Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический
университет имени П. О. Сухого»

Кафедра «Промышленная электроника»

Е. А. Храбров, Ю. Е. Котова

ЦИФРОВАЯ ЭЛЕКТРОНИКА

УЧЕБНОЕ ПОСОБИЕ

Электронный аналог печатного издания

Гомель 2013

УДК 621.397(075.8)
ББК 32.859я73
Х88

Рецензенты: зав. каф. электроники БГУИР канд. техн. наук,
доц. *С. В. Дробот*;
канд. техн. наук, доц. каф. «Электронные вычислительные
машины и системы» БрГТУ *В. И. Хведчук*

Храбров, Е. А.

Х88 Цифровая электроника : учебное пособие / Е. А. Храбров, Ю. Е. Котова ; М-во образования Респ. Беларусь, Гомель : ГГТУ им. П. О. Сухого, 2013. – 271 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: http://library.gstu.by. – Загл. с титул. экрана.

ISBN 978-985-535-222-9.

Содержатся необходимые сведения для изучения курса «Цифровая электроника» по следующим вопросам: системы счисления и способы кодирования, применяемые в цифровой электронике; основы булевой алгебры; способы минимизации логических схем; методы анализа и синтеза комбинационных, последовательностных и импульсных устройств на современной элементной базе.

Для студентов специальности 1-36 04 02 «Промышленная электроника» дневной и заочной форм обучения.

УДК 621.397(075.8)
ББК 32.859я73

ISBN 978-985-535-222-9

ВВЕДЕНИЕ

Учебным планом подготовки по специальности 1-36 04 02 «Промышленная электроника» установлена дисциплина «Цифровая электроника». Цель дисциплины – изучение студентами теоретических основ анализа и синтеза электронных цифровых и импульсных устройств на интегральных цифровых микросхемах малой и средней степени интеграции. В результате студенты должны усвоить следующие вопросы:

1. Сигналы цифровых устройств.
2. Математические основы цифровой электроники.
3. Элементная база цифровой электроники.
4. Функциональные устройства комбинационного типа.
5. Функциональные устройства последовательностного типа.
6. Генераторные устройства.
7. Цифровые устройства промышленной электроники.
8. Запоминающие устройства (ЗУ).
9. Аналого-цифровые и цифро-аналоговые преобразователи.

В настоящем учебном пособии излагаются основные теоретические сведения в соответствии с программой курса.

В пособии описываются элементарные логические функции и показано, как с помощью этих простых функций можно получить достаточно сложные выражения, описывающие требующиеся по техническим заданиям цифровые устройства.

Ввиду различий в условных графических обозначениях логических элементов, встречающихся в технической литературе, в настоящем пособии приводятся четыре варианта этих обозначений:

– отечественные в соответствии с ГОСТ 2.743–82;

– старые зарубежные обозначения, которые до настоящего времени встречаются в иностранной технической литературе;

– обозначения в соответствии с американской системой MILSPEC;

– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией (МЭК) и широко распространенной особенно среди европейских производителей электронной техники.

Актуальность использования устройств цифровой электроники

Все большее число приложений в электронике, как, впрочем, и большинство других технологий, используют цифровую технику, чтобы осуществлять операции, которые когда-то выполнялись при помощи аналоговой техники. Основные причины роста популярности цифровых технологий заключаются в следующем.

Цифровые системы, *как правило*, *легче разрабатывать*. Так происходит потому, что используемые схемы принадлежат к *ключевым схемам*, в которых важны не точные значения напряжения или тока, а лишь диапазон (*высокий* или *низкий* сигнал), в который они попадают.

Легко осуществить хранение информации. Хранение осуществляется с помощью специальных устройств и схем, которые могут считывать цифровую информацию и сохранять ее сколь угодно долго. Запоминающие устройства сверхбольшой емкости могут хранить миллиарды бит информации на сравнительно малом физическом пространстве. Аналоговые устройства хранения информации, наоборот, имеют крайне ограниченные возможности.

Большая точность. Цифровые системы могут оперировать любым необходимым количеством десятичных знаков путем простого увеличения числа ключевых схем. В аналоговых системах точность обычно ограничена тремя или четырьмя знаками, потому что значения тока или напряжения непосредственно зависят от номиналов компонент схемы и подвержены влиянию случайных флуктуаций напряжения (шумов).

Возможность запрограммировать действие. Достаточно легко спроектировать цифровые системы, в которых работа контролируется набором хранящихся команд, или программой. Аналоговые системы также можно программировать, но разнообразие и сложность имеющихся в распоряжении операций строго ограничены.

Цифровые схемы менее подвержены шумам. Паразитные флуктуации напряжения (шумы) некритичны для цифровых систем, потому что точные значения напряжения для них не столь важны (шум не настолько большой, чтобы нельзя было отличить *высокий* уровень сигнала от *низкого*). *Большее количество схемотехнических решений может быть изготовлено на интегральных схемах* (ИС). Конечно, и аналоговая схемотехника выиграла от бурного развития ИС-технологий, но ее относительная сложность, а также использование компонентов, которые не могут быть эффективно интегрированы (высокоемкие конденсаторы, прецизионные резисторы, катушки индуктивности, трансформаторы), помешали аналоговым системам добиться такой же высокой степени интеграции.

Достоинства и недостатки цифровой электроники

Когда сигналы, с которыми приходится иметь дело, либо являются непрерывными по самой своей природе (например, звуковые),

либо представляют собой непрерывно меняющееся напряжение, поступающее от измерительных приборов (например, от устройств для измерения температуры или светового излучения, биологических или химических зондов), то естественным является применение для их обработки аналоговых устройств.

Входной сигнал по своей природе может быть и чисто дискретным, например импульсы в детекторе частиц или «биты» информации, поступающие от ключа, клавиатуры или ЭВМ. В подобных случаях естественно и удобно использовать цифровую электронику.

Но все чаще и аналоговые сигналы стараются обрабатывать цифровыми устройствами. Причины, обусловливающие переход от аналоговых систем к цифровым, весьма разнообразны. Пожалуй, самым очевидным достоинством цифровых систем является снижение требований, предъявляемых к выходному сигналу двоичной схемы, по сравнению с выходом аналоговой схемы.

Информация в аналоговой схеме представлена в виде величины напряжения, и любые отклонения от требуемого значения, даже самые малые, приводят к потере, искажению информации. С двоичной схемой связан всего лишь один бит информации, значение которого определяется тем, в каком из двух возможных состояний находится выход схемы.

Рассмотрим передачу фиксированного количества информации, например 20 бит, посредством аналогового сигнала. Каждый бит информации позволяет сделать выбор между двумя равновероятными вариантами; поэтому с использованием 20 бит информации можно представить $2^{20} = 1048576$ возможных вариантов. Таким образом, весь диапазон выходного сигнала делят на 1048576 равных частей и, определяя, в какой части этого диапазона находится аналоговый выходной сигнал, выделяют 20 бит информации.

Для диапазона напряжений 0...10 В приращение (шаг квантования) составляет 10 мкВ, и сконструировать реальную схему с такой точностью в пределах обычно существующих помех и диапазона изменения рабочих температур практически невозможно или по крайней мере неоправданно дорого. Один из вариантов решения проблемы заключается в использовании 20 двоичных схем, каждый выход которых несет 1 бит информации. Хотя это и ведет к увеличению числа используемых схем, зато стоимость каждой такой схемы, а в результате и всего устройства, значительно снижается. Очевидно, что при обработке большого количества информации из-за разницы в стоимости предпочтение отдается двоичным, цифровым схемам.

Примером этого может служить переход к цифровым способам управления в бытовых телевизионных приемниках. Здесь практически все параметры телевизионного изображения и звука регулируются в настоящее время дискретно, устанавливаясь на одном из 64 допустимых уровней.

Управляющий сигнал поступает в телевизор, как правило, через систему дистанционного управления с помощью инфракрасного канала, передающего кодированные цифровые данные о кнопках, нажатых на переносном пульте для изменения того или иного параметра в телевизоре. В самом телевизоре значения всех параметров запоминаются в цифровом запоминающем устройстве и хранятся в нем сколь угодно долго. Такой стабильности и сохранности выбранных параметров настройки очень трудно было бы добиться аналоговыми средствами.

Другим не менее наглядным примером вытеснения аналоговых способов обработки информации цифровыми является широкое распространение так называемых компакт-дисков с лазерным считывающим устройством для высококачественной записи и воспроизведения музыки. Таким образом, магнитофоны и проигрыватели граммофонных пластинок заменяются CD-ROM проигрывателями.

Применяемые в них сложные алгоритмы устранения шумов и помех в воспроизводимых записях не приводят к существенному повышению стоимости этих устройств, так как вся цифровая обработка информации производится всего несколькими микросхемами большой интеграции (а порой и одной микросхемой), цена которых несравнимо ниже стоимости всего проигрывателя. Значительные первоначальные затраты на разработку вышеупомянутых алгоритмов и реализующих их микросхем сравнительно быстро окупились за счет массового производства проигрывателей.

Использование цифровой оперативной памяти для запоминания текущего фрагмента записи с целью устранения искажений, вносимых устройством считывания при тряске и вибрациях для носимых и возимых CD-ROM проигрывателей, позволяет добиться высококачественного воспроизведения надежными и недорогими средствами.

Можно считать, что цифровые способы обработки сигналов предпочтительнее применять в тех случаях, когда:

– входные сигналы имеют дискретный характер;

– требуется сложная и нелинейная обработка сигналов при массовом производстве устройств обработки сигналов.

ГЛАВА 1. СИГНАЛЫ ЦИФРОВЫХ УСТРОЙСТВ

1.1. Импульсные, цифровые, логические сигналы и устройства

1.1.1. Классификация радиотехнических сигналов

Применяемые в современной радиоэлектронике сигналы можно разделить на следующие классы:

– произвольные по величине и непрерывные по времени (рис. 1.1, *а*);

– произвольные по величине и дискретные по времени (рис. 1.1, *б*);

– квантованные по величине и непрерывные по времени (рис. 1.1, *в*);

– квантованные по величине и дискретные по времени (рис. 1.1, *г*).

Сигналы первого класса (рис. 1.1, *а*) иногда называют *аналоговыми*, так как их можно толковать как электрические модели физических величин, или непрерывными, так как они задаются по оси времени на несчетном множестве точек. Такие множества называются континуальными. При этом по оси ординат сигналы могут принимать любое значение в определенном интервале. Поскольку эти сигналы могут иметь разрывы, как на рис. 1.1, *а*, то, чтобы избежать некорректности при описании, лучше такие сигналы обозначать термином *континуальный*.

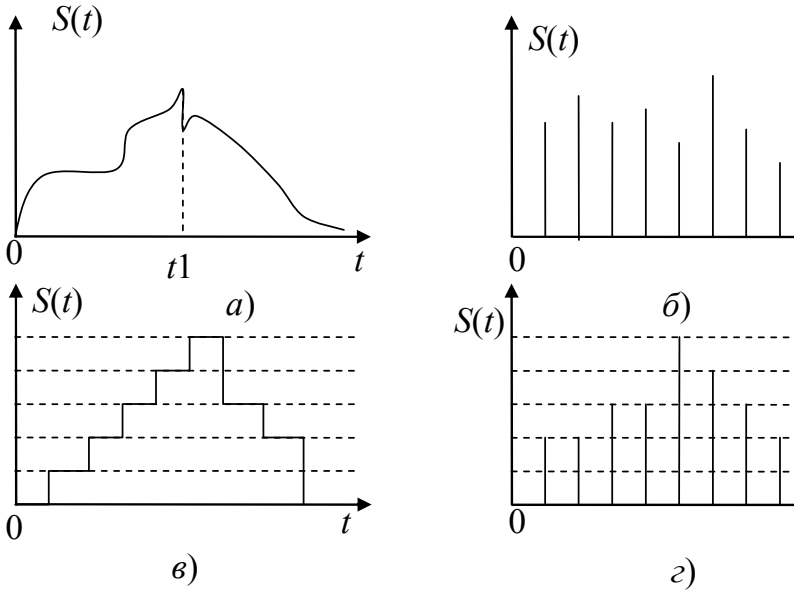


Рис. 1.1. Сигналы: *а* – произвольные по величине и по времени; *б* – произвольные по величине и дискретные по времени; *в* – квантованные по величине и непрерывные по времени; *г* – квантованные по величине и дискретные по времени

Итак, *аналоговый* сигнал $s(i)$ является функцией непрерывной переменной t, а *дискретный* сигнал $s(x)$ – функцией дискретной переменной x, принимающей только фиксированные значения. Дискретные сигналы могут создаваться непосредственно источником информации (например, дискретными датчиками в системах управления или телеметрии) или образовываться в результате дискретизации континуальных сигналов.

На рис. 1.1, *б* представлен сигнал, заданный при дискретных значениях времени t (на счетном множестве точек); величина же сигнала в этих точках может принимать любое значение в определенном интервале по оси ординат (как и на рис. 1.1, *а*). Таким образом, термин *дискретный* характеризует не сам сигнал, а способ задания его на временной оси.

Сигнал на рис. 1.1, *в* задан на всей временной оси, однако его величина может принимать лишь дискретные значения. В подобных случаях говорят о сигнале, *квантованном по уровню*. В дальнейшем термин *дискретный* будет применяться только по отношению к дискретизации по времени; дискретность же по уровню будет обозначаться термином *квантование*.

Дискретные сигналы, уровни которых могут принимать лишь счетное множество значений, называют *квантованными* сигналами.

Квантование используют при представлении сигналов в цифровой форме с помощью цифрового кодирования, поскольку уровни можно пронумеровать числами с конечным числом разрядов. Поэтому дискретный по времени и квантованный по уровню сигнал (рис. 1.1, *г*) называют *цифровым*.

Таким образом, можно различать континуальные (рис. 1.1, *а*), дискретные (рис. 1.1, *б*), квантованные (рис. 1.1, *в*) и цифровые (рис. 1.1, *г*) сигналы.

1.1.2. Основные процессы преобразования: дискретизация, квантование, кодирование

В современной радиоэлектронике широко распространены системы, в которых осуществляется *дискретизация* сигнала, например, при использовании импульсных методов передачи сообщения в радиосвязи. В системах с цифровой обработкой исходный континуальный сигнал преобразуется в дискретный сигнал. Выбор шага (темпа) T дискретизации производится на основании теоремы отсчетов.

Процедуру *дискретизации* (взятие выборок), осуществляемую с помощью электронного ключа, удобно рассматривать как умножение

функции $s(t)$ на вспомогательную периодическую последовательность $y(t)$ достаточно коротких тактовых импульсов. В качестве таких импульсов обычно рассматривают прямоугольные импульсы с длительностью τ_0, малой по сравнению с T. Таким образом, дискретизованный с шагом T сигнал можно определить выражением $s_T(t) = s(t)y_T(t)$.

Функции $s(t)$, $y_T(t)$ и $S_T(t)$ показаны на рис. 1.2.

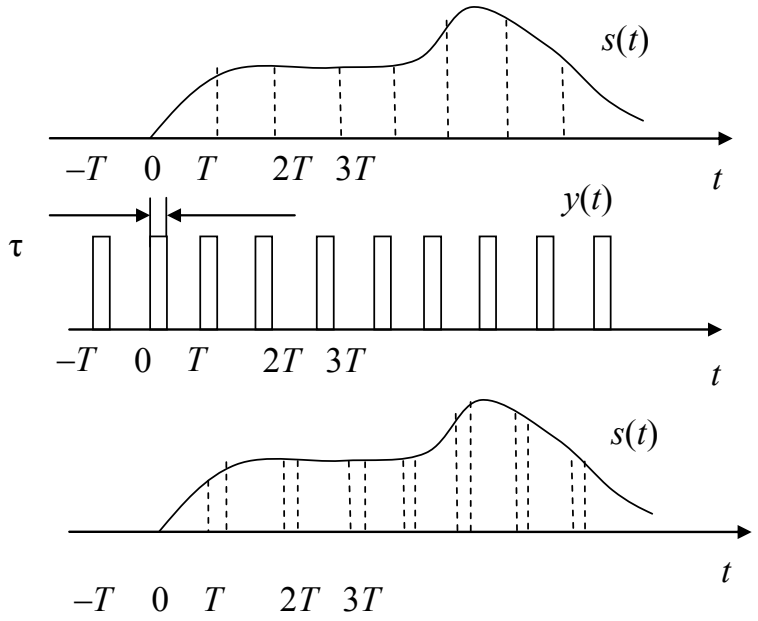


Рис. 1.2. Дискретизация сигнала как умножение на последовательность тактовых импульсов конечной длительности

1.1.3. Принцип аналого-цифрового преобразования информации

Цифровая система – это комбинация устройств, разработанных для обработки логической информации или физических величин, которые представлены в цифровой форме. Эти устройства чаще всего электронные, но могут быть механическими, магнитными или пневматическими. Наиболее распространенные цифровые системы в мире – это цифровые компьютеры и калькуляторы, цифровое оборудование по обработке аудио- и видеоданных, телефонные системы.

Аналоговая система содержит устройства, которые оперируют с физическими величинами, представленными в аналоговой форме. В аналоговой системе амплитуда выходного сигнала в колонке радиоприемника может иметь любое значение между нулем и максимальным пределом. Другие обычные аналоговые системы – это уси-

лители звука, устройства записи и воспроизведения на магнитной ленте, обычный плавный (реостатный) выключатель света.

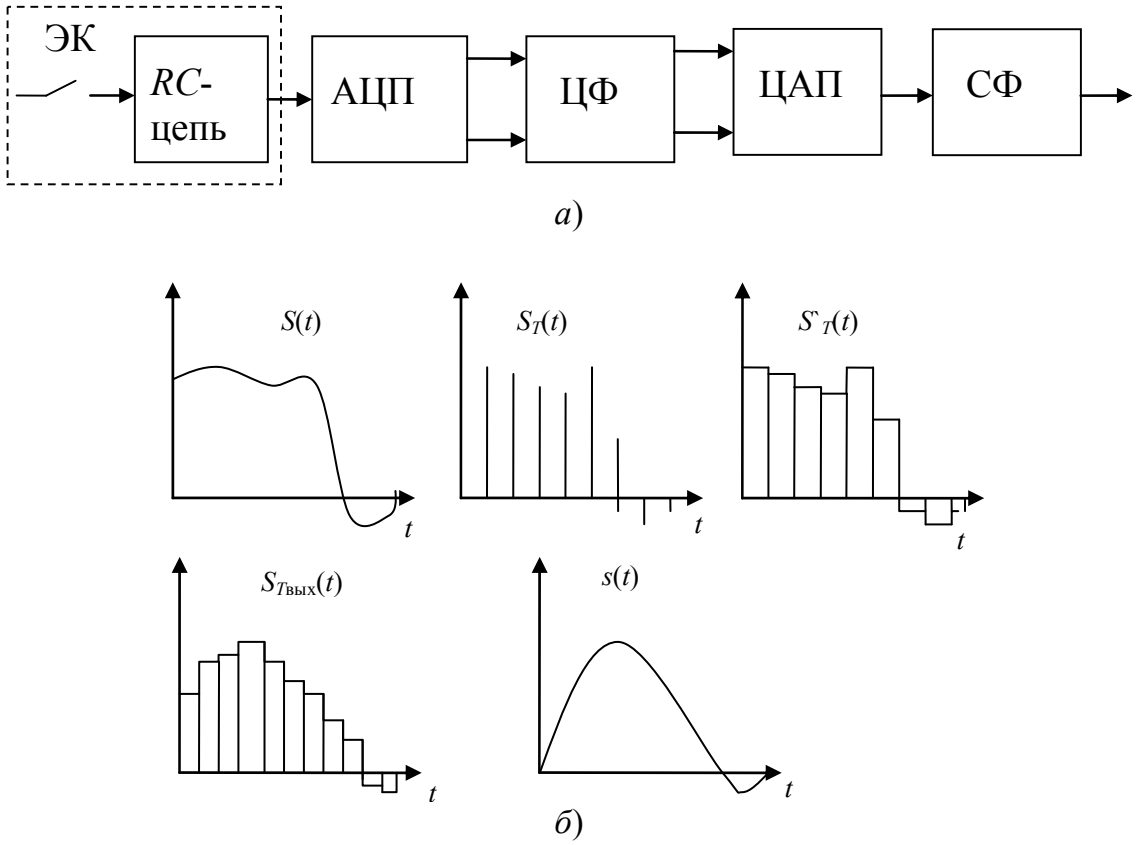


Рис. 1.3. Принцип аналогово-цифрового и цифро-аналогового преобразования на примере функциональной схемы цифрового фильтра:
а – схема цифрового фильтра; *б* – эпюры колебаний сигнала

Общее представление о принципе цифровой обработки континуального сигнала можно получить из схемы, изображенной на рис. 1.3, *а*, на котором также даны эпюры колебаний в различных точках схемы (рис. 1.3, *б*). Входной сигнал $s(t)$ подвергается сначала дискретизации по времени с помощью электронного ключа (ЭК), работающего с шагом T.

Сигнал $S_T(t)$ на выходе ЭК имеет вид последовательности равноотстоящих коротких импульсов, являющихся выборками (отсчетами) сигнала $s(t)$. Предполагается, что при выборе шага τ обеспечивается сохранение информации, содержащейся в континуальном сигнале $s(t)$.

Каждый отсчет запоминается в интегрирующей *RC*-цепи на время, необходимое для срабатывания аналого-цифрового преобразователя (АЦП). Это время должно быть не больше шага T. В результате на выходе *RC*-цепи получается ступенчатое колебание $S_T`(t)$. В АЦП каждый отсчет квантуется по уровню и преобразуется в кодо-

вое слово – двоичное число, составленное из r разрядов, каждый из которых представлен нулем или единицей (паузой или стандартным импульсом).

Квантование заключается в том, что отсчет измеряется и ему присваивается один уровень из общего числа возможных. Это число равно $2r$. Например, при $r = 10$ получается $2^{10} = 1024$ уровня. Каждому разряду соответствует своя шина, так что на выходе АЦП закодированный цифровой отсчет представлен в виде комбинации из бинарных чисел (пауз и импульсов), возникающих на r выходных шинах одновременно (параллельный код). Максимально возможному значению отсчета соответствует кодовое слово, составленное из r импульсов, нулевому значению отсчета – слово из r пауз. Точность представления отсчета тем выше, чем длиннее кодовое слово, т. е. чем больше в нем бинарных чисел.

Последовательность закодированных цифрами отсчетов поступает в цифровой фильтр (ЦФ), представляющий собой вычислительное устройство, в котором над кодовыми словами производятся определенные математические операции (сложение, умножение, а также задержка во времени), соответствующие заданному алгоритму. В результате этих операций на выходе ЦФ возникают новые кодовые слова, соответствующие профильтрованному сигналу.

В цифро-аналоговом преобразователе (ЦАП) каждое кодовое слово приводит в действие группу электронных ключей, которые управляют суммированием эталонных напряжений, соответствующих каждому из разрядов. В результате на выходе ЦАП воспроизводятся отсчеты в аналоговой форме.

Такое декодирование является процессом, обратным происходящему в АЦП.

Напряжение на выходе ЦАП $s_{T\text{вых}}(t)$ имеет ступенчатую форму, причем высота каждой ступени равна отсчету выходного сигнала в соответствующий момент времени.

Наконец, в четырехполюснике, который можно назвать синтезирующим фильтром (СФ), осуществляется преобразование дискретной последовательности в континуальный выходной сигнал $s_{bbix}(t)$.

Очевидно, что перечисленные выше преобразования, производимые над каждым отсчетом входного сигнала, должны выполняться за время, меньшее шага T. Кроме того, должна обеспечиваться строгая синхронность управления электронными ключами, используемыми для осуществления поразрядного сложения, вычитания и других операций над кодовыми словами.

Все это приводит к необходимости применения сложной системы синхронизации вспомогательных импульсных последовательностей, с помощью которых на каждом шаге T обеспечиваются стирание старой информации в двоичных элементах (например, в триггерах) и ввод в них новой информации.

1.2. Цифровые и логические сигналы

1.2.1. Параметры импульсных сигналов

К *импульсным сигналам* в общем случае принято относить различные электрические сигналы – периодические и непериодические, отличные по форме от гармонических (синусоидальных). Вопросами генерации, преобразования, передачи и применения импульсных сигналов занимается импульсная техника – важная область современной электронной техники. Форму различных импульсов для удобства уподобляют близким по виду геометрическим фигурам. Импульсная техника оперирует с прямоугольными, трапецеидальными, треугольными, пилообразными, ступенчатыми и другими видами импульсов. Для практики наибольшее значение имеют прямоугольные импульсы.

Понятие «*прямоугольный импульс*» – идеализированное, колебательные выбросы, обусловленные паразитными емкостями и индуктивностями, могут отсутствовать.

Параметры импульсов подразделяют на *электрические* и *временные* [1].

Электрические параметры характеризуют *полярность* импульса и его *рабочий уровень*. Поскольку большинство серий современных цифровых микросхем питается положительным относительно общего провода напряжением, полярность электрических импульсов в устройствах с такими микросхемами положительна (иначе говоря, мгновенное значение импульсного напряжения всегда больше нуля). Операционные усилители и некоторые типы компараторов напряжения питаются двуполярным напряжением, такие микросхемы пригодны для работы с разнополярными импульсами.

Микросхемы, питающиеся отрицательным напряжением, составляют небольшую часть изделий микроэлектронной промышленности и имеют ограниченное применение.

Импульсное напряжение может принимать то высокий, то низкий уровень. Уровень высокого напряжения обозначают U^1, низкого – U^0.

Амплитуда U_a характеризует наибольшее из мгновенных значений импульса относительно начального уровня.

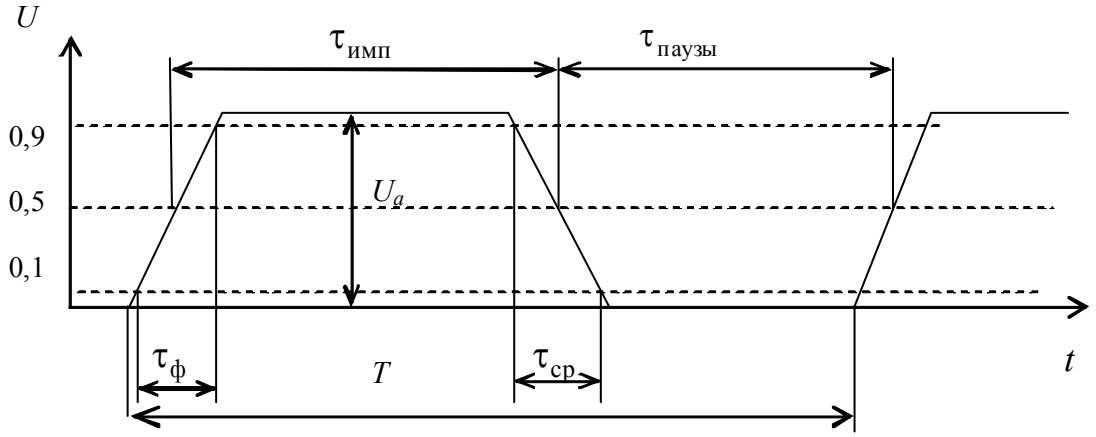


Рис. 1.4. Параметры импульса

Перепадом (напряжения, тока) называют быстрое изменение электрического состояния, соизмеримое по длительности с временем переходных процессов в цепи. Положительный перепад (от низкого уровня к высокому) называют *фронтом*, в таблицах и на схемах его изображают знаком _|‾ (реже ↑), а в тексте – 0.1. Отрицательный перепад (от высокого уровня к низкому) принято называть *срезом* (*спадом*, *задним фронтом*). В таблицах и на схемах срез обозначают знаком ‾|_ (реже ↓), а в тексте – 1.0.

Переходный процесс в реальном устройстве занимает какое-то время, зависящее от инерционных свойств активных элементов, а также от паразитных емкостей. Поэтому в действительности в длительность импульса входит время фронта $\tau_ф$ и среза $\tau_{ср}$.

На практике длительность фронта и среза импульса определяют с помощью осциллографа. Поскольку у реального импульса трудно точно выделить моменты начала и конца процесса, длительностью фронта и среза условно считают время изменения напряжения импульса от $0{,}1U_a$ до $0{,}9U_a$ или от $0{,}9U_a$ до $0{,}1U_a$.

В литературе можно также встретить термины «*крутизна фронта*» и «*крутизна среза*»:

$$K_ф = \frac{0{,}9U_a - 0{,}1U_a}{\tau_ф} = \frac{0{,}8U_a}{\tau_ф},\ \text{В/мкс},$$

$$K_{\text{ср}} = \frac{0{,}9U_a - 0{,}1U_a}{\tau_{\text{ср}}} = \frac{0{,}8U_a}{\tau_{\text{ср}}},\ \text{В/мкс}.$$

Длительность импульса, так называемую активную длительность, отсчитывают на уровне $0{,}5U_a$. При рассмотрении процессов генерирования и формирования импульсов длительность фронта и среза не учитывают, для простоты полагая ее равной нулю.

Импульсы, следующие в определенном порядке, образуют импульсную последовательность. В периодической последовательности импульсы и паузы между ними одинаковы. Сумма длительностей импульса и паузы $\tau_{\text{паузы}}$ определяет период следования T (рис. 1.4). Обратная величина $f = 1/T$ характеризует частоту следования импульсов.

Для характеристики периодической последовательности используют и дополнительные параметры: скважность импульсов $Q = T/\tau_{\text{имп}}$ и реже коэффициент заполнения $\gamma = \tau_{\text{имп}} /T$. Периодическую последовательность, у которой $\tau_{\text{имп}} = \tau_{\text{паузы}}$, т. е. $Q = 2$, называют меандром.

1.2.2. Формы представления цифровых сигналов

Множество допустимых уровней дискретного сигнала может быть любым, однако есть один самый распространенный тип дискретного сигнала, у которого возможны только два значения: ВЫСОКИЙ (HIGH, 1) и НИЗКИЙ (LOW, 0) уровни. Такой сигнал называют *двоичным*, а зачастую *цифровым* (строго говоря, цифровой сигнал может иметь не только два уровня).

В современных цифровых устройствах логические состояния представляются двумя уровнями напряжения (потенциалов): *высоким*, близким к напряжению источника питания, и *низким*, близким к нулю. Это так называемая потенциальная система представления информации, для которой характерны непосредственная связь между отдельными элементами схемы. Длительность потенциальных сигналов определяется частотой смены информации, а переключающими импульсами служат перепады напряжения от одного уровня к другому.

Два уровня напряжения, характеризующие логические состояния, определяются просто как более высокий H (англ. high – высокий) и низкий L (low – низкий). Эти два значения называют логическими уровнями. Существуют два вида так называемых соглашений в зависимости от того, каким уровнем напряжения кодировать логическую 1 (и, соответственно, логический 0). В соглашении положительной логики более высокий уровень напряжения (H) соответствует логической 1, а низкий – логическому 0. В соглашении отрицательной логики – наоборот.

Двоичные (цифровые) сигналы можно подразделить на числоимпульсные, последовательные и параллельные позиционные сигналы.

Самыми простыми двоичными сигналами являются числоимпульсные сигналы, у которых информационным параметром является количество импульсов N за время заданного постоянного интервала времени или за время между двумя определенными метками времени, причем ни длительность импульсов, ни их расположение при этом не имеют значения. На рис. 1.5 показан пример числоимпульсного двоичного сигнала, в котором отметками начала и конца интервалов, в которых производят подсчет числа импульсов сигнала, служат достаточно длительные паузы между пачками импульсов. Такой вид может иметь сигнал набора номера в телефонных аппаратах с импульсным набором, которые распространены в нашей стране.

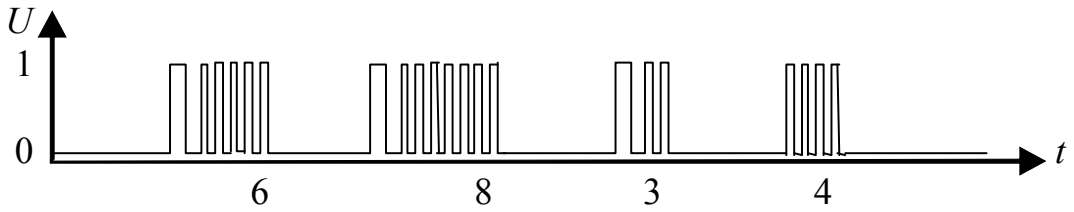


Рис. 1.5. Пример числоимпульсного двоичного сигнала, содержащего 6, 8, 3 и 4 импульса, соответственно, в первой, второй, третьей и четвертой посылках

Числоимпульсные сигналы настолько просты, что, как правило, когда говорят о двоичных или цифровых сигналах, подразумевают при этом один из двух позиционных видов двоичного сигнала – последовательный или параллельный двоичный сигнал, для которых уже имеет значение и длительность, и положение каждого единичного импульса.

В последовательном двоичном сигнале информация заключена во временной последовательности появления нулей и единиц внутри своей группы, приходящейся на определенное число тактов дискретного времени. В каждой такой группе первая позиция обычно соответствует самому младшему разряду, имеет самый малый вес, а все последующие позиции (номера дискретного времени внутри группы) имеют последовательно нарастающие веса вплоть до последней позиции с максимальным весом, соответствующей старшему разряду. Поскольку в двоичном сигнале имеется только два допустимых значения, то все веса, разряды кода являются степенями двойки.

На рис. 1.6 показан пример последовательного двоичного сигнала, поступающего с одного вывода какого-то цифрового устройства *DU* (digital unit). На этом рисунке, кроме сигналов по тактовой линии и самого последовательного двоичного сигнала, показано также цифровое устройство *DU* с его входами и выходом. Под каждым тактовым отрезком времени приведены их весовые коэффициенты.

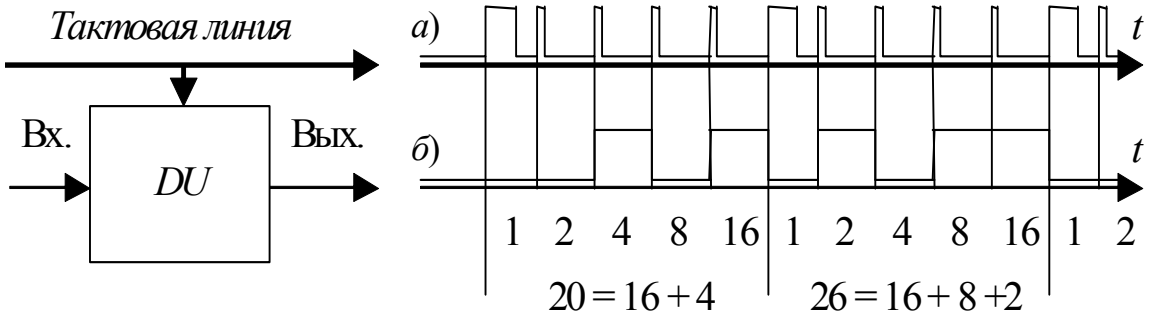


Рис. 1.6. Пример последовательного двоичного сигнала, поступающего с одного вывода цифрового устройства *DU* (digital unit):
а – сигнал, передаваемый по тактовой линии;
б – последовательность нулей и единиц, в первом интервале соответствующая числу 20, а во втором – 26

Последовательные двоичные сигналы хорошо использовать при передаче данных на большие расстояния, когда прокладывать многожильный кабель для параллельного двоичного сигнала становится достаточно дорого, ведь в отличие от последовательного параллельный сигнал передается одновременно по нескольким линиям, как это показано на рис. 1.7. В приведенном примере параллельного сигнала совокупность нулей и единиц во время первого тактового импульса соответствует числу 3, а во втором – 10.

Обычно передача цифровых данных на расстояние до нескольких метров производится с помощью параллельных сигналов, а на большие расстояния – с помощью последовательных сигналов.

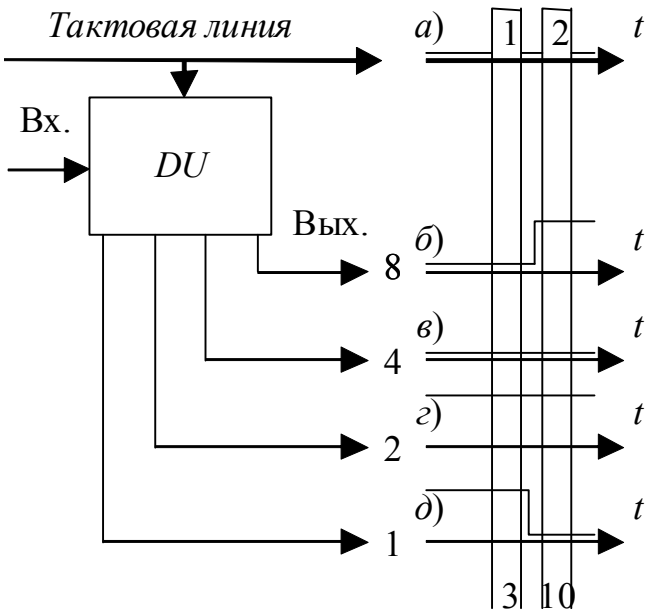


Рис. 1.7. Пример параллельного двоичного сигнала, поступающего одновременно с нескольких выводов цифрового устройства *DU*: *а* – сигнал, передаваемый по тактовой линии; *б* – сигнал, поступающий со старшего выхода цифрового устройства *DU*, которому соответствует весовой коэффициент 8; *в* – сигнал, поступающий со среднего выхода цифрового устройства *DU*, которому соответствует весовой коэффициент 4; *г* – сигнал, поступающий со среднего выхода цифрового устройства *D*, которому соответствует весовой коэффициент 2; *д* – сигнал, поступающий с младшего выхода цифрового устройства *DU*, которому соответствует весовой коэффициент 1

Как видно из трех вышеприведенных рисунков, самым быстрым способом передачи информации является использование параллельного сигнала, но при этом требуются самые большие аппаратные затраты.

Самым медленным, но зато и самым простым и требующим самые малые затраты оборудования, является применение числоимпульсного сигнала.

Последовательный позиционный двоичный сигнал является промежуточным между параллельным и числоимпульсным сигналами.

1.3. Комбинационные и последовательностные цифровые устройства

Все устройства, оперирующие двоичной (дискретной) информацией, подразделяются на два больших класса: *комбинационные схемы* (дискретные автоматы без памяти) и *последовательностные устройства* (дискретные автоматы с памятью).

Электронные цепи дискретного действия можно разделить на *цифровые* и *импульсные* устройства.

К импульсным устройствам можно отнести мультивибраторы *G* и одновибраторы (ждущие мультивибраторы) *G*1, компараторы напряжения *COMP*, триггеры Шмидта, генераторы, частота которых управляется напряжением *VCO*.

Все цифровые устройства подразделяются на *комбинационные* и *последовательностные* (с памятью).

Последовательностные устройства могут быть разделены на триггеры *T*, на счетчики *CT*, на регистры *RG* памяти и сдвига.

Комбинационные устройства в свою очередь бывают *кодирующие* и *арифметические*.

Кодирующие устройства делят на дешифраторы *DC*, мультиплексоры *MS*, шифраторы *CD* и преобразователи произвольных кодов (*PLA*, *PROM*, *X*/*Y*). Кодирующее устройство преобразует многоразрядный входной код в выходной код, построенный по иному закону. Работа кодирующего устройства задается таблицей истинности и не может быть описана достаточно простым алгоритмом (как, например, сумматор).

Арифметические устройства подразделяют на сумматоры *SM*, компараторы кодов *COMP*, схемы контроля четности и арифметико-логические устройства *ALU*.

Все *комбинационные схемы* (логические устройства) характеризуются отсутствием памяти. *Память* – свойство системы сохранять в течение требуемого времени значения сигналов, характеризующих внутреннее состояние цифрового устройства. Сигналы на выходах комбинационного устройства в любой момент времени однозначно определяются сочетанием сигналов на входах и не зависят от его предыдущих состояний. Схемным признаком таких устройств служит отсутствие цепей обратной связи, т. е. замкнутых петель для прохождения с выхода на входы. Примерами комбинационных схем могут служить логические элементы, электронные ключи, шифраторы, дешифраторы, мультиплексоры, демультиплексоры, большинство арифметических устройств.

Последовательностные устройства обладают памятью, и при смене информации на входах для предсказания сигналов на выходах необходимо знать и состояние, в котором устройство находилось до этого. Последовательностные устройства организуются из комбинационных устройств путем их надлежащего включения. Показателем принадлежности схемы к последовательностному типу служит наличие в ней обратных связей. Простейшими последовательностными устройствами являются триггеры. К этому же классу относятся регистры, счетчики, запоминающие устройства.

Работу схем, реализующих переключательные функции, принято рассматривать в безразмерном дискретном времени, для чего реальное время разбивается на интервалы, которые нумеруются, начиная с какого-то момента. Каждый такой промежуток называется тактовым интервалом или просто тактом.

Дискретное время складывается из отдельных тактов, длительность которых для характеристики работы устройства не имеет значения. Продолжительность отдельных тактов может быть различной, но на работе устройства это не сказывается.

Обновление информации на выходах происходит в момент начала нового такта. Временные задержки, обусловленные переходными процессами, обычно не учитывают, однако когда частота смены тактов велика и соизмерима с предельным быстродействием устройства, с ними приходится считаться.

В последовательностных устройствах за счет памяти функция внешних переходов определяется состоянием входов и выходов в двух соседних тактах – до и после воздействия входных сигналов. Эту пару тактов принято обозначать t^n и t^{n+1}. У сигналов, действующих в определенные такты, к наименованию вывода добавляют и номер такта. Так, запись $Q^n = 1$ означает, что на выводе Q в интервале времени $t^n \le t < t^{n+1}$ действует сигнал логической единицы.

ГЛАВА 2. МАТЕМАТИЧЕСКИЕ ОСНОВЫ ЦИФРОВОЙ ЭЛЕКТРОНИКИ

2.1. Системы счисления в цифровой электронике

Все современные системы счисления (кроме некоторых римских цифр) являются позиционными, т. е. в них одна и та же цифра в разных позициях (слева, справа) имеет разное значение. Например, в десятичном числе 55 левая цифра означает 50, а правая – только 5. В общем виде в позиционной системе счисления с основанием системы X число A можно представить в виде:

$$A = \sum_{i=0}^{n-1} a_i \cdot X^i ,$$

где n – количество разрядов числа A; a_i – коэффициенты каждого разряда, которые могут принимать значения от 0 до $X - 1$.

При необходимости основание системы счисления указывается внизу после числа в виде нижнего индекса.

2.1.1. Позиционные системы счисления, используемые в цифровых устройствах

• Четырехразрядное *десятичное* число:

$$5685_{10} = 5 \cdot 10^3 + 6 \cdot 10^2 + 8 \cdot 10^1 + 5 \cdot 10^0,$$

где $X = 10$ – основание системы счисления, $a_0 = 5$, $a_1 = 8$, $a_2 = 6$, $a_3 = 5$ – коэффициенты в каждом разряде, $n = 4$ – количество разрядов числа A.

• Трехразрядное *восьмеричное* число:

$$372_8 = 3 \cdot 8^2 + 7 \cdot 8^1 + 2 \cdot 8^0 = 250_{10},$$

где $X = 8$ (восьмеричная система счисления), коэффициенты в разрядах числа A: $a_0 = 2$, $a_1 = 7$, $a_2 = 3$; $n = 3$ – количество разрядов числа A.

• Двухразрядное *шестнадцатеричное* число:

$$4E_{16} = 4 \cdot 16^1 + 14 \cdot 16^0 = 78_{10},$$

где $X = 16$ (шестнадцатеричная система счисления), коэффициенты в разрядах числа A: $a_0 = E = 14$, $a_1 = 4$ (шестнадцатеричные цифры от 0 до 9 записываются так же, как и соответствующие десятичные цифры, а шестнадцатеричные цифры 10, 11, 12, 13, 14, 15 записываются заглавными латинскими буквами A, B, C, D, E, F, соответственно); $n = 2$ – количество разрядов числа A.

• Четырехразрядное двоичное число:

$$1101_2 = 1\cdot2^3 + 1\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 = 13_{10},$$

где $X = 2$ – двоичная система счисления, коэффициенты в разрядах числа A: $a_0 = 1$, $a_1 = 0$, $a_2 = 1$, $a_3 = 1$, $n = 4$ – количество разрядов числа A.

Двоичный разряд, который может принимать только два значения: 0 или 1, называют «бит», происходящее от сокращения BIT английских слов

BINARY DIGIT – «двоичная цифра». В английском языке слово *bit* означает также кусочек.

Десятичная система, к которой мы привыкли, основана на количестве пальцев рук и для применения в цифровой технике неудобна. В цифровой аппаратуре устройства обычно имеют два рабочих состояния и в них применяют двоичную систему счисления. Чем меньше основание системы счисления, тем менее компактный вид имеет запись какого-то числа, поэтому для сокращенной записи двоичных чисел обычно используют восьмеричную или шестнадцатеричную системы счисления. Восьмеричная и шестнадцатеричная системы счисления применяются в вычислительной технике из-за удобства представления больших двоичных чисел, поскольку их основания являются степенями двойки. Они являются как бы компромиссными между двоичной и десятичной системами счисления.

2.1.2. Преобразование чисел из одной системы счисления в другую. Кодирование сигналов

Преобразовать десятичное число в двоичное можно путем деления на 2 сначала самого числа, а затем каждого промежуточного частного. При этом каждый неделимый остаток дает очередную цифру соответствующего разряда искомого двоичного числа. Первый полученный таким образом остаток даст цифру младшего разряда, а последний – старшего разряда двоичного числа. Например, десятичное число 53_{10} преобразуем в двоичное (рис. 2.1).

Аналогично можно преобразовывать числа с другими основаниями.

На основе двоичной системы счисления в цифровой технике разработаны различные формы представления чисел, которые иногда называют кодами. Естественную запись двоичного числа называют двоичным кодом.

Так же как большие десятичные числа для удобства чтения разбивают при записи на тройки, так и большие двоичные числа обычно разбивают на четверки – тетрады. Две тетрады или восемь двоичных цифр называют байт – BYTE. Число $2^{10} = 1024$ называют в цифровой технике словом «килобит» в нарушение точного значения данного слова – 1000.

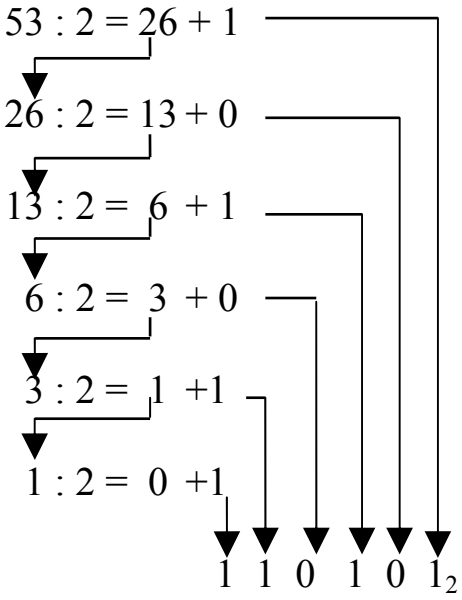


Рис. 2.1. Пример преобразования десятичное число 53_{10} в двоичное

В цифровой аппаратуре, в основном при индикации показаний десятичными цифрами или при задании параметров десятичными задатчиками, широко применяются различные двоично-десятичные коды.

Самый распространенный из них *BCD*-код (сокращенное BINARY CODED DECIMAL – двоично-кодированная десятичная цифра), который порой называют позиционным 8421-кодом, или натуральным двоично-десятичным кодом. В этом коде каждая десятичная цифра представляется своей отдельной тетрадой – четверкой двоичных цифр, например:

$$\begin{matrix} & 5 & 2 & 6 \\ 526_{10} = & 0101 & 0010 & 0110_{BCD}. \end{matrix}$$

С помощью 4-х битов можно составить 16 различных сочетаний единиц и нулей, а для десятичных цифр достаточно 10 сочетаний. 6 сочетаний избыточны, поэтому возможны различные позиционные двоично-десятичные коды. Отличаются эти коды порядком следования степеней основания в представлении десятичных цифр. Так *BCD*-код имеет следующие веса разрядов:

$$2^3, 2^2, 2^1, 2^0 = 8421.$$

Другой позиционный двоично-десятичный код – *код Эйкена* имеет следующие веса разрядов:

$$2^1, 2^2, 2^1, 2^0 = 2421.$$

Десятичное число 9_{10}, записанное в *BCD*-коде, равно 1001_{BC} , а это же число, записанное в коде Эйкена, имеет следующий вид: $1111_{\text{ЭЙКЕН}}$.

Еще один распространенный двоично-десятичный код – код с избытком 3 (EXCESS – 3 CODE). В нем каждая десятичная цифра кодируется двоичной тетрадой, в которой взвешенная сумма разрядов больше этой десятичной цифры на три. Так, десятичная цифра 9 записывается тетрадой 1100, для которой взвешенная сумма разрядов $8\cdot1 + 4\cdot1 + 2\cdot0 + 1\cdot0 = 12$, что на 3 больше 9.

Этот код называют самодополняющимся до 10, т. е. его младшие 5 цифр (от 0 до 4) являются зеркальным отражением, инверсией старших 5 цифр (от 5 до 9); например, число $0 = 0011_{EX3}$ является инверсным числу $9 = 1100_{EX3}$, число $1 = 0100_{EX3}$ является инверсным числу $8 = 1011_{EX3}$ и т. д. Достоинством кода с избытком 3 является его повышенная надежность при передаче информации в канале связи, поскольку количество единиц в его числах в среднем равно количеству нулей.

В двоично-кодированных датчиках перемещения или угла поворота часто применяется *код Грея* (GRAY CODE). В этом коде комбинации двоичных цифр, представляющие числа, соседние по величине, отличающиеся лишь в одной кодовой позиции, т. е. при последовательном переходе от одного числа к другому всегда изменяется только один из двоичных разрядов.

Число *B*, записанное в двоичном коде, можно преобразовать в число *G* в коде Грея с помощью следующего выражения:

$$G_i = B_i \oplus B_{i+1}.$$

Число *G*, записанное в коде Грея, можно преобразовать в число *B* в двоичном коде с помощью следующего выражения:

$$B_i = G_i \oplus B_{i+1}.$$

Здесь знак $\oplus$ означает сумму по модулю два, которая равна единице, если входные слагаемые разные, или – нулю, если они одинаковые, т. е.

$$0 \oplus 0 = 0$$

$$0 \oplus 1 = 1$$

$$1 \oplus 0 = 1$$

$$1 \oplus 1 = 0$$

Число в коде Грея можно также получить из двоичного кода следующим образом:

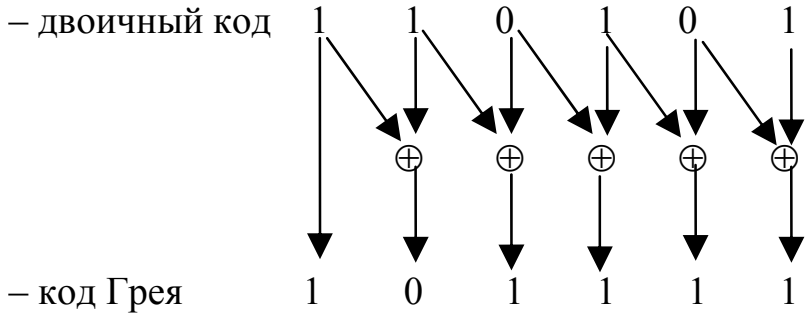


Рис. 2.2. Пример преобразования двоичного числа в код Грея

Обратное преобразование кода Грея в двоичный код производят по похожей схеме:

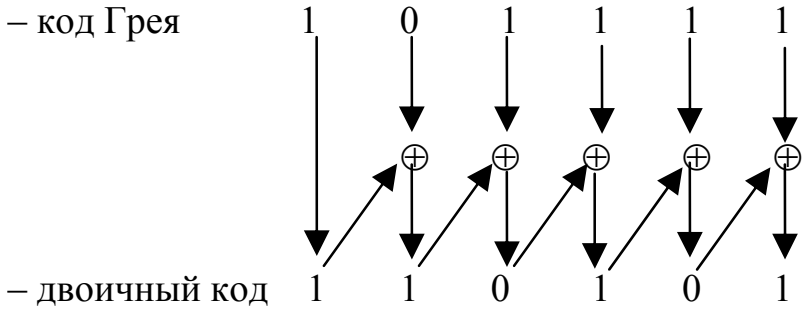


Рис. 2.3. Пример обратного преобразования числа в коде Грея в двоичное

Свойство кода Грея изменяться только в одном разряде при последовательном переходе от одного числа к другому ближнему определяет его преимущество перед другими кодами при использовании этого кода для построения кодирующих дисков и пластин. Очевидно, что такое свойство кода уменьшает число переключений считывающих устройств и снижает неоднозначность считывания кода. Код Грея нельзя отнести к позиционным кодам, поскольку в нем весовые значения единиц в различных позициях нельзя однозначно определить по формуле, приведенной в начале данного раздела.

2.2. Арифметические действия над многоразрядными двоичными числами

2.2.1. Сложение многоразрядных чисел

Сложение двух двоичных чисел осуществляется точно так же, как и в случае десятичных чисел. Более того, двоичное сложение даже проще, так как есть лишь несколько правил, которые необходимо будет запомнить. Рассмотрим десятичное сложение:

$$\begin{array}{r} 3\ 7\ 6 \\ +4\ 6\ 1 \\ \hline 8\ 3\ 7 \end{array}$$

Сначала складываются младшие значащие разряды, которые в данном примере дают в сумме 7. После этого прибавляются разряды во второй позиции справа. Их сумма равна 13, поэтому делается перенос 1 из разряда десятков в старший разряд. Складывая ее с цифрами, которые там уже имеются, получаем 8.

Аналогичные операции нужно выполнить и в случае двоичного сложения. Однако теперь дело приходится иметь только с четырьмя возможными вариантами сложения, так как в любой позиции можно складывать лишь два двоичных числа (бита):

0 + 0 = 0
1 + 0 = 1
1 + 1 = 10 = 0 + перенос 1 в следующий разряд
1 + 1 + 1 = 11 = 1 + перенос 1 в следующий разряд.

Последний случай иллюстрирует ситуацию, когда складываются единицы, стоящие в одном разряде, а к ним после переноса из младшего разряда добавляется еще одна единица. Теперь рассмотрим несколько примеров сложения пары двоичных чисел (в скобках рядом приведены их десятичные эквиваленты):

$$\begin{array}{rl} 011 & (3) \\ +\,110 & (6) \\ \hline 1001 & (9) \end{array} \qquad\qquad \begin{array}{rl} 1011 & (9) \\ +\ 1111 & (15) \\ \hline 11000 & (24) \end{array}$$

Рассматривать сложение более чем двух двоичных чисел нет никакой необходимости, так как все цифровые системы могут складывать лишь по два числа за такт. Если же надо сложить более двух чисел, то сначала складываются два первых из тех, а затем к сумме

прибавляется третье и т. д. Поскольку современные цифровые компьютеры могут осуществлять операцию сложения за время порядка наносекунды, то такой подход вполне оправдывает себя.

2.2.2. Представление чисел со знаком

В цифровых компьютерах двоичные числа выражаются набором состояний двоичных устройств хранения (триггеров). Каждое устройство представляет один бит, например, шестибитовый триггерный регистр может хранить двоичные числа от 000000 до 111111 (от 0 до 63 в десятичном представлении). Эта величина называется *модулем* числа. Большинство современных цифровых компьютеров и калькуляторов работают как с положительными числами, так и с отрицательными, поэтому требуется какой-то способ представления *знака* числа («+» или «–»). Обычно с этой целью к числу прибавляют еще один бит, который называется *знаковым битом*.

В общем случае если знаковый бит содержит 0, то число считается положительным, а если 1, то число будет отрицательным, как показано на рис. 2.4. Регистр *A* содержит биты 0110100. Крайний слева 0 (бит *A*6) – это знаковый бит, который представляет собой «+». Шесть остальных битов представляют модуль двоичного числа 1101002, что эквивалентно 5210. Таким образом, число, хранящееся в регистре *A*, будет равно +52. Аналогично число в регистре *B* будет равно –52, так как 1 в знаковом бите представляет знак «–».

Знаковый бит используется, чтобы показать принадлежность данного хранящегося двоичного числа к положительным или отрицательным числам. Числа на рис. 2.4 состоят из знакового бита и шести бит, представляющих модуль. Биты модуля – действительные двоичные эквиваленты десятичных значений. Такая система называется системой типа знак–модуль и служит для представления двоичных чисел со знаком.

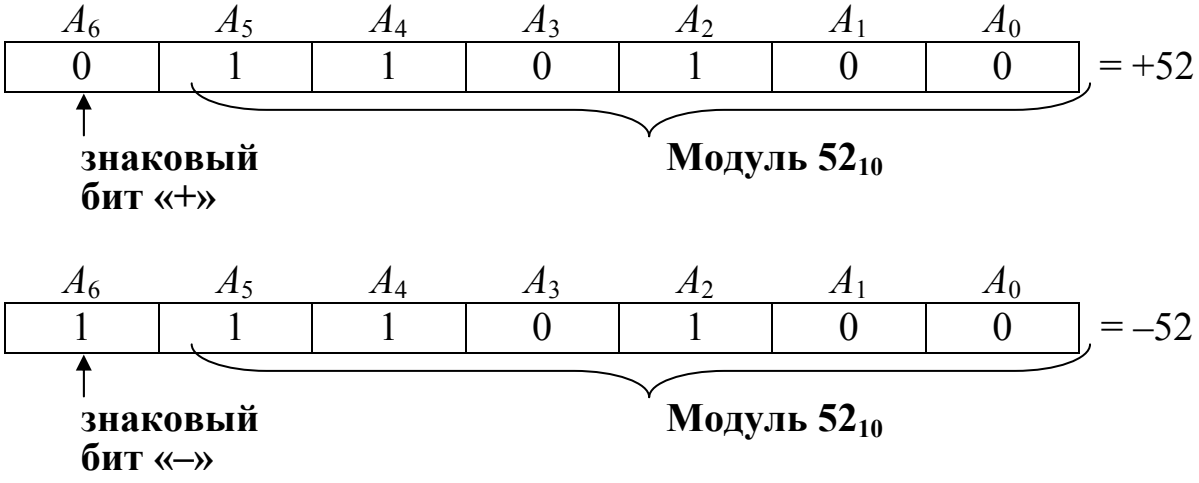


Рис. 2.4. Представление чисел со знаком в форме знак–модуль

Хотя эта система и относится к прямым методам представления чисел, калькуляторы и компьютеры обычно ее не используют, так как в этом случае реализация схем, как правило, более сложна, чем для других систем. Наиболее распространенная система представления чисел со знаком – дополнительный код, однако сначала необходимо научиться получать обратный и дополнительный коды двоичного числа.

2.2.3. Обратный код

Обратный код двоичного числа (l's complement) получают заменой каждого 0 на 1 и каждой 1 на 0. Иначе говоря, каждый бит числа изменяется на противоположный (обратный). Этот процесс показан далее:

1 0 1 1 0 1 двоичное число

↓ ↓ ↓ ↓ ↓ ↓

0 1 0 0 1 0 обратный код двоичного числа

Таким образом, обратный код числа 101101 будет равен 010010.

2.2.4. Дополнительный код

Дополнительный код двоичного числа (2's complement) получают из обратного кода путем добавления 1 к его младшему значащему биту. Процесс преобразования для числа $1011012 = 4510$ показан ниже:

	1	0	1	1	0	1	двоичный эквивалент 45
	0	1	0	0	1	0	каждый бит преобразуется в обратный код
+						1	прибавляется 1
	0	1	0	0	1	1	дополнительный код двоичного числа

Таким образом, число 010011 будет дополнительным кодом числа 101101.

А вот еще один пример преобразования двоичного числа в дополнительный код:

	1	0	1	1	0	1	двоичное число
	0	1	0	0	1	1	каждый бит преобразуется в обратный код
+						1	прибавляется 1
	0	1	0	1	0	0	дополнительный код двоичного числа

2.2.5. Представление чисел со знаком в системе дополнительных кодов

Дополнительный код применяется для представления чисел со знаком следующим образом.

Если число положительное, то его модуль представляется в естественной форме двоичного числа, а знаковый бит содержит 0, который ставится перед старшим знаковым битом.

Такой случай показан на рис. 2.5 для числа 45_{10}.

Если число отрицательное, то его модуль представляется при помощи дополнительного кода, а знаковый бит, содержащий 1, ставится перед старшим знаковым битом. Эта ситуация отражена на рис. 2.5 для числа -45_{10}.

Дополнительный код используется для представления чисел со знаками, потому что этот код (о чем будет сказано далее) позволяет осуществлять операцию вычитания через сложение. Это очень удобно, поскольку компьютер может использовать одну и ту же схемотехнику как для сложения, так и для вычитания, тем самым экономя аппаратную часть.

Рис. 2.5. Представление чисел со знаком с помощью дополнительного кода

2.2.6. Отрицание

Отрицанием называется логическая операция преобразования положительного числа в отрицательное или отрицательного числа в положительное. Если двоичные числа со знаком представляются в виде дополнительного кода, отрицание эквивалентно простому преобразованию в дополнительный код. Чтобы продемонстрировать это, преобразуем число +9. В двоичной форме вместе с битом знака это число будет выглядеть как 01001.

Если преобразовать его в дополнительный код, получим 10111. Ясно, что теперь это уже отрицательное число, так как знаковый бит, стоящий в первом разряде, равен 1. Действительно, число 10111 представляет собой –9, которое является отрицательным. Точно так же можно начинать и с числа –9, которое в двоичной форме будет выглядеть как 10111. Если преобразовать это число в дополнительный код, получим 01001, т. е. число +9. Порядок преобразований показан ниже:

двоичный код числа	→	0	1	0	0	1	= +9
преобразование в дополнительный код	→	1	0	1	1	1	= –9
снова осуществляем операцию отрицания	→	0	1	0	0	1	= +9

Таким образом, операция отрицания двоичного числа со знаком заключается в получении его дополнительного кода. Операция отрицания изменяет число на такое же число, но с противоположным знаком.

2.2.7. Сложение в системе дополнительных кодов

Теперь проанализируем, как осуществляются операции сложения и вычитания в цифровых машинах, которые для представления отрицательных чисел используют дополнительный код. Важно помнить, что в большинстве случаев со знаковым битом обращаются так же, как и с битами модуля.

Вариант I. *Два положительных числа*. Сложение двух положительных чисел выполняется непосредственно. Рассмотрим сложение чисел +9 и +4:

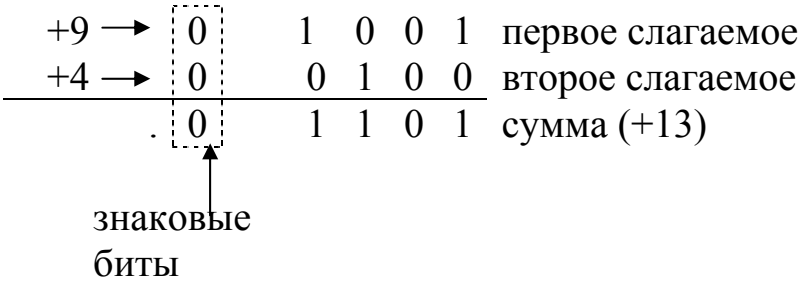


Рис. 2.6. Пример сложения двух положительных чисел

Обратите внимание, что оба знаковых бита как первого, так и второго слагаемого равны 0, поэтому и знаковый бит суммы равен 0, что говорит о ее принадлежности к положительным числам. Также следует заметить, что оба слагаемые должны иметь одинаковое количество бит, так как это необходимо для использования дополнительного кода. Поэтому всегда нужно добавлять нули слева от значащей части числа, чтобы уравнять количество битов обоих чисел.

Вариант II. *Положительное число и отрицательное число, меньшее по модулю*. Рассмотрим сложение двух чисел: +9 и –4. Число –4 выражается в виде дополнительного кода. Таким образом, сначала надо перевести +4(00100) в –4(11100).

В этом случае знаковый бит второго слагаемого содержит 1. Обратите внимание, что теперь оба знаковых бита участвуют в процессе сложения. В последнем разряде модуля возникает перенос, который, однако, не учитывается.

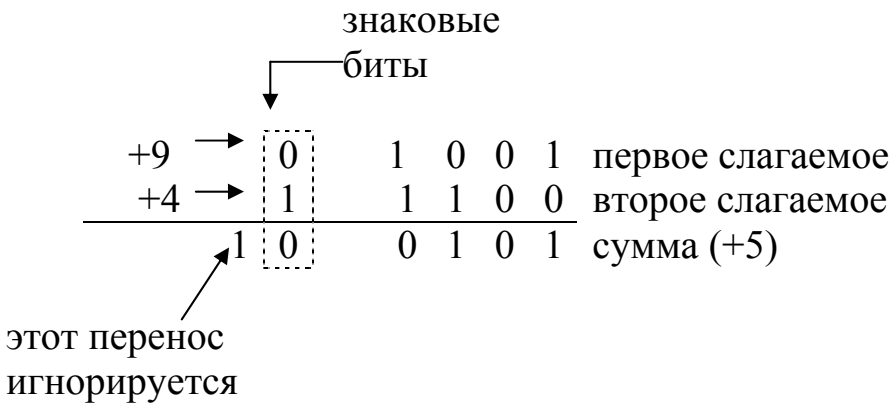


Рис. 2.7. Пример сложения положительного числа и отрицательного, меньшего по модулю

Вообще, при использовании дополнительного кода перенос из старшего разряда не учитывается, поэтому конечный результат равен 00101, что эквивалентно +5.

Вариант III. *Положительное число и отрицательное число, большее по модулю*. Рассмотрим сложение чисел –9 и +4.

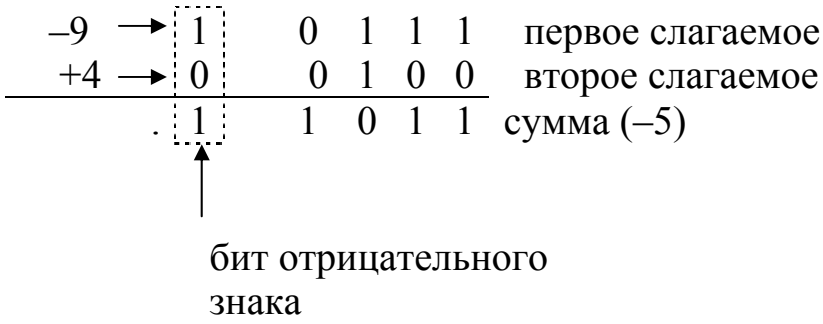


Рис. 2.8. Пример сложения положительного числа и отрицательного, большего по модулю

Полученная сумма имеет знаковый бит, содержащий 1, это говорит о том, что она отрицательна. Так как сумма меньше нуля, то она имеет вид дополнительного кода, т. е. только четыре младших бита (1011) представляют собой реальную величину числа (модуль). Чтобы получить абсолютное значение числа, необходимо применить операцию отрицания (получить дополнительный код числа) к величине 11011; результат будет равен 00101 = +5. Таким образом, число 11011 с учетом знакового бита представляет собой число –5.

Вариант IV. *Два отрицательных числа*.

Конечный результат будет отрицательным и представляется в виде дополнительного кода, т. е. его знаковый бит равен 1, отрицание этого числа (получение его дополнительного кода) даст 01101 = +13.

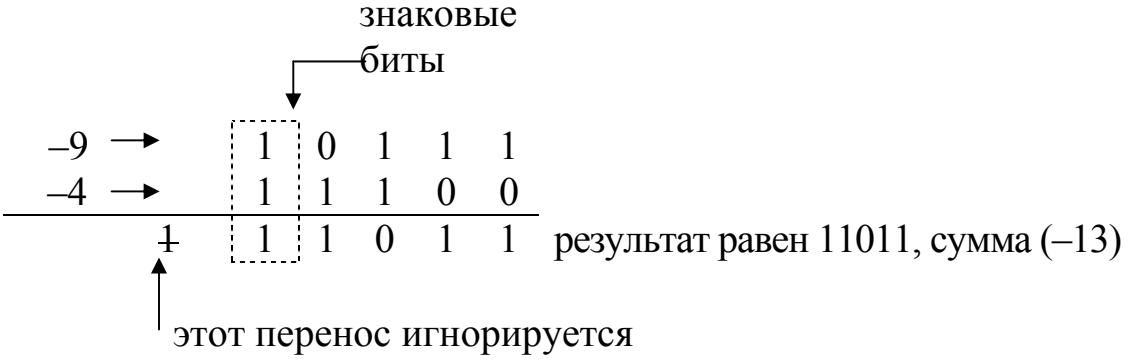


Рис. 2.9. Пример сложения двух отрицательных чисел

2.2.8. Вычитание в системе дополнительных кодов

Операция *вычитания* в системе дополнительных кодов по сути состоит из операции сложения и ничем не отличается от примеров, показанных выше. При вычитании одного двоичного числа (*вычитаемого*) из другого двоичного числа (*уменьшаемого*) следуют такой процедуре:

1. *Отрицание вычитаемого*. Эта операция заменяет вычитаемое эквивалентной по модулю величиной, но с противоположным знаком.

2. *Сложение полученной величины с уменьшаемым*. Сумма такого сложения и будет представлять собой *разность* между уменьшаемым и вычитаемым.

Так же как и во всех математических операциях с применением дополнительного кода, необходимо, чтобы оба числа содержали одинаковое количество бит. Рассмотрим ситуацию, когда от +9 отнимается +4.

2.2.9. Переполнение

В каждом из примеров сложения, приведенных выше, слагаемые числа состояли из знакового бита и четырех битов модуля. Результаты также состояли из знакового бита и тех же четырех битов модуля. Любой перенос в шестой разряд не учитывался. Во всех рассмотренных случаях модуль конечного результата можно было уместить в четыре бита. Теперь рассмотрим сложение чисел +9 и +8.

Ответ содержит знаковый бит, соответствующий отрицательному числу, который, как совершенно очевидно, неправильный, поскольку суммировались два положительных числа. В результате должно быть +17, но для записи такого числа потребуются больше четырех битов. Таким образом, возникает переполнение, которое переносит единицу в знаковый бит.

Условие переполнения может реализоваться только в случае сложения двух положительных или двух отрицательных чисел, в результате чего может получиться неправильный ответ. Переполнение легко обнаружить при проверке знакового бита – он должен совпадать со знаковыми битами обоих слагаемых.

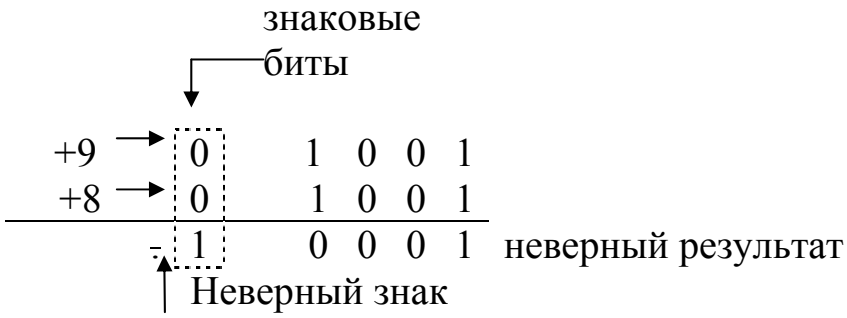


Рис. 2.10. Пример переполнения и неверного результата

Поскольку операция вычитания происходит в системе дополнительных кодов и осуществляется с помощью отрицания вычитаемого и последующего сложения полученной величины с уменьшаемым, переполнение может возникнуть только в том случае, если уменьшаемое и вычитаемое имеют разные знаки.

Например, если вычитается –8 из +9, то к числу –8 нужно применить операцию отрицания, в результате получим +8, которое затем сложится с +9, как было показано ранее. В этом случае возникнет переполнение разряда, которое приведет к неверному результату, так как модуль суммы окажется слишком большим.

2.3. Основы алгебры логики

2.3.1. Логические переменные и константы

Называемая в честь английского математика Дж. Буля *булевой алгеброй*, алгебра логики составляет теоретическую основу логики, теории алгоритмов и логического проектирования цифровых схем. Отличие *булевой алгебры* от традиционной главным образом состоит

в том, что первая оперирует с константами и переменными, принимающими только два возможных значения – 0 или 1.

Булева переменная – это параметр, который в различных случаях принимает значение 0 или 1. Эти переменные часто используют для представления уровня напряжения в проводнике или на контакте ввода-вывода схемы. Например, булево значение 0 может быть назначено для представления любого напряжения в диапазоне от 0 до 0,8 В, в то время как значение 1 характерно для представления любого напряжения в диапазоне от 2 до 5 В.

Таким образом, булевы 0 и 1 не являются реальными величинами, а представляют состояния переменной напряжения, или так называемые логические уровни. Напряжение в цифровой схеме представляет логический уровень 0 или логический уровень 1 – в зависимости от присущих ему действительных численных значений.

Два возможных значения логических переменных называют ИСТИНА (TRUE) и ЛОЖЬ (FALSE), иногда их называют ДА и НЕТ, но чаще всего их обозначают, соответственно, как 1 и 0. При этом следует помнить, что эти логические 0 и 1 не надо трактовать как числа, над ними нельзя производить арифметические действия.

В цифровой логике одновременно с 0 и 1 используются и другие термины. Некоторые из наиболее часто употребляемых обозначений показаны в табл. 2.1.

Таблица 2.1

Примеры наиболее часто употребляемых обозначений лог. 0 и лог. 1

Логический 0	Логическая 1
Ложь	Истина
Выключен	Включен
Низкий	Высокий
Нет	Да
Ключ открыт	Ключ закрыт

2.3.2. Основные логические операции

Булева алгебра оперирует двоичными переменными, которые условно обозначаются как 0 и 1 и подчиняются условию: $x = 1$, если $x \neq 0$, и $x = 0$, если $x \neq 1$. В ее основе лежит понятие переключательной, или булевой, функции вида $f(x_1, x_2, \ldots x_n)$ относительно аргументов x_1, x_2, $\ldots$ x_n, которая, как и ее аргументы, может принимать только два значения – 0 и 1. Как частный случай двоичные переменные могут постоянно сохранять одно из значений – 0 либо 1.

Логическая функция может быть задана словесно, алгебраическим выражением и таблицей, которая называется таблицей истинности.

Действия над двоичными переменными производятся по правилам логических операций. Между обычной, привычной нам, алгеброй и алгеброй логики имеются существенные различия в отношении количества и характера операций, а также законов, которым они подчиняются.

Простейших логических операций три: *отрицание* (инверсия, операция НЕ), *логическое умножение* (конъюнкция, операция И) и *логическое сложение* (дизъюнкция, операция ИЛИ). Более сложные логические преобразования можно свести к указанным операциям.

Операция отрицания выполняется над одной переменной и характеризуется следующими свойствами: функция $y = 1$ при аргументе $x = 0$ и $y = 0$, если $x = 1$. Обозначается отрицание чертой над переменной, с которой производится операция: $y = \bar{x}$ (игрек равен не икс). Соответственно, $\bar{y} = x$.

Операция логического умножения (конъюнкция) для двух переменных характеризуется табл. 2.2 и обозначается следующим образом: $0 \cdot 0 = 0$; $0 \cdot 1 = 0$; $1 \cdot 0 = 0$; $1 \cdot 1 = 1$, т. е. нулевое значение хотя бы одного из аргументов обеспечивает нулевой результат операции. Операция может быть распространена на большее число переменных.

Таблица 2.2

Таблица истинности логического умножения

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

Таблица 2.3

Таблица истинности логического сложения

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

Операцию *логического сложения* (дизъюнкции) определяет табл. 2.3. Обозначают ее таким способом: $y = x_1 \vee x_2$ либо $y = x_1 + x_2$. Первый способ предпочтителен, так как позволяет отличать логическое сложение от арифметического. Для двух переменных $0 \vee 0 = 0$;

$0 \vee 1 = 1$; $1 \vee 0 = 1$; $1 \vee 1 = 1$, т. е. равенство хотя бы одного аргумента логической единице определяет единичное значение всей функции.

Дизъюнкция, как и конъюнкция, может осуществляться с многими переменными.

Совокупность различных значений переменных называют набором. Булева функция n аргументов может иметь до $N = 2^n$ наборов. Поскольку функция принимает только два значения, общее число булевых функций n аргументов равно $2^N = 2^{2n}$. Таким образом, функция одного аргумента может иметь четыре значения: $y = x$, $y = \bar{x}$, $y = 1$ (константа 1); $y = 0$ (константа 0).

2.3.3. Законы, аксиомы и правила алгебры логики

Булева алгебра базируется на нескольких аксиомах, из которых выводят основные законы для преобразований с двоичными переменными. Обоснованность выбора этих аксиом подтверждается таблицами истинности для рассмотренных операций. Каждая аксиома представлена в двух видах, что вытекает из принципа дуальности (двойственности) логических операций, согласно которому операции конъюнкции и дизъюнкции допускают взаимную замену, если одновременно поменять логическую 1 на 0, 0 на 1, знак $\vee$ на •, а • на $\vee$.

Аксиомы операции отрицания: $\bar{0} = 1$; $\bar{1} = 0$.

Аксиомы операций конъюнкции и дизъюнкции:

$0 \cdot 0 = 0$; $\quad 1 \vee 1 = 1$;

$0 \cdot 1 = 1 \cdot 0 = 0$; $\quad 0 \vee 1 = 1 \vee 0 = 1$;

$1 \cdot 1 = 1$; $\quad 0 \vee 0 = 0$.

Законы булевой алгебры вытекают из аксиом и также имеют две формы выражения: для конъюнкции и дизъюнкции. Здесь они приводятся без доказательств. Их правильность легко проверить по таблицам истинности либо путем подстановки 0 и 1 вместо соответствующих значений переменных.

- Переместительный закон:

$$x_1 x_2 = x_2 x_1, \qquad x_1 \vee x_2 = x_2 \vee x_1.$$

- Сочетательный закон:

$$x_1(x_2x_3) = (x_1x_2)x_3 = x_1x_2x_3,$$
$$x_1(x_2 \vee x_3) = (x_1 \vee x_2)x_3 = x_1 \vee x_2 \vee x_3.$$

- Закон повторения (тавтологии):

$$xx = x, \qquad x \vee x = x.$$

- Закон обращения: если $x_1 = x_2$, то $\bar{x}_1 = \bar{x}_2$.
- Закон двойной инверсии: $\bar{\bar{x}} = x$.
- Закон нулевого множества:

$$x \cdot 0 = 0, \qquad x \vee 0 = 0.$$

- Закон универсального множества:

$$x \cdot 1 = 1, \qquad x \vee 1 = 1.$$

- Закон дополнительности:

$$x\bar{x} = 0, \qquad x \vee \bar{x} = 1.$$

- Распределительный закон:

$$x_1(x_2 \vee x_3) = x_1x_2 \vee x_1x_3,$$
$$x_1 \vee (x_2x_3) = (x_1 \vee x_2)(x_1 \vee x_3).$$

- Закон поглощения:

$$x_1(x_2 \vee x_3) = x_1x_2 \vee x_1x_3, \qquad x_1(x_1 \vee x_2) = x_1.$$

- Закон склеивания:

$$(x_1 \vee x_2)(x_1 \vee \bar{x}_2) = x_1, \; x_1x_2 \vee x_1\bar{x}_2 = x_1.$$

- Закон инверсии (закон де Моргана):

$$\overline{x_1x_2} = \bar{x}_1 \vee \bar{x}_2, \qquad \overline{x_1 \vee x_2} = \bar{x}_1\bar{x}_2.$$

2.3.4. Способы задания логических функций

Основных функций в булевой алгебре всего три: логическое умножение И, логическое сложение ИЛИ и отрицание НЕ.

Логическая функция может быть задана четырьмя способами:

1) словесно (описанием ситуации);

2) алгебраическим выражением;

3) таблицей истинности;

4) электрической схемой, состоящей из контактов переключателей.

Например:

1. Лифт можно вызвать, если закрыты двери лифта на первом этаже, и на втором этаже, и на третьем этаже.

2. Если закрытые двери на первом этаже обозначить как $A = 1$, на втором как $B = 1$, на третьем как $C = 1$, возможность вызвать лифт обозначить как $F = 1$, а логическую функцию И обозначить знаком умножения «·», то алгебраическое выражение будет иметь вид:

$$F = ABC.$$

3. В таблицу истинности в левой колонке заносятся все возможные комбинации входных аргументов, а в правой колонке записывают соответствующие этим комбинациям значения выходной функции. Входные комбинации записываются в порядке возрастания их значений от всех нулей до всех единиц сверху вниз. Таблица истинности, соответствующая данному примеру, будет иметь следующий вид (табл. 2.4):

Таблица 2.4

Задание функции *F = ABC* при помощи таблицы истинности

A	***B***	***C***	***F***
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

4. Электрическая контактная схема обладает хорошей наглядностью, но может быть легко построена лишь для самых простых логических функций. Для нашего примера эта схема может иметь следующий вид:

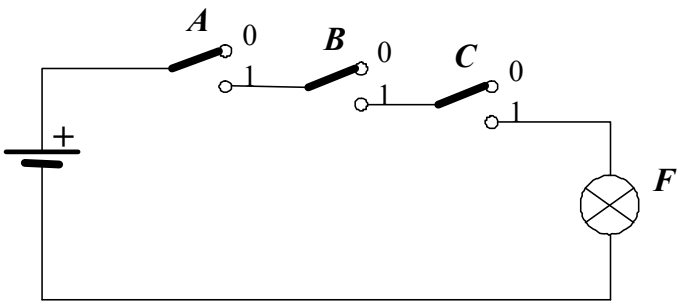


Рис. 2.11. Задание функции *F = ABC*
при помощи электрической контактной схемы

2.4. Элементарные функции алгебры логики одной и двух переменных

2.4.1. Логические элементы, реализующие элементарные функции алгебры логики

Наиболее часто встречаются следующие названия и буквенные обозначения функции И: *логическое умножение*, *конъюнкция*, *совпадение*, AND, И.

Возможные виды алгебраической записи функции И:

$$F = A \,\&\, B; \quad F = A \wedge B; \quad F = A \cdot B; \quad F = AB; \quad F = AB.$$

Контактная схема для функции И для двух переменных A и B:

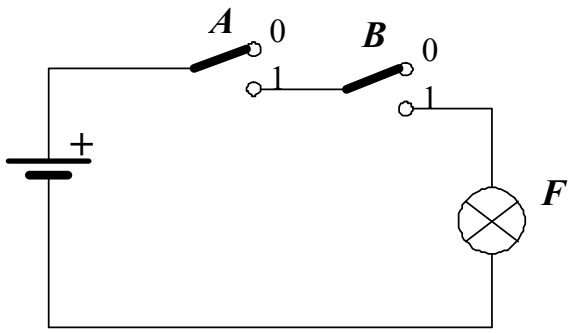


Рис. 2.12. Задание функции 2И при помощи электрической контактной схемы

Таблица 2.5

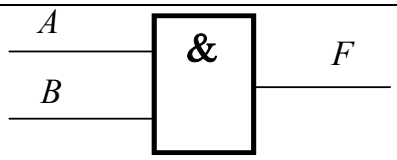
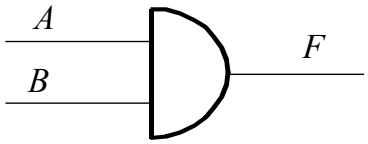
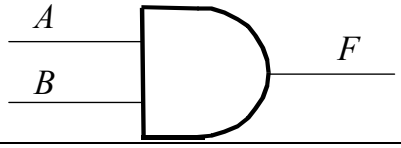
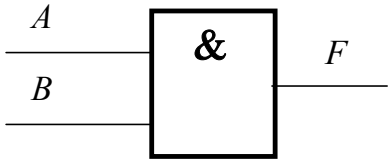
Таблица истинности функции 2И

A	***B***	***F***
0	0	0
0	1	0
1	0	0
1	1	1

В схемах логический элемент И (AND – gate) обозначают следующим образом (табл. 2.6):

Таблица 2.6

Условное графическое обозначение (УГО) логического элемента, реализующего функцию И

– в отечественных схемах согласно ГОСТ 2.743–91	
– старое зарубежное обозначение, которое до настоящего времени встречается в иностранной технической литературе согласно DIN 40700	
– обозначение в соответствии с американской системой MILSPEC 806B	
– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией – МЭК 117–15	

Наиболее часто встречаются следующие названия и буквенные обозначения функции ИЛИ: *логическое сложение*, *дизъюнкция*, OR, ИЛИ.

Алгебраическая запись функции ИЛИ: $F = A \vee B$; $F = A+B$.

Контактная схема для функции 2ИЛИ представлена на рис. 2.13.

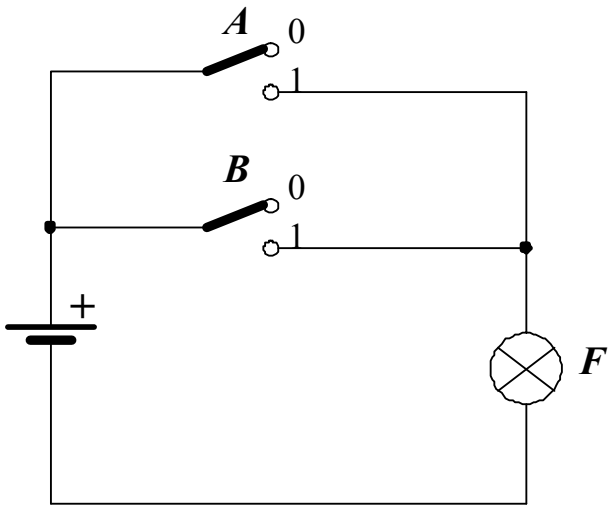


Рис. 2.13. Задание функции 2ИЛИ при помощи электрической контактной схемы

Таблица 2.7

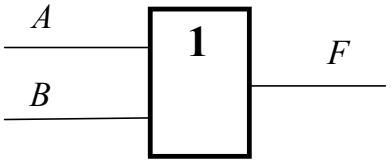
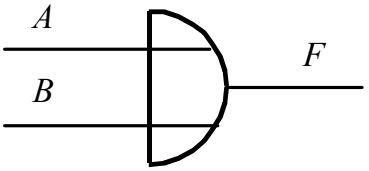
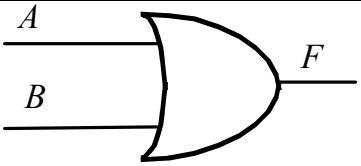
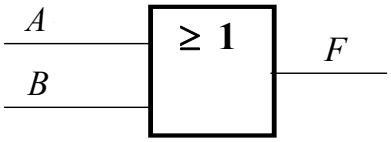
Таблица истинности функции 2ИЛИ

A	***B***	***F***
0	0	0
0	1	1
1	0	1
1	1	1

В схемах логический элемент ИЛИ (OR – gate) обозначают следующим образом (табл. 2.8):

Таблица 2.8

Условное графическое обозначение (УГО) логического элемента, реализующего функцию ИЛИ

– в отечественных схемах согласно ГОСТ 2.743–91	
– старое зарубежное обозначение, которое до настоящего времени встречается в иностранной технической литературе согласно DIN 40700	
– обозначение в соответствии с американской системой MILSPEC 806B	
– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией – МЭК 117–15	

Наиболее часто встречаются следующие названия и буквенные обозначения функции НЕ: *логическое отрицание*, *инверсия*, *дополнение*, NOT, НЕ.

