

Казанский федеральный университет  
Институт физики

**Ю.К. СИТНИКОВ**

**ЭЛЕКТРОННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ МАШИНЫ**

Арифметика

Логика

Элементная база

Основные узлы



**КАЗАНЬ**

**2015**

**УДК 681.31(075.8)**  
**ББК 32.973 я73**  
**С41**

**Рецензенты:**

к.т.н., доцент кафедры систем информационной безопасности  
**КНИТУ-КАИ Л.А. Александрова;**  
к.т.н., доцент кафедры радиофизики **И.С. Нугманов**

**Ситников Ю.К.**

**С41 Электронные вычислительные машины. Арифметика. Логика. Элементная база. Основные узлы /Ю.К. Ситников. – Казань: Изд-во Казан. ун-та, 2015. – 176 с.**

**ISBN 978-5-00019-413-3**

Работа является введением в дисциплину «электронные вычислительные машины и системы». Рассмотрены арифметические и логические основы разработки узлов вычислительных машин. На основе этого материала приводятся сведения об основных узлах, таких как регистры, счетчики, дешифраторы, сумматоры и т.п., реализуемых на интегральных микросхемах малой и средней степени интеграции. Рассматриваются способы и особенности их применения, то есть при изложении материала реализуется схемотехнический подход.

Предназначено для студентов, обучающихся по направлениям схемотехнического профиля, при выполнении выпускных квалификационных работ и магистерских диссертаций. Может быть использовано при изучении дисциплин «Компьютеры и системы», «Цифровые измерительные устройства» и «Цифровая электроника». Работа также может быть полезна инженерно-техническим работникам, занятым разработкой и производством электронной аппаратуры.

**УДК 681.31(075.8)**  
**ББК 32.973 я73**

**ISBN 978-5-00019-413-3**

© Ситников Ю.К., 2015  
© Издательство Казанского университета, 2015

## ПРЕДИСЛОВИЕ

Зародившаяся в середине XX века электронная вычислительная техника развивается очень быстро. Этому способствуют не только потребность в быстрой обработке больших объемов информации, но и развитие науки и техники. При разработке узлов вычислительных машин с самого начала использовались как достижения математики, так и достижения инженерных наук. В каждой новой модели вычислительных машин применялись элементы, не встречавшиеся в предшествующих моделях, а именно вакуумные, магнитные, полупроводниковые.

Благодаря удешевлению электронных вычислительных машин, оснащению их разнообразными программными средствами и удобными устройствами ввода и вывода информации они стали доступными специалистам различных профессий. За всем этим стоит огромный творческий труд разработчиков и технологов, организаторов производства, исследователей.

Знакомство с элементной базой и основными узлами вычислительных машин необходимо для понимания принципов их работы, анализа и синтеза устройств, без этого невозможна разработка новой техники.

Для расширения кругозора читателей автор не ограничивается описанием конкретного типа электронных вычислительных машин – ЭВМ, а рассматривает разнообразные варианты выполнения их узлов. При этом предлагаются несложные контрольные вопросы. Эти вопросы развивают самостоятельный подход к проблеме и на них надо стараться ответить. Не получается с первого раза – нужно вернуться к материалу и разобраться. Именно эта учебная работа позволяет достичь положительного результата, поскольку, если «человек учится сам», он достигает творческого уровня. Если есть возможность выполнения лабораторных работ (лабораторных исследований), то их обязательно необходимо проделать.

Автор будет считать свою работу по написанию книги выполненной, если кто-то из читателей приобретет уровень знаний, необходимый для участия в разработке новых вычислительных средств.

# 1. ВВЕДЕНИЕ В ПРЕДМЕТ

## 1.1. Обработка непрерывных и дискретных сообщений

Вычислительной машиной будем называть комплекс технических средств, объединенных общим управлением, предназначенный для переработки информации и для вычислений. Общее управление подразумевает наличие управляющих программ в памяти вычислительной машины. Таким образом, программы являются обязательным компонентом, находящимся в памяти вычислительной машины при ее работе.

Различают информацию, представленную в *непрерывном* или в *дискретном* виде. Информация передается или отображается в виде сигналов, которые могут быть как непрерывными, так и дискретными. Всякий сигнал есть изменение некоторого физического параметра во времени, то есть функция времени. Функция, называемая *непрерывной*, может принимать любые вещественные значения в диапазоне изменения аргумента (в нашем случае времени –  $t$ ). Таким образом, множество значений непрерывной функции бесконечно. *Дискретная* функция может принимать вещественные значения только при некоторых значениях аргумента. Каким бы малым ни выбирался интервал между соседними значениями аргумента, число значений дискретной функции конечно, если интервал изменения аргумента не бесконечный.

Вид обрабатываемого сигнала определяет принцип действия и структуру вычислительной машины. Машины, обрабатывающие непрерывные сигналы, называют *аналоговыми*, а машины, обрабатывающие дискретные сигналы – цифровыми. Предметом рассмотрения этой работы являются *электронные цифровые вычислительные машины*. Для краткости наряду с этим термином будем употреблять термин компьютер, хотя часто он применяется в более широком смысле.

## 1.2. Понятие о количестве информации

В теории цифровых вычислительных машин важным является понятие *количества информации*. Оно необходимо, например, при оценке параметров запоминающих устройств. Способ измерения информации, лежащий в основе теории информации, предложен К. Шенноном. Речь идет об измерении количества информации, содержащемся в одном случайном объекте относительно другого. Применительно к вычислительной технике проще всего рассмотреть объекты, которые являются случайными величинами, принимающими лишь конечное число значений.

Если  $X$  – случайная величина, принимающая значения  $X_1, X_2, \dots, X_n$  с вероятностями  $p_1, p_2, \dots, p_n$ , а  $Y$  – случайная величина, принимающая значения  $Y_1, Y_2, \dots, Y_m$  с вероятностями  $q_1, q_2, \dots, q_m$ , то информация  $J(X, Y)$ , содержащаяся в  $X$  относительно  $Y$ , записывается как

$$J(X, Y) = \sum_{ij} p_{ij} \log_2(p_{ij} / p_i q_j), \quad (1.1)$$

где  $p_{ij}$  – вероятность совмещения событий  $X_i = Y_j$ .

Величина

$$H(X) = J(X, X) = \sum_i p_i \log_2(1/p_i) \quad (1.2)$$

носит название *энтропии случайной величины*<sup>1</sup>. В вычислительной технике удобно пользоваться приведенным выше определением в следующей формулировке:

если объект может находиться в альтернативных состояниях  $1, 2, \dots, n$  с вероятностями  $p_1, p_2, \dots, p_n$ , соответственно, то количество информации, несущей представление о состоянии, в котором пребывает объект, определяется, как энтропия (1.2).

Если вероятности пребывания во всех состояниях одинаковы и этих состояний  $n$ , то

$$J = \sum p_i \log_2(1/p_i) = np \log_2(1/p) = \log_2 n, \quad (1.3)$$

так как в этом случае  $p = 1/n$ .

Таким образом, если число двоичных знаков (разрядов двоичного числа) равно  $J$ , то из этого следует, что возможно

$$n = 2^J$$

различных значений случайной величины.

Далее для обозначения разрядов двоичного числа вместо  $J$  будут использоваться и другие принятые обозначения.

### 1.3. Численные методы решения задач

Цифровая вычислительная машина может оперировать только с дискретными величинами, которые представляются в виде дискретных электрических сигналов. Это означает, что задача, которую предполагается решить на цифровой машине, не может быть записана в произвольной форме, например с использованием функций непрерывного аргумента и операций анализа, применяемым к этим функциям. Задачу следует привести к специальному виду, позволяющему свести решение задач к вычислениям.

Для этого необходимо воспользоваться численными методами, то есть методами приближенного решения математических задач, сводящегося к выполнению конечного числа элементарных операций над числами. Для конкретной задачи задается определенная последовательность опера-

---

<sup>1</sup> Здесь энтропия рассматривается, как мера неопределенности испытания, которое может иметь разные результаты.

ций над числами – *вычислительный алгоритм*. Язык записи этого алгоритма составляют числа и арифметические действия. Именно эта простота языка позволяет реализовать численные методы на вычислительных машинах.

При применении численных методов задача, подлежащая решению, сформулированная в исходном виде на общепринятом математическом языке (уравнений, функций, операторов), заменяется близкой к ней задачей. При этом фигурирующие в качестве элементарных операций арифметические действия выполняются обычно приближенно. Эти операции дополняются такими вспомогательными операциями, как запись результатов, выборка из таблиц и т. п. Числа задаются ограниченным набором цифр, записанных в некоторой позиционной системе счисления.

Наряду с арифметическими операциями, к элементарным операциям необходимо отнести операции *логические*, определяющие дальнейший ход вычислительного процесса в зависимости от результата предыдущих вычислений.

Таким образом, при применении численных методов числовая прямая заменяется дискретной системой чисел (сеткой чисел), а функция непрерывного аргумента – таблицей ее значений в этой сетке. Операции анализа, применяемые к непрерывным функциям, заменяются алгебраическими операциями над значениями функций в сетке. Вычислительный алгоритм содержит новый параметр – число шагов  $N$ , не присущий исходной задаче. Выбором  $N$  можно, в принципе, добиться любой близости результата выполнения вычислительного алгоритма к решению исходной задачи. Неточная реализация алгоритма, вызванная округлением, не меняет существенно его свойств.

#### **1.4. Возможность автоматического выполнения расчетов**

Информация записывается в общепринятом виде или в виде, отличном от общепринятого. Форма представления, отличающаяся от общепринятой, называется *кодом*. В цифровых вычислительных машинах в виде кодов представляется вся вводимая в нее информация, будь то числа, тексты, рисунки или звуковые сигналы.

Действия, которые выполняются над кодами, введенными в вычислительную машину, сводятся к запоминанию, передаче из одних устройств в другие и выполнению арифметических и логических операций. Все эти операции над кодами выполняются автоматически. Автоматически выполняется также переход к каждой следующей операции.

Процесс обработки информации автоматизирован с помощью программного управления. При этом порядок действий, которые необходимо выполнить для решения задачи, записывается в виде последовательности команд (инструкций). Эта последовательность команд называется *про-*

*граммой*. Команды в вычислительной машине представлены в виде **кодов операций**, присущих конкретной машине. Подготовленная программа (на бумаге или другом носителе информации) записывается с помощью специальных языков – **языков программирования**. Описание порядка действий называется **вычислительным алгоритмом**.

Команда предписывает некоторую операцию из числа операций, реализуемых цифровой вычислительной машиной – ЦВМ, и указывает числа (операнды), участвующие в операции. Реализация программы в ЦВМ сводится к поочередному извлечению команд из памяти вычислительной машины и выполнению их устройствами ЦВМ. **Программа, введенная в машину и хранимая в ее памяти, позволяет осуществить процесс решения задачи автоматически.**

ЦВМ может выполнять некоторый набор операций, достаточный для эффективной реализации некоторого класса алгоритмов. Число алгоритмов может быть бесконечным.

Если возможна смена программ, а система операций, реализуемых ЦВМ, обладает свойством полноты, то ЦВМ является универсальным вычислительным устройством (универсальной ЦВМ) с программируемыми функциями. Путем замены программ изменяются функции, выполняемые ЦВМ.

Наряду с универсальными машинами существуют *специализированные ЦВМ*, обычно предназначенные для работы в системе управления. Такие машины оснащены небольшим набором программ, размещенных в памяти постоянно. При работе в контуре управления ЦВМ периодически выполняет этот набор программ [4]. В цифровых **управляющих** машинах обычно не предусмотрена смена программ.

## 1.5. Алгоритм

Алгоритмом (см. п. 1.3) будем называть такую **конечную последовательность операций** (систему правил), которая позволяет с помощью численных методов получить решение поставленной задачи. Алгоритм – это точное предписание, определяющее процесс преобразования исходных данных в искомый результат. Алгоритм должен обладать такими свойствами, как [4]

- определенность, исключая произвол;
- массовость, то есть возможность обрабатывать любые исходные данные, принадлежащие некоторому множеству;
- результативность, означающая, что процесс при любых допустимых исходных данных приводит к искомому результату;
- дискретность, означающая, что определяемый процесс состоит из отдельных операций;
- простота операций, выполняемых на каждом шаге.

## 1.6. Алфавит и язык

Для записи алгоритмов в виде, пригодном для ввода в вычислительную машину, необходимо использовать специальный язык. Всякий язык характеризуется *алфавитом, синтаксисом и семантикой*. Для построения обозначений дискретного представления информации используется набор символов, называемый алфавитом. Информация, записываемая в виде последовательности символов некоторого алфавита, называется *символьной информацией*. Система правил, определяющая порядок построения выражений из символов алфавита, называется *синтаксисом*.

Для описания синтаксиса используются специальным образом записанные предложения, называемые *металингвистическими формулами*. Существует несколько разновидностей этих формул. Будем пользоваться формулами, состоящими из левой и правой частей, между которыми ставится знак  $=$ , означающий «это есть». В левой части указывается наименование определяемого синтаксического понятия. Справа записывается последовательность символов и выражений, составляющих это понятие. Для выделения наименований синтаксических понятий служат металингвистические скобки  $\langle, \rangle$ . Союз «или» обозначается знаком  $|$ . Например, запись времени суток 16.25 или 2.15:

$\langle \text{время} \rangle := \langle \text{час} \rangle \langle \text{минута} \rangle$

$\langle \text{час} \rangle := \langle \text{цифра} \rangle | \langle \text{цифра} \rangle \langle \text{цифра} \rangle$

$\langle \text{минута} \rangle := \langle \text{цифра} \rangle \langle \text{цифра} \rangle$

$\langle \text{цифра} \rangle := 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9$

Значение времени суток в общепринятой записи состоит из двух частей, разделенных точкой. Запись 25.01 удовлетворяет синтаксическим правилам, но недопустима в данной записи с точки зрения семантики. В данном случае нет взаимосвязи между приведенной записью и правилами счета времени. Семантику составляют правила, устанавливающие связь между корректно составленными выражениями и соответствующими объектами.

Алгоритмы, реализуемые машиной (машинные алгоритмы), представляются на языке машины (*машинном языке*).

Традиционный машинный язык основывается на цифровом алфавите. Оператор машинного языка, как указано в п. 1.4, называется *командой*. Последовательность команд интерпретируется машиной в виде последовательности действий. Запись алгоритмов на машинном языке характеризуется сильной детализацией описания вычислительного процесса. Такая детальная запись является трудоемкой. Для упрощения процесса записи алгоритмов используются *языки программирования*, ориентированные на машинное применение [17]. Эти машинно-ориентированные языки относятся к наиболее простым языкам. Степень близости языка программиро-

вания к машинному языку называют *уровнем* языка программирования. Сам машинный язык принято относить к нулевому уровню.

Следующий уровень языка называется *мнемокодами* (мнемоническими, то есть легко запоминающимися кодами). В этих языках цифровая запись кодов заменяется мнемонической. Мнемоническое описание применяется также для операндов. Операторы мнемокодов имеют структуру команд машинного языка. Мнемокод обычно называют языком ассемблера, поскольку он транслируется (преобразуется) в машинный язык с помощью специальной программы, называемой ассемблером<sup>2</sup>.

К машинно-ориентированным языкам относятся также автокоды. Автокоды по основным свойствам близки к мнемокодам, но в них имеется возможность описать некоторое действие, состоящее из нескольких шагов, и в дальнейшем записывать это действие как одну операцию. Благодаря этому запись алгоритмов становится более компактной и наглядной [4].

Для работы с автокодом, как и в случае мнемокода, необходима специальная программа-переводчик, называемая макроассемблером. Поэтому автокоды иначе называют макроассемблерами по имени этой программы. Использование машинно-ориентированного языка предполагает знание принципов функционирования машины. Этой необходимости можно избежать, так как существуют языки программирования, для которых характерна независимость средств языка от типа машины. Это так называемые *процедурно-ориентированные* языки. К ним относятся такие известные языки программирования, как фортран и паскаль.

Алгоритмический язык может быть также ориентирован на некоторый класс задач. Такие языки называют *проблемно-ориентированными*.

Проблемно- и процедурно-ориентированные языки называют языками *высокого уровня*. Вычислительная машина может выполнять предписания, если они записаны на машинном языке. Если алгоритм записан на другом языке, то он должен быть переведен на язык машины. Для перевода применяют специальные программы, называемые *трансляторами*.

## 1.7. Уровни описания ЦВМ

Вычислительная машина представляет собой сложный электронный комплекс, который содержит большое число компонентов образующих иерархическую систему функциональных узлов. Различные узлы современных вычислительных машин функционируют зачастую на основе различных физических принципов. Многочисленность компонентов и многообразие принципов функционирования не позволяют дать полное описание вычислительной машины на основе единого метода. Поэтому описание ЦВМ приходится производить с различной степенью детальности.

---

<sup>2</sup> Assembly – сборка, объединение.

Описав структуру и порядок функционирования некоторого объекта (узла ЦВМ), можно рассматривать его как элемент более сложного объекта. Вводя более высокий уровень описания, мы уменьшаем степень сложности этого описания.

Наиболее детальным уровнем описания являются *электрические схемы* с перечнями элементов. На этом уровне описания выполняются анализ и разработка отдельных узлов. Описание функционирования вычислительной машины как целого на этом уровне невозможно из-за громоздкости.

Следующим уровнем описания является уровень *логических схем*, на котором описание ЦВМ детализируется до элементарных операций, выполняемых над каждой единицей информации. В качестве языка описания в этом случае используются булевы функции.

Для описания отдельных устройств используется уровень *операционных схем*. При этом процесс функционирования устройств детализируется до элементарных операций над числами, называемых *микрооперациями*. На уровне операционных схем порядок функционирования описывается в форме программ, составленных из микроопераций – *микропрограмм*. Язык микроопераций предназначен для описания цифровых устройств, функционирование которых рассматривается на уровне регистров [9]. Иногда язык микроопераций также называют *языком регистровых передач*. С его помощью можно просто и наглядно описывать регистры, слова, массивы памяти, элементы и части машинных слов и массивов, элементы регистров и отдельные ячейки памяти, операции передачи машинных слов и частей слов.

Микрооперация описывается микрооператором и меткой, то есть идентификатором управляющего сигнала, вызывающего выполнение микрооперации. Например, двухместная операция, в которой участвуют два регистра В и С, описывается следующим образом:

$$G: P_{rA}[k \div k + 1] := P_{rB}[m \div m + 1] * P_{rC}[n \div n + 1].$$

Метка G и микрооператор отделены друг от друга двоеточием. Часть микрооператора, стоящая после знака присваивания ( $:=$ ), называется формулой микрооператора. Формула определяет преобразование. Преобразование здесь обозначено знаком \*. Слева от знака присваивания указывается регистр (или часть регистра), куда передается результат преобразования, выполненного по формуле микрооператора.

Структура ЦВМ как целого объекта описывается на уровне *структурных* схем. При этом перечисляется весь комплекс устройств, составляющих ЦВМ, и описывается порядок взаимодействия устройств в процессе выполнения каждой отдельной операции из системы команд машины.

Наиболее высоким уровнем представления команд машиной является *уровень программ*. Структура программ описывается последовательностью *операторов*, а описываемым объектом являются операционные системы.

### 1.8. Структурная схема цифровой вычислительной машины

Цифровая вычислительная машина содержит в своем составе такие основные устройства, как *процессор, оперативная память и внешние устройства*. К внешним устройствам относятся *устройства ввода (вывода) и внешние запоминающие устройства*. Процессор вычислительной машины состоит из *арифметико-логического устройства и центрального устройства управления*. Арифметико-логическое устройство служит для выполнения *преобразований чисел*, а центральное устройство управления в соответствии с программой обеспечивает *порядок выполнения операций*. Конструктивно процессор может быть выполнен на микросхемах разных технологий и разной степени интеграции. Однако по соображениям надежности, компактности и упрощения сборки предпочтение отдается большим интегральным схемам (БИС) – микропроцессорам.

Оперативная память служит для хранения информации, необходимой для выполнения программы в текущий момент времени. Другими словами, в оперативной памяти хранятся команды выполняемой программы и данные (операнды), необходимые для этой программы.

Внешние запоминающие устройства служат для хранения больших количеств информации, которая не используется процессором в текущий момент времени.

Устройства ввода и вывода позволяют реализовать различные способы ввода и вывода информации. Основным устройством ввода текстовой информации, включая тексты программ, является клавиатура. Наряду с клавиатурой применяются устройства, осуществляющие прямой ввод документов в графической форме – сканеры. При выводе информации она может быть представлена, как в виде твердой копии, так и оперативно. Для получения бумажной копии текста используются электрические печатающие устройства – принтеры, а для получения графических изображений – чертежные автоматы – плоттеры. Оперативное представление выводимой информации осуществляется дисплеями различной конструкции.

Возможности ЦВМ определяются параметрами перечисленных устройств. Основными характеристиками вычислительных машин являются перечень выполняемых операций, скорость их выполнения, количество хранимой информации и возможность смены программ.

Время выполнения операций зависит от их сложности. Такие операции, как сложение выполняются за малое время. Продолжительность выполнения операций умножения и деления, как правило, во много раз

больше, чем время, необходимое для выполнения сложения или вычитания. Производительность вычислительной машины оценивается количеством операций, выполняемых за единицу времени. Эта величина часто указывается отдельно для коротких и для длинных операций.

Связь между устройствами цифровой вычислительной машины может быть осуществлена несколькими способами.

Связь устройств может быть выполнена по принципу *каждый с каждым*. При такой схеме соединений каждая пара устройств-абонентов может обмениваться кодами независимо от других устройств. Этот способ является наиболее производительным, а система в целом наиболее надежной. В то же время такая система является наиболее сложной и дорогой.

Для упрощения системы связей без существенного снижения производительности можно объединить в группы устройства, имеющие близкую производительность. Для каждой такой группы выделяется система обмена информацией своего типа. В группе медленных устройств для связи применяются более простые, но медленные устройства. Для более быстрых устройств выделяются более производительные средства обмена информацией. В этом случае осуществляется управление распределением информации по группам устройств и между устройствами медленной группы. При управлении обменом информацией с внешними устройствами система связей вместе с управлением называется каналом.

Для наиболее простых и недорогих электронных вычислительных машин, например персональных ЭВМ, применяется единая система связей для всех устройств. Такая система называется *общей шиной*.

Рассмотрим подробнее систему с общей шиной (рис. 1.1). Показанная на рис. 1.1 схема существенно упрощена. На ней представлены основные устройства и шина обмена информацией.



Рис. 1.1. Система объединения устройств посредством общей шины

Линии для передачи сигналов управления на рис 1.1 не показаны.

## Вопросы для самопроверки

1. Что называется кодом?
2. Перечислите свойства алгоритма.
3. Опишите с помощью металингвистических формул способ записи даты рождения.
4. Нарисуйте функциональную схему цифровой вычислительной машины.
5. Из чего состоит процессор?
6. Для чего служит арифметико-логическое устройство?
7. Какие функции выполняет центральное устройство управления?
8. Для чего используется оперативная память?
9. Каковы функции канала?
10. Каково назначение внешних запоминающих устройств?
11. В чем заключается основная особенность специализированной ЦВМ в сравнении с универсальными?
12. Что называется командой?
13. Что называется операндом?
14. Что называется адресом?
15. Для чего в современных ЦВМ необходима программа транслятор?
16. Сколько можно записать различных двоичных значений случайной величины при использовании чисел, имеющих  $N$  разрядов?
17. Что называется алгоритмом?
18. Какие вы знаете современные языки программирования?
19. Какие уровни описания ЭВМ вам известны?
20. Почему нельзя описать устройство ЭВИ в целом с помощью принципиальной электрической схемы?
21. Что указывается на логических схемах?
22. Что указывается на принципиальных электрических схемах?
23. Какая схема является более детальной: принципиальная или логическая?
24. Нарисуйте структурную схему вычислительной системы с «общей шиной».
25. Перечислите достоинства и недостатки вычислительной системы со связями типа «каждый с каждым».

## 2. АРИФМЕТИЧЕСКИЕ ОСНОВЫ ЦВМ

### 2.1. Представление информации в вычислительных машинах

В вычислительных машинах информация записывается на машинном языке. Машинный язык основывается на цифровом алфавите. Наименьшей единицей информации, записываемой в вычислительной машине, является один *разряд* числа в системе счисления с основанием два, называемый *битом*. В каждом бите может быть записан один из двух символов: 0 или 1. Биты объединяются в группы.

Группа разрядов в оперативной памяти вычислительной машины, считываемых (записываемых) одновременно, называется *словом*. Обычно понятие машинного слова относится к коду определенной для данной вычислительной машины длины или, другими словами, содержащему фиксированное число разрядов. Группа битов (разрядов), имеющих определенное значение, называется *полем*. При операциях с буквенными символами каждому символу также сопоставляется цифровой двоичный код. В большинстве случаев коды символов имеют восемь разрядов. Восемьразрядный двоичный код называется *байтом*.

Совокупность групп полей, описывающих некоторое множество объектов (последовательность полей, байтов или слов, объединенных по некоторым признакам), называется *массивом*, или *файлом*<sup>3</sup>.

Машинные коды чисел и команд – слова – могут состоять из одного или нескольких байтов. Распространены вычислительные машины со словами, содержащими 8, 16 или 32 разряда. Наряду с терминами «слово» и «байт» при разделении и объединении слов применяют термины «полуслово», содержащее 16 разрядов, и «двойное слово», имеющее 64 разряда.

Натуральным единицам информации, которыми мы пользуемся в общепринятой записи: разряду, символу, числу, массиву чисел, в цифровых вычислительных машинах соответствуют разряд, слог (байт), слово (поле) и массив (файл).

### 2.2. Системы счисления

Счисление (нумерация) – это совокупность приемов наименования и обозначения чисел [19]. Использование большого количества символов для обозначения чисел неудобно. Применяемые в настоящее время системы счисления используют ограниченное количество символов для обозначения разных цифр.

В соответствии со сказанным *системой счисления* называется совокупность приемов и правил для записи чисел ограниченным количеством цифровых знаков.

---

<sup>3</sup> Понятие файла будет дополнительно уточнено при изучении запоминающих устройств.

Представление чисел в вычислительных машинах отличается от их представления в широко используемой десятичной системе счисления. Перед изучением машинных форм представления чисел рассмотрим свойства систем счисления.

Любая, предназначенная для практического применения система счисления, должна удовлетворять следующим требованиям:

- обеспечивать возможность представления любого числа в рассматриваемой системе;
- обеспечивать единственность представления каждого числа;
- обеспечивать простоту выполнения операций над числами.

Системы счисления делятся на *позиционные* и *непозиционные*.

В непозиционных системах счисления значение символа не зависит от его положения в записи числа. Все непозиционные системы принято называть *символическими*.

В позиционной системе счисления значение цифры определяется ее положением в числе. Один и тот же символ, находясь на разных позициях, определяет различные значения.

В общем случае правило построения числа может быть записано следующим образом:

$$A(q) = a_1q_1 + a_2q_2 + \dots a_nq_n, \quad (2.1)$$

Где  $q_i$  называется весом разряда, а символами  $a_i$  обозначены цифры.

Если в этом выражении положить  $q_i = q^i$ , то получим равенство, которому должна удовлетворять позиционная система счисления:

$$q_i = qq_{i-1}, \quad (2.2)$$

где  $q$  – *основание* системы счисления.

Основание системы счисления определяется количеством разных символов, необходимых для изображения числа в данной системе.

Произвольное число  $A(q)$  в системе счисления с основанием  $q$  записывается как

$$a_nq^n + a_{n-1}q^{n-1} + \dots a_1q^1 + a_0q^0 + a_{-1}q^{-1} + \dots a_{-m}q^{-m}, \quad (2.3)$$

$$\text{или } A(q) = \sum_{i=-m}^{i=n} a_i q^i, \quad (2.4)$$

где  $n$  и  $m$ , соответственно, количество целых и дробных значений разрядов числа. Для привычной нам десятичной системы

$$A(10) = \sum_{i=-m}^{i=n} a_i 10^i. \quad (2.5)$$

Обычно используется сокращенная запись числа  $A(q)$

$$a_n a_{n-1} \dots a_1 a_0 a_{-1} \dots a_{-m}. \quad (2.6)$$

Например,  $2 \times 100 + 3 \times 10 + 6 \times 1 + 1 \times 0,1 + 5 \times 0,01$  записывается как 236,15.

В системе счисления с основанием 2 используются цифры 0 и 1. Эта система называется *двоичной*.

Для любой позиционной системы счисления основание системы изображается с помощью цифр этой системы сочетанием символов 1 и 0. Поскольку в вычислительных машинах для представления чисел используется конечное число разрядов, то необходимо учитывать, какое максимальное число и минимальное число можно записать в конкретной разрядной сетке. Попытки выйти за этот диапазон приводят к ошибкам результата, за счет отбрасывания крайних значений. Поэтому машинная арифметика является *арифметикой конечных чисел*.

Для арифметики конечных чисел важным является понятие *диапазона представления чисел*. Диапазон представления чисел – это интервал числовой оси, заключенный между минимальным и максимальным числами, записанными при данном числе разрядов

$$A(q)_{\min} \leq \text{диапазон} \leq A(q)_{\max}, \quad (2.7)$$
$$A(q)_{\min} = -(q^n - 1) = 1 - q^n, \quad A(q)_{\max} = q^n - 1.$$

Вес разряда числа в позиционной системе счисления в соответствии с (2.2)

$$q_i = \frac{q^i}{q^0} = q^i, \quad (2.8)$$

где  $i$  – номер разряда, отсчитываемого от младшего разряда целой части, принимаемого за нулевой.

Поскольку количество единиц в разряде не может быть равным или большим  $q$ , то возникает необходимость передавать единицы в соседний разряд. Эта операция при сложении называется *переносом*, а при вычитании – *заемом*.

При проектировании вычислительных машин очень важен правильный выбор системы счисления. От этого выбора зависят скорость вычислений, необходимый объем памяти и сложность алгоритмов.

Для записи одного и того же числа в различных системах счисления необходимо различное число разрядов. При выборе системы счисления для ЭВМ учитывают, что

- основание системы счисления определяет количество устойчивых состояний, которое должен иметь элемент ЭВМ (устройство, которое используется для отображения разрядов числа);
- длина записи числа существенно зависит от основания системы счисления;
- система счисления должна обеспечивать простые алгоритмы выполнения операций.

Наиболее проста реализация арифметических действий в двоичной системе счисления. Правила выполнения арифметических действий видны

из приведенного примера, в котором прямоугольником отмечены единица переноса в старший разряд и единица заема из старшего разряда:

0+0=0	0-0=0	0x0=0
0+1=1	1-0=1	0x1=0
1+0=1	1-1=0	1x0=0
1+1=0 <span style="border: 1px solid black; padding: 0 2px;">1</span>	0-1=1 <span style="border: 1px solid black; padding: 0 2px;">1</span>	1x1=1

Умножение многоразрядных чисел осуществляется путем образования частных произведений и их последующего суммирования. В двоичной системе счисления умножение сводится к операциям сдвига и сложения. Например,

$$\begin{array}{r}
 11100101,01 \quad = 229,25D \\
 \underline{101,1} \quad = 5,5D \\
 11100101 \ 01 \\
 + 1110010101 \\
 \hline
 1110010101 \\
 10011101100,111 \quad = 1260,875
 \end{array}$$

В этом примере справа приведена запись сомножителей и произведения в десятичной системе счисления.

Деление сводится к операциям вычитания и сдвига. В качестве примера выполним в двоичной системе деление целого числа  $405D$  на  $19D^4$

$$\begin{array}{r}
 405D = 110010101 \overline{)10011} \quad = 19D \quad \text{делитель} \\
 \underline{-10011} \quad 10101 \quad = 21B \quad \text{частное} \\
 11001 \\
 \underline{-10011} \\
 11001 \\
 \underline{-10011} \\
 110 \quad = 6D \quad \text{остаток}
 \end{array}$$

Любая позиционная система счисления дает более компактную запись числа в сравнении с двоичной системой. Правила арифметики во всех позиционных системах счисления одинаковы, поэтому дальнейшие примеры не требуют пояснений. Запись чисел в различных системах счисления приведена в табл. 2.1. Рассмотрим приведенный выше пример для троичной системы, в которой применяются цифры 0, 1 и 2. В этом примере соответствие чисел приблизительное, поскольку имеется дробная часть (см. п. 2.3).

$$\begin{array}{r}
 229,25D * 22111,0202 \\
 \times 12,1111 = 5,5D \\
 \hline
 221110202 \\
 \hline
 1212221111 \\
 221110202 \\
 \hline
 1201122,1022122 = 1259,195D
 \end{array}$$

<sup>4</sup> Буквой *D* отмечено десятичное представление числа.



В текстах программ, инструкциях к программам часто применяют сокращенные однобуквенные обозначения систем счисления, как в рассматриваемом примере. Для двоичных чисел применяют обозначение –  $B$ , для восьмеричных –  $O$ , десятичных –  $D$ , а для шестнадцатеричных –  $H$ .

Системы счисления могут различаться по количеству символов, применяемых в разных разрядах. Если количество допустимых символов для всех разрядов одинаково, то такая система называется **однородной**.

Если в разных разрядах применяется разное количество цифр, то система называется **смешанной**.

Для обозначения каждой цифры может применяться отдельный символ или набор из нескольких символов. Если для каждой цифры имеется свой значок, то система называется **непосредственной**. Для непосредственной системы количество применяемых знаков равно количеству разных цифр, используемых в системе.

Если количество разных знаков меньше, чем количество цифр, применяемых в данной системе, то каждая цифра кодируется комбинацией нескольких символов. Например, цифры десятичной системы можно кодировать символами (цифрами) применяемыми в двоичной системе. Для изображения цифр десятичной системы часто применяется система, называемая **двоично-десятичной**. В этой системе каждая десятичная цифра кодируется четырьмя двоичными цифрами. Так двоично-десятичное представление числа  $125D$  состоит из трех четырехразрядных групп 0001 0010 0101.

Недостатком позиционных систем счисления является невозможность выполнения арифметических операций, как поразрядных. Поразрядной называют такую операцию, результат которой в любом разряде не зависит от результата выполнения этой операции в других разрядах. В позиционных системах счисления результат в одном разряде влияет на другие разряды через перенос и заем.

Существуют системы счисления, в которых арифметические операции выполняются поразрядно. Примером такой системы является **система остаточных классов** – СОК. В этой системе каждое число представляется остатками от его деления на взаимно простые числа, являющиеся основаниями системы счисления. Запись в остатках является символическим способом представления чисел.

Если ЦВМ должна оперировать с  $N$  целыми числами (от нуля до  $N-1$ ), то для записи этих чисел в системе остаточных классов следует выбрать несколько взаимно простых чисел  $X_1, X_2, \dots, X_n$  таких, чтобы их произведение было не меньше  $N$ . Число, которое необходимо записать в СОК, следует разделить на  $X_1$ , на  $X_2, \dots, X_n$ . Заданное число представляется в виде остатков от деления. Получается смешанная система с различными основаниями для различных разрядов. Например, если  $N = 15$ , в качестве

делителей могут быть выбраны наименьшие простые числа 2, 3 и 5 ( $2 \times 3 \times 5 > 15$ ).

Таблица 2.2

Запись чисел в системе остаточных классов

X =	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
X <sub>сок</sub>	000	111	220	301	410	100	211	320	401	010	121	200	311	420	001

Система остаточных классов относится к системам символическим, смешанным и кодированным. Сложение в СОК сводится к независимому суммированию разрядов чисел. Для примера переведем в СОК и просуммируем десятичные цифры 5 и 9 ( $5 = 021_{\text{сок}}$ ,  $9 = 401_{\text{сок}}$ ). Так как вес правого разряда равен 2, то сумма равна  $420_{\text{сок}} = 14D$ .

Рассмотрим примеры умножения в системе остаточных классов.

$$\begin{array}{r}
 301 \\
 * 220 \\
 \hline
 100
 \end{array}
 \quad
 \begin{array}{r}
 301 \\
 * 301 \\
 \hline
 401
 \end{array}
 \quad
 \begin{array}{r}
 021 \\
 * 111 \\
 \hline
 021
 \end{array}
 \quad
 \begin{array}{r}
 301 \\
 * 410 \\
 \hline
 200
 \end{array}$$

### 2.3. Перевод чисел из одной системы счисления в другую

Исходные данные могут быть представлены либо в виде двоичных чисел, либо как десятичные или шестнадцатеричные числа, либо как любой произвольный текст. Данные в ЦВМ, работающей в двоичной системе счисления, должны быть закодированы числами двоичной системы. Если построить ЦВМ, работа которой основана на другой системе счисления, например троичной, то данные должны быть закодированы числами этой системы. Наиболее часто исходные данные и результаты записывают в десятичной системе счисления. При вводе данных в вычислительную машину и при выводе данных делается перевод из одной системы счисления в другую.

*Запись числа в различных позиционных системах счисления отличается значением основания  $q$  и величинами коэффициентов  $a$  (см. п. 2.2).*

Правила перевода зависят от того, какая арифметика используется для перевода: арифметика исходной системы счисления или арифметика той системы, в которую преобразуется число.

При ручном переводе чисел определяют, какая максимальная степень нового основания входит в преобразуемое число (не превышает преобразуемое число). Затем проверяется, сколько раз входит каждая степень нового основания меньшая максимальной в оставшуюся после вычета старшей степени часть числа. Например, при переводе из десятичной системы счисления в троичную систему число  $89D$  представляется как  $10022$

$$89D = 1*3^4 + 0*3^3 + 0*3^2 + 2*3^1 + 2*3^0.$$

Разработано несколько способов для преобразования чисел в вычислительных машинах. Рассмотрим преобразования, выполняемые по правилам арифметики исходной системы счисления. Основание исходной системы обозначим  $m$ , а основание искомой –  $n$ . Пусть имеется целое число.

Искомая запись целого числа (в системе с основанием  $n$ )

$$A(n) = \beta_k n^k + \beta_{k-1} n^{k-1} + \dots + \beta_0 n^0. \quad (2.9)$$

Поделим число  $A(n)$  на  $n$

$$\frac{A(n)}{n} = \beta_k n^{k-1} + \beta_{k-1} n^{k-2} + \dots + \frac{\beta_0}{n}. \quad (2.10)$$

В результате деления получили частное и остаток. Обозначим частное  $A_1$  и запишем

$$\frac{A(n)}{n} = A_1 + \frac{\beta_0}{n}, \quad (2.11)$$

$$A(n) = A_1 n + \beta_0.$$

Аналогичным образом можно записать частное

$$A_1 = A_2 n + \beta_1. \quad (2.12)$$

Так как мы делим число на основание новой системы, то каждый получаемый остаток не содержит цифр, не имеющих в новой системе счисления. Иначе говоря, остаток уже записан в новой системе счисления. Каждый остаток есть цифра соответствующего разряда представления числа в системе с основанием  $n$ . Первый остаток дает самый младший разряд числа в искомой записи.

Таким образом, правило перевода целого числа из системы с основанием  $m$  в систему с основанием  $n$  заключается в последовательном делении числа и получаемых частных на основание новой системы  $n$ , которое для того, чтобы осуществить деление, представлено в исходной системе (записано цифрами исходной системы и по правилам исходной системы счисления). Деление осуществляется до тех пор, пока частное не станет меньше  $n$ . Старшей цифрой нового числа служит последнее частное, следующей цифрой – последний остаток. Например,

$$\begin{array}{r}
 10D \overline{) 3} \\
 \underline{9} \quad \underline{3} \overline{) 3} \\
 1 \quad \underline{3} \quad 1 \\
 0
 \end{array}
 \qquad
 \begin{array}{r}
 95D \overline{) 2} \\
 \underline{94} \quad \underline{47} \overline{) 2} \\
 1 \quad \underline{46} \quad \underline{23} \overline{) 2} \\
 \quad \quad 1 \quad \underline{22} \quad \underline{11} \overline{) 2} \\
 \quad \quad \quad 1 \quad \underline{10} \quad \underline{5} \overline{) 2} \\
 \quad \quad \quad \quad 1 \quad \underline{4} \quad \underline{2} \overline{) 2} \\
 \quad \quad \quad \quad \quad 1 \quad \underline{2} \quad 1 \\
 \quad \quad \quad \quad \quad \quad 0
 \end{array}
 \qquad
 95D = 1011111B$$

$$\begin{array}{r}
 10D = 101 \text{ (троичная)} \\
 1101001B \overline{) 1010} \\
 \underline{1010} \quad \underline{1010} \overline{) 1010} \\
 001100 \quad \underline{1010} \quad 1 \\
 \underline{1010} \quad \underline{0} \\
 00101
 \end{array}$$

$$1101001B = 105D$$

Для дробных чисел применяется умножение на основание новой системы счисления. Для перевода правильной дроби из системы с основанием  $m$  в систему с основанием  $n$  нужно умножить исходную дробь и дробные части получающихся произведений на основание  $n$ , представленное в исходной  $m$ -системе. Целые части получающихся произведений дают последовательность цифр в представлении дроби в новой системе. Обычно перевод осуществляется приближенно.

Так как в обоих случаях перевод сопровождается округлением, то результат получается меньше исходного числа. Если переводится смешанное число, то переводят отдельно целую и отдельно дробную части числа, каждую по своим правилам. При разработке ЭВМ способ перевода выбирается из соображений быстроты перевода и экономии оборудования.

$  \begin{array}{r}  0,1496D \sim 0,001001B \\  \times \quad 2 \\  \hline  0,2992 \\  \times \quad 2 \\  \hline  0,5984 \\  \times \quad 2 \\  \hline  1,1968 \\  \times \quad 2 \\  \hline  0,3963 \\  \times \quad 2 \\  \hline  0,7872 \\  \times \quad 2 \\  \hline  1,5744  \end{array}  $	$  \begin{array}{r}  0,001001B = 0,1406D \\  \times \quad 1010 \\  \hline  10010 \\  +1001 \\  \hline  1,011010 \\  \times \quad 1010 \\  \hline  110100 \\  +11010 \\  \hline  100,000100 \\  \times \quad 1010 \\  \hline  1000 \\  + \quad 100 \\  \hline  0,101000 \\  \times \quad 1010 \\  \hline  1010000 \\  +101000 \\  \hline  110,010000  \end{array}  $	<p>0,</p> <p>1</p> <p>4</p> <p>0</p> <p>6</p>
---	---	---

При вводе чисел в ЭВМ можно реализовать перевод из одной системы счисления в другую, используя промежуточные преобразования. Исходные данные могут быть представлены в десятичной, двоичной, символьной или иной форме. Эти данные вводятся либо по линиям связи, посредством клавиатуры, либо с помощью промежуточного носителя информации (магнитного диска, компакт диска, полупроводниковой памяти и т.п.). При этом они кодируются удобным для ввода в ЭВМ образом. Вводимые в ЭВМ данные кодируются цифрами двоичной системы, что позволяет ввести их в память ЭВМ и затем преобразовать программным путем. Например, десятичные числа при вводе в ЭВМ можно преобразовать к промежуточному двоично-десятичному виду (п. 2.2 и п. 2.5). При этом каждой цифре десятичной системы сопоставляется четырехразрядный двоичный код. В целом десятичному числу сопоставляется набор четырехразрядных кодов (*тетрады*). Так число  $125D = 1111101B$  запишется в

двоично-десятичном коде как 0001 0010 0101 (подробнее двоично-десятичный код рассматривается далее в п. 2.5).

Каждая тетрада имеет свой десятичный вес, соответствующий правилам представления числа в десятичной системе счисления. Каждую тетраду при преобразовании следует умножить на ее вес, записанный в двоичной системе счисления, и эти частные произведения сложить по правилам двоичной арифметики. В примере с числом 125 следует проделать следующие вычисления:

$$\begin{array}{r}
 100 = 1100100 \text{ В} \quad 10 = 1010 \text{ В} \quad 0001 \quad 1100100 \\
 * 0001 \quad * 0010 * 0101 + 10100 \\
 \hline
 \text{-----} \quad \text{-----} \quad \text{-----} \quad 101 \\
 1100100 \quad 10100 \quad 101 \quad \text{-----} \\
 1111101
 \end{array}$$

При таком способе преобразования вес тетрад (1100100, 1010, 0001) должен вводиться в ЭВМ вместе с программой перевода или постоянно хранится в ее памяти. Кроме программного способа применяется также табличный способ перевода, при котором в ЭВМ хранятся эквиваленты цифр системы и эквиваленты положительных и отрицательных степеней основания. При переводе подставляются эквиваленты и выполняются все необходимые умножения и сложения.

Наиболее просто перевести число в двоичную систему счисления из восьмеричной и шестнадцатеричной систем, так как основания этих систем равны степеням двойки. Каждая цифра восьмеричного кода может быть записана с помощью трех двоичных разрядов. Например, восьмеричное число  $745 = 111 \ 100 \ 101 \text{ В}$ . Таким же образом осуществляется преобразование шестнадцатеричного кода. Например,  $A4F1H = 1010 \ 0100 \ 1111 \ 0001 \text{ В}$ . Как видно из примера, каждой шестнадцатеричной цифре соответствуют четыре двоичных разряда. Поскольку основания этих систем 8 и 16 равны степеням двойки двоично-восьмеричная и двоично-шестнадцатеричная записи числа совпадают с представлением в двоичной системе. (Для сравнения переведите указанным ранее способом в двоичную систему счисления восьмеричное число 745 и шестнадцатеричное число A4F1).

Несложен и обратный переход от двоичной системы счисления к шестнадцатеричной (или восьмеричной). Двоичное число разделяется справа и слева от запятой на тетрады (или триады). Если крайняя правая и крайняя левая тетрады будут неполными, двоичное число справа и слева необходимо дополнить нулями. Каждая тетрада используется при этом для определения соответствующей шестнадцатеричной цифры. Например,

$$\begin{array}{r}
 1001101101101, 1010101101 \\
 0001 \ 0011 \ 0110 \ 1101, 1010 \ 1011 \ 0100 \\
 1 \ 3 \ 6 \ D, \ A \ B \ 4
 \end{array}$$

Перевод чисел в шестнадцатеричную систему счисления и обратный перевод осуществляются сравнительно просто и быстро в силу компактности шестнадцатеричной системы. Например,

$$\begin{array}{r}
 42225 \overline{)16} \\
 \underline{-32} \phantom{00} \\
 102 \phantom{00} \\
 \underline{-96} \phantom{00} \\
 62 \phantom{00} \\
 \underline{-48} \phantom{00} \\
 145 \phantom{00} \\
 \underline{-144} \phantom{00} \\
 1 \\
 \hline
 \phantom{00} A4F1H
 \end{array}
 \qquad
 \begin{array}{r}
 0111 \phantom{00} \phantom{00} \phantom{00} \phantom{00} \phantom{00} \phantom{00} \\
 \times 10000 \\
 \hline
 1110000 \\
 + \phantom{00} 1011 \\
 \hline
 1111011 = 123D
 \end{array}
 \qquad
 \begin{array}{r}
 1011 = 7BH \\
 \times 1 \\
 \hline
 1011 = BH = 11D
 \end{array}$$

В примере, расположенном слева, осуществляется последовательное деление числа 42225, записанного в десятичной системе счисления, на 16. Получается результат в шестнадцатеричной записи равен A4F1.

Во втором примере шестнадцатеричное число 1111011 переводится в двоичную систему. Число записано в виде двух тетрад 0111 1011. Поскольку, как указывалось выше, вес разрядов в двоичной и в шестнадцатеричной системах является степенями цифры 2, имеем сразу без перевода двоичную запись. Проверим: умножим цифры тетрад на их шестнадцатеричный вес, записанный в двоичной форме. Младшая тетрада 1011 имеет вес  $16^0 = 1$ . Выполним умножение на 1. Старшая тетрада имеет вес  $16^1 = 16$  или в двоичной форме 1 0000. Выполним умножение второй тетрады, то есть 0111, на ее вес. Получаем 1110000. Число в целом определяется как сумма, равная 1111011. Видим, что двоичная запись и исходная шестнадцатеричная запись одинаковы.

Если операции над десятичными числами необходимо выполнить без затрат времени на перевод в двоичную систему, то их преобразуют к двоично-десятичному виду. Арифметические действия выполняют непосредственно над этими двоично-десятичными кодами. (Действия над двоично-десятичными кодами можно выполнить в двоичном сумматоре, используя программную коррекцию или в аппаратно реализованном сумматоре для двоично-десятичных кодов).

Преобразование из двоичной системы в десятичную можно осуществить, используя двоично-десятичную систему как промежуточную. В соответствии со сказанным ранее применяется деление (или умножение) преобразуемого числа на основание десятичной системы, записанное цифрами исходной двоичной системы. Полученные остатки и частное (или, соответственно, целые части) дополняются слева нулями до четырех разрядов, образуя тетрады двоично-десятичного кода. Например,

$$\begin{array}{r}
 1111101 \overline{)1010} \\
 \underline{-1010} \quad 1100 \overline{)1010} \\
 1011 \quad \underline{-1010} \quad \rightarrow 1 \\
 \underline{-1010} \quad 00 \rightarrow 10 \quad \leftarrow 000 \\
 0 \rightarrow 101
 \end{array}
 \qquad
 \begin{array}{r}
 0,001 \\
 \times \underline{1010} \\
 000 \rightarrow 1,010 \\
 \quad \times \underline{1010} \\
 00 \rightarrow 10,100 \\
 \quad \quad \times \underline{1010} \\
 0 \rightarrow 101,000
 \end{array}$$

В рассмотренном примере после формирования тетрад получаются числа  $0001\ 0010\ 0101 = 125D$  и  $0,0001\ 0010\ 0101 = 0,125D$ .

## 2.4. Коды для представления чисел в ЭВМ

Представление числовой информации в вычислительной машине, как правило, невозможно без погрешностей. Эти погрешности необходимо знать и учитывать. Величина ошибки зависит от формы представления чисел и длины разрядной сетки, принятой для данной ЭВМ. Кроме того, форма представления числа и длина разрядной сетки определяют диапазон значений чисел, обрабатываемых ЭВМ. Поскольку количество разрядов в числах (длина разрядной сетки) в вычислительных машинах всегда фиксировано, то это приводит к некоторым дополнительным действиям при вычислениях. Иначе говоря, «бумажная арифметика» и машинная арифметика имеют некоторые различия, связанные с конечным числом разрядов машинных кодов. В связи с этим, как сказано ранее, машинную арифметику *называют арифметикой конечных чисел*.

При выполнении арифметических операций в ЭВМ необходимо учитывать знаки чисел. Чтобы избежать операции вычитания и упростить действия с отрицательными числами и, соответственно, упростить арифметико-логическое устройство, в ЭВМ применяются специальные коды. Если знак числа и мантиссу, представленную в виде правильной дроби, рассматривать как единое число, то все положительные числа имеют вид  $0\ X_1\ X_2 \dots X_{n-1}$  и лежат в интервале  $0 \leq X < 1$ , а отрицательные числа имеют вид  $1\ X_1\ X_2 \dots X_{n-1}$  и лежат в диапазоне  $1 \leq X < 2$ . Таким образом, отрицательному числу, находящемуся в интервале  $-1 < X \leq 0$ , сопоставлено положительное число  $[X]$ , находящееся в интервале  $1 \leq [X] \leq 1 + X$ , если знак минус обозначен единицей. Путем записи единицы в знаковый разряд отрицательное число сдвигается по числовой оси в область положительных чисел.

