МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра сетей и устройств телекоммуникаций

С.М.Лапшин

ПРОЕКТИРОВАНИЕ ЦИФРОВЫХ ИЗМЕРИТЕЛЬНЫХ УСТРОЙСТВ

УЧЕБНОЕ ПОСОБИЕ
по курсовому проектированию по курсам
«Цифровые устройства средств измерений», «Устройства цифровой техники»
для студентов специальностей «Метрология и стандартизация», «Телекоммуникационные системы»

Минск 2002

1 КОМБИНАЦИОННЫЕ ЦИФРОВЫЕ УСТРОЙСТВА

Дискретное устройство в общем случае может быть представлено как m-n полюсник с n входами и m выходами (рисунок 1).

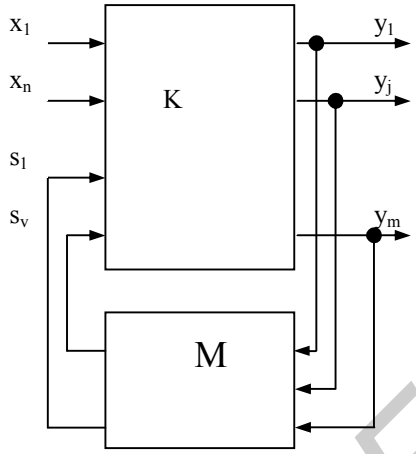


Рисунок 1

Модель дискретного устройства, отражающую только его свойства по переработке сигналов, называют дискретным автоматом. В таком автомате выделяются состояния входов $X = \{x_1,x_2\ldots..x_n\}$, выходов $Y=\{y_1,y_2\ldots\ldots y_m\}$, а также внутренние состояния элементов памяти $S=\{s_1,s_2\ldots s_v\}$, причем в дальнейшем будем считать, что как сигналы, так и элементы памяти являются двоичными.

В общем случае сигнал y_j на j-м выходе автомата, находящегося в статическом состоянии, однозначно определяется функцией выходов – зависимостью сигналов на каждом выходе от состояния входа X и внутренних состояний элементов памяти S в данный момент времени, т.е.

$$Zj(t)=f[X(t),S(t)]. \quad (1)$$

Если выходной сигнал определяется только состояниями входов, т.е. когда

$$Zj=f[(X(t)], \quad (2)$$

то такой автомат называется комбинационным (логическим).

Состояние автомата Y{X,S} – совокупность состояния входа и внутреннего состояния может быть устойчивым и неустойчивым. В устойчивом состоянии автомат будет оставаться до тех пор, пока не изменится состояние входов. При неустойчивом состоянии автомат стремится перейти в новое состояние. Дискретный автомат может переходить из одного неустойчивого состояния в другое неустойчивое, что может иметь место, например, при работе импульсного генератора.

Внутреннее состояние автомата в момент t, как и внутреннее состояние его отдельных элементов памяти, определяется функцией переходов, т.е. зависимостью внутренних состояний от состояний выходов в этот момент и внутреннего состояния автомата в предыдущий момент:

$$S(t) = \varphi[X(t),S(t-1)],$$

$$Sj(t) = \varphi j[X(t),S(t-1) \quad (3)$$

Если же элементы автомата реагируют на переход или обладают импульсным выходом, то функции перехода и выхода будут определяться соответствующими изменениями входных сигналов или внутренних состояний автомата.

Одной из наиболее общих моделей дискретного автомата является конечный автомат, задаваемый наборами входных и выходных сигналов (конечным множеством состояний входов и выходов), конечным множеством внутренних состояний, функциями выходов типа (1), (2) и переходов типа (3), а также некоторым начальным состоянием S_0.

2 МЕТОДЫ СИНТЕЗА ДИСКРЕТНЫХ АВТОМАТОВ

Методы синтеза и анализа дискретных автоматов строятся на основе алгебры логики, в которой рассматриваются свойства функций, принимающих, как и их аргументы, два значения: 1 и 0. При сопоставлении

функций алгебры логики с дискретными автоматами аргументы функции соответствуют входам, а функции выходам автомата. Так как реальные дискретные автоматы имеют конечное число входов, то мы будем рассматривать конечное число аргументов.