Возможные виды алгебраической записи функции НЕ:

$$F = \neg A;\ F = A',\ F = \overline{A}.$$

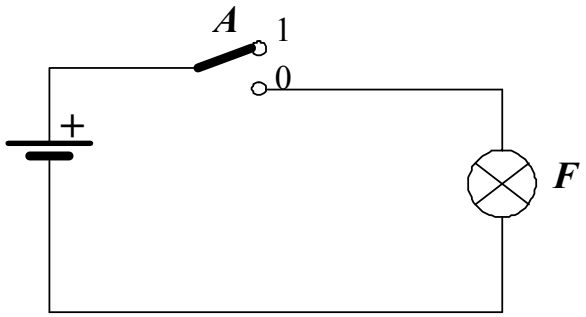


Рис. 2.14. Задание функции НЕ при помощи электрической контактной схемы

Таблица 2.9

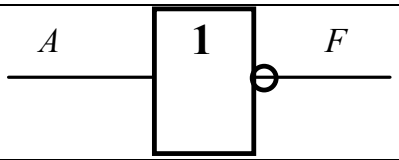
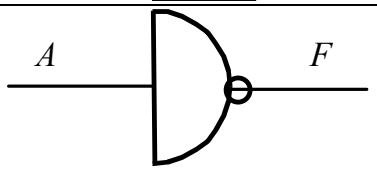
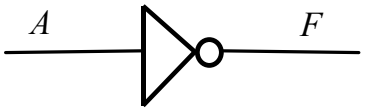
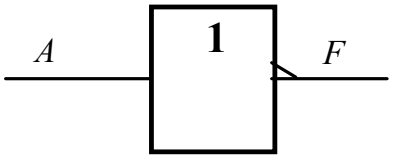
Таблица истинности функции НЕ

A	***F***
0	1
1	0

В схемах логический элемент НЕ (NOT) обозначают следующим образом (табл. 2.10):

Таблица 2.10

Условное графическое обозначение (УГО) логического элемента, реализующего функцию НЕ

– в отечественных схемах согласно ГОСТ 2.743–91	
– старое зарубежное обозначение, которое до настоящего времени встречается в иностранной технической литературе согласно DIN 40700	
– обозначение в соответствии с американской системой MILSPEC 806B	
– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией – МЭК 117–15	

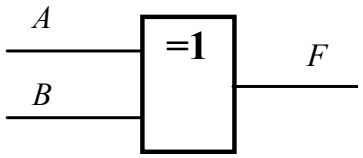
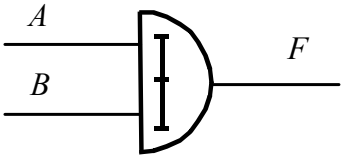
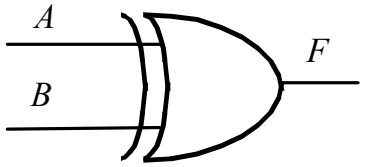
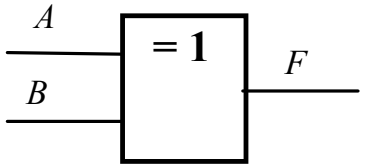
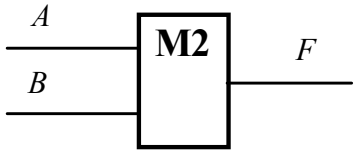
Хочется отметить еще один логический элемент для реализации двухвходовой функции «исключающее ИЛИ», булево выражение которого в двухвходовом случае совпадает с булевым выражением для сумматора по модулю два и имеет следующий вид:

$$F = \overline{A}B + A\overline{B}.$$

В схемах логический элемент «исключающее ИЛИ» обозначают следующим образом (табл. 2.11):

Таблица 2.11

Условное графическое обозначение логического элемента, реализующего функцию «исключающее ИЛИ»

– в отечественных схемах согласно ГОСТ 2.743–91	
– старое зарубежное обозначение, которое до настоящего времени встречается в иностранной технической литературе согласно DIN 40700	
– обозначение в соответствии с американской системой MILSPEC 806B	
– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией – МЭК 117–15	
Условное графическое обозначение логического элемента, реализующего двухвходовый сумматор по модулю два, который в логических схемах выполняет ту же функцию, что и «исключающее ИЛИ», в отечественных схемах имеет вид:	

Следует заметить, что данные элементы очень широко применяются в цифровой схемотехнике.

Таблица 2.12

Таблица истинности функции «исключающее ИЛИ»

A	***B***	***F***
0	0	0
0	1	1
1	0	1
1	1	0

2.4.2. Инвертирующие базисы, отрицательная логика

Три вышеописанные логические функции И, ИЛИ, НЕ, с помощью которых можно получить все остальные логические функции, называют *булевым базисом*.

Иногда объединяют две булевы функции (при этом одной из них является НЕ) и получившийся логический элемент считают базовым для получения всех остальных логических функций.

Элемент И–НЕ называют также: штрих Шеффера (Sheffer stroke), NAND (сокращение от NOT AND).

Алгебраическая запись функции И–НЕ: $F = A/B$; $F = \overline{AB}$.

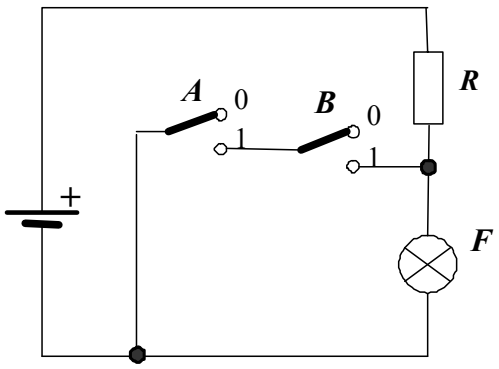


Рис. 2.15. Контактная схема для функции И–НЕ для двух переменных

Таблица 2.13

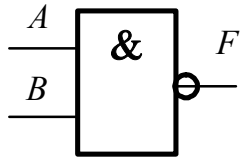
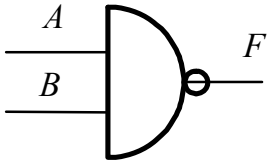
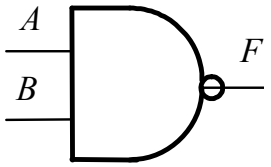
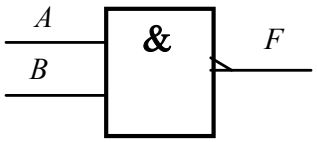
Таблица истинности функции 2И–НЕ

A	***B***	***F***
0	0	1
0	1	1
1	0	1
1	1	0

В схемах логический элемент И–НЕ (NOT AND) обозначают следующим образом (табл. 2.14):

Таблица 2.14

Условное графическое обозначение (УГО) логического элемента, реализующего функцию И–НЕ

– в отечественных схемах согласно ГОСТ 2.743–91	
– старое зарубежное обозначение, которое до настоящего времени встречается в иностранной технической литературе согласно DIN 40700	
– обозначение в соответствии с американской системой MILSPEC 806B	
– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией – МЭК 117–15	

Элемент ИЛИ–НЕ называют также: стрелка Пирса (Pierce arrow), NOR (сокращение от NOT OR).

Алгебраическая запись функции ИЛИ–НЕ: $F = A \downarrow B$; $F = \overline{A + B}$.

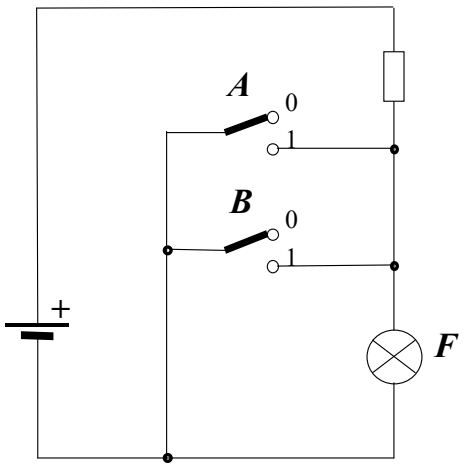


Рис. 2.16. Контактная схема для функции ИЛИ–НЕ для двух переменных

Таблица 2.15

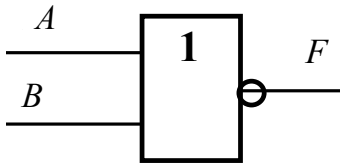
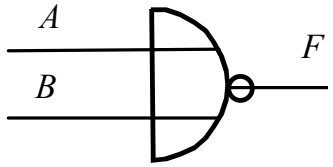
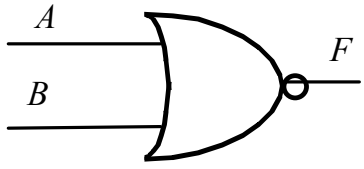
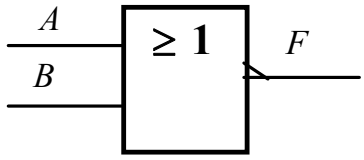
Таблица истинности функции 2ИЛИ–НЕ

A	***B***	***F***
0	0	1
0	1	0
1	0	0
1	1	0

В схемах логический элемент ИЛИ–НЕ (NOT OR) обозначают следующим образом (табл. 2.16):

Таблица 2.16

Условное графическое обозначение (УГО) логического элемента, реализующего функцию ИЛИ–НЕ

– в отечественных схемах согласно ГОСТ 2.743–91	
– старое зарубежное обозначение, которое до настоящего времени встречается в иностранной технической литературе согласно DIN 40700	
– обозначение в соответствии с американской системой MILSPEC 806B	
– обозначения в соответствии с системой, рекомендованной Международной Электротехнической Комиссией – МЭК 117–15	

Логика называется *положительной*, если высокий потенциал отображает единицу, а низкий – ноль. Если наоборот, высокий потенциал отображает ноль, а низкий – единицу, то логика называется *отрицательной*. Данное правило называют логическим соглашением.

Самым важным следствием применения отрицательной логики является то, что при переходе от положительной логики к отрицательной функция И превращается в ИЛИ и наоборот.

Это можно проиллюстрировать следующим образом:

– в положительной логике: в комнате зимой **Т**епло, если батареи отопления **В**ключены И окна **З**акрыты (**Т = ВЗ**);

– в отрицательной логике: в комнате зимой **НЕ Т**епло, если батареи отопления **НЕ В**ключены ИЛИ окна **НЕ З**акрыты (**Т** = $\overline{\mathbf{В}}$ + $\overline{\mathbf{З}}$).

Здесь И переходит в ИЛИ, когда входные аргументы и вывод отрицаются, при этом смысл выражения практически не меняется.

Благодаря этому переходу от И к ИЛИ и удается с помощью однотипных элементов инвертирующего базиса получать все остальные логические функции. Об этом говорят два постулата де 'Моргана:

$$\overline{AB} = \overline{A} + \overline{B}\ ;\ \overline{A + B} = \overline{A}\,\overline{B}.$$

Если логический элемент в положительной логике реализует функцию И, то в отрицательной логике этот же элемент реализует функцию ИЛИ, и наоборот, логический элемент ИЛИ положительной логики реализует функцию И в отрицательной логике.

Применение наряду с положительной логикой и отрицательной логики позволяет любое сложное логическое преобразование выполнить с применением только логических элементов И–НЕ или только ИЛИ–НЕ.

Покажем это хотя бы для функций булева базиса.

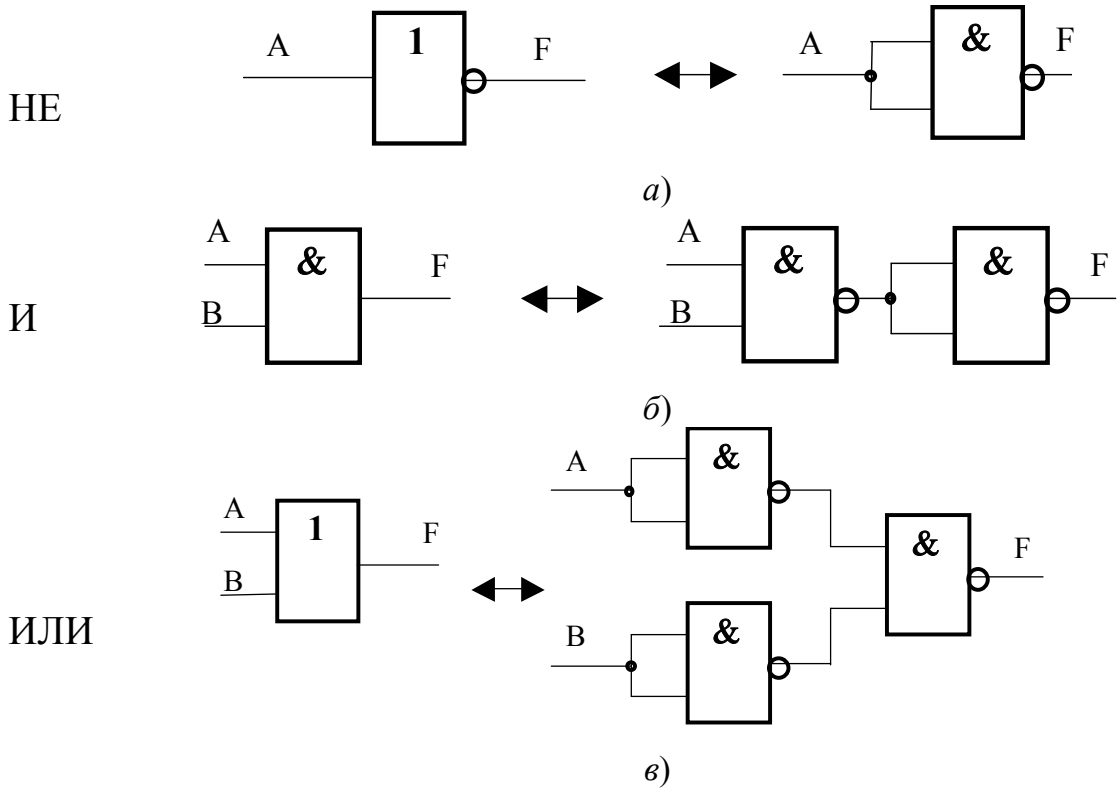


Рис. 2.17. Схемы, реализующие функции: *а* – НЕ; *б* – И; *в* – ИЛИ с помощью только логических элементов 2И–НЕ

2.4.3. Дизъюнктивные и конъюнктивные формы записи функций алгебры логики

Совершенной дизъюнктивной нормальной формой (СДНФ) называют наиболее полную форму записи логического выражения. Эта форма записи представляет собой сумму, каждое слагаемое которой является произведением всех входных аргументов или их инверсий, например:

$$F = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + AB\overline{C} + ABC.$$

СДНФ является избыточной, но логические функции, записанные в СДНФ, легко сравнивать между собой, их удобно преобразовывать в таблицы истинности и составлять по ним карты Карно. Булево выражение, полученное из таблицы истинности логической функции, имеет совершенную дизъюнктивную нормальную форму.

В некоторых случаях более удобной формой записи логического выражения является *совершенная конъюнктивная нормальная форма* (СКНФ). Это произведение сомножителей, каждый из которых является суммой всех входных аргументов или их инверсий, например:

$$F = (\overline{A} + B + \overline{C})(\overline{A} + B + C)(\overline{A} + \overline{B} + C)(A + B + C).$$

Так же как и СДНФ, СКНФ является избыточной.

2.4.4. Преобразование таблицы истинности в булево выражение

Допустим, имеется логическая функция F для трех переменных A, B и C, заданная в виде следующей таблицы истинности:

Таблица 2.17

Пример логической функции *F* для трех переменных *A*, *B* и *C*, заданной в виде таблицы истинности

Номер	*A*	*B*	*C*	*F*	Примечания
0	0	0	0	1	$P_0 = \overline{A}\,\overline{B}\,\overline{C}$
1	0	0	1	0	–
2	0	1	0	1	$P_2 = \overline{A}B\overline{C}$
3	0	1	1	1	$P_3 = \overline{A}BC$
4	1	0	0	0	–
5	1	0	1	0	–
6	1	1	0	0	–
7	1	1	1	1	$P_7 = ABC$

Из всех возможных восьми комбинаций входных переменных A, B и C данная функция F равна единице только для тех четырех комбинаций, которые записаны в виде логических произведений P_0, P_2, P_3 и P_7 в правой части таблицы, в разделе примечания. При остальных наборах входных переменных функция F равна нулю. Смысл каждого булева выражения в том, чтобы показать, при каких сочетаниях входных переменных или их инверсий заданная функция F равна единице. Поскольку функция будет иметь такое значение при любом из наборов P_0, P_2, P_3, P_7 независимо друг от друга, то их можно соединить между собой знаком ИЛИ логическим сложением:

$$F = P_0 + P_2 + P_3 + P_7.$$

Каждый из наборов P_0, P_2, P_3, P_7 является таким сочетанием входных переменных или их инверсий, которые только при совместном их воздействии обеспечивают единичное состояние выходной функции.

Следовательно, каждый такой набор состоит из всех входных переменных или их инверсий, связанных между собой функцией И логическим умножением:

$$P_0 = \overline{A}\,\overline{B}\,\overline{C}\,;\;\; P_2 = \overline{A}B\overline{C}\,;\;\; P_3 = \overline{A}BC\,;\;\; P_7 = ABC.$$

Исходя из этого, получаем результирующее выражение:

$$F = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + AB\overline{C} + ABC.$$

Как можно заметить, это выражение записано в СДНФ.

2.5. Минимизация логических функций

2.5.1. Минимизации алгебраическим способом

Основой минимизации алгебраическим способом является последовательное использование законов булевой алгебры и правил преобразований. Кроме законов, известных нам из обычной алгебры, в булевой алгебре типовыми приемами можно считать следующие:

1. *Многократное прибавление или умножение* какого-либо переменного или нескольких переменных, что не изменяет функцию, поскольку:

$$A + A + \ldots + A = A\,;\quad ABC + ABC + \ldots = ABC\,;$$

$$AAA \cdot \ldots = A\,;\quad ABC \cdot ABC \cdot \ldots = ABC\,.$$

2. *Умножение членов уравнения на сумму* $A + \overline{A} = 1$ или сложение их с $A \cdot \overline{A} = 0$.

3. Использование выражений и законов булевой алгебры:

$$\overline{\overline{A}} = A; \qquad A + \overline{A}B = A + B;$$

$$A + AB = A; \qquad A \cdot (A + B) = A;$$

$$A + (BC) = (A + B)(A + C); \qquad A + 1 = 1.$$

Пример минимизации логической функции алгебраическим способом:

$$F = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC = \overline{A}BC + A\overline{B}C + AB\overline{C} +$$
$$+ ABC + \underline{(ABC + ABC)}.$$

В форму записи функции добавили (*ABC + ABC*)*;* после этого перегруппируем слагаемые таким образом, чтобы вынести за скобки общий множитель:

$$F = \overline{A}BC = ABC + A\overline{B}C + ABC + AB\overline{C} + ABC =$$
$$= (A + \overline{A})BC + (B + \overline{B})AC + (C + \overline{C})AB =$$
$$= AB + DC + AC.$$

2.5.2. Минимизация логических функций по картам Карно

Для минимизации логических функций очень удобно пользоваться *картами Карно* или очень схожими с ними диаграммами Вейча.

Карта Карно изображает в виде графических квадратов (клеток) все возможные комбинации переменных, причем переменные, определяющие координаты клеток карты, размещают так, чтобы при переходе из одной клетки в соседнюю как по горизонтали, так и по вертикали изменялась только одна переменная.

Таблица истинности для четырех переменных включает 16 строк, следовательно, карта Карно должна состоять из 16 клеток, как показано на рис. 2.18.

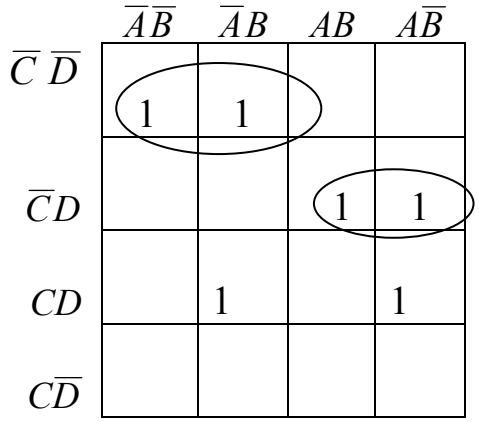


Рис. 2.18. Пример карты Карно для 4-х переменных

Если требуется получить карту Карно для какой-либо функции, сначала надо записать эту функцию в СДНФ, в совершенной дизъюнктивно нормальной форме или в виде таблицы истинности. Каждое слагаемое булева выражения в СДНФ, или каждая единица в столбце функции таблицы истинности, задается на карте Карно единицей в соответствующей клетке. Координаты этой клетки содержат те же входные переменные и их инверсии, что и данное слагаемое СДНФ булева выражения (или данная строка таблицы истинности).

У карты Карно для четырех переменных клетки крайнего левого столбца должны рассматриваться как соседние для клеток крайнего правого столбца, а клетки верхней строки – как соседние для клеток нижней строки. Другими словами, можно сказать, что эта карта расположена на поверхности цилиндра (склеили правый край карты с левым), изогнутого и растянутого так, что его верхний срез соединяется с нижним срезом; при этом цилиндр превращается в тор.

Правила упрощения заполненной карты Карно заключаются в следующем:

– соседние две, четыре, восемь или другое число единиц, равное степени двойки, обводят общим контуром;

– контур должен быть прямоугольным без изгибов или наклонов;

– каждый контур превращает все входящие в него единицы в одну, т. е. объединенные таким образом слагаемые СДНФ булева выражения дают одно слагаемое в упрощенном выражении;

– те входные переменные, которые входят в координаты данного контура совместно со своими инверсиями, исключаются из слагаемого, которое дает этот контур в упрощенном выражении.

Примеры упрощения булевых выражений с помощью карты Карно для 4-х переменных:

1. $F1 = \overline{A}BC\overline{D}_1 + AB\overline{C}\overline{D}_2 + \overline{A}B\overline{C}D_3 AB\overline{C}D_4 + \overline{A}\overline{B}C\overline{D}_5 + \overline{A}BC\overline{D}_6$.

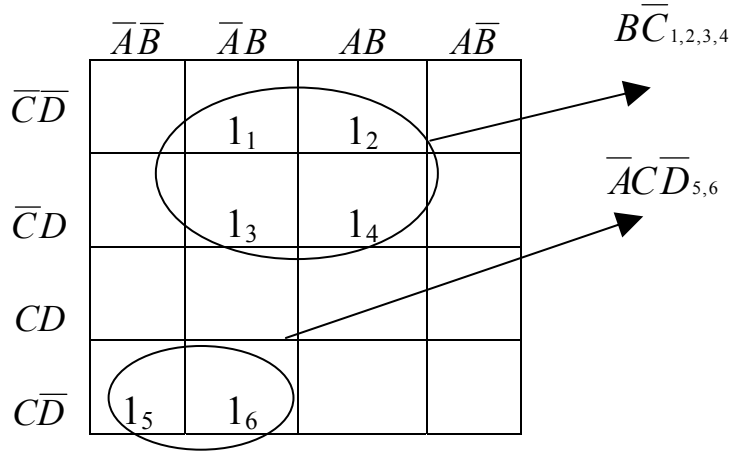


Рис. 2.19. Пример минимизации булевой функции $F1$ с помощью карты Карно для 4-х переменных

В первом примере минимизации булевой функции $F1$ (рис. 2.19) нижний контур из двух единиц 1_5 и 1_6, соответствующие пятому и шестому слагаемым в исходном булевом выражении, дает возможность опустить B и $\overline{B}$. После этого в нем остается произведение $\overline{A}CD$. В верхнем контуре из четырех единиц 1_1, 1_2, 1_3 и 1_4, соответствующие первым четырем слагаемым в исходном булевом выражении, попарно опускаются A и $\overline{A}$, D и $\overline{D}$, так что в результате этого верхний контур дает произведение $B\overline{C}$.

Полученное упрощенное выражение: $F1 = BC + \overline{A}C\overline{D}$,

$F2 = \overline{A}\overline{B}\overline{C}\overline{D}_1 + \overline{A}B\overline{C}\overline{D}_2 + AB\overline{C}\overline{D}_3 + \overline{A}\overline{B}C\overline{D}_4 + \overline{A}BC\overline{D}_5$.

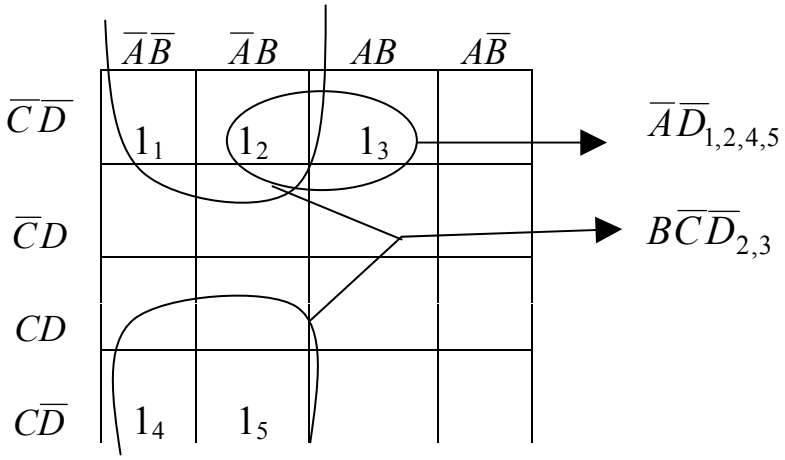


Рис. 2.20. Пример минимизации булевой функции *F*2
с помощью карты Карно для 4-х переменных

Полученное упрощенное выражение: $F2 = \overline{A}\,\overline{D} + B\overline{C}\,\overline{D}$.

Во втором примере минимизации булевой функции *F*2 (рис. 2.20) контур из двух единиц 1_2 и 1_3, соответствующих второму и третьему слагаемым в исходном булевом выражении, дает возможность опустить A и $\overline{A}$. После этого в нем остается произведение $B\overline{C}\,\overline{D}$. В контуре из четырех единиц 1_1, 1_2, 1_4 и 1_5, соответствующих другим четырем слагаемым из исходного булева выражения, попарно опускаются B и $\overline{B}$, C и $\overline{C}$, так что в результате этого верхний контур дает произведение $\overline{A}\,\overline{D}$. Карта Карно представляется в данном случае свернутой в цилиндр, в котором верхний край совмещается с нижним.

Этот пример показывает также, что контуры могут накладываться друг на друга (сколько угодно раз).

Таблица истинности для пяти переменных включает 32 строки, следовательно, карта Карно должна состоять из 32 клеток, как показано на рис. 2.21. Для шести переменных карта Карно содержит 64 клетки, как показано на рис. 2.22.

Пример упрощения булевых выражений с помощью карты Карно для 5-ти переменных:

Пусть задана функция

$$F3 = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}E_1 + \overline{A}\,\overline{B}\,\overline{C}DE_2 + A\overline{B}C\overline{D}\,\overline{E}_3 + \overline{A}\,\overline{B}C\overline{D}\,\overline{E}_4 + \overline{A}BC\overline{D}\,\overline{E}_5 + \overline{A}BCDE_6 +$$
$$+ \overline{A}BCD\overline{E}_7 + \overline{A}B\overline{C}DE_8 + \overline{A}B\overline{C}D\overline{E}_9 + AB\overline{C}DE_{10} + AB\overline{C}DE_{11} + ABCDE_{12} +$$
$$+ ABCD\overline{E}_{13} + A\overline{B}\,\overline{C}\,\overline{D}E_{14} + A\overline{B}\,\overline{C}DE_{15}.$$

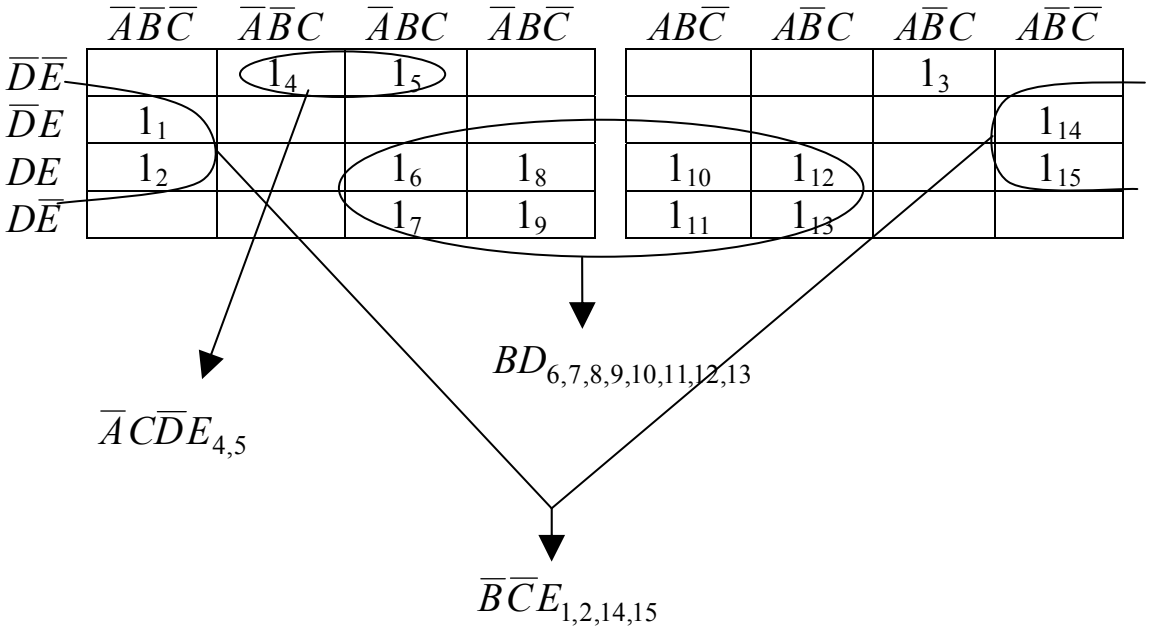


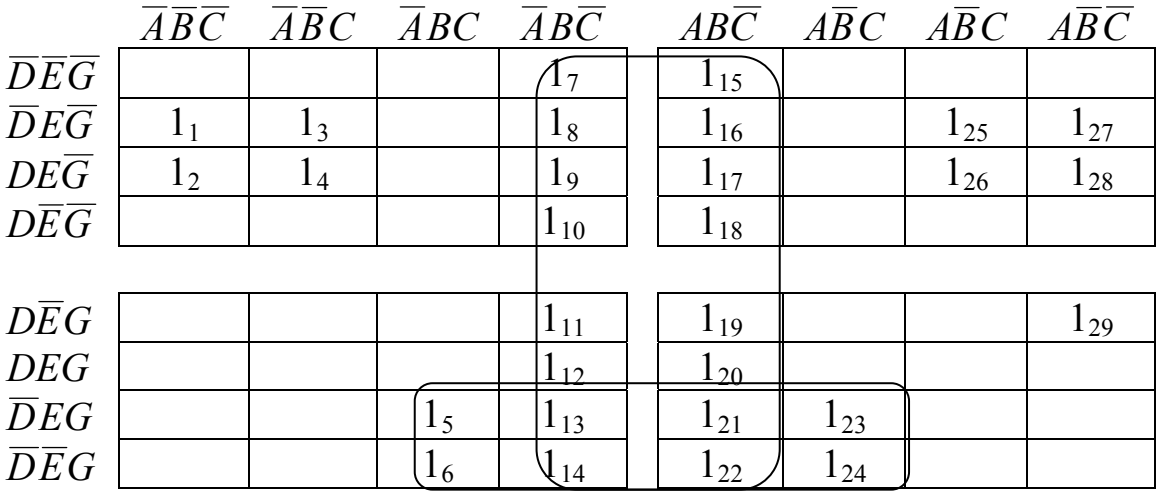
Рис. 2.21. Пример минимизации булевой функции *F*3
с помощью карты Карно для 5-ти переменных

Полученное упрощенное выражение:

$$F3 = A\overline{B}C\overline{D}\overline{E}_3 + \overline{B}\overline{C}E_{1,2,14,15} + \overline{A}CD\overline{E}_{4,5} + BD_{6,7,8,9,10,11,12,13}.$$

Пример упрощения булевых выражений с помощью карты Карно для 6-ти переменных:

$$F4 = \overline{A}\overline{B}\overline{C}\overline{D}E\overline{G}_1 + \overline{A}\overline{B}\overline{C}DE\overline{G}_2 + \overline{A}\overline{B}C\overline{D}E\overline{G}_3 + \overline{A}\overline{B}CDE\overline{G}_4 +$$
$$+ AB\overline{C}\overline{D}EG_5 + \overline{A}BC\overline{D}\overline{E}G_6 + \overline{A}B\overline{C}\overline{D}\overline{E}\overline{G}_7 + \overline{A}B\overline{C}\overline{D}E\overline{G}_8 +$$
$$+ \overline{A}B\overline{C}DE\overline{G}_9 + \overline{A}B\overline{C}D\overline{E}\overline{G}_{10} + \overline{A}B\overline{C}D\overline{E}G_{11} + \overline{A}B\overline{C}DEG_{12} +$$
$$+ \overline{A}B\overline{C}\overline{D}EG_{13} + \overline{A}B\overline{C}\overline{D}\overline{E}G_{14} + AB\overline{C}\overline{D}\overline{E}\overline{G}_{15} + AB\overline{C}\overline{D}E\overline{G}_{16} +$$
$$+ AB\overline{C}DE\overline{G}_{17} + AB\overline{C}D\overline{E}\overline{G}_{18} + AB\overline{C}D\overline{E}G_{19} + AB\overline{C}DEG_{20} +$$
$$+ AB\overline{C}\overline{D}E\overline{G}_{21} + AB\overline{C}\overline{D}\overline{E}G_{22} + A\overline{B}CDE\overline{G}_{23} + ABC\overline{D}EG_{24} +$$
$$+ A\overline{B}C\overline{D}E\overline{G}_{25} + + A\overline{B}CDE\overline{G}_{26} + A\overline{B}C\overline{D}E\overline{G}_{27} + A\overline{B}\overline{C}DE\overline{G}_{28} +$$
$$+ A\overline{B}\overline{C}D\overline{E}G_{29}.$$



$$F4 = \overline{B}eE\overline{G}_{1,2,3,4,25,26,27,28} + B\overline{D}G_{5,6,13,14,21,22,23,24} +$$

$$+ B\overline{C}_{7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22} + A\overline{C}DE\overline{G}_{19,29}$$

Рис. 2.22. Пример минимизации булевой функции *F*4
с помощью карты Карно для 6-ти переменных

2.6. Синтез комбинационных логических схем

2.6.1. Построение комбинационных логических схем по заданным булевым выражениям

Как правило, построение и расчет любой схемы осуществляется, начиная с ее выхода.

Допустим, задано булево выражение

$$F = \overline{B}A + B\overline{A} + C\overline{B}\,.$$

1. *Первый этап*: выполняется логическое сложение, логическую операция ИЛИ, считая входными переменными функции $\overline{B}A$, $B\overline{A}$ и $C\overline{B}$ (рис. 2.23).

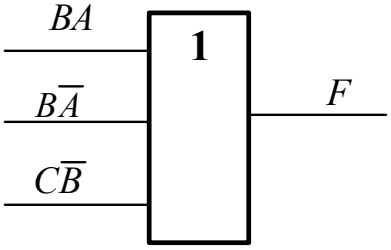


Рис. 2.23. Первый этап построения комбинационных логических схем
по заданному булеву выражению

2. *Второй этап*: к входам элемента ИЛИ подключаются логические элементы И, входными переменными которых являются уже A, B, C и их инверсии (рис. 2.24).

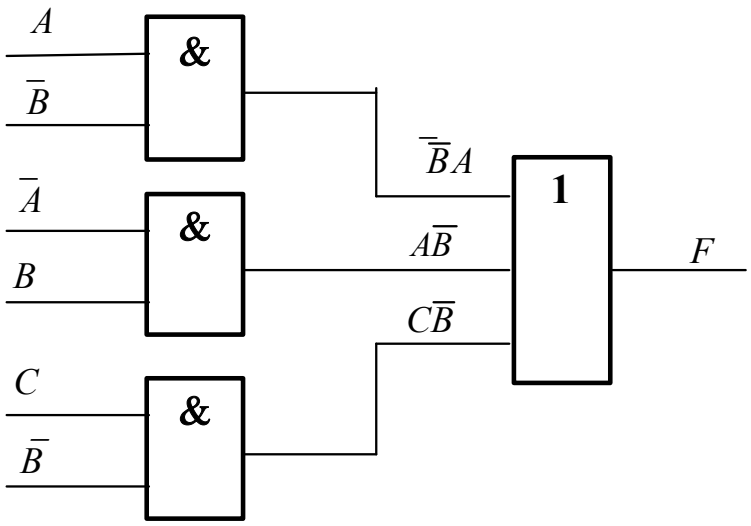


Рис. 2.24. Второй этап построения комбинационных логических схем по заданному булеву выражению

3. *Третий этап*: для получения инверсий $\overline{A}$ и $\overline{B}$ на соответствующих входах ставят инверторы (рис. 2.25).

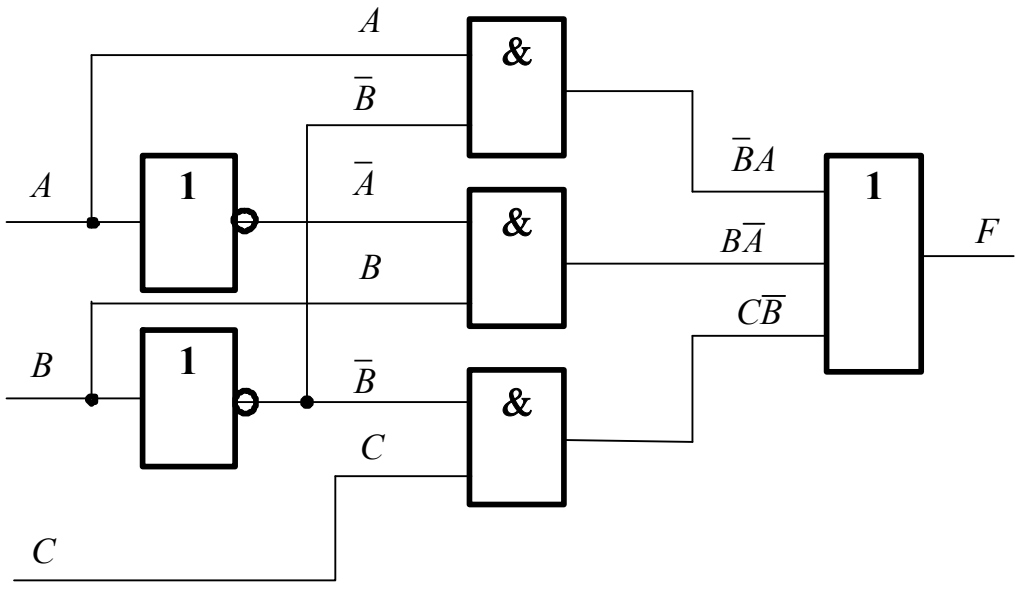


Рис. 2.25. Второй этап построения комбинационных логических схем по заданному булеву выражению

Данное построение основано на следующей особенности – поскольку значениями логических функций могут быть только нули и

единицы, то любые логические функции могут быть представлены как аргументы других более сложных функций.

Таким образом, построение комбинационной логической схемы осуществляется с выхода ко входу.

2.6.2. Особенности построения логических схем в инвертирующих базисах

Первой особенностью построения логических схем в инвертирующих базисах считается непрямая зависимость между простотой булева выражения и минимальностью соответствующей ему логической схемы. Другими словами, самое минимизированное булево выражение не всегда дает схему, минимальную по количеству инвертирующих логических элементов.

Для доказательства этого построим в инвертирующем базисе И–НЕ схему для реализации двухвходовой функции «исключающее ИЛИ», булево выражение которого в двухвходовом случае совпадает с булевым выражением для сумматора по модулю два и имеет следующий вид:

$$F = \overline{A}B + A\overline{B}.$$

Следует заметить, что данные элементы очень широко применяются в цифровой схемотехнике и вопрос минимизации их построения довольно актуален.

Поскольку исходное булево выражение для двухвходовой функции «исключающее ИЛИ», кроме функций И и НЕ, содержит и функцию ИЛИ, то, чтобы исключить ИЛИ, преобразуем его следующим образом:

$$F = \overline{A}B + A\overline{B} = \overline{\overline{\overline{A}B + A\overline{B}}} = \overline{\overline{\overline{A}B} + \overline{A\overline{B}}}.$$

Полученное выражение не является минимальным, а чтобы получить действительно минимальное выражение, произведем над исходным выражением следующий ряд преобразований:

$$F = \overline{A}B + A\overline{B} = \overline{A}B + A\overline{B} + A\overline{A} + B\overline{B} = \overline{A}(B + A) + \overline{B}(A + B) =$$

$$= (\overline{A} + \overline{B})(A + B) = \overline{AB} \cdot \overline{A\overline{B}}.$$

В соответствии с последним минимальным выражением построим в инвертирующем базисе схему из логических элементов И–НЕ (рис. 2.26):

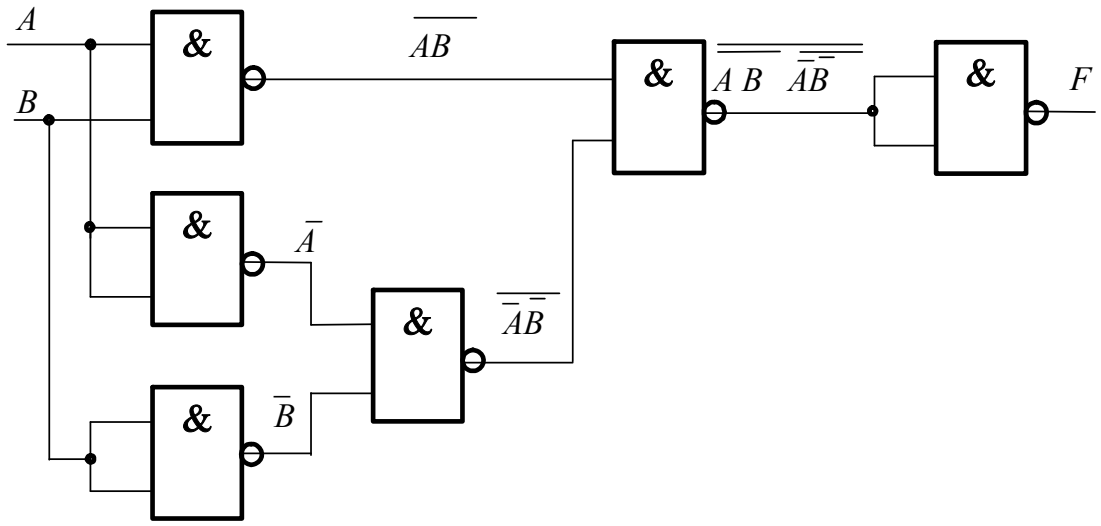


Рис. 2.26. Построение комбинационной логической схемы в инвертирующем базисе по преобразованному булеву выражению

Теперь попробуем получить такое булево выражение, которое могло бы привести к более простой логической схеме. Для этого над минимальным выражением произведем следующий ряд преобразований:

$$F = AB + \overline{A}\,\overline{B} = \overline{AB + \overline{A}\,\overline{B}} = \overline{AB + \overline{A}\,\overline{B} + A\overline{B}B + A\overline{A}B} =$$

$$= \overline{AB \cdot AB + AB \cdot \overline{B} + \overline{A}\,\overline{B} + \overline{A} \cdot AB} = \overline{AB(AB + A\overline{B}) + \overline{A}(AB + \overline{B})} =$$

$$= (AB + \overline{B})(AB + \overline{A}) = \overline{\overline{ABB}} \cdot \overline{\overline{ABA}}.$$

В соответствии с последним булевым выражением построим в инвертирующем базисе схему из логических элементов И–НЕ:

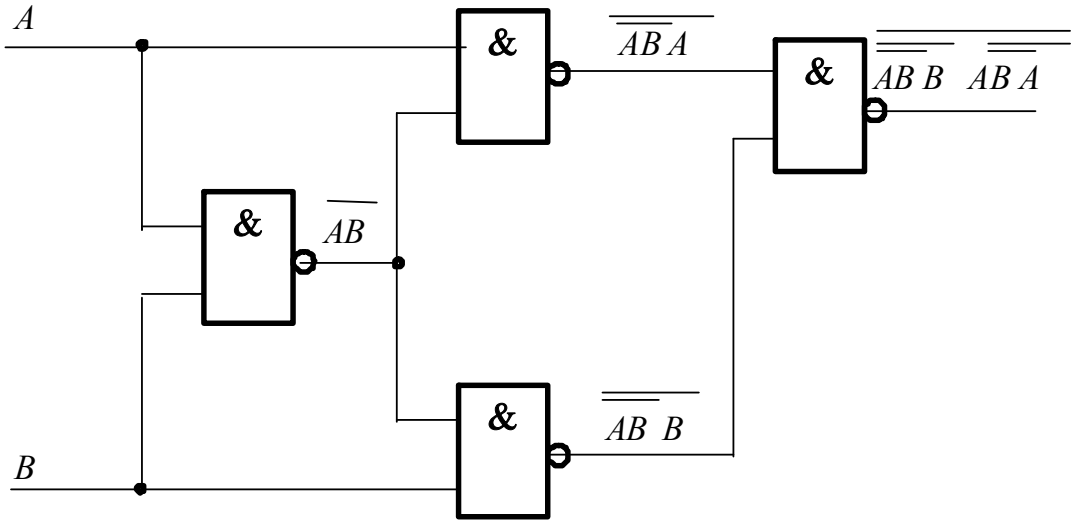


Рис. 2.27. Построение комбинационной логической схемы в инвертирующем базисе по преобразованному булеву выражению

Как видно, данная схема оказывается в полтора раза проще, чем предыдущая, несмотря на то, что булево выражение, в соответствии с которым она построена, явно сложнее, чем полученное ранее минимальное выражение. Таким образом, можно считать доказанным утверждение о том, что минимизированное булево выражение не всегда дает минимальную по количеству инвертирующих логических элементов схему.

Вторая особенность построения логических схем в инвертирующих базисах приводится без доказательства:

Если в произвольной цифровой схеме (комбинационной):

– проинвертировать все входные и выходные сигналы;

– все элементы И заменить на ИЛИ, а ИЛИ – на И, то реализуемая схемой функция не изменится.

ГЛАВА 3. ЭЛЕМЕНТНАЯ БАЗА ЦИФРОВОЙ ЭЛЕКТРОНИКИ

3.1. Классификация и обозначения цифровых микросхем

3.1.1. Основные определения

Интегральная микросхема – это микроэлектронное изделие, состоящее из активных (транзисторов) и пассивных (диодов, резисторов, конденсаторов) элементов, а также из соединяющих их проводников, которое изготавливается в едином технологическом процессе в объеме полупроводника или на поверхности диэлектрического основания, заключено в корпус и представляет собой неразделимое целое. Иногда ее называют интегральной схемой, иногда микросхемой, соответственно, возможны сокращенные обозначения ИМС, ИС, МС.

По технологии изготовления микросхемы делятся на три разновидности: полупроводниковые (самые распространенные), пленочные (почти не выпускаются) и гибридные (выпускают немного и выпуск сокращают).

В полупроводниковых микросхемах все элементы и их соединения изготавливаются в объеме (внутри) и частично на поверхности полупроводника. Иногда полупроводниковую микросхему называют твердотельной схемой, что является переводом с английского языка (solid state).

В пленочной микросхеме все элементы и их соединения выполнены в виде пленок из проводящих и диэлектрических материалов на диэлектрическом основании. В этих микросхемах нет транзисторов и диодов.

В гибридных микросхемах пассивные элементы и соединительные проводники изготавливают по пленочной технологии, а бескорпусные транзисторы и диоды, изготовленные отдельно по полупроводниковой технологии, соединяют тонкими проводами диаметром 0,04 мм с контактными площадками.

По функциональному назначению микросхемы делятся на две категории:

– аналоговые, обрабатывающие сигналы, изменяющиеся по закону непрерывной функции;

– цифровые, обрабатывающие цифровые сигналы.

Транзисторы, применяющиеся в цифровых микросхемах, бывают двух типов:

– обычные (*n–p–n* или *p–n–p*) биполярные транзисторы;

– полевые (униполярные) транзисторы.

В *цифровых* микросхемах применяются полевые транзисторы только с изолированным затвором, имеющие структуру: металл (затвор), диэлектрик (изоляция затвора), полупроводник (канал, сток–исток), сокращенно МДП, а так как в качестве диэлектрика используется окись кремния, то обычно эти транзисторы, а также микросхемы на них сокращенно называют МОП. Чаще всего в цифровых микросхемах используют пары МОП транзисторов, дополняющие друг друга по проводимости канала, такие микросхемы называют КМОП от слова комплементарный, что означает дополняющий.

В зависимости от элементов, на которых собраны входные и выходные каскады микросхем, от схемных особенностей этих каскадов цифровые микросхемы делятся на несколько групп или так называемых «*логик*» (здесь под словом «*логика*» подразумевается логический элемент или электронный ключ):

1. РТЛ – *резисторно-транзисторная логика*, в которой на входах стоит резистивный сумматор токов, реализующий для положительной логики функцию ИЛИ; выходной каскад собран на транзисторном инверторе.

2. ДТЛ – *диодно-транзисторная логика*, в которой на входах стоит несколько диодов, реализующих функцию И или ИЛИ; выходной каскад на транзисторах.

3. ТТЛ – *транзисторно-транзисторная логика*, в логических элементах которой ко входам подключены эмиттеры многоэмиттерного транзистора; с помощью этого многоэмиттерного транзистора реализуется функция И; выходной каскад собран на транзисторах.

4. ЭСЛ – *эмиттерно-связанная логика*, в которой на входах стоят транзисторы, эмиттеры которых связаны друг с другом.

5. *n*-МОП, *p*-МОП – МОП *логика*, все элементы которой выполнены на МОП (металл-окисел-полупроводник) транзисторах с проводимостью канала *n*-типа (*n*-МОП) или *p*-типа (*p*-МОП).

6. КМОП – логика, все элементы которой выполнены на двух типах МОП транзисторов *n*-МОП и *p*-МОП, дополняющих друг друга, *комплементарных*.

7. $И^2Л$ – интегральная инжекционная логика, в которой отсутствуют резисторы; инжекция носителей в область базы транзистора осуществляется с помощью активных генераторов тока, выполненных на *p–n–p* транзисторах, тогда как сам базовый инвертор – на *n–p–n* транзисторах.

По принятой у нас системе *обозначение микросхемы* должно состоять из четырех основных элементов:

1) цифра, соответствующая конструктивно-технологической группе (1, 5, 6, 7 – полупроводниковые микросхемы, из них 7 – бескорпусные; 2, 4, 8 – гибридные микросхемы; 3 – прочие, в том числе пленочные, вакуумные, керамические и т. д.);

2) две, а в последнее время три цифры, обозначающие порядковый номер разработки серии микросхем;

3) две буквы, обозначающие функциональное назначение микросхемы: первая буква соответствует подгруппе (порядка двадцати подгрупп), вторая – виду (от трех до семнадцати видов в подгруппе);

4) порядковый номер разработки данной микросхемы внутри своего вида в данной серии.

Номером серии микросхемы считают первые три или четыре цифры. Для микросхем, используемых в устройствах широкого применения, перед номером серии ставится буква К. Для характеристики материала и типа корпуса микросхемы после буквы К могут быть добавлены следующие буквы: Р – для пластмассового корпуса второго вида, М – для керамического, металлического и стеклокерамического корпуса второго типа. В конце обозначения микросхемы может быть добавлена буква, конкретизирующая один из основных ее параметров.

Например: КМ155ЛА3, К561ИЕ33, 564ЛА7, КР565РУ8Г.

Корпуса цифровых микросхем бывают в основном двух видов:

1. Планарные (плоские), у этих микросхем условное обозначение корпуса начинается с цифры 4; выводы числом от четырнадцати до сорока двух расположены с двух сторон микросхемы с шагом 1.25 мм, прямые, припаиваются, как правило, к дорожкам печатной платы на стороне установки микросхем; такие корпуса часто называют *SOIC* (small outline integrated cirquit – микросхема в малом корпусе с выводами, не лежащими в одну линию). Иногда такой тип корпуса называют сокращенно – *SO*.

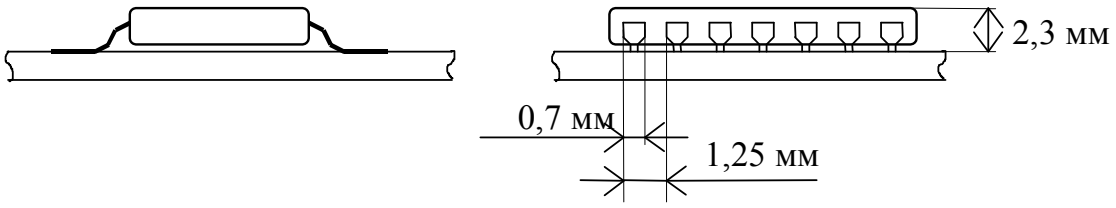


Рис. 3.1. Планарный корпус микросхемы

2. Корпус *DIP* – dual in line package – в две линии, расположенные выводы (иногда этот тип корпуса называют *DIL*, иногда чтобы указать, что корпус изготовлен из пластмассы – *PDIP*, plastic *DIP*), – корпус микросхемы, у которой обозначение корпуса начинается с цифры 2; выводы числом от четырнадцати до сорока двух с двух сторон микросхемы с шагом обычно 2,5 мм, изогнутые под углом 90º, припаиваются только в отверстиях печатных плат.

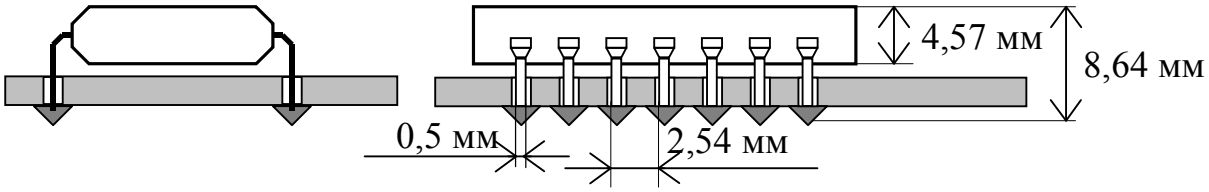


Рис. 3.2. DIP корпус микросхемы

Отечественные ТТЛ микросхемы в планарных корпусах часто имеют в обозначении серии вторую цифру 3 (133, 136), они обычно выпускаются для специального применения при температуре от –60 до 125 ºС, а в *DIP* корпусах имеют вторую цифру 5 (155,1531), выпускаются для широкого применения при температуре от –10 до 70 ºС.

Среди миниатюризированных современных корпусов микросхем, предназначенных для припаивания только на стороне установки микросхем, можно в качестве примера привести следующие:

– *SOIC* – small outline integrated circuit, при обозначении *SN…DW*.

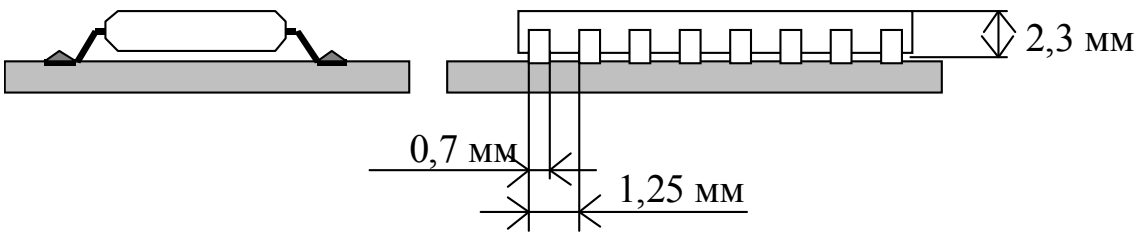


Рис. 3.3. SOIC корпус микросхемы

За рубежом в обозначении ТТЛ микросхем имеются числа 54 для микросхем специального (военного) применения и 74 – для широкого (гражданского) применения. Буквы в конце зарубежных обозначений означают: *L* – низкое потребление мощности, но низкое быстродействие; *H* – высокое быстродействие, но и большое потребление мощности; *S* – с диодами Шоттки (Schottky); *A* – улучшенные, перспективные от слова Advance (вольный перевод «аванс»); *F* – быстрые от слова Fast – быстрый. В обозначение зарубежных КМОП (*CMOS* – Complementary Metal-Oxide-Semiconductor) микросхем обычно входит число 40 (CD4011B).

Американская фирма «TEXAS INSTRUMENTS», крупнейший в мире разработчик и производитель цифровых микросхем средней интеграции, в одном из своих проспектов опубликовала график, приведенный на рис. 3.4, которым, по мнению специалистов этой фирмы, можно охарактеризовать историю развития и перспективы использования различных серий цифровых микросхем.

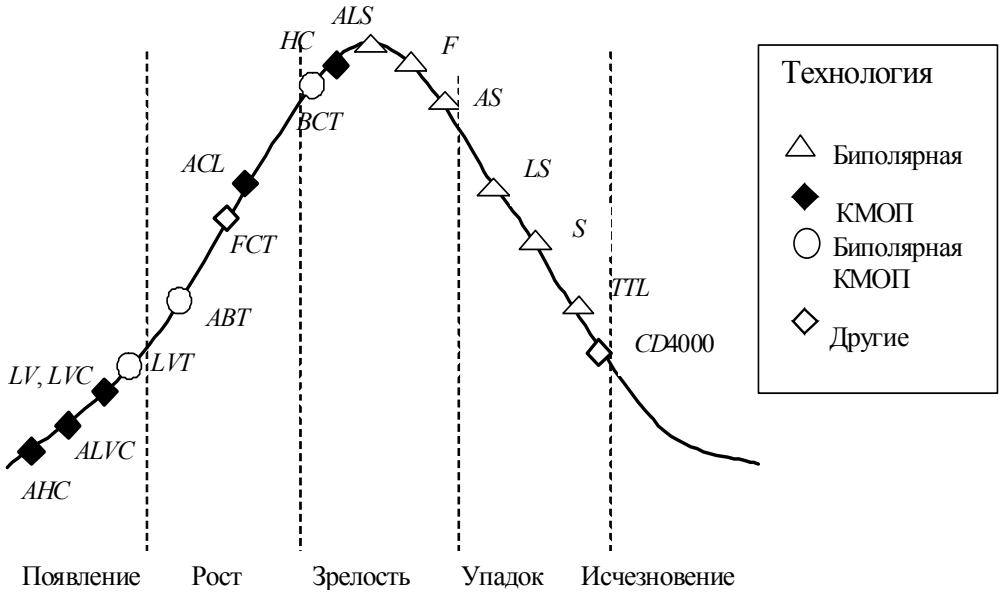


Рис. 3.4. Жизненный цикл микросхем различной технологии, по данным американской фирмы «TEXAS INSTRUMENTS»:
V/ LVC – низковольтная *CMOS* логика;
LVT – низковольтная технология;
ALVC – усовершенствованная низковольтная *CMOS* логика;
ABT – усовершенствованная *BiCMOS* технология;
BCT – *BiCMOS* технология;
F – биполярная технология серии 74*F*;
AC/ACT – усовершенствованная *CMOS* логика;
HC/HCT – высокоскоростная *CMOS* логика

3.1.2. Параметры цифровых микросхем

Параметры цифровых микросхем делятся на три группы: статические, динамические и интегральные.

Статические параметры цифровых микросхем характеризуют микросхему в статическом (установившемся) режиме. К ним относятся:

1. *Напряжение источника питания* $U_{пит}$ [В] и допуск на его изменение (для ТТЛ $U_{пит}$ = 5 В ± 5 %; для большинства же серий КМОП допустимо питание в пределах от 3 до 15 В).

2. *Входные и выходные допустимые напряжения* $U^0_{вх.max}$, $U^0_{вых.max}$, $U^1_{вх.min}$, $U^1_{вых.min}$ [В], для некоторых типов микросхем показаны на рис. 3.5.

На рис. 3.5 стрелками, направленными вниз, показаны типовые, наиболее часто встречающиеся значения уровней логического нуля и единицы, а стрелками, направленными вверх, показаны типовые значения порогов переключения микросхем.

3. *Входные и выходные токи* при лог. 0 и лог. 1 и их допуски: $I^0_{вх.max}$, $I^0_{вых.max}$, $I^1_{вх.min}$, $I^1_{вых.min}$.

4. *Коэффициент разветвления по выходу* (это число входов микросхемы данной серии, которые допустимо подключать к данному выходу микросхемы), обычно для ТТЛ $K_{разв}$ = 10, для КМОП $K_{разв}$ = 50…100.

5. *Коэффициент объединения по входу* $K_{об}$ (обычно это число входов данной микросхемы). Как правило, $K_{об}$ = 2; 3; 4 и 8. Если нужно другое число, то применяют специальные микросхемы – расширители или собирают схему по законам булевой алгебры.

6. *Потребляемая мощность* (*статическая*) обычно рассматривается как полусумма мощностей, потребляемых при нуле и при единице:

$$P_{пот} = \frac{(P^0_{пот} + P^1_{пот})}{2}.$$

Мощность, потребляемая микросхемами в реальных режимах работы, существенно зависит от частоты их переключения, как это показано на рис. 3.6.

7. Помехоустойчивость (статическая) $U_{пом}$ – допустимое напряжение помех на входе микросхемы, определяется из двух значений:

$$\left|U^1_{вых.min} - U^1_{вх.мах}\right| = U^1_{пом}, \text{ или } \left|U^0_{вых.мах} - U^0_{вх.min}\right| = U^0_{пом}.$$

Из этих двух значений выбирается меньшее. Эти значения помехоустойчивости даны для предельных значений питающих напряжений, температуры окружающей среды и других условий. Реальная помехоустойчивость микросхем примерно в два раза лучше, чем определяемая по приведенной формуле.

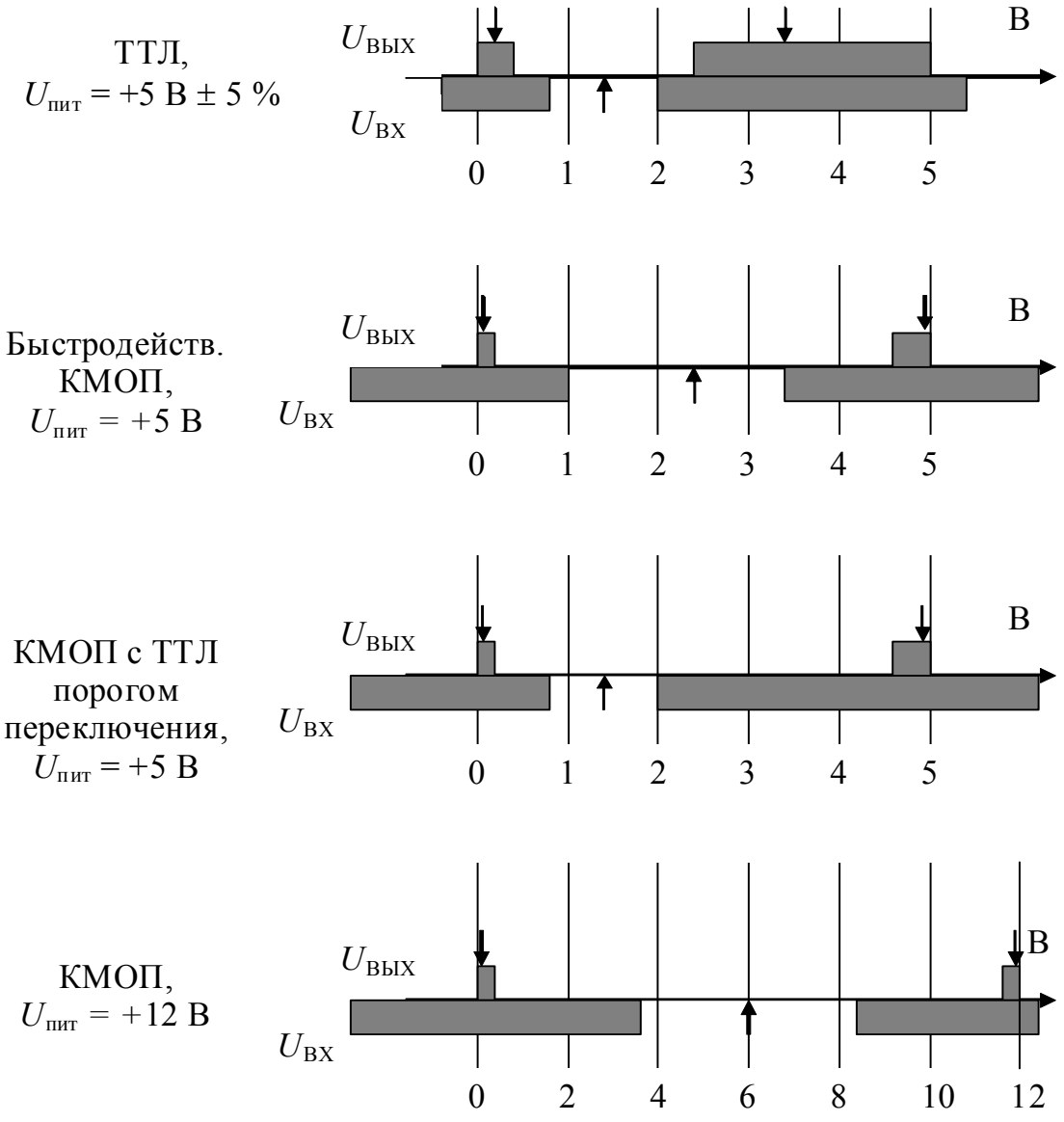


Рис. 3.5. Области допустимых значений входных и выходных напряжений ТТЛ и КМОП микросхем при напряжении их питания +5 В и +12 В

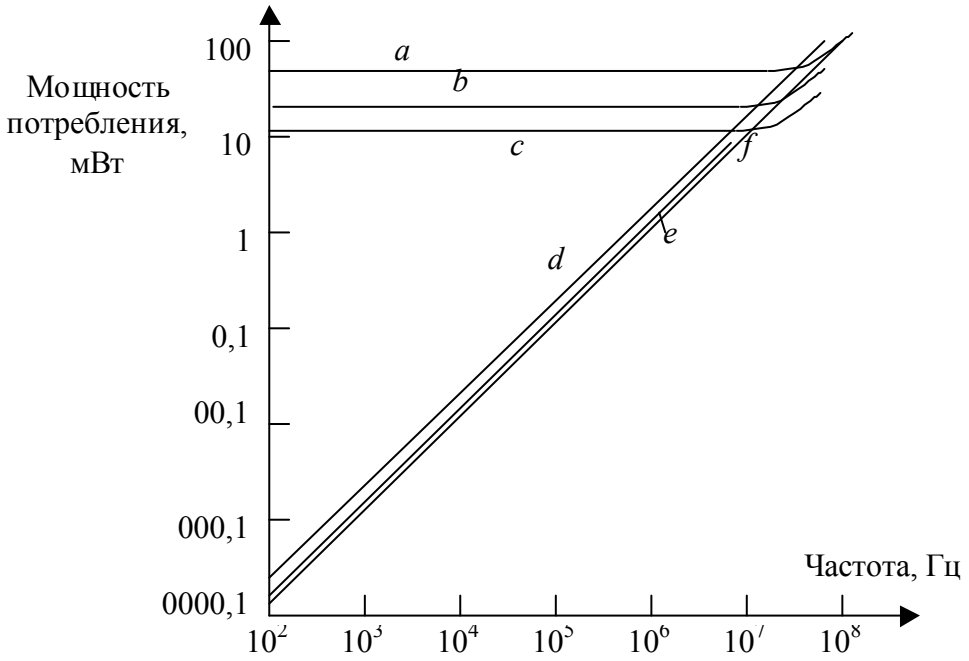


Рис. 3.6. Зависимость мощности, потребляемой микросхемами различных серий, от частоты переключения:
a – ТТЛ микросхемы серии КР1531 (*F*);
b – ТТЛ микросхемы серии К555 (*LS*);
c – ТТЛ микросхемы серии КР1533 (*ALS*);
d – КМОП микросхемы серии 1564 (*HCT*);
e – КМОП микросхемы серии К561 и КР1561 (74*C*);
f – КМОП микросхемы серии КР1554 (*ACT*)

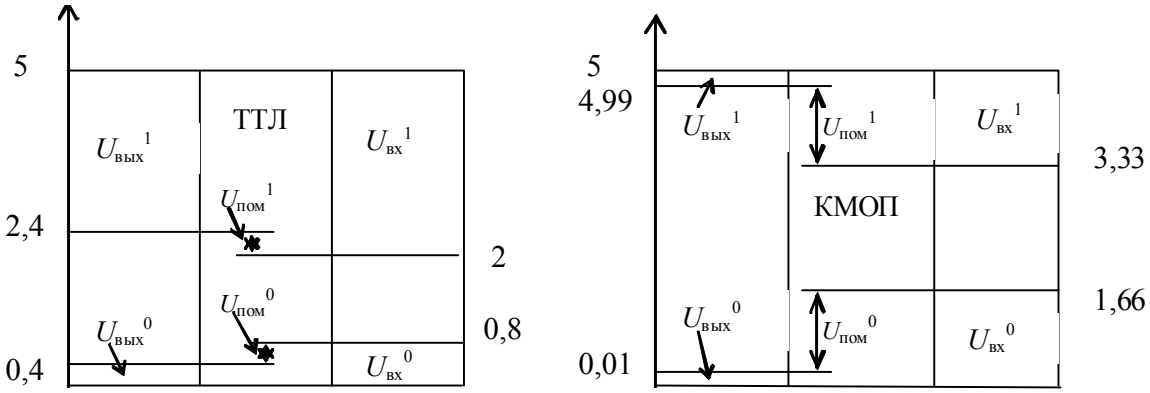


Рис. 3.7. Области допустимых значений входных и выходных напряжений ТТЛ и КМОП микросхем при напряжении их питания +5 В и напряжения статической помехи при нулевом и единичном уровнях

В зависимости от продолжительности помехи различают *статическую* и *динамическую помехоустойчивость*. Статическую помехоустойчивость связывают с помехами, длительность которых больше времени переходных процессов, а динамическую – с кратковре-

менными помехами. Динамическая помехоустойчивость лучше статической за счет того, что от короткого импульса помех микросхемы просто не успевает переключиться.

Динамические параметры цифровых микросхем определяют максимальную частоту смены входных состояний, при которой не нарушается нормальное функционирование микросхем. Временные диаграммы, иллюстрирующие способы определения временных динамических параметров, показаны на рис. 3.8.

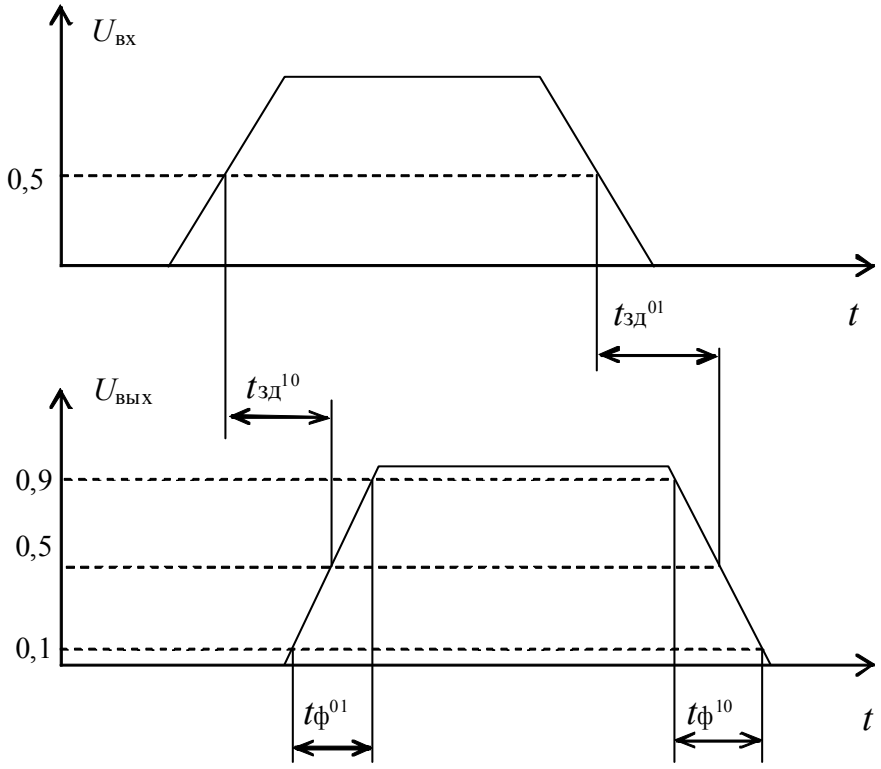


Рис. 3.8. Временные диаграммы, иллюстрирующие способы определения временных динамических интегральных микросхем

К *динамическим* параметрам цифровых интегральных микросхем обычно относят следующие:

1. *Среднее время* $t_{зд.р}$ *задержки распространения* переднего и заднего фронтов:

$$t_{зд.р} = \frac{t_{зд.р}^{0,1} + t_{зд.р}^{1,0}}{2}.$$

2. *Длительность* $t_{ф}^{0,1}$ *фронта* выходного сигнала при перепаде от нуля к единице (от 0,1 до 0,9 амплитуды сигнала) и $t_{ф}^{1,0}$ длительность

фронта выходного сигнала при перепаде от 1 к 0 (от 0,9 до 0,1 амплитуды сигнала), как это показано на рис. 3.6.

3. Максимальная частота переключения:

$$f_{\max} \le \frac{(t_{\text{ф}}^{0,1} + t_{\text{ф}}^{1,0})}{2}.$$

Интегральные параметры цифровых микросхем отражают уровень развития технологии и схемотехники, а также качество цифровых микросхем:

1. *Энергия переключения* $\text{Э}_{\text{пер}} = P_{\text{потр.ср}} \cdot t_{\text{зд.р}}$ [пДж].

2. *Степень интеграции* $N = \lg n$, где n – число простых логических элементов (2И–НЕ) на кристалле (при N = 2 микросхемы обычно называют схемами средней интеграции – СИС; при N = 3 микросхемы обычно называют схемами большой интеграции – БИС; при N = 4 микросхемы обычно называют схемами сверх большой интеграции – СБИС).

3.2. Схемы, параметры и характеристики базовых логических элементов стандартных серий цифровых

3.2.1. Диодные схемы И, ИЛИ, транзисторная схема НЕ, диодно-транзисторная логика

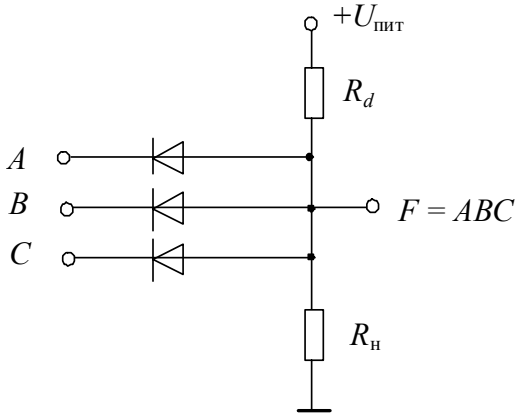


Рис. 3.9. Диодная схема И

Если на входах A и B и C схемы И будут высокие уровни, то на выходе F будет высокий уровень. Если хотя бы на одном из входов будет ноль напряжения, то на выходе будет низкий уровень, близкий к нулю, отличающийся от него на величину падения напряжения на

диоде $U_{пр}$= 0,5 В. Высокий уровень на выходе схемы И при высоких уровнях на всех входах будет в том случае, если $R_d < R_н$.

Если на входе *A* или *B* или *C* диодной схемы ИЛИ будет высокий уровень, то на выходе *F* также будет высокий уровень. Если же на всех входах будут низкие уровни, то и на выходе будет низкий уровень, но только при условии, обратном тому, которое предъявляется в схеме И, а именно $R_d > R_н$.

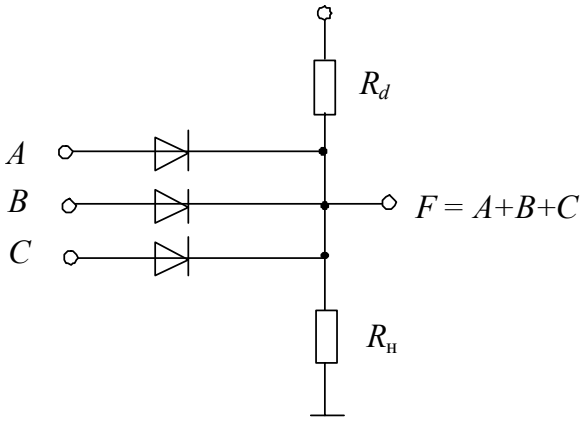


Рис. 3.10. Диодная схема ИЛИ

Инвертирующий усилитель на транзисторе включен по схеме с общим эмиттером и кроме усиления сигнала реализует логическую функцию НЕ. Ток базы $I_{б.гр}$ для данного $U_{пит}$ и данного $R_к$ определяет границу между областью активного (усилительного) режима и областью насыщения транзистора. Чем больше базовый ток по отношению к $I_{б.гр}$, т. е. больше степень насыщения транзистора, тем больше потребуется времени для рассасывания заряда в базе транзистора после его закрывания.

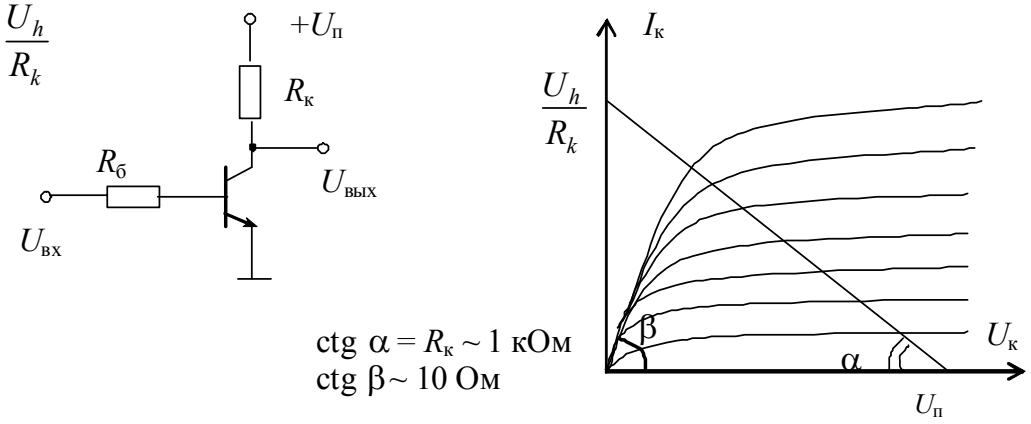


Рис. 3.11. Транзисторная схема НЕ и ее выходная характеристика

Обеспечить выходное напряжение логического нуля достаточно низким, чтобы иметь какой-то запас от помех, можно, если режим транзистора при логическом нуле на выходе устанавливать в области насыщения или вблизи ее. Транзисторы при изготовлении имеют большой разброс по усилению: в 3…5 раз. Выбирают номинал базового резистора $R_б$ таким, чтобы при подаче на вход инвертора заданного $U^1_{вх.min}$ даже транзисторы с минимальным усилением находились в области насыщения или вблизи ее, тогда все остальные транзисторы с неминимальным усилением входят в режим глубокого насыщения.

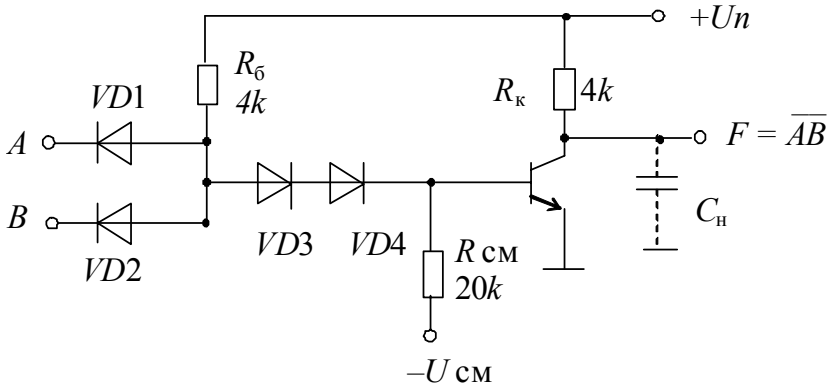


Рис. 3.12. Схема простейшего логического элемента диодно-транзисторной логики

При соединении входной диодной схемы И с выходным транзисторным инвертором получается одна из простейших логических микросхем. Диоды *VD*3, *VD*4 и цепь смещения $R_{см}$, $–U_{см}$ введены для обеспечения работоспособности и увеличения помехоустойчивости схемы при уровне логического нуля на входе.

Считаем, что вход данного логического элемента подключен к входу такого же логического элемента. Паспортные данные на такие ДТЛ микросхемы: $U^0_{вых.max}$ = 0,5 В, прямое падение на входном диоде $U_{пр.д.max}$= 0,9 В. Следовательно, при лог. 0 на выходе предыдущего логического элемента и, соответственно, на входе данного логического элемента напряжение после входного диода в точке *X* соединения анодов входных диодов может быть следующим:

$$U^0_X = U^0_{вых.max} + U_{пр.д.max} = 0{,}5 + 0{,}9 = 1{,}4 \text{ В}.$$

Если бы не было диодов *VD*3 и *VD*4, то напряжение логического нуля U^0_X в точке *X*, напрямую попадая на базу выходного транзистора, может быть слишком большим, чтобы этот транзистор закрылся, т. е.

может быть так, что выходной транзистор не будет закрываться ни при логической единице, ни при логическом нуле на его входе.

При наличии диодов *VD*3, *VD*4 и цепи смещения $R_{см}$, $-U_{см}$, ток, протекающий от точки *X* через диоды *VD*3 и *VD*4 и резистор $R_{см}$ к источнику смещения $-U_{см}$, создает на этих диодах падение напряжения, равное двойному $U_{пр.д.max}$, т. е. $U_{пр}$ = 1,8 В. Напряжение на базе транзистора $U_б$ при логическом нуле на входе при этом будет ниже напряжения в точке *X* на величину падения напряжения на открытых диодах *VD*3 и *VD*4 (исходя из направления тока в диодах), т. е.

$$U_б = U_X - 2U_{пр.д.max} = 1{,}4 - 1{,}8 = -0{,}4 \text{ В}.$$

Поскольку для начала открывания транзистора *n–p–n* необходимо положительное напряжение хотя бы 0,1 В, можно считать, что у данной микросхемы статическая помехоустойчивость составит:

$$U^0_{пом} = 0{,}1 - (-0{,}4) = 0{,}5 \text{ В}.$$

Главные недостатки таких схем:

а) низкая нагрузочная способность $K_{разв}$ = 5;

б) низкое быстродействие: $t^{0,1}_{зд.р}$ = 110 нс, $t^{1,0}{}_{зд.р}$ = 20 нс. Большая разница между задержками по фронтам 0,1 и 1,0 объясняется тем, что заряд емкости нагрузки логического элемента при закрывании выходного транзистора (фронт 0,1) происходит через коллекторный резистор $R_к$ = 4 кОм, а разряд – через открытый транзистор, имеющий существенно меньшее внутреннее сопротивление $R_{откр}$, чем $R_к$ ($R_к$ = 50...130 Ом, а $R_{откр} \approx 1$ Ом).

Усовершенствование микросхем ДТЛ путем усложнения выходного инвертора, как показано на рис. 3.13, позволило увеличить нагрузочную способность ($K_{разв}$ = 16) и быстродействие ($t^{0,1}_{зд.р} = t^{1,0}_{зд.р}$ = 40 нс).

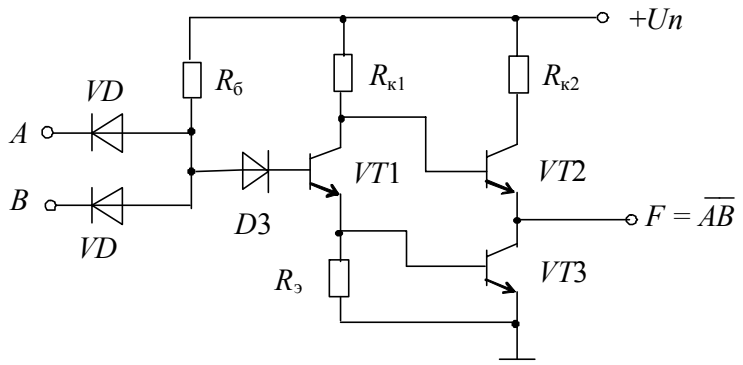


Рис. 3.13. Схема усовершенствованного логического элемента диодно-транзисторной логики

В этой схеме выходные транзисторы *VT*2 и *VT*3 собраны по двухтактной схеме (push-pull, – «тяни-толкай»). Они открываются поочередно. Транзистор *VT*1 называют фазоразделительным. Низкоомный резистор $R_{к2}$ ограничивает импульс тока, протекающий через транзисторы *VT*2 и *VT*3 при переключении, когда один из этих транзисторов открывается, а второй – закрывается.

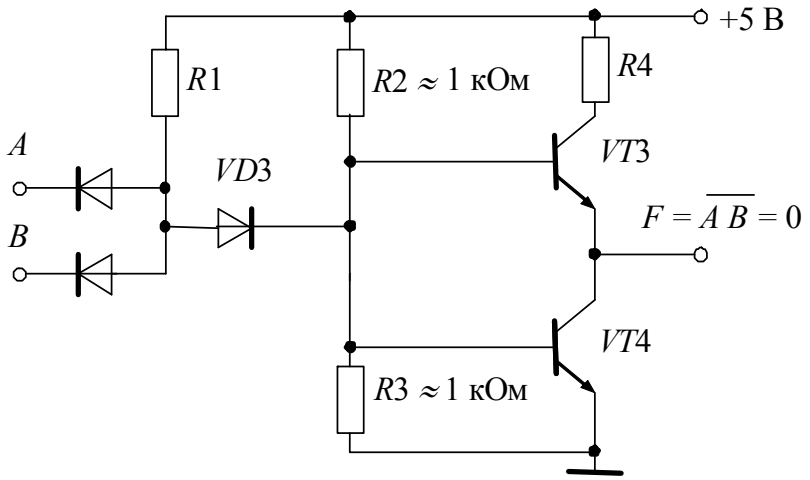


Рис. 3.14. Схема ДТЛ логического элемента при логических единицах на всех входах

Для анализа работы такого выходного каскада можно использовать следующий прием – допустить, что в одном случае (при логической единице на обоих входах *A* и *B*) фазоинвертирующий транзистор представляет собой короткое замыкание между коллектором и эмиттером, как показано на рис. 3.14, а в другом состоянии этот транзистор представляет собой разрыв, т. е. его как бы нет, что показано на рис. 3.15.

В первом случае, показанном на рис. 3.14, очевидно, что если не учитывать влияние верхнего выходного транзистора, то нижний выходной транзистор *VT*4 будет открыт. При этом напряжение на его коллекторе будет выше уровня нуля (хотя бы на долю вольта), следовательно, и потенциал эмиттера верхнего выходного транзистора будет выше уровня нуля, выше уровня эмиттера нижнего выходного транзистора. Поэтому можно считать, что для верхнего выходного транзистора напряжения на его базе будет недостаточно для его открывания, хотя этого напряжения и хватает для открывания нижнего выходного транзистора.

Следовательно, верхний выходной транзистор будет закрыт и не будет влиять на работу нижнего выходного транзистора, что подтверждает правильность сделанного выше предположения. Следова-

тельно, при логических единицах на всех входах ДТЛ логический элемент имеет на своем выходе напряжение логического нуля.

Анализ работы схемы, приведенной на рис. 3.15, еще более прост. Здесь сразу очевидно, что верхний выходной транзистор будет открыт, а нижний выходной транзистор – закрыт.

Следовательно, при хотя бы одном логическом нуле на входах ДТЛ логический элемент имеет на своем выходе напряжение логической единицы.