Рассмотренный случай относится к правильным дробям. Читатель может сам показать, что для целых чисел диапазон будет соответственно

$$0 \leq X < 2^n, 2^n \leq X < 2^{n+1} - 1.$$

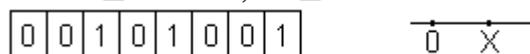


Рис. 2.1. Размещение числа в разрядной сетке и на числовой оси

Для кодирования отрицательных чисел и выполнения операций над ними применяют **дополнительный** и **обратный** коды. Положительные числа во всех кодах одинаковы. Исходное число,  $0, X_1, X_2, \dots, X_n$ , введенное в ЭВМ без преобразований, то есть в коде, называемом прямым, получается по следующей формуле:

$$[X]_{пр} = \begin{cases} X, & \text{если } X \geq 0 \\ 1-X, & \text{если } X < 0. \end{cases}$$

Можно также записать формулу для прямого кода, используя вес знакового разряда. Обозначим вес знакового разряда  $S$ . Для дробей  $S = 1$ , а для целых чисел  $S = 2^n$ . Тогда

$$[X]_{пр} = \begin{cases} X, & \text{если } X \geq 0 \\ S+[X], & \text{если } X < 0 \end{cases}$$

Дополнительный код отрицательного числа получается, если в знаковый разряд поставить единицу и во всех разрядах мантииссы сменить цифры двоичного кода, к младшему разряду мантииссы добавить единицу и выполнить переносы, включая перенос в знаковый разряд. Другими словами дополнительный код является дополнением мантииссы до единицы (в случае рассматриваемых здесь дробных мантиисс).

Для

$$\begin{aligned} X = -0, X_1 X_2 \dots X_n, \quad [X]_{доп} &= 1, \bar{X}_1 \bar{X}_2 \dots \bar{X}_n + 0,00\dots 1 \\ [X]_{доп} - X &= 1, \bar{X}_1 \bar{X}_2 \dots \bar{X}_n + 0,00\dots 1 + 0, X_1 X_2 \dots X_n = \\ &= 1,11\dots 1 + 0,00\dots 1 = 10,0 \end{aligned} \quad (2.13)$$

то есть  $[X]_{доп} = 10 + X$ ,  $[0]_{доп} = 0,00\dots 0$ ;  $[-0]_{доп} = 10$ .

В последнем выражении в знаковом разряде 0, а 1 левее знакового разряда теряется. Иначе говоря,  $[0]_{доп} = [-0]_{доп} = 0$ . Таким образом, ноль в дополнительном коде имеет только одно представление.

Следовательно,

$$[X]_{доп} = \begin{cases} X, & \text{если } X \geq 0 \\ 10+X, & \text{если } X < 0. \end{cases}$$

где 10 является двоичной записью числа 2. Представление чисел в дополнительном коде позволяет при суммировании оперировать со знаковым разрядом как с разрядом целой части числа.

Обратный код отрицательного двоичного числа является единичным дополнением (поразрядным дополнением). Обратный код отрицательного двоичного числа получается, если в знаковый разряд поставить единицу и во всех разрядах мантииссы поменять цифры.

Для  $X = -0, X_1 X_2 \dots X_n$ ;

$$\begin{aligned} [X]_{обр} &= 1, \bar{X}_1 \bar{X}_2 \dots \bar{X}_n; \\ [X]_{обр} - X &= 1, X_1 X_2 \dots X_n + (0, X_1 X_2 \dots X_n) = 1,11\dots 1; \\ [X]_{обр} &= 1,11\dots 1 + X \\ [X]_{обр} &= \begin{cases} X, & \text{если } X \geq 0 \\ 10-1*10^{-n}+X, & \text{если } X < 0 \end{cases} \end{aligned} \quad (2.14)$$

Последнее следует из того, что  $1,11\dots1 = 10^{-1} \times 10^{-n}$ .

Таким образом, обратный код является дополнением до значения ближайшего старшего разряда, уменьшенного на единицу младшего разряда.

В процессе вычислений могут возникнуть «положительный» и «отрицательный» нули.

$$+ 0,00\dots0 \text{ и } -0,00\dots0;$$

Представление «положительного» нуля одинаково для прямого, обратного и дополнительного кодов:

$$(+ 0) = 0,00\dots0;$$

Отрицательный ноль отобразится следующим образом:

в прямом коде  $(-0)_{\text{пр}} = 0,00\dots0;$

в обратном коде  $(-0)_{\text{обр}} = 1,11\dots1;$

в дополнительном коде  $(-0)_{\text{доп}} = 1,11\dots1 + 2^{-n} = 0,000\dots0,$   
поскольку перенос из знакового разряда теряется.

Из приведенных выше выражений следует, что в обратном коде ноль можно изобразить двояко.

Рассмотрение кодов, сделанное выше, выполнено для правильных дробей. Несложно убедиться, что коды для целых чисел получаются по тем же правилам. При этом дополнение для обратного кода делается до  $2^n - 1$ , а для дополнительного кода – до  $2^n$ , где  $n$  количество разрядов числа. Коды целых чисел представлены в табл. 2.3. В этой таблице мантисса представлена тремя разрядами. Старший (четвертый) разряд отведен для знака.

Из таблицы видно, что во всех случаях обратный код отрицательного числа отличается на единицу младшего разряда от дополнительного кода. Таблица также дает представление о диапазоне чисел в рассматриваемых кодах.

В таблице для заданной длины кодов видно, что диапазоны для рассматриваемых кодов различны (см. последнюю строку). Также из таблицы видна двойственность представления нуля в случае обратного кода. Последний столбец показывает целесообразность применения в сумматорах дополнительного кода.

Таблица 2.3

Коды целых чисел

Десятичное представление	Прямой код	Обратный код	Дополнительный код
+ 7	0111	0111	0111
+ 6	0110	0110	0110
+ 5	0101	0101	0101
+ 4	0100	0100	0100
+ 3	0011	0011	0011

+ 2	0010	0010	0010
+ 1	0001	0001	0001
+ 0	0000	0000	0000
-0	1000	1111	—
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	—	—	1000

Сложение обратных кодов чисел сопровождается переносом из старшего цифрового разряда в знаковый и переносом из знакового разряда в младший цифровой разряд. Такая цепочка переносов называется **циклическим переносом**.

При выполнении арифметических операций может возникнуть перенос из знакового разряда. Во избежание искажения кода числа это явление, называемое переполнением, необходимо обнаруживать. Переполнение разрядной сетки удобно обнаруживать в модифицированных кодах.

В модифицированном дополнительном коде для кода знака отводятся два разряда и знак минус обозначается двумя единицами. Знак плюс обозначается сочетанием 00. Циклический перенос не делается. Единица левее знаковых разрядов теряется. Сочетание значений 01 и 10 служат признаками переполнения.

В модифицированном обратном коде применяется такое же правило для знаков, как и в дополнительном. Сложение обратных кодов выполняется в сумматоре с циклическим переносом.

Обозначим слагаемые  $A$  и  $B$ , а сумму обозначим  $C$  и рассмотрим примеры сложения в прямом, обратном и дополнительном кодах для чисел, модуль которых меньше единицы, то есть для правильных дробей. Такие числа, как сказано выше, можно записать следующим образом:

$$[A]_{\text{пр}} = \begin{cases} A, & \text{если } A \geq 0 \\ 1+[A], & \text{если } A < 0 \end{cases}$$

При суммировании в прямом коде не делается перенос из старшего цифрового разряда в знаковый разряд.

$$\begin{array}{r}
A = -0,00110 \quad 1,00110 = [A]_{\text{пр}} \qquad 0,10111 = [A]_{\text{пр}} \\
+ \\
B = -0,01010 \quad 1,01010 = [B]_{\text{пр}} \qquad 0,00101 = [B]_{\text{пр}} \\
\hline
\qquad \qquad \qquad 1,10000 = [C]_{\text{пр}} \qquad 0,11100 = [C]_{\text{пр}} = C \\
C = -0,10000
\end{array}$$

Рассмотрим пример с переполнением при суммировании в прямом коде. Признак переполнения обозначим ф.

$$\begin{array}{r}
0,10110 = [A]_{\text{пр}} \\
+ \\
0,01101 = [B]_{\text{пр}} \\
\hline
\end{array}$$

ф,00011 Произошло переполнение. Результат неверен.

Если числа имеют разные знаки, то приходится делать операцию вычитания после предварительного сравнения кодов чисел. Результату приписывается знак большего числа. Эти дополнительные операции приводят к дополнительному усложнению сумматора.

Рассмотрим сложение чисел, представленных в дополнительном коде. Сумма дополнительных кодов есть дополнительный код результата. При сложении осуществляется перенос из старшего разряда цифровой части (мантиссы) в знаковый разряд.

$$\begin{array}{r}
0,01001 = [A]_{\text{доп}} = [A]_{\text{пр}} \\
+ \\
0,00101 = [B]_{\text{доп}} = [B]_{\text{пр}} \\
\hline
0,01110 = [C]_{\text{доп}} = [C]_{\text{пр}}
\end{array}$$

$A = -0,10010$   $1,01110 = [A]_{\text{доп}}$  Нет переноса из мантиссы в знаковый разряд

$$\begin{array}{r}
+ \\
B = 0,01010 \quad 0,01010 = [B]_{\text{доп}} \\
\hline
1,11000 = [C]_{\text{доп}}, C = -0,01000
\end{array}$$

$$\begin{array}{r}
A = 0,10001 = [A]_{\text{доп}} = [A]_{\text{пр}} \\
+ \\
B = 0,11111 = [B]_{\text{доп}} = [B]_{\text{пр}} \\
\hline
C = 1,10000
\end{array}$$

В последнем примере при сложении положительных чисел в результате переноса в знаковом разряде появилась единица. Результат неверен.

$$\begin{array}{r}
A = -0,01001 \quad 1,10111 = [A]_{\text{доп}} \\
+ \\
B = 0,01101 \quad 0,01101 = [B]_{\text{доп}} \\
\hline
0,00100 = [C]_{\text{доп}} = [C]_{\text{пр}}
\end{array}$$

Произошли переносы в знаковый разряд и из знакового разряда.

$$A = -0,10001\ 1,01111 = [A]_{\text{доп}}$$

+

$$B = -0,11010\ 1,00110 = [B]_{\text{доп}}$$

-----

$$10,10101 = [C]_{\text{доп}}$$

Произошел перенос из знакового разряда. Этот перенос в сумматоре, работающем в дополнительном коде, теряется. Получилось положительное число. Результат неверен.

Рассмотрим пример сложения обратных кодов чисел. Сумма обратных кодов чисел есть обратный код результата.

$$0,00100 = [A]_{\text{об}}$$

+

$$0,01100 = [B]_{\text{об}}$$

-----

$$0,10000 = [C]_{\text{об}}$$

$$A = -0,01100\ 1,10011 = [A]_{\text{об}}$$

+

$$B = 0,01001\ 0,01001 = [B]_{\text{об}}$$

-----

$$1,11100 = [C]_{\text{об}}, C = -0,00011$$

Рассмотрим еще пример с возникновением циклического переноса. Выбраны такие числа, чтобы возник перенос в знаковый разряд и из знакового разряда.

$$A = -0,01100\ 1,10011 = [A]_{\text{об}}$$

+

$$B = -0,01001\ 1,10110 = [B]_{\text{об}}$$

-----

$$1,01001$$

—1 → Единица циклического переноса.

После циклического переноса получается  $1.01010 = [C]_{\text{об}}$ ,

$$C = -0,10101.$$

Рассмотрим примеры операций с модифицированными кодами.

Модифицированный дополнительный код.

$$A = 0,1010\ [A]_{\text{доп. м.}} = 00,1010$$

+

$$B = -0,0101\ [B]_{\text{доп. м.}} = 11,1011$$

-----

$$100,0101$$

Единица, расположенная левее двух знаковых разрядов, отбрасывается, получаем  $[C]_{\text{доп. м.}} = [A + B]_{\text{доп. м.}} = 00,0101$ .

$$\begin{array}{r} A = -0,1010 [A]_{\text{доп. м.}} = 11,0110 \\ + \\ B = -0,0101 [B]_{\text{доп. м.}} = 11,1011 \\ \hline \end{array}$$

$$111,0001$$

С учетом потери единицы левее знаковых разрядов получаем 11,0001

$$\begin{array}{r} A = -0,1010 [A]_{\text{доп. м.}} = 11,0110 \\ + \\ B = -0,1010 [B]_{\text{доп. м.}} = 11,0110 \\ \hline \end{array}$$

$$110,1100$$

В результате отбрасывания левой единицы получаем 10,1100. Сочетание 10 в знаковых разрядах свидетельствует о переполнении разрядной сетки.

$$\begin{array}{r} A = 00,1001 \\ + \\ B = 00,1001 \\ \hline 01,0111 \end{array}$$

В этом примере сочетание 01 свидетельствует о переполнении разрядной сетки.

Модифицированный обратный код.

$$\begin{array}{r} A = 0,10001 [A]_{\text{об. м.}} = 00,10001 \\ + \\ B = -0,00111 [B]_{\text{об. м.}} = 11,11000 \\ \hline \end{array}$$

$$\begin{array}{r} \leftarrow 100,01001 \\ \hline \longrightarrow \end{array}$$

В результате циклического переноса из старшего знакового разряда в младший разряд мантиссы получаем положительное число 00,01010.

$$\begin{array}{r} A = -0,10110 [A]_{\text{об. м.}} = 11,01001 \\ + \\ B = -0,01011 [\underline{B}]_{\text{об. м.}} = 11,10100 \\ \hline \end{array}$$

$$\begin{array}{r} \left[ \leftarrow 110,11101 \right. \\ \left. \longrightarrow \right] \\ 10,11110 \end{array}$$

В знаковых разрядах признак переполнения. В результате циклического переноса получается 10,11110.

$$A = -0,10010 [A]_{\text{об. м.}} = 11,01101$$

+

$$B = -0,01011 [B]_{\text{об. м.}} = 11,10100$$

$$\begin{array}{r} \hline \left[ \leftarrow 111,00001 \right. \\ \left. \rightarrow \right] \end{array}$$

$$11,00010$$

Результат верен  $C = -0,11101$ .

$$A = 0,10011 [A]_{\text{об. м.}} = 00,1011$$

+

$$B = 0,11001 \quad [B]_{\text{об. м.}} = 00,1101$$

$$\begin{array}{r} \hline 01,01100 \end{array}$$

Результат неверен.

## 2.5. Выполнение операций над десятичными числами

Для выполнения операций над числами, представленными в десятичной системе счисления, применяют двоично-кодированные формы представления чисел (binary-coded decimal – BCD). Каждой десятичной цифре сопоставляются определенные комбинации двоичных цифр.

Система кодирования знаков некоторого алфавита, в том числе десятичных знаков, должна удовлетворять двум условиям:

А) в кодовом слове должна легко определяться граница между двумя рядом записанными знаками;

Б) каждому знаку должна соответствовать одна единственная комбинация цифр двоичного алфавита.

Двоично-десятичные коды делятся на коды *взвешенные* и *невзвешенные*. Во взвешенных кодах каждому из четырех разрядов двоичного набора  $a_{i+3}a_{i+2}a_{i+1}a_i$ , представляющего десятичную цифру, приписывается определенный вес  $g_i$ . Таким образом, значение десятичной цифры в двоичном представлении равно  $a_{i+3}g_{i+3} + a_{i+2}g_{i+2} + a_{i+1}g_{i+1} + a_i g_i$ . Использование взвешенных двоично-десятичных кодов облегчает перевод из одной системы счисления в другую. Для того чтобы было удобно выполнять операции над кодами и получился верный результат, двоично-десятичные коды должны обладать такими свойствами, как

– *единственность*, заключающаяся во взаимно однозначном соответствии десятичных цифр и кодирующих их наборов; для этого число разрядов двоичного числа должно быть не меньше, чем  $n = \log_2 k$ , где  $k$  –

основание системы счисления (для десятичной системы  $k = 10$ , а ближайшее целое  $n = 4$ ),

– **аддитивность**, заключающаяся в том, что двоично-десятичный код суммы должен быть равен сумме двоично-десятичных кодов слагаемых,

– **упорядоченность**, необходимая для выполнения логических операций; упорядоченность заключается в выполнении для двоично-десятичных кодов  $0B, 1B, \dots, 9B$  десятичных цифр  $0, 1, \dots$ , следующих условий:  $0B < 1B < \dots < 9B$  или  $0B > \dots > 9B$ ,

– **четность**, необходимая для умножения, деления, округления, проявляющаяся в том, что четным десятичным цифрам соответствуют только четные, либо только нечетные двоичные коды,

– **дополнительность**, необходимая для обнаружения переноса в старший разряд и для формирования обратного и дополнительного кодов; сущность дополнительной заключается в том, что сумма двоичного кода любой десятичной цифры и ее обратного двоичного кода должна быть равна двоичному коду цифры 9.

Рассмотрим простейший код, называемый кодом «8-4-2-1». При его использовании десятичные числа записываются в двоичной позиционной системе с естественным порядком весов. Этот код не обладает свойством самодополнительности. Поэтому обнаружение переноса в старший разряд затруднено.

Примером самодополняющегося кода служит невзвешенный код с избытком 3. В силу самодополнительности этот код удобен для выполнения преобразований над десятичными числами. В этом коде легко обнаруживается перенос, так как сумма двух слагаемых, каждое из которых берется с избытком 3, получается (на первом шаге формирования суммы в каждом десятичном разряде) с избытком 6, что исключает лишние комбинации кодов (1010, 1011, 1100, 1101, 1110, 1111). Чтобы сохранить в двоично-десятичных тетрадах избыток, равный 3, в тех тетрадах, из которых не было переноса, следует вычесть 3, а в тех тетрадах, где перенос осуществился, добавить 3. Перенос между тетрадами при этой коррекции игнорируется. Соответствие цифр десятичной системы и двоично-десятичных кодов легко устанавливается путем сложения двоичных тетрад, имеющих естественный порядок двоичного веса с кодом 0011B (3D).

0 1 2 3 4 5 6 7 8 9

0000 0001 0010 0011 0100 0101 0110 0111 1000 1001

0011 0100 0101 0110 0111 1000 1001 1010 1011 1100

Рассмотрим пример сложения чисел, представленных в коде с избытком 3

$$\begin{array}{r} 0100\ 0101\ 1000 = 125D \\ + 0101\ 0110\ 1010 = 237D \\ \hline \end{array}$$

1001 1100 ←0010 (перенос из тетрады в тетраду указан стрелкой)  
1101 1101 0011 (коррекция)

$$/0110 /1001\ 0101 = 326D$$

Знаком / показано игнорирование переноса из тетрады.

В четвертой строке примера вычитание цифры 3 делается путем сложения с ее дополнительным кодом 1101 (шестнадцатеричное представление). Код с избытком 3 мало удобен для преобразования чисел из одной системы счисления в другую.

Рассмотрим взвешенный код, называемый кодом 2421. Этот код является самодополняющимся. В этом коде удобно получать обратный и дополнительный коды и формировать перенос. Этот код не аддитивен. В коде 2421 десятичные цифры кодируются следующим образом:

0 1 2 3 4 5 6 7 8 9

0000 0001 0010 0011 0100 1011 1100 1101 1110 1111

(0101) (0110) (0111)

в скобках указаны коды, которые запрещены для устранения неединственности.

Если необходимо обнаруживать ошибки, то в коды вводятся избыточные разряды. Одним из кодов, пригодных для обнаружения одиночной ошибки, является код, называемый «2 из 5». В этом коде каждой десятичной цифре ставится в соответствие пятиразрядный двоичный код.

Воспользовавшись кодом 8421, рассмотрим выполнение операций десятичной арифметики. Десятичные числа трактуются обычно как целые числа со знаком. Отрицательные числа при использовании десятичной арифметики часто записываются в прямом коде. Но можно применять дополнительный или обратный код.

Двоично-десятичные коды можно записывать в упакованном формате. Упакованные двоично-десятичные коды хранятся по два двоично-десятичных кода (по 2 десятичные цифры) в одном байте в виде двух тетрад. При работе с указанным форматом необходимо обнаруживать перенос из третьего разряда каждого байта, то есть из старшего разряда младшей тетрады.

Наряду с упакованными BCD числами применяют неупакованные числа. В этом случае байт содержит только одну десятичную цифру. В этом формате в старшие тетрады операндов записываются нули. Например, цифра 5D записывается как 0000 0101. Этот формат удобен для умножения и деления BCD чисел.

В двоичном сумматоре десятичные цифры от 0 до 9 записываются шестнадцатеричными тетрадами и имеют значения от 0000 до 1001 соответственно. Коды 1010, 1011, 1100, 1101, 1110 и 1111 не могут использоваться в качестве кодов десятичных цифр и отводятся для знаков. Например, знак минус обозначается 1011 или 1101.

Максимальное возможное значение суммы двух десятичных цифр  $S = A + B + C \leq 19D$ . Пусть  $A + B < 10D$ , например  $0101 + 0011 = 1000$ . Сумма верна, перенос формироваться не должен. Если  $A + B = 10$ , например,  $0100 + 0110 = 1010B = 10D$ , то перенос отсутствует, сумма в двоично-десятичном коде неверна. Если  $10 < A + B + C < 15$ , например,  $1001 + 0110 = 1111B = 15D$ , то перенос отсутствует, сумма в двоично-десятичном коде неверна. Если  $A + B + C > 16$ , например,  $1001 + 1001 = 0001\ 0010$ , то сумма неверна и образовался перенос. Во втором и третьем случаях для образования переноса необходимо добавить 0110 (двоично-десятичный код 6), например,

$$0100 + 0110 + \underline{0110} = 1010 + \underline{0110} = 0001\ 0000.$$

В примере подчеркнут добавленный код цифры 6. Как видно при добавлении кода цифры 6 образовался перенос и получился правильный двоично-десятичный код суммы ( $6 + 4 = 10$ ),

Аналогично в третьем случае

$$0110 + 1001 + \underline{0110} = 1111 + \underline{0110} = 0001\ \underline{0101}.$$

Перенос из тетрады означает уменьшение содержимого тетрады на 16. В двоично-десятичном коде при переносе из тетрады для получения правильного результата значение тетрады должно уменьшаться лишь на  $10D$ . Разница (16-10) добавлена заранее. Если перенос из тетрады не произошел, то эту оставшуюся в тетраде избыточную шестерку надо вычесть из содержимого этой тетрады. Для этого делается суммирование содержимого тетрады с дополнительным двоично-десятичным кодом шести, то есть 1010. При этом перенос между тетрадами не осуществляется, так как каждая тетрада корректируется индивидуально.

Можно к полученной сумме добавлять не дополнительный, а обратный код цифры 6. В этом случае перенос не блокируется, а осуществляется циклический перенос. При работе с обратным кодом<sup>5</sup> при наличии переноса из тетрады для коррекции используется обратный код нуля, то есть 1111, а при отсутствии переноса – код 1001. Нижеприведенные примеры иллюстрируют оба способа.

---

<sup>5</sup> При применении сумматора, работающего с обратным кодом.

0001 0010 0100	0110 0011 0001	слагаемое A
+	+	
<u>0110 0110 0110</u>	<u>0110 0110 0110</u>	код 6
0111 1000 1010	1100 1001 0111	
+	+	
<u>0010 0111 0101</u>	<u>0001 1000 0101</u>	слагаемое B
1001 1111 1111	1110<0001 1100	
+	+	
<u>1010 1010 1010</u>	<u>1001 1111 1001</u>	корректирующие коды
0011 1001 1001	1000<0001<0101	
	→ > 0001	
	<u>1000 0001 0110</u>	

Так как вычислительные машины обрабатывают не только числовую информацию, но и текстовую, содержащую, кроме букв и цифр знаки препинания, математические и другие символы, необходимо применять коды для записи такой информации. Такие коды называют алфавитно-цифровыми. Алфавитно-цифровая информация преимущественно представляется восьмиразрядным слогом, называемым, как указывалось выше, *байтом*. Применение байтового формата позволяет записывать 256 различных символов. Полный алфавитно-цифровой код должен включать в себя 26 латинских строчных букв, 26 прописных латинских букв, 10 арабских цифр, 7 знаков пунктуации от 20 до 40 других символов ( + . \., /, % и т.п.). В вычислительной технике широко распространен Американский стандартный код для обмена информацией ASCII (American Standard Code for Information Interchange).

Алфавитно-цифровая информация может быть представлена кодами переменной длины, содержащими нужное число символов.

Для упрощения автоматизации обработки данных применяют весовой принцип кодирования символов. Весом символа называется двоичное число, соответствующее коду символа. При этом веса кодов букв возрастают в соответствии с алфавитным порядком. Например, код заглавной латинской буквы *A* равен  $65H$ , а код латинской буквы *B* –  $66H$ .

Если необходимо расположить список фамилий в алфавитном порядке, то при весовом принципе кодирования эта операция может быть выполнена машиной путем сравнения двоичных чисел, соответствующих кодовому описанию фамилий.

## 2.6. Формы представления чисел в ЭВМ

Требования точности, простоты масштабирования исходных данных, упрощения преобразований и повышения производительности ЭВМ противоречивы. Поэтому, как правило, не удается ограничиться одной формой машинных чисел. Наибольшее распространение получили формы представления чисел с запятой (точкой), положение которой фиксировано

относительно некоторого разряда – *F формат* (после младшего разряда – целые числа, перед старшим – правильные дроби), и форма представления чисел с плавающей запятой – *E формат*, по-другому называемый полулוגарифмической формой.

При представлении чисел в форме с фиксированной запятой (естественная форма представления чисел) один или два разряда отводятся для записи знака числа. Остальные разряды занимает мантисса числа. Если мантисса числа имеет  $n$  разрядов, то в такой разрядной сетке в случае запятой, фиксированной перед старшим разрядом числа, нельзя записать числа меньше  $2^{-n}$ . Меньшие числа образуют машинный нуль. В такой разрядной сетке не могут быть также представлены числа  $|X| \geq 1$ . Этот случай называется переполнением разрядной сетки. Диапазон чисел, представленных в этом формате, простирается от  $-(1-2^{-n})$  до  $-2^{-n}$  и от  $2^{-n}$  до  $2^n - 1$ .

Для целых чисел диапазон простирается от  $-(2^n - 1)$  до  $2^n - 1$ . Например, в микропроцессоре К580ИК80А, выпускавшемся в 80-е годы 20 века обрабатываются восьмиразрядные операнды. Такая разрядная сетка позволяет оперировать с числами от  $-128$  до  $+127$  (11111111...01111111). Двоичные числа без знака, применяемые в этом микропроцессоре наряду с числами со знаком, лежат в диапазоне от 0 до 255 (00000000...11111111). Упакованные десятичные числа (двоично-десятичные коды) без знака содержат по 2 десятичные цифры и лежат в диапазоне от 0 до 99.

Применение целых чисел позволяет более экономно использовать емкость оперативной памяти, в сравнении с применением дробных чисел. Выравнивание чисел по младшим разрядам уменьшает вероятность переполнения. Десятичные числа в случае представления в целочисленном виде выравниваются по младшим тетрадам.

Применение в ЭВМ десятичной арифметики выгодно при малом объеме вычислений, осуществляемых над большими массивами исходных данных. Благодаря отсутствию перевода в двоичную систему счисления и обратно получается значительный выигрыш времени. Если машинное слово имеет  $n$  двоичных разрядов для записи двоичных тетрад, то диапазон значений двоично-десятичных чисел ограничивается следующими пределами:

$$0 \leq |X| \leq 10^{n/4} - 1. \quad (2.16)$$

При естественной форме представления чисел вес разрядов всегда фиксирован. Если обозначить машинное представление числа квадратными скобками  $[X]$ , то число  $X = [X]K_X$ , где  $K_X$  – коэффициент, определяемый формой представления числа в вычислительной машине.

В нормальной форме (с плавающей запятой) число  $X$  запишется в следующем виде

$$X = \pm p, \pm t,$$

соответствующим записи

$$X = N^{\pm p} (\pm m),$$

где  $N$  – основание системы счисления,  $p$  – порядок числа, а  $m$  – мантисса числа. Основание системы счисления, возведенное в степень, равную порядку, называется характеристикой. Таким образом, значение числа равно произведению мантиссы на характеристику.

Обычно для удобства и для уменьшения погрешности вводят ограничения на диапазон изменения мантиссы

$$q^{-1} \leq [m] < 1. \quad (2.17)$$

Формат машинного слова с плавающей запятой содержит знаковые части и поля для мантиссы и порядка. Если условие (2.17) выполняется, то число называется *нормализованным*. Например, число  $-10110,1111$  ( $-22,9375D$ ) в форме с плавающей запятой в нормализованном представлении имеет вид

$$-0,101101111 \times 10^{00101}$$

Рассмотрим по частям последнее выражение. Основание характеристики  $10B = 2D$ . Порядок  $00101B = 5D$ . Характеристика положительная  $2^5 = 32$ . Мантисса отрицательная  $-(256 + 64 + 32 + 8 + 4 + 2 + 1)/512 = -0,716797$ . Умножаем характеристику на мантиссу. Результат, как ожидалось  $-22,9375$ .

Если для записи порядка выделено пять цифровых разрядов и один для знака, то в приведенном примере  $[p] = 000101$ . Машинное изображение мантиссы в этом случае с учетом знака  $[m_x] = 1101101111$ .

Соответствующую разрядную сетку можно представить в нижеследующем виде:

Зр	p4	p3	p2	p1	p0	Зm	m8	m7	m6	m5	m4	m3	m2	m1	m0
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

В приведенном примере  $Зm$  отмечен разряд знака мантиссы, а  $-Зр$  отмечен разряд для знака порядка.

Запишем число из примера в разрядную сетку. Учтем, что порядок положительный, а мантисса отрицательная.

0	0	0	1	0	1	1	1	0	1	1	0	1	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Обозначим число разрядов, отведенных для записи порядка (без знака)  $n$ , а число знаков мантиссы  $m$ . Тогда наименьшее абсолютное значение ненормализованного числа равно

$$2^{-m} 2^{-2^n},$$

где  $2^{-m}$  является наименьшим абсолютным значением (ненормализованной) мантиссы, а  $-2^n$  является отрицательным значением порядка, дающим наименьшее (дробное) значение характеристики. Например, если  $m = 9$ , а  $n = 5$ , то  $2^{-9} 2^{-32} = (1/2^9) \times 1/2^{32}$ . В разрядной сетке в этом случае запись имеет вид

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
Порядок						3m	СЗр	Мантисса						МЗр	

Диапазон представления нормализованных чисел лежит в пределах

$$-2^{-1}2^{2^n-1} \leq X \leq (1-2^{-m})2^{2^n-1}.$$

Общее возможное количество нормализованных чисел с плавающей запятой получается вычитанием из значения верхней границы значения, соответствующего нижней границе, с учетом знаков.

Для расширения диапазона представления чисел и для ускорения выполнения операций, содержащих сдвиги, можно применить основание системы счисления, равное степени цифры 2 ( $2^k = 4, 8, \dots; k = 2, 3, \dots$ ). При  $k = 4$  мантисса может рассматриваться как записанная двоичными тетрадами. В этом случае несложно выполнять ускоренные сдвиги мантиссы на число разрядов, кратное четырем. Каждый такой сдвиг мантиссы соответствует умножению или делению машинного числа на 16. Такое перемещение требует изменения порядка шестнадцатеричного числа на одну единицу (поскольку  $N = 16$ ).

В некоторых вычислительных машинах применяют смещенные порядки. При этом порядок числа представляется увеличенным на  $2^n$ . Порядок на числовой оси смещают в сторону положительных чисел.

Смещенный порядок

$$P_{см} = P + 2^n.$$

Диапазон значений порядка в этом случае ограничивается пределами

$$0 \leq P_{см} \leq 2^n - 1 + 2^n = 2^{n+1} - 1.$$

Например, если цифры порядка располагаются в шести разрядах, а знак в седьмом, то наименьший порядок  $-64D$  изобразится как 1 000000, нулевой порядок  $+ 0$  изобразится как 0 000000, а наибольший  $63D$  как 0 111111. При использовании смещенного порядка наименьшее значение  $- 64D + 64D$  изобразится как 0 000000, а наибольшее значение, равное  $+ 63D + 64D$ , изобразится как 1 111111. При таком изображении все порядки являются положительными числами. Это упрощает выполнение некоторых операций над порядками, так как отпадает необходимость в предварительном анализе их знаков.

При сложении чисел, представленных в форме с плавающей запятой, необходимо

- уравнивать порядки слагаемых,
- сложить мантиссы
- нормализовать результат.

При перемножении чисел, представленных в форме с плавающей запятой, перемножают мантиссы, а значения порядков суммируют.

## Вопросы для самопроверки

1. Что называется системой счисления?
2. Какими свойствами должна обладать система счисления?
3. Какая система счисления называется позиционной?
4. Что называется основанием позиционной системы счисления?
5. Какие системы счисления называются символическими?
6. Какая система счисления называется однородной?
7. Какой способ изображения цифр называется непосредственным?
8. Какая система счисления называется смешанной?
9. Какой способ изображения цифр называется кодированным?
10. Как кодируются числа в системе остаточных классов?
11. Переведите в троичную систему число  $100D$ . Результат проверьте, осуществив обратный переход.
12. Переведите в двоичную систему число  $0,1245D$ .
13. Почему преобразование десятичных чисел в двоичную систему при вводе в ЦВМ иногда осуществляется в два этапа?
14. Как строится дополнительный код отрицательного числа?
15. Как строится обратный код отрицательного числа?
16. Сложите целые двоичные числа  $-10110$  и  $+100001$ . Воспользуйтесь обратным кодом
17. Числа  $-10110$  и  $+100001$  сложите, воспользовавшись дополнительным кодом.
18. Какой код называется взвешенным?
19. Сложите числа  $22D$  и  $33D$ , представив их в двоично-десятичном коде.
20. Как записывается число в форме с фиксированной запятой?
21. Как записывается число в форме с плавающей запятой?
22. Запишите смешанное число  $-1011.011011$  в форме с плавающей запятой.
23. Как выполняется сложение чисел, записанных в форме с плавающей запятой?
24. Каков диапазон чисел, записанных в форме с фиксированной запятой, если запятая фиксирована правее младшего разряда?
25. Каков диапазон чисел, записанных в форме с плавающей запятой?
26. Как и для чего смещают порядок чисел, записанных в форме с плавающей запятой?
27. Каков диапазон двоичных восьмиразрядных чисел, представленных в дополнительном коде?
28. Каков диапазон восьмиразрядных двоичных чисел, представленных в обратном коде?

29. Почему при использовании двоичного сумматора для сложения десятичных чисел коррекция осуществляется кодом 0110В?
30. В чем заключается отличие сумматоров, предназначенных для обратных кодов, от сумматоров для сложения дополнительных кодов?
31. В двоичном сумматоре суммируются двоично-десятичные коды. Какой разряд в этом случае надо проверять, чтобы обнаружить десятичный перенос?

### 3. ЛОГИЧЕСКИЕ ОСНОВЫ ЦВМ

#### 3.1. Алгебра логики

Для формального описания работы цифровых устройств применяют аппарат алгебры логики, являющейся частью математической логики. Основным понятием алгебры логики является **высказывание**. Высказыванием будем называть некоторое предложение, о котором можно утверждать, что оно истинно или ложно. Высказывания можно обозначать символами, принятыми в математике, например  $X$ ,  $Y$ ,  $L$  и т.п. При этом, если высказывание истинное, то это обозначают единицей ( $X = 1$ ). Ложное высказывание обозначают нулем ( $X = 0$ ). В некоторых машинных программах применяют свои обозначения, например,  $T$  (True) или  $Y$  (Yes) вместо 1 и  $F$  (False) и  $N$  (No) вместо 0. **Логической переменной** будем называть такую величину  $X$ , которая может принимать значение только из набора 0 и 1,  $X = (0,1)$ .

#### 3.2. Логические функции

**Логическая функция** – это функция  $f(X_1, X_2, \dots, X_n)$  принимающая значение, равное 0 или 1 на наборе логических переменных  $X_1, X_2, \dots, X_n$ . **Набором** называют совокупность значений аргументов. Поскольку аргументы могут принимать только два значения, то область определения любой логической функции всегда конечна. Число наборов для любой функции от  $n$  двоичных аргументов равно  $2^n$ .

Так как функция алгебры логики имеет только два возможных значения и определена на  $2^n$  наборах, то разных двоичных функций от  $n$  двоичных переменных существует  $2^{2^n}$ .

Так, функций одной переменной всего 4, так как число наборов равно 2. Для функций двух аргументов число наборов равно  $2^2$ , поэтому функций от двух двоичных аргументов всего  $4^2$ . Функции одного и двух аргументов представлены в табл. 3.2.1 и 3.2.2.

Таблица 3.2.1

Функции одного аргумента  $f(X)$

Функция	Аргумент $X$		Обозначение функции	Наименование функции
	0	1		
$f_0$	0	0	0	Константа 0
$f_1$	0	1	$X$	Тождественная функция (переменная $X$ )
$f_2$	1	0	$\bar{X}$	Логическое отрицание (функция НЕ)
$f_3$	1	1	1	Константа 1

Таблица 3.2.2

Таблица функций двух переменных

Функция	X =	0	1	1	Обозначение функции	Название функции
	Y =	1	0	1		
f0	0	0	0	0	0	Константа 0
f1	0	0	0	1	$X \& Y, XY$	Конъюнкция, И
f2	0	0	1	0	$X \& \bar{Y}$	Запрет по Y, $X \uparrow \bar{Y}$
f3	0	0	1	1	X	Переменная X
f4	0	1	0	0	$Y \& \bar{X}$	Запрет по X, $Y \uparrow \bar{X}$
f5	0	1	0	1	Y	Переменная Y
f6	0	1	1	0	$X \oplus Y, X\bar{Y} + \bar{X}Y$	Сложение по модулю 2, отрицание равнозначности
f7	0	1	1	1	$X \vee Y, X + Y$	Дизъюнкция, ИЛИ
f8	1	0	0	0	$X \downarrow Y$	Стрелка Пирса, $\overline{X + Y}$
f9	1	0	0	1	$X \sim Y$	Эквивалентность, равнозначность, $X \equiv Y$
f10	1	0	1	0	$\bar{Y}$	Отрицание Y
f11	1	0	1	1	$Y \rightarrow X$	Импликация, вовлечение, $\bar{X} \& \bar{Y} + X \& \bar{Y} + X \& Y$
f12	1	1	0	0	$\bar{X}$	Отрицание X
f13	1	1	0	1	$X \rightarrow Y$	Импликация, $\bar{X} \& \bar{Y} + \bar{X} \& Y + X \& Y$
f14	1	1	1	0	$X/Y$	Штрих Шеффера, $\overline{X \& Y}$
f15	1	1	1	1	1	Константа 1

Число наборов для функции трех аргументов равно 8, а количество разных двоичных функций трех двоичных аргументов равно 256.

Логическое отрицание принято обозначать чертой над переменной ( $\bar{X}$ ) или символом  $\neg$  перед переменной ( $\neg X$ ).

Для логических функций применяют табличную или аналитическую запись. Каждой функции в таблице ставится в соответствие двоичное число. В таблице 3.2.2 мы видим эти числа в колонках для значений аргументов X и Y. Табличная форма записи функций алгебры логики наглядна, а на основе таблицы можно сделать аналитическую запись.

Из таблиц 3.2.1 и 3.2.2, как и из других аналогичных таблиц, видно, на каких наборах функция истинна, а на каких – ложна. Например, функция  $f_{13}$  в таблице 3.2.2, называемая *функцией импликации от  $X$  к  $Y$* , ложна тогда и только тогда, когда переменная  $X$  истинна, а переменная  $Y$  ложна.

### 3.3. Основные свойства логических функций

Рассмотрим основные свойства логических функций и законы алгебры логики. Воспользуемся в качестве аксиом следующими соотношениями:

$$X + 0 = X, X + 1 = 1, X + X = X, \neg X + X = 1 \\ X0 = 0, X1 = X, XX = X, \neg X X = 0 \\ \overline{\overline{X}} = X.$$

*Переместительный закон (свойство коммутативности)*

$$X + Y = Y + X, XY = YX.$$

*Сочетательный закон (свойство ассоциативности)*

$$X + (Y + Z) = (X + Y) + Z, X(YZ) = (XY)Z.$$

*Распределительный закон (свойство дистрибутивности)*

$$X(Y + Z) = XY + XZ, X + YZ = (X + Y)(X + Z).$$

Последнее выражение на первый взгляд неочевидно. Раскроем скобки:

$$(X + Y)(X + Z) = XX + XZ + XY + YZ.$$

Учитывая, что

$$XX = X,$$

запишем

$$X + XZ + XY + YZ = X(1 + Z + Y) + YZ.$$

Далее, учитывая, что  $1 + Z + Y = 1$ , запишем

$$X1 + YZ = X + YZ.$$

*Законы инверсии*, называемые часто законами де Моргана

$$\overline{XY} = \overline{X} + \overline{Y}, \overline{X + Y} = \overline{X} \& \overline{Y}.$$

Законы де Моргана и следствия из них справедливы для любого числа переменных.

*Законы поглощения*

$$X + XY = X, X(X + Y) = X. \quad (3.3.1)$$

Первое соотношение может быть переписано как

$$X(1 + Y) = X1 = X.$$

Второе соотношение сводится к первому  $X(X + Y) = XX + XY = X + XY$ .

*Формула склеивания* удобна при преобразовании выражений булевой алгебры

$$XY + X\overline{Y} = X. \quad (3.3.2)$$

Следующая формула также удобна при преобразовании алгебраических выражений

$$X + \overline{X}Y = X + Y.$$

Последнее выражение несложно проверить, приписав к левой его части выражение, равное нулю, то есть  $X\overline{X}$ , и, умножив  $X$  в левой части на выражение, равное единице, то есть  $1 + Y$ .

$$\begin{aligned} X + \overline{X}Y &= X1 + \overline{X}Y = X(1 + Y) + \overline{X}Y = X + XY + \overline{X}Y = \\ &= XX + \overline{X}X + XY + \overline{X}Y = X(X + \overline{X}) + Y(X + \overline{X}) = \\ &= (X + Y)(X + \overline{X}) = X + Y. \end{aligned}$$

Как указано выше, табличная форма записи булевых функций наглядна. В полной таблице записаны все возможные наборы аргументов и, соответственно, значения функции. В дальнейшем мы увидим, что по таблице можно получить аналитическую форму описания булевой функции.

Аналитическая форма записи удобна для преобразования и позволяет проанализировать свойства функции.

Рассмотрим аналитическое выражение функций алгебры логики. Пусть имеется функция  $\Phi(X_1, X_2, \dots, X_n)$ , равная 0 для одного из наборов аргументов и равная 1 для всех остальных наборов. Будем называть функцию  $\Phi$  **термом**. Терм, называемый дизъюнктивным, связывает все переменные, представленные в прямой или инверсной форме, знаком дизъюнкции, например,

$$\Phi = \overline{X}_1 + X_2 + X_3 + \overline{X}_4.$$

Конъюнктивный терм связывает переменные, представленные в прямой или инверсной форме, знаком конъюнкции, например,

$$F = X_1 \overline{X}_2 \overline{X}_3 X_4.$$

Любая таблично заданная функция алгебры логики может быть записана аналитически в виде дизъюнкции конъюнктивных термов. Запись логической функции в виде

$$f(X_1, X_2, \dots, X_n) = F_1 + F_2 + \dots + F_n \quad (3.3.3)$$

будем называть *объединением термов*.

Количество букв, входящих в терм, называется рангом термина [21].

Каждому набору аргументов, для которого  $f = 1$ , соответствует, хотя бы один элемент  $F_i = 1$ , а наборам, на которых  $f = 0$ , не соответствует ни одного элемента  $F_i = 1$ . Поэтому табличная форма представления логических функций однозначно отображается приведенной выше аналитической записью. Таким образом, заменяя каждую строку таблицы, в которой значение функции равно 1, конъюнктивным термом, соответствующим

этой строке, и осуществляя объединение этих термов, получаем из табличного описания логической функции ее аналитическое выражение.

Рассмотрим таблицу 3.3.3, описывающую функцию алгебры логики.

Таблица 3.3.3

Функция двух аргументов

	Первый аргумент	Второй аргумент	Значение функции
	A	B	F
Первый набор значений аргументов	0	0	1
Второй набор	0	1	0
Третий набор	1	0	1
Четвертый набор	1	1	0

В таблице 3.3.3 представлена функция двух аргументов  $A$  и  $B$ . Наборов аргументов четыре, и таблица содержит четыре строки с цифрами. Каждому набору (строке) соответствует значение функции, записанной в этой строке. Как видим, эта функция имеет только два набора аргументов, дающих значение функции, равное единице.

Подставим (в табл. 3.3.3) значения из строк, в которых  $F = 1$ , то есть из первой и третьей. Вместо нуля будем писать обозначение аргумента со знаком инверсии, а вместо единицы – без инверсии.

$$f(X_1, X_2, \dots, X_n) = F_1 + F_3 = \overline{A}B + A\overline{B}.$$

В таблице 3.3.3 имеются строка и столбец с пояснениями. Обычно эта строка и столбец отсутствуют, и таблица имеет более простой вид. Такая таблица называется **таблицей истинности** логической функции. Рассмотренная таблица содержит все наборы аргументов и поэтому называется **полной таблицей** (табл. 3.3.4).

Таблица 3.3.4.

Полная таблица истинности логической функции

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

### 3.4. Нормальные формы логических функций

Существуют формы логических функций, подчиняющихся некоторым правилам, позволяющим выделить их как класс так называемых **нормальных форм**. Дизъюнктивной нормальной формой – ДНФ, называется объединение термов, включающее конъюнктивные термы переменного ранга (различных рангов).

Количество термов, входящих в аналитическую запись функции, равно количеству единичных строк (столбцов) таблицы этой функции. Для примера рассмотрим таблицу истинности для трех аргументов (табл. 3.3.5).

Таблица 3.3.5

Таблица истинности функции трех аргументов

X	0	0	0	0	1	1	1	1
Y	0	0	1	1	0	0	1	1
Z	0	1	0	1	0	1	0	1
f(X,Y,Z)	1	0	0	1	1	0	0	0

Рассматривая столбцы этой таблицы, сделаем аналитическую запись функции f,

$$f(X, Y, Z) = F(0,0,0) + F(0,1,1) + F(1,0,0) = \overline{X}\overline{Y}\overline{Z} + \overline{X}YZ + X\overline{Y}\overline{Z}.$$

Это выражение записано в ДНФ.

Существует другая нормальная форма, называемая *конъюнктивной* – КНФ. Эта форма представляет собой объединение дизъюнктивных термов, включающее в себя термы разных рангов. Если все термы равны единице, то функция равна единице. Если хотя бы один из термов равен нулю, то и функция равна нулю.

Нормальные формы не дают однозначного представления функции. Для однозначного представления пользуются совершенными нормальными формами. Рассмотрим одну из них, называемую *совершенной дизъюнктивной нормальной формой* – СДНФ. Она обладает следующими свойствами:

- в ее записи нет двух одинаковых термов,
- ни один терм не содержит двух одинаковых переменных,
- ни один терм не содержит вместе с переменной ее отрицание,
- СДНФ содержит термы только максимального ранга (каждый аргумент входит в терм только в прямом или только в инверсном представлении).

### 3.5. Полные системы логических функций

Одни логические функции можно выражать через другие логические функции. Новые логические функции конструируются из других с помощью *суперпозиции*. Суперпозиция состоит в подстановке вместо аргументов булевой функции других переменных и в перенумерации (переименовании) аргументов.

Существуют наборы булевых функций, из которых методом суперпозиции можно получить любую булеву функцию. Такие функции образуют так называемую *функционально полную* систему логических функций.

Система функций алгебры логики  $\{f_1, f_2, \dots, f_m\}$  называется полной в некотором классе функций, если любая функция  $W$ , принадлежащая этому классу, может быть представлена суперпозицией функций  $f_1, f_2, \dots, f_m$  [1].

Рассмотрим свойства функций, образующих функционально полную систему. Все функции, существующие на наборах, соответствующих заданному числу аргументов, образуют *замкнутый класс*. Например, замкнутый класс образуют функции, приведенные в табл. 3.2.2. Замкнутый класс может содержать, а может и не содержать полную систему функций<sup>6</sup>.

Существует пять замкнутых классов особого вида, называемые *предполными классами*.

Предполные классы обладают следующим свойством: такой класс не совпадает с классом всех  $2^{2^n}$  функций, но если включить в него любую, не входящую в него функцию, то путем суперпозиции этой функции и функций предполного класса можно образовать новый класс, совпадающий с классом всех  $2^{2^n}$  функций.

Определение функционально полной системы часто дают в следующем виде: ***чтобы система была функционально полной, необходимо, чтобы она содержала по одной функции, не принадлежащей к каждому предполному классу (по крайней мере, по одной не принадлежащей к каждому классу)***.

Определение функционально полной системы можно дать в виде теоремы Поста – Яблонского [1] (см. приложение 1).

Классы функций обозначаются буквами  $K_0, K_1, S, L, M$ .

$K_0$  – класс функций, *сохраняющих константу ноль*. Для таких функций имеет место равенство  $f(0, 0, \dots, 0) = 0$ .

$K_1$  – класс функций, *сохраняющих константу единица*. Для таких функций имеет место равенство  $f(1, 1, \dots, 1) = 1$ .

$S$  – класс *самодвойственных* функций, то есть таких, *которые совпадают с двойственной себе функцией*. Для таких функций имеет место равенство  $f(X_1, X_2, \dots, X_n) = \overline{f(\overline{X}_1, \overline{X}_2, \dots, \overline{X}_n)}$  [1].

$M$  – класс монотонных функций. *Функция называется монотонной, если для любых двух наборов  $X_1$  и  $X_2$  таких, что  $X_1 \leq X_2$ , имеет место равенство  $f(X_1) \leq f(X_2)$* .

$L$  – класс линейных функций. *Функция называется линейной, если она представима в следующем виде:*

$$f(X_1, X_2, \dots, X_n) = a_0 \oplus a_1 X_1 \oplus \dots \oplus a_n X_n,$$

где  $a_i$  – постоянные 0 или 1.

---

<sup>6</sup> Здесь не приводится доказательство теоремы о функциональной полноте, а дается только упрощенное описание.

Для установления линейности функции необходимо воспользоваться алгеброй Жегалкина<sup>7</sup>. В этой алгебре используются операции конъюнкции ( $XY$ ), суммы по модулю 2 ( $X \oplus Y$ ), константы 1 (не зависящей от аргументов).

Запись функции стрелка Пирса в виде полинома Жегалкина выглядит следующим образом:

$$\overline{(X + Y)} = 1 \oplus (X + Y) = 1 \oplus X \oplus Y \oplus XY.$$

Это выражение является полиномом второй степени, следовательно, функция стрелка Пирса нелинейная.

Многие логические функции относятся одновременно к нескольким классам, поэтому в функционально полную систему может входить менее пяти функций. Принадлежность функции к определенным классам представлена в таблице свойств логических функций (см. приложение 3). Например, функции отрицания и конъюнкции  $\bar{X}$  и  $XY$ , из которых первая не сохраняет 0, не сохраняет 1 и не монотонна, а вторая нелинейна и не самодвойственна, образуют функционально полную систему. Функции штрих Шеффера и стрелка Пирса каждая в отдельности образуют функционально полную систему.