Функция алгебры логики f(x1,x2….xn) полностью определяется заданием ее значений на всех наборах аргументов. Поскольку число аргументов и число значений каждого аргумента конечно, конечна и область определения любой функции алгебры логики.

Функция алгебры логики может быть задана различными способами:

1 Табличный способ, когда функция задается в виде таблицы истинности (соответствия), которая содержит 2n строк по числу наборов аргументов, n столбцов по числу аргументов и один столбец значений функции. В таблице каждому набору аргументов соответствует значение функции.

2 Координатный способ, когда функция задается в виде координатной карты состояний, называемой картой Карно или диаграммой Вейча. Карта содержит 2n клеток по числу наборов значений переменных, каждая клетка определяется координатами строки и столбца, соответствующими определенному набору. Поскольку карта строится на плоскости, то все аргументы разбиваются на две группы так, что одна группа определяет координаты строки, а другая – столбца. В клетках карты проставляется значение функции на данном наборе (рисунок2).

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

Рисунок 2

3 Числовой способ, когда функция задается в виде десятичных или двоичных эквивалентов номеров тех наборов аргументов, на которых функция принимает значение 1.

4 Аналитический способ, когда функция задается в виде алгебраического выражения, получаемого путем применения алгебраических операций к переменным алгебры логики. Например, применяя операции конъюнкции, дизъюнкции и инверсии, можно задать функцию $f=x_1\&x_2\vee x_3\&x_4$.

2.1 Полностью и не полностью определенные функции

Если значение функции алгебры логики однозначно определено на всех возможных наборах значений ее аргументов, то она называется полностью определенной. Если же на некоторых наборах значение функции является безразличным и однозначно не определено, то такая функция называется н*е полностью*, или *частично определенной*. Для обозначения «безразличного» значения функции используются различные символы: при табличном и координатном способе проставляются знаки «~», «–», «∅». При числовом способе задания функции, если наборы значения аргументов, на которых функция принимает значение 1, обозначить Q, а наборы, на которых функция не определена – N, то не полностью определенную функцию можно задать: как $f=\{Q(N)\}$.

3 ЭЛЕМЕНТАРНЫЕ ФУНКЦИИ АЛГЕБРЫ ЛОГИКИ

Общее число функций алгебры логики (ФАЛ), которые можно построить для n аргументов, составляет $R=2^{2^n}$ при условии, что все значения функции полностью определены. Среди всех функций алгебры логики особое место занимают функции одной и двух переменных, называемые элементарными. Используя эти функции в качестве логических операций над переменными, можно построить различные функции алгебры логики от любого числа переменных.

Элементарные функции одной переменной. Если n=1, то число функций одной переменной равно $R=2^{2^1} = 4$. В таблице 1 приводятся таблицы истинности этих функций, названия этих функций и логических элементов.

Таблица 1. Логические функции одной переменной

x	0	1	Название	
			функции	логического элемента
f_0	0	0	Константа нуль	Генератор нуля
f_1	1	1	Константа единица	Генератор единицы
f_2	0	1	Повторение x	Повторитель
f_3	1	0	Инверсия x	Инвертор

Функции f_0 и f_1 не изменяют своих значений при изменении x и равны соответственно 0 и 1. Функция f_2 принимает те же значения, что и x, т.е. повторяет значение x. Функция f_3 принимает значения, противоположные x, т.е. осуществляет инверсию x.

3.1 Элементарные функции двух переменных

Если n=2, то число функций одной переменной равно $R=2^{2^2} =16$. В таблице 2 приведены аналитические представления и обозначения этих функций.