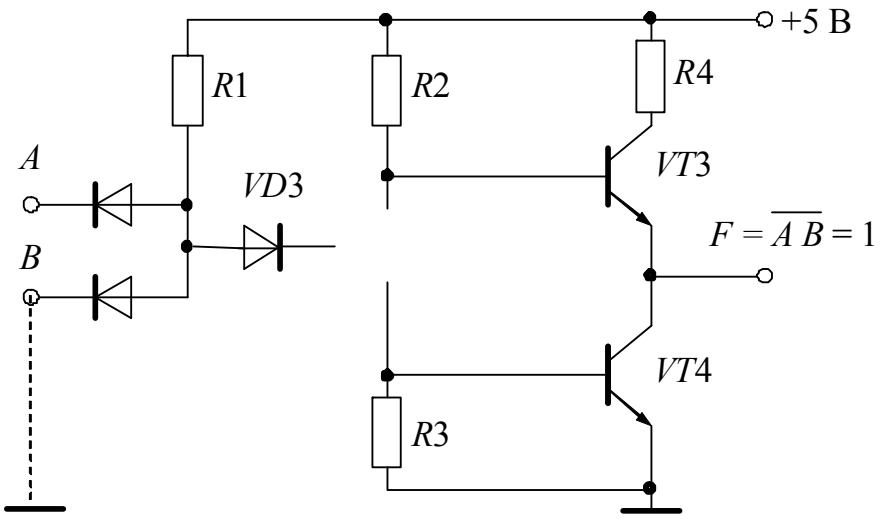


Рис. 3.15. Схема ДТЛ логического элемента при хотя бы одном логическом нуле на входах

3.2.2. Транзисторно-транзисторная логика

Основное отличие логических элементов ТТЛ от ДТЛ логических элементов состоит в том, что в ТТЛ логических элементах входные диоды заменены одним многоэмиттерным транзистором (МЭТ). В первом приближении *p–n* переходы база–эмиттер многоэмиттерного транзистора *VT*1 можно считать входными диодами, как у схем ДТЛ, а переход база–коллектор многоэмиттерного транзистора – помехозащитным диодом в базе фазоразделительного каскада. Так, эта схема ТТЛ и работает при уровне логической единицы на входах *A*, *B* и *C*. Когда же на один из входов поступает логический ноль, многоэмиттерный транзистор *VT*1 становится транзистором, включенным по схеме ОБ.

При таком переходе входа многоэмиттерного транзистора из логической единицы в логический ноль транзистор фазоразделительного каскада *VT*2 из открытого (насыщенного) состояния переходит в

закрытое, при этом накопленный в базе *VT*2 заряд быстро рассасывается через открытый многоэмиттерный транзистор *VT*1, что существенно уменьшает время переключения всего ТТЛ элемента по сравнению с ДТЛ.

Особенностью многоэмиттерного транзистора в открытом состоянии является отсутствие прямого взаимодействия эмиттеров между собой, так как их разделяют участки базы. Можно считать, что многоэмиттерный транзистор во включенном состоянии – это несколько транзисторов, имеющих общий коллектор.

Трехтранзисторный выходной инвертор ТТЛ отличается от рассмотренного ранее при анализе ДТЛ логического элемента введением диода *VD*1 в эмиттерную цепь верхнего транзистора *VT*4 двухтактного выходного каскада. Этот диод обеспечивает более надежное запирание транзистора *VT*4 при логическом нуле на входе логического элемента.

Развитие схемотехники ТТЛ вентиля (логического элемента) привело в основном к следующему:

1) введению антизвонных диодов (*VD*1…*VD*3 на рис. 3.17) на входах схемы (введены во всех сериях ТТЛ, название «антизвонные» встречается в различных источниках, в частности в [2, с. 31, 35]);

2) замене эмиттерного сопротивления *R*3 фазоразделительного каскада на генератор тока, схема которого собрана на *VT*6, *R*4, *R*5 (называется иногда динамической нагрузкой), который обеспечивает быстрое рассасывание накопленных зарядов в области базы нижнего транзистора *VT*5 выходного двухтактного каскада при закрывании этого транзистора *VT*5;

3) замене выходного эмиттерного повторителя верхнего транзистора *VT*4 выходного двухтактного каскада на составной эмиттерный повторитель на двух транзисторах *VT*3 и *VT*4 для уменьшения выходного сопротивления в состоянии логической единицы на выходе, для выравнивания этого сопротивления с сопротивлением при логическом нуле на выходе;

4) замене всех или части транзисторов логических элементов транзисторами Шоттки (Schottky) для уменьшения времени их переключения.

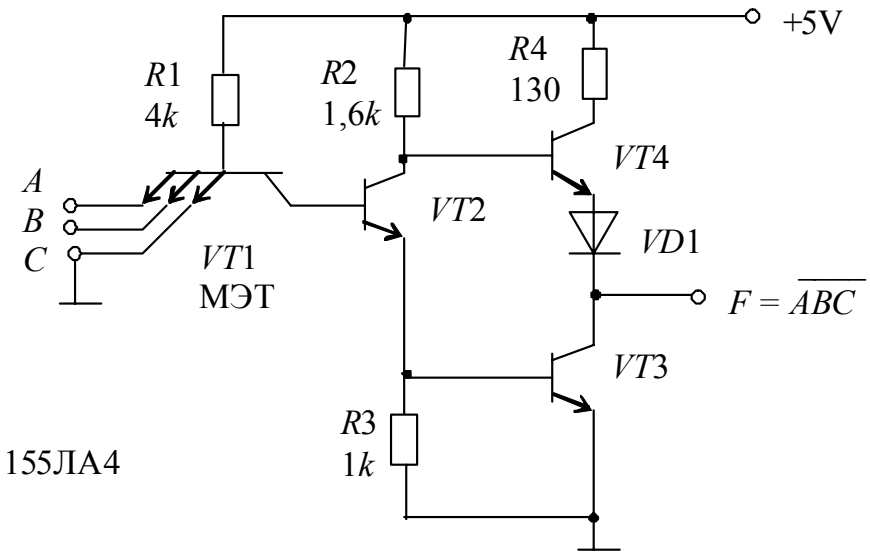


Рис. 3.16. Схема простейшего логического элемента транзисторно-транзисторной логики

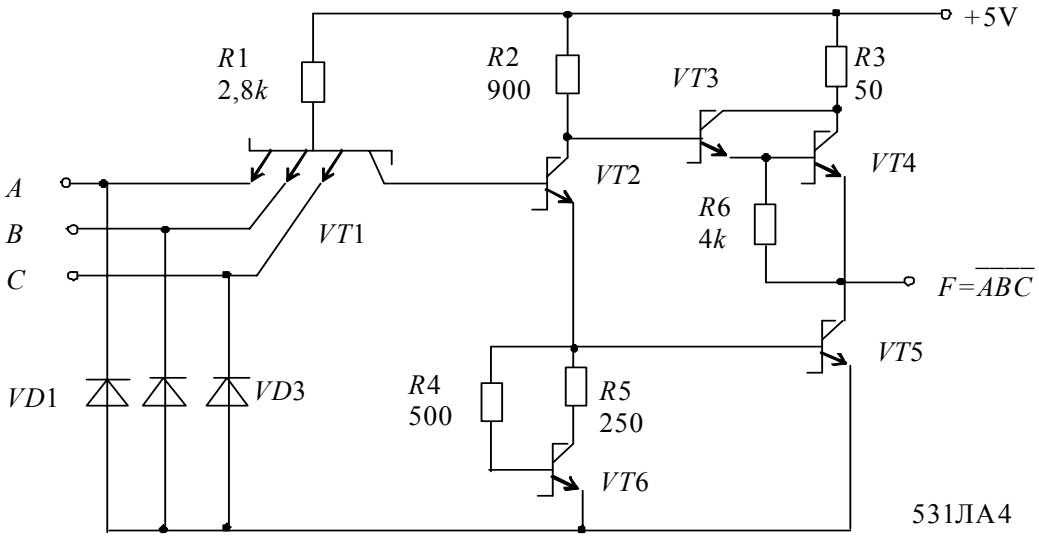


Рис. 3.17. Схема усовершенствованного логического элемента транзисторно-транзисторной логики

Комбинации 2, 3 и 4 изменений, а также значения внутренних резисторов определяют различия между сериями ТТЛ, при этом изменяются быстродействие и потребляемая микросхемами мощность.

3.2.3. Транзисторы Шоттки

Транзисторы Шоттки отличаются от обычных тем, что они не входят в глубокое насыщение, следовательно, в их базах в открытом состоянии накапливается мало носителей заряда, и в результате время их рассасывания меньше обычного.

Эффект Шоттки снижает напряжение открывания кремниевого *p–n* перехода от обычных 0,5...0,7 В до 0,2...0,3 В и значительно уменьшает время жизни неосновных носителей в полупроводнике. Эффект Шоттки основан на том, что в *p–n* переходе или рядом с ним присутствует очень тонкий слой металла, богатый элементами, свободный носителями.

Транзистор Шоттки можно представить как обычный транзистор с диодом Шоттки, включенным между его базой и коллектором, как показано на рис. 3.18.

При открывании транзистора базовый ток нарастает только до значения, лежащего на границе активного режима и области насыщения, а весь избыточный базовый ток отводится через открытый диод Шоттки через коллектор и эмиттер открытого транзистора на землю.

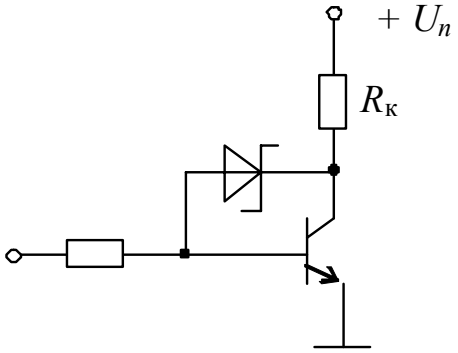


Рис. 3.18. Транзистор Шоттки, представленный как обычный транзистор с диодом Шоттки, включенным между его базой и коллектором

Чем сильнее откроется транзистор, т. е. чем меньше падение напряжения коллектор–эмиттер, тем больший ток отводится через диод Шоттки, минуя базу, на землю. Это приведет к закрыванию транзистора, так как уменьшение тока базы закрывает транзистор. Так образуется обратная связь, саморегулирующая режим работы транзистора, удерживая его от глубокого насыщения.

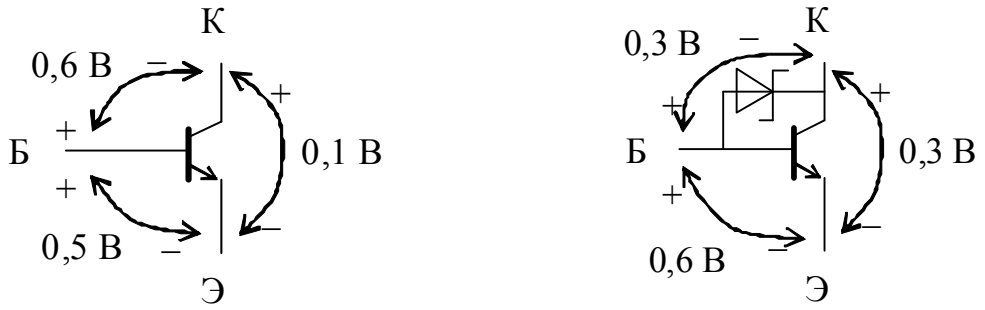
Сами диоды Шоттки имеют очень малые задержки включения и выключения. Накопление заряда в диодах Шоттки не происходит, так как протекающий в них ток вызван переносом основных носителей.

Когда транзистор заперт, потенциал коллектора выше потенциала базы, а значит, диод Шоттки смещен в обратном направлении и не влияет на работу транзистора.

Если в процессе отпирания транзистора потенциал коллектора становится ниже потенциала базы, диод Шоттки открывается и на нем устанавливается прямое напряжение $U_{пр}$. Поскольку это напряжение

меньше 0,5 В, то коллекторный переход практически заперт, а следовательно, не возникает режима насыщения и связанных с ним двойной инжекции и накопления избыточных зарядов. Благодаря этому при запирании транзистора исключается задержка, вызываемая рассеиванием избыточного заряда.

На рис. 3.19 показана разность потенциалов между выводами обычного транзистора и транзистора Шоттки, подтверждающая большее напряжение между коллектором и эмиттером транзистора Шоттки в открытом состоянии.



Обычный транзистор Транзистор Шоттки

Рис. 3.19. Разность потенциалов между выводами обычного транзистора и транзистора Шоттки

3.2.4. Состав ТТЛ логических микросхем малой интеграции

Логический элемент, служащий основой для построения других сложных микросхем, называется базовым. В ТТЛ базовым логическим элементом считается элемент 4–НЕ. Микросхема, содержащая в своем корпусе два таких логических элемента – 2 (4И–НЕ), имеет в своем обозначении символы ЛА1 (К155ЛА1) [4].

В ТТЛ сериях логические элементы И–НЕ распространены более других. Они производятся в виде самостоятельных микросхем, а также служат основой для построения сложных комбинационных и последовательностных микросхем средней и большой интеграции. Кроме базового элемента ЛА1 – 2 (4И–НЕ), в ТТЛ выпускаются логические элементы ЛА2 – 8И–НЕ, логические элементы ЛА3 – 4 (2И–НЕ), логические элементы ЛА4 – 3 (3И–НЕ).

Чтобы реализовать функцию ИЛИ, в логические элементы ТТЛ вводят несколько каскадов с фазоразделительными транзисторами, выходы этих транзисторов соединяют параллельно (коллекторы с коллекторами, эмиттеры с эмиттерами), как показано на рис. 3.20.

Если входные многоэмиттерные транзисторы таких микросхем имеют по несколько входов И, то получается схема логического элемента ЛР – И–ИЛИ–НЕ, очень распространенная в ТТЛ (в ТТЛ сериях выпускается тринадцать типов логических элементов И–ИЛИ–НЕ).

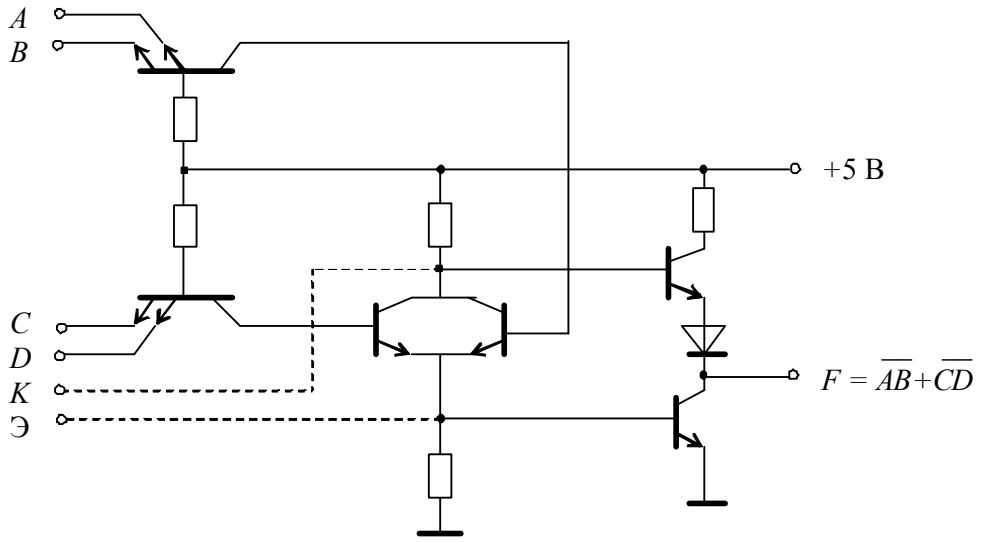


Рис. 3.20. Микросхема 2И–2ИЛИ–НЕ с выводами коллектора и эмиттера фазоразделительного каскада, предназначенными для подключения расширителей по ИЛИ

В некоторых видах этих микросхем И–ИЛИ–НЕ точки соединений коллекторов и эмиттеров подведены к выводам микросхемы, как показано на рис. 3.20, пунктирными линиями, что позволяет подключать дополнительные входные каскады с целью расширения по ИЛИ. При изображении логического элемента И–ИЛИ–НЕ или расширителей с выходами «коллектор» и «эмиттер» эти выводы отмечают наклонными крестами (рис. 3.21), в отличие от логических выводов. На обозначенных крестами выводах микросхем потенциалы могут отличаться от уровней, заданных требованиями для логических уровней.

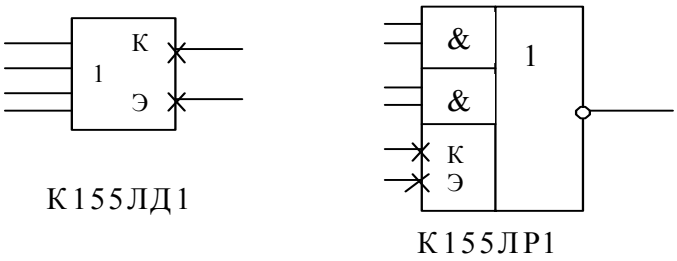


Рис. 3.21. Условные графические обозначения расширителя по ИЛИ и логического элемента 2И–2ИЛИ–НЕ, «нелогические» выводы которых отмечены наклонными крестами

Для расширения по входу ИЛИ в ТТЛ выпускают специальные расширители по ИЛИ на четыре И входа (ЛД1), показанный на рис. 3.21 и 3.22, и на восемь И входов (ЛД3) [3].

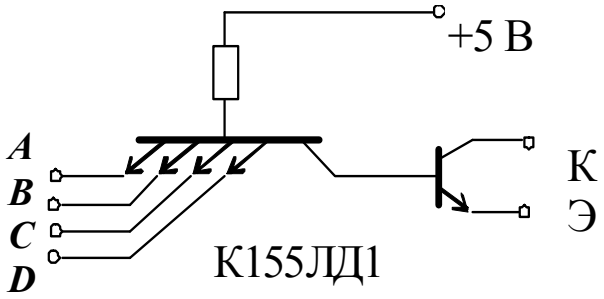


Рис. 3.22. Схема расширителя по ИЛИ с выходами коллектора и эмиттера, предназначенными для подключения параллельно к выводам коллектора и эмиттера фазоразделительного каскада логического элемента И–ИЛИ–НЕ

Подключив выводы коллектора и эмиттера фазоразделительного каскада расширителя 4И к выводам коллектора и эмиттера фазоразделительного каскада микросхемы 2И–2ИЛИ–НЕ, можно получить логический элемент (2–2–4)И–3ИЛИ–НЕ, как это показано на рис. 3.23.

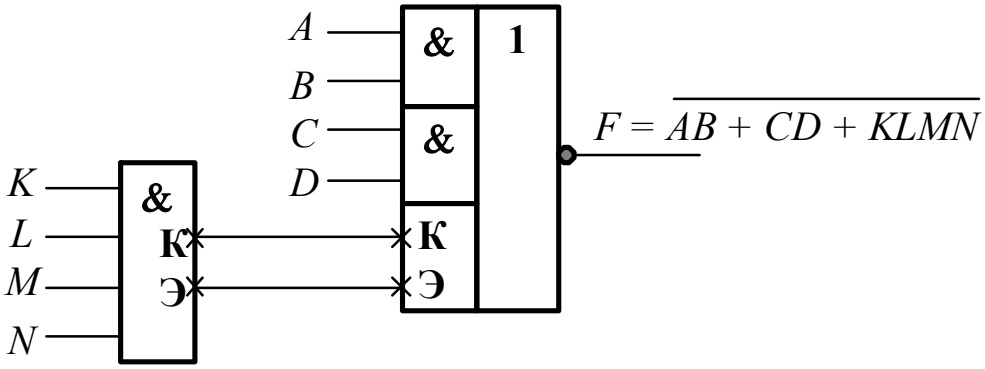


Рис. 3.23. Схема соединения четырехвходового расширителя по ИЛИ с выходами коллектора и эмиттера, подключенными параллельно к выводам коллектора и эмиттера фазоразделительного каскада логического элемента 2И–2ИЛИ–НЕ, создающего логический элемент (2–2–4)И–3ИЛИ–НЕ

Увеличение числа входов И (расширение по И) можно получить, пользуясь элементами из нескольких схем И–НЕ, пользуясь постулатами де Моргана. Тот же результат может быть получен путем подключения дополнительных внешних диодов и резистора к любому из И входов, как показано на рис. 3.24.

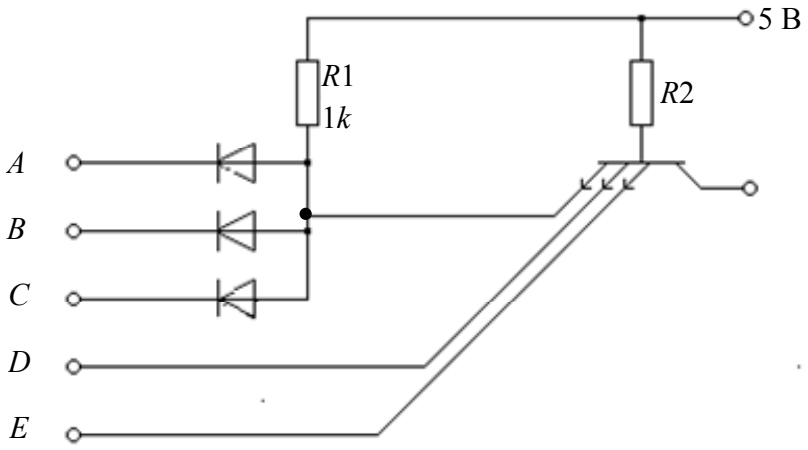


Рис. 3.24. Схема расширения по И с помощью дополнительных внешних диодов и резистора

Такое подключение снижает быстродействие и помехоустойчивость микросхемы по И входу, к которому подключены внешние диоды. Статическая помехоустойчивость ТТЛ-микросхем составляет 0,4 В. При подаче на один из дополнительных внешних диодов уровня логического нуля падение напряжения на этом открытом диоде практически полностью скомпенсирует напряжение статической помехоустойчивости, поэтому любая помеха при нуле на входе может привести к ложному срабатыванию.

Что касается быстродействия такого подключения, то здесь существенно (почти на порядок) увеличивается длительность фронта переключения из нуля в единицу на том входе логического элемента, к которому подключены дополнительные внешние диоды. Это происходит из-за того, что заряд паразитной емкости этого входа происходит не через верхний выходной транзистор предыдущей микросхемы, имеющий внутреннее сопротивление в открытом состоянии 50...150 Ом, а через внешний резистор величиной 1 кОм. Если дополнительный резистор взять меньшей величины, то при этом может недопустимо возрасти нагрузка на предыдущую микросхему в состоянии логического нуля на ее выходе.

3.2.5. Выходные каскады ТТЛ-микросхем

В микросхемах ТТЛ серий распространены следующие четыре типа выходных каскадов:

1. Обычный двухтактный выходной каскад, показанный на рис. 3.16.

2. Умощненный двухтактный выходной каскад, показанный на рис. 3.17, иногда его называют буферным выходным каскадом, иногда драйвером.

3. Выходной каскад с так называемым «открытым коллектором» (ОК), в котором из двух выходных транзисторов имеется только один, нижний. В условном графическом обозначении элементов, имеющих выход с открытым коллектором, обычно изображен ромб, подчеркнутый снизу горизонтальной чертой. Изредка используются и каскады с так называемым «открытым эмиттером» (ОЭ), в котором из двух выходных транзисторов имеется только один, верхний. В условном графическом обозначении элементов, имеющих выход с открытым эмиттером, изображен ромб, подчеркнутый сверху горизонтальной чертой.

4. Выходной каскад с так называемым третьим, *Z*-состоянием, которое не является ни нулевым, ни единичным состоянием выхода микросхемы. Это состояние характеризуется очень высоким выходным сопротивлением данной микросхемы, которая при этом как бы полностью отключена от своей нагрузки. В условном графическом обозначении элементов, имеющих выход с *Z*-состоянием, обычно изображен ромб, перечеркнутый посередине горизонтальной чертой.

Не разрешается соединять друг с другом выходы микросхем с двухтактными выходными каскады как обычными, так и умощненными. Этот запрет основан на двух причинах.

Если соединить между собой двухтактные выходы двух логических элементов, то, при поступлении на входы этих логических элементов разных сигналов, на их выходах должны быть разные состояния – на выходе одного логический ноль, на выходе другого – логическая единица, и неизвестно точно какое же состояние будет в точке их соединения (обычно «побеждает» ноль).

Ток перегрузки, короткого замыкания между ними будет в основном определяться резистором коллекторной нагрузки $R_к$ выходного транзистора, подающего в точку соединения логическую единицу. Главным назначением этого резистора является ограничение очень коротких импульсов сквозного тока, протекающего через два выходных транзистора в выходном каскаде логического элемента при их переключении, когда они оба приоткрыты, поэтому при продолжительной его перегрузке потребляемая или этим резистором, или одним из открытых транзисторов мощность превысит допустимое значение и при достаточно продолжительном состоянии какой-то из логических элементов может выйти из строя.

Еще более опасно подключение выходов логических элементов к корпусу или шине питания, однако на практике при ремонте плат, при поиске неисправных логических элементов многие электронщики

кратковременно (на несколько секунд) закорачивают пинцетом выходы логических элементов на корпус, чтобы подать логический ноль на входы подозрительных логических элементов и при этом предыдущие логические элементы не сгорают.

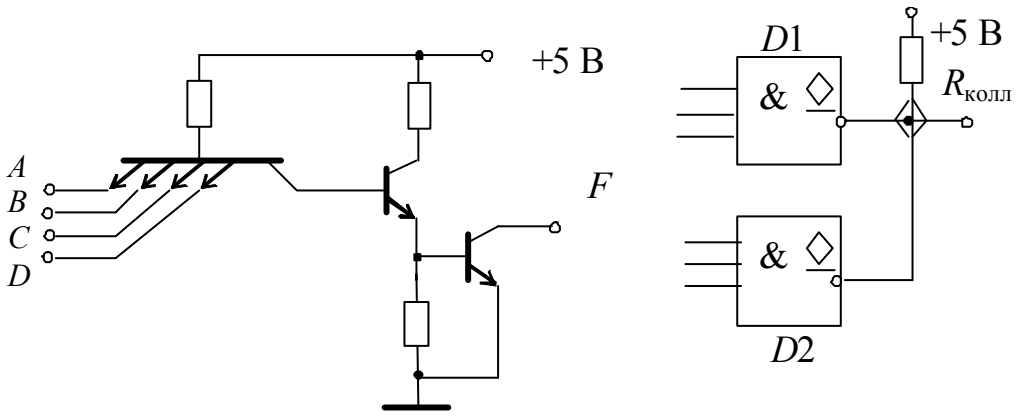
Никогда *нельзя закорачивать выходы логических элементов* ТТЛ *на шину питания*, *при этом выход предыдущей микросхемы*, *находящейся в нулевом состоянии*, *почти всегда* «*выгорает*». *Это касается всех типов выходных каскадов*.

Из четырнадцати типов логических элементов И–НЕ в ТТЛ шесть имеют выходы с открытым коллектором (ОК). Они применяются для включения реле, сегментов индикаторов, светодиодов и др. Они, в отличие от микросхем с обычным или умощненным двухтактным выходным каскадом, могут быть соединены друг с другом своими выходами без опасности выхода из строя.

Такое соединение выходов часто называют «монтажное ИЛИ», хотя с точки зрения положительной логики это схема И. Монтажным ИЛИ это соединение называют, возможно, потому, что включить ток в нагрузку (зажечь светодиод, включить реле) может микросхема *D*1 или микросхема *D*2 (рис. 3.25, *б*). Чтобы такое соединение нормально функционировало, обязательно должна быть нагрузка, подключенная к источнику питания.

Такое соединение иногда используют для подключения выходов нескольких микросхем к какой-либо информационной шине в ЭВМ. Тогда основное значение приобретает быстродействие такого подключения, которое при таком подключении ухудшается.

Ясно, что время задержки распространения $t^{0,1}_{\text{зд.р}}$ для схем с открытым коллектором и внешним $R_{\text{н}}$ значительно больше, чем у схем с обычным двухтактным выходным каскадом (для уменьшения $t^{0,1}_{\text{зд.р}}$ еще в ДТЛ – прообразе ТТЛ, ввели верхний транзистор в выходной каскад). Ясно также, что чем меньше сопротивление нагрузки $R_{\text{н}}$, тем меньше задержка, равная $t^{0,1}_{\text{зд.р}} = R_{\text{нагр}} \cdot C_{\text{нагр}}$, но $R_{\text{н}}$ можно уменьшить лишь до того предела, когда ток выходного транзистора логического элемента, определяемый из условия $I_{\text{вых}} = U_{\text{пит}} / R_{\text{нагр}}$, не достигает своего максимально допустимого значения $I^{0}_{\text{вых.max}}$. В условном графическом обозначении элементов, имеющих выход с открытым коллектором, обычно изображен ромб, подчеркнутый снизу, как это показано на рис. 3.25.



а) *б*)

Рис. 3.25. Схема логического элемента: *а* – с выходом с открытым коллектором (ОК); *б* – с параллельным подключением двух таких выходов к одной линии

У микросхем, на выходах которых из двух транзисторов двухтактного выходного каскада имеется верхний и отсутствует нижний транзистор; у микросхем с так называемым открытым эмиттером в условном графическом обозначении обычно изображен ромб, подчеркнутый сверху.

Имеются микросхемы, на выходе которых кроме двух обычных состояний логического нуля и логической единицы может быть и третье состояние *Z* с высоким выходным сопротивлением. На рис. 3.26 приведен пример такой схемы. Таблица истинности такой схемы имеет следующий вид:

*E*1	*E*2	*A*	*B*	*F*
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	*x*	*x*	*Z*
1	0	*x*	*x*	*Z*
1	1	*x*	*x*	*Z*

Здесь входы *E*1 и *E*2 (иногда такие входы обозначают двумя буквами *EO* – сокращенное ENABLE OUTPUT – разрешение выхода) управляют в соответствии с таблицей истинности состоянием выходного каскада.

В этой микросхеме два логических элемента: верхний, управляемый, и нижний, управляющий, транзисторы которого имеют обозначение, содержащее две цифры, разделенные точкой.

Если на оба входа – *E*1 и *E*2 – подана логическая единица, то в нижнем управляющем логическом элементе транзистор *VT*1.1 закрыт, *VT*1.2 открыт и *VT*1.3 открыт, следовательно, в верхнем логическом элементе напряжение на коллекторе *VT*2 близко к нулю, а тогда и напряжение на его эмиттере близко к нулю, значит, будут закрыты оба выходных транзистора *VT*3 и *VT*4.

Входы управления *Z*-состоянием *EO* следует отличать от входов *EI* (сокращенное ENABLE INPUT – разрешение входа, когда схема, аналогичная собранной на транзисторах *VT*1.1, *VT*1.2 и *VT*1.3, управляет по пунктирной линии одним из эмиттеров входного многоэмиттерного транзистора основного логического элемента, обеспечивая непрохождение сигналов *A* и *B*, отключая входы). При управлении по входам типа *EI* на выходе *F* будет состояние логического нуля, а не *Z*-состояние, как при управлении по входам типа *EO*.

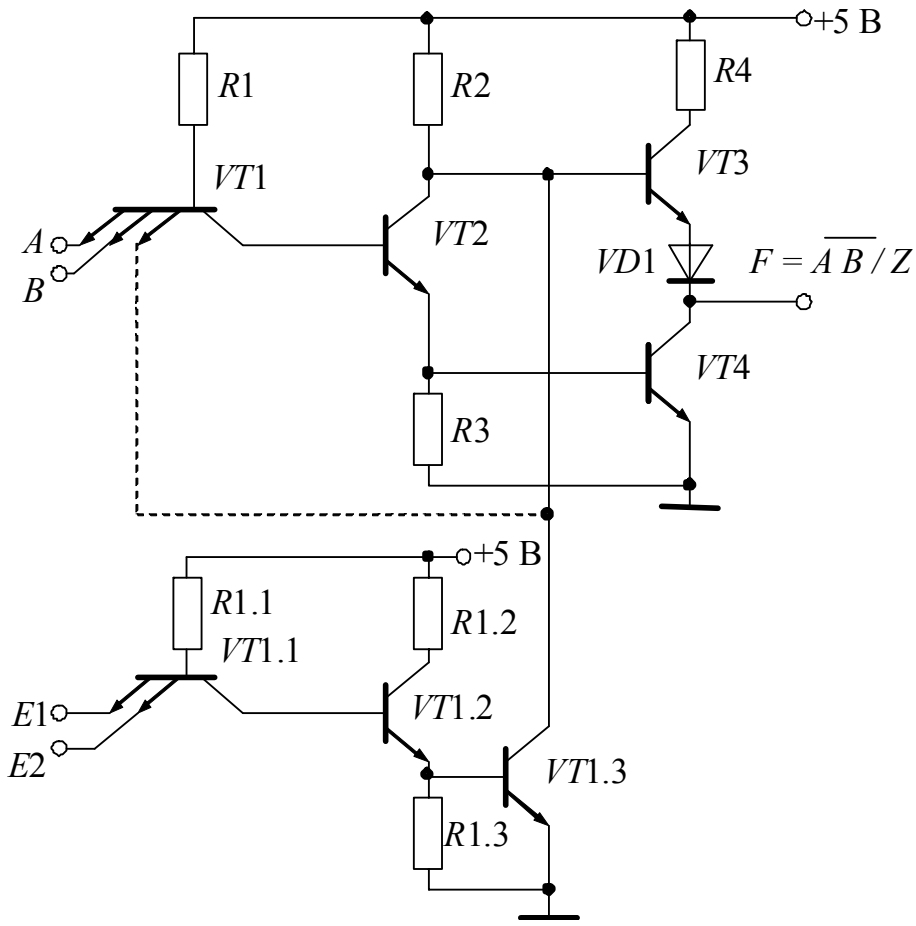


Рис. 3.26. Схема логического элемента с выходом, имеющим три возможных состояния: 0, 1 и «отключено» (*Z*-состояние), а также имеющего возможность «запрета по входу», показанную с помощью пунктирной связи

Связь, показанная на рисунке пунктирной линией, не обеспечивает *Z*-состояние, а обеспечивает только закрытие входных транзисторов (создавая так называемый запрет по входу).

Z-состояние необходимо, когда выходы нескольких логических элементов подключены к одной точке (информационной шине) и эти логические элементы работают поочередно. Итак, чтобы получить состояние *Z*, достаточно закрыть оба выходных транзистора логического элемента.

Некоторые логические элементы имеют повышенную (в 2...5 раз) нагрузочную способность, так, например, микросхемы ЛА6 – 2(4И–НЕ) и ЛА12 – 4(2И–НЕ) имеют повышенную нагрузочную способность ($K_{разв}$= 30). Их иногда называют *усилителями тока*, или *драйверами* (от английского driver – формирователь, водитель). Драйверы с *Z*-состоянием выхода широко применяются в микропроцессорных системах для подключения самого микропроцессора, памяти и внешних устройств к системным шинам адреса и данных. Такие драйверы называют *шинными формирователями* (bus driver).

Все микропроцессоры имеют двунаправленную шину данных, а значит, для соединения с этой шиной как со стороны микропроцессора, так и со стороны внешних устройств требуются двунаправленные драйверы, которые также называют приемопередатчиками. В английском языке *приемопередатчик* – transceiver происходит от слияния, комбинации частей слов *передатчик – transmitter* и приемник – receiver.

Внутри микросхем к каждому двунаправленному выводу микросхемы подключены вход одного и выход другого инвертора (или повторителя). Переключение этих выводов микросхемы с приема на передачу производится по сигналу, поступающему на управляющий вход микросхемы. В условном графическом обозначении микросхем около двунаправленных выводов обычно изображают двунаправленную стрелку, как это показано на рис. 3.27.

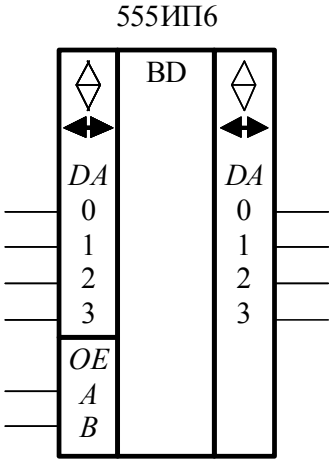


Рис. 3.27. Условное графическое обозначение двунаправленного шинного формирователя, драйвера (bus driver)

3.2.6. Входные каскады ТТЛ-микросхем

Иногда один или несколько входов логического элемента ТТЛ оказываются неиспользуемыми. Если это И входы, то их можно соединить с используемыми входами многоэмиттерного транзистора этого логического элемента, при этом входной ток практически не возрастает.

Входы И можно подключить (подвесить) через резистор R = 1 кОм (допускается подключать до 20 входов к одному резистору 1 кОм) к источнику питания +5 В, или непосредственно без резистора сколько угодно И входов можно подключать к дополнительному источнику +4 В.

Если И вход логического элемента ТТЛ ни к чему не подключен, то на нем самопроизвольно устанавливается уровень напряжения чуть выше границы логического нуля и логической единицы, так называемая «висячая единица», обычно примерно равная +1,5 В. При этом помехоустойчивость данного И входа логического элемента очень мала. В профессиональном жаргоне электронщиков термин «висячая единица» встречается достаточно часто, поэтому знание о смысле этого термина не повредит.

Неиспользуемые входы ИЛИ также можно подключать к используемым входам ИЛИ своего логического элемента, но при этом нагрузка на предыдущий выходной каскад увеличивается пропорционально числу входов ИЛИ (в отличие от входов И, количество которых из одного многоэмиттерного транзистора практически не меняет нагрузку на предыдущий логический элемент).

Свободный вход ИЛИ можно подключать к корпусной шине непосредственно или через резистор. Величину такого резистора можно найти по известным параметрам микросхем:

$$R^0_{\text{вх.max}} = U^0_{\text{вх.max}} \big/ I^0_{\text{вх.max}} .$$

Так, для большинства входов микросхем 155 серии $U^0_{\text{вх.max}} = 0{,}8$ В, $I^0_{\text{вх.max}} = 1{,}6$ мА, следовательно, максимальное значение такого резистора $R^0_{\text{вх.max}} = 500$ Ом, а минимальное, как уже упоминалось, $R_{\text{вх.min}} = 0$.

Включение такого резистора для создания $U^0_{\text{вх}}$ как на ИЛИ, так и на И входах применяется в импульсных устройствах с *RC* цепями. При этом емкость конденсатора *C* надо брать такой, чтобы длительность фронта не превышала значение $t_{\text{ф}} < 200$ нс (для 155 серии), иначе на выходе микросхемы может возникнуть «звон» – паразитные колебания.

3.2.7. Статические характеристики ТТЛ-микросхем

К статическим характеристикам относятся *входная*, *выходная* и *передаточная* характеристики.

• *Входная характеристика*

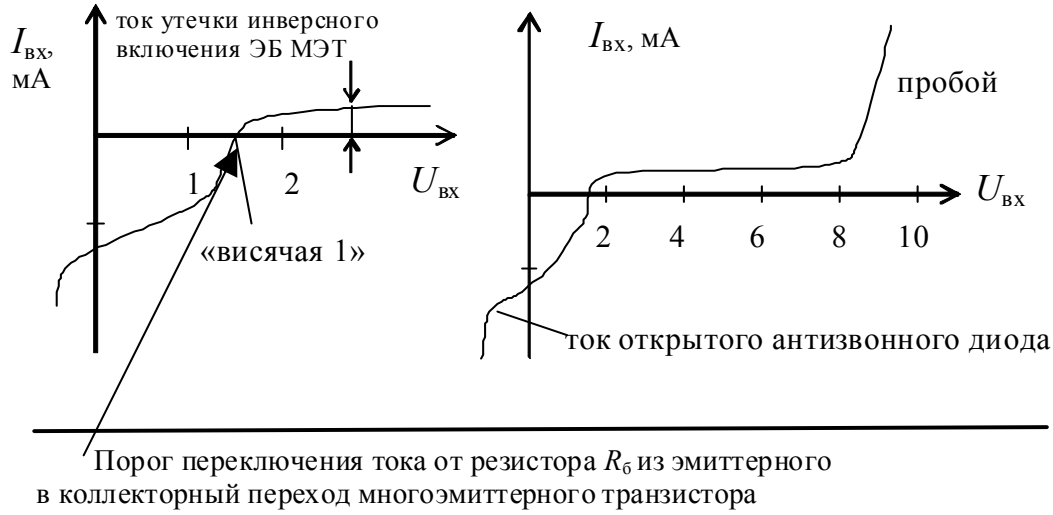


Рис. 3.28. Входная статическая характеристика ТТЛ логического элемента

Ток утечки инверсного включения перехода эмиттер–база многоэмиттерного транзистора максимальный и равен 40 мкА.

Здесь, как и на всех остальных характеристиках, направление тока, т. е. движение положительных электрических зарядов внутрь микросхемы, считается положительным, а наружу микросхемы – отрицательным (втекающий ток положительный, вытекающий отрицательный).

• *Выходная характеристика*

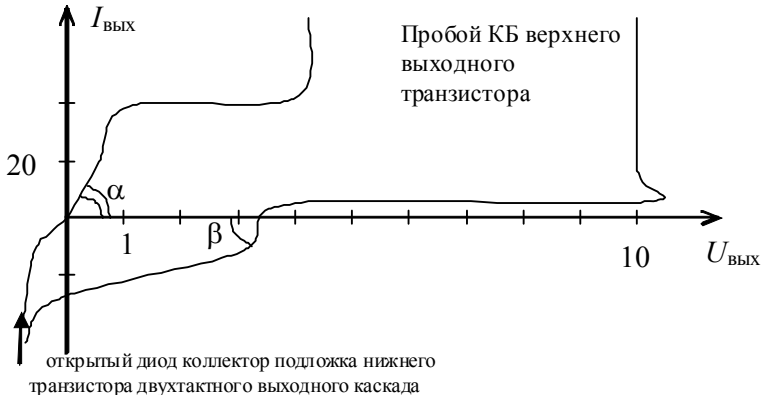


Рис. 3.29. Две ветви выходной характеристики логического элемента (верхняя ветвь относится к нулевому состоянию на выходе, нижняя – к единичному)

Наклоны участков характеристики определяются соответствующими сопротивлениями цепи микросхемы. Так, внутреннее сопротивление открытого нижнего выходного транзистора микросхемы определяет наклон верхней характеристики чуть выше нуля координат: $\operatorname{ctg}\alpha = R_i^0 \sim 10$ Ом для насыщенного транзистора *VT*5.

Внутренний резистор коллекторной нагрузки верхнего выходного транзистора микросхемы определяет наклон нижней ветви характеристики в области чуть ниже нуля ординат [5]:

$$\operatorname{ctg}\beta = R_{к2} = 130 \text{ Ом (для 155 серии)},$$

$$\operatorname{ctg}\beta = R_{к2} = 50 \text{ Ом (для 531 серии)}.$$

• Передаточная характеристика

Рис. 3.30. Передаточная статическая характеристика логического элемента ТТЛ

Здесь пунктиром показана характеристика логического элемента с генератором тока вместо эмиттерного сопротивления фазоразделительного каскада, а сплошной линией – характеристика логического элемента с обычным резистором в эмиттере фазоразделительного транзистора.

Из этой характеристики следует, что генератор тока вместо эмиттерного сопротивления в фазоразделительном каскаде не только повышает быстродействие логического элемента за счет более быстрого рассасывания накопленных зарядов в области базы нижнего транзистора выходного двухтактного каскада при закрывании этого транзистора, но и улучшает передаточную характеристику ТТЛ логического элемента.

3.3. Логика на МОП-транзисторах

В цифровых микросхемах если и применяются полевые транзисторы, то только с изолированным затвором (МОП) и только с индуцированным каналом.

МОП-транзисторы имеют четыре вывода:

1) *исток* – от которого начинают движение в канале носители электрического заряда;

2) *сток* – к которому движутся в канале носители заряда;

3) *затвор* – потенциал, на котором посредством электрического поля управляет толщиной канала (затворяет или отворяет путь протекания тока – движения носителей заряда: электронов или дырок);

4) *подложка* – полупроводниковый кремниевый кристалл, в объеме которого на малом расстоянии друг от друга методом диффузии сделаны две области с проводимостью, обратной проводимости подложки – это области стока и истока.

На поверхности подложки между стоком и истоком создают тонкий слой окисла кремния (кварцевого стекла SiO_2), а поверх него напыляют алюминиевый затвор. Когда на затвор такого МОП-транзистора относительно подложки подается отпирающее напряжение, то внутри подложки под затвором между стоком и истоком индуцируются (наводятся) носители заряда с проводимостью, противоположной проводимости подложки, но совпадающей с проводимостью стока и истока. Если теперь подключить сток и исток к источнику питания, то через них и через наведенный канал потечет ток.

Зависимости тока $I_с$ стока от напряжения затвор–подложка $U_з$ для транзистора с разной проводимостью канала приведены на рис. 3.31. и 3.32.

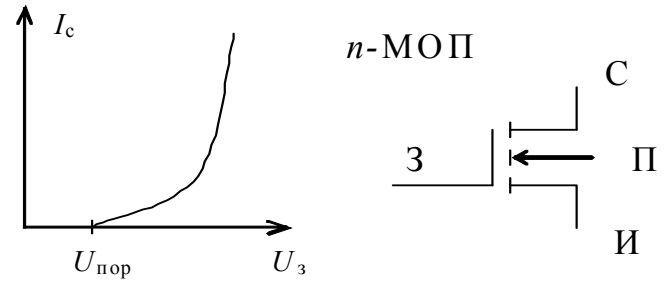


Рис. 3.31. Зависимость тока $I_с$ стока от напряжения затвор–подложка $U_з$ для транзистора с *p*-проводимостью подложки

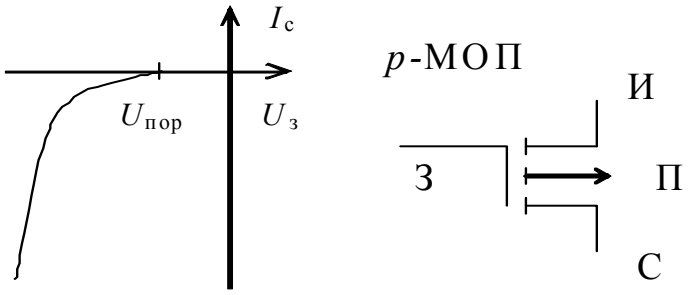


Рис. 3.32. Зависимость тока $I_с$ стока от напряжения затвор–подложка $U_з$ для транзистора с *n*-проводимостью подложки

На этих же рисунках показаны также и условные графические обозначения этих транзисторов. Следует обратить внимание на то, что вывод истока транзистора является как бы продолжением вывода его затвора. Зачастую линию канала вместо штриховой рисуют сплошной (как у МОП-транзисторов с встроенным каналом).

МОП-транзисторы называют в соответствии с проводимостью канала, а стрелки в их условных графических обозначениях соответствуют проводимости подложки, которая, как было сказано, для таких транзисторов противоположна проводимости канала. Направление стрелки показывает проводимость подложки (внутрь – транзистор *p*-типа, наружу – *n*-типа). Как известно, обозначения *n* – negative (отрицательный, при этом носители заряда – электроны), а *p* – positive (положительный, при этом носители заряда – дырки).

Значение порогового напряжения при питании +5 В обычно лежит в пределах: $U_{пор}$ = 1,5...3 В. При другом напряжении питания значение порогового напряжения $U_{пор} \approx 0{,}5\ U_{пит}$.

Для МОП-транзисторов характерны два существенных отличия их от обычных биполярных транзисторов:

– сопротивление канала практически линейно, т. е. ток стока почти линейно зависит от напряжения сток–исток;

– имеется почти полная взаимозаменяемость стока и истока, так как ток в канале может протекать в обоих направлениях в зависимости от полярности напряжения, приложенного к каналу.

В цифровых микросхемах эти особенности не являются определяющими, но помнить о них не вредно.

3.3.1. Инверторы на *n*-МОП- и *p*-МОП-транзисторах

Схема инвертора на *n*-МОП-транзисторе и передаточная характеристика такого каскада представлена на рис. 3.33.

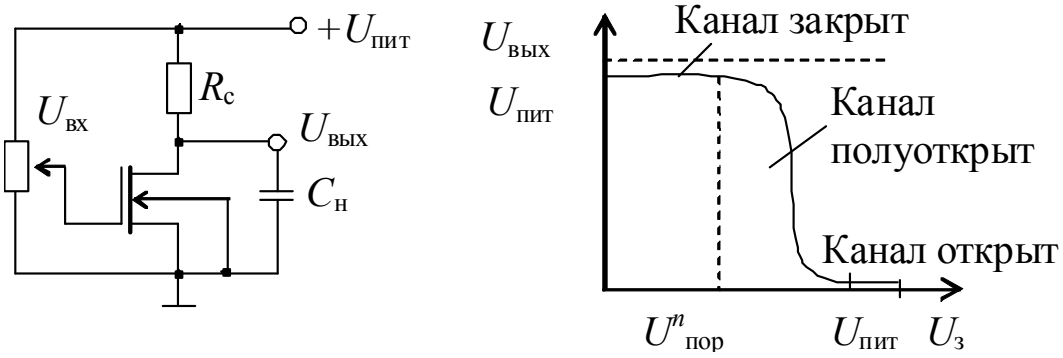


Рис. 3.33. Схема включения *n*-МОП-транзистора и передаточная характеристика такого каскада

Здесь $R_с$ – сопротивление стоковой нагрузки, аналогичное сопротивлению коллекторной нагрузки в *n-p-n*-транзисторах.

Оценим с помощью простых расчетов быстродействие такого каскада. Сопротивление канала $R_к$ меняется от десятков мОм в закрытом состоянии до 1…10 кОм в открытом при изменении входного напряжения $U_{зи}$ от 0 до $+U_{пит}$. Считаем, что $U^1_{вых} = U_{пит}$.

Если $R_с$ = 40 кОм, $R_{к.откр}$ = 4 кОм, $U_{пит}$ = 10 В, $C_{нагр}$ = 20 пФ, и поскольку $U^0_{вых} = U_{пит} \cdot R_{к.откр} / (R_с + R_{к.откр})$,

$$t_ф^{0,1} = R_с \cdot C_{нагр},\ t_ф^{0,1} = R_{к.откр} \cdot C_{нагр},\ f_{max} = \frac{1}{t_ф^{0,1} + t_ф^{0,1}},$$

то $U^0_{вых} = 10 \cdot 4/(40+4) = 0{,}9\,В$, $t_ф^{0,1} = 40\,кОм \cdot 20\,пФ = 800\,нс$,

$$t_{\text{ф}}^{0,1} = 4\ \text{кОм} \cdot 20\ \text{пФ} = 80\,\text{нс},\ f_{\max} = \frac{1}{t_{\text{ф}}^{0,1} + t_{\text{ф}}^{0,1}} = 1\,\text{МГц}.$$

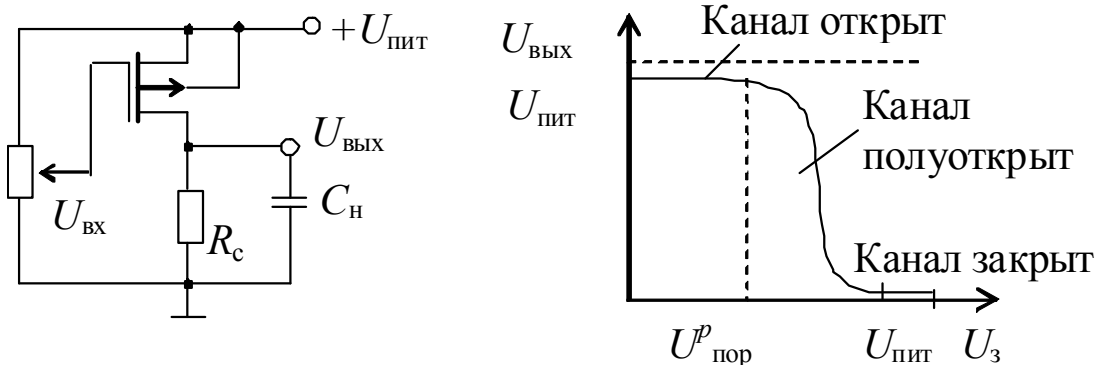


Рис. 3.34. Схема включения *p*-МОП-транзистора
и передаточная характеристика такого каскада

Так же как и в предыдущем случае, оценим с помощью простых расчетов быстродействие такого каскада.

Считаем, что $U_{\text{вх}} = U_{\text{пит}} - U_{\text{зи}}$.

Поскольку $U^{1}_{\text{вых}} = U_{\text{пит}} \cdot R_{\text{с}} / (R_{\text{с}} + R_{\text{к.откр}})$, $t_{\text{ф}}^{0,1} = R_{\text{к.откр}} \cdot C_{\text{нагр}}$,

$$t_{\text{ф}}^{0,1} = R_{\text{с}} \cdot C_{\text{нагр}},\ f_{\max} = \frac{1}{t_{\text{ф}}^{0,1} + t_{\text{ф}}^{0,1}},$$

то при $U_{\text{пит}}$ = 10 В:

$$R_{\text{с}} = 40\,\text{кОм},\ R_{\text{к.откр}} = 4\,\text{кОм},\ C_{\text{нагр}} = 20\,\text{пФ},$$

$$U^{1}_{\text{вых}} = 9{,}1\text{В},\ U^{0}_{\text{вых}} = 0\,\text{В},$$

$$t_{\text{ф}}^{0,1} = 80\text{нс},\ t_{\text{ф}}^{0,1} = 800\text{нс},\ f_{\max} = 1\,\text{МГц}.$$

Вместо резистора $R_{\text{с}}$ в действительности (в частности в серии К172) стоит специальный нагрузочный *p*-МОП транзистор с сопротивлением постоянно открытого канала $R_{\text{к.откр}}$ = 25…40 кОм.

Эти микросхемы имеют малое быстродействие ($t_{\text{зд.р}}$ = 1 мкс), большую мощность потребления (порядка 40 мВт/логический элемент) и уровни $U^{1}_{\text{вых}}$= 7,5 В, $U^{0}_{\text{вых}}$ = 2,3 В, не совместимые с уровнями ТТЛ микросхемами, поэтому в новых разработках серия К172 не применяется.

3.3.2. Инвертор на КМОП-транзисторах

Схема включения инвертора на взаимодополняющих, комплементарных *p*-МОП- и *n*-МОП-транзисторах, передаточная характеристика такого каскада и график импульса тока, протекающего через инвертор при его переключении, представлены на рис. 3.35.

Если учесть, что роль сопротивления стоковой нагрузки R_c при перезаряде $C_{нагр}$ играют поочередно *n*-МОП и *p*-МОП-транзисторы с сопротивлением открытого канала $R_{к.откр}$ на порядок меньшим, чем резисторы R_c в двух предыдущих случаях, то становится понятно, почему быстродействие КМОП логических элементов на порядок выше, чем у *n*-МОП и *p*-МОП логических элементов: $t_{зд.р}$ = 0,06 мкс для К561 при $C_н$ = 15 пФ, $U_{пит}$ = 5 В. В КМОП-инверторах $U^1_{вых} \approx U_{пит}$, а $U^0_{вых} \approx 0$.

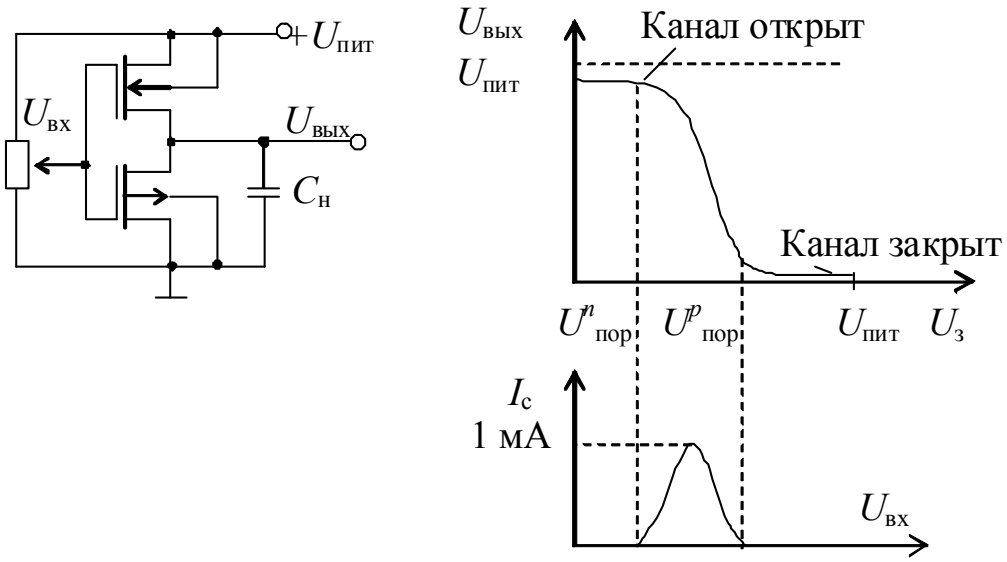


Рис. 3.35. Схема включения инвертора на комплементарных *p*-МОП- и *n*-МОП-транзисторах, передаточная характеристика такого каскада и график импульса тока, протекающего через инвертор при его переключении

Мощность, потребляемая логическим элементом КМОП в статическом режиме, близка к нулю ($P_{стат}$= 10^{-4}мВт / корпус), но с ростом частоты переключения схемы динамическая потребляемая мощность $P_{дин}$ растет в соответствии с формулой

$$P_{дин} = 2C_{нагр} f U^2_{пит},$$

где $C_{нагр}$ – эквивалентная емкость нагрузки микросхемы; f – частота переключения; $U_{пит}$ – напряжение питания.

Такая зависимость объясняется импульсами сквозного тока через оба транзистора в момент переключения.

Например, при частоте переключения f = 1 МГц, $U_{пит}$ = 10 В, $C_{нагр}$= 50 пФ потребляемая мощность достигает $P_{дин}$ = 10 мВт.

Очень высокое входное сопротивление МОП-транзисторов ($R_{вх} \approx 10^{12}$ Ом) ставит проблему защиты от пробоя изоляции входных цепей МОП-схем. Основная опасность идет от статического электричества, наводок силовых электромагнитных полей. Для защиты от пробоя на входе каждого КМОП логического элемента в составе микросхемы включают стабилитрон *VD*1. Его практически никогда не рисуют на схемах логических элементов, но помнить о том, что он есть в каждом КМОП логическом элементе, надо.

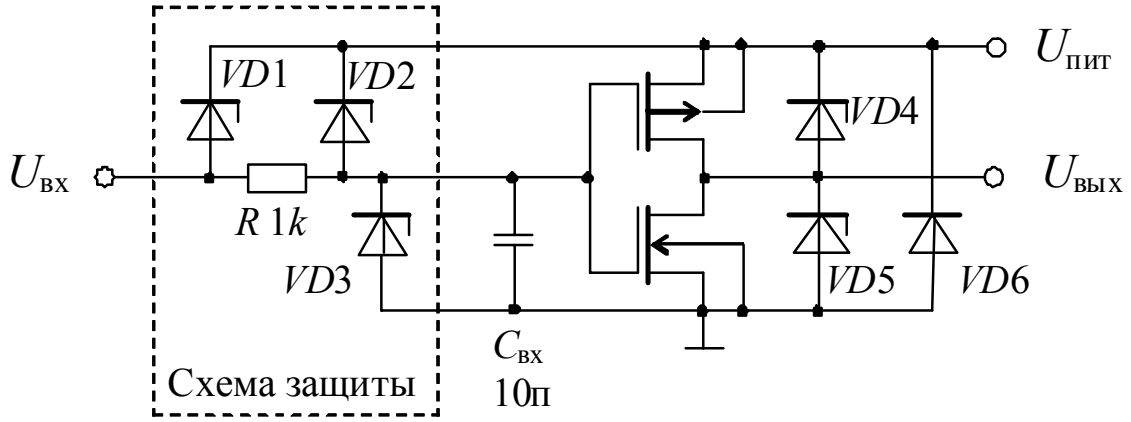


Рис. 3.36. Эквивалентная схема КМОП-инвертора

В структуре логического элемента КМОП заложены защиты как входов, так и выходов логического элемента. Они обеспечиваются технологией изготовления. На рис. 3.36 показана эквивалентная схема инвертора КМОП. Цепь *R*, *VD*1, *VD*2, *VD*3 представляет собой резисторно-диодную структуру с распределенными параметрами, введенную в кристалл логического элемента рядом с *p*-МОП и *n*-МОП транзисторами, и названа *схемой защиты*.

Эта цепь защищает вход инвертора от пробоя, резистор защищает выход предыдущего каскада от импульсной токовой перегрузки при заряде емкости $C_{вх}$. Диоды *VD*1, *VD*2 и *VD*3 изготовлены таким образом, что напряжение пробоя их в обратном включении составляет 25…50 В, и при сигналах на входе логического элемента, превышающих напряжение пробоя, эти диодные стабилитроны открываются и предохраняют входы полевых транзисторов от высоковольтных вы-

бросов, которые могут привести к пробою изоляции между затвором и каналом полевого транзистора.

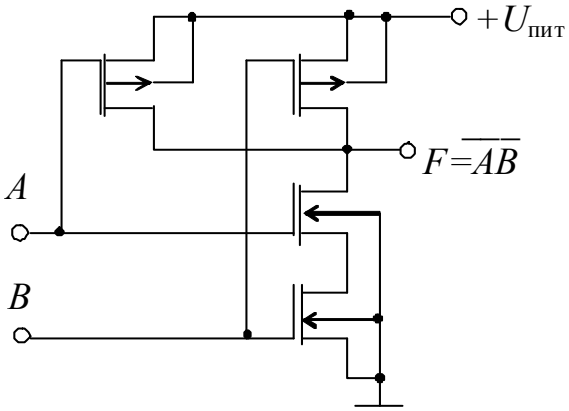
Различные *p–n* переходы, получившиеся при изготовлении областей стоков, истоков и кармана в подложке (кристалле кремния с *n*-проводимостью), на схеме показаны как *VD*4…*VD*6.

Предельно допустимый ток входных стабилитронов 10 мА, ток диода *D*6 больше. При неисправной полярности питания этот диод *VD*6 может сгореть сам или сжечь источник питания, если в источнике нет защиты от короткого замыкания по выходу.

Если потенциалы входов и выходов МОП-транзисторов не выходят за пределы 0…+$U_{пит}$, то об всех этих диодах можно забыть, поскольку напряжения пробоя этих встроенных стабилитронов превышает 25 В, т. е. больше напряжения питания.

3.3.3. КМОП логические элементы И–НЕ и ИЛИ–НЕ

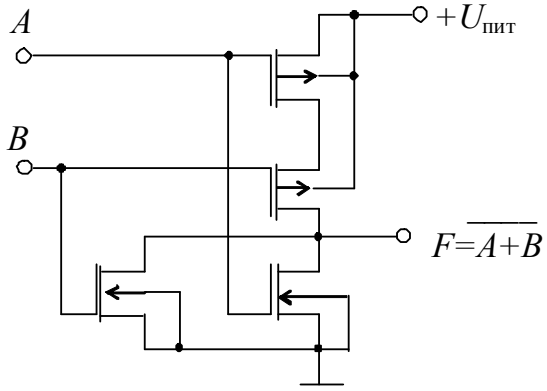
Схема КМОП логического элемента И–НЕ и его таблица истинности представлены на рис. 3.37.



A	***B***	***F***
0	0	1
0	1	1
1	0	1
1	1	0

Рис. 3.37. Схема КМОП логического элемента И–НЕ и его таблица истинности

Чтобы открыть оба нижних транзистора, надо, чтобы на входе *A* и входе *B* была логическая единица, при этом оба верхних транзистора закроются и на выходе *F* будет логический ноль. Если хотя бы на одном или на обоих входах будет логический ноль, то хотя бы один из нижних транзисторов закроется, один из верхних откроется, при этом на выходе будет логическая единица.



A	*B*	*F*
0	0	1
0	1	0
1	0	0
1	1	0

Рис. 3.38. Схема КМОП логического элемента ИЛИ–НЕ и его таблица истинности

Если на входах *A* или *B* (или на обоих) логическая единица, то откроется один или оба из нижних транзисторов, при этом закроется один или оба верхних транзистора, тогда на выходе логический ноль. Если на обоих входах логический ноль, то закроются оба нижних и откроются оба верхних транзистора, при этом на выходе логическая единица.

3.3.4. Двунаправленный ключ КМОП

Упрощенная схема КМОП двунаправленного ключа представлена на рис. 3.39.

Двунаправленным ключом называют также коммутатор цифровых и аналоговых сигналов, переключатель, ключ коммутации (КК). В состав ключа коммутации входят два инвертора, управляющие двумя параллельно включенными комплементарными транзисторами собственно ключа.

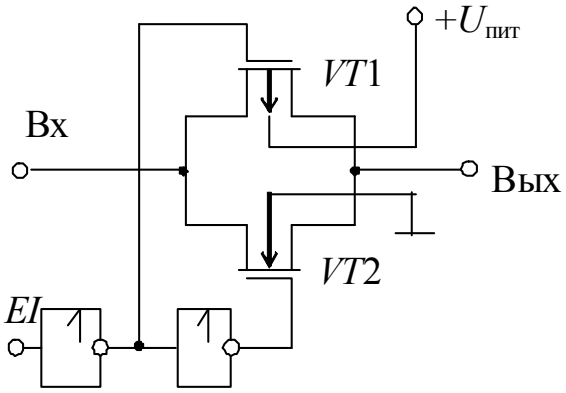


Рис. 3.39. Упрощенная схема КМОП двунаправленного ключа

Каждый из этих транзисторов имеет внутреннее сопротивление, существенно зависящее от коммутируемого напряжения, а параллельное их соединение имеет намного меньшую зависимость общего сопротивления от коммутируемого напряжения, как это показано на рис. 3.40.

Рассмотрим *два режима работы ключа коммутации*:

1. На управляющем входе *EI* = 1. На выходе первого управляющего инвертора 0, значит и верхний *p*-МОП-транзистор *VT*1 открыт, при этом на выходе второго инвертора 1, значит и нижний *n*-МОП-транзистор *VT*2 открыт. Следовательно, в ключе открыты оба транзистора.

2. На управляющем входе *EI* = 0. На выходе первого управляющего инвертора 1, значит и верхний *p*-МОП-транзистор *VT*1 закрыт, при этом на выходе второго инвертора 0, значит и нижний *n*-МОП-транзистор *VT*2 закрыт. Следовательно, в ключе оба транзистора закрыты.

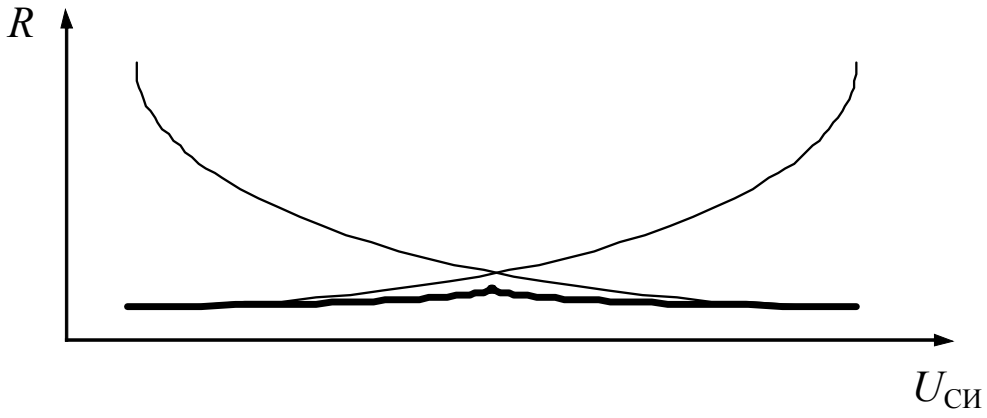


Рис. 3.40. Зависимости внутреннего сопротивления комплементарных транзисторов от напряжения сток–исток, показанные тонкими линиями, и заисимость их общего сопротивления, показанная жирной линией, от напряжения, коммутируемого ключом

Для ключа К176КТ1 характерны следующие параметры:

$$R_{\text{откр}} = 500 \text{ Ом};\ t_{\text{зд.р}} = 10\ldots25 \text{ нс}.$$

При $R_{\text{нагр}}$= 10 кОм, $f_{\text{сиг}}$ = 10 кГц развязку между входом и выходом можно определить из следующего условия:

$$20 \text{ дб} \cdot (U_{\text{вых.откр}} / U_{\text{вых.закр}}) \geq 65 \text{ дб}.$$

Сопротивление открытого ключа К561КТ3 меньше, чем ключа К176 КТ1, а именно $R_{\text{откр}}$ = 80 Ом.

3.3.5. Схемотехника КМОП-микросхем малой интеграции

В КМОП логике не один базовый элемент, как в ТТЛ (4И–НЕ), а два базовых элемента: *инвертор* НЕ и *ключ коммутации* КК. Напряжение между затвором и подложкой МОП транзистора открывает или закрывает канал транзистора, причем тот факт, открыт или закрыт транзистор, не зависит от напряжения на стоке и истоке (лишь бы не открылись диоды структуры полупроводникового кристалла). Это позволяет соединить каналы МОП-транзисторов последовательными цепочками, а также подавать на эти каналы любые напряжения (в пределах 0…+$U_{пит}$) с уверенностью, что все это не изменит состояние канала, не изменит его сопротивление.

Для примера построения КМОП-микросхем можно рассмотреть микросхему КМОП «исключающее ИЛИ» К176ЛП2. При анализе ее работы следует помнить, что из каждой пары транзисторов верхний *p*-МОП открывается логическим нулем, нижний *n*-МОП – логической единицей. В каждом из трех инверторов один из транзисторов открыт, второй закрыт, в ключе коммутации одновременно открываются или закрываются оба транзистора. При анализе работы схемы основное внимание надо уделить рассмотрению работу инвертора на *VT*5 и *VT*6, поскольку состояния транзисторов этого инвертора зависят не только от потенциалов на их затворах, но и от потенциалов на их истоках.

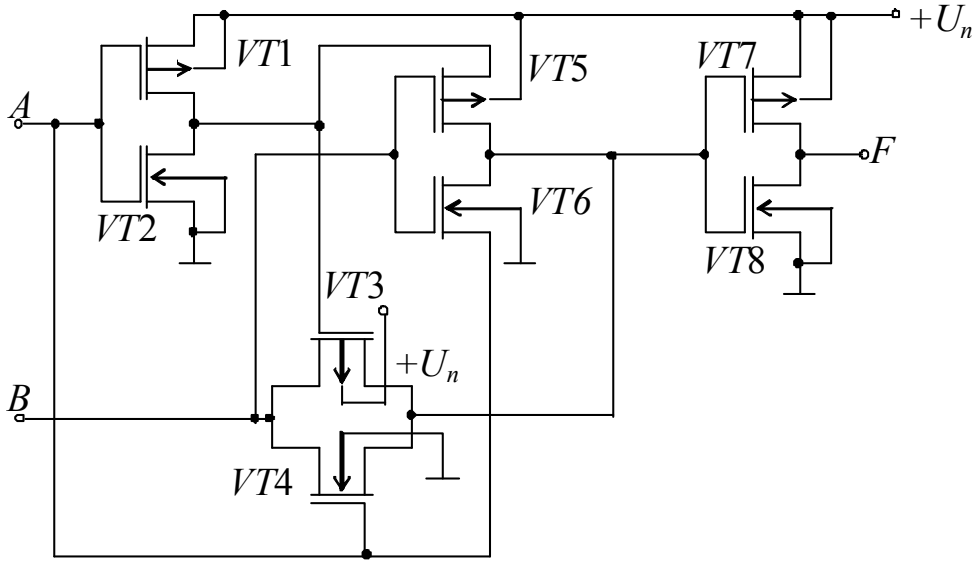


Рис. 3.41. Схема КМОП «исключающее ИЛИ» К176ЛП2

3.3.6. Разновидности выходов КМОП-микросхем и особенности их подключения

В КМОП, как и в ТТЛ, есть схемы с открытым стоком и *Z*-состоянием. Выходы логического элемента КМОП *не рекомендуется соединять непосредственно между собой*, как в ТТЛ. Особенно важно это для элементов с повышенным выходным током. Если требуется увеличить выходной ток, то допускается параллельное соединение входов и выходов логических элементов, но они должны быть из одного корпуса микросхемы. *Нельзя* применять емкости нагрузки $C > 500$ пФ для обычных логических элементов и $C > 5000$ пФ для буферных и высоковольтных оконечных элементов, поскольку такой конденсатор равноценен короткому замыканию для импульса тока; если же последовательно с выходом логического элемента включить гасящий резистор для ограничения импульса тока, то емкость нагрузки может быть любой.

При соединении выходов логического элемента с шиной питания или с корпусной шиной через один из выходных транзисторов протекает ток, ограничиваемый только сопротивлением открытого канала. Этот ток вызывает нагрев транзистора и всего логического элемента и может вызвать тепловой пробой, если рассеиваемая мощность превышает допустимую: 100 мВт для одного транзистора и 500 мВт для корпуса микросхемы. Благодаря отрицательному температурному коэффициенту тока канала МОП-транзисторов они обладают внутренней кратковременной защитой от нагрева. Ориентировочные значения токов короткого замыкания и рассеиваемой мощности даны в табл. 3.1, приведенной в [2].

Таблица 3.1

Токи короткого замыкания и рассеиваемая мощность МОП-транзисторов в зависимости от напряжения питания при 25 °С

	***p*-канальный**		***n*-канальный**	
$U_{пит}$, В	***I*, мА**	***P*, мВт**	***I*, мА**	***P*, мВт**
5	5,1	0,13	4,4	0,097
10	26,5	7,0	25	6,25
15	53	42,1	52,4	41,3

В некоторых КМОП-микросхемах выходные каскады имеют повышенную нагрузочную способность, такие микросхемы называют *буферными*, или *драйверами*. По сравнению с обычными микросхемами они имеют меньшее сопротивление открытого канала, как это можно видеть по табл. 3.2.

Таблица 3.2

Сопротивление открытого канала выходных транзисторов обычных и буферных микросхем в зависимости от напряжения питания

Напряжение питания, *В*	**5**	**15**
Сопротивление открытого канала, кОм:		
– при буферном выходном каскаде	1,5	0,5
– при обычном выходном каскаде	2,5	0,75

3.3.7. Подключение входов КМОП-микросхем

Входы КМОП-микросхем никогда не должны оставаться ни к чему не подключенными – это одно из обязательных правил.

Входное сопротивление КМОП-микросхем почти на десять порядков больше, чем ТТЛ-микросхем, и, соответственно, примерно во столько же раз сильнее влияют помехи на «висящий», ни к чему не подключенный вход КМОП-микросхемы, чем на такой же вход ТТЛ-микросхемы.

Из практики работы с микросхемами ТТЛ-серий известно, что на свободном, ни к чему не подключенном входе ТТЛ-микросхемы самопроизвольно устанавливается так называемая «висячая единица», т. е. напряжение, примерно равное полтора вольта. Это можно показать, анализируя входную статическую характеристику логического элемента ТТЛ, представленную на рис. 3.28, на которой при входном токе, равном нулю, напряжение примерно равно полутора вольтам.

Практика работы с КМОП-микросхемами показывает, что часто при обрыве дорожки платы, идущей к какому–либо входу КМОП-микросхемы, на выходе этой микросхемы появляется переменное напряжение с частотой питающей сети.

Иногда при обрыве соединения на входе КМОП-микросхемы напряжение на этом входе медленно дрейфует, плавает. При этом возможны так называемые «мерцающие» отказы в работе устройства, когда устройство нормально работает некоторое время, затем без какой-либо причины оно выходит из строя, а затем, через какое-то время устройство вновь начинает нормально работать.

Из-за тиристорного эффекта, присущего КМОП-микросхемам (особенно их первым сериям), устройства с этими микросхемами не разрешается вставлять в разъемы, на которых уже могут присутствовать напряжения питания и входные сигналы. В таких случаях возможны ситуации, когда напряжение на вход КМОП-микросхемы поступает раньше, чем напряжение питания. При этом полупроводнико-

вые структуры КМОП-инвертора могут открыться аналогично тиристору, включенному анодом к плюсу питания, а катодом – к земле.

Такой тиристор перегружает источник питания, а главное, перегревается сам, и вполне может через некоторое время (через секунду-другую) сгореть от теплового пробоя. По этой же причине устройства с КМОП-микросхемами не разрешается вытаскивать из разъемов, на которых присутствуют напряжения питания и входные сигналы.

Без потери работоспособности неиспользуемые И входы КМОП-микросхем могут быть подключены непосредственно к плюсу питания, а ИЛИ входы – к корпусной шине; при этом пороги переключения соседних входов немного смещаются. Можно неиспользованные входы КМОП-микросхем подключать параллельно соседним используемым, но при этом эти входы дополнительно нагружают выход предыдущей микросхемы пропорционально числу подключенных входов.

Очень большое входное сопротивление КМОП-микросхем позволяет при расчете разветвления сигналов с одного выхода на несколько входов пренебрегать активной, резистивной составляющей их входного сопротивления. Число входов, которые допустимо подключать к одному выходу КМОП-микросхемы, определяется эквивалентной входной емкостью одного входа и предельной емкостью нагрузки, при которой характеристики микросхем (в основном динамические) не выходят из заданных пределов.

Входная емкость большинства КМОП-микросхем составляет 5...15 пф, а максимальная емкость нагрузки, при которой характеристики микросхем не выходят из заданных пределов, обычно равна 500 пф, поэтому обычно коэффициент разветвления $K_{\text{разв}}$= 30...100.

Статическая помехоустойчивость КМОП-микросхем зависит от напряжения питания и увеличивается с его ростом. Допустимое напряжение помех можно выразить как долю напряжения питания таким образом:

$$U^0_{\text{пом}} \approx U^1_{\text{пом}} \approx U_{\text{пит}}/3.$$

Особенностью КМОП-микросхем является очень большой разброс и нестабильность напряжения переключения – область входных напряжений, в которой может находиться порог переключения КМОП-микросхем, составляет примерно треть напряжения питания (тогда как для ТТЛ-микросхем эта область на один-два порядка меньше).

3.4. Эмиттерно-связанная логика

Микросхемы *эмиттерно-связанной логики* (ЭСЛ) являются самыми быстродействующими из всех типов логик, и обеспечивается это за счет целого ряда особенностей этой логики.

Главная особенность ЭСЛ, повышающая ее быстродействие, заключается в том, что схема ее логического элемента основана на дифференциальном усилителе (балансном каскаде), дифференциальном переключателе тока, показанном на рис. 3.42, два транзистора которого переключают ток и не попадают в режим насыщения. Благодаря этому значительно сокращается время выхода транзисторов логического элемента из открытого состояния и существенно повышается общее быстродействие.

В эмиттерную цепь этих транзисторов включен генератор стабильного тока (ГСТ), который ограничивает величину тока, протекающего через тот из двух транзисторов, который открыт.

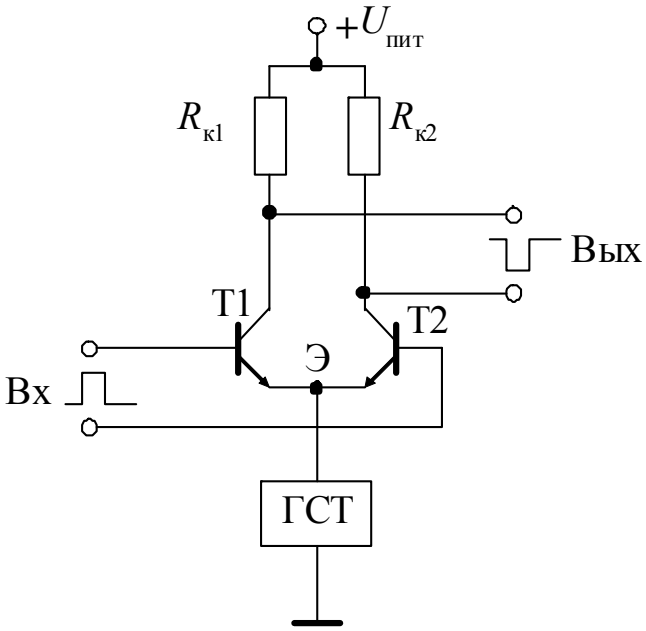


Рис. 3.42. Основа ЭСЛ логического элемента – дифференциальный усилитель, в эмиттерную цепь которого включен генератор стабильного тока

Величина тока, задаваемая ГСТ, и сопротивления резисторов коллекторных нагрузок каскадов выбраны такими, чтобы исключить режим насыщения транзисторов в открытом состоянии независимо от разброса усиления этих транзисторов, что невозможно обеспечить в КМОП и в обычных ТТЛ-сериях (кроме серий с транзисторами Шоттки).

В ЭСЛ-микросхемах имеется два противофазных выхода – прямой и инверсный, поэтому в устройствах на ЭСЛ-микросхемах отсутствуют промежуточные инверторы, которые в устройствах на ТТЛ и КМОП-микросхемах вносят дополнительную задержку и снижают быстродействие.

С целью уменьшения времени перезаряда паразитных емкостей за счет уменьшения выходного сопротивления ЭСЛ логических элементов в их схемы введены мощные эмиттерные повторители с сопротивлениями нагрузки малой величины 50 Ом.

Уменьшение задержки распространения в ЭСЛ-сериях достигается также и за счет уменьшения длительности фронтов выходных импульсов за счет уменьшения перепада напряжения на фронтах импульсов:

– в ЭСЛ-сериях:

$$U^1_{\text{вых}} = –0{,}96 \text{ В}, \; U^0_{\text{вых}} = –1{,}65 \text{ В},$$

$$\Delta U_{\text{вых}} = | \, U^1_{\text{вых}} – U^0_{\text{вых}} \, | = 0{,}69 \text{ В};$$

– в ТТЛ-сериях:

$$U^1_{\text{вых}} = 3{,}6 \text{ В}, \; U^0_{\text{вых}} = 0{,}1 \text{ В},$$

$$\Delta U_{\text{вых}} = | \, U^1_{\text{вых}} – U^0_{\text{вых}} \, | = 3{,}5 \text{ В};$$

– в КМОП-сериях при питании +9 В:

$$U^1_{\text{вых}} = 9 \text{ В}, \quad U^0_{\text{вых}} = 0 \text{ В},$$

$$\Delta U_{\text{вых}} = | \, U^1_{\text{вых}} – U^0_{\text{вых}} \, | = 9 \text{ В}.$$

Даже если бы крутизна фронтов выходных импульсов у микросхем ЭСЛ, ТТЛ и КМОП была бы одинаковой, то только за счет уменьшения перепада напряжения на фронтах импульсов длительности фронтов выходных импульсов в ЭСЛ были бы в пять раз меньше, чем в ТТЛ, и в тринадцать раз меньше, чем в КМОП.

Но уменьшение перепада напряжения на фронтах импульсов означает меньшую разницу между уровнями нуля и единицы, а значит и помехи меньшей величины могут привести к неправильному срабатыванию ЭСЛ-микросхем. Для *снижения влияния помех* в ЭСЛ-микросхемах применяются *следующие приемы*:

1. С корпусной шиной в ЭСЛ-микросхемах соединен положительный вывод питания, а не отрицательный, как во других логиках. При таком подключении чувствительность логического элемента к помехам по питанию (отрицательному) ослабляется благодаря наличию генератора стабильного тока, и чем выше степень его стабилизации, тем сильнее ослабляется эмиттерный синфазный сигнал помехи.

2. Порог переключения ЭСЛ логического элемента стабилизируется с помощью источника опорного напряжения (ИОН), который фиксирует напряжение на базе одного из транзисторов дифференциального переключателя тока.

3. Корпусная шина питания (положительного) маломощных чувствительных каскадов микросхем развязана от корпусной шина питания мощных выходных повторителей, являющихся основными источниками помех.

4. Сопротивления нагрузки выходных эмиттерных повторителей подключаются не к шине отрицательного питания, а к отдельному источнику смещения. Напряжение источника смещения меньше, чем на шине отрицательного питания, поэтому мощность, рассеиваемая этими сопротивлениями нагрузки, практически на порядок снижена.

Вышеуказанные средства обеспечивают приемлемую помехоустойчивость ЭСЛ-микросхем. Схема трехвходового логического элемента приведена на рис. 3.43.

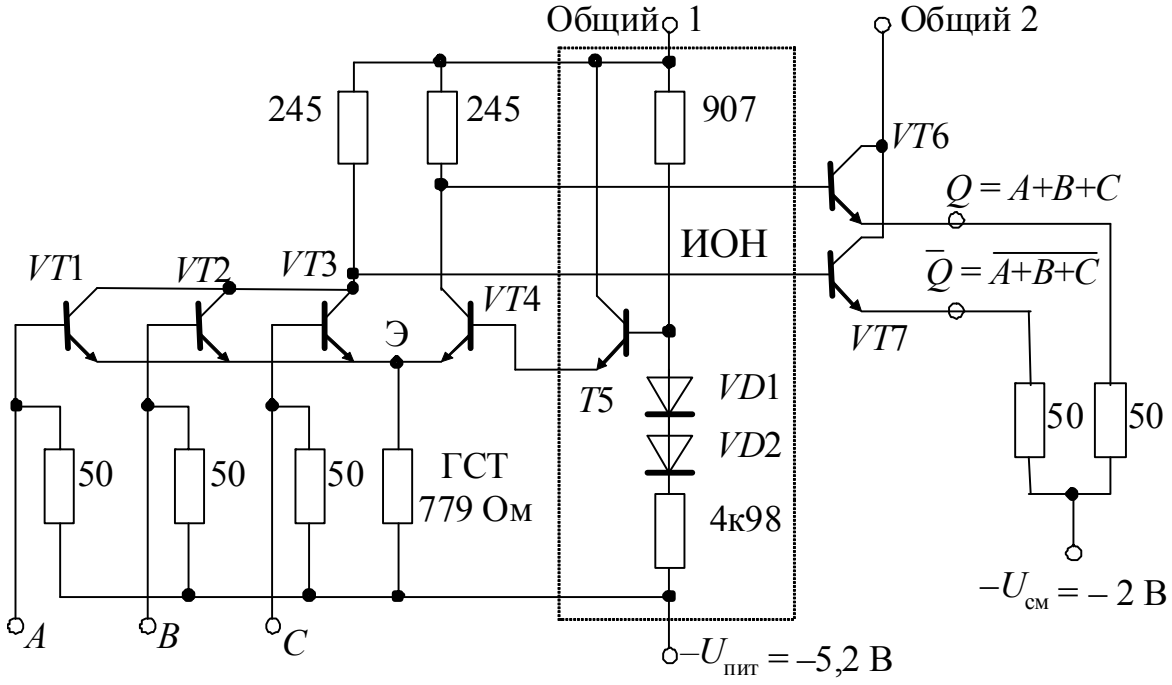


Рис. 3.43. Схема трехвходового логического элемента ЭСЛ серии К500

В первых ЭСЛ-сериях (отечественная серия К500) в качестве генератора стабильного тока (ГСТ) служил резистор, сопротивление которого больше, чем сопротивления коллекторных нагрузок.

В процессе развития ЭСЛ-серий (отечественная серия К1500) были применены следующие улучшения:

– стали использоваться более быстродействующие транзисторы с граничной частотой $f_T \approx 4{,}5$ ГГц, что обеспечивает $t_{зд} \approx 0{,}75$ нс (вместо $f_T \approx 1{,}5$ ГГц, которая обеспечивала $t_{зд} \approx 2$ нс в серии К500);

– стала применяться более плотная «упаковка», т. е. вместо десяти логических элементов на миллиметр поверхности кристалла микросхемы в серии К500 в следующей серии К1500 двадцать логических элементов на миллиметр поверхности кристалла, и это при том, что число транзисторов в одном логическом элементе увеличилось вдвое;

– стали использоваться более стабильные и сложные генераторы стабильного тока и источники опорного напряжения;

– напряжение питания уменьшили с 5,2 В до 4 В, но при этом обеспечивается совместимость между этими сериями по уровням логического нуля и единицы.

