МИНИСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ФЕРГАНСКИЙ ФИЛИАЛ ТАШКЕНТСКОГО УНИВЕРСИТЕТА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ИМЕНИ МУХАММАДА АЛ-ХОРАЗМИЙ

Факультет «Телекоммуникационные технологии и профессиональное образование»

Кафедра «Телекоммуникационный инжиниринг»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

К лабораторным работам

по предмету

“ЦИФРОВАЯ СХЕМОТЕХНИКА”

Фергана - 2018 г

Методическая указания написана на основе типовой учебной программы по предмету «Цифровая схемотехника», утвержденного приказом Министерство высшего и средне-специального образования Республики Узбекистан № ____ от «____» _____ 201___ г..

Данная рабочая учебная программа утверждена на Совете Ферганского филиала ТУИТ (протокол №____ от “___” _____ 201___ год).

Составитела:
Мамасодикова Н.Ю. — **ассистент кафедры «Телекоммуникационный инжиниринг» ФФ ТУИТ**

Рецензенты:
Мамасодиков Ю. — **доцент кафедры «Электроника и приборостроения» ФерПИ**

Кулдашев О.Х. — **доцент кафедры «Телекоммуникационный инжиниринг» ФФ ТУИТ**

Методические указанияобсуждена и одобрена на заседании кафедры «Телекоммуникационный инжиниринг» (протокол № ___ от « __» _____ 201___ г.) и рекомендована к рассмотрению на Учебно-Методическом Совете факультета «Телекоммуникационные технологии и профессиональное образования».

Зав. кафедрой: _____ (подпись) **Жураев Н.М.**

Методические указанияобсуждена и утверждена на заседании Учебно-Методического Совета факультета «Телекоммуникационные технологии и профессиональное образования» (протокол № ___ от «___» _____ 201___ г.)

Председатель Учебно-Методического Совета факультета : _____ (подпись) **Кулдашев О.Х**

СОГЛАСОВАНО:
Начальник учебно-методического отдела: _____ (подпись) **Умаров Ш.А.**

Лабораторная работа № 1.

Логические элементы И.

Цель работы: Изучить и научиться работать логическим элементом И.

Следующим простейшим логическим элементом является схема, реализующая операцию логического умножения "И":

F(x1,x2) = x1^x2

где символ ^ и обозначает функцию логического умножения. Иногда эта же функция записывается в другом виде:

F(x1,x2) = x1^x2 = x1·x2 = x1&x2.

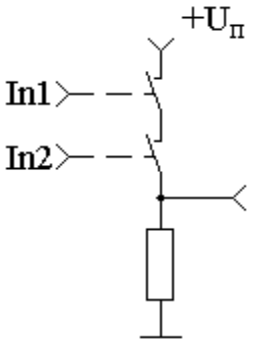
То же самое действие можно записать при помощи таблицы истинности, приведённой в таблице 2. В формуле, приведенной выше использовано два аргумента. Поэтому элемент, выполняющий эту функцию имеет два входа. Такой элемент обозначается "И". Для элемента "И" таблица истинности будет состоять из четырех строк (22 = 4).

Таблица истинности схемы, выполняющей логическую функцию "И"

In1	In2	Out
0	0	0
0	1	0
1	0	0
1	1	1

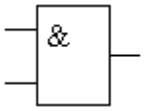
Как видно из приведённой таблицы истинности активный сигнал на выходе этого логического элемента появляется только тогда, когда и на входе X и на входе Y будут присутствовать логические единицы. То есть этот логический элемент действительно реализует операцию "И"

Проще всего понять, как работает логический элемент "И", при помощи схемы, построенной на идеализированных ключах с электронным управлением, как это показано на рисунке 2. В этой схеме ток будет протекать только тогда, когда оба ключа будут замкнуты, а значит, единичный уровень на выходе схемы появится только при двух логических единицах на входе.



Принципиальнае схема логического элемента "2И".

Условно-графическое изображение схемы, выполняющей логическую функцию "2И", на принципиальных схемах приведено на рисунке 3, и с этого момента схемы, выполняющие функцию “И” будут приводиться именно в таком виде. Это изображение не зависит от конкретной принципиальной схемы устройства, реализующей функцию логического умножения.



Условно-графическое изображение логического элемента "И".

Точно так же описывается и функция логического умножения трёх переменных:

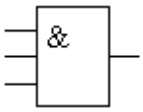
F(x1,x2,x3)=x1^x2^x3

Её таблица истинности будет содержать уже восемь строк (23 = 4). Таблица истинности трёхвходовой схемы логического умножения "И" приведена в таблице 3, а условно-графическое изображение на рисунке 4. В схеме же логического элемента "И", построенной по принципу схемы, приведённой на рисунке 2, придётся добавить третий ключ.

Таблица 3. Таблица истинности схемы, выполняющей логическую функцию "И"

In1	In2	In3	Out
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

Получить подобную таблицу истинности можно при помощи схемы исследования логического элемента "И", подобной схеме исследования логического инвертора.



Условно-графическое обозначение схемы, выполняющей логическую функцию "И".

Лабораторная работа № 2.

Логические элементы ИЛИ.

Цель работы: Изучить и научиться работать логическим элементом ИЛИ.

Следующим простейшим логическим элементом является схема, реализующая операцию логического сложения "ИЛИ":

$$F(x1,x2) = x1Vx2$$

где символ V обозначает функцию логического сложения. Иногда эта же функция записывается в другом виде:

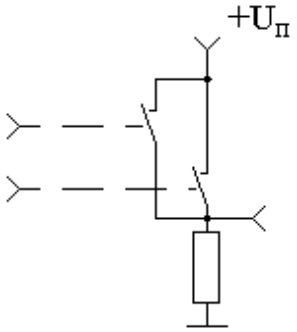
$$F(x1,x2) = x1Vx2 = x1+x2 = x1|x2.$$

То же самое действие можно записать при помощи таблицы истинности, приведённой в таблице 4. В формуле, приведенной выше использовано два аргумента. Поэтому логический элемент, выполняющий эту функцию имеет два входа. Такой элемент обозначается "ИЛИ". Для элемента "ИЛИ" таблица истинности будет состоять из четырех строк (22 = 4).

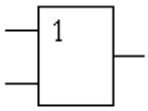
Таблица истинности логического элемента "ИЛИ".

In1	In2	Out
0	0	0
0	1	1
1	0	1
1	1	1

Как и в случае, рассмотренном для схемы логического умножения, воспользуемся для реализации схемы "2ИЛИ" ключами. На этот раз соединим ключи параллельно. Схема, реализующая таблицу истинности 4, приведена на рисунке 5. Как видно из приведённой схемы, уровень логической единицы появится на её выходе, как только будет замкнут любой из ключей, то есть схема реализует таблицу истинности.



Принципиальная схема логического элемента "ИЛИ".
Так как функция логического суммирования может быть реализована различными принципиальными схемами, то для обозначения этой функции на принципиальных схемах используется специальный символ "1".



Условно-графическое изображение логического элемента, выполняющего функцию "ИЛИ"

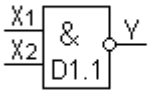
Лабораторная работа № 3.

Логические элементы И-НЕ.

Цель работы: Изучить и научиться работать логическим элементом И-НЕ.

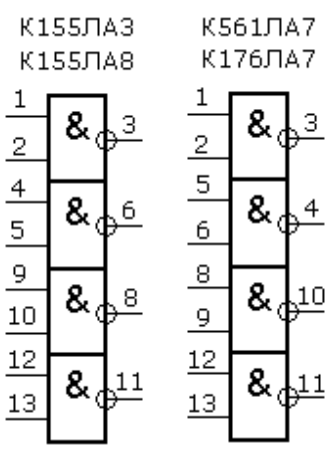
«И-НЕ» (NAND) – функция сложения с отрицанием (если на всех входах единица, то на выходе будет ноль, в противном случае на выходе всегда будет единица). Графическое обозначение элемента «И-НЕ» и его таблица истинности приведены слева.

По таблице истинности следует, что на выходе элемента «И-НЕ» будет логический ноль только в том случае, если на обоих входах будет логическая единица. Если хотя бы на одном входе ноль, то на выходе будет единица.

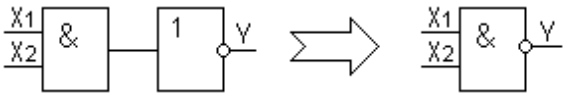


X1	X2	Y
0	0	1
1	0	1
0	1	1
1	1	0

Самой распространённой микросхемой ТТЛ, выполняющей функцию «2И-НЕ», является ИМС К155ЛА3, а микросхемами КМОП (комплементарный металлооксидный полупроводник) – ИМС К561ЛА7 и К176ЛА7, внутри которых имеется четыре элемента «2И-НЕ». Нумерация выводов этих микросхем показана снизу.

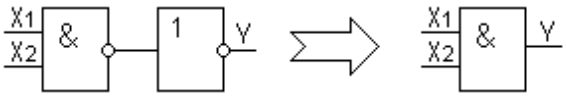


Сравнив таблицы истинности элемента «И-НЕ» и элемента «И» можно догадаться об эквивалентности схем:



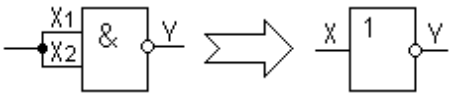
Добавив к элементу «И» элемент «НЕ» мы получили элемент «И-НЕ». Так можно собрать схему, если нам необходим элемент «И-НЕ», а у нас в распоряжении имеются только элементы «И» и «НЕ».

И наоборот:



Добавив к элементу «И-НЕ» элемент «НЕ» мы получили элемент «И». Так можно собрать схему, если нам необходим элемент «И», а у нас в распоряжении имеются только элементы «И-НЕ» и «НЕ».

Аналогичным образом, путём соединения входов элемента «И-НЕ» мы можем получить элемент «НЕ»:



Обратите внимание, что было введено новое в обозначении элементов – дефис, разделяющий правую и левую часть в названии «И-НЕ». Этот дефис непременный атрибут при инверсии на выходе (функции «НЕ»).

Лабораторная работа № 4.

Логические элементы ИЛИ-НЕ.

Цель работы: Изучить и научиться работать логическим элементом ИЛИ-НЕ.

«ИЛИ-НЕ» (NOR) – функция выбора (если хотя бы на одном из входов – единица, то на выходе – ноль, в противном случае на выходе всегда будет единица). Как вы поняли, элемент «ИЛИ-НЕ» выполняет функцию «ИЛИ», а потом инвертирует его функцией «НЕ».

Графическое обозначение элемента «ИЛИ-НЕ» и его таблица истинности приведена снизу.

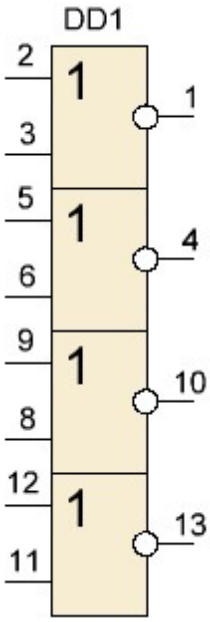
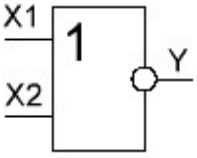


Таблица истинности для логического элемента ИЛИ-НЕ.

Вход X1	Вход X2	Выход Y
0	0	1
1	0	0
0	1	0
1	1	0

Изображение на схеме.

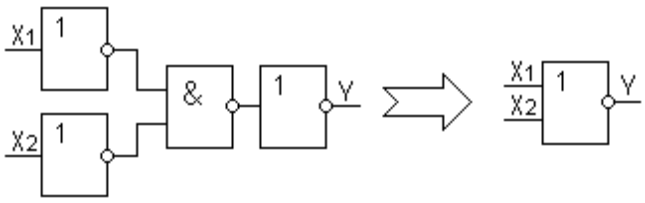


На зарубежный лад изображается так. Называют как NOR.

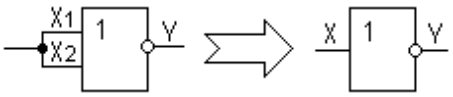
Мы имеем только один высокий потенциал на выходе, обусловленный подачей на оба входа одновременно низкого потенциала. Здесь, как и на любых других принципиальных схемах, кружочек на выходе подразумевает инвертирование сигнала. Так как схемы И – НЕ и ИЛИ – НЕ встречаются очень часто, то для каждой функции имеется своё условное обозначение. Функция И – НЕ обозначается значком "&", а функция ИЛИ – НЕ значком "1".

Для отдельного инвертора таблица истинности уже приведена выше. Можно добавить, что количество инверторов в одном корпусе может достигать шести.

Предположим, что нам в схеме необходим элемент, выполняющий функцию «ИЛИ-НЕ», но у нас есть в распоряжении только элементы «НЕ» и «И-НЕ», тогда можно собрать следующую схему, которая будет выполнять функцию «ИЛИ-НЕ»:



По аналогии с элементом «И-НЕ», путём соединения входов элемента «ИЛИ-НЕ» мы можем получить элемент «НЕ»:



Лабораторная работа №5.

Логические элементы НЕ-ИЛИ.

Цель работы: Изучить и научиться работать логическим элементом НЕ-ИЛИ.

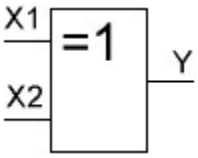
К числу базовых логических элементов принято относить элемент реализующий функцию «НЕ-ИЛИ». Иначе эта функция называется «неравнозначность».

Высокий потенциал на выходе возникает только в том случае, если входные сигналы не равны. То есть на одном из входов должна быть единица, а на другом ноль. Если на выходе логического элемента имеется инвертор, то функция выполняется противоположная – «равнозначность». Высокий потенциал на выходе будет появляться при одинаковых сигналах на обоих входах.

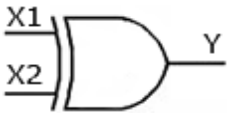
Таблица истинности.

Вход X1	Вход X2	Выход Y
0	**0**	**0**
1	**0**	**1**
0	**1**	**1**
1	**1**	**0**

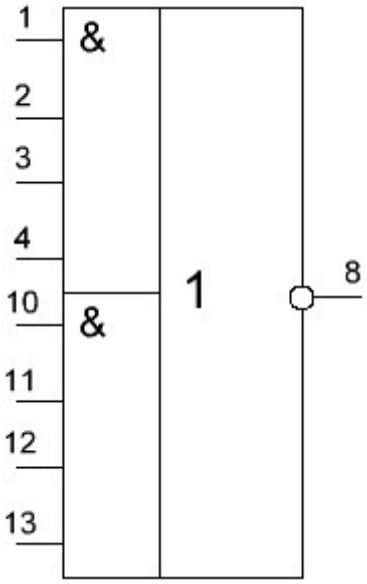
Эти логические элементы находят своё применение в сумматорах. «НЕ-ИЛИ» изображается на схемах знаком равенства перед единицей "=1".



На зарубежный манер "исключающее ИЛИ" называют XOR и на схемах рисуют вот так.



Кроме вышеперечисленных логических элементов, которые выполняют базовые логические функции очень часто, используются элементы, объединённые в различных сочетаниях. Вот, например, К555ЛР4. Она называется очень серьёзно 2-4И-2ИЛИ-НЕ.



Её таблица истинности не приводится, так как микросхема не является базовым логическим элементом. Такие микросхемы выполняют специальные функции и бывают намного сложнее, чем приведённый пример. Так же в логический базис входят и простые элементы "И" и "ИЛИ". Но они используются гораздо реже. Может возникнуть вопрос, почему эта логика называется транзисторно-транзисторной.

Если посмотреть в справочной литературе схему, допустим, элемента И – НЕ из микросхемы К155ЛА3, то там можно увидеть несколько транзисторов и резисторов. На самом деле ни резисторов, ни диодов в этих микросхемах нет. На кристалл кремния через трафарет напыляются только транзисторы, а функции резисторов и диодов выполняют эмиттерные переходы транзисторов. Кроме того в ТТЛ логике широко используются многоэмиттерные транзисторы. Например, на входе элемента И стоит четырёхэмиттерный транзистор.

Лабораторная работа № 6.

Синтез схем на основе Булевых выражений.

Цель работы: Изучить синтез схем и научиться работать на основе БУлевых выражений.

Любая логическая схема без памяти полностью описывается таблицей истинности. Эта таблица является исходной информацией для синтеза схемы на основе логических элементов «И», «ИЛИ», «НЕ». Для разработки требуемого цифрового устройства сначала на основе таблицы истинности записывают его логическое выражение. Затем с целью упрощения цифрового устройства минимизируют его логическое выражение и далее разрабатывают схему, реализующую полученное логическое выражение. Логические выражения можно получить двумя способами:

на основе совершенной дизъюнктивной нормальной формы (СДНФ);
на основе совершенной конъюнктивной нормальной формы (СКНФ).
Совершенная дизъюнктивная нормальная форма (СДНФ)
Функция представляется суммой групп. Каждая группа состоит из произведения, в которую входят все переменные.
Например:

$f(x1,x2,x3) = x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3$

Совершенная конъюнктивная нормальная форма (СКНФ)
Функция представляется произведением групп. Каждая группа состоит из суммы, в которую входят все переменные.
Например:

$f(x1,x2,x3) = (x1+x2+x3) \cdot (x1+x2+x3) \cdot (x1+x2+x3)$

Если схема имеет несколько выходов, то каждый выход описывается своей функцией. Такая система функций называется системой собственных функций. СДНФ составляется на основе таблицы истинности по следующему правилу: для каждого набора переменных, при котором функция равна 1, записывается произведение, в котором с отрицанием берутся переменные, имеющие значение «0».

Пример:

Таблица– Заданная таблица истинности.

x1	**x2**	**x3**	**y**
0	0	0	0
0	**0**	**1**	**1**
0	1	0	0
0	**1**	**1**	**1**
1	0	0	0
1	0	1	0
1	**1**	**0**	**1**
1	**1**	**1**	**1**

СДНФ:

$y = f(x1,x2,x3) = x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3$

СКНФ составляется на основе таблицы истинности по правилу: для каждого набора переменных, при котором функция равна 0, записывается сумма, в которой с отрицанием берутся переменные, имеющие значение 1.

Таблица– Заданная таблица истинности.

x1	**x2**	**x3**	**y**
0	**0**	**0**	**0**
0	0	1	1
0	**1**	**0**	**0**

0	1	1	1
1	**0**	**0**	**0**
1	**0**	**1**	**0**
1	1	0	1
1	1	1	1

СКНФ:

$y = f(x1,x2,x3) = (x1+x2+x3)\cdot(x1+\bar{x2}+x3)\cdot(\bar{x1}+x2+x3)\cdot(\bar{x1}+\bar{x2}+\bar{x3})$

На основе полученных выражений можно составить схему устройства, реализующего заданную функцию.

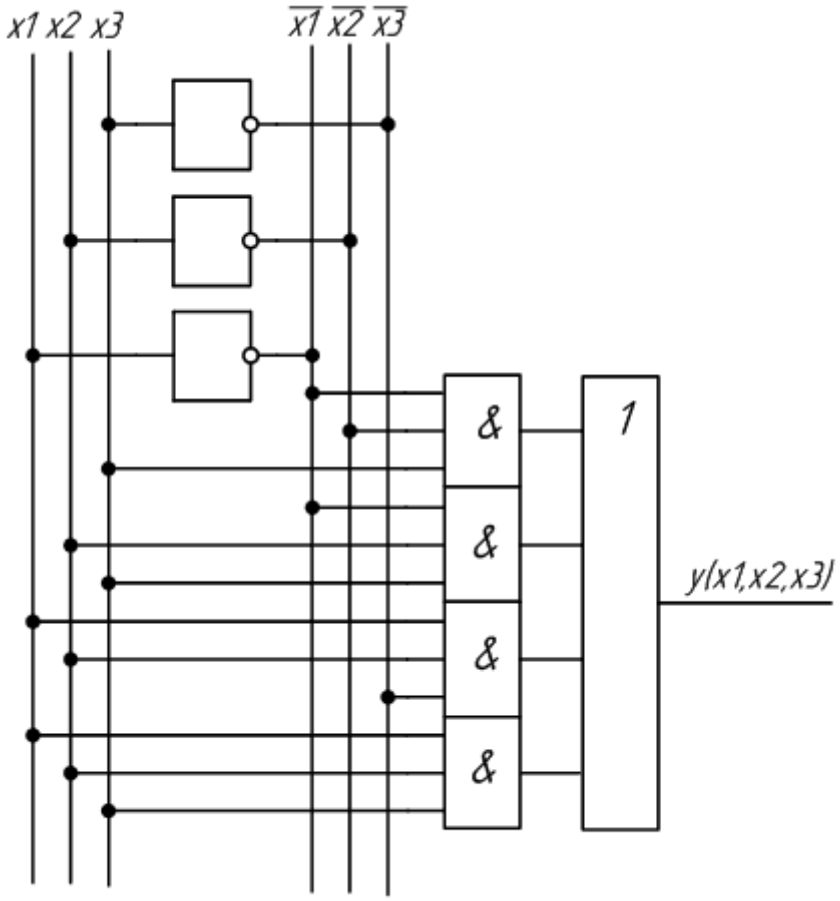


Схема устройства, полученная на основе СДНФ

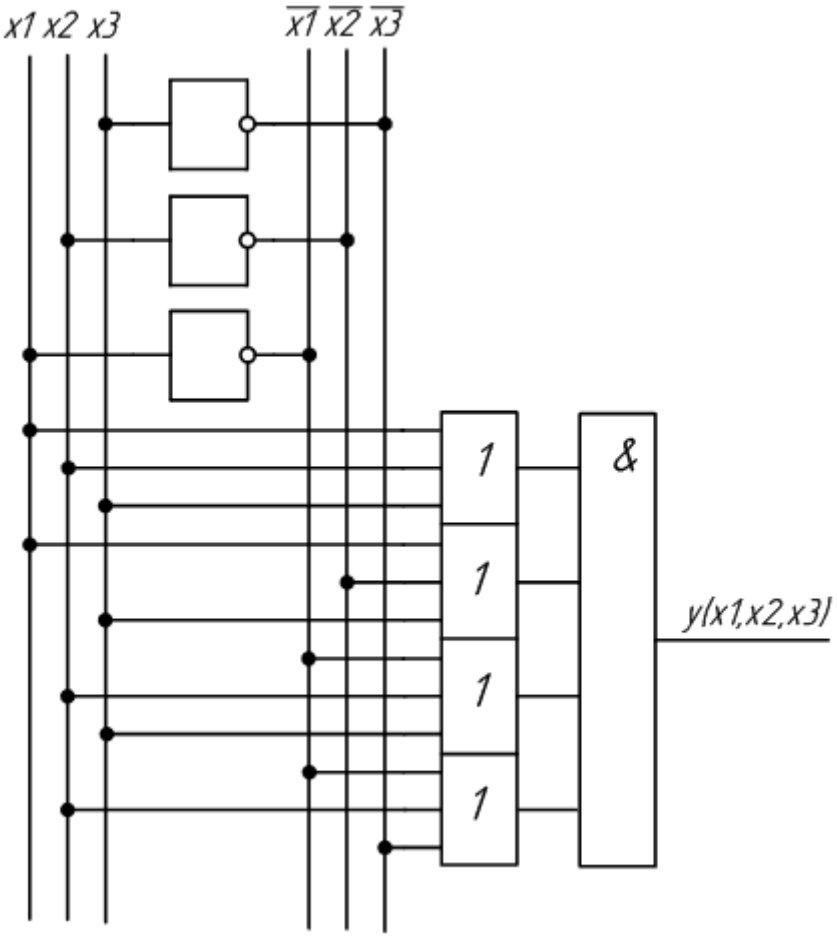


Схема устройства, полученная на основе СКНФ.

