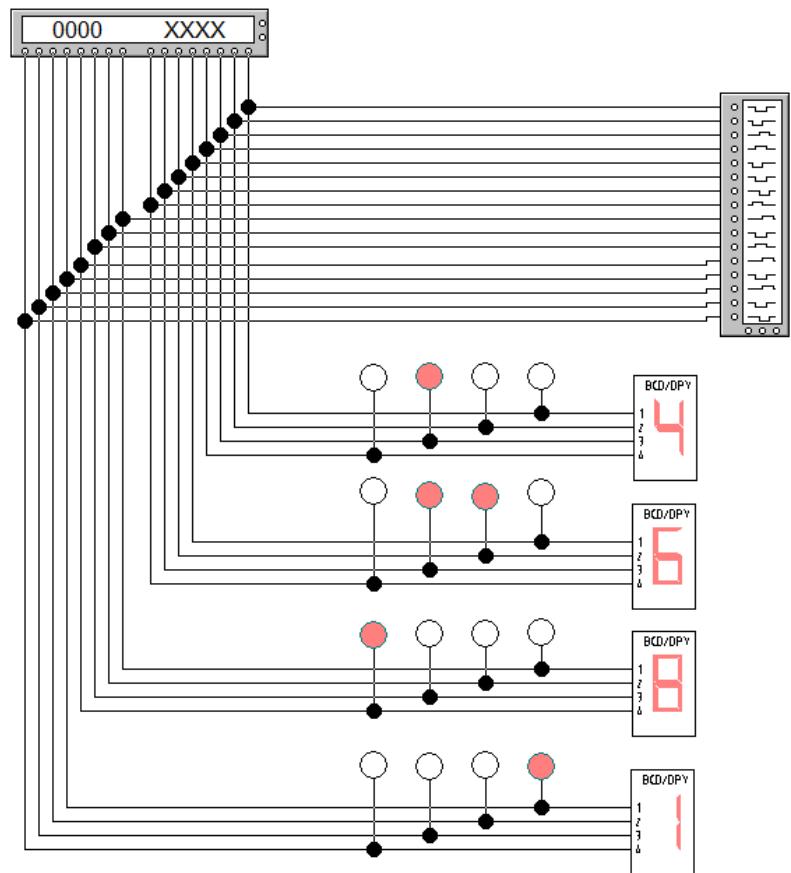


Рисунок 1.5 – Модель проверки кодирования результатов квантования аналогового сигнала (файл LAB2)

1.5 Порядок выполнения работы

- В соответствие со своим вариантом (номером бригады) выберите исходные данные из таблицы 1.1 и откройте файл LAB1.

Таблица 1.1 – Исходные данные для аналогового сигнала

Параметры	Номер бригады									
	1	2	3	4	5	6	7	8	9	10
U1, В	8	7	6	8	7	6	8	7	6	8
U2, В	5	4	3	5	4	3	5	4	3	5
f, Гц	50	100	80	20	60	40	10	70	90	30
f_t, Гц	400	800	600	200	400	300	100	800	900	400

Задайте напряжения источников U1, U2 (и частоту для U2) двойным щелчком левой клавиши мыши, открыв окна этих источников при появлении

символа “рука” **U2**
3 V/10 kHz/0 Deg (рисунок 1.6).

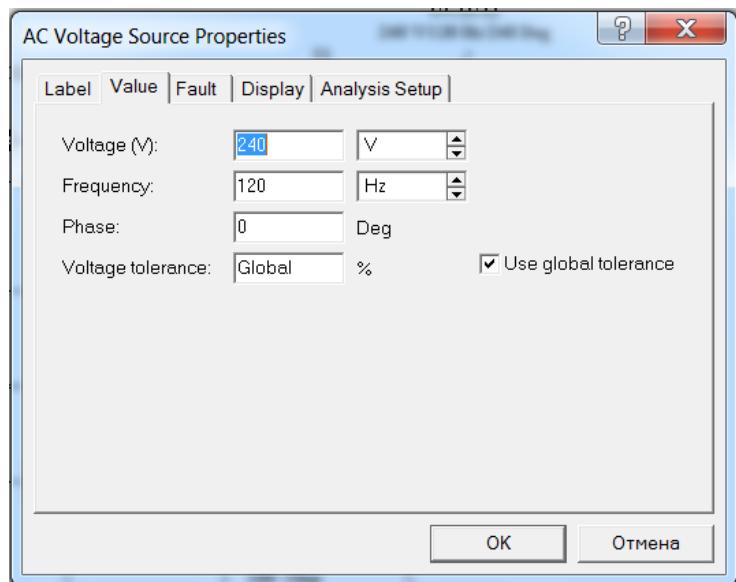


Рисунок 1.6 – Окно источника переменного тока

Включите схему клавишей в правом верхнем углу экрана. После заполнения экрана выключите схему. В верхней линейке панели инструментов нажмите значок , появится окно Analysis Graphs с изображением осциллографа. Откройте осциллограф двойным щелчком по иконке. Нажатием левой кнопки мыши на поле графика выделите пунктиром два-три периода гармонического сигнала.

Для удобства обработки результатов в верхней части окна экрана имеется ряд функциональных кнопок (рис.1.7):

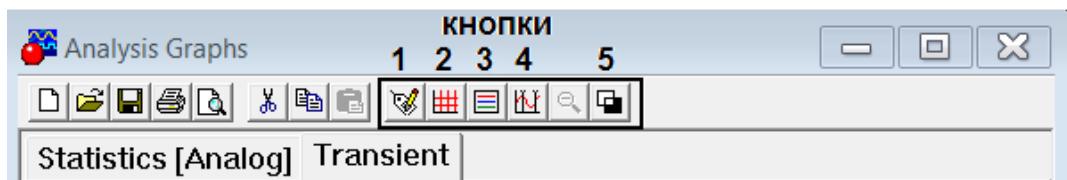


Рисунок 1.7 – Окно настроек

Кнопка 1 раскрывает диалоговое окно, позволяющее при закладке General установить ряд опций: *Font* – выбрать шрифт надписей; *Grid* – ширину сетки экрана и ее цвет; *Grid on*; *Cursors on* – удаление сетки и визиров с экрана; *Single Trace* – выбор для анализа отдельного графика, при этом номер графика выбирается установкой его номера в окошке Trace; *All Traces* – одновременный анализ всех графиков.

Кнопка 2 устанавливает и удаляет сетку экрана.

Кнопка 3 выводит информационное окно с номерами узлов и цветом соответствующих графиков (верхний правый угол экрана). Курсором мыши это окно можно переместить в любое место экрана.

Кнопка 4 выводит на экран два измерительных визира. За верхние треугольники визиры можно перемещать курсором мыши в любое место графиков и измерять значение координат точек.

Кнопка 5 изменяет цвет фона экрана.

На экран (правый верхний угол) выводится также цифровая информация о точках графика, на которые установлены визиры. Информация помещается в отдельном окне, содержание которого изменится в зависимости от положения визиров. Обозначения переменных величин информационного окна следующее:

x_1, y_1 – координаты точки графика для первого визира;

x_2, y_2 – координаты точки графика для второго визира;

$\min x, \max x; \min y, \max y$ – минимальные и максимальные значения координат графика;

dx – разность ($x_1 - x_2$).

Щелкните по кнопкам 2, 3 и 4.

2. Перенесите график в отчёт. Подготовьте и заполните таблицу 1.2. – Численные значения отсчетов в моменты квантования по времени. Шаг квантования по времени равен $dx = 1/ft$ (см. рис. 1.8). Используйте оба курсора для установки шага квантования, шагая ими поочередно через такт.

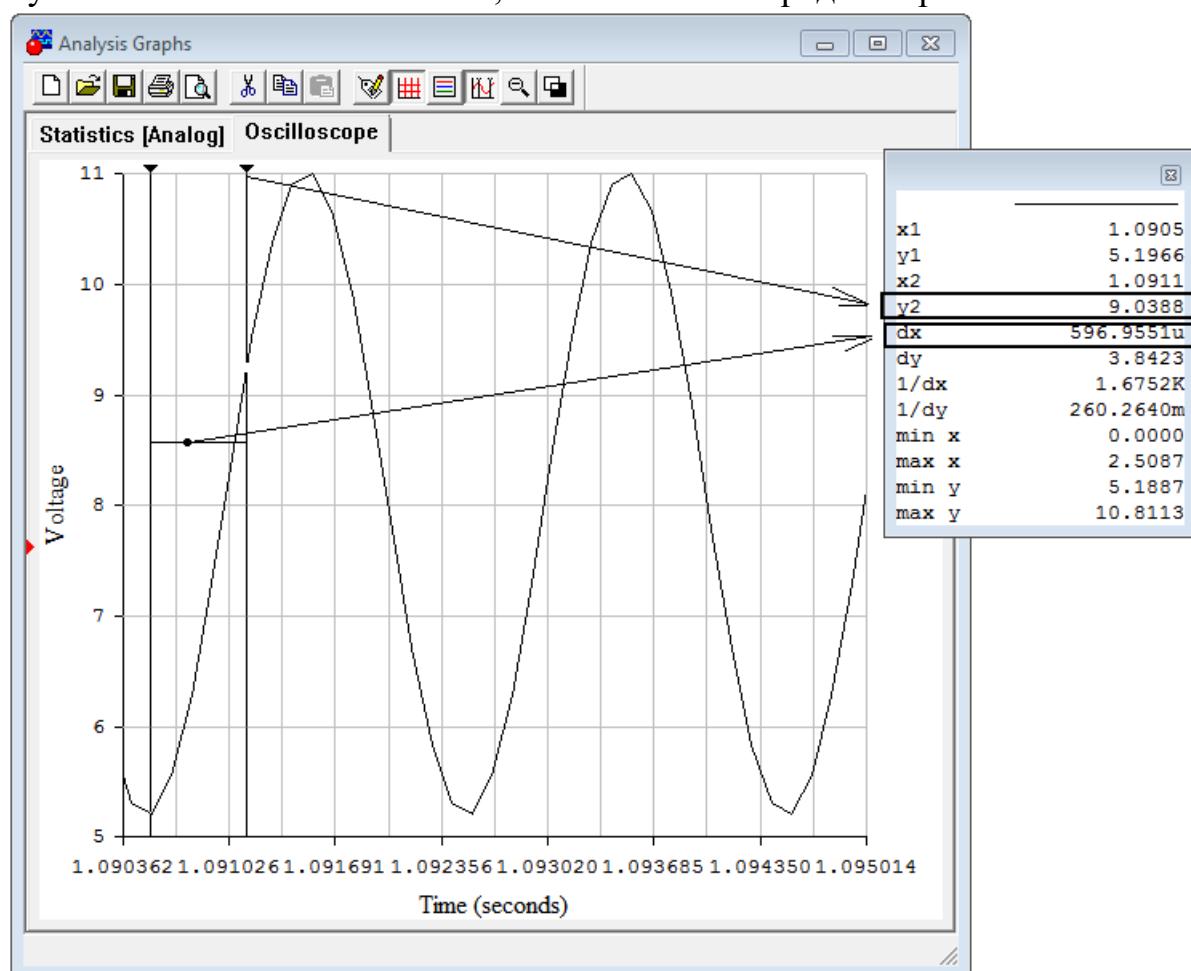


Рисунок 1.8 – Определение значений сигнала в моменты квантования

Полученные значения заносим во второй столбец таблицы 1.2 для десяти первых точек, начиная от минимального значения аналогового сигнала.

Таблица 1.2 – Результаты измерения, квантования и кодирования аналогового сигнала

Номер отсчета	Отсчеты сигнала (В)	Квантованный сигнал (В)	Четырехразрядный двоичный код
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

Далее выполняем квантование по уровню (округление) значений сигнала до ближайшего целого числа т.е. шаг квантования по уровню составляет один вольт и переводим в двоичное четырехразрядное число (код).

Постройте график цифрового сигнала (квантованный по уровню и по времени). Сопоставьте его с аналоговым сигналом.

3. Откройте файл LAB2 и в нем цифровой генератор как показано на рисунке 1.9. Нажмите кнопку Pattern и выполните очистку буфера посредством метки напротив команды Clear buffer и подтвердите, нажатием на кнопку Accept.

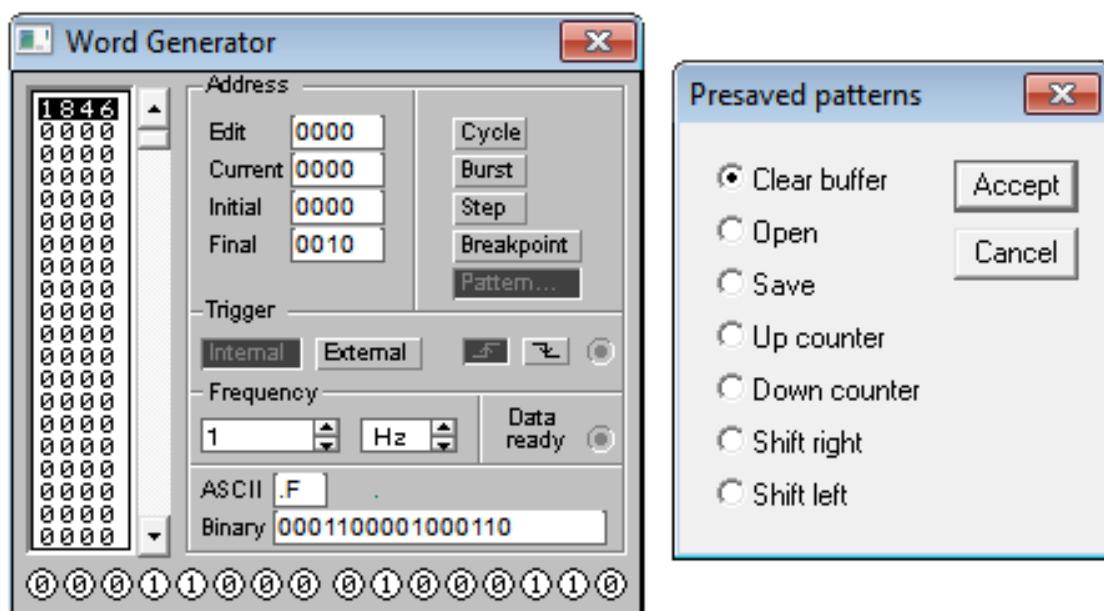


Рисунок 1.9 – Очистка буфера

Выполните установки в генераторе запуск по переднему фронту сигнала ; при внутреннем запуске (нажатии кнопки Internal) синхронизации. Установите частоту в окне Frequency 1 Hz. Нажмите step. Пошагово введите данные из последнего столбца таблицы 1.2 на экране, расположенным в левой части лицевой панели генератора слов. Сформированные слова выдаются на шестнадцать расположенных в нижней части прибора выходных клемм-индикаторов: - с индикацией в двоичном коде в строке окна binary; - в пошаговом (step), циклическим (cycle) или с выбранного слова до конца (при нажатии кнопки BURST) при заданной частоте посылок (1Гц). Установите режим Cycle. В окно Final загрузите число 10 (16 повторений).

4. Для просмотра временных зависимостей откройте логический анализатор.

Откройте окно параметров Clock нажатием на кнопку справа Set. Проверьте на соответствие установки рисунка 1.10 а. Далее аналогично откройте окно установки параметров панели Trigger и выполните проверку на соответствие рисунка 1.10 б. На лицевой панели анализатора в окне Clock per division установите единицу.

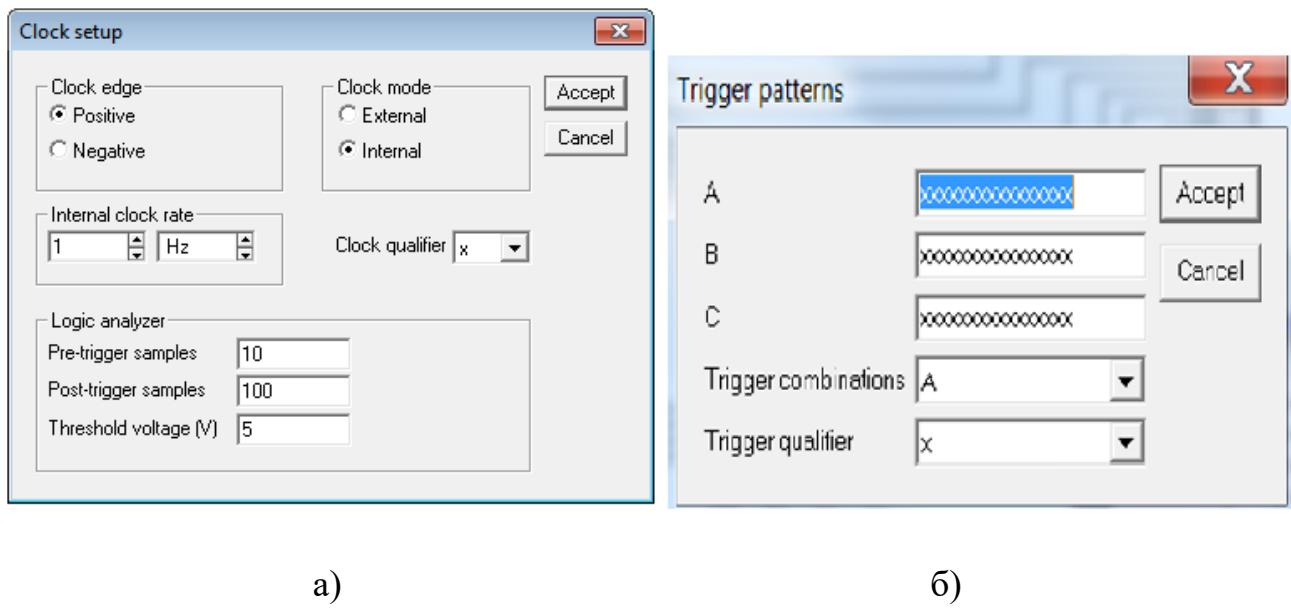


Рисунок 1.10 – Панели Clock (а) и Trigger (б)

5. Включите схему клавишей в правом верхнем углу экрана и после заполнения логического анализатора выключите. Проверьте соответствие временных зависимостей результатам кодирования в таблице 1.2 (рис. 1.11). Укажите и пронумеруйте последовательность кодов на логическом анализаторе в соответствие с номерами отсчетов.

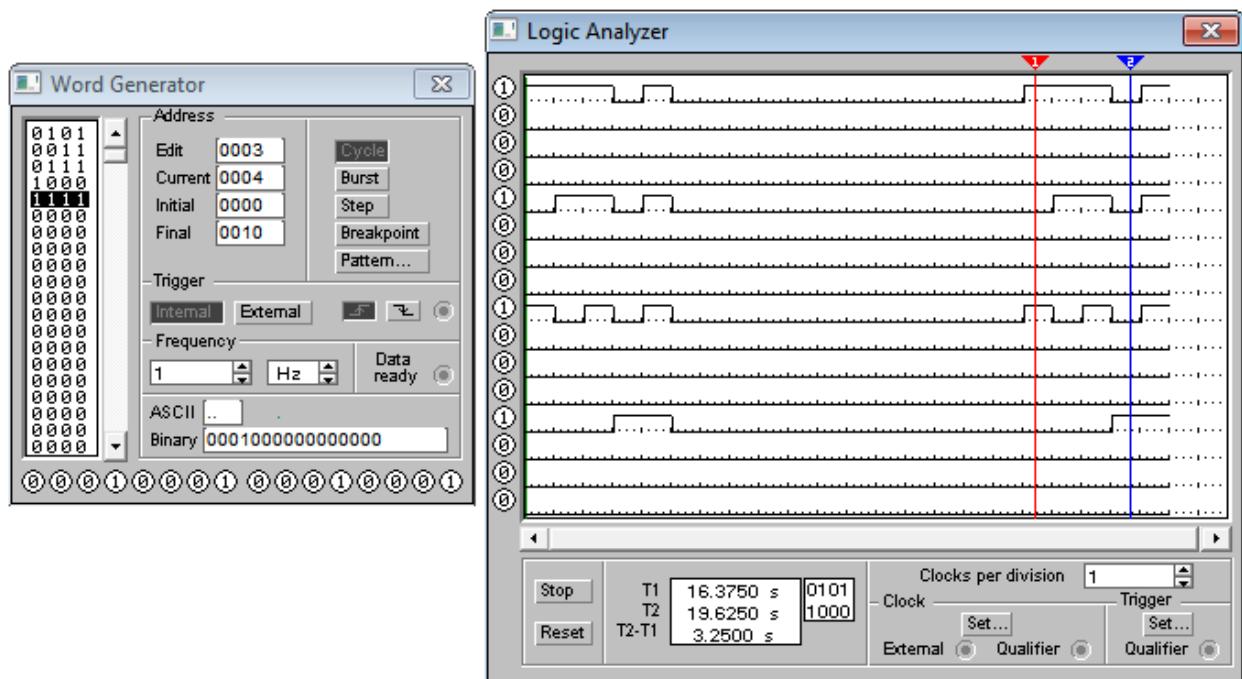


Рисунок 1.11 – Результаты кодирования

Можно просмотреть коды и на графическом анализаторе для этого нажмите на кнопку на рабочей панели.

Каждый разряд числа, заданного на генераторе, в логическом анализаторе представляется тетрадой двоичных разрядов. Младший разряд - верхний.

1.5 Результаты работы

Подготовьте отчет по лабораторной работе, который должен содержать исходные данные, схемы моделей, таблицу 1.2, временные зависимости входного цифрового сигнала и с экрана логического анализатора.

1.6 Контрольные вопросы

1. Какие существуют системы счисления?
2. Что такое разрядность?
3. Какая разрядность необходима для того, чтобы записать числа 255, 511, 1023?
4. Что произойдет, если попытаться записать число 256 в 7-разрядную ячейку (сетку)?
5. Как представить десятичное число минус 10 в прямом и обратном машинном коде в восьмиразрядной ячейке?

Лабораторная работа № 2

Схемная реализация логических элементов

2.1 Цель работы

Изучение элементов диодной и транзисторной логики.

2.2 Пояснения к работе

Алгебра логики - один из раздел математической логики. Её создателем является англичанин Джорж Буль (1815 - 1864), поэтому алгебру логики называют булевой алгеброй.

Начальным понятием булевой алгебры является высказывание.

Высказывание – это некоторое предложение, о котором можно утверждать истинно оно или ложно. Высказывание обозначают буквой (идентификатором). Например, два высказывания;

$$X_1 = <\text{Москва - столица России}>$$

$$X_1 = 1 \text{ -- истина}$$

$$X_2 = <\text{Луна больше Земли}>$$

$$X_2 = 0 \text{ -- ложь}$$

Если высказывание истинно, то его условно обозначают единицей, если ложно – нулем.

Логическая переменная – некоторая переменная величина **X**, которая может принимать одно из двух значений 0 или 1, то есть быть ложной или истинной $X=\{0,1\}$.

Логическая функция (булева функция, переключательная функция, функция алгебры логики) **n** переменных - это функция, которая может принимать одно из двух значений (0 или 1) на некотором наборе этих переменных $F(X_1, X_2, \dots, X_n) = \{0,1\}$.

Логическая функция задается таблицей истинности.

Таблица истинности – это совокупность всех возможных наборов (комбинаций) логических переменных и значений функции на этих наборах.

Например, логические функции одной переменной $n = 1$ – тривиальные функции.

Таблица 1 – Логические функции одной переменной

$F \setminus X$	0	1	Название функции
F_1	0	0	Const «0» - абсолютно ложная функция
F_2	0	1	Переменная икс - тождественная функция
F_3	1	0	«Не икс» - отрицание икс - инверсия икс
F_4	1	1	Const «1» - абсолютно истинная функция

Реализация этих функций показана на рисунке 2.1.

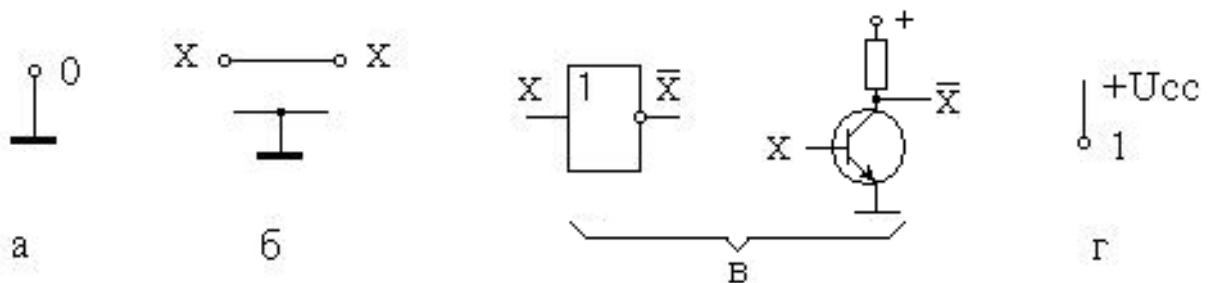


Рисунок 2.1 – Реализация функций одной переменной

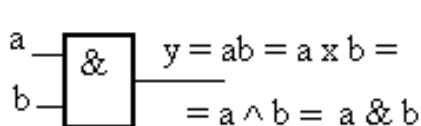
Функция F_1 всегда ложна (рис.2.1а), функция F_2 есть сама переменная икс (рис 2.1б), функция F_3 реализуется инвертором-транзисторный усилитель по схеме с ОЭ (рис.2.1в), функция F_4 всегда истинна (рис.2.1г).

В общем случае, если имеем n число независимых логических переменных, то можно составить $2^n = N$ различных наборов этих переменных, а так как на каждом из наборов функция может принимать значение 0 или 1 , то общее возможное число функций равно $L=2^N$. Так, при $n = 1$ число наборов $N = 2$, а число функций $L = 4$. Это тривиальные функции.

Рассмотрим логические функции двух переменных $n = 2$. Они относятся к элементарным функциям. Число наборов переменных равно $N = 2^2 = 4$, а число функций $L = 16$.

На практике имеют простую техническую реализацию и используются не все элементарные функции, а только основные (базисные) функции. Рассмотрим их.

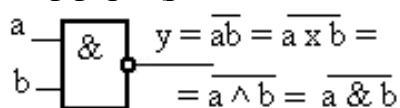
1. Логическое умножение, операция «И» – конъюнкция. Выполняется элементом – конъюнктором (рис. 2.2).



№\X	a	b	y
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

Рисунок 2.2 – Конъюнктор и его таблица истинности

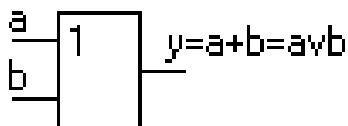
2. Операция Шеффера «И – НЕ» – отрицание конъюнкции. Выполняется элементом Шеффера (рис. 2.3).



№\X	a	b	y
0	0	0	1
1	0	1	1
2	1	0	1
3	1	1	0

Рисунок 2.3 – Элемент Шеффера и его таблица истинности

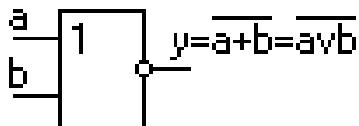
3. Логическое сложение, операция «ИЛИ» – дизъюнкция. Выполняется элементом – дизъюнктором (рис.2.4).



$\text{№}\backslash X$	a	b	y
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1

Рисунок 2.4 – Дизъюнктор и его таблица истинности

4. Операция Пирса - отрицание дизъюнкции. Логическое «ИЛИ – НЕ». Выполняется элементом Пирса (рис. 2.5).

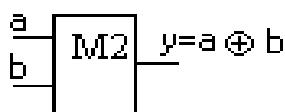


$\text{№}\backslash X$	a	b	y
0	0	0	1
1	0	1	0
2	1	0	0
3	1	1	0

Рисунок 2.5 – Элемент Пирса и его таблица истинности

5. Логическая неравнозначность или сумма по модулю два - **M2**.

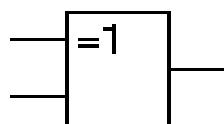
Выполняется сумматором по «модулю два» (рис. 2.6). Функция истинна на тех наборах, где число единиц нечетно.



$\text{№}\backslash X$	a	b	y
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Рисунок 2.6 – Сумматор по модулю два и его таблица истинности

Вместе с тем, в литературе встречается функция, так называемая, «исключающее ИЛИ», которая истинна, на тех наборах, где присутствует *исключительно* одна единица. Операция выполняется элементом «исключающее ИЛИ» (рис.2.7)



$\text{№}\backslash X$	a	b	y
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

Рисунок 2.7 – Элемент «исключающее ИЛИ» и его таблица истинности

Видно, что таблицы истинности совпадают. Но это только для двух переменных функции M2 и =1 – эквивалентны.

Составим таблицу истинности этих функций при числе переменных $n=3$.

№	a	b	c	M2	=1
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	0	0
4	1	0	0	1	1
5	1	0	1	0	0
6	1	1	0	0	0
7	1	1	1	1	0

Рисунок 2.8—Таблица истинности элементов M2 и =1 для трёх переменных

Видно, что они различаются в последнем наборе. При большем числе переменных это различие возрастает, поэтому функции M2 и =1 нельзя отождествлять.

Графическое изображение и условное обозначение логических элементов регламентируются ГОСТ 2.743-91 ЕСКД. Этот ГОСТ устанавливает следующие геометрические размеры (рис. 2.9).

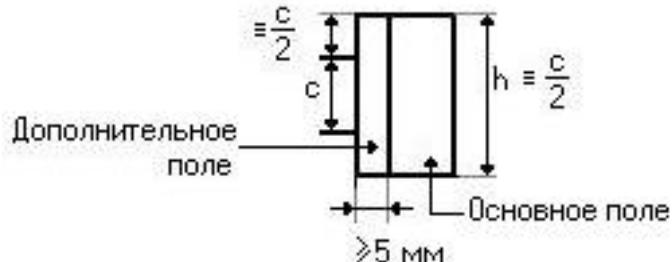


Рисунок 2.9 – Условное изображение логических элементов

Других ограничений на размеры логических элементов ГОСТ не накладывает.

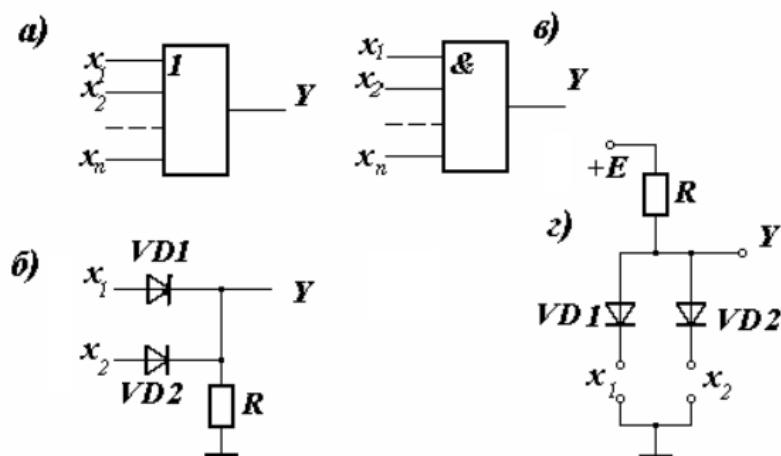
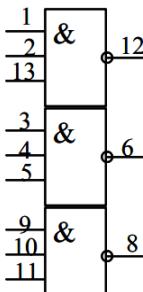
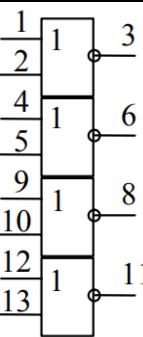
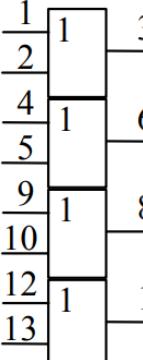
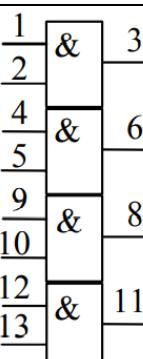


Рисунок 2.10 – Диодная реализация логических операций: дизъюнкции (а) и (б); конъюнкции (в) и (г).

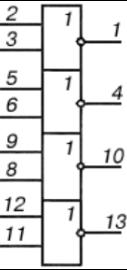
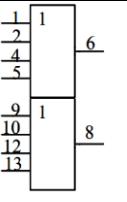
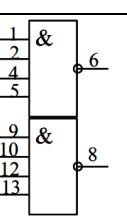
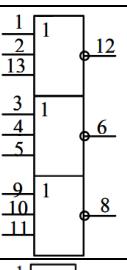
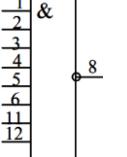
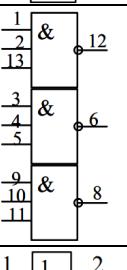
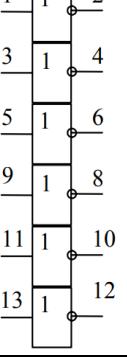
2.3 Порядок выполнения работы

1. Выпишите в соответствии со своим вариантом задания (номером бригады) тип логических элементов из таблицы 2.5.

Таблица 2.5 – Варианты задания

Номер бригады	Микросхема отечественная (аналог)	Логические элементы	Модель диода	Модель n-p-n транзистора	
1	K555ЛА4 (7410)	три логических элемента ЗИ-НЕ		1N4154	2N2712
2	K555ЛЕ1 (7402)	четыре логических элемента 2ИЛИ-НЕ		1N4305	2N2714
3	K555ЛЛ1 (7432)	Четыре логических элемента 2ИЛИ		1N4446	2N2923
4	K555ЛИ1 (7408)	четыре логических элемента 2И		1N4447	2N2924

Окончание таблицы 2.5

Номер бригады	Микросхема отечественная (аналог)	Логические элементы	Модель диода	Модель n-p-n транзистора	
5	<u>K555ЛЕ5</u> (7428)	четыре логических элемента 2ИЛИ-НЕ		1N4448	2N2925
6	KР555ЛИ6 (7421)	два логических элемента 4И		1N914	2N3393
7	<u>K555ЛА1</u> (7420)	два логических элемента 4И-НЕ		1N4454	2N3391
8	K555ЛЕ4 (7427)	три элемента ЗИЛИ-НЕ		1N4938	2N3392
9	<u>K155ЛА2</u> (7430)	логический элемент 8И-НЕ		1N914	2N3393
10	K155ЛА3 (7400)	три логических элемента ЗИ-НЕ		1N916	2N3394
11	<u>K555ЛН1</u> (7404)	шесть логических элементов НЕ		1N4449	2N3390

2. В соответствие с заданием таблицы 2.5 начертите схему одного логического элемента из своей микросхемы как показано на рисунке 2.11 элемент ЗИ-НЕ.