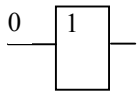
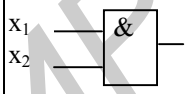
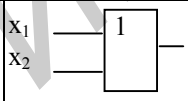
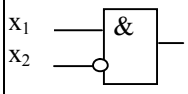
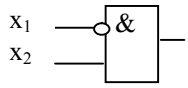
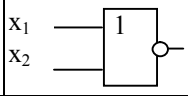
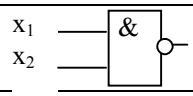
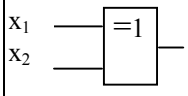
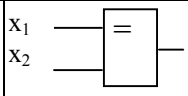
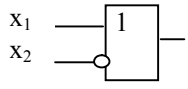
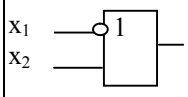
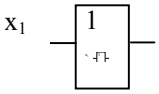
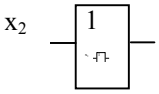
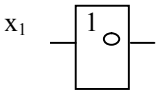
Логические функции обладают следующими свойствами: сохранение константы нуля, сохранение константы единицы, самодвойственность, линейность и монотонность.

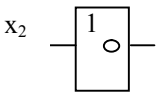
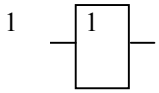
Логическая функция n переменных $f(x_1,x_2…x_n)$ называется сохраняющей константу нуля, если при нулевых значениях ее аргументов она принимает значение нуля: $f(0,0…0)=0$.

Логическая функция n переменных $f(x_1,x_2…x_n)$ называется сохраняющей константу единицы, если при единичных значениях ее аргументов она принимает значение единицы: $f(1,1…1)=1$.

Логическая функция n переменных $f^*(x_1,x_2\ldots x_n)$ называется двойственной к функции $f(x_1,x_2\ldots x_n)$, если имеет место равенство: $f^*(x_1,x_2\ldots x_n)=f(x_1,x_2\ldots x_n)$ Например, для функции дизъюнкции $f=x_1 \vee x_2$ двойственной будет функция конъюнкции $f^*=x_1 \& x_2$.

Таблица 2. Логические функции двух переменных

Функция	Аналитическое представление	Обозначение	Название элемента	Условное обозначение
Константа нуль	0	0	Генератор нуля	
Конъюнкция	$x_1 \& x_2$	$x_1 \& x_2$	Элемент «И»	
Дизъюнкция	$x_1 \vee x_2$	$x_1 \vee x_2$	Элемент “ИЛИ»	
Запрет	$x_1 \& x_2$	$x_1 \leftarrow x''_2$	Запрет	
Запрет	$X''_1 \& x_2$	$x_1 \rightarrow x_2$	Запрет	
Функция Вебба (стрелка Пирса)	$x_1 \vee x_2$	$x_1 \downarrow x_2$	Элемент ИЛИ-НЕ	
Функция Шеффера	$(x_1 \& x_2)''$	x_1 / x_2	Элемент И-НЕ	
Суммирование по модулю 2	$x_1 \& x''_2 \vee \vee x''_1 \& x_2$	$x_1 \oplus x_2$	Исключающее ИЛИ	
Эквивалентность	$x''_1 \& x''_2 \vee \vee x_1 \& x_2$	$x_1 \equiv x_2$	Эквивалентность	
Импликация	$x_1 \& x''_2$	$x_1 \rightarrow x_2$	Импликация	
Импликация	$x''_1 \& x_2$	$x_1 \leftarrow x_2$	Импликация	
Повторение x1	x_1	x_1	Повторитель	
Повторение x2	x_2	x_2	Повторитель	
Инверсия	Инверсия x_1	x''_1	Инвертор	

Инверсия	Инверсия x_2	x''_2	Инвертор	
Константа 1	1	1	Генератор единицы	

8 Логическая функция n переменных $f(x_1,x_2\ldots x_n)$ называется самодвойственной, если она совпадает с двойственной к ней функцией, т.е. имеет место равенство: $f(x_1,x_2\ldots x_n)=f^*(x_1,x_2\ldots x_n) = f''(x''_1,x''_2\ldots x''_n)$. Например, функция $f(x_1,x_2,x_3)=x_1x_2\vee x_1x_3\vee x_2x_3$ является самодвойственной, так как