С целью упрощения цифрового устройства применяют минимизацию функций. Используя законы алгебры логики, можно упростить исходную функцию.

$y(x1,x2,x3) = x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3 + x1 \cdot x2 \cdot x3 =$

$= x1 \cdot x3 \cdot (x2+x2) + x1 \cdot x2 \cdot (x3+x3) = x1 \cdot x3 + x1 \cdot x2$

На основе полученного выражения составим новую схему устройства.

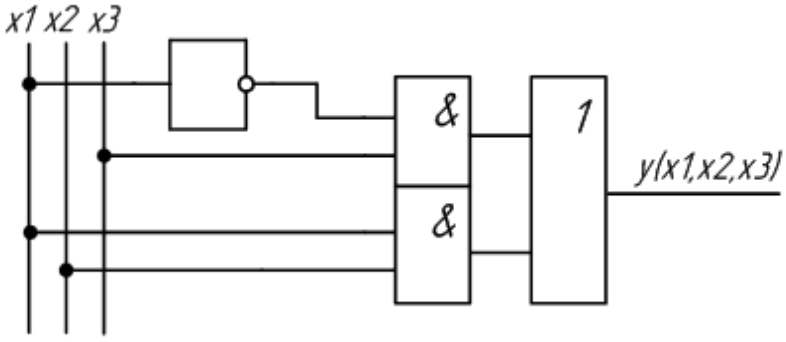


Схема устройства, полученная после минимизации логической функции

Лабораторная работа № 7.

Минимизация функций логической алгебры. Метод Квайна.

Цель работы: Изучить минимизацию функций логческой алгебры и научиться работать методом Квайна.

Метод Квайна — способ представления функции в ДНФ или КНФ с минимальным количеством членов и минимальным набором переменных.
Преобразование функции можно разделить на два этапа:

на первом этапе осуществляется переход от канонической формы (СДНФ или СКНФ) к так называемой сокращённой форме;
на втором этапе — переход от сокращённой формы к минимальной форме.

Первый этап (получение сокращённой формы).

Представим, что заданная функция fпредставлена в СДНФ. Для осуществления первого этапа преобразование проходит два действия:

1. Операция склеивания;
2. Операция поглощения.

Операция склеивания сводится к нахождению пар членов, соответствующих виду $w \cdot x$или $w \cdot \overline{x}$, и преобразованию их в следующие выражения:$w \cdot x \vee w \cdot \overline{x} = w \cdot (x \vee \overline{x}) = w$. Результаты склеивания *w* теперь играют роль дополнительных членов. Необходимо найти все возможные пары членов (каждый член с каждым).

Потом выполняется операция поглощения. Она основана на равенстве $w \vee w \cdot z = w \cdot (1 \vee z) = w$(член *w*поглощает выражение $\{w \cdot z$). Вследствие этого действия из логического выражения вычёркиваются все члены, поглощаемые другими переменными, результаты которых получены в операции склеивания.
Обе операции первого этапа могут выполняться до тех пор, пока это может быть осуществимо.

Применение этих операций продемонстрировано в таблице:

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

СДНФ выглядит так:

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

Результат операции склеивания нужен для преобразования функции на втором этапе (поглощения)

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

Членамирезультатасклеиванияявляются

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

Член $x_2 \cdot \overline{x_3}$ поглощает те члены исходного выражения, которые содержат $x_2 \cdot \overline{x_3}$, то есть первый и четвёртый. Эти члены вычёркиваются. Член $x_1 \cdot \overline{x_2}$ поглощает второй и третий, а член $x_1 \cdot x_3$ — пятый член исходного выражения.

Повторение обеих операций приводит к следующему выражению:

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

Здесь склеивается пара членов $x_1 \cdot \overline{x_2}$ и $x_1 \cdot x_2$ (склеивание пары членов $x_1 \cdot \overline{x_3}$ и $x_1 \cdot x_3$ приводит к тому же результату), результат склеивания*x1* поглощает 2-, 3-, 4-, 5-й члены выражения. Дальнейшее проведение операций склеивания и поглощения оказывается невозможным, сокращённая форма выражения заданной функции (в данном случае она совпадает с минимальной формой)

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

Члены сокращённой формы (в нашем случае это $x_2 \cdot \overline{x_3}$ и x_1*называются простыми импликантами* функции. В итоге, мы получили наиболее простое выражение, если сравнивать его с начальной версией — СДНФ. Структурная схема такого элемента показана на рисунке справа.

Второй этап (табличный) (получение минимальной формы).

Как и на первом этапе, в полученном равенстве могут содержаться члены, устранение которых никаким образом не повлияет на конечный результат. Следующий этап минимизации — удаление таких переменных. Таблица, представленная ниже, содержит значения истинности функции. По ней будет собрана следующая СДНФ.

x_1	x_2	x_3	x_4	$f(x_1, x_2, x_3, x_4)$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1

0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

СДНФ, собранная по этой таблице выглядит следующим образом:

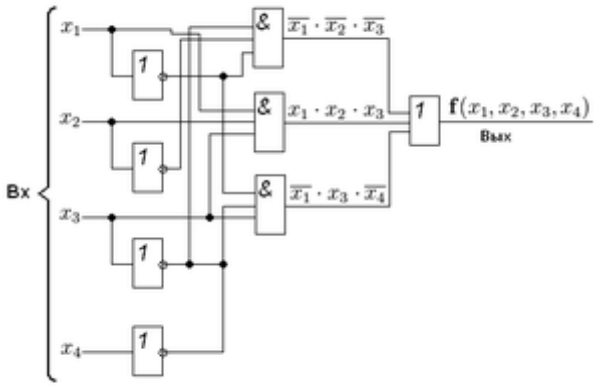
$$f(x_1, x_2, x_3, x_4) = \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4} \vee \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4 \vee \overline{x_1} \cdot \overline{x_2} \cdot x_3 \cdot \overline{x_4} \vee \overline{x_1} \cdot x_2 \cdot x_3 \cdot \overline{x_4}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

Конечное выражение достигается за счёт повторного использования операций склеивания и поглощения:

$$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \vee \overline{x_1} \cdot \overline{x_2} \cdot \overline{x_4} \vee \overline{x_1} \cdot x_3 \cdot \overline{x_4} \vee x_2 \cdot x_3 \cdot \overline{x_4} \vee x_1 \cdot x_2 \cdot x_3$$

Члены этого выражения являются простыми импликантами выражения. Переход от сокращённой формы к минимальной осуществляется с помощью импликантной матрицы.

Члены СДНФ заданной функции вписываются в столбцы, а в строки — простые импликанты, то есть члены сокращённой формы. Отмечаются столбцы членов СДНФ, которые поглощаются отдельными простыми импликантами. В следующей таблице простаяимпликанта$\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3}$ поглощает члены $\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot \overline{x_4}$ и $\overline{x_1} \cdot \overline{x_2} \cdot \overline{x_3} \cdot x_4$ (в первом и во втором столбцах поставлены крестики).



Структурная схема, соответствующая выражению в МДНФ (второй этап) при минимизации функции методом Квайна

Использование метода для получения минимальной КНФ.

Для получения Минимальной конъюнктивной нормальной формы (МКНФ), используя метод Куайна, вводятся следующие критерии:

для минимизации берётся не СДНФ, а СКНФ функции;
склеиваемые пары членов меняются на: $w \vee x$ или $w \vee \overline{x}$;
правило операции поглощения выглядит следующим образом:

$$z \cdot (z \vee y) = z \vee z \cdot y = z \cdot (1 \vee y) = z$$

Лабораторная работа № 8.

Минимизация функций логической алгебры с помощью карты Карно.

Цель работы: Изучить минимизацию функций логической алгебры и научиться работать с помощью карты Карно.

Карта Карно — графический способ минимизации переключательных (булевых) функций, обеспечивающий относительную простоту работы с большими выражениями и устранение потенциальных гонок. Представляет собой операции попарного неполного склеивания и элементарного поглощения. Карты Карно рассматриваются как перестроенная соответствующим образом таблица истинности функции. Карты Карно можно рассматривать как определенную плоскую развертку n-мерного булева куба.

Карты Карно были изобретены в 1952 году Эдвардом В. Вейчем и усовершенствованы в 1953 году Морисом Карно, физиком из «Bell Labs», и были призваны помочь упростить цифровые электронные схемы.

В карту Карно булевы переменные передаются из таблицы истинности и упорядочиваются с помощью кода Грея, в котором каждое следующее число отличается от предыдущего только одним разрядом.

Принципы минимизации.

Основным методом минимизации логических функций, представленных в виде СДНФ или СКНФ, является операция попарного неполного склеивания и элементарного поглощения. Операция попарного склеивания осуществляется между двумя термами (членами), содержащими одинаковые переменные, вхождения которых (прямые и инверсные) совпадают для всех переменных, кроме одной. В этом случае все переменные, кроме одной, можно вынести за скобки, а оставшиеся в скобках прямое и инверсное вхождение одной переменной подвергнуть склейке. Например:

$$\overline{X}_1 X_2 X_3 X_4 \vee \overline{X}_1 X_2 \overline{X}_3 X_4 = \overline{X}_1 X_2 X_4 (X_3 \vee \overline{X}_3) = \overline{X}_1 X_2 X_4 \cdot 1 = \overline{X}_1 X_2 X_4.$$

Аналогично для КНФ:

$$(\overline{X}_1 \vee X_2 \vee X_3 \vee X_4)(\overline{X}_1 \vee X_2 \vee \overline{X}_3 \vee X_4) = \overline{X}_1 \vee X_2 \vee X_4 \vee X_3 \overline{X}_3 = \overline{X}_1 \vee X_2 \vee X_4 \vee 0 = \overline{X}_1 \vee X_2 \vee X_4.$$

Возможность поглощения следует из очевидных равенств

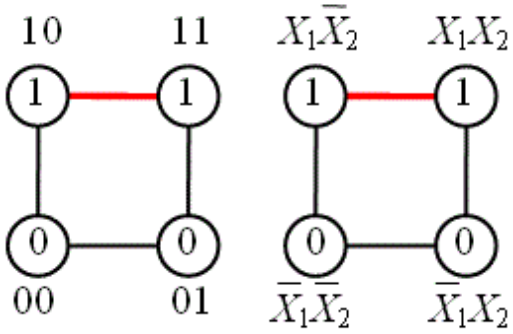
$$A \vee \overline{A} = 1; A\overline{A} = 0.$$

Таким образом, главной задачей при минимизации СДНФ и СКНФ является поиск термов, пригодных к склейке с последующим поглощением, что для больших форм может оказаться достаточно сложной задачей. Карты Карно предоставляют наглядный способ отыскания таких термов.

Как известно, булевы функции N переменных, представленные в виде СДНФ или СКНФ, могут иметь в своём составе 2N различных термов. Все эти члены составляют некоторую структуру, топологически эквивалентную N–мерному кубу, причём любые два терма, соединённые ребром, пригодны для склейки и поглощения.

На рисунке изображена простая таблица истинности для функции из двух переменных, соответствующий этой таблице 2-мерный куб (квадрат), а также 2-мерный куб с обозначением членов СДНФ и эквивалентная таблица для группировки термов:

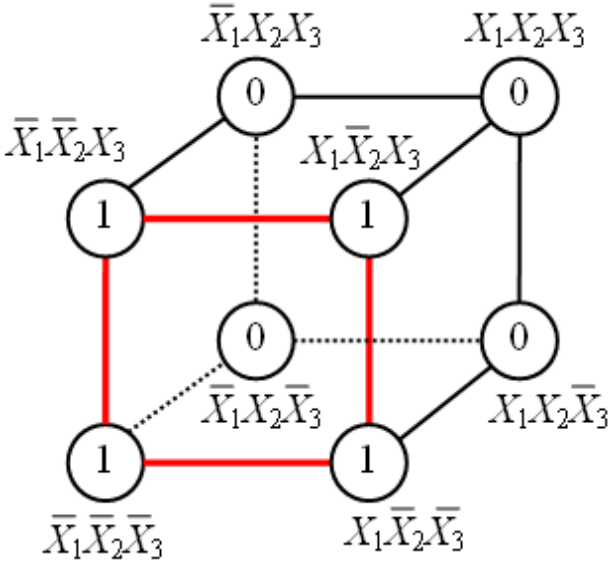
X_1	X_2	$f(X_1, X_2)$
0	0	0
0	1	0
1	0	1
1	1	1



X_1 \ X_2	0	1
1	1	1
0	0	0

В случае функции трёх переменных приходится иметь дело с трёхмерным кубом. Это сложнее и менее наглядно, но технически возможно. На рисунке в качестве примера показана таблица истинности для булевой функции трёх переменных и соответствующий ей куб.

X_1	X_2	X_3	$f(X_1, X_2, X_3)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

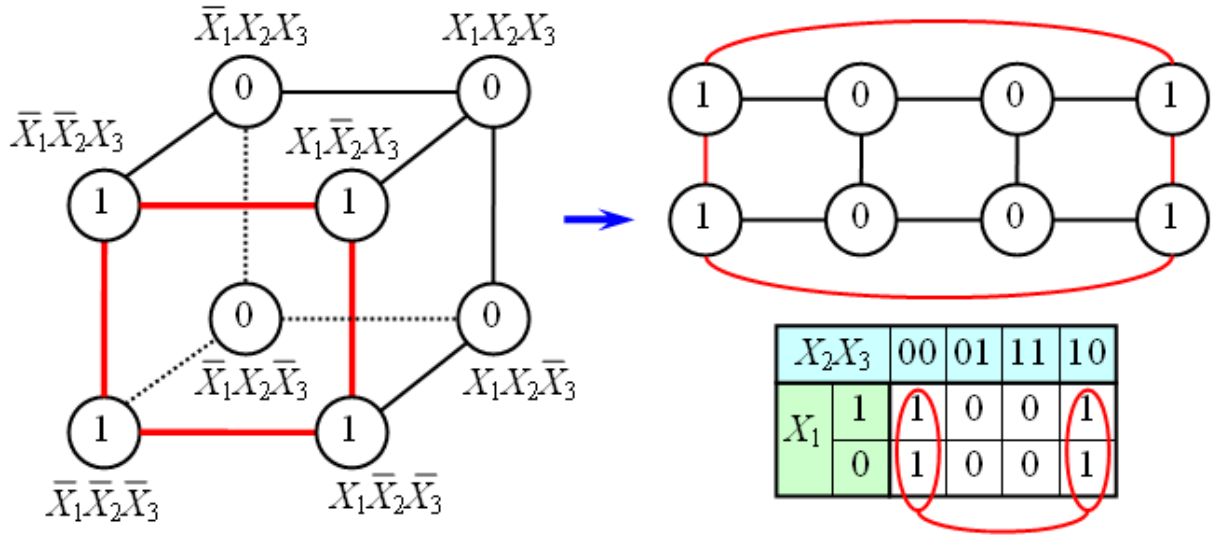


Как видно из рисунка, для трёхмерного случая возможны более сложные конфигурации термов. Например, четыре терма, принадлежащие одной грани куба, объединяются в один терм с поглощением двух переменных:

$$\overline{X}_1\overline{X}_2\overline{X}_3 \vee X_1\overline{X}_2\overline{X}_3 \vee \overline{X}_1\overline{X}_2X_3 \vee X_1\overline{X}_2X_3 =$$
$$= \overline{X}_2(\overline{X}_1\overline{X}_3 \vee \overline{X}_1X_3 \vee X_1\overline{X}_3 \vee X_1X_3) = \overline{X}_2(\overline{X}_1 \vee X_1)(\overline{X}_3 \vee X_3) = \overline{X}_2$$

В общем случае можно сказать, что 2K термов, принадлежащие одной K–мерной грани гиперкуба, склеиваются в один терм, при этом поглощаются K переменных.

Для упрощения работы с булевыми функциями большого числа переменных был предложен следующий удобный приём. Куб, представляющий собой структуру термов, разворачивается на плоскость как показано на рисунке. Таким образом появляется возможность представлять булевы функции с числом переменных больше двух в виде плоской таблицы. При этом следует помнить, что порядок кодов термов в таблице (00 01 11 10) не соответствует порядку следования двоичных чисел, а клетки, находящиеся в крайних столбцах таблицы, соседствуют между собой.



X_2X_3		00	01	11	10
X_1	1	1	0	0	1
	0	1	0	0	1

Аналогичным образом можно работать с функциями большего числа переменных.

Порядок работы с картой Карно.

Исходной информацией для работы с картой Карно является таблица истинности минимизируемой функции. Таблица истинности содержит полную информацию о логической функции, задавая её значения на всех возможных 2N наборах входных переменных X1 ... XN. Карта Карно также содержит 2N клеток, каждая из которых ассоциируется с уникальным набором входных переменных X1 ... XN. Таким образом, между таблицей истинности и картой Карно имеется взаимно однозначное соответствие, и карту Карно можно считать соответствующим образом отформатированной таблицей истинности.

В данном разделе в качестве примера используется функция четырёх переменных, заданная таблицей истинности, изображённой на рисунке а. Карта Карно для той же функции изображена на рисунке б.

а

X_1	X_2	X_3	X_4	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

б

X_1X_2 \ X_3X_4	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	0	1	1	0
10	1	0	0	1

в

X_1X_2 \ X_3X_4	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	0	1	1	0
10	1	0	0	1

г

X_1X_2 \ X_3X_4	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	0	1	1	0
10	1	0	0	1

Принципы склейки.

- Склейку клеток карты Карно можно осуществлять по единицам (если необходимо получить ДНФ) или по нулям (если требуется КНФ).
- Склеивать можно только прямоугольные области с числом единиц (нулей) 2n, где n — целое число, при этом рекомендуется брать максимальное из возможных значений n. В некоторых ситуациях в раскладке образуется единица или ноль, которую невозможно склеить с какой либо областью. В этом случае единица склеивается "сама с собой". Для карт Карно с числом переменных более четырёх могут получаться более сложные области, о чём будет сказано в следующих разделах.
- Область, которая подвергается склейке должна содержать только единицы (нули).
- Крайние клетки каждой горизонтали и каждой вертикали также граничат между собой (топологически карта Карно для четырёх переменных представляет собой тор) и могут объединяться в прямоугольники. Следствием этого правила является смежность всех четырёх угловых ячеек карты Карно для N=4. Если во всех четырёх угловых ячейках стоят единицы (нули) они могут быть объединены в квадрат, как показано на рис. 2в.
- Все единицы (нули) должны попасть в какую-либо область.
- С точки зрения минимальности ДНФ (КНФ) число областей должно быть как можно меньше (каждая область представляет собой терм), а число клеток в области должно быть как можно больше (чем больше клеток в области, тем

меньше переменных содержит терм. Терм размером 2n ячеек содержит N–n переменных).

- Одна ячейка карты Карно может входить сразу в несколько областей. Это следует из очевидного свойства булевых функций: повторение уже существующего слагаемого (сомножителя) не влияет на функцию:

$$A \vee A = A; A \cdot A = A.$$

- В отличие от СДНФ (СКНФ), ДНФ (КНФ) не единственны. Возможно несколько эквивалентных друг другу ДНФ (КНФ), которые соответствуют разным способам покрытия карты Карно прямоугольными областями.

Примеры.

У мальчика Коли есть мама, папа, дедушка и бабушка. Коля пойдёт гулять на улицу, тогда и только тогда, когда ему разрешат хотя бы двое родственников.
Для краткости обозначим родственников Коли через буквы:
мама — x1
папа — x2
дедушка — x3
бабушка — x4
Условимся обозначать согласие родственников единицей, несогласие - нулём.
Возможность пойти погулять обозначим буквой f, Коля идёт гулять — f = 1, Коля гулять не идёт — f = 0.
Составим таблицу истинности:

X1	X2	X3	X4	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Перерисуем таблицу истинности в 2-х мерный вид:

X1 X2 \ X3 X4	00	01	10	11
00				
01				
10				
11				

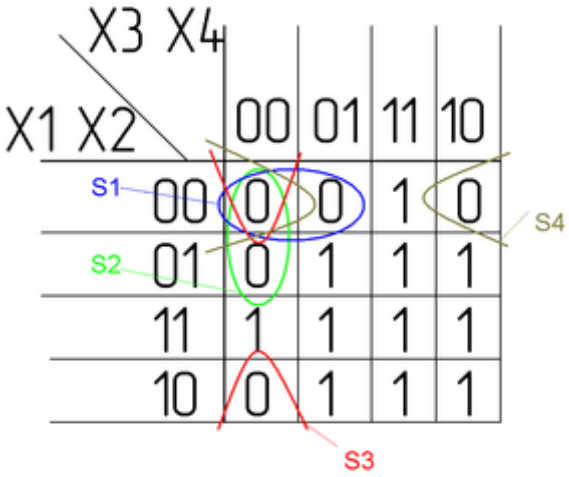
Переставим в ней строки и столбцы в соответствии с кодом Грея(последний и предпоследний столбец меняют местами). Получили Карту Карно:

X1 X2 \ X3 X4	00	01	11	10
00				
01				
11				
10				

Заполним её значениями из таблицы истинности(первая строка не соответствует таблице истинности так как f=0 и разрешения на гулять нет):

X1 X2 \ X3 X4	00	01	11	10
00	0	0	1	0
01	0	1	1	1
11	1	1	1	1
10	0	1	1	1

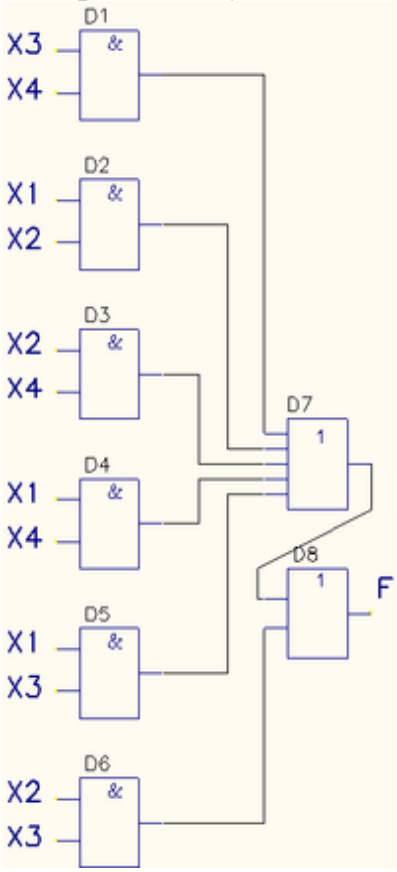
Минимизируем в соответствии с правилами:



1. Все области содержат 2^n клеток;
2. Так как Карта Карно на четыре переменные, оси располагаются на границах Карты и их не видно (подробнее смотри пример Карты на 5 переменных);
3. Так как Карта Карно на четыре переменные, все области симметрично осей — смежные между собой (подробнее смотри пример Карты на 5 переменных);
4. Области S3, S4, S5, S6 максимально большие;
5. Все области пересекаются (необязательное условие);
6. В данном случае рациональный вариант только один.