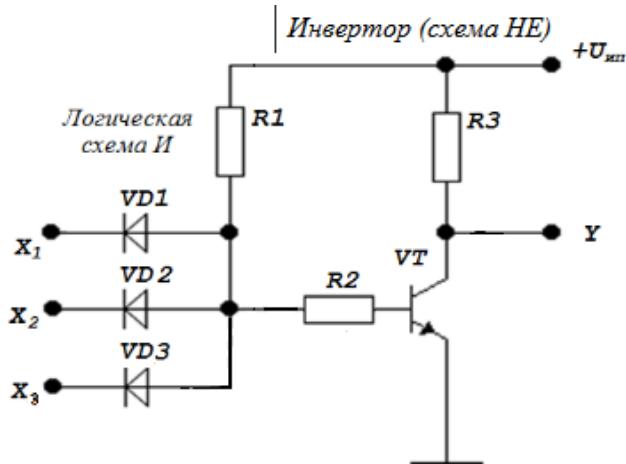
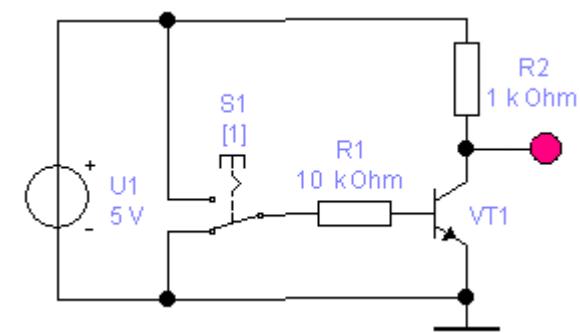


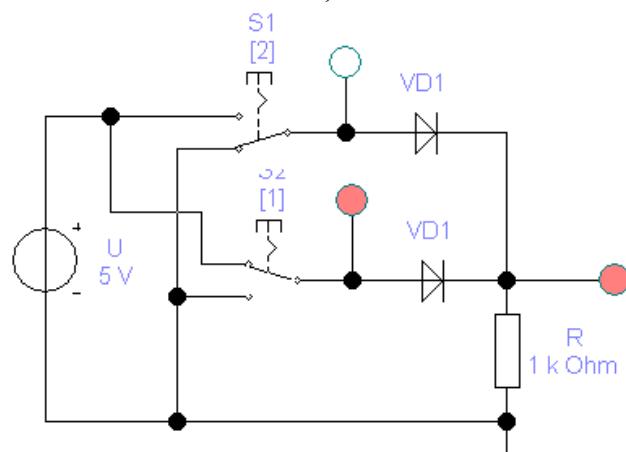
Рисунок 2.11 – Пример диодно-транзисторной реализации элемента ЗИ-НЕ

Запустите программу EWB.

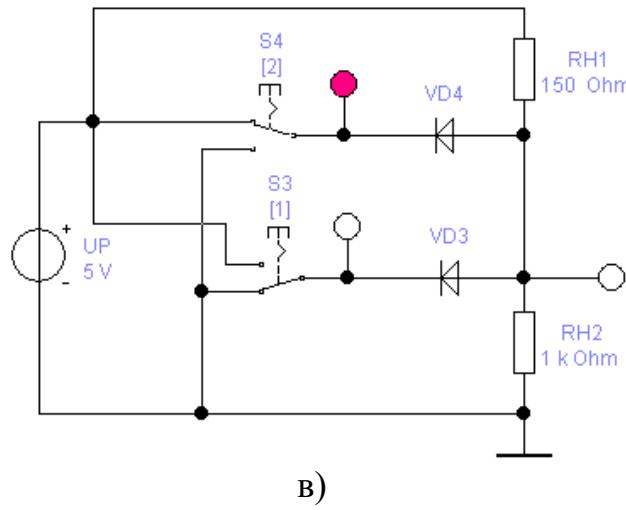
Для создания своего файла выберете курсором команду File, затем в ниспадающем меню строку New (или щелкнуть по пиктограмме) и сохраните файл по команде File > Save as в папке Цифра-model на рабочем столе, присвоив ему имя (латинскими буквами) с расширением EWB. Для построения схемы следует скопировать в буфер обмена имеющиеся элементы-файлы ne.ewb (НЕ), or.ewb (ИЛИ) и x.ewb (И) (рисунок 2.12).



a)



б)

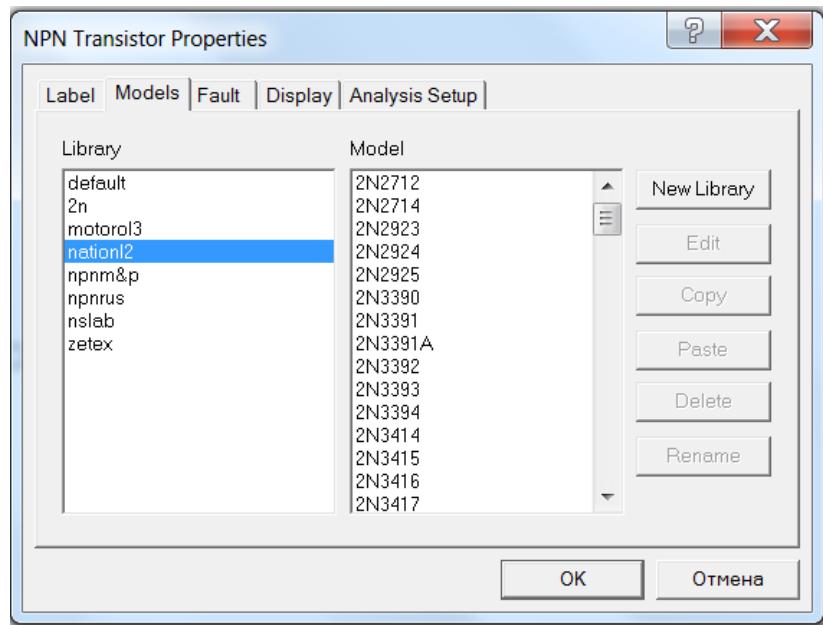


в)

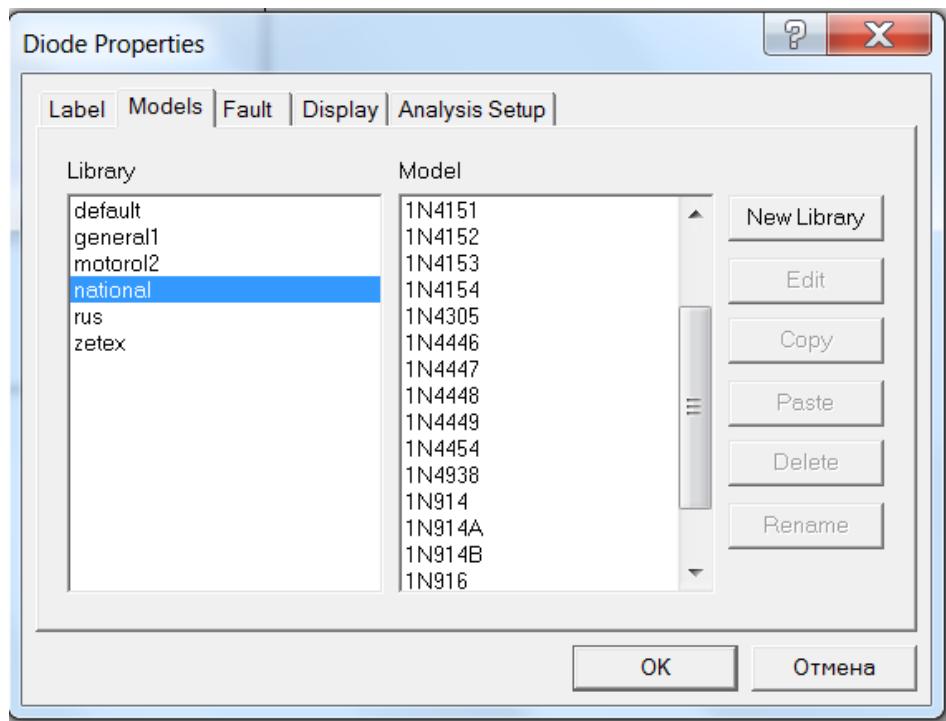
Рисунок 2.12 – Файлы реализации логических функций НЕ (а), ИЛИ (б) и И (в)

Для этого нажимаем File-Open- ne. EWB -OK – верхнюю клавишу(Use library model)-далее Edit – Select All – Edit-Copy (кнопка). Открываем свой файл –Fail-Open- имя файла- OK-Edit-Paste. Погасите цвет. Сохраните файл (Filt-Save). Аналогично поступаете с другими файлами. Составьте нужную схему.

Для дальнейшего редактирования файла Вам потребуется замена типов диодов, транзисторов на заданные по Вашему варианту. Сопротивления резисторов не меняем. Для этого элемент схемы выделяется щелчком левой кнопки мыши, его цвет меняется на красный. После появления окна свойств (Properties) выделенного элемента необходимо выбрать транзистор из библиотеки national2 (см. рис. 2.13 а) и диод из библиотеки national (рис. 2.13.б).



а)



б)

Рисунок 2.13 – Окна свойств элементов: транзисторов (а) и диодов (б)

3. После создания схемы, включите ее клавишей в правом верхнем углу экрана и используя ключи S1, S2,..., SI заполните таблицу истинности (табл. 2.6) для всех входных наборов, где разрядность равна количеству входов.

Таблица 2.6– Таблица истинности одного элемента микросхемы

x_1	x_2	...	x_n	Y
		...		
		...		
		...		
		...		

2.4 Результаты работы

Отчет должен содержать принципиальную схему логического блока и таблицу истинности, полученную в процессе моделирования.

2.5 Контрольные вопросы

- 1 Нарисуйте логическую схему И на диодах.
- 2 Нарисуйте логическую схему ИЛИ на диодах.
- 3 Нарисуйте логическую схему НЕ на биполярном транзисторе.
- 4 Что понимается под таблицей истинности логического блока?
- 5 Какие активные элементы используются для создания логических схем?

3 Лабораторная работа № 3

Синтез комбинационных цифровых устройств

3.1 Цель работы

Ознакомление с этапами синтеза комбинационных схем.

3.2 Пояснения к работе

Синтез комбинационных цифровых устройств состоит из нескольких этапов. На первом этапе требуется исходную функцию, заданную в словесной, табличной или другой формах, представить в виде логического выражения. Для этого рассмотрим основные свойства алгебры логики.

Основные свойства алгебры логики базируются на аксиомах и позволяют преобразовывать логические функции. Приведем здесь аксиомы и основные свойства алгебры логики. Заметим, что некоторые свойства, в силу их важности, в технической литературе трактуются как законы. Знак + это дизъюнкция, знак * это конъюнкция.

АКСИОМЫ алгебры логики:

$$\begin{array}{ll} 0 * 0 = 0 & 0 + 0 = 0 \\ 0 * 1 = 0 & 0 + 1 = 1 \\ 1 * 0 = 0 & 1 + 0 = 1 \\ 1 * 1 = 1 & 1 + 1 = 1 \end{array}$$

ЗАКОНЫ алгебры логики:

1. Закон одинарных элементов

$$\begin{array}{ll} 1 * X = X & 0 * X = 0 \\ 1 + X = 1 & 0 + X = X \end{array}$$

2. Законы отрицания:

а) закон дополнительных элементов

$$X + \overline{X} = 1 \quad X * \overline{X} = 0$$

б) двойное отрицание

$$\overline{\overline{1}} = 0 \quad \overline{\overline{0}} = 1 \quad \overline{\overline{1}} = 1 \quad \overline{\overline{0}} = 0 \quad \overline{\overline{X}} = X \quad \overline{\overline{X}} = \overline{X},$$

поэтому отрицание можно переносить из одной части равенства в другую;

в) закон двойственности (правило Моргана):

$$\overline{A + B + C} = \overline{A} * \overline{B} * \overline{C}$$

Читается так “Отрицание дизъюнкции есть конъюнкция отрицаний и наоборот - отрицание конъюнкции есть дизъюнкция отрицаний”:

$$\overline{A * B * C * D} = \overline{A} + \overline{B} + \overline{C} + \overline{D}$$

Правило справедливо для любого числа переменных.

3. Комбинационные законы.

Они во многом соответствуют обычной алгебре, но есть и отличия:

а) тавтологии (многократное повторение):

$$X + X + X + X = X$$

$$X * X * X * X = X$$

б) переместительности:

$$A + B + C + D = A + C + B + D$$

в) сочетательности: D

$$A + B + C + D = A + (B + C) + = A + B + (C + D)$$

г) распределительности:

$$X_1(X_2 + X_3) = X_1X_2 + X_1X_3$$

$$X_1 + X_2X_3 = (X_1 + X_2)(X_1 + X_3) = / \text{докажем это путём раскрытия скобок} / = \\ = X_1X_1 + X_1X_3 + X_1X_2 + X_2X_3 = X_1(1 + X_3 + X_2) + X_2X_3 = X_1 + X_2X_3$$

д) правило поглощения (одна переменная поглощает другие):

$$X_1 + X_1X_2X_3 = X_1(1 + X_2X_3) = X_1$$

е) правило склеивания (выполняется только по одной переменной):

$$A\bar{B}C + ABC = AC(\bar{B} + B) = AC$$

Так же как в обычной математике имеется старшинство операций:

- 1) действие в скобках;
- 2) операция с одним операндом (одноместная операция) – НЕ;
- 3) конъюнкция И;
- 4) дизъюнкция ИЛИ;
- 5) сумма по модулю два.

Операции одного ранга выполняются слева направо в порядке написания.

Понятие базиса

С помощью ограниченного набора элементарных функций можно представить любую, сколь угодно сложную функцию алгебры логики. Такой набор элементарных функций называют *базисом или функционально полным набором*.

Базисов может быть много:

- | | |
|---------------|------------------------------------|
| 1. И, ИЛИ, НЕ | 2. И, НЕ |
| 3. И – НЕ | 4. НЕ – И |
| 5. ИЛИ, НЕ | 6. ИЛИ – НЕ |
| 7. НЕ – ИЛИ | 8. «0», «1», НЕ, $\geq n$ и другие |

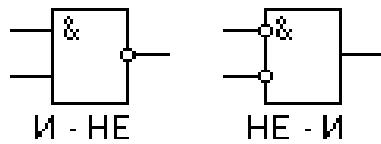


Рисунок 3.1 – Некоторые базисы

Используя законы алгебры логики, можно переходить от одного базиса к другому и реализовывать функции на разных элементах.

Например, пусть имеется элемент ЗИ-НЕ, а необходимо реализовать следующие функции (операции):

1. НЕ;
2. И (для двух переменных-2И);
3. ИЛИ (для двух переменных-2ИЛИ).

Реализуем эти операции:

1. Операция НЕ получается на основании закона тавтологии (рис.3.2).

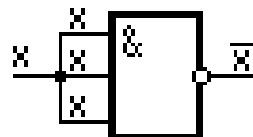


Рисунок 3.2 – Инвертор на элементе Шеффера(ЗИ-НЕ)

2. Операция И получается на основании законов тавтологии и двойного отрицания (рис. 3.3).

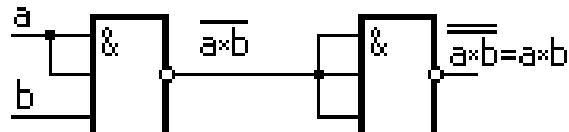


Рисунок 3.3 – Конъюнктор на элементах Шеффера

3. Операция ИЛИ получается на основании правила двойственности-искусственно делают двойное отрицание и нижнюю черту преобразуют по правилу Моргана $a + b = \overline{\overline{a} + \overline{b}} = \overline{\overline{a} * \overline{b}}$. Тогда получаем следующую реализацию (рис. 3.4).

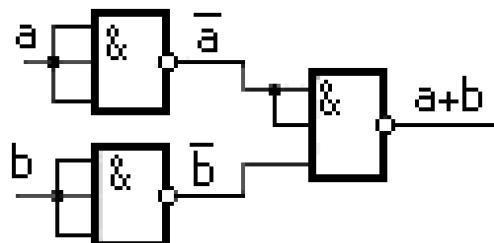


Рисунок 3.4 – Дизъюнктор на элементах Шеффера

Формы представления функций алгебры логики

Функции алгебры логики могут быть заданы различными способами:

- таблицей истинности;
- в аналитической форме;
- в числовом виде.

Таблица истинности уже была рассмотрена. В ней все наборы логических переменных следуют строго в порядке возрастания их двоичного номера и нумеруются целыми числами от 0 до $2^n - 1$, где n – число переменных функции.

При аналитической записи используются так называемые нормальные формы.

Для лучшего понимания материала введем некоторые понятия:

- терм – компонент выражения;
- ранг терма – число переменных, входящих в терм;
- элементарная дизъюнкция – дизъюнктивный терм или макстерм это дизъюнкция произвольного числа попарно независимых переменных.

Например, $(\bar{a} + \bar{b} + c + d)$ – макстерм 4-го ранга, $X_1 + \bar{X}_2 + \bar{X}_3$ – макстерм 3-го ранга, $(a + \bar{b} + c + \bar{a})$ – это не макстерм, т.к. переменные a и \bar{a} попарно зависимые;

- элементарная конъюнкция – конъюнктивный терм или минтерм – конъюнкция произвольного числа попарно независимых переменных.

Например, $X_1 X_2 X_3$ – минтерм 3-го ранга, $ad \bar{c} \bar{d}$ – это не минтерм, так как переменные d и \bar{d} зависимые.

Для аналитической записи функций используют две формы:

- 1) дизъюнктивную нормальную форму – ДНФ;
- 2) конъюнктивную нормальную форму – КНФ.

ДНФ это дизъюнкция минтермов различного ранга:

$$f(a, b, c) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b} + \bar{a}\bar{c} + b$$

КНФ это конъюнкция макстермов различного ранга:

$$f(X_1 X_2 X_3 X_4) = (X_1 + \bar{X}_2 + X_3)(\bar{X}_1 + \bar{X}_2 + X_3 + X_4)(X_1 + X_2)$$

Если все термы, входящие в нормальную форму, имеют одинаковый и максимальный ранг, равный числу переменных функции n , то такая форма называется совершенной. При этом минтерм называется конституентой (составляющей) единицы (КЕ), а макстерм конституентой нуля (КН).

$$F(a,b,c) = \bar{a}\bar{b}c + abc + a\bar{b}\bar{c} + a\bar{b}\bar{c} - \text{это СДНФ}$$

$$F(a,b,c,d) = (a + b + \bar{c} + d)(\bar{a} + b + \bar{c} + d)(\bar{a} + \bar{d} + \bar{c} + d) - \text{это СКНФ}$$

Таким образом, совершенная дизъюнктивная нормальная форма есть дизъюнкция конституент единицы, а СКНФ есть конъюнкция конституент нуля.

Совершенные формы составляются по таблице истинности функции. СДНФ составляется по такому правилу: для каждого набора переменных, на котором функция истинна, записывают минтерм ранга n , в котором с отрицанием берутся переменные, имеющие нулевые значения на данном наборе. Все минтермы объединяют дизъюнктивно.

Пусть, например, имеем произвольную функцию трёх переменных, заданную такой таблицей истинности (рис. 3.5).

$\# X$	a	b	c	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Рисунок 3.5 – Таблица истинности произвольной функции

Для номеров наборов $N=1, 3, 6$ и 7 получаем следующую СДНФ:

$$f(a,b,c) = \bar{\bar{a}}\bar{b}c + \bar{a}\bar{b}c + a\bar{b}\bar{c} + abc$$

СКНФ также записывают по таблице истинности по правилу: для каждого набора переменных, на котором функция ложна, записывают макстерм ранга n , в котором с отрицанием берутся переменные, имеющие единичные значения на данном наборе. Все макстермы объединяют конъюнктивно. Тогда для этой же функции, для номеров наборов $N = 0, 2, 4$ и 5 , получаем СКНФ:

$$f(a,b,c) = (a + b + c)(a + \bar{b} + c)(\bar{a} + b + c)(\bar{a} + b + \bar{c})$$

Очевидно, что СДНФ и СКНФ полностью дуальны.

Для компактной записи функций используют числовую форму, в которой задаются только номера наборов. Числовая форма для нашей СДНФ:

$$f(a,b,c) = V(1,3,6,7)$$

Числовая форма для СКНФ:

$$f(a,b,c) = \Lambda(0,2,4,5)$$

При физической реализации функций обычно используется СДНФ. Пусть например требуется построить логическую схему, которая реализует функцию с таблицей истинности рис. 3.5.

Для реализации схемы запишем СДНФ:

$$f(a,b,c) = \overline{\overline{abc}} + \overline{\overline{abc}} + \overline{abc} + abc$$

Каждый минтерм реализован своим конъюнктором. Инверсии на схеме выполнены отдельными элементами. Получаем (рис. 3.6):

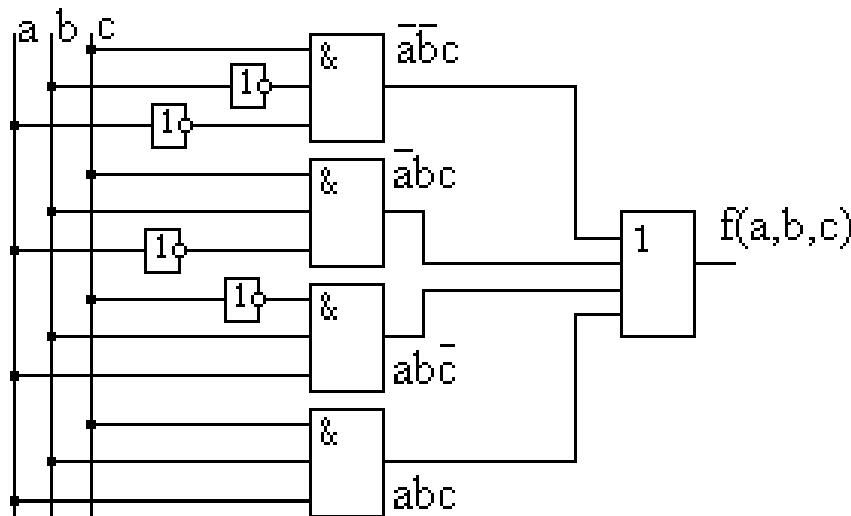


Рисунок 3.6 – Схемная реализация функции с таблицей истинности (рис.3.5)

3.3 Порядок выполнения работы

1. Выпишите согласно своему варианту задания (номеру бригады) функцию четырех переменных для синтеза комбинационной схемы из таблицы 3.2.

Таблица 3.2 – Варианты задания

Номер бригады	Функция F	Номер бригады	Функция F
1	$(\overline{A} \vee B) \oplus \overline{C} \wedge D \vee B$	6	$(A \oplus B) \oplus A \wedge B \vee D$
2	$(A \wedge B) \oplus (D \vee B) \wedge C$	7	$(A \vee \overline{B}) \oplus (\overline{A} \wedge \overline{B}) \wedge D$
3	$C \vee B \oplus \overline{A} \wedge D$	8	$(\overline{A} \wedge C) \oplus (D \vee B) \wedge C$
4	$\overline{(A \vee C)} \oplus A \wedge D$	9	$(C \oplus D) \vee (\overline{A} \oplus \overline{B}) \wedge B$
5	$A \vee B \oplus C \wedge D$	10	$D \oplus B \wedge (\overline{A} \oplus \overline{C}) \vee A$

Нарисуйте схему на листе бумаги на произвольных логических элементах, реализующую Вашу функцию учитывая старшинство операций.

Для создания своего файла выберете курсором команду File, затем в ниспадающем меню строку New (или щелкнуть по пиктограмме) и сохраните файл на диске D по команде File > Save As, присвоив ему имя(идентификатор), из латинских букв. Для построения схемы необходимо использовать источник питания для подачи на вход устройства логической «1» и землю – для подачи сигнала «0». При нажатии на кнопку Sources выберите эти элементы и переместите левой кнопкой мыши на рабочий стол как показано на рисунке 3.7.

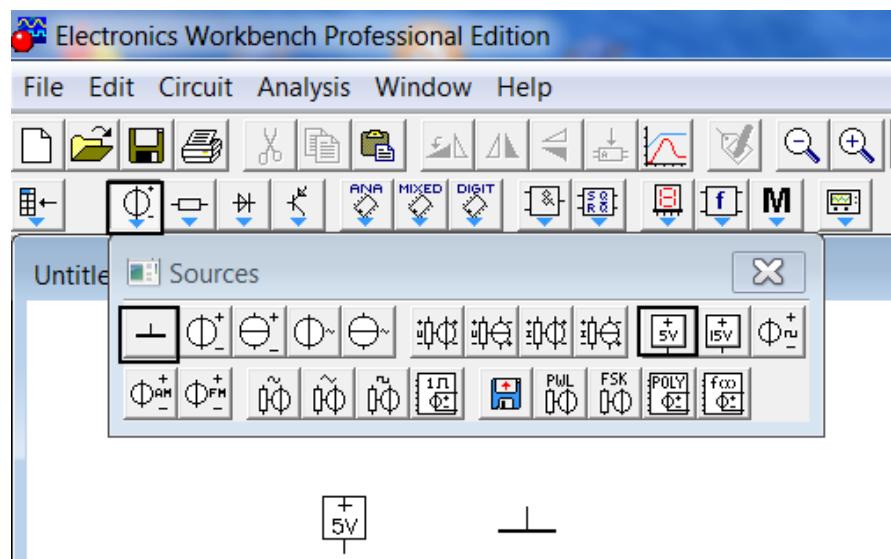


Рисунок 3.7 – Панель Sources

Для перебора команд на входе устройства используются четыре ключа, которые находятся на панели Basic (см. рис. 3.8). Присвойте им имена (Label) A,B,C,D и сигнал переключения(Value) 1,2,3,4. Логические элементы расположены на панели Logic Gates (см. рис.3.9).

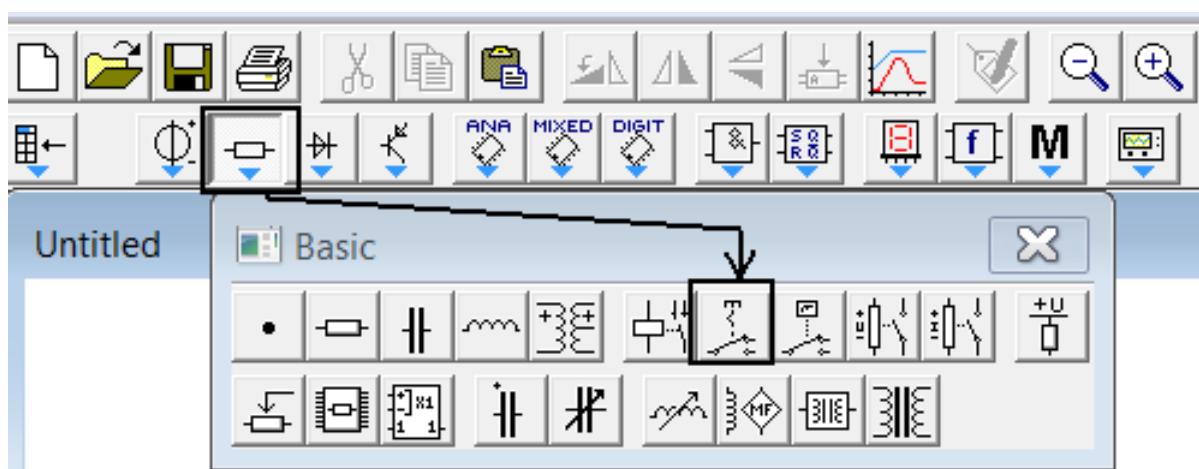


Рисунок 3.8 – Панель Basic

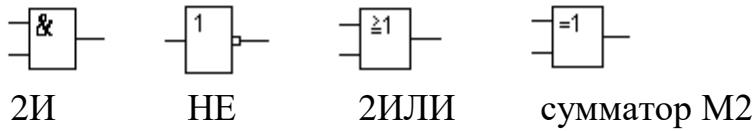
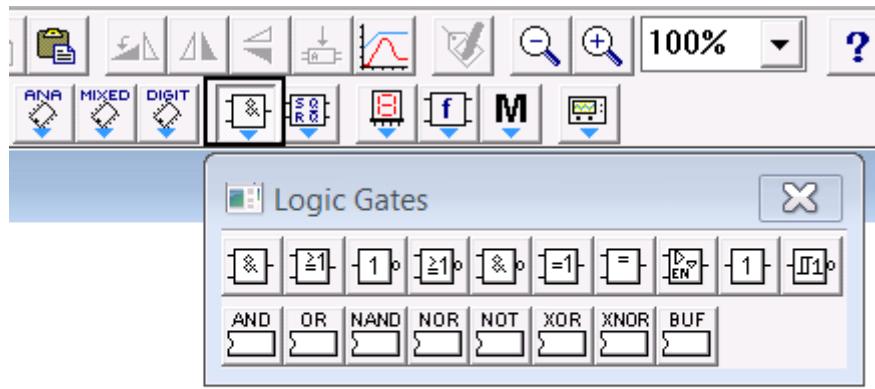


Рисунок 3.9 – Панель Logic Gates

Число входов логических элементов можно изменить. Например, для получения четырехходового элемента задаем количество входов, нажав левой кнопкой мыши на элементе, открываете вкладку Properties и подтвердить свой выбор как показано на рисунке 3.10.

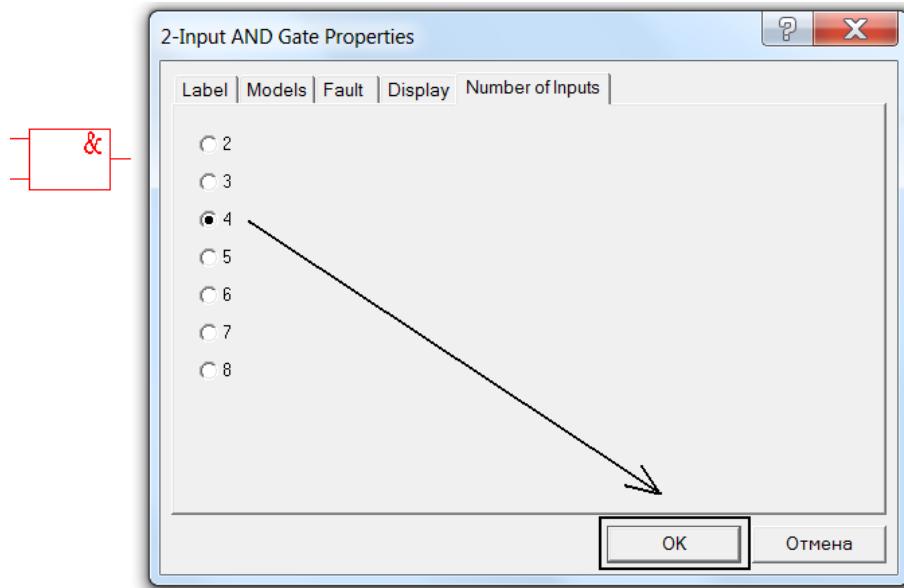


Рисунок 3.10 – Изменение количества входов логического элемента

Выполните соединение элементов схемы, для этого подведите мышь к концу вывода элемента до появления черной точки и нажав ее левую клавишу протяните линию до конца вывода другого элемента.

Для составления таблицы истинности Вашего комбинационного устройства потребуются индикаторы, которые расположены на панели Indicator (см. рис. 3.11).

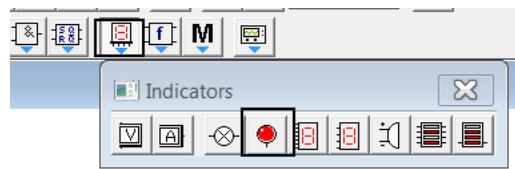


Рисунок 3.11 – Панель Indicator

Выполните соединение всех элементов Вашего комбинационного устройства и присоедините индикаторы как показано на рисунке 3.12.

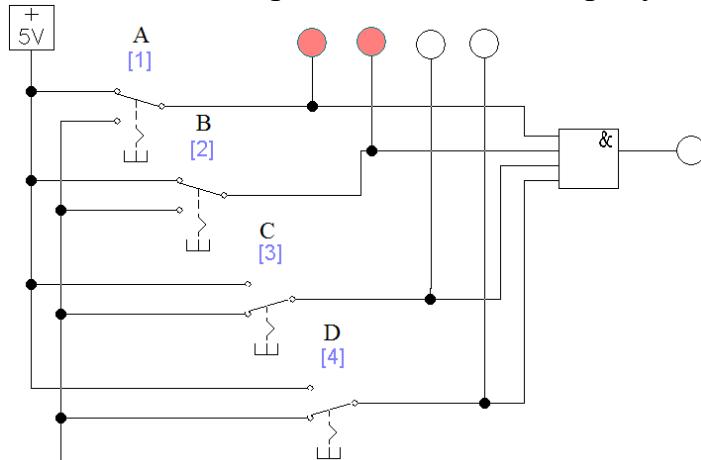


Рисунок 3.12 – Присоединение индикаторов

2. После создания схемы, включите ее клавишей в правом верхнем углу экрана и, используя ключи A,B,C,D заполните таблицу истинности (табл. 3.3) для всех наборов входных переменных.

Таблица 3.3 – Таблица истинности схемы

A	B	C	D	F
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	1	

Выключите макет.

Исходную логическую функцию, используя законы алгебры логики, приведите к базису И-НЕ, начертите схему на элементах 2И-НЕ.

3.4 Результаты работы

Отчет должен содержать принципиальную схему на произвольных логических элементах, таблицу истинности, полученную в процессе моделирования и преобразования исходной функции к элементам 2И-НЕ.

3.5 Контрольные вопросы

- 1 Сформулируйте законы алгебры логики, которых нет в обычной алгебре.
- 2 Для чего применяется двойное отрицание переменной (терма)?
- 3 Сформулируйте правило склеивания.
- 4 В чем отличие аксиом алгебры логики от правил двоичной арифметики?
- 5 Когда используют СКНФ, а когда СДНФ?

4 Лабораторная работа № 4

Минимизация логических устройств

4.1 Цель работы

Изучить методы минимизации логических устройств аналитическим способом и с помощью компьютерного программного моделирования.