$x_1x_2\vee x_1x_3\vee x_2x_3 = (x''_1x''_2\vee x''_1x''_3\vee x''_2x''_3)''$

Логическая функция n переменных $f(x_1,x_2\ldots x_n)$ называется монотонной, если для любой пары наборов значений ее аргументов $(x'_1,x'_2\ldots x'_n)$ и $(x`_1,x`_2\ldots x`_n)$, таких, что для любого $i=\{1,\ldots n\}$ $x'_i\geq x`_i$, справедливы равенства $f(x'_1,x'_2\ldots x'_n)\geq f(x`_1,x`_2\ldots x`_n)$ (для монотонно возрастающей) и $f(x'_1,x'_2\ldots x'_n)\leq \leq f(x`_1,x`_2\ldots x`_n)$ (для монотонно убывающей). Например, функция конъюнкции является монотонно возрастающей:

$$0\&1<1\&1.$$

Логическая функция n переменных $f(x_1,x_2\ldots x_n)$ называется линейной, если она представима в виде полинома: $f(x_1,x_2\ldots x_n) = c_0\oplus c_1x_1\oplus c_2x_2\oplus\ldots\oplus c_nx_n$, где каждый коэффициент c_i может принимать только два значения: 0 и 1. Например, полагая $c_0=0$, а $c_1=c_2=1$, получаем $f(x_1,x_2)=x_1\oplus x_2$, т.е. функцию «сложение по модулю 2», следовательно, эта функция является линейной.

Для функций алгебры логики справедливы следующие законы:

1) переместительный (коммутативный):

$x_1\vee x_2 = x_2\vee x_1;\ x_1x_2 = x_2x_1;$

2) сочетательный (ассоциативный):

$x_1\vee(x2\vee x3) = (x1\vee x2)\vee x3;$ $\qquad$ $x1(x2x3) = (x1x2)x3;$

3) распределительный (дистрибутивный):

$x_1\vee(x_2x_3) = (x_1\vee x_2)(x_1\vee x_3);$ $\qquad$ $x_1(x_2\vee x_3) = x_1x_2\vee x_1x_3;$

4) повторение (тавтология):

$x\vee x\vee\ldots.\vee x = x;$ $\qquad$ $xx\ldots.x = x$

5) инверсия (формулы Де Моргана):

$(x_1\vee x_2\vee\ldots.\vee x_n)'' = x''_1x''_2\ldots.x''_n;(x_1x_2\ldots x_n)'' = x''_1\vee x''_2\vee\ldots..\vee x''_n.$

3.2 Разложение функций

Первая формула разложения.

Любая ФАЛ может быть представлена в виде

$$f(x_1,x''_1,\ldots x_n,x''_n) = x_1 f(1,0,x_2,x''_2,\ldots x_n,x''_n) \vee x''_1 f(0,1,x_2,x''_2,\ldots x_n,x''_n) \quad (1)$$

Вторая формула разложения.

$$f(x_1,x''_1,\ldots x_n,x''_n) = [x_1 \vee f(1,0,x_2,x''_2,\ldots x_n,x''_n)] \& [x''_1 \vee f(0,1,x_2,x''_2,\ldots x_n,x''_n)] \quad (2)$$

На основании этих правил любой дискретный автомат может быть представлен состоящим из логических элементов конъюнкции, дизъюнкции и инверсии (декомпозиция дискретных автоматов). Пример декомпозиции дискретного автомата показан на рисунке 3.