В ЭСЛ-сериях неиспользованные входы можно оставлять ни к чему не подключенными, так как они внутри микросхем соединены с минусом питания через резисторы 50 кОм.

Выходы ЭСЛ-микросхем можно соединять друг с другом с учетом их полярности: прямые выходы можно соединять в монтажное ИЛИ, а инверсные выходы – в монтажное И.

3.5. Программируемые логические устройства

3.5.1. Структура программируемых логических устройств

Программируемые логические матрицы (ПЛМ) появились в середине 70-х гг. [8]. Основой их служит последовательность программируемых матриц элементов И и ИЛИ. В структуру входят также блоки входных и выходных буферных каскадов (БВх и БВых на рис. 3.44).

Входные буферы, если не выполняют более сложных действий, преобразуют однофазные входные сигналы в парафазные и формируют сигналы необходимой мощности для питания матрицы элементов И.

Выходные буферы обеспечивают необходимую нагрузочную способность выходов, разрешают или запрещают выход ПЛМ на внешние шины с помощью сигнала *OE*, а иногда выполняют и более сложные

действия. Основными параметрами ПЛМ (рис. 3.44) являются число входов *m*, число термов *l* и число выходов *n*.

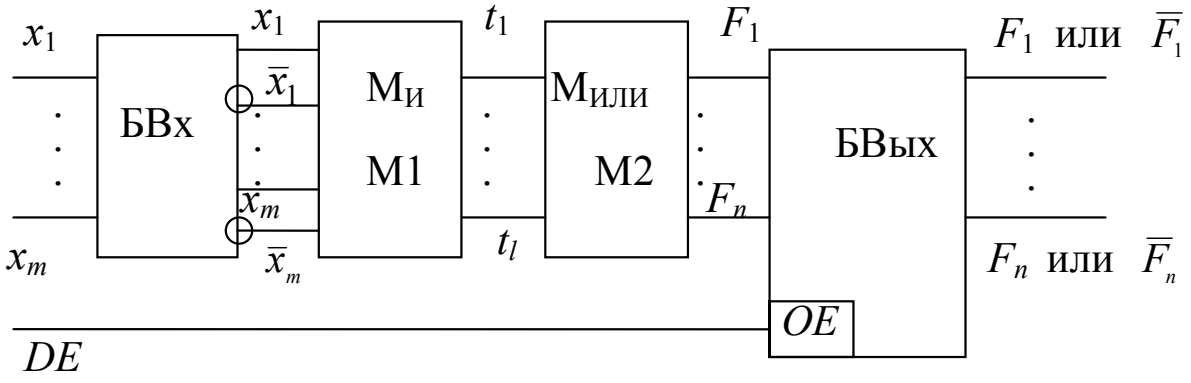


Рис. 3.44. Базовая структура ПЛМ

Переменные $x_1 \ldots x_m$ подаются через блоки входных буферных каскадов (БВх) на входы элементов И (конъюнкторов) и в матрице И образуются *l* термов. Под *термом* здесь понимается конъюнкция, связывающая входные переменные, представленные в прямой или инверсной форме. Число формируемых термов равно числу конъюнкторов или, что то же самое, числу выходов матрицы И. Термы подаются далее на входы матрицы ИЛИ, т. е. на входы дизъюнкторов, формирующих выходные функции. Число дизъюнкторов равно числу вырабатываемых функций *n*.

Таким образом, ПЛМ реализует дизъюнктивную нормальную форму (ДНФ) воспроизводимых функций. ПЛМ *способна реализовать систему n логических функций от m аргументов*, *содержащую не более l термов.* Воспроизводимые функции являются комбинациями из любого числа термов, формируемых матрицей И. Какие именно термы будут выработаны и какие комбинации этих термов составят выходные функции, определяется программированием ПЛМ.

Первые отечественные ПЛМ были выпушены в составе серии К556 (микросхемы *РТ*1, *РТ*2 схемотехнологии ТТЛШ с программированием-прожиганием перемычек). Их размерность 16 входов, 48 термов, 8 выходов, задержка около 50 нс. Микросхема РТ1 имеет выходы с открытым коллектором. Микросхема РТ2 имеет выходы с тремя состояниями.

На рис. 3.45 показано обозначение ПЛМ на схемах принципиальных электрических.

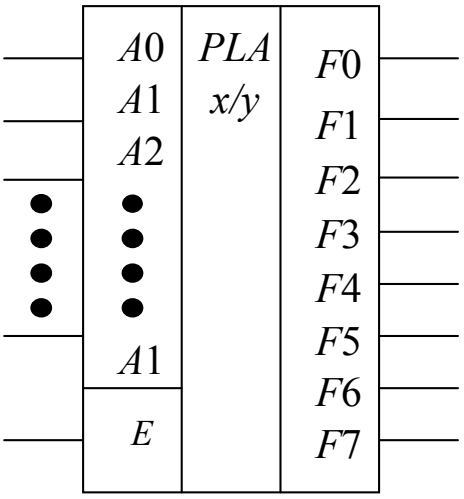


Рис. 3.45. Обозначение ПЛМ на принципиальных электрических схемах

Основу ПЛМ составляют матрицы элементов «И» и элементов «ИЛИ». На рис. 3.46 показана структурная схема ПЛМ К556РТ1.

Требуемые логические произведения формируются на шинах столбцов путем выжигания ненужных перемычек (обозначены зигзагом) между строками и столбцами. Требуемые логические суммы формируются матрицей элементов «ИЛИ», в которой на входах ЛЭ также выжигаются ненужные перемычки. Число термов (столбцов), как упоминалось выше, у ПЛМ К556РТ1, К556РТ2 всего 48. Следовательно, может быть получено до 48-и логических произведений. Число переменных и дополнений, которые могут участвовать в произведениях, определяется количеством адресных входов ПЛМ, которое равно 16. Следовательно, каждый элемент «И» имеет 32 входа (16 переменных и 16 дополнений), а каждый элемент «ИЛИ» – 48 входов (по числу столбцов ПЛМ). Всего таких элементов «И» – 8. Следовательно, ПЛМ формирует до 8 функций из логических сумм, в которых участвуют до 16 входных переменных и их дополнений.

Элементы «исключающее ИЛИ» позволяют получить прямой или инверсный выход в зависимости от наличия перемычек на втором входе элемента. Технологией изготовления ПЛМ обеспечивается на неподключенных входах элементов «И» – «1», а входах элементов «ИЛИ» – «0».

Элементами связи в матрице ИЛИ служат транзисторы, включенные по схеме эмитгерного повторителя относительно линий термов и образующие схему ИЛИ относительно выхода.

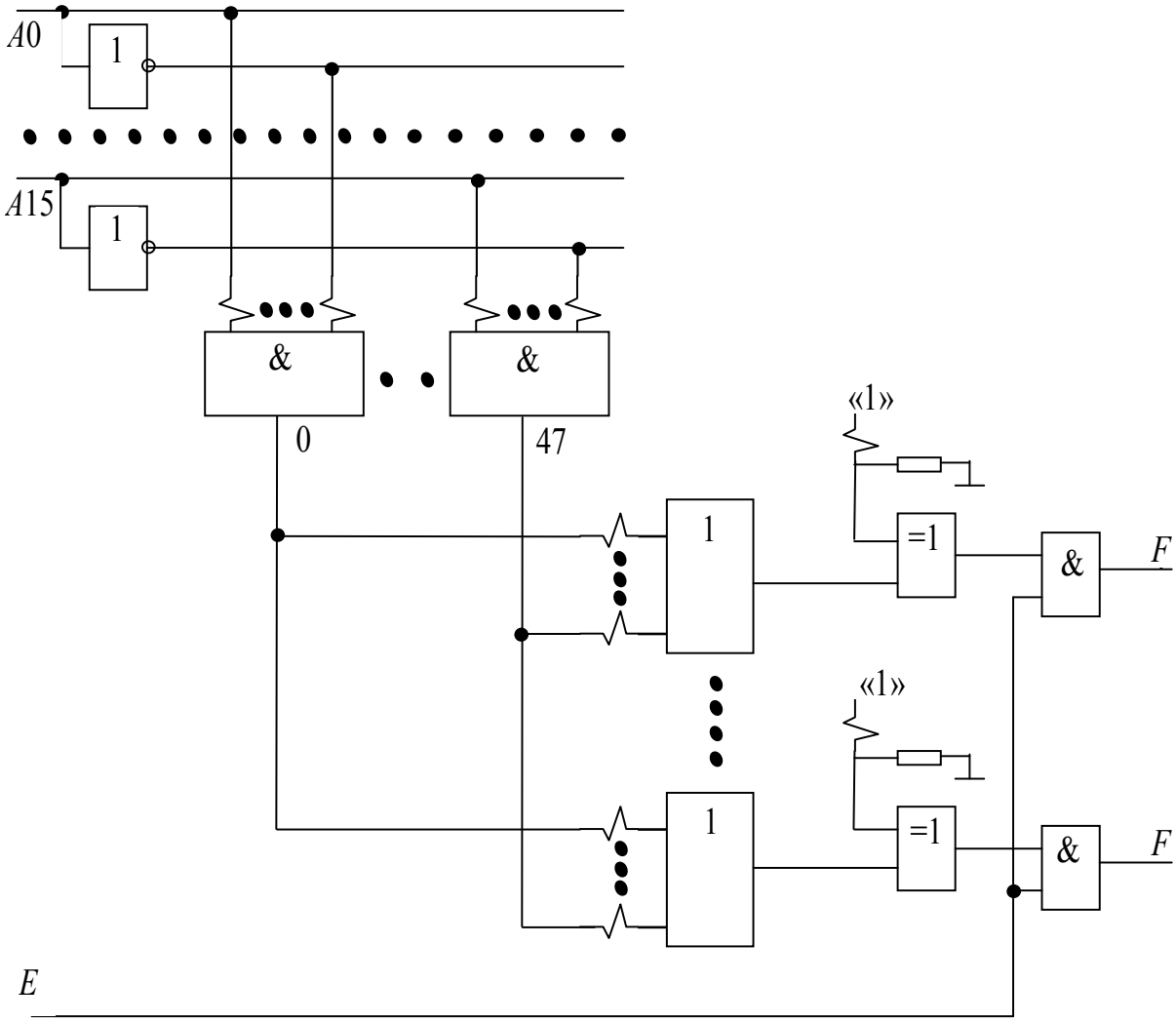


Рис. 3.46. Схема ПЛМ К556РТ1

Элементами связей в матрице И служат диоды, соединяющие горизонтальные и вертикальные шины. Совместно с резистором и источником питания цепи выработки термов образуют обычные диодные схемы И. До программирования все перемычки целы и диоды связи размещены во всех узлах координатной сетки. При любой комбинации аргументов на выходе будет ноль, так как на вход схемы подаются одновременно прямые и инверсные значения аргументов, а $x\bar{x} = 0$. При программировании в схеме оставляются только необходимые элементы связи, а ненужные устраняются пережиганием перемычек. Высокий уровень выходного напряжения (логическая единица) появляется только при наличии высоких напряжений на всех входах, низкое напряжение хотя бы на одном входе фиксирует выходное напряжение на низком уровне, так как открывается диод этого входа. Так выполняется операция И.

3.5.2. Программирование программируемых логических устройств

Программирование ПЛМ, выполняемое пользователем, проводится с помощью специальных устройств (программаторов), и сведения для них о данной ПЛМ должны иметь определенную форму. Имеются программаторы, которые принимают в качестве информации о ПЛМ таблицу функционирования (истинности), однако удобнее задавать сведения о самих перемычках. Символы, используемые при таком задании сведений для программирования ПЛМ:

H – переменная входит в терм в прямом виде, т. е. нужно оставить целой перемычку прямого входа и пережечь перемычку инверсного входа;

L – переменная входит в терм в инверсном виде, т. е. нужно сохранить перемычку у инверсного входа и пережечь у прямого;

«–» – переменная не входит в терм и не должна влиять на него, т. е. нужно пережечь перемычки обоих входов.

Оставление перемычек у обоих входов переменной как бы устраняет из матрицы соответствующую схему И, поскольку в силу равенства $x\bar{x} = 0$ выход этой схемы всегда нулевой и не влияет на работу матрицы ИЛИ, на вход которой подается;

A – указывается в выходном столбце (столбце функции) и свидетельствует о связи данной схемы И с выходом ПЛМ через матрицу ИЛИ. Перемычка должна быть сохранена;

« • » – указывает на то, что данная схема И не подключается к выходу и должна иметь пережженную перемычку в матрице ИЛИ.

3.5.3. Базовые матричные кристаллы (вентильные матрицы с масочным программированием)

Первые образцы базовых матричных кристаллов (БМК) появились в 1975 г. как средство реализации нестандартных схем высокопроизводительной ЭВМ без применения микросхем малого и среднего уровней интеграции. Разработка БМК, кроме того, позволила выполнить и нетиповые части машины на БИС.

Стоимость проектирования БИС/СБИС велика и достигает десятков или даже сотен миллионов долларов. Ясно, что производство БИС/СБИС становится рентабельным только при достаточно большом объеме их потребления, чего нет при разработке нестандартных

частей конкретных систем. Выход из создавшихся трудностей был найден на путях разработки БИС/СБИС, функционирование которых может быть приспособлено к решению той или иной задачи на заключительных этапах их производства. При этом полуфабрикаты производятся в массовом количестве без ориентации на конкретного заказчика. Придание полуфабрикатам индивидуального характера лишь на заключительных стадиях производства БИС/СБИС обходится значительно дешевле и требует значительно меньшего времени на проектирование. Такие БИС/СБИС называют *полузаказными* в отличие от полностью заказных.

Развитие полузаказных БИС/СБИС привело к появлению ряда их разновидностей. Применительно к БМК, это – *канальные*, *бесканальные* и *блочные архитектуры* [8].

Основа БМК первого поколения – совокупность регулярно расположенных на кристалле базовых ячеек (БЯ), между которыми имеются свободные зоны для создания соединений (каналы). Эта архитектура называется *канальной*. Базовые ячейки занимают внутреннюю область БМК, в которой они расположены по строкам и столбцам и содержат группы нескоммутированных элементов (транзисторов, резисторов и др.). В периферийной области кристалла размещены ячейки ввода-вывода, набор схемных компонентов которых ориентирован на реализацию связей БМК с внешними цепями.

Таким образом, БМК является заготовкой, которая преобразуется в требуемую схему выполнением необходимых соединений. Потребитель может реализовать на основе БМК некоторое множество устройств определенного класса, задав для кристалла тот или иной вариант рисунка межсоединений компонентов.

Первые БМК (фирмы Amdahl Corp., США) выполнялись по схемотехнике ЭСЛ, для которой полный процесс изготовления включал 13 операций с фотошаблонами. Для изготовления схемы на основе БМК (такие схемы называют МАБИС или БИСМ) требуются только 3 индивидуальных (переменных) шаблона для задания рисунка межсоединений. Соответственно этому сроки и стоимость проектирования МАБИС в 3...5 раз меньше, чем для полностью заказных БИС/СБИС.

Плата за сокращение сроков и стоимости проектирования – неоптимальность результата. МАБИС проигрывают по площади кри-

сталла и быстродействию полностью заказным схемам, так как часть их элементов оказывается избыточной (не используется в данной схеме), взаимное расположение элементов и пути межсоединений не являются наилучшими и т. д. Промышленное производство БМК широко развернулось с начала 80-х гг. Применяются схемотехнологии КМОП, ТТЛШ, ЭСЛ и др. В настоящее время уровень интеграции БМК достиг миллионов вентилей на кристалле. При проектировании БМК стремятся наилучшим образом сбалансировать число базовых ячеек, трассировочные ресурсы кристалла и число контактных площадок для подключения внешних выводов. Неудачные соотношения между указанными параметрами могут существенно ограничивать полноту использования ресурсов кристалла при построении МАБИС.

Трассировочная способность БМК определяется, прежде всего, площадью, отводимой для межэлементных связей в ортогональных направлениях. Учитывается и число слоев межсоединений. Недостаточная трассировочная способность приводит к уменьшению числа задействованных при построении МАБИС базовых ячеек. Избыточная трассировочная способность ведет к нерациональному использованию площади кристалла, что понижает уровень интеграции БМК и повышает его стоимость.

Примерно тоже можно сказать и о числе внешних выводов БМК. Для современных БМК может потребоваться до 500...600 внешних выводов. При проектировании БМК требуемые трассировочная способность и число внешних выводов рассчитываются по эмпирическим формулам, основанным на статистических данных, полученных из опыта построения систем различного назначения. Эта работа выполняется до изготовления БМК и в этом смысле не входит в компетенцию системотехника. Системотехник (потребитель) должен иметь представление о существующих БМК, их разновидностях и особенностях, а также о средствах и методике разработки МАБИС.

Остановимся подробнее на основных понятиях и определениях.

Базовая ячейка (БЯ) уже определялась как некоторый набор схемных элементов, регулярно повторяющийся на определенной площади кристалла. Этот набор может состоять из нескоммутированных элементов, а также из частично скоммутированных. Базовые ячейки внутренней области БМК именуются матричными базовыми ячейками (МБЯ),

ячейки периферийной зоны – периферийными базовыми ячейками (ПБЯ). Применяются два способа организации ячеек БМК:

– из элементов МБЯ может быть сформирован один логический элемент, а для реализации более сложных функций используются несколько ячеек;

– из элементов МБЯ может быть сформирован любой функциональный узел, а состав элементов ячейки определяется схемой самого сложного узла.

Функциональная ячейка (ФЯ) – функционально законченная схема, реализуемая путем соединения элементов в пределах одной или нескольких БЯ. *Библиотека функциональных ячеек* – совокупность ФЯ, используемых при проектировании МАБИС. Эта библиотека создается при разработке БМК и избавляет проектировщика МАБИС от работы по созданию на кристалле тех или иных типовых подсхем, так как предоставляет для их реализации готовые решения. Библиотека содержит большое число (сотни) функциональных элементов, узлов и их частей. Пользуясь библиотекой, проектировщик реализует схемы, работоспособность которых уже проверена, а параметры известны. Работая с библиотекой, он ведет проектирование на функционально-логическом уровне, поскольку проблемы схемотехнического уровня уже решены при создании библиотеки. Библиотечные элементы имеют различную сложность (логические элементы, триггеры, более сложные узлы или их фрагменты). В состав библиотечного элемента могут входить одна или несколько БЯ. Площадь библиотечного элемента кратна площади БЯ. При проектировании МАБИС функциональная схема изготовляемого устройства, как принято говорить, должна быть покрыта элементами библиотеки.

Эквивалентный вентиль (ЭВ) – группа элементов БМК, соответствующая возможности реализации логической функции вентиля (обычно это двухвходовый элемент И–НЕ либо ИЛИ–НЕ). Понятие «эквивалентный вентиль» предназначено для оценки логической сложности БМК [8].

Каналы трассировки – пути на БМК для возможного размещения межсоединений.

3.5.4. Программируемые пользователем вентильные матрицы

Программируемые пользователем вентильные матрицы (ППВМ или *FPGA*) топологически сходны с канальными БМК. В их внутренней области размещается множество регулярно расположенных идентичных конфигурируемых логических блоков (КБЛ), между которыми проходят трассировочные каналы, а на периферии кристалла расположены блоки ввода/вывода.

К наиболее известным *FPGA* относятся БИС/СБИС семейств *XC*2000, *XC*3000, *XC*4000, *XC*5000 и Spartan фирмы Xilinx, которая в 1985 г. впервые выпустила ППВМ с триггерной памятью конфигурации. Среди ППВМ с перемычками типа anlifuse следует отметить семейства *ACTI*, *I*200*XL*, *ACT*3, 3200*DX* фирмы Actel, используемые, в частности, в космической аппаратуре США.

Свойства и возможности ППВМ зависят в первую очередь от характера их конфигурируемых логических блоков и системы межсоединений.

В качестве конфигурируемых логических блоков (далее для краткости просто ЛБ – логические блоки) используются:

– транзисторные пары, простые логические вентили И–НЕ, ИЛИ–НЕ и т. п. Такие ЛБ называют *SLC* – Simple Logic Cells;

– логические модули на основе мультиплексоров;

– логические модули на основе программируемых ПЗУ, такие ЛБ называют *LUTs* – Look-Up Tables.

Важной характеристикой логических блоков является их «зернистость» (Granularity). Другой важной характеристикой считается «функциональность» (Functionality).

Первое свойство связано с тем, насколько «мелкими» будут те части, из которых можно «собирать» нужные схемы, второе – с тем, насколько велики логические возможности логических блоков.

Примером наиболее мелкозернистого может служить логический блок фирмы Crosspoini Solutions. Блок содержит цепочки транзисторов с *p*- и *n*-каналами. ЛБ – пара из транзисторов разного типа проводимости. Между цепочками транзисторов имеются трассировочные каналы, в которых могут быть реализованы необходимые межсоединения элементов.

Мелкозернистость ЛБ ведет к большей гибкости их использования, возможностям реализовать воспроизводимые функции разными

способами, получая разные варианты в координатах «площадь кристалла – быстродействие». В то же время мелкозернистость ЛБ усложняет систему межсоединений ППВМ в связи с большим числом программируемых точек связи.

Крупнозернистый блок семейства *XC*4000*E* в качестве основы имеет три табличных функциональных логических преобразователя, а также ряд программируемых мультиплексоров и два триггера.

В ППВМ с триггерной памятью реконфигурации, как правило, применяют крупнозернистые блоки. В таких блоках реализуются более сложные функции, что ведет к упрощению программируемой части межсоединений. В то же время труднее полностью использовать логические элементы блоков, что ведет к потерям площади кристалла и быстродействия. Иными словами, меняя зернистость, можно выиграть в одном и проиграть в другом.

Примерами более крупнозернистых ЛБ могут служить используемые в семействе микросхем *ACT* фирмы Actel.

Системы межсоединений (системы коммутации), как и логические блоки, реализуются в широком диапазоне архитектурных и технологических решений. Линии связей в *FPGA* обычно сегментированы, т. е. составлены из проводящих сегментов (участков, не содержащих ключей) различной длины, соединяемых друг с другом программируемым элементом связи (ключом). Малое количество сегментов ведет к недостаточно эффективному использованию логических блоков, слишком большое – к появлению большого числа программируемых ключей в линиях связи, что увеличивает затраты площади кристалла и вносит дополнительные задержки сигналов.

Короткие сегменты затрудняют реализацию длинных связей, длинные – коротких. Поэтому целесообразна иерархическая система связей с несколькими типами межсоединений для передач на разные расстояния. Целью построения системы связей является обеспечение максимальной коммутируемости блоков при минимальном количестве ключей и задержек сигналов, а также предсказуемость последних, облегчающая проектирование.

Наличие ключей и схем для их программирования усложняет межсоединения ППВМ сравнительно с межсоединениями БМК.

Критерий трассировочной способности системы межсоединений отображает возможность создания в ППВМ множества схем типового

применения (только с помощью программируемых ключей, так как сегментная часть соединений стандартная). Быстродействие ППВМ существенно зависит от задержек сигналов в связях. Ключ в лиии связи имеет схему замещения в виде *RC*-звена. В последовательной цепочке *RC*-звеньев задержка зависит от числа звеньев квадратично, поэтому цепи с большим числом ключей в них особенно нежелательны. Может оказаться целесообразным разбиение длинной линии на несколько коротких с помощью промежуточных буферных каскадов.

Вначале развитие ППВМ связывалось с переносом концепции БМК в область малотиражной аппаратуры, но в дальнейшем в связи с появлением репрограммируемости стало ясным, что это нечто большее. Это, в частности, видно из приведенных ниже примеров.

Рассмотренные области применения не являются исключительной прерогативой ППВМ, иногда возможно применение других СБИС ПЛ для решения перечисленных задач.

В различной аппаратуре встречаются ситуации, в которых те или иные блоки работают поочередно. Например, средства кодирования и декодирования при записи и чтении данных, использующие помехоустойчивые коды. Обе функции никогда не выполняются одновременно. Поэтому не обязательно иметь два устройства (кодер и декодер), а можно иметь одну ППВМ с двумя разными конфигурациями, хранимыми в ПЗУ. То есть одна и та же аппаратная часть может выполнять различные преобразования после соответствующей перестройки.

При отладке устройств традиционно пользовались изготовлением прототипа и программными моделями. Изготовление прототипа – сложная и дорогостоящая задача, но с его помощью можно вести тестирование с реальными сигналами и на высоких скоростях, наблюдая фактические возможности устройства. Программное моделирование лишено указанных достоинств, но проще и дешевле. Модели легко изменяются для удаления ошибок в проекте, и в них просто обеспечивается хорошая наблюдаемость процессов в объекте исследования.

Применение ППВМ в задачах логической эмуляции дает сочетание достоинств обоих классических подходов. Система из ППВМ легко создается и изменяется, но, с другой стороны, может работать с реальными сигналами и частотами их изменения. Однако по затратам труда и времени создание системы на ППВМ сложнее, чем создание программной модели. Поэтому программные модели не зачеркивают-

ся появлением репрограммируемых ППВМ. Следует также помнить, что полные свойства окончательно изготовленного устройства логическая эмуляция на ППВМ отобразить не может, так как временные характеристики зависят от конкретной трассировки схемы, чего еще нет на этапе логической эмуляции. Таким образом, логическая эмуляция не отменяет прежние методы разработки и тестирования схем, а лишь хорошо дополняет их.

Одним из применений ППВМ можно считать обогащение элементной базы цифровых устройств новыми микросхемами типа *FPIC* (*FPID*) – Field Programmable Interconnect Circuits (Devices).

Микросхемы *FPIC* содержат программируемые соединения и блоки ввода/вывода, но не имеют логических блоков. Они предназначены для произвольного соединения своих внешних выводов согласно программированию. Для окончательно изготовляемых продуктов это не является необходимым, но при отработке прототипов и в системах с динамически реконфигурируемой структурой такие микросхемы бесспорно полезны. Соединяя СБИС ПЛ через *FPIC*, можно легко изменять их межсоединения, чего не обеспечивают технологии с жесткой трассировкой (печатные платы и др.). Область применения *FPIC* более узка, чем у таких СБИС ПЛ, как *FPGA* и *CPLD*, соответственно, тиражность их производства ниже и стоимость выше.

ГЛАВА 4. ФУНКЦИОНАЛЬНЫЕ УСТРОЙСТВА КОМБИНАЦИОННОГО ТИПА

Как говорилось ранее, все цифровые устройства подразделяются на *комбинационные* и *последовательные* (с памятью).

Комбинационные устройства в свою очередь бывают *кодирующие* и *арифметические.*

Среди *арифметических* устройств наиболее широко распространены:

– сумматоры, обозначения микросхем которых в схемах содержат буквы *SM*, а на корпусах этих микросхем ставят обозначения, в которых имеются буквы ИМ;

– компараторы кодов, обозначения микросхем которых в схемах содержат буквы *COMP* или двойные знаки равенства = =, а на самих корпусах этих микросхем ставят обозначения, в которых имеются буквы *СА* или ИП;

– схемы контроля четности, обозначения микросхем которых в схемах содержат буквы и цифры М2, а на корпусах этих микросхем ставят обозначения, в которых имеются буквы ИМ;

– арифметическо-логические устройства, обозначения микросхем которых в схемах содержат буквы *ALU*, а на корпусах этих микросхем ставят обозначения, в которых имеются буквы ИП.

Кодирующее устройство преобразует многоразрядный входной код в выходной код, построенный по иному закону. Работа кодирующего устройства задается таблицей истинности и не может быть описана достаточно простым алгоритмом (как, например, сумматор).

Кодирующие устройства подразделяются на:

– дешифраторы, обозначения микросхем которых в схемах содержат буквы *DC*, а на самих корпусах этих микросхем ставят обозначения, в которых имеются буквы ИД;

– мультиплексоры, на корпусах этих микросхем ставят обозначения, в которых имеются буквы *MS*, а в обозначениях этих микросхем в схемах содержатся буквы КП;

– шифраторы, обозначения микросхем которых в схемах содержат буквы *CD*, а на корпусах этих микросхем ставят обозначения, в которых имеются буквы ИВ;

– преобразователи произвольных кодов, обозначения микросхем которых в схемах содержат буквы *X*/*Y*, *PLA* или *PROM*, а на самих корпусах этих микросхем ставят обозначения, в которых имеются буквы РЕ, РФ или РР.

4.1. Арифметические устройства

Особенность арифметических устройств состоит в том, что двоичным сигналам, двум известным состояниям низкого и высокого уровня приписываются не только логические, но и арифметические значения 0 и 1 и действия над ними подчиняются законам арифметики.

К арифметическим относятся устройства, выполняющие арифметические действия с двоичными числами: сложение, вычитание, умножение, деление, возведение в степень, взятие логарифма и т. д. К арифметическим относятся также специальные устройства:

– для определения четности (паритета);

– для сравнивания двух многоразрядных чисел (компараторы);

– для мажоритарного контроля (кворум-элементы).

Все арифметические операции с двоичными числами сводятся к сложению: вычитание – это сложение двух чисел, одно из которых представлено в обратном или дополнительном коде; умножение – это многократное сложение и сдвиг; деление – это многократное вычитание и сдвиг. Степенные, логарифмические и другие сложные функции в цифровой электронике, в ЭВМ вычисляются по приближенным формулам, включающим сложение и умножение.

4.1.1. Сумматор

Сумматором (от SUMMER, SUMMATOR) называется схема, предназначенная для сложения чисел в двоичном коде.

Сумматор двух одноразрядных слагаемых называется *полусумматором* и обозначается *HS* – HALF SUM – половина суммы.

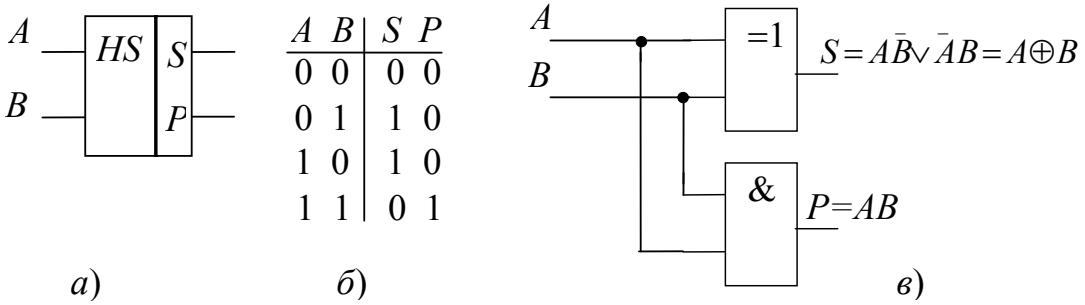


Рис. 4.1. Полусумматор *HS*: *а* – условное обозначение; *б* – таблица истинности; *в* – схема полусумматора

Выход $P = AB$, называют *переносом* (переполнением) и иногда обозначают *C* или *CR* (от CARRY – перенос).

Чтобы не путать обозначения логических и арифметических действий при описании арифметических устройств, знаком «+» будем

обозначать только арифметическую сумму, а знаком $\vee$ – логическую функцию ИЛИ, т. е. логическое сложение. Знак $\oplus$ будет означать суммирование по модулю 2 для арифметических двоичных переменных и функцию исключающие ИЛИ для логических.

При сложении двух многоразрядных двоичных чисел, кроме двух входов слагаемых, в сумматоре каждого разряда должен быть предусмотрен еще вход для переноса из младшего разряда. Полусумматор имеет только два входа и пригоден для суммирования только одноразрядных слагаемых и не пригоден для суммирования всех других разрядов слагаемых.

Для сложения двух слагаемых любой разрядности с учетом переноса из младшего разряда предназначен одноразрядный полный сумматор, или просто сумматор.

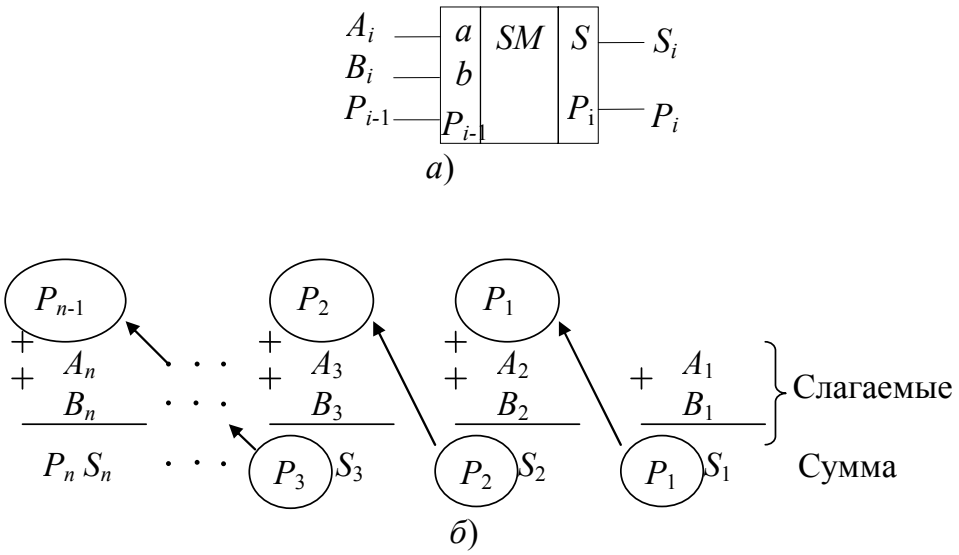
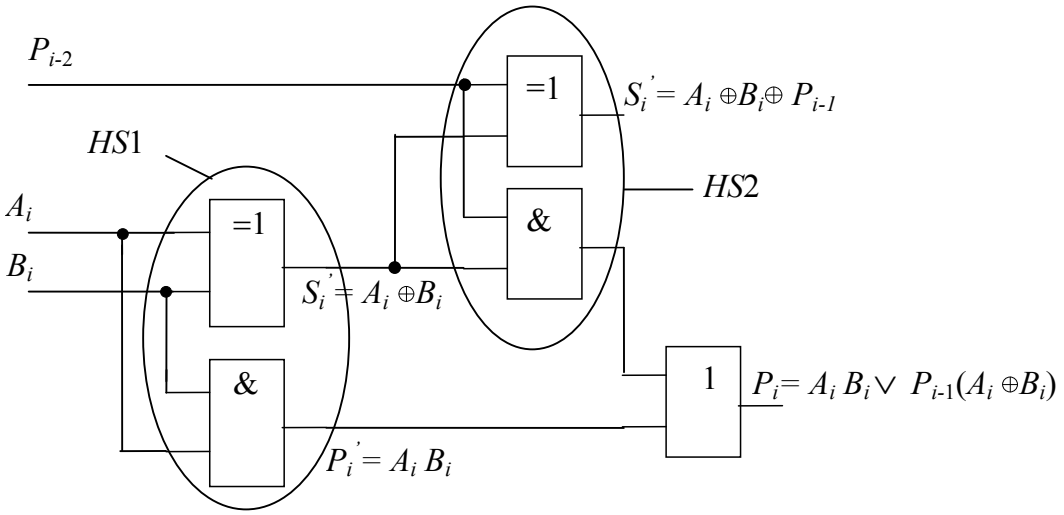


Рис. 4.2. Полный сумматор: *а* – условное обозначение сумматора; *б* – порядок сложения им многоразрядных чисел

Сумматор можно получить из двух полусумматоров (рис. 4.3). Первый полусумматор *HS*1 складывает два слагаемых и выдает промежуточную сумму $S_i^{'}$ и бит переноса $P_i^{'}$. Второй полусумматор *HS*2 суммирует бит переноса предыдущего разряда P_{i-1} с промежуточной суммой S_{i-1}, в результате получается полная сумма S_i. Бит переноса получаем при участии двух полусумматоров и дополнительных логических элементов ИЛИ.



а)

N	P_i	A_i	B_i	S_i	P_i
0	0	0	0	0	0
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	1

б)

Рис. 4.3. Сумматор, полученный из двух полусумматоров: *а* – условное обозначение; *б* – его таблица истинности

По схеме запишем булевы выражения для выходов S_i и P_i:

$$S_i = \overline{P}_{i-1}\overline{A}_i B_i \vee \overline{P}_{i-1} A_i \overline{B}_i \vee \overline{P}_{i-1}\overline{A}_i \overline{B}_i \vee P_{i-1} A_i B_i,$$

$$P_i = \overline{P}_{i-1} A_i B_i \vee P_{i-1}\overline{A}_i B_i \vee P_{i-1} A_i \overline{B}_i \vee P_{i-1} A_i B_i.$$

Булевы выражения для выходов S_i и P_i после минимизации и с учетом друг друга выглядят следующим образом:

$$S_i = \bar{P}_i\, A_i \vee \bar{P}_{i-1}\, \bar{B}_i \vee \bar{P}_i\, P_{i-1} \vee P_{i-1} A_i B_i,$$

$$P_i = P_{i-1} A_i \vee P_{i-1} B_i \vee A_i B_i.$$

Схема сумматора, построенная по этим минимизированным выражениям на логических элементах И–ИЛИ–НЕ, используется в качестве основы в микросхемах К155ИМ1, К155ИМ2, К155ИМ3 и представлена на рис. 4.4.

Здесь выходы $\overline{S_i}$ и $\overline{P_i}$ инверсные, что требует применения дополнительных инверторов.

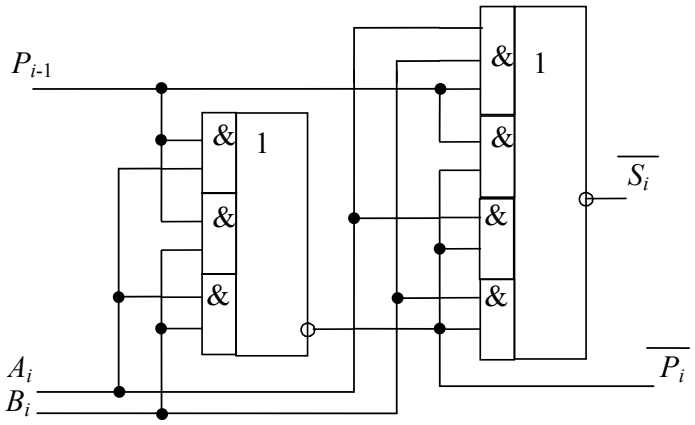


Рис. 4.4. Схема сумматора, построенная по минимизированным выражениям на логических элементах И–ИЛИ–НЕ

4.1.2. Последовательное и параллельное суммирование

Одноразрядный сумматор может быть использован для суммирования многоразрядных двоичных чисел, если они представлены последовательным кодом (рис. 4.5), в котором младшие разряды следуют раньше старших. В этом случае сигнал с выхода переноса подается на вход переноса этого же сумматора через цепь задержки, обеспечивающую хранение бита переноса на время одного такта следования импульсов входных цифровых слагаемых. Это самый медленный способ суммирования многоразрядных чисел, но он самый экономичный по аппаратурным затратам.

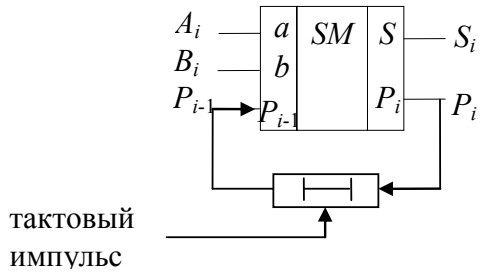


Рис. 4.5. Последовательное суммирование

В случае *параллельного суммирования* число сумматоров должно быть равно числу разрядов суммируемых чисел. Быстродействие ограничено задержкой переноса ($T_{\text{зд.общ}} = nt_{\text{зд}}P_i$), так как формирование сигналов суммы S_n и переноса P_n старшего разряда не может произойти до тех пор, пока сигнал переноса младшего разряда не распространится последовательно по всей цепочке.

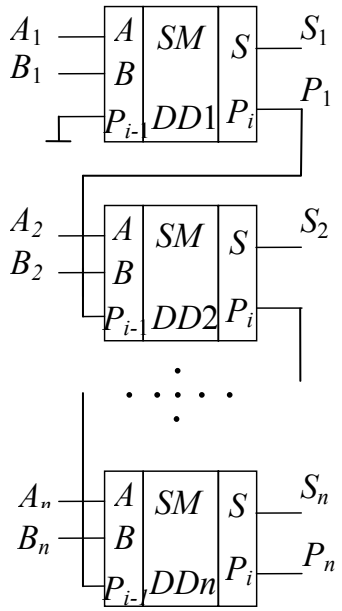


Рис. 4.6. Параллельное суммирование с последовательным переносом

Время задержки переноса можно уменьшить, вводя *параллельный перенос*, для чего применяют *схемы ускоренного переноса* (СУП). В этих схемах перенос каждого разряда вырабатывается независимо от переноса соседнего младшего разряда. Он формируется из слагаемых данного разряда и из входного переноса всего сумматора (в предыдущем разряде он подавался на вход переноса младшего разряда).

Для описания работы сумматора с параллельным переносом введем две вспомогательные функции – γ и π:

γ – функция генерации переноса (от «CARRY GENERATORION» – *CRG)*; $\gamma_i = A_i B_i$:

$\gamma_i = 1$, если в данном *i*-м разряде сумматора генерирует перенос и если $A_i = B_i = 1$;

$\gamma_i = 0$ во всех других случаях;

π – функция прозрачности или распространения («CARRY PROPOGATION» – *CRPP*);

$\pi = A_i \vee B_i$, иногда принимают $\pi = A_i \oplus B_i$.

На конечный результат суммирования не влияет никакой из этих двух вариантов вычисления π. Один из этих вариантов схемной реализации выбирают из соображения удобства его применения в конкретной логике ТТЛ или МОП.

π = 1 когда тракт переноса данного *i*-го разряда сумматора прозрачен для сигнала $P_{i\text{-}1}$ переноса предыдущего $i - 1$ разряда в последующий $i + 1$разряд.

Процесс формирования ускоренного переноса описывается уравнением $P_i = \gamma_i \vee \pi_i\gamma_{i-1} \vee \pi_i\pi_{i-1}\gamma_{i-2} \vee \ldots \vee \pi_i\pi_{i-1}\ldots\pi_1\pi_2\gamma_0$.

На практике функции γ и π – это промежуточные сигналы разрядов схемы сумматора, которые используют для формирования сигнала параллельного переноса всего сумматора.

4.1.3. Вычислитель (субтрактор)

В устройствах дискретной техники операция вычитания обычно заменяется сложением уменьшаемого с вычитаемым, когда последнее представлено в дополнительном коде. Устройство, выполняющее данную функцию, называют вычислитель.

Пусть надо получить разность двух чисел *A–B*, где *A* и *B* – *n*-разрядные двоичные числа.

Переведем число *B* в дополнительный код: $B_{\text{доп}} = \overline{B} + 1$.

Если $B + B_{\text{доп}} = B + \overline{B} + 1 = \underbrace{1111\ldots1+1}_{n\text{-разрядное число}} = \underbrace{10}_{(n+1)\text{-й разряд}}000\ldots0,$

$$\underbrace{B = 1000}_{n\text{ разрядов}}\ldots0 - \underbrace{B_{\text{доп}} = 111\ldots1}_{n+1\text{ разряд}} + 1 - \bar{B} - 1.$$

$$A–B = \underbrace{A - 1000}_{n+1}\ldots\underbrace{1 + \bar{B}}_{B_{\text{доп}}} + 1 = A + \underbrace{B_{\text{доп}} - 1000}_{n\,+\,1\text{ разряд}}\ldots0.$$

Рис. 4.7. Пример получения разности *n*-разрядных двоичных чисел

Это значит, что для вычитания двух чисел достаточно произвести их сложение, однако при этом вычитаемое должно быть представлено в дополнительном коде.

Если разность, полученная при таком способе вычитания, положительна, т. е. $A > B$, то результат будет представлен в прямом коде,

а в разряде, старше старшего, т. е. в (n+1)-м разряде образуется 1, которой можно пренебречь.

Если при таком способе вычитания разность отрицательна, т. е. $A < B$, то результат будет представлен в дополнительном коде, при этом 1 в (n+1)-м разряде не образуется (рис. 4.8).

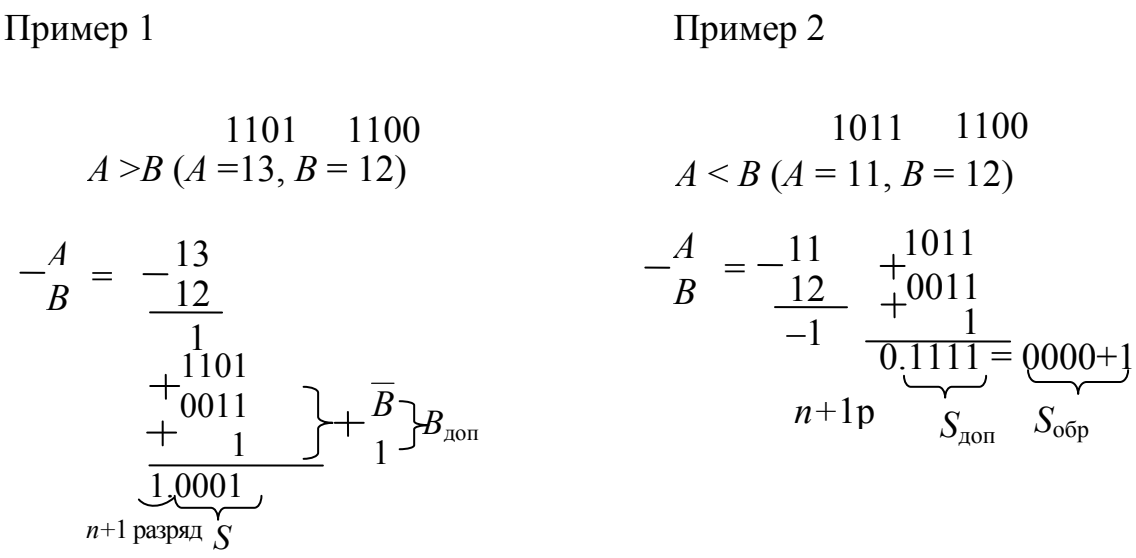


Рис. 4.8. Примеры вычитания

4.1.4. Умножение и деление двоичных чисел

Пример умножения двух чисел представлен ниже (рис. 4.9):

$$\begin{array}{rcl} \times\ 1101_2 & = & \times\ 13_{10} \\ \underline{\quad 101_2} & = & \underline{\quad\ 5_{10}} \\ 1101 & & \\ 0000 & & \\ \underline{1101\quad\ } & & \underline{\quad\ } \\ 1000001_2 & & 65_{10} \end{array}$$

Рис. 4.9. Пример умножения двух двоичных чисел

Так как частичное произведение многоразрядного числа на 1 равно этому числу, а умножение на 0 дает нули во всех разрядах, то операция умножения сводится к операциям сдвига и сложения частичных произведений.

Схема умножителя четырехразрядного числа $A_4A_3A_2A_1$ на трехразрядное число $B_3B_2B_1$ реализовано на микросхемах типа К155ИМ3. Семиразрядное произведение $M_7M_6M_5M_4M_3M_2M_1$ на выходе умножителя формируется за счет параллельного умножения умножаемого A

на каждый разряд множителя *B* логическими элементами 2–И и сложение промежуточных произведений со сдвигом на один разряд – сумматорами *SM DD*1 и *DD*2.

Применение логических элементов И для выполнения арифметической операции умножения в данном случае допустимо, поскольку в рамках одного разряда и арифметическое умножение, и логическое (функции И, конъюнкция) подчиняются общим правилам.

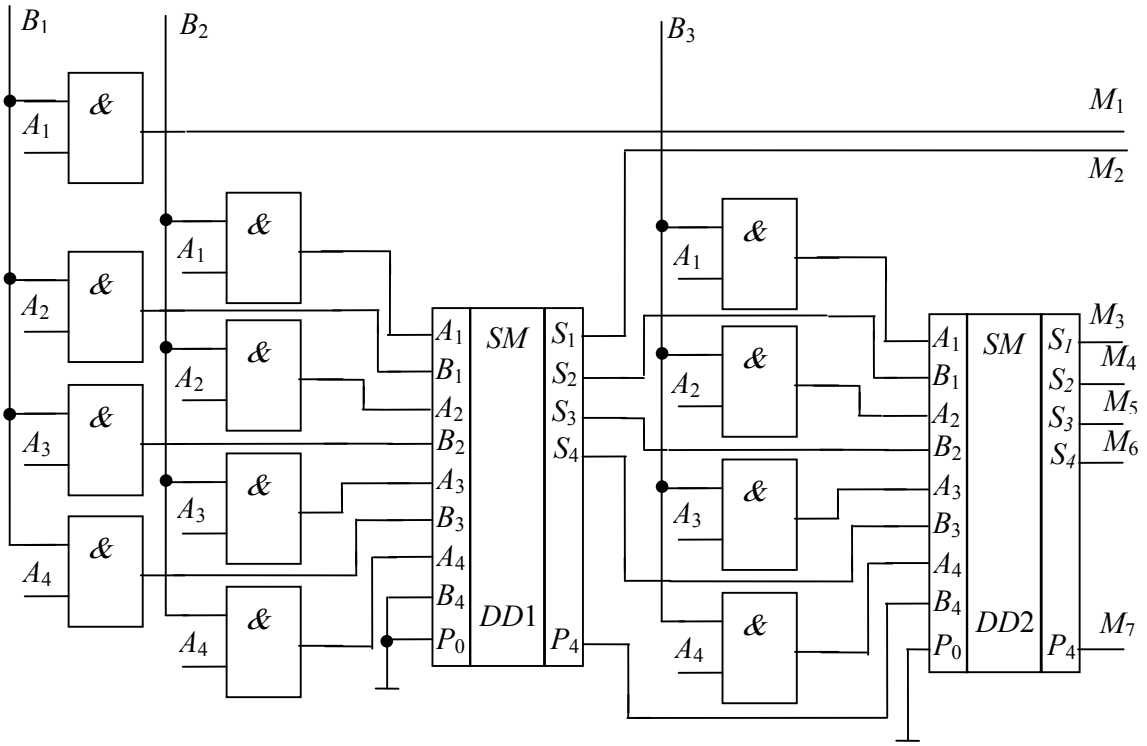


Рис. 4.10. Структура микросхемы К155ИМ3

Двоичное деление сводится к следующим операциям: вычитанию делителя из делимого, сдвигу делителя на один разряд, сравнению полученного остатка с делителем (рис. 4.11).

$$
\begin{array}{l|l}
\underline{-110111_2=55_{10}} & \underline{101=5_{10}} \\
101 & 1011=11_{10} \\
\hline
111 & \\
\underline{101} & \\
101 & \\
\underline{101} & \\
000 &
\end{array}
$$

Рис. 4.11. Пример деления двух двоичных чисел

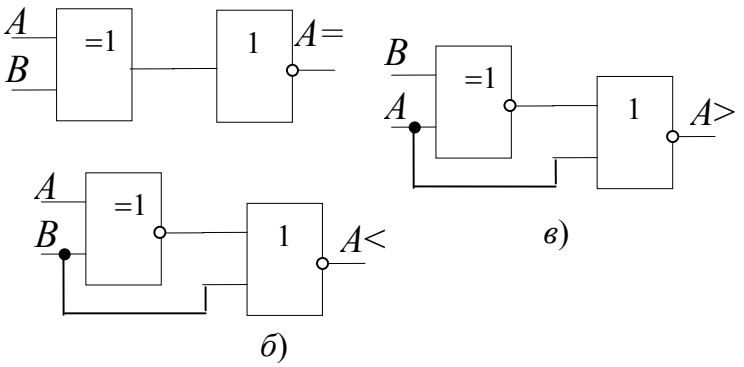
Если остаток меньше делителя, делитель сдвигается еще на один разряд, затем снова делитель сравнивается с остатком, и если остаток больше делителя, то из него вычитается делитель, затем опять следует сдвиг делителя, вновь сравнение, так до тех пор, пока после сдвига делителя его младший разряд сравняется с младшим разрядом делимого. В тех разрядах, где сравнение делителя с остатком показало, что остаток меньше делителя, в данный разряд результата деления – частного – записывается нуль, в остальные разряды – 1. Итак, деление сводится к вычитанию, сдвигу и сравнению.

4.1.5. Компаратор

Компаратор – схема сравнения двоичных чисел (от «COMPARY» – сравнивать, сличать). Компаратор сравнивает два многоразрядных числа A и B по величине. Обычно схема имеет три выхода: $A < B$, $A = B$, $A > B$; в зависимости от соотношения входных чисел A и B активный уровень появляется на одном из этих выходов. Для одноразрядных чисел из таблицы истинности видно, что $A =$ есть инверсия результата суммы по модулю два.

A	B	$A=$	$A<$	$A>$
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

а)



б) *в*)

Рис. 4.12. Результат сравнения одноразрядных двоичных чисел:
а – таблица истинности; *б*, *в* – схемы получения

Компаратор может быть построен как вычислитель на микросхеме сумматора, только при этом второе число B подается на вход в дополнительном коде.

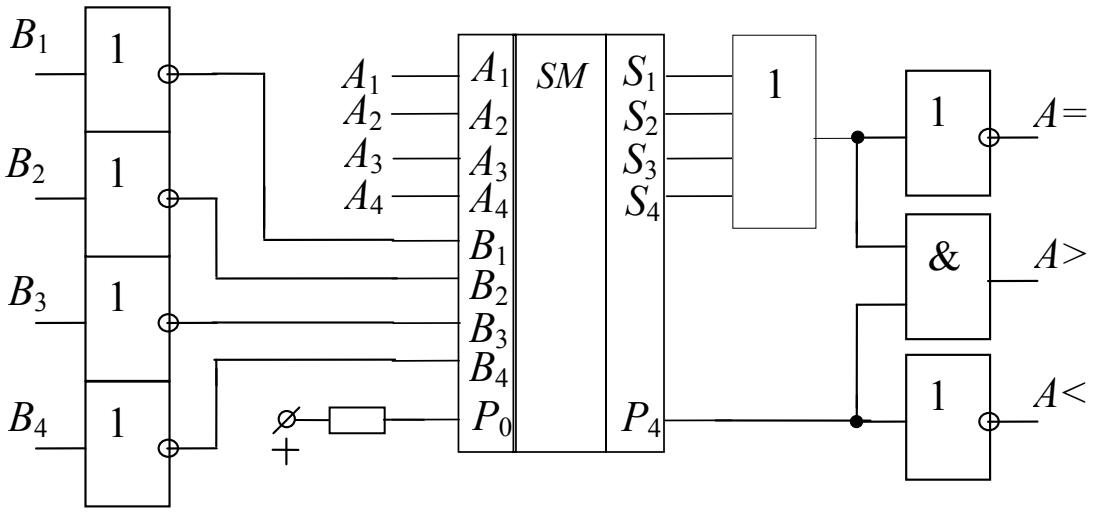


Рис. 4.13. Схемная реализация компаратора
при помощи сумматора К555СП1

Если вспомнить раздел о вычитании двоичных чисел, то станет понятно, почему из сигнала переноса получаются сигналы $A > B$ и $A < B$. Входы $A <$, $A =$ и $A >$ микросхем компараторов предназначены для наращивания разрядности.

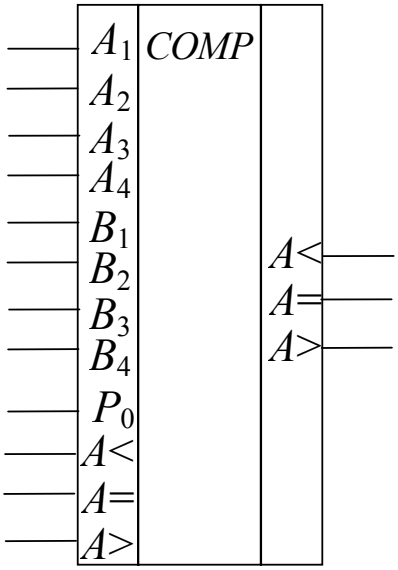


Рис. 4.14. Условное обозначение микросхемы компаратора К561ИП2

4.1.6. Контроль четности

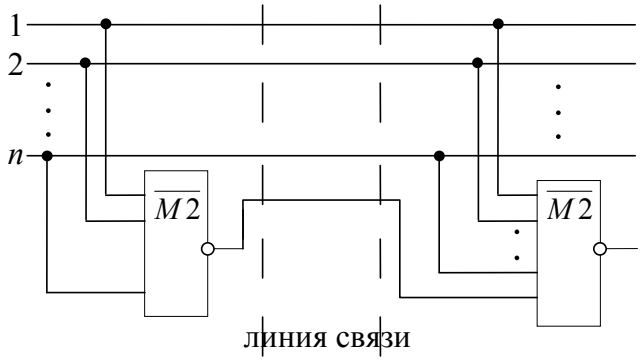
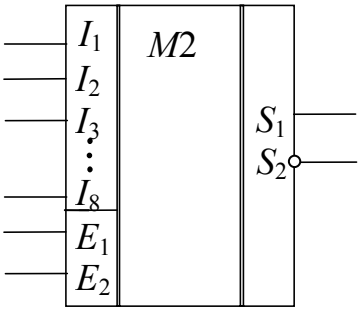


Рис. 4.15. Схемное решение контроля четности

Практически любой канал передачи данных или любое запоминающее устройство (ЗУ), если они не имеют какого-либо более сильного метода контроля, защищены *контролем по четности*, по *паритету* (*PARITY* – соответствие).

На n-м входном элементе $\overline{M2}$ формируется признак четности n-разрядного числа, передаваемого по линии связи. Вместе с n-разрядным числом передают и (n+1)-й разряд, содержащий признак четности. Такое передаваемое (n+1)-разрядное число всегда имеет нечетное число единиц. На приемном конце линии связи (или после чтения из памяти записывающего устройства) у полученного (n+1)-разрядного числа проверяют четность (n+1)-разрядным элементом $\overline{M2}$. Если на выходе этого элемента логическая 1, т. е. входное (n+1)-разрядное число – четное, то делают вывод об ошибке в канале связи или в ЗУ.

К155ИП2



Входы			Выходы	
данных	управл.			
Число единиц на входах $I1$–$I8$	$E1$	$E2$	$S1$	$S2$
четное	1	0	1	0
не четное	1	0	0	1
четное	0	1	0	1
не четное	0	1	1	0
X	1	1	0	0
X	0	0	1	1

а) *б*)

Рис. 4.16. Микросхема К155ИП:
а – условное обозначение; *б* – таблица состояний

Это дает возможность исключить неверные данные, затребовать повторную передачу и т. п.

Разрешающие входы микросхем компараторов $E1$ и $E2$ иногда обозначают как $W1$ и $W2$, или как EE и EO; а выходы $S1$ и $S2$ могут быть обозначены как $\sum E$ (четная сумма) и $\sum O$ (нечетная сумма).

Микросхемы для контроля четности, кроме n-разрядного элемента $M2$, дополняют схемами выбора признака контроля: по четности или по нечетности.

4.1.7. Узлы мажоритарного контроля

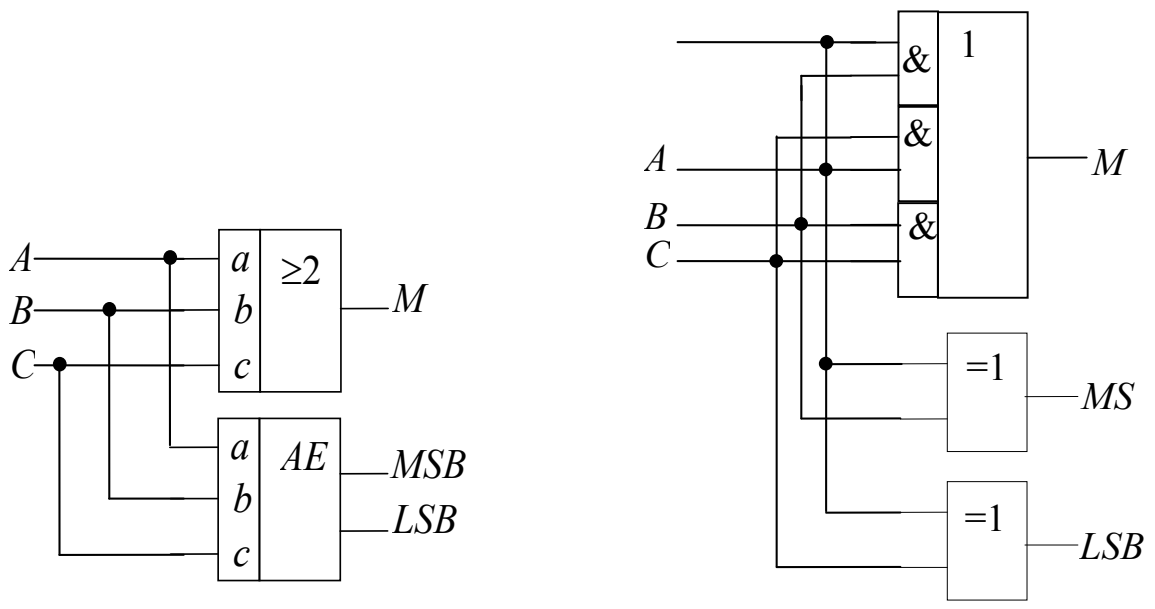
Узлы мажоритарного контроля являются простейшими схемами, позволяющими не только определять, но и исправлять ошибки, произошедшие при передаче или обработке данных.

При этом используются три канала – при передаче, или три блока – при обработке данных. Выходные данные этих каналов поступают на узел мажоритарного контроля (*узел мажорирования*, *кворум-элемент*), который формирует из них один выходной сигнал по принципу голосования «два из трех». Таким образом, выходной сигнал равен логической 1, если на двух или трех выходах логическая 1.

В состав узла мажоритарного контроля обычно вводят элемент, позволяющий оценить номер отказавшего (несогласного с остальным) канала или блока (*AER* – ADRESS ERROR). Его выходы – *MSB* – (MOST SIGNIFICAT BTT) – наиболее значимый разряд и *LSB* – (LEAST BIT) – наиболее значимый разряд двоичного кода, обозначающего номер отказавшего канала.

Булевы выражения, реализуемые на выходах данной схемы, записываются следующим образом:

$$M = AB \vee BC,\ MSB = A \oplus B,\ LSB = A \oplus C.$$



а) *б*)

A	*B*	*C*	*M*	*MSB*	*LSB*
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	0	1	1
1	0	1	1	1	0
1	1	0	1	0	1
1	1	1	1	0	0

в)

Рис. 4.17. Реализация узла мажоритарного контроля:
а – при помощи элемента *AE*;
б – его схемная реализация на логических элементах;
в – его таблица истинности

4.1.8. Арифметико-логические устройства (ALU – ARITHMETIC and LOGIK UNIT)

В микропроцессорной технике АЛУ (рис. 4.18) являются базовыми элементами. Входы $A_0A_1A_2A_3$ и $B_0B_1B_2B_3$ являются информационными (т. е. на них подаются 4-х-разрядные двоичные числа); входы $S_0S_1S_2S_3$ задают логические и арифметические операции, производимые над этими числами. Вход M регулирует, какие функции выполняются (при $M = H$ (высокий) микросхема выполняет логические функции) от четырехразрядных входных переменных A и B. Выбор

конкретной функции из 16 возможных задается набором входных управляющих сигналов $S0\ldots S3$:

$$A;\ \overline{A};B;\ \overline{A};AB;\ \overline{AB};A\vee B;\ \overline{A\vee B};A\,\overline{B};A\vee\overline{B};A\oplus B\ldots\ .$$

При $M = L$ (низкий) АЛУ ИП3 производит арифметические операции над четырехразрядными двоичными числами A и B. Набор (комбинация) входных сигналов $S0\ldots S3$ определяет одну из 16 возможных операций:

$$A-1;AB-1;\ A\,\overline{B}-1;-A;\ A+\overline{B};\ A+AB;\ A+A_{\text{сдв}}\ldots\ .$$

Здесь $A_{\text{сдв}}$ равно A, каждый разряд которого сдвинут в направлении более высокого разряда.

C_n – вход, а C_{n+4} – выход переноса.

G и P – выходы функций γ генерации переноса и π – прозрачность переноса для подключения схемы ускоренного переноса К155 ИП4.

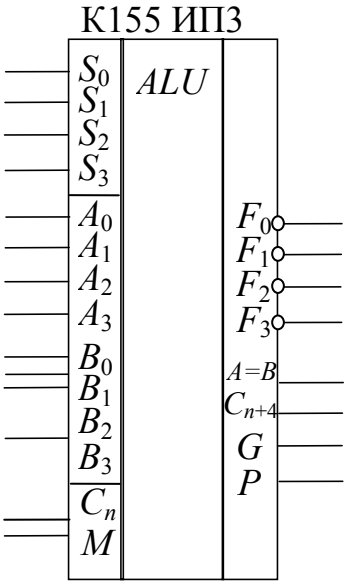


Рис. 4.18. Условно-графическое обозначение микросхемы АЛУ К155ИП3

4.2. Преобразователи двоичных кодов

4.2.1. Преобразователь кода Грея в двоичный позиционный

Код Грея, *отраженный двоичный код*, *рефлексный* (от *reflect* – отражать) код для первых десяти чисел представлен в табл. 4.1.

В последовательности чисел кода Грея все разряды (*abc*), кроме самого левого, подчиняются следующему правилу: любая сплошная группа разрядов, считая справа, по некоторому закону перебирает все свои возможные комбинации, а затем начинает перебирать их в обратном порядке. Так, разряды кода Грея *b* и *c* сначала перебирают комбинации 00, 01, 11, 10, затем – те же комбинации в обратном порядке: 10, 11, 01, 00.

Закон справедлив вплоть до группы в один разряд: разряд с изменяется сначала в одну сторону – 0,1, затем обратно – 1,0 и снова в том же цикле. Этот закон объясняет одно из названий кода «отраженный» и позволяет строить последовательность чисел кода любой разрядности. Код Грея *не позиционный*, т. е. веса его разрядов не определяются занимаемыми ими местами, как в обычном двоичном коде, который относится к классу позиционных.

Таблица 4.1

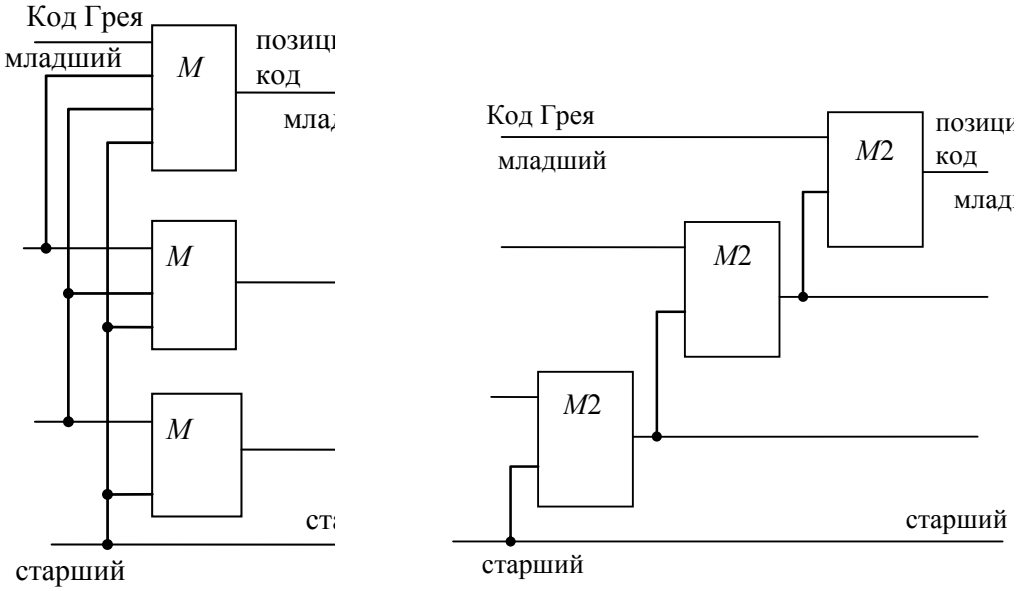
Таблица кода Грея для 8 чисел

Десятичное число	**Двоичное число (*abc*)**	**Число в Коде Грея (*abc*)**
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

В коде Грея при переходе между любыми соседними числами изменяется значение всегда только одного разряда. Благодаря этому свойству код широко применяется в преобразователях углового положения вала в цифровой код, построенных на базе кодового диска или кодового барабана. В оптическом кодовом диске единицы и нули кодируются прозрачными и непрозрачными областями.

Два варианта схем преобразования кода Грея в позиционный показаны на рис. 4.19. Первый (параллельная схема, рис. 4.19, *а*) имеет меньшую задержку, но дороже по затратам оборудования, второй (последовательная схема, рис. 4.19, *б*) с большей задержкой, но экономичнее.

Переводить числа из позиционного кода в код Грея приходится значительно реже. Алгоритм перевода прост: каждый *i*-й, считая слева, разряд числа в коде Грея равен сумме по модулю 2 *i*-го и (*i*–1)-го слева разрядов того же числа, представленного позиционным кодом.



а) *б*)

Рис. 4.19. Преобразователи кода Грея в двоичный позиционный:
а – параллельная схема; *б* – последовательная схема

4.3. Дешифраторы

Кодирующим устройством называют логический узел, преобразующий многоразрядный входной код в выходной код, построенный по иному закону. Название в большой мере условно, поскольку любое цифровое устройство преобразует некоторый входной код в некоторый выходной, т. е. является кодовым преобразователем. Традиционно это название применяется к узлам, работа которых не описывается достаточно простым алгоритмом, как, например, работа сумматора, а задается таблицей соответствия входов и выходов.

Дешифратором, или *декодером* (decoder), чаще всего называют кодирующее устройство, преобразующее двоичный код в унарный. Из всех *m* выходов дешифратора активный уровень имеется только на

одном, а именно на том, номер которого равен поданному на вход двоичному числу. На всех остальных выходах дешифратора уровни напряжения неактивные. Условное изображение дешифратора (3–8) показано на рис. 4.20.

Если декодер имеет n входов, m выходов и использует все возможные наборы входных переменных, то $m = 2^n$. Такой декодер называют *полным* в отличие от *неполного*, использующего лишь часть возможных наборов и имеющего соответственно меньшее число выходов и внутренних схемных элементов.

Дешифратор используют, когда нужно обращаться к различным цифровым устройствам, и при этом номер устройства – его адрес – представлен двоичным кодом. Входы декодера (их иногда называют *адресными входами*) часто нумеруют не порядковыми номерами, а в соответствии с весами двоичных разрядов, т. е. не 1, 2, 3, 4, 5, ..., а 1, 2, 4, 8, 16... . Число входов и выходов декодера указывают таким образом: декодер 3–8 (читается «три в восемь»); 4–16; 4–10 (это неполный декодер).

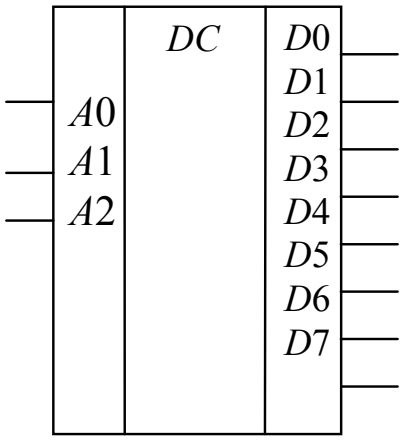


Рис. 4.20. Обозначения дешифратора (3–8)

Формально описать работу дешифратора можно, задав список функций, отрабатываемых каждым из его выходов.

Булевы выражения выходных функций дешифратора 2–4:

$$D0 = \overline{A1} \cdot \overline{A0}, \quad D1 = \overline{A1} \cdot A0,$$

$$D2 = A1 \cdot \overline{A0}, \quad D3 = A1 \cdot A0.$$

Реализация этих четырех выражений с помощью четырех двухвходовых элементов И дает наиболее простой по структуре дешифратор, называемый *линейным* (рис. 4.21).

Если дешифратор состоит из элементов И–НЕ, то на его выходах будут отрабатываться не сами функции Y_i (табл. 4.2), а их инверсии, т. е. активным уровнем выхода будет низкий. Ликвидировать инверсии на выходах можно или подключив инверторы, или построив дешифратор по двойственной схеме – на элементах ИЛИ–НЕ. Число входных инверторов и буферных усилителей при этом не изменится.

Таблица 4.2

Таблица истинности дешифратора 3–8 с инверсными выходами

$A2$	$A1$	$A0$	$\overline{Y7}$	$\overline{Y6}$	$\overline{Y5}$	$\overline{Y4}$	$\overline{Y3}$	$\overline{Y2}$	$\overline{Y1}$	$\overline{Y0}$
0	0	0	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	0	1
0	1	0	1	1	1	1	1	0	1	1
0	1	1	1	1	1	1	0	1	1	1
1	0	0	1	1	1	0	1	1	1	1
1	0	1	1	1	0	1	1	1	1	1
1	1	0	1	0	1	1	1	1	1	0
1	1	1	0	1	1	1	1	1	1	1

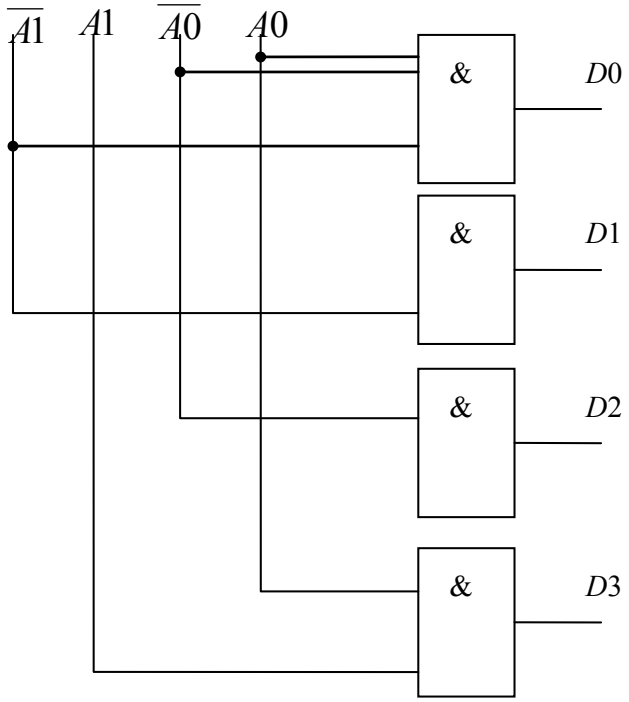


Рис. 4.21. Схема линейного дешифратора 2–4 на логических элементах И

Дешифраторы часто имеют *разрешающий* (управляющий, стробирующий) вход E (от *enable* – давать возможность). При $E = 1$ дешифратор работает как обычно, при $E = 0$ на всех выходах устанавли-

ваются неактивные уровни независимо от поступившего кода адреса. Вход *E* часто выполняют инверсным.

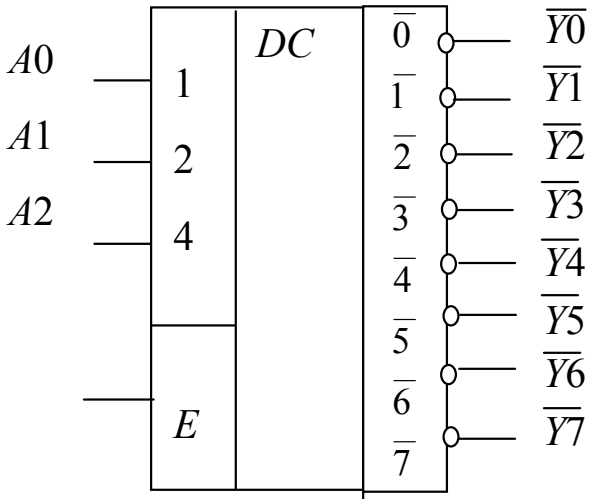


Рис. 4.22. Условно-графическое обозначение дешифратора 3–8 с инверсными выходами и прямым разрешающим входом

Иногда входов *E* бывает несколько, причем часть их может быть прямыми, а часть инверсными входами. Тогда их обычно отделяют на левом (входном) поле горизонтальной чертой от остальных входов:

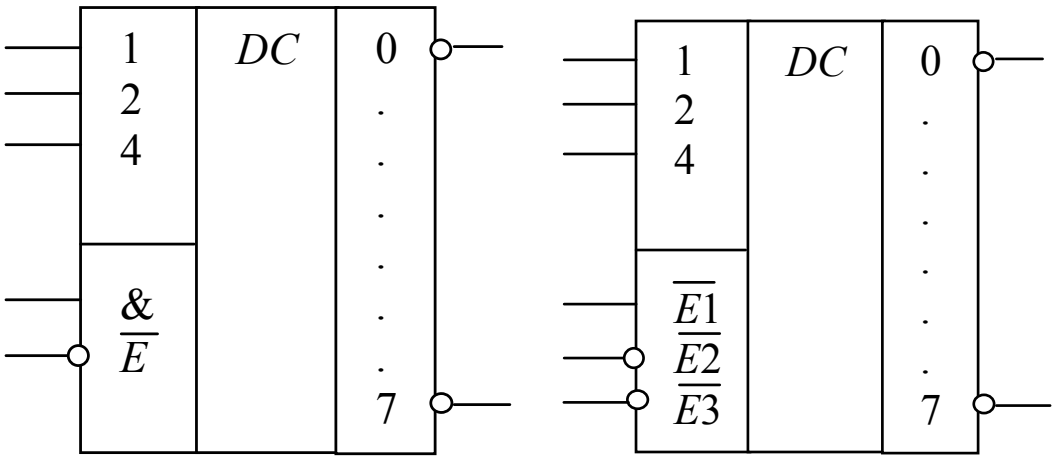


Рис. 4.23. Пример обозначения дешифраторов с несколькими разрешающими входами

Микросхемы дешифраторов часто имеют не один, а два или даже три входа *E* разрешения, причем некоторое из них прямые, а другие – инверсные. Такие входы удобно использовать при наращивании дешифратора, собирая как бы каскадный дешифратор, но вместо первого каскада можно использовать входы разрешения микросхем – дешифраторов.

На рис. 4.24 показана группа из пяти дешифраторов, соединенных в два каскада. Вся группа работает как дешифратор 5–32. Два старших разряда адреса a_{16} и a_8 расшифровываются дешифратором 2–4 *DC*4, который по входам *E* управляет четырьмя дешифраторами 3–8 второго каскада. Младшие разряды адреса a_4, a_2, a_1 поступают на все дешифраторы второго каскада, но открытым по входу *E* оказывается лишь один из них. Ему и будет принадлежать единственный из всех 32 активный выход. Так, при поступлении кода $a_{16}\ a_8\ a_4\ a_2\ a_1 = 01111$ у *DC*4 сигнал появится на выходе 1, и по входу *E* будет открыт *DC*1. Остальные дешифраторы второго каскада будут заперты. Разряды адреса $a_4\ a_2\ a_1 = 111$ вызовут появление 1 на выходе 7 дешифратора *DC*1, т. е. на выходе 15 всего составного дешифратора, что соответствует заданному адресу. Принцип используется при построении дешифраторов на много выходов из микросхем дешифраторов с меньшим числом выходов.

В рассмотренном примере 5-разрядный адрес был разбит на две группы в 2 и 3 разряда, и это определило структуру всей схемы. В общем случае многоразрядный адрес можно разбить на группы различными способами, и каждому способу будет соответствовать свой вариант схемы многокаскадного (не обязательно двухкаскадного) дешифратора. Варианты будут отличаться задержкой и аппаратурными затратами, и можно ставить задачу выбора оптимальной в заданной серии элементов структуры.

В предельном случае при числе каскадов, равном разрядности адреса, получается *пирамидальный дешифратор*, имеющий максимально возможную задержку. Достоинством его является использование только двухвходовых элементов И, что определило его широкое распространение на заре цифровой техники. В современных многовходовых логических базисах пирамидальная схема дешифратора не выдерживает конкуренции с другими структурами даже по оборудованию, не говоря уже о задержке.

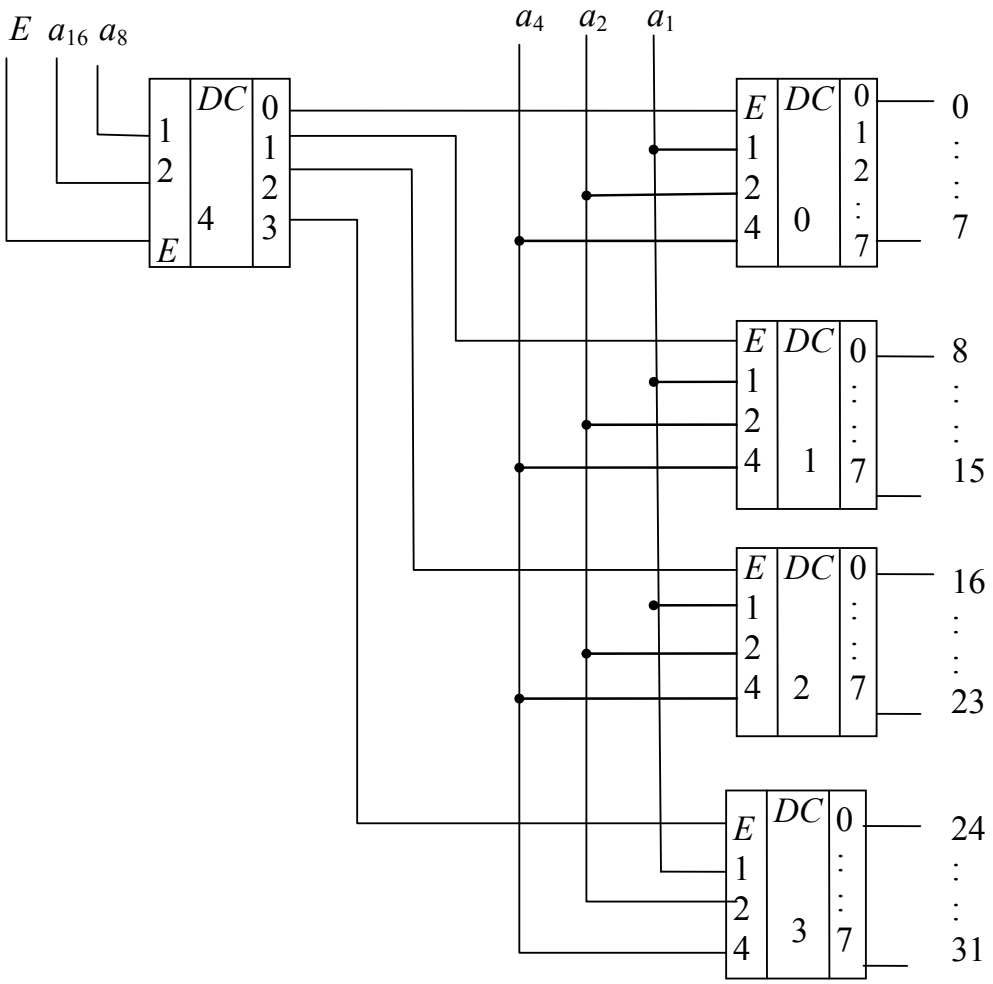


Рис. 4.24. Каскадное соединение дешифраторов

На рис. 4.25 показан двухкаскадньнй дешифратор 4–16, второй каскад которого собран по схеме *прямоугольного дешифратора*. Разряды адреса разбиты на две группы, каждая из которых независимо от другой расшифровывается своим *дешифратором первого каскада* *DC*1 и *DC*2. При любой комбинации значений входных переменных оказываются выбранными одна строка *m* и один столбец сетки, в узлах которой расположены элементы И второй ступени. В результате каждый входной набор активирует выход единственного соответствующего ему элемента И. Такую сетку из элементов И и называют *прямоугольным*, или *матричным* дешифратором.

При использовании во второй ступени элементов И–НЕ выходы дешифратора будут инверсными. Их можно сделать прямыми, построив сетку второго каскада по двойственному варианту, на элементах ИЛИ–НЕ; тогда инверсными должны быть выходы дешифраторов первого каскада.

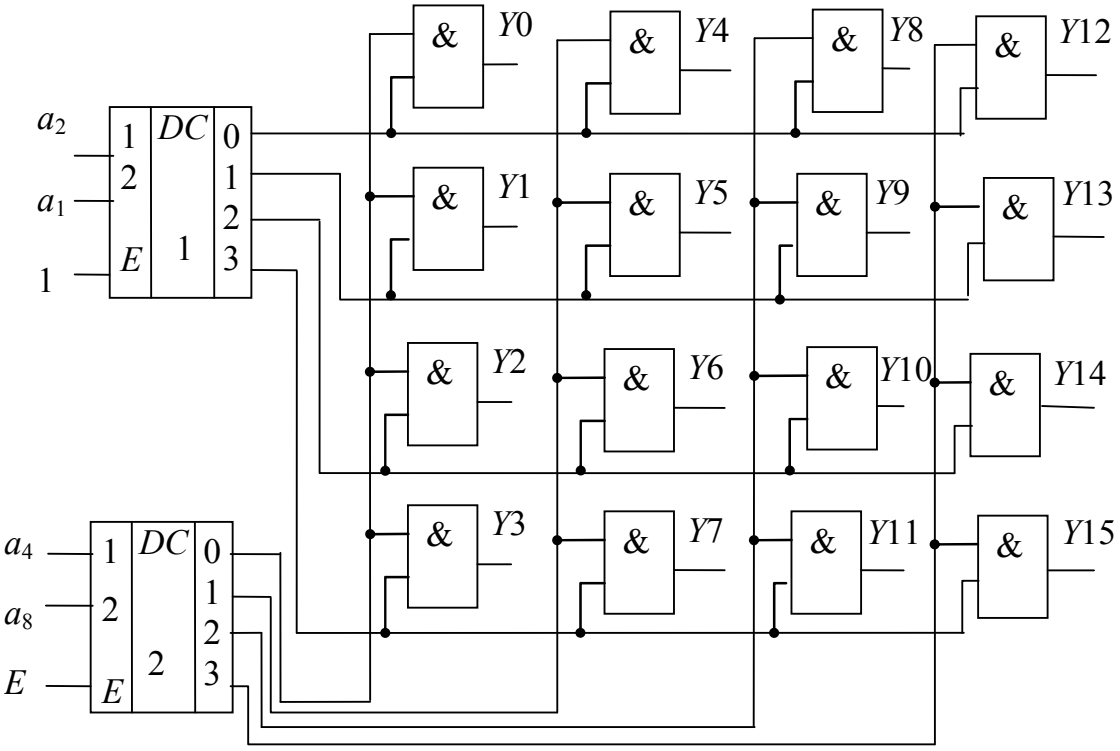


Рис. 4.25. Двухкаскадный дешифратор с прямоугольным (матричным) дешифратором во втором каскаде

Делить разряды адреса между *DC*1 и *DC*2 нужно по возможности поровну: чем ближе прямоугольник второго каскада к квадрату, тем при том же числе выходных элементов И меньше сумма его строк и столбцов, т. е. меньше число выходов дешифраторов первого каскада. В качестве входа *E* всего двухкаскадного дешифратора удобно использовать разрешающий вход одного из дешифраторов первого каскада. При этом запираются или все строки, или все столбцы.

Линейный дешифратор имеет минимальную задержку. Но у него максимальные из всех типов дешифраторов аппаратурные затраты.

Матричный (прямоугольный) дешифратор самый экономичный по оборудованию, имеет среднюю задержку.

Каскадный дешифратор имеет наибольшую задержку при числе каскадов больше двух, но удобен при использовании готовых микросхем – дешифраторов.