4.2 Пояснения к работе

Прямой способ построения схемы по структурной формуле обычно не дает удовлетворительных результатов с практической точки зрения: применяются разнотипные логические элементы с произвольным числом входов. Задача минимизации сводится к тому, чтобы после получения аналитической формы записи структурной формулы $Y = f(x_1, x_2, \dots, x_n)$, выполнить ее минимизацию, т. е. найти такую форму записи, которая потребует при реализации наименьшего числа элементов. Рассмотрим такой пример. Пусть требуется построить логическую схему, которая реализует следующую таблицу истинности (рис. 4.1)

$\# \backslash X$	a	b	c	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Рисунок 4.1 – Таблица истинности для логической схемы

Для реализации схемы запишем СДНФ:

$$f(a, b, c) = \overline{\overline{a}}\overline{b}\overline{c} + \overline{a}\overline{b}\overline{c} + a\overline{b}\overline{c} + ab\overline{c}$$

Каждый минтерм реализован своим конъюнктором. Инверсии выполнены на входах, чтобы не пользоваться дополнительными инверторами и не загромождать схему. Получаем (рис. 4.2):

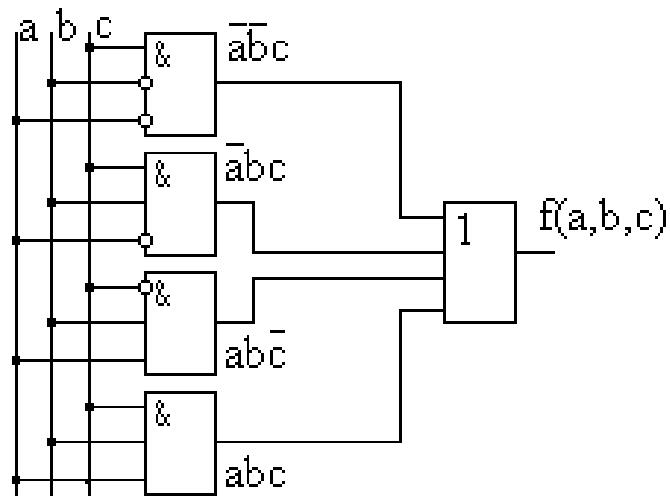


Рисунок 4.2 – Схемная реализация таблицы истинности (рис.4.1)

Сложность логической схемы принято оценивать общим числом входов всех ее элементов, которые условно называют *ценой схемы*. Инверторы не учитываются.

Найдём, чему равна цена нашей схемы: $\Pi_1=12+4=16$

Она складывается из числа входов всех конъюнкторов (элементы первого уровня) и числа входов дизъюнктора (элемент второго уровня).

Число входов элементов 1-го уровня равно числу символов в записи функции. Число входов элементов 2-го уровня равно числу термов в записи функции. То есть цену схемной реализации можно подсчитать сразу по исходной логической формуле.

Используя законы алгебры логики, попытаемся упростить исходную функцию. Для этого сгруппируем и вынесем за скобки:

$$f(a, b, c) = \bar{a}c(\bar{b} + b) + ab(\bar{c} + c) = \bar{a}c + ab$$

Очевидно, что реализация такой формулы значительно проще, ее цена $\Pi_2=4+2=6$.

Логическая формула с наименьшим числом логических связей называется *минимальной*.

Таким образом, получаем минимальную дизъюнктивную нормальную форму (МДНФ).

Процесс отыскания минимальной формы называется *минимизацией логической функции*, или просто *минимизацией*.

Минимизировать функции можно тремя методами:

- 1) расчетным путем, используя законы алгебры логики;
- 2) графическим путем (метод карт Карно или диаграмм Вейча), используя специальные карты;
- 3) расчетно-графическим путем (метод Квайна и его модификации).

Расчетный метод мы уже разобрали выше. Метод Квайна используется при числе переменных больше шести, хорошо алгоритмизируется и программируется. На его основе разработаны системы автоматизированного проектирования и различные стандартные программы минимизации логических функций любого числа переменных (в том числе и EWB). Но они не всегда доступны и не оправданы при малом числе независимых переменных. Метод карт Карно хорошо работает при числе переменных меньше шести, прост и удобен для оперативного использования. Тем более, что большинство устройств, с которыми имеет дело разработчик, оперируют именно с малым числом переменных (3...5). Поэтому рассмотрим этот метод подробнее.

Метод карт Карно

Карта Карно (минимизирующая карта) – это развертка некоторой объемной фигуры на плоскости. Карта Карно состоит из клеток, число которых равно числу наборов переменных функции. Каждая клетка соответствует строго определённому набору.

Например, карта Карно одной переменной (рис 4.3):

$n = 1$, число наборов $N = 2^n = 2$.



Рисунок 4.3 – Карта Карно одной переменной

Карта Карно двух переменных (рис.4.4): $n = 2$, число наборов $N = 2^2 = 4$.



Рисунок 4.4 – Карта Карно двух переменных

Крайние клетки, соответствующие комбинациям 00 и 10, являются соседними и отличаются одной переменной a .

Переменные в карте могут располагаться произвольно, но *любые соседние по вертикали или по горизонтали клетки могут отличаться не более чем одной переменной* (склеиваются по одной переменной).

Карта Карно трёх переменных (рис.4.5): $n = 3$, число наборов $N = 2^3 = 8$.

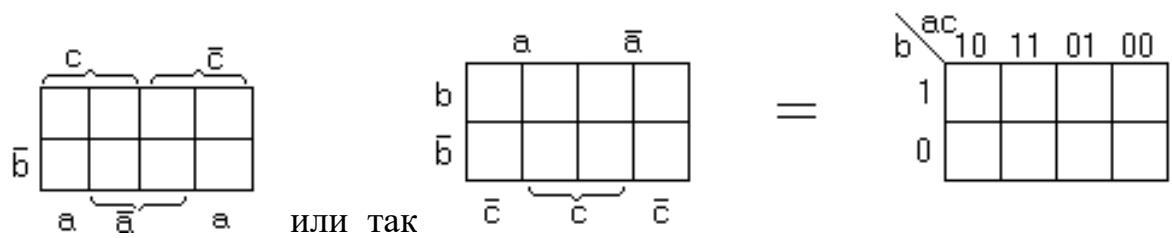


Рисунок 4.5 – Карта Карно трёх переменных

Если имеется функция трёх переменных, заданная следующей таблицей истинности (рис. 4.6):

$\# \backslash X$	A	B	C	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1

Рисунок 4.6 – Таблица истинности произвольной функции

Тогда соответствующая ей карта Карно выглядит так (рис. 4.7):

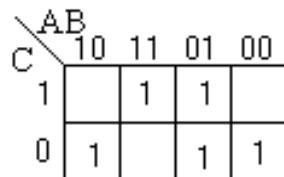


Рисунок 4.7 – Карта Карно функции рис. 4.6

Обычно нули в карту не пишут, а заносят только единицы. Карта Карно четырёх переменных (рис. 4.8): $n = 4$, $N = 16$.

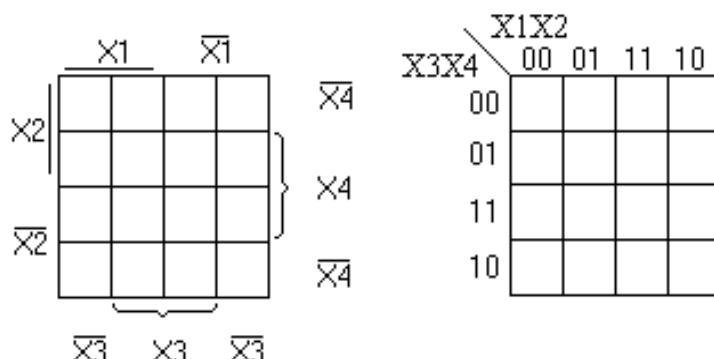


Рисунок 4.8 – Карта Карно для функции четырёх переменных

В этих картах переменные расположены по-разному, но они обе правильны, так как любые соседние клетки по горизонтали или вертикали отличаются только одной из переменных. Клетки верхнего ряда соседние с клетками нижнего ряда, а клетки крайнего правого столбца соседние с клетками крайнего левого столбца. Удобно считать X₁-старшая, а X₄ – младшая переменная т.е. X₁X₂X₃X₄

Карта Карно пяти переменных (рис. 4.9): n = 5, N = 32. Это две шестнадцатиклеточных карты, отличающиеся только одной (пятой) переменной.

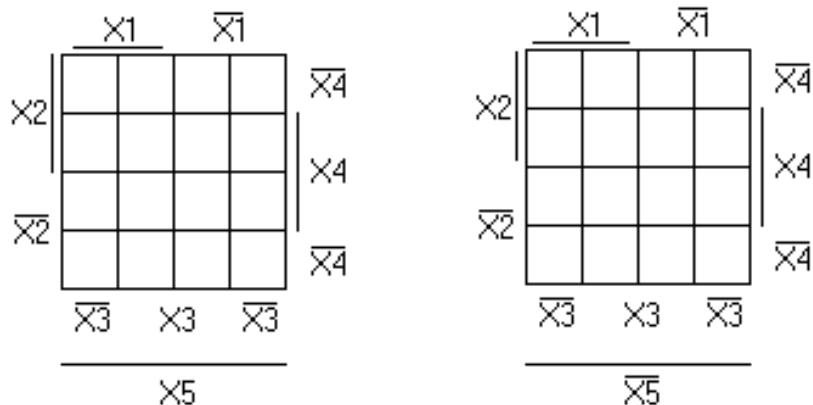


Рисунок 4.9 – Карта Карно функции пяти переменных

Можно составить карту Карно и для шести переменных, но в этом обычно нет необходимости.

Карту Карно заполняют по таблице истинности – обычно пишут только единицы, в остальных клетках подразумеваются нули.

После заполнения карты к минимизации. Суть минимизации: охватить все единицы карты Карно наименьшим числом кубов наиболее высокого ранга. Из каждого куба выписывают минтерм общих переменных. Минтермы объединяют дизъюнктивно.

Куб – это прямоугольный или квадратный контур, содержащий клетки только с единицами:

одна единица – куб «0»-го ранга, так как $2^0 = 1$;

две единицы - куб «1»-го ранга, т.к. $2^1 = 2$;

четыре единицы - куб «2»-го ранга, т.к. $2^2 = 4$;

восемь единиц - куб «3»-го ранга, т.к. $2^3 = 8$;

шестнадцать единиц - куб «4»-го ранга, т.к. $2^4 = 16$ и т.д.

Куб не может содержать другое число единиц и клеток с нулями. Одни и те же единицы одновременно могут принадлежать нескольким кубам, чтобы ранг куба был наибольшим. Тогда форма будет именно минимальной.

Пример 1.

Найти МДНФ такой функции: F(a,b,c)=V(1,3,6,7) . Составим её таблицу истинности, которая приведена на рис. 4.10.

$\# \backslash X$	a	b	c	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Рисунок 4.10 – Таблица истинности функции

Заполняем карту Карно.

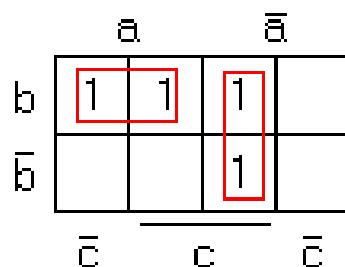


Рисунок 4.11 – Карта Карно функции рис. 4.10

Здесь следует провести два куба первого ранга и составить МДНФ:

$$f_{\min} = ab + \bar{a}c$$

Запишем СДНФ исходной функции и преобразуем её по законам алгебры логики $f = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + ab\bar{c} + abc = \bar{a}c + ab$. Получим тот же результат.

Представим карту Карно этой же функции по-другому (рис.4.12)

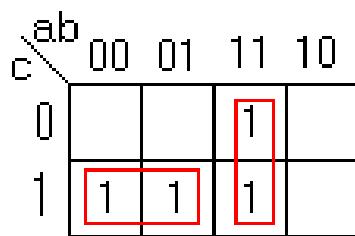


Рисунок 4.12 – Карта Карно функции рис. 4.10

Видно, что результат такой же: $f_{\min} = ab + \bar{a}c$.

Пример 2.

Минимизировать функцию трёх переменных: $F(a,b,c) = \wedge(0,4,5)$. Начинаем с составления таблицы истинности (рис. 4.13).

$\# \backslash X$	a	b	c	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Рисунок 4.13 – Таблица истинности

Карта Карно будет такой:

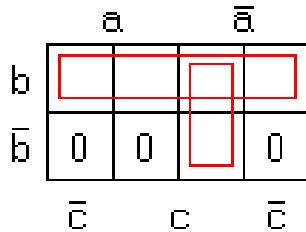


Рисунок 4.14 – Карта Карно функции (рис. 4.13)

И соответствующая минимальная форма $F_{\min} = b + \bar{a}\bar{c}$. Здесь первый куб содержит четыре единицы (ранг куба равен $r = 2$). Число переменных в минтерме равно $(n - r)$, где $n = 3$, $n - r = 3 - 2 = 1$ – количество переменных из первого куба.

Из второго куба выписываем $n - r = 3 - 1 = 2$ – две переменных.

Составим СДНФ и выполним склеивание минтермов (конституент единицы) каждого с каждым:

$$\begin{aligned} f(a,b,c) &= \overline{\bar{a}\bar{b}\bar{c}} + \overline{\bar{a}\bar{b}\bar{c}} + \overline{\bar{a}\bar{b}\bar{c}} + \overline{a\bar{b}\bar{c}} + a\bar{b}\bar{c} + \\ &= \overline{\bar{a}\bar{c}} + \overline{ab} + \overline{b\bar{c}} + \overline{bc} + ab = \overline{\bar{a}\bar{c}} + b + b = \overline{\bar{a}\bar{c}} + b \end{aligned}$$

Первый склеиваем со вторым, затем с третьим, с четвёртым и т.д. Далее второй склеиваем с третьим, с четвёртым и т.д. Все минтермы пройдут через склеивание. После первого склеивания выполняется второе и т.д. пока оно возможно. Напомним, что минтермы для склеивания могут отличаться только одной переменной.

Пример 3.

Минимизировать функцию двух переменных:

$$f_1 = \overline{X_1 * X_2} + \overline{X_1} * \overline{X_2} + \overline{X_1} * \overline{X_2}$$

Первоначально приведем логическую формулу к нормальному виду СДНФ:

$$f_1 = \overline{X_1 * X_2} * \overline{\overline{X_1} * \overline{X_2}} + \overline{X_1} * \overline{X_2} = (\overline{X_1} + \overline{X_2}) * (X_1 + X_2) + \overline{X_1} * \overline{X_2} = \\ = \overline{X_1} * X_1 + \overline{X_1} * X_2 + X_1 * \overline{X_2} + X_2 * \overline{X_2} + \overline{X_1} * \overline{X_2} = \overline{X_1} * X_2 + X_1 * \overline{X_2} + \overline{X_1} * \overline{X_2}$$

Далее составим карту Карно (рис. 4.15).

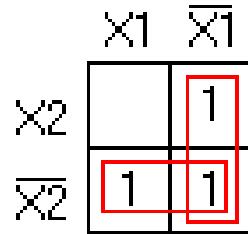


Рисунок 4.15 – Карта Карно функции

По которой получаем $f_{\min} = \overline{X_1} + \overline{X_2}$. Этот же результат можно получить склеиванием минтермов (выполните самостоятельно).

Пример 4.

Минимизировать функцию четырёх переменных, карта Карно которой представлена на рис. 4.16.

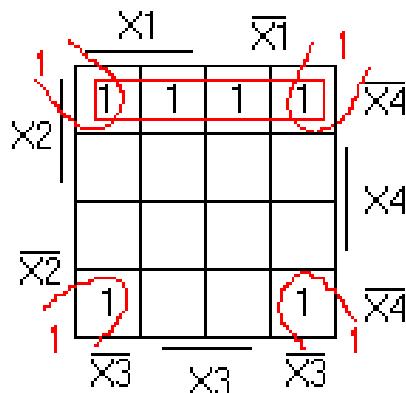


Рисунок 4.16 – Карта Карно исходной функции

Проводим два контура второго ранга и получаем

$$f_{\min} = \overline{X_3} * \overline{X_4} + X_2 * \overline{X_4}$$

Цена схемы равна $\Pi = 4 + 2 = 6$.

Можно в карте Карно объединить нули (если их число меньше чем единиц), но при этом получаем инверсную функцию: $f_{\min} = \overline{X_4} + \overline{X_2} * X_3$. Здесь проведены два куба – третьего и второго ранга. Цена схемы получается меньше. Ц = 2 + 2 = 4. Её реализация на произвольных элементах имеет вид (рис.4.17).

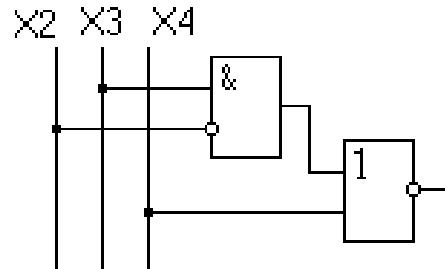


Рисунок 4.17 – Схемная реализация функции (рис. 4.16)
Отрицание можно перенести в правую часть, что не отражается на цене.

$$f_{\min} = \overline{X_4} + \overline{X_2} * X_3$$

Чем меньше цена, тем лучше. Поэтому минимизировать по карте Карно следует и по единицам, и по нулям. К реализации принимают формулу с наименьшей ценой.

Пример 5.

Имеем функцию пяти переменных, заданную картой Карно (рис. 4.18).

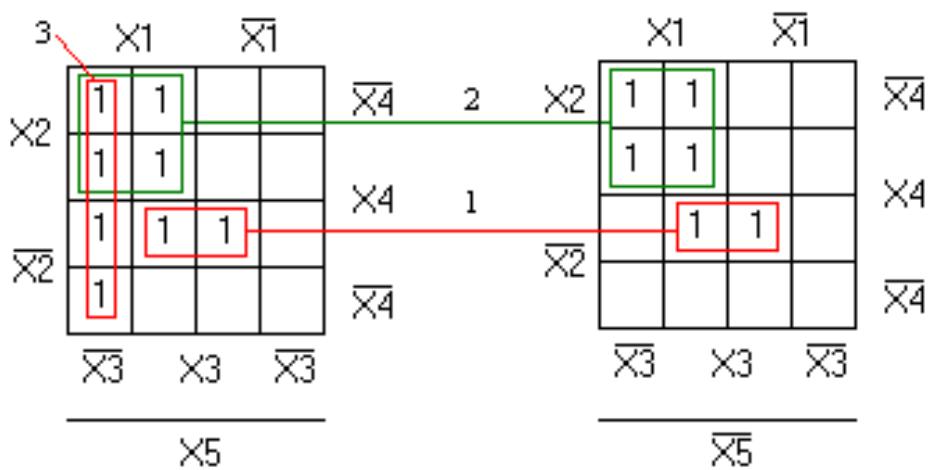


Рисунок 4.18 – Карта Карно исходной функции

Здесь проводим три куба: 1 – куб второго ранга, 2 – куб третьего ранга, 3 – куб второго ранга. Записываем минимальную форму:

$$f_{\min} = \overline{X_2}X_3X_4 + X_1X_2 + X_1\overline{X_3}X_5.$$

Находим цену $\Pi = 8 + 3 = 11$. Найдите цену схемы после объединения нулей.

На практике встречаются функции, значения которых на некоторых наборах нас не интересуют. Такие функции называются не полностью определенными или недоопределенными. Таблицы истинности таких функций обычно дополняют нулями или единицами в карте Карно, чтобы получить наиболее минимальную форму.

4.3 Порядок выполнения работы

1. Выпишите в соответствии со своим вариантом задания (номером бригады) логическую функцию устройства из таблицы 4.1.

Таблица 4.1 – Варианты задания

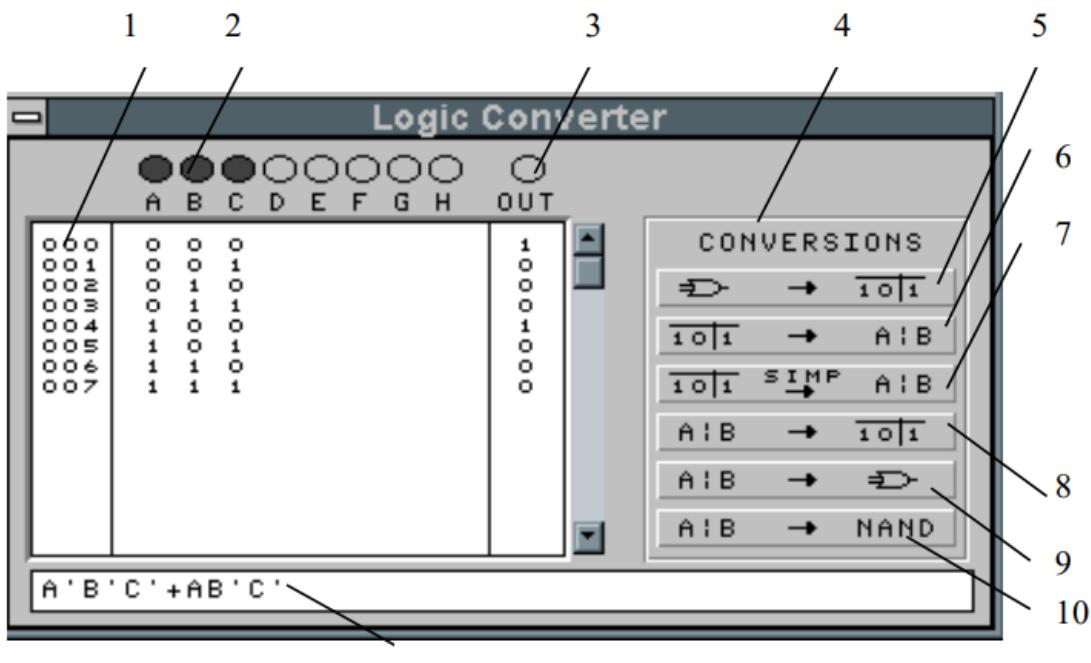
Номер бригады	Функция F
1	$\overline{A} \cdot \overline{B} + A \cdot \overline{D}B + DB + C$
2	$A \cdot (\overline{A} + B) + B \cdot (\overline{B} + C) + C$
3	$\overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot D \cdot C + A \cdot \overline{C}$
4	$\overline{A} \cdot \overline{B} + \overline{A} \cdot C + B \cdot \overline{D}$
5	$\overline{A} \cdot D + C \cdot B + \overline{A} \cdot \overline{B}$
6	$(\overline{A} \cdot \overline{B} + D \cdot \overline{B})C + A \cdot B$
7	$(C + B) \cdot (A + \overline{B}) \cdot (\overline{A} + D)$
8	$A \cdot \overline{B} + \overline{A} \cdot \overline{B} + \overline{C} + B \cdot D$
9	$(A \cdot \overline{B} + C \cdot D)(\overline{A} + B) \cdot \overline{C}$
10	$A \cdot B \cdot C + D(\overline{A} + B)$

Для создания своего файла выберете курсором команду File, затем в ниспадающем меню строку New (или щелкнуть по пиктограмме ) и сохраните файл по команде File > Save As, присвоив ему имя из латинских букв в папке временного хранения Student (C:\Users\Пользователь\Desktop\Цифра). Для выполнения минимизации откройте меню Instruments (рис. 4.19) и переместите на рабочее поле логический конвертор.



Рисунок 4.19 – Логический конвертор (Logic Converter)

Откройте переднюю панель логического конвертора (рис. 4.20).



1 – номер входного набора; 2 – активные аргументы; 3 – выходное значение функции; 4 – панель конвертирования; 5 – кнопка для построения таблицы истинности по заданной схеме; 6 – кнопка для получения логического выражения в форме СДНФ по таблице истинности; 7 – кнопка для упрощения таблицы истинности; 8 – кнопка для построения таблицы истинности по заданной формуле; 9 – кнопка для построения схемы в базисе И-ИЛИ-НЕ по формуле, заданной в окне 11; 10 – кнопка для построения схемы в базисе И-НЕ по формуле, заданной в окне 11

Рисунок 4.20 – Передняя панель логического конвертора

Для задания функции из таблицы 4.1 в экран-строку (11 на рис. 4.20), предварительно активизируйте в верхней строке нужные переменные (на рис. 4.20 это А В С) и набирайте из этих символов формулу в нижней строке. При задании формул логическое сложение задается символами «|» или «+», отрицание – символом «'». При умножении двух аргументов они пишутся друг за другом без каких-либо символов. Для того чтобы выполнить инверсию суммы двух аргументов, их необходимо предварительно взять в скобки.

Например, выражение $A \vee B \cdot C$ в окне 11 задания формул должно выглядеть следующим образом: « $(A|BC)'$ » или « $(A+BC)'$ ».

2. Для получения таблицы истинности нажимаем на клавишу 8 → . Заносим полученные данные в столбец Y .

Таблица 4.2 – Таблица истинности заданной функции

A	B	C	D	Y
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	1	

3. Для минимизации логической функции с помощью Workbench нажмите кнопку (7 на рис. 4.20) и снова заполните таблицу истинности 4.2.

4. Далее выполняем команды преобразования таблицы истинности в логическую схему как показано на рисунке 4.21, выбираем команду

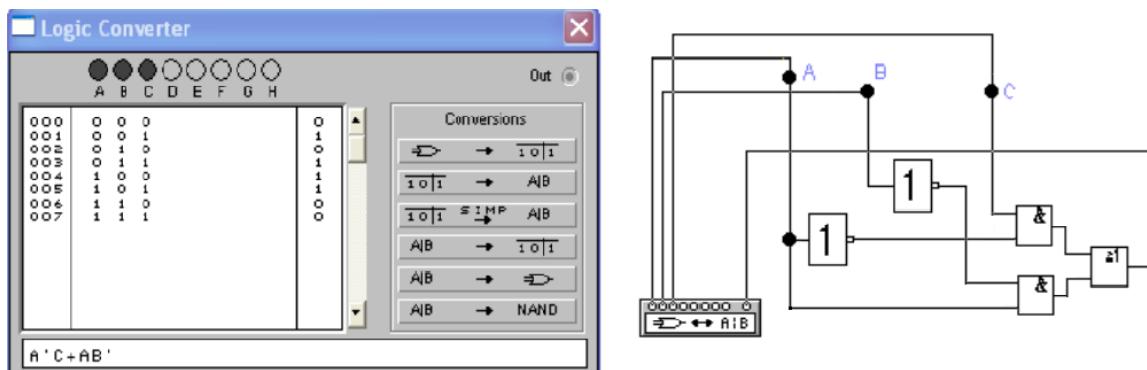


Рисунок 4.21 –Пример преобразования таблицы истинности в логическую схему

5. Выполните аналитическую минимизацию, используя аксиомы и тождества алгебры логики, сопоставьте полученные результаты с компьютерной минимизацией. Сделайте выводы.

4.4 Результаты работы

Подготовьте отчет по лабораторной работе, который должен включать формулу функции, таблицы истинности, полученную схему и аналитическую минимизацию.

4.5 Контрольные вопросы

1. Что такое минимально-полный набор логических элементов и из каких элементов он состоит?
2. На основании правила де Моргана для двух переменных запишите правило де Моргана для трех логических переменных, докажите его корректность.
3. Как выглядит принципиальная схема устройства, если при минимизации был получен результат $F = 1$?

5 Лабораторная работа № 5

Исследование дешифратора и шифратора

5.1 Цель работы

Целью работы является изучение дешифраторов и шифраторов, возможностей наращивания разрядности серийных микросхем дешифраторов и шифраторов путем каскадного их включения, а также методов исследования средствами системы схемотехнического моделирования Electronics Workbench.

5.2 Пояснения к работе

Дешифратор (декодер) служит для преобразования n -разрядного позиционного двоичного кода в единичный выходной сигнал на одном из 2^n выходов. Потому полный декодер имеет n -входов и 2^n -выходов.

Составим таблицу истинности декодера при $n = 2$ (рис. 5.1).

№	a	b	Y0	Y1	Y2	Y3
0	0	0	1	0	0	0
1	0	1	0	1	0	0
2	1	0	0	0	1	0
3	1	1	0	0	0	1

Рисунок 5.1 – Таблица истинности декодера

Для каждого выхода декодера составляют свою функцию. Эта система ФАЛ называется системой собственных функций. Составим систему по единичным значениям функций (ФАЛ для каждого выхода):

$$y_0 = \bar{a} * \bar{b}$$

$$y_1 = \bar{a} * b$$

$$y_2 = a * \bar{b}$$

$$y_3 = a * b$$

По этой системе несложно построить схему дешифратора, которая представляет собой четыре двухвходовых конъюнктора (рис. 5.2):

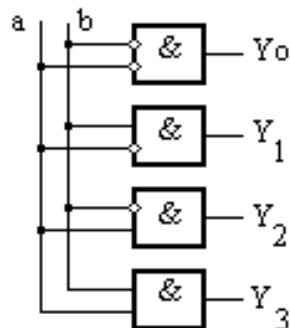


Рисунок 5.2 – Схемная реализация декодера

Условное обозначение декодера показано на рисунке 5.3.

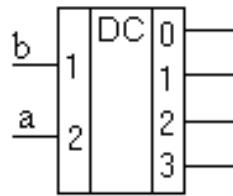


Рисунок 3.3 – Условное обозначение декодера 2x4

Декодеры выпускают в виде отдельных микросхем. Например, ИМС К155ИД3 полный декодер 4*16 (рис.5.4).



Рисунок 5.4 – Декодер К155ИД3

Обратите внимание, что четырёхразрядное двоичное число abcd подаётся на входы по старшинству (согласно своему весу).

Декодер можно синтезировать с управляющим входом, например, с клапаном V. Когда сигнал V=1 декодер работает как обычно, если V=0, то на всех выходах будут нули независимо от сигналов a и b.

Для этого составим таблицу истинности (рис. 5.5).

№	V	a	b	Y0	Y1	Y2	Y3
0	1	0	0	1	0	0	0
1	1	0	1	0	1	0	0
2	1	1	0	0	0	1	0
3	1	1	1	0	0	0	1
4	0	-	-	0	0	0	0

Рисунок 5.5 – Таблица истинности декодера с входом V

Система собственных функций:

$$y_0 = V * \bar{a} * \bar{b}$$

$$y_1 = V * \bar{a} * b$$

$$y_2 = V * a * \bar{b}$$

$$y_3 = V * a * b$$

По этим выражениям нетрудно построить схему (рис. 5.6).

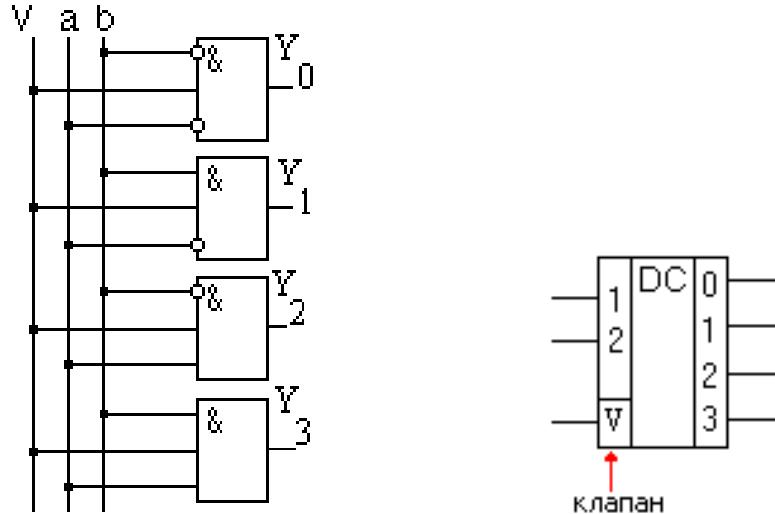


Рисунок 5.6 – Декодер с управляемым входом V (valve - клапан)

Декодер является простейшей схемой, но на его основе создают другие, более сложные комбинационные устройства.

Шифратор (кодер) выполняет функцию, обратную декодеру, то есть преобразует непозиционный (унитарный, единичный) двоичный код (Z_i) в n -разрядный позиционный(ab).

Составим таблицу истинности шифратора при $n = 2$ (рис. 5.7).

№	Z_0	Z_1	Z_2	Z_3	a	b
0	1	0	0	0	0	0
1	0	1	0	0	0	1
2	0	0	1	0	1	0
3	0	0	0	1	1	1

Рисунок 5.7 – Таблица истинности кодера

Синтезируем шифратор. Для этого запишем систему собственных функций:

$$a = \overline{Z_0} * \overline{Z_1} * Z_2 * \overline{Z_3} + \overline{Z_0} * \overline{Z_1} * \overline{Z_2} * Z_3$$

$$b = \overline{Z_0} * Z_1 * \overline{Z_2} * \overline{Z_3} + \overline{Z_0} * \overline{Z_1} * \overline{Z_2} * Z_3 ,$$

по которой и составим схему (рис. 5.8).

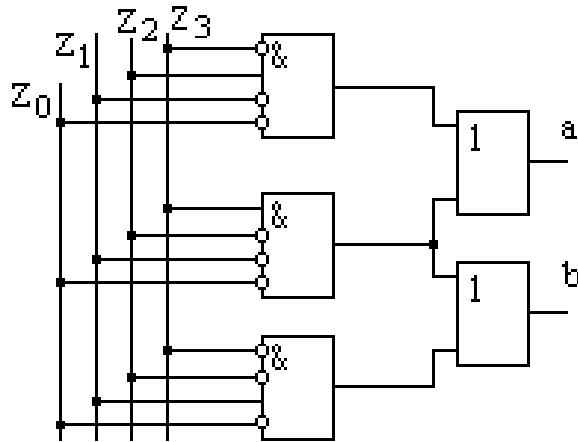


Рисунок 5.8 – Схема шифратора при $n = 2$ (или 4×2)

Поскольку на вход шифратора поступает единичный код, то при синтезе шифратора выходные сигналы для схемы рис. 5 можно записать в виде

$$a = z2 + z3$$

$$b = z1 + z3$$

Условное обозначение шифратора и пример микросхемы приведены на рис. 5.9.

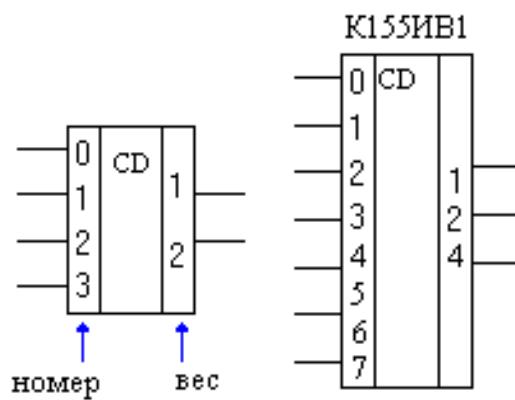


Рисунок 5.9 – Условное обозначение шифраторов 4×2 и 8×3

Шифраторы имеются во многих сериях микросхем (К555ИВ3, 533ИВ2).