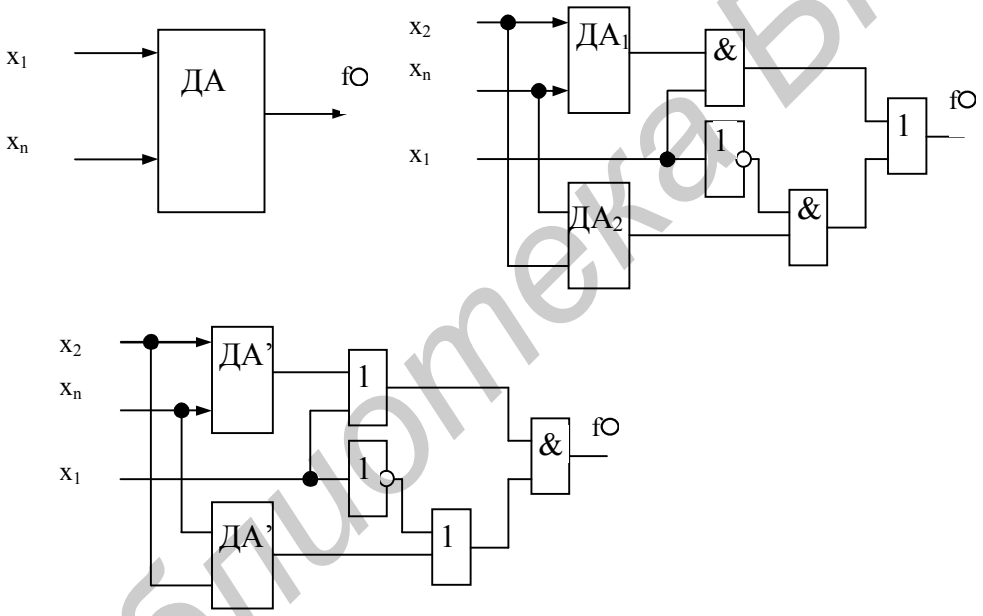


Рисунок 3

Из формул разложения вытекают следующие важные для практики формулы:

$$x_1 \vee f(x_1,x''_1,\ldots x_n,x''_n) = x_1 \vee f(0,1,\ldots x_n,x''_n), \quad (3)$$

$$x_1 \vee f(x_1,x_1,\ldots x_n,x_n) = x_1 \vee f(1,0,\ldots x_n,x_n). \quad (4)$$

Из (3),(4) можно получить следующие формулы преобразования, которые называют законами поглощения:

$x_1 \vee x_1 x_2 = x_1$; $\qquad x''_1 \vee x_1 x_2 = x''_1 \vee x_2$;

$x_1 \vee x''_1 x_2 = x_1 \vee x_2$; $x''_1 \vee x''_1 x_2 = x''_1$;

3.3 Нормальные формы функций алгебры логики

Рассмотрим ФАЛ, тождественную единице. Пусть мы имеем две переменные x1 и x2. Тогда в силу (x1∨x1=1) можно записать:

$x_1 \vee x''_1 = 1$ и $x_2 \vee x''_2 = 1$.

Кроме того, $1 \bullet 1 = 1$ и $(x_1 \vee x''_1)(x_2 \vee x''_2) = 1$, т.е. $1 = x_1 x_2 \vee x_1 x''_2 \vee x''_1 x_2 \vee x''_1 x''_2$.

10 Каждый член полученного разложения единицы – это конъюнкция, принимающая значение 1 только на одном наборе переменных; логическое произведение любой пары конъюнкций равно нулю, поскольку каждая из них отличается значением, по крайней мере, одной переменной. Полученные члены разложения называются конституентами разложения единицы или просто конституентами или минтермами.

Обратимся теперь к формуле (1) и проведем разложение функции n переменных. В итоге получим:

$$f(x_1,x_2,..x_n) = f(1,1,\ldots 1)x_1 x_2 \ldots x_n \vee f(0,1,\ldots 1)x''_1 x_2 \ldots x_n \vee \ldots \vee f(0,0\ldots 0)x''_1 x''_2 .. x''_n.$$

Каждый член разложения – это произведение значения функции при определенном наборе значений переменных и конституента разложения единицы. Таким образом, любая ФАЛ может быть представлена дизъюнкцией конституентов. Такое представление называется совершенной дизъюнктивной нормальной формой (СДНФ).

Рассмотрим ФАЛ, тождественную нулю. Пусть мы имеем две переменные x_1 и x_2, тогда в силу $(x \bullet x = 0)$ имеем $x_1 x''_1 = 0$ и $x_2 x''_2 = 0$. Так как $0 \vee 0 = 0$, то $x_1 x''_1 \vee x_2 x''_2 = 0$. Применим распределительный закон:

$0 = (x_1 \vee x_2)(x_1 \vee x''_2)(x''_1 \vee x_2)(x''_1 \vee x''_2)$.

Каждый член полученного разложения - это дизъюнкция, принимающая значение нуля только на одном наборе переменных; логическая сумма любой пары дизъюнкций равна единице. Полученные

члены разложения называются конституентами разложения нуля, антиконституентами или макстермами.