Дешифраторы, выпускаемые в виде отдельных микросхем, имеют буквенное обозначение ИД. В сериях ТТЛ, в которых элементы И–НЕ наиболее технологичны, дешифраторы обычно имеют инверсные выходы, т. е. активный низкий уровень выходного сигнала. В КМДП-сериях, где элементы ИЛИ–НЕ не менее технологичны, чем И–НЕ, дешифраторы чаще имеют прямые выходы. Стремление воз-

можно полнее использовать выводы типовых корпусов определяет размеры декодеров, выпускаемых в виде СИС. Обычно это 3–8, 4–10, сдвоенный 2–4, а также 4–16, но уже в корпусе с 24 выводами.

4.4. Шифраторы

Классический шифратор (*ENCODER – CD*) выполняет функцию обратную дешифратору – преобразует входной унарный код в выходной двоичный, т. е. при подаче активного сигнала только на один из выходов шифратора на его выходах появляется двоичный код, соответствующий номеру этого активного входа.

Полный шифратор имеет 2^n входов, на которые могут поступать единицы (только не одновременно), и n входов, на которых формируется двоичный код, соответствующий номеру активного входа (на котором формируется единица).

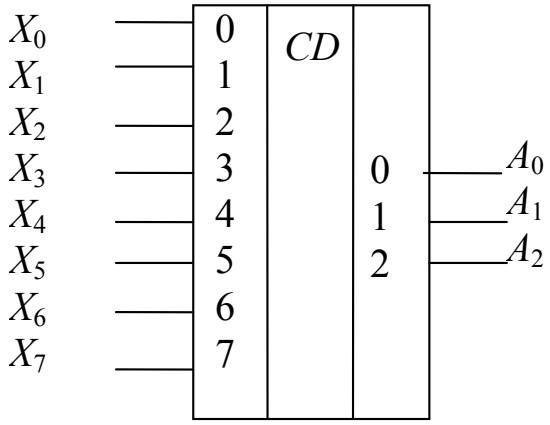


Рис. 4.26. Условно-графическое обозначение шифратора

Таблица 4.3

Таблица состояний дешифратора

№	$X_7\,X_6\,X_5\,X_4\,X_3\,X_2\,X_1\,X_0$	$A_2\,A_1\,A_0$
0	0 0 0 0 0 0 0 1	0 0 0
1	0 0 0 0 0 0 1 0	0 0 1
2	0 0 0 0 0 1 0 0	0 1 0
3	0 0 0 0 1 0 0 0	0 1 1
4	0 0 0 1 0 0 0 0	1 0 0
5	0 0 1 0 0 0 0 0	1 0 1
6	0 1 0 0 0 0 0 0	1 1 0
7	1 0 0 0 0 0 0 0	1 1 1

Булевы выражения выходных функций запишутся следующим образом:

$$A_0 = \overline{X}_0 X_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 \overline{X}_7 + \overline{X}_0 \overline{X}_1 \overline{X}_2 X_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 \overline{X}_7 +$$
$$+ \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 X_5 \overline{X}_6 \overline{X}_7 + \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 X_7;$$

$$A_1 = \overline{X}_0 \overline{X}_1 X_2 \overline{X}_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 \overline{X}_7 + \overline{X}_0 \overline{X}_1 \overline{X}_2 X_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 \overline{X}_7 +$$
$$+ \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 \overline{X}_5 X_6 \overline{X}_7 + \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 X_7;$$

$$A_2 = \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 X_4 \overline{X}_5 \overline{X}_6 \overline{X}_7 + \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 X_5 \overline{X}_6 \overline{X}_7 +$$
$$+ \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 X_5 X_6 \overline{X}_7 + \overline{X}_0 \overline{X}_1 \overline{X}_2 \overline{X}_3 \overline{X}_4 \overline{X}_5 \overline{X}_6 X_7.$$

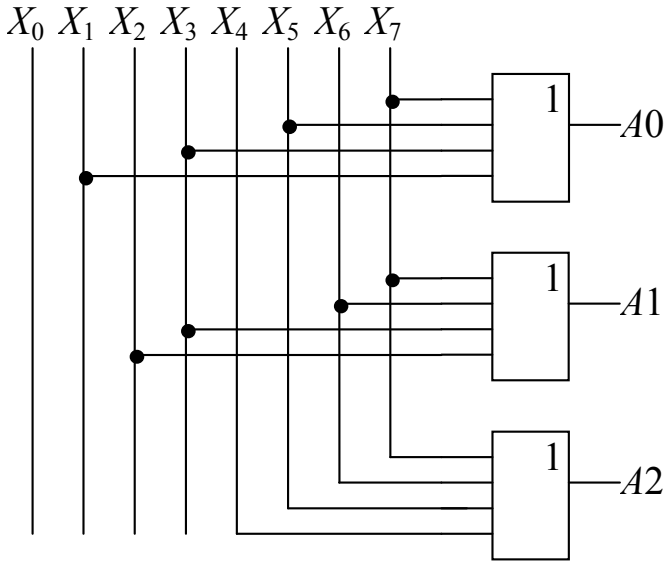


Рис. 4.27. Схема шифратора на логических элементах

Схемы шифратора должны состоять из трех логических элементов 4–ИЛИ, на каждый из входов которых поступает сигнал с ЛЭ 8–И. Поскольку в определении шифратора, в его таблице истинности заложено условие, что сигнал может подаваться только на один вход шифратора, т. е. что одновременно на несколько входов подаваться не должно, то ЛЭ 8–И можно исключить и схема шифратора будет состоять из трех ЛЭ 4–ИЛИ (рис. 4.27). Здесь под сигналом подразумевается логическая единица, так как ничего не сказано о типе логики, а следовательно логика подразумевается положительной.

Классический шифратор не выпускают в виде отдельных микросхем, однако он входит в состав программируемых логических матриц (ПЛМ), которые выпускают очень широко.

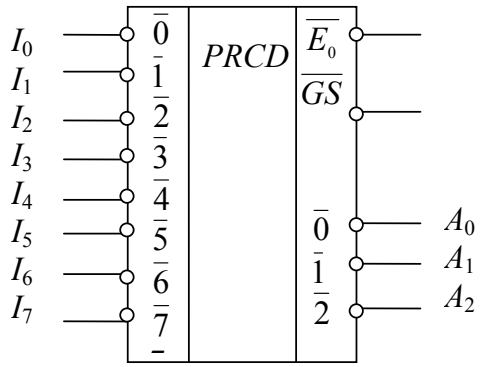


Рис. 4.28. Условно-графическое обозначение приоритетного шифратора

Выпускаются микросхемы *приоритетных шифраторов* (рис. 4.28), в которых условие влияния только одной выходной лог.1 на схему шифратора обеспечивается дополнительной схемой приоритета на его входе. В приоритетном шифраторе входной код может содержать сколько угодно единиц, но выходной код шифратора будет соответствовать номеру того входа, на который подается старшая единица. Это не обязательно будет старший разряд входного кода. Приоритет – преимущественное право.

На входе приоритетного шифратора стоит схема выделения старшей единицы, в которой все входные единицы, кроме старшей, заменяются нулями.

Пример: *PRCD* К155ИВ1. Здесь имеется выход $\overline{GS}$ – GROUP SIGNAL – групповой сигнал, на этом выходе активный уровень, в важном случае это низкий уровень, когда есть активный (низкий) уровень хотя бы на одном из разрядов входного кода шифратора.

Таблица 4.4

Таблица состояний приоритетного шифратора К155ИВ1

Входы									**Выходы**		
$\overline{EI}$	$\overline{I_0}$	$\overline{I_1}$	$\overline{I_2}$	$\overline{I_3}$	$\overline{I_4}$	$\overline{I_5}$	$\overline{I_6}$	$\overline{I_7}$	$\overline{A_2}\,\overline{A_1}\,\overline{A_0}$	$\overline{EO}$	$\overline{GS}$
1	*x*	*x*	*x*	*x*	*x*	*x*	*x*	*x*	1 1 1	1	1
0	1	1	1	1	1	1	1	1	1 1 1	0	1
0	*x*	*x*	*x*	*x*	*x*	*x*	*x*	0	0 0 0	1	0
0	*x*	*x*	*x*	*x*	*x*	*x*	0	1	0 1 0	1	0
0	*x*	*x*	*x*	*x*	*x*	0	1	1	0 0 1	1	0
0	*x*	*x*	*x*	*x*	0	1	1	1	0 1 1	1	0
0	*x*	*x*	*x*	0	1	1	1	1	1 0 0	1	0
0	*x*	*x*	0	1	1	1	1	1	1 0 1	1	0
0	*x*	0	1	1	1	1	1	1	0 1 1	1	0
0	0	1	1	1	1	1	1	1	1 1 1	1	0

Выходы $\overline{EO}$ и $\overline{GS}$ образуют тракт групповых сигналов и служат для наращивания разрядности.

Наращивание разрядности шифраторов основано на том обстоятельстве, что все цифры, начиная с восьми, имеют «1» в старшем разряде. Следовательно, выход *GS*-шифратора, образующего входы *I*8–*I*15, может быть использован для образования старшего разряда A_3. На рис. 4.29 показана схема получения 4-разрядного шифратора на микросхемах К555ИВ1.

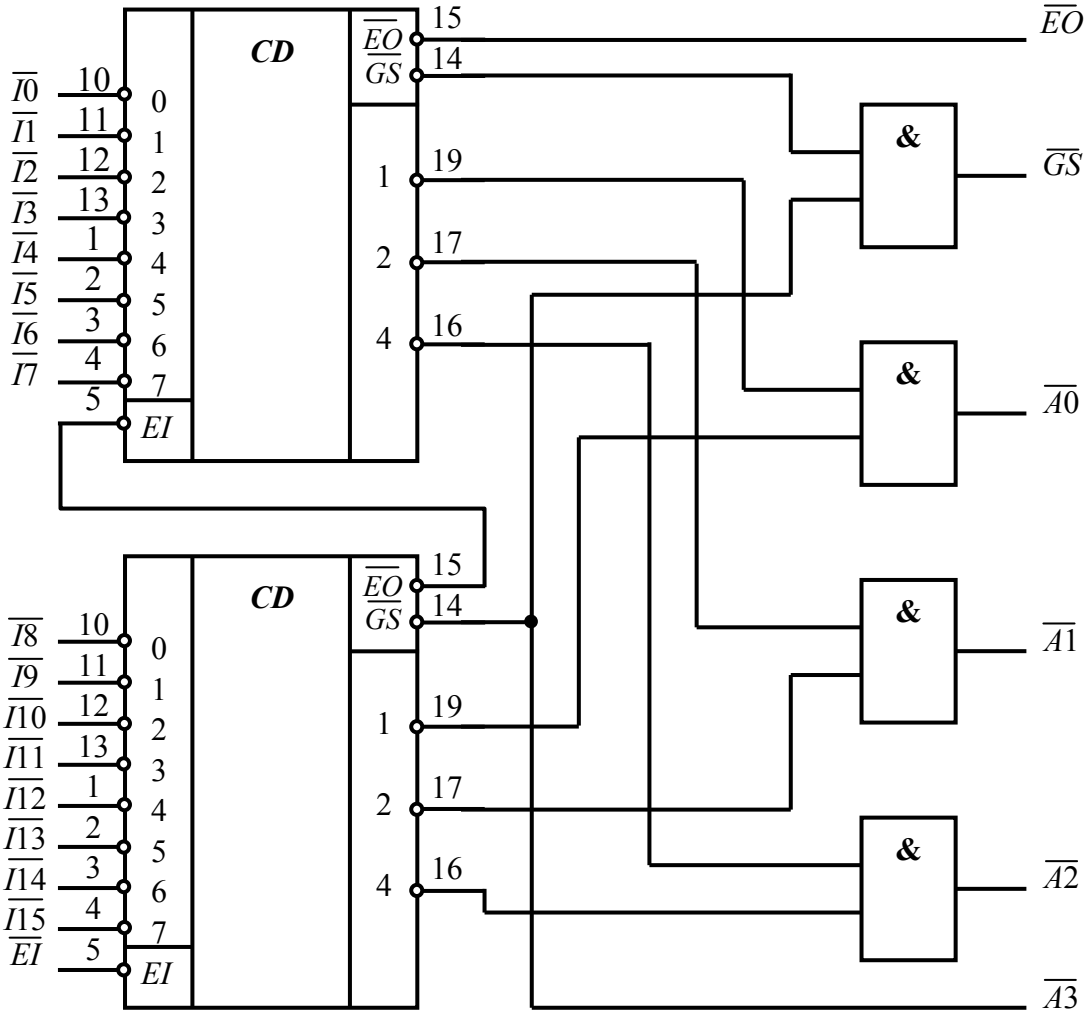


Рис. 4.29. Пример наращивания разрядности шифратора на микросхемах К555ИВ1

4.5. Мультиплексоры и демультиплексоры

Мультиплексор – это функциональное устройство, осуществляющее подключение (коммутацию) одного из нескольких входов данных к выходу. Номер выбранного входа соответствует коду, подаваемому на адресные входы мультиплексора. На рис. 4.30 приведен пример условного обозначения мультиплексора на схемах.

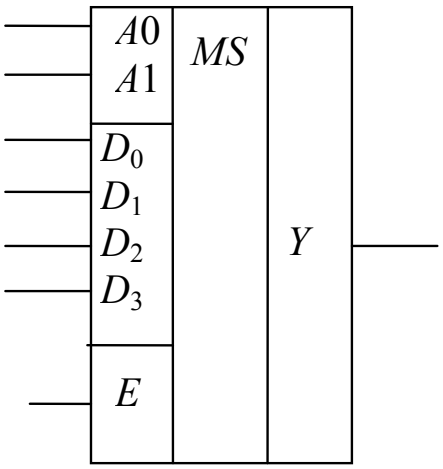


Рис. 4.30. Условное обозначение мультиплексора на схемах

Мультиплексированием (MULTIPLEX) называют передачу данных от нескольких источников по одному каналу поочередно.

Мультиплексор (*MS*) – устройство, осуществляющее подключение одного из входов данных к выходу в зависимости от комбинации на адресных входах.

В цифровой технике мультиплексор имеет m информационных входов данных D_0, D_1, D_2, D_3, …, D_m; n адресных входов A_1, A_2, A_3, …, A_n $m = 2^n$ и один выход данных.

Двоичный код на входах адреса определяет номер того входа данных, с которого информация проходит на выход мультиплексора в этот момент.

Обозначение *MS* отличается от *MUX*, которого требует ГОСТ, потому что в нем отражена еще одна функция – выборка, селекция данных из определенного, указанного адресом источника, которая обозначается *S* – SELECT – выбирать. Обозначение *MS* встречается чаще всего, но иногда бывает *MUX* и изредка *SL*.

Таблица истинности мультиплексора с двумя адресными входами приведена ниже.

Таблица 4.5

Таблица состояний мультиплексора

E	A_1	A_0	***Y***
1	0	0	*D*0
1	0	1	*D*1
1	1	0	*D*2
1	1	1	*D*3

В состав мультиплексора обязательно входит дешифратор адреса в том или ином виде. В первом варианте мультиплексора, приведенном на рис. 4.31, дешифратор адреса выделен в отдельный узел.

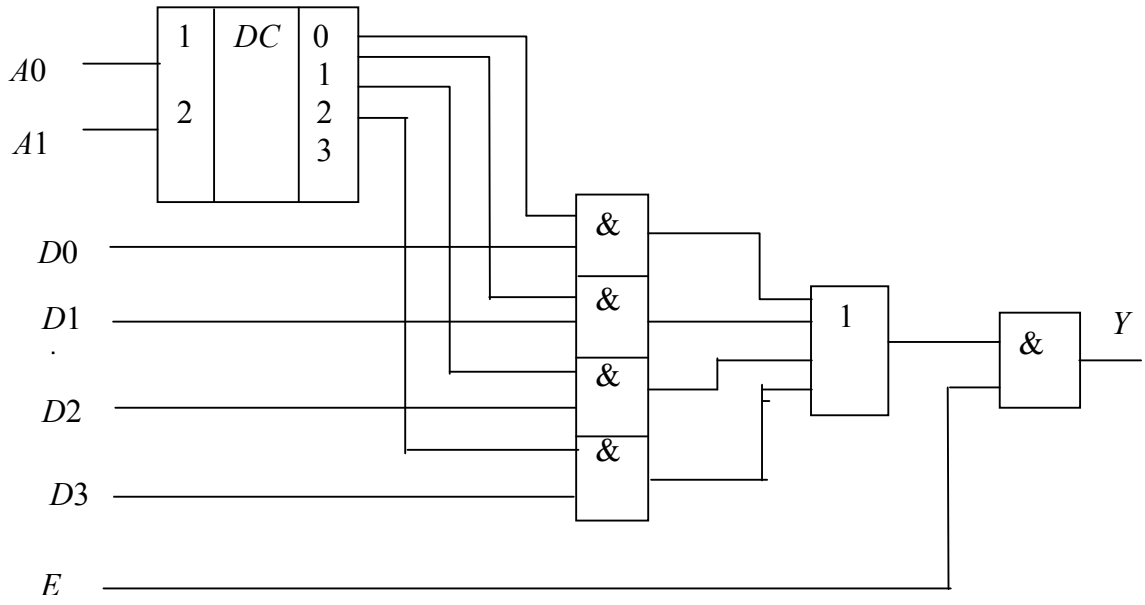


Рис. 4.31. Структурная схема мультиплексора,
построенная при помощи дешифратора адресных выходов

Как известно, линейный дешифратор состоит из группы инверторов по адресным входам и линейки – столбца – элементов И.

Столбец элементов И мультиплексора, приведенного на рис. 4.32, может выполнять функцию линейного дешифратора, если эти элементы И, но с большим числом входов, подключить к выходам инверторов по адресным линиям в соответствии с таблицей истинности дешифратора.

Разрешающий вход *E* может быть осуществлен за счет дополнительных входов конъюнкторов, включенных вместе, или за счет дополнительного конъюнктора на выходе мультиплексора. Первый способ сложнее, но обеспечивает меньшую задержку распространения сигнала.

Наличие разрешающего входа *E* позволяет увеличивать число информационных входов вдвое путем последовательного соединения разрешающих входов двух мультиплексоров (наращивание разрядности).

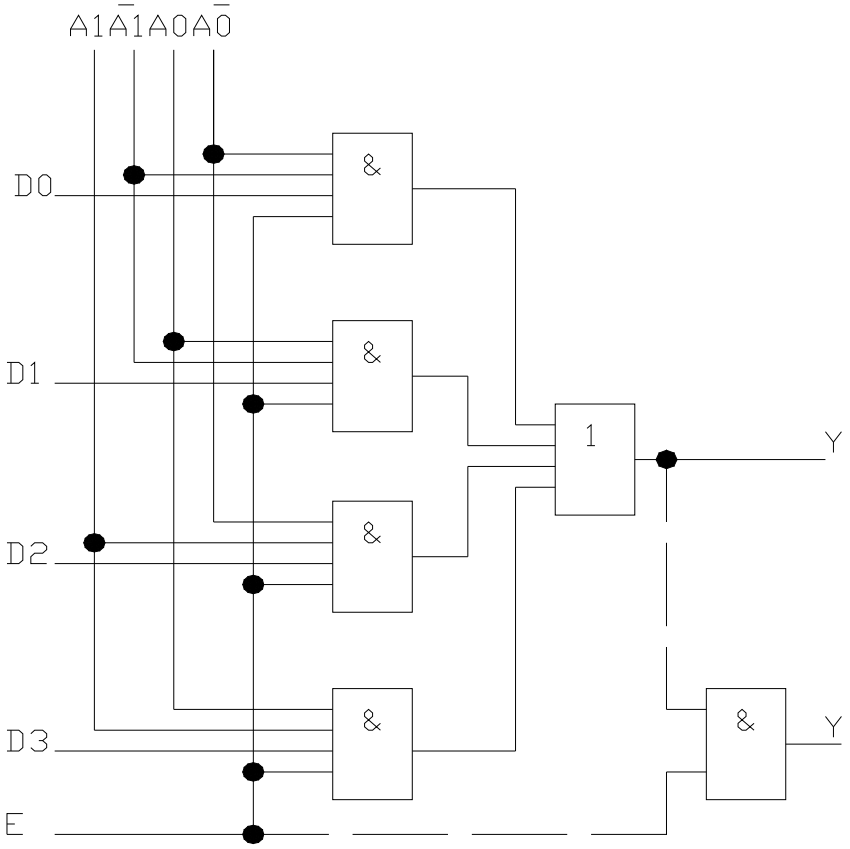


Рис. 4.32. Схема мультиплексора 4 : 1 в базисе И, ИЛИ, НЕ

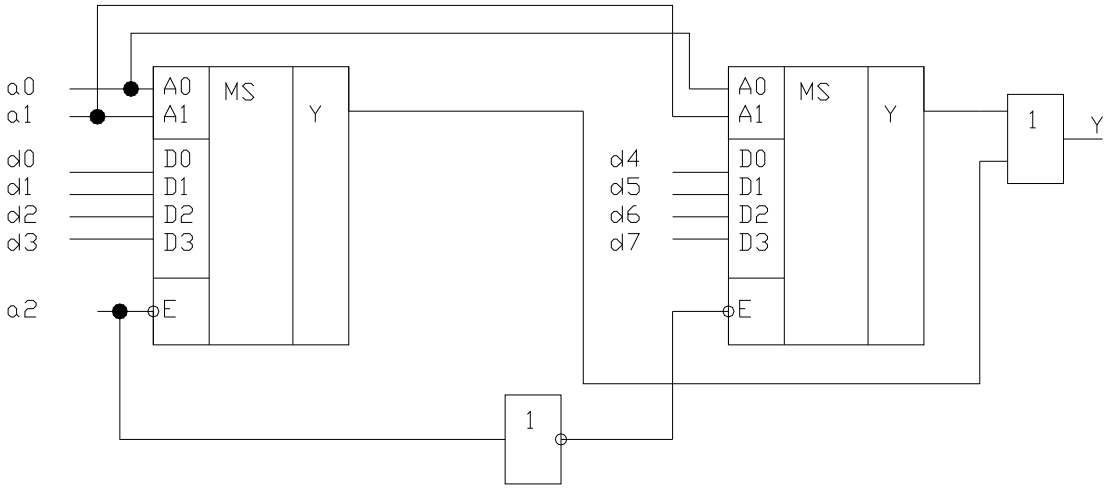


Рис. 4.33. Схема мультиплексора 8 : 1
с последовательным соединением разрешающих входов

Другой способ наращивания разрядности мультиплексора использует ступенчатый (пирамидальный) принцип построения.

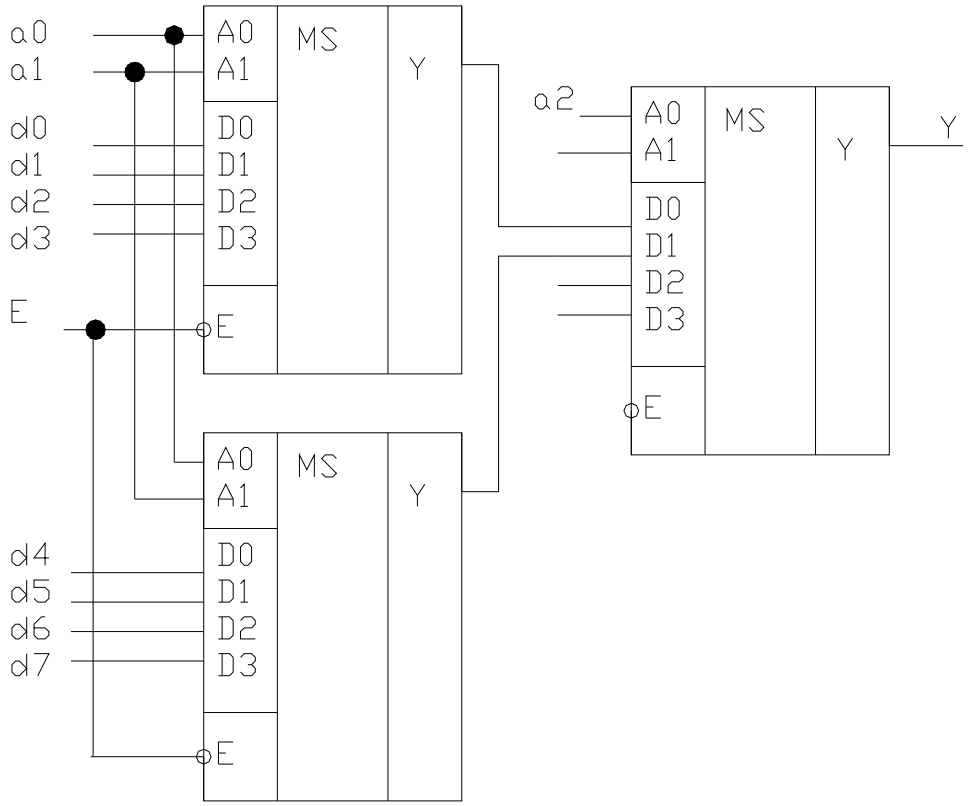


Рис. 4.34. Схема пирамидального мультиплексора 8 : 1

В пирамидальном мультиплексоре каждая ступень, начиная с первой, имеет больше входов, чем последующая. Применяются две, три и более ступени. Младшие разряды кода адреса подаются на адресные входы первой ступени, а последующим ступеням соответствуют старшие разряды адресного кода.

Кроме прямого назначения, мультиплексор может быть использован как универсальный логический элемент. Возможность реализации заданной логической функции основана на том, что на адресные входы мультиплексора можно подавать входные переменные, зная, какой выходной уровень должен отвечать каждой комбинации этих переменных. Предварительно установив на входах данных уровни напряжения «1» и «0» согласно таблице истинности, получим устройство, реализующее заданную логическую функцию.

X_0	X_1	F
0	0	0
0	1	1
1	0	1
1	1	0

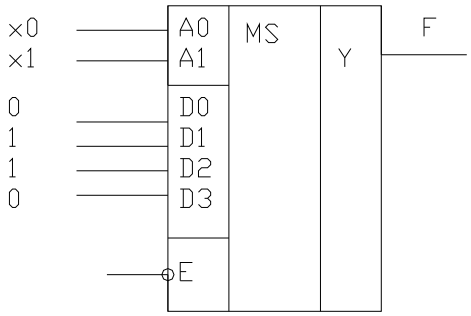


Рис. 4.35. Схема устройства, реализующего заданную логическую функцию согласно таблице истинности

Демультиплексоры в функциональном отношении противоположны мультиплексорам. Здесь сигналы с одного информационного входа распределяются в желаемой последовательности по нескольким выходам. Выбор нужной выходной шины, как и в мультиплексоре, обеспечивается кодом на адресных входах. При m адресных входах демультиплексор в зависимости от конструкции может иметь до 2^m выходов.

Идею работы демультиплексора поясняет рис. 4.36. Вход x – информационный, вход A – адресный, и сигнал на этом входе определяет, на каком из выходов будут формироваться сигналы, повторяющие x. Когда $A = 1$, верхний элемент И заперт и на его выходе $F_0 = 0$; нижний элемент, напротив, открыт и работает как повторитель информационных сигналов. При $A = 0$ заперт нижний элемент, а верхний пропускает входную информацию. Демультиплексоры с большим числом выходов работают по тому же принципу, только имеют более сложную схему.

Если у демультиплексора 1 : 4 на информационном входе поддерживать низкий (неактивный) уровень или на разрешающем поддерживать высокий (активный) уровень, то устройство будет работать как дешифратор 2 : 4. Таким образом, между демультиплексором и дешифратором нет принципиальной разницы, а различие сводится к виду сигналов на одиночном входе: если они меняются во времени – это демультиплексор, если нет – дешифратор. У дешифраторов этот вход нередко отсутствует и выходные сигналы на активном выходе имеют одно, наперед известное значение. Обозначается демультиплексор, как правило, символами *DMX*.

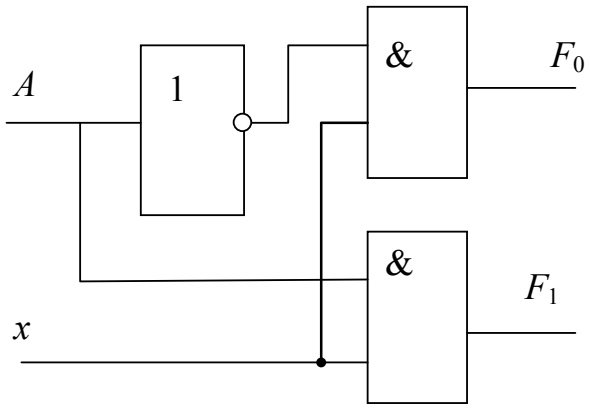


Рис. 4.36. Принцип построения демультиплексора

Демультиплексоры, как микросхемы средней степени интеграции, широко применяются в информационно-измерительной технике. Как и мультиплексоры, они часто используются в сочетании со счетчиками и регистрами. Они служат в качестве коммутаторов – распределителей информационных сигналов и синхроимпульсов, для демультиплексирования данных и организации адресной логики в постоянных запоминающих устройствах, а также для преобразователей двоично-десятичного кода в десятичный с целью управления индикаторными устройствами.

ГЛАВА 5. ФУНКЦИОНАЛЬНЫЕ УСТРОЙСТВА ПОСЛЕДОВАТЕЛЬНОСТНОГО ТИПА

Как уже упоминалось ранее, цифровые устройства делятся на *комбинационные* и *последовательностные*. Все *последовательностные* устройства обладают памятью. *Память* – свойство сохранять в течение неограниченного времени внутреннее состояние устройства. Работа последовательностных схем определяется как текущими входными состояниями, так и предыдущими, информация о которых хранится в памяти. Последовательностные схемы называются еще *конечными автоматами*.

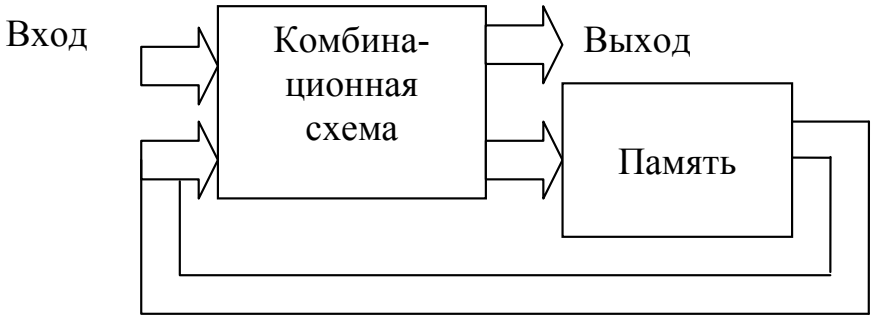


Рис. 5.1. Структура последовательностных схем

5.1. Триггеры

5.1.1. Классификация триггеров

Триггер – представляет собой логическую схему с положительной обратной связью, которая может неограниченно долго находиться в одном из двух устойчивых состояний (0 и 1), которые обеспечиваются положительной обратной связью, а не входным сигналом. Триггер скачком, лавинообразно, меняет одно состояние на другое под воздействием входного сигнала.

Триггеры делятся на *симметричные* и *несимметричные* (триггеры Шмитта). Триггеры Шмитта будут рассмотрены вместе с импульсными устройствами.

Симметричные триггеры подразделяются:

– по способу записи информации:

1) асинхронные;

2) синхронные – тактируемые;

– по способу управления записью информации:

1) со статическим управлением;

2) с динамическим управлением – по фронту (переднему или заднему);

3) с двухступенчатым управлением – двумя тактовыми импульсами или по двум фронтам;

– по способу организации логических связей:

1) элементарные, с раздельной установкой 0 и 1 – *RS*-триггеры;

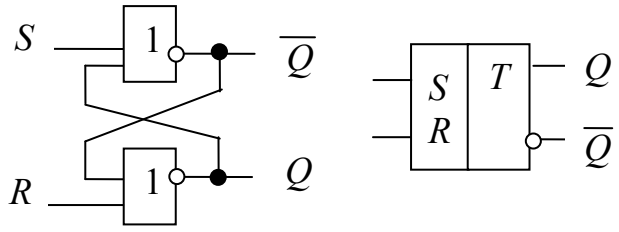
2) с приемом информации по одному входу – *D*-триггеры;

3) со счетным входом – *T*-триггеры;

4) универсальные с раздельной установкой 0 и 1 – *JK*-триггеры.

5.1.2. Асинхронный *RS*-триггер

Строится на элементах ИЛИ–НЕ. Вход S – (SET) – называют входом установки (устанавливает на выходе Q лог. 1); вход R – (RESET) – входом возврата или сброса (устанавливает на выходе Q лог. 0).



а) *б*)

Рис. 5.2. *RS*-триггер с прямыми входами: *а* – схемная реализация; *б* – условно-графическое обозначение

Первые две строки таблицы состояний (табл. 5.1), когда $S = 0$ и $R = 0$ – это *режим хранения*, при этом состоянии триггера не меняется $Q_{n\text{н}} = Q_n$.

Таблица 5.1

Таблица состояний *RS*-триггера

№	Такт t_n			Такт t_{n+1}
	S_n	R_n	Q_n	Q_{n+1}
1	0	0	0	0
2	0	0	1	1
3	0	1	0	0
4	0	1	1	0
5	1	0	0	1
6	1	0	1	1
7	1	1	0	н/0
8	1	1	1	н/0

Следующие две строки (3 и 4) – перевод триггера в 0-*сброс*, т. е. $Q_{n\text{н}} = 0$, независимо от состояния Q_n ($Q_n = 0$ или $Q_n = 1$), потому что $R = 1$, а $S = 0$.

Последующие две строки (5 и 6) – перевод триггера в 1 – *установка*, т. е. $Q_{n\text{н}} = 1$, независимо от предыдущего состояния Q_n ($Q_n = 0$ или $Q_n = 1$), так как $S = 1$, а $R = 0$.

Две последние строки таблицы называют *состоянием неопределенности* (н/о).

Если на *S*- и *R*- входы подать лог.1, то на обоих выходах Q и $\overline{Q}$ появятся нули. Если теперь одновременно снять лог.1 со входов *R* и *S*, то оба элемента ИЛИ–НЕ начнут переключаться в единичное состояние, каждый стремясь при этом оставить своего «партнера» в нуле. Какой партнер одержит в этом «поединке» победу, будет зависеть от их коэффициентов усиления, скорости переходных процессов и ряда других неизвестных заранее факторов. Для разработчика схемы результирующее состояние триггера оказывается неопределенным, поэтому комбинация $R = 1$, $S = 1$ считается *запрещенной* и в обычных условиях ее не используют. В некоторых справочниках эту комбинацию даже называют *неустойчивой*, хотя пока она держится на входах, схема вполне устойчива. Комбинацию $R = 1$, $S = 1$ допускается применять только когда обеспечено не одновременно, а строго определенное снятие лог.1 с *R*- и *S*-входов.

	$\overline{S}_n\overline{R}_n$	$\overline{S}_n R_n$	$S_n R_n$	$S_n\overline{R}_n$
$\overline{Q}_n$			н/0	1
Q_n	1		н/0	1

Рис. 5.3. Карта Карно, построенная по таблице состояний

По таблице состояний триггера, так же как и по таблице истинности комбинационных схем, можно построить карту Карно. В двух последних строчках таблицы состояний отражена неопределенность состояния триггера не во время подачи лог.1 на входы *R*- и *S*-, а после одновременного снятия лог.1 с этих входов. Пока же на входы *R*- и *S*- подана лог.1, состояние выходов определено. Предположим, что в этом состоянии $\overline{Q} = 0$, $Q = 0$, тогда карта Карно примет вид, представленный на рис. 5.4.

	$\overline{S}_n\overline{R}_n$	$\overline{S}_nR_n$	S_nR_n	$S_n\overline{R}_n$
$\overline{Q}_n$				1
Q_n	1			1

Рис. 5.4. Карта Карно, построенная при условии $\overline{Q}=0$, Q = 0 при R = 1, S = 1

После минимизации из карты Карно запишем:

$$Q_{n+1}=S_n\overline{R}_n+Q_n\overline{R}_n=(S_n+Q_n)\overline{R}_n=\overline{\overline{R}_n+\overline{S_n+Q_n}}=\overline{R_n+\overline{S_n+Q_n}}.$$

Схема, соответствующая этому булеву выражению, приведена на рис. 5.2, *а*.

Если в карте Карно вместо состояния неопределенности $Q_{n\text{н}}=1$ и $\overline{Q}_{n\text{н}}=1,$ то получим карту Карно, изображенную на рис. 5.5.

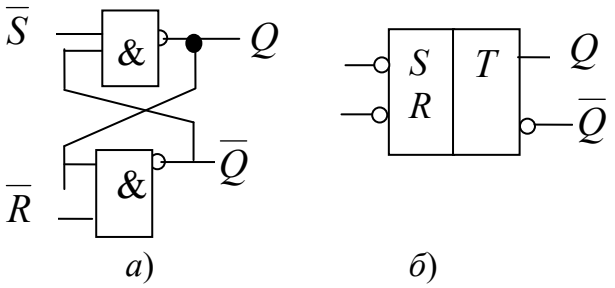
	$\overline{S}_n\overline{R}_n$	$\overline{S}_nR_n$	S_nR_n	$S_n\overline{R}_n$
$\overline{Q}_n$			1	1
Q_n	1		1	1

Рис. 5.5. Карта Карно, построенная при условии $\overline{Q}=1$, Q = 1 при R = 1, S = 1

После минимизации получим выражение:

$$Q_{n+1}=S+\overline{R}Q=\overline{\overline{S}\cdot\overline{\overline{R}\cdot Q}}.$$

Схема, построенная по этому выражению, является базовой для *RS*-триггеров в ТТЛ и приведена на рис. 5.6, *а*.



а) *б*)

Рис. 5.6. RS-триггер с инверсными входами:
а – схемная реализация; *б* – условно-графическое обозначение

5.1.3. Асинхронный двухступенчатый *JK*-триггер

В отличие от *RS*-триггеров *JK*-триггеры не боятся, когда на оба их входа поступает лог.1 – они при этом переключаются в состояние, противоположное текущему.

В данном *JK*-триггере (рис. 5.7) изменение состояния происходит с задержкой, равной длительности входных сигналов, вызывающих переход в новое состояние. Роль задержки выполняет второй *RS*-триггер, который в отсутствии входных лог.1 копирует, повторяет состояние первого *RS*-триггера, а в присутствии хотя бы одной лог.1 на входах он «отключается» от первого *RS*-триггера и сохраняет свое состояние.

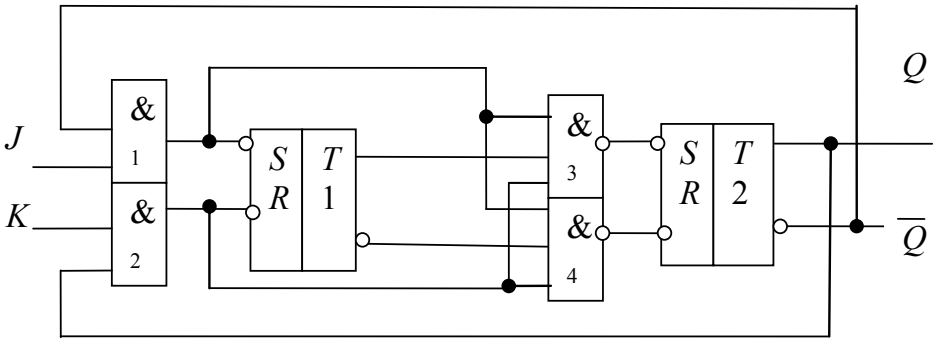


Рис. 5.7. Структура асинхронного двухступенчатого *JK*-триггера

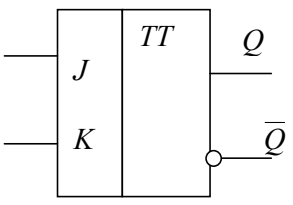
Для того чтобы подчеркнуть, что в данный *JK*-триггер входит два *RS*-триггера, в условно-графическом обозначении *JK*-триггера вводятся две буквы: *TT*. Булево выражение для данного триггера имеет следующий вид:

$$Q_{n+1} = J_n \overline{Q}_n + \overline{K}_n Q_n .$$

Карта Карно, построенная по данному булеву выражению, и условно-графическое обозначение *JK*-триггера представлены на рис. 5.8.

	$\overline{J}\,\overline{K}$	$\overline{J}K$	JK	$J\overline{K}$
$\overline{Q}$			1	1
Q	1			1

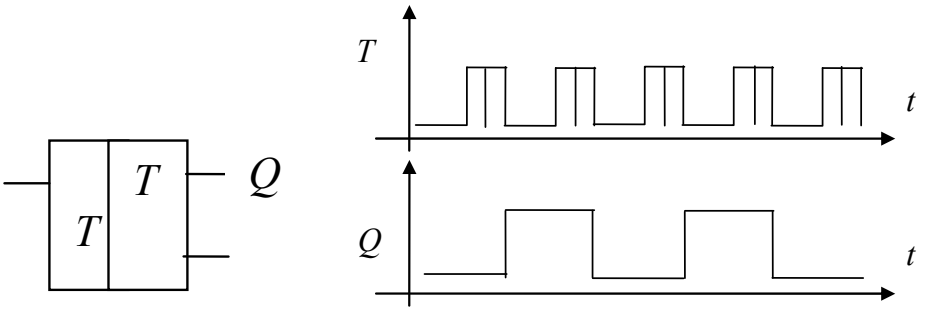
а)



б)

Рис. 5.8. JK-триггер: *а* – карта Карно; *б* – условно-графическое обозначение

Если у *JK*-триггера объединить входы *J* и *K*, то получится *T*-триггер, условно-графическое обозначение которого, а также временные диаграммы, поясняющие его работу, представлены на рис. 5.9.



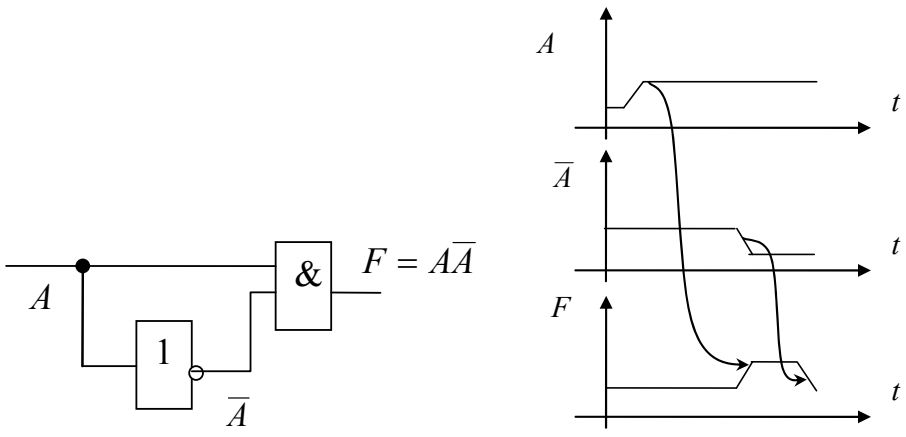
а) *б*)

Рис. 5.9. T-триггер: *а* – условно-графическое обозначение; *б* – временные диаграммы

5.1.4. Переходные процессы. Гонки (RACES)

Гонки (*состязания фронтов*) бывают на участках схем, где сигнал разветвляется и получившееся два сигнала распространяются по двум независимым цепочкам элементов, а затем оба сигнала «встречаются» на входах одного элемента. Из-за различия сумм задержек в этих цепочках на выходе схемы могут появляться импульсы помехи фронта.

На рис. 5.10 представлена цепочка из двух логических элементов и временные диаграммы ее работы.



а) *б*)

Рис. 5.10. Схема на логических элементах: *а* – реализующая функцию $F = A\overline{A}$; *б* – временные диаграммы ее работы

Задержка схемы складывается из задержек срабатывания логических элементов и задержек распределения сигнала по дорожкам

платы. Последние пока учитывают только для схем на ЭСЛ. В некоторых схемах не учет задержек приводит к неправильной работе этой схемы, с учетом задержек обнаруживаем на выходе импульс, помехи с длительностью, близкой к $t_{зд.р}$ инвертора. Линии со стрелками показывают причинно-следственные связи в цепочке переключений.

На заводе-изготовителе микросхем есть выходной контроль продукции, где отбраковывают микросхемы, у которых задержка распространения превышает допуск. Можно предположить, что микросхемы с задержкой, меньшей в 8 раз чем максимально допустимая, никогда не встретятся, но заводы-изготовители такую гарантию не дают. Поэтому обычно считают, что задержка ЛЭ может быть от нуля до $t_{зд.р.max}$.

На рис. 5.11 представлены временные диаграммы переключения логического элемента И с различным временем распространения сигнала по разным входам. Здесь X – переходный процесс, или время неопределенности.

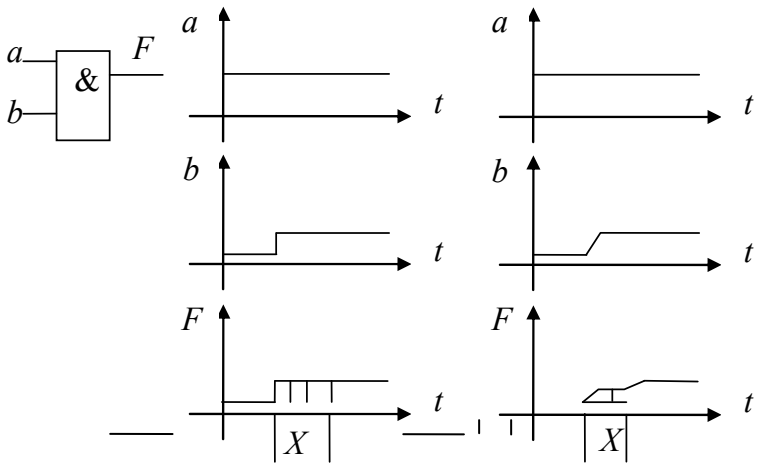


Рис. 5.11. Временные диаграммы работы логического элемента И

Существует три способа *борьбы с гонками*:

1. *Построение противогоночных схем*, например, в месте встречи фронтов от двух цепочек ставят логический элемент ИЛИ, через который просто раньше проходит тот фронт, что пришел первым, т. е. помеха не возникает.

2. *Учет минимального времени задержки*, которое нигде и не оговорено; однако если одна цепочка состоит из числа N логических элементов, которое больше числа M элементов другой цепочки, соблюдается условие $N \geq M$.

3. *Введение синхронизации* (*тактирования*) – является самым универсальным. Поэтому большинство микросхем последовательных устройств делают синхронными.

5.1.5. Синхронный *RS*-триггер

Все синхронные триггеры имеют один или два асинхронных входа *R*- и *S*-, прямые или инверсные, причем эти *RS*-асинхронные входы имеют наивысший приоритет в своем воздействии на триггер по отношению ко всем основным входам триггера.

На рис. 5.12 представлена структурная схема синхронного *RS*-триггера и временные диаграммы его работы.

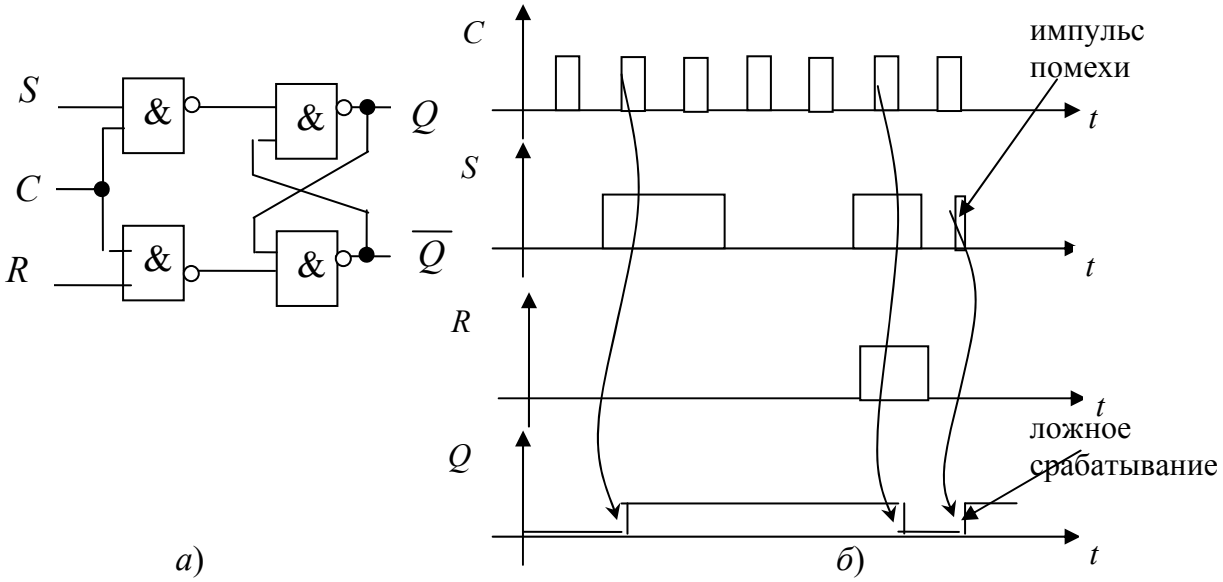
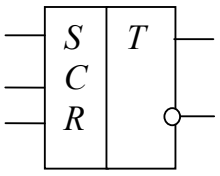


Рис. 5.12. Синхронный *RS*-триггер: *а* – структурная схема; *б* – временные диаграммы его работы

Сигналы на выходах *R*-, *S*- являются разрешающими, подготовительными, а сигнал на входе *C* – *исполнительный*, по которому происходит переключение триггера. *C* – от английского *CLOCK* – часы; иногда этот вход может быть обозначен как *L* – *LOAD* – загрузка, иногда – *H*-HOLD – захват, удержание (рис. 5.13).



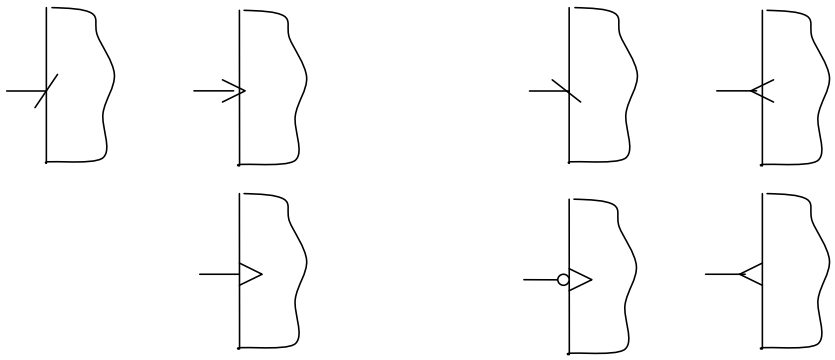
а)

C_n	S_n	R_n	Q_{n+1}	
0	*X*	*X*	Q_n	Режим хранения
1	0	0	Q_n	
1	0	1	0	
1	1	0	1	
1	1	1	н/0	

б)

Рис. 5.13. Синхронный *RS*-триггер: *а* – условно-графическое изображение; *б* – его таблица состояний

На рис. 5.14 представлены возможные обозначения входов синхронизации.



а) *б*)

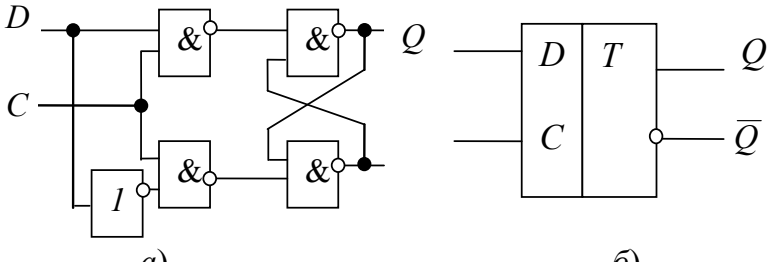
Рис. 5.14. Обозначения входов синхронизации: *а* – прямой вход – срабатывание по фронту C^{01}; *б* – обратный вход – срабатывание по срезу импульса C^{10}

5.1.6. Синхронный *D*-триггер со статическим управлением

Триггеры со статическим управлением, т. е. *прозрачные*, обычно называют по-англ. LATH – защелка.

Триггеры с динамическим управлением (срабатывание по фронту, по срезу), или двухступенчатые, т. е. непрозрачные, называют обычно FLIP – FLOP (щелчок-хлопок).

Выход Q синхронного D-триггера (рис. 5.15) по тактовому сигналу на C-входе принимает то же состояние, что и на D-входе (D – от английского DATA – данные, иногда DELAY – задержка, так как сигнал задерживается на выходе Q до прихода следующего импульса на синхровход C). D-триггер получается из синхронного RS-триггера путем объединения через инвертор R- и S-входов.



а) *б*)

Рис. 5.15. D-триггер: *а* – структурная схема; *б* – условно-графическое обозначение

При $C = 0$ триггер находится в режиме хранения, т. е. изменения сигнала на D-входе не влияют на Q-выход триггера. При $C = 1$ этот

D-триггер прозрачен для сигнала на *D*-входе: этот сигнал сразу же появляется на *Q*-выходе, как только появится на *D*-входе.

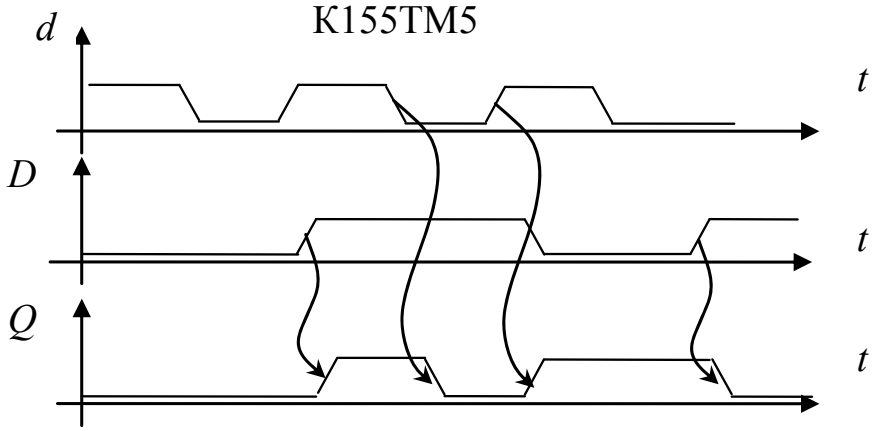


Рис. 5.16. Временные диаграммы работы *D*-триггера на примере К155ТМ7

Прозрачность триггера может привести к срабатыванию его от помех, как, например, в предыдущем разделе, в случае, когда прозрачный *RS*-триггер сработал от импульса помехи (рис. 5.12, диаграмма *б*). Минимизированная для ТТЛ схема *D*-триггера имеет некоторую склонность к ложным срабатываниям, если фронт сигнала на синхровходе C^{1-0} слишком пологий, а порог срабатывания логического элемента *D*1 выше, чем порог срабатывания логического элемента *D*2. Такая ситуация называется *гонками по входу* (рис. 5.17).

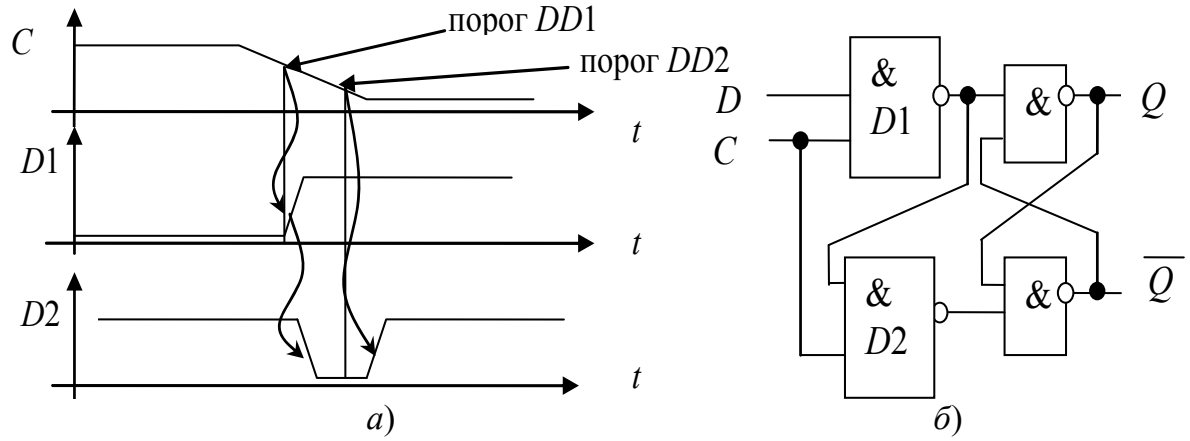


Рис. 5.17. Гонки по входу: *а* – временные диаграммы; *б* – структурная схема

5.1.7. Синхронный двухступенчатый *JK*-триггер

JK-триггеры относятся к универсальным устройствам. Их универсальность имеет двойственный характер. Во-первых, эти триггеры

с равным успехом могут быть использованы в различных цифровых устройствах: регистрах, цифровых счетчиках, делителях частоты и т. д., во-вторых, путем определенного соединения выводов они легко обращаются в триггеры других типов. Структурная схема *JK*-триггера с динамическим управлением приведена на рис. 5.18.

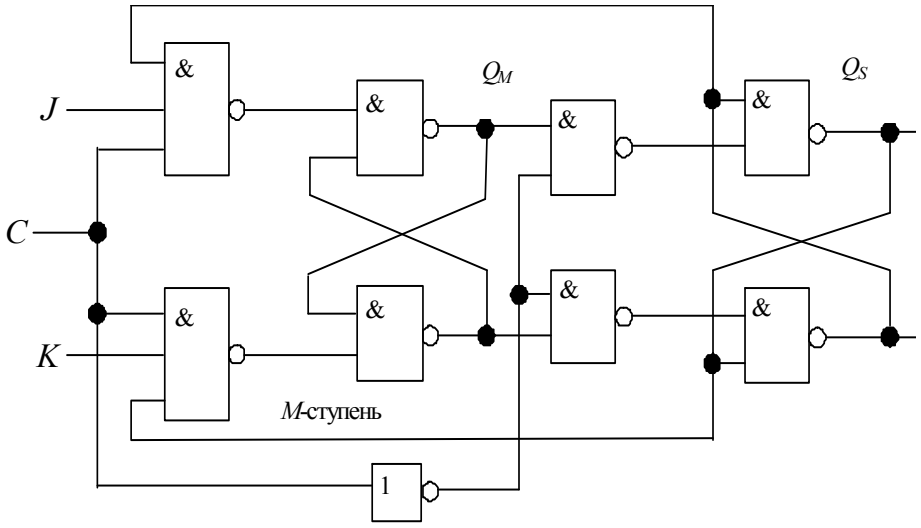


Рис. 5.18. Синхронный двухступенчатый *JK*-триггер

В двухступенчатом триггере каждый асинхронный вход *R*- или *S*- воздействует сразу на триггеры обеих ступеней, а также и на одну из входных схем и исключает возможность противодействия с их стороны сигналам с входов *R*- и *S*-.

На рис. 5.19 приведены временные диаграммы работы синхронного двухступенчатого *JK*-триггера.

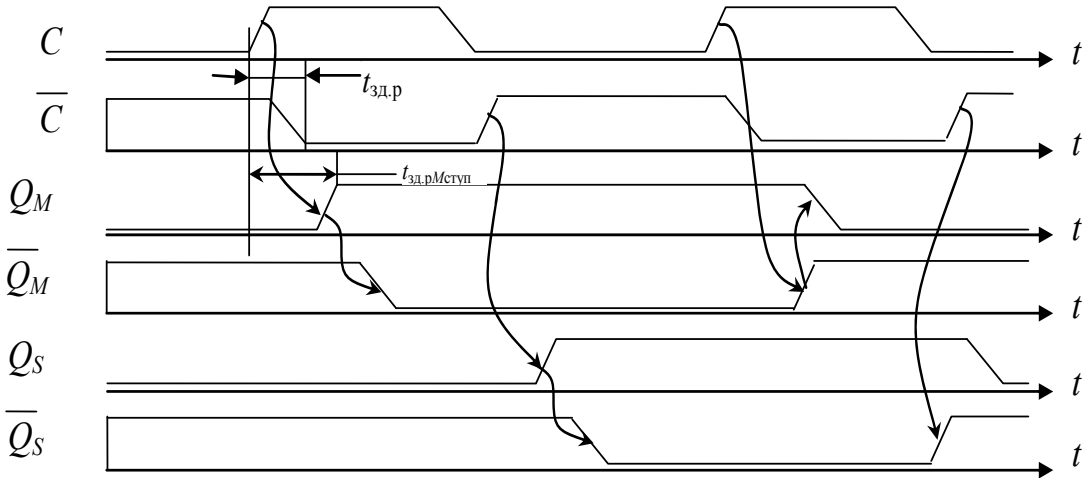


Рис. 5.19. Временные диаграммы работы синхронного двухступенчатого *JK*-триггера

1. При $J = 0$ и $K = 0$ сигнал лог.1 на C-входе не может открыть входные логические элементы И–НЕ, т. е. триггер не меняет состояния – осуществляется режим хранения.

2. При $J = 1$, $K = 0$, первый RS-триггер M-ступени переключается в единицу ($Q_M = 1$) при поступлении на C-вход лог.1 (фронт C^{01}), а второй RS-триггер S-ступени переключается в 1 ($Q_S = 1$) при фронт C^{10} [режим установки].

3. При $J = 0$, $K = 1$ первый RS-триггер M-ступени переключается в ноль ($Q_M = 0$) по C^{01}, на выходе S-ступени: $Q_S = 0$, по спаду C^{10} [сброс].

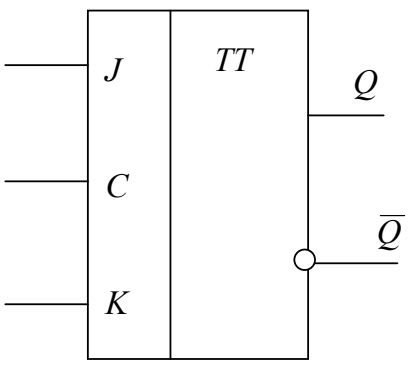
4. При $J = K = 1$ сигналы с выходов Q_S и $\overline{Q_S}$ открывают для сигнала на C-входе именно тот логический элемент И–НЕ, пройдя через который триггер переключается в противоположное положение в два этапа: сначала при C^{01} переключается Q_M, а затем по C^{10} – Q_S [счетный режим, или T-режим].

Триггер непрозрачен, т. е. сигналы, поступающие на входы J и K, не проходят сразу же на выход Q_S-триггера, как это происходит в синхронных RS- и D-триггерах, рассмотренных ранее.

В этом триггере видны параллельные пути распространения фронтов сигнала со входа C, которые являются причиной гонок. Чтобы микросхемы таких JK-триггеров не давали сбоев, в них опасность гонок ликвидируется за счет нормирования минимальной задержки M-ступени, или специально введенной гарантированной разницы в порогах срабатывания инвертора и входных схем И ступени M. Делают так, чтобы инвертор переключался при более низком уровне синхросигнала, чем входные схемы И, и тогда нужная последовательность срабатывания инвертора и M-ступени обеспечивается за счет конечной длительности фронтов C-сигнала. Инвертор реагирует на фронт C^{01} сигнала раньше, а на его спад C^{10} позже, чем это делает M-ступень.

C	C	J	K	Q_{n+1}
x	x	0	0	Q_n
↓	⌐	0	1	0
↓	⌐	1	0	1
↓	⌐	1	1	$\overline{Q}_n$

а)



б)

Рис. 5.20. Синхронный двухступенчатый JK-триггер:
а – таблица переключений; *б* – условно-графическое обозначение

5.1.8. Шестиэлементный триггер

Это самый распространенный *D*-триггер ТТЛ. Другие его названия – триггер Вебба, триггер с самоблокировкой, схема трех триггеров. Является непрозрачным. Переключение выхода *Q*-триггера происходит по фронту C^{01}. Схема противогоночная и допускает любые сочетания задержек своих элементов, так как в ней отсутствуют параллельные пути для гонок.

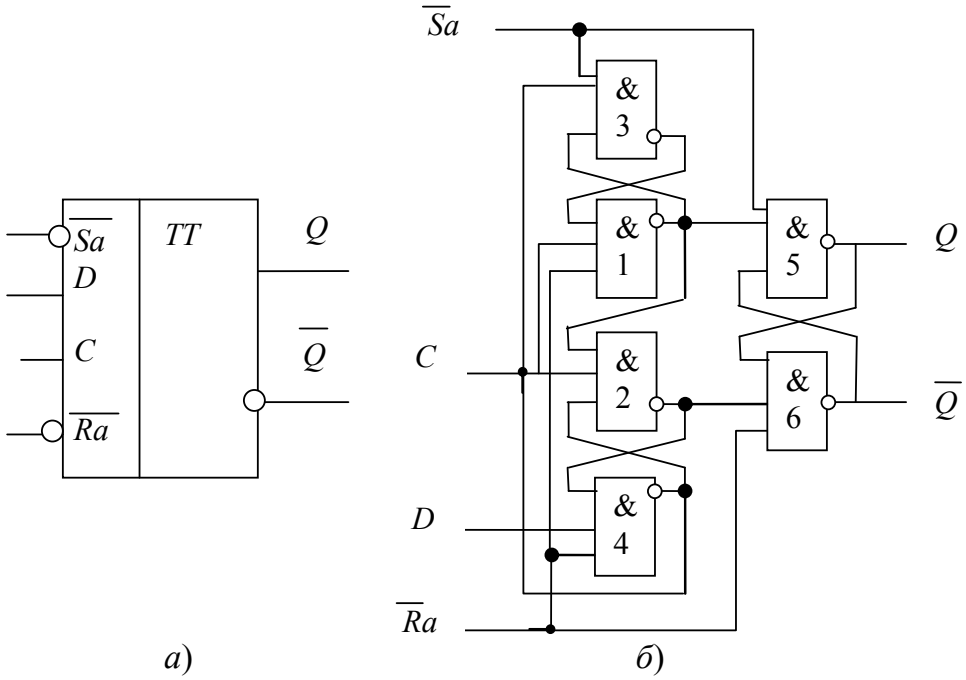


Рис. 5.21. Шестиэлементный *D*-триггер:
а – условно-графическое изображение; *б* – структурная схема

Назначение логических элементов (ЛЭ):

ЛЭ 5,6 – выходной *RS*-триггер с инверсными входами;

ЛЭ 1,2 – входные схемы И–НЕ *RS*-триггера;

ЛЭ 3,4 – вспомогательные логические элементы И–НЕ, они подготавливают один из выходных логических элементов И–НЕ для открытия его очередным фронтом C^{01}.

Вначале, пока на синхровходе нет сигнала $C = 0$, триггер хранит предыдущее состояние, например $Q = 0$ (рис. 5.22). Изменение сигнала на *D*-входе меняет лишь состояния логических элементов 3 и 4, но не влияет на выходной *RS*-триггер на ЛЭ 5,6.

По первому фронту C^{01} открывается ЛЭ 1 и, так как $D = 1$, при этом он подключает выходной *RS*-триггер в единичное состояние ($Q = 1$), т. е. происходит установка триггера.

По второму C^{01} фронту открывается ЛЭ 2 и, так как $D = 0$, при этом и триггер переключается в ноль ($Q = 0$), т. е. происходит сброс триггера.

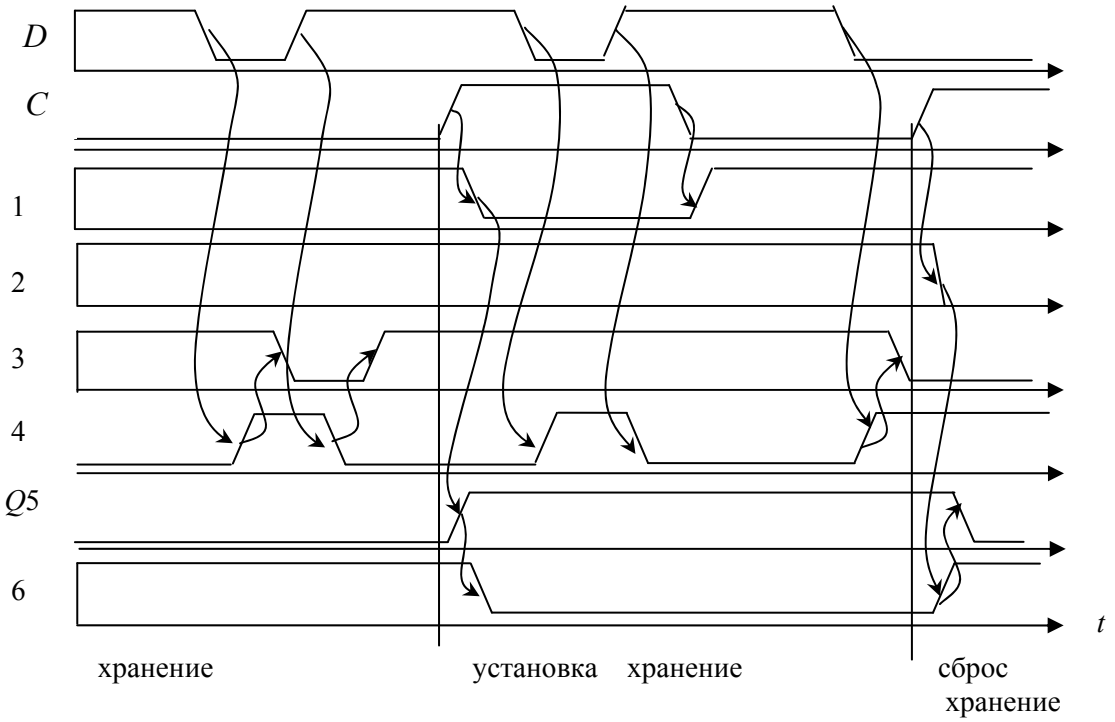


Рис. 5.22. Временные диаграммы работы шестиэлементного *D*-триггера

Изменение состояния на *D*-входе, пока на синхровходе сохраняется лог.1 ($C = 1$), не влияет на общее состояние триггера, так же как и при $C = 0$. Следовательно, триггер переключается только фронтом C^{01}.

5.1.9. Схемы взаимного преобразования триггеров

На рис. 5.23 представлены способы взаимного преобразования триггеров.

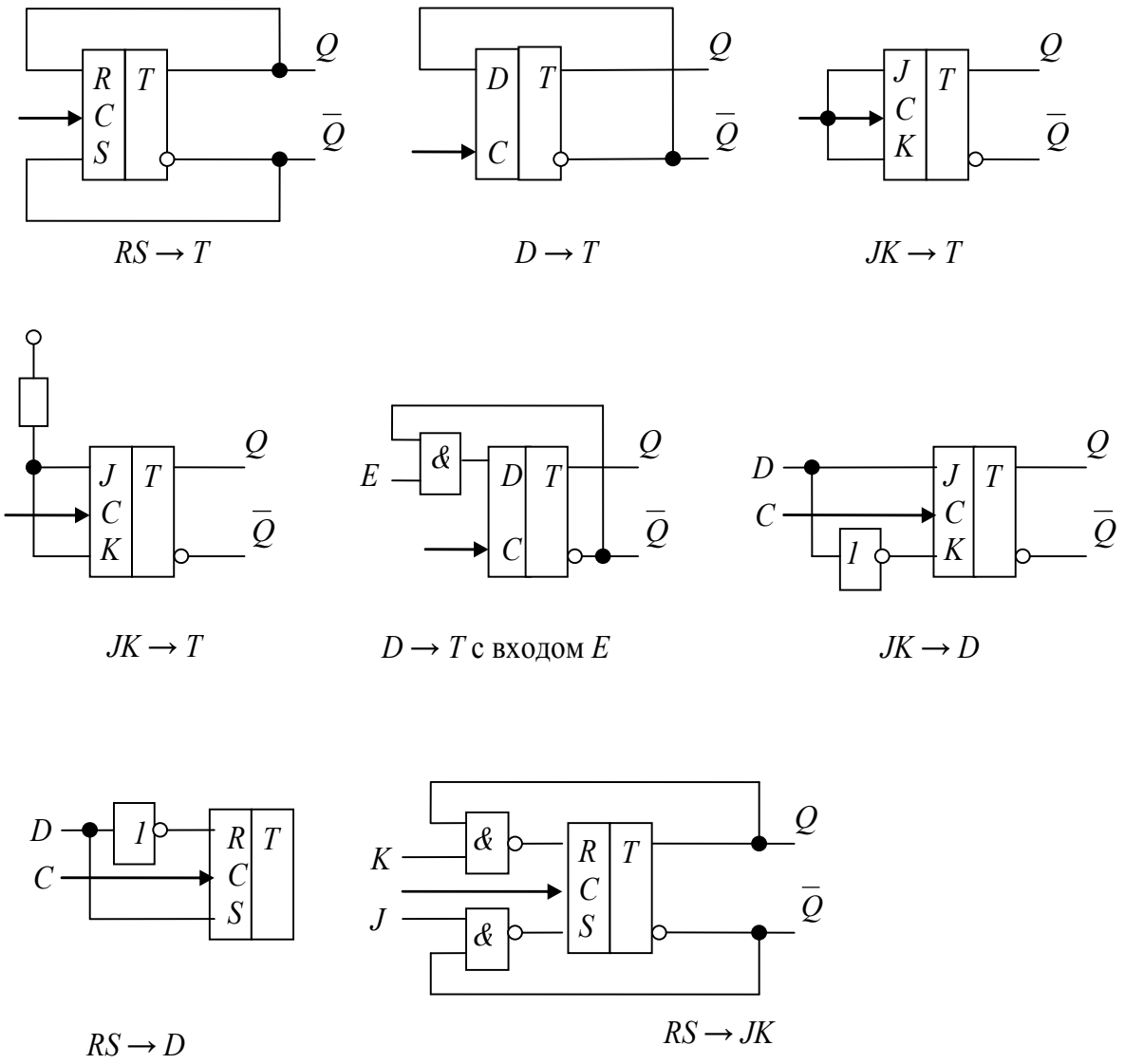


Рис. 5.23. Способы взаимного преобразования триггеров

5.2. Регистры

5.2.1. Регистры памяти

Для запоминания многоразрядных двоичных слов необходимое число *D*-триггеров объединяют вместе и рассматривают их как единый функциональный узел – *регистр памяти*. Здесь каждый триггер хранит один разряд двоичного числа (слова).

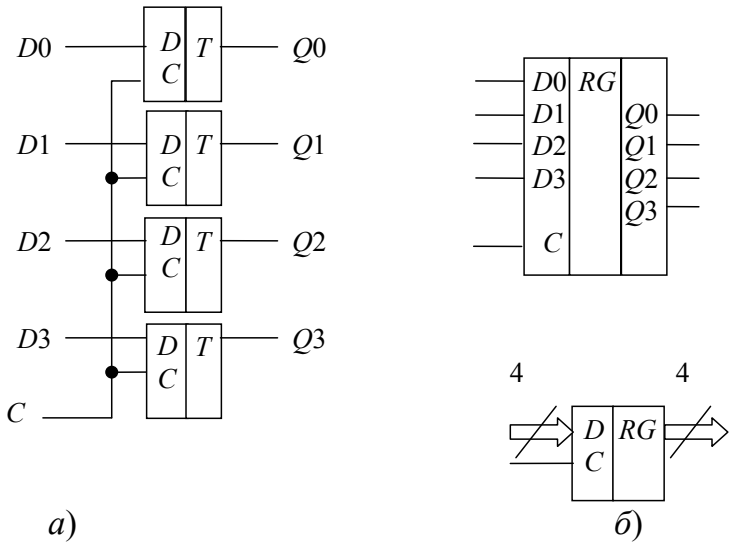


Рис. 5.24. Регистр: *а* – функциональная схема; *б* – его условно-графическое изображение

Назначение регистров памяти – хранить двоичную информацию. Эти регистры представляют собой набор синхронных триггеров, каждый из которых хранит один разряд двоичного числа. Ввод (запись, загрузка) и вывод (считывание) информации производится одновременно во всех разрядах параллельным кодом.

Запись обеспечивается тактовым импульсом. С приходом очередного тактового импульса происходит обновление записанной информации. Функциональная схема регистра и его условно-графическое изображение приведены на рис. 5.24.

На рис. 5.25 приведена функциональная схема микросхемы К155ТМ8, которая содержит четыре *D*-триггера. Микросхема имеет 4 раздельных информационных входа, прямые и инверсные выходы Q и $\overline{Q}$ от каждого триггера и общие для всех триггеров выходы синхронизации и установки нулей (сброса). Запись данных, поступивших на информационные входы, происходит одновременно во всех триггерах по фронту 01 сигнала на входе C, при этом на входе R должен быть высокий уровень (лог.1).

Установка триггеров в нулевое состояние выполняется сигналом НИЗКОГО уровня (лог.0) на входе R. При использовании микросхемы К155ТМ8 в качестве четырехразрядного регистра памяти она изображается как на рис. 5.25.

5.2.2. Регистры сдвига

Кроме операции хранения данных регистры могут использоваться и для операции сдвига данных с целью преобразования параллельного кода в последовательный и наоборот. SHIFT REGISTER – *сдвиговый регистр*, получают путем цепочечного соединения триггеров.

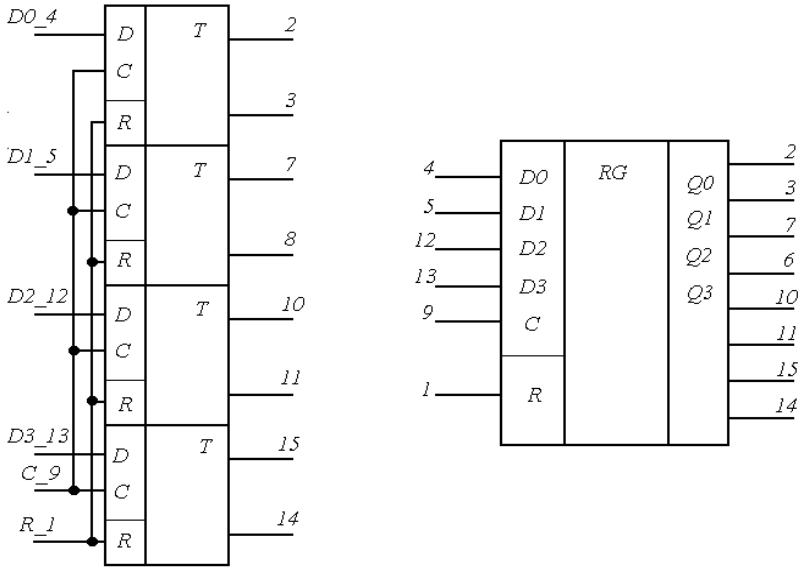
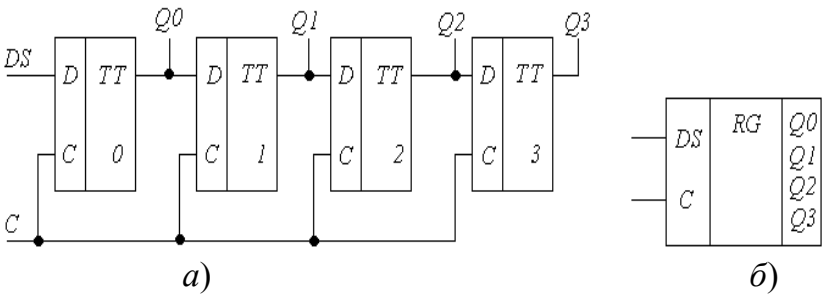


Рис. 5.25. Функциональная схема микросхемы К155ТМ8 и ее УГО

Сдвиг состоит в том, что по сигналу синхроимпульс происходит одновременно перезапись содержимого каждого триггера в соседний триггер. При этом не меняется само двоичное слово (число), записанное в регистре, оно лишь сдвигается вправо на один разряд, и только содержимое последнего триггера *TT*3 пропадает из регистра, а на вход первого *TT*0 поступает новый бит.

Сдвиговые регистры, или *регистры сдвига*, которые получаются путем последовательного соединения непрозрачных *D*-триггеров, предназначены для сдвига данных с целью преобразования двоичного последовательного кода в параллельный и наоборот. Суть сдвига состоит в том, что по сигналу синхроимпульса происходит одновременная перезапись содержимого каждого триггера в соседний триггер. При этом не меняется само двоичное слово (число), записанное в регистре, оно лишь сдвигается на один разряд, и только содержимое последнего триггера *TT*3 пропадает из регистра, а на вход первого *TT*0 поступает новый бит (рис. 5.26).



а) *б*)

Рис. 5.26. Регистр сдвига: *а* – схема на непрозрачных триггерах; *б* – условное графическое обозначение

Сдвигающий регистр, в отличие от регистра памяти, должен обязательно состоять из непрозрачных триггеров, иначе при первом же сигнале сдвига бит, поступивший на вход первого триггера регистра сдвига, сразу же пройдет на выход этого триггера и, соответственно, на вход второго триггера, а значит и на выход второго триггера и т. д. до последнего триггера регистра сдвига.

В некоторых сдвигающих регистрах сдвиг может производиться только в одну сторону, а в некоторых регистрах – *реверсивных* – и вправо, и влево.

Буквенные обозначения входов (рис. 5.27) означают:

P/*S* – PARALLEL/SERIAL – параллельная/последовательная загрузка (*P*/*S* = 1 – парал.);

SIO – SERIAL INPUT 0 – последовательный вход триггера 0;

*SI*3 – SERIAL INPUT 3 – последовательный вход триггера 3;

DSL – DATA SHIFT LEFT – сдвиг данных влево;

DSR – DATA SHIFT RIGHT – сдвиг данных вправо;

PE – PARALLEL ENABLE – разрешение параллельной записи.

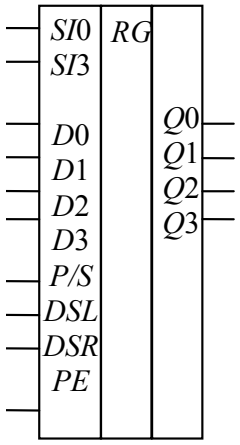


Рис. 5.27. Условное графическое обозначение реверсивного регистра

Чтобы обеспечить реверс сдвига и параллельную запись сразу во все триггеры, к *D*-входу каждого *i*-го триггера подключен мультиплексор, который при подаче лог.1 на один из управляющих входов – *SL* (Shift Left – сдвиг влево), *SR* (Shift Right – сдвиг вправо) или *PL* (Parallel load – параллельная загрузка) подключает вход *i*-го триггера, соответственно, к выходу младшего соседа (направление *A*), старшего соседа (направление *B*) или к выходу параллельной загрузки P_i (рис. 5.28). Точка *A* самого младшего триггера (разряда) является входом *DR*, точка *B* самого старшего разряда – входом *DL* (входы *DL* и *DR* используются для наращивания разрядности реверсивных регистров). Выход *i*-го триггера подключен к соответствующим входам мульти-

плексоров соседних разрядов. По *C*-сигналу триггеры регистра принимают информацию с направлений, определяемых мультиплексорами.

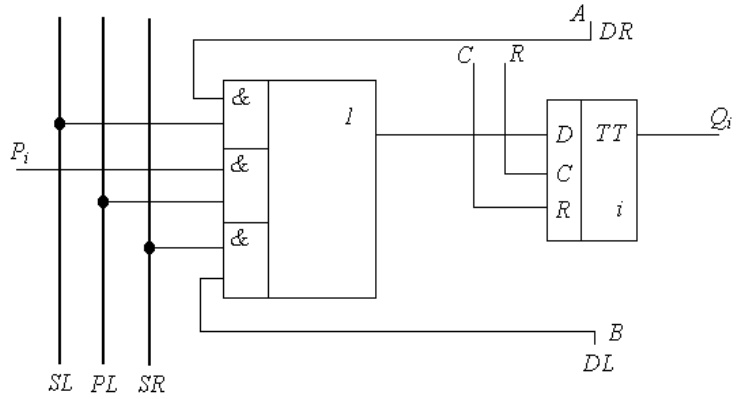


Рис. 5.28. Схема одного разряда реверсивного регистра

При разрешении передачи данных через нижний вход каждого мультиплексора (рис. 5.29) производят параллельную запись в триггеры регистра. При разрешении на среднем входе мультиплексора можно сдвигать информацию в регистре вправо, а при открытом верхнем элементе мультиплексора – влево.

Принято считать, что слева расположены старшие разряды двоичного слова, числа. Поэтому сдвиг влево увеличивает значение каждого исходного бита.

При сдвиге вправо на один разряд значение каждого бита как бы уменьшается вдвое.

На рис. 5.29 представлен регистр, способный осуществлять параллельную и последовательную запись данных.

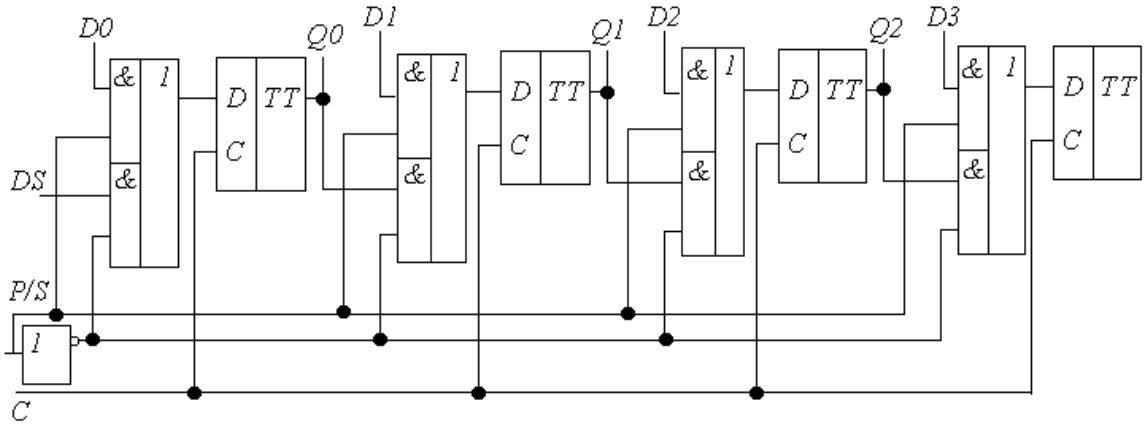


Рис. 5.29. Сдвиговый регистр с возможностью параллельного и последовательного ввода данных

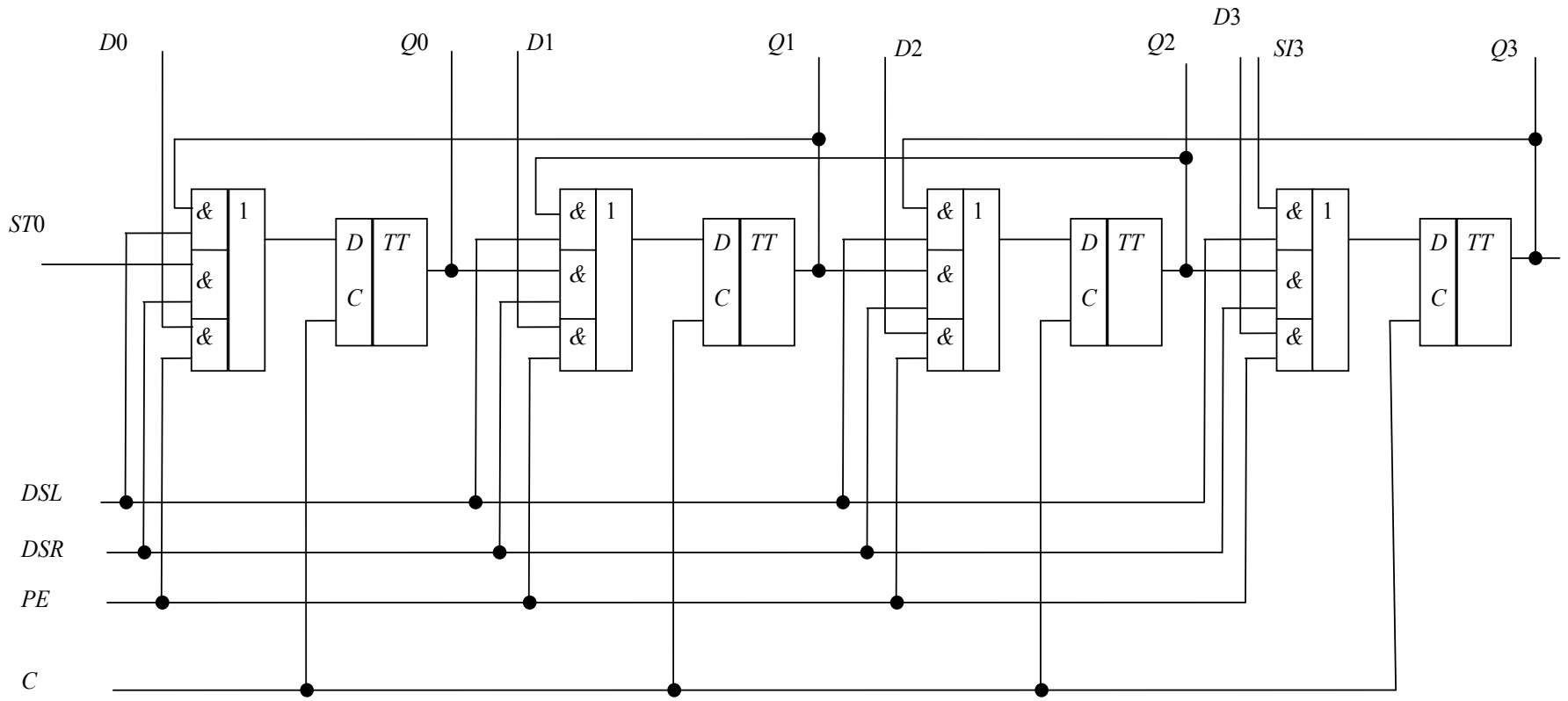


Рис. 5.30. Реверсивный четырехразрядный регистр

На рис. 5.29 *DS*-Data Serial – вход для последовательного ввода данных, *P*/*S* – Parallel/Serial – выбор режима.