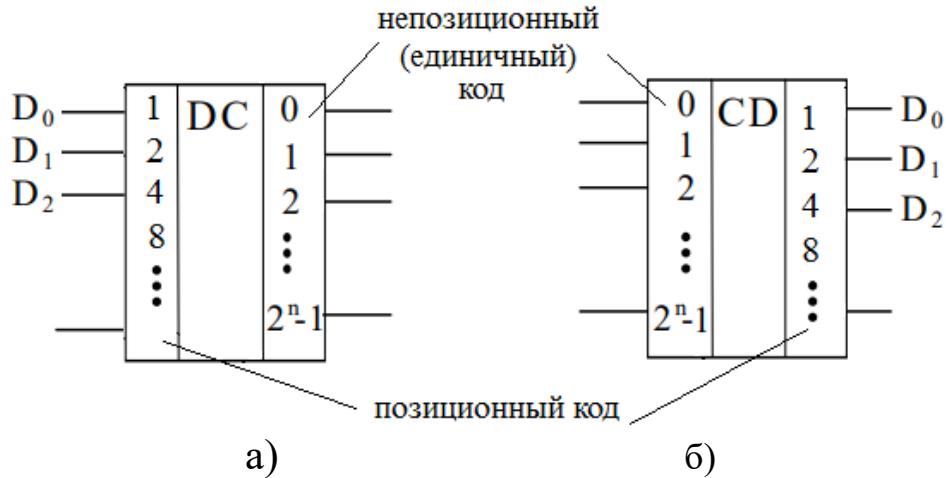


Рисунок 5.10 – Сопоставление дешифратора (а) и шифратора (б)

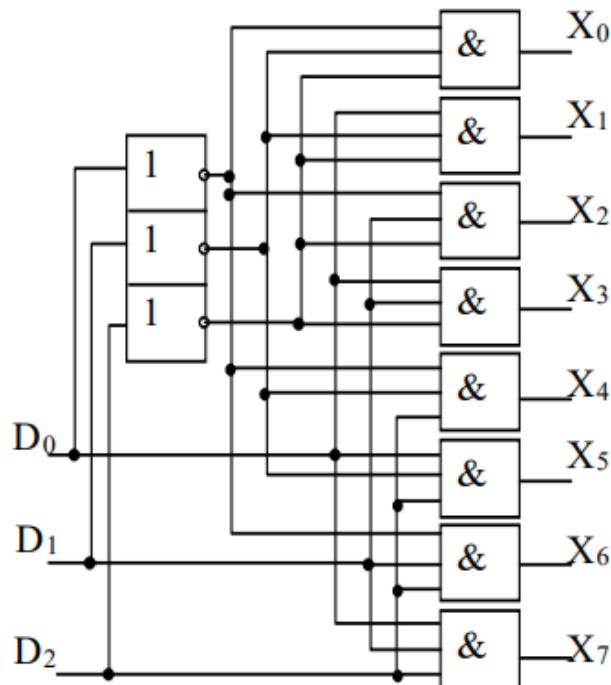


Рисунок 5.11 – Трехразрядный дешифратор (3x8)

Таким образом, трехразрядный дешифратор строится на восьми трехходовых схемах И (3И), младший разряд - нулевой.

5.3 Описание модели дешифратора и шифратора (файл decoder1.ewb)

Схема модели дешифратора и шифратора (в формате EWB) представлена на рисунке 5.12.

Схема содержит следующие элементы:

- цифровой генератор слов (Word Generator), предназначенный для генерации 16-ти разрядных двоичных слов, которые набираются на экране, расположенным в левой части лицевой панели;



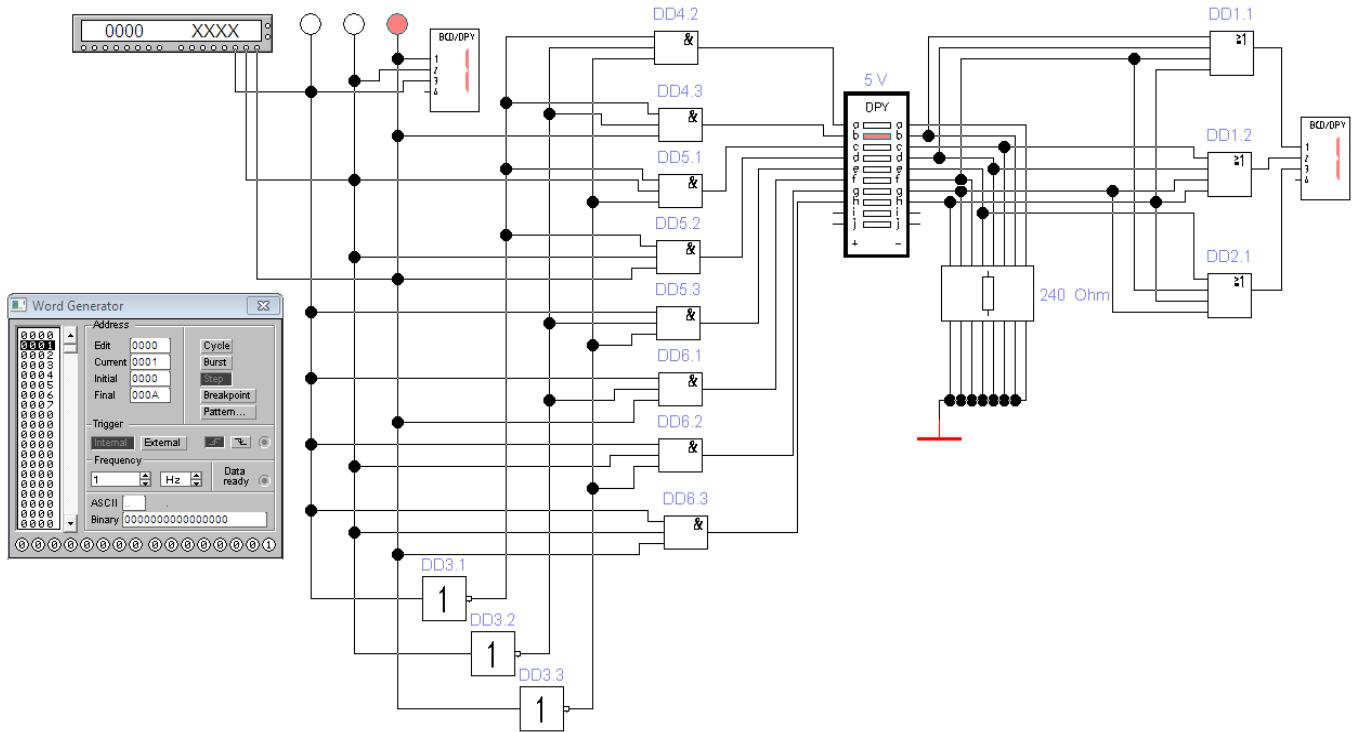
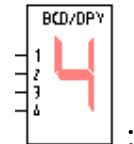


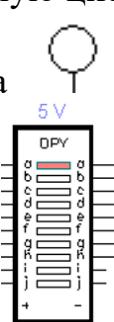
Рисунок 5.12 – Модели дешифратора 3x8 на элементах ЗИ и шифратора на элементах 4ИЛИ

– семисегментный индикатор с преобразователем четырехразрядного

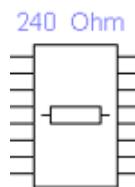


двоичного числа в шестнадцатеричную цифру ;

– три логических индикатора ;



– светодиодный индикатор с резисторной нагрузочной



– матрицей – набор из восьми одинаковых по величине

резисторов; – имеет десять градаций уровня и по десять выводов с каждой стороны, выводы слева соответствуют анодам светодиодов, справа – катодам. Обычно такие индикаторы используются для визуального отображения уровня сигнала.

Шифратор 8x3 построен на основании следующих уравнений и ТИ (рис. 5.13), поскольку на входе единичный сигнал, то оставшиеся клетки ТИ можно заполнить нулями, а выходные сигналы записать в виде:

$$\begin{cases} x_1 = y_4 + y_5 + y_6 + y_7 \\ x_2 = y_2 + y_3 + y_6 + y_7 \\ x_3 = y_1 + y_3 + y_5 + y_7 \end{cases}$$

y_0	y_1	y_2	y_3	y_4	y_5	y_6	y_7	x_1	x_2	x_3
1								0	0	0
	1							0	0	1
		1						0	1	0
			1					0	1	1
				1				1	0	0
					1			1	0	1
						1		1	1	0
							1	1	1	1

Рисунок 5.13 - ТИ шифратора 8x3

5.4 Порядок выполнения работы

1 Откройте файл decoder1.ewb и в нем генератор слов как показано на рисунке 5.14. Нажмите кнопку Pattern и выполните очистку буфера посредством метки напротив команды Clear buffer и подтвердите, нажатием кнопки Accept.

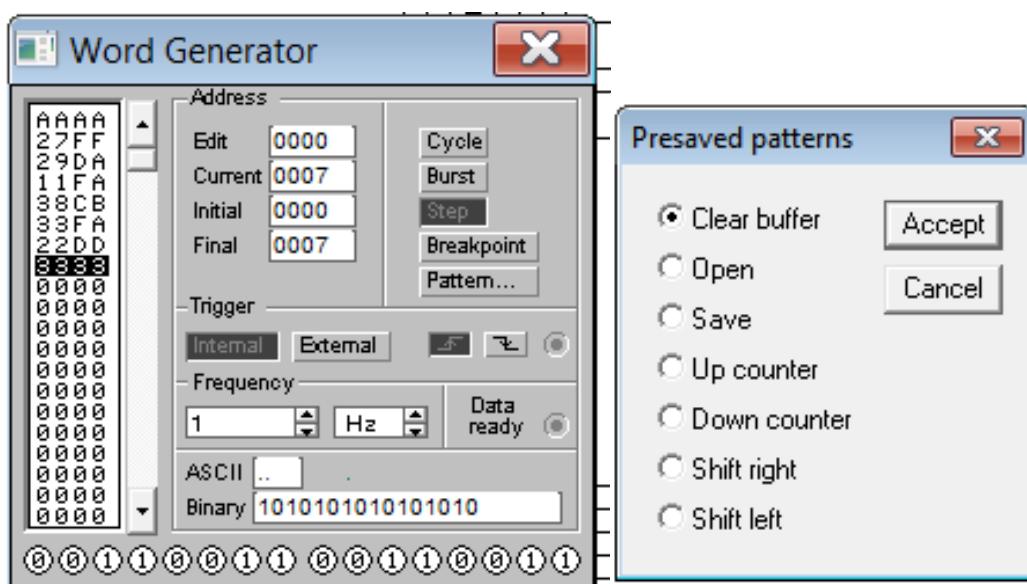
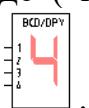


Рисунок 5.14 – Очистка буфера и ввод слов

Выполните установки в генераторе: запуск по переднему фронту сигнала ; при внутреннем запуске (нажатии кнопки Internal) синхронизации. Установите частоту в окне Frequency 1 Hz. Нажмите step. Пошагово введите в последовательные ячейки, расположенные в левой части лицевой панели генератора слов цифры 0,1,2,3,4,5,6,7. Сформированные слова выдаются на шестнадцать расположенных в нижней части прибора выходных клемм-индикаторов: - с индикацией в двоичном коде в строке окна binary; - в пошаговом (step), циклическим (cycle) или с выбранного слова до конца (при нажатии кнопки BURST) при заданной частоте посылок (1Гц). Установите режим Cycle. В окно Final загрузите число 000A (10 повторений с 0000).

2. Переведите клавишу в правом верхнем углу экрана в положение «1». В пошаговом режиме (step) генератора, заполните таблицы истинности декодера и кодера в двоичном виде (по показаниям окна Binary генератора



слов) и в десятичном по дисплею .

3. Откройте файл decoder2.ewb (рис 5.15). В генераторе слов задайте последовательность входных цифр 0,...7, соответствующую входным двоичным кодам, подаваемым на дешифратор. Работу дешифратора исследуйте в пошаговом режиме (step). Составьте таблицу истинности учитывая, что младшим разрядом является Y0.

Условное обозначение и логическая схема ИМС K555ИД7 приведены на рис. 5.16 а и 5.16 б соответственно. Сигналы управления E1 и E2 – инверсные. Сигнал лог.0 при разрешающем сочетании на входах E1 E2 E3 появится на том выходе дешифратора номер которого соответствует десятичному эквиваленту кода, поданному на адресные входы 1, 2, 4. Наличие трех управляющих входов позволяет наращивать разрядность дешифратора путем соединения нескольких ИМС.

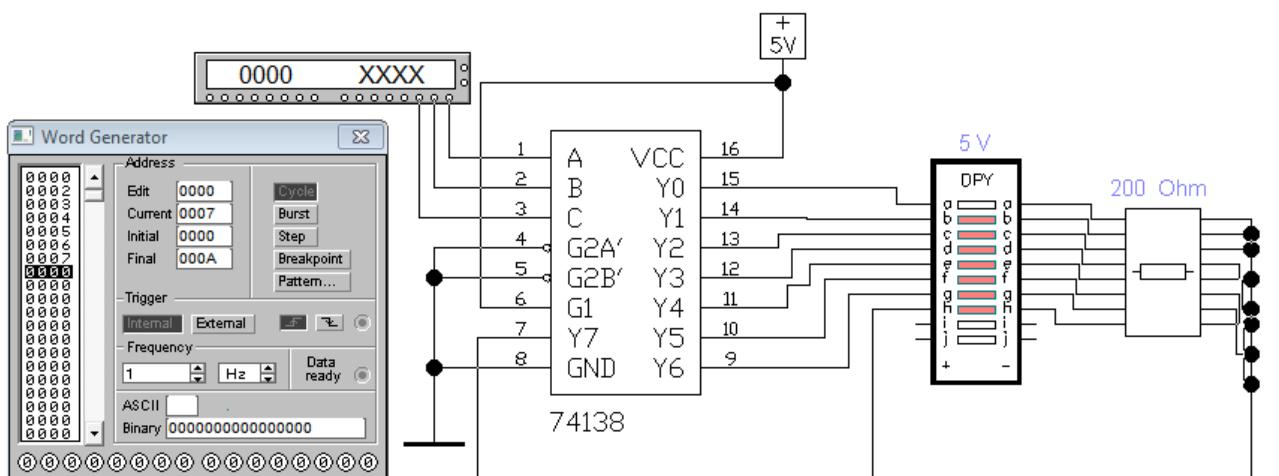


Рисунок 5.15 – Схема исследования декодера К555ИД7 (74138).

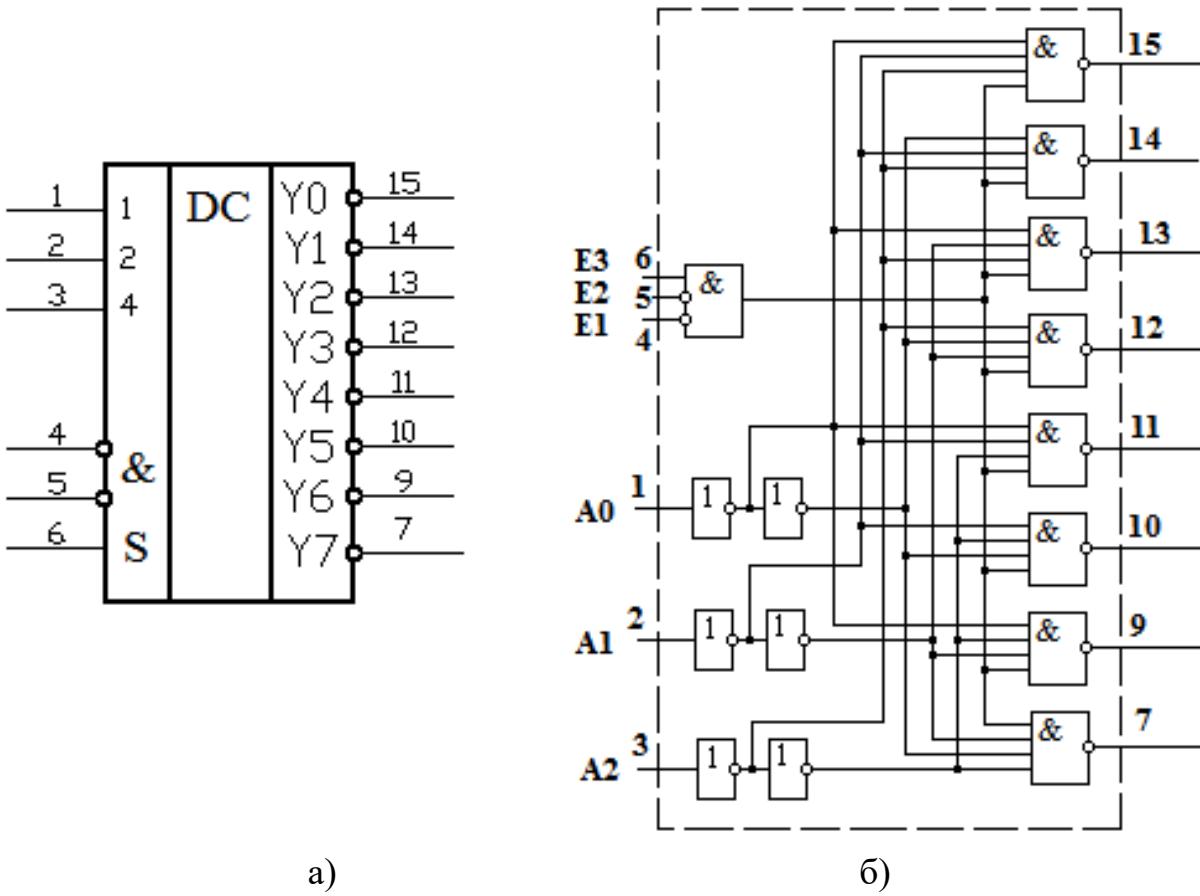


Рисунок 5.16 – Условное обозначение и схема декодера К555ИД7

5.5 Результаты работы

Подготовьте отчет по лабораторной работе, который должен включать исходные данные, ТИ исследуемых схем, схемы и результаты проверки.

5.6 Контрольные вопросы

1. Что такое DC и CD, каково их назначение?
2. Чему равно число входных и выходных линий полного DC?
3. В чем принципиальное отличие полного и неполного декодера?
4. Составьте ТИ декодера 4x16 .
5. Построить схему заданного полного или неполного дешифратора для двух- и трехразрядных чисел.
6. Построить схему шифратора преобразующий унитарный код в двоичный.
7. Построить схему заданного полного или неполного дешифратора на стандартных микросхемах дешифраторов.

6 Лабораторная работа № 6

Исследование работы мультиплексора и демультиплексора

6.1 Цель работы

Целью работы является изучение функционального назначения и устройства и мультиплексора и демультиплексора средствами системы схемотехнического моделирования Electronics Workbench.

6.2 Пояснения к работе

Мультиплексор (MUX) – это функциональный узел, осуществляющий подключение (коммутацию) одного из нескольких информационных входов к одному выходу y . На выход такого устройства передаётся сигнал того информационного входа, номер которого задан на адресных входах x_1 и x_2 двоичным кодом. Условное изображение мультиплексора на четыре информационных входа и его схема показаны на рисунке 6.1, а и б.

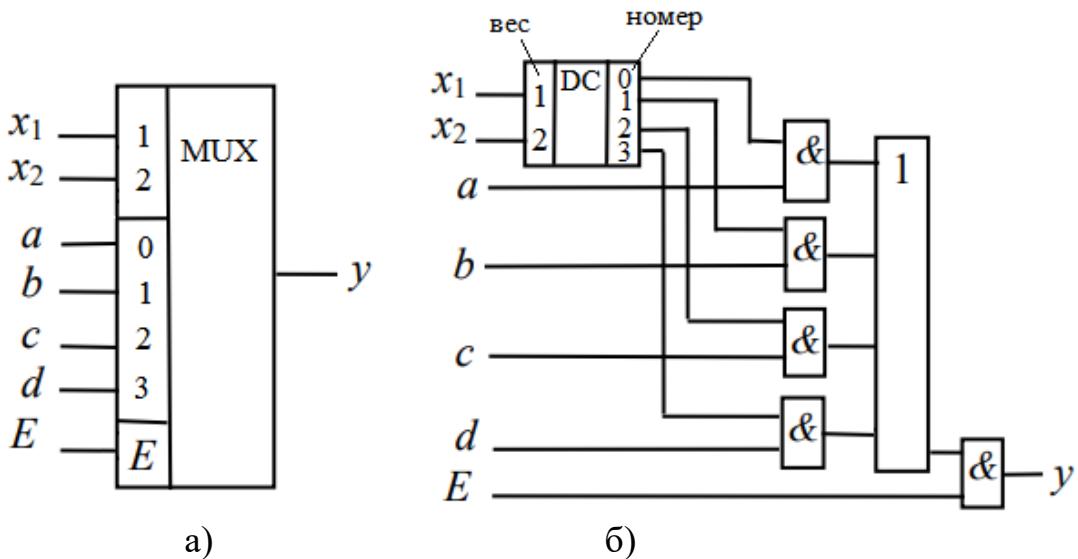


Рисунок 6.1 –Условное обозначение мультиплексора (а) и его структура (б)

При $x_1 = 0$ и $x_2 = 0$, $y = a$; при $x_1 = 1$ и $x_2 = 0$, $y = b$; при $x_1 = 0$ и $x_2 = 1$, $y = c$ и при $x_1 = 1$ и $x_2 = 1$, $y = d$. Здесь x_2 старший бит двухразрядного адреса (адрес x_2 x_1 , а не x_1 x_2).

Выходной сигнал мультиплексора описывается выражением:

$$y = \overline{ax_1x_2} + \overline{bx_1x_2} + \overline{cx_1x_2} + \overline{dx_1x_2} \quad (6.1)$$

Вход E – разрешающий: при $E = 1$ мультиплексор работает как обычно, при $E = 0$ выход узла находится в неактивном состоянии и на выходе мультиплексора ноль. Серийные узлы выпускаются с числом адресных входов $n = 2, 3$ и 4 при возможном числе 2^n коммутируемых входов. При

необходимости коммутировать большее количество входов используют несколько мультиплексоров. Мультиплексоры находят широкое применение в устройствах отображения и передачи информации а также в системах управления. Так как мультиплексор реализует СДНФ (выражение 6.1), то он является универсальным прибором и на нем можно реализовывать различные функции, подавая на информационные входы нули или единицы в соответствии с таблицей истинности функции, а на адресные входы – аргументы функции.

Демультиплексор (DMUX) выполняет функцию, обратную функции мультиплексору, т. е. производит коммутацию одного информационного входа на один из 2^n выходов, где n – число адресных входов. Демультиплексор имеет один информационный вход D и несколько выходов, причем вход подключается к выходу с номером, указанным в адресе. В качестве примера на рисунке 6.2, а дано условное графическое обозначение демультиплексора, имеющего четыре выхода, а ТИ приведена в табл. 6.1. Пользуясь таблицей 6.1, запишем переключательные выходные функции:

$$y_0 = D \bar{x}_1 x_2; \quad y_1 = D \bar{x}_1 \bar{x}_2; \quad y_2 = D x_1 \bar{x}_2; \quad y_3 = D x_1 x_2. \quad (6.2)$$

Функциональная схема демультиплексора, реализующая эти выражения, приведена на рисунке 6.2, б.

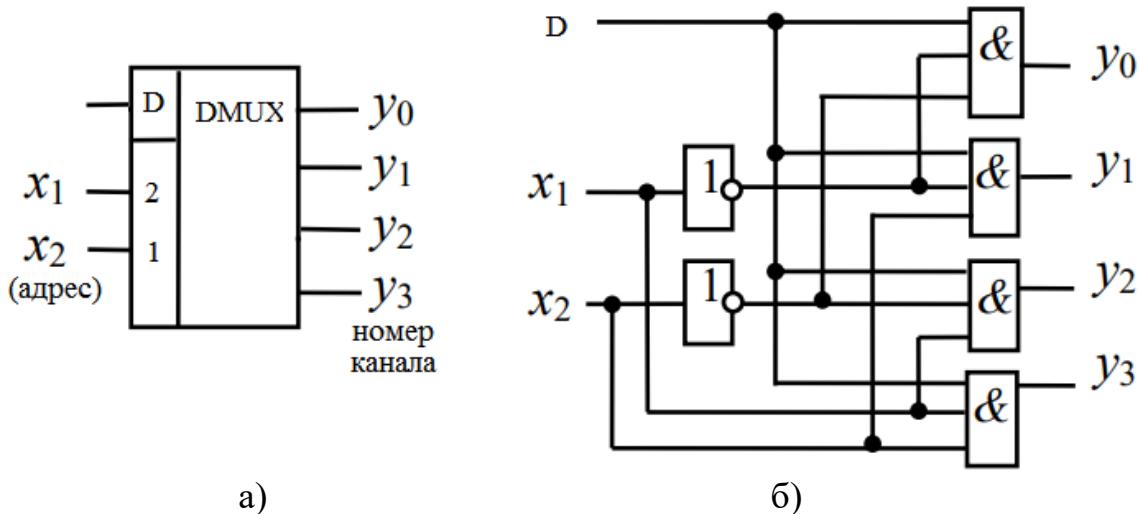


Рисунок 6.2 –Условное обозначение (а) и функциональная схема (б)
демультиплексора

Таблица 6.1– ТИ демультиплексора

D	x_1	x_2	y_3	y_2	y_1	y_0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Нетрудно видеть, что вход D может играть роль информационного или управляющего. В последнем случае DMUX превращается в декодер с

разрешающим входом, поэтому промышленностью выпускаются ИМС дешифратор/демультиплексор, так как это одно и тоже.

Если требуемое число выходов DMUX превышает имеющееся в выпускаемых интегральных микросхемах, то используют каскадное включение DMUX. На рисунке 6.3, а показано каскадное включение демультиплексоров (а) и мультиплексоров (б). Для 16 каналов используется четырехразрядный адрес $x_1 x_2 x_3 x_4$ – младший разряд x_4 .

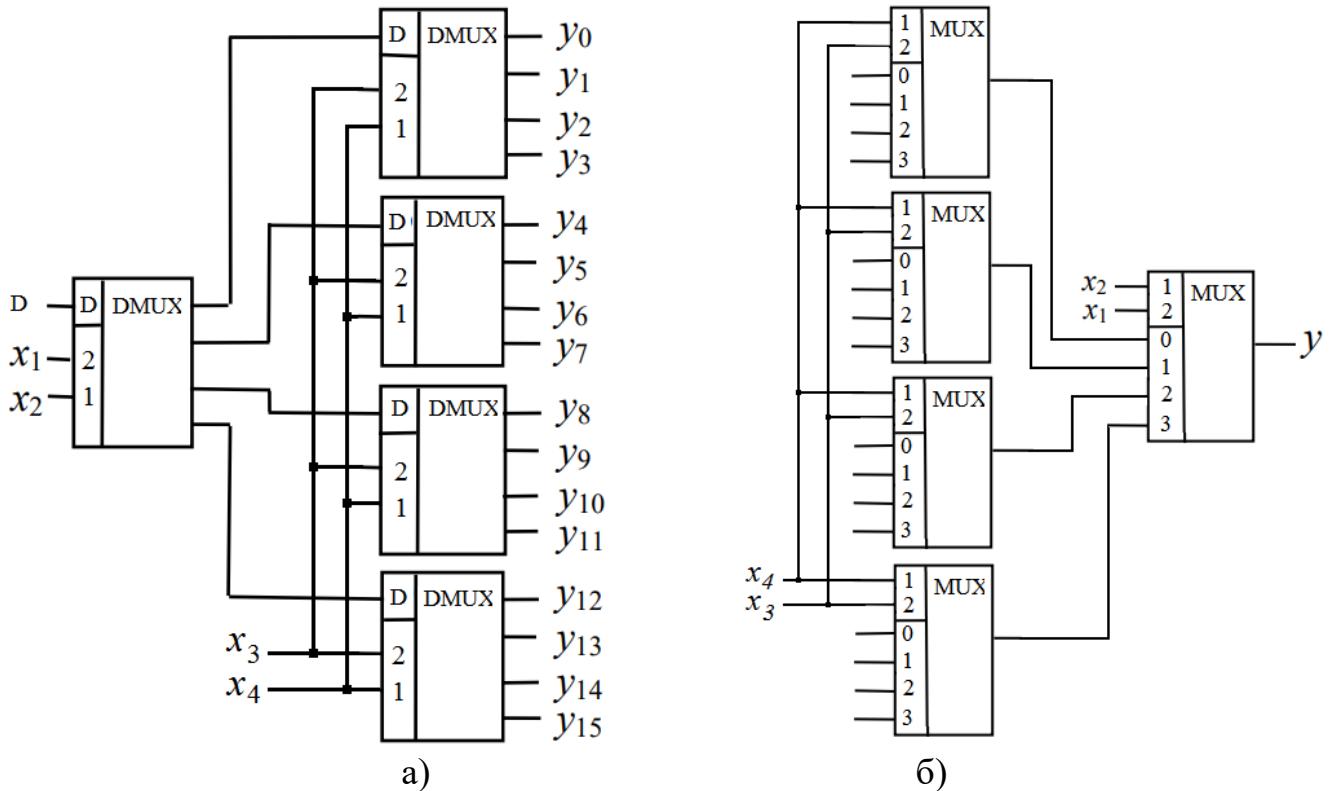
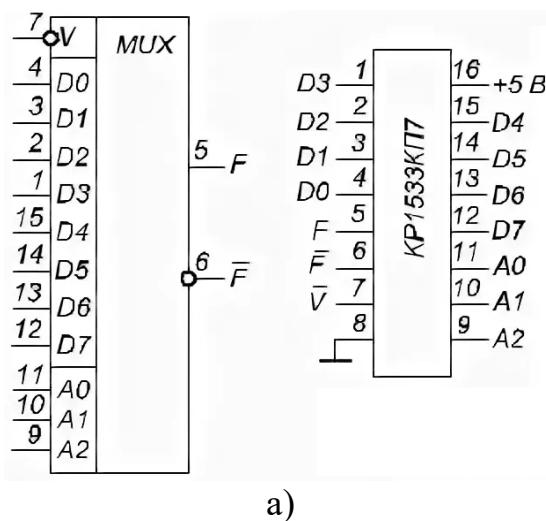


Рисунок 6.3 – Каскадное включение DMUX и MUX

Промышленностью выпускаются ИМС мультиплексоров с числом адресных входов 2, 3, 4. Например. K555КП7, 1533КП7 аналоги ИМС серии 74151. Это восьмиканальные мультиплексоры, их условное обозначение приведено на рис 6.4.



Мультиплексор 74151 (Digital -> MUX)

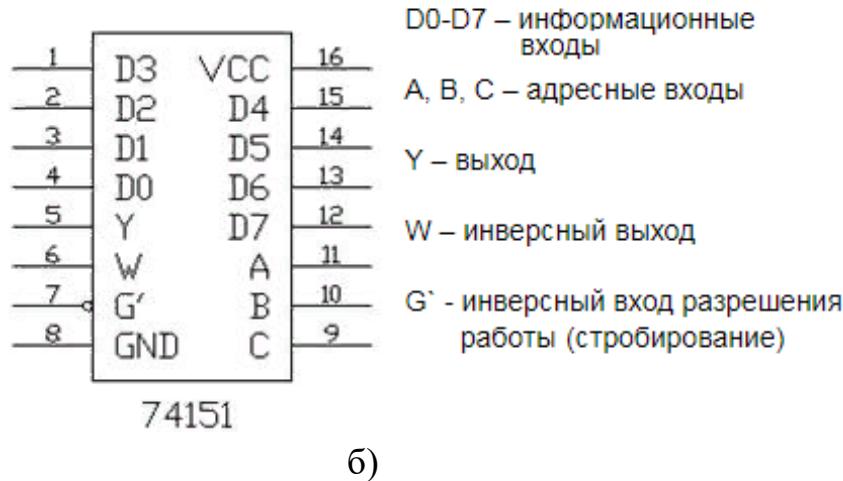


Рисунок 6.4 – Восьмиканальные мультиплексоры

6.3 Описание модели мультиплексора (файл multiplex.ewb) и демультиплексора (файл demultiplex.ewb)

Схема исследования мультиплексора (модель в формате EWB) представлена на рисунке 6.5. Схема содержит следующие элементы:

- источник положительного потенциала 5В;

– цифровой генератор слов (Word Generator), предназначенный для генерации 16-ти разрядных двоичных слов, которые набираются на экране, расположенным в левой части лицевой панели;

- логические пробники .