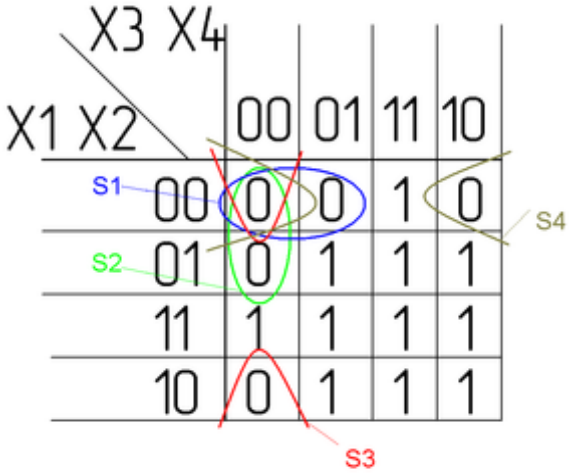
$$f(X1, X2, X3, X4) = S1 \vee S2 \vee S3 \vee S4 \vee S5 \vee S6 =$$
$$= X3X4 \vee X1X2 \vee X2X4 \vee X1X4 \vee X1X3 \vee X2X3$$

Теперь по полученной минимальной ДНФ можно построить логическую схему:

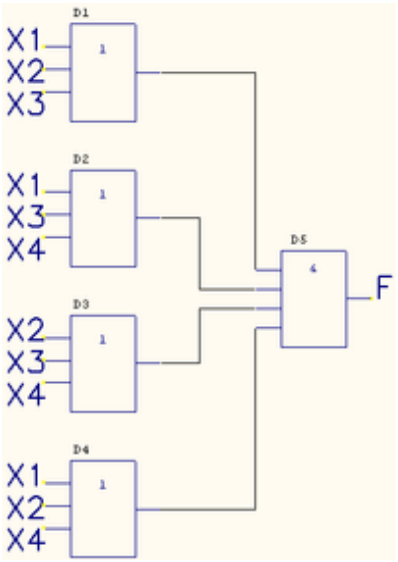


Из-за отсутствия в наличии шести-входового элемента ИЛИ, реализующего функцию дизъюнкции, пришлось каскадировать пяти- и двух-входовые элементы(D7, D8).

Составим мин. КНФ:



$$f(X1, X2, X3, X4) = (S1)\ (S2)\ (S3)\ (S4) =$$
$$= (X1 \vee X2 \vee X3)(X1 \vee X3 \vee X4)(X2 \vee X3 \vee X4)(X1 \vee X2 \vee X4)$$

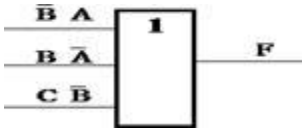


Лабораторная работа № 9.

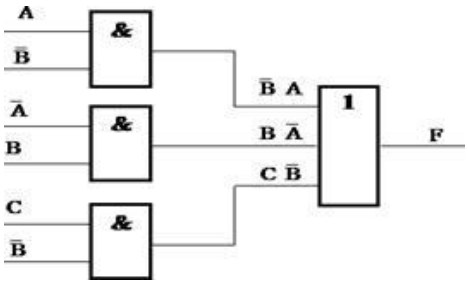
Построение комбинационных схем используя логических элементов.

Цель работы: Изучить и научиться построению комбинационных схем, используя логических элементов.

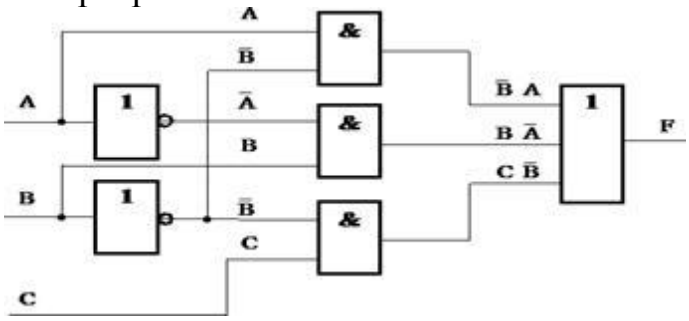
Комбинационные схемы - это схемы, у которых выходные сигналы Y = (y1, y2, ..., ym) в любой момент дискретного времени однозначно определяются совокупностью входных сигналов X = (x1, x2,..., xn), поступающих в тот же момент времени t. Реализуемый в КС способ обработки информации называется комбинационным потому, что результат обработки зависит только от комбинации входных сигналов и формируется сразу при поступлении входных сигналов. Поэтому одним из достоинств комбинационных схем является их высокое быстродействие. Преобразование информации однозначно описывается логическими функциями вида Y=f(X). Логические функции и соответствующие им комбинационные схемы подразделяют на регулярные и нерегулярные структуры. Регулярные структуры предполагают построение схемы таким образом, что каждый из ее выходов строится по аналогии с предыдущими. В нерегулярных структурах такая аналогия отсутствует. Многие регулярные структуры положены в основу построения отдельных ИС малой и средней степени интеграции или отдельных функциональных частей БИС и СБИС. Из регулярных комбинационных схем наиболее распространены дешифраторы, шифраторы, схемы сравнения, комбинационные сумматоры, коммутаторы и др.
Для построения любой КС необходима таблица истинности ее функционирования (составляется или задается), затем составляется функция зависимости каждого выхода схемы от входа (в форме СДНФ, которую затем можно перевести в упрощенную форму) и производится построение схемы на определенных логических элементах (чаще всего на И-НЕ и ИЛИ-НЕ). Как правило, построение и расчет любой схемы осуществляется начиная с ее выхода. Допустим задано булево выражение : $F = \bar{B}A + B\bar{A} + C\bar{B}$. Первый этап: выполняется логическое сложение (т.е. логическая операция ИЛИ), считая входными переменными функции $\bar{B}A$, $B\bar{A}$, $C\bar{B}$.



Второй этап: к входам элемента ИЛИ подключаются логические элементы И, входными переменными которых являются уже A, B, C и их инверсии:



Третий этап: для получения инверсий $\bar{A}$ и $\bar{B}$ на соответствующих входах ставят инверторы:



Как видно из построения, любые логические функции могут быть представлены как аргументы других более сложных функций, и наоборот: любую сколь угодно сложную функцию можно представить как совокупность стандартных функций.

Лабораторная работа №10.

Закон Де – Моргана и их применение в построении комбинационных схем.

Цель работы: Ознакомится с правиламиДе – Моргана и научиться применить их в построении комбинационных схем.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Законы двойственности или, другими словами, законы Де – Моргана, открытые шотландским логиком Огастосом де Морганом в девятнадцатом веке нашли широко применение в исчислении высказываний, в теории множеств, в теории автоматов, в теории алгоритмов и других областях науки и техники. Они относятся к одним из широко применяемых инструментов при оптимизации алгоритмов, систем автоматики, при проектировании радиорелейных системах и в других практических применениях.

Огастес де Морган первоначально заметил, что в классической пропозициональной логике справедливы следующие соотношения:

not (P and Q) = (not P) or (not Q)

not (P or Q) = (not P) and (not Q)

Обычная запись этих законов в формальной логике:

$$\neg(P \wedge Q) = (\neg P) \vee (\neg Q),$$
$$\neg(P \vee Q) = (\neg P) \wedge (\neg Q),$$

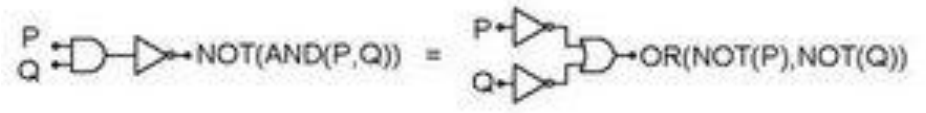
или

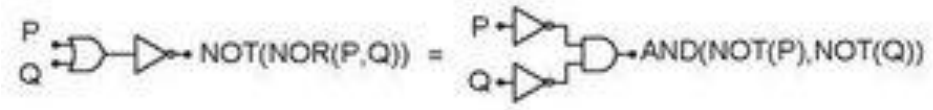
$$\overline{x \wedge y} = \overline{x} \vee \overline{y}$$
$$\overline{x \vee y} = \overline{x} \wedge \overline{y}$$

в теории множеств:

$$(A \cap B)^C = A^C \cup B^C,$$
$$(A \cup B)^C = A^C \cap B^C,$$

в схемотехнической логике:





Вышеназванные соотношения в пропозициональной логике формулировались следующим образом:

«Противоречащая противоположность дизъюнктивого суждения — конъюнктивное суждение, составленное из противоречащих противоположностей частей дизъюнктивого суждения (Thecontradictoryoppositeof a disjunctivepropositionis a conjunctivepropositioncomposedofthecontradictoriesofthepartsofthedisjunctiveproposition)» (Уильям Оккам, SummaLogicae).

Де – Морган разделил это тождество на две части и обосновал их как отдельные логические тождества, которые, впоследствии получили название «законов Де-МорганаПервый закон де Моргана гласит: "Отрицание конъюнкции высказываний равнозначно дизъюнкции отрицаний этих высказываний", что выражается следующей формулой:

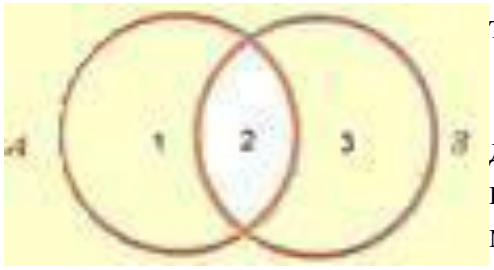
$\overline{A \wedge B} \equiv A' \vee B'$; Здесь знак $\vee$ обозначает союз "и", символизирует операцию "конъюнкция", а знак $\wedge$ обозначает союз "или", символизирует знак "дизъюнкция" - черта сверху буквы - знак отрицание, знак $\equiv$ означает равнозначность.

Конъюнкция или, иными словами, логическое умножение это операция математической логики,, соединяющая два и более высказываний. при помощи

союза, сходного с союзом "и", в новое сложное высказывание, которое истинно тогда и только тогда, когда каждое из исходных высказываний истинно., и ложно тогда, когда по крайней мере одно из исходных высказываний ложно.

Если истинное высказывание обозначить цифрой 1, а ложное цифрой 0, то таблица истинного значения конъюнкции будет выглядеть так:

	B	A ∨ B
1	1	1
1	0	0
0	1	0
0	0	0



Откуда следует справедливость первой теоремы де Моргана.

Графически конъюнкцию можно представить двумя наложенными областями, где элементы во второй области принадлежат и множеству А и множеству В. Отрицание принадлежности элементов в этой области, возможно тогда и только тогда, когда этих элементов нет или в области А, или в области В, или в обеих областях одновременно, что записывается `A ∨`B.

Второй закон де Моргана, $\overline{A \vee B}$ = `A ∧ `B говорит, что отрицание дизъюнкции равнозначно конъюнкции отрицаний этих высказываний.

Применение законов де-Моргана.

При реализации цифровых устройств на интегральных микросхемах широко используются базисы **И-НЕ** или **ИЛИ-НЕ**. Для этого минимизированные логические функции путем преобразований приводятся к соответствующему виду.

Пусть минимальная ДНФ функция

$$F(A, B, C) = AB \vee BC \vee AC.$$

Применим к этому выражению двойное отрицание и теорему де Моргана

$$F = \overline{\overline{F}} = \overline{\overline{AB \vee BC \vee AC}} = \overline{(\overline{AB})(\overline{BC})(\overline{AC})}.$$

Как видно, функция F включает только операции **И-НЕ**, и ее реализация в базисе **И-НЕ** имеет вид (рис. 8)

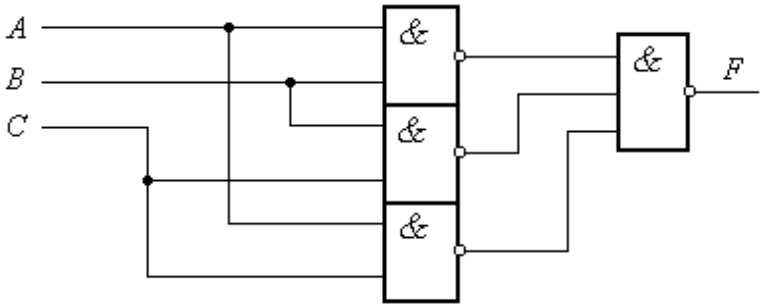
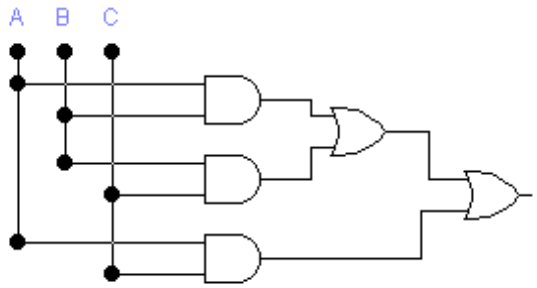
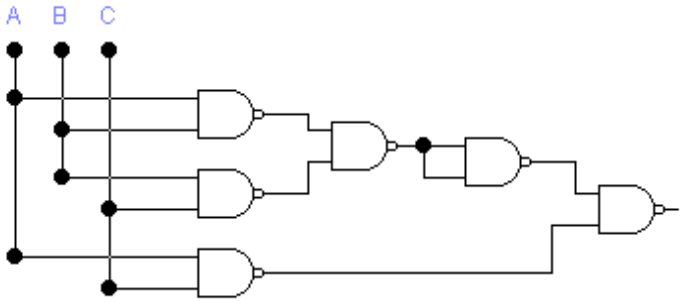


Схема реализации функции $F = AB \vee BC \vee AC$ в базисе И-НЕ



Реализация на базисе И,ИЛИ,НЕ (1 базис)



Реализация на базисе И-НЕ (2 базис)

Аналогичным образом от КНФ функции можно перейти к ее форме, удобной для реализации в базисе **ИЛИ-**

Заключение.

Огастес де Морган первоначально заметил, что в классической пропозициональной логике справедливы следующие соотношения:

не (a и b) = (не a) или (не b)
не (a или b) = (не a) и (не b)

В математике это выглядит так:

$\neg(a \wedge b) = \neg a \vee \neg b$
$\neg(a \vee b) = \neg a \wedge \neg b$ или по другому: $\overline{(a \wedge b)} = \overline{a} \vee \overline{b}$
$\overline{(a \vee b)} = \overline{a} \wedge \overline{b}$

В теории множеств:

$\overline{A \cap B} = \overline{A} \cup \overline{B}$
$\overline{A \cup B} = \overline{A} \cap \overline{B}$ или по другому: $(A \cap B)^C = A^C \cup B^C,$
$(A \cup B)^C = A^C \cap B^C.$

Эти правила также действительны для множества элементов:

$\overline{\bigcap_{i \in I} A_i} = \bigcup_{i \in I} \overline{A_i}$ и $\overline{\bigcup_{i \in I} A_i} = \bigcap_{i \in I} \overline{A_i}$.

В исчислении предикатов:

$\neg \forall x \, P(x) \equiv \exists x \, \neg P(x),$
$\neg \exists x \, P(x) \equiv \forall x \, \neg P(x).$

Следствия:

Используя законы Де Моргана можно выразить конъюнкцию через дизъюнкцию и три отрицания. Аналогично можно выразить дизъюнкцию:

$a \wedge b = \neg(\neg a \vee \neg b)$
$a \vee b = \neg(\neg a \wedge \neg b)$

Лабораторная работа № 11

Полу суммирующие и полно суммирующие логические элементы

Цель работы: Ознакомится с полу суммирующими и полно суммирующими логическими элементами.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Счётные устройства: двоичные и недвоичные, десятичные, асинхронные и синхронные, суммирующие и вычитывающие, реверсивные счётные устройства.

Счетчиком называется устройство, предназначенное для подсчета числа входных сигналов и хранения в определенном двоичном коде этого числа.

Счетчики - это цифровые автоматы, внутренние состояния которых определяются только количеством сигналов "1", пришедших на вход. Сигналы "0" не изменяют их внутренние состояния.

Триггер Т-типа является простейшим счетчиком, который считает до двух. Счетчик, образованный цепочкой из m триггеров, сможет подсчитывать в двоичном коде 2^m входных импульсов. Каждый из триггеров в этой цепочке называют разрядом счетчика.

Основная характеристика счетчика – модуль счета, или емкость счетчика $K_{сч}$.. Это количество поступивших входных сигналов, которое возвращает счетчик в исходное состояние.

Количество триггеров, необходимое для реализации счетчика, равно $m = \log_2 ,K_{сч}.$, где m – ближайшее большее целое число.

Цифровые счетчики классифицируются следующим образом:

- по модулю счета: двоичные, двоично-десятичные или с другим основанием счета, недвоичные с постоянным модулем счета, с переменным модулем счета;
- по направлению счета: суммирующие, вычитающие, реверсивные;
- по способу организации внутренних связей: с последовательным переносом, с параллельным переносом, с комбинированным переносом, кольцевые.

Классификационные признаки независимы и могут встречаться в различных сочетаниях: например, суммирующие счетчики бывают как с последовательным, так и с параллельным переносом и могут иметь двоичный, десятичный и иной модуль счета.

В суммирующем счетчике каждый входной импульс увеличивает число, записанное в счетчик, на единицу (для счетчиков с естественным порядком счета) и на единицу и более для счетчиков с произвольным порядком счета.

Вычитающий счетчик действует обратным образом: двоичное число, хранящееся в счетчике, с каждым поступающим импульсом уменьшается. Переполнение счетчика наступает при поступлении на его вход количества импульсов большего $K_{сч}$..

Реверсивный счетчик может работать в качестве суммирующего и вычитающего. Эти счетчики имеют дополнительные входы для задания направления счета.

Счетчики могут быть как асинхронными, так и синхронными.

Последовательные счетчики.

Рассмотрим работу суммирующего двоичного счетчика ($K_{сч.} = 2^m$) с естественным порядком счета и с $K_{сч.} = 8$. Для его построения необходимо $m = \log_2 8 = 3$ триггера, что соответствует трем разрядам двоичного числа.

Таблица состояний такого счетчика имеет вид, причем входной сигнал x^nобозначим через 1, Q_3^n – старший разряд, Q_1^n – младший разряд.

x^n	Q_3^n	Q_2^n	Q_1^n	Q_3^{n+1}	Q_2^{n+1}	Q_1^{n+1}
1	0	0	0	0	0	1
1	0	0	1	0	1	0
1	0	1	0	0	1	1

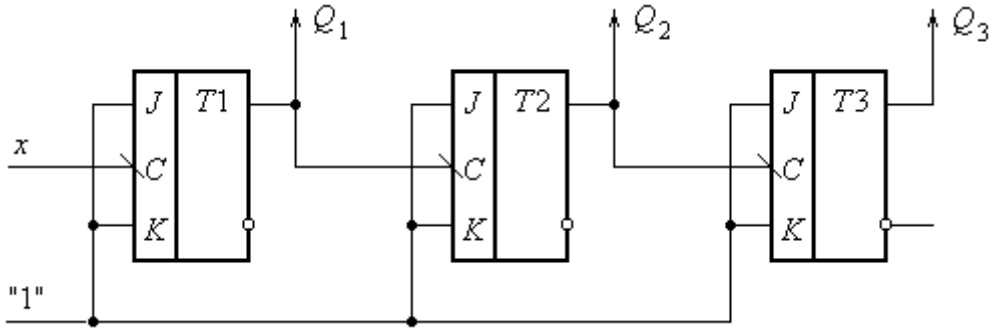
1	0	1	1	1	0	0
1	1	0	0	1	0	1
1	1	0	1	1	1	0
1	1	1	0	1	1	1
1	1	1	1	0	0	0

Из анализа таблицы видно:

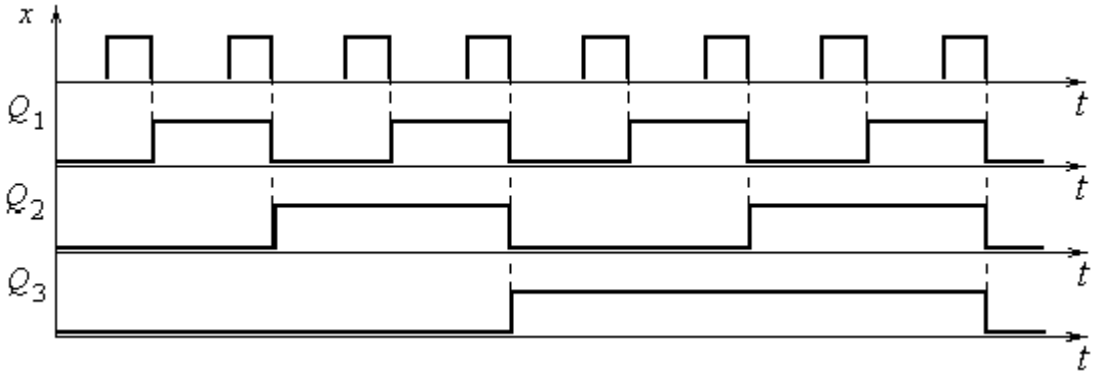
- триггер младшего разряда Q1 переключается от каждого входного сигнала;
- второй разряд Q2 переключается через два входных сигнала;
- третий разряд Q3 переключается через четыре входных сигнала.

Таким образом, частота переключения каждого следующего триггера уменьшается вдвое. Следовательно, счетчик можно построить как цепочку последовательно включенных счетных триггеров.

Построим такой счетчик на JK-триггерах, работающих в счетном режиме



А



Б

Последовательный суммирующий счетчик на JK-триггерах – а; временная диаграмма его работы – б

Данный счетчик может работать как вычитающий. Для этого необходимо сигналы на входы последующих разрядов подавать с инверсных выходов триггеров предыдущих разрядов.

Так как полученный счетчик – асинхронный, то каждый его триггер срабатывает с задержкой относительно входного сигнала. Поэтому по мере продвижения сигнала от младшего разряда к старшему эта задержка суммируется и может произойти искажение информации, в виде несоответствие числа уже поступивших в счетчик импульсов и кода на его выходах. В общем случае суммарная задержка пропорциональна числу триггеров, что снижает быстродействие счетчика.

Счетчики с параллельным переносом.

Для повышения быстродействия счетчики выполняются синхронными с параллельным переносом (или параллельными).

Их особенность заключается в том, что выходы всех предшествующих разрядов соединяются с входами триггера последующего разряда, поэтому длительность переходного процесса определяется только длительностью переходного процесса одного разряда и не зависит от количества триггеров.

Отсюда следует, что параллельные счетчики – синхронные.

Структура параллельного счетчика не столь очевидна, как структура последовательного счетчика, и для ее выявления необходима определенная процедура синтеза.

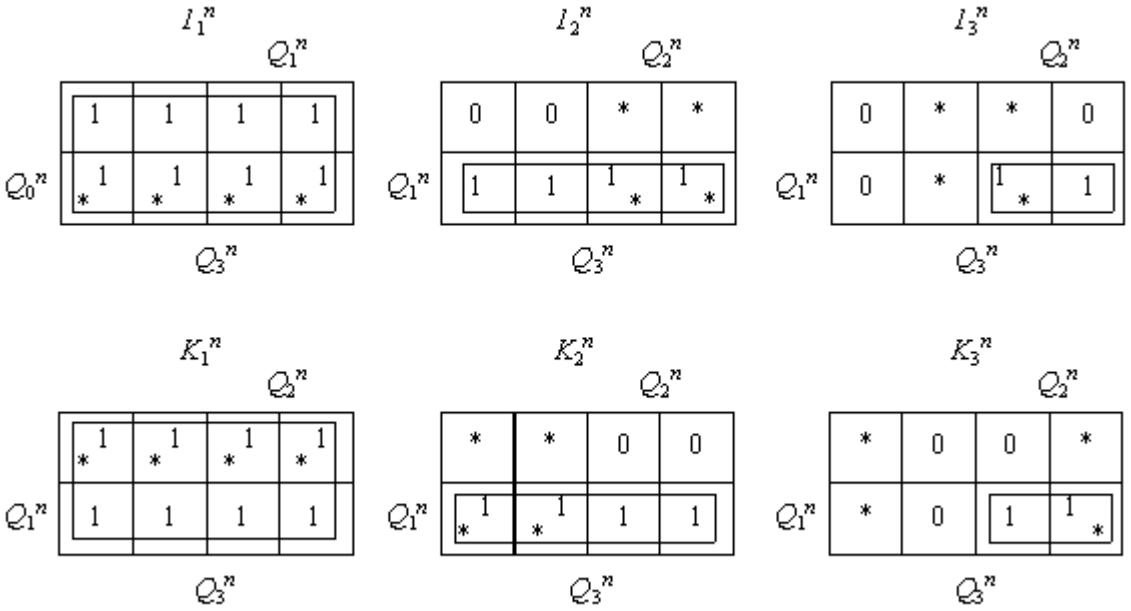
Суммирующий счетчик. Процедура синтеза включает следующие операции:

Определяется необходимое количество разрядов m. В данном случае m = log2 8 = 3.