### 3.6. Минимизация логических выражений

Важной задачей является минимизация логических выражений. При минимизации (упрощении) логическую функцию, заданную в виде таблицы или формулы, приводят к простейшей формуле, соответствующей заданной функции. Под *простейшей* будем понимать формулу, содержащую наименьшее количество элементарных логических функций И, ИЛИ, НЕ (в базисе булевой алгебры). Не существует общих методов, позволяющих стандартными приемами определить вид этой простейшей формулы. Наиболее разработаны методы отыскания простейших формул в классе нормальных форм. К сожалению, получающаяся в результате преобразования формула необязательно простейшая. Чтобы попытаться получить более простую запись, надо начать с исходной записи и применить другой порядок преобразований<sup>8</sup>.

Необходимость упрощающих преобразований следует из того, что ***формулы алгебры логики являются математическими моделями, на основе которых строятся узлы вычислительных машин.***

Для дизъюнктивных нормальных форм под минимальными формами понимаются ДНФ, содержащие наименьшее суммарное число переменных во всех термах.

---

<sup>7</sup> Дополнительные сведения об алгебре Жегалкина см. в приложении.

<sup>8</sup> Получающиеся в результате преобразований логических выражений формулы, называются тупиковыми формами.

Одним из наиболее распространенных методов нахождения сокращенных ДНФ является метод Квайна. Для реализации этого метода функция представляется в виде СДНФ. Чтобы привести функцию к СДНФ, применяют *операцию разворачивания*, заключающуюся в умножении некоторых членов на выражение  $X + \bar{X}$ , равное 1. К парам термов применяют последовательно операции *склеивания* (3.3.2) и элементарного *поглощения* (3.3.1). В СДНФ проводят все возможные операции склеивания конъюнкций ранга  $n$  ( $n$  – число аргументов функции). Получаются конъюнкции  $n-1$  ранга. К ним применяются операции поглощения. Затем применяются все возможные операции склеивания конъюнкций  $n-1$  ранга и т. д.

Возникает вопрос о числе разных тупиковых форм, которые могут быть получены для конкретной функции. Поиск наименьшей тупиковой формы путем перебора может оказаться очень трудоемким.

Для облегчения преобразований пользуются таблицами различного вида, например, диаграммами Вейча. Диаграммы Вейча и другая разновидность диаграмм, называемая картами Карно, являются графическим средством для упрощения логических выражений или преобразования таблиц истинности в соответствующую логическую схему сравнительно простым и упорядоченным способом.

В таких таблицах число клеток равно числу всех возможных наборов аргументов. Так таблица для функций трех аргументов содержит 8 клеток, а таблица для функций четырех аргументов – 16, как показано на рис. 3.1. Для каждого набора аргументов в таблице должна быть своя клетка. Столбцы и строки таблицы помечают наименованиями аргументов.

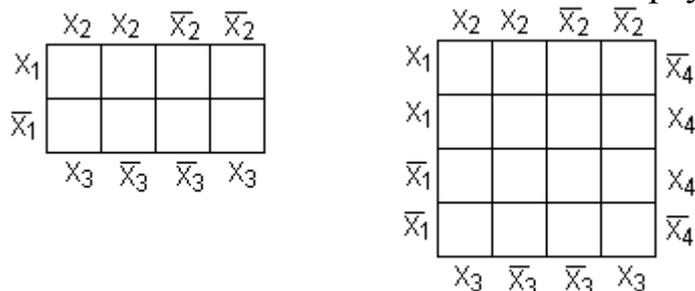


Рис. 3.1. Варианты диаграмм для минимизации функций

Для каждого набора аргументов в клетки диаграммы подставляются соответствующие значения функции, то есть 0 или 1. Любая пара склеивающихся единичных термов располагается в соседних горизонтальных или вертикальных клетках. Для функции трех аргументов соседними являются также клетки, расположенные в первом и последнем столбцах. Для диаграммы функции трех аргументов возможно склеивание по 2 или по 4 клетки (по 2 или по 4 терма).

Для диаграммы функции четырех аргументов соседними считаются клетки не только первого и последнего столбцов, но и верхней и нижней строк. Для диаграммы функций четырех аргументов возможно склеивание по 2, 4 или 8 термов.

Процесс отыскания минимальной ДНФ заключается в том, чтобы всю совокупность единиц диаграммы Вейча накрыть наименьшим числом наиболее коротких произведений.

Заполненная диаграмма для функции трех аргументов

$$f = X_1 X_2 X_3 + \bar{X}_1 \bar{X}_2 X_3 + \bar{X}_1 X_2 X_3 + \bar{X}_1 \bar{X}_2 \bar{X}_3 + X_1 X_2 \bar{X}_3$$

приведена на рис. 3.2.

	$X_2$	$X_2$	$\bar{X}_2$	$\bar{X}_2$
$X_1$	1	1	0	0
$\bar{X}_1$	1	0	1	1
	$X_3$	$\bar{X}_3$	$\bar{X}_3$	$X_3$

Рис. 3.2. Пример заполненной диаграммы Вейча.

Из диаграммы следует, что

$$f = \bar{X}_1 \bar{X}_2 + X_1 X_2 + X_2 X_3.$$

Для упрощения процесса разметки столбцов и строк диаграммы можно использовать циклические коды<sup>9</sup>. Свойством циклического кода является то, что при каждой записи нового значения кода в сравнении с предыдущей записью изменяется значение только одного разряда. Примером циклического кода служит следующий двухразрядный двоичный код 00 01 11 10. Здесь записаны все 4 возможные значения, и далее код начнет повторяться. Пример разметки показан на рис. 3.3.

	$X_1 X_2$	00	01	11	10
$X_3 X_4$	00				
	01				
	11				
	10				

Рис. 3.3. Диаграмма, столбцы и строки которой помечены с помощью циклического кода

При минимизации логических функций с помощью диаграмм следует придерживаться следующих общих правил [23]:

– следует стремиться к тому, чтобы число объединяемых ячеек было как можно больше. Чем больше ячеек объединено, тем проще выражение функции;

<sup>9</sup> Циклический код может непрерывно повторяться, например код Грея.

- число объединяемых ячеек кратно степени 2;
  - объединяемые ячейки образуют прямоугольную область;
  - группами должны быть охвачены все единицы. Если какая-то единица не может быть ни с чем объединена, соответствующий терм остается в исходном виде;
  - одни и те же единицы могут многократно использоваться в разных группах;
  - общее число групп ячеек должно быть как можно меньше.
- Объединять можно либо единицы, либо нули, в зависимости от того, какой вариант проще. Далее, если не возникнет специальных случаев, будем ориентироваться на вариант с группировкой единиц.

### Вопросы для самопроверки

1. Какое понятие является основным в алгебре логики?
2. Чему равно число наборов для функции  $n$  двоичных аргументов?
3. Сколько существует двоичных функций  $n$  двоичных аргументов?
4. Запишите распределительный закон для логических функций.
5. Запишите законы инверсии.
6. Запишите законы поглощения.
7. Что называется нормальной дизъюнктивной формой логических функций?
8. Какие логические функции образуют функционально полную систему? Приведите примеры.
9. Какие функции называются линейными?
10. Перечислите, к каким классам функций относится функция Стрелка Пирса?
11. Что понимается под простейшей формой логической функции?
12. Что понимают под простейшей формой ДНФ?
13. Минимизируйте аналитически выражение  $XY\bar{Z} + X\bar{Y}\bar{Z} + \bar{X}\bar{Y}Z$ .
14. Минимизируйте с помощью диаграммы Вейча выражение  $\bar{X}\bar{Y}Z + XYZ + \bar{X}\bar{Y}\bar{Z} + X\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} + XY\bar{Z}$ .
15. Минимизируйте с помощью диаграммы Вейча выражение  $\bar{X}YZV + \bar{X}YZ\bar{V} + \bar{X}YZV + XYZ\bar{V}$ .
16. Является ли функционально полной система функций  $F = XY$ ,  $G = \bar{X}$ ?
17. Является ли функционально полной система функций  $F = 1$ ,  $G = X\bar{Y} = \bar{X}Y$ ?
18. Можно ли с помощью функции Штрих Шеффера составить выражения для получения других функций булевой алгебры?
19. Имеются 2 двухвходовые схемы совпадений (схемы И). На первую поданы сигналы  $A$  и  $B$ , на вторую поданы сигналы  $C$  и  $D$ . Выход-

ные сигналы этих схем поданы на входы двухвходовой схемы ИЛИ. Запишите выражение для сигнала на выходе схемы ИЛИ.

20. Имеются 2 двухвходовые схемы совпадений (схемы И). На первую поданы сигналы А и В, на вторую поданы сигналы С и D. Выходные сигналы этих схем поданы на входы двухвходовой схемы ИЛИ. С первой схемы И сигнал на схему ИЛИ подан непосредственно. Сигнал с выхода другой схемы И подан на вход схемы ИЛИ через инвертор. Запишите выражение для сигнала на выходе схемы ИЛИ.

## 4. ЭЛЕМЕНТНАЯ БАЗА ЦВМ

### 4.1. Комбинационные схемы

Значение логической функции определяется набором (комбинацией) значений аргументов. Можно построить устройство, сигнал, на выходе которого будет связан с входными сигналами так же, как значение логической функции со значениями ее аргументов. Такое устройство называется комбинационной схемой, рис. 4.1.

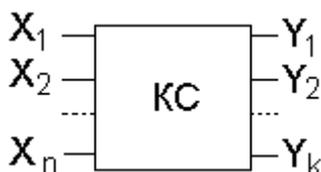


Рис. 4.1. Условное графическое обозначение комбинационной схемы

Комбинационную (логическую) схему можно описать таблицей истинности или булевым выражением. Для рассмотрения физики работы комбинационных схем нужно перейти на уровень принципиальных электрических схем. Предварительно рассмотрим математическое описание схемы.

На рис. 4.1 входы обозначены  $X_1, \dots, X_n$ . Выходы обозначены  $Y_1, \dots, Y_k$ . Входные сигналы обозначим  $P$ . В соответствии с рисунком, будем рассматривать устройство с  $n$ -входами и  $k$ -выходами. Каждый из входных сигналов может принимать значение из набора символов, образующих входной алфавит

$$P = \{P_1, P_2, \dots, P_l\}. \quad (4.1)$$

Выходные сигналы могут принимать значения из конечного набора символов, составляющих выходной алфавит.

$$S = \{S_1, S_2, \dots, S_m\}. \quad (4.2)$$

В качестве символов этих алфавитов будем использовать двоичные цифры.

В комбинационных схемах совокупность выходных сигналов (выходное слово  $Y$ ) в дискретный момент времени  $t_i$  однозначно определяется входными сигналами (входным словом  $X$ ), поступившими на входы в дискретный момент времени  $t_i$ .

Результат, с точностью до задержки распространения сигнала в схеме и времени переключения, вырабатывается сразу после подачи сигналов на вход. Другими словами, закон функционирования комбинационной схемы определяется соответствием между словами ее входного и выходного алфавитов. В (4.3) показана аналитическая форма задания соответствия.

$$Y_1 = f_1(X_1, X_2, \dots, X_n) \quad (4.3)$$

## 4.2. Цифровые автоматы

**Цифровой автомат**, в отличие от комбинационной схемы, имеет конечное число различных внутренних состояний. Внутренние состояния характеризуются значениями токов (или напряжений) электронных компонентов, образующих цифровой автомат. Набор различных состояний конечен и может быть описан. Обозначим конечный алфавит состояний

$$Q = \{q_0, q_1, \dots, q_r\}.$$

Под воздействием входного слова (входного кода) цифровой автомат переходит из одного состояния в другое<sup>10</sup> и выдает выходное слово [17].

В цифровых автоматах реализуется накапливающий принцип обработки информации, при котором информация накапливается в течение нескольких тактов работы. В каждом отдельном такте вырабатывается и запоминается в виде соответствующего состояния автомата промежуточный результат. Наконец в последнем такте запоминается окончательный результат переработки информации.

Выходное слово цифрового автомата (сочетание сигналов на выходных контактах) в дискретный момент времени  $t_i$  определяется входным словом, поступившим в этот момент времени и внутренним состоянием автомата, которое является результатом воздействия на автомат входных слов в предыдущие моменты времени. Комбинация входного слова и текущего состояния автомата в данном такте определяет не только выходное слово, но и то состояние, в которое перейдет автомат к началу следующего такта.

Цифровой автомат обязательно содержит запоминающие элементы, фиксирующие состояние, в котором автомат находится. Кроме запоминающих элементов, цифровой автомат содержит комбинационные схемы (рис. 4.2).

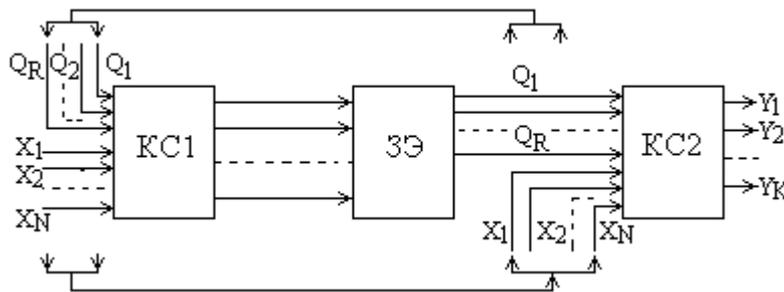


Рис. 4.2. Структура цифрового автомата

На рисунке представлены две комбинационные схемы: КС1 и КС2. Входные сигналы, после прохождения через комбинационную схему КС1, переводят автомат (запоминающие элементы автомата) в новое состояние. Аргументами для КС1 служат символы входного слова и состояния запоминающих элементов.

<sup>10</sup> Возможно, что входное слово оставляет состояние автомата без изменений.

Комбинационная схема КС2 формирует входное слово. Аргументами для нее служат символы входного слова и состояния запоминающих элементов.

Для задания (описания работы) цифрового автомата необходимо указать:

алфавит входных сигналов,  $P = (P_1, P_2, \dots, P_l)$ ,

алфавит выходных сигналов,  $S = (S_1, S_2, \dots, S_m)$ ,

алфавит состояний  $Q = (q_0, q_1, \dots, q_p)$ ,

начальное состояние автомата  $q_0$ ,

функции переходов  $A(q, X)$ ,

функции выходов  $B(q, X)$ .

Входной сигнал  $X$  принимает значения из своего алфавита  $P$ , выходной сигнал  $Y$  принимает значения из алфавита  $S$ . Функция перехода описывает, в какое состояние перейдет автомат, находящийся в состоянии  $q_i$ , если подан сигнал  $X_j$ . Функция выходов описывает сигнал на выходных контактах в зависимости от состояния  $q_i$  и входного сигнала  $X_j$ .

Алфавиты  $P$ ,  $S$  и  $Q$  и функции  $A$  и  $B$  однозначно определяют зависимость состояния автомата  $q(t+1)$ <sup>11</sup> и выходного сигнала  $Y(t)$  от состояния автомата  $q(t)$  и входного сигнала  $X(t)$  в момент дискретного времени  $t$ . Рассмотренный автомат, называемый автоматом Мили, отличается сложностью функционирования. Широкое применение в вычислительной технике находят более простые автоматы, называемые автоматами Мура.

Действие автомата Мура описывается уравнениями для состояния

$$Q(t+1) = A[q(t), X(t)],$$

и сигнала

$$Y(t) = B[q(t)].$$

Выходной сигнал для автомата Мура  $Y(t)$  в момент дискретного времени  $t$  зависит только от состояния автомата  $q(t)$  в этот момент времени и не зависит от входного сигнала  $X(t)$ .

Функции переходов и функции выходов могут быть заданы таблицей переходов и таблицей выходов (рис. 4.3). В этих таблицах столбцы соответствуют всем состояниям (каждый столбец одному состоянию), а строки – всем сигналам из соответствующего алфавита. На пересечении строки входного сигнала  $P_i$  и столбца состояния  $q_j$  записывается состояние, в которое перейдет автомат из состояния  $q_j$  под действием входного сигнала  $P_i$ , а в таблице выходов записывается выдаваемый при этом сигнал.

В двоичном автомате на каждый вход  $X_i$  может поступать сигнал  $P_i$ , равный 0 или 1. Сигнал, не воздействующий на вход можно не отмечать, то есть не выделять для него строку. Сигнал 1, поданный на вход  $X_1$ , отмечаем просто, как  $X_1$  (вместо того, чтобы писать «значение  $P_i$  на

---

<sup>11</sup> Эта запись означает «состояние в момент времени  $t+1$ ».

входе  $X_i$  равно единице»). Для выходов единицу, например на  $S_1$ , будем обозначать просто  $S_1$ . Нули в таблицу записывать не будем,  $S_0 := S_0 = 1$ ,  $S_1 := S_1 = 1$  и т.д.

	$q_0$	$q_1$	$q_2$	$q_3$
$X_1$	$q_1$	$q_2$	$q_3$	$q_0$
$X_2$	$q_0$	$q_0$	$q_0$	$q_0$
$X_3$	$q_3$	$q_0$	$q_1$	$q_2$
$X_4$	$q_0$	$q_1$	$q_2$	$q_3$

	$q_0$	$q_1$	$q_2$	$q_3$
$X_1$	$S_1$	$S_2$	$S_3$	$S_0$
$X_2$	$S_0$	$S_0$	$S_0$	$S_0$
$X_3$	$S_3$	$S_0$	$S_1$	$S_2$
$X_4$	$S_0$	$S_1$	$S_2$	$S_3$

Рис. 4.3. Таблица переходов и таблица выходов цифрового автомата

Для автомата Мура таблицу переходов и таблицу выходов можно свести в одну таблицу. Эту таблицу называют *отмеченной* таблицей переходов (рис. 4.4). Она содержит дополнительную строку, в которой каждое состояние автомата отмечается соответствующим выходным сигналом.

Значения входного сигнала X	Значения выходного сигнала Y			
	$S_0$	$S_1$	$S_2$	$S_3$
	Состояния			
	$q_0$	$q_1$	$q_2$	$q_3$
$X_1$	$q_1$	$q_2$	$q_3$	$q_0$
$X_2$	$q_0$	$q_0$	$q_0$	$q_0$

Рис. 4.4. Отмеченная таблица переходов

Таблицы переходов и выходов полностью задают автомат. В них наряду с функциями переходов и выходов указываются алфавит состояний, входной и выходной алфавиты и начальное состояние.

При построении узлов ЦВМ, являющихся цифровыми автоматами, в качестве элементов памяти используются элементарные автоматы. Элементарными автоматами служат автоматы Мура с двумя состояниями, двумя различными выходными сигналами, обладающие полными системами переходов.

Автоматом с полной системой переходов называется автомат, который для любых состояний  $q_i$  и  $q_j$  имеет входной сигнал, переводящий автомат из состояния  $q_i$  в состояние  $q_j$ .

Другое, часто применяемое название автоматов с памятью – **последовательностные схемы**.

Наглядно задать закон функционирования автомата с памятью можно с помощью направленного графа. Для автомата Мура в узлах графа записывают состояния автомата, а около узлов значения выходных сигнала-

лов. Переходы между состояниями (узлами) обозначаются ветвями (ребрами). Ветви отмечают входными сигналами. Например, сохранение состояния  $Z_j$  изображается, как показано на рис. 4.5. Если под действием сигнала  $X_i$  происходит переход автомата из состояния  $Z_j$  в состояние  $Z_k$ , то это обозначается, как показано на рис. 4.6.

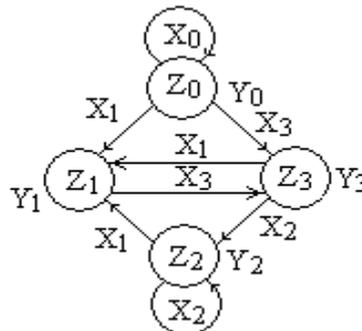
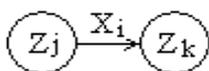
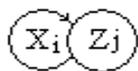


Рис. 4.5. Сохранение состояния

Рис. 4.6. Переход

Рис. 4.7. Граф автомата Мура

Граф автомата Мура изображен на рис. 4.7. Такой граф является наглядной диаграммой состояний.

При малом числе состояний графическое изображение закона функционирования является удобным. При большом числе состояний более удобна табличная форма описания.

### 4.3. Способы физического представления информации

Непрерывным требованием к физическим аналогам символов двоичного алфавита является возможность надежного распознавания двух различных значений сигнала. Известны три способа физического представления информации: *потенциальный, импульсный и динамический*.

Значение сигнала изменяется во времени. Время, протекающее между двумя сменами сигнала, называется длительностью такта (периода) представления информации.

Потенциальный сигнал сохраняет постоянный уровень в течение всего такта представления информации. Значение потенциального сигнала в переходные моменты (на границах тактов) не является определенным.

При импульсном способе представления информации единичное и нулевое значения двоичной переменной изображаются или положительными и отрицательными электрическими импульсами, или наличием и отсутствием импульсов в соответствующей точке схемы. Импульсные сигналы должны появляться в заданные моменты дискретного времени.

При динамическом способе представления информации двум возможным значениям переменной соответствуют наличие или отсутствие серии импульсов или синусоидальных колебаний, заполняющих весь период представления.

В перечисленных способах представления информации наряду с амплитудой в качестве информационного параметра могут быть использованы и используются фаза и частота сигнала.

Рассмотрим импульсный и потенциальный сигналы.

Импульсный сигнал характеризуется следующими параметрами:

Амплитудой  $U_m$ ;

Временем нарастания (фронта),  $\tau_{\phi 0}$ ;

Длительностью среза (спада),  $\tau_{c0}$ ;

Длительностью импульса (по основанию),  $\tau_{и0}$ ;

Величиной спада вершины импульса  $\Delta U$ ;

Активной длительностью импульса  $\tau_n$ , измеряемой на уровне  $0,5U_m$ .

При наблюдении импульсов на экране осциллоскопа удобно оперировать активными длительностями нарастания и спада, измеряемыми по моментам пересечения уровней  $0,1U_m$  и  $0,9U_m$ . При этом  $\tau_{\phi} = t_{0,9} - t_{0,1}$  и  $t_c = t_{0,1} - t_{0,9}$ , как показано на рис. 4.8.

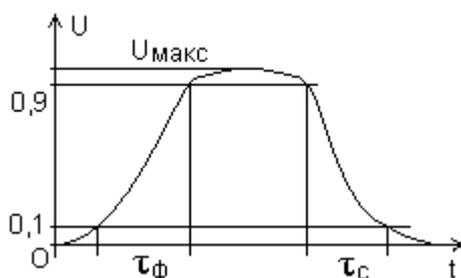


Рис. 4.8. Основные параметры импульсного сигнала

Эти же параметры относятся и к потенциальному сигналу. Кроме того, потенциальный сигнал характеризуется разностью верхнего и нижнего уровней напряжения  $U_c$ . Поскольку возможны отклонения от заданных верхнего и нижнего уровней сигнала, то обычно ограничивают две зоны. Зону единичного сигнала ограничивают снизу величиной  $U_{1мин}$ , минимальным допустимым значением. Зону нулевого сигнала ограничивают сверху величиной  $U_{0макс}$ , максимальным допустимым значением. Между этими значениями лежит зона неопределенности, рис. 4.9.

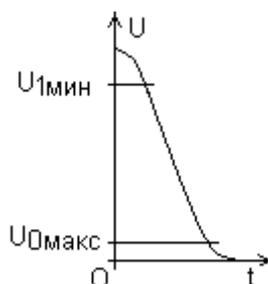


Рис. 4.9. Дополнительные параметры потенциального сигнала

В соответствии с типом используемых сигналов цифровые устройства могут быть построены на динамических, импульсных, потенциальных или импульсно-потенциальных компонентах. В современных цифровых вычислительных машинах широко применяются интегральные микросхемы потенциального типа.

#### 4.4. Технические аналоги функций алгебры логики

В вычислительной технике нашли применение наиболее просто реализуемые устройства, воспроизводящие функции алгебры логики. При необходимости другие более сложные устройства строятся путем композиции из простых логических схем.

На схемах логические элементы обозначают прямоугольниками. Входные контакты (входы) указываются слева, а выходы – справа. Инверсия выходного сигнала обозначается кружком со стороны выхода схемы. Внутри прямоугольника указывается название схемы или выполняемая ею функция. Название схемы обозначается условными символами или аббревиатурой.

Число входов логического элемента равно числу аргументов воспроизводимой им логической функции.

Рассмотрим основные логические функции.

Схема, реализующая операцию дизъюнкции, называется *логической схемой ИЛИ*. Логическое выражение, описывающее ее работу, приведено ранее в таблице логических функций. Условное обозначение двухвходовой схемы ИЛИ приведено на рис. 4.10.

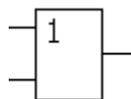


Рис. 4.10. Условное обозначение схемы ИЛИ

Двухвходовая диодная схема ИЛИ для положительных потенциальных и импульсных сигналов приведена на рис. 4.11.

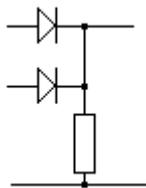


Рис. 4.11. Двухвходовая диодная схема ИЛИ

Для рассмотрения свойств схемы выпишем часть табл. 3.2.2 для функции  $f_7$  в ином виде. Высокий потенциал обозначим буквой  $B$ , а низкий (нулевой) – буквой  $H$ . Из таблицы видно, что рассматриваемая схема для высокого потенциала реализует функцию дизъюнкции, а для низкого – функцию конъюнкции.

X	Y	F
H	H	H
H	B	B
B	H	B
B	B	B

Резистор нагрузки  $R$  часто в явном виде в схеме не присутствует. Его роль играет входное сопротивление следующего каскада схемы. Поэтому при рассмотрении схемы

ИЛИ в составе некоторого устройства в виде реальных компонентов остаются только диоды, рис. 4.12.

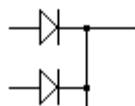


Рис. 4.12

Схема, реализующая операцию конъюнкции, называется *схемой И* или *схемой совпадений*. Она обозначается, как показано на рис. 4.13. В таблице логических функций двух аргументов она обозначена  $f_1$ . Диодная схема, выполняющая эту функцию для двух входных сигналов, приведена на рис. 4.14.

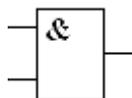


Рис. 4.13. Условное изображение схемы совпадений

X	Y	F
H	H	H
H	B	H
B	H	H
B	B	B

Составим, пользуясь обозначениями  $H$  и  $B$ , таблицу для логической функции  $f_1$ . Из таблицы видно, что схема конъюнкции также меняет свою функцию в зависимости от того, какой сигнал является действующим  $H$  или  $B$ . Это свойство взаимозаменяемости схем И и ИЛИ называется *дуальностью* (двойственностью).

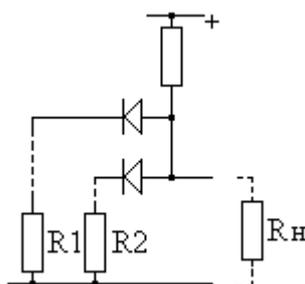


Рис. 4.14. Диодная схема совпадений для положительных сигналов

В потенциальной системе элементов сопротивления  $R_1$  и  $R_2$  являются обычно элементами устройства, служащего источником сигнала для схемы И, а  $R_n$  – элементом схемы нагрузки. Поэтому в сложных схемах схема совпадения имеет физически вид, представленный на рис. 4.15.

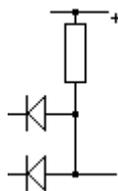


Рис. 4.15. Выделенная отдельно схема совпадения

Свойство дуальности схем И и ИЛИ позволяет в ряде случаев уменьшить число инверторов в многокаскадных схемах.

Инвертором называется устройство, реализующее операцию инверсии – НЕ. Эта схема обозначается, как показано на рис. 4.16.

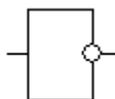


Рис. 4.16. Условное обозначение инвертора

Схема инвертора реализуется на одном транзисторе, работающем в ключевом режиме, рис. 4.17. Величины сопротивлений резисторов  $R_1$  и  $R_2$  и потенциала смещения базы  $E_{см}$  выбираются таким образом, чтобы при нулевом уровне сигнала на входе потенциал базы был ниже уровня эмиттера на величии ну приблизительно 0,1–0,3 В. При единичном уровне сигнала на входе ток базы должен быть равен току базы насыщения  $I_{бн}$ . Как видно из рисунка, в случае применения биполярного транзистора, он включается по схеме с общим эмиттером.

В интегральной микросхемотехнике инвертор применяется как в составе схем И-НЕ (например микросхемы К555ЛА3 и К1533ЛА3), так и в виде отдельных инверторов, например микросхема К555ЛН1 (6 инверторов).

Из рассмотренных логических схем синтезируются более сложные логические устройства. Последовательное соединение логических схем соответствует суперпозиции логических функций. В качестве базового элемента микросхем ТТЛ часто применяется схема, реализующая функцию штрих Шеффера. Эта схема получается путем последовательного включения схем И и НЕ. В интегральных микросхемах, изготавливаемых по технологии, принятой для биполярных транзисторов схема И выполняется на многоэмиттерном транзисторе. В качестве примера рассмотрим  $\frac{1}{4}$  часть микросхемы К155ЛА3 рис. 4.18.

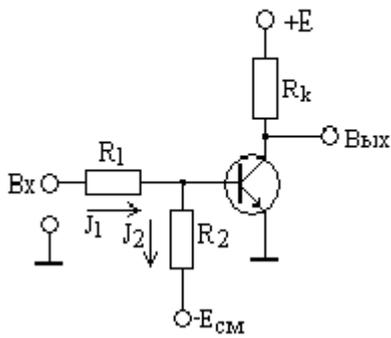


Рис. 4.17. Инвертор

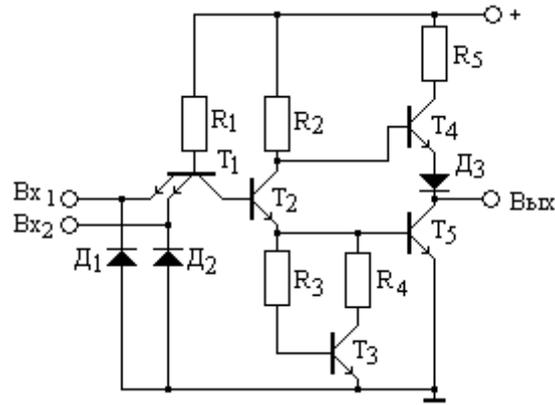


Рис. 4.18. Микросхема К155ЛА3

При последовательном соединении выходной сигнал одного элемента является входным для следующего элемента.

При рассмотрении многокаскадных устройств из-за большого числа элементов схемы представление на уровне принципиальных электрических схем неудобно. Поэтому далее сложные схемы будем рассматривать на логическом уровне. Обратимся к рис. 4.19. Здесь выходной сигнал  $G$  получен как функция промежуточных сигналов  $E$  и  $F$  и как функция входных сигналов  $A$ ,  $B$ ,  $C$  и  $D$ .

$$G = \overline{E + F} = \overline{AB + CD} = ABCD.$$

Двухкаскадные схемы принято обозначать прямоугольниками, разделенными на две части вертикальной чертой. Эти части соответствуют первой и второй ступеням, как показано на рис. 4.20.

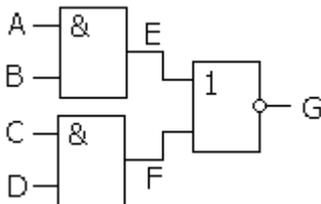


Рис. 4.19. Двухкаскадная логическая схема

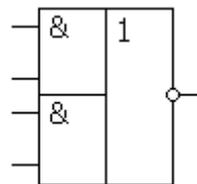


Рис. 4.20. Условное обозначение двухкаскадной логической схемы

Более подробно двухкаскадное соединение логических элементов рассматривается при изучении регистров.

#### 4.5. Передача информации между элементами цифровой вычислительной машины

Соединение логических элементов осуществляется с помощью электрических цепей. Для передачи бита информации используется одна электрическая линия. Передача информации осуществляется при протекании электрического тока. Поэтому цепь, состоящая из передатчика и приемника, должна быть замкнута, рис. 4.21.

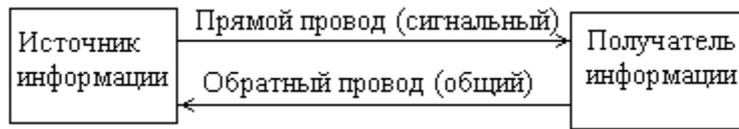


Рис. 4.21. Соединение источника и получателя информации

Обратный провод может также называться нулевым или землей. Если говорят «одна электрическая линия» или «однопроводная цепь», это не означает, что проводник действительно один. Эти термины говорят о том, что имеется одна замкнутая цепь для протекания тока. Эти термины относятся к представлению устройств на уровне логических схем, которые на рисунке (только на рисунке) соединяются одной линией, показывающей направление передачи информации.

Однопроводная цепь, по которой передается сигнал, представляющая прямое или обратное представление двоичной переменной называется *однофазной*, рис. 4.22. Двухпроводная цепь, по которой одновременно передаются оба сигнала, представляющие прямое и обратное значения двоичной переменной, называется *парафазной*<sup>12</sup>, рис. 4.23.

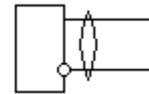
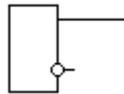


Рис. 4.22. Условное обозначение Рис.4.23. Двухфазная (парафазная) цепь однофазной цепи

Совокупность электрически независимых цепей, предназначенных для передачи слова (поля, байта), называется *шиной*, рис. 4.24. Для управления процессом передачи информации (передачи или блокирования передачи) в цепи передачи включаются *вентили* (схемы И), рис. 4.25, возбуждаемые сигналами управления операциями (микрооперациями) передачи.

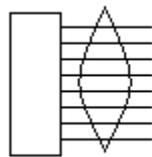


Рис. 4.24. Шина Рис. 4.25. Схема И в качестве вентиля

#### 4.6. Последовательный и параллельный способы представления и передачи информации

Машинное слово может быть представлено последовательным или параллельным кодом. При последовательном способе представления информации каждый временной такт предназначается для представления

<sup>12</sup> Парафазная цепь физически имеет обратный проводник для сигнала каждой фазы или один общий проводник для обеих фаз.

одного разряда слова. Номер разряда при этом отображается номером такта и отсчитывается от некоторого нулевого положения, совпадающего с началом представления слова. Возможны два способа представления информации: младшими разрядами вперед или старшими разрядами вперед. При последовательном способе представления информации все разряды слова фиксируются по очереди одним и тем же элементом и проходят по одному каналу передачи информации.

При параллельном способе представления информации все разряды слова обрабатываются в одном временном такте, фиксируются отдельными элементами и проходят через отдельные каналы, каждый из которых служит для представления и передачи только одного разряда слова. Устройства параллельного действия требуют большего количества аппаратуры в сравнении с последовательными устройствами, но работают значительно быстрее, чем последовательные устройства.

Из-за несовпадения продолжительности переходных процессов в различных цепях не может быть обеспечена одновременность появления новых сигналов на всех входах принимающих эти сигналы устройств.

Если схемы не содержат запоминающих устройств, то есть являются комбинационными схемами, в них может быть реализован только асинхронный способ передачи информации.

В устройствах, содержащих запоминающие элементы (которые могут сохранять значения битов информации), используются как асинхронный, так и синхронный способы передачи информации. При синхронном способе передача информации осуществляется управляемым образом под воздействием специальных синхронизирующих сигналов. Для синхронизации информации могут быть использованы элементы задержки. Условное обозначение элемента задержки на схемах приведено на рис. 4.26. Синхронная передача сигналов, осуществляемая с помощью элементов задержки, является однотоктной.

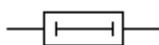


Рис. 4.26. Условное обозначение элемента задержки

Новый такт начинается лишь после того, как в предыдущем такте завершится выработка комбинационной схемой выходного слова и его запоминание путем установки запоминающих элементов в соответствующие устойчивые состояния.

С помощью синхронизирующих сигналов обеспечивается передача полученной в предыдущем такте информации с запоминающих элементов на входы комбинационных схем одновременно с сигналами, поступающими на их входы от других устройств.

Синхронная передача сигналов осуществляется также с помощью промежуточных запоминающих элементов. При этом нужны дополни-

тельные синхронизирующие сигналы. В устройствах с промежуточными запоминающими устройствами, построенных на основе потенциальной системы элементов, применяют, как правило, двухтактную передачу сигналов. При этом применяют две последовательности синхронизирующих сигналов или фронт и срез синхронизирующих импульсов одной последовательности. Если последовательностей синхронизирующих сигналов две, то импульсы разных последовательностей сдвинуты один относительно другого. Длительность синхронизирующих сигналов должна быть больше, чем время переходных процессов элементов устройства.

Принцип однократной передачи информации проиллюстрирован на рис. 4.27.

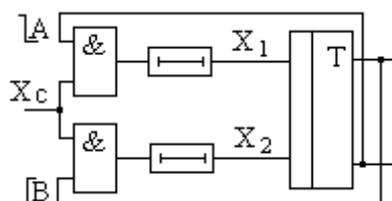


Рис . 4.27. Устройство с однократной передачей сигналов.

В устройстве, показанном на рис. 4.27, осуществляется однократная передача сигнала, поступающего по цепи обратной связи. Элемент задержки задерживает сигнал на время, большее длительности входного сигнала  $X_c$ , исключая тем самым ложное срабатывание. Поэтому в течение длительности сигнала  $X_c$  информация на входах  $A$  и  $B$  не успевает смениться.

Принцип двухтактной передачи сигналов пояснен на рис. 4.28.

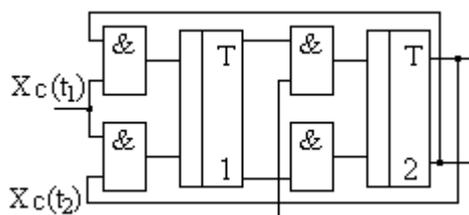


Рис. 4. 28. Устройство с двухтактной передачей сигнала

В устройстве, изображенном на рис. 4.28, сигнал переписи информации из элемента  $T_1$  в элемент  $T_2$ . должен быть подан после окончания переходного процесса в  $T_1$ .

#### 4.7. Системы логических элементов

Наборы логических элементов, электрические характеристики которых согласованы таким образом, чтобы их можно было соединять, создавая более сложные устройства, образуют **системы логических элементов**. Для системы логических элементов так же, как и для логических функций, вводится понятие **функциональной полноты**.

Система логических элементов обладает функциональной полнотой, если при помощи конечного числа элементов можно построить произвольную комбинационную схему с любым законом функционирования.

Для построения цифровых автоматов система элементов, кроме функциональной полноты для комбинационных схем, должна еще содержать запоминающий элемент (элементарный автомат с двумя устойчивыми состояниями).

Для удобства проектирования цифровых устройств системы элементов чаще всего делают избыточными.

К основным характеристикам логических элементов и систем логических элементов относятся:

- значения питающих напряжений и сигналов для представления логического нуля и логической единицы;
- коэффициенты объединения по входам И и ИЛИ ( $m, 1$ );
- нагрузочная способность (коэффициент разветвления);
- рассеиваемая мощность;
- быстродействие, характеризуемое средним временем задержки распространения сигнала и
- помехоустойчивость.

Коэффициент объединения показывает число входов, имеющих у данной логической схемы. Коэффициент разветвления показывает, сколько входов однотипных элементов можно подключить к выходу данного логического элемента. Среднее время задержки распространения сигнала определяется как суммарное время задержки сигнала в последовательной цепочке логических элементов, поделенное на число элементов в этой цепочке.

Логические элементы должны обладать устойчивостью к действию помех. Помехоустойчивость определяется величиной дополнительного сигнала, который может быть добавлен к входному сигналу логического элемента, прежде чем выходное напряжение этого логического элемента станет выше максимального уровня логического нуля или ниже минимального уровня логической единицы.

#### **4.8. Триггерные устройства ЦВМ**

В качестве запоминающих элементов в наборах элементов применяют элементарные автоматы с двумя устойчивыми состояниями, однозначно определяющими выходной сигнал. Будем далее для таких цифровых автоматов сигнал на выходе и внутреннее состояние обозначать одним символом  $Q$ . Применяемые в вычислительной технике элементарные автоматы называются триггерами.

Если автомат имеет  $n$  входов, на которые могут быть поданы сигналы 0 и 1, а на его выходе возможны 5 состояний 0, 1,  $\bar{Z}(S)$ ,  $Z(S)$  и неопределенное, то можно задать  $5^{2^n}$  различных законов функционирования автомата. При  $n = 1$  можно задать 25 различных элементарных автоматов, а при  $n = 2$  число таких автоматов равно 625. Многие из этих автоматов не находят применения, а другие – тривиальны.

Рассмотрим триггеры, имеющие от одного до трех независимых входных зажимов и два выходных. Из возможного многообразия таких устройств широкое применение нашли триггеры типов  $RS$ ,  $T$ ,  $JK$ ,  $D$  и  $DV$ , имеющие один или два информационных выхода.

#### 4.9. RS триггер

Триггером типа  $RS$  называется триггер с установочными входами. Он применяется в качестве запоминающего элемента. Этот триггер построен таким образом, что при подаче на его входы определенной комбинации сигналов, можно принудительно приводить его в необходимые состояния. Триггер называется в соответствии с назначением его входов, которое обозначается буквами  $S$  (Set – установка) и  $R$  (Reset – сброс). На вход  $S$  подается сигнал, обеспечивающий установку триггера в состояние, принятое за единицу, а на вход  $R$  подается сигнал, устанавливающий триггер в исходное (нулевое) состояние.

Структурная схема  $RS$  триггера, выполненного на логических элементах ИЛИ-НЕ, и его условное обозначение приведены на рис. 4.29. Вертикальной чертой разделяются поля для обозначения входов  $R$  и  $S$  и для обозначения триггера –  $T$ .

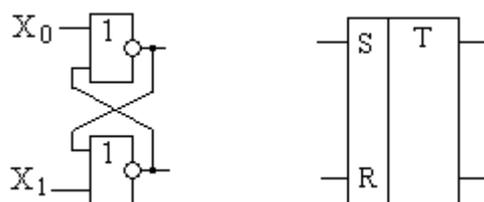


Рис. 4.29. Структурная схема триггера на элементах ИЛИ-НЕ и условное графическое обозначение триггера на схемах

Под воздействием сигнала  $X_0 = 1$  триггер устанавливается в нулевое состояние, обозначаемое  $Q = 0$ . Под воздействием сигнала  $X_1 = 1$  триггер типа  $RS$  устанавливается в состояние  $Q = 1$ . Переходы  $RS$  триггера (смена состояний) представлены в табл. 4.9.1.

Таблица переходов  $RS$  триггера

$t_n$			$t_{n+1}$
$Q_n$	$R_n$	$S_n$	$Q_{n+1}$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	x
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	x

Сочетание сигналов  $X_0 = 1$  и  $X_1 = 1$  является запрещенным. После такого воздействия состояние триггера однозначно не определяется. При подаче сигналов  $X_0 = 0$  и  $X_1 = 0$  одновременно триггер устанавливается в состояние хранения информации, то есть его состояние не изменяется.

Сигналы и состояния в  $n$ -м такте обозначены буквой  $n$ , а состояние, в которое устройство переходит в следующем такте работы, соответственно, отмечено  $n + 1$ . Запрещенные состояния обозначены буквой  $x$ .

На основе таблицы дадим аналитическое описание работы  $RS$  триггера в виде дизъюнктивной нормальной формы, записывая термы, как конъюнкции значений переменных для строк таблицы, в которых  $Q_{n+1} = 1$ . При записи формулы опустим индексы у символов  $S$  и  $R$ .

$$\bar{Q}_{n+1} = \bar{Q}_n \bar{R} S + Q_n \bar{R} S + Q_n \bar{R} \bar{S} = \bar{R} S + \bar{R} Q_n = \bar{R}(S + Q_n) \quad (4.9.1)$$

Чтобы избежать неопределенности, устройства, содержащие  $RS$  триггеры, построенные на элементах ИЛИ-НЕ, строят таким образом, чтобы на входах  $R$  и  $S$  исключалась комбинация сигналов  $RS = 1$ . С учетом этой оговорки функционирование триггера описывается равенствами (4.9.1) и (4.9.2).

$$R_n S_n = 0 \quad \text{или} \quad \bar{S}_n + \bar{R}_n = 1 \quad (4.9.2)$$

Выражение (4.9.2) указывает на то, что на любом шаге сигнал либо на входе  $R$ , либо на входе  $S$ , либо на обоих входах должен быть равен нулю.

Рассматриваемый  $RS$  триггер является статическим асинхронным устройством и реагирует на сигнал, имеющий уровень логической единицы.

Триггер, изображенный на рис. 4.29, часто называют *защелкой* [20].

Закон функционирования, как указывалось выше, можно описывать с помощью графов. Граф функционирования рассматриваемого  $RS$  триггера изображен на рис. 4.30.

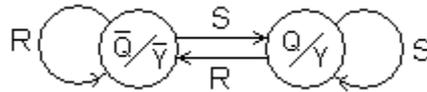


Рис. 4.30. Граф асинхронного RS триггера

Ветви графа помечены символами входных сигналов, а узлы символами состояний и выходных сигналов. На рис. 4.30 выходные сигналы обозначены  $Y$  и  $\bar{Y}$ .

Если на обозначении триггера входные контакты помечены кружочками, как показано на рис. 4.31, то это означает, что действующим сигналом является логический 0. Такой триггер может быть реализован на логических элементах, осуществляющих операции И и НЕ, что соответствует функции  $F14$ <sup>13</sup>. Схема триггера показана также на рис. 4.31. Этот триггер изменяет свое состояние при подаче сигналов 0 и 1 или 1 и 0 и сохраняет свое состояние при сигналах  $R = 1, S = 1$ . Независимо от исходного состояния триггер устанавливается в новое состояние в соответствии с входными сигналами. Поэтому в таблице состояний для этого триггера можно опустить колонку, помеченную  $Q_0$ .

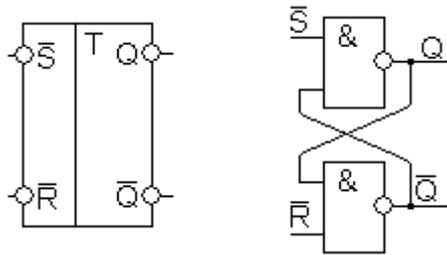


Рис. 4.31. Условное обозначение и логическая схема RS триггера, для которого активным входным уровнем является 0

Запишем сокращенную таблицу переходов RS триггера.

$\bar{R}_n$	$\bar{S}_n$	$Q_{n+1}$	
0	0	X	Запрещено
0	1	0	Установка 0
1	0	1	Установка 1
1	1	$Q_n$	Хранение

Триггер является асинхронным. Переходный процесс начинается в момент изменения сигналов на входах. Если на входы подать сигналы  $S = 0$  и  $R = 0$ , то состояние, в которое перейдет триггер, при поступлении сигналов  $R = 1, S = 1$  окажется непредсказуемым (новое состояние определяется помехами). Если же сигналы  $R = 1, S = 1$  подать после комбинаций 01 или 10, то состояния, инициированные этими комбинациями сигналов, сохраняются.

<sup>13</sup> Смотрите выше таблицу булевых функций двух переменных.

Как указывалось выше, описание вычислительной машины и ее узлов можно делать разной степени детальности. Описание  $RS$  триггера на уровне принципиальной электрической схемы будет приведено далее в разделе, посвященном оперативной памяти. Условное обозначение асинхронного  $RS$  триггера, схема, составленная из логических элементов, таблица переходов и направленный граф приведены в этом разделе. В дополнение к этим способам рассмотрим диаграмму состояний для асинхронного  $RS$  триггера, выполненного на элементах И-НЕ, рис. 4.32.

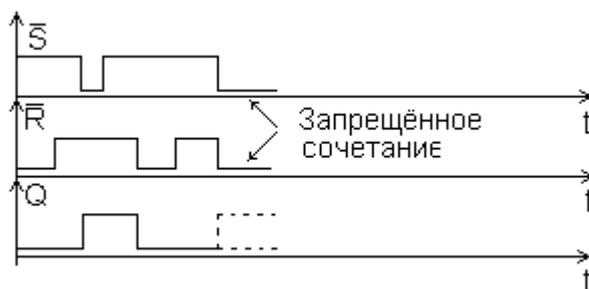


Рис. 4.32. Временная диаграмма входных и выходного сигналов для  $RS$  триггера, выполненного на элементах И-НЕ

Цифровые системы могут работать в асинхронном или синхронном режимах. В асинхронных системах состояния на выходах могут изменяться в любое время, когда меняется входной сигнал.

В синхронных устройствах, как указывалось ранее, дополнительно применяют специальные сигналы, которые точно задают момент времени установки выходного сигнала. Такие сигналы называют сигналами синхронизации или тактовыми.

Тактовые сигналы обычно представляют собой последовательность прямоугольных<sup>14</sup> импульсов. Большая часть устройств в системе изменяет свое состояние в момент поступления одного из фронтов тактовых импульсов. В соответствии с моментом времени формирования фронты называют передними и задними. Задний фронт часто называют срезом, а передний – просто фронтом. Большинство синхронных импульсных устройств синхронизируется фронтом.

Ранее были рассмотрены триггеры – защелки, не имеющие входа для синхронизирующего импульса. Триггеры являющиеся синхронными имеют специальный вход для подачи сигнала синхронизации. В целом синхронный триггер может иметь синхронные управляющие (информационные) входы, входы предварительной установки – асинхронные и вход для сигналов синхронизации. Схема такого триггера приведена на рис. 4.33.

<sup>14</sup> Не следует забывать, что импульсы на самом деле имеют *приблизительно* прямоугольную форму.

Она имеет дополнительный асинхронный вход  $R_1$ , служащий для установки триггера в нулевое состояние. Схема построена на элементах ИЛИ-НЕ. По синхронным входам  $R$  и  $S$  и по входу тактового сигнала триггер переключается нулевым уровнем, а по асинхронному входу  $R_1$  под действием уровня логической единицы. Когда  $R_1$  равно 1, то на вход синхронизации нельзя подавать сигнал 0. Клапаны 1 и 2 образуют схему входной логики.

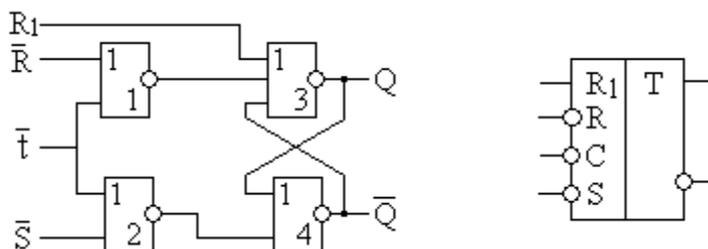


Рис. 4.33. Логическая схема и условное обозначение синхронного  $RS$  триггера

Часть схемы, выполненная на вентилях 3 и 4, представляет собой рассмотренный ранее асинхронный  $RS$  триггер. Ниже, при рассмотрении других типов триггеров, мы увидим, что этот элемент является базовым для других типов статических триггеров. Статическими называются триггеры, обладающие устойчивыми состояниями.

Функционирование синхронного триггера описывается двумя таблицами: синхронной и асинхронной. Асинхронная таблица определяет действие сигналов, подаваемых на вход  $R_1$ . Если  $R_1 = 0$ , то следует пользоваться синхронной таблицей, так как в этом случае можно подавать сигналы на вход синхронизации и состояние триггера определяется сочетанием сигналов на входах  $R$  и  $S$ . Так как для синхронных входов действующим сигналом является 0 (см. рис. 4.33), то сочетание сигналов  $R = 0, S = 0$  недопустимо.