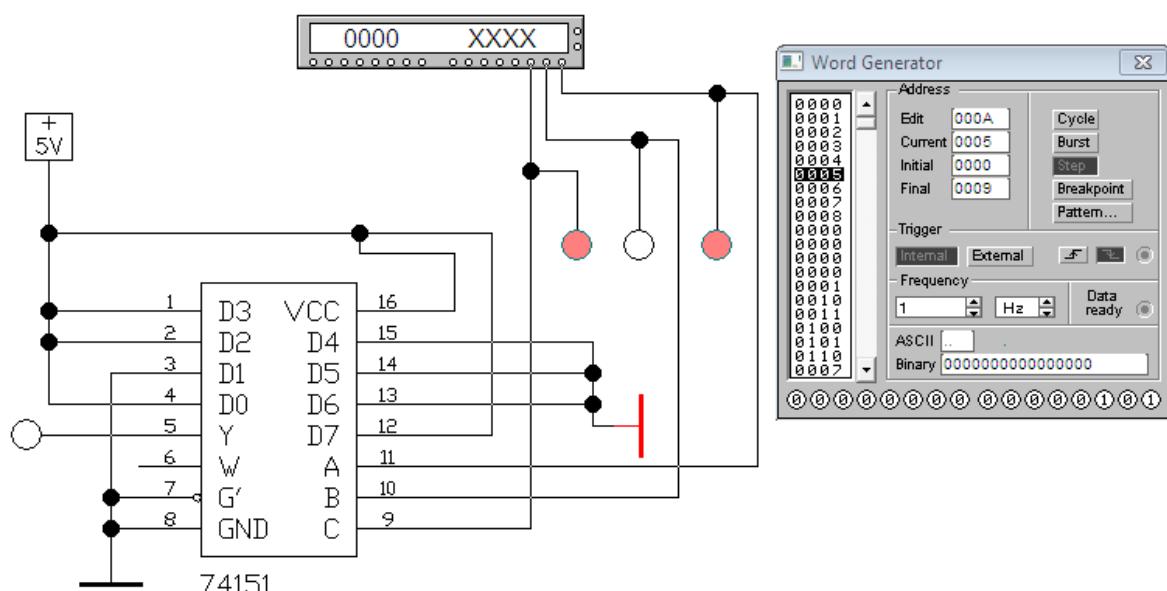


Рисунок 6.5 – Пример схемы включения мультиплексора

Схема модели демультиплексора (модель в формате EWB) представлена на рисунке 6.6. Схема содержит следующие элементы:

- источник положительного потенциала 5В;

- цифровой генератор слов (Word Generator)

предназначенный для генерации 16-ти разрядных двоичных слов, которые набираются на экране, расположенным в левой части лицевой панели;

- восемь логических пробников

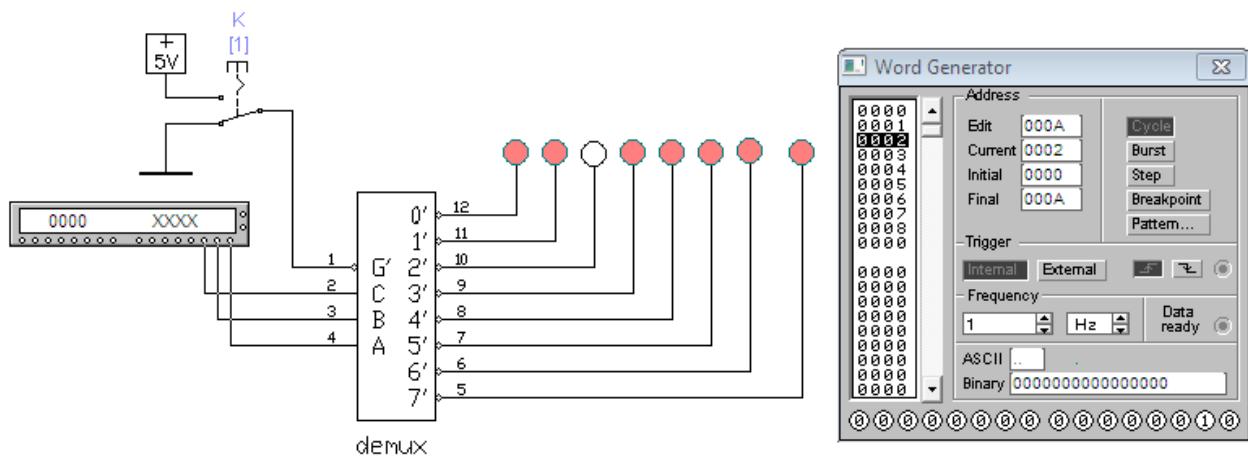


Рисунок 6.6 – Модель демультиплексора

6.4 Порядок выполнения работы

1. Выпишите в соответствии со своим вариантом задания (номером бригады) исходные данные из таблицы 6.2.

Таблица 6.2 – Варианты задания

Номер бригады	D ₀	D ₁	D ₂	D ₃	D ₄	D ₅	D ₆	D ₇
1	1	1	0	1	0	0	1	0
2	0	1	1	1	0	0	1	0
3	1	0	1	0	1	0	1	0
4	1	0	1	1	0	0	0	1
5	0	0	0	1	1	0	1	1
6	1	0	0	0	1	0	0	1
7	0	0	1	0	1	1	0	1
8	1	0	0	1	0	1	1	1
9	0	1	0	1	0	1	0	1
10	0	1	1	0	1	0	1	0

2. Откройте файл multiplex.ewb и выполните соединение информационных входов D_i с источником питания (лог. 1) и с заземлением (лог. 0) в соответствии со своим вариантом (см. табл. 6.2).

3. Двойным щелчком мыши откройте цифровой генератор слов как показано на рисунке 6.7. Нажмите кнопку Pattern и выполните очистку буфера посредством метки напротив команды Clear buffer и подтвердите, нажатием на кнопку Accept.

4. Выполните установки в генераторе: запуск по переднему фронту сигнала ; при внутреннем запуске (нажатии кнопки Internal) синхронизации. Установите частоту в окне Frequency 1 Hz. Намите step. Пошагово введите числа (адреса) от 0 до 7 из трех левых столбцов таблицы 6.3 на экране, расположеннном в левой части лицевой панели генератора слов.

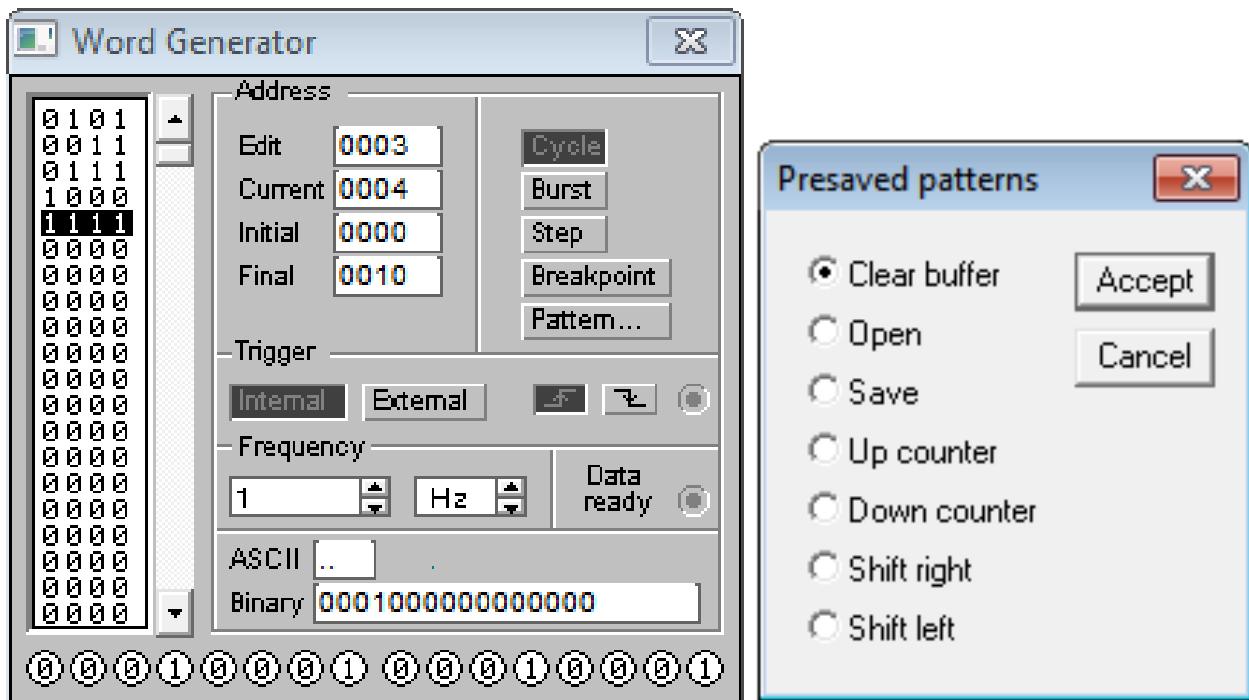


Рисунок 6.7 – Очистка буфера и ввод слов

Сформированные слова выдаются на шестнадцать расположенных в нижней части прибора выходных клемм-индикаторов: - с индикацией в двоичном коде ; - в пошаговом (step), циклическим (cycle) или с выбранного слова до конца (при нажатии кнопки BURST) при заданной частоте посылок (1Гц). Установите режим Cycle. В окно Final загрузите число 000A (десять чисел начиная с 0000).

Таблица 6.3 – Результаты проверки мультиплексора

Адрес			Выход	
C	B	A	дано	факт
0	0	0	$D_0 =$	
0	0	1	$D_1 =$	
0	1	0	$D_2 =$	
0	1	1	$D_3 =$	
1	0	0	$D_4 =$	
1	0	1	$D_5 =$	
1	1	0	$D_6 =$	
1	1	1	$D_7 =$	

Работу мультиплексора исследуйте в пошаговом режиме (step), фиксируя сигналы по индикаторам. Результаты занесите в таблицу 6.3.

5. Откройте файл demultiplex.ewb. Двойным щелчком мыши откройте цифровой генератор слов как показано на рисунке 6.7. Нажмите кнопку Pattern и выполните очистку буфера посредством метки напротив команды Clear buffer и подтвердите, нажатием на кнопку Accept.

Выполните установки в генераторе: запуск по переднему фронту сигнала ; при внутреннем запуске (нажатии кнопки Internal) синхронизации. Установите частоту в окне Frequency 1 Hz. Намите step. Пошагово введите адреса из трех левых столбцов таблицы 6.3 на экране, расположенным в левой части лицевой панели генератора слов. Сформированные слова выдаются на шестнадцать расположенных в нижней части прибора выходных клеммах - с индикацией в двоичном коде в строке окна binary; - в пошаговом (step), циклическим (cycle) или с выбранного слова до конца (при нажатии кнопки BURST) при заданной частоте посылок (1Гц). Установите режим Cycle. В окно Final загрузите число 000A (десять чисел-адресов начиная с 0000).

6. Проверку демультиплексора проводите в пошаговом режиме (step) и по сигналам пробников-индикаторов заполните таблицу 6.4. Работу можно остановить выключателем K, который управляется клавишей 1.

Таблица 6.4 – Результаты проверки демультиплексора

Адрес			Выходы								
C	B	A	0	1	2	3	4	5	6	7	
0	0	0									
0	0	1									
0	1	0									
0	1	1									
1	0	0									
1	0	1									
1	1	0									
1	1	1									

6.5 Результаты работы

Подготовьте отчет по лабораторной работе, который должен содержать исходные данные, схемы измерений, таблицы измерений.

6.6 Контрольные вопросы

- 1 Сколько выходов может иметь демультиплексор, если число адресных входов равно a ?
- 2 Что такое мультиплексор?
- 3 Что такое демультиплексор?
- 4 Приведите УГО мультиплексора и поясните все его выводы.
- 5 Приведите УГО демультиплексора и обозначение его выводов.

\

7 Лабораторная работа № 7

Синтез и исследование преобразователей кодов

7.1 Цель работы

Изучение этапов синтеза комбинационных схем с несколькими выходами для преобразования кодов и приобретение навыков в экспериментальном исследовании таких схем.

7.2 Пояснения к работе

Операция изменения кода числа называется его перекодированием. Интегральные микросхемы, выполняющие эти операции, называются преобразователями кодов. Преобразователи кодов бывают простые и сложные. К простым относятся преобразователи, которые выполняют стандартные операции изменения кода чисел, например, преобразований двоичного кода в одинарный или обратную операцию (кодер - декодер). Сложные преобразователи кодов выполняют нестандартные преобразования кодов и их схемы приходится разрабатывать каждый раз с помощью алгебры логики. Будем считать, что преобразователи кодов имеют n входов и k выходов. Соотношения между n и k могут быть любыми: $n=k$, $n < k$ и $n > k$. При преобразовании кодов чисел с ними могут выполняться различные дополнительные операции, например, умножение на весовые коэффициенты. Примером такого преобразования является преобразование двоично-десятичного кода в двоичный. Весовые преобразователи кодов используются при преобразовании числовой информации. Интегральные микросхемы преобразователей кодов выпускаются только для наиболее распространенных операций:

- преобразователи двоично-десятичного кода в двоичный код;
- преобразователи двоичного кода в двоично-десятичный код;
- преобразователи двоичного кода в код управления сегментными индикаторами;
- преобразователи двоичного или двоично-десятичного кода в код управления шкальными или матричными индикаторами.

В качестве примера рассмотрим преобразователь двоичного кода в код управления семисегментным цифровым индикатором, приведенный на рисунке 7.1 а. Сам индикатор представляет собой полупроводниковый прибор, в котором имеются семь сегментов, выполненных из светодиодов. Включением и выключением отдельных сегментов можно получить светящееся изображение отдельных цифр или знаков. Конфигурация и расположение сегментов индикатора показаны на рисунке 7.1 а. Каждой цифре соответствует свой набор включения определенных сегментов индикатора. Соответствующая таблица приведена на рисунке 7.1 б. В этой таблице также приведены двоичные коды соответствующих цифр.

Такие индикаторы позволяют получить светящееся изображение не только цифр от 0 до 9, но других знаков, используемых в 8- и 16-ричной системах счисления. Для управления индикаторами выпускаются интегральные микросхемы типов К514ИД2, 176ИД3, 564ИД4, 564ИД5 и др. Шкальные индикаторы представляют собой линейку светодиодов с одним общим анодом

или катодом. Преобразователи двоичного кода в код управления шкальным индикатором обеспечивают перемещение светящегося пятна, определяемое двоичным кодом на адресных входах.

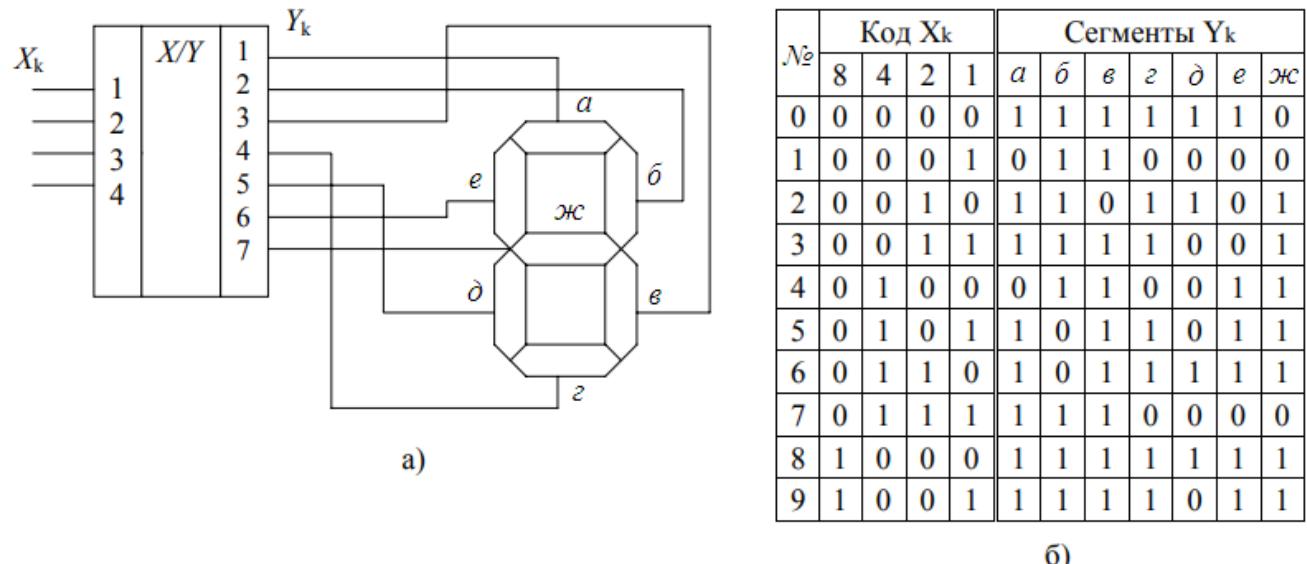


Рисунок 7.1 – Схема преобразователя кода для семисегментного индикатора (а) и таблица истинности (б)

7.3 Описание схемы подключения для исследования преобразователя кода

Схема для исследования преобразователя кода представлена на рисунке 7.2 и содержит следующие элементы:

- идеальный источник положительного потенциала + 5В;
- преобразователь кода X/Y, синтезированный по Вашему варианту;
- индикаторы-пробники.

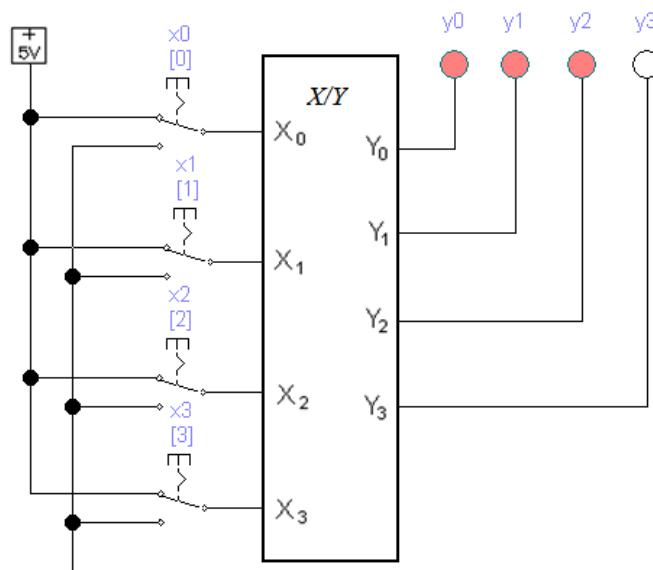


Рисунок 7.2 – Схема подключения для исследования преобразователя кода

7.4 Порядок выполнения работы

1. Задание на работу: Синтезировать преобразователь BCD кода в четырехразрядный заданный двоичный код.

Для этого выпишите в соответствии со своим вариантом задания (номером бригады) исходные данные из таблицы 7.1.

Таблица 7.1 – Варианты заданий

Десятичная цифра, (BCD или 8421)- х ₀ х ₁ х ₂ х ₃	Разновидности D-кодов (варианты заданий) –Y ₀ Y ₁ Y ₂ Y ₃									
	1	2	3	4	5	6	7	8	9	10
	Код Грея	2421	5121	Код с изб. 3	4221	5211	5421	53-21	75-31	Не взвеш. код
0(0000)	0000	0000	0000	0011	0000	0000	0000	0000	0000	0000
1(0001)	0001	0001	0001	0100	0001	0001	0001	0001	0001	0001
2(0010)	0011	0010	0010	0101	0010	0011	0010	0111	0110	0010
3(0011)	0010	0011	0011	0110	0011	0101	0011	1010	0111	0011
4(0100)	0110	0100	0111	0111	0110	0111	0100	0101	1010	0110
5(0101)	0111	10011	1000	1000	0111	1000	1000	1000	0100	0111
6(0110)	0101	1100	1001	1001	1010	1001	1001	1001	0101	1001
7(0111)	0100	1101	1010	1010	1011	1011	1010	1111	1000	1101
8(1000)	1100	1110	1011	1011	1110	1101	1011	1100	1001	1110
9(1001)	1000	1111	1111	1100	1111	1111	1100	1101	1110	1111

2. Выполните синтез комбинационной схемы для четырех переключательных функций (для каждого разряда кода) четырех переменных (табл. 7.1). Приведите минимизированные выходные функции Y₀, Y₁, Y₂, Y₃, к произвольному базису и начертите схему.

Для создания своего файла выберете курсором команду File, затем в ниспадающем меню строку New (или щелкнуть по пиктограмме ) и сохраните файл по команде File > Save As, присвоив ему имя латинскими буквами и далее ОК. Для построения схемы необходимо использовать источник питания для подачи на вход устройства логической «1» и землю – для подачи сигнала «0». При нажатии на кнопку Sources выберите эти элементы и переместите левой кнопкой мыши на рабочий стол как показано на рисунке 7.3. Для перебора кодов на входе устройства используются четыре ключа, которые находятся на панели Basic (см. рис. 7.4). Небходимые логические элементы расположены на панели Logic Gates (см. рис. 7.5).

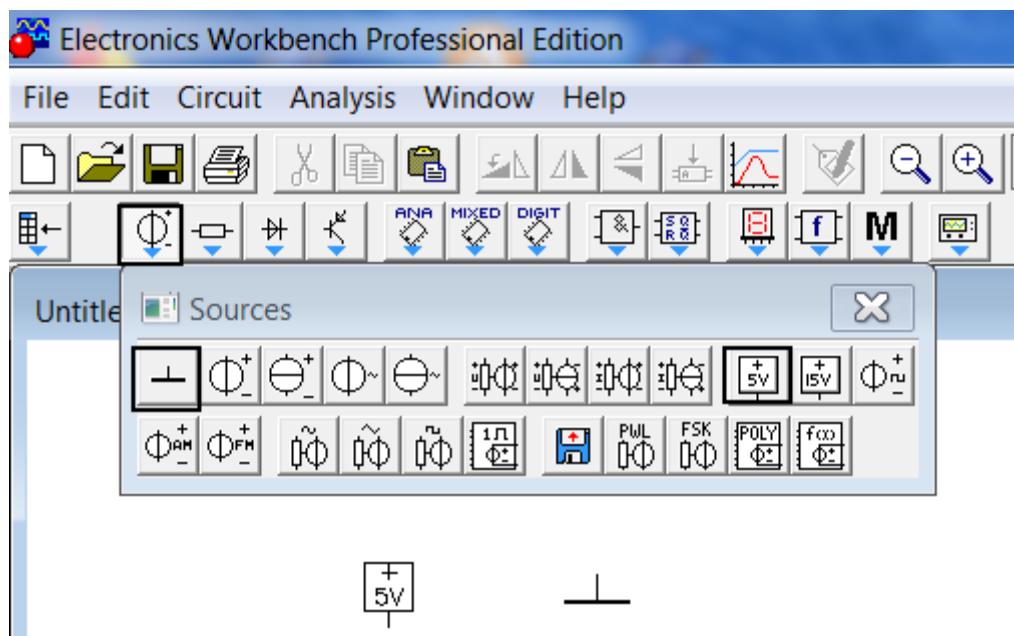


Рисунок 7.3 – Панель Sources

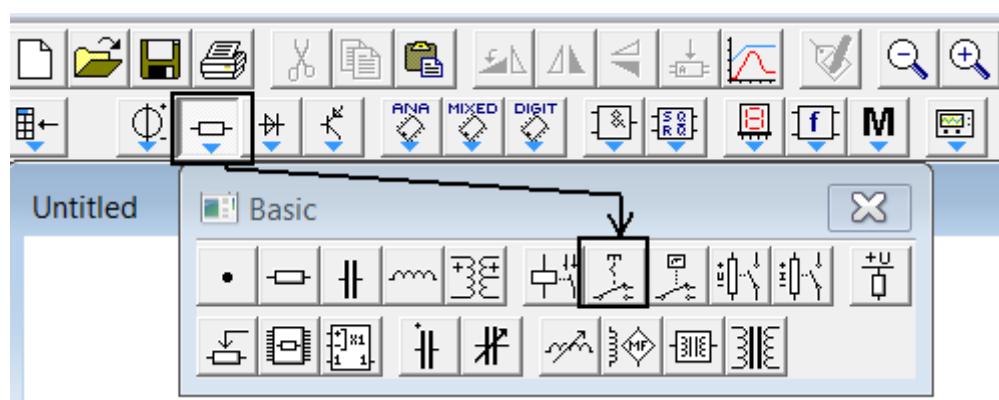


Рисунок 7.4 – Панель Basic

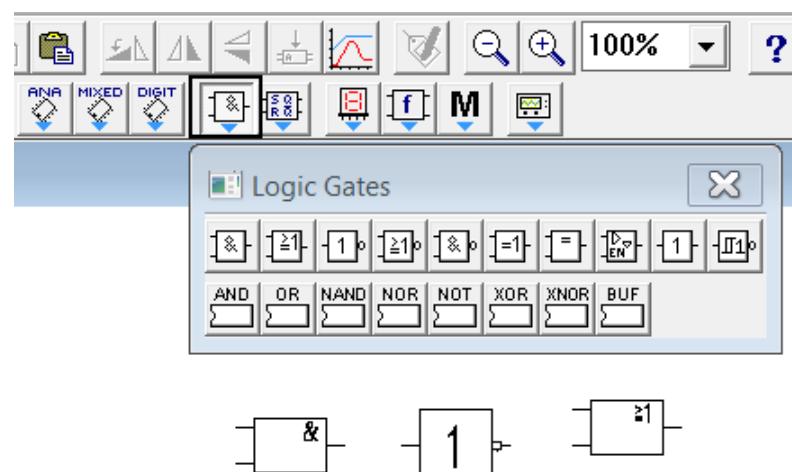


Рисунок 7.5 – Панель Logic Gates

Для изменения числа входов логических элементов следует нажать левой кнопкой мыши на элементе, открыть вкладку Properties и подтвердить свой выбор как показано на рисунке 7.6.

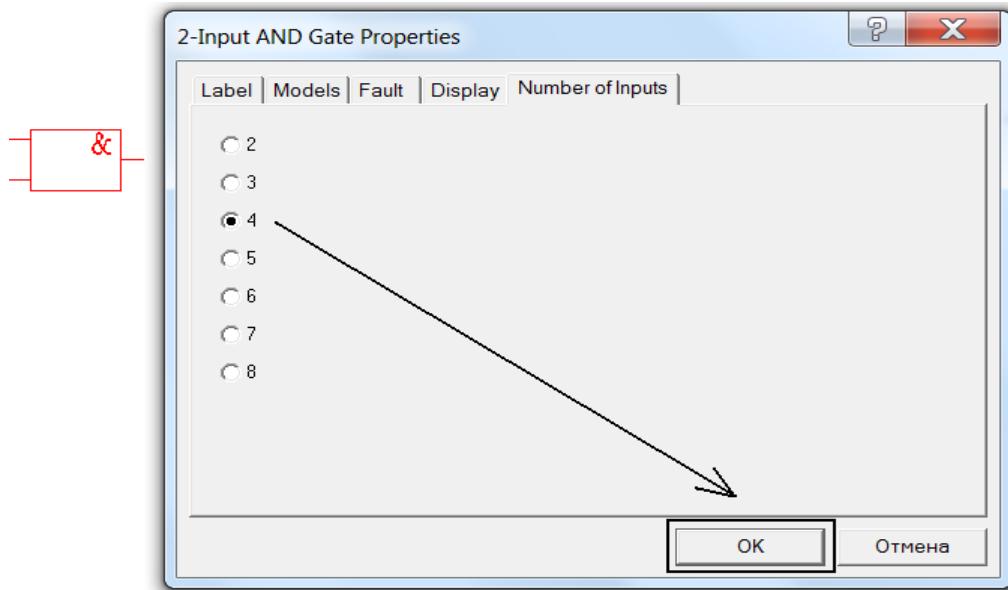


Рисунок 7.6 – Изменение количества входов в логическом элементе

Выполните соединение элементов схемы, для этого подведите мышь к концу вывода элемента до появления черной точки и нажав левую клавишу протяните линию до конца вывода другого элемента. Для контроля выходного кода установите двоичные индикаторы и пронумеруйте их как показано на рис. 7.7

Пронумеруйте ключи, нажав кнопку Label на панели Properties, введите клавиши управления ключами во вкладке Value как показано на рисунке 7.7.

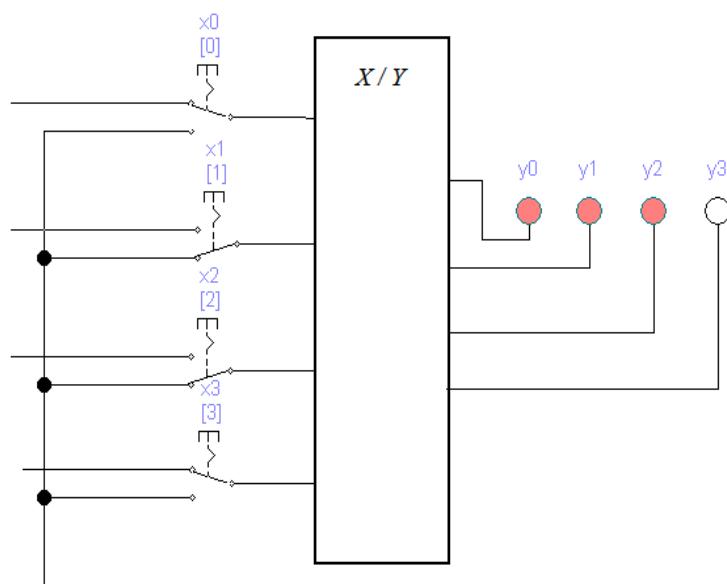


Рисунок 7.7 – Ввод параметров ключей

Выполните соединение всех элементов Вашего комбинационного устройства. Включите симулятор и проверьте ТИ (табл. 7.1).

Войдите в меню Instruments (рис. 7.8) и переместите на рабочее поле логический конвертор, выполните его соединение с одним из выходов Вашего преобразователя как показано на рисунке 7.9.



Рисунок 7.8 – Логический конвертор (Logic Converter)

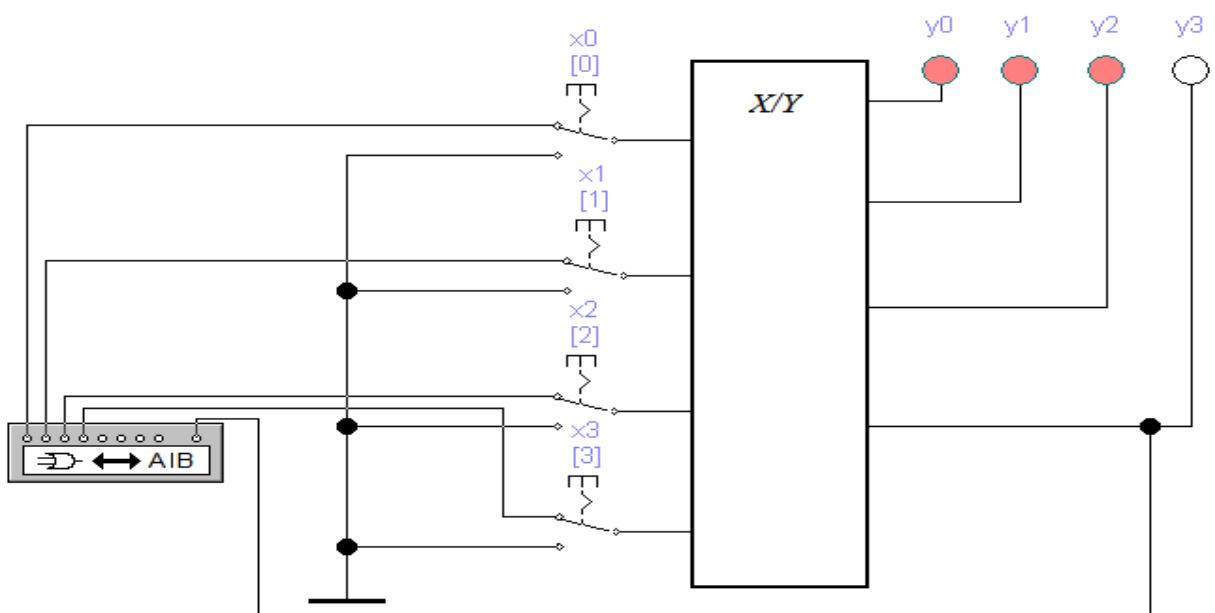


Рисунок 7.9 – Подключение конвертора

3. Откройте переднюю панель логического конвертора. Активируйте кнопки ABCD путем их нажатия - они соответствуют входным переменным $x_0x_1x_2x_3$.

Для получения таблицы истинности нажимаем на клавишу . Последовательно присоединяя выход конвертора к индикаторам Y_n заполните ТИ и сравните её с заданием.

Каждую из функций можно минимизировать. Для этого нажмите кнопку и из нижней строки конвертора выпишите минимальную форму.

Проделайте это для всех выходов и сравните с результатами, полученными при ручной минимизации. Все преобразования приведите в отчете по лабораторной работе.

7.5 Результаты работы

Подготовьте отчет по лабораторной работе, который должен содержать исходные данные, результаты синтеза преобразователя кода (карты Карно, обоснование выбора элементов, схему преобразователя), схему подключения для исследования преобразователя кода в формате EWB, результаты проверки таблиц истинности.

7.6 Контрольные вопросы

1. В чем сущность синтеза комбинационных схем?
2. В чем специфика синтеза многовыходных комбинационных схем?
3. Почему задача синтеза неоднозначна?
4. Какие критерии минимизации применяют при синтезе комбинационной схемы?
5. Для чего предназначены преобразователи кодов?

8 Лабораторная работа № 8

Матричная реализация логических функций

8.1 Цель работы

Целью работы является знакомство с программируемыми логическими матрицами (ПЛМ).

8.2 Пояснения к работе

В качестве функциональных узлов в цифровых вычислительных устройствах широко используются матричные схемы.

Матричная схема – сетка ортогональных проводников в местах пересечения которых, могут быть установлены элементы с односторонней проводимостью (ЭОП) – диоды или транзисторы. Возьмем такую матрицу (рис.8.1) и назовем её матрицей M1.

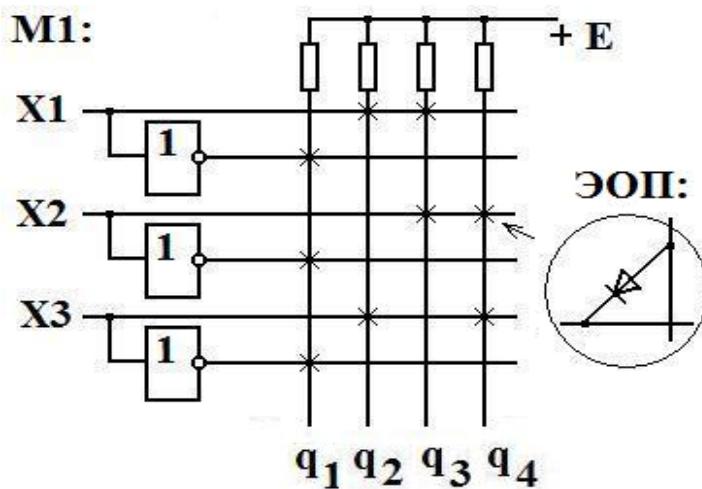


Рисунок 8.1 – Матрица конъюнкторов

Такой способ соединения некоторых узлов позволяет получить на выходах q_i любую конъюнкцию входных переменных. При указанных на рисунке местах установки ЭОП получаем:

$$q1 = \overline{x1} * \overline{x2} * \overline{x3}$$

$$q2 = x1 * x3$$

$$q3 = x1 * x2$$

$$q4 = x2 * x3$$

Возьмем другую матрицу и назовем её M2 (рис.8.2)

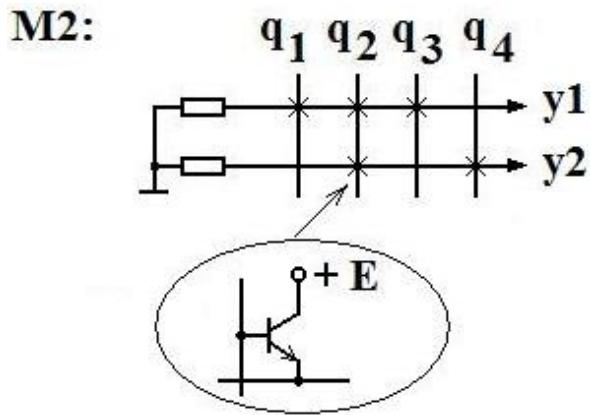


Рисунок 8.2 – Матрица дизъюнкторов

Эта матрица позволяет реализовать любую дизъюнкцию переменных q_i на выходах y_1 и y_2 . Здесь :

$$y_1 = q_1 + q_2 + q_3 \quad y_2 = q_2 + q_4$$

Если соединить эти матрицы, как показано на рис. 8.3, то можно получить логическую функцию представленную в ДНФ.