Составляются карты Карно для функций переходов триггеров каждого разряда. Карта переходов строится по таблице состояний и отображает переход триггера$Q_i^n \rightarrow Q_i^{n+1}$ в каждом такте в зависимости от состояний остальных триггеров в такте n

	Для $Q_1^n \rightarrow Q_1^{n+1}$					Для $Q_2^n \rightarrow Q_2^{n+1}$					Для $Q_3^n \rightarrow Q_3^{n+1}$			
			Q_2^n					Q_2^n					Q_2^n	
	01	01	01	01		00	00	11	11		00	11	11	00
Q_1^n	10	10	10	10	Q_1^n	01	01	10	10	Q_1^n	00	11	10	01
		Q_3^n					Q_3^n					Q_3^n		

Выбирается тип триггера, например, JK-триггер, для построения счетчика. Используя матрицу переходов JK-триггера, для каждого входа триггера составляются карты Карно, в клетках которых проставляются сигналы, необходимые для обеспечения переходов триггеров, указанных в одноименных клетках карт функций переходов.



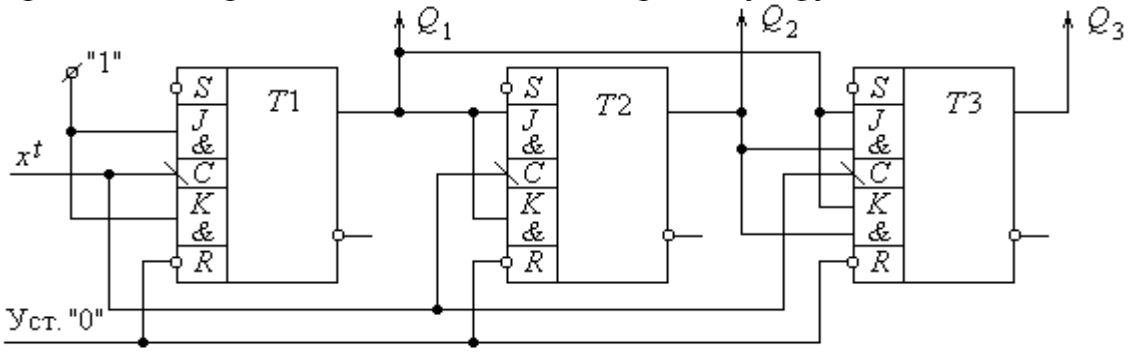
Например, для переходов 01 JK-триггера согласно его матрице переходов необходимо подать сигнал J = 1, а сигнал на входе K может быть любым (* – звездочка), поэтому в верхнюю левую клетку карты Карно для J1 проставляют единицу, а для K1 – звездочку и т.д.

Проводится минимизация логических функций входов в картах Карно с целью получения их аналитических представлений, показывающих связи между входами и выходами всех триггеров, составляющих счетчик.

В процессе минимизации производится доопределения функций там, где это целесообразно, единицами в клетках со звездочками.

В результате получены следующие функции входов триггеров счетчика:

Строится электрическая схема счетчика, реализуя функции



входов

.

Параллельный суммирующий двоичный счетчик с K сч. = 8

В качестве триггеров выбраны универсальные JK-триггеры (микросхема К155ТВ1), особенностью которых является наличие логики типа ЗИ на входах J и K и дополнительных R S входов с инверсным асинхронным управлением.

Вычитающий счетчик. Синтез вычитающего счетчика, работающего в соответствии с таблицей переходов обратной таблице 13, включает все рассмотренные выше процедуры и дает следующие функции входов:

J1 = K1 =1

J2 = K2 = $\overline{Q_1}$

J3 = K3 = $\overline{Q_1}\,\overline{Q_2}$.

Таким образом, вычитающий счетчик отличается от суммирующего тем, что сигналы на входы J и K последующих триггеров необходимо подавать с инверсных выходов триггеров предшествующих разрядов. Так как исходное состояние вычитающего счетчика – единицы во всех разрядах, то организуется общая шина установки по -входам.

Реверсивный счетчик - Такой счетчик должен, в зависимости от сигналов управления, обеспечивать или режим суммирования, или режим вычитания входных сигналов.

Из сравнения функций входов, полученных ранее для суммирующего и вычитающего параллельных счетчиков с Kсч.=8, следует, что сами функции имеют один и тот же вид, только в случае вычитающего счетчика берутся инверсные значения переменных. Следовательно, реверсивный счетчик должен содержать схему управления, обеспечивающую подключение либо прямых, либо инверсных выходов ко входам последующих разрядов, в зависимости от сигналов управления направлением счета T.

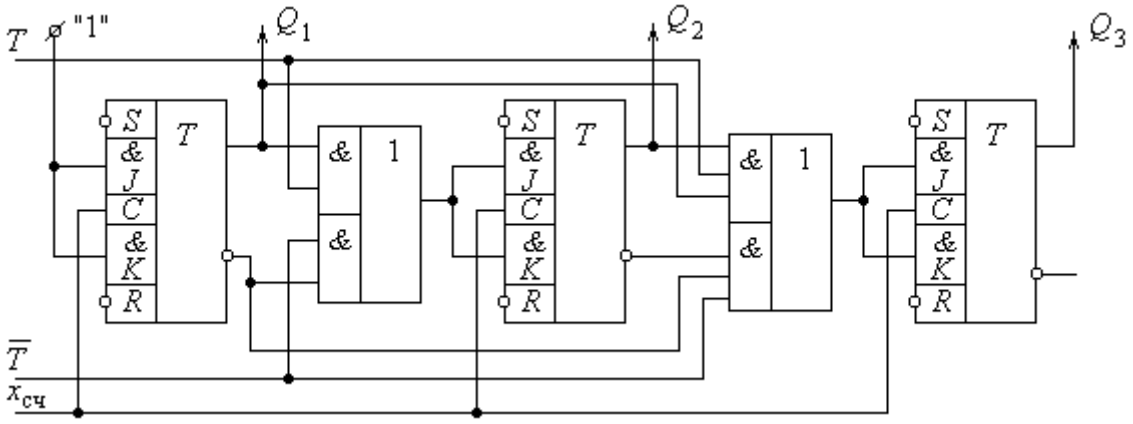
Функция входов для реверсивного счетчика будет иметь вид:

K1 = J1 = 1,

J2 = K2 = TQ1 $\vee \overline{T}\overline{Q_1}$,

J3 = K3 = TQ1 Q2 $\vee \overline{T}\overline{Q_1}\,\overline{Q_2}$,

а его схема представлена на рисунке.



Реверсивный двоичный параллельный счетчик с K сч. = 8.

Счетчик работает в режиме суммирования при T = 1 и в режиме вычитания при T = 0.

Недвоичные счетчики. Счетчик, имеющий K сч. № 2m, называется недвоичным. Состояния (2m – K сч.) являются избыточными и исключаются внутри счетчика с помощью обратных связей. Задача синтеза таких счетчиков сводится к определению вида необходимых обратных связей и минимизации их числа.

Рассмотрим пример синтеза суммирующего счетчика с K сч. = 3.

Определяем необходимое количество триггеров:

$m = \log_2 3 = 1{,}58$

Округляем m до двух.

Находим число избыточных состояний:

$2^2 - 3 = 1$

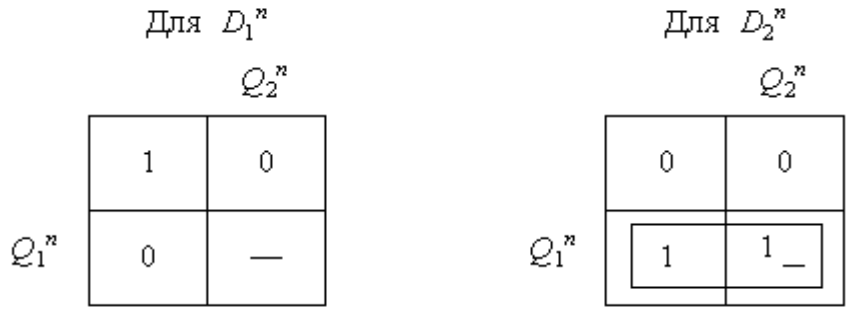
Из числа возможных состояний счетчика исключим, например, состояние

Q1 = Q2 = 1

Строим таблицу переходов счетчика:

Составляем карты переходов триггеров счетчика, проставляя в клетках, соответствующим исключенным наборам, прочерк:

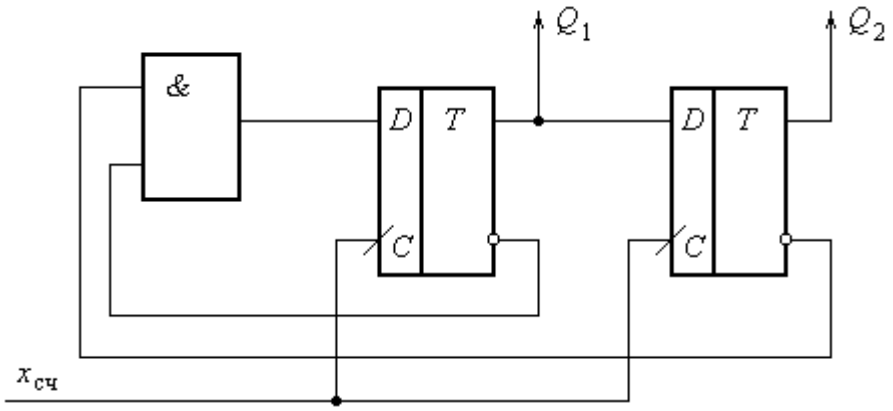
Выбираем тип триггеров (D-триггер). Используя матрицу переходов D-триггера и построенные карты переходов триггеров счетчика, строим карты функций входов триггеров:



Находим функции входов триггеров счетчика:

$D_1 = \overline{Q_1}\,\overline{Q_2}, \quad D_2 = Q_1$

Строим схему счетчика:



Параллельный недвоичный счетчик с K сч. = 3 на D-триггерах.

Как видно из схемы, исключение из состояний счетчика двоичного числа 11 достигается подачей сигналов с инверсных выходов первого и второго разрядов на вход первого разряда.

При использовании в счетчике триггеров JK-типа функции входов имеют вид:

J1 = $\overline{Q}_2$, J2 = Q1 , K1 = K2 = 1.

Лабораторная работа № 12
Построение смех в программе ElectronicsWorkbench

Цель работы: Изучение программной среды ElectronicsWorkbench и научиться строить схемы с помощью программы.

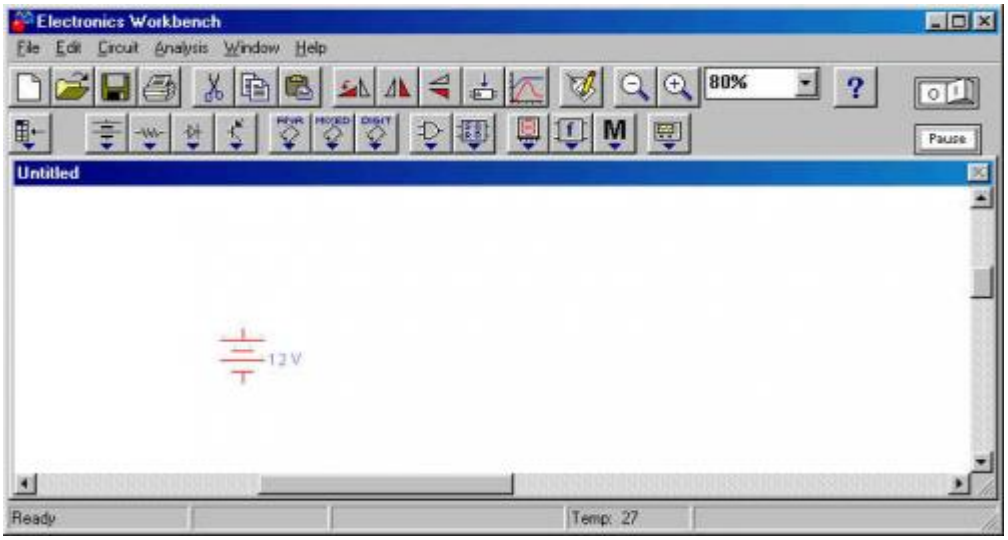
Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоритическая часть.

Программа EWB создана в 1989 г. канадской фирмой InteractiveImageTechnologies для схемотехнического моделирования аналоговых и цифровых радиоэлектронных устройств различного назначения. Существует несколько программ подобного рода (SystemView, Wicro-Cap, DesignLab, Aplac и др.), однако EWB отличается тем, что она позволяет использовать КИП, по внешнему виду, органам управления и характеристикам максимально приближенные к их промышленным аналогам. Все описания на английском языке. Схема составляется из библиотек компонентов или путем изменения уже готовых схем. Выделение элемента – щелчок левой кнопки мыши. Удаление – клавишаDel или соответствующая команда контекстного меню. При составлении схемы и ее упрощения путем оформления подсхем моделирование ее работы начинается щелчком выключателя.

Используемые тип файлов - .ewb. Эти файлы обычно находятся в каталоге Samples. Например, файл Rs-ff.ewb содержит схему RS-триггера на логических элементах ИЛИ-НЕ.

Структура окна.



Основную, центральную часть окна приложения занимает рабочая область, в которой собирается и тестируется электронная схема. Над рабочей областью расположена панель кнопок. С помощью кнопок, имеющихся на этой панели, можно изменять содержимое окна набора компонентов. Изображения на кнопках показывают, какие типы компонентов появятся в наборе при нажатии соответствующей кнопки. Справа в этом же ряду расположен переключатель, позволяющий включать и выключать собранную схему. Переключение осуществляется с помощью щелчка мышью на переключателе. Над иконками приборов расположено меню.

В ElectronicsWorkbench сборка схемы осуществляется в рабочей области. Электронные компоненты для сборки схемы берутся из меню, содержащего набор компонентов. Содержимое набора компонентов можно изменить нажатием соответствующих кнопок, расположенных непосредственно над окнами. Чтобы переместить требуемый компонент в рабочую область, нужно поместить на него курсор и нажать левую клавишу мыши. Затем, удерживая клавишу в нажатом состоянии, "перетащить" элемент, двигая мышь, в требуемое положение в рабочей области и отпустить клавишу.

Чтобы осуществить какие-либо операции над элементом его необходимо выделить. Выделение элемента осуществляется щелчком мыши на элементе, при этом он окрашивается в красный цвет.

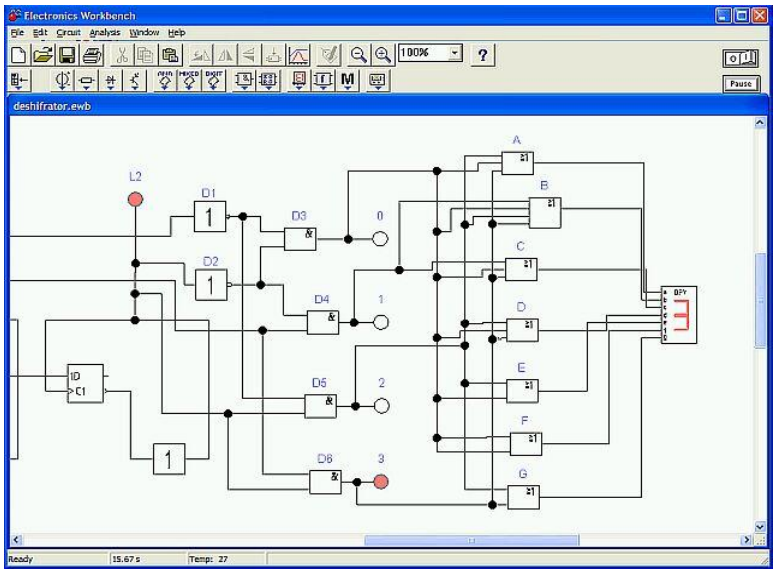
Если необходимо повернуть элемент, нужно сначала его выделить, а затем использовать комбинацию клавиш [Ctrl+R], нажатие которых приводит к повороту элемента на 90°.

Для удаления элемента его также необходимо сначала выделить, а затем нажать клавишу [Del] и в ответ на запрос о подтверждении удаления нажать кнопку подтверждения или отмены удаления.

Все электронные компоненты характеризуются своими параметрами, определяющими их поведение в схеме. Чтобы задать эти параметры нужно дважды щелкнуть мышью на нужном элементе, в результате чего появится диалоговое окно, в котором необходимо выбрать или записать требуемые параметры и закрыть его нажатием кнопки "Ok".

Чтобы соединить между собой выводы элементов подведите курсор к нужному выводу, при этом, если к этому выводу действительно можно подсоединить проводник, на нем появится маленький черный кружок. При появлении кружка нажмите левую клавишу мыши и, не отпуская ее, протащите курсор к другому выводу. Когда на другом выводе тоже появится черный кружок, отпустите клавишу, и эти выводы автоматически будут соединены проводником. Если вывод элемента нужно подсоединить к уже имеющемуся проводнику, то подведите курсор мыши при нажатой клавише к этому проводнику, при этом также в том месте, где можно сделать подсоединение появится маленькая окружность. В этот момент отпустите клавишу, и в схеме автоматически образуется проводящее соединение между проводниками, обозначенное черным кружком.

Программная часть.



Лабораторная работа №13

Кодеры, шифраторы и дешифраторы.

Цель работы: Ознакомится с работой кодирующих и декодирующих устройств (шифраторов и дешифраторов).

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Дешифратор – это устройство, предназначенное для преобразования двоичного кода в напряжение логической единицы (логического нуля) на том выходе, номер которого совпадает со значением двоичного кода на входе. При *n* входах в *полном*

дешифраторе имеется 2^n выходов, т.е. для каждой комбинации входных сигналов имеется соответствующий выход. Дешифратор, у которого при *n* входах число выходов меньше 2^n, называется *неполным*. Другое название дешифратора - *декодер*. Принцип работы полного трехразрядного дешифратора рассмотрим на примере его таблицы истинности.

входы			выходы							
3	2	1	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Соответствующие таблице истинности ФАЛ имеют вид

$$Y_0 = \overline{X_3} \cdot \overline{X_2} \cdot \overline{X_1}$$

$$Y_1 = \overline{X_3} \cdot \overline{X_2} \cdot X_1$$

$$Y_2 = \overline{X_3} \cdot X_2 \cdot \overline{X_1}$$

$$Y_3 = \overline{X_3} \cdot X_2 \cdot X_1$$

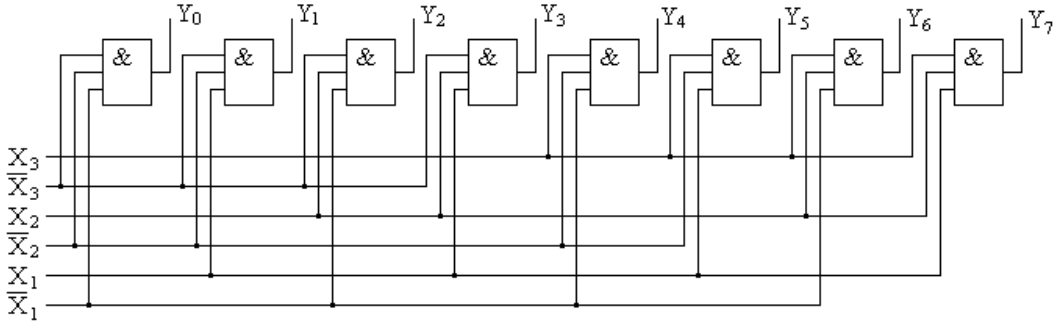
$$Y_4 = X_3 \cdot \overline{X_2} \cdot \overline{X_1}$$

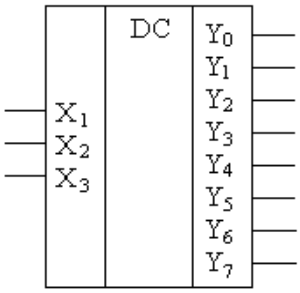
$$Y_5 = X_3 \cdot \overline{X_2} \cdot X_1$$

$$Y_6 = X_3 \cdot X_2 \cdot \overline{X_1}$$

$$Y_7 = X_3 \cdot X_2 \cdot X_1.$$

Структурная схема трехразрядного дешифратора, синтезированная на основании полученных ФАЛ приведена на рис. 4.10,*а*, а его УГО - на рис. 4.10,*б*.





б)

Рис. 4.10.Структурная схема и УГО трехразрядного дешифратора.

В общем случае логические уравнения для выходных переменных дешифратора *n*-разрядного числа имеют вид

$$Y_0 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot \overline{X_2} \cdot \overline{X_1}$$

$$Y_1 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot \overline{X_2} \cdot X_1$$

$$Y_2 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot X_2 \cdot \overline{X_1}$$

$$Y_3 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot X_2 \cdot X_1$$

……

$$Y_{2^n-1} = X_n \cdot X_{n-1} \cdot \_ \cdot X_2 \cdot X_1.$$

Построенные по полученным формулам дешифраторы называются *линейными*. К преимуществу линейных дешифраторов можно отнести высокое быстродействие, поскольку входные переменные одновременно поступают на все элементы И. Одновременно, без дополнительных задержек, формируется и результат на выходах этих элементов. Очевидно, что для реализации линейного дешифратора *n*-разрядного числа необходимо иметь 2^n логических элементов И с *n*-входами. В существующих микросхемах логических элементов количество входов ограничено. Следовательно, ограничена и разрядность реализуемых на их основе линейных дешифраторов, что является недостатком. Кроме того, недостатком является и то, что предыдущие элементы, работающие на входы дешифратора, должны иметь высокую нагрузочную способность, т.е. должны быть рассчитаны на подключение большого числа логических элементов И. Каждый из входов дешифратора подключен к $0{,}5 \cdot 2^n$ логическим элементам И. Поскольку нагрузочная способность базовых логических элементов ИС не превышает величины N=10¸20, то максимальная разрядность дешифрируемых чисел для линейных дешифраторов n=4¸5.

Указанного недостатка лишены *пирамидальные дешифраторы*. Принцип построения этих дешифраторов состоит в том, что сначала строят линейный дешифратор для двухразрядного числа X_1, X_2, для чего необходимы 2^2=4 двухвходовые схемы И. Далее, каждая полученная конъюнкция логически умножается на входную переменную X_3 в прямой и инверсной форме. Полученная конъюнкция снова умножается на входную переменную X_4 в прямой и инверсной форме и т.д. Наращивая таким образом структуру, можно построить пирамидальный дешифратор на произвольное число входов. На рис. 4.11 приведена структура пирамидального дешифратора для трех разрядов.

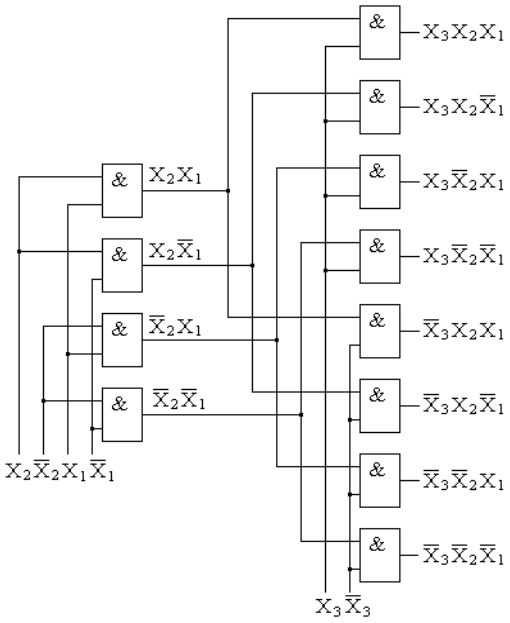


Рис. 4.11. Пирамидальный дешифратор для трехразрядного числа.