Асинхронная таблица		
$R$	$Q_{n+1}$	Комментарии
1	0	Установка нуля
0	Состояние триггера задаётся синхронной таблицей	

Синхронная таблица			
$t$		$t+1$	Комментарии
$R_n$	$S_n$	$Q_{n+1}$	
0	0	X	Запрещено
0	1	0	Установка 0
1	0	1	Установка 1
1	1	$Q_n$	Хранение

Поэтому либо  $R$ , либо  $S$ , либо оба сигнала должны быть равны 1. Это можно описать выражением  $\bar{R} \& \bar{S} = 0$  или  $R + S = 1$ . Длительность переходного процесса в синхронном одноканальном триггере равна трем временам задержки распространения сигнала элементами И-НЕ (ИЛИ-НЕ), если не учитывать задержку распространения в соединительных проводниках.

В ранее рассмотренном асинхронном триггере время переходного процесса (переключения) равно двум временам задержки.

Если триггер переключается в момент поступления фронта синхроимпульса, то вход сигнала синхронизации помечается треугольником, как показано на рис. 4.34.

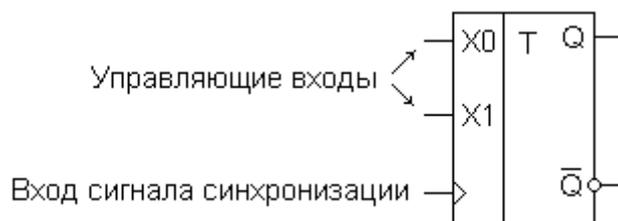


Рис. 4.34. Входы синхронного триггера.  
Треугольник означает синхронизацию фронтом

Неидеальность формы импульсов, разброс параметров триггеров, а также время, в течение которого заряжается входная емкость триггера, требуют соблюдения некоторых временных соотношений между моментом подачи сигнала на синхронный управляющий вход и вход сигнала синхронизации. Рассмотрим эти времена: **время установки** триггера и **время удержания** триггера.

**Время установки** (предустановки [24]) – это интервал времени, предшествующий управляющему фронту синхронизирующего сигнала. В течение этого времени сигнал на информационном входе не должен изменяться (рис. 4.35). Другими словами синхронизирующий сигнал не должен поступить слишком рано.

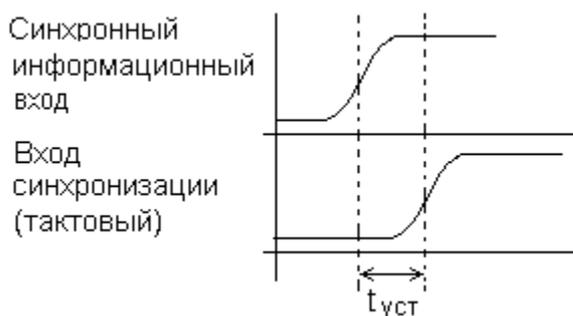


Рис. 4.35. Время установки синхронного триггера.

**Время удержания** (выдержки [24]) – это интервал времени, после поступления управляющего фронта тактового сигнала. В течение этого времени информационный вход должен оставаться в одном и том же состоянии, то есть сигнал на этом входе не должен меняться (рис. 4.36).

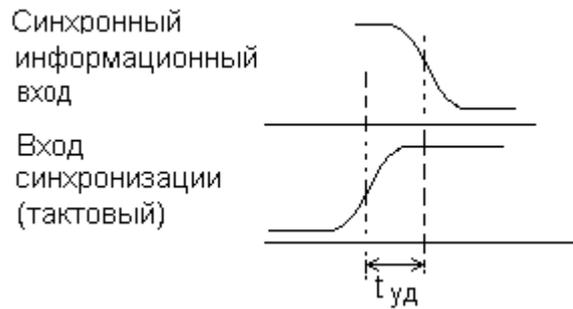


Рис. 4.36. Время удержания синхронного триггера

При осциллографических измерениях эти интервалы замеряются до достижения тактовым импульсом 50% своей амплитуды.

Таким образом, чтобы синхронный триггер правильно реагировал на сигнал в момент прихода тактового импульса, управляющие входы должны оставаться в стабильном состоянии (не изменять уровень потенциала), по крайней мере, в течение интервала времени, равного  $t_{уст\ мин}$ , до прихода нужного фронта тактового импульса и как минимум в продолжение интервала времени, равного  $t_{уд\ мин}$ , после прихода нужного фронта [20].

Двухтактный *RS* триггер рассмотрим в исполнении на клапанах И-НЕ, рис. 4.37.

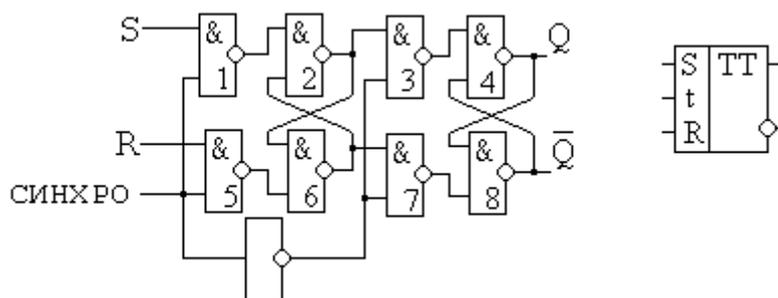


Рис. 4.37. Двухтактный *RS* триггер

Эта схема иллюстрирует двухтактную передачу информации под действием фронта и среза синхронизирующего импульса. Этот триггер часто называют *MS* триггером (master-slave). Как следует из схемы, двухтактный триггер собран из двух базовых *RS* триггеров. Действующим сигналом в данном случае является уровень единицы. При подаче единицы на вход СИНХРО входная информация записывается в первый триггер, выполненный на элементах 2 и 6. В это время на входную логику второго триггера (элементы 3 и 7) действует инвертированный синхросигнал. Поэтому запись во второй триггер заблокирована. По окончании действия синхросигнала, на входной логике второго триггера будет высокий уровень синхросигнала, и информация скопируется из первого триггера во второй. Переходы двухтактного *RS* триггера приведены в табл. 4.9.2.

Сокращенная таблица переходов RS триггера

t		t + 1	Примечания
S	R	$Q_{n+1}$	
0	0	$Q_n$	Хранение
0	1	0	Установка 0
1	0	1	Установка 1
1	1	X	Запрещено

Используя базовый *RS* триггер, можно сконструировать другие типы триггеров

#### 4.10. Триггер типа *T*

Триггером типа *T* называют триггер со счетным входом. Этот триггер изменяет свое состояние каждый раз при подаче на вход единичного сигнала. При подаче на вход периодической последовательности импульсов осуществляется деление частоты в два раза. Условное графическое обозначение *T* триггера приведено на рис. 4.38.



Рис. 4.38. Обозначение *T* триггера в схемах. Слева синхронный *T* триггер, справа синхронный *T* триггер с входами установки и сброса

С учетом начального состояния описание работы *T* триггера дается следующей таблицей, где  $Q(t)$  состояние триггера в момент времени  $t$ ,  $X(t)$  – входной сигнал,  $Q(t + 1)$  – состояние, в которое переходит триггер под действием этого сигнала.

$Q(t)$	$X(t)$	$Q(t + 1)$
0	0	0
0	1	1
1	0	1
1	1	0

Из таблицы следует, что *T* триггер осуществляет суммирование исходного состояния и входного сигнала по модулю 2 (функция f6 в табл. 3.2.2)

$$Q(t+1) = \overline{X(t)}Q(t) + X(t)\overline{Q(t)}.$$

Простая структура  $T$  триггера приведена на рис. 4.39. В этом триггере используются два одноканальных синхронных  $RS$  триггера. В схеме образованы два канала для распространения информации. Это верхний и нижний ряды элементов на рис. 4.39. Сигнал с выхода  $Q$  по цепи обратной связи подается из

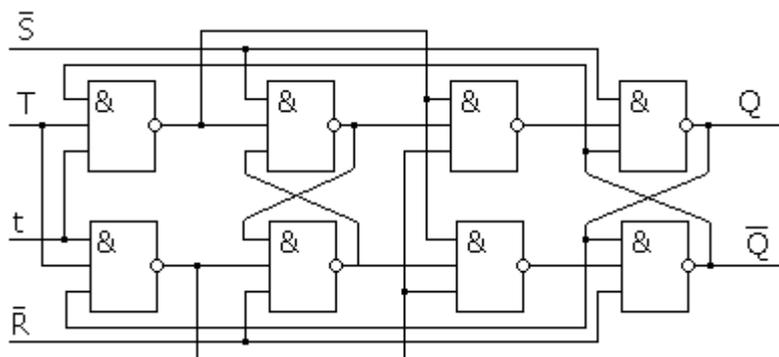


Рис. 4.39. Структура двухтактного  $T$  триггера

одного канала в другой канал. Соответственно сигнал с выхода  $\bar{Q}$  подается в первый канал. Поэтому при каждом воздействии на входы триггер изменяет свое состояние. Инверсные входы  $\bar{S}$  и  $\bar{R}$  служат для асинхронной установки в заданное состояние. Для этих входов одновременная подача нулевых сигналов на оба входа запрещена. Триггер работает в синхронном режиме, когда на оба асинхронных входа поданы сигналы, имеющие уровень логической единицы. В первый триггер информация записывается при  $t = 1$ . Перенос информации во второй триггер происходит при  $t = 0$ . Поэтому передача информации между триггерами должна осуществляться с задержкой сигналов синхронизации. Во время переходного процесса в триггере входные сигналы должны быть неизменными. Общее время задержки распространения сигнала для этого триггера составляет  $6t_3$ .

Триггер, схема которого приведена на рис. 4.39, может применяться в асинхронном режиме. Для этого на вход  $T$  надо подать уровень логической единицы, а информационный сигнал подать на вход  $t$ .

#### 4.11. $D$ триггер

Триггер типа  $D$  называют элементом задержки. Этот цифровой автомат функционирует таким образом, что его состояние и выходной сигнал в момент времени  $t + 1$  повторяет значение входного сигнала в момент времени  $t$ , то есть  $Q(t + 1) = X(t)$ . Выходное состояние зависит только от сигнала на входе и не зависит от предыдущего состояния.

Триггерные устройства  $D$  типа в интегральной схемотехнике применяются для построения одноканальных счетчиков, сдвигающих регистров,

распределительных устройств. При построении  $D$  триггеров применяются преимущественно две схемы: схема типа «M-S» и схема с тремя базовыми триггерами.

Рассмотрим вариант со схемой на трех триггерах.

Схема  $D$  триггера (микросхема К155ТМ2) приведена на рис. 4.40. Этот триггер является триггером, синхронизируемым по фронту тактового сигнала. Элементы 5 и 6 образуют основной триггер, элементы 1, 2, 3 и 4, верхний и нижний коммутирующие триггеры. Как видно из рисунка, схема является несимметричной.

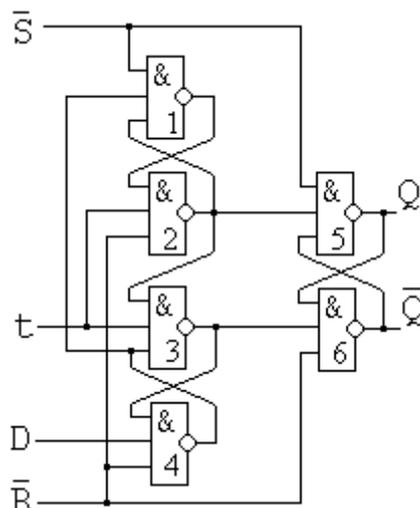


Рис 4.40. Структура микросхемы К155ТМ2

Триггер, выполненный на элементах 1 и 2, служит для записи 1 в основной триггер. Триггер, выполненный на элементах 3 и 4, – для записи нуля. Запись информации во вспомогательные триггеры происходит при переходе синхронизирующего сигнала от уровня «0» к уровню «1».

Синхронизирующий импульс, поданный после смены уровня на входе  $D$ , то есть в  $i + 1$  такте, устанавливает триггер в новое состояние. Естественная задержка в элементах схемы составляет  $4t_3$ . Если у такого триггера соединить выход  $\bar{Q}$  с входом  $D$ , то он будет работать как асинхронный триггер со счетным входом (рис. 4.41).

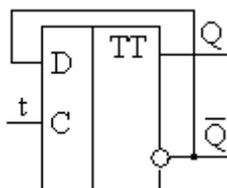


Рис. 4.41. Включение  $D$  триггера в качестве асинхронного счетного триггера

При использовании схемы, приведенной на рис. 4.40, в синхронном режиме, на входах  $\bar{R}$  и  $\bar{S}$  следует установить сигнал «Лог. 1». При асин-

хронной установке триггера в заданное состояние на один из асинхронных входов подается сигнал с уровнем «Лог. 0».

В качестве  $D$  триггера можно применить синхронный двухтактный  $RS$  триггер. Для этого к входу  $R$  необходимо подключить инвертор, как показано на рис. 4.42.

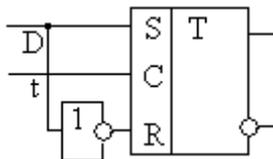


Рис. 4.42. Построение  $D$  триггера на основе  $RS$  триггера

Развернем подробнее эту схему (рис. 4.43) и дополнительно рассмотрим действие входа  $C$ .

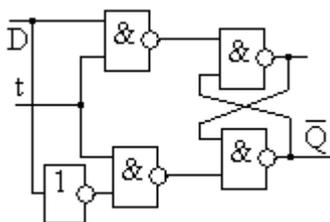


Рис. 4.43. Логическая схема  $D$  триггера, выполненного на основе  $RS$  триггера

Подав на вход  $t$  через некоторое время после того, как на вход  $D$  был подан информационный сигнал, сигнал, имеющий уровень логического нуля, мы отключаем триггер от входа  $D$ . Принято говорить, что триггер блокируется сигналом нулевого уровня по входу  $t$ . При сигнале на входе  $t$ , равном «Лог. 1», триггер прозрачен для сигнала, подаваемого на вход  $D$ . Такой  $D$  триггер принято называть **триггер-защелка**.

#### 4.12. $JK$ триггер

Универсальный триггер, способный работать, как в режиме  $RS$  триггера, так и в режиме  $T$  триггера, называется  $JK$  триггером. В качестве примера рассмотрим схему К155ТВ1 (рис. 4.44а). По входам  $J$  и  $K$  микросхема содержит пятиходовые схемы совпадения, что позволяет объединять на входах сигналы от разных источников. Три входа из каждой пяти соединены с внешними контактами микросхемы. Входы  $J$  и  $K$  синхронные. Для асинхронной установки триггера имеются входы  $\bar{R}$  и  $\bar{S}$ . Сигналы, подаваемые на эти входы, действуют непосредственно на элементы 4 и 8 выходной ступени. Чтобы устройство могло работать в режиме триггера типа  $T$ , в нем имеются перекрестные обратные связи с выхода  $Q$  на вход  $K$  и с выхода  $\bar{Q}$  на вход  $J$ .

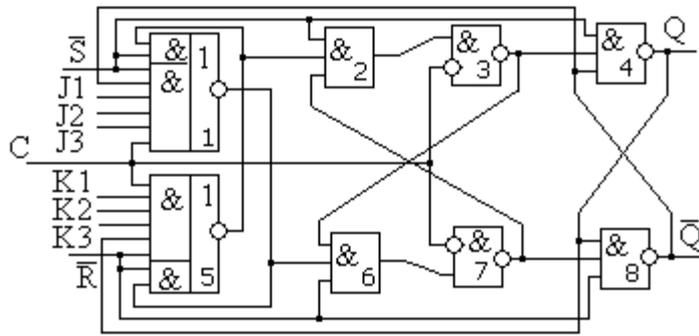


Рис. 4.44а. Структура *JK* триггера. (Микросхема К155ТВ1)

Упрощенный вариант схемы *JK* триггера приведен на рис. 4.44б.

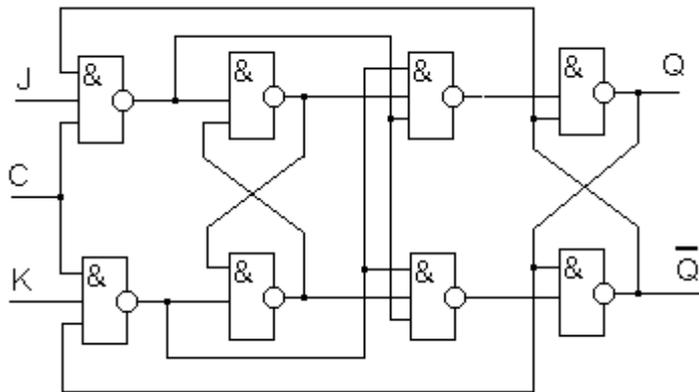


Рис. 4.44.б. Упрощенная структура *JK* триггера.

Работа триггера, изображенного на рис. 4.44.а, описывается синхронной и асинхронной таблицами.

Синхронная таблица

Асинхронная таблица

Т		Т + 1	Состояние	Асинхронная таблица			
J	K	$Q_n + 1$		R1	S1	Q	Состояние
0	0	$Q_n(t)$	Хранение	0	0	X	Запрещено
0	1	0	Установка 0	0	1	0	Установка 0
1	0	1	Установка 1	1	0	1	Установка 1
1	1	$Q_n(t)$	Инверсия	1	1	Синхронная таблица	

По входам  $\bar{R}_1$ ,  $\bar{S}_1$  триггер реагирует на нулевой уровень. При  $R_1 = S_1 = 1$  сигнал на входе *J* устанавливает триггер в единичное состояние. Сигнал 1 на входе *K* устанавливает триггер в нулевое состояние,

независимо от предыдущего состояния. Таким образом, входы  $J$  и  $K$  соответствуют входам  $S$  и  $R$  триггера типа  $RS$ . В этом триггере можно одновременно подавать единицы на входы  $J$  и  $K$ . При этом  $JK$  триггер будет функционировать, как триггер со счетным входом.

На основании таблицы переходов  $JK$  триггера его функционирование (функцию переходов) можно представить в виде логической функции от переменных, соответствующих состоянию и входным сигналам для одного и того же момента времени  $t$ .

$$Q(t+1) = \overline{J(t)}\overline{K(t)}Q(t) + J(t)\overline{K(t)} + J(t)K(t)\overline{Q(t)} = \\ = J(t)\overline{K(t)} + J(t)Q(t) + \overline{K(t)}Q(t)$$

Триггер типа  $JK$  функционирует как  $RS$  триггер, если наложить ограничение  $JK = 0$ . Для построения  $D$  триггера надо сигнал на вход  $K$  подать через инвертор, как показано на рис 4.45. Схема синхронного  $T$  триггера, построенного на основе  $JK$  триггера, изображена на рис. 4.46.

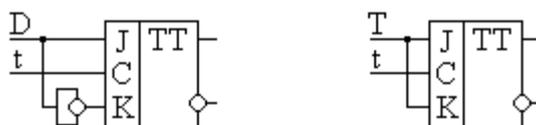


Рис. 4.45. Триггер типа  $RS$  Рис. 4.46. Триггер типа  $T$

### 4.13. $DV$ триггер

Еще одним универсальным триггером является  $DV$  триггер. Этот триггер имеет дополнительный вход  $V$  и, соответственно, дополнительный клапан в сравнении с  $D$  триггером. Если на вход  $V$  подан сигнал, соответствующий логической единице, то  $DV$  триггер работает в соответствии с таблицей переходов для  $D$  триггера.

Если  $V = 0$ , то триггер сохраняет свое текущее состояние. Если нет входного сигнала на входе  $D$  ( $D = 0$ ), то «лог 1» на входе  $V$  устанавливает  $DV$  триггер в нулевое состояние. В отличие от  $JK$  триггера комбинация входных сигналов  $V = D = 1$  не переводит  $DV$  триггер в противоположное состояние, а устанавливает его в состояние 1. Схема  $DV$  триггера приведена на рис. 4.47.

В таблице переходов  $DV$  триггера две последние строки описывают его работу в режиме  $D$  триггера. Условное обозначение  $DV$  триггера приведено на рис. 4.48.

Кроме  $D$  триггера с помощью  $DV$  триггера можно реализовать синхронный и асинхронный варианты  $T$  триггера.

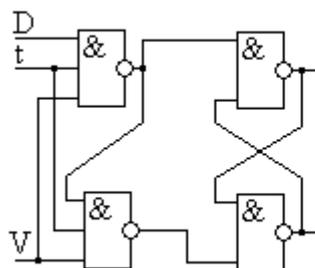


Рис. 4.47. Логическая схема  $DV$  триггера

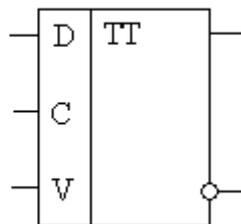


Рис.4.48. Условное обозначение  $DV$  триггера

Таблица переходов для  $DV$  триггера

t		t + 1	
V	D	Q(t)	Режим
0	0	Q(n)	Хранение
0	1	Qt)	Хранение
1	0	0	Установка 0
1	1	1	Установка 1

Асинхронный  $T$  триггер, построенный на основе  $DV$  триггера, изображен на рис. 4.49.

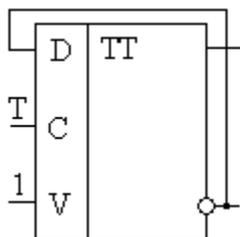


Рис. 4. 49. Построение асинхронного  $T$  триггера на основе  $DV$  триггера

### Вопросы для самопроверки

1. Что называется комбинационной схемой?
2. В чем отличие цифрового автомата от комбинационной схемы?
3. Как строится отмеченная таблица переходов цифрового автомата.
4. Какой автомат называется автоматом с полной системой переходов?
5. Составьте произвольную отмеченную таблицу переходов автомата, обладающего тремя состояниями. Постройте граф работы этого автомата.
6. Нарисуйте условное обозначение схемы совпадений.
7. Нарисуйте диодную логическую схему «И» для отрицательных сигналов.

8. Нарисуйте диодную логическую схему «ИЛИ» для положительных сигналов.

9. Нарисуйте схему инвертора для отрицательных сигналов.

10. Какую логическую функцию будет осуществлять двухступенчатая схема, в первой ступени которой применены две двухвходовые схемы «ИЛИ-НЕ», а во второй – двухвходовая схема «И-НЕ»?

11. Почему необходима синхронизация при передаче информации из одних устройств в другие?

12. Что характеризует коэффициент разветвления логических схем?

13. Что характеризует коэффициент объединения логических схем?

14. Начертите условное графическое обозначение  $RS$  триггера.

15. Начертите схему асинхронного  $RS$  триггера, выполненного на логических элементах, реализующих функцию «ИЛИ-НЕ».

16. Нарисуйте схему синхронного однотактного  $RS$  триггера, выполненного на элементах, реализующих функцию «И-НЕ».

17. Начертите схему  $D$  триггера.

18. Начертите схему  $DV$  триггера.

19. Начертите схему двухтактного  $RS$  триггера.

20. Начертите граф, описывающий работу асинхронного  $RS$  триггера.

21. Имеется диодная схема совпадений для положительных сигналов. Напряжение питания + 5 В. Сигналы на входах изменяются от + 0,2 В (лог. 0) до 3 В (лог 1). Чему равно выходное напряжение при логическом нуле на всех входах?

22. Имеется диодная схема совпадений для положительных сигналов. Напряжение питания + 5 В. Сигналы на входах изменяются от + 0,2 В (лог. 0) до 3 В (лог 1). Чему равно выходное напряжение при логической единице на всех входах?

23. Имеется интегральная микросхема К155ТВ1 (триггер типа  $JK$ ). Используя ее условное графическое обозначение, нарисуйте, как ее надо включить, чтобы получить триггер типа  $T$ ?

24. Опишите различие в схемах однотактных и двухтактных триггеров типа  $RS$ . Опишите, в каких случаях следует использовать триггеры с двухтактной синхронизацией?

25. Нарисуйте структурную схему синхронного двухтактного триггера типа  $JK$ . Составьте для нее таблицу переходов. По таблице переходов составьте аналитическое выражение для  $Q(J,K)$  и нарисуйте временную диаграмму, отражающую зависимость выходных сигналов от сигнала синхронизации и от сигналов на входах  $J$  и  $K$ .

## 5. РЕГИСТРЫ

Регистр является устройством, предназначенным для запоминания и хранения информации. Цифровой код в регистре может храниться сколько угодно долго до прихода нового кода. Если разряд числа выражается цифрой двоичной системы, то для хранения значения этого разряда необходим элемент с двумя устойчивыми состояниями. В современных микросхемах регистров применяют полупроводниковые статические или динамические триггеры. Поэтому при выключении питания хранящаяся в регистре информация пропадает. Условное графическое обозначение регистра приведено на рис. 5.1.

Триггеры регистров могут быть выполнены на полевых или биполярных транзисторах. Регистры выполняют на RS, JK и D триггерах. Триггеры типов RS и JK имеют отдельные входы для управления установкой в каждое из устойчивых состояний. Во многих случаях триггеры имеют несколько вспомогательных устройств, позволяющих выполнять следующие дополнительные операции:

- сброс (установку регистра в нулевое состояние),
- прием кода из другого устройства,
- передачу кода в другое устройство,
- преобразование кода числа,
- сдвиг кода числа вправо, влево, реверсируемый сдвиг,
- преобразование последовательного кода в параллельный,
- преобразование параллельного кода в последовательный.

При приеме информации в регистр могут выполняться поразрядные (логические) операции, такие как:

- логическое сложение,
- логическое умножение,
- суммирование по модулю 2.

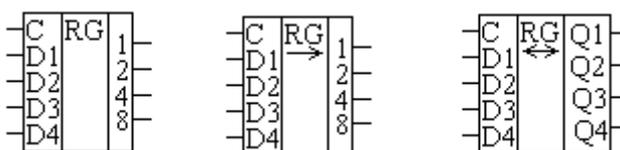


Рис. 5.1. Условное обозначение регистров

Стрелка на условном графическом обозначении указывает направление сдвига. Регистр, в котором возможно переключение направления сдвига (реверсивный), отмечен двунаправленной стрелкой.

В регистрах применяют RS триггеры, JK и D триггеры.

Если информация, поступающая в регистр, записывается по однопроводной схеме, то перед занесением информации необходимо осуще-

ствить предварительное гашение (сброс) регистра, так как сигнал лог 0 при подаче на вход  $S$  (вход установки в единичное состояние), не сбрасывает ранее записанную единицу. Сброс делается по асинхронному входу установки в ноль, обозначаемому  $R$ .

На рис. 5.2 изображен трехразрядный регистр, выполненный на асинхронных  $RS$  триггерах с шиной сброса. Из рисунка видно, что простейший многоразрядный регистр состоит из отдельных одноразрядных устройств хранения информации, не связанных между собой информационными шинами.

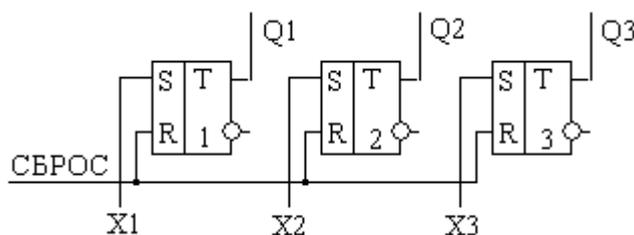


Рис. 5.2. Три разряда регистра с шиной сброса

Информация в такой схеме подается параллельно (одновременно) на входы  $S$  всех триггеров. Для безошибочного приема информации сигналы  $X_1$ ,  $X_2$  и  $X_3$  необходимо задерживать по отношению к сигналу СБРОС на время установления триггеров.

В большинстве практических случаев реализуются не все возможные функции регистров. На рис. 5.3 приведена схема регистра, выполненного на синхронных (синхронизируемых) триггерах, в которой прием информации осуществляется после предварительной установки в 0 при подаче синхронизирующего (тактового) сигнала *запись*.

Для получения на выходе регистра по выбору прямого или обратного кода, к выходам триггеров присоединяются логические схемы И и ИЛИ, как показано на рис. 5.4.

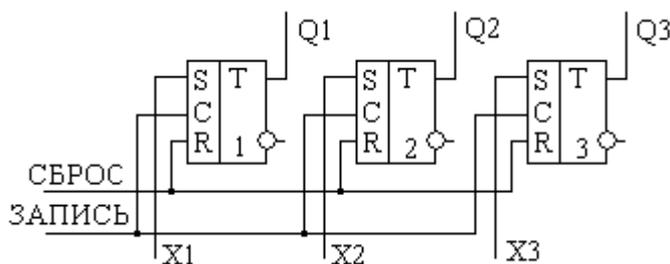


Рис. 5.3. Схема тактируемого регистра

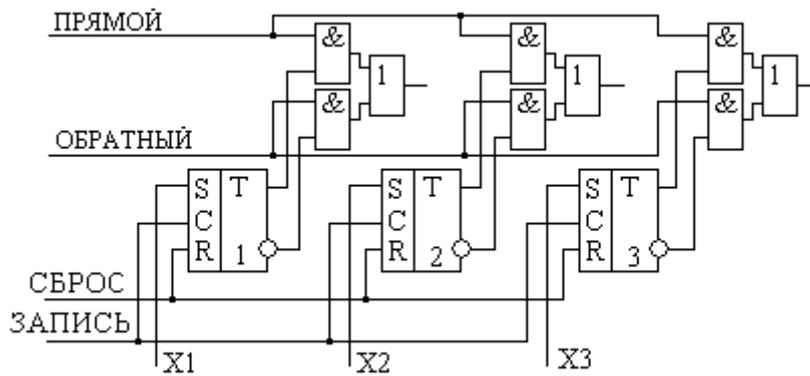


Рис. 5.4. Схема регистра с выдачей прямого или инверсного сигнала

В этом регистре нельзя одновременно подавать логические единицы на входы «ПРЯМОЙ» и «ОБРАТНЫЙ», так как в этом случае на выходе будет код «все единицы». При подаче логического нуля на оба входа выдача кода блокируется, и на всех выходах будут нули.

При передаче информации по двухпроводной схеме на выходах регистра получают парафазный код. На рис. 5.5 показаны разряды регистра, информация с которого снимается в прямом парафазном коде.

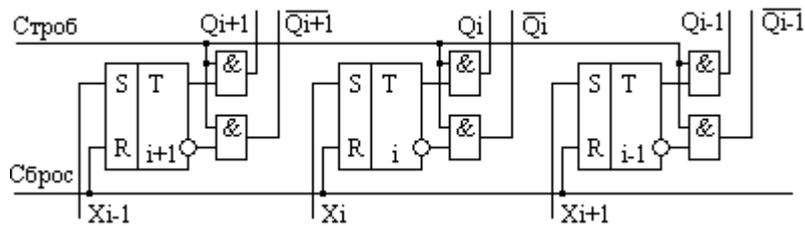


Рис. 5.5. Часть разрядов регистра с парафазной выдачей информации

Чтение информации осуществляется в то время, когда сигнал строб равен единице. Если задание момента чтения не нужно, то схемы совпадения и сигнал СТРОБ можно исключить и снимать парафазный код непосредственно с выходов триггеров.

При применении синхронных триггеров, для осуществления передачи из регистра в регистр парафазным кодом, достаточно непосредственно соединить выходы передающего регистра с входами регистра приемника информации, как показано на рис. 5.6.

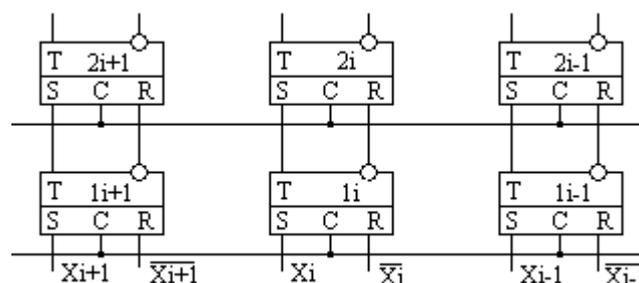


Рис. 5.6. Непосредственная передача парафазного кода из регистра в регистр

Для передачи кода во второй регистр необходимо на триггеры второго регистра подать сигнал синхронизации. В это время сигнал синхронизации на входы первого регистра подаваться не должен.

Интегральные микросхемы регистров имеют в каждом разряде только один выходной контакт. Информация на выходе представляется в прямом однофазном коде. При необходимости получения обратного или парафазного кодов необходимо к выходам регистров подключать дополнительные инверторы.

Осуществление сдвига информации в регистре поясняется с помощью рис. 5.7, на котором изображен сдвигающий регистр, выполненный на одноктактных *RS* триггерах. В каждом разряде такого регистра необходимо применить два триггера, а сдвиг осуществлять посредством подачи двух синхронизирующих сигналов. На рис. 5.7 верхние триггеры являются основными.

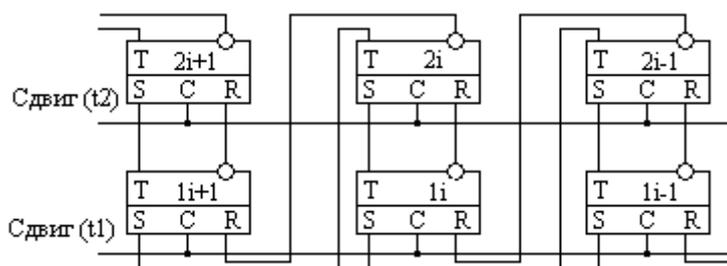


Рис. 5.7. Сдвигающий регистр на парах одноктактных *RS* триггеров

При сдвиге сначала подается импульс, разрешающий прием информации в нижние (по рисунку) триггеры, а затем, после завершения переходных процессов, подается сигнал записи в основные триггеры. Таким образом, код сдвигается на один разряд влево.

Сдвигающие регистры, кроме сдвига кодов, позволяют выполнять преобразование из последовательного кода в параллельный и наоборот. В первом случае код при сдвиге снимается с выхода последнего триггера разряд за разрядом. Во втором случае последовательный код подается на вход первого разряда и сдвигается. При таком последовательно-параллельном преобразовании информация снимается одновременно с выходов всех разрядов.

В зависимости от типа триггеров и способа связи между ними может быть построено несколько различных вариантов сдвигающих регистров. Информацию в последовательном коде можно вводить в регистр старшими или младшими разрядами вперед. Вывод из регистра последовательного кода можно также осуществить старшими или младшими разрядами вперед.

Из рассмотрения рис. 5.7 видно, что построение сдвигающего регистра предпочтительнее делать на *RS* триггерах двухтактного типа, содер-

жащих основной и вспомогательный триггеры<sup>15</sup>. Схема такого регистра приведена на рис. 5.8. В этой схеме при воздействии фронта синхроимпульса код записывается во вспомогательные триггеры, а при воздействии среза синхронизирующего импульса – в основные. Каждый триггер на рис. 5.8 является двухтактным и состоит из двух одноктактных триггеров, о чем говорит обозначение ТТ.

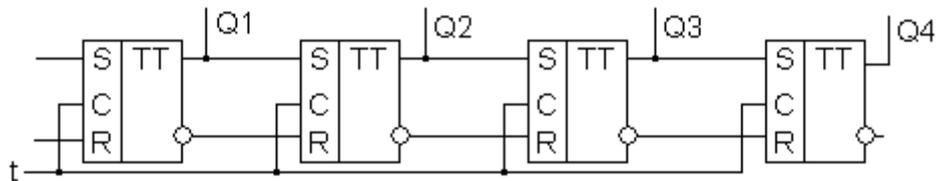


Рис. 5.8. Сдвигающий регистр на двухтактных *RS* триггерах

По такой же схеме составляется сдвигающий регистр на триггерах типа *JK*. Двухтактный сдвигающий регистр, построенный на *JK* триггерах, приведен на рис. 5.9.

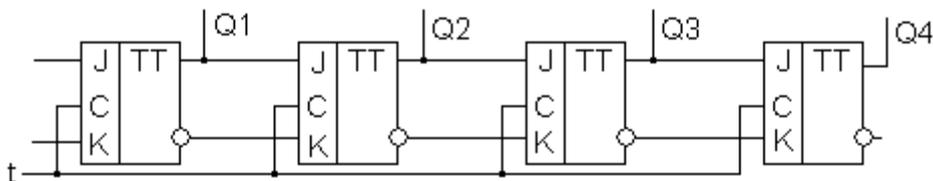


Рис. 5.9. Двухтактный сдвигающий регистр, выполненный на триггерах типа *JK*

Так как триггеры типа *D* имеют один управляющий (информационный) вход, сдвигающий регистр на таких триггерах получается проще. Схема такого регистра приведена на рис. 5.10.

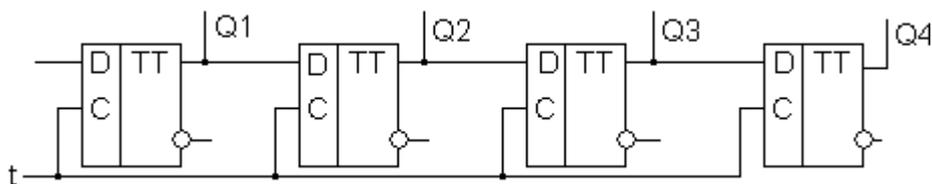


Рис. 5.10. Сдвигающий регистр на *D* триггерах

Наряду с регистрами, в которых сдвиг кода осуществляется в одном направлении, применяются регистры, в которых предусмотрено изменение направления сдвига. Такие регистры называют реверсивными. Для обеспечения возможности переключения между разрядами регистра добавляются логические схемы, как показано на рис. 5.11. Направление

<sup>15</sup> См. также рис. 4.36.

сдвига задается сигналом, подаваемым на вход управления этих логических схем. В реверсивном регистре сигнал на вход  $D_i$  переписывается или из соседнего младшего или из соседнего старшего разряда, в зависимости от значения логического уровня на входе управления  $D_i = Q_{i-1}B + Q_{i+1}\bar{B}$ .

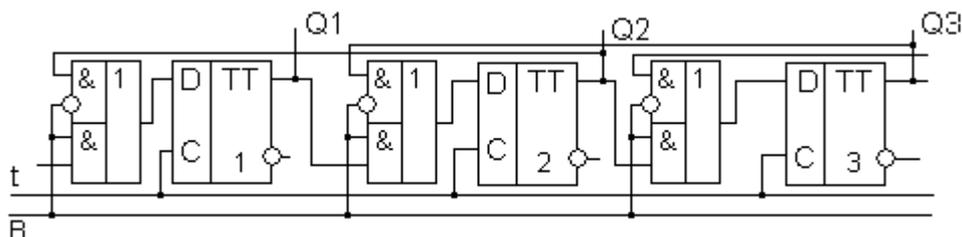


Рис. 5.11. Три разряда реверсивного сдвигающего регистра, выполненного на  $D$  триггерах

В этом регистре при  $B = 1$ , открыты нижние (по рисунку) схемы совпадения и происходит сдвиг кода вправо. При  $B = 0$  открыты верхние схемы совпадения и производится сдвиг кода влево.

Управлять направлением сдвига можно по двум линиям, например сигналами с выходов разных устройств. На рис. 5.12 приведена схема реверсивного сдвигающего регистра на  $JK$  триггерах, в которой применены отдельные входы для подачи управляющих сигналов, определяющих направление сдвига кода.

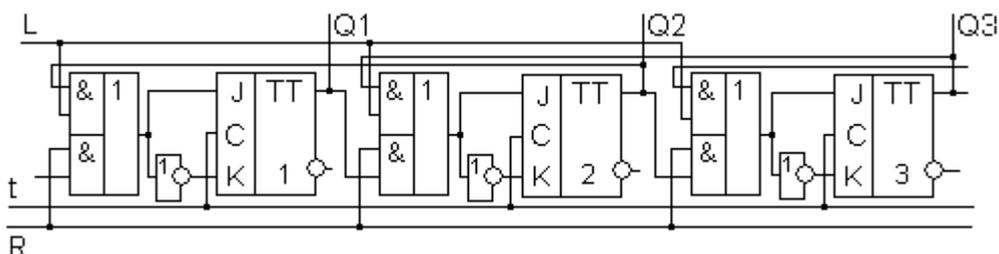


Рис.5.12. Реверсивный сдвигающий регистр с отдельными входами управления направлением сдвига кода

При подаче сигналов  $R = 1, L = 0$  информация сдвигается вправо, а при подаче сигналов  $R = 0, L = 1$  информация сдвигается влево. Если сигналы  $R$  и  $L$  оба равны нулю, сдвиг кода невозможен. Одновременная подача единиц на оба входа управления недопустима, поскольку происходит сбой кода.

При соединении последовательного выхода сдвигающего регистра с его последовательным входом получается циклический регистр сдвига, иначе называемый кольцевым счетчиком или кодовым кольцом. Простейшая схема кодового кольца приведена на рис. 5.13. Это кодовое кольцо называется счетчик Джонсона, а циркулирующий в кольце код – кодом Джонсона.

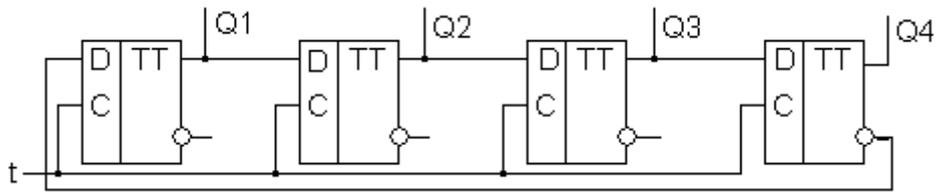


Рис.5.13. Кольцевой счетчик, работающий в коде Джонсона

Такой счетчик (кодовое кольцо) производит пересчет по модулю  $2N$ , где  $N$  – число разрядов счетчика. Последовательность смены состояний разрядов этого кодового кольца приведена в таблице 5.1.

Сдвиг информации в кодовом кольце происходит под воздействием каждого сигнала  $X_{сч}$ .

Кодовое кольцо может быть построено таким образом, что по кольцу будет продвигаться только одна единица. Схема такого кольцевого счетчика изображена на рис. 5.14, а смена состояний приведена в табл. 5.2.

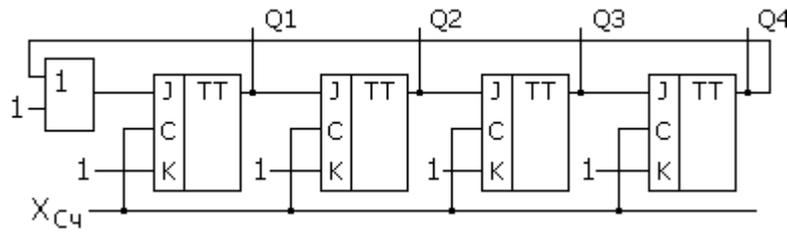


Рис. 5.14. Кодовое кольцо с предварительным занесением единицы

В этом счетчике один из триггеров всегда находится в состоянии 1, а остальные – в состоянии 0. Единица должна быть записана в счетчик предварительно с помощью схемы ИЛИ. На входе триггера, следующего за триггером, находящимся в состоянии 1, действуют сигналы  $J = 1, K = 1$ . Поэтому при подаче импульса на вход синхронизации он переходит из нулевого состояния в единичное. На входах других триггеров действуют сигналы  $J = 0, K = 1$  и они переходят в состояние 0 (или остаются в состоянии 0).

Таблица 5.1

$X_{сч}$	Q4	Q3	Q2	Q1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	1	1	1
4	1	1	1	1
5	1	1	1	0
6	1	1	0	0
7	1	0	0	0

Таблица 5.2

$X_{сч}$	Q4	Q3	Q2	Q1
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0
4	0	0	0	1

На выходе каждого из триггеров единица появляется один раз за четыре такта. При сбое единицы или появлении лишних единиц искаженная информация будет циркулировать по кольцу. Поэтому в кольцевых счетчиках принимают меры по устранению сбоев. Одним из способов коррекции является добавление, как показано на рис. 5.15, логической схемы, разрешающей перепись единицы из последнего триггера в первый, только при условии, что все остальные находятся в состоянии 0. В этой схеме на входе первого триггера сохраняется нулевой сигнал, пока хотя бы один из триггеров  $T1...Tn$  находится в единичном состоянии. Единица на входе первого триггера появляется тогда, когда все остальные триггеры установятся в нулевое состояние.

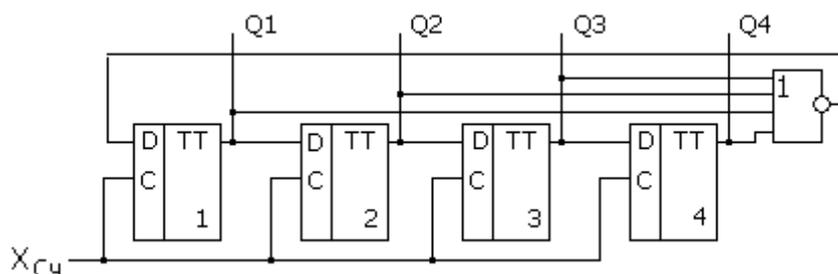


Рис. 5.15. Кодовое кольцо со схемой коррекции

Порядок смены состояний для схемы на рис. 5.15 приведен в табл. 5.3. Из таблицы видно, что благодаря дополнительной логической схеме кодовое кольцо стало иметь 5 различных состояний, включая состояние 0000 и осуществлять пересчет на 5.

Таблица 5.3.

Состояния кодового кольца со схемой коррекции

$X_{Cч}$	Q4	Q2	Q3	Q1	ИЛИ-НЕ
0	0	0	0	0	0
1	1	0	0	0	0
2	0	0	0	0	1
3	0	0	0	1	0
4	0	0	1	0	0
5	0	1	0	0	0

Передача информации из регистра в регистр может сопровождаться выполнением логических операций. Логические операции выполняются поразрядно, поэтому достаточно рассмотреть один разряд в регистре-передатчике и один разряд в регистре-приемнике. Схема, осуществляющая логическое сложение при передаче кода из регистра в регистр, приведена на рис. 5.16.

$$[RG_1] + [RG_2] \rightarrow RG_2$$

Обратите внимание на синтаксис записи операции. В скобках записано содержимое регистров. Знаком « + » отмечено логическое сложение. Стрелка означает «результат операции помещается в».

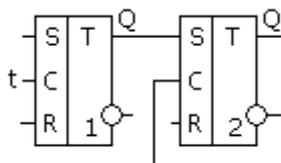


Рис. 5.16. Выполнение логического сложения при передаче кода из регистра в регистр

Информация из  $RG_1$  передается в  $RG_2$  прямым кодом. Предварительный сброс  $RG_2$  в 0 не делается. Схема, приведенная на рис. 5.16, выполнена на синхронных  $RS$  триггерах.

Если применяются асинхронные триггеры, то необходимо ввести в цепи связи схемы совпадения (ключи), как показано на рис. 5.17. Операция при этом выполняется при подаче 1 на управляющий вход ключа.

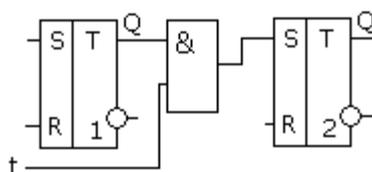


Рис. 5.17. Реализация логического сложения при обмене информацией в асинхронных регистрах

Содержимое регистров при выполнении этой операции приведено в табл. 5.4. В этой таблице  $[RG_{2n}]$  означает содержимое второго регистра в  $n$ -м такте, а  $[RG_{2n+1}]$  – содержимое второго регистра в следующем такте.

Таблица 5.4

Логическое сложение		
$[RG_1]$	$[RG_{2n}]$	$[RG_{2n+1}]$
0	0	0
0	1	1
1	0	1
1	1	1

При осуществлении логического умножения содержимое первого регистра необходимо передать в  $RG_2$  в обратном коде без предварительного гашения  $RG_2$ ,

$$\overline{[RG_1]} \& [RG_2] \rightarrow RG_2. \quad (5.3)$$

Схема такой передачи приведена на рис. 5.18. Ее функционирование описано в табл. 5.5.

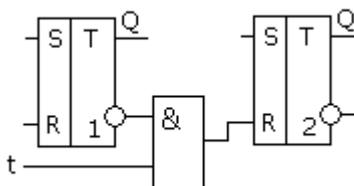


Рис. 5.18. Схема логического умножения

Для осуществления сложения по модулю 2 (поразрядное сложение) триггеры принимающего регистра должны иметь счетный вход, как показано на рис. 5.19. Функционирует эта схема в соответствии с выражением

$$[RG1] \oplus [RG2] \rightarrow RG2 \quad (5.4)$$

и табл. 5.6.

Таблица 5.5

$[RG_1]$	$[RG_{2n}]$	$[RG_{2n+1}]$
0	0	0
0	1	0
1	0	0
1	1	1

Таблица 5.6

$[RG_1]$	$[RG_{2n}]$	$[RG_{2n+1}]$
0	0	0
0	1	1
1	0	1
1	1	0

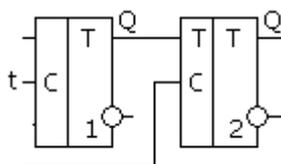


Рис. 5.19. Реализация двоичного сложения

Прямой код из  $RG_1$  подается на счетный вход  $RG_2$  без предварительного гашения  $RG_2$ .

Входные и выходные цепи рассмотренных выше регистров содержат логические схемы, служащие для управления направлением и моментом времени передачи сигнала. Наиболее удобно такие цепи выполнять на элементах логического И. Однако в некоторых сериях микросхем, логические элементы содержат инверторы. При использовании таких схем для получения нужного уровня сигнала приходится применять двухкаскадное соединение элементов И-НЕ (ИЛИ – НЕ). Работа таких схем описана с помощью табл. 5.7.

В первых четырех равенствах даны выражения для логических функций в случае, когда 1 представлена высокими уровнями входного и выходного сигналов.

Таблица 5.7

Двухкаскадное соединение логических схем

Схема	1	0
A	$Y = X_1X_2 + X_3X_4$	$Y = (X_1 + X_2)(X_3 + X_4)$
B	$Y = X_1X_2X_3X_4$	$Y = X_1 + X_2 + X_3 + X_4$
C	$Y = (X_1 + X_2)(X_3 + X_4)$	$Y = X_1X_2 + X_3X_4$
D	$Y = X_1 + X_2 + X_3 + X_4$	$Y = X_1X_2X_3X_4$

Если для этих схем единице сопоставить низкий уровень напряжения, а нулю – высокий, то действие двухкаскадных схем будет описываться следующими четырьмя выражениями. Эти выражения легко проверяются с помощью правил Моргана. Соответствующие схемы представлены на рис. 5.20.

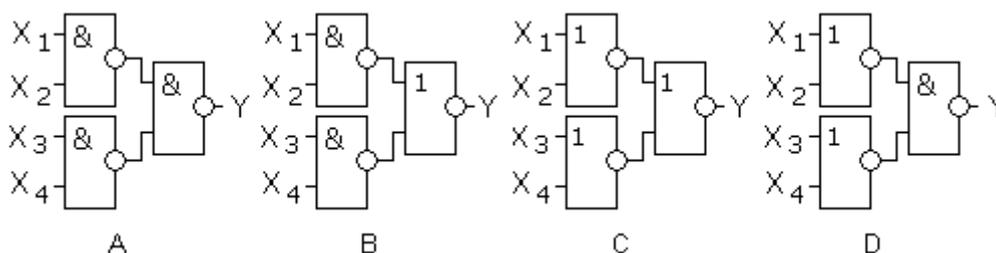


Рис. 5. 20. Соединение логических элементов с инверсией на выходе в соответствии с табл. 5.7

В качестве примера схемы регистра рассмотрим микросхему К155ИР1. Ее условное обозначение приведено на рис. 5.21. Функциональное назначение микросхем ИР1 – четырехразрядный универсальный сдвигающий регистр. Микросхема имеет следующие входы и выходы: V1 – вход для ввода информации в последовательном коде, D1, D2, D3 и D4 – входы разрядов регистра для ввода информации в параллельном коде, V2 – вход выбора режима (подачей сигнала на этот вход задаются направление сдвига и способ ввода кода), C1 – вход синхронизации для сдвига в сторону младших разрядов, контакты 13 (Q1), 12 (Q2), 11 (Q3) и 10 Q(4) – выходы разрядов регистра. Код можно вводить параллельным или последовательным способом.