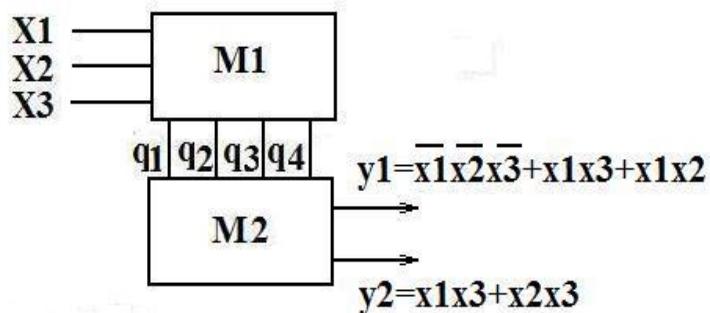


Рисунок 8.3 – Матричная реализация ДНФ

Это двухуровневая матричная структура. Имеется матрица (слой) конъюнкторов и матрица (слой) дизъюнкторов. Реализация логических функций в таких структурах сводится к определению точек пересечения проводников, где должны быть установлены ЭОП. Эта процедура называется настройкой матриц или программированием, а сами матрицы называются ПЛМ-программируемые логические матрицы. В зависимости от способа настройки матриц их делят на следующие типы:

-М – масочно- программируемые. В них ЭОП устанавливают при изготовлении самих матриц на предприятии. Эти соединения изменены быть не могут. Простая технология и высокая надежность;

-П – программируемые пользователем. При изготовлении матрицы ЭОП установлены во всех пересечениях и пользователь по своему усмотрению устраниет(выжигает) не нужные связи на специальном программаторе. Это более гибкая и оперативная система, но надежность таких соединений ниже, чем у масочных;

- Р – репрограммируемые матрицы. Здесь ЭОП выполнены на МОП- транзисторах с плавающим затвором. При программировании затворы заряжают лавинной инжекцией (обратимым пробоем окружающего затвор изолирующего слоя), т.е. электрическим способом, переводя транзистор в состояние замкнут/разомкнут. Такой заряд сохраняется очень долго. Эти матрицы допускают перезапись информации. Стирать информацию можно путем ультрафиолетового или рентгеновского облучения всей матрицы через кварцевое окошко (матрица типа РФ) или электрическим способом, разряжая каждый затвор (матрица типа РР).

Обычно ПЛМ изображают в другом виде (рис.8.4).

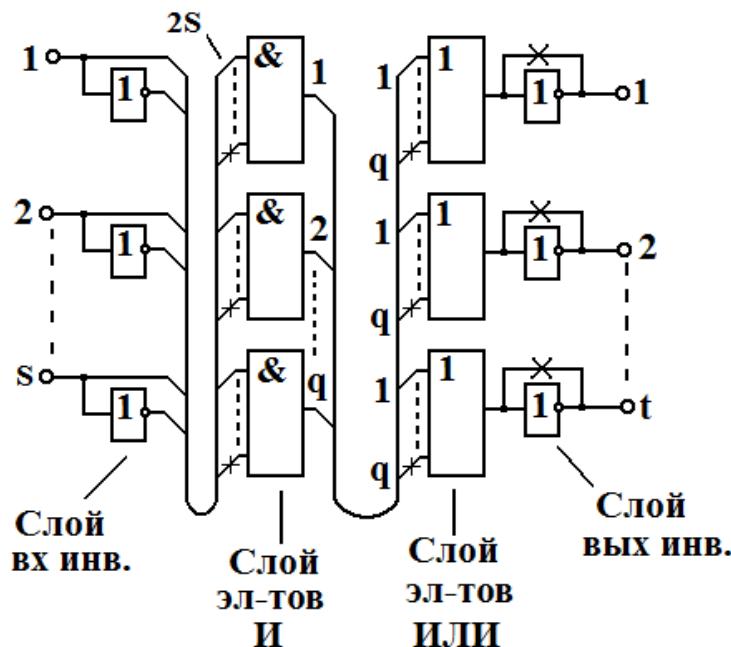


Рисунок 8.4 – Практическая схема ПЛМ

На рисунке приведена практическая схема, где крестиком обозначены места удаления ненужных связей. ПЛМ с S входами, t выходами и q промежуточными шинами называется ПЛМ(s,t,q). Если матрица M1 настроена на реализацию функции полного дешифратора ($q=2^S$), то такая ПЛМ называется ПЗУ (постоянное запоминающее устройство). На вход поступает код(адрес) и снимается код (данные), который определяется программированием матрицы M2.

Информационная емкость ПЛМ определяется площадью матриц (числом пересечений ортогональных проводников). Для ПЛМ емкость равна:

$$V_{PLM} = V_{M1} + V_{M2} = 2S \cdot q + q \cdot t = q(2S + q) ,$$

Для ПЗУ учитывается только площадь матрицы M2:

$$V_{ПЗУ} = q \cdot t = 2^S \cdot t .$$

Для матрицы ПЛМ (16,8,48) типа 556РТ1

$$V = q(2S + t) = 48(2 \cdot 16 + 8) = 1920 \text{бит}$$

Для ПЗУ с таким же числом входных и выходных линий получим

$$V = 2^S \cdot t = 2^{16} \cdot 8 = 65536 \cdot 8 = 64 \text{ K} \times 8 = 512 \text{ Кбит}$$

В таблице 8.1 приведены параметры некоторых ПЛМ и ПЗУ.

Таблица 8.1 Примеры матричных схем

№	Тип	S шт	t шт	q шт	t _{зад} нс	Примечание Р _{потр}
1	ППЛМ 556РТ1	16	8	48	70	ТТЛШ 0.44 мВт/бит
2	ППЗУ 556РТ16	13 (8к x 8)	8	-	85	ТТЛШ 0.015 мВт/бит
3	МПЗУ K568РЕ5	17 (128к x 8)	8	-	200	280 мкВт/бит
4	РПЗУ 1636 РР2	16М (2М x 8)	16	-	65	КМОП +3...3,6 В 3/150 мВт(хр/чт)
5	РПЗУ K573 РФ9	128к x 8	128	-	350	ЛИЗМОП t _{стир} =30 мин 0.5 мкВт / бит

8.3 Порядок выполнения работы

- Выпишите в соответствии со своим вариантом задания (номером бригады) логические функции из таблицы 8.2.

Таблица 8.2 – Варианты задания

Номер бригады	Y1	Y2	Номер бригады	Y1	Y2
1	V(0,2,7)	V(1,2,3,7)	6	Λ(1,2,5,7)	Λ(3,4,6,7)
2	Λ(0,2,4,6)	V(3,5,6)	7	V(4,5,6,7)	V(0,1,2,5)
3	Λ(2,4,6,7)	Λ(1,2,5,7)	8	Λ(1,2,4,5)	Λ(1,3,5,6)
4	V(1,2,4,7)	Λ(0,3,5,6)	9	V(2,3,7)	Λ(3,5,6,7)
5	Λ(0,3,4,7)	V(0,2,3,6)	10	Λ(1,3,5,6)	V(2,4,7)

- Составьте таблицу истинности функций и запишите две СДНФ

- Подсчитайте число различных мinterмов в обеих функциях и, тем самым, определили требуемое число промежуточных линий - q .
- Откройте файл matrica.ewb и расставьте ЭОПы в нужных пересечениях М1 и М2. При необходимости добавьте ещё линии q.

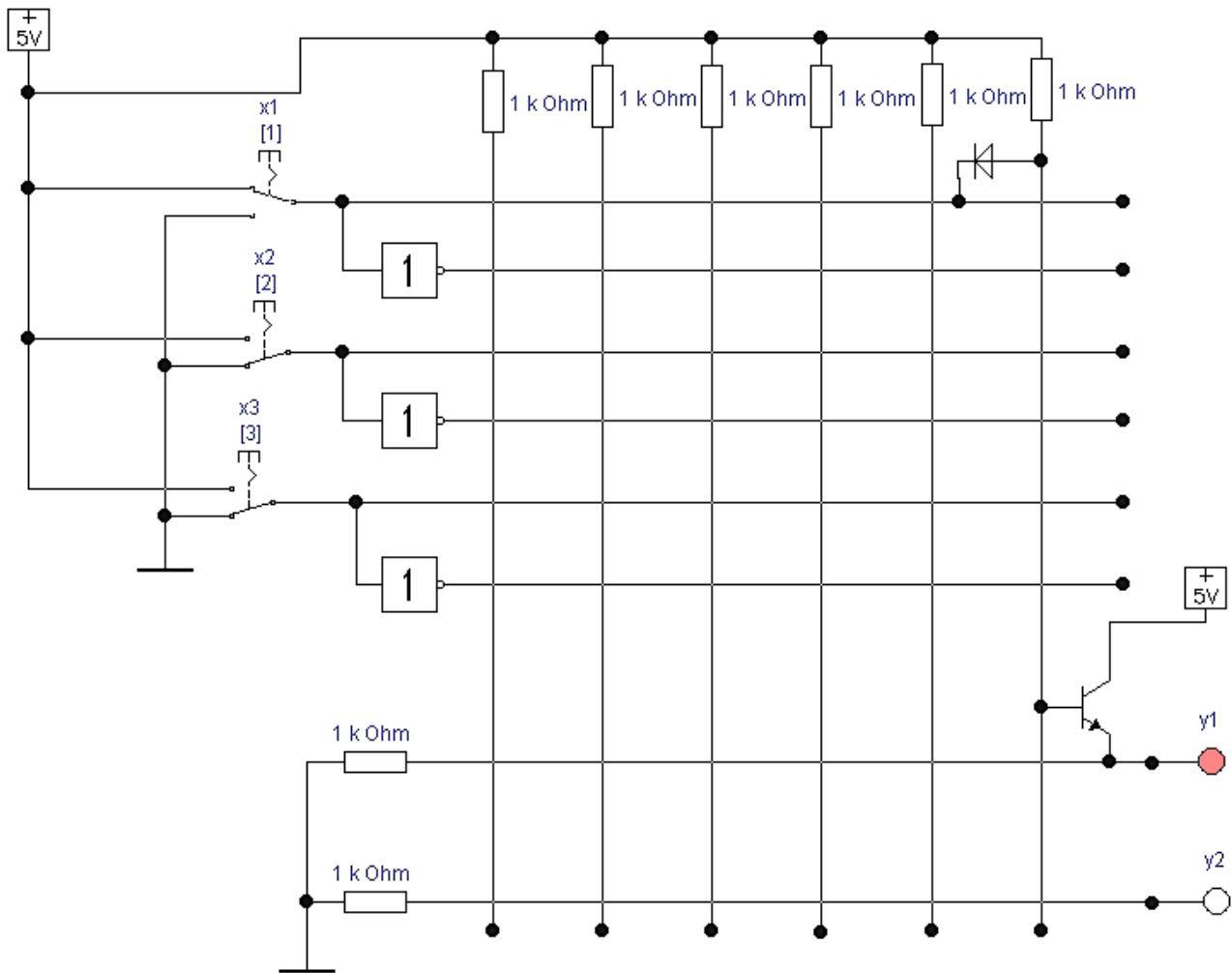


Рисунок 8.5 Файл matrica.ewb – схема-заготовка с примером постановки ЭОП

- Включите схему и убедитесь в правильности исходной ТИ. Сохранять изменения в схеме НЕ НАДО!
- Изобразите на бумаге схему ПЛМ для реализации следующей ТИ.

Адрес x1x2x3	Данные			
	Y1	Y2	Y3	Y4
0 0 0	0	1	1	0
0 0 1	0	1	1	1
0 1 0	1	0	0	0
0 1 1	1	1	0	1
1 1 0	0	1	0	0

Считаем, что x_1, x_2, x_3 – входные переменные(адрес), а y_1, y_2, y_3, y_4 – выходные переменные (данные).

8.4 Результаты работы

Подготовьте отчет по лабораторной работе, который должен включать исходные данные, ТИ, список минтермов, полученную схему и результаты проверки.

8.5 Контрольные вопросы

1. Что такое ПЛМ и ПЗУ?
2. Чему равна информационная емкость ИМС 1 и 3 таблицы 8.1?
3. В чем принципиальное отличие ИМС 3 и 5 таблицы 8.1?
4. Достоинства и недостатки масочных ПЗУ?
5. К какому типу матриц принадлежит ИМС 4 из таблицы 8.1?

9 Лабораторная работа № 9

Арифметические сумматоры

9.1 Цель работы

Целью работы является освоение процедуры синтеза одноразрядных и многоразрядных арифметических устройств - полусумматоров и полных сумматоров. Приобретение навыков работы в виртуальной электронной лаборатории EWB.

9.2 Пояснения к работе

Сумматор – это узел ЭВМ, предназначенный для сложения кодов двоичных чисел. Сумматоры делятся на последовательные (накапливающие) и параллельные (комбинационные). Накапливающие сумматоры имеют низкое быстродействие, поэтому они рассматриваться не будут. В комбинационных сумматорах слагаемые поступают на входы одновременно, а на выходе получается код суммы. После снятия слагаемых результат пропадает. Эти устройства не обладают памятью и строятся на логических элементах.

Составим таблицу истинности устройства для сложения двух одноразрядных чисел a и b (рис. 9.1).

№	a	b	p	s
0	0	0	0	0
1	0	1	0	1
2	1	0	0	1
3	1	1	1	0

Рисунок 9.1 – Таблица истинности для сложения двух цифр

Здесь p – перенос в старший разряд, s – значение суммы. Устройство, реализующее эту таблицу истинности, называют двоичным полусумматором. Его можно синтезировать по ФАЛ для каждого из выходов:

$$p = a * b$$
$$s = a * \bar{b} + \bar{a} * b = a \oplus b$$

Составим схему на произвольных элементах (рис. 9.2).

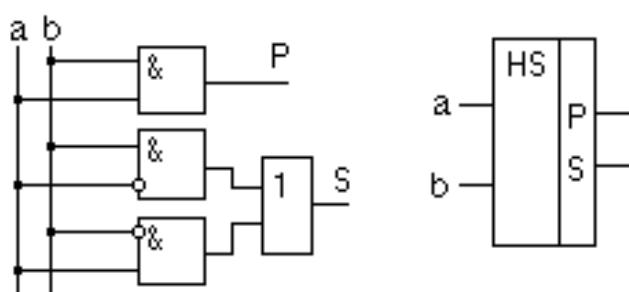


Рисунок 9.2 – Схемная реализация и условное обозначение полусумматора

При сложении многоразрядных чисел необходимо складывать три двоичных цифры в каждом разряде: два слагаемых и единицу переноса из предыдущего разряда P_{i-1} . Наличие этой единицы переноса несколько меняет таблицу сложения двоичных чисел (рис. 9.3).

№	a_i	b_i	P_{i-1}	P_i	S_i
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

Рисунок 9.3 – Таблица истинности для сложения трёх цифр

Система собственных функций:

для суммы: $S_i = \overline{a_i} * \overline{b_i} * P_{i-1} + \overline{a} * b * \overline{P}_{i-1} + a_i * \overline{b_i} * \overline{P}_{i-1} + a_i * b_i * P_{i-1}$
для переноса (рис. 9.4).

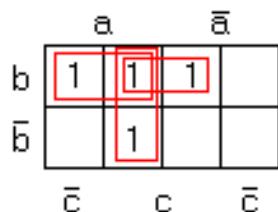


Рисунок 9.4 – Карта Карно для цепи переноса

Минимальная форма по этой карте $P_i = a_i * b_i + a_i * P_{i-1} + b_i * P_{i-1}$.

Уравнение для S_i не минимизируется. Устройство, реализующее эти ФАЛ, называется сумматором (полный сумматор). Он имеет три входа и два выхода. Цена сумматора по уравнениям составляет Ц = 25. Путем совместной минимизации уравнений S_i и P_i удается снизить цену до 20 и в таком виде выпускаются микросхемы сумматоров. Например, K555ИМ5 – полный одноразрядный сумматор (рис. 9.5).

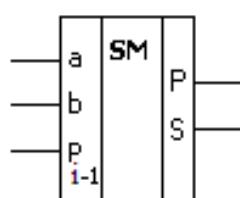


Рисунок 9.5 – Полный сумматор

Для сложения многоразрядных чисел сумматор составляют из одноразрядных.

Пусть требуется сложить два четырёхразрядных двоичных числа: А и В.

$$A = a_3 a_2 a_1 a_0$$

$$B = b_3 b_2 b_1 b_0$$

Составим схему сумматора (рис. 9.6).

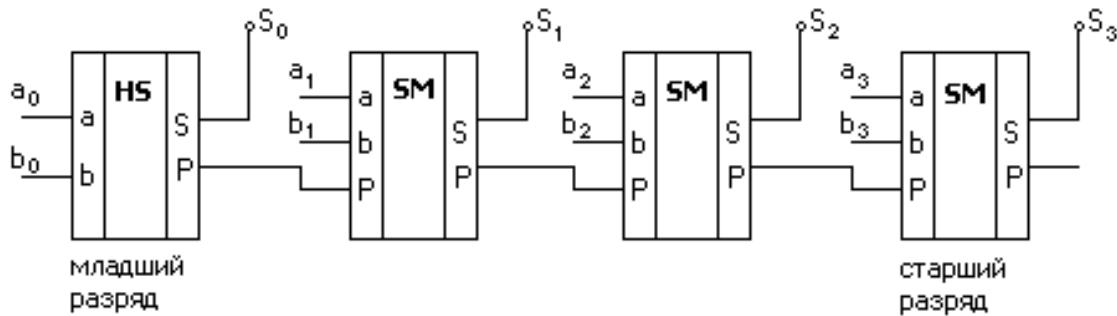


Рисунок 9.6 – Многоразрядный сумматор

Получился многоразрядный сумматор с последовательным переносом. Такие сумматоры выпускают в виде отдельных микросхем. Например, ИМС К155 ИМ3 четырёхразрядный сумматор с последовательным переносом. Время сложения чисел определяется временем распространения переноса и равно 55 нс (для четырёх разрядов). С ростом числа разрядов быстродействие сумматора уменьшается, так как цепь переноса последовательная.

Вспомним формулу переноса:

$$P_i = a_i * b_i + a_i * P_{i-1} + b_i * P_{i-1}$$

Найдём эти переносы:

$$P_0 = a_0 * b_0$$

$$P_1 = a_1 * b_1 + a_1 * P_0 + b_1 * P_0 = a_1 * b_1 + a_1 * a_0 * b_0 + b_1 * a_0 * b_0$$

$$P_2 = a_2 * b_2 + a_2 * P_1 + b_2 * P_1 = \dots$$

Видно, что имея только коды слагаемых, можно формировать перенос в любом разряде, не дожидаясь его появления в предыдущем разряде, причём с помощью только двухуровневой схемы (один слой конъюнкторов и один дизъюнктор). Схема вырабатывающая сигналы переноса называется **схемой ускоренного переноса** (параллельного переноса). Она может быть встроена в сумматор (сумматор с параллельным переносом) или выпускаться отдельно. Например, ИМС К555 ИМ6 – четырёхразрядный сумматор с параллельным переносом. Время сложения чисел равно 27 нс.

При большом числе разрядов сложность схемы ускоренного переноса сильно возрастает. Поэтому сумматор разбивают на группы по 4 или 8 разрядов. Внутри группы выполняют параллельный перенос, а между группами параллельный или последовательный. Такие сумматоры называют сумматорами с групповым переносом.

Многоразрядный сумматор условно обозначают так (рис. 9.7).

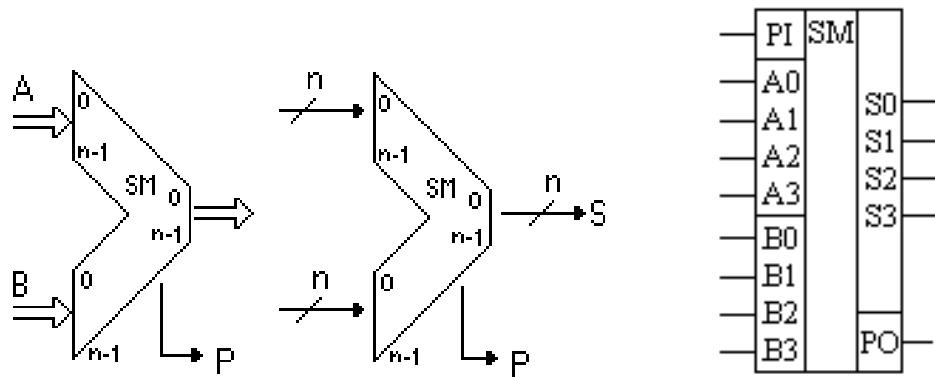


Рисунок 9.7 – Условное обозначение многоразрядного сумматора

С помощью сумматоров можно не только складывать, но и вычитать двоичные числа. При использовании дополнительных кодов операцию вычитания двух положительных чисел заменяют операцией суммирования положительного и отрицательного чисел, при этом получение дополнительного кода числа является элементарной операцией. Для этого необходимо проинвертировать двоичное число и добавить к нему в младший разряд +1.

Схема вычитателя числа А из числа В приведена на рисунке 9.8а, а схема вычитателя числа В из числа А на рисунке 9.8б.

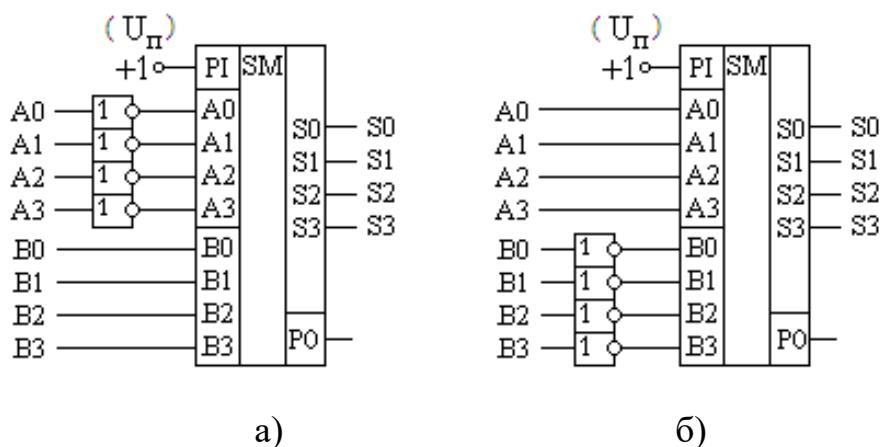


Рисунок 9.8. Схема вычитателя числа А из числа В(а) и числа В из А (б)

Схема инкремент/декремент

Возьмём три полусумматора и соединим их как показано на рис. 9.9.

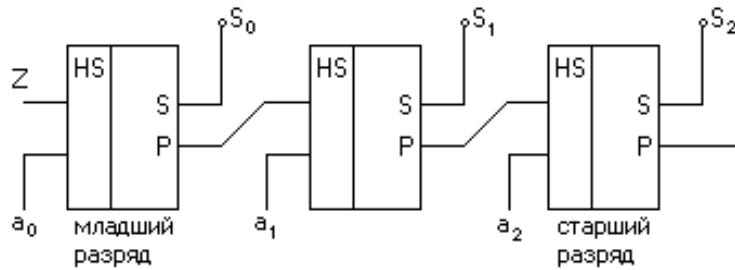


Рисунок 9.9 – Схема инкремент / декремент

Подавая на управляющий вход Z ноль или единицу, проанализируем состояние выхода при различных входных сигналах (рис. 9.10).

Z	Входной код			Выходной код		
	a_2	a_1	a_0	S_2	S_1	S_0
0	1	1	0	1	1	0
0	0	0	1	0	0	1
1	1	1	0	1	1	1
1	0	0	1	0	1	0

Рисунок 9.10 – Соответствие сигналов на входе и выходе схемы инкремент / декремент

Если на вход Z поступает 0, то число на выходе пройдет без изменений. Если на вход Z подать 1, то эта единица добавляется к младшему разряду числа (инкремент - плюс 1).

Если числа на входе и выходе проинвертировать, то мы получаем схему декремент (декремент это минус 1).



Эта схема самостоятельного значения не имеет, но широко используется как составная часть арифметико-логических устройств.

9.3 Порядок выполнения работы

- Откройте файл summator.ewb (рис 9.11). Проверьте таблицу истинности полусумматора на элементе M2 (рис 9.11а).
- Самостоятельно получите логическую формулу и логическую схему полусумматора в базисе И-НЕ, соберите её и проверьте правильность работы.
- Проверьте полусумматор и сумматор по схеме рис 9.11б.

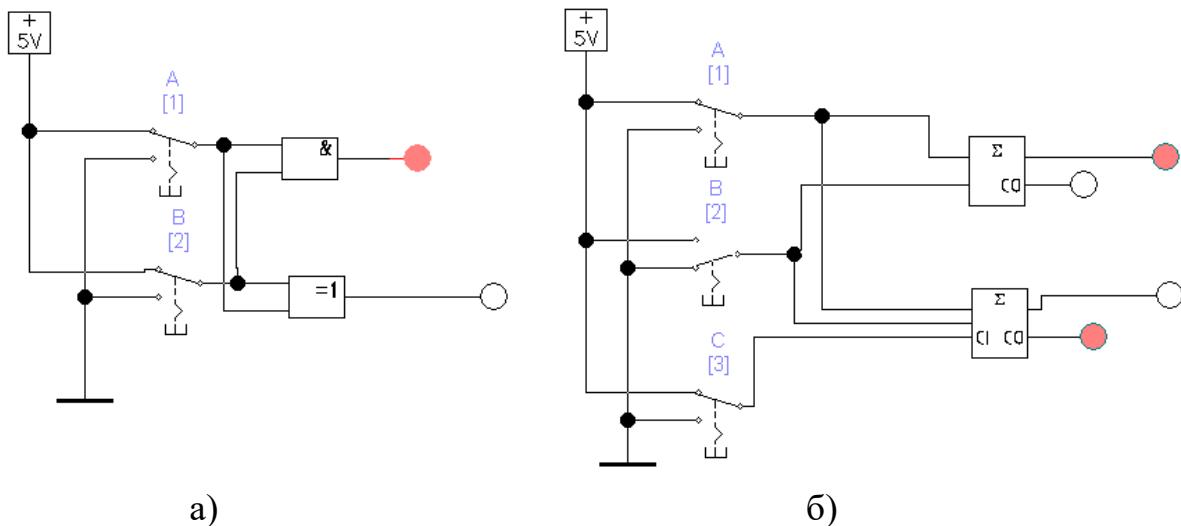


Рисунок 9.11 – Схема для проверки полусумматора и сумматора

Убедитесь в правильности работы схем по таблицам истинности.

4. Составьте схему параллельного трехразрядного сумматора (см. рис 9.6). Самостоятельно выберите и реализуйте на схеме вариант подачи трехразрядных чисел на входы сумматоров (от генератора кодовых слов или с помощью источника постоянного напряжения 5V и элемента заземления). Соберите и протестируйте схему.

9.4 Результаты работы

Отчет должен включать: 1. Цель работы. 2. Логические схемы, таблица истинности и логические формулы полусумматора. 3. Логические схемы, таблица истинности и логические формулы полного сумматора. 4. Логическая схема трехразрядного сумматора с тестовыми приборами. 5. Личные выводы по работе: что нового для себя узнали, что научились делать.

9.5 Контрольные вопросы

1. Чем отличаются сумматоры по модулю два, полусумматоры и полные сумматоры?
2. Перечислите процедуры синтеза логических схем полусумматора и полного сумматора.
3. Как осуществляется переход от таблицы истинности к логической формуле и далее к логической схеме в желаемом базисе?
4. Из каких соображений делается выбор для представления таблицы истинности логическими формулами в виде СДНФ и СКНФ?
5. Какие способы минимизации логических функций использованы в данной работе?
6. Какие машинные коды используются в компьютерах для представления чисел?
7. Как с помощью сумматора вычислить разность двух чисел?

10 Лабораторная работа № 10

Исследование триггеров

10.1 Цель работы

Целью работы является изучение различных типов триггеров, режимов их работы, особенностей переключения их состояний и приобретение практических навыков использования средств системы схемотехнического моделирования Electronics Workbench.

10.2 Пояснения к работе

Последовательностные устройства – это устройства с памятью. В них выходной сигнал определяется не только текущим состоянием входа как в комбинационных схемах, но и рядом предыдущих значений. Простейшим последовательностным устройством является триггер. Его особенностью является, способность бесконечно долго находится в одном из двух устойчивых состояний. Приняв одно состояние за ноль, другое за единицу, можно считать, что триггер хранит один бит информации.

Триггер характеризуется:

число входов $P \leq 3$

число выходов (один) Q и \bar{Q} .

число внутренних состояний два Q и \bar{Q} .

характеристическое уравнение: $Q(t+1) = \delta(Z(t), Q(t))$

где t - текущее время

$t+1$ - следующий момент времени

$Z(t)$ - входной сигнал в момент времени t

$Q(t)$ - состояние триггера в текущий момент времени

$Q(t+1)$ - состояние триггера, в которое он перейдет в следующий момент времени ($t+1$).

Триггеры классифицируют по ряду признаков:

1) по логическому функционированию (RS-триггер, это триггер с установочными входами; Т-триггер, это счетный триггер; D-триггер, это триггер передачи (задержки); JK и DV-триггеры это универсальные триггеры; RST, TV и другие это комбинированные триггеры).

2) по способу записи информации в триггер (синхронные и асинхронные). В синхронных триггерах добавлен управляющий вход C , который не является логическим. Сигнал на его входе разрешает перейти триггеру в нужное состояние.

3) по числу ступеней триггеры делятся на одноступенчатые (однотактные) и двухступенчатые (двухтактные).

Общее теоретическое количество возможных триггеров очень велико $W = 5^{2^P}$, где p – число информационных входов, если $p=1$, то $W=25$, если $p=2$, то $W=625$. Но это теоретическое число, так триггер может находиться в одном

из пяти возможных состояний: Q , \bar{Q} , *, 1, 0. Большинство из этих триггеров не имеют технического применения. Например, при $R = 1$ в настоящее время синтезировано и применяется только два типа. При $R = 2$ имеют физический смысл только 24, из них синтезировано 8, среди которых два универсальных. С помощью них решаются все возникающие технические задачи. В других триггерах потребности пока не возникало.

10.2.1 Асинхронные триггеры

Основным триггером, на котором базируются все остальные триггеры, является RS-триггер. Он имеет два логических входа: R - установка 0 (от слова Reset), S - установка 1 (от слова Set).

Простейший RS – триггер составляется из двух логических элементов, охваченных перекрёстной, положительной обратной связью (рис. 9.1)

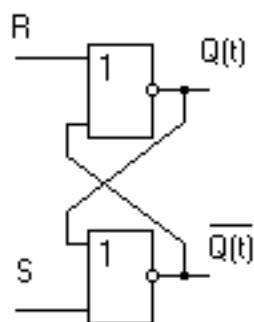


Рисунок 10.1 – Простейший RS-триггер

Один из выходов называют прямым, а другой – обратным. Состояние триггера определяется состоянием прямого выхода.

Составим таблицу истинности RS – триггера, учитывая, что имеем три независимых переменных R, S, Q(t) (рис. 10.2).

Такт t			Такт t+1	пояснения
R	S	Q(t)	Q(t+1)	
0	0	0	0	Режим хранения информации $R=S=0$
0	0	1	1	
0	1	0	1	Режим установки единицы $S=1$
0	1	1	1	
1	0	0	0	Режим установки нуля $R=1$
1	0	1	0	
1	1	0	*	$R=S=1$ запрещенная комбинация
1	1	1	*	

Рисунок 10.2 – Таблица истинности простейшего триггера

Составим характеристическое уравнение триггера по единичным значениям сигнала $Q(t+1)$. Для этого заполним карту Карно (рис. 10.3):

R	\bar{R}		
S	*	*	1
\bar{S}			1
<hr/>			
$Q(t)$			

Рисунок 10.3 – Кarta Карно

получаем $Q(t+1) = S + \bar{R} * Q(t)$ - характеристическое уравнение RS – триггера. Такой триггер условно изображается в следующем виде (рис. 10.4)

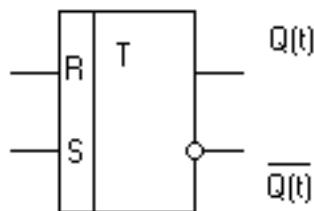


Рисунок 10.4 – Обозначение RS - триггера

Очевидно, что по характеристическому уравнению триггера его схему можно составить в любом базисе. Составим схему на элементах 2И-НЕ. Для этого в уравнении нужно избавиться от дизъюнкции. Применим двойное отрицание.

$$Q(t+1) = \overline{\overline{S + \bar{R}} * Q(t)} = \overline{\bar{S}} * \overline{\bar{R}} * \overline{Q(t)}$$

Учитывая, что сигналы $Q(t)$ и $Q(t+1)$ одна и та же физическая точка схемы, но в разные моменты времени, то эти точки соединяют, как показано на рисунке 10.5.

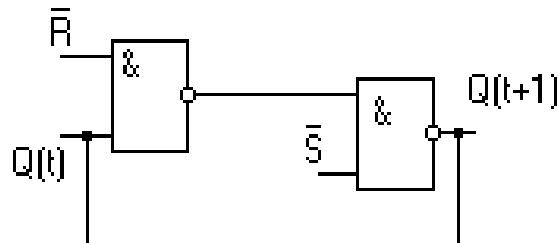


Рисунок 10.5 – Построение триггера на элементах 2И-НЕ

Получился дуальный триггер, в котором эффективным значением входного сигнала является нуль (рис. 10.6).