Характерным отличием пирамидальных дешифраторов от линейных является использование только двухвходовых логических элементов вне зависимости от разрядности дешифрируемого числа. В то же время количество логических элементов в пирамидальном дешифраторе больше. Однако следует иметь ввиду, что количество логических элементов, располагаемых в одном корпусе ИС, определяется главным образом требуемым количеством выводов. Следовательно, в одном корпусе ИС можно расположить большее число двухвходовых элементов, чем трехвходовых, четырехвходовых и т.д. И значит, пирамидальная структура дешифратора по числу корпусов ИС может оказаться более предпочтительной, чем линейная.

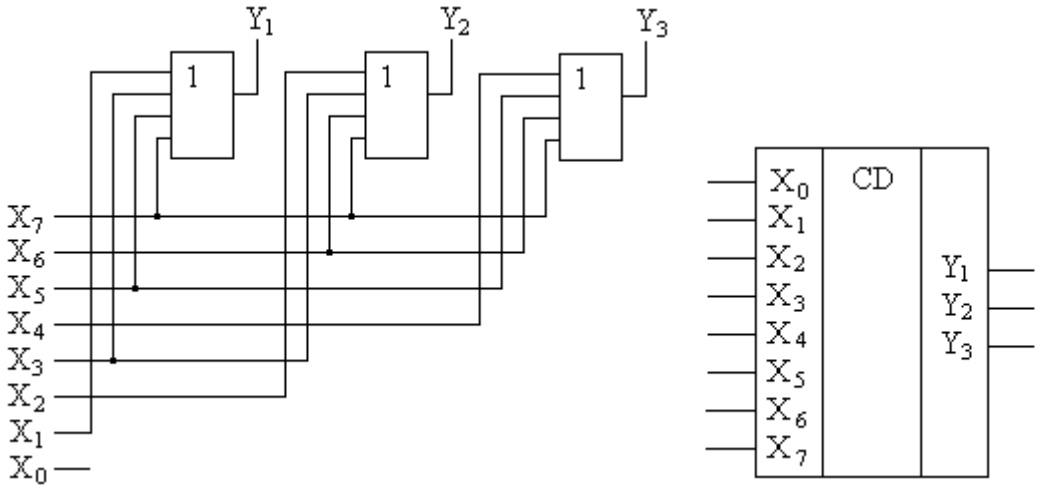
Шифраторы выполняют задачу обратную той, которую выполняют дешифраторы: появление логической единицы (логического нуля) на определенном входе приводит к появлению соответствующей кодовой комбинации на выходе. Также как и дешифраторы, шифраторы бывают полными и неполными. Работа восьмивходового полного шифратора задается следующей таблицей истинности:

Входы								**Выходы**		
X_7	X_6	X_5	X_4	X_3	X_2	X_1	X_0	Y_3	Y_2	Y_1
0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**
0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**
0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**1**
0	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**
1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**

На основании таблицы истинности можно записать ФАЛ, задающие работу восьмивходового шифратора:

$$Y_1 = X_1 \vee X_3 \vee X_5 \vee X_7$$
$$Y_2 = X_2 \vee X_3 \vee X_6 \vee X_7$$
$$Y_3 = X_4 \vee X_5 \vee X_6 \vee X_7.$$

Синтезированная на основании приведенных логических уравнений структурная схема шифратора представлена на рис. 4.12,*а*, а его условное графическое обозначение – на рис. 4.12,*б*.



а) *б*)

Рис. 4.12.Структура и УГО восьмивходового шифратора.

Заключение.

Шифратор (кодер) преобразует сигнал на одном из входов в n-разрядное двоичное число. Функциональная схема шифратора, преобразующего десятичные цифры в 4-разрядное двоичное число, приведена на рисунке 1.33,а, а его условное обозначение – на рисунке 1.33,б. При появлении сигнала логической единицы на одном из десяти входов на четырех выходах шифратора будет присутствовать соответствующее двоичное число. Пусть сигнал логической единицы подан на вход 7. Тогда на выходах логических элементов DD1.1, DD1.2, DD1.3 будут сигналы логических единиц, а на выходе элемента DD1.4 – сигнал логического нуля. Таким образом, на выходах 8, 4, 2, 1 шифратора мы получим двоичное число 0111.

Некоторые из шифраторов снабжаются входом стробирования. Наличие входа стробирования позволяет выделять сигнал в определенный момент времени.

Дешифратор (декодер) преобразует код, поступающий на его входы, в сигнал только на одном из его выходов. Дешифратор n-разряд-ного двоичного числа имеет 2^n выходов. Функциональная схема дешифратора на 16 выходов приведена на рисунке 1.34,а. По такой функциональной схеме построена микросхема К155ИД3. Условное обозначение этой микросхемы на принципиальных схемах приведено на рисунке 1.34,б. Для преобразования сигнала необходимо на входы V1 и V2 микросхемы подать сигналы логических нулей.

Лабораторная работа №14-15
Мультиплексоры и демультиплексоры.

Цель работы: Ознакомится с работой мультиплексора и демультиплексора.
Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

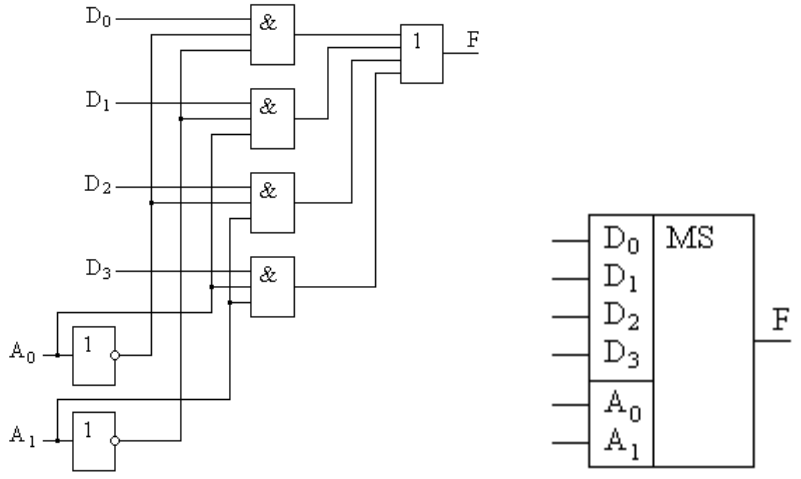
Мультиплексор - комбинационное цифровое устройство, которое обеспечивает передачу на единственный выход F одного из нескольких входных сигналов D_j в соответствии с поступающим адресным кодом A_i. При наличии n адресных входов можно реализовать $M=2^n$ комбинаций адресных сигналов, каждая из которых обеспечивает выбор одного из M входов. Чаще всего используются мультиплексоры «из 4 в 1» (n=2, M=4), «из 8 в 1» (n=3, M=8), «из 16 в 1» (n=4, M=16). Правило работы мультиплексора «из 4 в 1» можно задать таблицей истинности:

Входы		Выход
A_1	A_0	F
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Логическое выражение для выходной функции, заданной таблицей, можно записать в виде

$$F = D_0\bar{A}_1\bar{A}_0 \vee D_1\bar{A}_1A_0 \vee D_2A_1\bar{A}_0 \vee D_3A_1A_0.$$

В соответствии с полученной формулой для реализации мультиплексора можно использовать логические элементы И, ИЛИ, НЕ. Синтезированная структурная схема мультиплексора показана на рис. 4.13,а, а его условное графическое обозначение – на рис. 4.13,*б*.



а) *б*)

Рис. 4.13.Структура и УГО мультиплексора «из 4 в 1».

Мультиплексирование при большом числе входов можно выполнить пирамидальным каскадированием мультиплексоров, как это показано на рис. 4.14. На рисунке показано каскадирование мультиплексоров «из 4 в 1» для реализации функции мультиплексирования «из 16 в 1».

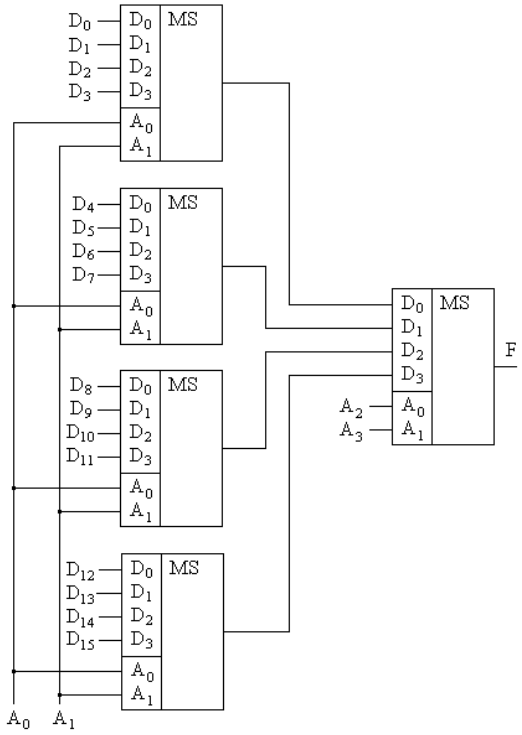
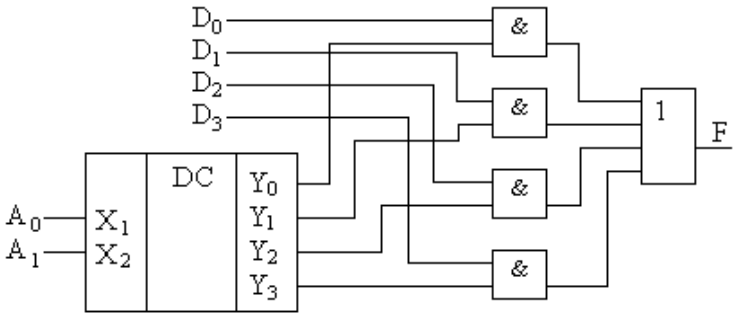


Рис. 4.14.Пирамидальное каскадирование мультиплексоров.

Мультиплексоры первого уровня управляются адресными сигналами A_0 и A_1, а мультиплексоры второго – адресными сигналами A_2 и A_3. Каждый из мультиплексоров первого уровня выбирает один из четырех разрядов D_j. Первый мультиплексор выбирает один из разрядов D_0 – D_3, второй мультиплексор – один из разрядов D_4 – D_7 и т.д. Выходы с мультиплексоров первого уровня объединяются в мультиплексоре второго уровня, который осуществляет окончательную коммутацию и формирование выходного сигнала F.

Мультиплексор можно реализовать, используя дешифратор и схемы И и ИЛИ (рис. 4.15). Дешифратор формирует логическую единицу на одном из выходов согласно входному двоичному коду. Сигналы с выходов дешифратора являются стробирующими, т.е. разрешающими сигналами для схемы совпадения единиц, реализованной на двухвходовых элементах И. Логическая единица будет формироваться на выходе только того элемента И, на один вход которого подается единица с выхода дешифратора и на второй вход – единица с соответствующего входа D_j. Для объединения выходов всех элементов И в один выход F, служит элемент ИЛИ. На его выходе формируется логическая единица, если таковая присутствует на опрашиваемом в данный момент входе D_j.



Демультиплексор выполняет функцию, обратную мультиплексору, т.е. в соответствии с принятой адресацией A_i направляет информацию с единственного входа D на один из M выходов F_j. При этом на остальных выходах будут логические нули (единицы). Принцип работы демультиплексора «из 1 в 4» иллюстрируется таблицей истинности:

Входы		Выходы			
A_1	A_0	F_3	F_2	F_1	F_0

0	0	0	0	0	*D*
0	1	0	0	*D*	0
1	0	0	*D*	0	0
1	1	*D*	0	0	0

Логические выражения для каждого из выходов можно представить в виде:

$$F_0 = D\bar{A}_1\bar{A}_0$$

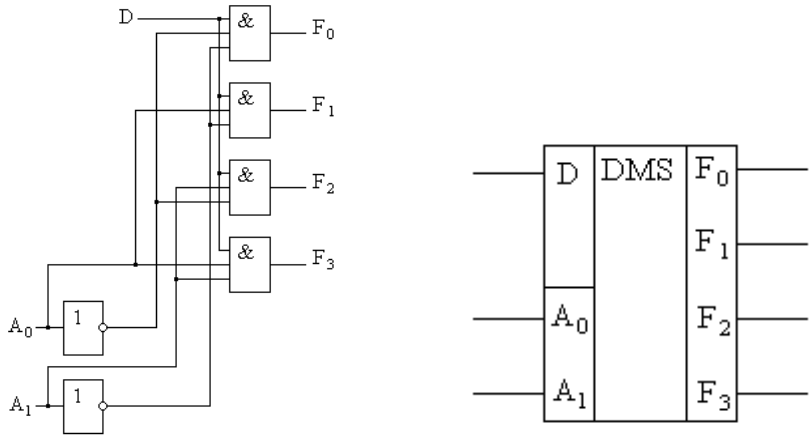
$$F_1 = D\bar{A}_1A_0$$

$$F_2 = DA_1\bar{A}_0$$

$$F_3 = DA_1A_0.$$

Структурная схема, реализующая демультиплексор «из 1 в 4» приведена на рис. 4.16,*а*, а его условное графическое обозначение – на рис. 4.16,*б*.

Как и в случае мультиплексора, схему демультиплексора можно реализовать с помощью дешифратора. Действительно, ФАЛ демультиплексора отличается отФАЛ дешифратора только наличием входного сигнала *D* в конъюнкциях с адресными входами. Следовательно, объединив выходы дешифратора с входом *D* с помощью стробирующих элементов И, можно получить демультиплексор (рис. 4.17). Мультиплексоры и демультиплексоры часто называют еще цифровыми *коммутаторами*.



а) *б*)

Рис. 4.16. Структурная схема и УГО демультиплексора «из 1 в 4».

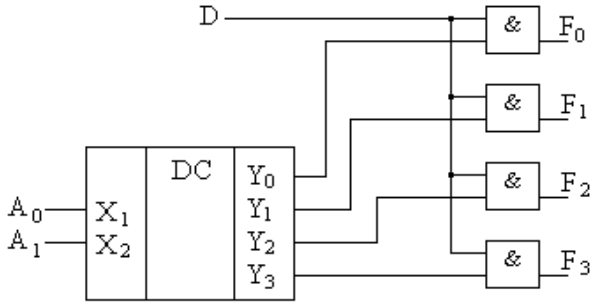


Рис. 4.17.Реализация демультирлексора на базе дешифратора.

Заключение.

Мультиплексор - цифровые позиционные переключатели, т.е. мультиплексор назначение коммутировать на одну выходную линию сигналы от различных выходных источников, следовательно мультиплексор имеет 3 группы входов :
1)информационные,
2)адресные – двоичный код, на котором определяется какой из информационных входов подключен к выходу;
3) стробирующие (разрешающие)
При разрядности адреса n, число информационных входов 2^n.

Демультиплексор – комбинационное логическое устройство, предназначенное для управляемой передачи данных от одного источника информации на несколько выходных каналов.

В общем случае:
Демультиплексор имеет один информационный вход, n адресных входов, 2^n выхода и вход разрешения.

Лабораторная работа №16.

Логические элементы и представление их в виде схем.

Цель работы: Ознакомится с логическими элементами и представлением их в виде схем.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

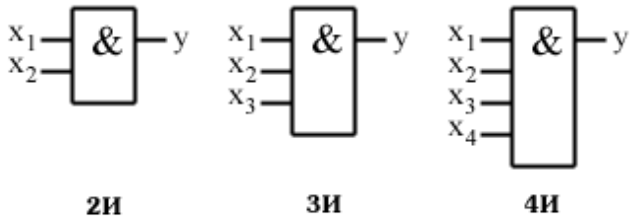
Теоретическая часть.

Логическим элементом называется электрическая схема, выполняющая какую-либо логическую операцию (операции) над входными данными, заданными в виде уровней напряжения, и возвращающая результат операции в виде выходного уровня напряжения. Так как операнды логических операций задаются в двоичной системе счисления, то логический элемент воспринимает входные данные в виде высокого и низкого уровней напряжения на своих входах. Соответственно, высокий уровень напряжения (напряжение логической 1) символизирует истинное значение операнда, а низкий (напряжение логического 0) - ложное. Значения высокого и низкого уровней напряжения определяются электрическими параметрами схемы логического элемента и одинаковы как для входных, так и для выходных сигналов. Обычно, логические

элементы собираются как отдельная интегральная микросхема. К числу логических операций, выполняемых логическими элементами относятся конъюнкция (логическое умножение, И), дизъюнкция (логическое сложение, ИЛИ), отрицание (НЕ) и сложение по модулю 2 (исключающее ИЛИ). Рассмотрим основные типы логических элементов.

Элемент И

Логический элемент И выполняет операцию логического умножения ьюнкция) над своими входными данными и имеет от 2 до 8 входов и один од (как правило, выпускаются элементы с двумя, тремя, четырьмя и восемью ами). На рис. 1. изображены условные графические обозначения (УГО) ческих элементов И с двумя, тремя и четырьмя входами соответственно. енты И обозначаются как NИ, где N - количество входов логического

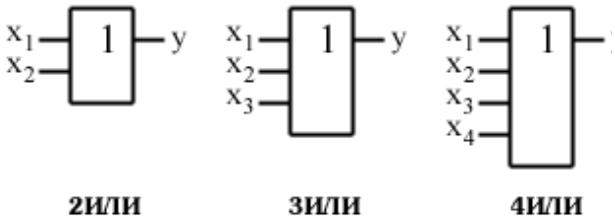


2И **3И** **4И**

ента (например, 2И, 3И, 8И и т.д.).

Элемент ИЛИ

Логический элемент ИЛИ выполняет операцию логического сложения (дизъюнкция) над своими входными данными и, также как и логический элемент И, имеет от 2 до 8 входов и один выход. На рис. 2. изображены УГО логических элементов ИЛИ с двумя, тремя и четырьмя входами соответственно. Элементы ИЛИ обозначаются также, как и элементы И (2ИЛИ, 4ИЛИ и т.д.).



2ИЛИ **3ИЛИ** **4ИЛИ**

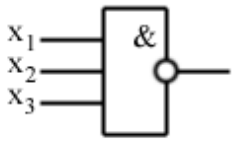
Элемент НЕ (инвертор)

Логический элемент НЕ выполняет операцию логического отрицания над своими входными данными и имеет один вход и один выход. Иногда его называют инвертор, так как он инвертирует входной сигнал. На рис. 3 изображено УГО элемента НЕ.

НЕ

Элемент И-НЕ

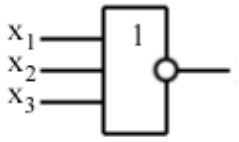
Логический элемент И-НЕ выполняет операцию логического умножения над своими входными данными, а затем инвертирует (отрицает) полученный результат и выдаёт его на выход. Таким образом, можно сказать, что логический элемент И-НЕ - это элемент И с инвертором на выходе. УГО элемента 3И-НЕ приведено на рис. 4.



ЗИ-НЕ

Элемент ИЛИ-НЕ

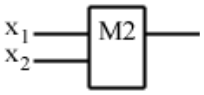
Логический элемент ИЛИ-НЕ выполняет операцию логического сложения над своими входными данными, а затем инвертирует (отрицает) полученный результат и выдаёт его на выход. Таким образом, можно сказать, что логический элемент ИЛИ-НЕ - это элемент ИЛИ с инвертором на выходе. УГО элемента ЗИЛИ-НЕ приведено на рис. 5.



ЗИЛИ-НЕ

Элемент сложения по модулю 2

Этот логический элемент выполняет логическую операцию сложения по модулю 2 и, как правило, имеет 2 входа и один выход. Такой элемент, в основном, используется в схемах аппаратного контроля. УГО элемента приведено на рис. 6.



Рассмотрим создание простейшей логической схемы на примере создания простого арифметического сумматора.

При сложении двоичных чисел в каждом разряде образуется сумма и при этом возможен перенос в старший разряд. Введем обозначения слагаемых (A, B), переноса (P), и суммы (S). Тогда таблица сложения одноразрядных двоичных чисел с учетом переноса в старший разряд выглядит следующим образом:

Слагаемые		Перенос	Сумма
A	B	P	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Отсюда видно, что перенос P=A*B, для СКНФ S=(A'+B')*(A+B);
дляСДНФ S=A'*B+A*B'.

Логические схемы с входами A,B и выходами P, S для соответствующих формул:

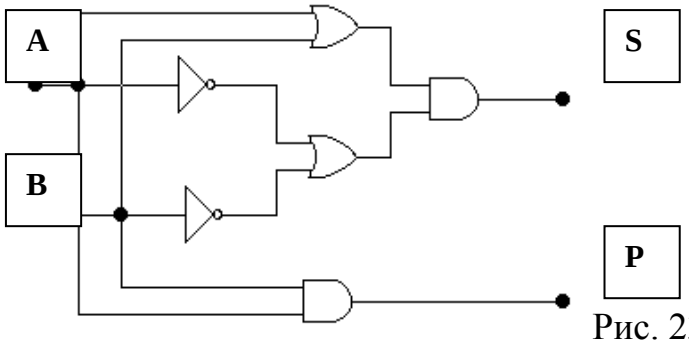


Рис. 23

A

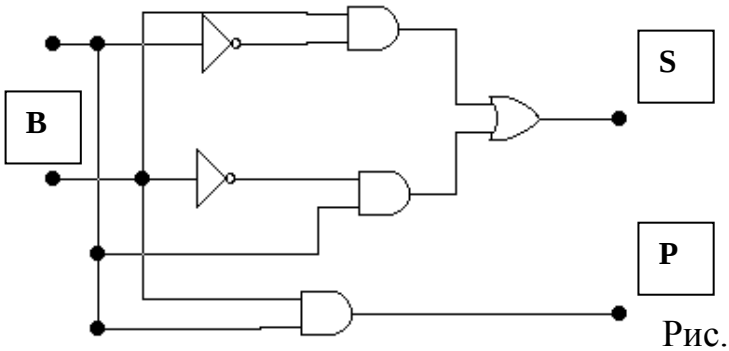


Рис. 24

Преобразуем формулу S=(A'+B')*(A+B), используя, закон де Моргана: S=(A*B)'*(A+B).

Логическая схема для данной формулы (рис. 25):

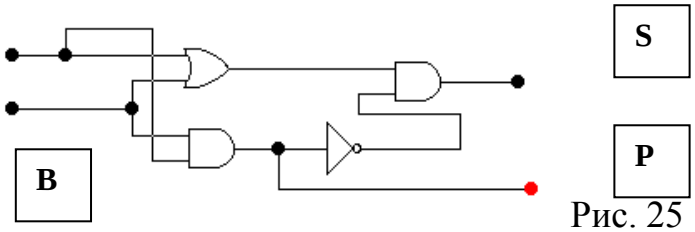


Рис. 25

Как видим, что для реализации арифметического устройства сложения достаточно четырех логических элементов.

Приведенные логические устройства называются полусумматорами.

Полный одноразрядный сумматор должен иметь три входа: A, B - слагаемые и P_0 - перенос из младшего разряда и два выхода сумму S и перенос P.

Слагаемые		Перенос из дшего разряда	Перенос	Сумма
A	B	P_0	P	S
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0

1	0	1	1	0
1	1	1	1	1

Заключение.

Логические схемы нужны для того чтобы в наглядной графической форме отобразить последовательность выполнения операций при вычислении логических формул.

Входящие слева линии и цифры около них обозначают значения операндов, линия справа и соответствующая цифра - результат операции (значение на выходе логических элементов).

Правило построения логических схем:

1) Определить число логических переменных.

2) Определить количество базовых логических операций и их порядок.

Лабораторная работа №17.

Объединение и изменение кодов.