Если сигнал на входе V2 равен нулю, то под действием сигналов, подаваемых на вход C1, осуществляется ввод последовательного кода, подаваемого на вход V1, и сдвиг этого кода в сторону старших разрядов реги-

стра. Если сигнал на входе  $V2$  равен единице, то под действием сигналов, подаваемых на вход  $C2$ , осуществляется ввод параллельного кода через входы  $D1 - D4$ . В этом режиме можно осуществить сдвиг кода в сторону младших разрядов, если с помощью внешних перемычек соединить выход  $Q2$  с входом  $D1$ ,  $Q3$  с  $D2$  и  $Q4$  с  $D3$ .

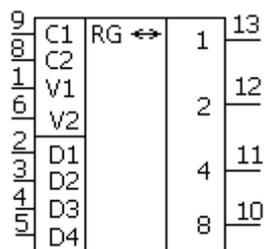


Рис. 5.21. Условное графическое изображение регистра типа ИР1

Длительность сигналов, подаваемых на входы  $C1$  и  $C2$ , должна быть меньше длительности сигналов, подаваемых на входы  $V1$  и  $D2 - D4$ , так как нельзя изменять значения сигналов на информационных входах во время действия синхросигналов. Логическая структура микросхемы К155ИР1 приведена на рис 5.22.

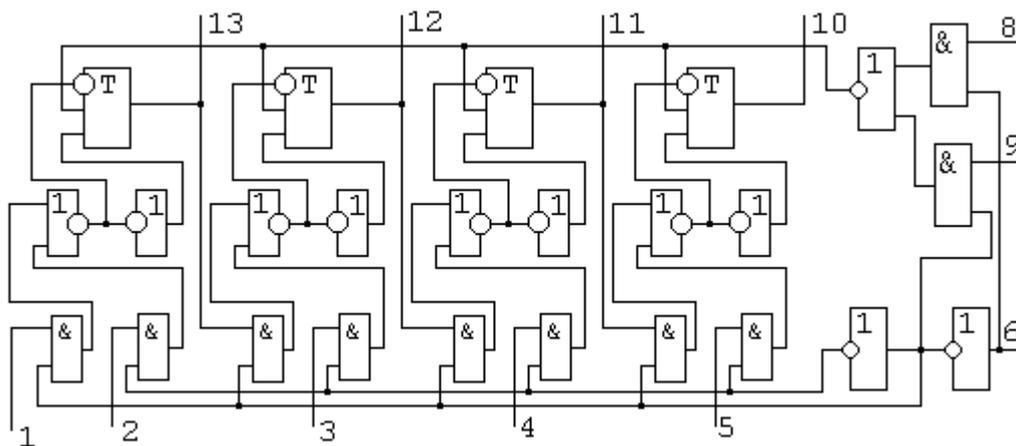


Рис. 5.22. Структура регистра К155ИР1

### Вопросы для самопроверки

1. Возможно ли с помощью регистра преобразовать последовательный код в параллельный?
2. Почему в сдвигающих регистрах применяется двухтактный способ сдвига информации?
3. Можно ли в реверсивном сдвигающем регистре одновременно осуществить сдвиг кода в обе стороны?

4. Какой коэффициент пересчета имеет простейшее кодовое кольцо, в котором вход первого триггера соединяется с инверсным выходом последнего триггера?

5. Можно ли реализовать регистр на  $RS$  триггерах?

6. Можно ли реализовать регистр на  $D$  триггерах?

7. Можно ли реализовать регистр на  $JK$  триггерах?

8. Можно ли реализовать регистр на  $T$  триггерах?

9. Имеется микросхема регистра К155ИР1. На кристалле микросхемы нет мультиплексора для переключения направления сдвига кода. Как настроить регистр для сдвига кода влево?

10. Имеется регистр на 4 синхронных триггерах типа  $RS$ . Выход  $Q$  четвертого триггера соединен с входом  $R$  первого триггера. Инверсный выход четвертого триггера соединен с входом  $S$  первого триггера. Нарисуйте диаграмму переходов этого регистра. Составьте таблицу переходов. Что изменится, если в схему добавить еще один триггер?

11. Можно ли с помощью регистра преобразовать параллельный код в последовательный?

12. Что такое кольцевой счетчик?

13. Как устроен и как работает счетчик для получения кода Джонсона?

14. Как устроена схема коррекции для устранения сбоев в кольцевом счетчике?

15. С помощью какой схемы заносится единица в кольцевой счетчик перед началом его работы?

16. Имеется регистр с прямыми и инверсными выходами в каждом разряде. Какие схемы надо добавить, чтобы с помощью внешнего сигнала получать прямой или обратный код?

## 6. СЧЕТЧИКИ

Счетчиком называется цифровой автомат, выполняющий подсчет единичных сигналов, подаваемых на его вход. При этом в счетчике формируется и запоминается двоичный код числа поступивших сигналов. Таким образом, счетчик осуществляет преобразование числоимпульсного кода в двоичный код. Некоторые разновидности счетчиков могут также выполнять функции приема и выдачи параллельного кода. На электрических схемах счетчики принято изображать, как показано далее, на рис. 6.12.

Счетчики применяются в ЦВМ для образования последовательности адресов оперативной памяти, для счета количества циклов выполнения операций и как делители частоты. Счетчики применяются в системе синхронизации и в составе таймеров. По направлению переходов счетчики принято подразделять на *суммирующие* и *вычитающие*. Счетчик, в котором реализуется микрооперация счета вида  $C := C + 1$ , называется суммирующим. Вычитающим называется счетчик, реализующий микрооперацию  $C := C - 1$ . Счетчик, в котором можно реализовать переключение с одной микрооперации на другую, называется *реверсивным*.

Счет может быть организован асинхронным или синхронным способом.

Рассмотрим суммирующие счетчики. Примером асинхронного суммирующего счетчика является приведенный на рис.6.1 счетчик на *JK* триггерах,

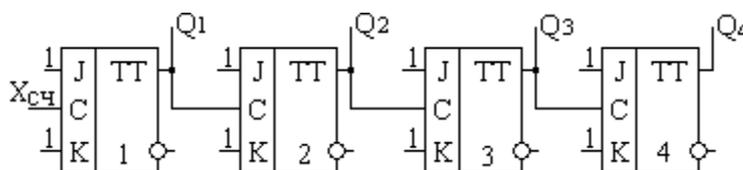


Рис. 6.1. Структура асинхронного суммирующего счетчика

Как следует из рисунка, *JK* триггеры включены в режиме асинхронного *T* триггера. Каждый разряд счетчика при этом делит частоту поступающего сигнала на 2.

Первым триггером счетчика является триггер младшего разряда, четвертый – старшего. Таблица состояний 4-разрядного счетчика имеет 16 строк в соответствии с количеством состояний этого автомата.

На входы *J* и *K* триггеров постоянно подаются сигналы, имеющие уровень логической единицы. Сигнал с выхода *Q* подается на вход синхронизации следующего триггера. Триггер *i*-го разряда переходит из нулевого состояния в единичное, если подан входной сигнал и все триггеры с номерами меньше *i* находятся в состоянии 1, Это записывается в виде следующего условия:

$$X_{сч} \& Q_1 \& Q_2 \& \dots \& Q_{i-1} = 1. \quad (6.1)$$

Временная диаграмма работы счетчика приведена на рис. 6.2. Нетрудно видеть, что каждый разряд счетчика понижает частоту сигнала в два раза. Рассматриваемый счетчик является счетчиком с последовательным переносом.

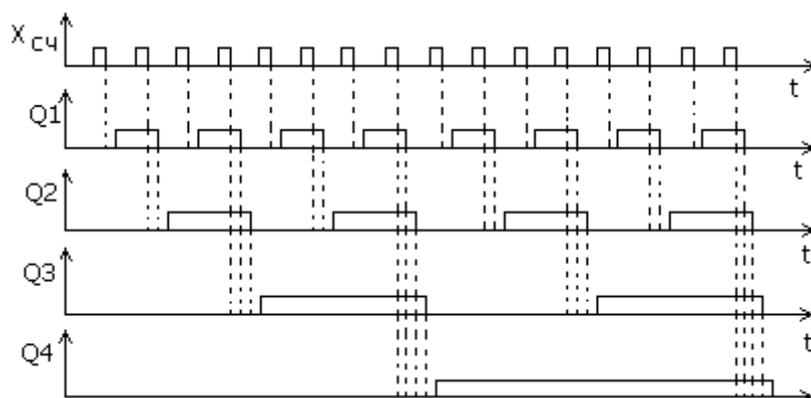


Рис 6.2. Временная диаграмма работы счетчика (показывает нарастание времени задержки переключения по мере увеличения числа разрядов)

В асинхронном счетчике длительность переходного процесса зависит от количества разрядов. Так как каждый триггер задерживает сигнал на некоторое конечное время (рис. 6.2), то с увеличением числа разрядов счетчика, будет возрастать задержка появления сигнала на выходе по отношению к моменту времени поступления сигнала на первый триггер. При большом числе разрядов и высокой частоте следования входных сигналов работа такого счетчика на дешифрирующие схемы осложняется, так как управляющий сигнал "строб дешифрации" должен задерживаться относительно момента поступления очередного импульса на время, не меньшее максимальной длительности переходного процесса для всего счетчика.

На временной диаграмме штриховыми линиями отмечено запаздывание изменения состояния триггеров относительно моментов изменения состояния триггеров младших разрядов и относительно входного сигнала. Время включения  $N$  разрядов последовательного счетчика

$$\Theta_{Nn} = \Theta + N\Theta_T, \quad (6.2)$$

где  $\Theta$  – время переключения сигнала,  $\Theta_T$  – время переключения триггера.

Последовательность смены состояний двоичного суммирующего счетчика приведена в табл. 6.1.

Смена состояний суммирующего счетчика

Хсч	Q4	Q3	Q2	Q1	Хсч	Q4	Q3	Q2	Q1
0	0	0	0	0	8	1	0	0	0
1	0	0	0	1	9	1	0	0	1
2	0	0	1	0	10	1	0	1	0
3	0	0	1	1	11	1	0	1	1
4	0	1	0	0	12	1	1	0	0
5	0	1	0	1	13	1	1	0	1
6	0	1	1	0	14	1	1	1	0
7	0	1	1	1	15	1	1	1	1

Для сокращения времени переходного процесса в счетчиках добавляют специальные цепи для сигналов переноса. На рис. 6.3 и рис. 6.4 приведены схемы синхронных двоичных счетчиков, имеющие схемы переноса, называемого сквозным. В таких счетчиках распространение переноса, наряду с цепочкой триггеров, происходит и по обходным логически цепям.

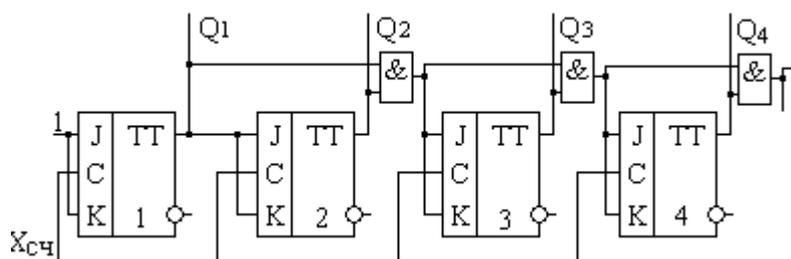


Рис. 6.3. Синхронный двоичный счетчик со сквозным переносом на  $JK$  триггерах

Переключение каждого триггера (рис. 6.3) возможно, если на его информационных входах  $J$  и  $K$  имеется сигнал 1. На входы первого триггера подан сигнал «константа 1». Поэтому триггер первого разряда работает как асинхронный.

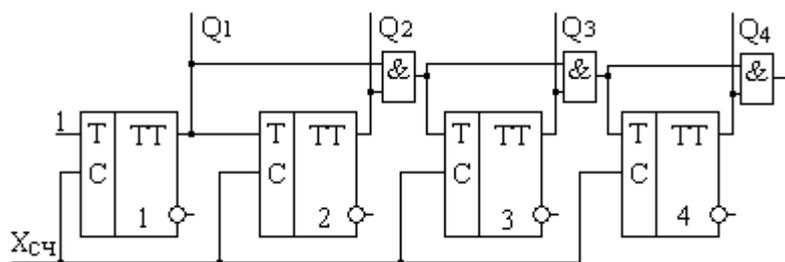


Рис. 6.4. Синхронный счетчик со сквозным переносом, выполненный на триггерах типа  $T$

При переключении триггеров из состояния 1 в состояние 0 соответствующая схема И закрывается. Соответственно, при переключении триггера в состояние 1, эта схема открывается. Сигнал переноса, задерживаясь на каждом клапане И, распространяется по цепи переноса.

Длительность переходного процесса определяется временем задержки сигнала в клапанах И цепей переноса. Для распространения сигнала переноса после момента подачи сигнала необходимо время

$$\Theta_{Nc} = \Theta + \Theta_T(N - 1)t_3, \quad (6.3)$$

где  $t_3$  – время задержки сигнала на схеме И.

Время, определяемое по формуле (6.3), меньше чем время для последовательного счетчика (6.2) на величину

$$(N - 1) \cdot (\Theta_T - t_3). \quad (6.4)$$

В счетчиках со сквозным переносом при распространении сигналов по параллельным цепям (через триггеры и через клапаны И) возможно появление ложных сигналов на выходах. Такой процесс называется *гонками*, или состязаниями. Состязания приводят к ограничению числа разрядов, охваченных цепями сквозного переноса. Характер распространения сигнала в рассматриваемом счетчике остается последовательным. Сигнал переноса проходит последовательно через все обходные цепи переноса. Выигрыш во времени получается за счет того, что задержка в клапане И в несколько раз меньше, чем в триггере.

Для схем, изображенных на рис. 6.3 и 6.4, условие переноса на вход второго разряда

$$P_2 = Q_1, \quad (6.5)$$

где  $Q_1$  – состояние (и выходной сигнал) триггера первого разряда счетчика.

Условие появления сигнала переноса на входе третьего разряда счетчика

$$P_3 = Q_1 Q_2. \quad (6.6)$$

Значение  $Q_2$ , равное единице, разрешает распространение переноса по обходным цепям. Это условие называется условием *транзита*. Обозначим транзит мимо второго разряда  $T_2$ , и учитывая, что

$$T_2 = Q_2 = 1, \quad (6.7)$$

запишем

$$P_3 = P_2 T_2.$$

Запишем условие поступления переноса на вход четвертого разряда

$$P_4 = P_3 T_3 = P_2 T_2 T_3 = Q_1 Q_2 Q_3, \quad (6.8)$$

Для произвольного номера триггера условие поступления сигнала на его вход (формирование сигнала на выходе предыдущего каскада) запишется как:

$$P_{i+1} = P_i T_i = P_2 T_2 T_3 \dots T_i. \quad (6.9)$$

Таким образом,  $P_{i+1}$  есть конъюнкция выходных сигналов всех предыдущих разрядов.

Если последнее выражение переписать в виде

$$P_{i+1} = (\dots((P_2 T_2) T_3) \dots) T_i, \quad (6.10)$$

то перенос реализуется с помощью двухвходовых схем совпадения, как показано на рис. 6.3 и 6.4. Логическую схему переноса можно синтезировать непосредственно по выражению (6.9). В этом случае перенос в счетчике осуществляется параллельным (одновременным) способом. При этом на входах клапанов И, управляющих переносом, действуют одновременно сигналы всех предыдущих разрядов.

Счетчики с такой организацией переноса называются счетчиками с одновременным (параллельным) переносом (рис. 6.5). Схема с одновременным переносом имеет наиболее высокое быстродействие.

Длительность переходного процесса в счетчике с одновременным переносом равна длительности переходного процесса в одном разряде счетчика

$$\Theta_{\text{Нсч}} = \Theta + \Theta_{\text{T}}.$$

Счетчик с одновременным переносом может быть реализован с помощью многовходовых  $JK$  триггеров. Схема такого счетчика приведена на рис. 6.6. В этой схеме выходы всех разрядов с номерами от 1 до  $i-1$  соединяются с  $J$  и  $K$  входами разряда с номером  $i$ .

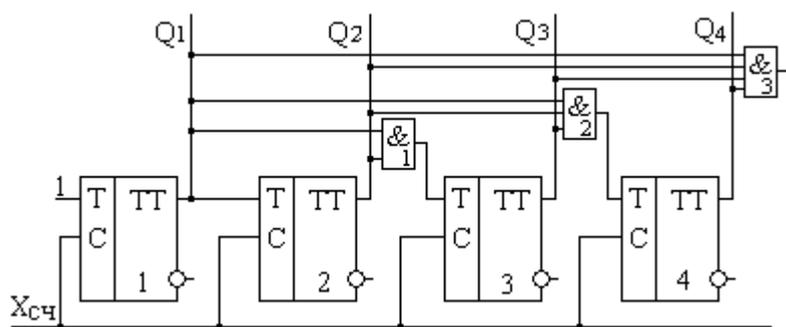


Рис. 6.5. Счетчик с одновременным переносом на Т триггерах

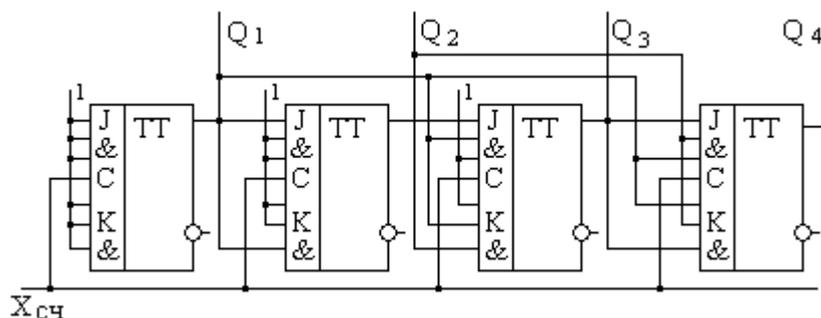


Рис. 6.6. Счетчик с одновременным переносом на  $JK$  триггерах, имеющих схемы И на  $J$  и  $K$  входах

Если в счетчике, изображенном на рис. 6.6, используются триггеры без функции конъюнкции на входах  $J$  и  $K$ , то цепи переноса надо выполнить, как показано на рис. 6.5, с помощью отдельных схем совпадения.

Так как число входов схем И (или входов  $J$  и входов  $K$  в  $JK$  триггерах) и нагрузочная способность триггеров ограничена, то разрядность счетчиков с параллельным переносом невелика.

При числе разрядов счетчика более чем на один превышающем максимальное число входов  $J$  и  $K$ , счетчик разделяют на группы разрядов. Внутри групп осуществляется одновременный перенос. Перенос между группами, возможно осуществить любым способом, в зависимости от требуемого быстродействия счетчика. Такая организация переноса<sup>16</sup> называется групповой. На рис 6.7 изображена схема счетчика, в котором между группами осуществляется сквозной перенос.

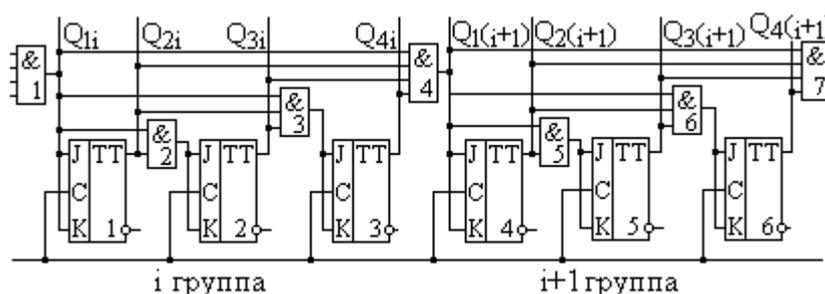


Рис. 6.7. Счетчик с групповым переносом (между группами сквозной перенос).

Наряду со счетчиками, имеющими коэффициент пересчета, равный степени цифры 2, находят применение счетчики с другими коэффициентами пересчета. Если необходимо создать счетчик по модулю  $M$  для  $M$ , не являющегося степенью цифры 2, то есть,

$$2^{n-1} < M < 2^n,$$

то такой счетчик должен содержать не менее  $n$  разрядов. При этом  $m = 2^n - M$  состояний счетчика будут "лишними" (запрещенными) для счетчика по модулю  $M$ . Поэтому счетчик должен быть разработан таким образом, чтобы в процессе работы не могли возникнуть запрещенные состояния.

При разработке схем счетчиков можно использовать методику структурного синтеза конечных автоматов. Структурный синтез подразумевает составление структурной схемы автомата на основе функций автомата, которые необходимо реализовать.

Структурный синтез заключается в выборе типов элементарных автоматов, составлении функций возбуждения каждого автомата и функций кодирования выходов заданного автомата.

<sup>16</sup> Кроме этого раздела перенос рассматривается при изучении сумматоров.

Для исключения избыточных состояний можно воспользоваться введением обратных связей. Вторым способом построения счетчиков с модулем счета  $M \neq 2^n$  заключается в использовании в качестве строительных кирпичиков счетчиков по модулю  $2^{n+1}$ . Эти счетчики позволяют увеличивать модуль счета на единицу. Третьим способом является проектирование счетчиков на основе таблиц переходов. Таблица переходов записывается для всех состояний счетчика. То есть количество строк таблицы равно числу состояний. В первую строку таблицы записывается начальное состояние. Затем записывается состояние, в которое должен перейти счетчик при подаче на вход следующего сигнала. В таблице должны быть указаны текущие и последующие состояния всех триггеров. По таблице переходов для выбранного типа триггеров определяются уровни сигналов, которые необходимо подавать на входы разрядов. Для выполнения упражнений с различными модулями счета в приложении 6 приведена пошаговая инструкция.

Если имеются микросхемы счетчиков с входами сброса, то можно синтезировать счетчик, в котором исключение избыточных состояний осуществляется подачей выходных сигналов на вход сброса.

При необходимости работать с десятичными числами, то есть, получения коэффициента деления счетчика, кратного 10, применяют десятичные (двоично-десятичные) счетчики. Каскады таких десятичных счетчиков выполняют на двоичных элементах. Поскольку двоично-десятичный код составляется из четырех двоичных цифр, то разряд десятичного счетчика содержит 4 триггера. Счетчик, выполненный на четырех триггерах, имеет 16 состояний. При десятичном счете состояния  $A, B, C, D, E$  и  $F$  являются избыточными. Поэтому в десятичные счетчики вводят цепи, исключающие эти 6 избыточных состояний (метод введения обратных связей и управляющих вентилях). На рис. 6.8 изображен один десятичный разряд суммирующего синхронного двоично-десятичного счетчика, работающего в коде 8421. Исключение избыточных состояний выполнено с помощью цепи обратной связи, охватывающей три старших разряда. Сигнал обратной связи снимается с инверсного выхода счетчика.

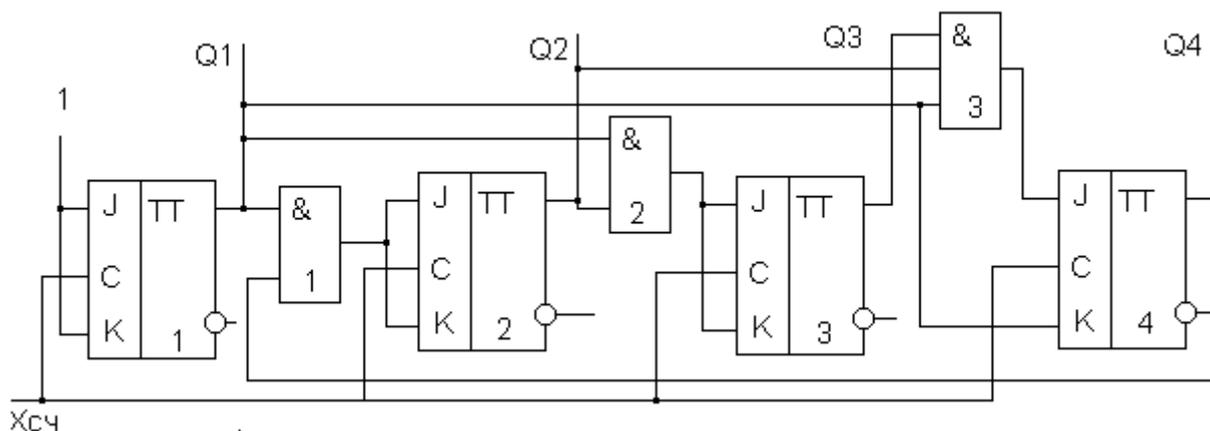


Рис. 6.8. Двоично-десятичный счетчик

В этом счетчике счет первых восьми импульсов происходит как в счетчиках, рассмотренных выше, так как четвертый триггер находится в состоянии 0 и сигнал с его инверсного выхода поддерживает первую схему совпадений в открытом состоянии. Когда в счетчике записан код 111 на входы  $J$  и  $K$  четвертого триггера поданы единичные сигналы. Поэтому при подаче восьмого  $X_{сч}$  четвертый триггер переключается в состояние 1. Так как переключение четвертого триггера происходит под действием сигнала  $X_{сч}$ , то он переключается до того, как сбрасываются первые три триггера. После сброса первых трех триггеров счетчик находится в состоянии 1000. В этом случае сигнал с инверсного выхода четвертого триггера не поддерживает первую схему совпадений в открытом состоянии и переключение второго и третьего триггеров становится невозможным. При состоянии счетчика 1000 на обоих входах четвертого триггера присутствуют нулевые сигналы, и он находится в режиме хранения. Поэтому девятый импульс  $X_{сч}$  не изменяет состояние четвертого триггера, он только переключает первый триггер. Счетчик устанавливается в состояние 1001. В этом состоянии на входе  $J$  четвертого триггера присутствует сигнал 0, поступающий со второй схемы совпадений, а на входе  $K$  – сигнал 1, поступающий непосредственно с выхода первого разряда. Поэтому десятый импульс  $X_{сч}$  переключает  $T4$  в 0. Одновременно в 0 переключается первый триггер. Счетчик устанавливается в исходное состояние 0000.

Вычитающие счетчики отличаются от суммирующих межкаскадными соединениями. Сигнал на вход  $i$ -го триггера подается с инверсного выхода  $i-1$ -го триггера. На рис. 6.9 представлена схема двоичного вычитающего счетчика. Счетчик построен по асинхронной схеме с последовательным переносом. Пересчитываемые импульсы подаются на вход синхронизации первого триггера. Счетчик выполнен на  $JK$  триггерах, имеющих входы асинхронной установки в состояние 1, обозначенные  $S$  и входы для установки триггеров в 0 (сброса), обозначенные  $R$ . Счетчик имеет управляющий вход  $B$ , позволяющий сигналом 1 разрешать счет, а сигналом 0 запрещать передачу сигнала в следующий разряд (блокировка). Последовательность состояний счетчика приведена в табл. 6.2.

В вычитающем счетчике, при переходе  $i$ -го триггера из состояния 0 в состояние 1, триггер с номером  $i + 1$  устанавливается в 0, если он находился в состоянии 1 (строка табл. 6.2, соответствующая третьему импульсу  $X_{сч}$ ). Таким образом, код в счетчике уменьшается на единицу.

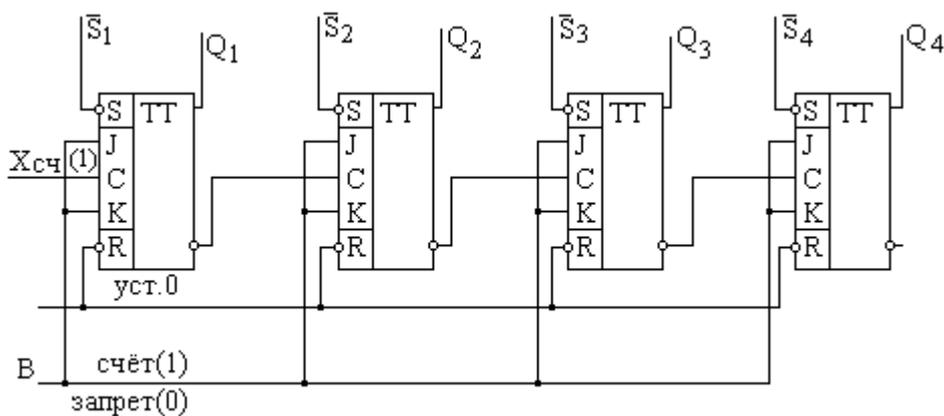


Рис. 6.9. Вычитающий счетчик на *JK* триггерах

Таблица 6.2

Последовательность смены состояний вычитающего счетчика

Xсч	Q4	Q3	Q2	Q1	Xсч	Q4	Q3	Q2	Q1
0	0	0	0	0	8	1	0	0	0
1	1	1	1	1	9	0	1	1	1
2	1	1	1	0	10	0	1	1	0
3	1	1	0	1	11	0	1	0	1
4	1	1	0	0	12	0	1	0	0
5	1	0	1	1	13	0	0	1	1
6	1	0	1	0	14	0	0	1	0
7	1	0	0	1	15	0	0	0	1
16	0	0	0	0					

Реверсивный счетчик имеет цепи управления, служащие для переключения направления счета. Представленный на рис 6.10 счетчик является двоичным реверсивным асинхронным счетчиком. Каждый триггер счетчика имеет входы предварительной установки - *S*, предназначенные для приема параллельного кода. Выбор операции (суммирование – вычитание) определяется значениями сигналов на входах «*C-1*» и «*C + 1*». При сочетании "*C-1*" = 1, "*C + 1*" = 0 инверсный сигнал с *i*-го триггера подается на вход синхронизации *i + 1*-го триггера. Это режим вычитающего счетчика

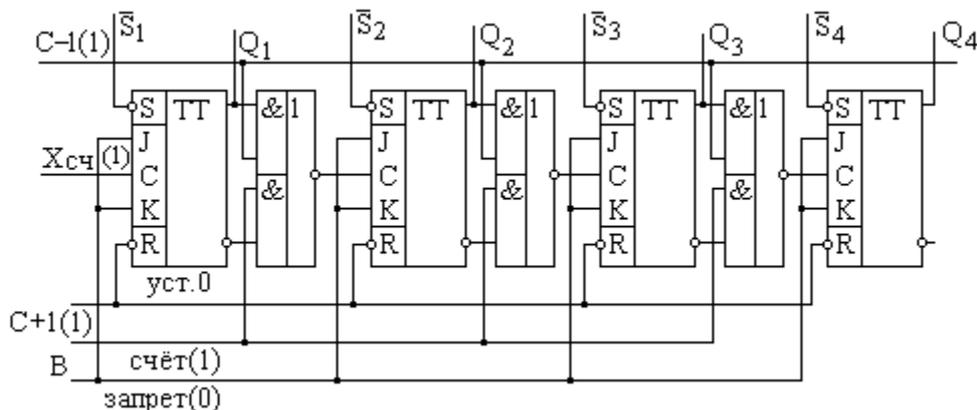


Рис 6.10. Схема реверсивного счетчика

В виде интегральных микросхем выпускаются счетчики различного назначения и производительности. На рис. 6.11 и 6.12 приведены схема и условное обозначение микросхемы К155ИЕ5.

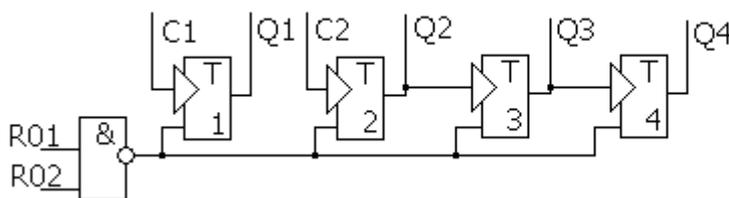


Рис. 6.11. Структурная схема счетчика К155ИЕ5

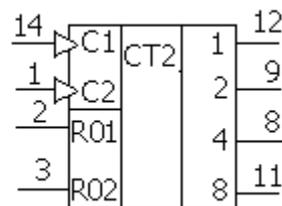


Рис. 6.12. Условное обозначение счетчика К155ИЕ5

У этого счетчика выход триггера младшего разряда не соединен с входом следующего (второго) разряда. Вход второго триггера соединен с внешним контактом. Кроме того, имеется общая цепь установки триггеров в 0, управляемая двухвходовой схемой И-НЕ.

Наличие отдельного триггера позволяет с помощью внешних соединений создавать как четырехразрядный счетчик, так и отдельно одноразрядный и трехразрядный счетчики. Двухвходовая цепь сброса обеспечивает получение различных коэффициентов деления. Сброс триггеров осуществляется сигналом, имеющим уровень логического нуля на входе схемы И-НЕ. Таким образом, сочетания сигналов 00, 01 и 10 на входах  $R_{01}$  и  $R_{02}$  не устанавливают разряды счетчика в 0.

Рассмотрим несколько примеров получения различных коэффициентов деления. Соединим выход  $Q_1$  с входом второго триггера, замыкая контакты 12 и 1 ( $Q_1$  и  $C_2$ ). При этом получается схема, приведенная на рис. 6.1, позволяющая получить на выходах 12, 9, 8 и 11 частоты следования импульсов, соответственно, в 2, 4, 8 и 16 раз более низкие, чем на контакте 14. Такой же результат будет получен, если соединить контакты 11 и 14. Указанные выше соотношения частот будут наблюдаться на контактах 9, 8, 11 и 12.

Для деления частоты в три раза необходимо выполнить соединения в соответствии с рис. 6.13.

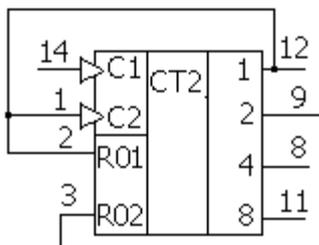


Рис. 6.13. Делитель с коэффициентом, равным 3

Временная диаграмма работы этой схемы представлена на рис. 6.14. Для составления такой диаграммы необходимо знать, по фронту или по срезу импульса изменяют свое состояние триггеры, применяемые в счетчике. В рассматриваемом счетчике (К155ИЕ5) изменение состояния триггера происходит в момент среза входного импульса. На рис. 6.14 на эюре входных импульсов проставлены порядковые номера этих импульсов. На приводимых на рисунке эюрах с помощью нулей и единиц отмечены состояния счетчика (см. табл. 6.1). На этом рисунке и двух следующих диаграммах проставлены номера контактов (12 и 9), на которых наблюдаются эти сигналы.

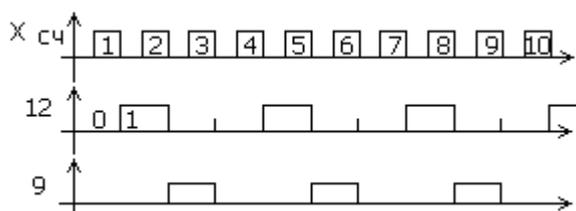


Рис. 6.14. Диаграмма работы делителя на 3

Из рис. 6.1 и рис. 6.14 следует, что в исходном состоянии на входах  $R_{01}$  и  $R_{02}$  нулевой уровень. При поступлении первого импульса счетчик устанавливается в состояние «1» (0001) и на входах  $C_2$  и  $R_{01}$  наблюдается уровень «1». При поступлении второго импульса  $X_{сч}$  первый триггер изменяет свое состояние, что приводит к изменению состояния второго триггера. Счетчик устанавливается в состояние 2 (0010), на входах  $C_2$  и  $R_{01}$  устанавливается уровень «0», на входе  $R_{02}$  – уровень «1». Поступление третьего импульса  $X_{сч}$  переключает первый триггер, и на входе  $R_{01}$  с задержкой относительно среза входного импульса появляется сигнал «1». Таким образом на входах схемы И-НЕ одновременно наблюдаются единичные сигналы, что приводит к установлению разрядов счетчика в «0» через время, равное задержке распространения сигнала через схему И-НЕ и время сброса триггеров. Для микросхем, выполненных на транзисторно-транзисторных элементах, время задержки распространения сигнала  $t_3$  приблизительно 10 нс, время включения триггера  $t_{вк} = 40$  нс, а время выключения (сброса)  $t_{сб}$  составляет 25 нс. В итоге счетчик устанавливается в 0 через 75 нс после среза входного импульса, а сигнал на контакте 9 существует не более 35 нс. При наблюдении входных импульсов с помощью осциллографа этот сигнал может быть не обнаружен из-за своей малой длительности. На приведенных диаграммах он обозначен штрихом. Из диаграммы видно, что частота импульсов, наблюдаемых на контактах 9 и 12, в три раза ниже, чем частота импульсов на входе. Длительность выходных импульсов равна периоду следования входных импульсов –  $T$ , а

импульсы на выходе  $Q_2$  сдвинуты на время  $T$  относительно импульсов на выходе  $Q_1$ . Аналогичным образом, соединяя выходные зажимы с входами  $R_{01}$  и  $R_{02}$  в других сочетаниях, можно осуществить деление на 5, 6, 9, 10 и 12. Используя первый триггер отдельно, можно только с помощью триггеров 2, 3 и 4 получить деление в 3, 5 и 6 раз.

Для осуществления деления на 7 необходимо 12-й контакт соединить с 1 и, кроме того, сигналы с выходов 12, 9 и 8 подать на входы 2 и 3 ( $R_{01}$ ,  $R_{02}$ ). Так как схема совпадения в цепи сброса имеет только два входа, непосредственное соединение указанных контактов невозможно. Однако можно воспользоваться двухвходовой схемой И-НЕ. Пример счетчика-делителя на 7 приведен на рис. 6.15, а диаграмма его работы на рис. 6.16. Аналогичным образом реализуются схемы для деления на 11, 13 и 14. Используя первый триггер отдельно, можно с помощью триггеров 2, 3 и 4 также получить деление на 7.

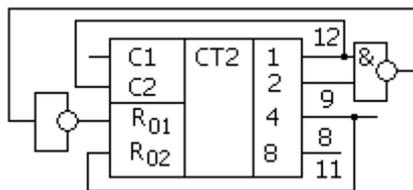


Рис. 6.15. Пример схемы соединений для деления на 7

Для того чтобы построить счетчик с коэффициентом деления 15, необходимо использовать сигналы со всех четырех выходов счетчика. Схема счетчика-делителя на 15 приведена на рис. 6.17, а временные диаграммы – на рис. 6.18. В этом случае придется использовать 4 схемы И-НЕ.

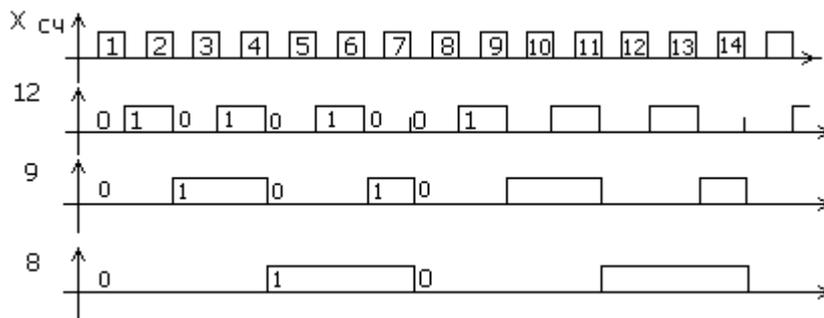


Рис. 6.16. Диаграмма смены состояний счетчика при делении на 7

Если четырехразрядный счетчик образован соединением 11 и 14 контактов, то можно получить еще 11 вариантов построения счетчиков с различными коэффициентами пересчета.

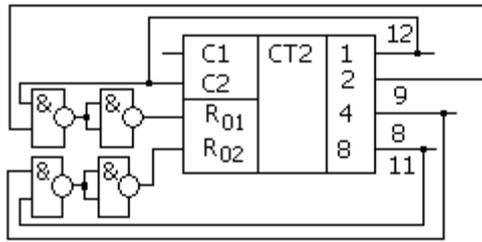


Рис. 6.17. Счетчик с коэффициентом деления равным 15

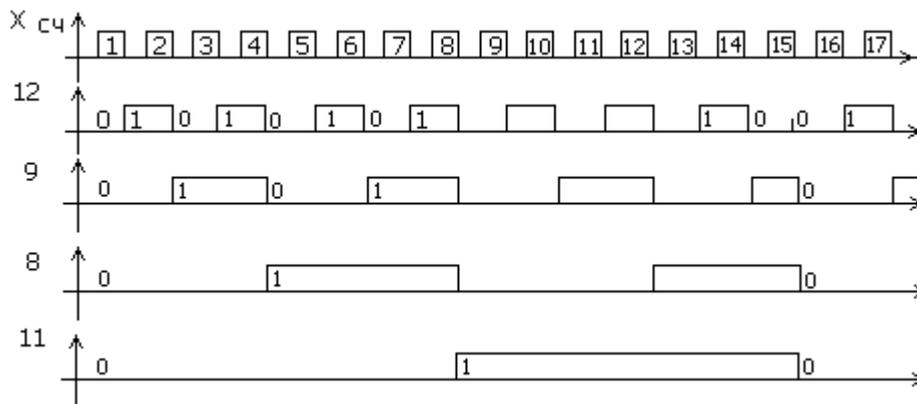


Рис. 6.18. Диаграмма состояний счетчика по модулю 15

Принципиальных отличий от рассмотренных ранее вариантов они не имеют, однако могут оказаться иногда более целесообразными с точки зрения расположения проводников печатного монтажа. Вместо двухвходовых схем И-НЕ могут быть применены с незначительными отличиями трех- и четырехвходовые схемы И-НЕ либо схемы совпадения<sup>17</sup>.

Оба входа  $R_{01}$  и  $R_{02}$  можно подключить к выводу одной логической схемы. Если эти входы не используются, их необходимо соединить (для микросхем транзисторно-транзисторной логики) с общим проводом.

### Вопросы для самопроверки

1. Какие типы триггеров можно использовать в счетчиках?
2. Какое преобразование кода осуществляет двоичный счетчик?
3. Какие способы ускорения сигнала переноса применяются в двоичных счетчиках?
4. Что называют модулем счета?
5. Почему обходные цепи для распространения сигнала переноса не могут иметь большую длину?
6. Чем отличается вычитающий счетчик от суммирующего?
7. В чем заключается эффект гонок? К чему он может привести?

<sup>17</sup> Читателям рекомендуется самостоятельно проделать эксперименты с получением различных коэффициента пересчета.

8. Чем отличается реверсивный счетчик от суммирующего и вычитающего счетчиков?

9. Какое устройство включается между разрядами реверсивного счетчика?

10. Можно ли построить на двоичных элементах счетчик с модулем счета, отличным от степени цифры 2?

11. Сколько нужно двоичных разрядов для построения счетчика с модулем счета, равным 10?

12. Какой модуль счета имеет четырехразрядный двоичный счетчик, построенный на четырех триггерах?

13. Какие элементы надо добавить в двоичный счетчик, чтобы получить счетчик с модулем 10?

14. Входы какого назначения встречаются в счетчиках кроме входа для подсчитываемых сигналов и входа для подачи сигнала синхронизации?

## 7. СУММАТОРЫ

### 7.1. Техническая реализация сложения двоичных кодов

При суммировании кодов многоразрядных чисел в каждом разряде, кроме младшего, происходит суммирование трех цифр: цифры первого слагаемого, цифры второго слагаемого и единицы переноса, поступившей из соседнего младшего разряда числа. В результате сложения этих цифр на выходе разряда получается цифра суммы  $S_i$  и при некоторых сочетаниях слагаемых, единица переноса в соседний, старший,  $i + 1$ -й разряд, обозначаемый далее  $P_{i+1}$ . Обозначим цифры слагаемых в  $i$ -м разряде  $a_i$  и  $b_i$ . Перенос, возникающий за счет сложения  $a_i$  и  $b_i$ , обозначим  $c_i$ . Так как перенос возникает при сложении двух единиц, то  $c_i = a_i \& b_i$ . Этот перенос называют *собственным*. Кроме собственного переноса перенос образуется при поступлении единицы переноса из младшего  $i-1$ -го разряда в  $i$ -й разряд и наличия хотя бы одного слагаемого ( $a_i$  или  $b_i$ ), равного единице. Перенос, поступающий на вход  $i$ -го разряда из  $i-1$ -го, обозначим  $p_i$ . Условие распространения переноса через  $i$ -й разряд обозначают  $T_i$  и называют условием *транзита*

$$T_i = a_i + b_i.$$

Перенос из  $i$ -го разряда в  $i + 1$  разряд составляется из собственного и транзитного переносов

$$P_{i+1} = c_i + p_i T_i = a_i b_i + p_i (a_i + b_i).$$

Соответствие выходных и входных значений для одного разряда двоичного сумматора приведено в табл. 7.1 (таблице истинности).

В этой таблице введены два дополнительных столбца для указания того, что выполнен собственный перенос  $c_i$  и для указания того, что выполнено условие транзита  $T_i$ . На основе таблицы запишем логические формулы, описывающие образование суммы и переноса.

Таблица 7.1.

Таблица истинности одноразрядного двоичного сумматора

$p_i$	$a_i$	$b_i$	$S_i$	$P_{i+1}$	$c_i$	$T_i$
0	0	0	0	0	0	0
0	0	1	1	0	0	1
0	1	0	1	0	0	1
0	1	1	0	1	1	1
1	0	0	1	0	0	0
1	0	1	0	1	0	1
1	1	0	0	1	0	1
1	1	1	1	1	1	1

Формулы, записываемые на основе полной таблицы истинности, получаются в совершенной дизъюнктивной нормальной форме. Для суммы следует взять те строки, где сумма равна 1, то есть строки 1, 2, 5 и 8. Для переноса воспользуемся 4, 6, 7 и 8 строками. Для простоты записи опустим в формулах индексы при переменных.

$$S_i = \bar{a}\bar{b}\bar{p} + \bar{a}b\bar{p} + \bar{a}b p + abp \quad (7.2)$$

$$P_{i+1} = ab\bar{p} + \bar{a}bp + a\bar{b}p + abp$$

Дважды добавляя во второе выражение произведение  $abp$ , упростим формулу для переноса.

$$S = \bar{a}b\bar{p} + \bar{a}b p + \bar{a}b p + abp \quad (7.2a)$$

$$P_{i+1} = ab + bp + ap$$

В соответствии с (7.2a) для синтеза сумматора необходимы логические схемы И-ИЛИ. Для работы сумматора, построенного по (7.2a) на его входы необходимо подавать прямые и инверсные значения переменных. Вариант схемы сумматора приведен на рис. 7.1.

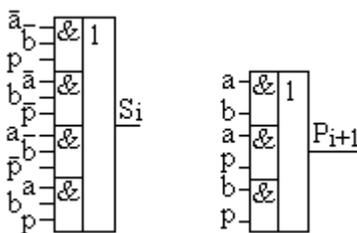


Рис. 7.1. Комбинационный сумматор, выполненный на элементах И-ИЛИ.

Сумматор, синтезированный непосредственно по выражению (7.2a) является сумматором комбинационного типа. Оба слагаемых и сигнал переноса должны в таких сумматорах подаваться на входы одновременно. Сигнал на выходе сумматора с точностью до времени задержки логическими схемами существует на входе в течение того же времени, что и входные сигналы. Если задержка сигнала в одном логическом элементе  $t_3$ , то задержка сигнала переноса и сигнала суммы относительно момента переключения входных сигналов равна  $2 t_3$ . Приведенный на рис. 7.1 вариант сумматора является самым быстрым. Он синтезирован по бесскобочной форме булевых выражений, по дизъюнктивной нормальной форме, дающей двухступенчатую схему.

Сумматор на рис. 7.1 не имеет средств для синхронизации и, следовательно, сложение происходит в момент поступления слагаемых. Поэтому при необходимости выполнения сложения в заданный момент времени следует применять стробирование входных сигналов. Недостатком таких сумматоров является то, что результат не хранится в самом сумматоре. Поэтому в процессе суммирования результат должен записываться в ре-

гистр, предназначенный для хранения кода суммы. Такой регистр обычно называют *аккумулятором*.

Комбинационные сумматоры обычно применяют в тех случаях, когда регистры выполнены на триггерах потенциального типа. Достоинством комбинационного сумматора является его высокая производительность. Особенно важно получение малой задержки сигнала в цепях переноса, так как основные потери времени в многоразрядных сумматорах, как мы увидим далее, происходят при переносе. Обозначение одноразрядного сумматора приведено на рис. 7.2.

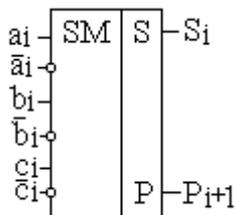


Рис. 7.2. Условное обозначение одноразрядного сумматора

Как следует из выражений 7.2 и рис.7.1 значения суммы и переноса формируются независимо друг от друга.

Так как сумматор является важнейшим узлом вычислительных машин, рассмотрим другие варианты и возможности построения сумматоров. Так, если в серии интегральных микросхем имеются только элементы инверсной логики, то, преобразуя выражения для суммы и переноса (7.2а) к виду

$$\begin{aligned}
 S_i &= \overline{\overline{a}b}p + \overline{a}\overline{\overline{b}p} + \overline{\overline{a}b}p + \overline{a}\overline{\overline{b}p} = \overline{\overline{a}b \& \overline{a}b \& \overline{a}b \& \overline{a}b} \\
 P_{i+1} &= \overline{\overline{a}b} + \overline{a}\overline{\overline{b}p} + \overline{\overline{b}p} = \overline{\overline{a}b \& \overline{a}p \& \overline{b}p}
 \end{aligned}
 \tag{7.3}$$

получаем схему, изображенную на рис. 7.3.

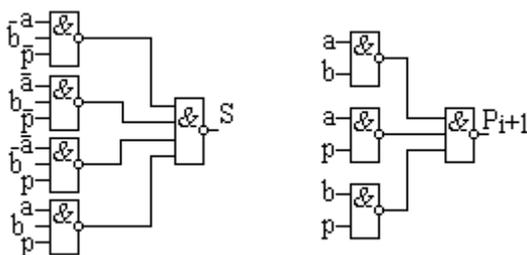


Рис. 7.3. Один разряд двоичного комбинационного сумматора, построенный на элементах инверсной логики.

Преобразуя выражения (7.2) можно получить различные исходные соотношения для построения одноразрядного комбинационного сумматора. Эти выражения можно, например, переписать в конъюнктивной форме:

$$S_i = (a+b+p)(a+\bar{b}+\bar{p})(\bar{a}+b+\bar{p})(\bar{a}+\bar{b}+p),$$

$$P_{i+1} = (a+b)(a+p)(b+p). \quad (7.4)$$

В этом случае первая ступень сумматора строится на схемах ИЛИ, а вторая – на схемах И, как показано на рис. 7.4.