Используя (2), проведем разложение функции n переменных:

$$F(x_1,x_2\ldots.x_n)=[x_1\vee\ldots\vee x_n\vee f(0,0\ldots0)][x''1\vee\ldots x''_n\vee f(1,1\ldots1)].$$

Каждый член разложения – это логическая сумма антиконституента и значения функции при данном наборе значений переменных. Таким образом, любая ФАЛ может быть представлена конъюнкцией антиконституентов. Такое представление ФАЛ называется совершенной конъюнктивной нормальной формой СКНФ.

3.4 Минимизация функций алгебры логики

При инженерном проектировании обычно встает задача оптимизация структуры автомата, т.е. получение экономичной надежной технической реализации. Это означает, что из нескольких схем автоматов следует выбрать ту, которая содержит меньшее число элементов, а при одинаковом числе элементов – ту, которая имеет наименьшее число входов используемых элементов. Решение задачи оптимизации структуры автомата связано с проблемой минимизации ФАЛ, которую данный автомат реализует.

Минимизация – это процесс нахождения такого эквивалентного выражения функции, которое содержит минимальное число вхождений переменных. Большинство методов минимизации ориентировано на получение минимальной ДНФ, однако доказано, что что минимальное выражение функции в ДНФ будет также минимальным либо отличаться на одно вхождение переменной в классе других форм функции.

Определения:

1 Число переменных, входящих в конъюнкцию ДНФ, называется рангом конъюнкции. Ранг конъюнкции СДНФ равен числу аргументов функции.

2 Две конъюнкции называют соседними, если они различаются значением только одной переменной, например, $x_1x_2x_3$ и $x_1x''_2x_3$.

3 Логическая операция вида Ax∨Ax”=A называется операцией полного склеивания по переменной x, а вида Ax∨Ax”=Ax∨Ax”∨A – неполного склеивания по x.

4 Конъюнкция (n-1) переменной, полученная в результате склеивания соседних конституентов, называется минтермом (n-1) ранга; конъюнкция (n-j) переменных, полученная склеиванием соседних минтермов [n-(j-1)]-го ранга, называется минтермом (n-j)-го ранга.

5 Конституенты (или минтермы) функции, к которым операция склеивания неприменима, называются простыми, или первичными, импликантами.

12 6 Простой импликант называется существенным, если он образован склеиванием таких конституентов, что по крайней мере для одного из них эта операция была единственной.

7 Нормальная дизъюнктивная форма функции, составленная из простых импликантов, некоторые члены которых могут быть избыточными и удалены без нарушения эквивалентности исходной функции, называется сокращенной.

8 Нормальная дизъюнктивная форма функции называется тупиковой, если ни один ее член не может быть удален без нарушения эквивалентности ее исходной функции. К минтермам тупиковой ДНФ операция склеивания неприменима.

9 Тупиковая ДНФ называется минимальной, если она содержит наименьшее число вхождений переменных по сравнению с другими тупиковыми формами этой функции. Минимальная форма функции обязательно содержит существенные импликанты.

В настоящее время существуют два направления в решении задачи минимизации. Первое состоит в определении простых импликантов, построении из них тупиковых форм и определении путем их перебора минимальной ДНФ. Второе направление состоит в определении всех

существенных импликантов, отыскании недостающих для реализации заданной функции простых импликантов и построении тупиковой ДНФ. Каждое из направлений включает ряд методов. Рассмотрим некоторые из них.

3.4.1 Табличный метод

Метод предполагает координатное представление функции и использует свойства карты состояний, состоящие в том, что для любой клетки карты существует соседняя по строке и столбцу такая, что координаты этих клеток отличаются значением только одной переменной. Это свойство позволяет осуществлять на карте минимизацию функции, отображая процесс склеивания соседних конституентов и минтермов, соответствующих клеткам карты, путем построения различных объединений соседних клеток.