Цель работы: Ознакомится с работой кодирующих и декодирующих устройств (шифраторов и дешифраторов).

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Дешифратор – это устройство, предназначенное для преобразования двоичного кода в напряжение логической единицы (логического нуля) на том выходе, номер которого совпадает со значением двоичного кода на входе. При *n* входах в *полном дешифраторе* имеется 2^n выходов, т.е. для каждой комбинации входных сигналов имеется соответствующий выход. Дешифратор, у которого при *n* входах число выходов меньше 2^n, называется *неполным*. Другое название дешифратора - *декодер*. Принцип работы полного трехразрядного дешифратора рассмотрим на примере его таблицы истинности.

ходы			ыходы							
3	2	1	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Соответствующие таблице истинности ФАЛ имеют вид

$$Y_0 = \overline{X_3} \cdot \overline{X_2} \cdot \overline{X_1}$$

$$Y_1 = \overline{X_3} \cdot \overline{X_2} \cdot X_1$$

$$Y_2 = \overline{X_3} \cdot X_2 \cdot \overline{X_1}$$

$$Y_3 = \overline{X_3} \cdot X_2 \cdot X_1$$

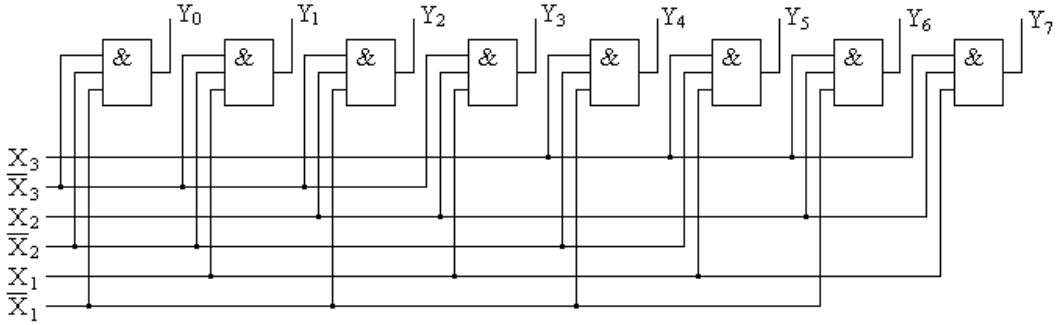
$$Y_4 = X_3 \cdot \overline{X_2} \cdot \overline{X_1}$$

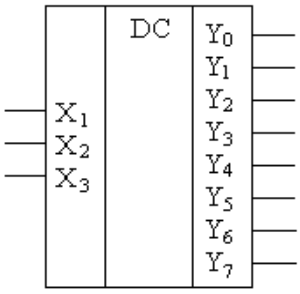
$$Y_5 = X_3 \cdot \overline{X_2} \cdot X_1$$

$$Y_6 = X_3 \cdot X_2 \cdot \overline{X_1}$$

$$Y_7 = X_3 \cdot X_2 \cdot X_1.$$

Структурная схема трехразрядного дешифратора, синтезированная на основании полученных ФАЛ приведена на рис. 4.10,*а*, а его УГО - на рис. 4.10,*б*.





б)

Рис. 4.10.Структурная схема и УГО трехразрядного дешифратора.

В общем случае логические уравнения для выходных переменных дешифратора *n*-разрядного числа имеют вид

$$Y_0 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot \overline{X_2} \cdot \overline{X_1}$$

$$Y_1 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot \overline{X_2} \cdot X_1$$

$$Y_2 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot X_2 \cdot \overline{X_1}$$
$$Y_3 = \overline{X_n} \cdot \overline{X_{n-1}} \cdot \_ \cdot X_2 \cdot X_1$$
.....
$$Y_{2^n-1} = X_n \cdot X_{n-1} \cdot \_ \cdot X_2 \cdot X_1.$$

Построенные по полученным формулам дешифраторы называются *линейными*. К преимуществу линейных дешифраторов можно отнести высокое быстродействие, поскольку входные переменные одновременно поступают на все элементы И. Одновременно, без дополнительных задержек, формируется и результат на выходах этих элементов. Очевидно, что для реализации линейного дешифратора *n*-разрядного числа необходимо иметь 2^n логических элементов И с *n*-входами. В существующих микросхемах логических элементов количество входов ограничено. Следовательно, ограничена и разрядность реализуемых на их основе линейных дешифраторов, что является недостатком. Кроме того, недостатком является и то, что предыдущие элементы, работающие на входы дешифратора, должны иметь высокую нагрузочную способность, т.е. должны быть рассчитаны на подключение большого числа логических элементов И. Каждый из входов дешифратора подключен к $0{,}5 \cdot 2^n$ логическим элементам И. Поскольку нагрузочная способность базовых логических элементов ИС не превышает величины N=10¸20, то максимальная разрядность дешифрируемых чисел для линейных дешифраторов n=4¸5.

Указанного недостатка лишены *пирамидальные дешифраторы*. Принцип построения этих дешифраторов состоит в том, что сначала строят линейный дешифратор для двухразрядного числа X_1, X_2, для чего необходимы 2^2=4 двухвходовые схемы И. Далее, каждая полученная конъюнкция логически умножается на входную переменную X_3 в прямой и инверсной форме. Полученная конъюнкция снова умножается на входную переменную X_4 в прямой и инверсной форме и т.д. Наращивая таким образом структуру, можно построить пирамидальный дешифратор на произвольное число входов. На рис. 4.11 приведена структура пирамидального дешифратора для трех разрядов.

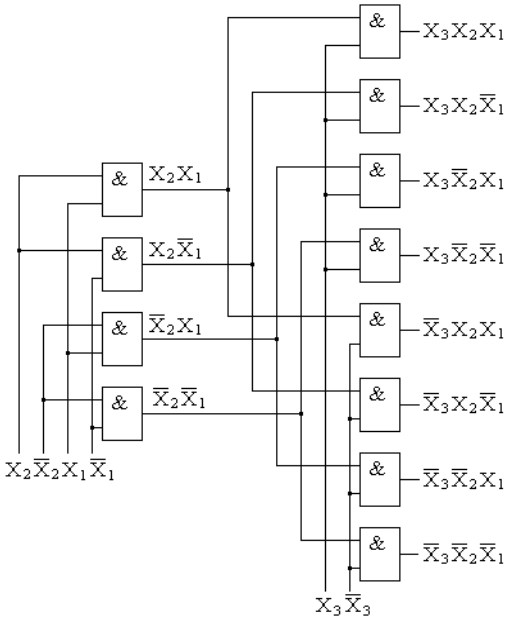


Рис. 4.11. Пирамидальный дешифратор для трехразрядного числа.

Характерным отличием пирамидальных дешифраторов от линейных является использование только двухвходовых логических элементов вне зависимости от разрядности дешифрируемого числа. В то же время количество логических элементов в пирамидальном дешифраторе больше. Однако следует иметь ввиду, что количество логических элементов, располагаемых в одном корпусе ИС, определяется главным образом требуемым количеством выводов. Следовательно, в одном корпусе ИС можно расположить большее число двухвходовых элементов, чем трехвходовых, четырехвходовых и т.д. И значит, пирамидальная структура дешифратора по числу корпусов ИС может оказаться более предпочтительной, чем линейная.

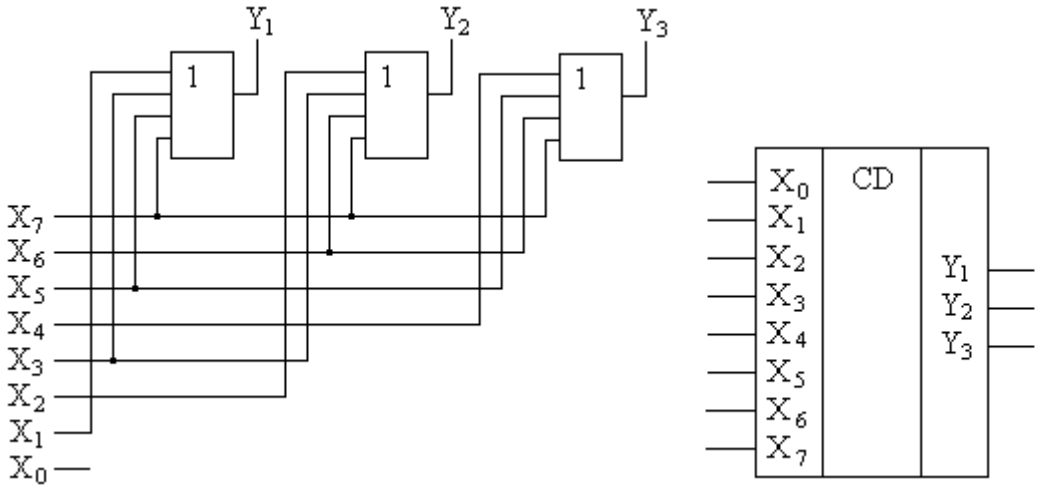
Шифраторы выполняют задачу обратную той, которую выполняют дешифраторы: появление логической единицы (логического нуля) на определенном входе приводит к появлению соответствующей кодовой комбинации на выходе. Также как и дешифраторы, шифраторы бывают полными и неполными. Работа восьмивходового полного шифратора задается следующей таблицей истинности:

Входы								**Выходы**		
X_7	X_6	X_5	X_4	X_3	X_2	X_1	X_0	Y_3	Y_2	Y_1
0	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
0	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**
0	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**1**	**0**
0	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**1**
0	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
0	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**1**
0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**0**
1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**

На основании таблицы истинности можно записать ФАЛ, задающие работу восьмивходового шифратора:

$$Y_1 = X_1 \vee X_3 \vee X_5 \vee X_7$$
$$Y_2 = X_2 \vee X_3 \vee X_6 \vee X_7$$
$$Y_3 = X_4 \vee X_5 \vee X_6 \vee X_7.$$

Синтезированная на основании приведенных логических уравнений структурная схема шифратора представлена на рис. 4.12,*а*, а его условное графическое обозначение – на рис. 4.12,*б*.



а) *б*)

Рис. 4.12.Структура и УГО восьмивходового шифратора.

Заключение.

Шифратор (кодер) преобразует сигнал на одном из входов в n-разрядное двоичное число. Функциональная схема шифратора, преобразующего десятичные цифры в 4-разрядное двоичное число, приведена на рисунке 1.33,а, а его условное обозначение – на рисунке 1.33,б. При появлении сигнала логической единицы на одном из десяти входов на четырех выходах шифратора будет присутствовать соответствующее двоичное число. Пусть сигнал логической единицы подан на вход 7. Тогда на выходах логических элементов DD1.1, DD1.2, DD1.3 будут сигналы логических единиц, а на выходе элемента DD1.4 – сигнал логического нуля. Таким образом, на выходах 8, 4, 2, 1 шифратора мы получим двоичное число 0111.

Некоторые из шифраторов снабжаются входом стробирования. Наличие входа стробирования позволяет выделять сигнал в определенный момент времени.

Дешифратор (декодер) преобразует код, поступающий на его входы, в сигнал только на одном из его выходов. Дешифратор n-разряд-ного двоичного числа имеет 2^n выходов. Функциональная схема дешифратора на 16 выходов приведена на рисунке 1.34,а. По такой функциональной схеме построена микросхема К155ИД3. Условное обозначение этой микросхемы на принципиальных схемах приведено на рисунке 1.34,б. Для преобразования сигнала необходимо на входы V1 и V2 микросхемы подать сигналы логических нулей.

Лабораторная работа №18

Триггеры и их виды.

Цель работы: Ознакомится с работой триггеры.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Триггерами называют большой класс электронных устройств, обладающих способностью длительно находиться в одном их двух или более устойчивых состояний и чередовать их под воздействием внешних сигналов.

Понятие «триггер» охватывает много устройств, которые существенно различаются между собой по выполняемым функциям, способам управления, по электрическим и конструктивным параметрам. Отличительной особенностью триггера как функционального устройства является свойство запоминания двоичной информации. Под памятью триггера подразумевают способность оставаться в одном из двух состояний после прекращения действия переключающего сигнала. Приняв одно из состояний за 1, а другое за 0, можно считать, что триггер хранит (помнит) один разряд числа, записанного в двоичном коде.

Триггеры подразделяются на две большие группы – динамические и статические. Названы они так по способу представления выходной информации. Динамические триггеры в настоящее время имеют ограниченное применение. К статическим триггерам относят устройства, каждое состояние которых характеризуется неизменными уровнями выходного напряжения (выходными потенциалами): высоким – близким к напряжению питания и низким – около нуля.

Статические (потенциальные) триггеры, в свою очередь, подразделяются на две неравные по практическому значению группы – симметричные и несимметричные триггеры. Оба класса реализуются на двухкаскадном усилителе с положительной обратной связью, а названием своим они обязаны способам организации внутренних электрических связей между элементами схемы.

Симметричные триггеры отличает симметрия схемы и по структуре, и по параметрам элементов обоих плеч. Для несимметричных триггеров

характернанеидентичность параметров элементов отдельных каскадов, а также и связей между ними.

Симметричные статические триггеры составляют основную массу триггеров, используемых в современной радиоэлектронной аппаратуре, и именно они рассматриваются в данном разделе.

Классификация триггеров проводится по признакам логического функционирования и способу записи информации.

По логическому функционированию различают триггеры типов RS, D, T, JK и др. Кроме того, используются комбинированные триггеры, в которых совмещаются одновременно несколько типов, и триггеры со сложной входной логикой (группами входов, связанных между собой логическими зависимостями).

В комбинированных триггерах совмещаются несколько режимов. Например, триггер типа RST — счетный триггер, имеющий также входы установки сброса.

По способу записи информации различают асинхронные (нетактируемые) и синхронные (тактируемые) триггеры. Будет ли триггер синхронным или асинхронным, зависит от схемы управляющего устройства.

У асинхронных триггеров имеются только информационные (логические) входы. Асинхронные триггеры отличает свойство срабатывать сразу после изменения сигналов на входах, не считая короткого времени задержки распространения в элементах, образующих триггер.

У синхронных триггеров смены сигналов на информационных входах еще недостаточно для срабатывания. Необходим дополнительный командный импульс, который подается на синхронизирующий, или, как его чаще называют, тактирующий вход.

Синхронизирующие (тактирующие) сигналы вырабатываются специальным генератором тактовых импульсов, которые и задают частоту смены информации в дискретном времени t^1, t^2, …, t^{n-1}, t^n, t^{n+1}. В эти моменты обновляется информация на выходах триггера, которая поступает на входы последующих устройств.

Синхронизация обеспечивает привязку ко времени и объединяет в общем ритме работу многих узлов аппаратуры, что позволяет во многих случаях ее существенно упростить.

Для асинхронного триггера тактом считается интервал времени между очередными сменами входных сигналов, причем длительность тактов не регламентируется. Асинхронные триггеры воспринимают непрерывный входной сигнал (1 или 0), независимо от его длительности, как один сигнал. В случае синхронного триггера входной сигнал неизменного уровня, длящийся n тактов, обрабатывается как последовательность нескольких n отдельных сигналов одного знака. Например, код 11001, поступающий последовательно на вход синхронного триггера, за счет тактирования будет так и воспринят. Этот же код на входах асинхронного триггера будет воспринят как 101, так как без временной привязки последовательность логических 1 или 0 не отличить от одиночных логических 1 и 0.

Синхронные триггеры, сравнительно с асинхронными, обладают также более высокой помехоустойчивостью. Опрокидывание синхронных триггеров происходит только при участии тактовых импульсов, длительность которых намного меньше периода следования. В остальное время входные сигналы, равно как и помехи различного происхождения, на триггер не влияют. В случае же асинхронного управления опрокидывание может произойти в любой момент времени, будь то полезный сигнал на входе или помеха.

По способу восприятия тактовых сигналов триггеры делятся на управляемые уровнем и управляемые фронтом. Управление уровнем означает, что при одном уровне тактового сигнала триггер воспринимает входные сигналы и реагирует на них, а при другом не воспринимает и остается в неизменном состоянии При управлении фронтом разрешение на переключение дается только в момент перепада тактового сигнала (на его фронте или спаде). В остальное время независимо от уровня тактового сигнала триггер не воспринимает входные сигналы и остается в неизменном состоянии. Триггеры, управляемые фронтом, называют также триггерами с динамическим управлением.

Динамический вход может быть прямым или инверсным. Прямое динамическое управление означает разрешение на переключение при изменении тактового сигнала с нулевого значения на единичное, инверсное – при изменении тактового сигнала с единичного значения на нулевое.

По характеру процесса переключения триггеры делятся на одноступенчатые и двухступенчатые.

В одноступенчатом триггере переключение в новое состояние происходит сразу, в двухступенчатом – по этапам. Двухступенчатые триггеры состоят из входной и выходной ступеней. Переход в новое состояние происходит в обеих ступенях поочередно. Один из уровней тактового сигнала разрешает прием информации во входную ступень при неизменном состоянии выходной ступени. Другой уровень тактового сигнала разрешает передачу нового состояния из входной ступени в выходную.

С синхронизацией (тактированием) триггера связаны два важных параметра — время предустановки tSU (Set-UpTime) и время выдержки tH (HoldTime). Важность этих параметров обуславливается еще и тем, что они свойственны не только триггерам, но и другим устройствам. Время tSU — это интервал до поступления синхросигнала, в течение которого информационный сигнал должен оставаться неизменным . Время выдержки tH — это время после поступления синхросигнала, в течение которого информационный сигнал должен оставаться неизменным. Соблюдение времен предустановки и выдержки обеспечивает правильное восприятие триггером входной информации.

Ряд других временных параметров триггеров непосредственно связан с задержками сигнала при прохождении через триггер и не требует специальных пояснений.

Заключение.

Словесное описание работы триггера занимает много места и лишено наглядности. Для случая триггера с несколькими независимыми входами оно становится еще неудобнее. Проще и нагляднее функциональная зависимость между сигналами на входах и выходах может быть показана другими способами:

Временные диаграммы.

Полная таблица переходов (ПТП).

Сокращенная таблица переходов (СТП).

Матрица переходов.

Граф переходов.

Характеристическая функция выходов.

Временные диаграммы особенно наглядны, когда количество независимых входных сигналов невелико. Помимо переключательных функций с их помощью в увеличенном временном масштабе можно показывать задержки фронтов, обусловленные переходными процессами в элементах схемы. Для устройств с несколькими входами и выходами временные диаграммы становятся громоздкими и теряют наглядность. Таблицы переходов (полные и сокращенные) позволяют представить в наглядном виде функциональную зависимость между входными и выходными сигналами в двух соседних тактах – t и t+1. Сокращенную таблицу переходов получают из полной за счет исключения из нее очевидных сведений, так что по сокращенной таблице можно восстановить полную. Обычно опускают столбец, характеризующий состояние выхода Q', т.к. оно обратно состоянию выхода Q. Нередко также в левой половине таблицы пропускают столбец состояния выхода Q(t) в такте t, но в правой, когда нужно, сопоставляют состояния выходов в соседних тактах t+1 и t.

Лабораторная работа № 19.

Тема: RS триггеры и их применение.

Цель работы: Изучение RS и JK триггеров и их применение.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

.

Теоретическая часть.

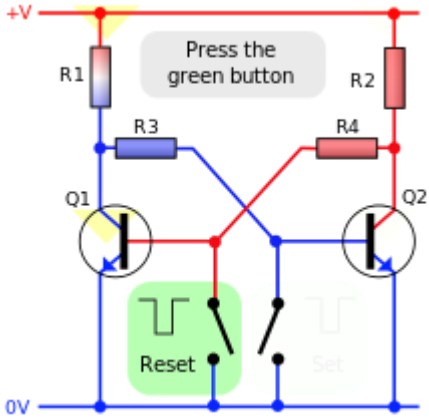
Триггер (триггерная система) — класс электронных устройств, обладающих способностью длительно находиться в одном из двух устойчивых состояний и чередовать их под воздействием внешних сигналов. Каждое состояние триггера легко распознаётся по значению выходного напряжения. По характеру действия

триггеры относятся к импульсным устройствам — их активные элементы (транзисторы, лампы) работают в ключевом режиме, а смена состояний длится очень короткое время.

Отличительной особенностью триггера как функционального устройства является свойство запоминания двоичной информации. Под памятью триггера подразумевают способность оставаться в одном из двух состояний и после прекращения действия переключающего сигнала. Приняв одно из состояний за «1», а другое за «0», можно считать, что триггер хранит (помнит) один разряд числа, записанного в двоичном коде.

При включении питания триггер непредсказуемо принимает (с равной или неравной вероятностью) одно из двух состояний. Это приводит к необходимости выполнять первоначальную установку триггера в требуемое исходное состояние, то есть подавать сигнал сброса на асинхронные входы триггеров, счётчиков, регистров, и т.д. (например, с помощью RC-цепочки), а также учитывать, что ячейки ОЗУ, построенного на триггерах (память статического типа), содержат после включения произвольную информацию.

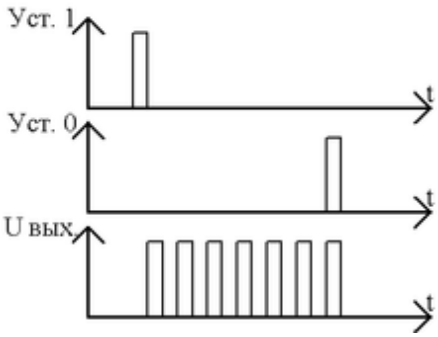
При изготовлении триггеров применяются преимущественно полупроводниковые приборы (обычно биполярные и полевые транзисторы), в прошлом — электромагнитные реле, электронные лампы. С появлением технологии производства микросхем малой и средней степени интеграции был освоен выпуск обширной номенклатуры триггеров в интегральном исполнении. В настоящее время логические схемы, в том числе с использованием триггеров, создают в интегрированных средах разработки под различные программируемые логические интегральные схемы (ПЛИС). Используются, в основном, в вычислительной технике для организации компонентов вычислительных систем: регистров, счётчиков, процессоров, ОЗУ.



RS-триггер
(R1, R2 = 1 kΩ, R3, R4 = 10 kΩ).

Триггеры подразделяются на две большие группы — динамические и статические. Названы они так по способу представления выходной информации.

Динамический триггер представляет собой управляемый генератор, одно из состояний которого (единичное) характеризуется наличием на выходе непрерывной последовательности импульсов определённой частоты, а другое (нулевое) — отсутствием выходных импульсов. Смена состояний производится внешними импульсами.



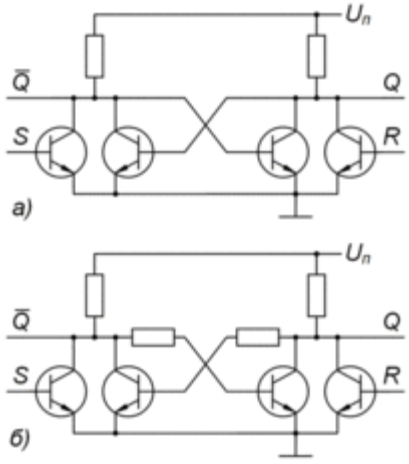
Временная диаграмма работы динамического триггера.

К статическим триггерам относят устройства, каждое состояние которых характеризуется неизменными уровнями выходного напряжения (выходными потенциалами): высоким — близким к напряжению питания и низким — около нуля. Статические триггеры по способу представления выходной информации часто называют потенциальными.

Статические (потенциальные) триггеры, в свою очередь, подразделяются на две неравные по практическому значению группы — симметричные и несимметричные триггеры. Оба класса реализуются на двухкаскадном двухинверторном усилителе с положительной обратной связью, а названием своим они обязаны способам организации внутренних электрических связей между элементами схемы.

Симметричные триггеры отличает симметрия схемы и по структуре, и по параметрам элементов обоих плеч. Для несимметричных триггеров характернанеидентичность параметров элементов отдельных каскадов, а также и связей между ними.