При помощи подобных регистров можно осуществлять преобразование параллельного двоичного кода в последовательный (рис. 5.31).

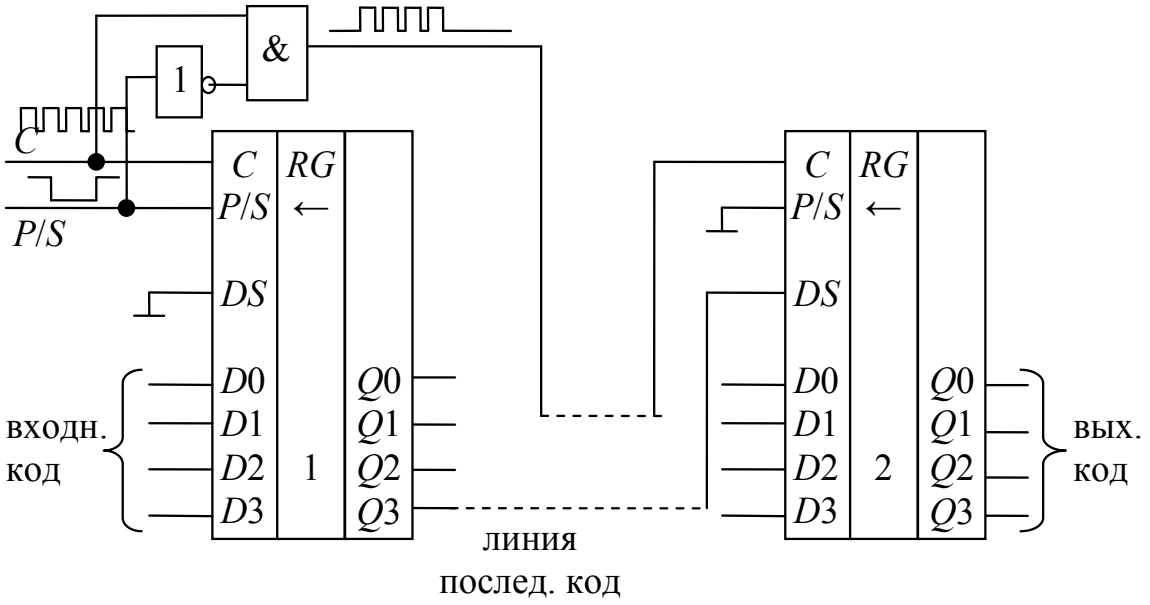


Рис. 5.31. Преобразование параллельного четырехразрядного кода в последовательный

Данные параллельным кодом загружаются в регистр *RG*1, пока *P*/*S* = 1. Затем при *P*/*S* = 0 *RG*1 переходит в режим сдвига (последовательный режим) и в течение четырех импульсов на *C*-входе *RG*1 содержимое этого регистра с его *Q*3-выхода разряд за разрядом передается на *RG*2, на *DS*-вход, где записывается последовательным кодом благодаря подаче четырех синхроимпульсов на *C*-вход *RG*2. После этого данные могут быть считаны параллельным кодом с выходов *Q*0, … *Q*3 регистра *RG*2.

Чтобы четырехразрядное двоичное число, представленное параллельным кодом, преобразовать в последовательный код, надо сначала разряды этого числа подать на входы *D*0, *D*1, *D*2, *D*3 регистра, а одновременно с этим подать разрешение на параллельную запись, т. е. *P*/*S* = 1, и подать один или несколько импульсов на *C*-вход регистра.

После этого нужно запретить параллельную запись в регистр и тем самым разрешить последовательный режим регистра, т. е. *P*/*S* = 0. Теперь при каждом импульсе на *C*-входе регистра выход *Q*3 будет выдавать поочередно разряды записанного двоичного числа.

Обычно последовательный код начинается с младшего разряда, а в данном регистре первым на выход поступит разряд числа, записанный ранее в триггер *ТТ*3, значит при параллельной записи числа в регистр следует проследить, чтобы младший разряд был записан в триггер *ТТ*3, а старший – в *ТТ*0.

5.2.3. Кольцевые схемы

Так называемый кольцевой счетчик (распределитель) представляет собой сдвиговый регистр, у которого информационный вход соединен с выходом последней ступени, образуя замкнутое кольцо, как это показано на рис. 5.32.

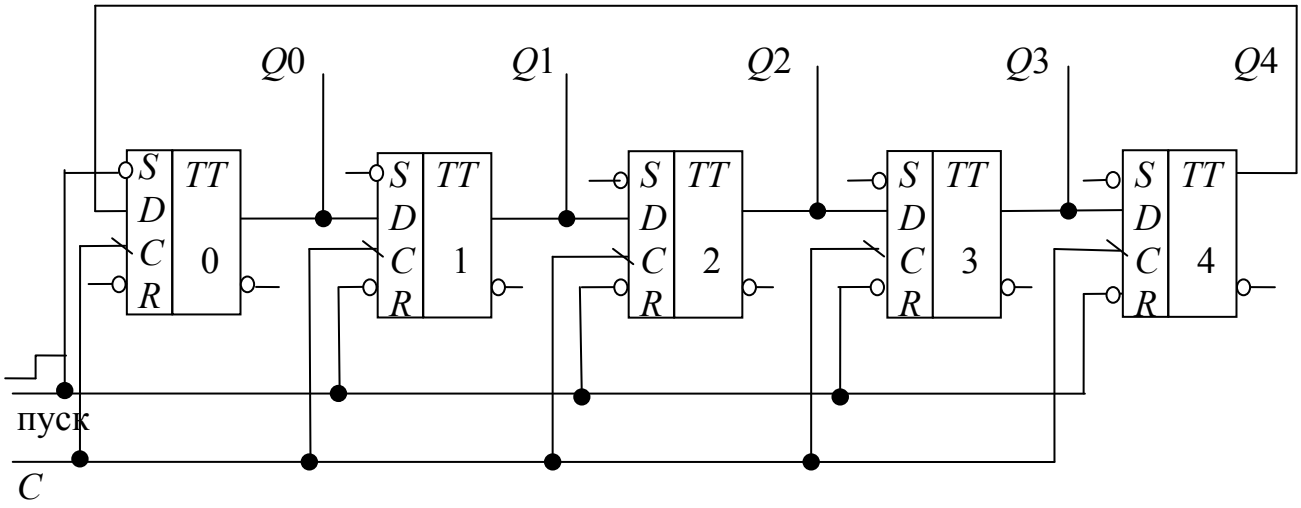


Рис. 5.32. Функциональная схема кольцевого счетчика

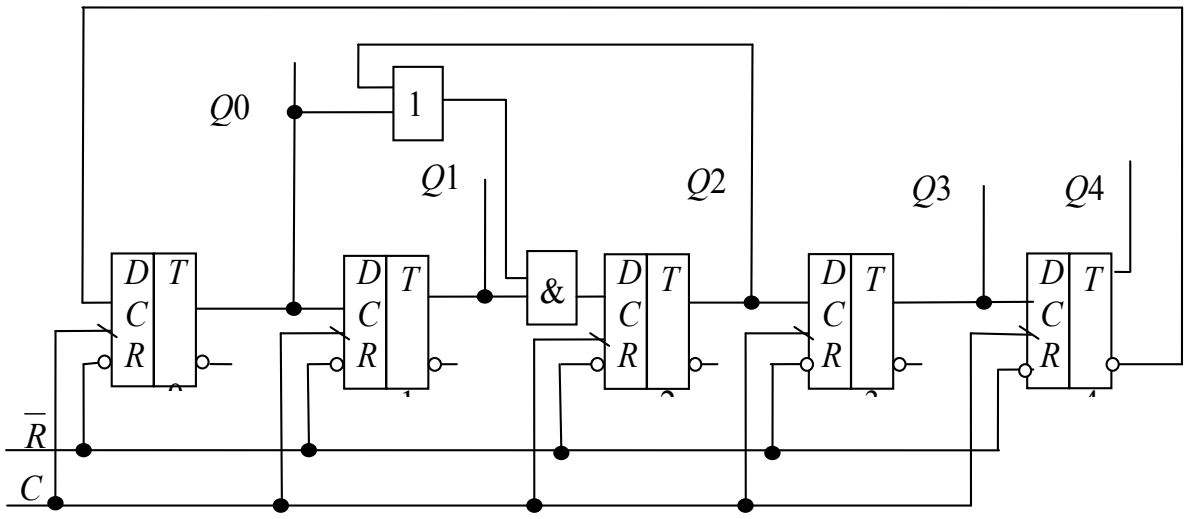
До начала работы в триггер *ТТ*0 записывается лог.1, а в остальные триггеры – лог.0. Коэффициент счета $K_{сч} = m$ (у двоичного счетчика $K_{сч} = 2^m$), где m – число триггеров.

Достоинство данной схемы: унарный код обеспечивается на выходе без помощи дешифратора и представлен активным уровнем на том выходе, номер которого соответствует подсчитанному числу импульсов.

Недостаток: повышенный расход триггеров. Второй недостаток кольцевого счетчика в том, что если в результате какой-либо помехи во время работы счетчика в нем нарушится слово, бегущее по кольцу (в данном примере это одна лог.1 и четыре лог.0), то уже неправильное слово будет циркулировать по кольцу счетчика.

Временные диаграммы работы кольцевого счетчика представлены на рис. 5.33.

Счетчик Джонсона (рис. 5.33) – это кольцевой счетчик, у которого одна из связей между триггерами сделана перекрестной, т. е. вход одного из триггеров соединен не с прямым, а с инверсным выходом предыдущего триггера.



$K_{сч} = 2m$

Рис. 5.33. Функциональная схема счетчика Джонсона

Временные диаграммы работы счетчика Джонсона представлены на рис. 5.34.

В отличие от предыдущего кольцевого счетчика в счетчике Джонсона по кольцу бежит волна единиц и нулей. Счетчик Джонсона тоже боится зацикливания ошибочных состояний, хотя и в меньшей степени, чем простой кольцевой счетчик. Для исправления ошибок, чтобы ошибка циркулировала в счетчике меньше половины кольца, в счетчик введены ЛЭ ИЛИ и И, которые обеспечивают возвращение счетчика к правильной работе.

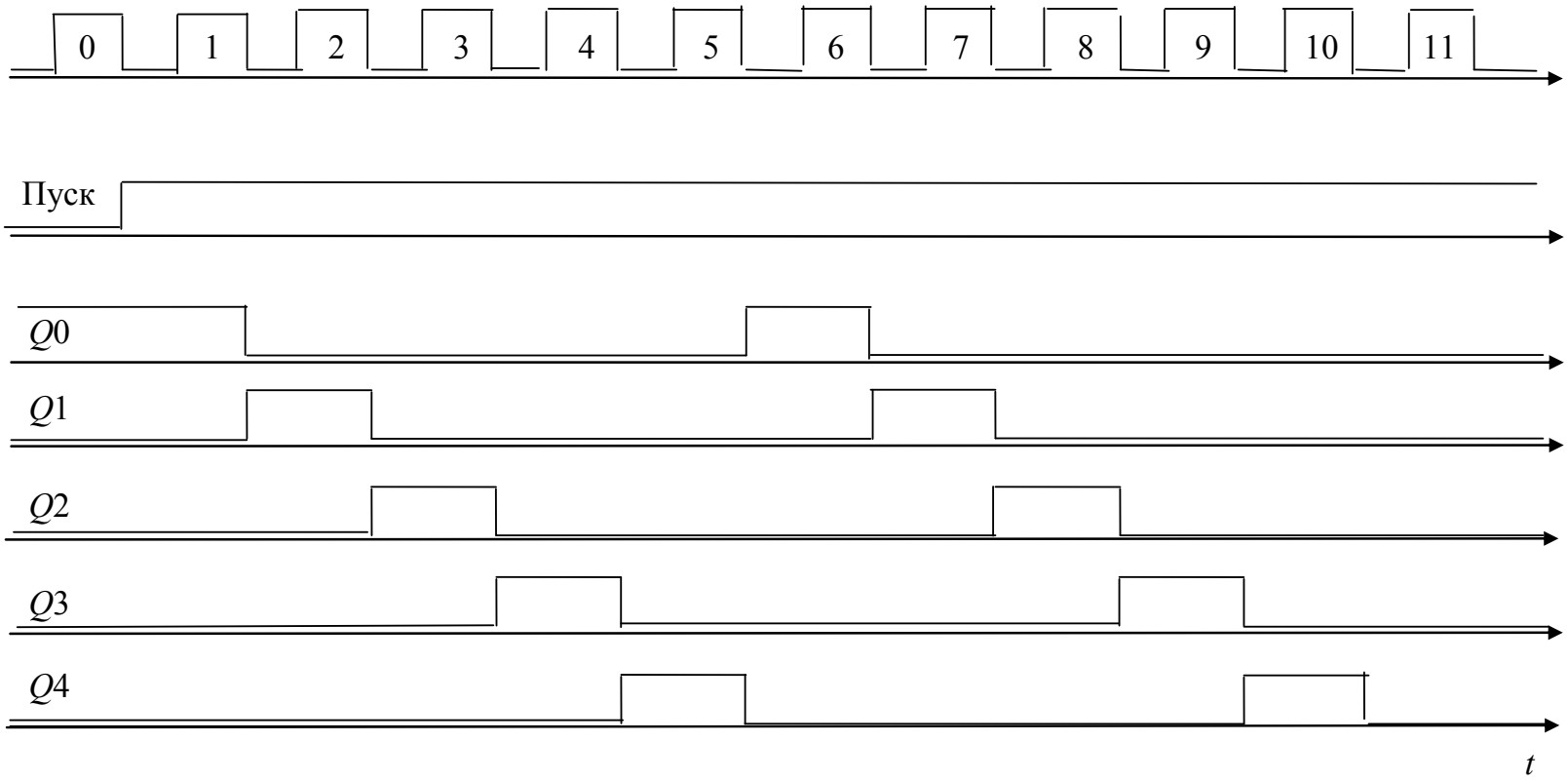


Рис. 5.34. Временные диаграммы работы кольцевого счетчика

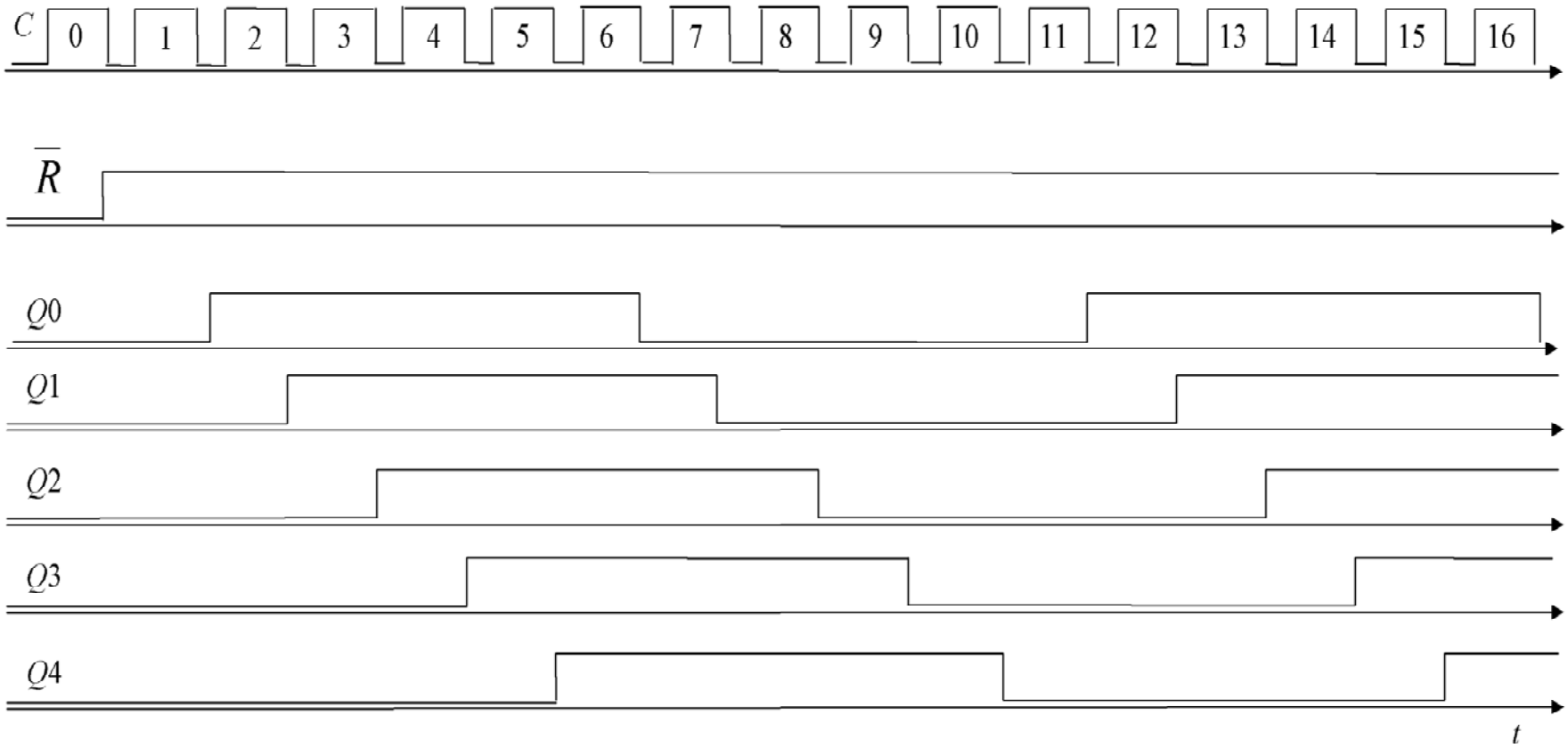


Рис. 5.35. Временные диаграммы работы счетчика Джонсона

В отличие от предыдущего кольцевого счетчика, чтобы получить унарный код на выходах, счетчик Джонсона должен иметь в своем составе дешифратор. Каждый *i*-й элемент И такого линейного дешифратора имеет только 2 входа, один из которых подключен к инверсному выходу *i*-го триггера, а второй – к прямому выходу (*i*–1)-го триггера (для двоичного счетчика такой дешифратор состоял бы из ЛЭ И с четырьмя входами).

Подобный счетчик на 5 триггерах с полным линейным дешифратором на 10 выходов – это К561ИЕ8. Счетчик Джонсона на четырех триггерах с дешифратором на 8 – это К561ИЕ9.

5.2.4. Полиномиальный счетчик

Полиномиальным счетчиком называют пересчетную схему из сдвигающего регистра при введении в него обратных связей посредством сумматоров по модулю 2.

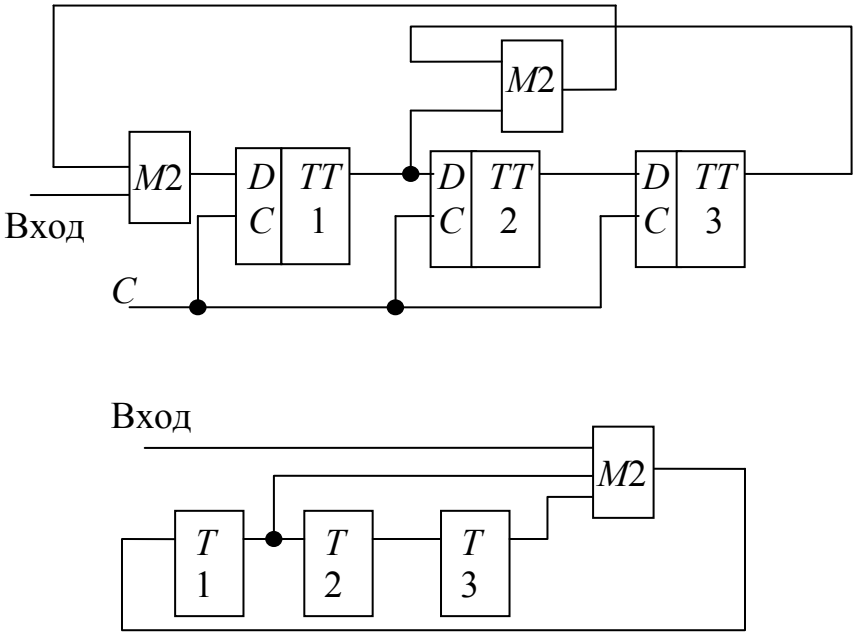


Рис. 5.36. Функциональная схема полиномиального счетчика

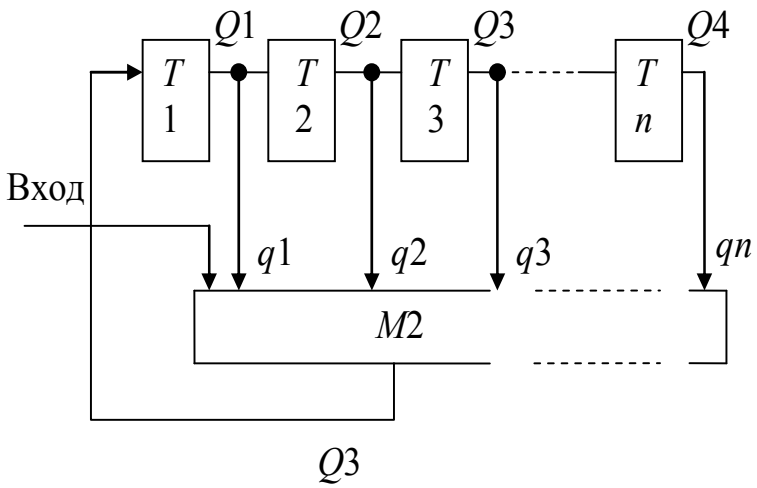
Пока на вход схемы не подается лог.1 и все триггеры в нулевом состоянии, счетчик не реагирует на синхроимпульсы.

При подаче на вход 1 счетчик по очередному синхроимпульсу переходит в состояние 100 и далее, при нулевом входном сигнале, в каждый такт происходит смена состояний триггеров по некоторому закону.

Счетчик входит в цикл и генерирует последовательность состояний в соответствии с таблицей переключений длиной 7 тактов. Порядок смены нулей и единиц в последовательности и ее длина зависит от числа триггеров и того, между какими разрядами заведены обратные связи.

N	Вход	*Q*1	*Q*2	*Q*3
	0	0	0	0
	..	..	..	..
	0	0	0	0
0	1	0	0	0
1	0	1	0	0
2	0	1	1	1
3	0	1	1	1
4	0	0	1	1
5	0	1	0	0
6	0	0	1	1
7	0	0	0	0
8	0	1	0	0
9	0	1	1	1
10	0	1	1	0

а)



б)

Рис. 5.37. N-разрядный полиномиальный счетчик: *а* – таблица переключений; *б* – функциональная схема

После окончания одной последовательности счетчик начинает генерировать вторую такую же, третью и т. д. до тех пор, пока его триггеры не будут сброшены в ноль (на схеме цепь сбора не показана).

Некоторые обратные связи, как и в данном примере, обеспечивают формирование счетчиком из n триггеров последовательность максимальной длины, M-последовательность длиной 2^n–1 тактов.

Таблица 5.2

Длина *M*-последовательностей в зависимости от разрядности

Разрядность	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**12**	**16**	**20**
Номера обратных связей	3, 1	4, 3	5, 4, 3, 2	6, 5	7, 3, 2, 1	8, 4, 3, 2, 1	9, 8, 7, 6, 5, 1	10, 3	3, 5, 6, 12	16, 12, 9, 7	20, 3
Длина последовательностей	7	15	31	63	127	255	511	1023	4095	65635	1048575

5.3. Счетчики

5.3.1. Определение и классификация

Счетчиком называется устройство, код на выходах которого отображает число импульсов, поступивших на счетный вход.

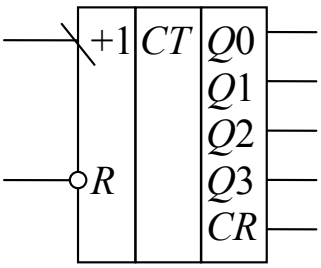


Рис. 5.38. Условно-графическое обозначение счетчика

Счетчик можно назвать сумматором число-импульсного кода, поступающего на его вход. Иногда счетчик называют делителем частоты.

T-триггер является примером простейшего одноразрядного счетчика. Выход *CR* – CARRY – обозначает перенос, переполнение. Вход +1 – суммирующий счетный вход, вход *R* – общий сброс. Каждый счетчик характеризуется коэффициентом счета (модулем $K_{сч}$). Для двоичного счетчика $K_{сч} = 2$, для *m*-разрядного счетчика $K_{сч} = 2^m$. При поступлении на счетный вход импульсов состояние выходов суммирующего счетчика постепенно меняется от всех нулей до всех единиц, при поступлении следующего импульса счетчик вырабатывает сигнал переполнения *CR* и устанавливается в состояние всех нулей на выходах.

В общем случае код на входе счетчика соответствует числу $M = N_{вх} - iK_{сч}$, где $N_{вх}$ – число импульсов, поступивших на вход счетчика, а i = 0,1,2,3, …, n.

Число импульсов на выходе переноса *CR* будет в $K_{сч}$ раз меньше, чем число импульсов на его счетном входе. Отсюда следует. что частота импульсов на выходе счетчика $f_{вых}$ меньше частоты импульсов на его счетном входе $f_{вх}$ в $K_{сч}$ раз: $f_{вых} = \frac{f_{вх}}{K_{сч}}$.

Счетчики классифицируются следующим образом:

1. По модулю пересчета:

а) двоичные $K_{сч} = 2^m$;

б) двоично-десятичные $K_{сч} = 10^m$;

в) с управляемым модулем счета;

г) с постоянным произвольным $K_{сч}$.

2. По направлению счета:

а) суммирующие;

б) вычитающие;

в) реверсивные.

3. По способу организации внутренних связей:

а) с непосредственной связью (с последовательным переносом);

б) с параллельным переносом;

в) кольцевые на регистрах сдвига.

4. По способу защиты от гонок и помех на входе:

а) асинхронные;

б) синхронные.

5.3.2. Суммирующие счетчики

Рассмотрим суммирующий счетчик К155ИЕ5 с последовательным переносом (рис. 5.39а–5.39в).

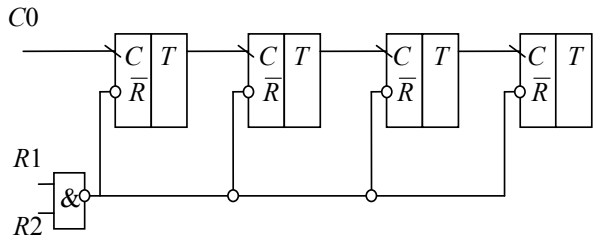


Рис. 5.39а. Счетчик К155ИЕ5 (структурная схема)

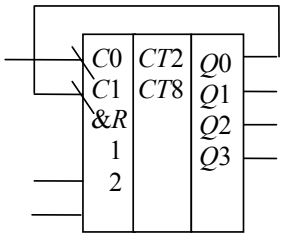


Рис. 5.39б. Счетчик К155ИЕ5 (условно-графическое изображение)

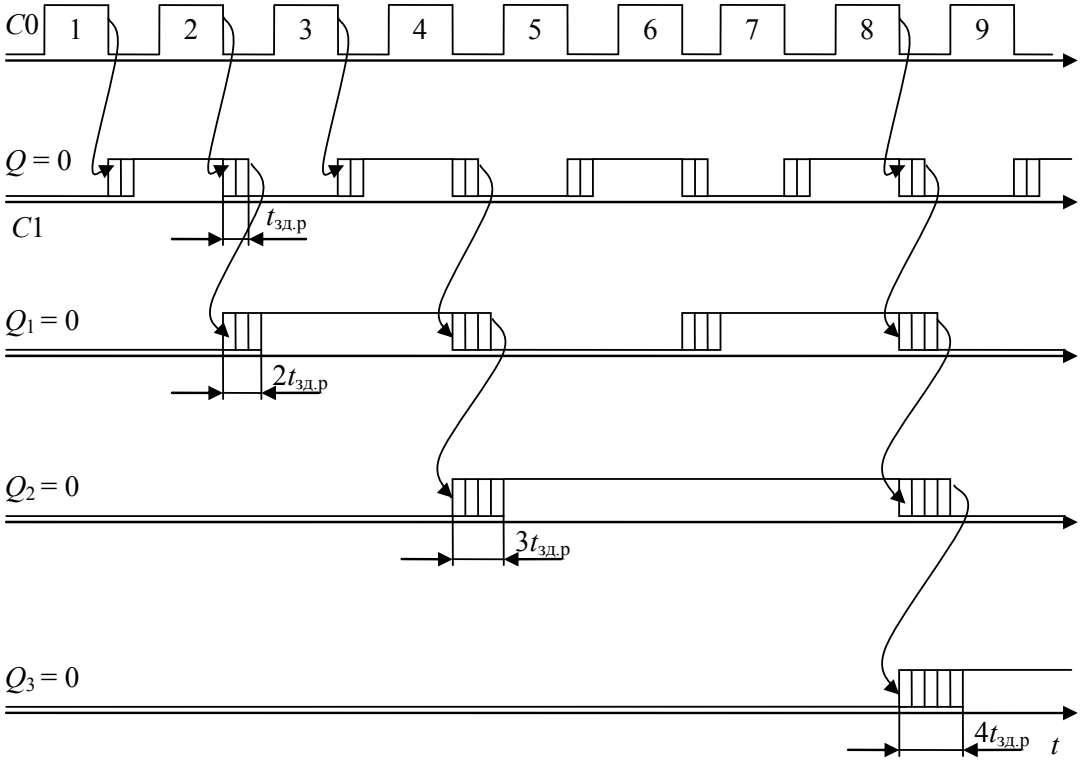


Рис. 5.39в. Счетчик К155ИЕ5 (временные диаграммы)

Переключение триггеров в счетчике по фронтам C^{10} происходит поочередно, *последовательно*, поэтому задержка переключения 3-го разряда (считая с нулевого) в 4 раза больше задержки переключения одного триггера относительно входного сигнала. Такие счетчики не бывают синхронными, так как быстро накапливающаяся задержка может привести к ложным срабатываниям в последних триггерах.

Роль выходного переноса, выполняет выход $Q3$, старшего (последнего) разряда (триггера). Число импульсов N, поступивших на счетный вход, отображается двоичным кодом на выходах счетчика следующим образом:

$$N = Q_3 2^3 + Q_2 2^2 + Q_1 2^1 + Q_0 2^0,$$

где $Q = 0; 1$.

Рассмотрим синхронный суммирующий счетчик К155ИЕ10 с параллельным переносом (рис. 5.40).

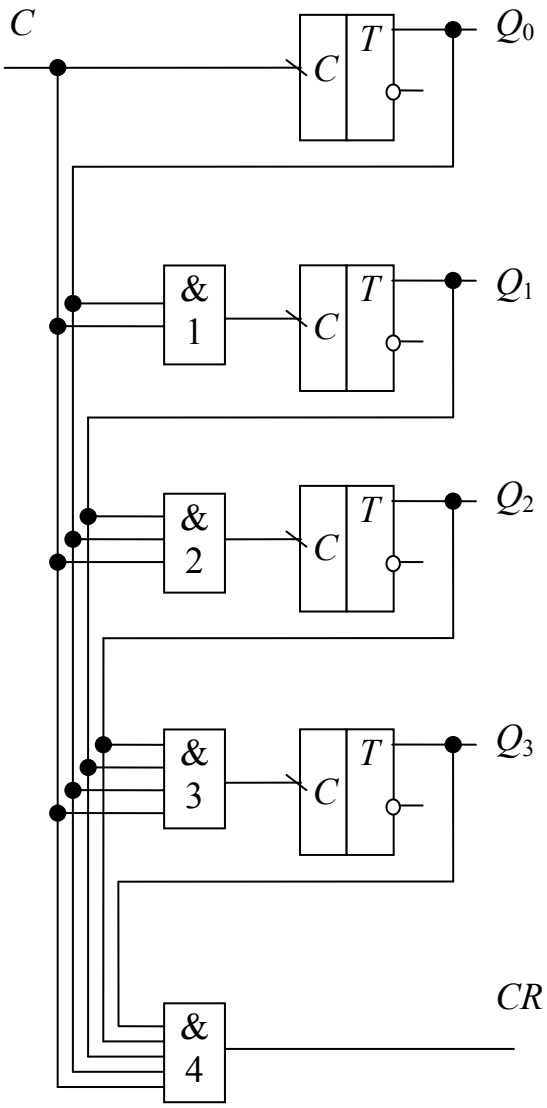


Рис. 5.40. Структура счетчика К155ИЕ10 с параллельным переносом

Входной сигнал *С* поступает параллельно на входы всех логических элементов И схем и там, где эти логические элементы И открыты, он вызывает одновременное переключение триггеров. Кроме входного сигнала *С* на входы ЛЭ И поданы сигналы с выходов всех предыдущих, более младших триггеров, поэтому при поступлении на входы сигнала *С* изменят свое состояние те триггеры, перед которыми все более младшие триггеры находятся в состоянии лог.1.

В этом счетчике задержка переключения не накладывается от триггера к триггеру, как в предыдущем счетчике 155ИЕ5. Часто этот счетчик называют синхронным.

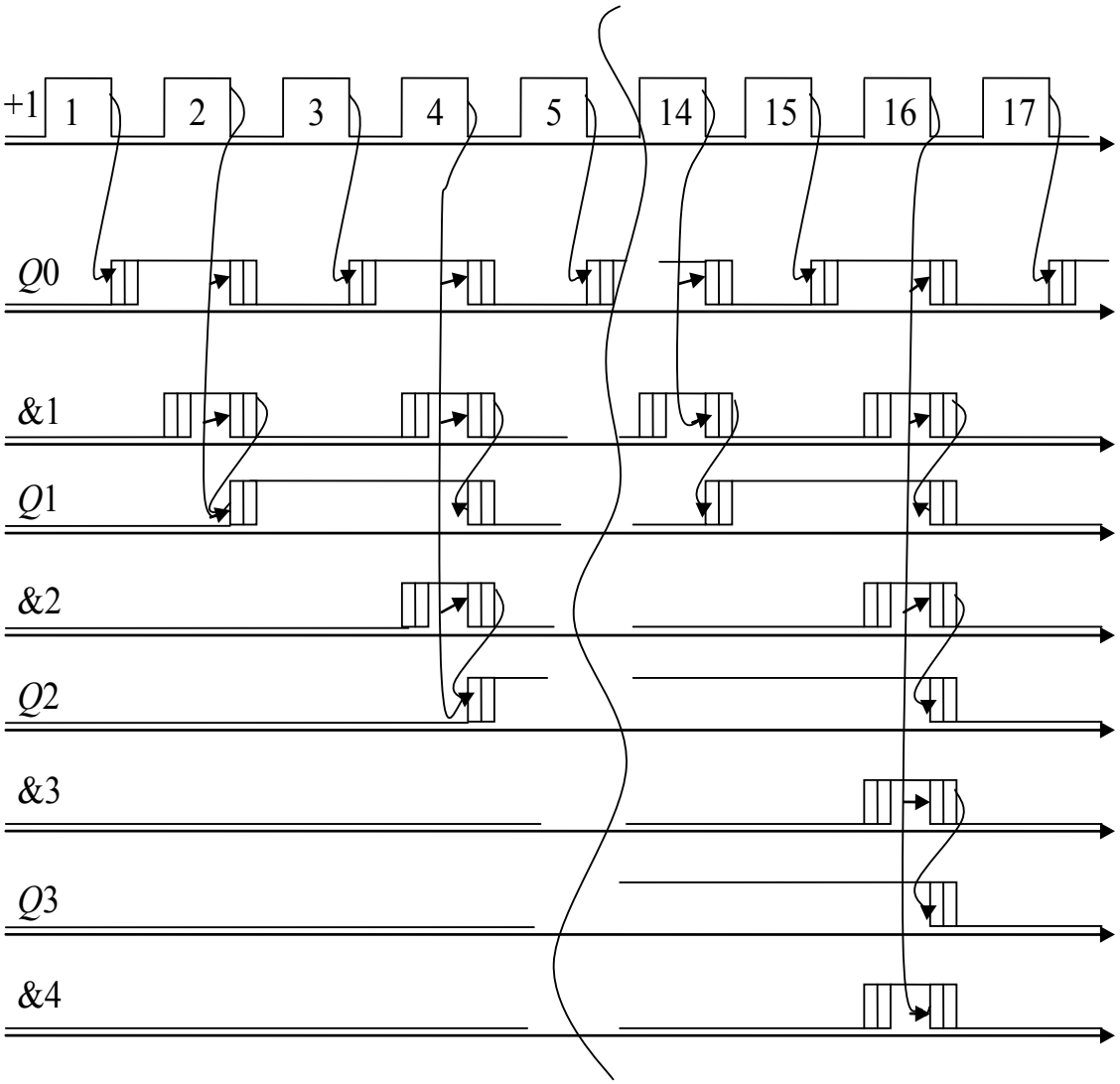


Рис. 5.41. Временные диаграммы работы счетчика К155ИЕ10

5.3.3. Реверсивные счетчики

До этого рассматривались лишь суммирующие счетчики, код на выходах которых, начинаясь со всех нулей, увеличивает свое значение на единицу при каждом импульсе на счетном входе. Наибольшее значение выходного кода двоичного счетчика состоит из единиц во всех разрядах, если не ограничивать его коэффициентом счета.

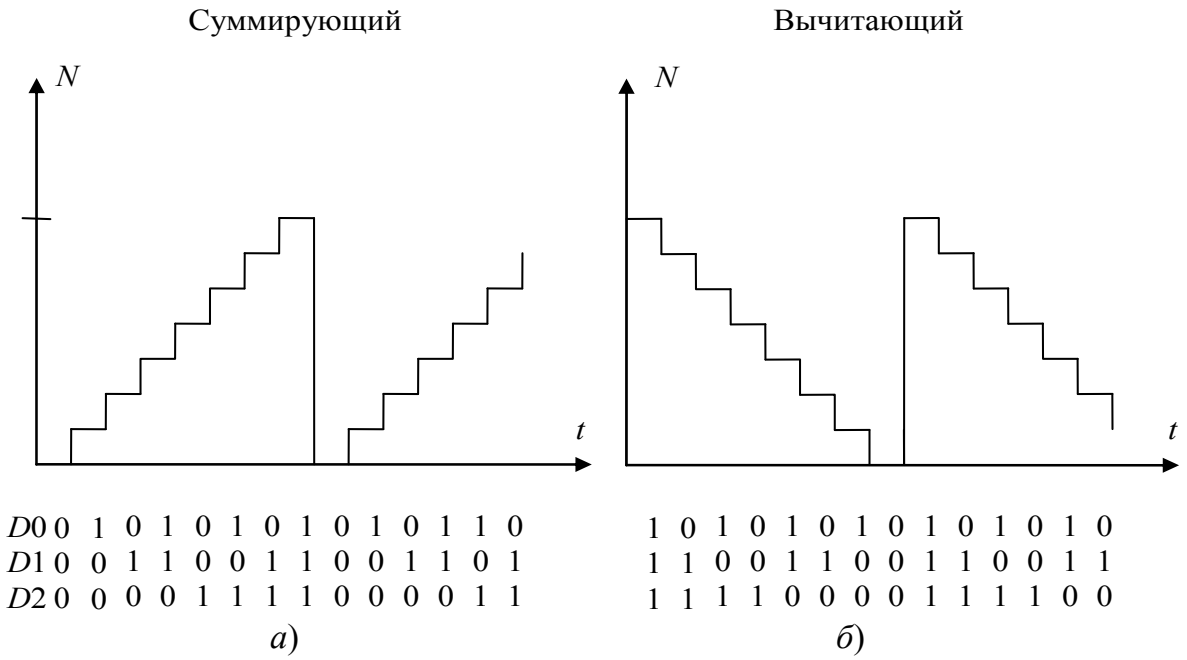


Рис. 5.42. Временные диаграммы работы: *а* – суммирующего счетчика; *б* – вычитающего счетчика

Вычитающий счетчик с каждым счетным импульсом уменьшает на единицу код на своих выходах. Отдельными микросхемами вычитающие счетчики не выпускают, но такие счетчики входят в состав более сложных микросхем, как, например, К561ИЕ15, счетчик с переключаемым коэффициентом пересчета. В тех случаях, когда не уточняется, о каком счетчике идет речь, подразумеваются суммирующие счетчики.

Счетчики, которые могут считать в обоих направлениях, называют реверсивными.

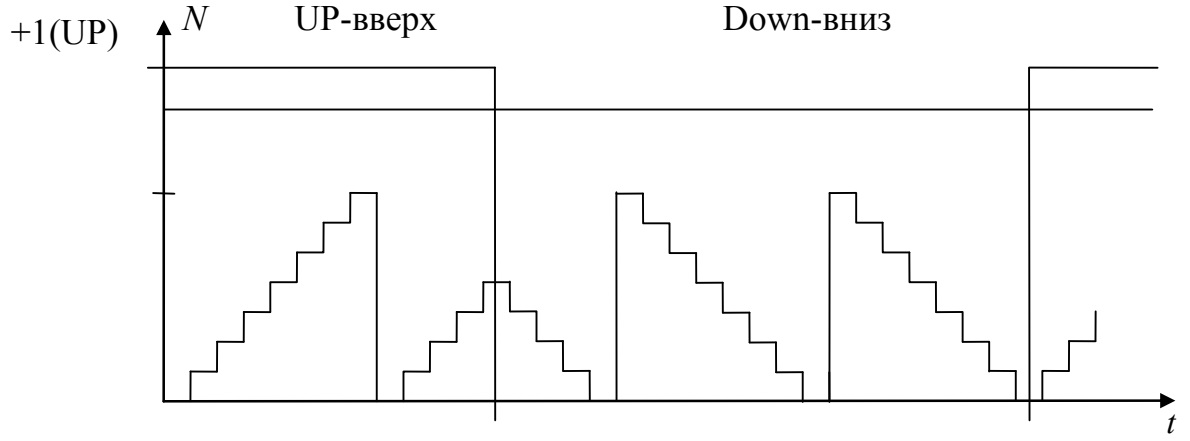
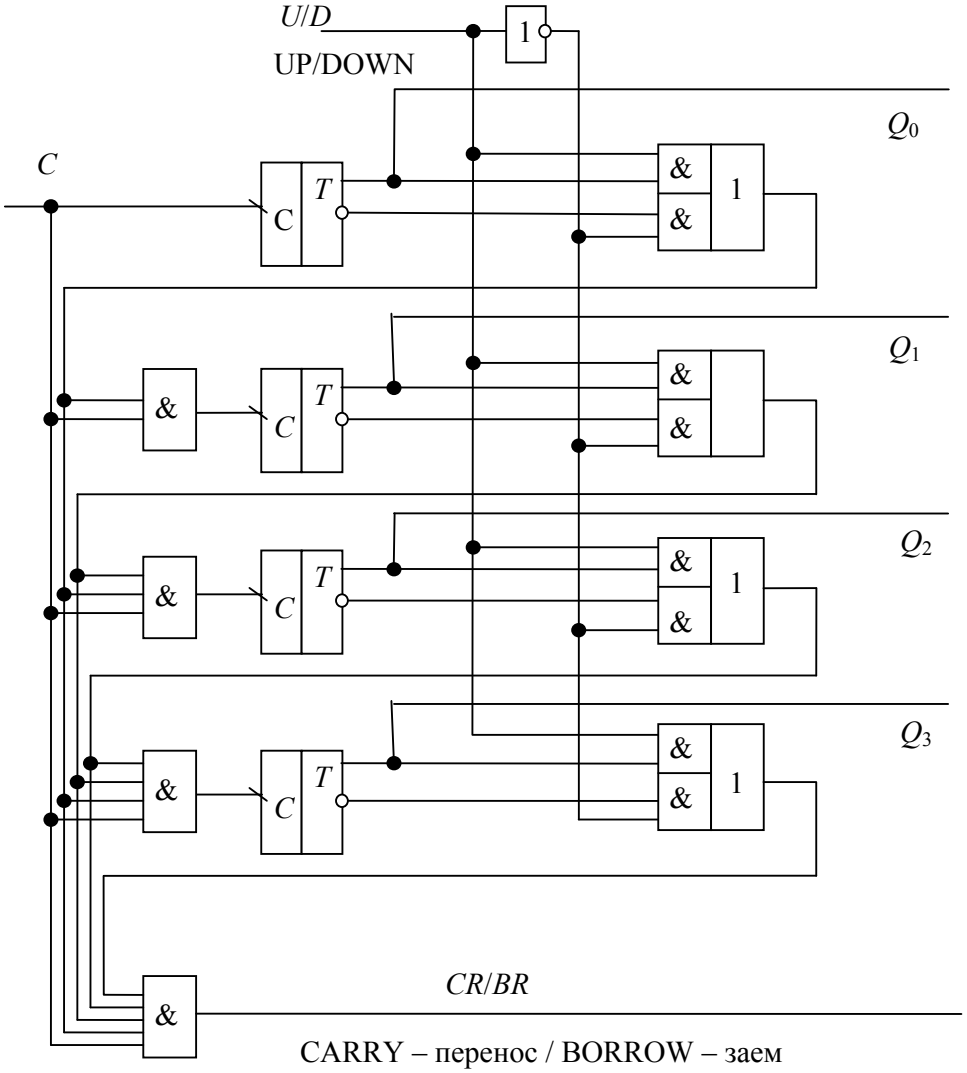


Рис. 5.43. Временные диаграммы работы реверсивного счетчика

Если среди выводов суммирующего счетчика есть и прямые, и инверсные выходы триггеров, то этот счетчик можно использовать как вычитающий счетчик; для этого просто вместо прямых выходов использовать инверсные выходы триггеров.



а)

б)

Рис. 5.44. Реверсивный счетчик К155ИЕ11: *а* – структура; *б* – условно-графическое обозначение

Здесь импульсы, поступающие на вход *C*, или суммируются, или вычитаются счетчиком в зависимости от состояния входа *U*/*D*, если на входе *U*/*D* лог.1 счетчик суммирует, если *U*/*D* = 0 – вычитает, а выход *CR*/*BR* является, соответственно, выходом переноса или заема.

В микросхемах К155ИЕ6, К155ИЕ7 счетные импульсы для суммирования подают на вход +1 (*U*), а импульсы для вычитания – на вход –1 (*D*) а выходы «≥» (*PU*) и «≤» (*PD*) являются, соответственно, раздельными выходами переноса и заема.

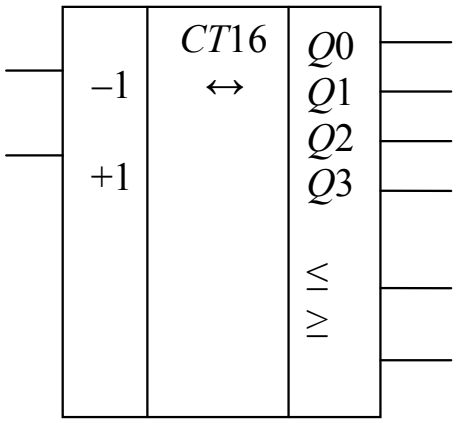


*Рис. 5.45.*Условно-графическое изображение счетчика К155ИЕ6

5.3.4. Счетчики с регулируемым модулем счета

Счетчики с регулируемым коэффициентом пересчета строят двумя основными способами:

• *Схема со сбросом счетчика в ноль дополнительным логическим элементом* И.

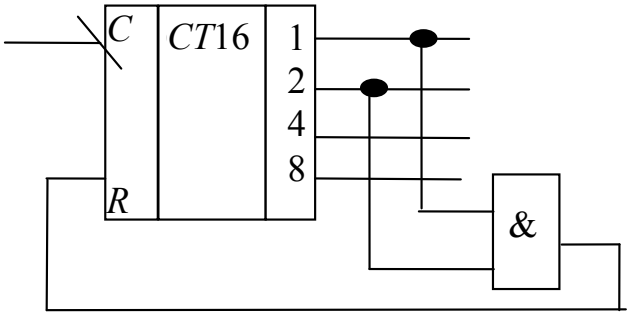


Рис. 5.46. Схема сброса счетчика логическим элементом И

Коэффициент пересчета равен сумме тех весов разрядов счетчика, которые подключены к сбросовой схеме И.

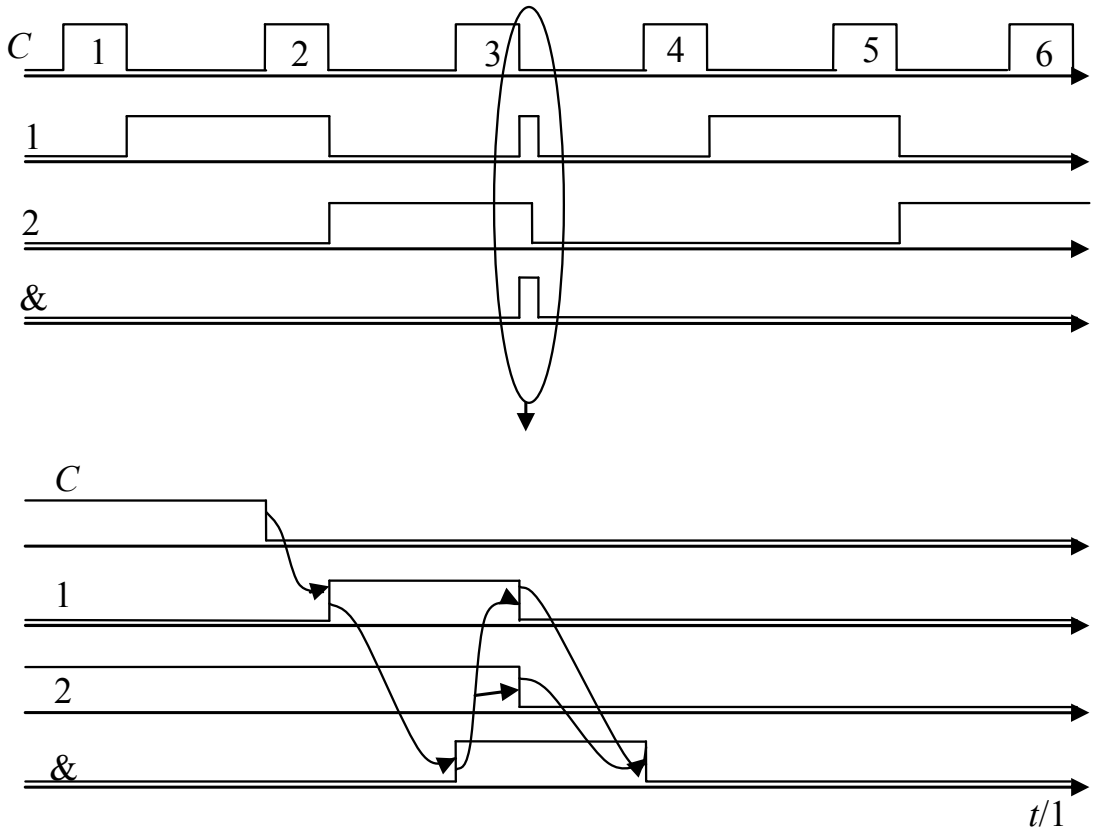


Рис. 5.47. Временные диаграммы, поясняющие недостатки работы схемы сброса счетчика дополнительным ЛЭ И

Недостатки схемы сброса счетчика ЛЭ И:

1. На выходе младшего триггера из всех подключенных к сбросовой схеме логического элемента И будут присутствовать импульсы помехи в момент сброса.

2. Если в качестве сигнала переноса использовать импульсы с выхода сбросовой схемы И, то эти импульсы могут оказаться слишком короткими для надежного срабатывания последующих схем.

• Схема с загрузкой в счетчик дополнения $\overline{K_{\text{сч}}} = 2^m - K_{\text{сч}}$.

Двоичный счетчик перед началом счета и в конце каждого импульса переноса загружают кодом, соответствующим дополнению $K_{\text{сч}}$ до 2^m, где m – число разрядов счетчика.

Недостатком данного способа является неестественность последовательности выходного кода, так как он меняется не от всех нулей до $K_{\text{сч}}$, а от $\overline{K}_{\text{сч}}$ до всех единиц.

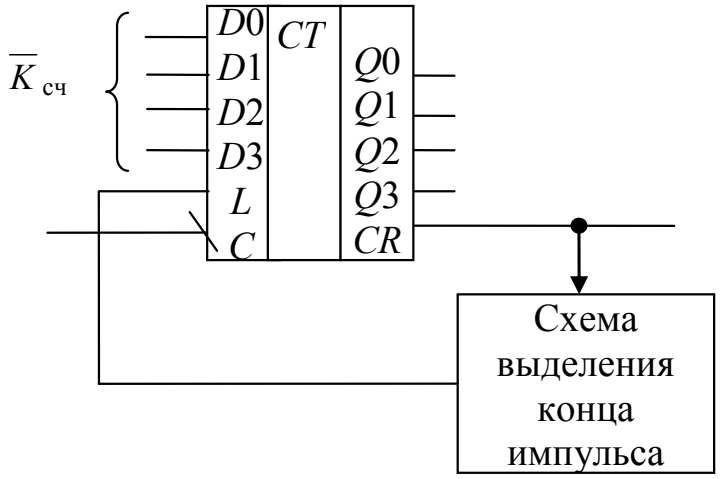


Рис. 5.48. Схема с загрузкой в счетчик дополнения $\overline{K_{\text{сч}}} = 2^m - K_{\text{сч}}$

Этот способ используется там, где не нужны значения выходов кода – в счетчиках – делителях частоты. Здесь выход *L* – LOAD – загрузка данных (см. синхронный *RS*-триггер).

5.3.5. Счетчики с постоянным произвольным модулем счета

Такие счетчики чаще всего разрабатывают на коэффициенты пересчета, равные 10 или 6. Для их реализации обычно используют свойства входов *D*- и *JK*-отдельных триггеров, а не приведенные выше два способа реализации изменения коэффициента счета.

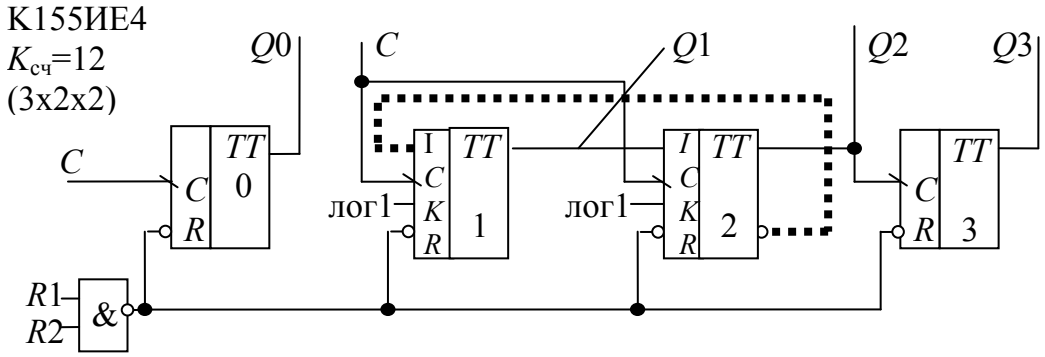


Рис. 5.49. Структурная схема счетчика К155ИЕ4

Триггеры *TT*1 и *TT*2 охвачены цепью обратной связи, показанной пунктирной линией на рис. 5.49, за счет которой их общий модуль счета равен трем. Модуль счета трех триггеров *TT*1, *TT*2 и *TT*3 равен 6, а модуль счета всего счетчика К155ИЕ4 равен 12.

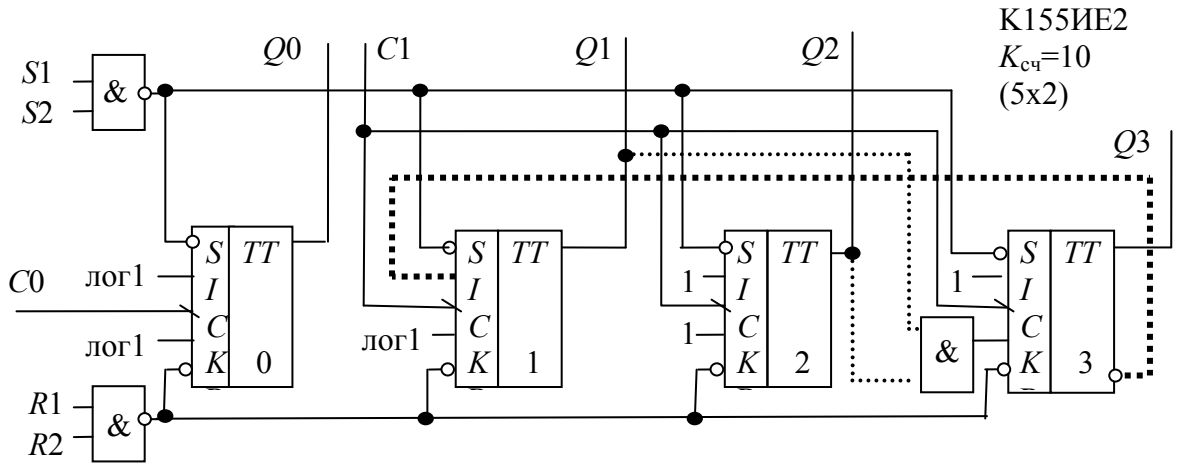


Рис. 5.50. Структурная схема счетчика К155ИЕ2

В двоично-десятичном счетчике 155ИЕ2 три триггера *TT*1, *TT*2 и *TT*3, охваченные обратной связью, показанной жирной пунктирной линией и прямой связью, показанной тонкой пунктирной линией (рис. 5.50), образуют счетчик с коэффициентом счета $K_{сч} = 5$.

5.4. Цифровые автоматы

5.4.1. Общие сведения о конечных цифровых автоматах. Основные понятия теории конечных автоматов

Автоматами называют схемы, выходы которых зависят не только от значений входов в данный момент, но и от комбинаций значений входов в определенные прошлые моменты времени. Автомат в некотором смысле помнит прошлые воздействия в отличие от комбинационной схемы (КС), выход которой определяется значениями входов только в настоящее время (разумеется, после окончания переходных процессов). *Автомат имеет память*, комбинационная схема памяти не имеет.

Тем или иным содержимым памяти автомата определяется его *внутреннее состояние*, или просто *состояние*. Внешнее проявление различных состояний – это различные реакции автомата на одни и те же воздействия.

Обобщенная структурная схема цифрового автомата показана на рис. 5.51.

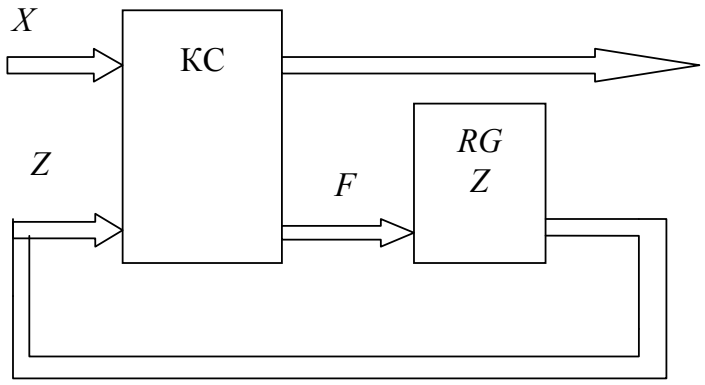


Рис. 5.51. Структурная схема цифрового автомата

Выходной код Y, вырабатываемый КС, есть функция не только *входного кода X*, но и *кода состояния Z*, который хранится в *регистре состояний RG Z.* Этот регистр и есть *память* автомата. Чем больше его емкость, тем богаче спектр поведения автомата, шире разнообразие его реакций на одни и те же входные воздействия, полнее учет прошлого опыта.

Новое состояние Z_{i+1}, в которое переходит автомат после очередного входного воздействия, т. е. новое содержимое *RGZ*, задается *кодом перехода F*. Код *F*, так же как и выходной код *Y*, есть функция и входных сигналов, и состояния автомата непосредственно перед его переходом в новое.

Цифровые автоматы (последовательностные схемы), по способу воздействия на них входных информационных сигналов подразделяются на *три основных класса*:

– асинхронные потенциальные автоматы;

– синхронные автоматы;

– асинхронные импульсные автоматы.

Каждый из классов автоматов можно разделить на несколько типов по другим признакам.

На *асинхронный потенциальный автомат* воздействия производят непосредственно его входные информационные сигналы до тех пор, пока сохраняется их активный уровень.

В *синхронном автомате* используется единый для всего автомата тактовый сигнал, который осуществляет на него импульсное воздействие в соответствии с уровнями входных информационных сигналов, т. е. в синхронном автомате последние только управляют воздействиями, а момент времени воздействия полностью определяется тактовым сигналом. Далее информационные сигналы будут называться просто *входными сигналами* автомата.

Асинхронный импульсный автомат отличается от потенциального тем, что входные потенциальные сигналы производят на него импульсные воздействия в момент их изменения. Такие воздействия в синхронных автоматах производит только один сигнал – тактовый, а в асинхронных импульсных автоматах любой из входных сигналов может управлять воздействиями и производить их.

Так как воздействия активных уровней входных сигналов на асинхронный потенциальный автомат происходят непрерывно, то при их синтезе необходим учет переходных процессов, вызываемых этими воздействиями. В синхронных же автоматах изменения входных сигналов не могут вызвать переходного процесса без участия тактового сигнала, который задает моменты времени, в которые автомат воспринимает значения входных сигналов. Тактовый сигнал поступает на синхронный автомат от внешнего генератора, не синхронизированного с его входными сигналами. Поэтому в момент воздействия на автомат тактового сигнала входные сигналы не должны изменяться. При выполнении этого требования переходные процессы в синхронных автоматах можно не рассматривать, обеспечив выбором соответствующей частоты тактового сигнала окончание переходного процесса к следующему моменту его воздействия. Таким образом, в синхронных автоматах вместо непрерывного времени вводится в рассмотрение *дискретное время*, задаваемое генератором тактового сигнала.

Для проектирования цифровых автоматов используются не только ЛЭ, но и *элементы памяти* (ЭП), в качестве которых чаще всего используются триггеры. Элементы памяти и триггеры являются элементарными автоматами, поэтому они, как и цифровые автоматы, делятся на те же три класса.

Классы автоматов различаются только типом используемых в них ЭП:

– с потенциальным воздействием входных сигналов;

– импульсным воздействием только одного тактового сигнала;

– с импульсным воздействием нескольких входных сигналов (может быть и одного).

Наиболее фундаментальной и сложной является теория асинхронных потенциальных автоматов, и только ее методами можно синтезировать триггеры с импульсным восприятием изменений входных потенциальных сигналов, например тактового.

Такие триггеры необходимы для структурного синтеза автоматов, принадлежащих к другим классам. Основные понятия теории автоматов являются общими для всех классов автоматов, различия же

появляются в основном на этапе их структурного синтеза из-за особенностей законов функционирования используемых триггеров.

В отличие от КС значения выходных сигналов автомата в данный момент времени зависят не только от значений входных сигналов в этот же момент времени, но и от предыдущих их значений. Из этого следует, что цифровые автоматы реализуют функциональную связь уже не между отдельными значениями входных и выходных сигналов, а между их последовательностями. Таким образом, в отличие от КС работу автоматов следует рассматривать во времени. Чтобы значения выходных сигналов зависели от предыдущих значений входных сигналов, автомат должен обладать памятью, в которой сохраняется информация о предыдущих входных воздействиях. Эта информация используется в автомате в виде совокупности сигналов, вырабатываемых памятью и называемых *внутренними сигналами*.

На рис. 5.52, *а* показана основная модель асинхронного потенциального автомата, которая состоит из Комбинационной схемы КС и *элементов задержки D* входных сигналов Q_r^+ на время Δt, включенных в обратных связях КС. При этом основная модель синхронного автомата (рис. 5.52, *б*) отличается от асинхронной только тем, что на элемент памяти ЭП типа *D* подается еще тактовый сигнал.

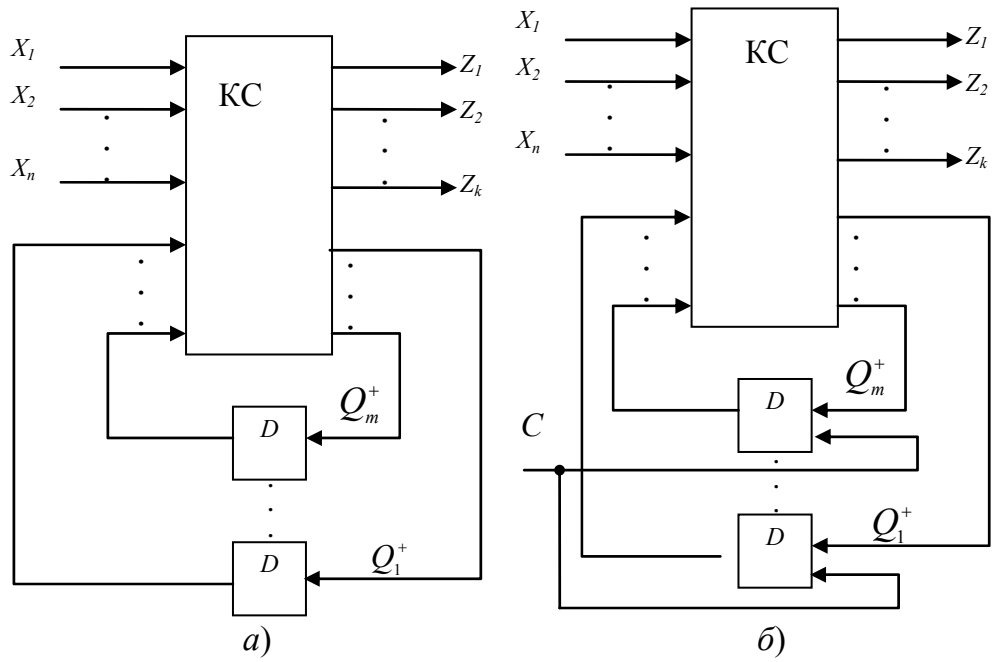


Рис. 5.52. Основная модель: *а* – асинхронного потенциального автомата; *б* – синхронного автомата

Элементы задержки производят запоминание внутренних сигналов КС $D_r = Q_r^+ (r = 1, 2, ..., m)$ на время Δt, т. е. они являются элементами памяти. Эти сигналы появляются на входах КС (выходах ЭП) через время Δt и могут вызвать изменение ее выходных сигналов. Понятно, что если сигнал $Q_r = Q_r(t)$, то сигнал $Q_r^+ = Q_r(t + \Delta t)$.

Практическое применение асинхронных автоматов существенно затруднено из-за сильного влияния на их работу задержек сигналов в цепях автоматов, создающих статические и динамические риски, гонки элементов памяти (неодновременность срабатывания ЭП даже при одновременной подаче на них входных сигналов) и др. В итоге характерным свойством асинхронного автомата является то, что при переходе из одного устойчивого состояния в другое он обычно проходит через промежуточные нестабильные состояния. Нельзя сказать, что методы борьбы с нежелательными последствиями рисков и гонок в асинхронных АП отсутствуют, но все же обеспечение предсказуемого поведения АП – сложная проблема. В более или менее сложных АП асинхронные схемы встречаются очень редко, а в простейших схемах применяются. Примером могут служить асинхронные *RS*-триггеры.

В синхронных автоматах каждое состояние устойчиво и переходные временные состояния не возникают. Концепция борьбы с последствиями рисков и гонок в синхронных автоматах проста – прием информации в элементы памяти разрешается только после завершения в схеме переходных процессов. Это обеспечивается параметрами синхроимпульсов, задающих интервалы времени для завершения тех или иных процессов. В сравнении с асинхронными, синхронные АП значительно проще в проектировании.

На сегодняшний день и достаточно длительную перспективу основным путем построения АП следует считать применение тактирования, т. е. синхронных автоматов.

5.4.2. Автоматы Мили и Мура

Детерминированные преобразователи дискретной информации, реализующие некоторые заданные алфавитные операторы, называют *дискретными* (*конечными*) автоматами. Конечные автоматы, рассматриваемые безотносительно к их внутренней структуре, обычно называют *абстрактными конечными автоматами*.

Автоматом с памятью называют автомат, описываемый функциями переходов и выходов, оператор которого является оператором

с памятью. Выходные слова автомата с памятью зависят не только от входных слов, но и от последовательности их поступления.

Автомат с памятью имеет множество внутренних состояний, в которое он переходит под воздействием слов входного алфавита. Наличие множества внутренних состояний придает автомату способность запоминания входной информации, поступившей на вход автомата в прошлом. В зависимости от вида функции выходов автоматы с памятью делятся на *автоматы Мили* и *автоматы Мура*.

Автоматом Мили называется автомат, выходные слова которого зависят как от внутренних состояний автомата, так и от значений входных слов.

Автоматом Мура называется автомат, выходные слова которого зависят только от внутренних состояний автомата и не зависят непосредственно от входных слов.

5.4.3. Синтез цифровых автоматов

Задачу синтеза автомата средней сложности удобно разбить на три части [7]:

– *формализация задания*;

– *кодирование состояний*;

– *синтез комбинационной схемы.*

Эти части не полностью автономны, и работа над одной из них обычно требует коррекции результатов двух других.

Формализация задания автомата

Автоматы обычно строят для управления работой различных технических объектов: исполнительных механизмов, приводов, электронных устройств, вплоть до целых ЭВМ. В этой роли автоматы называют программными датчиками, генераторами управляющих сигналов, блоками (устройствами) управления. Управляющим воздействием на объект являются выходы автомата *Y*. На входы автомата *X* поступают команды оператора (или вышестоящего автомата), сигналы, описывающие окружающую обстановку, сигналы о ситуациях, возникающих в управляемом объекте. Автомат проектируется под конкретный объект с его конкретным алгоритмом управления, и сложность проектирования автомата – именно в его нестандартности. *Задание на автомат* сначала дается в виде словесного описания закона управления, и, как правило, первый вариант задания оказывается очень неполным, допускающим неоднозначное толкование. Поэтому сначала задание нужно *формализовать.* В процессе этой работы задание уточняется, корректируется. Чтобы с этим справиться, разработ-

чик или сам должен хорошо знать условия работы объекта, или иметь на этапе формализации тесный контакт с заказчиком. Чтобы проиллюстрировать характер возникающих при синтезе автомата вопросов, процедура синтеза будет излагаться на примере простого объекта с хорошо понятными алгоритмом и условиями работы.

Задание. Спроектировать схему кодового замка двери комнаты. Соленоид должен оттягивать ригель замка после поочередного нажатия двух (а не трех, как обычно: это для упрощения) из десяти кнопок, а именно 1 и 3. В общем случае обе цифры должны быть различными. В исходное состояние схема возвращается единичным уровнем сигнала Д, снимаемого с контакта при открывании двери.

Это и есть первичная, словесная, интуитивно ясная и воспринимающаяся как достаточно полная формулировка задания. Требуемый автомат исключительно прост, и схему его можно построить на интуитивном уровне понимания без какой-либо методики.

Проектируемая схема не может быть просто комбинационной. Это автомат, поскольку на один и тот же входной сигнал, например на нажатие кнопки 3, схема должна реагировать по-разному: как на неверный сигнал, если это первое нажатие, как на верный, если перед этим была нажата кнопка 1, и никак не реагировать после того, как соленоид уже сработал, а дверь еще не открыта.

Процесс *формализации задания* удобно разбить на этапы.

1-й этап формализации – формирование списка входных сигналов X и выходных Y, которые будут использоваться при разработке схемы. В данном случае формально на вход воздействует сигнал открытой двери Д и 10 кнопок. Однако можно ожидать, что избирательно автомат будет реагировать лишь на пять входных ситуаций: сигнал правильной первой цифры П1 (цифра 1), сигнал правильной второй цифры П2 (цифра 3), сигнал неправильной первой цифры Н1 (любая цифра, кроме 1, в том числе и 3), сигнал неправильной второй цифры Н2 (любая цифра, кроме 3, в том числе и 1) и сигнал Д. Для упрощения дальнейшей работы рационально весь набор ситуаций X, фактически воздействующих на автомат, сформировать заранее на вспомогательных логических схемах, как показано на рис. 5.53. На цепочку элементов, начинающуюся с шины питания $U_{\text{ип}}$, внимания пока обращать не нужно. Аналогично формируется массив выходных сигналов Y, который в примере представлен единственным выходным уровнем P.

2-й этап формализации – определение требуемого задачей числа состояний автомата. В данном случае решение очевидно: три со-

стояния. *Z*0 – *начальное* состояние, состояние ожидания первого правильного сигнала; *Z*1 – первый правильный сигнал получен, ожидание второго; *ZP* – оба сигнала получены, вырабатывается единичный уровень выходного сигнала *P* на усилитель соленоида замка.

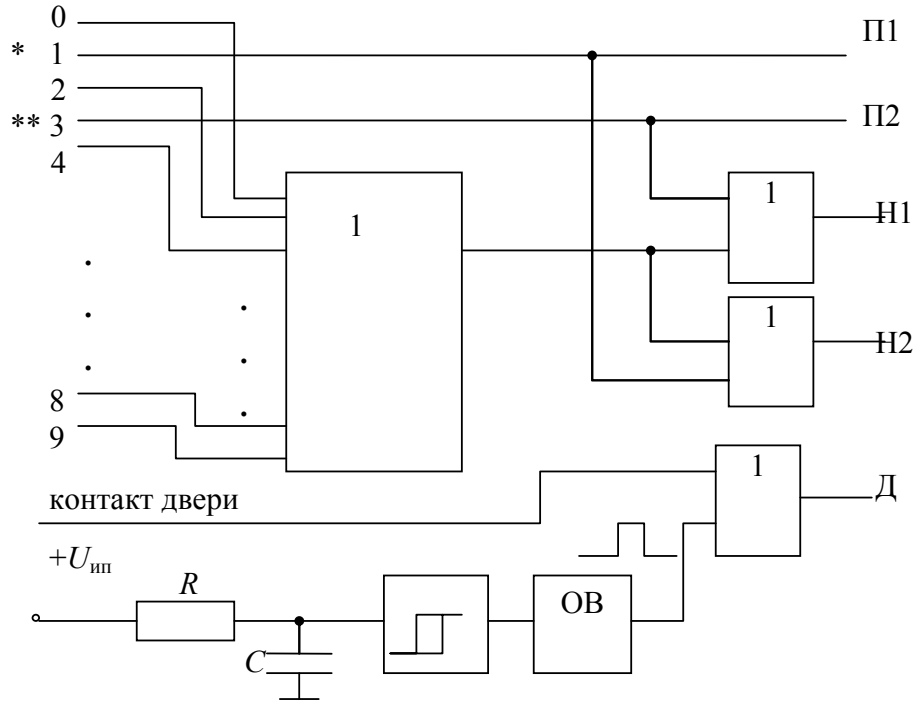


Рис. 5.53. Формирование входного набора сигналов

Если бы отпирающая последовательность состояла из трех нажатий кнопок, то добавилось бы еще одно состояние. Можно, если этого требуют условия работы, ввести еще состояние – тревога, в которое автомат переходит в случае набора неверной цифры и при этом выдает сигнал куда следует. Кстати, как автомат должен поступать в случае неправильного набора – это первый вопрос, ответа на который не было в первичной формулировке задания, который возникает при проектировании и требует выяснения у заказчика. Допустим, ответ таков: при неправильном наборе схема сбрасывается в начальное состояние.

При проектировании сложных автоматов число состояний становится очевидным далеко не сразу, а лишь в процессе изучения задания, по мере более глубокого понимания алгоритма автомата и уяснения разнообразия его реакций на одинаковые входные сигналы.

3-й этап формализации – построение предварительного, упрощенного *графа* или *таблицы* автомата (*синтез абстрактного автомата*). В примере с замком лучше начинать с графа, поскольку при числе состояний в пределах 10–15 граф обычно нагляднее таблицы.

В *графовом представлении* функционирования автомата (рис. 5.54) состояния автомата отображаются *вершинами графа*, а возможные переходы автомата из одного состояния в другое – *дугами графа*. Направления переходов обозначены стрелками. Дуги помечены входными сигналами-условиями, при которых эти переходы выполняются.

Выходные сигналы, если они связаны только с определенным состоянием, изображаются внутри кружка состояния, порождающего этот сигнал. Если кроме состояния выходной сигнал зависит еще и от входных сигналов, то он изображается выходящей из соответствующего состояния стрелкой, помеченной порождающим входным сигналом. В рассматриваемом примере разумно полагать, что выходной сигнал *P* однозначно связан с состоянием автомата *ZP*, т. е. *P* = 1, тогда, когда автомат находится в состоянии *ZP* независимо от действующих при этом входных сигналов.

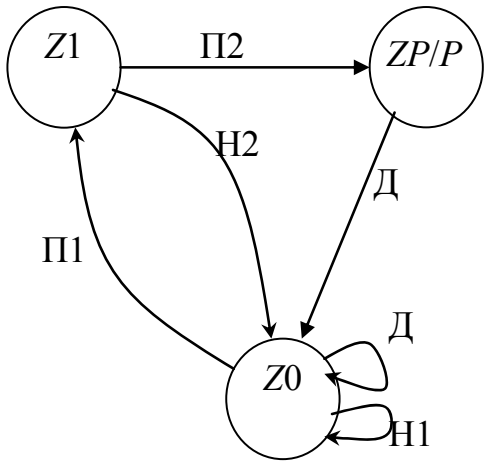


Рис. 5.54. Граф переходов автомата

Граф на рис. 5.54 есть символическое отражение словесного описания автомата. Пока на вход не поступит сигнал П1, автомат находится в начальном состоянии *Z*0. Сигнал П1 переводит его в состояние *Z*1, откуда он при поступлении неверного сигнала Н2 вернется в *Z*0, а при поступлении второго правильного сигнала П2 перейдет в состояние *ZP* разрешения открыть замок. Сигнал Д, возникающий при открывании двери, сбрасывает автомат в *Z*0, подготавливая его к следующему циклу работы. Никакие другие сигналы не выводят автомат из состояния *ZP*: пусть такова воля заказчика. В принципе можно задать и иные условия.

Для строгого указания на тот факт, что, несмотря на поступление некоторых сигналов, автомат тем не менее не изменяет своего со-

стояния, на графе переходов около этих состояний чертят петли, помеченные соответствующими сигналами. Примеры двух таких петель показаны у вершины *Z*0 (рис. 5.54). Строгое начертание графа потребовало бы еще одной петли П1 у вершины *Z*1 и трех петель – П1, П2 и Н2 – у вершины *ZP* (или хотя бы двух – П2 и Н2, поскольку сигнал П1 входит в состав Н2). Петли сильно загромождают чертеж, и их, как правило, не изображают, а лишь подразумевают, полагая, что при поступлении любого сигнала, кроме тех, которыми помечены стрелки перехода из данного состояния, автомат своего состояния не изменит. Несколько позже будет показано, что замалчивание существования петель оправдано и с позиций построения логической схемы.

Наряду с *графовым* используют *табличное описание* автомата, общепринятая форма которого приведена в табл. 5.3.

Таблица 5.3

Табличное описание автомата

Входы	**Состояния**		
	***Z*0**	***Z*1**	***ZP***
П1	*Z*1	*Z*1	*ZP*
П2	*Z*0	*ZP*	*ZP*
Н2	*Z*0	*Z*0	*ZP*
Д	*Z*0	*Z*0	*ZP*
Выход	–	–	*P*

Строки таблицы отображают входные сигналы, столбцы – состояния. На пересечении строки и столбца записывается новое состояние, в которое автомат перейдет из состояния данного столбца под воздействием входного сигнала данной строки. В последней строке приведены те выходные сигналы, которые однозначно связаны с состоянием. Если выходной сигнал связан еще и со входным, то он изображается на соответствующем пересечении и отделяется от нового состояния косой чертой, как знаменатель дроби. Если в таблице заполнены все клетки, то говорят, что автомат *определен полностью.* В задании на *недоопределенный* автомат в некоторых клетках могут стоять кресты – символы безразличия. В процессе построения логической схемы разработчик доопределяет таблицу исходя из соображений качества схемы – аналогично доопределению таблицы логической функции.

Граф и *таблица* – два взаимно эквивалентных способа формального описания поведения автомата. Каждый из них задает реак-

цию проектируемого автомата на каждый же входной сигнал однозначно, без опасности разночтения. Чтобы граф был строго эквивалентен таблице, на нем нужно изобразить в явном виде все петли.

Описанный процесс построения графа или таблицы называют этапом *абстрактного синтеза* автомата или этапом *синтеза абстрактного автомата*. *Абстрактный автомат* – это еще не устройство и даже не схема. Это лишь математическая модель, это алгоритм функционирования некоторого преобразователя кодовых последовательностей. Если входы в табл. 5.3 (или на графе) отождествить с буквами некоторого, входного алфавита, а выходы – с буквами выходного алфавита (в случае замка входной алфавит состоит лишь из двух Р и ПРОБЕЛ), то абстрактный автомат – это закон преобразования цепочек букв (т. е. слов) входного алфавита в цепочки букв (слова) выходного алфавита. Примерно такая постановка задачи и послужила в свое время исходным толчком к созданию теории автоматов. Создание абстрактного автомата – это первый ощутимый результат на пути построения схемы автомата. Дальнейшие шаги разработки автомата, приблизительно до уровня получения его функциональной схемы принято называть этапом *структурного синтеза.*

4-й этап формализации – построение полного графа (полной таблицы) автомата. Формальное описание абстрактного автомата, полученное на предыдущем этапе, как правило, оказывается недостаточно полным для того, чтобы непосредственно по нему можно было вести синтез логической схемы. Обычно требуется еще рассмотреть и формализовать следующие не учтенные абстрактной моделью моменты: вопрос о *неиспользуемых состояниях* автомата, *начальная установка* его после включения питания, возможность *совпадения входных сигналов* во времени.

Неиспользуемые (*запрещенные*) *состояния*. Пусть известно, что три рабочих состояния автомата хранятся в двухзарядном триггерном регистре. При этом четвертая из возможных комбинаций состояний триггеров не используется, в цепочке переходов автомата не участвует и о ней вроде бы можно и не говорить. Однако если из-за воздействия помехи в регистре ошибочно возникнет это четвертое состояние, автомат в нем «зависнет», и никакие внешние сигналы уже не смогут вернуть его в рабочий цикл. Говорят, что в графе переходов существует *изолированная вершина*. В грамотно спроектированной схеме изолированных вершин или циклов из нескольких таких вершин быть не должно. От неиспользуемых вершин нужно сделать «стоки» в рабочий цикл графа, и это должно быть отражено на самом графе.

В рассматриваемом примере на новом чертеже полного графа (рис. 5.55) лишняя вершина *ZЛ* уже показана явно и подключена к остальной части графа дугами Д и $П1 \cdot \overline{Н1} \cdot \overline{Д}$. Это решение допустимое и неплохое: если в автомате вдруг возникнет состояние *ZЛ*, то или первое же открывание двери вернет автомат в начальное состояние, или нажатие первой верной кнопки переведет его в *ZI*. Почему вместо простого условия перехода П1 использовано более сложное – $П1 \cdot \overline{Н1} \cdot \overline{Д}$, будет ясно немного позже.

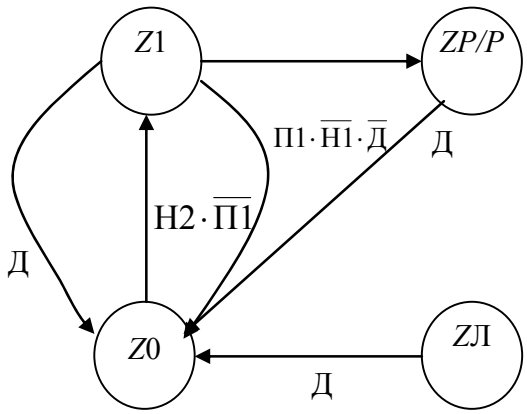


Рис. 5.55. Полный граф переходов автомата

Начальная установка. При включении питания триггеры регистра, а вместе с ними и весь автомат оказываются в случайном состоянии. При некоторых применениях допустимо устанавливать автоматы в начальное состояние *Z*0, выполнив первый холостой установочный прогон до конца цикла. Если эффект, вызываемый таким неполным циклом, начавшимся с произвольной промежуточной точки, недопустим, то автомат снабжают цепями *начальной установки* (*начального сброса*). На графе переходов для этого вводят дуги переходов от каждой вершины к нулевой, которые помечают еще одним специально введенным входным сигналом – *сигналом начального сброса.*

В примере с кодовым замком начальная установка путем холостого прогона чревата неприятностями. Если автомат при включении питания с равной вероятностью устанавливается в любое состояние, в том числе и в *ZP*, то злоумышленник, выворачивая в обеденный перерыв на несколько секунд пробки и пробуя затем дверь, будет проникать в любую защищенную комнату в среднем за четыре попытки. Для начальной установки замка специальной цепи сброса можно не вводить. Роль сигнала сброса может выполнить сигнал Д: дуги Д от вершин *ZP* и *ZЛ* к *Z*0 уже есть и если добавить еще одну дугу Д от *Z*1

к *Z*0, то цель, которой служит автомат, не пострадает: хотя открываемая неожиданно выходящим из комнаты дверь и прервет интеллектуальный процесс набора кода другим лицом, желающим попасть в комнату, но в результате дверь все равно окажется открытой и желание набиравшего – выполненным. Отметим, что решение использовать тракт Д еще и таким образом требовало знания реальных условий работу автомата, а не вытекало непосредственно из текста заданий. Дуга от *Z*1 к *Z*0 показана на рис. 5.55. Вообще, в цифровой аппаратуре цепи начального сброса автоматов делают как с ручным запуском, вводя специальную кнопку СБРОС, так и с автоматическим, от включения питания.