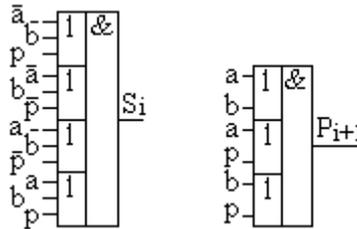


Рис. 7.4. Один разряд комбинационного двоичного сумматора на схемах ИЛИ-И

Рассмотрим возможности преобразования цепей переноса. На основе использования цепей собственного и транзитного переноса, в соответствии с выражением  $P_{i+1} = c_i + p_i T_i = a_i b_i + p_i (a_i + b_i)$ , построена схема, приведенная на рис. 7.5.



Рис. 7.5. Схема переноса с цепями транзита и собственного переноса

В схеме на рис. 7.5 общее число входов всех элементов равно 8, а в схемах, представленных на рис. 7.1–7.4, элементы цепей переноса имеют по 9 входов. Сигнал переноса, формирующийся в случае выполнения условия транзита, задерживается по отношению к моменту поступления входных сигналов на  $3t_3$ .

Схемы на рис. 7.1, рис. 7.3 и рис. 7.4 содержат по пять трехвходовых схем, по три двухвходовых и по одной четырехвходовой. Общее число входов равно 25. Если для синтеза сумматора воспользоваться выражениями для суммы и для переноса, преобразованными к такому виду, чтобы в них появились одинаковые слагаемые, то можно и для формирования суммы и для формирования переноса частично применить одни и те же узлы.

Запишем и преобразуем выражение для инверсии переноса

$$\overline{P_{i+1}} = \overline{ab+ap+bp} = \overline{ab+a p+b p}. \quad (7.5)$$

К такому же виду можно привести выражение для переноса, частично совпадающее с выражением для суммы

$$\begin{aligned} \overline{abp} + \overline{abp} + \overline{bp} &= \overline{abp} + \overline{abp} + \overline{bp} + \overline{bp} = \overline{b(ap + p)} + \overline{p(ab + b)} = \\ &= \overline{ab} + \overline{bp} + \overline{ap} + \overline{bp} = \overline{ab} + \overline{bp} + \overline{ap}. \end{aligned} \quad (7.6)$$

Поэтому в качестве правила для получения инверсии переноса воспользуемся выражением:

$$\overline{p_{i+1}} = \overline{abp} + \overline{abp} + \overline{bp}. \quad (7.7)$$

Схема, синтезированная по последнему выражению и по выражению для суммы (формула 7.2), изображена на рис. 7.6. Она содержит одну двухвходовую логическую схему, пять трехвходовых, одну четырехвходовую и один инвертор.

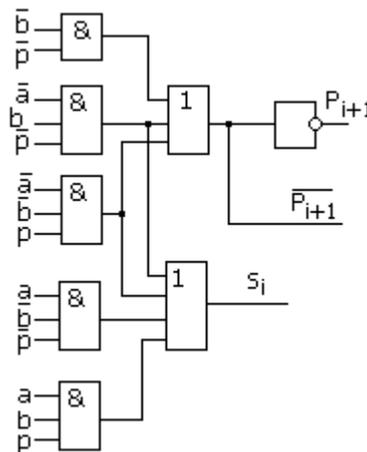


Рис.7.6. Схема сумматора с 22 входами

Если в выражении для суммы два сомножителя в третьем слагаемом заключить в скобки, то число входов уменьшится еще на единицу, как показано на рис. 7.7. Обратите внимание на особенности соединения элементов на этом рисунке.

$$S_i = \overline{abp} + \overline{abp} + a(\overline{bp}) + abp. \quad (7.8)$$

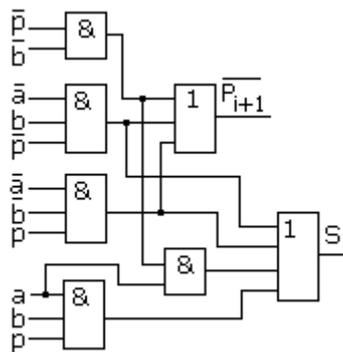


Рис. 7.7. Схема сумматора с 21 входом

Для получения суммы можно воспользоваться частью схемы, формирующей перенос. Чтобы убедиться в этом, в выражении для суммы выделим группу слагаемых, совпадающих с выражением для переноса. Для этого к каждой из первых трех конъюнкций добавим конъюнктивные слагаемые, равные нулю и содержащие соответствующую входную переменную без знака инверсии.

$$\begin{aligned}
 S_i &= \overline{ab}p + \overline{a}b\overline{p} + a\overline{b}\overline{p} + abp = (\overline{ab}p + \overline{a}p\overline{p} + \overline{b}p\overline{p}) + (\overline{ab}\overline{p} + \overline{a}b\overline{b} + \overline{p}b\overline{b}) + \\
 &(\overline{ab}\overline{p} + \overline{a}a\overline{b} + \overline{a}a\overline{p}) + abp = abp + b(\overline{a}\overline{p} + \overline{a}b + \overline{b}\overline{p}) + a(\overline{ab} + \overline{a}\overline{p} + \overline{b}\overline{p}) + \\
 &p(\overline{ab} + \overline{a}\overline{p} + \overline{b}\overline{p}) = abp + (a + b + p)\overline{(ab + ap + bp)} = abp + \\
 &(a + b + p)\overline{P_{i+1}}.
 \end{aligned}
 \tag{7.9}$$

Схема сумматора, синтезированная по выражению 7.9, приведена на рис. 7.8. В этом сумматоре нет необходимости в подаче на вход инверсий слагаемых и переноса. Задержка сигнала переноса происходит в двух элементах, а задержка сигнала суммы – в пяти.

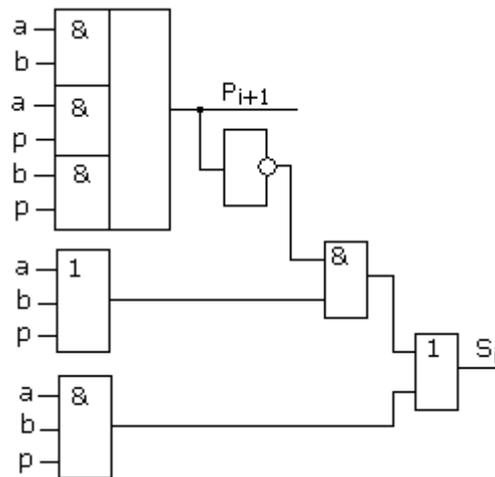


Рис. 7.8. Схема сумматора, не использующая инверсные входные переменные

Приведение выражений для переноса к виду

$$p_{i+1} = ab + (a + b)p \tag{7.10}$$

и выражений для суммы к виду

$$S_i = (ab)p + [(a + b) + p]\overline{p_{i+1}} \tag{7.11}$$

позволяет синтезировать схему, отличающуюся значительной экономичностью. Из приведенных выражений видно, что в схеме, формирующей сумму, и в схеме, формирующей сигнал переноса, во входных каскадах следует применить одинаковые узлы или использовать один и тот же узел для обеих частей схемы. Такая схема изображена на рис. 7.9.

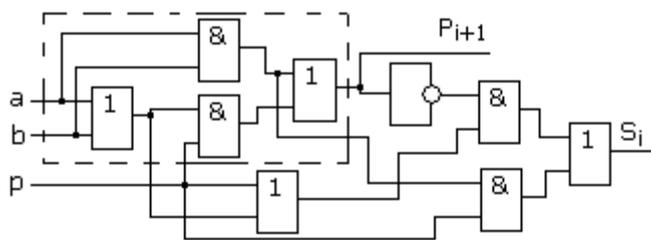


Рис.7.9. Экономичная схема сумматора

Общее число входов в этой схеме равно 17. При этом нет ни одного элемента более чем с двумя входами. На рис. 7.9 штриховой линией выделена схема уже рассмотренная ранее (см. рис. 7.5).

На логических схемах типа И-НЕ, имеющих открытый коллекторный выход, можно выполнить устройство более простое в сравнении с устройством, приведенным на рис. 7.9. В качестве коллекторной нагрузки схем с открытым коллекторным выходом применяют внешний резистор, один для нескольких открытых выходов. При этом получается логическая схема, называемая «проводное ИЛИ». Включение транзисторных вентилях с открытым коллектором, имеющих общую нагрузку, приведено на рис. 7.10. Если из числа выходных транзисторов ТЗ, имеющих общую нагрузку, хотя бы один открыт, то на выходе будет низкий потенциал, то есть логический 0. В целом схема, изображенная на рис. 7.10, реализует логическую функцию И-ИЛИ-НЕ.

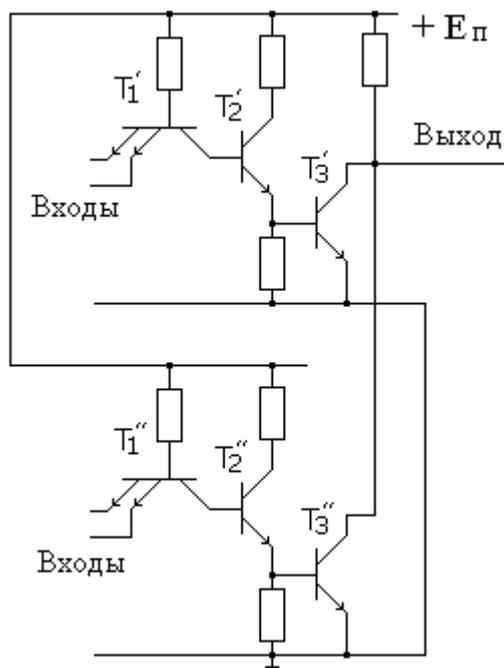


Рис. 7.10. Объединение двух базовых элементов с открытым коллектором в одну схему

Подавая на вход сумматора, реализованного на элементах И-НЕ, прямые значения переменных, получим на выходе инверсные значения суммы и переноса [27].

$$\begin{aligned} \overline{P_{i+1}} &= \overline{a_i b_i + b_i p_i + a_i p_i}, \\ \overline{S_i} &= \overline{a_i b_i p_i + (a_i + b_i + p_i) p_{i+1}} = \\ &= \overline{a_i b_i p_{i+1} p_{i+1} + b_i p_{i+1} + p_i p_{i+1}}. \end{aligned} \quad (7.26)$$

Выражение (7.26) позволяет синтезировать схему сумматора (рис. 7.11), содержащую 7 элементов с общим числом входов, равным 15.

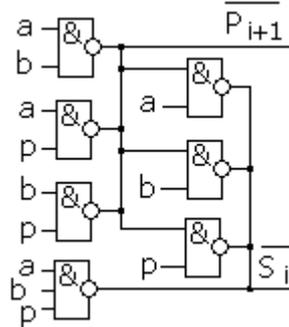


Рис.7.11.Один разряд комбинационного сумматора на схемах И-НЕ с открытым коллектором

Используя таблицу, аналогичную таблице 7.1, синтезируем комбинационный вычитатель. Таблица истинности 7.2 для вычитателя имеет следующие обозначения:  $a_i$  – уменьшаемое,  $b_i$  – вычитаемое,  $d_i$  – единица, занимаемая в младший,  $i-1$  разряд,  $d_{i+1}$  – единица, занимаемая из старшего,  $i+1$  разряда, а  $R_i$  – разность.

Таблица 7.2

Вычитатель: таблица истинности				
$d_i$	$A_i$	$b_i$	$R_i$	$d_{i+1}$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

Опуская индексы, запишем выражение для разности и заема:

$$\begin{aligned} R_i &= \overline{a}b\overline{d} + a\overline{b}\overline{d} + \overline{a}b\overline{d} + a\overline{b}d, \\ d_{i+1} &= \overline{a}b\overline{d} + \overline{a}b\overline{d} + \overline{a}b\overline{d} + a\overline{b}d. \end{aligned} \quad (7.12)$$

преобразуем выражение для цифры заема к виду:

$$d_{i+1} = \overline{a}b(\overline{d} + d) + \overline{a}d(\overline{b} + b) + bd(\overline{a} + a) = \overline{a}b + \overline{a}d + bd. \quad (7.13)$$

Схема вычитателя, синтезированная на основе выражения (7.13), приведена на рис.7.12.

Сравнивая выражение для разности с выражением для суммы, видим, что  $S_i$  и  $R_i$  отличаются только обозначениями  $p$  и  $d$ .

Следовательно, вычитание может быть выполнено тем же устройством, что и сложение, если вместо  $p$  и  $\bar{p}$  на соответствующие входы подать  $d$  и  $\bar{d}$ .

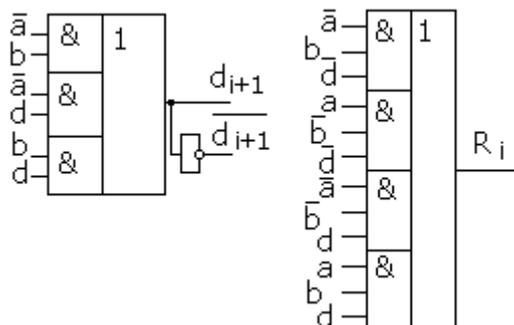


Рис. 7.12. Схема вычитателя

Выражения для заема отличаются от выражения для переноса тем, что первые два слагаемых вместо « $a$ » содержат « $\bar{a}$ ». В таком вычитателе при прямом вычитании из меньшего числа большего результат получается в дополнительном коде, например,

$$\begin{array}{r}
 23D \ 0100111 \\
 - 34D \ -100010 \\
 \hline
 -11D \ 110101 \text{ в дополнительном коде} \\
 -001011 \text{ в прямом коде}
 \end{array}$$

Признаком вычитания большего числа из меньшего является заем 1 из разряда вне разрядной сетки числа.

Вводя переключатели  $K$  и  $L$  в выражения для заема и, соответственно, в схемы для реализации заема (переноса) получаем логическое выражение и схему, осуществляющую либо сложение, либо вычитание в зависимости от выбора подаваемого управляющего сигнала. При этом и сигнал заема и сигнал переноса подаются на вход  $p$ . Сигналы  $K$  и  $L$  не могут принимать одинаковые значения ( $K = \bar{L}$ ).

Схема одного разряда комбинационного сумматора-вычитателя приведена на рис. 7.13. При подаче логической 1 на вход  $K$  схема работает как сумматор, а при подаче логической 1 на вход  $L$  – как вычитатель.

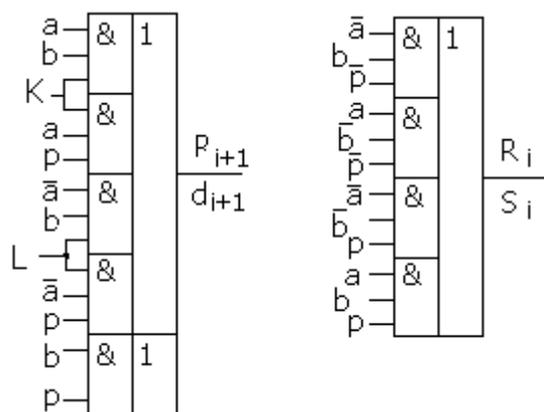


Рис. 7.13. Двоичный сумматор – вычитатель (один разряд)

Наряду с комбинационными сумматорами существуют накапливающие сумматоры. Одноразрядный накапливающий сумматор производит суммирование поочередно поступающих на его вход цифр слагаемых и переноса. Такой сумматор запоминает результат суммирования. Накапливающий сумматор строится на основе триггера со счетным входом, реализующим функцию сложения по модулю 2 ( $T$  – триггер). Функциональная схема одного разряда накапливающего сумматора изображена на рис. 7.14.

После поступления на вход триггера цифры (сигнала) первого слагаемого  $a_i$  через время  $t$ , превышающее время установления триггера, на счетный вход триггера подается цифра второго слагаемого  $b_i$ . Поскольку в триггере уже хранится цифра первого слагаемого, он реализует функцию

$$F_1 = a \oplus b. \quad (7.14)$$

Для простоты в выражении (7.14) и далее индексы опущены.

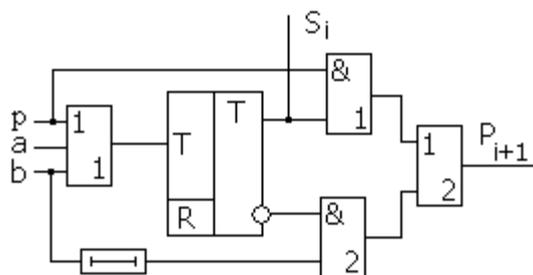


Рис. 7.14. Накапливающий сумматор, выполненный на счетном триггере

При подаче слагаемых может сформироваться сигнал переноса (если  $a = 1$  и  $b = 1$ ), соответствующий первому слагаемому в выражении (7.10).

После подачи слагаемых подается сигнал переноса. Если  $b = 0$  и  $a + b = 1$ , то перенос при наличии переноса из младшего разряда, формирует-

ся схемой И<sub>1</sub>. После подачи  $p_i$  суммирование завершается. При этом получаем

$$\begin{aligned} f_2 &= f_1 \oplus (\bar{a}\bar{b} + \bar{a}b)\bar{p} + (\bar{a}b + ab)p = \bar{a}\bar{b}\bar{p} + \bar{a}b\bar{p} + (\bar{a}\bar{b})(\bar{a}b)p = \\ &= \bar{a}\bar{b}\bar{p} + \bar{a}b\bar{p} + (\bar{a} + b)(a + \bar{b})p = \bar{a}\bar{b}\bar{p} + \bar{a}b\bar{p} + \bar{a}bp + abp = S_i. \end{aligned} \quad (7.14a)$$

Выражение (7.14a) дает соотношение для получения суммы, совпадающее с выражением (7.2), полученным по таблице истинности для сумматора. После подачи сигнала переноса на выходе схемы И<sub>1</sub> получаем

$$f_3 = f_1 p = (\bar{a}\bar{b} + \bar{a}b)p = \bar{a}\bar{b}p + \bar{a}bp. \quad (7.15)$$

Сигнал  $f_3$ , если он появляется, существует на выходе, пока существует на входе сигнал  $p_i$  (с точностью до задержки в схемах И<sub>1</sub> и ИЛИ<sub>2</sub>). В соответствии с выражением для  $f_3$  формируется сигнал переноса, если есть условия для распространения переноса (транзита). Выражение (7.15) соответствует второму и третьему слагаемым в выражении (7.2).

Слагаемое  $b_i$  задерживается элементом задержки на время переходных процессов в триггере и объединяется с  $\bar{f}_1$  схемой И<sub>2</sub>.

$$f_4 = \bar{f}_1 b = (\bar{a}\bar{b} + \bar{a}b)b = (\bar{a}\bar{b})(\bar{a}b)b = (\bar{a} + b)(a + \bar{b})b = ab. \quad (7.16)$$

Элемент И<sub>2</sub> реализует функцию  $f_4$ . Суммируя  $f_3$  и  $f_4$  и учитывая, что,  $\bar{b}p + b = p + b$ , а  $\bar{a}p + a = p + a$ , получаем

$$\begin{aligned} P_{i+1} &= \bar{a}\bar{b}p + \bar{a}bp + ab = \bar{a}\bar{b}p + \bar{a}bp + ab + ab = \\ &= a(\bar{b}p + b) + b(\bar{a}p + a) = ap + ab + bp + ab = ab + ap + bp. \end{aligned} \quad (7.17)$$

В сумматорах накапливающего типа при сложении одно из слагаемых обычно находится в сумматоре (как результат предыдущей операции). При подаче на вход сумматора кода добавляемого числа образуется сумма этого числа с числом, ранее находившимся в сумматоре. Таким образом, можно сложить любое количество чисел. Полученная сумма сохраняется после удаления кода слагаемого. Сигнал переноса не сохраняется.

Достоинством такого сумматора является простота суммирования с накоплением результата. Недостатком накапливающего сумматора является необходимость использования не менее двух тактов для образования суммы, что увеличивает время, затрачиваемое на сложение.

В зависимости от способа передачи чисел в ЭВМ многоразрядные числа суммируются параллельным или последовательным способом. Последовательный сумматор преобразует последовательные коды слагаемых в последовательный код суммы. Последовательный сумматор наиболее просто строится на основе одноразрядного комбинационного сумматора.

Рассмотрим последовательный комбинационный сумматор, схема которого изображена на рис. 7.15. Этот сумматор состоит из одноразрядного сумматора и схемы запоминания переноса. На входы последователь-

ного комбинационного сумматора в каждом такте операции сложения поступают одноименные разряды слагаемых и единица переноса из младшего разряда, который суммировался в предыдущем такте. При передаче переноса из одного разряда в другой осуществляется запоминание переноса на время одного такта сложения для того, чтобы этот сигнал поступил на вход одновременно со следующим разрядом слагаемых.

Суммирование в последовательных сумматорах требует столько тактов сложения, сколько разрядов в числе.

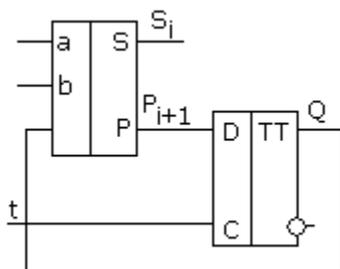


Рис.7.15. Последовательный комбинационный сумматор

Сократить время, затрачиваемое на операцию суммирования, можно путем одновременного сложения кодов слагаемых во всех разрядах одновременно. Так как для получения результата в каждом разряде необходимо выполнение всех переносов в младших разрядах, то получение окончательного значения суммы происходит после осуществления всех переносов.

Для одновременного суммирования всех разрядов слагаемых применяют сумматоры параллельного типа. В таком сумматоре для каждого разряда числа имеется отдельный одноразрядный сумматор. Выход переноса каждого одноразрядного сумматора соединяется с входом переноса соседнего старшего разряда. Функциональная схема сумматора параллельного типа показана на рис. 7.16.

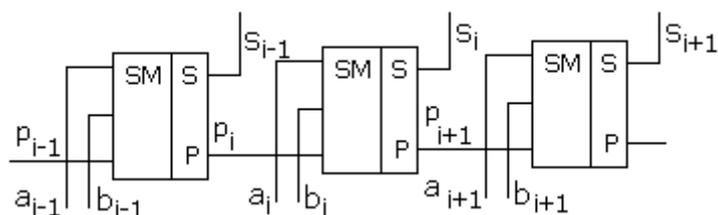


Рис. 7.16. Сумматор параллельного типа

В параллельных сумматорах могут использоваться как комбинационные, так и накапливающие элементы. На рис. 7.16 нетрудно видеть, что перенос, образовавшийся в одном из разрядов, распространяется к старшим разрядам по цепочке сумматоров. Например, если сложить число, состоящее из единиц во всех разрядах, с числом, содержащим единицу лишь

в младшем разряде, то сигнал переноса пройдет через все разряды. Только после этого можно начинать новое суммирование. Быстродействие параллельных сумматоров определяется в основном скоростью распространения сигнала переноса. Поэтому в параллельных сумматорах особое внимание надо обращать на выбор структуры цепей распространения переноса. В сумматоре, изображенном на рис. 7.16, осуществляется последовательный перенос от разряда к разряду. В параллельном сумматоре с последовательным переносом в наихудшем случае время, затрачиваемое на перенос:

$$T_{пер} = \tau N,$$

где  $\tau$  – время распространения переноса в одном разряде, а  $n$  – число разрядов сумматора.

Для ускорения переноса можно воспользоваться одним из двух способов: уменьшением числа элементов, через которые проходит сигнал переноса в каждом разряде, или передачей сигнала переноса по обходным цепям непосредственно на входы соответствующих разрядов.

Рассмотрим два разряда сумматора, реализованного на элементах инверсной логики, рис. 7.17.

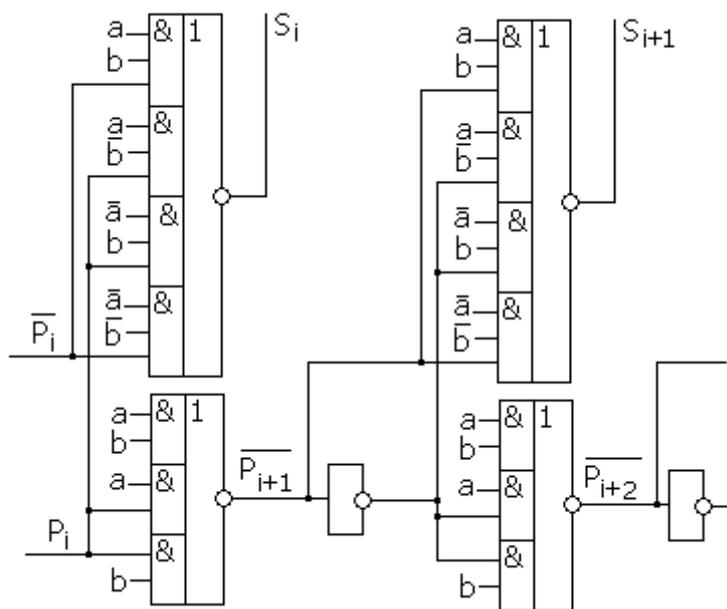


Рис. 7.17. Два разряда комбинационного сумматора на элементах инверсной логики

Соотношения для построения этой схемы могут быть получены из 7.2:

$$S_i = \overline{\overline{ab}p + \overline{a\bar{b}}p + \overline{a\bar{b}}p + \overline{ab}p},$$

$$\overline{P_{i+1} = \overline{ab + ap + bp}};$$
(7.18)

$$P_{2i+1} = \overline{\overline{a_{2i}b_{2i} + a_{2i}p_{2i} + b_{2i}p_{2i}}}; \quad (7.19)$$

для чего запишем выражение для суммы, с использованием выражения для инверсии переноса. При преобразованиях воспользуемся добавлением произведений, равных нулю.

$$\begin{aligned} S_{2i} &= \overline{a_{2i}b_{2i}p_{2i}} + \overline{a_{2i}b_{2i}p_{2i}} + \overline{a_{2i}b_{2i}p_{2i}} + \overline{a_{2i}b_{2i}p_{2i}} = \\ &= (\overline{a_{2i}b_{2i}p_{2i}} + \overline{a_{2i}p_{2i}p_{2i}} + \overline{b_{2i}p_{2i}p_{2i}}) + (\overline{a_{2i}b_{2i}p_{2i}} + \overline{a_{2i}b_{2i}b_{2i}} + \\ &+ \overline{p_{2i}b_{2i}b_{2i}}) + (\overline{a_{2i}b_{2i}p_{2i}} + \overline{a_{2i}a_{2i}b_{2i}} + \overline{a_{2i}a_{2i}p_{2i}}) + \overline{a_{2i}b_{2i}p_{2i}} = \\ &= \overline{a_{2i}b_{2i}p_{2i}} + \overline{b_{2i}(a_{2i}p_{2i} + a_{2i}b_{2i} + b_{2i}p_{2i})} + \overline{a_{2i}(b_{2i}p_{2i} + \\ &+ a_{2i}b_{2i} + a_{2i}p_{2i})} + \overline{p_{2i}(a_{2i}b_{2i} + a_{2i}p_{2i} + b_{2i}p_{2i})} = \\ &= \overline{a_{2i}b_{2i}p_{2i}} + (\overline{a_{2i} + b_{2i} + p_{2i}})(\overline{a_{2i}b_{2i} + a_{2i}p_{2i} + b_{2i}p_{2i}}) = \\ &= \overline{a_{2i}b_{2i}p_{2i}} + (\overline{a_{2i} + b_{2i} + p_{2i}})[\overline{(a_{2i} + b_{2i})(a_{2i} + p_{2i})(b_{2i} + p_{2i})}] = \\ &= \overline{a_{2i}b_{2i}p_{2i}} + (\overline{a_{2i} + b_{2i} + p_{2i}})P_{2i+1}. \end{aligned} \quad (7.20)$$

На основе приведенных выражений получаем выражение для сумм для  $2_i$ -го и для  $2_{i+1}$ -го разрядов и для инверсии переноса на входе  $2_{i+2}$  разряда

$$\begin{aligned} S_{2i} &= \overline{\overline{a_{2i}b_{2i}p_{2i} + (a_{2i} + b_{2i} + p_{2i})P_{2i+1}}} = \\ &= \overline{(\overline{a_{2i} + b_{2i} + p_{2i}}) \& (\overline{a_{2i}b_{2i}p_{2i}} + P_{2i+1})} = \\ &= \overline{a_{2i}b_{2i}p_{2i} + a_{2i}p_{2i+1} + b_{2i}p_{2i+1} + p_{2i}p_{2i+1}}, \end{aligned} \quad (7.21)$$

$$\begin{aligned} S_{2i+1} &= \overline{a_{2i+1}b_{2i+1}p_{2i+1} + a_{2i+1}p_{2i+2} + b_{2i+1}p_{2i+2} + \\ &+ p_{2i+1}p_{2i+2}}, \end{aligned} \quad (7.22)$$

$$P_{2i+2} = \overline{a_{2i+1}b_{2i+1} + a_{2i+1}p_{2i+1} + b_{2i+1}p_{2i+1}}. \quad (7.23)$$

Если в нечетных разрядах использовать для формирования переноса схемы, выполненные в соответствии с соотношением (7.19), а в четных разрядах, выполненные в соответствии с соотношением (7.23), то сигнал переноса при распространении по сумматору проходит через два элемента в каждом разряде, элемент И-ИЛИ-НЕ и элемент И-НЕ (НЕ).

В формулах (7.21) и (7.22) соотношения для суммы приведены к такому виду, чтобы для ее образования использовать только сформированные сигналы переноса или его инверсии без дополнительного инвертирования. Схема, изображенная на рис. 7.18, основана на выражениях (7.19), (7.21), (7.22) и (7.23). При получении выражений (7.18) – (7.23) использовано легко проверяемое соотношение

$$\overline{P_{i+1}} = \overline{ab + ap + bp} = \overline{a} \& \overline{b} + \overline{a} \& \overline{p} + \overline{b} \& \overline{p}. \quad (7.24)$$

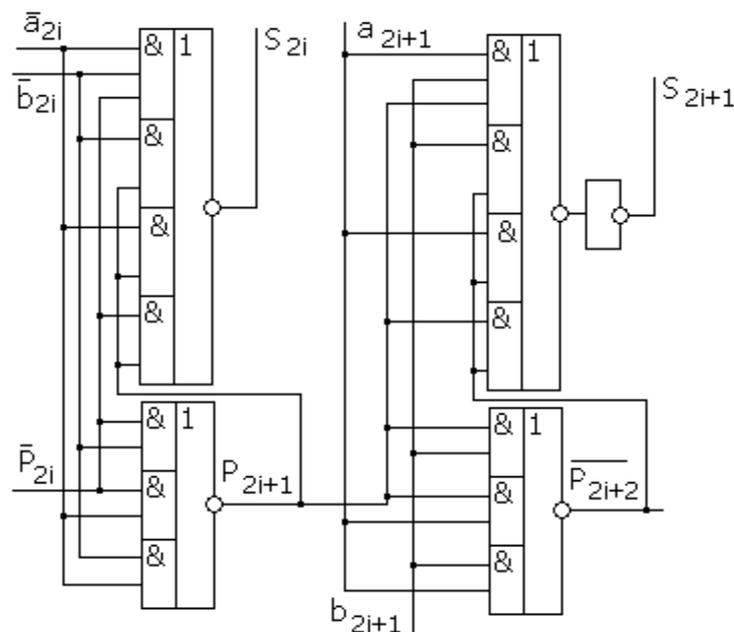


Рис. 7.18. Использование разных схем в четных и нечетных разрядах (для получения прямого значения суммы в каскаде с нечетным номером добавляется инвертор)

Функцию переноса  $P_{i+1}$  в соответствии с (7.10) можно записать следующим образом:

$$P_{i+1} = a_i b_i + p_i (a_i + b_i) = c_i' + p_i T_i, \quad (7.25)$$

где  $c_i'$  – перенос, сформированный непосредственно в  $i$ -ом разряде и называемый собственным переносом из  $i$ -го разряда,  $T_i$  – условие транзита, а  $p_i T_i$  – перенос, переданный из предыдущих разрядов, называемый *транзитным переносом*.

Выражение (7.25) является описанием одного звена цепи описания переноса. Из этого выражения видно, что цепь передачи переноса может быть составлена из схем И и ИЛИ. Если разделить каждый одноразрядный сумматор на две части: на собственно сумматор с входами  $a_i$ ,  $b_i$  и  $p_i$  и выходами  $S_i$  и  $T_i$  и на цепи переноса, то придем к схеме, изображенной на рис. 7.19. Такой сумматор называется *сумматором со сквозным переносом*.

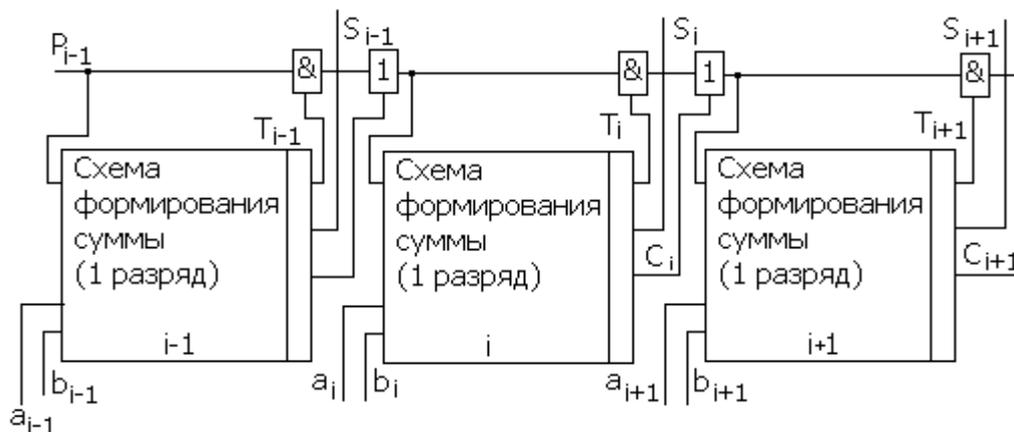


Рис. 7.19. Схема сумматора со сквозным переносом

В сумматорах со сквозным переносом быстродействие повышается за счет упрощения цепи распространения переноса. Последовательный характер распространения переноса в этой схеме сохраняется. Для большего ускорения переноса можно перенос в каждый разряд формировать отдельной комбинационной схемой, на вход которой подаются значения, получаемые в разрядах слагаемых. Для  $N$ -разрядного сумматора необходима  $N-1$  такая схема.

Рассмотрим логические формулы, описывающие перенос, начиная с переноса во второй разряд. Эти выражения можно составить в соответствии с (7.25). Учтем, что

$$\begin{aligned}
 C_1' &= P_2, \\
 P_2 &= a_1 b_1, \\
 P_3 &= C_2 + T_2 p_2 = C_2 + T_2 C_1, \\
 P_4 &= C_3 + T_3 p_3 = C_3 + T_3 C_2 + T_3 T_2 p_2, \\
 &\dots\dots\dots \\
 P_{i-1} &= C_{i-2} + T_{i-2} p_{i-2}, \\
 P_i &= C_{i-1} + T_{i-1} p_{i-1}.
 \end{aligned} \tag{7.26}$$

Последовательно подставляя выражения с меньшими индексами в выражения с большими индексами, получим выражение

$$\begin{aligned}
 P_i &= C_{i-1} + T_{i-1} C_{i-2} + T_{i-1} T_{i-2} C_{i-3} + T_{i-1} T_{i-2} T_{i-3} C_{i-4} + T_{i-1} T_{i-2} \dots \\
 &\dots T_3 C_2 + T_{i-1} T_{i-2} \dots T_3 T_2 C_1.
 \end{aligned} \tag{7.27}$$

В соответствии с (7.27) перенос  $c_j$ , из любого разряда младше  $i$ -го, может стать причиной переноса, поступающего на  $i$ -й разряд, если все разряды между источником собственного переноса и  $i$ -м разрядом транзитны для переноса, то есть выполняется условие

$$C_j = 1, T_k = 1 \quad (j < k < i). \tag{7.28}$$

Комбинационная схема, непосредственно реализующая формулы (7.26), вырабатывает перенос за время, необходимое для распространения сигнала через два логических элемента. Сумматор, построенный в соответствии с выражением (7.26), называют сумматором с одновременным переносом. Цепи переноса реализуются с помощью схем И и ИЛИ, на входы которых подаются переменные

$$T_i = a_i + b_i, C_i = a_i b_i. \tag{7.29}$$

Так как выражения вида  $a_i b_i$  и  $a_i + b_i$  могут быть использованы также и для формирования значений суммы в разрядах, то в комбинационных схемах целесообразно применять двухступенчатое формирование суммы. При этом первая ступень используется для формирования промежуточных значений  $a_i b_i$ ,  $a_i + b_i$  и для суммирующей части и для цепей переноса. Трехразрядный сумматор, построенный по такой схеме, изображен на рис. 7.20. Как на рис. 7.17, 7.19 на 7.20, младшие разряды изображены слева.

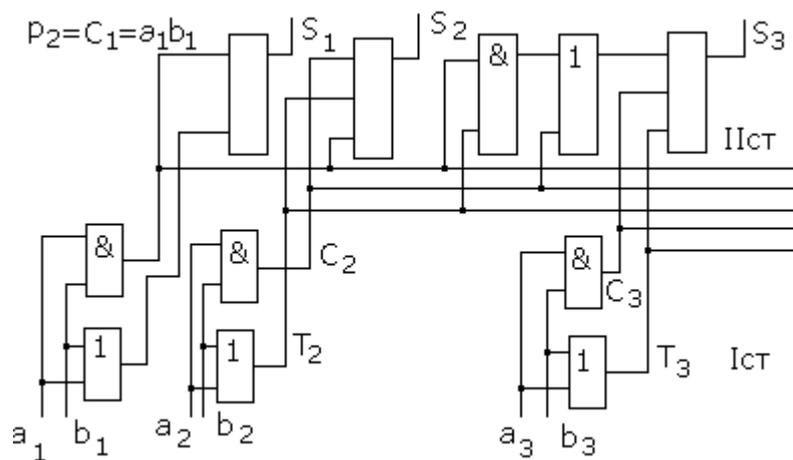


Рис. 7.20. Трехразрядный сумматор, построенный по двухступенчатой схеме

Так как первый разряд сумматора не имеет входа переноса, то в соответствии с выражением

$$S_1 = (a_1 + b_1)\overline{a_1 b_1} = a_1 \overline{b_1} + \overline{a_1} b_1 \quad (7.30)$$

он может быть выполнен на двух элементах, как показано на рис. 7.21. Таковую схему называют полусумматором.

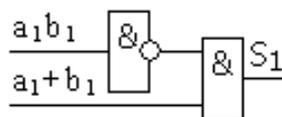


Рис. 7.21. Первый разряд двоичного сумматора

В остальных разрядах сумматор целесообразно выполнить в соответствии с выражением (7.31), следующим из (7.9),

$$S_i = (a_i b_i) p_i + [(a_i + b_i) + p_i] \overline{p_{i+1}}, \quad (7.31)$$

как показано на рис. 7.22.

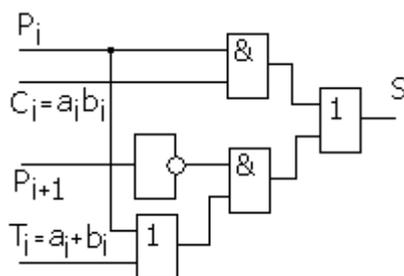


Рис. 7.22. Структура разряда сумматора, соответствующая выражению (7.31)

В схеме, изображенной на рис. 7.20, цепи переноса построены, как показано на рис. 7. 23.

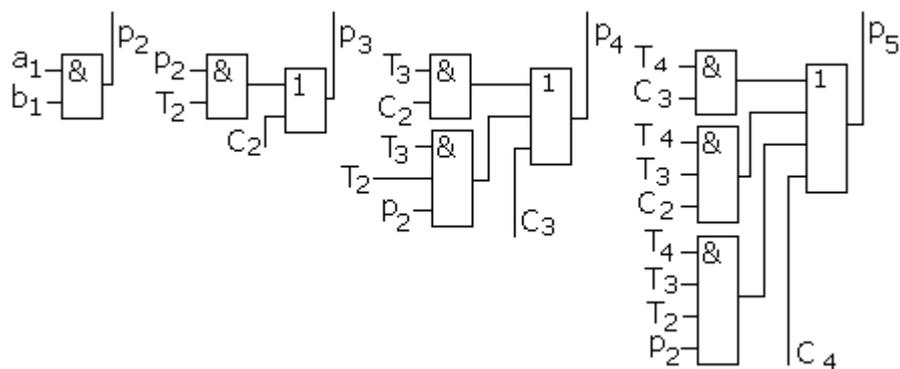


Рис. 7.23. Схемы одновременного переноса

Сумматоры с параллельным (одновременным) переносом требуют больших затрат оборудования. На рис. 7.23 видно, что число входов логических схем растет при переходе к старшим разрядам сумматора.

Компромиссным решением между сумматором со сквозным переносом и сумматором с одновременным переносом является сумматор с групповым переносом. Необходимость в сумматорах с групповым переносом вызвана ограниченной нагрузочной способностью схем и ограничением на число входов логических схем И.

Разделим сумматор, имеющий  $N$  разрядов на группы по  $m$  разрядов в каждой, и введем следующие обозначения:  $\Pi_k^{sp}$  – перенос на входе  $k$ -й группы (на входе младшего разряда  $k$ -й группы);  $P_{kj}$  – перенос на входе  $j$ -го разряда  $k$ -й группы;  $T_{kj}$  – признак транзита переноса через  $j$ -й разряд  $k$ - группы.

Переносы, поступающие во все разряды  $k$ -й группы, вычисляются по приведенным ниже формулам (7.32)

$$P_{km} = c_{k,m-1} + T_{k,m-1} p_{k,m-1} \quad (7.32)$$

Выражение (7.32) отличается от выражения (7.26) тем, что первым уравнением является уравнение для  $\Pi_{k1}$ , а не для  $\Pi_{k2}$ , то есть учитывается перенос, поступающий на вход группы.

На вход следующей,  $k + 1$  группы должен поступать перенос

$$\Pi_{k+1}^{gp} = c'_{km} + T_{km} \cdot p_{km}. \quad (7.33)$$

Формулы (7.32) являются описанием сквозного переноса внутри группы, а (7.33) – сквозного переноса на вход следующей группы. Выполнив подстановки, аналогичные сделанным в формулах (7.26), получим описание схемы одновременного переноса, действующего в пределах одной группы.

$$\begin{aligned}
p_{k,j+1} &= c'_{kj} T_{kj} p_{kj} + T_{kj} T_{k,j-1} p_{k,j-1} + \dots \\
&+ T_{k,j} T_{k,j-1} \dots T_{k2} p_{k,2} + T_{kj} \dots T_{k2} T_{k1} \Pi_k^{lp}.
\end{aligned}
\tag{7.34}$$

Приведенное выражение описывает параллельный перенос внутри группы на вход ее  $j + 1$  разряда.

На вход следующей группы поступает перенос:

$$\begin{aligned}
\Pi_{k+1}^{lp} &= c'_{km} + T_{km} c'_{k,m-1} + T_{km} T_{k,m-1} c'_{k,m-2} + \dots \\
&\dots + T_{km} T_{k,m-1} \dots T_{k2} c'_{k1} T_{km} T_{k,m-1} \dots T_{k2} T_{k1} \Pi_k^{lp}.
\end{aligned}
\tag{7.35}$$

Если ввести обозначения

$$\begin{aligned}
c_k^{lp} &= c'_{km} + T_{km} c'_{k,m-1} + \dots + T_{km} T_{k,m-1} + \dots \\
&+ T_{k2} c'_{k1}, \\
T_k^{lp} &= T_{km} T_{k,m-1} \dots T_{k2} T_{k1},
\end{aligned}
\tag{7.36}$$

то последнее выражение можно записать в виде:

$$\Pi_{k+1}^{lp} = c_k^{lp} + T_k^{lp} \Pi_k^{lp},
\tag{7.37}$$

где  $c_k^{lp}$  – собственный перенос из группы разрядов с номером  $k$ , а  $T_k^{lp}$  – признак транзита переноса через группу разрядов с номером  $k$ .

Выражение (7.37) является описанием сквозного переноса между группами. Разворачивая это выражение в последовательность формул, аналогичную (7.20) и (7.21), и производя преобразования, такие как сделаны выше, получим описание одновременного (параллельного) переноса между группами. Для построения схемы одновременного переноса внутри группы, в которой вырабатываются сигналы  $C^{lp}$  и  $\Pi^{lp}$ , запишем выражения (7.33) и (7.37) для четырехразрядной группы:

$$\begin{aligned}
P_{k1} &= \Pi_k^{lp}, \\
P_{k2} &= c'_{k1} + T_{k1} \Pi_k^{lp}, \\
P_{k3} &= c'_{k2} + T_{k2} c'_{k1} + T_{k2} T_{k1} \Pi_k^{lp}, \\
P_{k4} &= c'_{k3} + T_{k3} c'_{k2} + T_{k3} T_{k2} c'_{k1} + T_{k3} T_{k2} T_{k1} \Pi_k^{lp}, \\
\Pi_{k+1}^{lp} &= c'_{k4} + T_{k4} c'_{k3} + T_{k4} T_{k3} c'_{k2} + \\
&+ T_{k4} T_{k3} T_{k2} c'_{k1} + T_{k4} T_{k3} T_{k2} T_{k1} \Pi_k^{lp}, \\
T_k^{lp} &= T_{k4} T_{k3} T_{k2} T_{k1}; \\
C_k^{lp} &= c'_{k4} + T_{k4} c'_{k3} + T_{k4} T_{k3} c'_{k2} + T_{k4} T_{k3} T_{k2} c'_{k1}.
\end{aligned}
\tag{7.39}$$

Как следует из этих выражений, для построения цепей переноса в рассматриваемом случае требуется 14 схем совпадения, в том числе 4 с четырьмя входами, одна с пятью и 4 схемы ИЛИ, в том числе одна с пятью входами и две с четырьмя. В целом, сумматор, кроме этих схем, содержит еще суммирующую часть и схемы, служащие для формирования значений  $C'_{ki} = a_{ki} b_{ki}$  и  $T_{ki} = a_{ki} + b_{ki}$ , не показанные на приведенном ниже рис. 7.24.

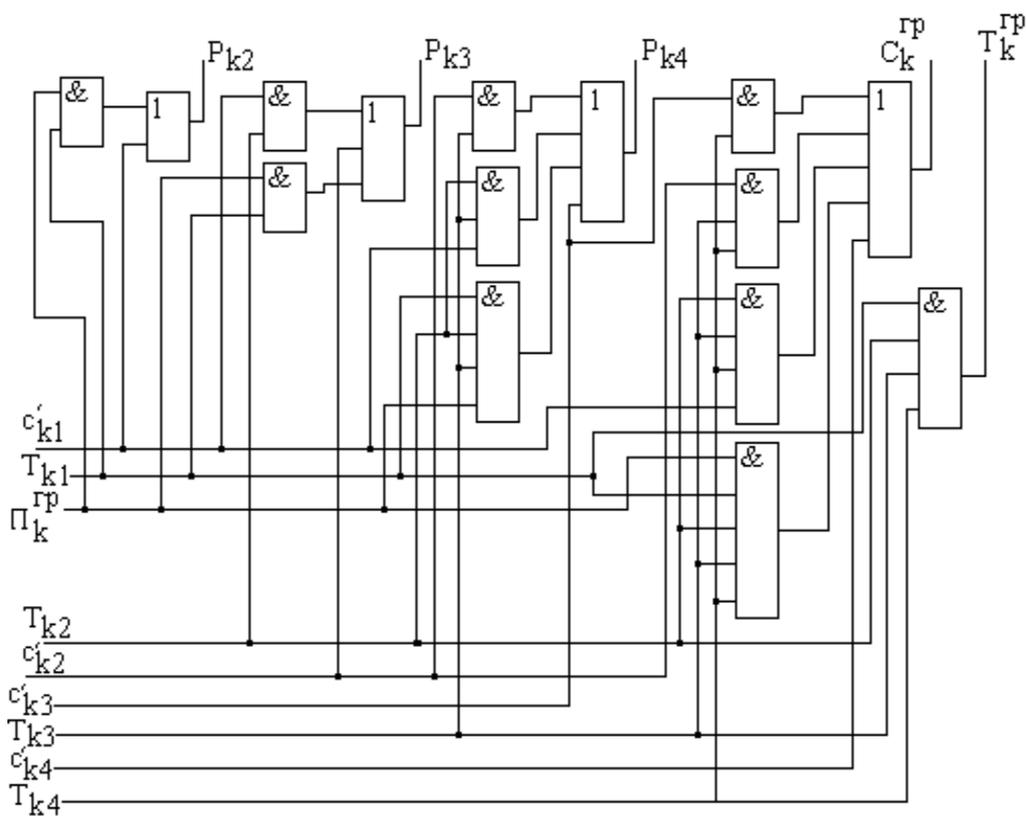


Рис. 7.24. Схема формирования группового переноса, использующая произведения и суммы аргументов соответствующих разрядов

Для выработки сигналов переноса, подаваемых на входы групп можно использовать схемы переноса между группами (вторую ступень схемы переноса), воспринимающую входные сигналы  $C^{гр}$  и  $T^{гр}$ . Схема переноса второй ступени может быть построена так же, как схема переноса внутри группы (второй ступени), либо в виде схемы сквозного переноса, либо в виде схемы одновременного переноса.

Группы из  $m$  двоичных разрядов можно объединить в новые более крупные группы с двухступенчатой схемой переноса для каждой из таких групп и схемой переноса третьей ступени, объединяющей эти крупные группы. Схема сквозного переноса между группами изображена на рис. 7.25. Схема одновременного переноса между группами изображена на рис. 7.26. На рис. 7.27 изображена схема сквозного переноса внутри групп и между группами.

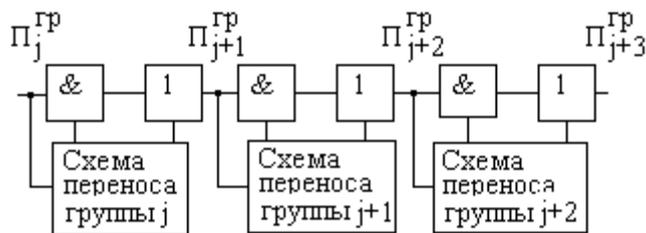


Рис. 7.25. Схема сквозного переноса между группами



Рис. 7.26. Схема одновременного переноса между группами

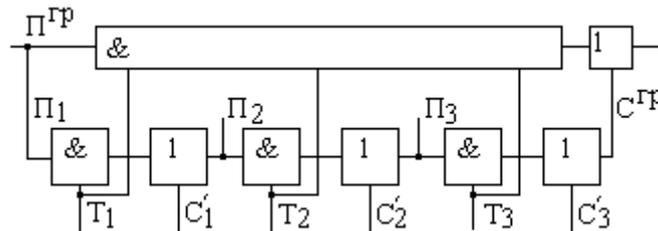


Рис. 7.27. Схема сквозного переноса внутри групп и между группами

Время, необходимое для выработки переноса в многоступенчатой схеме переноса, складывается из выработки переноса в каждой ступени, начиная с младшей (первой), исходных данных для следующей ступени переноса, вплоть до последней. Кроме того, затрачивается время на выработку в каждой ступени, начиная со старшей, входных переменных для предыдущей ступени, вплоть до выработки в первой ступени входных переносов на отдельные двоичные разряды сумматора<sup>18</sup>.