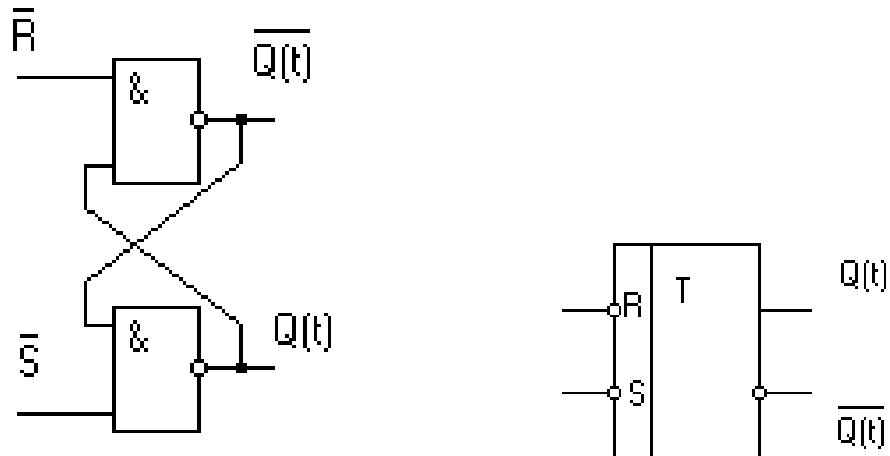


Рисунок 10.6 – Триггер RS на элементах И-НЕ

В нем комбинация входных сигналов $R=S=0$ является запрещенной, а $R=S=1$ - режим хранения. Это триггер с инверсным управлением.

Разновидностью RS-триггера являются: S-триггер, который при $R = S = 1$ активному уровню, переходит в единичное состояние; R - триггер, который при $R = S = 0$ активному уровню, переходит в нулевое состояние; E - триггер, который при $R = S = 1$ активному уровню, сохраняет предыдущее состояние $Q(t+1)=Q(t)$.

Составим словарь переходов RS-триггера, который показывает, какие сигналы следует подавать на входы, чтобы перевести триггер в нужное состояние. Словарь заполняют на основе ТИ. В нем всегда четыре строки (для любых триггеров). Словарь RS- триггера по рис.10.1 приведён на рисунке 10.7. Здесь прочерк означает безразличное состояние входа.

$Q(t)$	R	S	$Q(t+1)$
0	-	0	0
0	0	1	1
1	1	0	0
1	0	-	1

Рисунок 10.7 – Словарь переходов RS – триггера

Т-триггер

Счетный триггер (от слова topple - переключать). Имеет один информационный вход - Т. Функционирует согласно такой таблице истинности (рис. 10.8):

Такт t		Такт t+1
T	Q	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

Рисунок 10.8 – Таблица истинности T – триггера

Когда сигнал на входе T=0, то триггер сохраняет своё состояние. Когда на входе T= 1, то триггер меняет состояние на противоположное. Условное изображение T- триггера (рис. 10.9).

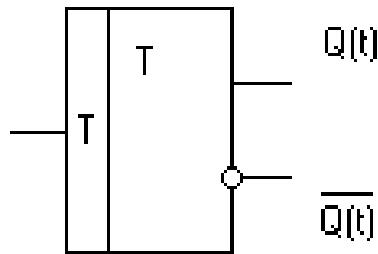


Рисунок 10.9 – Условное изображение T - триггера

Его характеристическое уравнение $Q(t + 1) = \bar{T} * Q(t) + T * \overline{Q(t)} = T \oplus Q(t)$. Счетный триггер выполняет сложение по модулю два (М2) входного сигнала и внутреннего состояния триггера Q (t). Составим схему T-триггера на основе RS-триггера. Для этого требуется создать комбинационную схему, преобразующую сигнал T в сигналы R и S, как показано на рисунке 10.10.

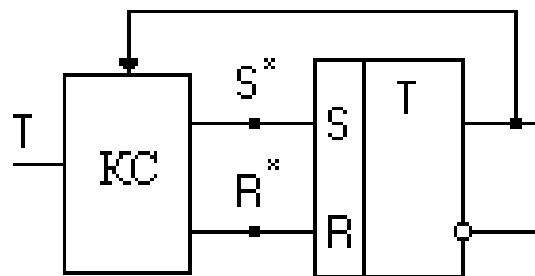


Рисунок 10.10 – Схема для синтеза T - триггера

Воспользуемся словарём переходов RS – триггера и пропишем в таблице истинности (рис. 10.11) такие сигналы R^*S^* , чтобы обеспечить требуемые переходы RS – триггера.

Такт t				Такт t+1
T	Q	R*	S*	Q(t+1)
0	0	-	0	0
0	1	0	-	1
1	0	0	1	1
1	1	1	0	0

Рисунок 10.11 – Таблица истинности для синтеза T - триггера

По единичным значениям сигналов R^* и S^* составим логические функции $R^* = T * Q$ и $S^* = T * \bar{Q}$, по которым легко составить схему (рис. 10.12).

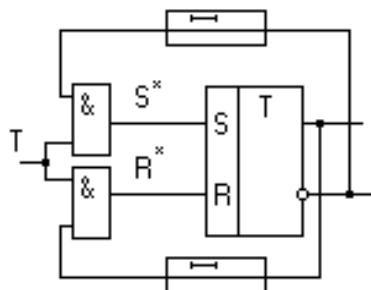


Рисунок 10.12 – Схемная реализация T – триггера на основе RS - триггера

Элементы задержки в цепи обратной связи на время равное длительности входного сигнала T необходимы, так как логические функции не учитывают фактор времени. Сигнал с выхода не должен поступить на вход раньше, чем закончится сигнал T. Такая задержка, в схемах реальных триггеров обеспечивается путём блокировки цепи обратной связи на время действия входного сигнала. Таким образом, получаем счётный триггер.

Словарь переходов T-триггера (рис. 10.13):

Q(t)	T	Q(t+1)
0	0	0
0	1	1
1	1	0
1	0	1

Рисунок 10.13 – Словарь переходов T – триггера

JK-триггер

Триггер универсальный. Он имеет два информационных входа J - установка единицы (Jerk – включение) и K - установка нуля (Kill –

выключение). В отличие от RS-триггера комбинация входных сигналов J=K=1 переводит триггер в противоположное состояние. Здесь нет запрещенных комбинаций входных сигналов J и K.

Составим таблицу истинности JK-триггера (рис. 10.14) и реализуем его на RS-триггере.

Такт t					Такт t+1
J	K	Q(t)	R*	S*	Q(t+1)
0	0	0	-	0	0
0	0	1	0	-	1
0	1	0	-	0	0
0	1	1	1	0	0
1	0	0	0	1	1
1	0	1	0	-	1
1	1	0	0	1	1
1	1	1	1	0	0

Рисунок 10.14 – Таблица истинности JK - триггера

Построим схему JK-триггера на основе RS- триггера (рис. 10.15). Для этого воспользуемся словарем перехода RS- триггера и в таблицу истинности запишем требуемые сигналы R * и S *.

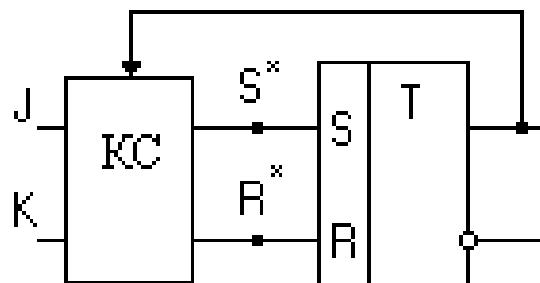


Рисунок 10.15 – JK- триггер на основе RS - триггера

Получаем систему собственных функций

$$R^* = \bar{J} * K * Q(t) + J * K * \bar{Q}(t) = K * Q(t)$$

$$S^* = J * \bar{K} * \bar{Q}(t) + J * K * \bar{Q}(t) = K * \bar{Q}(t)$$

и по ней составляем схему (рис. 10.16)

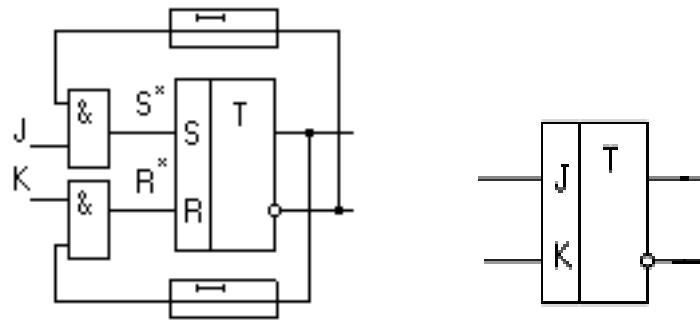


Рисунок 10.16 – Схема и обозначение JK - триггера

Словарь переходов JK – триггера (рис. 10.17):

$Q(t)$	J	K	$Q(t+1)$
0	0	-	0
0	1	-	1
1	-	1	0
1	-	0	1

Рисунок 10.17 – Словарь переходов JK - триггера

Из словаря переходов видно, что JK-триггер предоставляет больше возможности по комбинации входных сигналов, чем RS - триггер, поэтому схемные решения на его основе получаются более компактными.

10.2.2 Синхронные триггеры

В них добавлен управляющий вход, который не является логическим, он только разрешает перейти триггеру в нужное состояние. Словари переходов и таблицы истинности при этом не меняются. Например, синхронный RS-триггер (рис. 10.18)

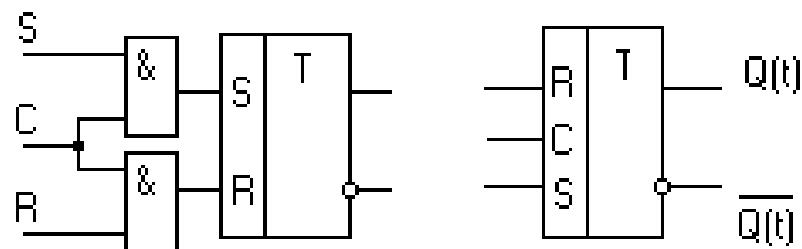


Рисунок 10.18 – Синхронный RS - триггер

Синхронный JK-триггер (рис. 10.19).

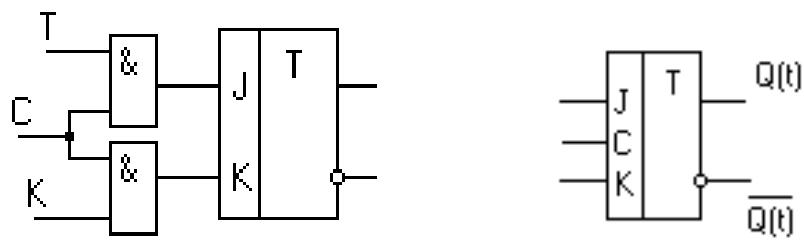


Рисунок 10.19 – Синхронный JK – триггер

Объединение входов J и K даёт T – вход (рис. 10.20), то есть получаем синхронный счётный триггер.

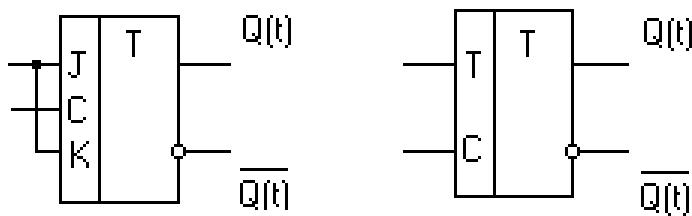


Рисунок 10.20 – Синхронный T - триггер

D-триггер

Триггер имеет один информационный вход - D (Delay – задержка, Drive – передача). Бывают только *синхронные*. Состояние информационного входа D передается на выход только с приходом синхроимпульса.

Составим таблицу истинности D-триггера и синтезируем его на базе RS – триггера (рис.10.21).

Такт t				Такт t+1
D	Q(t)	R*	S*	Q(t+1)
0	0	-	0	0
0	1	1	0	0
1	0	0	1	1
1	1	0	-	1

Рисунок 10.21 – Таблица истинности D - триггера

Состояние входа D передается на выход без учета внутреннего состояния триггера. Его характеристическое уравнение $Q(t+1) = D * \bar{Q}(t) + D * Q(t) = D$. Построим D-триггер на основе RS – триггера (рис. 10.22). Поступаем аналогично, как в JK и T – триггерах.

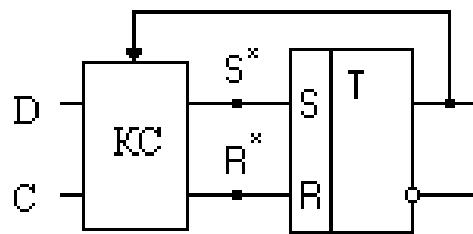


Рисунок 10.22 – Синтез D – триггера на основе RS - триггера

Записываем по единицам

$$R^* = \overline{D} * Q(t) + \overline{D} * \overline{Q(t)} = \overline{D}$$

$$S^* = D * \overline{Q(t)} + D * Q(t) = D$$

Составляем схему триггера (рис. 10.23)

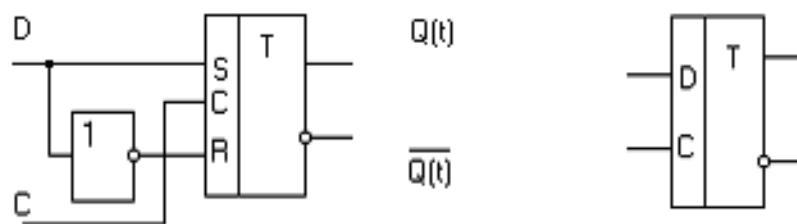


Рисунок 10.23 – D – триггер и его изображение

Назначение D – триггера запоминать и передавать входной сигнал. При этом происходит задержка во времени, но она зависит от момента появления сигнала на входе D и может быть различной. Проследим это на эпюрах (рис. 10.24).

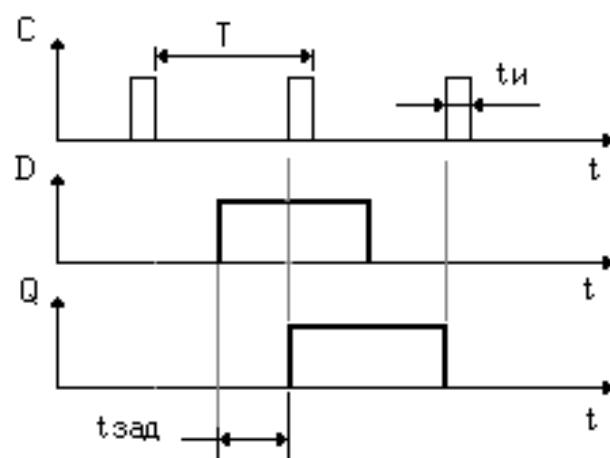


Рисунок 10.24 – Задержка сигнала в D - триггере

Максимальное время задержки равно длительности паузы между синхроимпульсами: $T_{зад} \leq (T - t_n)$

Двухтактные триггеры (MS-триггеры)

Двухтактный триггер состоит из двух ступеней – двух триггеров: первая ступень – M (master - хозяин), а вторая ступень – S (slave – помощник). Тип двухтактного триггера определяется типом триггера первой ступени. Возьмём двухтактный RS – триггер (рис. 10.25).

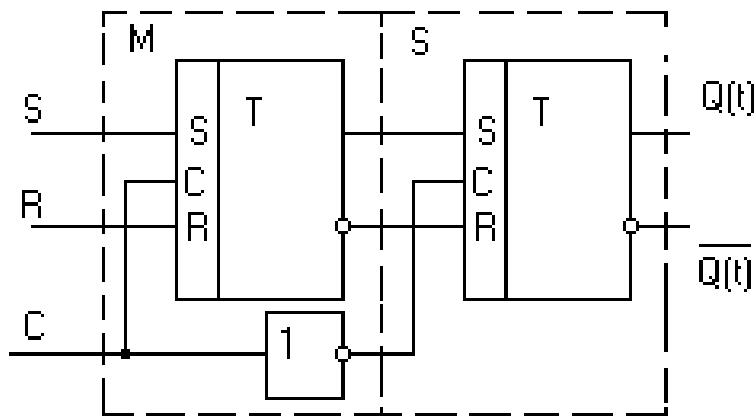


Рисунок 10.25 – Двухтактный RS - триггер

Если сигнал $C=1$, то первая ступень находится в режиме приема информации, а вторая - в режиме хранения, так как сигнал синхронизации на её входе равен нулю. Если $C=0$, то первый триггер переходит в режим хранения, а второй - в режим приема информации и копирует состояние первого триггера. Именно в этот момент информация появляется на выходе триггера (Q). Двухтактный триггер обозначается двумя буквами T (рис. 10.26).

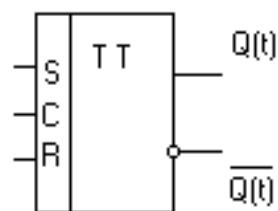


Рисунок 10.26 – Изображение двухтактного RS - триггера

Зачем нужны двухтактные триггеры? Во-первых, они имеют высокую помехоустойчивость, а во-вторых, с помощью двухтактного например D – триггера можно задержать сигнал на время равное периоду синхронизации (в однотактных - только на время паузы). Это поясняется эпюрами рис. 10.27

$$t_{34D} \leq T$$

Имея двухтактный D – триггер, несложно получить асинхронный T – триггер (рис. 10.28).

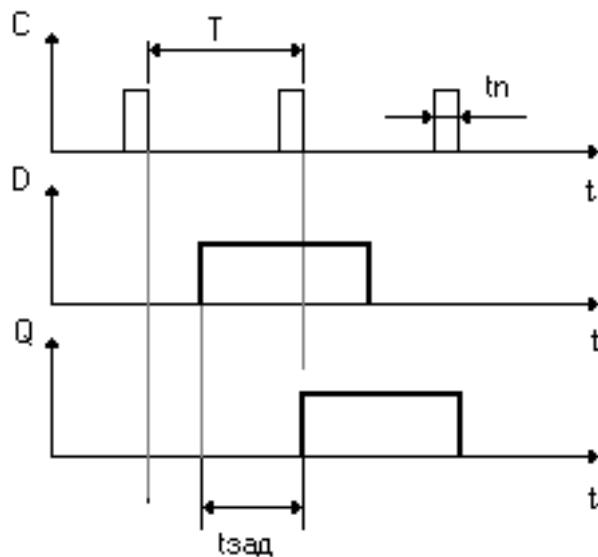


Рисунок 10.27 – Задержка сигнала в двухтактном D - триггере

Имея двухтактный D – триггер, несложно получить асинхронный T – триггер (рис. 10.28).

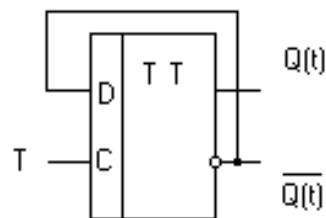


Рисунок 10.28 – Асинхронный T – триггер на основе двухтактного D - триггера

Вообще, любой триггер можно синтезировать на любом другом типе триггера, воспользовавшись словарём переходов последнего.

10.2.3 Способы управления триггерами

В зависимости от того какая часть синхросигнала используется для опрокидывания триггера, все триггеры делят на статические и динамические. Статические триггеры управляются уровнем сигнала – это потенциальные триггеры. Они опрокидываются, когда уровень синхросигнала выше некоторого порога (рис.10.29).

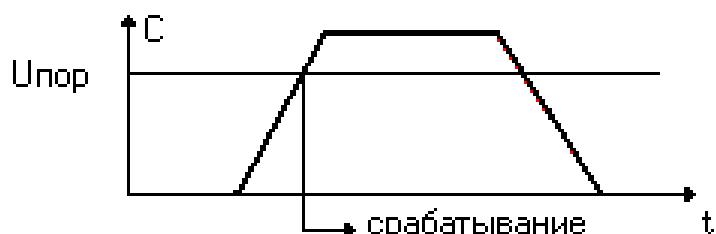


Рисунок 10.29 – Потенциальное управление

В некоторых случаях это неудобно, поэтому в триггерах с динамическим управлением информация воспринимается только во время перехода синхросигнала $0 \rightarrow 1$ или наоборот. Все остальное время информационные входы блокированы, как в двухтактных триггерах ($0 \rightarrow 1$ прямое динамическое управление; $1 \rightarrow 0$ обратное динамическое управление). Обозначения синхровходов (рис. 10.30):

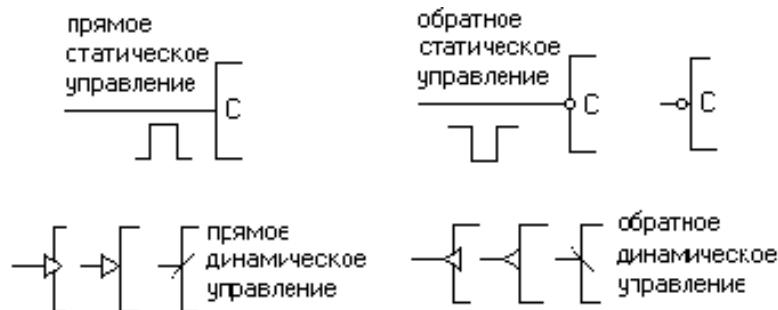


Рисунок 10.30 – Обозначения синхровходов триггеров

Синхровход в двухтактных триггерах всегда обозначается как статический. Рассмотрим влияние типа управления на выходные сигналы триггера. Возьмем пять D-триггеров с различными типами синхровходов, как показано на рисунке 10.31. Видно, что реакция триггера с обратным динамическим управлением и двухтактного триггера одинакова. Очевидно, что помехоустойчивость триггеров с динамическим управлением выше, чем статических. В динамических триггерах помеха может пройти на выход только, если она по времени накладывается на фронт синхроимпульса.

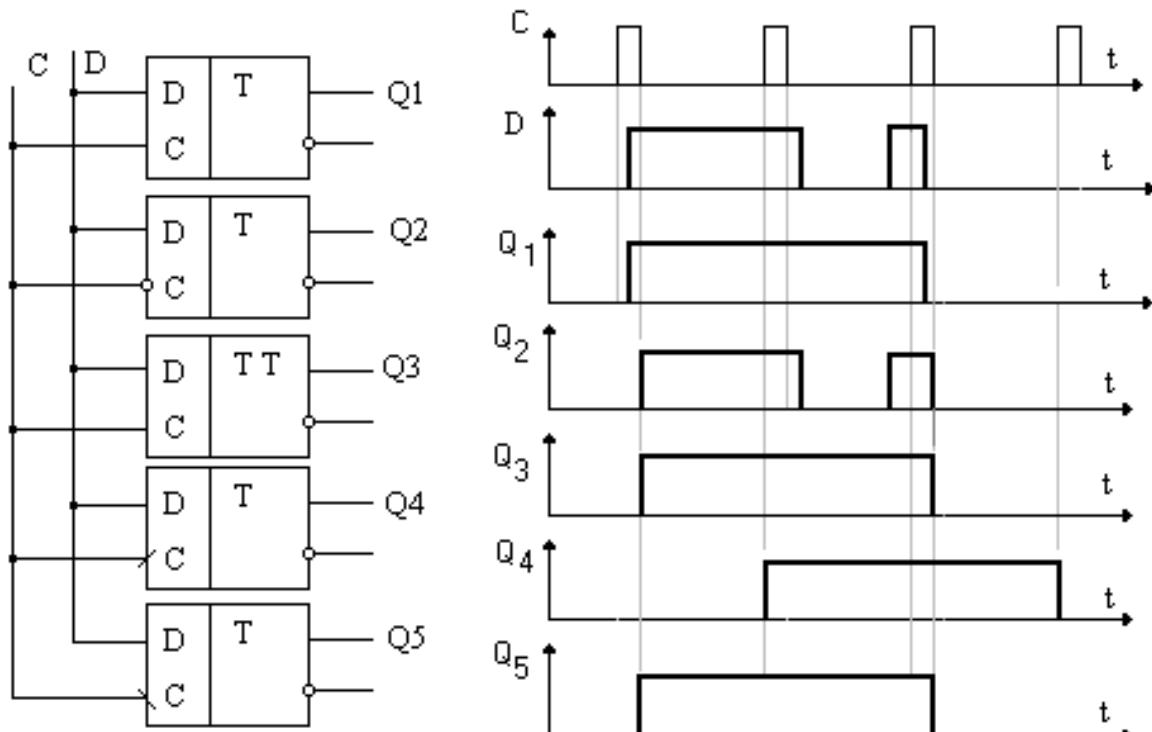


Рисунок 10.31 – Влияние типа управления на выходной сигнал D – триггера

10.2.4. Модели триггеров в программе EWB

В библиотеке EWB триггеры представлены тремя типами: RS, JK и D, показанными на рис. 10.32.

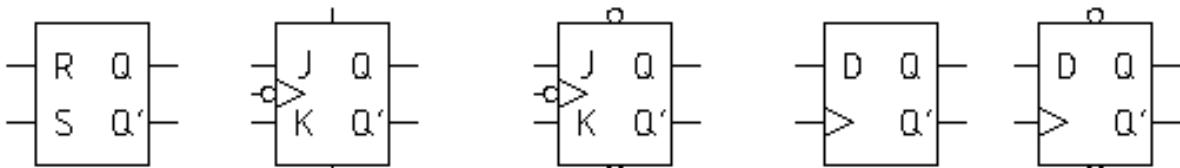


Рисунок 10.32 – Модели триггеров программы EWB

Назначение выводов триггеров соответствуют нашим обозначениям в тексте. Приведем лишь некоторые уточняющие сведения. Для JK триггера: > — тактовый вход; ввод сверху — асинхронная предустановка триггера в единичное состояние ($Q=1$); ввод внизу — асинхронная предустановка триггера в нулевое состояние (очистка триггера, после которой $Q' = 1$). Наличие кружочков на изображениях выводов обозначает, что активными являются сигналы низкого уровня, а для тактового входа — что переключение триггера производится по заднему фронту тактового импульса.

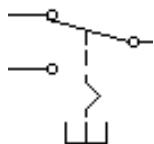
10.3 Порядок выполнения работы

Лабораторная работа предполагает исследование четырех триггеров: RS-триггера, D, и JK. Для исследования используются триггеры из библиотеки EWB. Схема измерений представлена на рисунке 10.33. (файл triggery.EWB)

Схема содержит следующие элементы:

- идеальный источник положительного потенциала 5В;

[1]



- ключи , управляемые клавишами (1,2,3,4) для набора входных сигналов триггеров;

- генератор синхросигналов (G) частотой 0,2 Гц с амплитудой 5 вольт;
- DD1 – синхронный JK-триггер, включенный по схеме T-триггера;
- DD2 – синхронный JK-триггер, включенный по схеме D-триггера;
- DD3 – синхронный D-триггер;
- DD4 – асинхронный RS – триггер, дополненный двумя элементами “И” и инвертором, что превращает его в синхронный D-триггер;
- логический анализатор.

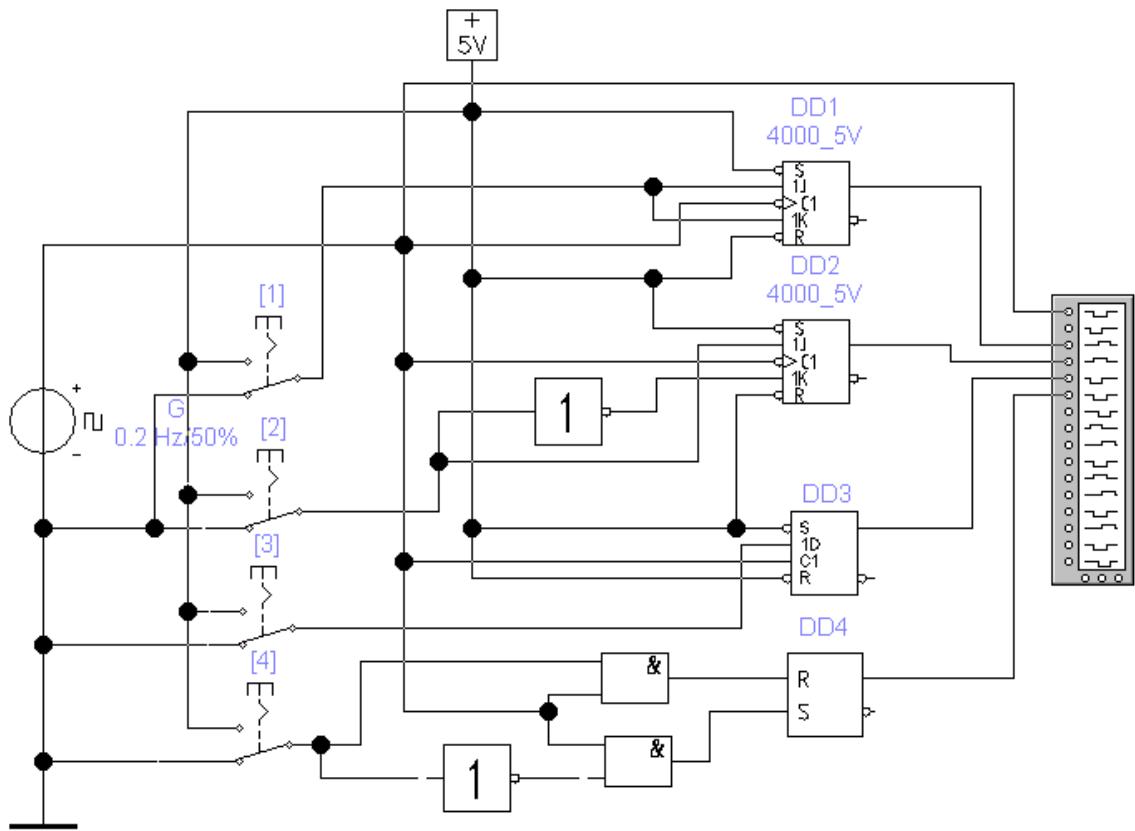


Рисунок 10.33 – Схема для исследования триггеров

Порядок выполнения работы:

1. Двойным щелчком откройте логический анализатор (рис.10.34).

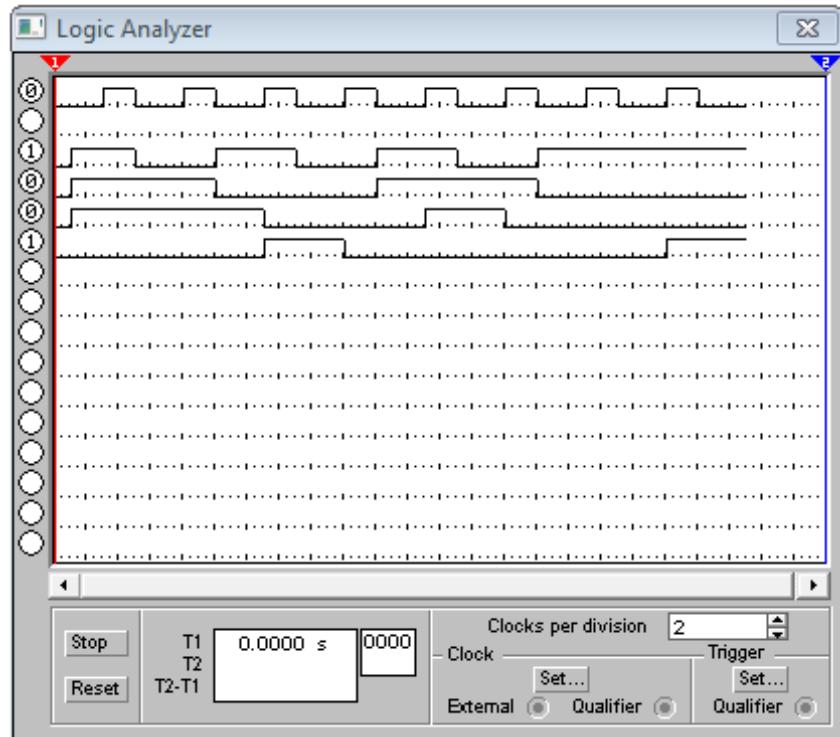


Рисунок 10.34 – Выходные сигналы триггеров

Войдите в настройки блока Clock – Set: выберите опции Positive, Internal, как показано на рис. 10.35

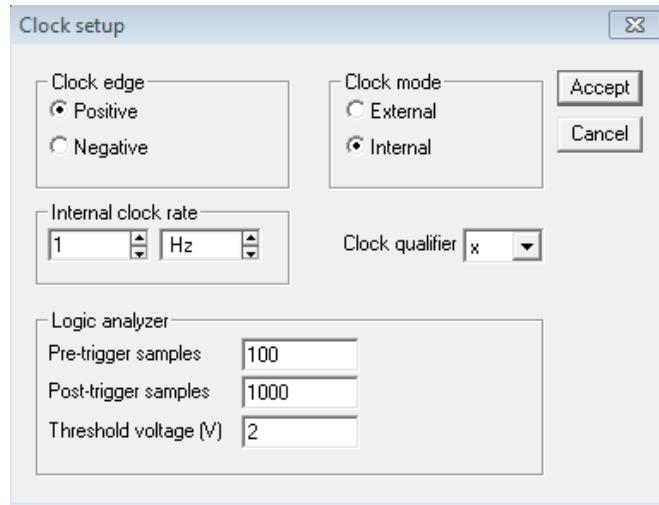


Рисунок 10.35 – Опции установок логического анализатора

Нажмите Accept. В блоке Trigger нажмите Set. Установите настройки согласно рис. 10.36

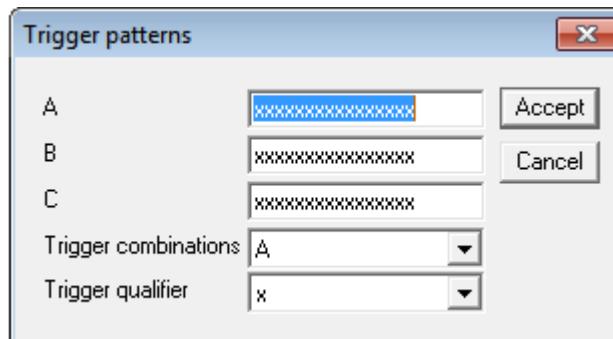


Рисунок 10.36 – Опции настроек в блоке Trigger

Нажмите Accept и выход.

2. Включите схему клавишей в правом верхнем углу экрана. Все ключи переведите в верхнее положение. Дождитесь заполнения экрана и выключите питание. Поясните все выходные сигналы. Все ключи переведите в нижнее положение. Дождитесь заполнения экрана и выключите питание. Поясните все выходные сигналы.

10.4 Результаты работы

Подготовьте отчет по лабораторной работе, который должен содержать исходные данные, схему измерений, временные зависимости с экрана

логического анализатора, пояснения временных зависимостей для каждого триггера.