Последний вариант показан на рис. 5.53. После включения питания *RC*-цепочка с постоянной времени порядка секунды через триггер Шмитта запускает одновибратор ОВ, импульс которого через элемент ИЛИ поступает в тракт Д и устанавливает автомат в начальное состояние. И еще один метод: в электрических схемах БИС триггеры иногда выполняют немного асимметричными, так, чтобы в процессе появления напряжения питания они сами устанавливались в 0.

Совпадение входных сигналов во времени. Если автомат создается не для обработки текстов, где буквы поступают строго по одной, а как управляющее устройство, которое должно реагировать на изменения окружающей обстановки, то возможности совпадения во времени входных сигналов, по крайней мере от независимых источников, исключить нельзя. Однако в таблице переходов и выходов абстрактного автомата реакция на возможные совпадения сигналов никак не отражена, поэтому схема, построенная по такой таблице, может реагировать на совпадения входных сигналов любым, самым неожиданным образом. Чтобы этого не произошло, разработчик должен предусмотреть адекватные реакции автомата на эти случаи и отразить их в таблице или на графе переходов.

Формальным приемом учета возможности совпадения сигналов, который часто рекомендуют к применению на этапе структурного синтеза автомата, является переход к новому набору абстрактных входных сигналов, число которых равно числу всех комбинаций реальных сигналов. В случае замка пять входных сигналов с помощью дешифратора можно превратить в 2^5, т. е. в 32 абстрактных сигнала, которые уже будут появляться строго по одному. Для описания автомата в этом случае потребуется таблица в 32 строки, столько же дуг

(включая петли) будет и у графа. Соответственно усложнится процедура синтеза схемы.

Этот прием оправдан и даже необходим, когда разработчик приписывает определенный смысл именно *комбинациям* входных сигналов, как, например, при кодировании A-разрядным кодом команды 2^k запускаемых этой командой различных операций. Однако в устройствах автоматики намного чаще определенный смысл приписывается не комбинации сигналов, а именно отдельному сигналу (перегрев, ошибка, дверь открыта и т. п.) вне связи с другими. Поэтому наиболее адекватной реакцией при совпадении сигналов часто оказывается реакция на наиболее важный из них как на единственный в данный момент, а все остальные сигналы игнорируются. Если приоритет не определен четко самим содержанием задачи, он может быть назначен произвольно. Ранжирование входных сигналов по приоритетам снимает с разработчика большой труд по осознанию требуемой акции автомата на каждую комбинацию различных входных сигналов и существенно упрощает как таблицу, так и схему автомата. При этом ранжировать требуется не все сигналы, а лишь те, которые порождают *конфликтные ситуации*, т. е. при своем совпадении вызывают на графе переходов ситуации неопределенности или генерации. Для каждого состояния автомата конфликтующие группы сигналов в общем различны и начинать работу нужно с выявления этих групп.

В рассматриваемом примере в состоянии $Z0$ конфликтуют две пары сигналов: П1 с Д и П1 с Н1. При одновременном появлении любой пары сигнал П1 будет стремиться перевести автомат в состояние $Z1$, а Д или Н1 – оставить в $Z0$. По смыслу работы замка сигналам Д и Н1 нужно назначить более высокий приоритет, чем сигналу П1, т. е. разрешить переход в $Z1$ по сигналу П1 лишь при условии, что дверь закрыта ($\overline{\text{Д}}$) и кроме правильной кнопки не нажата больше никакая другая, а тем более сразу все – ладонью ($\overline{\text{Н1}}$). Для этого переход $Z0$–$Z1$, который на рис. 5.54 был помечен безусловным сигналом П1, на уточненном графе (рис. 5.55), помечается уже конъюнкцией $\text{П1} \cdot \overline{\text{Н1}} \cdot \overline{\text{Д}}$. Снова для формализации задания потребовалось знание условий работы объекта.

При рассмотрении вершин $Z0$ и $Z1$ конфликтующей парой оказываются сигналы П1 и Н2. Оба они порождаются нажатием кнопки 1 (рис. 5.53), но один из них вызывает переход $Z0$–$Z1$, а второй – обратно, $Z1$–$Z0$. Все время, пока нажата кнопка 1, схема будет находиться в режиме генерации. Чтобы состояние $Z1$ сделать устойчивым, необхо-

димо сигналу П1 в этом состоянии дать более высокий приоритет, чем сигналу Н2. Для этого дуга *Z*1–*Z*0 помечается функцией $Н2 \cdot \overline{П1}$, и тогда вернуться в *Z*0 от неверно нажатой кнопки автомат сможет лишь после отпускания кнопки П1.

В состоянии *Z*1 сигналы Д и $Н2 \cdot \overline{П1}$ не конфликтуют, поскольку заняты одним общим делом и совпадение их времени не вызовет осложнений. Поэтому здесь введений приоритетов не требуется.

В неполном графе, показанном на рис. 5.54, переход *Z*1–*ZP* выполняется по условию П2. Однако в этом случае нажимать кнопку П2 и переходить в *ZP* можно, не отпустив предварительно кнопки П1. Нажатая кнопка П1 не дает автомату сброситься из *Z*1 в *Z*0 даже при одновременном нажатии любой неверной кнопки. Значит, фактически существенным оказывается знание лишь первой цифры. После ее нажатия злоумышленник, не отпуская первой кнопки, нажимает ладонью все остальные. Неверная кнопка замка не сбросит, а верная его откроет. Чтобы вторая цифра была столь же значимой, что и первая, в графе на рис. 5.55 переход *Z*1–*ZP* делается по условию $П2 \cdot \overline{Н2}$, т. е. переход в *ZP* по нажатию П2 осуществится, только если П1 отпущена и вообще не нажата никакая другая кнопка, кроме П2.

В *Z*1 конфликт возникает между нажатием кнопки П2 (сигнал $П2 \cdot \overline{Н2}$) и сигналом Д. Конфликт ликвидируется, если одному из сигналов, например Д, назначить более высокий приоритет. Это отражено в условии перехода *Z*1–*ZP*: $П2 \cdot \overline{Н2} \cdot \overline{Д}$.

Введение приоритетов позволило ликвидировать все возможные сбои при совпадении входных сигналов во времени, не увеличив при этом числа дуг графа. Граф, показанный на рис. 5.55, считается окончательным. Цель его – проиллюстрировать сам процесс формализации задания. Характерно, что для построения графа понадобился ряд сведений о реальных условиях работы автомата, которые не были изложены в первоначальном задании и, вообще говоря, требовали согласования с заказчиком. Построенный граф представлен в табличной форме в табл. 5.4. Поскольку входные сигналы теперь уже не абстрактные, таблица не получается столь изящной, как табл. 5.3. Построением полной таблицы автомата, охватывающей все его возможные (в том числе и нежелательные) состояния и все комбинации входных сигналов, способных повлиять на работу, и заканчивается эта формализации задания.

Таблица 5.4

Полная таблица автомата

№	Аргументы						Функции	
	Состояние в данный момент	Входы автомата					Следующее состояние	Выход *P*
		П1	Н1	П2	Н2	Д		
1	*Z*0	1	0	*X*	*X*	0	*Z*1	0
2	*Z*1	*X*	*X*	*X*	*X*	1	*Z*0	0
3	*Z*1	0	*X*	*X*	1	*X*	*Z*0	0
4	*Z*1	*X*	*X*	1	0	0	*ZP*	0
5	*ZP*	*X*	*X*	*X*	*X*	1	*Z*0	1
6	*Z*Л	*X*	*X*	*X*	*X*	1	*Z*0	0
7	*Z*Л	1	0	*X*	*X*	0	*Z*1	0

В табл. 5.4 символ *X* обозначает, что как следующее состояние автомата, так и его выходной сигнал, относящиеся к данной строке, не должны зависеть от значения входного сигнала того столбца, в котором стоит этот символ.

Регистр состояний автомата

К вопросам разработки регистровой части автомата принято относить выбор числа триггеров регистра и способ кодирования состояния. Минимально возможное число триггеров равно ближайшему большему целому от двоичного логарифма числа состояний. Максимальное число триггеров равно числу состояний, при этом каждое состояние кодируется единицей в одном из триггеров. В среднем с ростом числа триггеров в пределах указанного диапазона уменьшается число логических элементов, требующихся для дешифрации состояний. При максимальном числе триггера дешифрации вообще не требуется. Кроме соображений экономичности увеличению числа триггеров сверх необходимого минимума способствует использование микросхем регистров с числом разрядов, кратным четырем.

При кодировании состояний, если нет других соображений, вершины графа переходов нумеруются в произвольном порядке и номера кодируются по двоичной системе. Однако объем комбинационной схемы можно уменьшить, если для вершин, связанных большим числом дуг переходов, подбирать коды, отличающиеся возможно меньшим числом разрядов. Тогда для обеспечения каждого перехода придется заводить сигналы на входы меньшего числа триггеров. Иногда для уменьшения числа переключаемых триггеров увеличивают общее их число. Алгоритма нахождения наиболее экономичного решения, кроме полного перебора вариантов, не существует. Разработчики используют метод проб и ошибок и свой опыт.

Распространено мнение, что в автомате не будет гонок и он может быть сделан асинхронным, без системы синхронизации, если его состояния закодировать так, чтобы при любом переходе изменялось значение только одного разряда. Это так называемое противогоночное кодирование состояний автомата. В случае разветвленного графа такое кодирование требует введения дополнительных триггеров в регистр.

В современной элементной базе задержка триггера соизмерима с задержкой одного логического элемента, поэтому задержка комбинационной схемы средней сложности превышает задержку триггера. В этих условиях основное число помех гоночного типа зарождается в недрах КС автомата и противогоночное кодирование его состояний здесь бессильно. Построение КС, в которых гонки невозможны, весьма трудоемко и реально выполнимо лишь для очень простых схем. Кроме того, такие схемы требуют больше оборудования, и эта разница уже при достаточно простых схемах превышает затраты оборудования на генератор синхросигналов.

Учитывая все сказанное, число триггеров регистра состояний кодового замка принимается равным двум. Кодирование его состояний: *Z*0 = 00; *Z*1 = 01; *ZP* = 11; *ZЛ* = 10. Коды для *Z*0 и *Z*1 и *Z*1 и *ZP* выбраны отличающимися лишь одним разрядом (соседнее кодирование), чтобы уменьшить, как уже говорилось, объем логических схем: именно эти состояния связаны наибольшим числом дуг. Поскольку устройство легко разместится на одной плате, т. е. расфазировки активных фронтов можно не опасаться, система синхронизации выбирается однофазной, а триггеры – соответственно, синхронные *JK*-триггеры. Генератором синхросигналов может служить мультивибратор на логических микросхемах. Входные сигналы привязываются к синхросерии с помощью схем-синхронизаторов. Дребезга контактов схема не боится: таблица автомата построена так, что снятие, а потом и повторная подача входного сигнала не нарушают работы автомата.

ГЛАВА 6. ГЕНЕРАТОРНЫЕ УСТРОЙСТВА

6.1. Импульсные устройства

6.1.1. Схемы приема внешних сигналов

Импульсная схема отличается от цифрового устройства тем, что в ней главными являются не булева алгебра или арифметика работы устройства, не коды на его выходах, а длительность формируемых этой схемой импульсов, образование выходных фронтов, более коротких, чем на входе, задержки этих фронтов, их дифференцирование, ограничение и т. п.

Внешние сигналы можно разделить на:

– сигналы от кнопок, переключателей, контактов реле;

– сигналы от аналоговых датчиков, генераторов и усилителей;

– сигналы от удаленных на десятки и сотни метров источников двоичных данных;

– сигналы от микросхем другого типа логики.

При замыкании контактов обычно имеется дребезг, т. е. контакт вибрирует от удара при замыкании. После первого касания контакт размыкается, затем снова замыкается и так несколько раз. Частота коммутации при дребезге зависит от жесткости пружины и массы контакта и обычно лежит в пределах сотен герц – единиц килогерц.

При необходимости избавиться от дребезга ставят *RS*-триггеры.

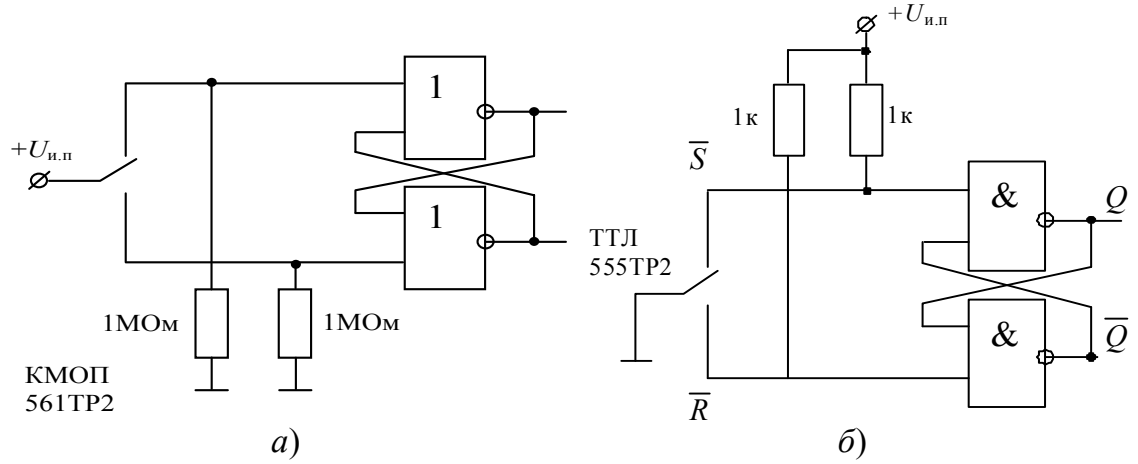


Рис. 6.1. Примеры схем избавления от дребезга контактов:
а – для КМОП; *б* – для ТТЛ

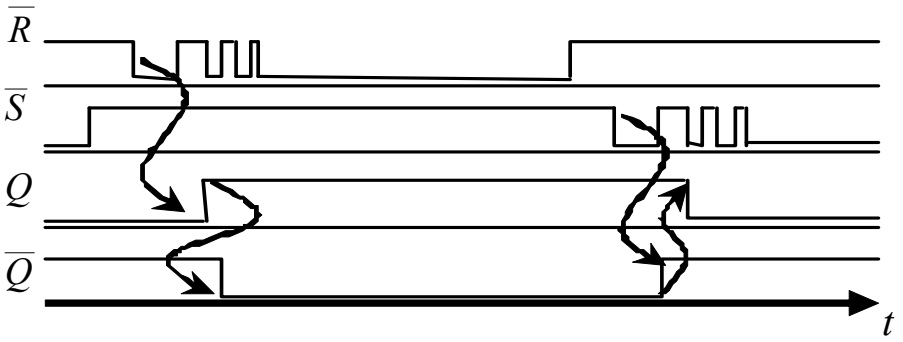


Рис. 6.2. Временные диаграммы работы триггера

У сигналов, *поступающих от аналоговых датчиков*, усилителей и генераторов, могут быть фронты более пологие, чем допустимо для данных цифровых микросхем.

При этом на входных цифровых микросхемах могут возникнуть «вспышки» генерации на фронтах, эти «вспышки» генерации могут перегреть кристалл микросхемы, создают сильные помехи для работы других микросхем.

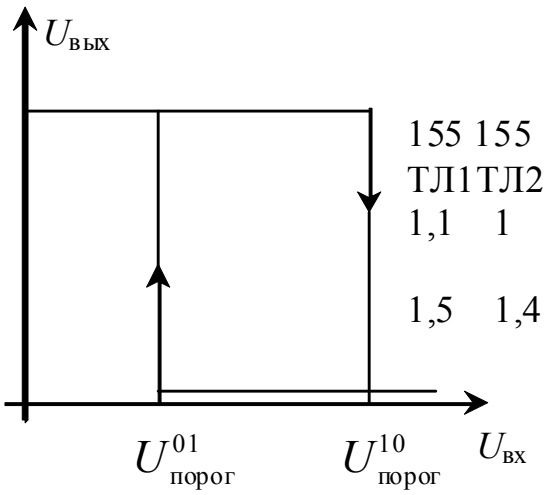


Рис. 6.3. Передаточная характеристика триггера Шмитта

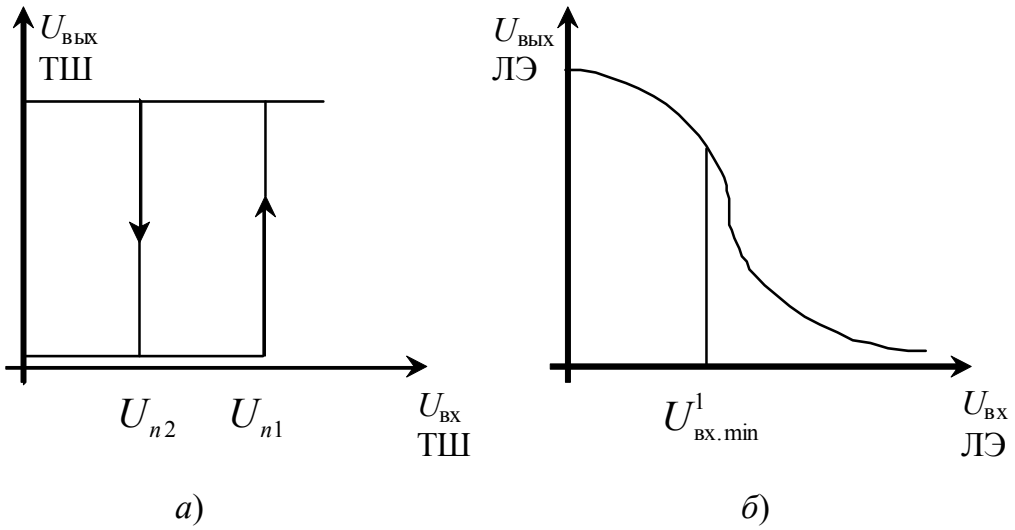
Чтобы независимо от крутизны фронтов входного сигнала получить четкий и крутой фронт на выходе, применяют триггеры Шмитта, которые имеют передаточную характеристику с гистерезисом (рис. 6.3).

В табл. 6.1 приведены значения пороговых напряжений передаточной характеристики триггера Шмитта для ТТЛ и для КМОП микросхем.

Таблица 6.1

Значения пороговых напряжений триггера Шмитта

Серия ИМС	ТТЛ				КМОП		
	155	**555**	**531**	**1006 ВИ1**	**564**		
$U^{01}_{\text{порог}}$ В	0,9	0,8	1,2	$\frac{U_{\text{и.п}}}{3}$	2,2	4,2	6
$U^{10}_{\text{порог}}$ В	1,7	1,6	1,8	$\frac{2U_{\text{и.п}}}{3}$	2,85	5,2	7,3



а) *б*)

Рис. 6.4. Различие передаточных характеристик: *а* – триггера Шмитта; *б* – логического элемента

На рис. 6.5 представлены временные диаграммы выходных напряжений логического элемента и триггера Шмита при изменении входного напряжения.

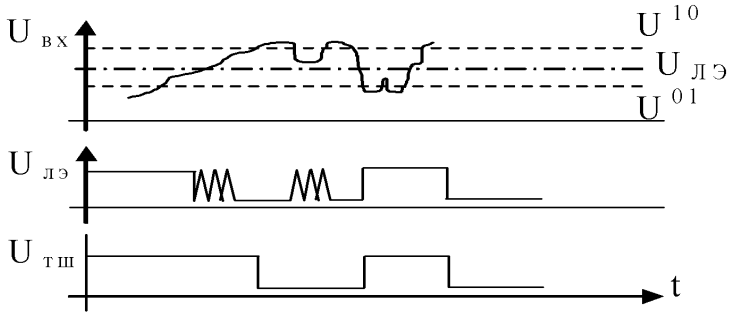


Рис. 6.5. Временные диаграммы выходных напряжений логического элемента и триггера Шмитта

Расчет пороговых напряжений можно произвести по приведенным ниже формулам:

$$U_{п1} = \frac{R1 + R2}{R2} U^{1}_{\text{вх. min}};$$

$$U_{п2} = \frac{R1 + R2}{R2} U^{1}_{\text{вх. min}} - \frac{R1}{R2} U_{\text{и.п}};$$

$$U_{п1} = (1 + \frac{R1}{R2}) \frac{U_{\text{и1.п}}}{2};\ U_{п2} = (1 - \frac{R1}{R2}) \frac{U_{\text{и1.п}}}{2}.$$

Для ТТЛ берут R_1 = 330; R_2 = 2,2…3,3 кОм.

За счет положительной обратной связи триггер Шмитта переключается лавинообразно, обеспечивая этим крутые фронты сигнала на своем выходе (без вспышек генерации), а благодаря гистерезису – разности порогов срабатывания – он защищен от не очень больших помех.

Эти положительные свойства триггеров Шмитта обуславливают применение их *при приеме сигналов от удаленных источников данных*, когда возможны выбросы, всплески на фронтах, звон из-за неидеального согласования сопротивлений передатчика, линии и приемника, искажение фронтов из-за неполного согласования спектра сигнала с частотной характеристикой линии.

Как известно, все обычные микросхемы не допускают подачу на вход сигналов, напряжение которых выходит за рамки питания корпуса. Это объясняется возможностью перегрузки диодов, входящих в состав этих микросхем либо с защитными, или антизвонными, целями, либо образующихся по технологии изготовлении микросхем. Если входной сигнал может выходить за пределы ноль-питания, то надо последовательно с входом ставить резистор, ограничивающий входной ток на уровне ≤10 мА, а если сигнал больше питания, то у ТТЛ $J_{\text{мак.вх}} < 1$ мА для 155 и 531серий и $J_{\text{мак.вх}} < 0{,}1$ мА 555,1531,1533.

Чтобы избежать перегрузок схем КМОП, следует всегда подавать питающее напряжение на них раньше, чем входные сигналы.

Для перехода от микросхем ТТЛ *к* КМОП при питании и тех и других от + 5 В можно использовать ТТЛ схемы со стандартным выходом, но для надежности рекомендуется между выходом ТТЛ и источником питания включать резистор величиной порядка 3,3 кОм. Если схема КМОП питается напряжение больше 5 В, то передающая ТТЛ схема должна иметь выходы с открытым коллектором, при этом выходной транзистор должен быть высоковольтным ($U \geq$ 15 В). Это схемы 155ЛА7, ЛА8, ЛА10, ЛА11, ЛА13, ЛА18, ЛН2, ЛН3, ЛН5.

Обратный переход от КМОП к ТТЛ сделать сложнее, так как большинство КМОП ИС имеют малые выходные токи и выходы их не могут быть нагружены даже на один ТТЛ вход серии 155. Но есть три обычные ИС 561ЛН1, ЛН2, которые можно нагрузить на 2 входа 155 серии ТТЛ ($J_{\text{вых.КМОП}} \geq 3{,}2$ мА) и 564ЛА10 – на 10 ТТЛ входов 155 серии. Кроме них в КМОП сериях есть довольно много специализированных преобразователей уровня: 176ПУ1, ПУ2, ПУ3, ПУ5, 561ПУ4, 564ПУ6,7,8.

Для перехода от ЭСЛ к ТТЛ и обратно надо обязательно пользоваться специальными микросхемами – преобразователями уровня, входящими в состав серий ЭСЛ (К500ПУ124, ПУ125).

6.2. Генераторы импульсов

6.2.1. Генераторы импульсов на логических элементах

Большинство импульсных устройств на логических элементах основано на том, что логические элементы – это своего рода усилители с коэффициентом усиления 20…100 и с частотой среза у ТТЛ схем $f_{\text{среза}} \approx 5\ldots50$ МГц, а у КМОП $f_{\text{среза}} \approx 0{,}1\ldots1$ МГц. Аналитический расчет линейного режима цифровой микросхемы нецелесообразен, так как конкретные характеристики элементов могут существенно отличаться от типовых, ведь завод-изготовитель эти характеристики не регламентирует. На практике нужный режим находят подбором сопротивления внешних резисторов или введением стабилизирующих обратных связей.

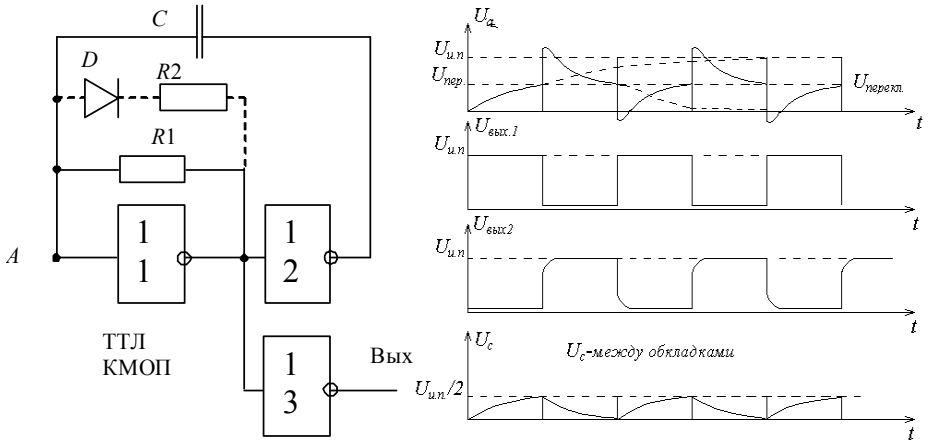


Рис. 6.6. Генератор импульсов: *а* – схема генератора с конденсатором в цепи обратной связи; *б* – временные диаграммы его переключений

Для ТТЛ R_1 выбирают R_1 = 150…680 Ом, для КМОП R_1 = = 10 кОм…10 МОм. Резистор R_1 выполняет две функции: смещает рабочую точку логического элемента 1 на крутой участок передаточной характеристики, обеспечивая этим мягкое самовозбуждение, и вместе с конденсатором C служит времязадающим элементом. Длительность каждого полупериода колебаний $T1$ и $T2$ примерно равна $2R_1C$.

Для ТТЛ микросхем этот процесс несколько ассиметричен, поскольку при входном напряжении ЛЭ1 большем, чем напряжение переключения $U_{\text{перекл}}$, входной ток его не превышает десятков микроампер – это ток утечки закрытого эмиттерного перехода МЭТ, т. е. среднее входное сопротивление ЛЭ1 здесь порядка 100 кОм. При входном напряжении меньшем, чем напряжение переключения, входной ток возрастает почти до 1 мА, следовательно, среднее входное сопротивление здесь порядка 1 кОм, и низкое входное сопротивление ЛЭ, действуя параллельно времязадающему резистору R_1, уменьшает время заряда конденсатора C в полупериод $T2$ на 10–20 % (по сравнению с $T1$). Для выравнивания длительностей полупериодов $T1$ и $T2$ иногда параллельно резистору R_1 подключают цепочку из диода D и резистора R_2, где R_2 выбирают в 5 и 10 раз большем, чем R_1.

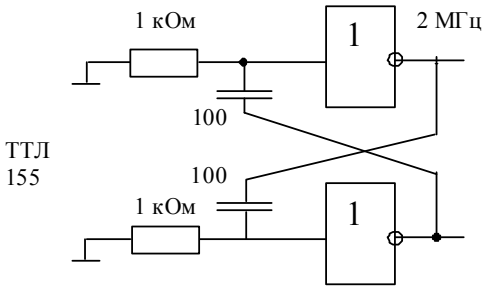


Рис. 6.7. Схема генератора на симметричном мультивибраторе

В данном симметричном мультивибраторе (рис. 6.7) нет отрицательной обратной связи для стабилизации рабочей точки ЛЭ на крутом участке переходной характеристики, чтобы обеспечить мягкое самовозбуждение, как в предыдущей схеме (рис. 6.6).

В данной схеме прохождение постоянной составляющей входного тока ЛЭ через резистор 1 кОм создает на нем падение напряжения порядка 10,83, это и есть смещение рабочей точки на сравнительно крутой участок переходной характеристики. Невысокая стабильность рабочей точки такого генератора приводит к нестабильному самовозбуждению, поэтому данную схему не рекомендуется использовать, несмотря на то что она довольно часто встречается в литературе.

Нестабильность частоты приведенных ранее генераторов имеет значение 10–20 %. Несколько уменьшить нестабильность частоты можно, если ввести между инверторами резисторы (рис. 6.8), которые фиксируют коэффициенты усиления каскадов и уменьшают влияние входного сопротивления и усиления каскадов на частоту генерируемых импульсов.

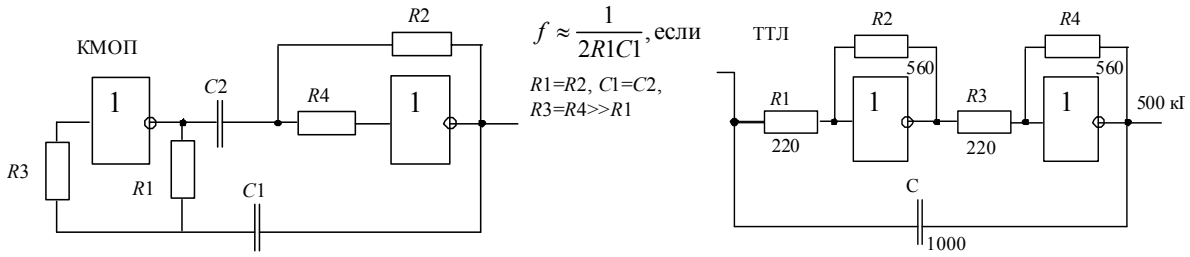
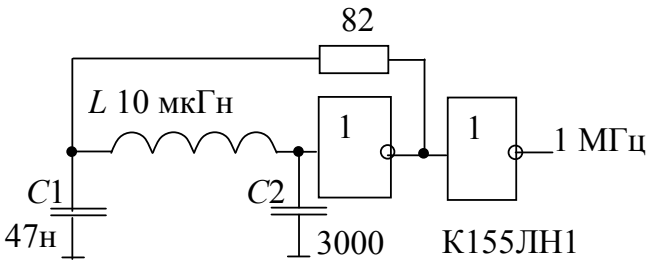


Рис. 6.8. Улучшенные схемы генераторов

Еще лучшей стабильностью частоты – примерно 10^{-3} – обладают генераторы на основе линий задержки или колебательного контура с катушкой индуктивности и конденсаторами (рис. 6.9).



$$f \approx \frac{1}{2\pi\sqrt{LC_э}};\ C_э = \frac{C1C2}{C1 + C2}.$$

Рис. 6.9. Схема генератора на основе колебательного контура

Наилучшую стабильность частоты имеют генераторы с кварцевым резонатором вместо времязадающего конденсатора.

Резисторы R_1 и R_2 выбирают так же, как и для генератора с емкостью в обратной связи. Фильтрирующий конденсатор $C_ф$ ставят в ТТЛ схемах на частотах ниже 1 МГц для предотвращения звона на фронтах импульсов (против вспышек генерации частотой ≈ 30 МГц).

Переходной конденсатор $C_{пер}$ выбирают из расчета: $X_{cпер} = \frac{1}{\omega_p C_{пер}} << R2$ на рабочей частоте ω_p.

Подстрочный конденсатор $C_{\text{подстр}}$ позволяет точно подстроить частоту генератора в небольших пределах.

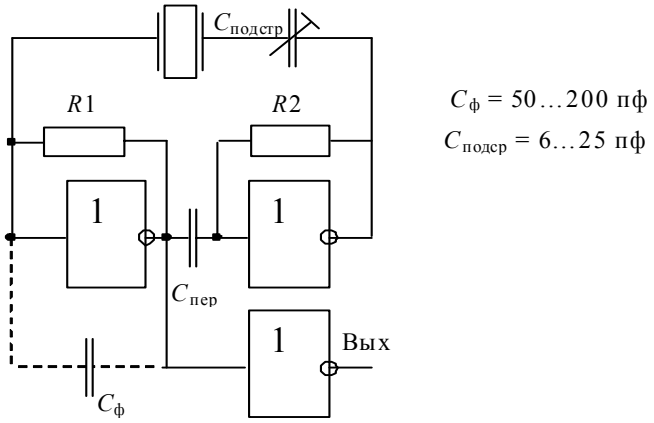


Рис. 6.10. Схема генератора с кварцевым резонатором

Обычные кварцевые резисторы имеют относительную нестабильность частоты резонанса порядка 10^{-6}. Основным дестабилизирующим фактором является зависимость частоты кварца от его температуры. Так, очень распространенный в часах кварцевый резонатор РК72 на 32,786 кГц имеет допустимое относительное изменение частоты $\frac{\Delta f}{f} = 36 \cdot 10^{-6}$ в диапазоне температур 1…50 ºС (допустимое относительное изменение частоты колебаний за первый год $\pm\ 3 \cdot 10^{-6}$).

Это значит, что часы, работающие с таким кварцем при температуре 0…50 ºС, могут иметь неточность хода за сутки: $\Delta T = T\frac{\Delta f}{f} = 86400 \cdot 36 \cdot 10^{-6} = 3{,}1$ с, где $T = 24 \cdot 60 \cdot 60 = 86400$ с.

Частотомеры обычно имеют более высокую стабильность частоты опорного генератора, порядка 10^{-8}, для этого кварцевые резонаторы, а то и опорные генераторы, целиком термостатируют, т. е. помещают в термостат – сосуд Дюара (термос), подогревают до температуры ≈ 60 ºС и поддерживает эту температуру с точностью до 0,1…0,001 ºС.

6.2.2. Формирователи коротких импульсов

Простейшим формирователем коротких импульсов длительностью t_2, образующихся на фронтах 01 входных импульсов, которые имеют длительность $t_1 > t_2$, является дифференцирующая *RC*-цепочка (рис. 6.11).

В ТТЛ-схемах таких формирователей резистор R должен быть достаточно мал, чтобы с учетом падения напряжения на нем от протекания входного тока ЛЭ $I^0_{\text{вх.max}} = 1{,}6\,m\text{A}$ входное напряжение этого ЛЭ в отсутствие сигнала не вышло за пределы допустимого $I^0_{\text{вх.max}} = 0{,}8$ В, т. е. $R \le \dfrac{0{,}8}{1{,}6} = 0{,}5\,\text{кОм}$.

Длительность импульсов t_2 в таких формирователях на 155 серии должна быть не более 200 нс, чтобы не возникло вспышек генерации (так называемого «звона») на выходе ЛЭ2 из-за слишком полого фронта 10 на его входе.

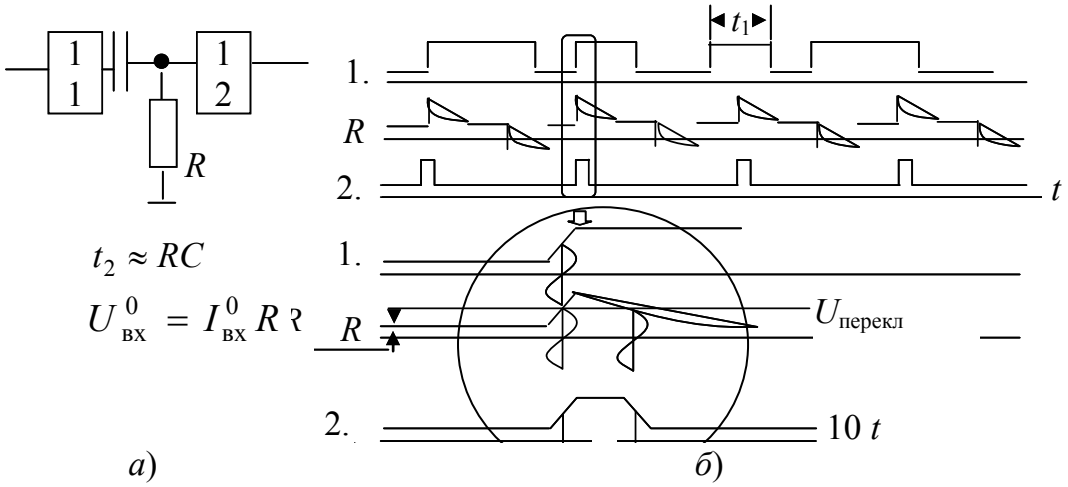


Рис. 6.11. Формирователь коротких импульсов: *а* – на основе *RC*-цепи; *б* – его временные диаграммы

Если ЛЭ2 в схеме на рис. 6.11 – триггер Шмитта, то длительность может быть любой, но не больше длительности входных импульсов.

Для формирования коротких импульсов на фронтах 01 входных импульсов можно также использовать схему с *RC*-интегратором.

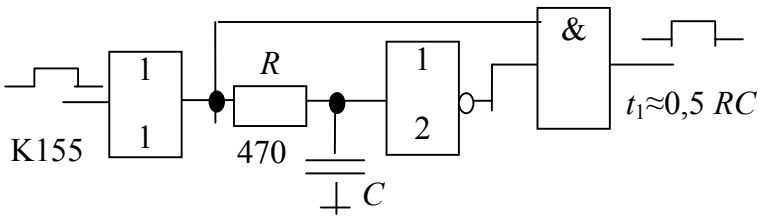


Рис. 6.12. Формирователь коротких импульсов на основе схемы с *RC*-интегратором

На рис. 6.12 резистор должен быть еще меньше, чем в предыдущей схеме (рис. 6.11), так как резистор подключен не к корпусу, а к выходу предыдущего ЛЭ, у которого выходное напряжение лог.0 $U^0_{\text{вх.max}} = 0{,}4\,\text{В}$.

Требования к длительности t_2 те же, что и в предыдущей схеме.

6.3. Одновибраторы

6.3.1. Схемы одновибраторов

Если от коротких входных импульсов длительностью t_1 надо получить более длинные импульсы t_2, то используют *одновибраторы*. От простых формирователей, рассмотренных ранее, они отличаются наличием положительной обратной связи, обеспечивающей лавинообразные процессы переключения, чем достигается большая крутизна фронтов и отсутствие на них вспышек генерации при длительности выходных импульсов вспышек, чем допустимо для предыдущих формирователей.

Одновибраторы обычно делают на основе *RS*-триггеров, которые входят в состав различных серий ИМС и могут быть собраны из логических элементов. В одновибраторах либо одна из перекрестных связей *RS*-триггера заменяется *RS*-цепью, либо *RC*-цепочка образует дополнительную, третью обратную связь. Длительность выходных импульсов определяется по приведенной ниже формуле:

$$t_2 = RC \ln \frac{U^1_{\text{вых}} - U^0_{\text{вых}}}{U_{\text{перекл}} - U^0_{\text{вых}}} \approx 0{,}7\ RC.$$

Если вскоре после образования выходного импульса такого одновибратора на вход подать еще один импульс запуска, то длительность нового выходного импульса будет короче, чем у предыдущего настолько, насколько не завершился процесс зарядки конденсатора к началу нового импульса, т. е. не закончился процесс восстановления.

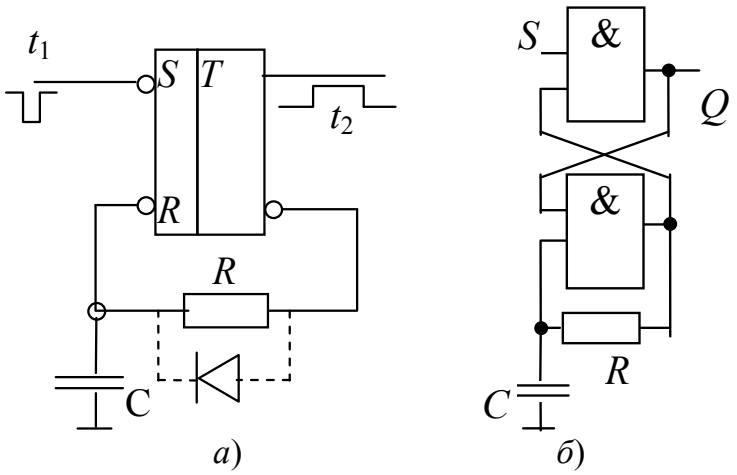


Рис. 6.13. Одновибраторы: *а* – на основе *RS*-триггера; *б* – на основе логических элементов

Чтобы длительность нового выходного импульса уменьшилась не более чем на 10 % по отношению к предыдущему, следует период входных импульсов иметь не менее чем $t_2 + 3RC$, т. е. время восстановления $t_{\text{восст}} = 3RC$. Для уменьшения времени восстановления параллельно резистору ставят диод (рис. 6.13, *а*). На рис. 6.14 изображены временные диаграммы работы одновибратора на основе *RS*-триггера.

На рис. 6.15 приведены примеры схемной реализации одновибраторов на логических элементах.

На рис. 6.16 приведены примеры схемной реализации одновибраторов на основе *D*-триггера.

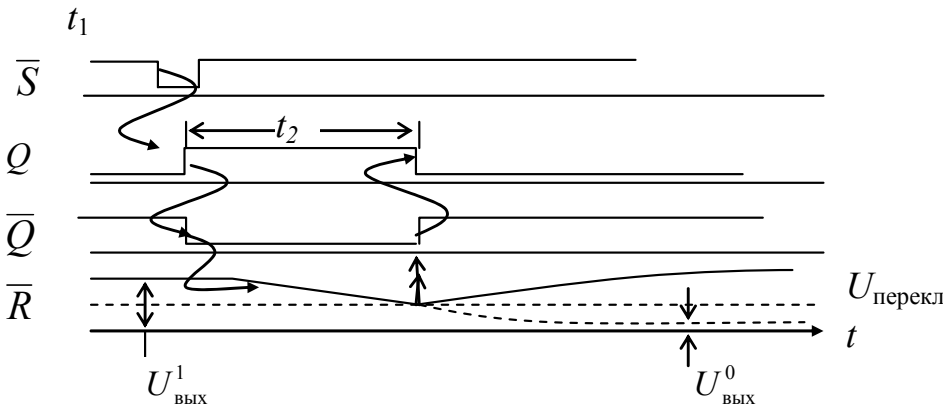


Рис. 6.14. Временные диаграммы работы одновибратора на основе *RS*-триггера

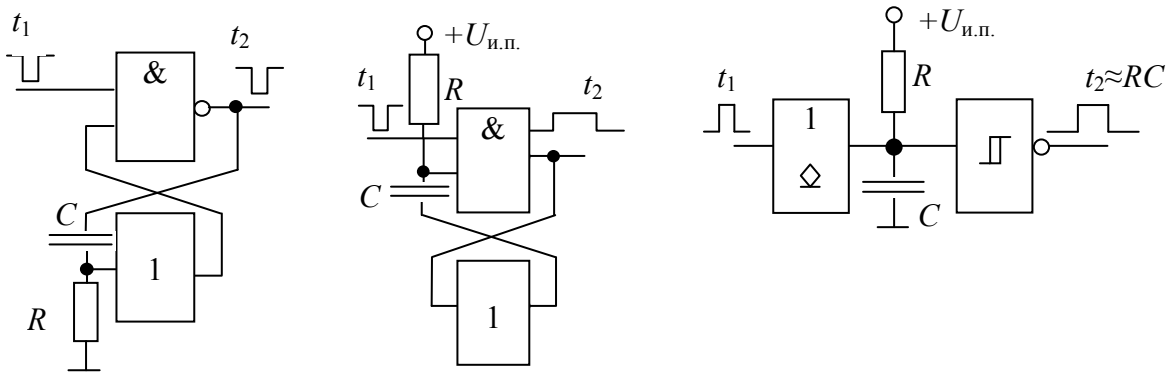


Рис. 6.15. Примеры реализации одновибраторов на логических элементах

Если на плате имеются свободные *D*-триггеры, то на них можно собрать такие одновибраторы.

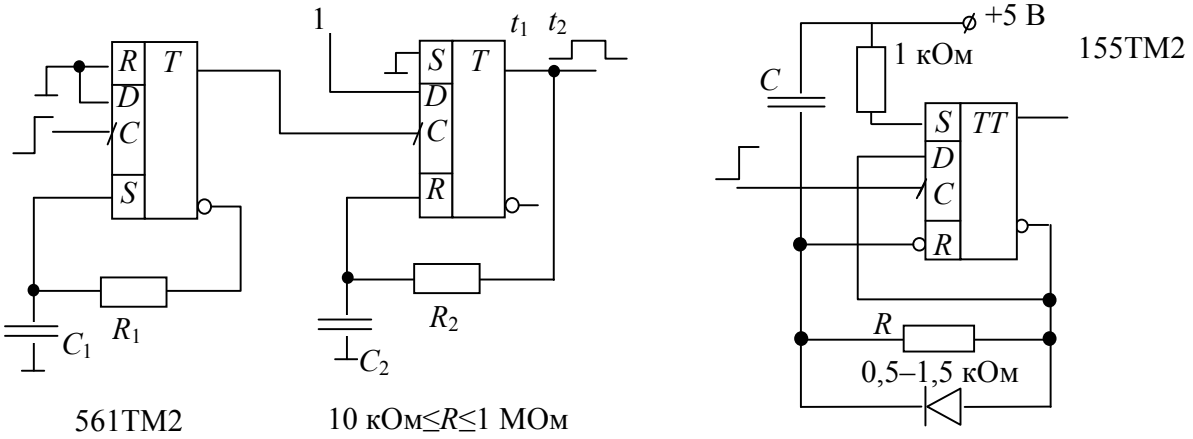
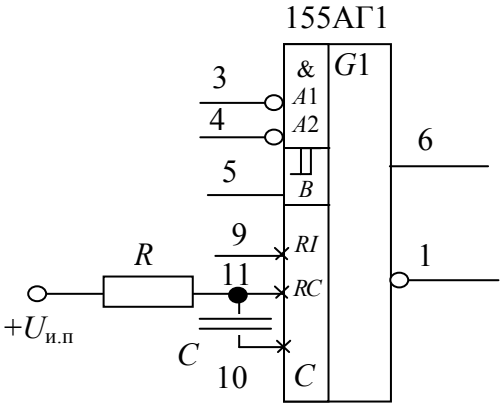


Рис. 6.16. Примеры реализации одновибраторов на основе *D*-триггера

6.3.2. Микросхемы одновибраторов ТТЛ

Специализированные микросхемы-одновибраторы – таймеры – выпускают в различных сериях ТТЛ и КМОП. Они обладают широкими функциональными возможностями, но самое главное, имеют значительно меньшие погрешности длительности выходных импульсов, слабую зависимость длительности от температуры, питающего напряжения, от времени и от замены микросхемы. Так, для 155АГ1 приводятся данные о погрешности $t_{имп}$ порядка 0,5 %, тогда как обычные одновибраторы, собранные на ЛЭ И–НЕ, ИЛИ–НЕ, *D*-триггерах или триггерах Шмитта, имеют погрешность длительности импульса порядка 10–20 %.



а)

*A*1	*A*2	*B*	*Q*
0	*X*	1	0
X	0	1	0
X	*X*	0	0
1	1	*X*	0
1	↓	1	⎍
↓	*X*	1	⎍
0	0	↑	⎍
X		↑	⎍

б)

Рис. 6.17. Одновибратор 155АГ1: *а* – условное графическое обозначение; *б* – таблица переключений

Одновибратор 155АГ1 может срабатывать от положительных фронтов по входу *B* или от обратных им импульсов по инверсным входам *A*1 и *A*2 в соответствии с таблицей.

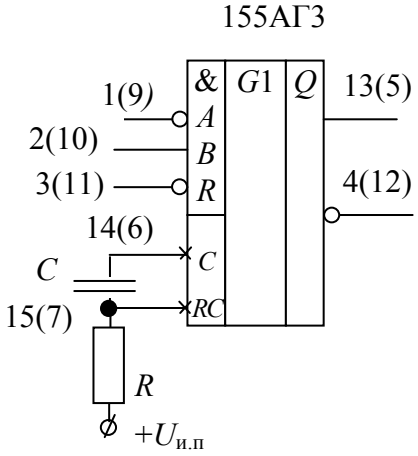
Входная схема с триггером Шмитта обеспечивает надежный запуск одновибратора по входу *B* при пологих фронтах импульсов запуска.

Внешние резистор *R* и конденсатор *C* определяют длительность выходного импульса $t_{имп} \approx 0{,}7RC$, они могут быть в пределах $R = 1{,}5\ldots50$ кОм; $C = 10$ пф $\ldots1000$ мкФ.

В микросхеме 155АГ1 (рис. 6.18) имеется внутренний резистор *RI* (INTERNAL – внутренний), включенный между выводами *RI* и *RC*, сопротивлением примерно 2 кОм. Во многих случаях можно обойтись этим внутренним резистором, без внешнего резистора *R*, для этого надо к $U_{и.п}$ подключать вывод *RI* внутреннего резистора (вместо внешнего резистора *R*).

Минимальная длительность выходного импульса микросхемы 155АГ1 составляет 35 нс.

Когда одновибратор 155АГ1 запущен, то во время формирования выходного импульса он не реагирует на новые входные импульсы по любому из запускающих входов *A*1, *A*2 или *B*, т. е. он – одновибратор без перезапуска, в отличие от одновибраторов 155АГ3 (рис. 6.18) и 564АГ1.



R	A	B	Q
0	X	X	0
X	1	X	0
X	X	0	0
1			
1		1	
	0	1	

а) *б*)

Рис. 6.18. Одновибратор 155АГ3: *а* – условное графическое обозначение; *б* – таблица переключений

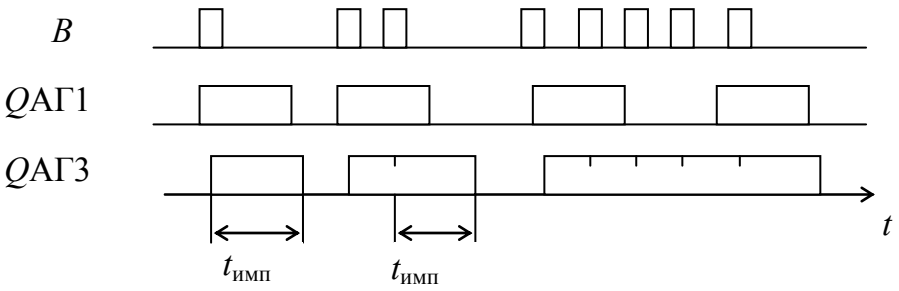


Рис. 6.19. Временные диаграммы перезапуска одновибратора 155АГ3

Различия выходных импульсов одновибраторов 155АГ1 и 155АГ3 при поступлении на их входы *B* одинаковых импульсных последовательностей показывает разницу между работой одновибраторов без перезапуска (АГ1) и с перезапуском (АГ3).

В одном корпусе 155АГ3 два одновибратора, которые отличаются от 155АГ1 отсутствием входного триггера Шмитта и внутреннего резистора. Одновибратор 155АГ3 допускает повторный запуск во время, пока не закончилось формирование выходного импульса; при этом длительность выходного импульса увеличивается на интервал времени между первым и последующими запусками. Этот процесс называется *перезапуском*.

Если импульсы на вход одновибратора с перезапуском поступают быстрее, чем закончится формирование выходного импульса одновибратора от предыдущего входного импульса, то на выходе его будет один «нескончаемый импульс». Таким образом, можно выделять интервалы между импульсами больше, чем заданный интервал.

Мультивибратор с перезапуском называют иногда *детектором подавления импульсов*.

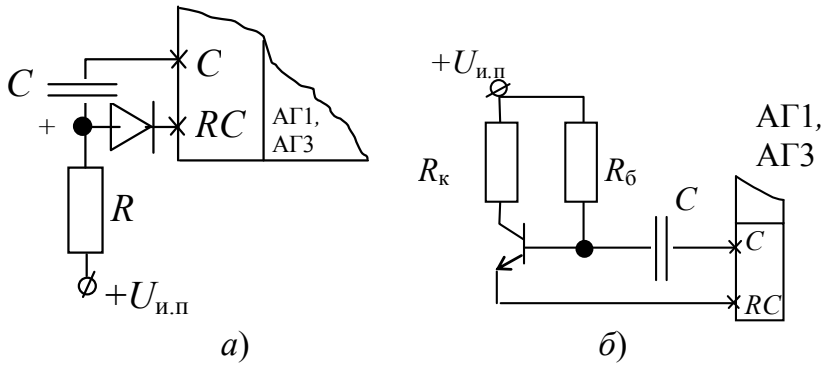


Рис. 6.20. Примеры подключения входов одновибраторов:
а – использование электролитического конденсатора;
б – с использованием транзисторного повторителя

При использовании электролитических конденсаторов в качестве времязадающих, для защиты от обратного напряжения, на конденсаторе рекомендуют включать диод перед *RC*-входом микросхем (рис. 6.20, *а*).

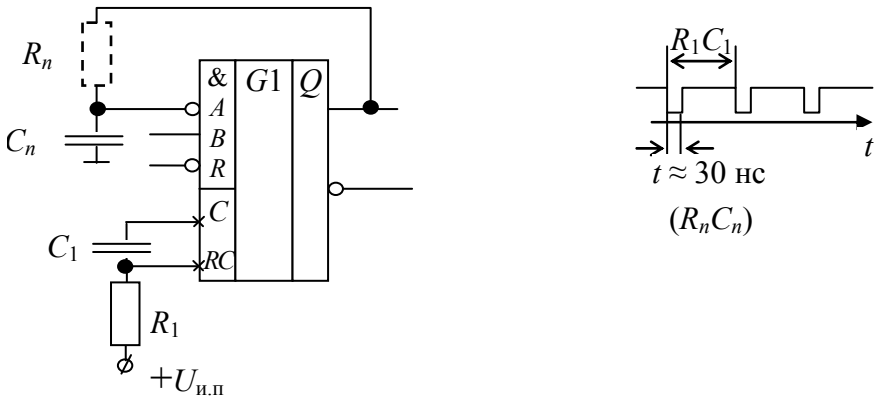


Рис. 6.21. Автогенератор коротких инверсных импульсов

Транзисторный повторитель, подключенный к *RC*-входу вместо времязадающего резистора *R* (рис. 6.20, *б*), позволяет на два порядка увеличить постоянную времени, а следовательно, и длительность выходного импульса – без увеличения емкости *C*. Коллекторный резистор $R_к \approx$ 1кОм, резистор в цепи базы повторителя $R_б \leq 5$ МОм (считая это усиление транзистора $h_{21Э} \geq 100$, а обратный ток его коллекторного перехода $I_{к0} < 1$ МкА).

Автогенератор коротких инверсных импульсов (рис. 6.21) длительностью $t^0_{имп} \approx 30$ нс с периодом повторения их $T \approx R_1C_1$. Чтобы

увеличить длительность $t^0_{\text{имп}}$ в цепи обратной связи можно включить интегрирующую R_nC_n-цепочку, но погрешность длительности этих импульсов будет такой же, как и при использовании логических элементов, т. е. приблизительно 10…20 %, в то время как интервал между импульсами $T = R_1C_1$ будет выдерживаться с погрешностью ~ 1%.

Генератор на двух одновибраторах 155АГ3 (рис. 6.22) работает в трех режимах:

1. $G = \overline{R} = 1$ автоколебательный.

2. $\overline{R} = 1$ стартстопный – генератор пачки импульсов на интервале $G = 1$.

3. $\overline{R} = 0$ одновибратор, формирующий выходной импульс из перепада 01 сигнала G.

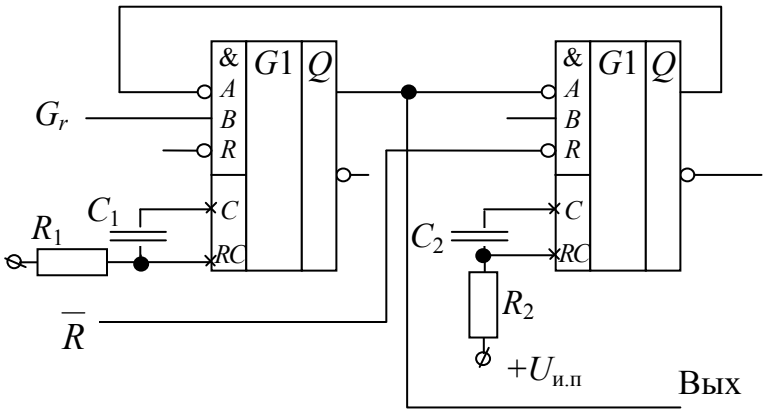


Рис. 6.22. Генератор на двух одновибраторах 155АГ3

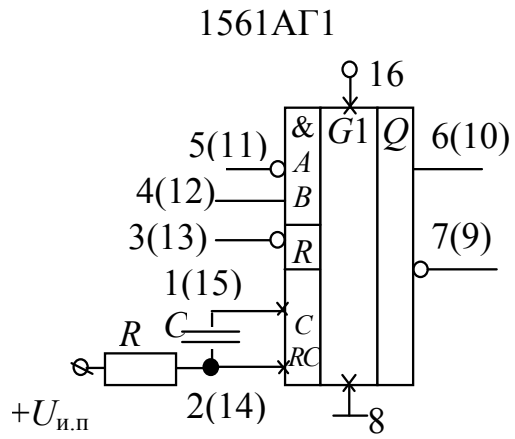
6.3.3. Микросхемы одновибраторов КМОП

В корпусе 1561АГ1 (рис. 6.23) два одновибратора с возможностью работы с перезапуском и без перезапуска. Каждый одновибратор имеет два взаимоинверсных входа запуска A для ⎺_ и B для _/⎺ фронтов запуска и вход R для сброса уже начавшегося импульса или для запрета для последующих входных импульсов.

Режим без перезапуска обеспечивается за счет обратной связи с выходов одновибратора на один из его входов в соответствии с таблицей – строки 5 и 7 – это режим без перезапуска, все остальные строки таблицы соответствуют работе с перезапуском.

Сопротивление R рекомендуют брать 10 кОм $\le R \le$ 10 МОм, емкость 15 пФ $\le C \le$ 100 пФ.

Длительность выходного импульса $t \approx 0{,}5\ RC$ при $C > 0{,}01$ мкФ и $t_2 \approx RC$ при $C \approx 100$ пФ.



N	*R*	*A*	*B*	*Q*
1	0	*X*	*X*	0
2	1	1		
3	1		0	
4	1	1		
5	1	$\overline{Q}$		
6	1		0	
7	1		$\overline{Q}$	

а) *б*)

Рис. 6.23. Одновибратор 1561АГ1: *а* – условное графическое обозначение; *б* – таблица переключений

Схема формирования импульса длительностью $T_2 \approx R_2C_2$, задержанного от начала входного импульса на время задержки $T_1 \approx R_1C_1$, выполнена на двух половинках 1561АГ1 (рис. 6.24).

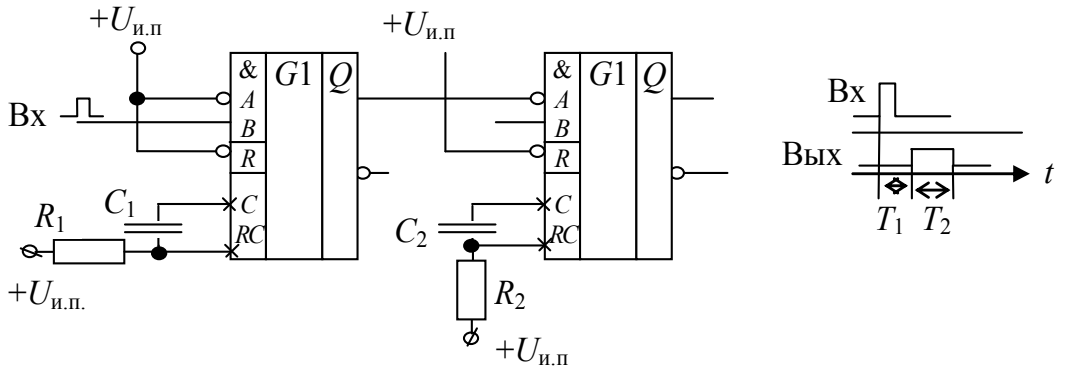


Рис. 6.24. Схема формирования импульса на двух половинах 1561АГ1 и временная диаграмма ее работы

Автогенератор импульсов, длительность которых определяется времязадающей цепью R_1C_1 на одновибраторе 1, а продолжительность пауз T_2 между импульсами задается постоянной времени R_2C_2 на одновибраторе 2, как показано на рис. 6.25.

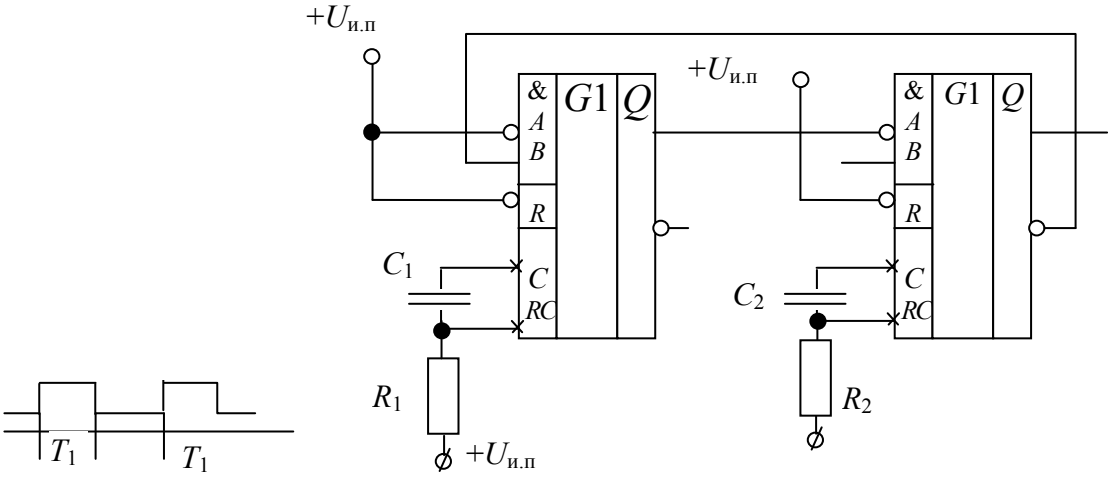
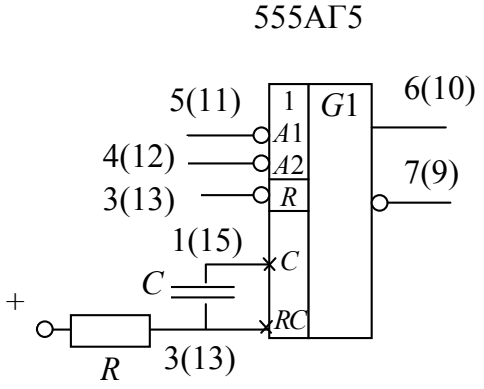


Рис. 6.25. Схема формирования импульса

Микросхема 555АГ5 представляет собой два одновибратора в одном корпусе с перезапуском. Его условное графическое обозначение и таблица переключений представлены на рис. 6.26.



R	A1	A2	Q
0	X	X	0
1	⎍	1	⎍
1	0	⎎	⎍
1	⎍	1	⎍
1	0	⎍	⎍

$1k \le R24$ кОм
$0 \le C \le \infty$
При R = 10 кОМ,
C = 1 нф,
τ = 4,1…4,5 мкс

а) *б*)

Рис. 6.26. Одновибратор 555АГ5: *а* – условное графическое обозначение; *б* – таблица переключений

Прецизионный таймер 1006ВИ1 (рис. 6.28) (очень часто в литературе – зарубежной и отечественной – называют *NE*555 по аналогу выпускаемому фирмой «Signetics») выполнен по биполярной технологии (на *n-p-n* и *p-n-p* транзисторах). Он состоит из *RS*-триггера, мощного двухтактного выходного каскада, выходного каскада с открытым коллектором и двух аналоговых компараторов *C*1 и *C*2 (рис. 6.27).

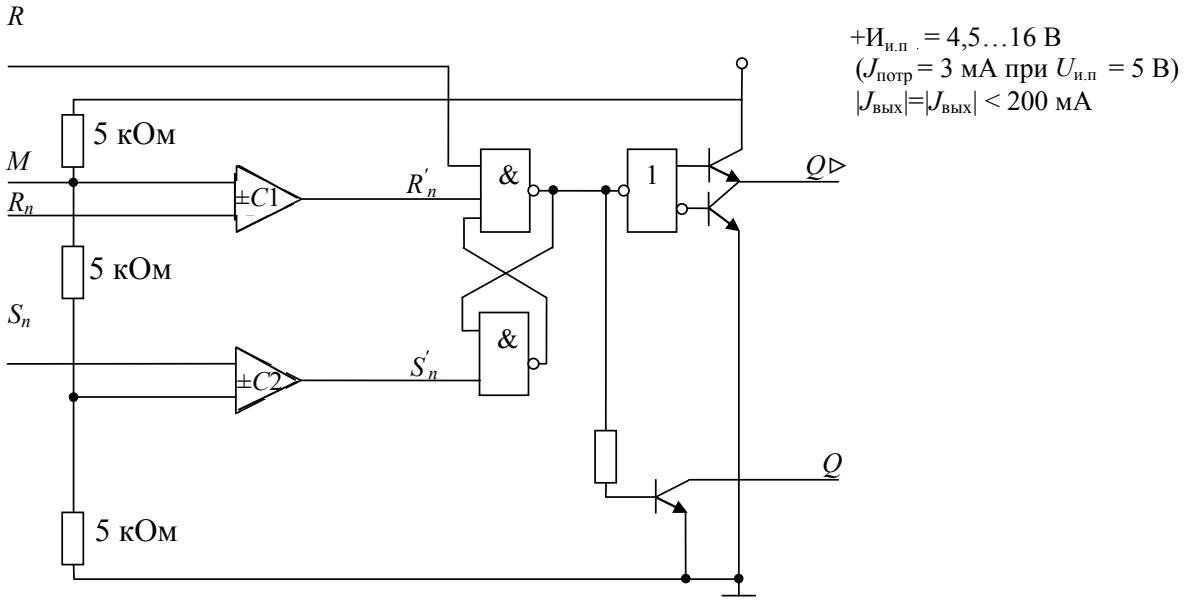


Рис. 6.27. Функциональная схема прецизионного таймера 1006ВИ1

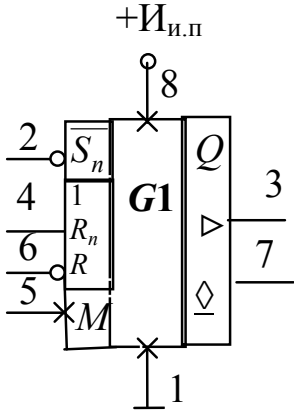


Рис. 6.28. Условное графическое обозначение прецизионного таймера 1006ВИ1

Всю схему таймера 1006ВИ1 можно рассматривать как асинхронный потенциальный *RS*-триггер с двумя аналоговыми входами S_n R_n и одним цифровым входом $\overline{R}$ сброса. Компаратор *C*2, подключаемый к входу S_n, имеет низкое быстродействие, поэтому длительность входного сигнала $\overline{S_n}$ = 0 должна быть не менее 10 мкс, чтобы компаратор сработал, поэтому над обозначением этого входа $\overline{S_n}$ рисуют символ задержки.

Модулирующий вход *M* используется для одновременного изменения порогов срабатывания компараторов по входам R_n и $\overline{S_n}$. Пока *M* вход никуда не подключен, пороги компараторов определяются

напряжением питания микросхемы. Напряжение порога компаратора $C1$ $U_{n1} = \frac{2}{3} U_{\text{и.п}}$, компаратора $C2$ $U_{n2} = \frac{1}{3} U_{\text{и.п}}$.

Если на вход M подать напряжение U_m от источника сигнала с низким внутренним выходным сопротивлением $R_i << 5$ кОм, то порог компаратора $C1$ станет равным этому напряжению $U_{n1} = U_M$, а порог нижнего компаратора $C2$ будет $U_{n2} = \frac{1}{2} U_M$.

Цифровой вход $\overline{R}$ позволяет сбрасывать выходные сигналы в ноль независимо от всех других входов, приоритет входа $\overline{R}$ цифрового самый высокий.

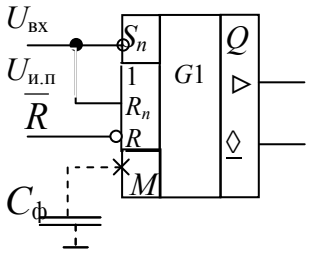


Рис. 6.29. Включение 1006ВИ1 в качестве прецизионного триггера Шмитта

В исходном состоянии на выходах на рис. 6.29 показано включение 1006ВИ1 в качестве прецизионного триггера Шмитта с порогами $U_{n1} = \frac{2}{3} U_{\text{и.п}}$, $U_{n1} = \frac{1}{3} U_{\text{и.п}}$. Конденсатор $C_{\text{ф}}$ подключен к входу M, дополнительно фильтрует от помех напряжение порогов срабатывания.

Если на вход M подать напряжение, то пороги переключения триггера Шмитта будут меняться в соответствии с этим напряжением.

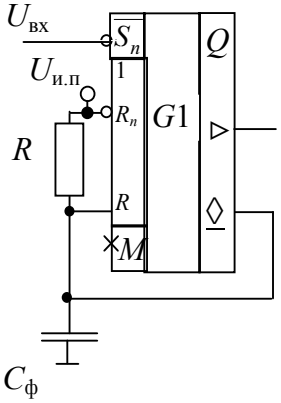


Рис. 6.30. Включение таймера 1006ВИ1 в качестве одновибратора

На рис. 6.30 показано включение таймера 1006ВИ1 в качестве одновибратора.

В исходном состоянии на выходах лог.0, и поэтому открытый транзистор выходного каскада с открытым коллектором удерживает конденсатор C в разряженном состоянии. При подаче на вход сигнала $\overline{S_n} = 0$ RS-триггер устанавливает выходы в единичное состояние и конденсатор C начинает заряжаться через резистор R по закону: $U_c = U_{\text{и.п}}(1 - e^{-\frac{t}{RC}})$.

На рис. 6.31 изображены временные диаграммы работы таймера 1006ВИ1 при включении его в качестве одновибратора.

Как только напряжение U_c, подаваемое через R_n вход на компаратор $C1$, достигнет порога срабатывания этого компаратора $U_{n1} = \frac{2}{3} U_{\text{и.п}}$, срабатывает компаратор $C1$ и сбрасывает RS-триггер в ноль, что вызывает быстрый разряд конденсатора C через открытый транзистор выходного каскада с открытым коллектором.

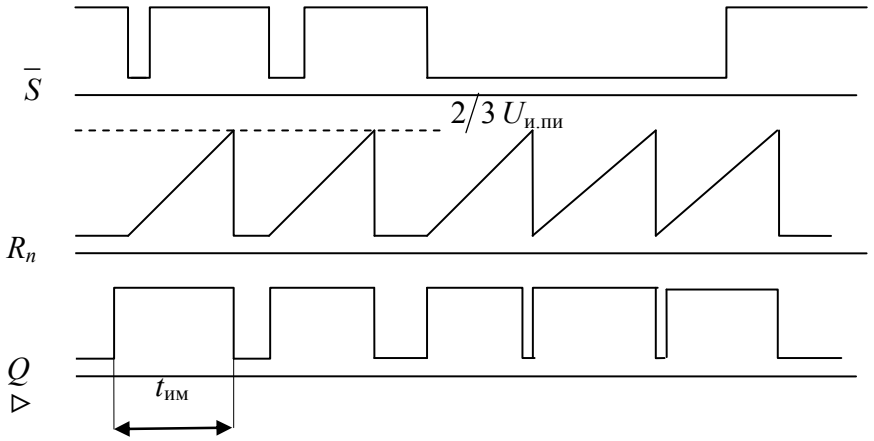


Рис. 6.31. Временные диаграммы работы таймера 1006ВИ1 при включении его в качестве одновибратора

Длительность $t_{\text{имп}}$ импульса $Q = 1$ зависит от соотношения

$$\frac{2}{3} U_{\text{и.п}} = U_{\text{и.п}}(1 - e^{-\frac{t}{RC}})$$

и независимо от напряжения питания определяется как

$$t_{\text{имп}} = RC \ln 3.$$

Если на вход M подать постоянное или медленно меняющееся по сравнению с частотой входных импульсов напряжение U_M, то длительность $t_{имп}$ станет определяться из выражения

$$U_M = U_{и.п}(1 - e^{-\frac{t_{имп}}{RC}}),$$

откуда

$$t_{имп} = -RC\ln(1 - \frac{U_M}{U_{и.п}}).$$

Если длительность сигнала запуска $\overline{S_n} = 0$ больше $t_{имп}$, то в течение этого сигнала одновибратор срабатывает несколько раз. Чтобы избежать этого, можно цифровой вход R сброса соединить с аналоговым входом $\overline{S_n}$.

Схема автогенератора прямоугольных импульсов со скважностью, равной двум, представлена на рис. 6.32.

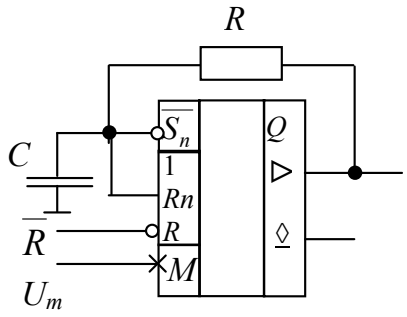


Рис. 6.32. Схема автогенератора прямоугольных импульсов со скважностью, равной двум

Скважность, равная двум, означает, что период повторения импульсов в два раза больше длительности импульса (меандр). $t_{имп} = t_{нагр} = 0{,}7RC = RC\ln 2$. В таком включении при подаче внешнего напряжение U_m на вход M длительность импульса $t_{имп}$ при $Q = 1$ уменьшается или растет вместе с уменьшением или ростом напряжения U_m по закону

$$t_{имп} = RC\ln\frac{U_{и.п} - U_M}{U_{и.п} - \frac{U_M}{2}}.$$

На рис. 6.33 представлены временные диаграммы работы таймера 1006ВИ1 при включении его в качестве автогенератора прямоугольных импульсов со скважностью, равной двум.

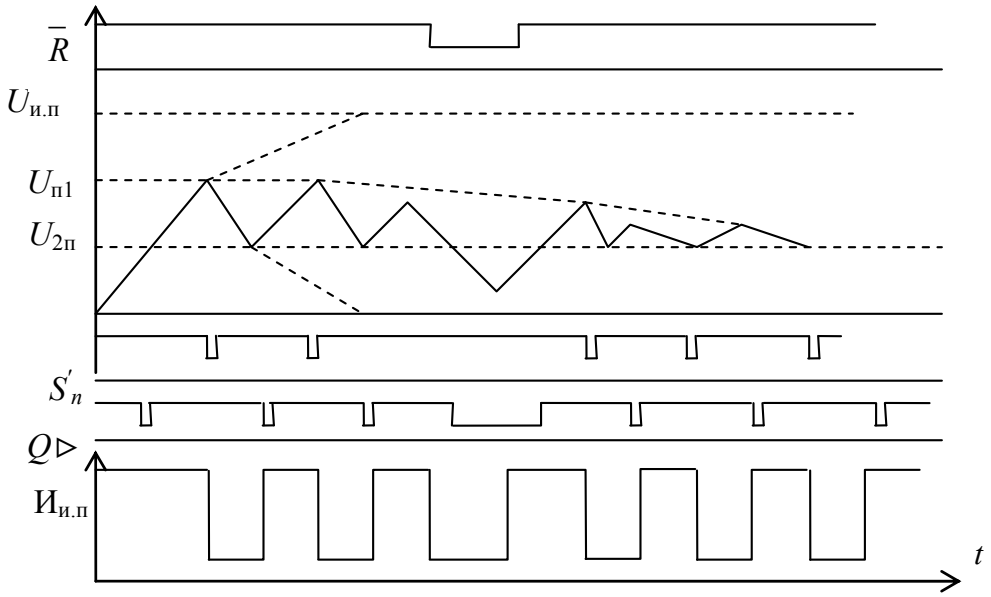


Рис. 6.33. Временные диаграммы работы таймера 1006ВИ1 при включении его в качестве автогенератора прямоугольных импульсов со скважностью, равной двум

Длительность паузы $t_{пауз}$ при изменении напряжения U_M на входе M не изменится и равно $t_{пауз} = RC\text{Ln}2$.

Неизменность $t_{пауз}$ или изменение напряжения U_M на входе M объясняется тем, что хотя напряжение второго порога $U_{п2}$ меняется при изменении U_M, но соотношение между ними не меняется: $\dfrac{U_{п1}}{U_M} = \dfrac{1}{2}$.

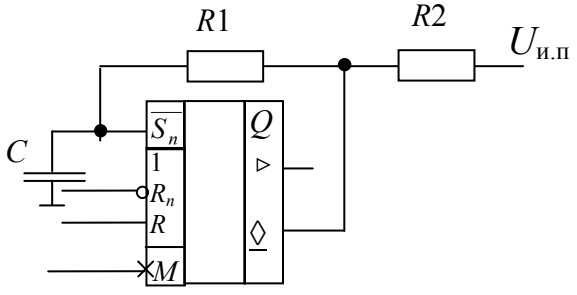


Рис. 6.34. Схема с автогенератора с обратной связью через выход с открытым коллектором

Для обратной связи в автогенераторах часто используют выход с открытым коллектором. В такой схеме (рис. 6.34) при $Q = 1$ длительность импульса определяется выражением: $t_{имп} = (R1+R2)C\mathrm{Ln}2$, а длительность пауз – при $Q = 0$: $T_{пауз} = R1C\mathrm{Ln}2$.

Схема одновибратора с перезапуском с ЛЭ с открытым коллектором на входе 1006ВН1 представлена на рис. 6.35.

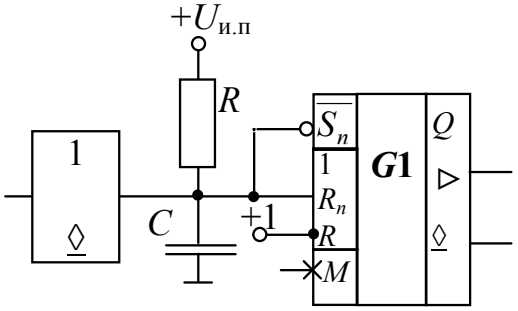


Рис. 6.35. Схема одновибратора с перезапуском с ЛЭ с открытым коллектором на входе 1006ВН1

6.4. Генераторы псевдослучайных последовательностей

6.4.1. Псевдослучайные последовательности

Генераторы псевдослучайных последовательностей (или линейные автоматы на основе сдвигающих регистров) используются в устройствах тестового диагностирования цифровых устройств, для решения математических задач методом Монте-Карло, при моделировании систем с учетом случайного разброса их параметров и в ряде других случаев.

Ряд названий, относящихся к генераторам псевдослучайных последовательностей, связан с понятием линейных комбинационных функций, клеточных автоматов и т. п. По определению, к линейным переключательным функциям относятся те, для которых полином Жегалкина имеет степень не выше первой. Такие функции содержат конъюнкции единичной длины и могут быть представлены в виде

$$F(x_1,...,x_n) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus ... \oplus a_nx_n,$$

где a принимают значения 0 или 1 ($i = 0...n$).

Автомат *линеен*, если схемы выработки функций выходов и функций возбуждения D-триггеров, образующих память автомата, линейны (эти схемы составлены только из сумматоров по модулю 2).

Возможная реализация автономного линейного автомата – сдвигающий регистр с обратными связями через сумматоры по модулю 2. Такие автоматы применяются в генераторах псевдослучайных последовательностей и устройствах тестового контроля ЦУ, где они обеспечивают получение циклических кодов.

Случайные сигналы могут быть аналоговыми или цифровыми. Цифровой сигнал при этом представляется случайной последователь-ностью, элементами которой могут быть символы 0 и 1 или многораз-рядные числа. Первому варианту обычно присваивается наименование *случайной последовательности*, второму – *случайных чисел.*

Случайные сигналы характеризуются законами распределения, среди которых важное место занимает равномерный закон, так как сигналы с таким законом распределения имеют не только непосредственное применение, но и служат для получения сигналов с другими законами распределения путем определенной математической обработки.

Истинно случайные последовательности и числа генерируются на основе физических явлений (флуктуационных шумов в электронных приборах, радиоактивного излучения и др.), что сложно реализуется и не обеспечивает стабильности статистических характеристик.

При генерации *псевдослучайных последовательностей и чисел* сигнал детерминирован, но его характеристики близки к характеристикам истинно случайного сигнала. Генерация псевдослучайных сигналов проще и надежнее. Структура сдвигающего регистра с линейной обратной связью показана на рис. 6.36.

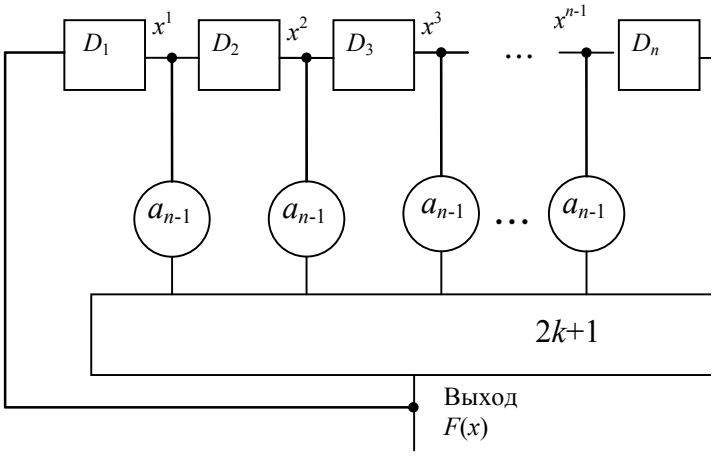


Рис. 6.36. Структура сдвигающего регистра с линейной обратной связью

Выходная последовательность снимается с входа триггера D_1, она же повторяется со сдвигами в других точках тракта, образованного *D*-триггерами. Аргументами линейной функции являются величины,

различающиеся только сдвигами на то или иное число тактов, что отмечается показателем степени при аргументе *x*. Структуре схемы ставится в соответствие полином

$$F(x) = x^n + a_1 x^{n-1} + \ldots + a_{n-1} x + 1 .$$

Начав с любого исходного состояния, можно вычислить последующие. На входе левого триггера окажется значение функции $F(x)$, соответствующее исходному состоянию, в других – результат сдвига на один разряд. Как только опять возникает состояние, идентичное исходному, все начнет повторяться, т. е. устройство работает периодично. Период последовательности зависит от коэффициентов a_i. Обычно используют максимальный период. Автомат с *n*-триггерами может иметь 2^n–1 состояний. В данном автомате состояние всех нулей должно быть исключено, так как из него схема никогда не сможет выйти. Поэтому для данного автомата максимальный период составит 2^{n-1}, а соответствующая ему последовательность называется *последовательностью максимальной длины*, или *М-последовательностью.*

К *М-последовательности* приводят многие варианты схемы. Их поиск основан на изучении полинома, соответствующего схеме. Чтобы генерировалась *М*-последовательность, полином должен быть неприводимым и примитивным. Таких полиномов много: при $n = 8$ их 16, при $n = 16$ уже 2048 и т. д. Среди множества полиномов целесообразно отыскать такие, у которых минимальное число ненулевых коэффициентов a_i, поскольку это упрощает схему.

Для генерации *М*-последовательностей схемой с одним элементом сложения по модулю 2 рассчитаны таблицы. Элемент имеет два входа, один из которых подключен к выходу последнего триггера регистра (иначе его наличие в схеме теряет смысл), а второй подключен к разряду с номером *i*. Если перевести вход элемента с выхода разряда номер *i* на выход разряда номер *n–i*, то будет генерироваться последовательность с обратным порядком следования двоичных символов, поэтому в приводимой таблице (табл. 6.2) указаны номера разрядов *i* или *n–i.*

Таблица 6.2

Таблица для генерации *М*-последовательностей схемой с одним элементом сложения по модулю 2

n	4	5	6	7	9	10	11	15	17	18	20
i* или *n–i	1	2	1	1, 3	4	3	2	1, 4, 7	3	7	3

6.4.2. Схемы генераторов псевдослучайной последовательности (ГПСП)

Схема генератора псевдослучайной последовательности, соответствующего первому столбцу таблицы (рис. 6.37, *а*), останавливается сбросом всех триггеров и запускается импульсом старта, записывающим единицу через элемент сложения по модулю 2 в левый триггер. На рис. 6.37, *б* показана схема такого же ГПСП, но обладающего свойством самозапуска. С выхода любого триггера ГПСП можно снять последовательность 111101011001000, соответствующую $M = 2^4 - 1 = 15$. Для схемы ГПСП с 20 разрядами $M = 1048575$. Если длина последовательности превышает емкость памяти системы, то псевдослучайную последовательность не отличить от случайной.

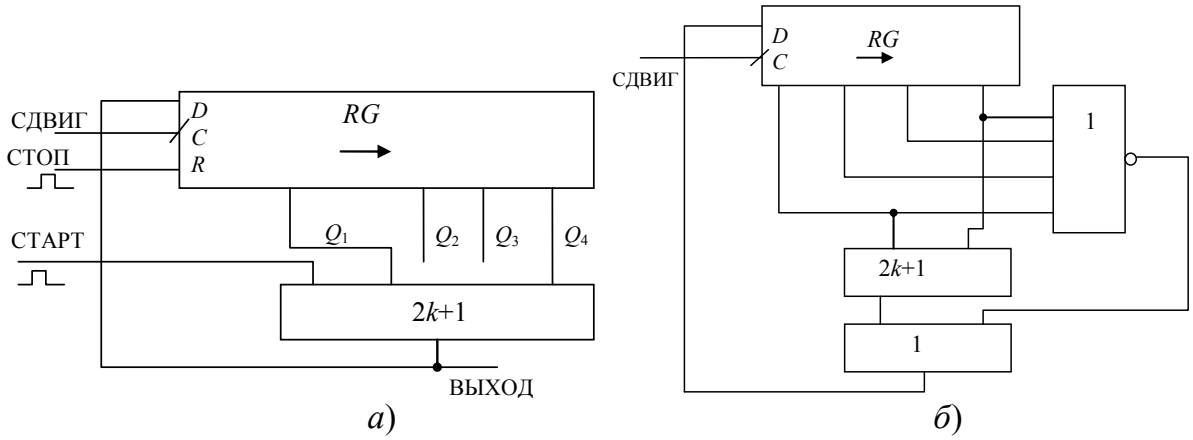


Рис. 6.37. Схемы четырехразрядных генераторов *M*-последовательностей: *а* – запускающегося стартовым импульсом; *б* – самозапускающегося

Генерируемые последовательности имеют число единиц, на единицу превышающее число нулей (следствие исключения состояния «все нули»); группы одинаковых символов появляются в них с той же частотой, что и в случайной последовательности равновероятных двоичных символов; нормированная автокорреляционная функция качественно подобна этой функции белого шума $R(\tau) = 0$ при больших M и τ, не кратных M.

6.4.3. Генераторы псевдослучайных чисел (ГПСЧ)

Генераторы псевдослучайных чисел строят по последовательному, параллельному и смешанному способам. В первом случае число (слово) образуется за несколько тактов. Из образованной в регистре последовательности для получения m-разрядного слова получают результат путем S сдвигов, где $S > m$, что дает отсутствие корреляции

между соседними словами. Период последовательности слов равен наименьшему общему кратному чисел S и M. Для получения максимального периода число S выбирается взаимнопростым к M.

В генераторах параллельного типа псевдослучайные числа генерируются в каждом такте. Очевидным решением было бы использование m генераторов псевдослучайных двоичных последовательностей для образования отдельных разрядов случайных чисел, однако существуют более простые решения, например, клеточные автоматы.

Линейные автоматы на основе сдвигающих регистров используются также в *сигнатурных анализаторах*, являющихся средствами тестового диагностирования цифровых устройств, требующих подачи на них специальных воздействий, с помощью которых проверяется правильность работы устройства. С ростом сложности устройств длина тестовых последовательностей и объем оборудования, обеспечивающего генерацию тестов и анализ результатов, увеличиваются и возникает задача сжатия информации при тестировании. Эта задача решается, в частности, с использованием линейных автоматов на основе регистров сдвига в сигнатурных анализаторах.