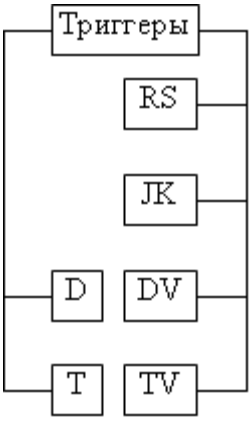
Симметричные статические триггеры составляют основную массу триггеров, используемых в современной радиоэлектронной аппаратуре. Схемы симметричных триггеров в простейшей реализации (2х2ИЛИНЕ) показаны на рисунке.



Симметричные триггеры: а) с непосредственной связью между каскадами; б) с резистивной связью

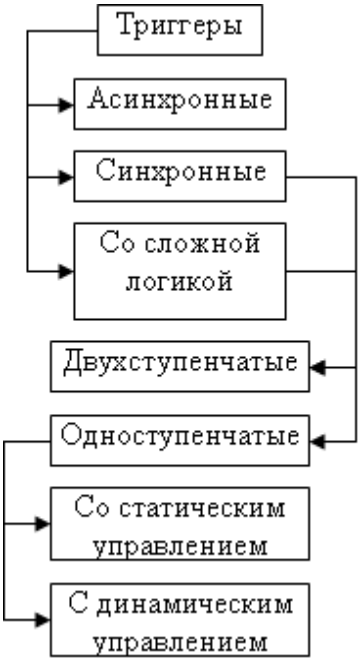
Основной и наиболее общий классификационный признак — функциональный — позволяет систематизировать статические симметричные триггеры по способу организации логических связей между входами и выходами триггера в определённые дискретные моменты времени до и после появления

входных сигналов. По этой классификации триггеры характеризуются числом логических входов и их функциональным назначением.



Функциональная классификация триггеров.

Вторая классификационная схема, независимая от функциональной, характеризует триггеры по способу ввода информации и оценивает их по времени обновления выходной информации относительно момента смены информации на входах.



Классификация триггеров по способу ввода информации

Каждая из систем классификации характеризует триггеры по разным показателям и поэтому дополняет одна другую. К примеру, триггеры RS-типа могут быть в синхронном и асинхронном исполнении.

Асинхронный триггер изменяет своё состояние непосредственно в момент появления соответствующего информационного сигнала(ов), с некоторой задержкой равной сумме задержек на элементах, составляющих данный триггер.

Синхронные триггеры реагируют на информационные сигналы только при наличии соответствующего сигнала на так называемом входе синхронизации С (от англ. clock). Этот вход также обозначают термином «такт». Такие информационные сигналы называют синхронными. Синхронные триггеры в свою

очередь подразделяют на триггеры со статическим и с динамическим управлением по входу синхронизации С.

Триггеры со статическим управлением воспринимают информационные сигналы при подаче на вход С логической единицы (прямой вход) или логического нуля (инверсный вход).

Триггеры с динамическим управлением воспринимают информационные сигналы при изменении (перепаде) сигнала на входе С от 0 к 1 (прямой динамический С-вход) или от 1 к 0 (инверсный динамический С-вход). Также встречается название «триггер управляемый фронтом».

Одноступенчатые триггеры (latch, защёлки) состоят из одной ступени представляющей собой элемент памяти и схему управления, бывают, как правило, со статическим управлением. Одноступенчатые триггеры с динамическим управлением применяются в первой ступени двухступенчатых триггеров с динамическим управлением. Одноступенчатый триггер на УГО обозначают одной буквой Т.

Двухступенчатые триггеры (flip-flop, шлёпающие) делятся на триггеры со статическим управлением и триггеры с динамическим управлением. При одном уровне сигнала на входе С информация, в соответствии с логикой работы триггера, записывается в первую ступень (вторая ступень заблокирована для записи). При другом уровне этого сигнала происходит копирование состояния первой ступени во вторую (первая ступень заблокирована для записи), выходной сигнал появляется в этот момент времени с задержкой равной задержке срабатывания ступени. Обычно двухступенчатые триггеры применяются в схемах, где логические функции входов триггера зависят от его выходов, во избежание временны́х гонок. Двухступенчатый триггер на УГО обозначают двумя буквами ТТ.

Триггеры со сложной логикой бывают также одно- и двухступенчатые. В этих триггерах наряду с синхронными сигналами присутствуют и асинхронные. Такой триггер изображён на рис. 1, верхний (S) и нижний (R) входные сигналы являются асинхронными.

Лабораторная работа № 20.

Тема: JK триггер.

Цель работы: Изучение JK триггера.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

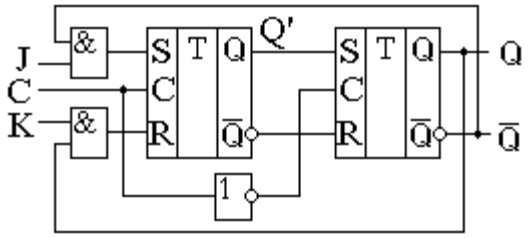
JK-триггер работает так же как RS-триггер, с одним лишь исключением: при подаче логической единицы на оба входа J и K состояние выхода триггера изменяется на противоположное, т.е. выполняется операция инверсии. Вход J (от англ. Jump — прыжок) аналогичен входу S у RS-триггера. Вход K (от англ. Kill — отключение) аналогичен входу R у RS-триггера. Таблица истинности jk триггера практически совпадает с таблицей истинности синхронного RS-триггера. Для того чтобы исключить запрещённое состояние, его схема изменена таким образом, что

при подаче двух единиц jk триггер превращается в счётный триггер. Это означает, что при подаче на тактовый вход C импульсов этот триггер изменяет своё состояние на противоположное. Таблица истинности jk триггера приведена в таблице.

Таблица истинности jk триггера.

C	K	J	Q(t)	Q(t+1)	Пояснения
0	x	x	0	0	Режим хранения информации
0	x	x	1	1	
1	0	0	0	0	Режим хранения информации
1	0	0	1	1	
1	0	1	0	1	Режим установки единицы J=1
1	0	1	1	1	
1	1	0	0	0	Режим записи нуля K=1
1	1	0	1	0	
1	1	1	0	1	K=J=1 счетный режим триггера
1	1	1	1	0	

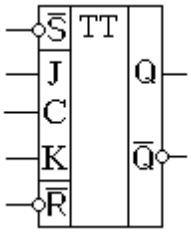
Один из вариантов схемы JK-внутренней триггера приведен на рисунке. Эта схема удобна для изучения принципов работы данного триггера в счетном режиме.



Внутренняя схема jk триггера

Для реализации счетного режима в схеме введена перекрестная обратная связь с выходов второго триггера на входы R и S первого триггера. Благодаря обратной связи на входах R и S первого триггера никогда не может возникнуть запрещенная комбинация, а то, что она перекрестная, вводит новый режим работы — счетный. При подаче на входы j и k логической единицы одновременно JK-триггер переходит в счетный режим, подобно T триггеру.

Приводить временные диаграммы работы JK-триггера не имеет смысла, так как они совпадают с приведёнными ранее временными диаграммами RS- и T-триггера. Условно-графическое обозначение JK-триггера приведено на рисунке.



Условно-графическое обозначение jk триггера

На этом рисунке приведено обозначение типовой цифровой микросхемы К1554ТВ9, выполненной по ТТЛ технологии. В промышленно выпускающихся микросхемах обычно кроме входов j и k реализуются входы RS-триггера, которые позволяют устанавливать jk триггер в заранее определённое исходное состояние.

В названиях отечественных микросхем для обозначения jk триггера присутствуют буквы ТВ. Например, микросхема К1554ТВ9 содержит в одном корпусе два jk триггера. В качестве примеров иностранных микросхем, содержащих jk триггеры можно назвать такие микросхемы, как 74HCT73 или 74ACT109.

Так как jk триггер является универсальной схемой, то рассмотрим несколько примеров ее использования. Начнем с примера использования этого триггера в качестве обнаружителя коротких импульсов.

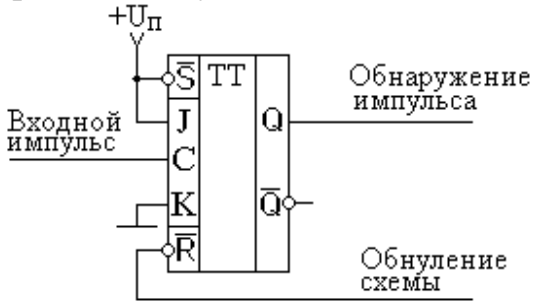


Схема обнаружения короткого импульса

В данной схеме при поступлении на вход "С" импульса триггер переходит в единичное состояние, которое затем может быть обнаружено последующей схемой (например, микропроцессором). Для того, чтобы привести схему в исходное состояние, необходимо подать на вход R уровень логического нуля.

Теперь рассмотрим пример построения на jk триггере ждущего мультивибратора. Один из вариантов подобной схемы приведен на рисунке.

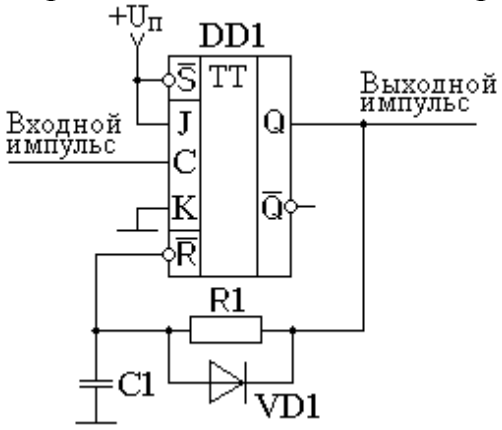


Схема ждущего мультивибратора, собранного на jk триггере.

Схема работает подобно предыдущей схеме. Длительность выходного импульса определяется постоянной времени RC цепочки. Диод VD1 предназначен для быстрого восстановления исходного состояния схемы (разряда емкости С). Если быстрое восстановление схемы не требуется, например, когда длительность выходных импульсов гарантированно меньше половины периода следования входных импульсов, то диод VD1 можно исключить из схемы ждущего мультивибратора.

В качестве последнего примера применения универсального jk триггера, рассмотрим схему счетного Т-триггера. Схема счетного триггера приведена на рисунке.

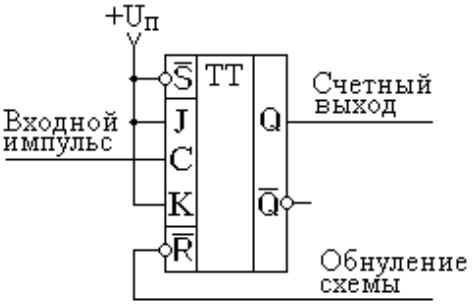


Схема счетного триггера, построенного на jk триггере.

В схеме, приведенной на рисунке 5, для реализации счетного режима работы триггера на входы J и K подаются уровни логической единицы. Если эти входы вывести в качестве отдельного входа, то они образуют отдельный вход разрешения счета T

Лабораторная работа № 21-22.

Тема: D и T триггеры и их применение.

Цель работы: Изучение D и T триггеров и их применение.

D-триггеры также называют триггерами задержки(от англ. Delay).

D-триггер синхронный.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

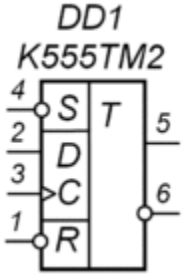
D	Q(t)	Q(t+1)
0	0	0
0	1	0
1	0	1
1	1	1

D-триггер (D от англ. delay — задержка, либо от data — данные) — запоминает состояние входа и выдаёт его на выход. D-триггеры имеют, как минимум, два входа: информационный D и синхронизации C. Вход синхронизации C может быть статическим (потенциальным) и динамическим. У триггеров со статическим входом C информация записывается в течение времени, при котором уровень сигнала C=1. В триггерах с динамическим входом C информация записывается только в течение перепада напряжения на входе C. Динамический вход изображают на схемах треугольником. Если вершина треугольника обращена в сторону микросхемы (прямой динамический вход), то триггер срабатывает по фронту входного импульса, если от неё (инверсный динамический вход) — по срезу импульса. В таком триггере информация на выходе может быть задержана на один такт по отношению к входной информации.Так как информация на выходе остаётся неизменной до прихода очередного импульса синхронизации, D-триггер называют также триггером с запоминанием информации или триггером-защёлкой. Рассуждая чисто

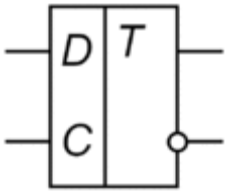
теоретически, парафазный (двухфазный) D-триггер можно образовать из любых RS- или JK-триггеров, если на их входы одновременно подавать взаимно инверсные сигналы.

D-триггер в основном используется для реализации защёлки. Так, например, для снятия 32 бит информации с параллельной шины, берут 32 D-триггера и объединяют их входы синхронизации для управления записью информации в защёлку, а 32 D входа подсоединяют к шине.

В одноступенчатых D-триггерах во время прозрачности все изменения информации на входе D передаются на выход Q. Там, где это нежелательно, нужно применять двухступенчатые (двухтактные, Master-Slave, MS) D-триггеры.



Пример условного графического обозначения (УГО) D-триггера с динамическим синхронным входом C и с дополнительными асинхронными инверсными входами S и R.



Условное графическое обозначение D-триггера со статическим входом синхронизации C.

D-триггер двухступенчатый.

В одноступенчатом триггере имеется одна ступень запоминания информации, при этом, в состоянии записи триггер "прозрачен", т.е. все изменения на входе триггера повторяются на выходе триггера, что может привести к ложным срабатываниям устройств стоящих после триггера. В двухступенчатом триггере две ступени. Вначале информация записывается в первую ступень, все изменения на входе триггера во вторую ступень до сигнала перезаписи не попадают, затем, после перехода D-триггера первой ступени в режим хранения, информация переписывается во вторую ступень и появляется на выходе, что позволяет избежать состояния "прозрачности". Двухступенчатый триггер обозначают ТТ. Если первая ступень двухступенчатого D-триггера выполнена на статическом D-триггере, то двухступенчатый D-триггер называют двухступенчатым D-триггером со статическим управлением, а если на динамическом D-триггере, то двухступенчатый D-триггер называют двухступенчатым D-триггером с динамическим управлением.

T-триггеры.

T-триггер (от англ. Toggle — переключатель) часто называют счётным триггером, так как он является простейшим счётчиком до 2.

T-триггер асинхронный.

Асинхронный Т-триггер не имеет входа разрешения счёта - Т и переключается по каждому тактовому импульсу на входе С.

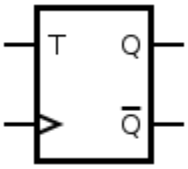
Т-триггер синхронный.

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

Синхронный Т-триггер, при единице на входе Т, по каждому такту на входе С изменяет своё логическое состояние на противоположное, и не изменяет выходное состояние при нуле на входе Т. Т-триггер можно построить на JK-триггере, на двухступенчатом (Master-Slave, MS) D-триггере и на двух одноступенчатых D-триггерах и инверторе.

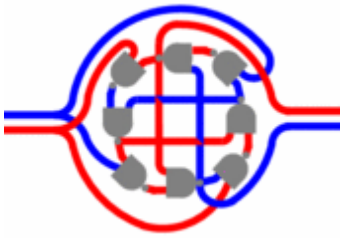
Как можно видеть в таблице истинности JK-триггера, он переходит в инверсное состояние каждый раз при одновременной подаче на входы J и K логической 1. Это свойство позволяет создать на базе JK-триггера Т-триггер, объединяя входы J и K.

В двухступенчатом (Master-Slave, MS) D-триггере инверсный выход Q соединяется со входом D, а на вход C подаются счётные импульсы. В результате триггер при каждом счётном импульсе запоминает значение Q, то есть будет переключаться в противоположное состояние.



Условное графическое обозначение (УГО) синхронного Т-триггера с динамическим входом синхронизации С на схемах.

Т-триггер часто применяют для понижения частоты в 2 раза, при этом на Т вход подают единицу, а на С — сигнал с частотой, которая будет поделена на 2.



Работа схемы асинхронного двухступенчатого Т-триггера с парафазным входом на двух парафазных D-триггерах на восьми логических вентилях 2И-НЕ. Слева — входы, справа — выходы. Синий цвет соответствует 0, красный — 1

Лабораторная работа № 23.

Тема: Счётчики. Виды счётчиков, принцип их работы и сфера применения.

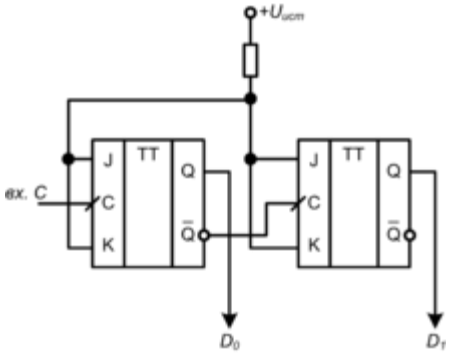
Цель работы: Изучить счётчики, виды счётчиков, принцип их работы и сфера применения.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Счётчик числа импульсов — устройство, на выходах которого получается двоичный (двоично-десятичный) код, определяемый числом поступивших импульсов. Счётчики могут строиться на двухступенчатых D-триггерах, T-триггерах и JK-триггерах.

Основной параметр счётчика — модуль счёта — максимальное число единичных сигналов, которое может быть сосчитано счётчиком. Счётчики обозначают через СТ (от англ. counter).



Двухразрядный двоичный асинхронный суммирующий счётчик с последовательной организацией переноса на JK-триггерах. Наклонная черточка на C-входе JK-триггеров указывает, что изменение состояния триггеров происходит по фронту сигнала.

Счётчики классифицируют:

1. по числу устойчивых состояний триггеров
- на двоичных триггерах
- на троичных триггерах
- на n-ичных триггерах
2. по модулю счёта:
- двоично-десятичные (декада);
- двоичные;
- с произвольным постоянным модулем счёта;
- с переменным модулем счёта;
3. по направлению счёта:
- суммирующие;
- вычитающие;
- реверсивные;
4. по способу формирования внутренних связей:
- с последовательным переносом;
- с ускоренным переносом;

- с параллельным ускоренным переносом;
- со сквозным ускоренным переносом;
- с комбинированным переносом;
- кольцевые;

5. по способу переключения триггера:

- синхронные;
- асинхронные;
- Счётчик Джонсона.

Двоичные счетчики.

Схему двоичного счетчика можно получить с помощью формального синтеза, однако более наглядным путём представляется эвристический. Таблица истинности двоичного счетчика — последовательность двоичных чисел от нуля до 2n-1, где n — разрядность счётчика. Наблюдение за разрядами чисел, составляющих таблицу, приводит к пониманию структурной схемы двоичного счетчика. Состояния младшего разряда при его просмотре по соответствующему столбцу таблицы показывают чередование нулей и единиц вида 01010101…, что естественно, так как младший разряд принимает входной сигнал и переключается от каждого входного воздействия. В следующем разряде наблюдается последовательность пар нулей и единиц вида 00110011… . В третьем разряде образуется последовательность из четверок нулей и единиц 00001111… и т. д. Из этого наблюдения видно, что следующий по старшинству разряд переключается с частотой, в два раза меньшей, чем данный.

Известно, что счетный триггер делит частоту входных импульсов на два. Сопоставив этот факт с указанной выше закономерностью, видим, что счетчик может быть построен в виде цепочки последовательно включенных счетных триггеров. Заметим, кстати, что согласно ГОСТу входы элементов изображаются слева, а выходы справа. Соблюдение этого правила ведет к тому, что в числе, содержащемся в счетчике, младшие разряды расположены левее старших.
Счетчики с последовательно-параллельным переносом.

В связи с ограничениями на построение счетчиков с параллельным переносом большой разрядности широкое распространение получили счетчики с групповой структурой, или счетчики с последовательно-параллельным переносом. Разряды таких счетчиков разбиваются на группы, внутри которых организуется принцип параллельного переноса. Сами же группы соединяются последовательно с использованием конъюнкторов, формирующих перенос в следующую группу при единичном состоянии всех триггеров предыдущих. При единичном состоянии всех триггеров группы приход очередного входного сигнала создаст перенос из этой группы. Эта ситуация подготавливает межгрупповойконъюнктор к прямому пропусканию входного сигнала на следующую группу.

В наихудшем для быстродействия случае, когда перенос проходит через все группы и поступает на вход последней,

$t_{УСТ} = t \cdot (Í — 1) + t_{ГР}$,

где Í — число групп, $t_{ГР}$ — время установления кода в группе.

В развитых сериях ИС обычно имеется по 5…10 вариантов двоичных счетчиков, выполненных в виде четырёхразрядных групп (секций). Каскадирование секций может выполняться путём их последовательного включения по цепям переноса, организации параллельно-последовательных переносов или для более сложных счетчиков с двумя дополнительными управляющими входами разрешения счета и разрешения переноса путём организации параллельных переносов и в группах, и между ними.

Особенностью двоичных счетчиков синхронного типа является наличие ситуаций с одновременным переключением всех его разрядов (например, для суммирующего счетчика при переходе от кодовой комбинации 11…1 к комбинации 00…0 при переполнении счетчика и выработке сигнала переноса). Одновременное переключение многих триггеров создает значительный токовый импульс в цепях питания ЦУ и может привести к сбою в их работе. Поэтому в руководящих материалах по использованию некоторых БИС/СБИС программируемой логики, в частности, имеется ограничение на разрядность двоичных счетчиков заданной величиной k. При необходимости применения счетчика большей разрядности рекомендуется переходить к коду Грея, для которого переходы от одной кодовой комбинации к другой сопровождаются переключением всего одного разряда. Правда, для получения результата счета в двоичном коде придется использовать дополнительно преобразователь кода, но это является платой за избавление от токовых импульсов большой интенсивности в цепях питания.

Лабораторная работа № 24.

Тема: Микросхемы постоянной памяти.

Цель работы: Изучить микросхемы постоянной памяти.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Постоянная память - электронная память для долговременного хранения программ и данных.

В постоянной памяти хранится информация, записанная на предприятии-изготовителе, она должна быть неизменна в течение длительного времени.

По типу исполнения.

Массив данных совмещён с устройством выборки (считывающим устройством), в этом случае массив данных часто в разговоре называется «прошивка»:

микросхема ПЗУ;

Один из внутренних ресурсов однокристальной микро ЭВМ (микроконтроллера), как правилоFlashROM.

Массив данных существует самостоятельно:

Поле магнитных ячеек
Компакт-диск;
перфокарта;
перфолента;
Штрих-коды;
монтажные «1» и монтажные «0».
По разновидностям микросхем ПЗУ.
По технологии изготовления кристалла:
ROM — (англ. read-onlymemory, постоянное запоминающее устройство), масочное ПЗУ, изготавливается фабричным методом. В дальнейшем нет возможности изменить записанные данные.
PROM — (англ. programmableread-onlymemory, программируемое ПЗУ (ППЗУ)) — ПЗУ, однократно «прошиваемое» пользователем.
EPROM — (англ. erasableprogrammableread-onlymemory, перепрограммируемое/репрограммируемое ПЗУ (ППППЗУ/РПЗУ)). Например, содержимое микросхемы К573РФ1 стиралось при помощи ультрафиолетовой лампы. Для прохождения ультрафиолетовых лучей к кристаллу в корпусе микросхемы было предусмотрено окошко с кварцевым стеклом.
EEPROM — (англ. electricallyerasableprogrammableread-onlymemory, электрическистираемоеперепрограммируемоеПЗУ).Память такого типа может стираться и заполняться данными несколько десятков тысяч раз. Используется в твердотельных накопителях. Одной из разновидностей EEPROM является флеш-память (англ. flashmemory).
ПЗУ на магнитных доменах, например К1602РЦ5, имело сложное устройство выборки и хранило довольно большой объём данных в виде намагниченных областей кристалла, при этом не имея движущихся частей (см. Компьютерная память). Обеспечивалось неограниченное количество циклов перезаписи.
NVRAM, non-volatilememory — «неразрушающаяся» память, строго говоря, не является ПЗУ. Это ОЗУ небольшого объёма, конструктивно совмещённое с батарейкой. В СССР такие устройства часто назывались «Dallas» по имени фирмы (англ.), выпустившей их на рынок. В NVRAM современных ЭВМ батарейка уже конструктивно не связана с ОЗУ и может быть заменена.
По виду доступа:
С параллельным доступом (parallelmode или randomaccess): такое ПЗУ может быть доступно в системе в адресном пространстве ОЗУ. Например, К573РФ5;
С последовательным доступом: такие ПЗУ часто используются для однократной загрузки констант или прошивки в процессор или ПЛИС, используются для хранения настроек каналов телевизора, и др. Например, 93С46, AT17LV512A.
По способу программирования микросхем (записи в них прошивки):
Непрограммируемые ПЗУ;
ПЗУ, программируемые только с помощью специального устройства — программатора ПЗУ (как однократно, так и многократно прошиваемые). Использование программатора необходимо, в частности, для подачи нестандартных и относительно высоких напряжений (до +/- 27 В) на специальные выводы.