Если схемы переноса во всех ступенях построены по принципу одновременного переноса, то время выработки сигналов в каждой ступени не зависит от количества разрядов. Полное время переноса в этом случае определяется количеством ступеней переноса. В двухступенчатой схеме это время равно  $6t_3$ . В первой ступени переноса затрачивается время  $2t_3$  на выработку переносов на входы второй ступени, во второй ступени –  $2t_3$  на выработку переносов на входы первой ступени и  $2t_3$  на выработку в первой ступени сигналов переноса, подаваемых на входы двоичных разрядов сумматора. В трехступенчатой схеме это время равно  $10t_3$  или  $8t_3$ , если третья ступень содержит только 2 группы.

Последний случай соответствует 32 разрядному сумматору, разделяемому на группы по 4 разряда ( $m = 4$ ), и образованию во второй ступени переносов для четверок групп.

Примерами микросхем сумматоров являются микросхемы К155ИМ1 и К155ИМ2. Микросхема К155ИМ2 представляет собой полный двухразрядный комбинационный сумматор, изображенный на рис 7.28. Он функционирует в соответствии с выражениями:

<sup>18</sup> Не путайте ступень схемы переноса с разрядом сумматора.

$$S_1 = a_1 b_1 p_1 + (a_1 + b_1 + p_1) \overline{p_2},$$

$$\overline{p_2} = \overline{a_1 b_1 + b_1 p_1 + a_1 p_1},$$

$$S_2 = \overline{p_2 p_3 + b_2 p_3 + a_2 p_3 + a_2 b_2 p_2} = a_2 b_2 p_2 + (a_2 + b_2 + p_2) \overline{p_3}, \quad (7.40)$$

На рис. 7.28 первый разряд расположен справа, а второй – слева. Вход переноса в первом разряде предусмотрен для того, чтобы из таких микросхем можно было собирать многоразрядные сумматоры.

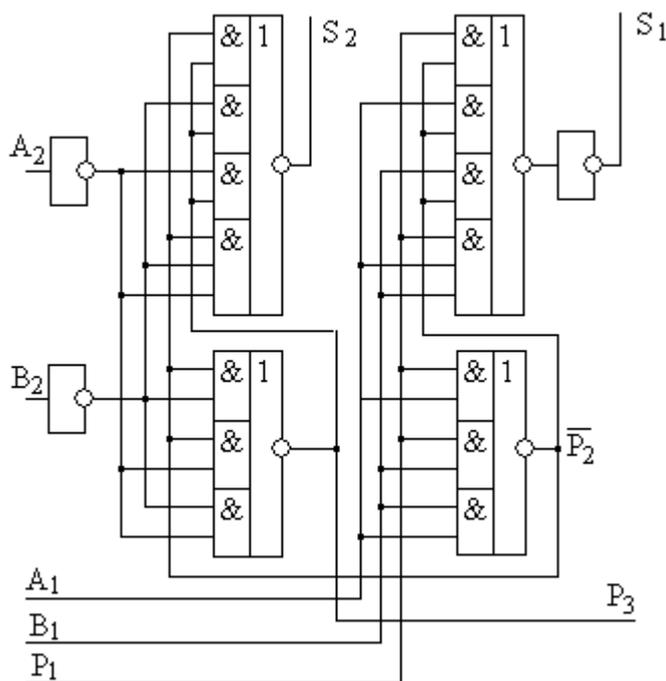


Рис. 7.28. Структура микросхемы двухразрядного сумматора

Условное графическое изображение микросхемы К155ИМ2 приведено на рис. 7.29.

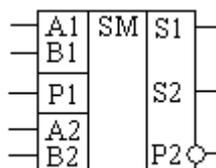


Рис. 7.29. Условное обозначение микросхемы сумматора (К155ИМ2)

Схема полного комбинационного сумматора К155ИМ3 является последовательным соединением двух схем, представленных на рис. 7.28 (7.18).

## 7.2. Суммирование двоично-десятичных кодов

Для выполнения операций над числами, представленными в десятичной системе счисления, могут быть использованы сумматоры как комбинационного, так и накапливающего типов. Десятичные сумматоры комбинационного типа строятся на основе двоичных комбинационных сумматоров, к которым добавляются логические схемы для коррекции значения суммы и формирования переноса.

Десятичные сумматоры накапливающего типа строятся на основе двоичных счетчиков, охваченных цепями обратной связи

Рассмотрим комбинационный двоичный сумматор, работающий в коде 8421. Для построения одного десятичного разряда необходимо четыре разряда двоичного сумматора. Четыре разряда двоичного сумматора осуществляют сложение по модулю 16. Иначе говоря, сигнал переноса из такой тетрады (из старшего двоичного разряда) сформируется, если сумма равна или больше  $16D$ .

Десятичный перенос (перенос по правилам десятичной системы счисления) должен сформироваться, если сумма получилась в пределах от  $10D$  до  $15D$  включительно. Поэтому при программной коррекции сумм десятичных кодов для правильного формирования десятичного кода суммы и сигнала переноса к кодам слагаемых добавляют код цифры 6, как пояснено выше в разделе «Арифметические основы ЦВМ». Формирование единицы переноса из двоичной тетрады означает уменьшение содержимого тетрады на 16, а не на 10, как необходимо при десятичном счете.

В связи с необходимостью коррекции десятичный сумматор состоит из двух ступеней. Вторая ступень содержит лишь три двоичных разряда: второй, третий и четвертый. Код суммы на выходе первой ступени анализируется с помощью конъюнкторов и дизъюнкторов. Если есть признаки десятичного переноса,

$$p_{10} = S'_4(S'_2 + S'_3) = S'_2S'_4 + S'_3S'_4, \quad (7.41)$$

то в соответствие с этими признаками формируется единица переноса и этот сигнал подается во второй и третий разряды второй ступени сумматора, где суммируется с результатом, полученным в первой ступени. Подчей единицы во второй и третий разряды тетрады осуществляется коррекция кодом 0110 (двоичным кодом десятичной цифры 6).

С выхода второй ступени снимается скорректированный двоично-десятичный код суммы. Если слагаемые образуют значение сумм  $S_{10} < 10$ , то перенос не формируется и код суммы проходит через вторую ступень без коррекции. Код суммы после коррекции имеет истинное двоично-десятичное значение, так как сумма дополняется до значения  $16D = 10H$  или большего. Так как при коррекции был добавлен код цифры 6, то код правильной десятичной суммы при переносе уменьшается на  $10D$ , то есть так,

как должно быть при десятичном счете. Десятичный сумматор, построенный в соответствии с рассмотренным методом, изображен на рис. 7.30.

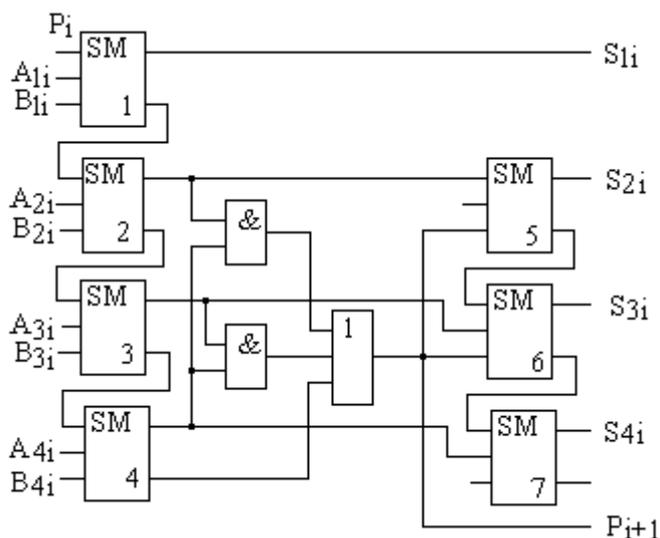


Рис. 7.30. Разряд десятичного сумматора, построенного на одnorазрядных двоичных сумматорах

### Вопросы для самопроверки

1. Составьте таблицу соответствия входных и выходных величин для одnorазрядного двоичного сумматора.
2. Запишите логические формулы для суммы и переноса для одnorазрядного двоичного сумматора.
3. Какие логические устройства содержит сумматор комбинационного типа?
4. Как определить задержку выходных сигналов относительно входных в одnorазрядном комбинационном сумматоре?
5. Сохраняется ли результат в комбинационном сумматоре?
6. Запишите таблицу соответствия выходных и входных сигналов для вычитателя.
7. Запишите логические выражения для разности и заема.
8. Одновременно или поочередно следует подавать входные слагаемые на входы комбинационного сумматора?
9. На основе какого устройства строится накапливающий сумматор?
10. Сохраняется ли результат сложения в накапливающем сумматоре?
11. Как суммируются в ЭВМ многоразрядные числа?
12. Одновременно или поочередно следует подавать слагаемые на вход накапливающего сумматора?
13. Какие элементы содержит последовательный сумматор?
14. Какие способы ускорения переноса применяют в параллельных сумматорах?
15. Какой перенос называется последовательным?

16. Какой перенос называется сквозным?
17. Какой перенос называется одновременным?
18. Какой перенос называется групповым?
19. Сколько разрядов двоичного сумматора необходимо для реализации одного разряда десятичного сумматора, работающего с двоично-десятичным кодом 8421?



Условные графические изображения дешифраторов приведены на рис. 8.2. Дешифратор может быть без стробирования и со стробированием. В последнем случае дешифратор дополнительно имеет вход С. Сигнал, подаваемый на этот вход, разрешает или не разрешает преобразование.

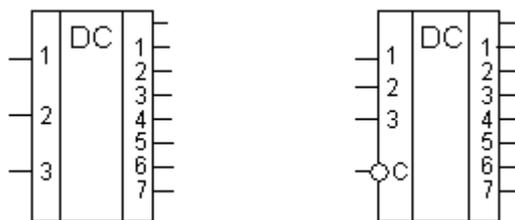


Рис. 8.2. Условное графическое обозначение дешифратора (слева без стробирования, справа со стробированием)

Рассмотрим различные схемы дешифраторов, оценим аппаратные затраты и быстродействие.

Аппаратные затраты, необходимые для создания логического устройства, будем оценивать количеством логических элементов в схеме  $N$  и общим количеством входов этих элементов  $h$ . Быстродействие комбинационных устройств будем оценивать на основе самой длинной цепи из всех возможных путей распространения сигнала. Для многокаскадных дешифраторов быстродействие определяется величиной, обратной произведению времени задержки сигнала в одном элементе  $t_3$  на число каскадов дешифратора  $K$ .

Для линейного дешифратора  $K = 1$ , число логических схем (клапанов)  $N = 2^N$ , общее число входов (цена дешифратора):

$$h = N \cdot 2^N, \quad (8.2)$$

а задержка сигнала равна  $t_3$ .

Если число разрядов машинного слова больше числа входов логических элементов, то логические схемы необходимо делать составными. В этом случае дешифратор уже будет не линейным, а каскадным. Каскадное включение схем конъюнкции осуществляют двумя способами, в соответствии с которыми выходные каскады дешифратора называют прямоугольными или пирамидальными.

В прямоугольном каскадном дешифраторе формат входного слова разделяется на слоги. Для каждого слога составляются линейные дешифраторы. На линейных дешифраторах для каждого слога образуются все входные значения. Например, для  $d_0$  получаем выражение:

$$d_0 = (\bar{X}_N * \bar{X}_{N-1} * \dots * \bar{X}_K)(\bar{X}_{K-1} * \bar{X}_{K-2} * \dots * \bar{X}_1). \quad (8.3)$$

Выходные сигналы линейных дешифраторов, образующих первую ступень устройства, объединяются во второй ступени, состоящей из прямоугольной дешифрирующей матрицы. Вторая ступень дешифратора вы-

полняет операции конъюнкции частичных (промежуточных) выходных значений, сформированных линейными дешифраторами первой ступени. Схема прямоугольного дешифратора приведена на рис. 8.3.

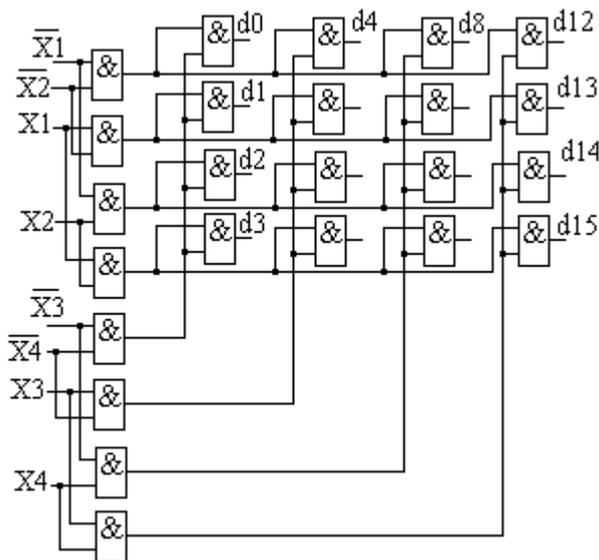


Рис. 8.3. Прямоугольный двухкаскадный дешифратор

В двухступенчатом дешифраторе, при четном  $n$ , дешифраторы первой ступени имеют равное число логических элементов и входов. Половина первой ступени содержит  $2^{N/2}$  логических элементов (схем совпадения). Каждая половина первой ступени использует половину входного кода, поэтому ее число входов равно  $(N/2)2^{N/2}$ . Число входов первой ступени в целом

$$h = N2^{N/2}. \quad (8.4)$$

Вторая ступень дешифратора выполняется на двухвходовых схемах совпадения.

Так как число выходов каждой части первой ступени равно числу элементов половины первой ступени  $2^{N/2}$ , то для их конъюнкции необходимо число элементов, равное

$$2^{N/2}2^{N/2} = 2^N, \quad (8.5)$$

а число входов второй ступени:

$$2 \cdot 2^N = 2N^{+1}. \quad (8.6)$$

Общее число входов двухступенчатого дешифратора

$$H = 2^{N+1} + N2^{N/2}. \quad (8.7)$$

Сравнивая (8.7) с (8.2), видим, что отношение числа входов линейного и двухступенчатого дешифраторов убывает с ростом  $N$ .

$$\frac{2^{N+1} + N \cdot 2^{N/2}}{N \cdot 2^N} = \frac{2 \cdot 2^N}{N \cdot 2^N} + \frac{N \cdot 2^{N/2}}{N \cdot 2^N} = \frac{2}{N} + \frac{1}{2^{N/2}}. \quad (8.8)$$

При  $n = 4$  двухступенчатый дешифратор имеет входов логических элементов на 25% меньше, чем линейный.

Время задержки двухступенчатого дешифратора равно  $2t_3$ .

Другим типом дешифратора является пирамидальный дешифратор. Схема пирамидального дешифратора имеет многокаскадную структуру и в каждом каскаде образует частичные значения, в виде конъюнкций частичных выходных значений и цифры одного из разрядов, не участвовавших в образовании предыдущих частичных значений.

Для получения выражения, по которому синтезируют пирамидальный дешифратор, поставим в выражении (8.1) скобки следующим образом:

$$d_0 = (((\dots(\bar{X}_n \bar{X}_{n-1}) \dots) \bar{X}_3) \bar{X}_2) \bar{X}_1,$$

Если необходимо, чтобы дешифратор срабатывал в момент поступления строб-сигнала<sup>19</sup>, надо добавить в последнее выражение еще один аргумент. Обозначим аргумент, соответствующий строб-сигналу  $Y_s$ , тогда

$$d_0 = (((\dots((Y_s \bar{X}_n) \bar{X}_{n-1}) \dots) \bar{X}_3) \bar{X}_2) \bar{X}_1. \quad (8.9)$$

Остальные строки таблицы истинности не записываем из-за простоты выражения. Схема пирамидального дешифратора приведена на рис. 8.4.

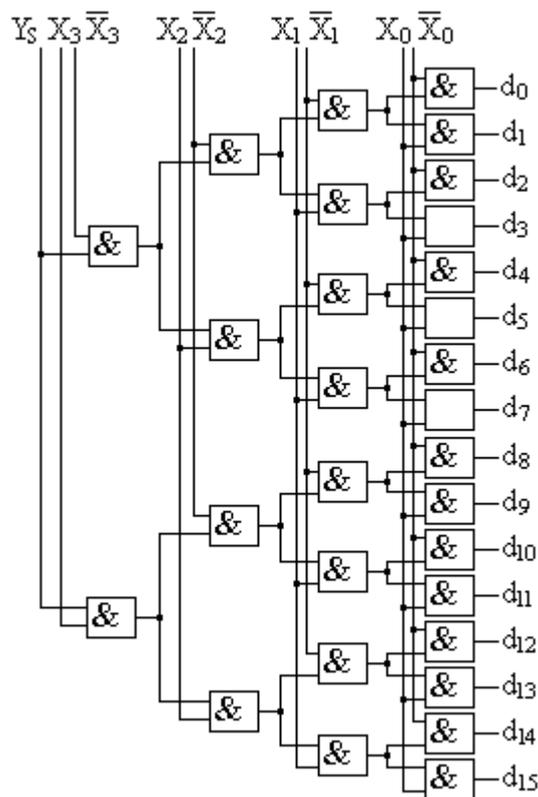


Рис. 8.4. Схема пирамидального дешифратора со стробированием. Для стробирования служит сигнал  $Y_s$  (см. левую линию)

В пирамидальном дешифраторе количество ступеней (каскадов) равно числу разрядов входного кода. Пирамидальный дешифратор является наиболее медленным видом дешифратора. Однако он позволяет приме-

<sup>19</sup> Стробирование – метод выделения некоторого интервала на временной оси.

нять простые двухходовые клапаны. Цена пирамидального дешифратора со стробированием:

$$h = 2N = 2(2^1 + 2^2 + \dots + 2^N) = 4(2^0 + 2^1 + 2^2 + \dots + 2^{N-1}) = 4(2^N - 1) \quad (8.10)$$

Цена пирамидального дешифратора без стробирования (без первой ступени на рис. 8.4)

$$h = 2(2^2 + 2^3 + \dots + 2^N) = 8(2^{N-1} - 1). \quad (8.11)$$

Сравним последнее выражение с (8.2):

$$\frac{8(2^{N-1} - 1)}{N \cdot 2^N} = \frac{8 \cdot 2^{N-1}}{N \cdot 2^N} - \frac{8}{N \cdot 2^N} = \frac{4 \cdot 2^N}{N \cdot 2^N} - \frac{8}{N \cdot 2^N} = \frac{4}{N} \left(1 - \frac{2}{2^N}\right). \quad (8.12)$$

При  $N = 4$  число входов пирамидального дешифратора на одну восьмую меньше, чем число входов линейного дешифратора. Для дешифратора со стробированием это отношение имеет вид (сравниваем 8.10 и 8.2):

$$\frac{4(2^N - 1)}{N \cdot 2^N} = \frac{4}{N} \left(1 - \frac{1}{2^N}\right). \quad (8.13)$$

Сравним двухступенчатый дешифратор с пирамидальным дешифратором без стробирования.

$$\frac{2^{N+1} + N \cdot 2^{N/2}}{8(2^{N-1} - 1)} \approx \frac{2^{N+1}}{8 \cdot 2^{N-1}} + \frac{N \cdot 2^{N/2}}{8 \cdot 2^{N-1}} = \frac{2 \cdot 2^N}{4 \cdot 2^N} + \frac{N \cdot 2^{N/2}}{4 \cdot 2^N} = \frac{1}{2} + \frac{N}{4 \cdot 2^{N/2}} = \frac{1}{2} \left(1 + \frac{N}{2^{1+N/2}}\right). \quad (8.14)$$

С ростом  $N$  последнее выражение стремится к  $1/2$ , следовательно, для больших  $N$  цена пирамидального дешифратора в два раза выше, чем двухступенчатого. Аналогичный результат получается и для дешифратора со стробированием. Для дешифратора со стробированием число входов на 4 больше, чем для дешифратора без стробирования.

Время задержки распространения сигнала для пирамидального дешифратора со стробированием равно  $Nt_3$ , а для дешифратора без стробирования  $(N-1)t_3$ .

### Вопросы для самопроверки

1. Какие логические схемы применяются в дешифраторах.
2. Почему нельзя построить линейный дешифратор для кода с большим количеством разрядов.
3. Запишите логическое выражение для описания двухкаскадного дешифратора.

## 9. АППАРАТНОЕ ПРЕОБРАЗОВАНИЕ КОДОВ ИЗ ДВОИЧНОЙ ФОРМЫ В ДЕСЯТИЧНУЮ

Двоичные числа, содержащие 4 или менее разрядов, преобразуются в двоично-десятичную форму непосредственно, как одна тетрада. Двоичные числа, имеющие более четырех разрядов, можно преобразовать в двоично-десятичную форму с помощью сдвигающих регистров. Для этого двоичное число надо сдвигать справа налево старшими разрядами вперед, записывая («вдвигая») его в регистр последовательно, разряд за разрядом, начиная с младшего (правого) разряда регистра. Сдвиги надо повторять до тех пор, пока младший разряд двоичного кода не запишется в младший разряд регистра. Регистр можно рассматривать как двоично-десятичную разрядную сетку. Когда единица пересекает границу между двоично-десятичными (десятичными) разрядами, то возникает ошибка, так как при переходе через границу десятичных разрядов значение этой единицы изменяется с 8 на 16, тогда как для двоично-десятичного кода это значение должно измениться с 8 до 10 (рис. 9.1)<sup>20</sup>.



Рис. 9.1. Сдвиг двоичного кода в двоично-десятичной разрядной сетке

Следовательно, необходима коррекция содержимого тетрады. Для коррекции содержимого тетрады необходимо в тетраду добавлять цифру 6, так как ее двоично-десятичное (десятичное) содержимое уменьшается на 6. Пусть преобразуется двоичный код 11111. Это код числа 31D. После сдвига на 3 разряда в регистре из трех тетрад будет код 0000 0000 0111, то есть код десятичного числа 007. Это правильный код – переносов между тетрадами не было и младшая тетрада в старшем разряде содержит 0. После четырех сдвигов в младшей тетрада будет недопустимый код 1111, а в целом, в трех тетрадах – код 0000 0000 1111. Соответственно, после пятого сдвига будет недопустимый (по правилам десятичной арифметики) код 0000 0001 1111. Чтобы не было тетрад, содержащих недопустимые значения, надо добавлять цифру 6 при переносе единицы в следующую тетраду и затем после каждого сдвига. Например, сдвинув код 0000 0000 0111, то есть, выполнив четвертый сдвиг, получим код, как отмечено выше, содержащий неверное значение в младшей тетраде (разряд единиц в десятичной системе счисления). Прибавив к младшей тетраде код 0110, получим в регистре код 0000 0001 0101. Это двоично-десятичный код числа 15D. Это верный результат,

<sup>20</sup> Обратите внимание, что при переходе из 3 разряда в 4, из 7 разряда в 8, правила переноса десятичные, а при других переносах – двоичные.

он соответствует вдвинутому в регистр коду. После пятого сдвига будет код 0000 0010 1011. Прибавив к младшей тетраде код 0110, получим 0000 0011 0001, что соответствует  $31D$ . Результат верный.

Вместо добавления кода 6 можно добавлять код 3 (0011) перед сдвигом. После одного сдвига этот код 0011 преобразуется в код 0110.

Рассмотрим коррекцию содержимого тетрад при преобразовании с помощью кода 0011. При анализе необходимости коррекции перед сдвигом выясняется, произойдет ли при сдвиге переход единицы через границу между тетрадами. Если перед сдвигом в тетраде записано 4 (0100) или меньше, то при сдвиге переход единицы через границу между тетрадами не произойдет.

Если же перед сдвигом в тетраде код  $5D$ ,  $6D$  или  $7D$ , то поскольку старший разряд тетрады при таких кодах содержит 0, то переход единицы через границу не произойдет. Однако в этих вариантах в тетрадах могут записаться недопустимые значения, а именно, 10, 12, 14 или 11, 13, 15.

Поэтому для тетрад, содержащих коды 5, 6 и 7 необходимо сделать коррекцию содержимого перед сдвигом путем прибавления кода 0011.

Если в тетраде записан код  $8D$  или  $9D$ , то необходимо скорректировать результат перехода единицы через границу между тетрадами. Эта коррекция также осуществляется предварительно кодом 0011. Таким образом, коррекция для всех случаев одинакова. Все варианты исчерпываются числами 5, 6, 7, 8 и 9. Эти варианты можно представить в виде таблицы входных и выходных кодов, описывающих работу корректирующего устройства. Корректирующее устройство должно подключаться к каждой четырехразрядной секции сдвигающего регистра. Проанализируем преобразование числа 11101010 ( $234D$ ), приведенного на рис. 9.2. Правило работы корректирующего элемента запишем в табл. 9.1.

Таблица 9.1

Правило работы корректирующего элемента

Десятичная цифра	Вход				Выход				Функция
	X1	X2	X3	X4	Y4	Y3	Y2	Y1	
X	X1	X2	X3	X4	Y4	Y3	Y2	Y1	Y
0	0	0	0	0	0	0	0	0	X
1	0	0	0	1	0	0	0	1	X
2	0	0	1	0	0	0	1	0	X
3	0	0	1	1	0	0	1	1	X
4	0	1	0	0	0	1	0	0	X
5	0	1	0	1	1	0	0	1	X + 3
6	0	1	1	0	1	0	1	0	X + 3
7	0	1	1	1	1	0	1	1	X + 3
8	1	0	0	0	1	0	0	0	X + 3
9	1	0	0	1	1	1	0	1	X + 3



Чтобы построить матрицу соответствия входных и выходных кодов для схемы, изображенной на рис. 9.4, надо подавать на входы  $X_1, X_2, \dots, X_5$  коды от 000000 до 111111.

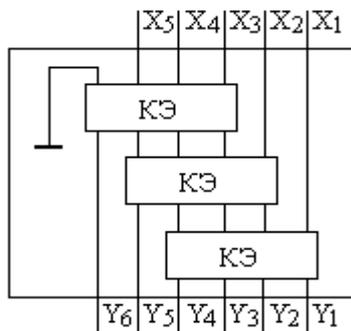


Рис. 9.4. Корректирующие элементы микросхемы K155PP7

Анализируя результат подачи кода, следует учитывать, что нулевой разряд входного кода на схему не подается. Условное графическое изображение этой микросхемы приведено на рис. 9.5.

С помощью одного экземпляра интегральной микросхемы можно преобразовать шестизначное двоичное число. Рассмотрим пример работы схемы при подаче во входные цепи кода 110101, рис. 9.6.

10	A	X/Y	Y1	1
11	B		Y2	2
12	C		Y3	3
13	D		Y4	4
14	E		Y5	5
			Y6	6
15	запр			7
16	Епит		общ	8

Рис.9.5. Условное графическое изображение микросхемы преобразователя

В прямоугольниках, обозначающих корректирующие элементы, сверху записаны коды, подаваемые на входы корректирующих элементов, а снизу коды, формируемые на выходах корректирующих элементов.

В первом и третьем элементах код шестерки дополняется до девяти. Во втором элементе коррекция не требуется, так как поступает код числа 3.

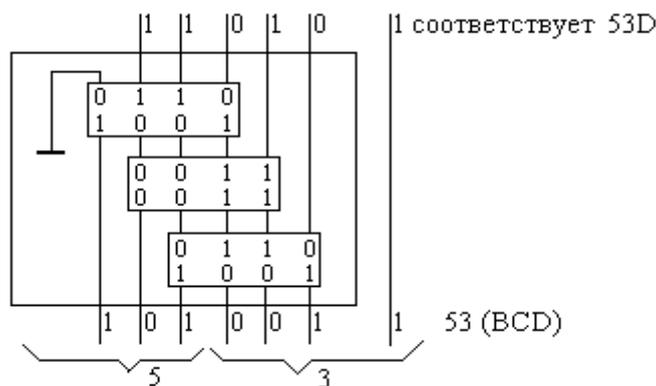


Рис. 9.6. Преобразование шестизначного двоичного числа

При необходимости преобразования многоразрядных двоичных чисел нужно воспользоваться каскадным соединением микросхем преобразователей. Рассмотрим преобразование в двоично-десятичную форму восьмиразрядного двоичного числа. Для этого, как показано на рис. 9.7, необходимо воспользоваться тремя экземплярами микросхемы.

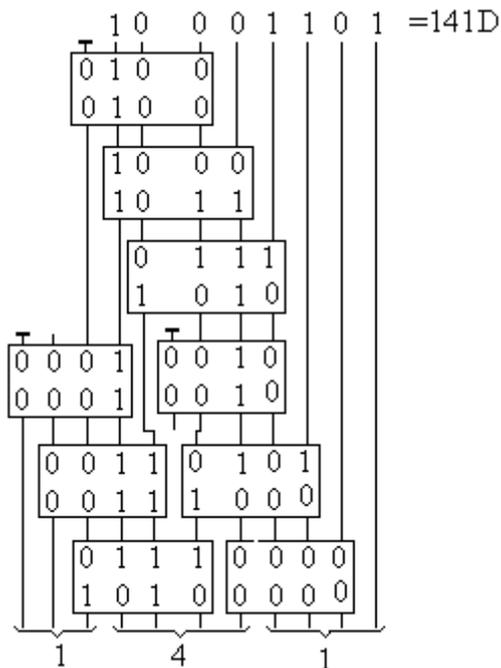


Рис. 9.7. Преобразование восьмиразрядного двоичного числа в двоично-десятичную форму

Микросхема преобразователя кодов имеет открытые коллекторные выходы, поэтому при ее применении необходимо выходные контакты подключать через резисторы к источнику питания. В соответствии с этим реальная электрическая схема выглядит, как показано на рис. 9.8.

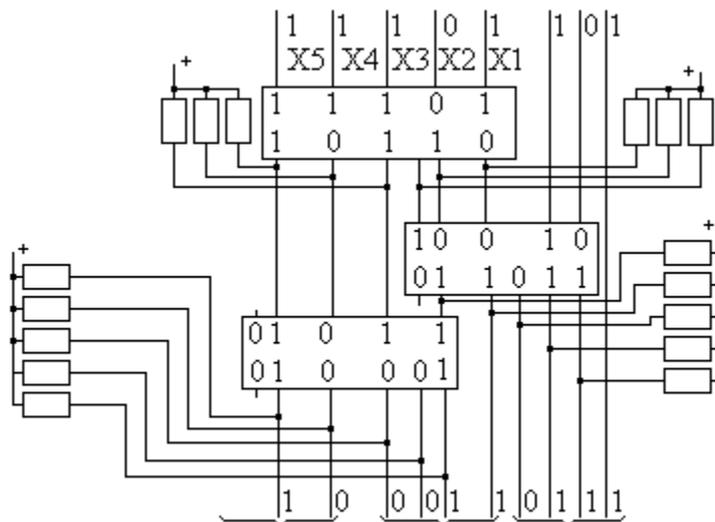


Рис. 9.8. Включение микросхемы преобразователя кодов

Из рассмотрения схемы на рис. 9.7 видно, что на каждую декаду и на каждый шаг сдвига необходимо по одному корректирующему элементу.

Формальные алгоритмы построения многокаскадных схем на основе рассмотренных преобразователей в литературе не встречаются. Однако для количества разрядов в словах, принятых в ЭВМ различных классов, эти схемы известны.

В качестве еще одного примера рассмотрим схему девятиразрядного преобразователя (рис.9.9). В ней применены четыре микросхемы К155ПР7.

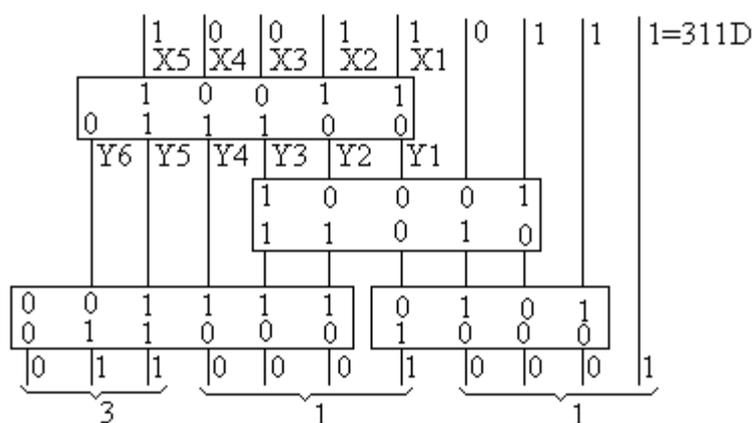


Рис.9.9. Девятиразрядный преобразователь кодов

Кратко рассмотрим преобразование двоично-десятичных кодов в двоичные. Такое преобразование можно выполнить посредством нескольких последовательных делений двоично-десятичного кода на 2. Так как отдельные десятичные цифры в этом случае уже представлены в двоичном коде, то деление на 2 можно выполнить, сдвигая код вправо на один разряд. Самый правый бит, выдвинутый из разрядной сетки, является искомым значением разряда (уже представленного в двоичном виде). Если при сдвиге единица пересекает границу между тетрадами, то возникает ошибка. При переходе от десятков к единицам значение разряда должно уменьшаться в два раза, то есть от 10 до 5. Но в случае двоичного числа единица приобретает вес, равный 8. Следовательно, для коррекции нужно вычитать 3. При коррекции, если в старшем разряде тетрады стоит единица, то вычитают три. Пользуясь этими правилами, можно составить таблицу соответствия выходов и входов для корректирующего элемента. Прделаем пример. Двоично-десятичный код числа 763 записывается в следующем виде: 0111 0110 0011. Этот код преобразуем в двоичный. Деление на 2 осуществим сдвигом кода в сторону младших разрядов.

Старшая тетрада		Младшая тетрада	Двоичное значение
0111	0110	0011	
0011	1011	0001	1

В первой строке таблицы помещена двоично-десятичная запись исходного числа. Во второй строке записан результат после одного сдвига в сторону младших разрядов. В той тетраде, где в старшем разряде 1, делаем коррекцию вычитанием кода 0011, и выполняем второй сдвиг.

Старшая тетрада		Младшая тетрада	Двоичное значение
0011	1011	0001	1
0011	1000	0001	1
0001	1100	0000	11

Во второй строке приведен результат коррекции (вычитания), а в третьей – результат второго сдвига.

Продолжим сдвиги и коррекцию.

Старшая тетрада		Младшая тетрада	Двоичное значение
0001	1001	0000	11
0000	1100	1000	011
0000	1001	0101	011
0000	0100	1010	1011

Делаем коррекцию в младшей тетраде.

Старшая тетрада		Младшая тетрада	Двоичное значение
0000	0100	0111	1011
0000	0010	0011	11011
0000	0001	0001	111011
0000	0000	1000	1111011

После коррекции в младшей тетраде и сдвигах получаем

Старшая тетрада		Младшая тетрада	Двоичное значение
0000	0000	0101	1111011
0000	0000	0000	101111011

В комбинационных схемах сдвиг двоично-десятичных разрядов достигается путем соответствующего соединения одинаковых комбинационных схем. Корректирующие схемы состояются из двух корректирующих элементов. Каждая схема реализуется в виде ПЗУ на 32 байта. Примером является микросхема К155ПР6.

### **Вопросы для самопроверки**

1. Сколько двоичных разрядов необходимо для кодирования одного десятичного разряда?
2. Каким кодом делается коррекция при преобразовании двоичного кода в двоично-десятичный код?
3. Вычитается или добавляется корректирующий код при преобразовании чисел из двоичной системы в двоично-десятичную?
4. Можно ли осуществлять коррекцию двоичным кодом 0011 при преобразовании из двоичного кода в двоично-десятичный?
5. Можно ли осуществлять коррекцию двоичным кодом цифры 3 при преобразовании двоично-десятичного кода в двоичный код?

## 10. МУЛЬТИПЛЕКСОРЫ

Выше в разделах РЕГИСТРЫ и СЧЕТЧИКИ приводились схемы реверсивных вариантов этих устройств. Между триггерами в реверсивных устройствах имеются переключатели, выполненные на комбинационных схемах. Эти переключатели принято называть мультиплексорами. Мультиплексор имеет несколько сигнальных (информационных) входов, один или более управляющих входов и один выход. Мультиплексор позволяет передавать сигнал с одного из входов на выход. Простейшая схема мультиплексора приведена на рис. 10.1.

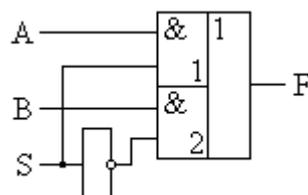


Рис. 10.1. Мультиплексор с двумя сигнальными входами A и B

Этот мультиплексор имеет два входа для переключаемых сигналов A и B и вход S, на который подается сигнал управления. При логической единице на входе S, на выход проходит сигнал A. Если  $S = 0$ , то проходит сигнал B. Таблица истинности (таблица 10.1) позволяет получить булеву функцию, описывающую работу рассматриваемого устройства. Диаграмма работы приведена на рис.10.2.

Таблица 10.1

Таблица истинности мультиплексора

S	A	B	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$$F = \bar{S}\bar{A}B + \bar{S}AB + S\bar{A}\bar{B} + SAB = \bar{S}B + SA. \quad S = 0, F = B; S = 1, F = A$$

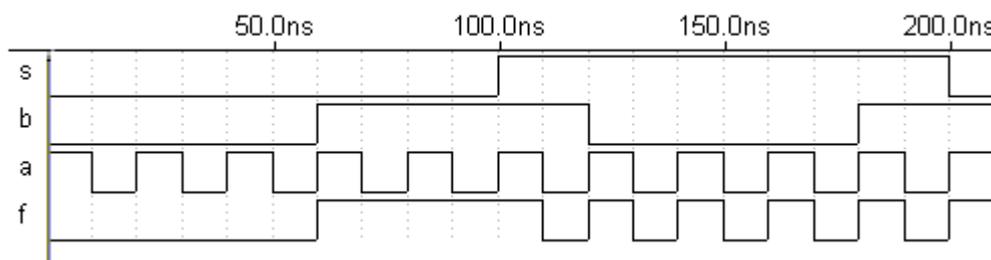


Рис. 10.2. Диаграмма работы мультиплексора с двумя сигнальными входами

Добавляя инверторы и используя трехходовые логические элементы, осуществляющие функцию конъюнкции, можно сделать инвертор с 4 сигнальными входами [20]. Схема такого устройства приведена на рис. 10.3.

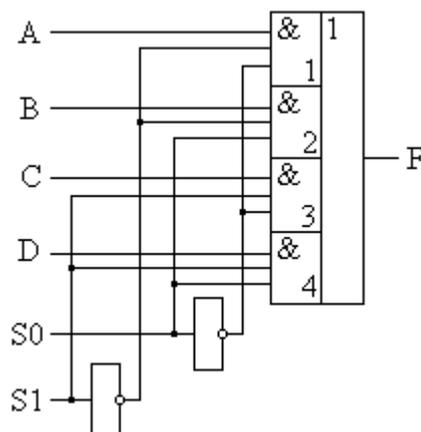


Рис. 10.3. Мультиплексор с четырьмя сигнальными входами

Из рис. 10.3 следует, что при  $S0 = 0$  и  $S1 = 0$   $F = A$ , при  $S0 = 1$  и  $S1 = 0$   $F = B$ , при  $S0 = 0$  и  $S1 = 1$   $F = C$ , а при  $S0 = 1$  и  $S1 = 1$   $F = D$ .

Перепишем последние соотношения в другом виде:

$$F = A = \overline{S1} \& \overline{S0}, \quad F = B = \overline{S1} \& S0, \quad F = C = S1 \& \overline{S0}, \quad F = D = S1 \& S0.$$

Эти выражения были уже получены при рассмотрении линейного дешифратора. Изменим входную цепь мультиплексора, применив двухразрядный линейный дешифратор, как показано на рис. 10.4 [26]. В мультиплексоре с дешифратором на входе применены более простые, двухходовые схемы совпадения в отличие от схемы на рис. 10.3, где мы видим трехходовые схемы.

Интегральные микросхемы мультиплексоров изготавливаются на основе технологий ТТЛ, ТТЛШ и КМОП. Микросхемы мультиплексоров обозначаются буквами КП, следующими непосредственно за номером серии, например К155КП1 (шестнадцатиразрядный мультиплексор со стробированием), К1533КП11А (селектор-мультиплексор 2x1 с тремя состояниями на выходе, 4 разряда). На схемах микросхемы мультиплексоров обозначаются буквами *MS*.

С помощью выпускаемых промышленностью интегральных схем мультиплексоров, включая их параллельно и объединяя входы управления можно создавать многоразрядные мультиплексоры.

Мультиплексоры можно применять не только для коммутации входных сигналов, но и для других целей. Мультиплексоры используют в операциях, связанных с выбором данных, при маршрутизации, задании последовательности передачи данных, преобразовании кодов, получении логических функций и т.п. [20].

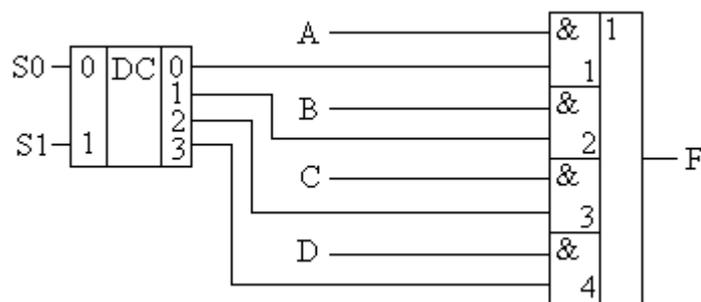


Рис. 10.4. Мультиплексор с адресными входами, активируемыми с помощью дешифратора

Рассмотрим в качестве примера формирование логической функции. Реализовать с помощью мультиплексора логическую функцию можно непосредственно по таблице истинности. Воспользуемся мультиплексором с восемью входами. Это позволит синтезировать устройство для получения функции трех аргументов. Входные переменные (значения аргументов функции) будем подавать на входы  $S_0$ ,  $S_1$  и  $S_2$ . Уровни на этих входах будут определять, какие данные будут появляться на выходе  $F$ .

Составим таблицу истинности для функции, которую нужно реализовать (таблица 10.2). Читатель для примера может сформировать свою функцию.

В приведенной таблице истинности единица на выходе появляется в случае поступления наборов 010, 011 и 110. В остальных пяти случаях на выходе должен быть 0. В соответствии с таблицей входы  $I_0$ ,  $I_1$ ,  $I_4$ ,  $I_5$  и  $I_7$  должны быть подключены к источнику с напряжением, соответствующим логическому нулю (рис. 10.5). Их можно просто подключить к общему проводу («земле»). Напряжение с уровнем логической единицы должно поступать на выход с входов  $I_2$ ,  $I_3$  и  $I_6$ .

Таблица 10.2

Таблица для синтеза формирователя

Input	C, S2	B, S1	A, S0	F
I0	0	0	0	0
I1	0	0	1	0
I2	0	1	0	1
I3	0	1	1	1
I4	1	0	0	0
I5	1	0	1	0
I6	1	1	0	1
I7	1	1	1	0

Уровень напряжения, соответствующий логической единице можно получить непосредственно от источника питания.

На основе таблицы запишем дизъюнктивную форму булевой функции

$$F = \bar{C} \& B \& \bar{A} + \bar{C} \& B \& A + C \& B \& \bar{A}.$$

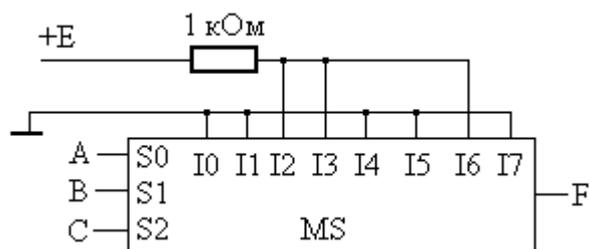


Рис. 10.5. Получение булевой функции с помощью мультиплексора

Работу устройства, работающего в соответствии с функцией, представленной таблицей истинности и булевым выражением, можно наблюдать по диаграмме, рис. 10.6, полученной компьютерным моделированием<sup>21</sup>. Линия Nul на диаграмме отображает уровень логического сигнала, необходимого для задания нуля на входах I0, I1, I4, I5, I7 и инверсного ему значения на входах I2, I3 и I6.

С помощью такого мультиплексора может быть реализована любая таблица истинности для функции трех аргументов.

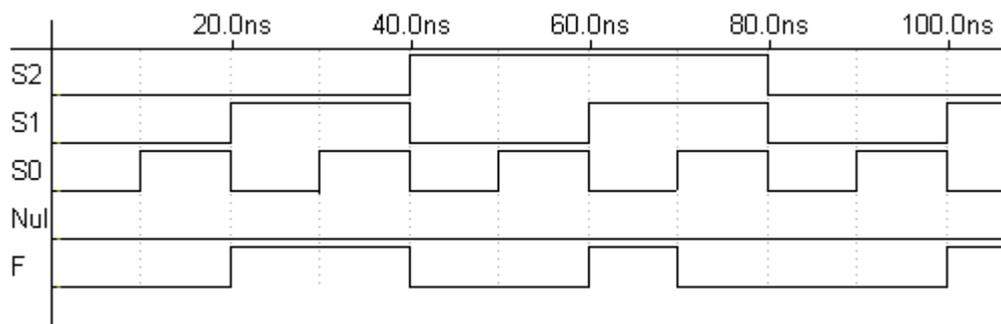


Рис. 10.6. Диаграмма логических уровней на входах аргументов S0, S1, S2 и выходе F

Рассмотрим пример преобразования параллельного двоичного кода в последовательный (рис. 10.7). Устройство должно передавать на выход значения разрядов кода один за другим, начиная с младшего или со старшего разряда. Для этого на выход мультиплексора должны подключаться его входы в порядке возрастания или убывания весов разрядов.

<sup>21</sup> Для моделирования использовалась программная система MAX+PlusII.

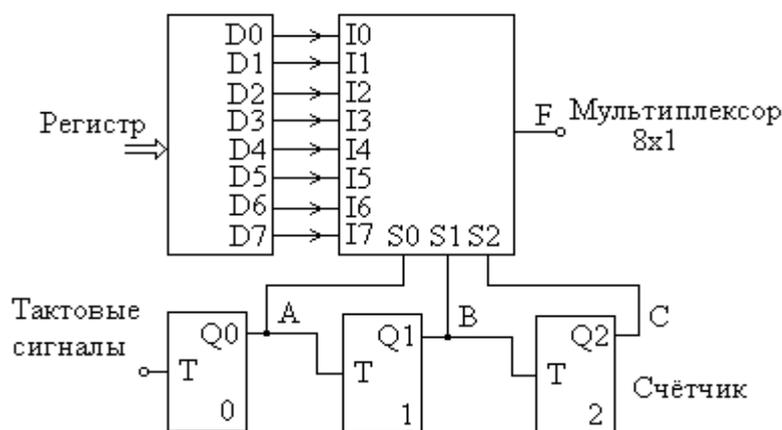


Рис. 10.7. Преобразование параллельного двоичного кода в последовательный код

К сигнальным входам мультиплексора необходимо подключить регистр, предназначенный для хранения исходного кода в течение времени преобразования, а на управляющие входы подавать сигналы со счетчика тактов.

Если мультиплексор имеет восемь входов, то для преобразования одного байта потребуется восемь тактов.

Трехразрядный счетчик преобразует последовательность тактовых импульсов в последовательность сигналов на управляющих входах мультиплексора. Каждому такту работы устройства соответствует свое состояние входов  $S_0$ ,  $S_1$  и  $S_2$  и появление очередного бита на выходе  $F$ . Диаграмма выходных логических уровней приведена на рис. 10.8.

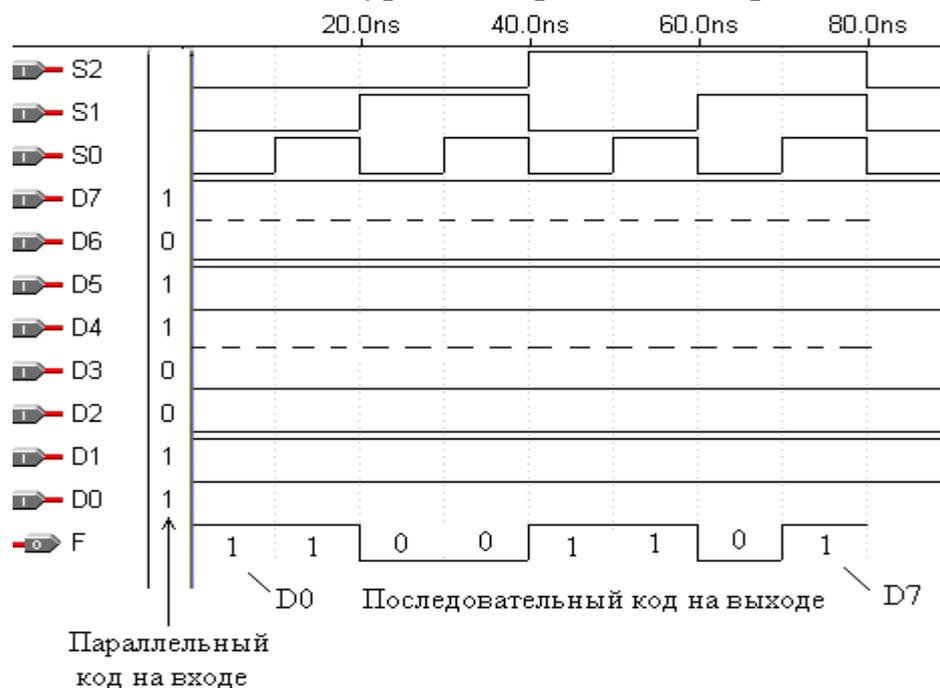


Рис. 10.8. Диаграмма сигналов на входах и выходе мультиплексора при преобразовании параллельного восьмиразрядного двоичного кода в последовательный код

## 11. ПРОВЕРКА КОДОВ НА ЧЕТНОСТЬ

При работе вычислительных машин происходит передача кодов из одних устройств в другие. Линии передачи подвергаются действию помех, создаваемых различными системами. Сигналы, являющиеся помехами, могут быть импульсными, синусоидальными или шумовыми. В силу действия помех всегда имеется вероятность появления ошибок в принимаемом сигнале. Если приемник воспринял принятый сигнал за логическую единицу, а передатчик передавал на самом деле 0, то в силу того, что неизвестно, что передавал приемник, ошибка останется незамеченной. По этой причине во многих цифровых системах используются методы обнаружения ошибок [20]. Одним из наиболее простых методов является метод проверки кодов на четность.

При проверке кодов на четность к передаваемой кодовой группе разрядов добавляется специальный разряд, в который на передающей стороне записывается 0 или 1. Этот разряд кода обычно называют битом четности. Его значение определяется количеством единиц, имеющих в передаваемой кодовой группе.

Если число единиц в коде нечетное, то в дополнительный разряд записывается 1. В целом получающаяся кодовая посылка содержит четное число единиц.

Если кодовая группа содержит четное количество единиц, то в дополнительный разряд записывается 0. Такой способ часто называют «положительная четность».

Значение специального разряда проверяется на принимающей стороне и сравнивается с количеством единиц в полученной группе разрядов. Проверяется не само количество единиц, а делается анализ: четно это количество или нет. Результат проверки сопоставляется со значением сигнала, поступившего в дополнительном разряде.

Если при передаче произойдет изменение одного из разрядов, то количество единиц в принятых вместе слове и дополнительном разряде, нарушится. На выходе схемы контроля четности сформируется единичный сигнал, указывающий на ошибку при передаче. В этом случае можно воспользоваться повторной передачей слова.