10.5 Контрольные вопросы

1. Какое устройство называется последовательностным?
2. Как из D-триггера получить T-триггер?
3. Как из JK - триггера получить T-триггер, D-триггер?
4. Объясните, почему в запрещенном состоянии сигналы на обоих выходах RS-триггера одинаковы?
5. Что означает термин «переключение по отрицательному фронту»?
- 6 Чем отличаются одно и двухтактные триггеры. К каким из них принадлежит каждый из исследуемых триггеров?
7. Поясните назначение входов R и S у первых трех триггеров?
8. Что такое R-триггер, S-триггер, E- триггер?

11 Лабораторная работа № 11

Регистры

11.1 Цель работы

Целью работы является изучение способов построения регистров. Получение навыков синтеза регистров на триггерах различных типов и моделирование режимов работы параллельного регистра.

11.2 Пояснения к работе

Регистр представляет собой набор триггеров, охваченных общей цепью синхронизации. Триггеры называют разрядами регистра.

По способу ввода / вывода информации различают регистры:

1) параллельные (регистры хранения, информация вводится и выводится одновременно по всем разрядам);

2) последовательные (регистры сдвига, информация, бит за битом «проталкивается» через регистр и выводится так же последовательно, бит за битом);

3) комбинированные (параллельный ввод, последовательный вывод или наоборот).

По способу представления информации регистры делятся на однофазные и парафазные.

В однофазном регистре информация представляется в прямом или обратном (инверсном) виде, в парафазных и в том, и в другом виде одновременно, то есть в информации всегда имеются нули и единицы.

На рис.11.1 изображён трёхразрядный регистр хранения на RS-триггерах (это парафазный регистр, так как есть Q и \bar{Q}).

Регистр обозначают идентификатором (имя регистра) $RG [1 - 18]$, $P1[1 - 6]$. Здесь [...] количество разрядов.

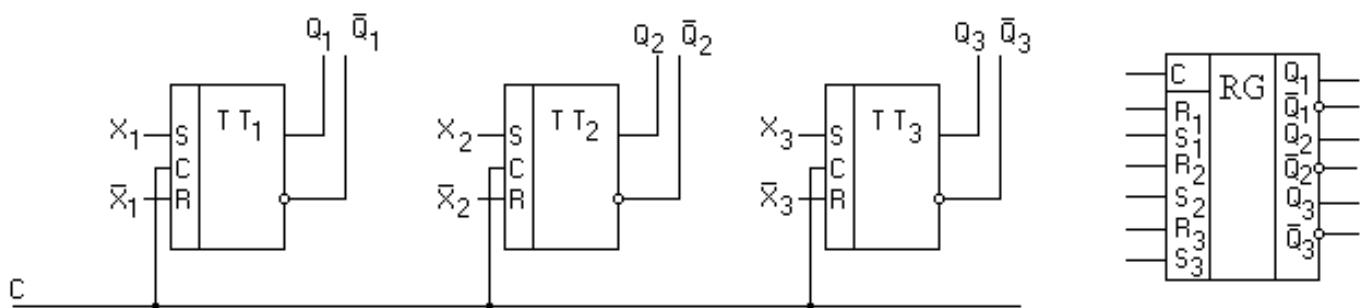


Рисунок 11.1 – Парафазный регистр хранения

Совокупность всех входов образует входную шину, а выходов – выходную шину. Условное обозначение регистра показано на рис. 11.2.

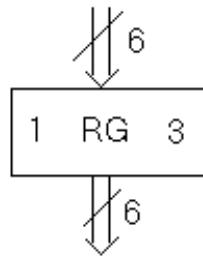


Рисунок 11.2 – Условное обозначение регистра

На схемах регистры и их соединение изображают так, как показано на рис. 11.3.

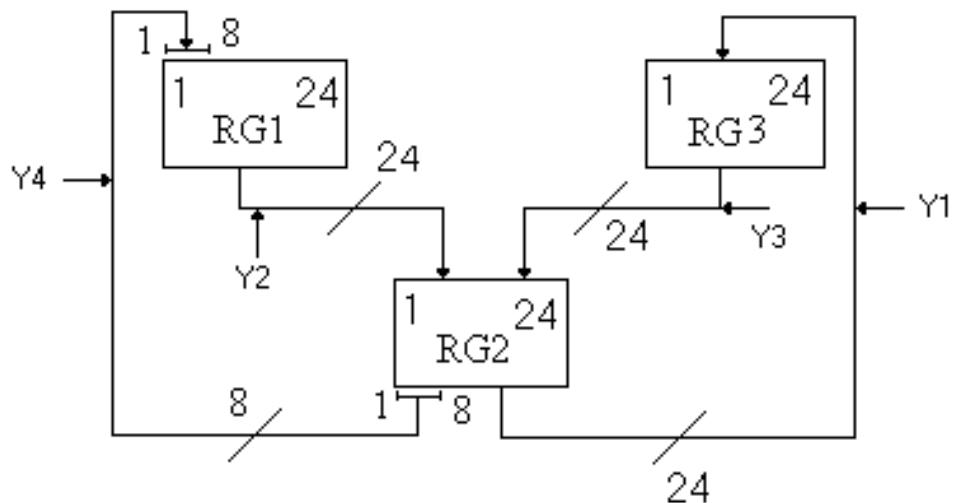


Рисунок 11.3 – Соединение регистров хранения

В этой схеме с тремя 24-разрядными регистрами выполняются следующие микрооперации передачи:

- Y1: $RG3 := RG2$
- Y2: $RG2 := RG1$
- Y3: $RG2 := RG3$
- Y4: $RG1[1-8] := RG2[1-8]$

Если часть разрядов в регистре имеют самостоятельное значение, то они называются субрегистром или подрегистром ($RG2[1-8]$ – подрегистр регистра $RG2[1-24]$). Регистры хранения могут быть построены на триггерах любых типов.

Синтез регистров хранения

Возьмем два четырехразрядных регистра хранения P1 и P2 на JK-триггерах и соединим их между собой с помощью конъюнкторов (рис. 11.4).

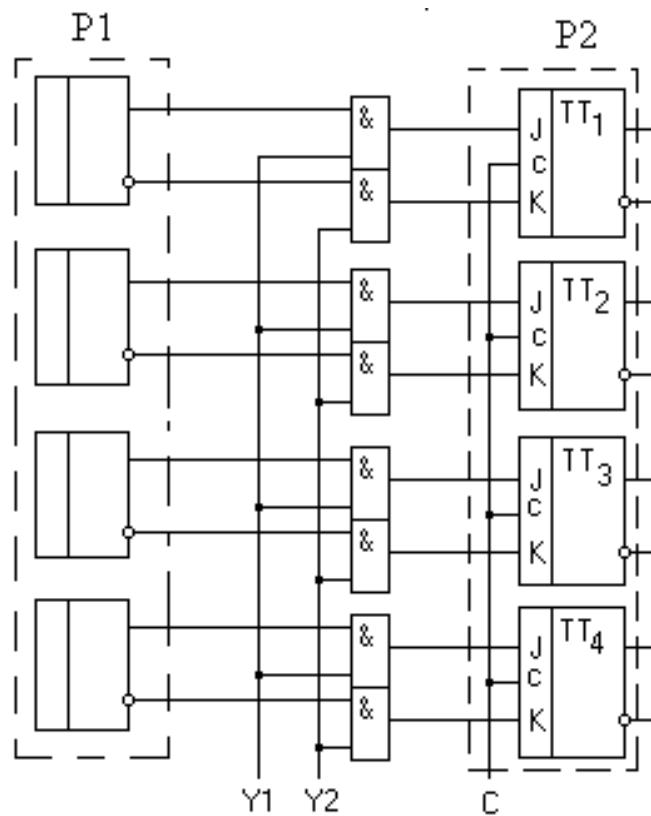


Рисунок 11.4 – Соединение регистров с помощью конъюнкторов

Входы Y_1 и Y_2 являются дополнительными управляющими входами – входами сигналов микроопераций (микрооперация - это элементарная функциональная операция, выполняемая за один такт под действием одного управляющего сигнала). На схемах такое соединение регистров условно обозначают следующим образом (рис. 11.5).

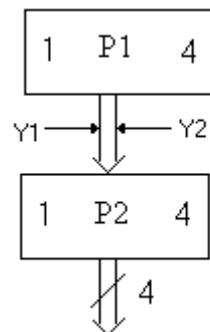


Рисунок 11.5 – Соединение регистров на операционных схемах

Рассмотрим, как будут влиять сигналы Y_1 и Y_2 на передачу информации с регистра P_1 на регистр P_2 .

Пусть, например, $Y_1=Y_2=0$. Тогда P_2 находится в режиме хранения своей информации, так как на выходах всех конъюнкторов имеются нули (на входах $J = K = 0$). Если $Y_1=1$, $Y_2=0$, то на входах $K = 0$ (нижние конъюнкторы), а состояние входов J рассмотрим по таблице (рис. 11.6), в которой состояние регистра определяется прямыми выходами триггеров.

Такт t			Такт t+1
P1	P2	Входы P2	Регистр P2
0	0	0 (J) 0 (K)	0
0	1	0 (J) 0 (K)	1
1	0	1 (J) 0 (K)	1
1	1	1 (J) 0 (K)	1

Рисунок 11.6 – Состояния входов и выходов регистра P2

Если на входе триггера $J = 1$, то на выходе P2 тоже будет единица. Таким образом, выполняется микрооперация дизъюнкции над содержимым регистров Y1: $P2 := P1 \vee P2$

Возьмём другой случай $Y1=0, Y2=1$ (рис. 11.7).

Такт t			Такт t+1
P1	P2	Входы P2	Регистр P2
0	0	0 1	0
0	1	0 1	0
1	0	0 0	0
1	1	0 0	1

Рисунок 11.7 – Состояния входов и выходов регистра P2

Видно, что выполняется другая микрооперация, а именно – конъюнкция

$$Y2: \quad P2 := P1 \wedge P2$$

Нетрудно видеть, что если $Y1=Y2=1$, то $P1 := P2$, то есть выполняется микрооперация передачи.

Мы выполнили анализ регистра, то есть соединили регистры и ответили на вопрос, что будет при таком соединении.

На практике обычно ставится задача не анализа, а синтеза, то есть как соединить регистры между собой, чтобы выполнялась требуемая микрооперация.

Синтез регистра сводится к синтезу одного разряда, так как все остальные разряды идентичны. Пусть, например, требуется синтезировать регистр хранения на Т-триггерах для выполнения следующих микроопераций между входной информацией - $X_{вх}$ и содержимым регистра - $Q(t)$:

- 1) передача;
- 2) дизъюнкция (V);
- 3) конъюнкция (Λ).

Составим такую схему (рис. 11.8).

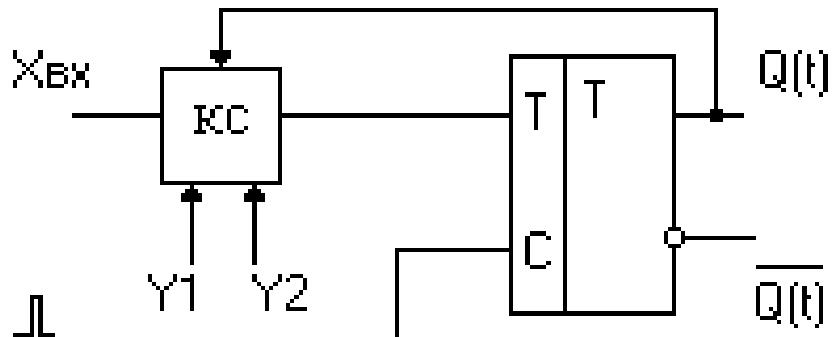


Рисунок 11.8 – Схема для синтеза одного разряда регистра

Здесь КС – комбинационная схема, которую и надо создать.

Составим таблицу истинности работы нашего устройства, но сначала необходимо произвольно закодировать управляющие сигналы, например, так (рис. 11.9).

Y1	Y2	Микрооперация
0	0	Передача
0	1	V
1	1	Λ

Рисунок 10.9 – Кодировка микроопераций

Составляем таблицу истинности для дальнейшего синтеза (рис. 11.10).

X _{вх}	Y1	Y2	Q(t)	Q(t+1)	T*
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	0	0
0	0	1	1	1	0
0	1	0	0	*	-
0	1	0	1	*	-
0	1	1	0	0	0
0	1	1	1	0	1
1	0	0	0	1	1
1	0	0	1	1	0
1	0	1	0	1	1
1	0	1	1	1	0
1	1	0	0	*	-
1	1	0	1	*	-
1	1	1	0	0	0
1	1	1	1	1	0

Рисунок 11.10 – Таблица истинности для синтеза регистра

Колонку с сигналом T^* - выходным сигналом комбинационной схемы заполняем на основании словаря переходов Т-триггера (рис. 11.11) и сигналов Q(t) и Q(t+1) таблицы истинности рис 11.10

Q(t)	T	Q(t+1)
0	0	0
0	1	1
1	1	0
1	0	1

Рисунок 11.11 – Словарь переходов Т - триггера

Составим карту Карно (рис. 11.12) и минимизируем функцию для T^* :

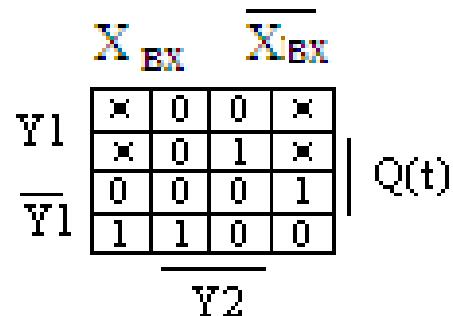


Рисунок 11.12 – Карта Карно для регистра

Получаем минимальную форму:

$$T^* = X_{BX} * \overline{Y_1} * \overline{Q(t)} + \overline{X}_{BX} * \overline{Y_2} * Q(t) + \overline{X}_{BX} * Y_1 * Q(t),$$

по которой и составляем комбинационную схему одного разряда регистра (рис. 11.13).

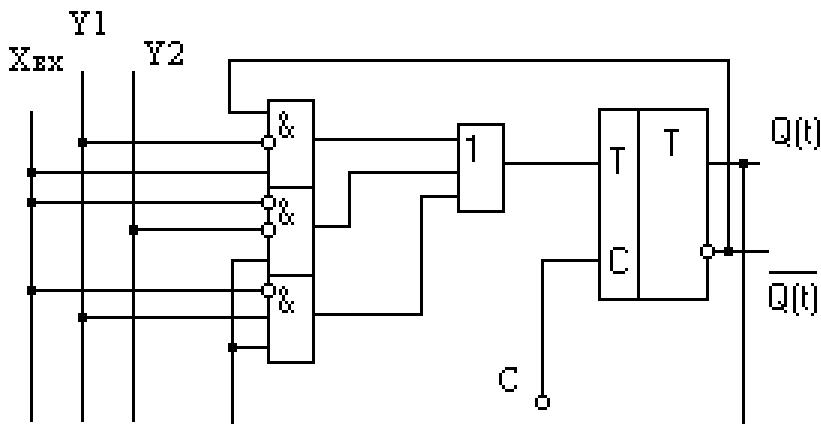


Рисунок 11.13 – Схема одного разряда регистра

По такому алгоритму регистры хранения можно создавать для выполнения различных микроопераций на триггерах любого типа.

Регистры сдвига

Служат для сдвига кода слова, занесённого в регистр. В зависимости от направления сдвига регистры делят на:

- 1) регистры со сдвигом вправо;
- 2) регистры со сдвигом влево;
- 3) реверсивные регистры.

Причём различают ещё и сдвиги арифметические, логические и циклические.

При арифметическом сдвиге смещаются все разряды кода, кроме знакового (крайнего левого бита). При логическом сдвиге смещаются все разряды кода, включая знаковый бит. При циклическом сдвиге крайние разряды соединены между собой, поэтому выдвигающийся бит помещается на освободившееся место.

При арифметическом и логическом сдвигах выдвигающиеся биты теряются, а освободившиеся места заполняются нулями.

Микрооперация сдвига записывается следующим образом:

Y1: P2:= сдв ЛП (2) P2 - сдвиг логический вправо на два бита содержимого регистра P2;

Y2: P1:= сдв АЛ (1) P1 - сдвиг арифметический влево на один бит содержимого регистра P1;

Y3: RG := сдв ЦП (1) RG – сдвиг циклический вправо на один бит содержимого регистра RG.

Регистр сдвига наиболее прост выполнить на D-триггерах (рис. 11.14).

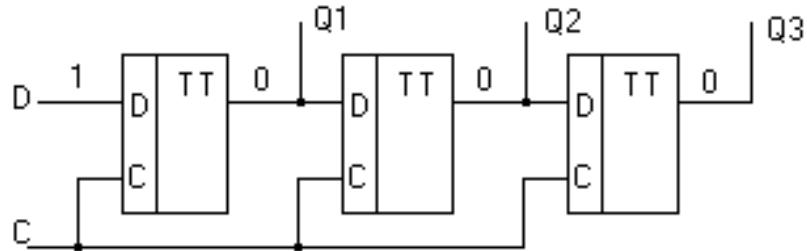


Рисунок 11.14 – Регистр сдвига

С приходом синхроимпульса код смещается вправо на один разряд. Регистр сдвига можно выполнить и на JK-триггерах. Для чего с помощью инвертора организуют D-вход (рис. 11.15).

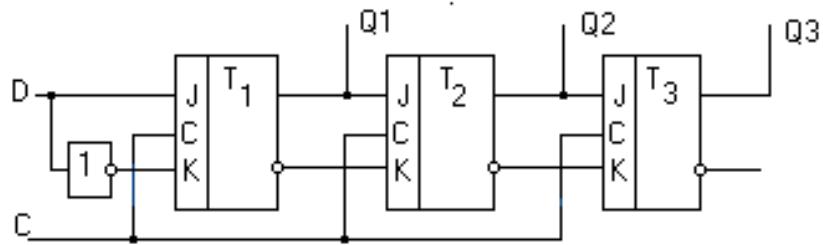


Рисунок 11.15 – Регистр сдвига на JK-триггерах

Так как цепь синхронизации является общей для всего блока (или узла), а микрооперацию сдвига требуется выполнить не всегда, то делают дополнительный вход через конъюнктор (рис. 11.16). При $V=1$ сдвиг будет выполнен по синхросигналу. При $V=0$ сдвига не будет.

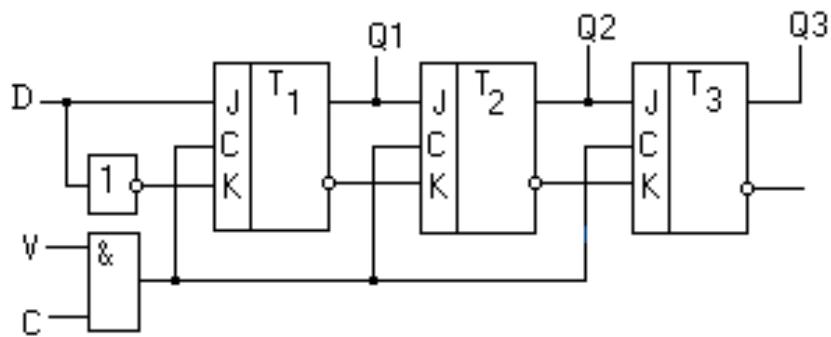


Рисунок 11.16 – Регистр сдвига на JK-триггерах с конъюнктором

Регистр сдвига на операционной схеме обозначают так (рис. 11.17).

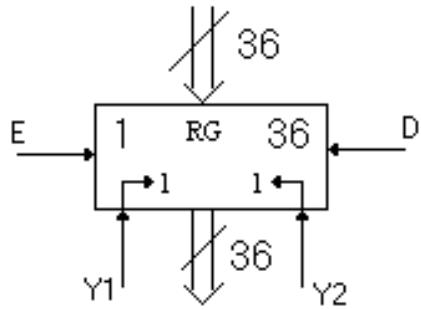


Рисунок 11.17 – Регистр сдвига на схеме

На этом рисунке обозначено: Y1 – вход микрооперации сдвига вправо на 1 бит; Y2 – вход микрооперации сдвига влево на 1 бит; D, E – дополнительная информация, которая помещается в освобождающиеся разряды. Тогда происходит составление нового слова на регистре:

$$\begin{aligned} Y1: \quad P1 &= \text{сдв ЛП (1)} \ P2 \mid E \\ Y2: \quad P2 &= \text{сдв ЛЛ (1)} \ P1 \mid D \end{aligned}$$

Микрооперация составления нового слова называется конкатенацией.

Регистры сдвига широко используются для организации последовательных АЛУ, преобразований кодов и выпускаются в виде отдельных микросхем (К155ИР1, К555ИР11 и др.).

Если вход и выход регистра сдвига соединить между собой, то код будет непрерывно циркулировать по замкнутому контуру, получается так называемый динамический регистр. Возьмём трёхразрядный регистр сдвига (рис. 11.18).

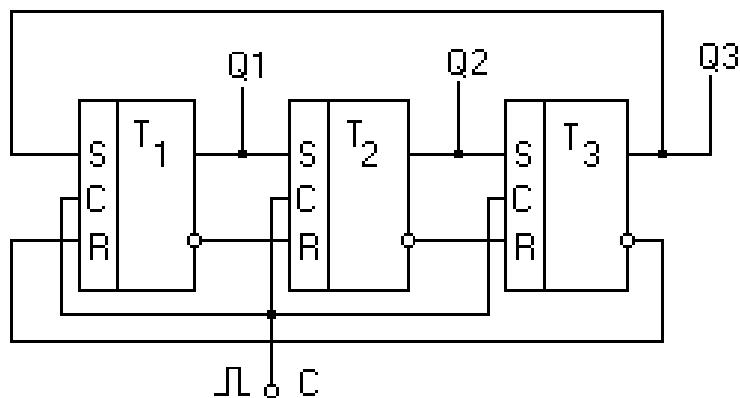


Рисунок 11.18 – Динамический регистр

Пусть в исходном состоянии на регистре записан код 1 0 0 (такт № 0). В каждом последующем такте эта единица будет перемещаться (рис. 11.19).

Такт	Q 1	Q 2	Q 3	t _{цсх} < t _{цц}	t _{цсх} > t _{цц}
0	1	0	0	1 0 0	1 0 0
1	0	1	0		
2	0	0	1	0 0 1	
3	1	0	0		
4	0	1	0	0 1 0	0 1 0
5	0	0	1		
6	1	0	0	1 0 0	
7	0	1	0		
8	0	0	1	0 0 1	0 0 1
9	1	0	0		
10	0	1	0	0 1 0	
11	0	0	1		
12	1	0	0	1 0 0	1 0 0
13	0	1	0		
14	0	0	1	0 0 1	
15	1	0	0		
16	0	1	0	0 1 0	0 1 0
17	0	0	1		
18	1	0	0	1 0 0	
19	0	1	0		

Рисунок 11.19 – Движение кода в динамическом регистре

Видно, что код на регистре повторяется через n -тактов, где $n = 3$ число разрядов регистра.

Существуют два понятия:

- 1) период циркуляции кода на регистре, он равен числу разрядов ($t_{цц} = n$);
- 2) период цикла схемы это время, через которое код считывается из регистра ($t_{цсх}$).

В зависимости от соотношения $t_{цц}$ и $t_{цсх}$ возможны три режима работы:

- 1) $t_{цц} = t_{цсх}$ - режим хранения информации;
- 2) $t_{цсх} < t_{цц}$ - режим со сдвигом влево (регистр со сдвигом влево);
- 3) $t_{цсх} > t_{цц}$ - режим со сдвигом вправо.

Получился универсальный регистр, но быстродействие его хуже чем у обычного регистра сдвига.

Динамические n – разрядные регистры соединяются в блоки по m штук (как показано на рис. 11.20) и записывают в них слова «поперёк». Всего можно записать n штук m – разрядных слов. Эти слова появляются в некотором сечении регистров через время равное n тактов. Получили так называемую регистровую память, или цифровую линию задержки (ЦЛЗ) на n тактов.

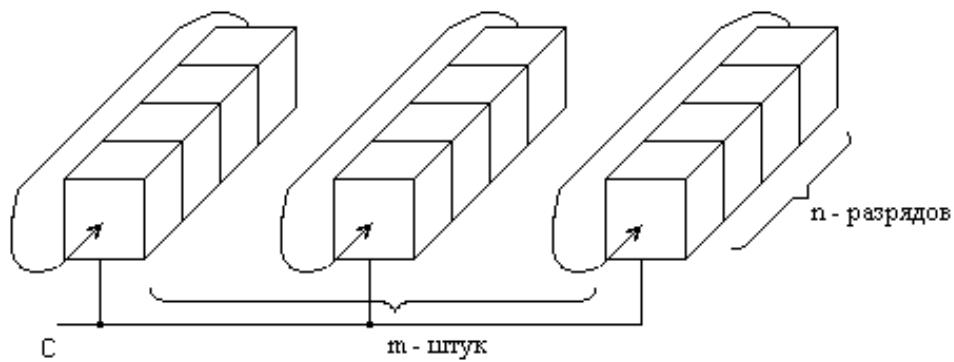


Рисунок 11.20 – Цифровая линия задержки

Для этих целей специально выпускаются регистры. Например, микросхема К144ИР3 – регистр на 64 бит (задержка на 64 такта).

11.3 Исследование работы параллельного регистра на D-триггерах

Схема модели параллельного регистра (registr2.ewb) реализована на четырех D-триггерах (рис. 11.21).

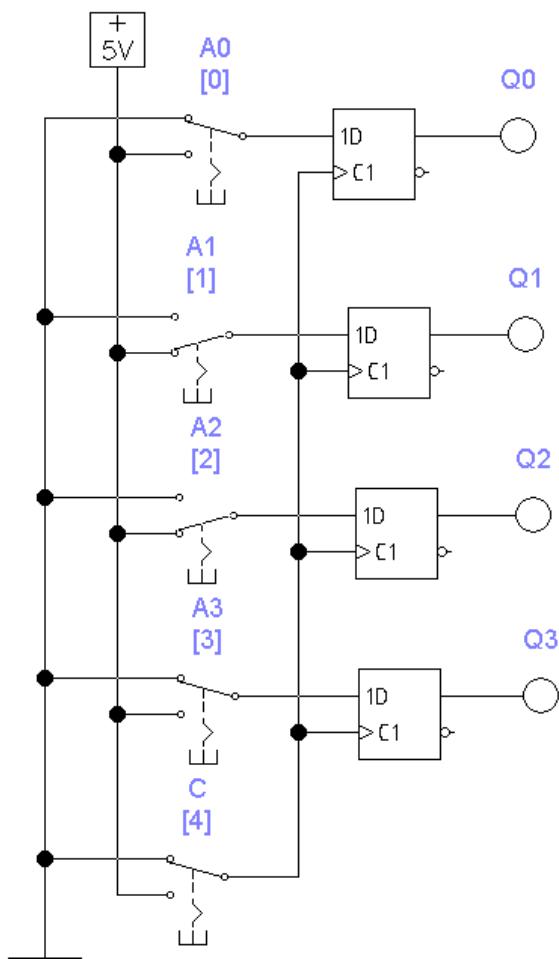
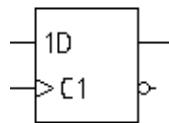


Рисунок 11.21 – Схема модели параллельного регистра (registr2.ewb)

Схема содержит следующие элементы:

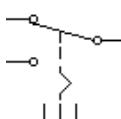
- идеальный источник положительного потенциала 5В;



- четыре D-триггера ;



- четыре логических пробника ;



- пять ручных переключателей , управляемых клавишами 0,...4.

11.4 Порядок выполнения работы

1 . В соответствии с номером бригады выпишите исходное десятичное число из таблицы 1

Таблица 1 – Исходные данные к работе.

Номер бригады	1	2	3	4	5	6	7	8	9	10
Десятич. число	10	9	7	8	6	15	11	12	13	14
Тип триггера	JK	D	T	RS	JK	D	T	RS	D	T
Микро-операции	Передача Λ V	M2 Λ Передача	Уст 1 V M2	Передача M2 Уст 0	Λ M2 Передача	Λ V Хран	V Хран M2	Λ Передача V	V M2 Λ	M2 Передача V

2. Подготовьте такую таблицу

Входы				Выходы			
A0	A1	A2	A3	Q0	Q1	Q2	Q3

Откройте файл registr2.ewb. Используя клавиши 0,1,2,3 ключей A0, A1, A2, A3 наберите на входе регистра исходное число в двоичном коде.

3. Подайте синхросигнал клавишей “4” для перевода триггеров в другое состояние. Результаты занесите в таблицу.

4. Из таблицы 1 выпишите тип триггера и перечень микроопераций согласно номеру бригады. Синтезируйте один разряд регистра хранения для выполнения требуемых микроопераций между входным сигналом и

содержимым триггера, учитывая словари переходов заданных триггеров (таблица 2) и пример синтеза регистра хранения, рассмотренный выше.

Таблица 2 - Словари переходов триггеров

Q(t)	Сигналы на входах триггера для перевода его в нужное состояние						Q(t+1)	
	JK-триггер		RS-триггер		T-триггер	D-триггер		
	J	K	R	S				
0	0	-	-	0	0	0	0	
0	1	-	0	1	1	1	1	
1	-	1	1	0	1	0	0	
1	-	0	0	-	0	1	1	

Прочерк означает безразличное состояние входа

Все этапы синтеза и преобразований приведите в отчете.

2 Откройте файл registr2.ewb . Схема приведена на рис.11.22

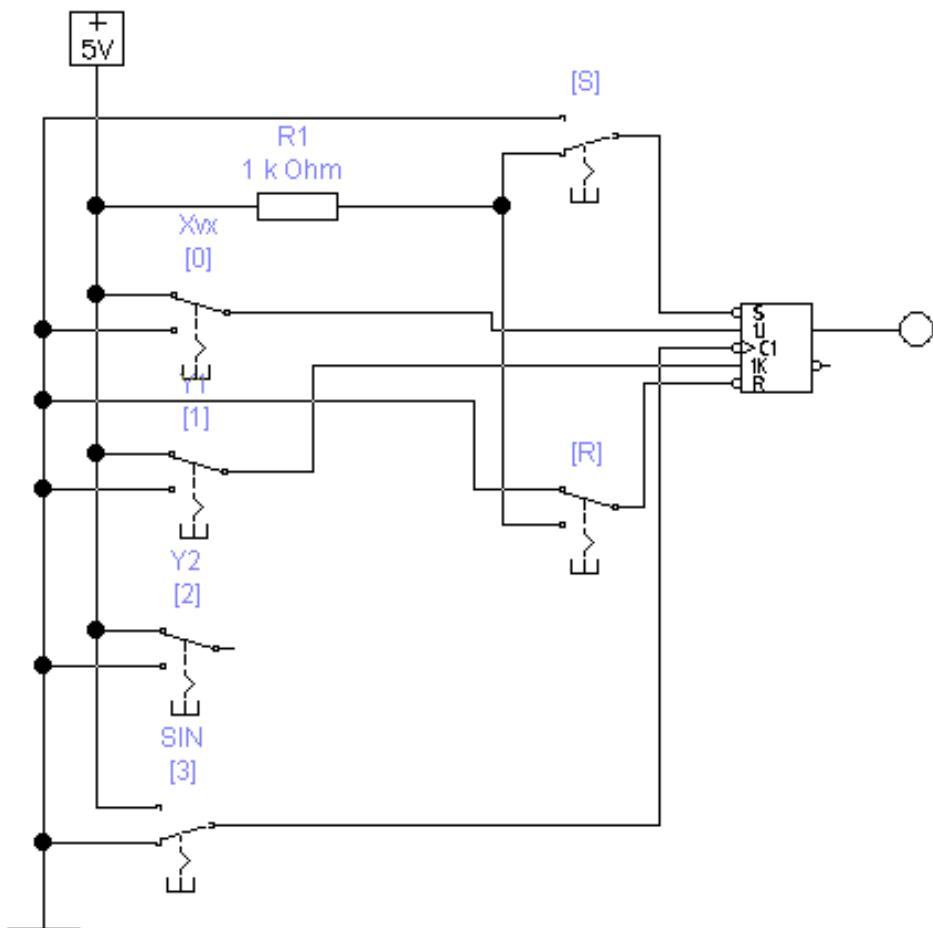


Рисунок 11.22 – Схема для анализа одного разряда регистра (registr2.ewb)

В этой схеме ключи S и R служат для начальной установки триггера в исходное состояние и используются при проверке заданной таблицы истинности. Внесите изменения в схему рис. 11.22 для получения своего варианта. Помните, что T-триггер получается из JK- триггера при объединении входов J и K. D- триггер также следует построить на JK триггере путем подключения входа K через инвертор ко входу J - получается общий D вход. Разряд регистра, синтезированный на RS-триггере при реализации выполняется также на JK- триггере без каких – либо изменений.

Путем перебора входного (X_{vх}) и управляющих (Y₁, Y₂) сигналов проверьте все наборы исходной ТИ.

11.5 Результаты работы

Подготовьте отчет по лабораторной работе, который должен содержать исходные данные, все логические преобразования, схемы моделей, таблицы измерений.

11.6 Контрольные вопросы

1. Какие типы регистров вы знаете?
2. Объясните принцип действия регистра хранения.
3. Какое количество информации хранится в десятиразрядном регистре?
4. Чем отличается парафазный регистр от однофазного?
5. Чем отличается параллельный регистр от последовательного?

12 Литература

- 1 Смирнов Ю.А. Основы микроэлектроники и микропроцессорной техники / Ю.А. Смирнов, С.В. Соколов, Е.В. Титов. – СПб.: Лань, 2013. 496 с.
- 2 Потехин В. А.. Схемотехника цифровых устройств: учебное пособие для вузов.- Томск: В-Спектр, 2012. - 250 с.
- 3 Сажнев, А. М. Цифровые устройства и микропроцессоры: учебное пособие для вузов/ А. М. Сажнев. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2023. — 139 с.
- 4 Булдаков А. Н. Цифровые устройства и микропроцессоры : учебное пособие / А. Н. Булдаков. – Томск : ТУСУР, 2022. – 218 с

Лариса Геннадьевна Рогулина
Александр Михайлович Сажнёв

Электротехника, электроника и схемотехника

Часть 2. Цифровые устройства

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Редактор:

Корректор:

Подписано в печать 24.06.2024,
формат бумаги 60x84/16, отпечатано на ризографе, шрифт №10,
изд.л. 13,4 , заказ № 58 , тираж 50. СибГУТИ
630102, Новосибирск, ул. Кирова, 86