Внутрисхемно (пере)программируемые ПЗУ (ISP, in-systemprogramming) — такие микросхемы имеют внутри генератор всех необходимых высоких напряжений, и могут быть перепрошиты без программатора и даже без выпайки из печатной платы, программным способом.

Микросхема масочного ПЗУ NEC D23128C в компьютере ZX Spectrum

Микросхема EPROM Intel 1702 с ультрафиолетовым стиранием.

Микросхема EPROM AMD AM2716 выпущенная в 1979 году.

В постоянную память часто записывают микропрограмму управления техническим устройством: телевизором, сотовым телефоном, различными контроллерами, или компьютером (BIOS или OpenBoot на машинах SPARC).

BootROM — прошивка, такая, что если её записать в подходящую микросхему ПЗУ, установленную в сетевой карте, то становится возможна загрузка операционной системы на компьютер с удалённого узла локальной сети. Для встроенных в ЭВМ сетевых плат BootROM можно активировать через BIOS.

ПЗУ в IBM PC-совместимых ЭВМ располагается в адресном пространстве с F600:0000 по FD00:0FFF

Постоянные запоминающие устройства стали находить применение в технике задолго до появления ЭВМ и электронных приборов. В частности, одним из первых типов ПЗУ был кулачковый валик, применявшийся в шарманках, музыкальных шкатулках, часах с боем.

С развитием электронной техники и ЭВМ возникла необходимость в быстродействующих ПЗУ. В эпоху вакуумной электроники находили применение ПЗУ на основе потенциалоскопов, моноскопов, лучевых ламп. В ЭВМ на базе транзисторов в качестве ПЗУ небольшой ёмкости широко использовались штепсельные матрицы. При необходимости хранения больших объёмов данных (для ЭВМ первых поколений — несколько десятков килобайт) применялись ПЗУ на базе ферритовых колец (не следует путать их с похожими типами ОЗУ). Именно от этих типов ПЗУ и берёт своё начало термин «прошивка» — логическое состояние ячейки задавалось направлением навивки провода, охватывающего кольцо. Поскольку тонкий провод требовалось протягивать через цепочку ферритовых колец для выполнения этой операции применялись металлические иглы, аналогичные швейным. Да и сама операция наполнения ПЗУ информацией напоминала процесс шитья.

Лабораторная работа № 25.

Тема: Микросхемы оперативной памяти.

Цель работы: Изучить микросхемы оперативной памяти.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

Оперативная память (англ. RandomAccessMemory, RAM, память с произвольным доступом) или операти́вноезапомина́ющееустро́йство (ОЗУ); комп. жарг. па́мять, операти́вка — энергозависимая часть системы компьютерной памяти, в которой во время работы компьютера хранится выполняемый машинный код (программы), а также входные, выходные и промежуточные данные, обрабатываемые процессором.

Обмен данными между процессором и оперативной памятью производится:

- непосредственно;
- через сверхбыструю память 0-го уровня — регистры в АЛУ, либо при наличии аппаратного кэша процессора — через кэш.

Содержащиеся в современной полупроводниковой оперативной памяти данные доступны и сохраняются только тогда, когда на модули памяти подаётся напряжение. Выключение питания оперативной памяти, даже кратковременное, приводит к искажению либо полному разрушению хранимой информации.

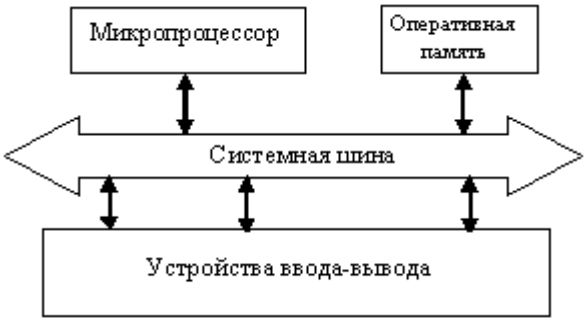
Энергосберегающие режимы работы материнской платы компьютера позволяют переводить его в режим сна, что значительно сокращает уровень потребления компьютером электроэнергии. В режиме гибернации питание ОЗУ отключается. В этом случае для сохранения содержимого ОЗУ операционная система (ОС) перед отключением питания записывает содержимое ОЗУ на устройство постоянного хранения данных (как правило, жёсткий диск). Например, в ОС Windows XP содержимое памяти сохраняется в файл hiberfil.sys, в ОС семейства Unix — на специальный swap-раздел жёсткого диска.

В общем случае, ОЗУ содержит программы и данные ОС и запущенные прикладные программы пользователя и данные этих программ, поэтому от объёма оперативной памяти зависит количество задач, которые одновременно может выполнять компьютер под управлением ОС.

Оперативное запоминающее устройство, ОЗУ — техническое устройство, реализующее функции оперативной памяти.

ОЗУ может изготавливаться как отдельный внешний модуль или располагаться на одном кристалле с процессором, например, в однокристальных ЭВМ или однокристальных микроконтроллерах.

Модули ОЗУ для ПК.



Простейшая схема взаимодействия оперативной памяти с ЦП.

ОЗУ большинства современных компьютеров представляет собой модули динамической памяти, содержащие полупроводниковые ИС ЗУ, организованные по принципу устройств с произвольным доступом. Память динамического типа дешевле, чем статического, и её плотность выше, что позволяет на той же площади кремниевого кристалла разместить больше ячеек памяти, но при этом её быстродействие ниже. Статическая память, наоборот, более быстрая память, но она и дороже. В связи с этим основную оперативную память строят на модулях динамической памяти, а память статического типа используется для построения кэш-памяти внутри микропроцессора.

Память динамического типа.

Экономичный вид памяти. Для хранения разряда (бита или трита) используется схема, состоящая из одного конденсатора и одного транзистора (в некоторых вариантах два конденсатора). Такой вид памяти, во-первых, дешевле (один конденсатор и один транзистор на 1 бит дешевле нескольких транзисторов триггера), и, во-вторых, занимает меньшую площадь на кристалле (там, где в SRAM размещается один триггер, хранящий 1 бит, можно разместить несколько конденсаторов и транзисторов для хранения нескольких бит). Но DRAM имеет и недостатки. Во-первых, работает медленнее, поскольку, если в SRAM изменение управляющего напряжения на входе триггера сразу очень быстро изменяет его состояние, то для того, чтобы изменить состояние конденсатора, его нужно зарядить или разрядить. Перезаряд конденсатора гораздо более длителен (в 10 и более раз), чем переключение триггера, даже если ёмкость конденсатора очень мала. Второй существенный недостаток — конденсаторы со временем разряжаются. Причём разряжаются они тем быстрее, чем меньше их электрическая ёмкость и больше ток утечки, в основном, утечка через ключ.

Именно из-за того, что заряд конденсатора динамически уменьшается во времени, память на конденсаторах получила своё название DRAM — динамическая память. Поэтому, дабы не потерять содержимое памяти, заряд конденсаторов

периодически восстанавливается («регенерируется») через определённое время, называемое циклом регенерации (обычно 2 мс). Для регенерации в современных микросхемах достаточно выполнить циклограмму «чтения» по всем строкам запоминающей матрицы. Процедуру регенерации выполняет процессор или контроллер памяти. Так как для регенерации памяти периодически приостанавливается обращение к памяти, это снижает среднюю скорость обмена с этим видом ОЗУ.

Память статического типа.

ОЗУ, которое не надо регенерировать (обычно схемотехнически выполненное в виде массива триггеров), называют статической памятью с произвольным доступом или просто статической памятью. Достоинство этого вида памяти — скорость. Поскольку триггеры являются соединением нескольких логических вентилей, а время задержки на вентиль очень мало, то и переключение состояния триггера происходит очень быстро. Данный вид памяти не лишён недостатков. Во-первых, группа транзисторов, входящих в состав триггера, обходится дороже, чем ячейка динамической памяти, даже если они изготавливаются групповым методом миллионами на одной кремниевой подложке. Кроме того, группа транзисторов занимает гораздо больше площади на кристалле, чем ячейка динамической памяти, поскольку триггер состоит минимум из 2 вентилей (шести-восьми транзисторов), а ячейка динамической памяти — только из одного транзистора и одного конденсатора. Используется для организации сверхбыстродействующего ОЗУ, обмен информацией с которым критичен для производительности системы.

Лабораторная работа №26.

Тема: Принципы построения логических устройств.

Цель работы: Ознакомится с принципами построения логических устройств.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

Теоретическая часть.

При проектировании сложных электронных устройств используется принцип декомпозиции задачи. Он сводится к последовательной разработке структурной, функциональной и принципиальной схемы устройства. Цифровое устройство реализуется аппаратными средствами в виде совокупности интегральных микросхем комбинационного и последовательностного типов.

Схема электрическая структурная (код схемы Э1) определяет основные функциональные части устройства, их назначение и взаимосвязь. Используется для общего ознакомления с изделием.

Схема электрическая функциональная (код схемы Э2) разъясняет процессы, протекающие в отдельных функциональных частях изделия или в изделии в

целом. Используется для изучения принципов работы устройства, а также при наладке, контроле, ремонте.

Схема электрическая принципиальная (код схемы Э3) определяет полный состав элементов и связей между ними и дает детальное представление о принципах работы изделия. Она служит исходным документом при разработке других конструкторских документов (печатных плат, сборочных чертежей, схем соединений и т.п.).

Разрешается разрабатывать совмещенные схемы, когда на схемах одного типа изображают фрагменты схем других типов.

Большой практический интерес представляют цифровые устройства, реализующие некоторый алгоритм обработки информации, т.е. выполняющие упорядоченную последовательность определенных операций над поступающими данными. При построении таких устройств целесообразно использовать принцип микропрограммного управления, состоящий в следующем:

· любая операция, реализуемая устройством, рассматривается как сложное действие, которое разделяется на последовательность элементарных действий, называемых микрооперациями;

· для управления порядком следования операций используются оповестительные сигналы — логические условия, принимающие значения 1 или 0 в зависимости от результата выполнения микроопераций;

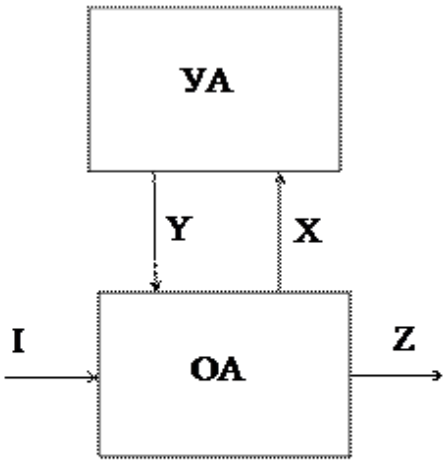


Рис. 1.1 — Структурная схема цифрового автомата

процесс выполнения операций в устройстве описывается в виде алгоритма, представленного в терминах микроопераций и логических условий и называемого микропрограммой;· микропрограмма дает путь к определению структуры устройства, его реализации на выбираемой элементной базе.При использовании описанного принципа принято делить цифровое устройство (рис. 1.1) на операционный автомат (ОА) и управляющий автомат

(УА).Операционный автомат предназначен для хранения поступающей информации I, выполнения заданного набора микроопераций, выработки логических условий X и выходных сигналов Z.Управляющий автомат генерирует последовательность управляющих сигналов Y в соответствии с заданной программой и значениями логических условий X.Цифровое устройство реализуется аппаратно-программны-ми средствами с использованием микропроцессорных комплектов интегральных схем.Микропроцессор (МП) — выполненное в виде большой интегральной схемы (БИС) цифровое устройство, предназначенное для обработки информации в соответствии с хранимой в памяти программой. Он реализует принцип микропрограммного управления и содержит на кристалле основные элементы операционного и управляющего автомата. Уровень микрокоманд часто скрыт от пользователя, который разрабатывает программу работы микропроцессора на уровне команд. Но полезно помнить, что каждая команда выполняется за определенное число тактов (микрокоманд). Вместе с памятью и устройствами ввода/вывода информации МП образует микропроцессорную систему.Микропроцессорные системы можно разделить на микроЭВМ и микроконтроллеры. Микроконтроллеры — специализированные устройства с программой, зашитой в ПЗУ, выполняющие задачи управления в реальном масштабе времени. МикроЭВМ — более универсальные устройства с развитыми средствами диалогового общения с человеком (клавиатура, дисплей и т.п.), легко перестраиваемые на решение новых задач. В изучаемой дисциплине основное внимание уделяется встроенным микропроцессорным системам управления на базе микроконтроллеров. Применение однокристальных микроконтроллеров в устройствах бытовой и медицинской электроники, в устройствах управления технологическим оборудованием, преобразователями электрической энергии, в измерительных приборах обеспечивает достижение исключительно высоких показателей эффективности при низкой стоимости.Любой микроконтроллер содержит центральный процессор, память и интерфейс ввода/вывода (рис. 1.2). ПЗУ хранит основную программу, подпрограммы, таблицы, константы. ОЗУ используется для хранения результатов промежуточных вычислений, массивов данных, поступающих от датчиков, либо подготовленных к выдаче внешним устройствам. Генератор тактовых импульсов (ГТИ) синхронизирует работу всей микропроцессорной системы. Интерфейс (ИФ) используется для сопряжения с внешними устройствами (ВУ) по временным и электрическим параметрам и представляет собой набор шин (портов), специальных сигналов и алгоритмов обмена

информацией.

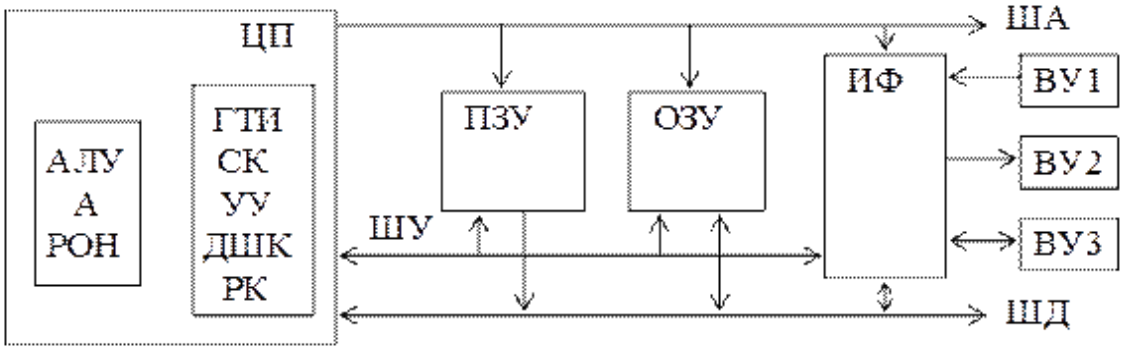


Рис. 1.2 — Структурная схема микропроцессорной системы

Основу центрального процессора (ЦП) составляет арифметико-логическое устройство (АЛУ), позволяющее выполнять арифметические, логические операции и операции сдвига над данными, представленными в двоичном коде. В состав операционной части входят также регистры общего назначения (РОН) и основной рабочий регистр — аккумулятор (А).

Заключение.

В процессе развития интегральной электроники выделилось несколько типов схем логических элементов, имеющих достаточно хорошие характеристики и удобных для реализации в интегральном исполнении, которые служат элементной базой современных цифровых микросхем.

Базовые элементы, независимо от их микросхемотехники и особенностей технологий изготовления, строятся в одном из базисов (как правило, в базисе **И–НЕ** или **ИЛИ–НЕ**).

Базовые элементы выпускаются в виде отдельных микросхем, либо входят в состав функциональных узлов и блоков, реализованных в виде СИС, БИС, СБИС.

В процессе реализации базовые логические элементы строят из двух частей: входной логики, выполняющей операции **И** или **ИЛИ**, и выходного каскада, выполняющего операцию **НЕ.**

Входная логика может быть выполнена на диодах, биполярных и полевых транзисторах. В зависимости от этого различают:

- транзисторно-транзисторную логику (ТТЛ, ТТЛШ),
- интегральную инжекционную логику (ИИЛ, $И^2Л$),
- логику на МДП-транзисторах (МДП, МОП),
- МОП-транзисторная логика на комплементарных транзисторах (КМОП-логика).

В перечисленных группах логических элементов в качестве выходного каскада используется ключевая схема (инвертор). Другая группа логических элементов основана на переключателях тока – эмиттерно-связанная логика (ЭСЛ-логика).

В основе схемы ЭСЛ лежит переключатель тока, в одно из плеч которого включено параллельно несколько транзисторов. Эти транзисторы равноправны – отпирание любого из них (или всех вместе) приводит к изменению логического состояния переключателя. Поэтому ЭСЛ-элементы выполняют логическую функцию **ИЛИ-НЕ.**

Вследствие ненасыщенного режима работы транзисторов логический перепад в схеме не превышает 0,65В.

Лабораторная работа №27.

Тема: Основы проектирования цифровых устройств.
Построение комбинационных схем используя логических элементов.

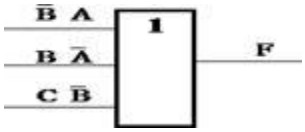
Цель работы: Изучить и научиться Основы проектирования цифровых устройств.

Необходимые принадлежности: Персональный компьютер, программное обеспечение BloodshedDevC++, вертуальная лаборатория ElectronicWorkbench, принтер.

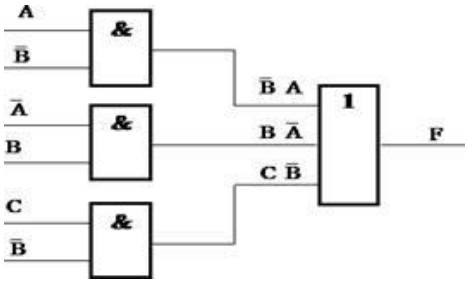
Теоретическая часть.

Комбинационные схемы - это схемы, у которых выходные сигналы Y = (y1, y2, ..., ym) в любой момент дискретного времени однозначно определяются совокупностью входных сигналов X = (x1, x2,..., xn), поступающих в тот же момент времени t. Реализуемый в КС способ обработки информации называется комбинационным потому, что результат обработки зависит только от комбинации входных сигналов и формируется сразу при поступлении входных сигналов. Поэтому одним из достоинств комбинационных схем является их высокое быстродействие. Преобразование информации однозначно описывается логическими функциями вида Y=f(X). Логические функции и соответствующие им комбинационные схемы подразделяют на регулярные и нерегулярные структуры. Регулярные структуры предполагают построение схемы таким образом, что каждый из ее выходов строится по аналогии с предыдущими. В нерегулярных структурах такая аналогия отсутствует. Многие регулярные структуры положены в основу построения отдельных ИС малой и средней степени интеграции или отдельных функциональных частей БИС и СБИС. Из регулярных комбинационных схем наиболее распространены дешифраторы, шифраторы, схемы сравнения, комбинационные сумматоры, коммутаторы и др.

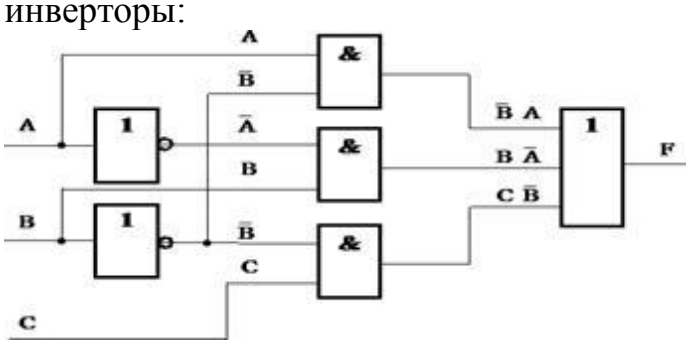
Для построения любой КС необходима таблица истинности ее функционирования (составляется или задается), затем составляется функция зависимости каждого выхода схемы от входа (в форме СДНФ, которую затем можно перевести в упрощенную форму) и производится построение схемы на определенных логических элементах (чаще всего на И-НЕ и ИЛИ-НЕ). Как правило, построение и расчет любой схемы осуществляется начиная с ее выхода. Допустим задано булево выражение : $F = \bar{B}A + B\bar{A} + C\bar{B}$. Первый этап: выполняется логическое сложение (т.е. логическая операция ИЛИ), считая входными переменными функции $\bar{B}A$, $B\bar{A}$, $C\bar{B}$.



Второй этап: к входам элемента ИЛИ подключаются логические элементы И, входными переменными которых являются уже A, B, C и их инверсии:



Третий этап: для получения инверсий $\bar{A}$ и $\bar{B}$ на соответствующих входах ставят инверторы:



Как видно из построения, любые логические функции могут быть представлены как аргументы других более сложных функций, и наоборот: любую сколь угодно сложную функцию можно представить как совокупность стандартных функций.

ИНФОРМАЦИОННО – МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ.

1. Основные источники

1. Гусев.В.Г.,Гусев Ю.М. Электроника . М. Высшая школа. 1992.
1. Манаев.Е.И. Основы радиоэлектроники. М. Радто и связь. 1995.
2. Забродин Ю.С. Промышленная электроника М. Высшая школа 1982.
3. Малахов В.П. Электронные цепи непрырывного и импульсного дейстия Киев Лыбидь 1991
4. Горбачёв Т.Н.,Чаплыгин Е.Е Промышленная электроника М. Энергоатомиздат 1988.
5. Ерофеев Ю.Н. Импульсные и цифровые устройства М. Высшая школа. 1989
6. Фролкин Л.Г. Импульсные и цифровые устройства М. Высшая школа. 1991
7. Краснопрошина А.А.Скаржепа В.А. Электроника и микросхемотехника. Киев Высшая школа. 1989.

2. Дополнительные источники

1. Гусев.В.Г.,Гусев Ю.М. Электроника . М. Высшая школа. 1992.
8. Манаев.Е.И. Основы радиоэлектроники. М. Радто и связь. 1995.
9. Забродин Ю.С. Промышленная электроника М. Высшая школа 1982.
10. Малахов В.П. Электронные цепи непрырывного и импульсного дейстия Киев Лыбидь 1991
11. Горбачёв Т.Н.,Чаплыгин Е.Е Промышленная электроника М. Энергоатомиздат 1988.
12. Ерофеев Ю.Н. Импульсные и цифровые устройства М. Высшая школа. 1989
13. Фролкин Л.Г. Импульсные и цифровые устройства М. Высшая школа. 1991
14. Краснопрошина А.А.Скаржепа В.А. Электроника и микросхемотехника. Киев Высшая школа. 1989.

3. Информационно-ресурсные источники

1. http.://www./sxem .net
2. http.://www./ziyo.net
3. http.://www./ techno. Edu.ru/db/sect/111