Если в схемах ГПСП ввести дополнительно внешний вход на элементы $2k+1$, то получится устройство для аппаратного выполнения операции деления полиномов по правилам арифметики по модулю 2. При этом входная последовательность, состоящая из нулей и единиц, трактуется как полином, содержащий те степени переменной x, которым соответствуют единицы. Например, последовательность 10010010 соответствует полиному $x^7 \oplus x^4 \oplus x$. Этот полином делится на так называемый порождающий полином, определяемый структурой схемы, как показано ранее. В результате деления в регистре записывается остаток $R(x)$.

Проверка логической схемы производится следующим образом. На ее входы от генератора псевдослучайных чисел подается известная последовательность. Выход схемы подключается к узлу деления полиномов. После деления в регистре остается остаток (сигнатура). Сигнатура для исправной схемы известна. Сравнение полученного остатка с этой сигнатурой (эталонным остатком) позволяет сделать заключение о правильности работы схемы. С помощью специальных процедур наряду с обнаружением ошибок можно производить и их поиск.

Сдвигающие регистры с обратными связями, выполняющие операции над полиномами, применяются также для построения и анализа циклических кодов, применяемых для обнаружения и коррекции ошибок в цифровых устройствах.

ГЛАВА 7. ЦИФРОВЫЕ УСТРОЙСТВА ПРОМЫШЛЕННОЙ ЭЛЕКТРОНИКИ

7.1. Синтезаторы частот

7.1.1. Цифровой синтезатор частот на основе суммирования импульсных последовательностей

Синтезом частот называется формирование дискретного множества частот из одной или нескольких опорных частот $f_{оп}$ (рис. 7.1). *Опорной* называется высокостабильная частота автогенератора, обычно кварцевого.

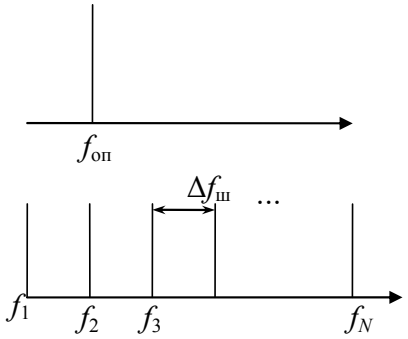


Рис. 7.1. Пример синтеза частот

Синтезатором частот называется устройство, реализующее процесс синтеза. Синтезатор является непременной частью современных радиопередатчиков и радиоприемников систем радиосвязи, радионавигации, радиолокации и другого назначения.

Основными параметрами синтезатора являются:

– *диапазон частот выходного сигнала*;

– *количество N* и *шаг сетки частот* $\Delta f_ш$;

– долговременная и кратковременная *нестабильность частоты*;

– *уровень побочных составляющих* в выходном сигнале;

– *время перехода* с одной частоты на другую.

В современных синтезаторах количество формируемых дискретных частот может достигать десятков тысяч, а шаг сетки изменяться от десятков герц до десятков и сотен килогерц. Долговременная нестабильность частоты, определяемая кварцевым автогенератором, составляет обычно 10^{-6}, а в специальных случаях $10^{-8}...10^{-9}$. В больших пределах меняется и диапазон частот в зависимости от назначения радиотехнической аппаратуры, в которой используется синтезатор.

В последнее время при создании синтезаторов частот, выполненных по *принципу прямого синтеза*, стали широко использовать цифровые методы. Примером может служить синтезатор частот, построенный на основе суммирования импульсных последовательностей. Структурная схема такого синтезатора, выполненного полностью на цифровых интегральных микросхемах, приведена на рис. 7.2. Эпюры соответствующих импульсных последовательностей изображены на рис. 7.3.

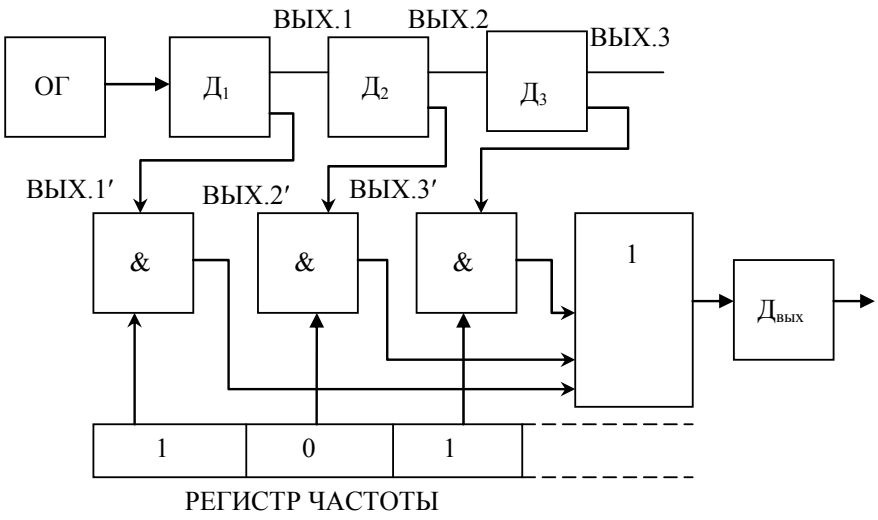


Рис. 7.2. Цифровой синтезатор частот на основе суммирования импульсных последовательностей

Сигнал высокостабильного опорного генератора (ОГ) (рис. 7.2) поступает на триггерный счетчик-делитель (Д), состоящий из *n* двоичных разрядов (на рис. 7.2 показано три разряда). На выходе делителя каждого разряда ($Д_1$, $Д_2$, $Д_3$) получаются две последовательности импульсов (например, на выходе 1 и 1′ (рис. 7.3), сдвинутые на $T/2$ (где T – период соответствующей импульсной последовательности). Частота импульсной последовательности на выходе каждого делителя в 2 раза меньше частоты входной импульсной последовательности.

С выходов делителей 1', 2', 3' и т. д. импульсные последовательности поступают на один вход логического элемента И. На другой вход этой схемы поступает лог.0 или лог.1 с регистра частоты. Если в регистре частоты записана лог.1, то соответствующая импульсная последовательность (на рис. 7.2 импульсные выходные последовательности с делителей $Д_1$ и $Д_3$) проходит на логический элемент ИЛИ. Если же записан лог.0, то ЛЭ И закрыт и импульсная последовательность на него не проходит (на рис. 7.2 выходную импульсную после-

довательность с делителя $Д_2$). Следовательно, на выходе ЛЭ ИЛИ происходит суммирование соответствующих последовательностей в соответствии с заданным кодом частоты.

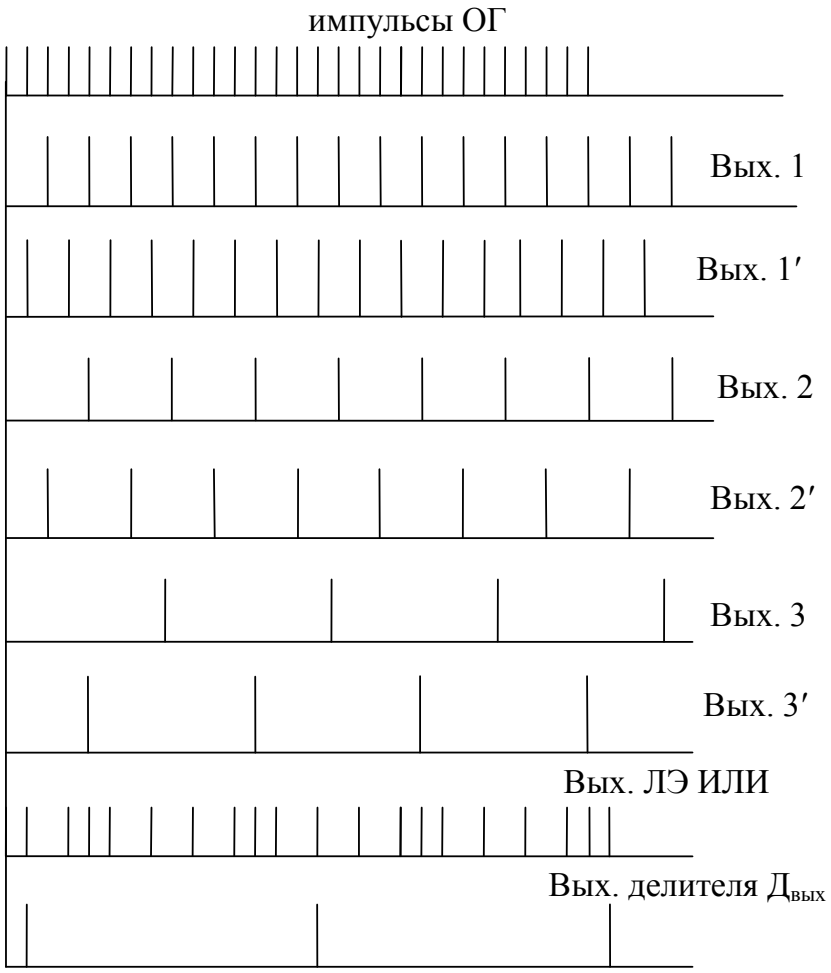


Рис. 7.3. Эпюры импульсных последовательностей в цифровом синтезаторе частот

В результате получается импульсная последовательность с неравномерной расстановкой импульсов (на рис. 7.3 выход ЛЭ ИЛИ), средняя частота импульсов которых определяется управляющим кодом, записанным в регистре частоты. Для уменьшения неравномерности импульсов на выходе ЛЭ ИЛИ включают делитель частоты ($Д_{вых}$ на рис. 7.2 с коэффициентом деления N). На выходе такого делителя импульсная последовательность более равномерная (на рис. 7.3 выход делителя $Д_{вых}$). Чем выше коэффициент деления, тем больше равномерная выходная импульсная последовательность и тем меньше уровень побочных частот в выходном спектре синтезатора. Но при этом частота синтезатора понижается.

7.1.2. Синтезатор частот с цифровым формированием отсчетов синтезируемого колебания

Другой разновидностью синтезатора частот, в котором использован цифровой принцип формирования частот, является синтезатор с цифровым формированием отсчетов синтезируемого колебания. Структурная схема такого синтезатора приведена на рис. 7.4.

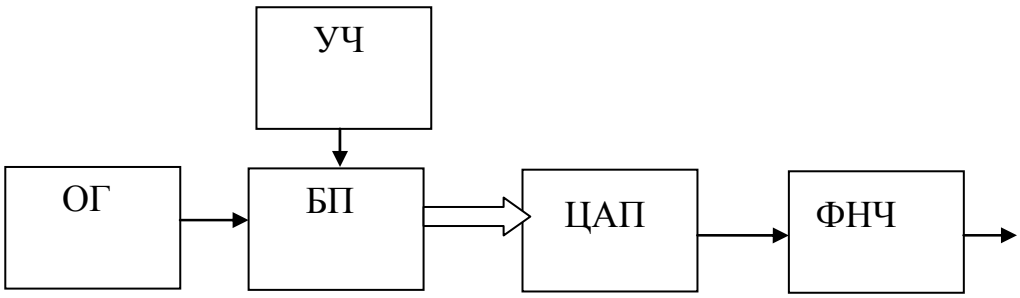


Рис. 7.4. Синтезатор частот с цифровым формированием отсчетов

В блоке памяти (БП) хранятся отсчеты синусоиды (данные о значении синусоиды при различных фазах). По определенной программе в соответствии с кодом частоты, записанным в блоке установки частоты (УЧ), вычисляются текущие значения синусоиды. Обычно БП выполняется в виде микропроцессорного устройства, которое используется как счетчик времени (накопитель фазы). Частота f в импульсной последовательности на выходе цифро-аналогового преобразователя (ЦАП) кратна шагу сетки частот: $f = k\Delta f$, где $k = 1, ..., N$.

Поскольку на выходе синтезатора необходимо формирование синусоидального колебания, то после БП включается ЦАП (рис. 7.4). Для устранения побочных частот после блока ЦАП включен фильтр нижних частот (ФНЧ), который фильтрует тактовую частоту, ее гармоники и комбинационные частоты. Число отсчетов синусоиды 2^N определяется объемом памяти блока вычисления отсчетов (БП). Если все отсчеты синусоиды считываются с частотой $f_{\text{ОГ}} = \dfrac{1}{T_{\text{ОГ}}}$, то период импульсной последовательности на выходе блока ЦАП $T = T_{\text{ОГ}} 2^N$, где $T_{\text{ОГ}}$ – период сигнала ОГ. Следовательно, минимальная частота импульсной последовательности

$$f_{\min} = \Delta f = \frac{1}{T_{\text{ОГ}} 2^N}.$$

Изменяя число импульсов ОГ, считываемых за период $T_{ОГ}$ (т. е. – число N), можно изменить частоту импульсной последовательности на выходе ЦАП.

Минимальное число импульсов ОГ равно двум, следовательно:

$$f_{\max} = \frac{1}{2T_{ОГ}}.$$

Верхняя частота определяется граничной частотой цифровых микросхем и блока ЦАП. С повышением выходной частоты необходимо увеличивать быстродействие ЦАП. Поскольку на выходе синтезатора нет деления частоты, то его граничная частота с отсчетами синтезируемого колебания оказывается выше, чем в синтезаторах, построенных на основе суммирования импульсных последовательностей.

7.1.3. Цифровой синтезатор частоты, построенный на основе метода косвенного синтеза

На практике используются также синтезаторы частоты, построенные на основе метода косвенного синтеза (иногда называемого методом анализа). Такие синтезаторы содержат в своем составе подстраиваемый по частоте автогенератор, охваченный петлей фазовой автоподстройки частоты (ФАПЧ).

Основной способ построения синтезатора основывается на применении схемы импульсно-фазовой автоподстройки частоты и элементов вычислительной техники. Укрупненная структурная схема синтезатора такого типа с одним кольцом фазовой автоподстройки частоты приведена на рис. 7.5.

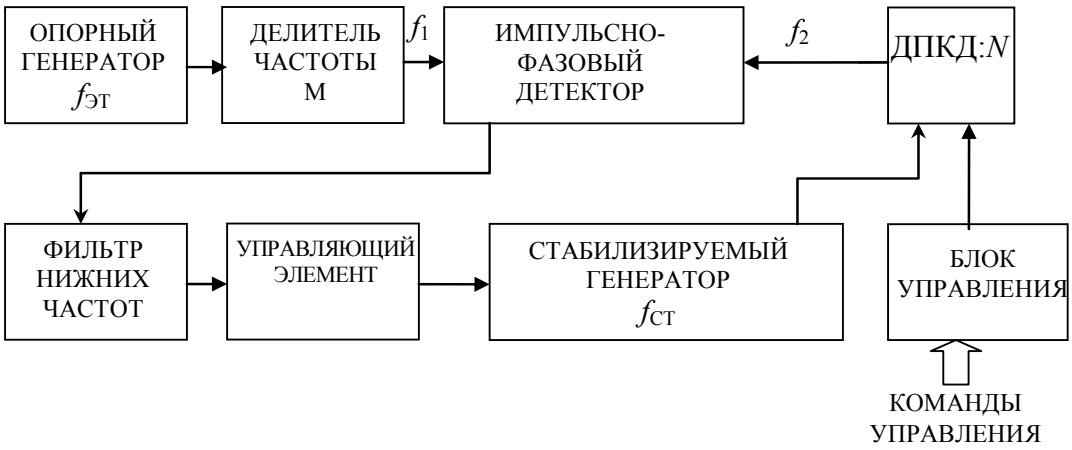


Рис. 7.5. Структурная схема синтезатора с одним кольцом фазовой автоподстройки частоты

В данной схеме ДПКД – это делители с переменным коэффициентом деления – *K*-разрядный программируемый цифровой счетчик. Назначение других звеньев схемы ясно из сделанных на них надписей. В блоке управления осуществляется прием и хранение данных программирования и формирование кодового сигнала, по которому устанавливается значение коэффициента деления *N* в зависимости от поступившей на синтезатор команды.

В результате действия фазовой автоподстройки частоты устанавливается равенство частот f_1 и f_2 сигналов, поступающих на вход импульсно-фазового детектора, что позволяет записать следующее соотношение для частот стабилизируемого и эталонного автогенераторов с учетом значений коэффициентов деления:

$$\frac{f_{\text{СТ}}}{N} = \frac{f_{\text{ЭТ}}}{M} \quad \text{или} \quad f_{\text{СТ}} = \frac{N}{M} f_{\text{ЭТ}}.$$

Согласно этому выражению шаг сетки частот $\Delta f = \frac{f_{\text{ЭТ}}}{M}$. Меняя управляемое значение *N*, устанавливают требуемое значение частоты стабилизируемого генератора, который с помощью управляющего элемента может перестраиваться в требуемом диапазоне частот.

7.2. Фазовые и частотные детекторы

7.2.1. Частотная и фазовая модуляция дискретных сообщений

При передаче дискретной, в том числе цифровой кодированной информации – комбинации двоичных сигналов, состоящей из лог.1 и лог.0, модуляцию также называют *манипуляцией сигнала*, а устройство, реализующее данный процесс, как модулятором, так и *манипулятором*. Кроме того, процесс манипуляции называют также телеграфным режимом работы, соответственно, заменяя название АМ на АТ, ЧМ на ЧТ, ФМ на ФТ. Три названных метода манипуляции высокочастотного сигнала имеют разный уровень помехоустойчивости, определяемую как вероятность ошибки принятого символа на выходе приемника от соотношения мощностей полезного сигнала и белого шума на входе демодулятора.

Поскольку метод амплитудной манипуляции (АМ) по помехоустойчивости существенно уступает двум другим, то в современных системах радиосвязи используют, в основном, только два метода манипуляции: частотный (ЧМ) и фазовый (ФМ). Причем в качестве ФМ

обычно используют ее разновидность – относительную фазовую модуляцию (ОФМ), называемую также фазоразностной. При ОФМ при передаче логической 1 фаза несущего колебания скачком изменяется на $\Delta\varphi$, например, на π, по отношению к фазе предыдущего бита, а при передаче логического 0 – фаза остается той же, что и у предыдущего бита.

Общим для обоих видов манипуляции (ЧМ и ФМ) является скорость передачи сообщения V, равная количеству передаваемых элементарных посылок (бит) в секунду (бит/с = бод), или длительность элементарной посылки $\tau = \frac{1}{V}$ (рис. 7.6, *а*). Кроме того, ЧМ характеризует дискрет частоты $\Delta F = F_1 - F_2$ (рис. 7.6, *б*), а ФМ – девиация или дискрет фазы $\Delta\varphi$ (рис. 7.6, *в*), позволяющие различать лог.1 и лог.0.

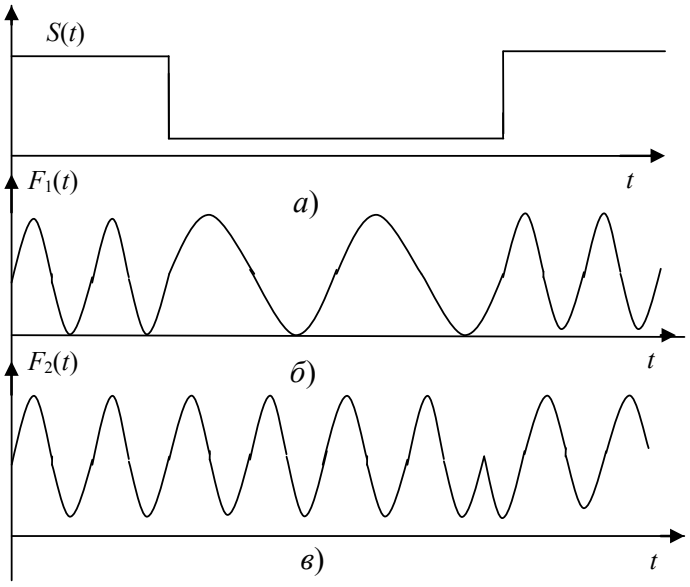


Рис. 7.6. Пример фазовой манипуляции: *а* – скорость передачи сообщения; *б* – дискрет частоты; *в* – дискрет фазы

Фазовая манипуляция (ФМ). В зависимости от дискрета фазы $\Delta\varphi$ наиболее часто используются следующие разновидности ФМ, приведенные в табл. 7.1.

Таблица 7.1

Разновидности фазовой манипуляции

Значение, $\Delta\varphi$	**Русское название**
π	Бинарная ФМ
$\pi/2$	Квадратурная ФМ
$\pi/2$	Квадратурная ФМ со смещением

При бинарной ФМ возможно два значения начальной фазы сигнала: 0 или π, что позволяет различить единичный бит информации: лог.1 или лог.0.

При квадратурной модуляции возможно четыре значения начальной фазы сигнала: 0, π/2, π, $3\pi/2$ или при смещении первого значения фазы на π/4 другая комбинация: π/4, $3\pi/4$, 5π/4, 7π/4. Поэтому здесь можно различить комбинацию из двух битов информации согласно табл. 7.2.

В результате при квадратурной ФМ, объединяя нечетный бит с четным или одновременно передавая битовые комбинации от двух источников, можно по сравнению с бинарной ФМ в два раза увеличить объем передаваемой информации за тот же по длительности сеанс связи.

Таблица 7.2

Таблица значений квадратурной модуляции

Кодовая комбинация	**ФМ без смещения**	**ФМ при смещении на π/4**	**ЧМ**
11	0	π/4	*F*1
01	π/2	3π/4	*F*2
10	π	5π/4	*F*3
00	3π/2	7π/4	*F*4

Так, 1-й символ, определяемый с помощью *N* бит (в частности, *N* = 8 или 16), передается без начального смещения фазы, 2-й символ – со смещением, 3-й символ – снова без смещения и т. д. (табл. 7.2). Формирование ФМ сигнала как бинарного, так и квадратурного вида возможно с помощью процессора по специальной программе.

ГЛАВА 8. ЗАПОМИНАЮЩИЕ УСТРОЙСТВА

8.1. Основные сведения. Система параметров запоминающих устройств

Запоминающие устройства (ЗУ) служат для хранения информации и обмена ею с другими цифровыми устройсвами (ЦУ). Микросхемы памяти в общем объеме выпуска ИС занимают около 40 % и играют важнейшую роль во многих системах различного назначения. Микросхемы и системы памяти постоянно совершенствуются как в области схемотехнологии, так и в области развития новых архитектур. В настоящее время созданы и используются десятки различных типов ЗУ. Важнейшие параметры ЗУ находятся в противоречии. Так, например, большая информационная емкость не сочетается с высоким быстродействием, а быстродействие в свою очередь не сочетается с низкой стоимостью. Потому системам памяти свойственна *многоступенчатая иерархическая структура*, и в зависимости от роли того или иного ЗУ его реализация может быть существенно различной.

В наиболее развитой иерархии памяти компьютера можно выделить следующие уровни:

– *регистровые* ЗУ, находящиеся в составе процессора или других устройств (т. е. внутренние для этих блоков), благодаря которым уменьшается число обращений к другим уровням памяти, реализованным вне процессора и требующим большего времени для операций обмена информацией;

– *кэш-память*, служащая для хранения копий информации, используемой в текущих операциях обмена. Высокое быстродействие кэш-памяти повышает производительность компьютера;

– *основная память* (оперативная, постоянная, полупостоянная), работающая в режиме непосредственного обмена с процессором и по возможности согласованная с ним по быстродействию. Исполняемый в текущий момент фрагмент программы обязательно находится в основной памяти;

– *специализированные* виды памяти, характерные для некоторых специфических архитектур (многопортовые, ассоциативные, видеопамять и др.);

– *внешняя память*, хранящая большие объемы информации. Эта память обычно реализуется на основе устройств с подвижным носителем информации (магнитные и оптические диски, флешки и др.).

Важнейшие параметры запоминающих устройств

Информационная емкость – максимально возможный объем хранимой информации. Выражается в битах или словах (в частности, в байтах). Бит хранится запоминающим элементом (ЗЭ), а слово – запоминающей ячейкой (ЗЯ), т. е. группой ЗЭ, с которым возможно лишь одновременное обращение. Добавление к единице измерения множителя «К» (кило) означает умножение на $2^{,0} = 1024$, а множителя «М» (мега) – умножение на 2^{20}– 1048576.

Организация запоминающих устройств – произведение числа хранимых слов на их разрядность. Видно, что это дает информационную емкость ЗУ, однако при одной и той же информационной емкости организация ЗУ может быть различной, так что организация является самостоятельным важным параметром.

Быстродействие (*производительность*) ЗУ оценивают временами считывания, записи и длительностями циклов чтения/записи.

Время считывания – интервал между моментами появления сигнала чтения и слова на выходе ЗУ.

Время записи – интервал после появления сигнала записи, достаточный для установления ЗЯ в состояние, задаваемое входным словом. Минимально допустимый интервал между последовательными чтениями или записями образует соответствующий цикл. Длительности циклов могут превышать времена чтения или записи, так как после этих операций может потребоваться время для восстановления необходимого начального состояния ЗУ.

Время чтения, записи и длительности циклов – традиционные параметры. Для некоторых современных ЗУ они должны быть дополнены новыми. Причиной является более сложный характер доступа к хранимым данным, когда обращение к первому слову некоторой группы слов (пакета) требует большего времени, чем обращение к последующим. Для таких режимов вводят параметр *времени доступа при первом обращении* (Latency) и *темпа передач* для последующих слов пакета (Bandwidth). Темп передач в свою очередь оценивается двумя значениями – *предельным* (внутри пакета) и *усредненным* (с учетом Latency). С уменьшением пакета усредненный темп снижается, все более отличаясь от предельного.

Помимо указанных основных параметров для ЗУ указывают еще целый набор временных интервалов. Перечисленные выше динамические параметры являются *эксплуатационными* (измеряемыми). Кроме них, существует ряд *режимных параметров*, обеспечение которых необходимо для нормального функционирования ЗУ, поскольку оно имеет

несколько сигналов управления, для которых должно быть обеспечено определенное взаимное расположение во времени. Для этих сигналов задаются длительности и ограничения по взаимному положению во времени.

Один из возможных наборов сигналов ЗУ включает следующие сигналы:

• A – адрес, разрядность которого определяется числом ячеек ЗУ, т. е. максимально возможным числом хранимых в ЗУ слов. Для ЗУ типично число ячеек, выражаемое целой степенью двойки. Адрес является номером ячейки, к которой идет обращение. Очевидно, что разрядность адреса связана с числом хранимых слов N соотношением $n = \log 2N$ (имеется в виду максимально возможное число хранимых слов). Например, ЗУ с информационной емкостью 64К слов имеет 16-разрядные адреса, выражаемые словами $A = A_{15}A_{14}A_{13}\ldots A_0$;

• CS – (Chip Select), или CE (Chip Enable), который разрешает или запрещает работу данной микросхемы;

• R/W – (Read/Write) задает выполняемую операцию (при единичном значении – чтение, при нулевом – запись);

• DI и DO (Data Input) и (Data Output) – шины входных и выходных данных, разрядность которых m определяется организацией ЗУ (разрядностью его ячеек). В некоторых ЗУ эти линии объединены.

Требования к взаимному временному положению двух сигналов (A и B) задаются временами предустановки, удержания и сохранения.

Время предустановки сигнала A относительно сигнала B $t_{su(A–B)}$ есть интервал между началами обоих сигналов.

Время удержания $t_{H(A–B)}$ – это интервал между началом сигнала A и окончанием сигнала B.

Время сохранения $t_{V(A–B)}$ – интервал между окончанием сигнала A и окончанием сигнала B.

Длительности сигналов обозначаются как t_w (индекс от слова *Width* – ширина).

Для ЗУ характерна такая последовательность сигналов. Прежде всего подается адрес, чтобы последующие операции не коснулись какой-либо другой ячейки, кроме выбранной. Затем разрешается работа микросхемы сигналом CS (CF) и подается строб чтения Записи R/W (взаимное положение сигналов CS и R/W для разных ЗУ может быть различным). Если задана, например, операция чтения, то после подачи перечисленных сигналов ЗУ готовит данные для чтения, что требует определенного времени. Задний фронт сигнала R/W, положение которого во времени должно обеспечивать установление правильных данных на выходе ЗУ, считывает данные.

Индексом *A* (от слова Access) обозначаются согласно стандарту времена доступа – интервалы времени от появления того или иного управляющего сигнала до появления информационного сигнала на выходе. *Время доступа относительно сигнала адреса* обозначается, если следовать правилу, как $t_{A(A)}$, но часто просто как t_A. Аналогично этому, *время доступа относительно сигнала CS*, т. е. $t_A(cs)$ часто обозначается просто как t_{CS}. Время t_A называют также временем выборки, а время t_{CS} – временем выбора.

Кроме отмеченных параметров для ЗУ используется и ряд других (уровни напряжений, токи, емкости выводов, температурный диапазон и т. д.), которые не требуют специального рассмотрения, так как они традиционны для цифровой схемотехники. Исключение составляет *свойство энергонезависимости*, т. е. способности ЗУ сохранять данные при отключении напряжения питания энергонезависимость может быть естественной, т. е. присущей самим ЗУ, или искусственной, достигаемой введением резервных источников питания, автоматически подключаемых к накопителю ЗУ при снятии основного питания.

8.2. Классификация запоминающих устройств

Для классификации ЗУ важнейшим признаком является способ доступа к данным.

При адресном доступе код на адресном входе указывает ячейку, с которой ведется обмен. Все ячейки адресной памяти в момент обращения равнодоступны. Эти ЗУ наиболее разработаны, и другие виды памяти часто строят на основе адресной с соответствующими модификациями.

Адресные ЗУ *делятся на RAM* (Random Access Memory) *и ROM* (Read-Only Memory). Русские синонимы термина *RAM*: ОЗУ (оперативные ЗУ) или ЗУПВ (ЗУ с произвольной выборкой). Оперативные ЗУ хранят данные, участвующие в обмене при исполнении текущей программы, которые могут быть изменены в произвольный момент времени. Запоминающие элементы ОЗУ, как правило, не обладают энергонезависимостью.

В *ROM* (русский эквивалент – ПЗУ, т. е. постоянные ЗУ) содержимое либо вообще не изменяется, либо изменяется, но редко и в специальном режиме. Для рабочего режима – это «память только для чтения».

RAM делятся на статические и динамические. В первом варианте запоминающими элементами являются триггеры, сохраняющие свое

состояние, пока схема находится под питанием и нет новой записи данных. Во втором варианте данные хранятся в виде зарядов конденсаторов, образуемых элементами МОП-структур. Саморазряд конденсаторов ведет к разрушению данных, поэтому они должны периодически (каждые несколько миллисекунд) регенерироваться. В то же время плотность упаковки динамических элементов памяти в несколько раз превышает плотность упаковки, достижимую в статических *RAM*.

Регенерация данных в динамических ЗУ осуществляется с помощью специальных контроллеров. Разработаны также ЗУ с динамическими запоминающими элементами, имеющие внутреннюю встроенную систему регенерации, у которых внешнее поведение относительно управляющих сигналов становится аналогичным поведению статических ЗУ. Такие ЗУ называют *квазистатическими*.

Статические ЗУ называются *SRAM* (Static RAM), а динамические – *DRAM* (Dynamic RAM).

Статические ОЗУ можно разделить на асинхронные, тактируемые и синхронные (конвейерные). В *асинхронных* сигналы управления могут задаваться как импульсами, так и уровнями. Например, сигнал разрешения работы *CS* может оставаться неизменным и разрешающим на протяжении многих циклов обращения к памяти. В *тактируемых* ЗУ некоторые сигналы обязательно должны быть импульсными, например, сигнал разрешения работы *CS* в каждом цикле обращения к памяти должен переходить из пассивного состояния в активное (должен формироваться фронт этого сигнала в каждом цикле). Этот тип ЗУ называют часто синхронным. Здесь использован термин «тактируемые», чтобы «освободить» термин «синхронные» для новых типов ЗУ, в которых организован *конвейерный* тракт передачи данных, синхронизируемый от тактовой системы процессора, что дает повышение темпа передач данных в несколько раз.

Динамические ЗУ *характеризуются наибольшей информационной емкостью и невысокой стоимостью*, *поэтому именно они используются как основная память компьютеров*. Поскольку от этой памяти требуется высокое быстродействие, разработаны многочисленные архитектуры повышенного быстродействия, перечисленные в классификации.

Статические ЗУ в 4...5 раз дороже динамических и приблизительно во столько же раз меньше по информационной емкости. Их достоинством является высокое быстродействие, а типичной областью использования – схемы кэш-памяти.

Постоянная память типа *ROM* (*M*) программируется при изготовлении методами интегральной технологии с помощью одной из исполь-

зуемых при этом масок. В русском языке ее можно назвать памятью типа ПЗУМ (ПЗУ масочные). Для потребителя это в полном смысле слова постоянная память, так как изменить ее содержимое он не может.

В следующих трех разновидностях *ROM* в обозначениях присутствует буква *P* (от Programmable). Это *программируемая пользователем память* (в русской терминологии ППЗУ – программируемые ПЗУ). Ее содержимое записывается либо однократно (в *PROM*), либо может быть заменено путем стирания старой информации и записи новой (в *EPROM* и *EEPROM*). В *EPROM* стирание выполняется с помощью облучения кристалла ультрафиолетовыми лучами, ее русское название РПЗУ-УФ (репрограммируемое ПЗУ с УФ-стиранием). В *EEPROM* стирание производится электрическими сигналами, ее русское название РПЗУ-ЭС (репрограммируемое ПЗУ с электрическим стиранием). Английские названия расшифровываются как Electrically Programmable ROM и Electrically Erasable Programmable ROM. Программирование *PROM* и репрограммирование *EPROM* и *EEPROM* производятся в обычных лабораторных условиях с помощью либо специальных программаторов, либо специальных режимов без специальных приборов (для *EEPROM*). *Память типа* Flash по запоминающему элементу подобна памяти типа *EEPROM* (или иначе E^2PROM), но имеет структурные и технологические особенности, позволяющие выделить ее в отдельный вид.

Запись данных и для *EPROM* и для E^2PROM производится электрическими сигналами.

В ЗУ *с последовательным доступом* записываемые данные образуют некоторую очередь. Считывание происходит из очереди слово за словом либо в порядке записи, либо в обратном порядке. Моделью такого ЗУ является последовательная цепочка запоминающих элементов, в которой данные передаются между соседними элементами.

Прямой порядок считывания имеет место в *буферах FIFO* с дисциплиной «первый пришел – первый вышел» (First In – First Out), а также в файловых и циклических ЗУ.

Разница между памятью *FIFO* и файловым ЗУ состоит в том. что в *FIFO* запись в пустой буфер сразу же становится доступной для чтения, т. е. поступает в конец цепочки (модели ЗУ). В файловых ЗУ данные поступают в начало цепочки и появляются на выходе после некоторого числа обращений, равного числу элементов в цепочке. При независимости операций считывания и записи фактическое расположение данных в ЗУ на момент считывания не связано с каким-либо внешним признаком. Поэтому записываемые данные объединяют в

блоки, обрамляемые специальными символами конца и начала (файлы). Прием данных из файлового ЗУ начинается после обнаружения приемником символа начала блока.

В циклических ЗУ слова доступны одно за другим с постоянным периодом, определяемым емкостью памяти. К такому типу среди полупроводниковых ЗУ относится видеопамять (*VRAM*).

Считывание в обратном порядке свойственно стековым ЗУ, для которых реализуется дисциплина «последний пришел – первый вышел». Такие ЗУ называют *буферами LIFO* (Last In – First Out).

Время доступа к конкретной единице хранимой информации в последовательных ЗУ представляет собою случайную величину. В наихудшем случае для такого доступа может потребоваться просмотр всего объема хранимых данных.

Ассоциативный доступ реализует поиск информации по некоторому признаку, а не по ее расположению в памяти (адресу или месту в очереди). В наиболее полной версии все хранимые в памяти слова одновременно проверяются на соответствие признаку, например, на совпадение определенных полей слов (тегов – от англ. *tag*) с признаком, задаваемым входным словом (теговым адресом). На выход выдаются слова, удовлетворяющие признаку. Дисциплина выдачи слов, если тегу удовлетворяют несколько слов, а также дисциплина записи новых данных могут быть разными. Основная область применения ассоциативной памяти в современных компьютеров – кэширование данных.

Технико-экономические параметры ЗУ существенно зависят от их схемотехнологической реализации. По этому признаку также возможна классификация ЗУ, однако удобнее рассматривать этот вопрос применительно к отдельным типам памяти.

8.3. Структура постоянных запоминающих устройств

Программируемые логические матрицы (ПЛМ), рассмотренные ранее, и *постоянные запоминающие устройства* (ПЗУ) предназначены для реализации кодовых преобразователей по принципу декодер-кодер.

В ПЛМ используются неполные дешифраторы. Так, в К556 РТ1 из 16 входов дешифратора полного, который имеет $2^{16} = 65536$ ЛЭ И, используется всего 48.

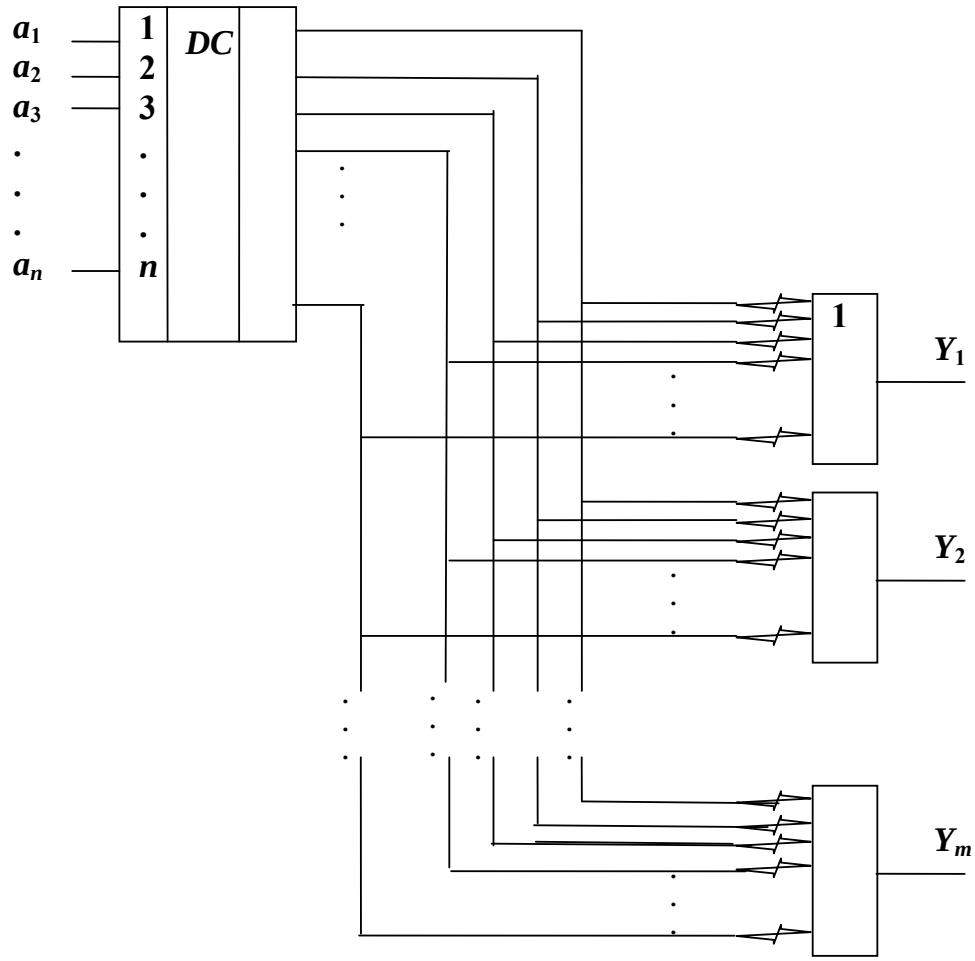


Рис. 8.1. Структура ПЗУ

Исторически первыми стали выпускаться микросхемы с полными дешифраторами на входе. Их называют *постоянными запоминающими устройствами* ПЗУ – READ ONLY MEMORY – (память только для чтения) *ROM*. По принятой терминологии входной код ПЗУ называют адресом, а выходной – разрядами хранимого слова. Емкость ПЗУ обычно описывают как произведение числа возможных слов, равного 2^n, где n – число входов, на число разрядов выходного слова m, например, для КР 556 РТ1 емкость (256 х 4).

Самые первые ПЗУ масочного типа программировались на заводе-изготовителе в соответствии с маской, закрывающей или открывающей те или иные связи на входах ЛЭ ИЛИ. И сейчас выпускаются такие ИМС. Например, К155РЕ21 – преобразователь 6^x разрядного входного двоичного кода в 12-разрядный код управления матричным индикатором 5 х 7 типа АЛС 340А в соответствии с буквами русского алфавита, РЕ 22 – в соответствии с цифрами десятичными от 0 до 9, алгебраическими и другими знаками. К568 РЕ 1 – таблица синусов.

Выпускаются программируемые ПЗУ – PROGRAMMABLE ROM, в которых выжигаемые перемычки позволяют потребителю самому внести информацию в микросхему так, как описано выше для ПЛМ.

Пример программирования: 4-разрядный двоичный входной код преобразуем в семисегментный код индикатора.

В ПЛМ Л556 РТ1 16 входов, которые напрямую или через входные инверторы подключены к входам 48 ЛЭ И входного дешифратора ПЛМ. Используем лишь первые 4 входа $a_1a_2a_3a_4$, следовательно, выжигаемые перемычки, соединяющие остальные входы микросхем и их инверсии с входами нужных нам 10 ЛЭ И дешифратора, должны быть разрушены. Входы остальных 38 ЛЭ И могут быть подключены как угодно, ведь эти ЛЭ И не используются.

Таблица 8.1

Таблица состояний преобразования 4-разрядного двоичного входного кода в семисегментный код индикатора

№	Входы	Выходы							
		g	*f*	*e*	*d*	*c*	*b*	*a*	
	a_{16} … a_5 a_4 a_3 a_2 a_1	Y_8	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1
0	*x* … *x* 0 0 0 0	*x*	0	1	1	1	1	1	1
1	*x* … *x* 0 0 0 1	*x*	0	0	0	0	1	1	0
2	*x* … *x* 0 0 1 0	*x*	1	0	1	1	0	1	1
3	*x* … *x* 0 0 1 1	*x*	1	0	0	1	1	1	1
4	*x* … *x* 0 1 0 0	*x*	1	1	0	0	1	1	0
5	*x* … *x* 0 1 0 1	*x*	1	1	0	1	1	0	1
6	*x* … *x* 0 1 1 0	*x*	1	1	1	1	1	0	1
7	*x* … *x* 0 1 1 1	*x*	0	0	0	0	1	1	1
8	*x* … *x* 1 0 0 0	*x*	1	1	1	1	1	1	1
9	*x* … *x* 1 0 0 1	*x*	1	1	0	1	1	1	1

Входной дешифратор из 10 ЛЭ И в ответ на 4-разрядный входной двоичный код должен вырабатывать на своих выходах, т. е. на первых десяти вертикальных шинах ПЛМ, унитарный код, другими словами, только на одном их этих десяти выходов должна быть лог.1, причем номер этого выхода дешифратора соответствует двоичному коду на входе.

Из 8 ЛЭ ИЛИ, составляющих выходной шифратор ПЛМ используем только первые 7, больше нет необходимости. Каждые их этих 7 ЛЭ ИЛИ имеет 48 входов, из которых в нашем примере используем только первые 10, остальные 38 в принципе должны быть отключены от своих вертикальных шин путем разрушения перемычек на этих входах.

Используемые в примере 10 входов каждого их 7 ЛЭ ИЛИ подключаются к вертикальным шинам – выходам дешифратора в соответствии с таблицей истинности. К примеру, выходы ЛЭ ИЛИ формирующего выходной сигнал *Y*5, соответствующий сегменту *C*, надо оставить подключенными к вертикальным шинам с номерами 0, 2, 6 и 8. Булево выражение имеет вид:

$$Y5 = \bar{a}_1\,\bar{a}_2\,\bar{a}_3\,\bar{a}_4 + \bar{a}_1\,a_2\,\bar{a}_3\,\bar{a}_4 + a_1\,\bar{a}_2\,a_3\,\bar{a}_4 + \bar{a}_1\,\bar{a}_2\,\bar{a}_3\,a_4.$$

Программируемые инверторы, реализованные на исключающих ИЛИ, до прожига, когда один их входов исключающего ИЛИ подключен через перемычку к шине питания, инвертируют сигналы, проходящие через них. Если не требуется инвертирование, то все перемычки, соединяющие входы = 1 с шиной питания, должны быть разрушены.

Существуют репрограммируемые ПЗУ – РППЗУ, *EPROM* – ERASABLE, стираемое PROGRAMMABLE READ ONLY MEMORY. В них обеспечивается неоднократное программирование, так как нет выжигаемых бесповоротно перемычек. Время гарантированного сохранения информации бывает от нескольких месяцев до нескольких лет. Стирание чаще всего производится облучением кристалла микросхемы РППЗУ ультрафиолетовым светом через специальное стеклянное окошко сверху корпуса микросхемы. В остальное время окошко должно быть заклеено хотя бы изолентой, чтобы дневной или электрический свет не стер информацию.

8.4. Буфер FIRST IN – FIRST OUT (*FIFO*)

Запоминающие устройства служат для хранения очередей данных и называются также буферными регистрами. *FIFO* – FIRST IN FIRST OUT – «первым вошел–первым вышел» (синоним – очередь в столовой).

Буфер типа *FIFO* ставится между источником и приемником данных, когда потери информации не допустимы, а приемник не может быстро обработать данные. Строится на основе регистровых файлов с раздельной адресацией записи и чтения.

Сигнал «поставить в очередь», поступая на вход *WE* разрешения записи, записывает этим данные, стоящие на входе *DI* в тот регистр памяти, номер которого *CT*1 – счетчик хвоста. По срезу этого импульса «поставить в очередь» выходной код *CT*1 увеличивается на 1, определяя адрес записи следующих данных.

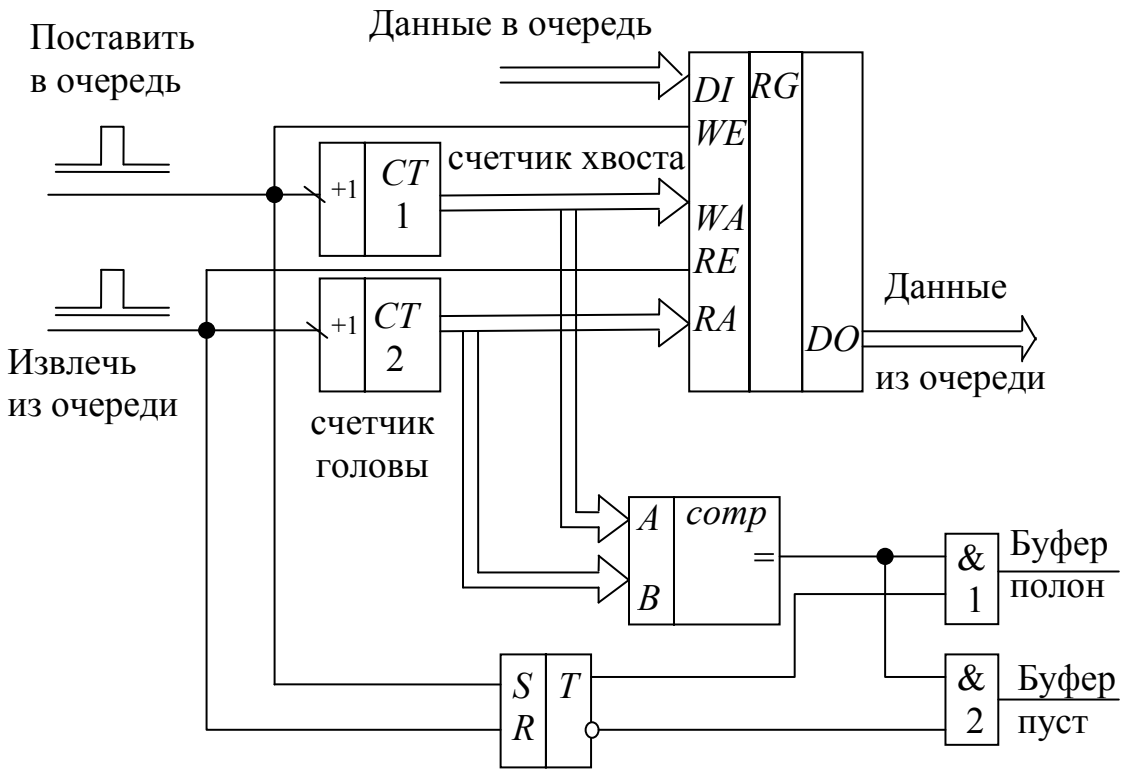


Рис. 8.2. Структурная схема буфера типа *FIFO*

При сигнале «извлечь из очереди», поданном на *RE* – разрешения чтения, на выходе *DO* появляется слово, которое хранилось в регистре по адресу, указанному выходным кодом *CT*2 – счетчика головы. По срезу импульса «извлечь из очереди» код этого счетчика *CT*2 увеличивается на 1, подготавливая к чтению следующее слово.

Если коды на выходах счетчиков *CT*1 хвоста и *CT*2 головы равны, что определяет компаратор, то на выходе ЛЭ И1 будет сигнал «буфер полон», если последним обращением к буферу был импульс «поставить в очередь». Это значит, что больше записывать в буфер нельзя, надо срочно подключать еще один буфер *FIFO*. Если же при *CT*1 = *CT*2 последним было «извлечь из очереди», то данный буфер *FIFO* пуст и обращаться за данными к этому буферу нет смысла.

ГЛАВА 9. АНАНАЛОГО-ЦИФРОВЫЕ И ЦИФРО-АНАЛОГОВЫЕ ПРЕОБРАЗОВАТЕЛИ

9.1. Сравнительный обзор цифровых и аналоговых систем

Цифровая величина принимает значение, которое является одним из двух возможных, например, 0 или 1, *низкий* или *высокий* уровень напряжения, ложь или истина и т. п. В действительности цифровые величины (к примеру, уровень напряжения) могут принимать любые значения из некоего диапазона (считается, что все значения из этого диапазона соответствуют данной цифровой величине). Например, для устройств на ТТЛ известно, что напряжение от 0 до 0,8 В соответствует лог.0; напряжение от 2 до 5 В – лог.1.

Любое значение напряжения, попадающее в диапазон от 0 до 0,8 В, будет расцениваться как лог.0, а любое значение напряжения, попадающее в интервал от 2 до 5 В, будет расцениваться как лог.1. Точные значения не существенны, потому что цифровые схемы одинаково реагируют на все напряжения, находящиеся в одном и том же диапазоне значений.

Аналоговая величина, наоборот, может принимать любое значение в непрерывном диапазоне величин и, что самое важное, интерес представляет именно ее точное значение. Например, пусть с выхода аналогового термометра было снято напряжение 2,76 В, которое соответствует, скажем, температуре 27,6 °С. Если бы было зафиксировано другое значение напряжения, например, 2,34 или 3,78 В, то оно бы представляло собой совершенно другую температуру. Иначе говоря, каждое возможное значение аналоговой величины представляет отдельный интерес. Можно привести еще один пример: выходной сигнал звукового усилителя подается в акустическую систему. Подаваемое напряжение – аналоговая величина, потому что каждое возможное значение напряжения формирует новый звук в акустической системе.

Большинство физических переменных по своей природе являются аналоговыми и могут принимать любое значение из определенного диапазона. В качестве примеров можно назвать температуру, давление, интенсивность света, звуковые сигналы, координату, скорость вращения или течения. Цифровые системы осуществляют все операции над такими величинами исключительно в цифровой форме, используя для этого элементы цифровой схемотехники. Любую ин-

формацию, которая должна быть введена в цифровую систему, необходимо сначала преобразовать в цифровую форму. Выходные же сигналы цифровой системы выражены в цифровом виде изначально. Когда цифровая система, такая как компьютер, должна использоваться для мониторинга и/или контроля за выполнением физического процесса, то приходится сталкиваться с различием между цифровой природой компьютера и аналоговой природой переменных данного процесса. Эта мысль наглядно проиллюстрирована на рис. 9.1. Приведенная блок-схема имеет пять элементов, составляющих систему мониторинга и контроля физических переменных с помощью компьютера.

1. *Датчик*, или *преобразователь*. Любая физическая переменная обычно не является величиной, каким-то образом связанной с электричеством. *Преобразователем* называется устройство, которое преобразует физические переменные в электрические. Можно назвать сразу несколько наиболее распространенных преобразователей: терморезисторы, фотоприемники, фотодиоды, измерители потока и датчики давления, тахометры.

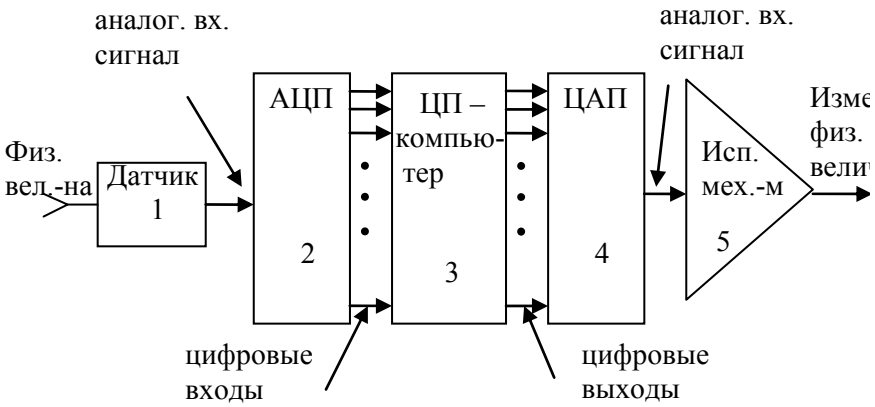


Рис. 9.1. Пример сопряжения компьютера и аналогового мира

Электрический выходной сигнал преобразователя представляет собой ток или напряжение в аналоговой форме. Значения тока пропорциональны физической величине, за которой нужно следить. Пусть, например, физическая величина представляет собой температуру воды в большой емкости, которая наполняется одновременно горячей и холодной водой. И пусть температура воды колеблется от 8 до 15 °С. Терморезистор и соответствующая схема преобразуют измеренное значение температуры воды в напряжение, величина которого варьируется от 800 до 1500 мВ. Обратите внимание, что на выходе преобразователя наблюдается сигнал, прямо пропорциональный температуре, т. е. изменение темпе-

ратуры на 1 °С приводит к изменению выходного напряжения на 10 мВ. Такой коэффициент пропорциональности был выбран исключительно с целью удобства.

2. *Аналого-цифровой преобразователь* (АЦП). Аналоговый электрический сигнал с выхода преобразователя служит входным сигналом для АЦП. АЦП предназначен для преобразования аналоговых сигналов в цифровые. Цифровой сигнал на выходе этого устройства состоит из определенного количества бит, которые соответствуют величине аналогового сигнала. Например, АЦП можно использовать для преобразования аналоговых уровней напряжения (от 800 до 1500 мВ) в двоичные значения от 01010000 (80) до 10010110 (150). Обратите внимание, что двоичный выходной сигнал АЦП пропорционален аналоговому напряжению и каждая единица цифрового сигнала соответствует 10 мВ.

3. *Компьютер.* Цифровой сигнал, величина которого соответствует состоянию физической переменной, передается из АЦП на компьютер, сохраняющий цифровое значение величины и обрабатывающий двоичный код в соответствии с заложенной в него программой. Программа может производить какие-то расчеты или операции над поступившим сигналом, в результате чего компьютер сформирует выходной сигнал, который можно использовать для контроля температуры.

4. *Цифро-аналоговый преобразователь* (ЦАП). Цифровой сигнал с выхода компьютера подается на ЦАП, который преобразует его в пропорциональное аналоговое напряжение или ток. Например, компьютер выдал цифровой сигнал, значения которого колебались от 00000000 до 11111111. ЦАП преобразует этот сигнал в напряжение, величина которого будет колебаться от 0 до 10 В.

5. *Исполнительный механизм*. Аналоговый сигнал с выхода ЦАП обычно поступает на какое-либо устройство или схему, которая служит исполнительным механизмом и управляет физической переменной. В примере с температурой воды исполнительный механизм может контролировать кран, регулирующий поток горячей воды, согласно уровню аналогового напряжения от ЦАП. Скорость потока воды будет варьироваться в зависимости от величины аналогового напряжения: при 0 В вода вообще не будет поступать в емкость, а при 10 В скорость заполнения будет максимальной.

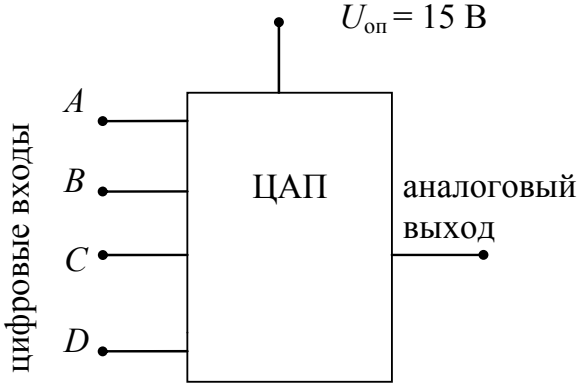
Таким образом, функция АЦП и ЦАП состоит в *сопряжении* полностью цифровых систем (например, компьютеров) с аналоговыми системами в окружающем мире. С недавних пор эта функция стала еще более важной, поскольку микрокомпьютеры вошли в такие области контроля за процессами, где ранее их использование казалось просто невозможным.

9.2. Цифро-аналоговое преобразование

9.2.1. Соотношения входного и выходного сигналов ЦАП

По сути, цифро-аналоговое преобразование состоит в том, что устройство (ЦАП) преобразует *цифровой* код (который представляет собой ту или иную величину и выражается непосредственно в двоичном или в двоично-десятичном кодах) в напряжение или ток, значения которых пропорциональны цифровому сигналу. На рис. 9.2, *а*, изображен четырехбитовый ЦАП.

Следует обратить внимание на вход опорного напряжения $U_{оп}$. Этот вход используется для того, чтобы задать величину полномасштабного или максимального значения напряжения, которое может быть сформировано на выходе ЦАП. Цифровые сигналы *A*, *B*, *C* и *D* обычно поступают на преобразователь с выходов регистра или другой цифровой системы. Эти четыре бита могут представлять $2^4 = 16$ различных двоичных чисел, как показано на рис. 9.2, *б*. Каждому представленному таким образом числу на выходе ЦАП соответствует свое уникальное значение напряжения. Фактически аналоговый выходной сигнал, $U_{вых}$ может быть равен стольким вольтам, сколько бит поступило на вход преобразователя. Однако этот сигнал может и вдвое превышать двоичное число на входе или еще как-то соотноситься с ним; все зависит от выбранного масштаба. Та же идея справедлива и в случае, если ЦАП формирует сигнал в виде уровней тока, а не напряжения.



а)

A	***B***	***C***	***D***	$U_{вых}$, **В**
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

б)

Рис. 9.2. Четырехбитовый ЦАП: *а* – схема; *б* – значения его выходного сигнала

В общем случае величина выходного аналогового сигнала равна:

$$\textit{аналоговый сигнал} = K \cdot \textit{цифровой сигнал},$$

где K – коэффициент пропорциональности, постоянная величина для заданного ЦАП, подключенного к фиксированному источнику опорного напряжения. Информация аналогового выходного сигнала может быть представлена, как уже известно, или током, или напряжением. Если это напряжение, то коэффициент K будет иметь размерность вольт, а в случае с током – измеряться в амперах.

Для ЦАП, показанного на рис. 9.2, $K = 1$ В, следовательно:

$$U_{\text{вых}} = (1\text{В}) \cdot \textit{цифровой сигнал}.$$

Можно использовать это значение коэффициента, чтобы подсчитать значение $U_{\text{вых}}$ для любых значений цифрового сигнала. Например, если на вход поступила кодовая комбинация 1100_2–12_{10}, то

$$U_{\text{вых}} = (1\text{В}) \cdot 12 = 12 \text{ В}.$$

9.2.2. Аналоговый выходной сигнал

Если быть точным, сигнал на выходе ЦАП нельзя назвать аналоговым потому, что даже при условии фиксированного опорного напряжения он может принимать только определенные значения: скажем, один из 16 возможных уровней выходного напряжения $U_{\text{вых}}$, как это показано в таблице на рис. 9.2 (в данном случае сигнал будет цифровым). Однако с увеличением количества входов число возможных значений выходного параметра (тока или напряжения) может быть увеличено, а разность между двумя ближайшими значениями – уменьшена. Это позволит формировать выходной сигнал, все более близкий к аналоговой величине, которая последовательно изменяется в каком-то установленном интервале значений. Другими словами, выходной сигнал ЦАП является «псевдоаналоговым». В литературе он называется аналоговым, однако это только аппроксимация реальной аналоговой величины.

9.2.3. Весовые коэффициенты входных битов

Следует отметить, что для ЦАП, показанного на рис. 10.2, каждый цифровой вход вносит свой вклад в формирование выходного аналогового сигнала. Это легко заметить, если внимательно изучить ситуации, когда активным является только один вход устройства

(табл. 9.1). Сигналам на каждом из входов присваиваются соответствующие *весовые коэффициенты*, или *веса*, которые определяются позицией данного бита в двоичном числе, поданном на ЦАП.

Таблица 9.1

Таблица соответствия лог.1 на входах и выходного аналогового сигнала

A	***B***	***C***	***D***	$U_{вых}$, **В**
0	0	0	1	1
0	0	1	0	2
0	1	0	0	4
1	0	0	0	8

Таким образом, вход *D*, который представляет собой младший значащий бит, имеет вес, равный 1 В; вход *C* имеет вес, равный 2 В; вход *B* – 4 В; а вход *A*, который представляет собой старший значащий бит, имеет наибольший вес, равный 8 В. Эти веса последовательно увеличиваются вдвое для каждого нового бита, начиная с младшего.

Итак, можно сделать следующий вывод: напряжение $U_{вых}$ представляет собой сумму весов входных битов. Например, чтобы найти значение $U_{вых}$ для входного кода 0111, нужно сложить веса единичных битов *A*, *B*, *C* и *D*, которые в сумме дадут 4 В + 2 В + 1 В = 7 В.

9.2.4. Разрешающая способность – (величина шага квантования)

Разрешающей способностью цифро-аналогового преобразователя называется наименьшее изменение аналогового сигнала на выходе в ответ на изменение цифрового сигнала на входе. Еще раз взглянув на таблицу (рис. 9.2, *б*), можно увидеть, что для приведенного ЦАП разрядность составила 1 В, так как при изменении входного сигнала напряжение $U_{вых}$ не может изменяться менее чем на 1 В. Разрядность ЦАП всегда равна весу младшего значащего бита.

Часто этот параметр еще называют величиной шага квантования, так как он представляет собой наименьшую величину выходного напряжения (или тока), на которую изменится выходной сигнал устройства при изменении входного значения на один шаг. Это очень хорошо проиллюстрировано на рис. 9.3, где изображен ЦАП, на входы которого поступают сигналы с выходов четырехбитового двоичного счетчика.

Счетчик непрерывно изменяет свои состояния (всего их 16) в моменты поступления импульсов синхронизации, а сигнал на выходе ЦАП представляет собой ступенчатую функцию, которая с каждым шагом возрастает на 1 В. Когда счетчик достигнет состояния 1111, на выходе ЦАП будет наблюдаться максимальный сигнал, равный 15 В; этот случай соответствует полномасштабному выходному сигналу. Когда счетчик сбросится в 0000, выход ЦАП также вернется в состояние 0 В. Разрешающая способность, или величина шага квантования, представляет собой высоту одной ступени функции (рис. 9.3); в данном случае каждый шаг равен 1 В.

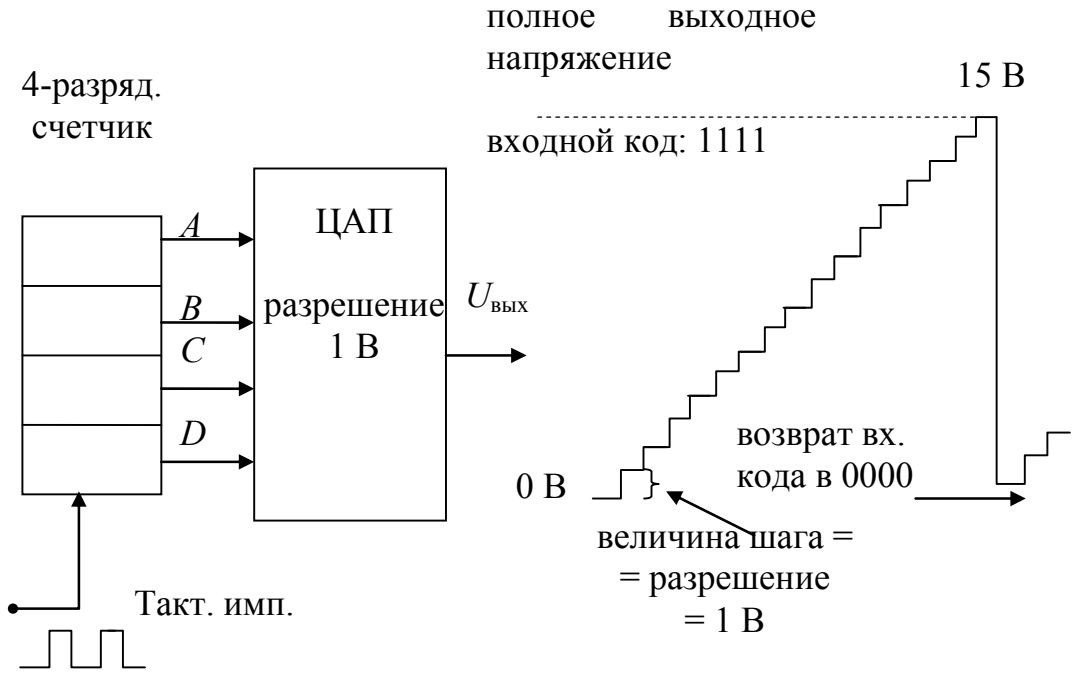


Рис. 9.3. Форма сигнала на выходе ЦАП, управляемого сигналами с двоичного счетчика

Таким образом, разрешающая способность (или величина шага) представляет собой тот же параметр, что и коэффициент пропорциональности между входным и выходным сигналами преобразователя:

$$\textit{аналоговый сигнал} = K \cdot \textit{цифровой сигнал}.$$

Новая интерпретация этого выражения заключается в том, что входной цифровой сигнал равен количеству шагов, где K – напряжение, или ток, каждого шага; аналоговый сигнал будет произведением этих двух параметров:

$$\textit{разрешающая способность} = K = \frac{A_{\text{П.С}}}{2^n - 1},$$

где $A_{\text{П.С}}$ – величина полномасштабного аналогового сигнала, а n – количество бит.

9.2.5. Разрешающая способность

Хотя *разрешающая способность* может быть выражена в виде величины напряжения, или тока, приходящейся на каждую ступень выходного сигнала, часто более удобно выражать ее в процентах относительно *полномасштабной величины напряжения*. Рассмотрим ЦАП, показанный на рис. 9.3. Он имеет полномасштабный выходной сигнал, величина которого составляет 15 В (в случае подачи на вход цифрового кода 1111). Величина шага квантования равна 1 В. Запишем процентную разрешающую способность следующим образом:

$$\textit{процентная разрешающая способность} =$$

$$= \frac{\textit{величина шага квантования}}{\textit{полномасштабный сигнал}} \cdot 100\,\% = \frac{1\,\text{В}}{15\,\text{В}} \cdot 100\,\% = 6{,}67\ \%.$$

Физический смысл разрешающей способности можно пояснить следующим образом. Цифро-аналоговый преобразователь не может формировать выходной сигнал, значения которого будут лежать в непрерывном диапазоне величин, поэтому, как уже говорилось, такой сигнал не является действительно аналоговым.

ЦАП выдает конечный набор дискретных выходных значений. В предыдущем разделе, где на примере измерения температуры воды рассматривались отличия аналоговых и цифровых величин, компьютер формировал выходной сигнал в дискретном (цифровом) виде, а затем уровни напряжения этого сигнала преобразовывались в аналоговый сигнал со значениями от 0 до 10 В, который контролировал работу крана с горячей водой. Разрядность (количество бит) рассмотренного ЦАП показывала, сколько различных значений напряжения может формировать и передавать на исполнительный механизм компьютер. При использовании шестибитового ЦАП и диапазона напряжений от 0 до 10 В всего будет возможно 63 различных ступени по 0,159 В каждая. При использовании восьмибитового ЦАП будет возможно уже 255 ступеней по 0,039 В каждая. Чем большим будет количество бит, тем лучшую разрешающую способность (и тем меньшую величину шага) будет иметь преобразователь.

Разработчик цифровых систем должен уметь определять необходимую для работы конкретной схемы разрядность, которая показывает, насколько близко цифровой сигнал с выхода ЦАП соответствует данному аналоговому сигналу. Обычно стоимость цифро-аналогового преобразователя возрастает с увеличением его разрешающей способности, поэтому следует использовать минимально необходимое количество бит.

9.3. Аналого-цифровое преобразование

9.3.1. Аналого-цифровой преобразователь

Аналого-цифровым преобразователем (АЦП) называется устройство, которое преобразует аналоговый входной сигнал в цифровой код, соответствующий этому сигналу. Процесс аналого-цифрового преобразования обычно более сложный и занимает больше времени, чем цифроаналоговое преобразование; кроме того, существует достаточно много различных методов выполнения этой операции [9].

В некоторых широко применяемых типах АЦП используются ЦАП, которые являются составной частью такого устройства. На рис. 9.4 показана функциональная схема этого класса преобразователей.

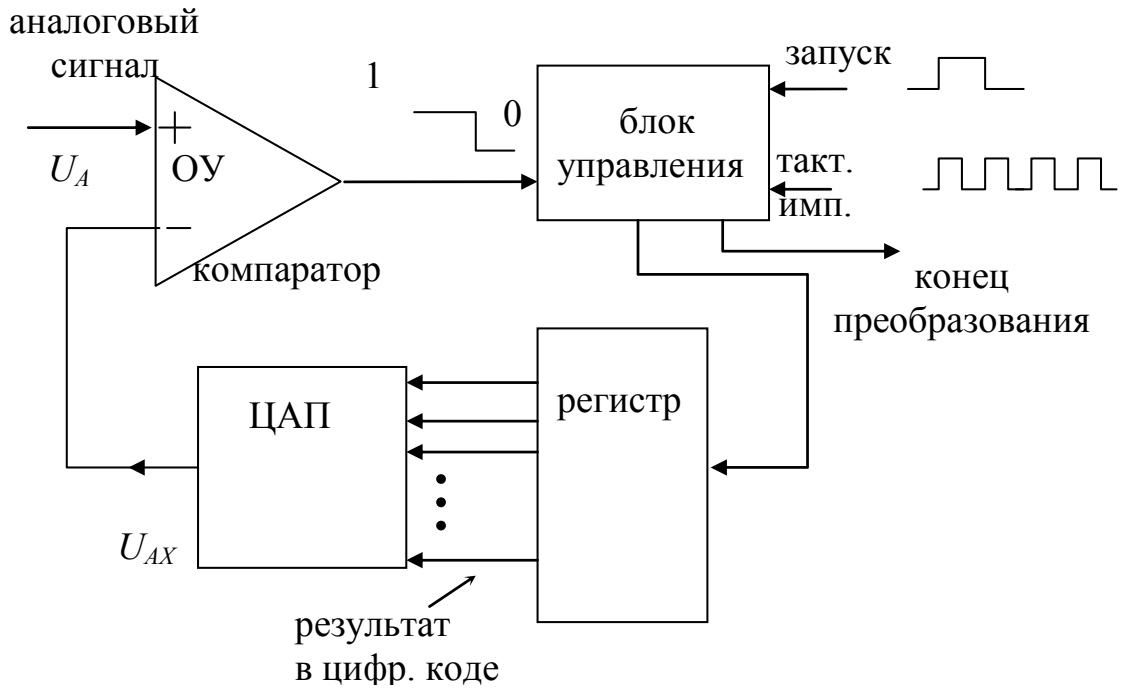


Рис. 9.4. Функциональная схема одного из классов аналого-цифровых преобразователей

Согласование во времени обеспечивается внешним сигналом синхронизации. Устройство управления содержит схемные элементы, с помощью которых происходит формирование заданной последовательности операций в ответ на импульс запуска (START COMMAND), поступление которого начинает процесс преобразования. Компаратор на операционном усилителе имеет два *аналоговых* входа и один *цифровой* выход, состояние которого зависит от того, какой из поданных аналоговых сигналов больше.

Процесс работы такого аналого-цифрового преобразователя можно описать в несколько этапов:

1. Импульс запуска (START COMMAND) начинает работу схемы.

2. Устройство управления со скоростью, определяемой частотой сигнала синхронизации, последовательно изменяет двоичные числа, которые поступают на регистр.

3. Двоичное число из регистра преобразуется с помощью ЦАП в аналоговое напряжение U_{AX}.

4. Компаратор сравнивает величину напряжения U_{AX} с аналоговым напряжением U_A. Пока $U_{AX} < U_A$, с выхода компаратора снимается сигнал с *высоким* уровнем напряжения. Если же U_{AX} превысит U_A на напряжение, равное пороговому (U_T), то на выходе компаратора установится сигнал с *низким* уровнем, который остановит процесс записи чисел в регистр. В этой точке напряжение U_{AX} будет приблизительно равно напряжению U_A. Число, записанное в регистр непосредственно перед этой операцией сравнения, будет представлять собой цифровой эквивалент U_A с погрешностью, которая определяется разрядностью и точностью системы.

5. Устройство управления формирует сигнал окончания преобразования (*EOC*), который показывает, что преобразование выполнено.

Вариации этой схемы аналого-цифрового преобразования могут отличаться только способом, которым устройство управления изменяет числа, записываемые в регистр. Однако все основные принципы преобразования останутся теми же, а число, соответствующее цифровой величине напряжения, всегда снимается с запоминающего регистра после окончания процесса преобразования.

9.3.2. Разрешающая способность и точность аналого-цифровых преобразователей

Очень важно знать виды ошибок, связанных с любыми измерительными процессами. Постоянным источником погрешностей интегрирующего метода аналого-цифрового преобразования является

тот факт, что разрешающая способность или величина шага квантования АЦП, которая представляет собой наименьшую величину измерения, определяется соответствующим параметром встроенного ЦАП. Представьте себе, что кто-то пробует измерить рост баскетболистов с помощью 30-сантиметровой линейки без делений. Если указывать рост только целым количеством таких линеек и при этом округлять его в сторону увеличения, то окажется, что все игроки, чей рост превышает 1,8 м, будут записаны как 2-метровые великаны! Точно так же и выходное напряжение U_{AX} представляет собой ступенчатую функцию, которая дискретно возрастает до тех пор, пока не превысит входной сигнал U_A. Уменьшая величину шага квантования, можно уменьшить погрешность, но между реальной (аналоговой) величиной и полученным цифровым значением всегда будет небольшая разница. Этот вид ошибки называется *погрешностью квантования*. Итак, величина U_{AX} – это лишь приближение к значению, поэтому что касается U_A, можно ожидать, что в лучшем случае эти два напряжения будут отличаться не более чем на 10 мВ, так как это соответствует величине шага квантования встроенного ЦАП.

Погрешность квантования, которая может быть уменьшена с помощью увеличения разрядности счетчика и ЦАП, иногда еще называют *погрешностью младшего значащего бита с коэффициентом* +1, указывая на то, что результат может отклоняться от истинного значения на величину, равную весу младшего разряда.

Посмотрим на проблему точности с другой стороны. Входной сигнал U_A может принимать *бесконечное* число значений из диапазона от 0 В до полномасштабного сигнала. Приближенное значение U_{AX}, однако, может принимать только ограниченное число дискретных значений, т. е. какой-то небольшой диапазон аналоговых значений U_A имеет одно и то же фиксированное дискретное значение, которое соответствует этому диапазону.

Другими словами, напряжение U_A должно измениться как минимум на 10 мВ (величина разрешающей способности), чтобы схема могла отреагировать на изменение входного сигнала соответствующим изменением цифрового кода на выходе.

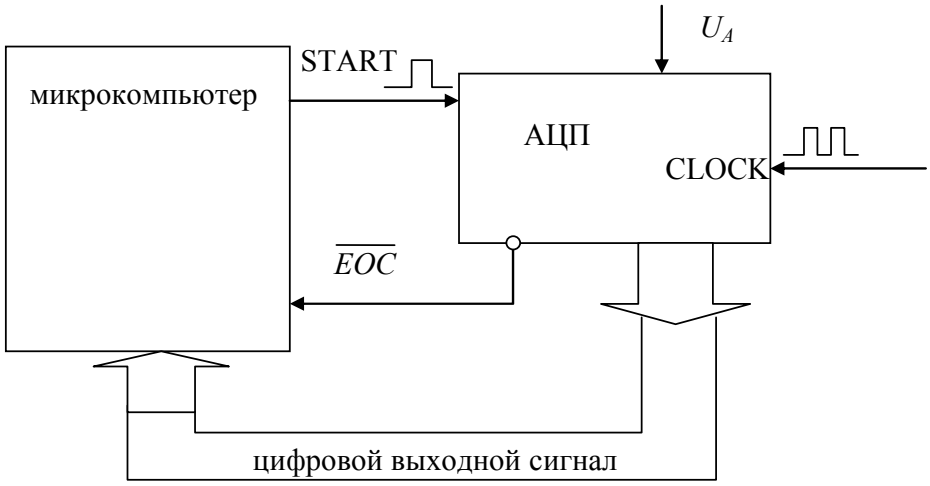
Так же как и в ЦАП, понятие *точности* не связано с понятием разрешающей способности напрямую, а зависит от прецизионности компонентов схемы (например, от точности изготовления компаратора, резисторов и ключей встроенного ЦАП, напряжения опорного источника питания и т. д.). Погрешность, равная 0,01 % полномасштаб-

ного сигнала, говорит о том, что результат с выхода АЦП соответствует входному сигналу с погрешностью 0,01 % от величины полномасштабного сигнала, а ошибка преобразования вызвана неточностью параметров внутренних компонентов преобразователя. Эта ошибка *прибавляется* к погрешности квантования, которая зависит от разрешающей способности. Оба указанных источника ошибок аналого-цифровых преобразователей обычно имеют один и тот же порядок величины.

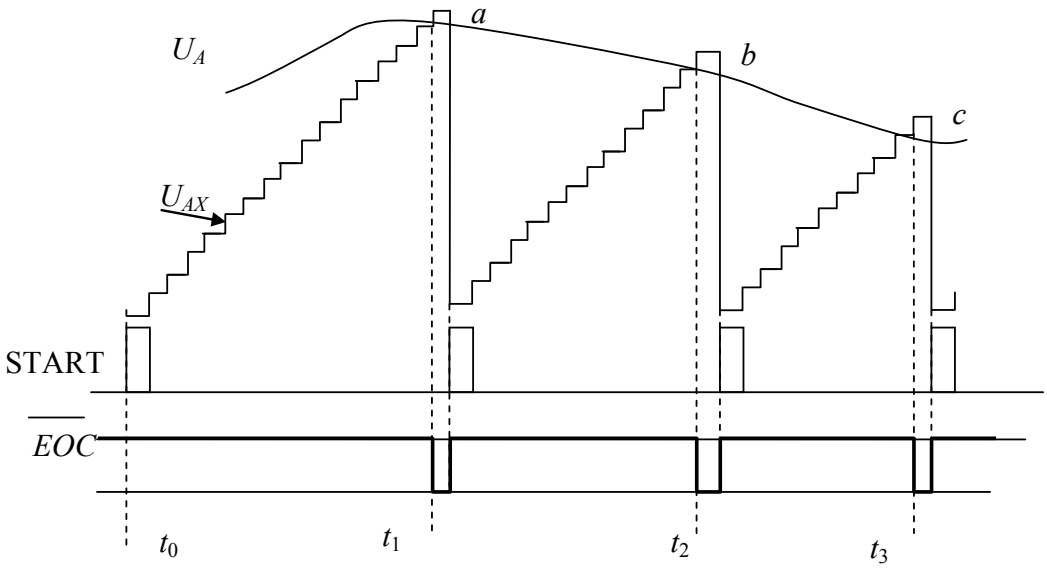
9.3.3. Сбор данных

Существует множество приложений, в которых аналоговые данные должны быть *оцифрованы* (т. е. преобразованы в цифровой вид) и переданы в память компьютера. Процесс, с помощью которого информация в цифровой форме поступает на компьютер, называется *сбором данных.* Запись значения одной точки данных называется выборкой, а сама точка называется *отсчетом*. Компьютер может осуществлять над данными разнообразные операции, зависящие от области применения информации. Сохраняя данные (например, при цифровой записи аудио- или видеоинформации) либо отображая сигналы на цифровом осциллографе, встроенный микрокомпьютер может запоминать данные и передавать их на ЦАП, чтобы позднее воспроизвести исходный аналоговый сигнал. Управляя этим процессом, компьютер может также обрабатывать информацию, например, осуществлять над ней какие-то арифметические действия, чтобы определить, какой выход следует возбудить.

На рис. 9.5, *а* показано, как подключается микропроцессор к интегрирующему АЦП для сбора данных. Для того чтобы начать очередной процесс преобразования, компьютер формирует импульсы запуска (START). Сигнал окончания преобразования $\overline{EOC}$ с выхода АЦП поступает на компьютер, который отслеживает состояние этого сигнала для того, чтобы обнаружить момент, когда цикл аналого-цифрового преобразования будет завершен; затем уже в цифровой форме компьютер передает данные с выхода АЦП в память.



а)



б)

Рис. 9.5. Сбор данных на основе компьютера: *а* – типичная система; *б* – формы сигналов показывают, каким образом компьютер начинает очередной цикл преобразования и загружает цифровые данные в память после завершения этого цикла

Формы сигналов на рис. 9.5, *б* показывают, каким образом на компьютер поступает оцифрованный аналоговый сигнал U_A. На рисунке ступенчатая функция U_{AX}, которая формируется в аналого-цифровом преобразователе, наложена на аналоговый сигнал U_A. Процесс начинается в момент t_0, когда компьютер выдает импульс запуска, обозначающий начало преобразования. Цикл заканчивается в момент t_1, когда ступенчатая функция превышает уровень сигнала U_A; в этот же момент сигнал $\overline{EOC}$ переходит в состояние с низким уровнем напряжения. Спад (задний фронт) сигнала $\overline{EOC}$ сообщает компью-

теру о том, что АЦП получил цифровой эквивалент напряжения U_A в точке *a* и компьютер загружает полученные данные в память.

Через некоторое время после момента компьютер формирует новый импульс запуска, начиная тем самым следующий цикл преобразования. Обратите внимание, что при этом ступенчатая функция спадает до 0, а сигнал $\overline{EOC}$ возвращается в состояние с высоким уровнем, так как импульс START сбрасывает счетчик АЦП. Второй цикл преобразования закончится в момент t_2, когда ступенчатая функция вновь превысит сигнал U_A. Компьютер загрузит новые цифровые данные в память; теперь они соответствуют точке *b*. Процесс повторяется в моменты t_3, t_4 и т. д.

Полный цикл преобразования, в ходе которого компьютер формирует импульсы запуска, отслеживает логические уровни сигнала окончания преобразования $\overline{EOC}$ и загружает данные из АЦП в память, выполняется под управлением программы, запускаемой перед началом работы. В частности, такая программа сбора данных определяет, сколько точек данных аналогового сигнала будет сохранено в памяти.

На рис. 9.5, *б* АЦП функционирует с максимальной скоростью – новый импульс запуска формируется сразу после того, как компьютер получает новые выходные данные АЦП. Обратите внимание, что время преобразования каждого цикла отличается от предыдущих показателей, так как аналоговый сигнал может изменять свои уровни. Проблема этого метода сохранения информации заключается в том, что для восстановления исходного сигнала необходимо знать точное время, когда цифровые данные должны быть выведены из памяти. Обычно при хранении оцифрованного сигнала отчеты берутся через равные интервалы времени со скоростью как минимум в два раза превышающей максимальную частоту аналогового сигнала. Цифровая система запомнит сигнал в качестве списка значений данных.

Полученный выходной сигнал достаточно точно соответствует исходному аналоговому, поскольку аналоговый сигнал не претерпевал каких-либо резких изменений между точками снятия отсчетов. Если бы сигнал содержал высокочастотные составляющие, то данный АЦП не смог бы уловить быстрые всплески или падения сигнала и восстановленный цифровой результат намного меньше соответствовал бы действительности.

ЛИТЕРАТУРА

1. Янсен, Й. Курс цифровой электроники : в 4-х т. / Й. Янсен. – М. : Мир, 1987.
2. Зельдин, Е. А. Цифровые интегральные микросхемы в информационно-измерительной аппаратуре / Е. А. Зельдин. – Л. : Энергоатомиздат, 1986. – 93 с.
3. Пухальский, Г. И. Проектирование дискретных устройств на интегральных микросхемах : справочник / Г. И. Пухальский, Т. Я. Новосельцева. – М. : Радио и связь, 1990. – 304 с.
4. Цифровые интегральные микросхемы : справочник / М. И. Богданович [и др.]. – Минск : Беларусь, 1991. – 493 с.
5. Шило, В. Л. Популярные цифровые микросхемы : справочник / В. Л. Шило. – М. : Радио и связь, 1989. – 352 с.
6. Потемкин, И. С. Функциональные узлы цифровой автоматики / И. С. Потемкин. – М. : Энергоатомиздат, 1988. – 320 с.
7. Захаров, Н. Г. Синтез цифровых автоматов : учеб. пособие / Н. Г. Захаров, В. Н. Рогов. – Ульяновск : УлГТУ, 2003.
8. Угрюмов, Е. П. Цифровая схемотехника / Е. П. Угрюмов. – СПб. : БХВ-Петербург, 2004. – 528 с.
9. Точи, Р. Цифровые системы. Теория и практика / Р. Точи ; пер. с англ. – 8-е изд. – М. : Изд. дом «Вильямс», 2004. – 1024 с.
10. ОСТ 11340.909–80. Отраслевой стандарт. Микросхемы интегральные полупроводниковые серий 131, К131, 155, К155, 158, К158. Руководство по применению.
11. ОСТ 11340.907–80. Отраслевой стандарт. Микросхемы интегральные серий 564. Руководство по применению.
12. Сапаров, В. Е. Системы стандартов в электросвязи и радиоэлектронике : учеб. пособие для вузов / В. Е. Сапаров, Н. А. Максимов. – М. : Радио и связь, 1985. – 248 с.
13. Разработка и оформление конструкторской документации РЭА : справ. пособие / Э. Т. Романычева [и др.]. – М. : Радио и связь, 1984. – 256 с.
14. Храбров, Е. А. Методические указания и задания по выполнению контрольной работы для студентов специальности 20.05 по курсу «Основы цифровой электроники» заочного факультета / Е. А. Храбров, Н. А. Красовская. – Гомел. политехн. ин-т, 1994. – 34 с.

СОДЕРЖАНИЕ

Учебное электронное издание комбинированного распространения

Учебное издание

Храбров Евгений Александрович
Котова Юлия Евгеньевна

ЦИФРОВАЯ ЭЛЕКТРОНИКА

Учебное пособие

Электронный аналог печатного издания

Редактор *А. В. Власов*
Компьютерная верстка *Е. Б. Ящук*

Подписано в печать 30.12.13.
Формат 60x84/16. Бумага офсетная. Гарнитура «Таймс».
Ризография. Усл. печ. л. 15,81. Уч.-изд. л. 16,68.
Изд. № 33.
http://www.gstu.by

Издатель и полиграфическое исполнение:
Издательский центр
Учреждения образования «Гомельский государственный
технический университет имени П. О. Сухого».
ЛИ № 02330/0549424 от 08.04.2009 г.
246746, г. Гомель, пр. Октября, 48