Если при передаче возникло изменение в двух разрядах, то четность количества единиц сохраняется [26] и передача будет считаться безошибочной. Следует учитывать, что вероятность возникновения двух или более ошибок гораздо ниже, чем одной.

Проверку удобно делать с помощью устройства, реализующего функцию суммирования по Mod2 (функция  $f_6$ ). Эта функция, как показано ранее, записывается в виде  $F = \overline{X_0} \cdot X_1 + X_0 \cdot \overline{X_1}$ .

## 12. ОСНОВНЫЕ УСТРОЙСТВА

Для выполнения вычислений и обработки информации необходимы два основных устройства: процессор и согласованное с процессором устройство для хранения информации. Для расширения возможностей вычислительной машины и удобства работы с ней, кроме устройства хранения информации и процессора, в ее состав входят устройства ввода и вывода и дополнительные устройства памяти. Основные устройства (процессор и память) часто называют центральной частью, остальные устройства относят к периферии. Центральная часть и периферия обмениваются сигналами управления и кодами через систему управления вводом и выводом.

Процессор содержит два основных устройства: центральное устройство управления и арифметико-логическое устройство (АЛУ).

Запоминающее устройство, работающее совместно с процессором, относится к устройствам, позволяющим работать с каждым отдельным кодом команды и каждым отдельным кодом обрабатываемого числа (операндом). Это устройство называется оперативным запоминающим устройством (ОЗУ), иногда основной оперативной памятью.

Устройства ввода и вывода позволяют вводить информацию, редактировать ее и отображать в виде твердой копии или оперативно на экране.

Дополнительная периферийная память используется для увеличения объема хранимой информации, долговременного ее хранения, в том числе когда компьютер выключен. Устройства периферийной памяти называются долговременными запоминающими устройствами (ДЗУ) или накопителями.

Совокупность устройств, предназначенных для хранения информации, образует систему взаимосвязанных запоминающих устройств – память ЭВМ.

Устройства управления и программные средства управления образуют систему управления.

## ПРИЛОЖЕНИЯ

### Приложение 1

#### Теорема Поста – Яблонского

Теорема. *Чтобы система функций  $\{f_1, f_2, \dots, f_m\}$  была полной, необходимо и достаточно, чтобы она содержала следующие функции: не сохраняющую константу ноль, не сохраняющую константу единица, не являющуюся самодвойственной, не являющуюся линейной, не являющуюся монотонной.*

### Приложение 2

#### Основные свойства операций в алгебре Жегалкина

В алгебре Жегалкина используются операции конъюнкции, суммы по модулю 2 и константы 1. Основными свойствами этих операций являются:

- коммутативность,  $X \oplus Y = Y \oplus X$ ;
- ассоциативность,  $(X \oplus Y) \oplus Z = X \oplus (Y \oplus Z)$ ;
- дистрибутивность,  $X(Y \oplus Z) = (XY) \oplus (XZ)$ .

В качестве аксиом в алгебре Жегалкина принимаются следующие соотношения:  $X1 = X$ ,  $X \oplus 0 = X$ ,  $X \oplus X = 0$ ,  $XX = X$ .

Связь между алгеброй Жегалкина и алгеброй, построенной на операциях конъюнкции, дизъюнкции и отрицания (алгебра Буля), устанавливается с помощью следующих соотношений:

$$X + Y = X \oplus Y \oplus XY, \quad \bar{X} = X \oplus 1, \quad X \oplus Y = \bar{X}Y + X\bar{Y}.$$

Эти выражения легко проверить, воспользовавшись правилами, приведенными ранее (в разделе **3.3.Основные свойства логических функций**). Выражение  $X \oplus Y = \bar{X}Y + X\bar{Y}$  – это функция f6 из табл. 3.2.2.

### Приложение 3

#### Классы логических функций

	В алгебре Буля	$K_0 K_1 S M L$	В алгебре Жегалкина
f0	0	+ - - + +	0
f1	$X_1 X_2$	+ + - + -	$X_1 X_2$
f2	$X_1 \bar{X}_2$	+ - - - -	$X_1 \oplus X_1 X_2$
f3	$X_1$	+ + + + +	$X_1$
f4	$\bar{X}_1 X_2$	+ - - - -	$X_2 \oplus X_1 X_2$
f5	$X_2$	+ + + - +	$X_2$
f6	$\bar{X}_1 X_2 + X_1 \bar{X}_2$	+ - - - +	$X_1 \oplus X_2$

f7	$X_1\bar{X}_2 + \bar{X}_1X_2 + X_1X_2$	+ + - + -	$X_1 \oplus X_2 \oplus X_1X_2$
f8	$\bar{X}_1 + X_2$	- - - - -	$1 \oplus X_1 \oplus X_2 \oplus X_1X_2$
f9	$X_1X_2 + \bar{X}_1\bar{X}_2$	- + - - +	$1 \oplus X_1 \oplus X_2$
f10	$\bar{X}_2$	- - + - +	$1 \oplus X_2$
f11	$\bar{X}_1\bar{X}_2 + X_1\bar{X}_2 + X_1X_2$	- + - - -	$1 \oplus X_1 \oplus X_1X_2$
f12	$\bar{X}_1$	- - + - +	$1 \oplus X_1$
f13	$\bar{X}_1\bar{X}_2 + \bar{X}_1X_2 + X_1X_2$	- + - - -	$1 \oplus X_2 \oplus X_1X_2$
f14	$\bar{X}_1\bar{X}_2$	- - - - -	
f15	1	- + - + +	1

Так как два элемента (f8 и f14) из рассмотренной таблицы обладают системой свойств, присущих полной системе, то любым из этих двух элементов можно пользоваться для построения других элементов. Наряду с таким подходом по соображениям стоимости, устранения избыточности используют и другие варианты. Удобно воспользоваться таблицей дополнений до полной системы. В левой колонке этой таблицы записана функция, а в правой указывается дополняющая функция [22].

Таблица дополняющих функций (элементов)

Функция (логический элемент)	Обозначение	Дополняющий элемент
Инвертор	НЕ	ИЛИ либо И
Схема ИЛИ	ИЛИ	НЕ
Схема И	И	НЕ
Схема импликации	ЕСЛИ	Нулевой
Схема запрета	НЕТ	Единичный
Схема равнозначности	~	Нулевой, И либо ИЛИ
Схема неравнозначности	$\oplus$	Единичный, И либо ИЛИ

#### Приложение 4

### Синтез цифровых автоматов: проектирование синхронных счетчиков

#### Пошаговая инструкция к структурному синтезу счетчиков

**Шаг 1.** Выбираем элементную базу, например *JK* триггер.

**Шаг 2.** Составляем таблицу возбуждений для выбранного типа триггера. Сигнал, значение которого безразлично, то есть результат не зависит от того, имеет этот сигнал значение 0 или значение 1, будем обозначать буквой Б.

Таблица возбуждений для JK триггера

Изменение выходного сигнала	Текущее состояние Q(n)	Следующее состояние Q(n + 1)	Необходимый сигнал на входе J	Необходимый сигнал на входе K
0 → 0	0	0	0	Б
0 → 1	0	1	1	Б
1 → 0	1	0	Б	1
1 → 1	1	1	Б	0

Если выбран D триггер, надо составить таблицу возбуждений для D триггера.

**Шаг 3.** Определение необходимого количества разрядов.

Разработаем для примера счетчик с 5 состояниями: 000, 001, 010, 011, 100. Для пятого состояния (100) необходимо 3 разряда. Далее состояния разрабатываемого счетчика повторяются: 000, 001 и т.д.

**Шаг 4.** Составление диаграммы переходов, отражающей все возможные состояния [20], том числе и те, которые не входят в состав заданных 5 состояний, Рис. П4.1.

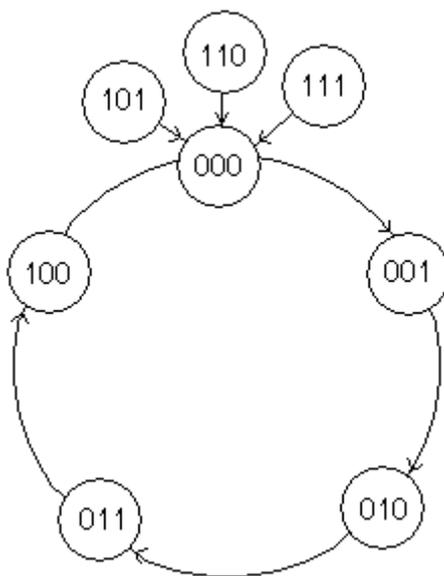


Рис. П4.1. Диаграмма переходов счетчика по Mod5

Не обязательно диаграмма состояний должна быть круговой. Можно даже в простых случаях сразу перейти к составлению таблицы переходов, отражающей текущие и последующие состояния, пропустив рисование диаграммы.

**Шаг 5.** Составление таблицы по диаграмме переходов.

Должны быть перечислены все возможные текущие состояния и последующие состояния.

В левой части таблицы отображаются *все возможные* состояния. Эти состояния обозначаются как текущие состояния. В правой части таблицы записываются все последующие состояния для каждого текущего состояния. Последующие состояния могут быть получены, в частности, из диаграммы переходов (см. шаг 4). Например, первая строка показывает, что у текущего состояния 000 последующим состоянием является 001. Пятая строка показывает, текущее состояние 100, для которого последующее состояние 000. В шестой, седьмой и восьмой строках записаны все три нежелательных состояния: 101, 110 и 111. **Для этих состояний последующим состоянием является состояние 000.**

Таблица текущих и последующих состояний

Номер строки	Текущее состояние ( $t_i$ )			Последующее состояние ( $t_{i+1}$ )		
	разр. 2	разр. 1	разр. 0	разр. 2	разр. 1	разр. 0
1	0	0	0	0	0	1
2	0	0	1	0	1	0
3	0	1	0	0	1	1
4	0	1	1	1	0	0
5	1	0	0	0	0	0
6	1	0	1	0	0	0
7	1	1	0	0	0	0
8	1	1	1	0	0	0

### Шаг 6.

Добавить столбцы к таблице текущих и последующих состояний со значениями для каждого входа  $J$  и для каждого входа  $K$ . Поскольку в рассматриваемом примере речь идет о 3 триггерах типа JK, следует добавить 6 столбцов (3 столбца для входов  $J_i$  и 3 столбца для входов  $K_i$ ). Эту полную таблицу будем называть *таблицей возбуждений* триггеров. Содержимое последних 6 столбцов указывает состояния входов каждого триггера (то есть необходимые сигналы на входах  $J$  и  $K$ ). Из левой части таблицы определяется, с какого триггера можно взять эти сигналы. При этом получившиеся выражения могут потребовать сложных устройств. Для упрощения схемы может потребоваться минимизация получившихся булевых выражений.

Таблица возбуждений триггеров разрядов счетчика (схемы)

Номер строки	Текущее состояние ( $t_i$ )			Последующее состояние ( $t_{i+1}$ )			Сигналы, которые необходимо подать на входы					
	разр. 2	разр. 1	разр. 0	разр. 2	разр. 1	разр. 0	$J_2$	$K_2$	$J_1$	$K_1$	$J_0$	$K_0$
1	0	0	0	0	0	1	0	Б	0	Б	1	Б
2	0	0	1	0	1	0	0	Б	1	Б	Б	1

3	0	1	0	0	1	1	0	Б	Б	0	1	Б
4	0	1	1	1	0	0	1	Б	Б	1	Б	1
5	1	0	0	0	0	0	Б	1	0	Б	0	Б
6	1	0	1	0	0	0	Б	1	0	Б	Б	1
7	1	1	0	0	0	0	Б	1	Б	1	0	Б
8	1	1	1	0	0	0	Б	1	Б	1	Б	1

Проделаем анализ первой строки таблицы. Текущее состояние 000 переходит в последующее состояние 001 при поступлении тактового импульса. В этот момент триггер второго разряда переходит из состояния 0 в состояние 0. Из таблицы возбуждений  $JK$  триггера (см. шаг 2) следует, что для такого перехода  $J_2$  должен быть в состоянии 0 (на  $J_2$  должен быть подан «лог. 0»). При этом на  $K_2$  можно подать любой уровень сигнала (безразлично, «лог. 0» или «лог. 1»). В триггере первого разряда осуществляется также переход  $0 \rightarrow 0$ ). Поэтому  $J_2 = 0$ , а  $K_2 = Б$ . Триггер нулевого разряда переключается из состояния 0 в состояние 1. Для осуществления такого перехода необходимо, чтобы выполнялись условия  $J_0 = 1$ , а  $K_0 = Б$ .

Аналогичные рассуждения необходимо применить ко второй и третьей строкам.

Четвертая строка описывает переход из текущего состояния 011 в последующее состояние 100, при этом триггер второго разряда переходит из лог 0 в лог. 1. Обращаемся к таблице возбуждений  $JK$  триггера. Из нее следует, что для осуществления перехода надо на вход  $J_2$  подать 1, а  $K_2$  может быть в любом состоянии, то есть  $J_2 = 1$ ,  $K_2 = Б$ .

Триггеры нулевого и первого разрядов переходят из 1 в 0. В соответствии с таблицей возбуждений  $J = Б$ , а  $K = 1$ .

Таким образом, мы получили логические уровни, которые следует подавать на входы триггеров.

**Шаг 7.** Разработка логических схем, с помощью которых подаются сигналы на входы  $J$  и  $K$ .

В последней таблице перечислены 6 входов  $J$  и  $K$  ( $J_0, K_0, J_1, K_1, J_2$  и  $K_2$ ). Необходимо рассмотреть каждый из них, как выход логической схемы, на вход которой подключены выходы разрядов (второго, первого и нулевого). Другими словами, например для нулевого разряда, надо записать с какого разряда (и с какого выхода) следует подать сигнал, на вход  $J_0$  и на вход  $K_0$ , чтобы получить заданное состояние нулевого разряда. Необходимо составить схему для каждого входа.

Рассмотрим разработку логических схем для нулевого разряда.

Сначала необходимо рассмотреть текущие состояния всех разрядов, а также требуемые логические уровни, которые необходимо подавать на вход  $J_0$  в каждом случае.

Эту информацию можно выделить из последней таблицы и представить в виде таблицы истинности. Эта таблица будет показывать требуемые логические уровни на входе  $J_0$  в каждом случае. Переписываем в таблицу весь столбец  $J_0$  и в каждую строку соответствующее текущее состояние. Как было отмечено выше, в некоторых случаях безразлично, какой уровень будет подан на вход  $J_0$ . Для разработки логической схемы для входа  $J_0$  следует сначала определить, из полученной таблицы истинности, его зависимость от состояний выходов триггеров 0, 1 и 2. Для этого можно дополнительно воспользоваться картой Карно и сделать упрощения.

Таблица истинности (часть таблицы возбуждений)

Текущие состояния			$J_0$
разряд 2 (C)	разряд 1 (B)	разряд 0 (A)	
0	0	0	1
0	0	1	Б
0	1	0	1
0	1	1	Б
1	0	0	0
1	0	1	Б
1	1	0	0
1	1	1	Б

Поскольку разрядов 3, в карте Карно должно быть 8 ячеек. Перенесим данные в карту Карно. Разметим карту с учетом того, что в ней не должно быть одинаковых ячеек. Для удобства работы с картой вместо номеров разрядов используем латинские буквы A, B и C, начиная с младшего.

Карта Карно

$\bar{B}\bar{C}$	1	Б
$\bar{B}C$	0	Б
$BC$	0	Б
$B\bar{C}$	1	Б
	$\bar{A}$	A

В этой карте только 2 единицы (подкрашенные клетки). Безразличные состояния – Б обозначим как 1. Тогда получаем  $J_0 = \bar{C}$ .

Обратим внимание, что в таблице возбуждений для  $K0$  нет нулей. Безразличные состояния в колонке для  $K0$  заменим 1. Тогда получается, даже не записывая карту Карно, что  $K0$  всегда равно 1.

Составим карты для первого и второго разрядов.

Для  $J2$  используем две последние ячейки в колонке  $A$ . Заменяя  $B$  на 1, получаем  $J2 = AB$ .

$\bar{B}\bar{C}$	0	0
$\bar{B}C$	Б	Б
$BC$	Б	Б
$B\bar{C}$	0	1
	$\bar{A}$	$A$

Для  $K2$ , записав вместо Б единицы, получаем  $K2 = 1$

$\bar{B}\bar{C}$	Б	Б
$\bar{B}C$	1	1
$BC$	1	1
$B\bar{C}$	Б	Б
	$\bar{A}$	$A$

Для  $J1$ , записав в правую нижнюю ячейку единицу, получаем  $A\bar{C}$ .

$\bar{B}\bar{C}$	0	1
$\bar{B}C$	0	0
$BC$	Б	Б
$B\bar{C}$	Б	Б
	$\bar{A}$	$A$

Для  $K1$  поставим 1 в правую верхнюю ячейку и во вторую строку таблицы. Получаем  $K1 = A + C$ .

$\bar{B}\bar{C}$	Б	Б
$\bar{B}C$	Б	Б
$BC$	1	1
$B\bar{C}$	0	1
	$\bar{A}$	$A$

**Шаг 8.** Эти выражения, полученные с помощью карт Карно, не упрощаются, поэтому переходим к рисованию схемы по этим выражениям.

В соответствии с логическими выражениями для  $K1$  используем функцию дизъюнкции, а для  $J1$  и  $J2$  – функцию конъюнкции. Для  $J0$  необходимо использовать инверсию  $C$ . На входы  $K0$  и  $K2$  в соответствии с полученными значениями постоянно подается лог. 1. Полученная схема представлена на рис. П.4.2, а диаграмма состояний на рис. П.4.3.

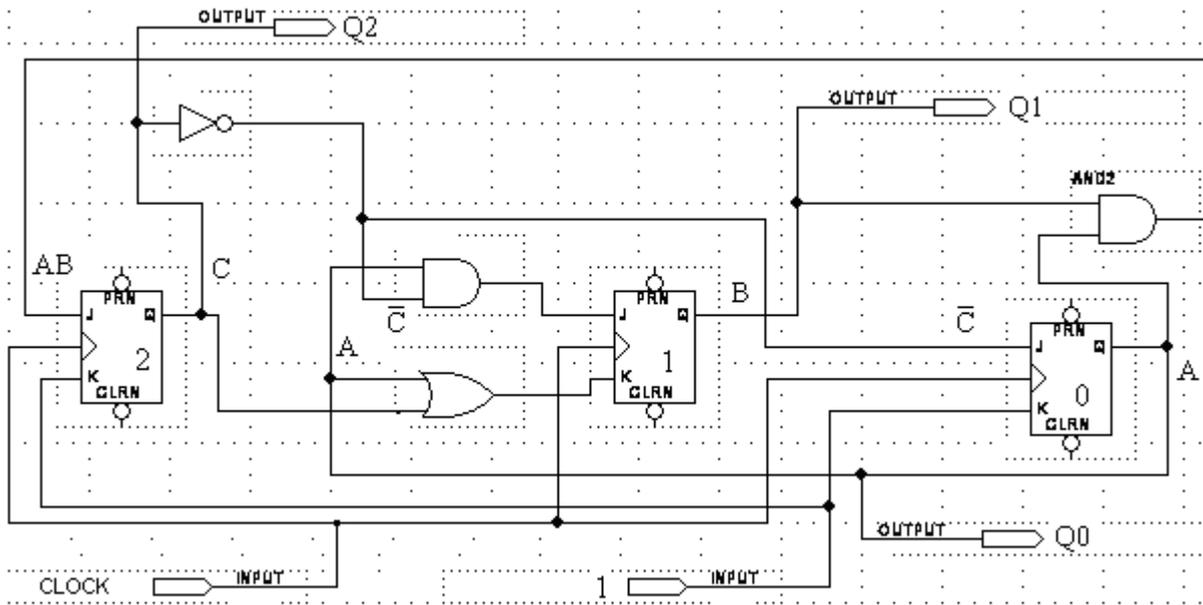


Рис. П.4.2. Схема счетчика по Mod5

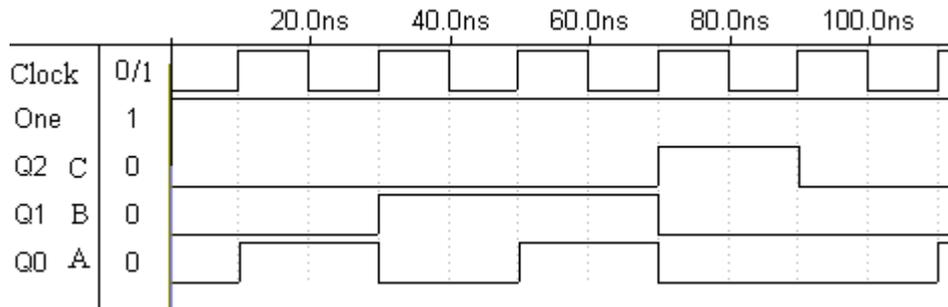


Рис. П4.3. Диаграмма состояний счетчика по Mod5

## Приложение 5

Применение схем И-ИЛИ-НЕ для построения триггеров.

Построить *RS* триггер можно на элементах, реализующих функцию И-ИЛИ-НЕ. В число функций такого элемента входит отрицание (НЕ). Таким образом, двух экземпляров элемента достаточно для построения асинхронного *RS* триггера. Ранее, в основном тексте, были рассмотрены такие триггеры на элементах ИЛИ-НЕ и на элементах И-НЕ. Понятно, что предлагаемый элемент обладает функциональной избыточностью.

Рассмотрим рис. П.5.1.

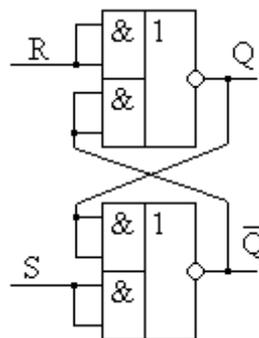


Рис. П.5.1. Асинхронный RS триггер на клапанах И-ИЛИ-НЕ

Лишние входы в этой схеме объединены с действующими. Такое объединение, однако, может быть нежелательным. Если имеется опасность перегрузки источников сигналов  $R$  и  $S$ , то следует избыточные входы присоединить к источнику постоянного напряжения, имеющего уровень логической единицы. Такое включение одновременно уменьшит время переключения триггера. Читателю предоставляется возможность самому начертить эту схему.

Элементы, реализующие логическую функцию И-ИЛИ-НЕ, позволяют построить также синхронный одноклапчатый  $RS$  триггер. Используя избыточность количества входов, добавим не только вход синхронизации, но и входы асинхронной установки триггера. Схема триггера представлена на рис. П.5.2.

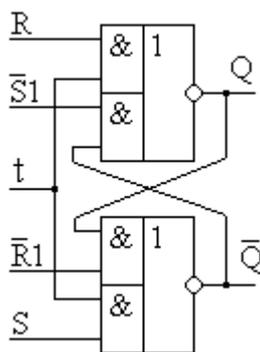


Рис. П.5.2. Синхронный одноклапчатый триггер на элементах И-ИЛИ-НЕ

На рисунке асинхронные входы обозначены  $\bar{R}_1, \bar{S}_1$ . Для синхронной работы триггера на эти входы необходимо подать напряжения с уровнем логической единицы.

Сочетание сигналов  $RS = 1$  в этой схеме запрещено. Иначе говоря, необходимым является выполнение условия  $RS = 0$ . Синхронная установка в 1 осуществляется подачей единицы на вход  $S$  (Если  $S \& t = 1$ , то  $Q = 1$ ).

## Приложение 6

### Системы счисления и экономичность оборудования

Позиционные системы счисления имеют ряд серьезных преимуществ перед непозиционными системами. Основными из этих преимуществ являются удобство выполнения таких арифметических операций, как сложение, вычитание, умножение, деление, извлечение корней и др. [3]

Привычная в обиходе десятичная позиционная система счисления не является наиболее удобной для реализации в цифровых машинах. Любой из символов, применяемых в цифровых машинах для записи чисел, должен отображаться в виде одного или нескольких состояний некоторого технического элемента. Эти состояния должны быть устойчивы и четко различимы. Элемент с десятью четко различимыми состояниями является сложным техническим устройством, при работе с которым состояние

определяется путем измерения. Напротив, имеется достаточно много устройств с двумя четко различимыми состояниями. В этих элементах различие между отдельными фиксированными состояниями имеет качественный, а не количественный характер. Поэтому запоминание чисел на двоичных устройствах может быть реализовано достаточно надежно. Простота и надежность явились главными причинами широкого применения двоичных элементов и двоичной системы счисления в цифровой технике.

Несколько подробнее остановимся на экономичности двоичной системы. Экономичность той или иной системы счисления элементов часто оценивают произведением количества цифр на количество разрядов, необходимых для отображения натурального ряда целых чисел в пределах от 0 до  $N$ . Для выбора оценки с помощью количества таких *цифро-разрядов* имеется несколько причин. Так, в ряде случаев оборудование, необходимое для хранения в цифровой машине одной цифры, пропорционально значению основания системы счисления. Так, например, в первых вычислительных машинах, работавших на основе десятичной системы счисления, для хранения одной десятичной цифры, применяли десять двоичных запоминающих элементов (триггеров). Если уменьшать основание системы счисления, то число триггеров, необходимое для хранения одной цифры, в рассматриваемом случае будет пропорционально уменьшаться. Исключение составляет двоичная система счисления, позволяющая реализовать надежное хранение одной цифры на одном триггере, а не на двух. Оборудование, необходимое для хранения любого числа от 0 до  $N$ , пропорционально произведению количества цифр, используемых в системе счисления на количество разрядов числа.

Таким образом, число цифро-разрядов характеризует объем оборудования цифровой машины и ее быстродействие.

При основании системы счисления  $P$  и числе разрядов  $n$  могут быть представлены  $N = P^n$  различных целых чисел в пределах от 0 до  $P^n - 1$ . Для заданного количества чисел  $N$  количество цифро-разрядов при основании  $P$  равно

$$pn = p \log_p N = \frac{P}{\log_N p} .$$

Дифференцируя приведенное выражение по основанию  $p$ , и приравнявая производную нулю, получаем минимальное значение при  $P = e = 2,718\dots$  Конечно значительно удобнее, когда основание системы счисления является целым, а не дробным числом. Пусть  $N = 10$ , а основания  $P$  – целые. Тогда, подсчитывая  $\frac{P}{\log_{10} P}$ , получаем следующие значения:

$P$	2	3	4	5	8	10	12	16
$\frac{P}{\log_{10} P}$	6,64	6,29	6,64	7,15	8,86	10	11,12	13,29

Выбор основания 10 не влияет на соотношение чисел во второй строке этой таблицы. При изменении основания все эти числа изменились бы пропорционально. Если все оборудование для хранения одного разряда троичной системы было бы в  $3/2$  раза больше, чем оборудование для хранения одного разряда двоичной системы, то в целом троичная машина оказалась бы на 5% экономичнее двоичной машины. Пять процентов получается при сравнении столбиков с весом 4 и с весом 3,

$$\frac{5,64 - 5,29}{5,29} \approx 0,055.$$

Однако элемент, физическая природа которого позволяет выделить три четко различимых состояния, оказывается менее надежным, чем двоичный элемент. Это означает, что оборудование для хранения одного троичного разряда приходится увеличить не в  $3/2$  раза, а в 2 или в 3 раза в сравнении с оборудованием для реализации двоичной системы. Иначе говоря, для надежной реализации троичного разряда надо два (двоичное изображение троичной цифры) или даже три триггера (изображение «одинарным» кодом). Если в процессе развития вычислительной техники появятся существенно троичные элементы, то рассуждения, приведенные в этом абзаце, придется пересмотреть.

## Приложение 7

### Компоненты, не рассмотренные в основной части текста

Развитие вычислительной техники происходит по нескольким направлениям, основными из которых являются производительность, количество обрабатываемой и хранимой информации, надежность и т.п. Достигновения в этих направлениях, в частности, обеспечиваются совершенствованием элементной базы.

Элементная база вычислительных средств на протяжении всего времени их развития претерпела ряд изменений. Так, электровакуумные приборы сменились полупроводниковыми, от дискретных компонентов произошел переход к интегральным микросхемам, интегральные микросхемы на биполярных транзисторах вытесняются микросхемами, выполненными на полевых транзисторах. При этом развитии, не все идеи, предлагаемые исследователями и разработчиками, находят практическое применение. Это естественно, так как серийные изделия должны удовлетворять большому количеству требований: технологических, надежных, стоимостных и других. Однако с развитием технологий, применением других исходных материалов, проведением дополнительных исследований некоторые идеи становятся реализуемыми и находят практическое применение.

В связи со сказанным выше, рассмотрим два устройства, которым не уделено внимание в основном тексте. Это параметрон и криотрон.

### **Параметрон**

Параметроном называется колебательный контур, в котором осуществляется периодическое изменение значения одного из параметров: индуктивности, емкости или сопротивления. Изменение значения параметра осуществляется с помощью внешнего сигнала. Этот процесс называется накачкой [29].

Если контур имеет частоту собственных колебаний  $f_0$ , то частота накачки  $f_n$  выбирается приблизительно в два раза выше. Под действием этого сигнала происходит периодическое изменение значения одного из энергоемких параметров контура. При этом в контуре возникают колебания, когерентные по отношению к колебаниям сигнала накачки. Существенным с точки зрения применения в цифровых системах является свойство фазы этих параметрических колебаний. Фаза возбужденных колебаний может принимать одно из двух отличающихся на  $180^\circ$  значений. Это значение фазы может сохраняться сколь угодно долго, пока подается сигнал накачки.

Поскольку в рассматриваемом случае, возможны только два стабильных значения фазы, параметрон может применяться как логический элемент или как запоминающая ячейка. На основе параметронов были созданы не только запоминающие элементы, но и регистры, сумматоры и элементы управления<sup>22</sup>.

С точки зрения теории цепей параметрон является нелинейной резонансной системой. По типу нелинейного элемента различают индуктивные, емкостные и резистивные параметроны. В индуктивных параметронах применяют катушки с ферритовыми сердечниками или магнитную пленку. Емкостные параметроны строятся на параметрических полупроводниковых диодах или сегнетоэлектрические конденсаторы. Резистивные параметроны содержат туннельные или другие полупроводниковые диоды с вольтамперной характеристикой, имеющей падающий участок.

Частота переключения параметрона (тактовая частота  $f_T$ ) пропорциональна частоте накачки и меньше ее приблизительно в 20–50 раз. Наименьшая тактовая частота наблюдается в индуктивных параметронах. Наивысшая тактовая частота достигается в резистивных параметронах. Индуктивные параметроны изготавливают также на тонких магнитных пленках.

В индуктивных параметронах на тонких магнитных пленках или в емкостных параметронах, выполненных на полупроводниковых диодах, тактовая частота достигает 150 МГц.

---

<sup>22</sup> В 1954 году параметрон был запатентован Э. Гото (Япония).

Наиболее надежными и дешевыми являются одноконтурные индуктивные параметроны, построенные на ферритовых сердечниках. Такие параметроны потребляют мощностью 15–50 МВт и работают на тактовой частоте,  $f_T = 100 \text{ кГц}$ . Более экономичны (3–6 МВт) емкостные параметроны на конденсаторах. Такие параметроны имеют более высокое быстродействие, достигающие  $f_T = 5 \text{ МГц}$ .

В настоящее время интерес к параметронам усиливается благодаря последним достижениям в области нанотехнологий.

### **Криотрон**

Устройство и работа криотронов основана на эффекте сверхпроводимости, то есть исчезновении электрического сопротивления у металлов при охлаждении их до температуры, называемой критической. Например, для ртути эта температура равна  $4,2^\circ \text{ К}$ . При достижении этой температуры, сопротивление резко падает, до столь малого уровня, что его не удается измерить чувствительными приборами [30].

Сверхпроводимость обнаруживается у металлов, которые в нормальных условиях являются не очень хорошими проводниками. Такие достаточно хорошие проводники, как медь серебро и золото в сверхпроводящее состояние не переходят. В зависимости от типа металла критические температуры лежат в диапазоне от  $0,6$  до  $8^\circ \text{ К}$ . Свойство скачкообразного изменения сопротивления наблюдается у таких металлов, как алюминий, индий, кадмий, ниобий, олово, ртуть, свинец и тантал.

Сверхпроводящее состояние может быть разрушено воздействием внешнего магнитного поля. Для этого его напряженность должна достигать некоторого критического значения  $H_k$ , своего для каждого типа сверхпроводника.

Рассмотренные свойства сверхпроводников положены в основу криотрона. Простейший криотрон выполняется в виде танталового стержня, длиной приблизительно 25 мм и диаметром 0,25 мм. На него наматывается однослойная обмотка, состоящая из 250 витков изолированной ниобиевой проволоки. Эта проволока имеет диаметр 0,08 мм. Устройство помещается в криостат с жидким гелием. Таким образом, поддерживается температура катушки, равная  $4,2^\circ \text{ К}$ . При такой температуре тантал и ниобий являются сверхпроводниками.

Если через катушку пропустить ток, равный приблизительно 300 мА, то, при указанном выше числе витков, создается магнитное поле напряженностью 32 А/см. При такой напряженности магнитного поля тантал скачком перейдет из состояния сверхпроводимости в состояние нормальной проводимости. Сопротивление танталового стержня при этом мгновенно изменится от нуля до некоторой конечной величины.

Направление тока в обеих цепях криотрона никак не влияет на его работу.

Для построения триггера управляющая обмотка одного криотрона соединяется с танталовым стержнем другого криотрона. Сопротивление обмотки при этом всегда должно оставаться равным нулю, независимо от изменяющегося магнитного поля. Иначе говоря, независимо от величины поля обмотка должна оставаться в сверхпроводящем состоянии. В связи с этим в качестве материала обмотки выбирается ниобий, имеющий высокую критическую температуру, равную  $8^{\circ}\text{К}$ . Кроме того, для ниобия характерна большая величина критической напряженности поля при рабочей температуре  $4,2^{\circ}\text{К}$ . Для криотрона принято условное графическое изображение, показанной на рис. П.7.1. Криотроны обозначены  $C_1$  и  $C_2$ , обмотки обозначены  $O_1$  и  $O_2$ .

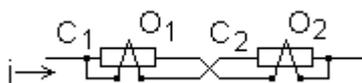


Рис. П.7.1. Схема с двумя устойчивыми состояниями, построенная на двух криотронах

Чтобы один криотрон мог управлять другим криотроном, надо, чтобы при пропускании по стержню тока  $i$ , создающего при протекании в обмотке другого криотрона критическое поле, сверхпроводимость в стержне первого криотрона не разрушалась.

Из рассмотрения схемы криотрона видно, что для протекания тока имеются две параллельные ветви. Если разрушить сверхпроводимость одного стержня и, тем самым, перевести в сверхпроводящее состояние другой стержень, то триггер окажется в другом состоянии. Произошедшее перераспределение токов в ветвях триггера, которое устойчиво сохраняется до нового внешнего воздействия.

Криотрон рассмотренной конструкции из-за наличия многовитковой обмотки является малотехнологичным. Кроме того, из-за большой индуктивности многовитковой катушки, он имеет существенную постоянную времени  $L/R$ .

Более производительным и технологичным является тонкопленочный криотрон [30]. Пленочные криотроны обладают очень малой индуктивностью и высоким сопротивлением. Переключаются пленочные криотроны за время менее микросекунды. Пленочные криотроны могут быть изготовлены по интегральной технологии с обеспечением высокой удельной плотности устройств хранения информации.

## Некоторые термины и сокращения

Able – название шестнадцатеричной цифры A (числовое значение 10).

Algol (Algorithmic Language) – Алгол, язык программирования высокого уровня.

ALU (Arithmetical and Logical Unit) – арифметико-логическое устройство.

APL (A Programming Language) – АПЛ, язык программирования высокого уровня.

ASCII (American Standard Code for Information Interchange) – американский стандартный код для обмена информацией. (Иногда произносится «АСКИ»).

B (binary) – двоичный, представленный в двоичной системе счисления

Вак (Backup) – резервная копия (иногда применяется в виде расширения имени файла).

Basic (Beginners All Purpose Symbolic Instruction Code) – Бейсик, простой для изучения и применения язык программирования, ориентированный на диалоговую работу.

BCD (Binary – Coded Decimal) – двоично-десятичный код.

Binary – двоичный.

Bit (Binary Digit) – разряд двоичного числа.

CP/M (Control Program for Microcomputers) – операционная система для микроЭВМ, написанная в командах микропроцессора Intel 8080.

D (decimal) – десятичный

DD (Double–Density Disk) – диск для записи с удвоенной плотностью.

Delay – задержка.

DS (Double–Sided Disk) – двухсторонняя дискета.

DOS (Disk Operating System) – дисковая операционная система, загружаемая с дисков и обеспечивающая работу с дисками для прикладных программ.

EEPROM (Electrically Erasable Programmable Read–Only Memory) – электрически стираемое программируемое постоянное запоминающее устройство, ЭСППЗУ.

EGA (Enhanced Graphics Adapter) – усовершенствованный графический адаптер – дисплейный адаптер для ПЭВМ, совместимый с IBM PC, обеспечивающий разрешение 640x350 точек с 156 цветами.

EI (Enable Interrupt) – разрешенное прерывание.

Erase – стирать (запись); разрушать (информацию) to erase a file from a disk — стирать файл на диске, удалять файл с диска

H (hexadecimal) – шестнадцатеричный

Hold time – время удержания, то есть наименьшее время, в течение которого данные должны быть установлены и стабильны после появления фронта тактового импульса

FIFO (First-In, First-Out) – стековая память обратного магазинного типа; дисциплина очереди «первым пришел – первым обслужен»

File – файл, упорядоченный набор записей или иная совокупность данных, хранящаяся в компьютерной системе под общим именем.

Fortran – процедурно ориентированный язык программирования высокого уровня

Implication – импликация; логическая операция, принимающая значение «ложь» только если первый аргумент истинен, а второй – ложен.

Karnaugh Map – карта Карно (используемая при минимизации булевых функций)

LIFO (Last In First Out) – «последним пришел – первым обслужен»; «звенья следуют в порядке, обратном порядку их поступления» (принцип стека); применяется в математике, информатике и экономике

Memory – память; запоминающее устройство

Minterm – элементарная конъюнктивная форма, минтерм

Minterm Form – дизъюнктивная нормальная форма, ДНФ

MS program – программа в магнитном ЗУ

O (octal) – восьмеричный, представленный в позиционной восьмеричной системе счисления

Pascal – процедурно ориентированный язык программирования высокого уровня

PIT – впадина, «кратер» (темное, неотражающее пятно на поверхности CD-ROM)

PROM (programmable read-only memory) – программируемая постоянная память, программируемое постоянное запоминающее устройство, ПЗУ

R (reset) – восстановление, возврат [возвращение] в исходное положение или состояние; установка в (состояние) «0»

RAM (random-access memory) – память [запоминающее устройство] с произвольной выборкой, ЗУПВ; оперативное запоминающее устройство, ОЗУ

Random – случайный; произвольный; нерегулярный

ROM (read-only memory) – ПЗУ, постоянная память; постоянное запоминающее устройство

S (set) – ставить, устанавливать

Set-Up Time – время предустановки, задержка сигнала синхронизации по отношению к моменту установки информационного сигнала

Supercomputer – суперкомпьютер, суперЭВМ – термин, означающий класс наиболее мощных из существующих компьютеров. Суперкомпьютеры используются, как правило, для решения научных задач, моделирования, в компьютерной графике и т. п.

Switch – переключатель

## Приложение 9

### Персоналии

**Брусенцов Николай Петрович** (р. 7 февраля 1925) – возглавлял разработку троичной ЭВМ «Сетунь» (1958), заслуженный научный сотрудник МГУ. Кандидат технических наук. В 1970 г. его группой разработчиков была сделана модернизированная машина «Сетунь-70». Всего 24 команды обеспечивали полную универсальность, эффективность и удобство программирования. Производство вычислительных машин Сетунь осуществлялось Казанским заводом математических машин.

**Буль Джордж** (1815–1864), английский математик и логик, один из основоположников математической логики; разработал алгебру логики, основу описания функционирования цифровых вычислительных машин.

**Лебедев Сергей Алексеевич** (20 октября (2 ноября) 1902 – 3 июля 1974) – основоположник вычислительной техники в СССР, директор ИТ-МиВТ, академик АН СССР (1953) и АН УССР (12.02.1945) – разработчик МЭСМ (Малой электронной счетной машины), первой ЭВМ в СССР и континентальной Европе, основатель советской компьютерной промышленности.

**Нейман** (Джон фон Нейман) (1903–1957) – венгеро-американский математик сделавший важный вклад в квантовую физику, квантовую логику, функциональный анализ, теорию множеств, информатику, экономику и другие отрасли науки. Наиболее известен как автор современной архитектуры компьютеров (так называемая архитектура фон Неймана), применением теории операторов к квантовой механике (алгебра фон Неймана), а также как создатель теории игр и концепции клеточных автоматов

**Пост Эмиль Леон** (Post Emil Leon) (11.02.1897— 21.04.1954) — американский математик и логик; один из основателей многозначной логики (1921); основные труды по математической логике: *алгебра Поста, классы Поста* функций алгебры логики; предложил абстрактную вычислительную машину – машину Поста.

**Рамеев Башир Искандерович** (01.05.1918 – 16.05.1994) – советский ученый-изобретатель, разработчик первых советских ЭВМ (Стрела, Урал-1). Под его руководством были разработаны арифметическое устройство и память на магнитном барабане. Машина «Стрела» была первой ЭВМ, освоенной в промышленном производстве в СССР.

В 1957 г. в Пензе был начат выпуск ЭВМ «Урал-1». Главным конструктором являлся Б.И. Рамеев.

**Тьюринг Алан Мэтисон** (Alan Mathison Turing) (23.06.1912 – 07.06.1954) – английский математик, логик, криптограф, оказавший существенное влияние на развитие информатики. В 1936 г. им предложена абстрактная вычислительная «Машина Тьюринга», которую можно считать моделью компьютера общего назначения, и позволившая формализовать понятие алгоритма.

**Шеннон Клод Элвуд** (30.04.1916 – 24.02.2001) – американский ученый и инженер. Один из создателей математической теории информации

**Яблонский Сергей Всеволодович** (06.12.1924, Москва) – институт прикладной математики АН СССР. Специалист в области дискретной математики и математических вопросов кибернетики. Основные труды относятся к исследованию общих вопросов теории управляющих систем, вопросам контроля и надежности управляющих систем, изучению алгоритмических трудностей синтеза управляющих систем.

## ЛИТЕРАТУРА

1. *Поспелов Д.А.* Логические методы анализа и синтеза схем / Д.А. Поспелов. – М.; Л.: Энергия, 1964. – 320 с.
2. *Карцев М.А.* Арифметика цифровых машин / М.А. Карцев. – М.: Наука, 1969. – 575 с.
3. *Папернов А.А.* Логические основы ЦВТ / А.А. Папернов. – М.: Сов. радио, 1972. – 592 с.
4. *Майоров С.А.* Принципы организации цифровых машин / С.А. Майоров, Г.И. Новиков. – Л.: Машиностроение, ЛО, 1974. – 432 с.
5. *Савельев А.Я.* Арифметические и логические основы цифровых автоматов / А.Я. Савельев. – М.: Высшая школа, 1980. – 255 с.
6. *Ситников Ю.К.* Основные узлы ЭЦВМ / Ю.К. Ситников. – Казань: Изд-во Казан. ун-та, 1981. – 52 с.
7. *Самофалов К.Г.* Цифровые электронные вычислительные машины / К.Г. Самофалов, В.И. Корнейчук, В.П. Тарасенко. – 2-е изд. Киев: Вища школа, 1983. – 455 с.
8. *Ситников Ю.К.* Введение в курс электронных цифровых вычислительных машин: учеб. пособие / Ю.К. Ситников. – Казань: Изд-во КГУ, 1985. – 76 с.
9. *Каган Б.М.* Электронные вычислительные машины и системы / Б.М. Каган. – М.: Энергоатомиздат, 1985. – 552 с.
10. Янсен Й. Курс цифровой электроники: в 4-х т. Т.1. Основы цифровой электроники на ИС / пер. с голланд. – М.: Мир, 1987. – 334 с.
11. *Пухальский Г.И.* Проектирование дискретных устройств на интегральных микросхемах: Справочник / Г.И. Пухальский, Т.Я. Новосельцева. – М.: Радио и связь, 1990. – 304 с.
12. *Ситников Ю.К.* Основы цифровой вычислительной техники / Ю.К. Ситников. – Казань: Изд-во Казан. ун-та, 1992. – 168 с.
13. *Гук М.* Аппаратные средства IBM PC. Энциклопедия / М. Гук. – СПб.: Питер, 2003. – 816 с.
14. *Таненбаум Э.* Архитектура компьютера / Э. Таненбаум. – СПб.: Питер, 2002. – 704 с.
15. *Столлицс В.* Структурная организация и архитектура компьютерных систем / пер. с англ. – М.: Вильямс, 2002. – 896 с.
16. *Букреев И.Н.* Микроэлектронные схемы цифровых устройств / И.Н. Букреев, Б.М. Мансуров, В.И. Горячев. – М.: «Сов. радио», 1975 – 368 с.
17. *Каган Б.М.* Цифровые вычислительные машины и системы: учеб. пособие для вузов / Б.М. Каган, М.М. Каневский. – М.: Энергия, 1973. — 680 с.

18. *Брей Б.* Микропроцессоры Intel: 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro Processor, Pentium II, Pentium III, Pentium 4. Архитектура, программирование и интерфейсы. Шестое издание / пер. с англ. СПб.: БХВ-Петербург, 2005 – 1328 с.
19. БСЭ, третье издание, т. 25, 1976.
20. *Точки, Рональд Дж., Уидмер Нил С.* Цифровые системы. Теория и практика, 8-е изд.: пер с англ. – М.: Вильямс, 2004. — 1024 с.
21. *Бронштейн И.Н.* Справочник по математике для инженеров и учащихся втузов: 13-е изд., исправ. / И.Н. Бронштейн, К.А. Семендяев. – М.: Наука, 1986. – 544 с.
22. *Зимин В.А.* Электронные вычислительные машины (основы теории и расчета). / В.А. Зимин. – Изд. 2-е. – М.: «Машиностроение», 1971. – 776 с.
23. *Нарышкин А.К.* Цифровые устройства и микропроцессоры: Учеб пособие для студ. высш. учеб. заведений / А.К. Нарышкин. – М.: Академия, 2006. – 320 с.
24. *Угрюмов Е.П.* Цифровая схемотехника / Е.П. Угрюмов. – СПб.: БХВ-Петербург, 2002. – 528 с.
25. *Ситников С.Ю.* Интегральные микросхемы в информационно-измерительной аппаратуре: учеб. пособие / С.Ю.Ситников, Ю.К. Ситников. – Казань: Казан. гос. энерг. ун-т, 2013. – 132 с.
26. *Лехин С.Н.* Схемотехника ЭВМ / С.Н. Лехин. – СПб.: БХВ-Петербург, 2010.— 672 с.
27. Преснухин Л.Н. Цифровые вычислительные машины: учеб. пособие для втузов / Л.Н. Преснухин, П.В. Нестеров. – М., Высшая школа, 1974. – 415 с.
28. *Угрюмов Е.П.* Цифровая схемотехника: учеб. пособие для вузов / Е.П. Угрюмов. – 3-е изд., перераб. и доп. // СПб.: БХВ-Петербург, 2010. – 816 с.:
29. *Вишневецкий А.И.* Параметроны и их применение в устройствах связи / А.И. Вишневецкий, Г.М. Немецкий. – М.: Связь, 1968. – 135 с.
30. *Крайзмер Л.П.* Устройства хранения дискретной информации / Л.П. Крайзмер. – изд. 2. – Л.: Энергия, 1969. – 312 с.

## СОДЕРЖАНИЕ

<b>ПРЕДИСЛОВИЕ</b>	3
<b>1. ВВЕДЕНИЕ В ПРЕДМЕТ</b>	4
1.1. Обработка непрерывных и дискретных сообщений	4
1.2. Понятие о количестве информации	4
1.3. Численные методы решения задач	5
1.4. Возможность автоматического выполнения расчетов	6
1.5. Алгоритм	7
1.6. Алфавит и язык	8
1.7. Уровни описания ЦВМ	9
1.8. Структурная схема цифровой вычислительной машины	11
Вопросы для самопроверки	13
<b>2. АРИФМЕТИЧЕСКИЕ ОСНОВЫ ЦВМ</b>	14
2.1. Представление информации в вычислительных машинах	14
2.2. Системы счисления	14
2.3. Перевод чисел из одной системы счисления в другую	20
2.4. Коды для представления чисел в ЭВМ	25
2.5. Выполнение операций над десятичными числами	32
2.6. Формы представления чисел в ЭВМ	36
Вопросы для самопроверки	40
<b>3. ЛОГИЧЕСКИЕ ОСНОВЫ ЦВМ</b>	42
3.1. Алгебра логики	42
3.2. Логические функции	42
3.3. Основные свойства логических функций	44
3.4. Нормальные формы логических функций	46
3.5. Полные системы логических функций	47
3.6. Минимизация логических выражений	49
Вопросы для самопроверки	52
<b>4. ЭЛЕМЕНТНАЯ БАЗА ЦВМ</b>	54
4.1. Комбинационные схемы	54
4.2. Цифровые автоматы	55
4.3. Способы физического представления информации	58
4.4. Технические аналоги функций алгебры логики	60
4.5. Передача информации между элементами цифровой вычислительной машины	63
4.6. Последовательный и параллельный способы представления и передачи информации	64
4.7. Системы логических элементов	66
4.8. Триггерные устройства ЦВМ	67
4.9. RS триггер	68
4.10. Триггер типа T	75

4.11. D триггер	76
4.12. JK триггер	78
4.13. DV триггер	80
Вопросы для самопроверки	81
<b>5. РЕГИСТРЫ</b>	83
Вопросы для самопроверки	94
<b>6. СЧЕТЧИКИ</b>	96
Вопросы для самопроверки	108
<b>7. СУММАТОРЫ</b>	110
7.1. Техническая реализация сложения двоичных кодов	110
7.2. Суммирование двоично-десятичных кодов	132
Вопросы для самопроверки	133
<b>8. ДЕШИФРАТОРЫ</b>	135
Вопросы для самопроверки	139
<b>9. АППАРАТНОЕ ПРЕОБРАЗОВАНИЕ КОДОВ ИЗ ДВОИЧНОЙ ФОРМЫ В ДЕСЯТИЧНУЮ</b>	140
Вопросы для самопроверки	147
<b>10. МУЛЬТИПЛЕКСОРЫ</b>	148
<b>11. ПРОВЕРКА КОДОВ НА ЧЕТНОСТЬ</b>	153
<b>12. ОСНОВНЫЕ УСТРОЙСТВА</b>	154
<b>ПРИЛОЖЕНИЯ</b>	155
Приложение 1. Теорема Поста – Яблонского	155
Приложение 2. Основные свойства операций в алгебре Жегалкина	155
Приложение 3. Классы логических функций	155
Приложение 4. Синтез цифровых автоматов: проектирование синхронных счетчиков	156
Приложение 5. Применение схем И-ИЛИ-НЕ для построения триггеров	162
Приложение 6. Системы счисления и экономичность оборудования	163
Приложение 7. Компоненты, не рассмотренные в основной части текста	165
<b>Параметрон</b>	166
<b>Криотрон</b>	167
Приложение 8. Некоторые термины и сокращения	169
Приложение 9. Персоналии	171
<b>ЛИТЕРАТУРА</b>	173