Совокупность 2^i соседних клеток карты состояний, таких, что по крайней мере одна переменная имеет для всех клеток одинаковое значение, называется подкубом. Каждый 2^i –клеточный подкуб позволяет исключить i переменных. Действительно, подкуб, состоящий из двух клеток, соседних по вертикали или горизонтали, характеризуется тем, что координаты его клеток отличаются значением только одной переменной, а остальные переменные имеют одинаковое значение. Следовательно, двухклеточный подкуб позволяет исключить одну переменную. На рисунке 4 приведены примеры двухклеточных подкубов для функции трех переменных.

Четырехклеточный подкуб содержит клетки, каждая из которых является соседней относительно двух других; координаты клеток различаются значениями только двух переменных, причем они принимают все возможные комбинации значений. На рисунке 5 приведены примеры четырехклеточных подкубов для функции четырех переменных. Восьмиклеточный подкуб содержит клетки, каждая из которых является соседней относительно трех других; координаты клеток отличаются

значениями только трех переменных, причем они принимают восемь возможных комбинаций значений. На рисунке 6 приведены примеры восьмиклеточных подкубов для функции четырех переменных.

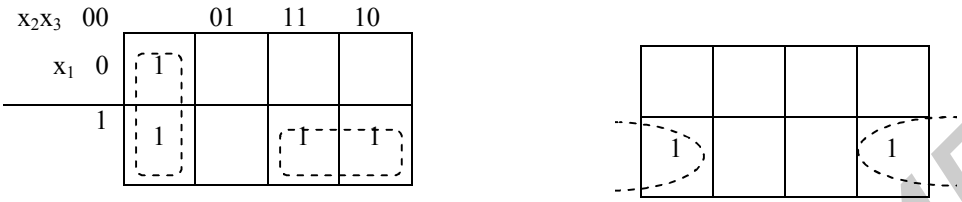


Рисунок 4

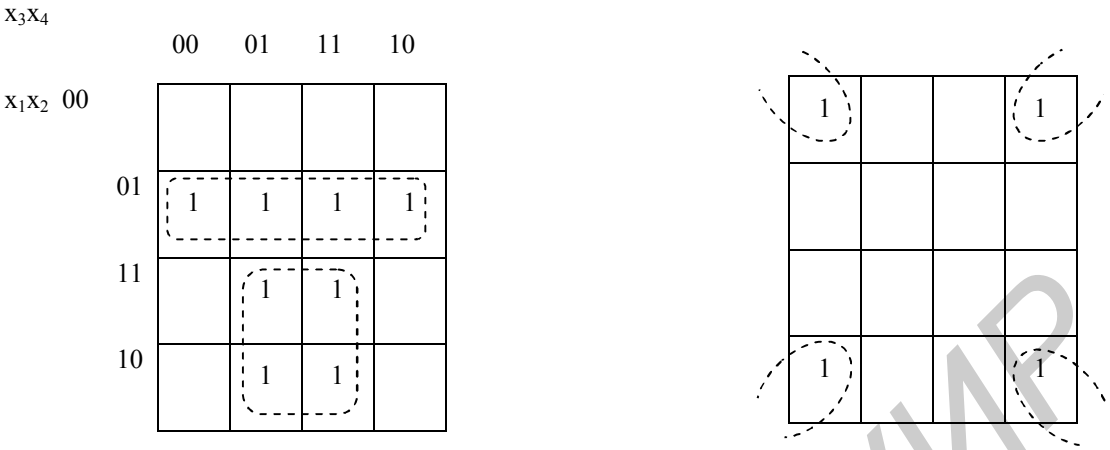


Рисунок 5

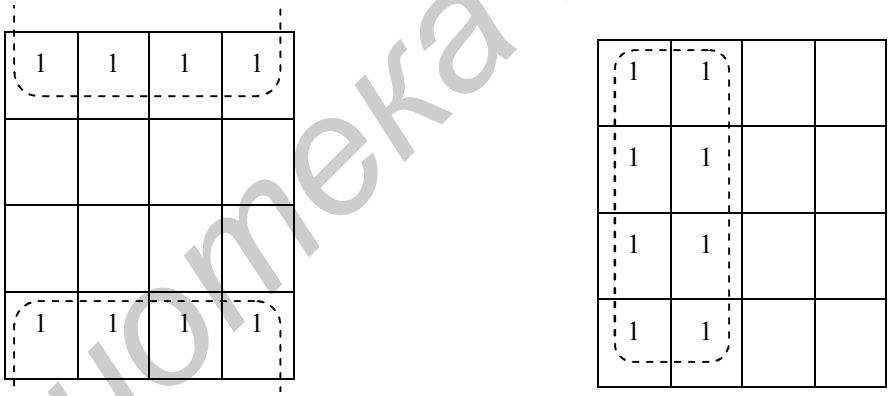


Рисунок 6

Чтобы определить вклад подкуба в функцию, необходимо определить неизменную часть координат строки и столбца объединенных клеток и взять конъюнкцию соответствующих переменных, причем если значение переменной x_i в координате равно 1, то берется переменная без инверсии, если же значение x_i в координате равно 0, то берется инверсия переменной.

При проведении минимизации можно получить различные тупиковые ДНФ, которые могут оказаться и неминимальными. Чтобы

избежать этого, нужно иметь в виду, что: 1) при построении подкубов следует учитывать прежде всего те клетки с записью 1, которые могут быть включены в один подкуб; 2) при выборе подкубов всегда следует отдавать предпочтение подкубу максимально возможного размера; 3) при построении подкубов целесообразно выбирать такие, которые охватывают возможно большее число клеток, содержащих 1. При минимизации не полностью определенных функций следует учитывать целесообразность включения в подкубы соседних клеток со знаком «~» лишь в тех случаях, когда они позволят сформировать подкуб клеток с 1 либо большего размера, либо такой подкуб, который охватит клетки с 1, ранее не включенные ни в один подкуб.

3.4.2 Алгебраический метод

Этот метод относится к первому направлению в решении задачи минимизации и основан на теореме и методе Квайна, модифицированном в 1956 г. Мак-Класки. При минимизации по данному методу последовательно проводится ряд операций:

Формирование простых импликантов. При выполнении этого этапа удобнее всего использовать задание функции числовым способом; если же она задана другим способом, то необходимо определить десятичные и двоичные эквиваленты конституентов данной функции. После этого выполняются следующие процедуры:

1) упорядочиваем двоичные номера в группы в соответствии с числом m единиц, содержащихся в их коде;

2) располагаем группы в порядке возрастания m, начиная с m=0;

3) производим попарное сравнение двоичных номеров всех членов группы с индексом m с членами только группы m+1;

4) если сравниваемые двоичные коды различаются только в одном разряде, то в следующий столбец остатков записывают двоичный код с прочерком “-“ на месте этого разряда (этот код соответствует минтерму ранга n-1);

5) все двоичные коды номеров, участвующих в операции сравнения при условии их склеивания, отмечаем знаком «√».

Пункты 3-5 повторяем для всех групп последовательно в порядке возрастания m. Все не отмеченные знаком «√» двоичные коды соответствуют

простым импликантам. После того как сформирован первый столбец остатков, к укороченным двоичным кодам этого столбца вновь применяется эта процедура, начиная с п.3, и формируется второй столбец остатков. Так повторяется до тех пор, пока среди двоичных кодов можно будет обнаружить сравнимые (т.е. такие, которые содержат прочерки в одних и тех же разрядах и различаются значением одного разряда).

Все не отмеченные двоичные коды определяют простые импликанты.

Пример. Пусть имеем функцию, заданную числовым способом:

$$f=\{2,3,4,6,10,12,14(7,8)\}x_4x_3x_2x_1.$$

В этой функции есть условные номера, поэтому при записи будем отмечать соответствующие коды знаком «~». Переходим к двоичным эквивалентам десятичных номеров конституентов, упорядочиваем их в группы по числу единиц и выполняем последовательно все этапы процедуры.

2	0010	∨	2,3	001-	∨	2,3,6,7	0-1-	A
4	0100	∨	2,6	0-1-	∨	2,6,10,14	--10	B
~8	1000	∨	2,10	-010	∨	4,6,12,14	-1-0	C
3	0011	∨	4,6	01-0	∨	8,10,12,14	1--0	D
6	0110	∨	4,12	-110	∨			
10	1010	∨	8,10	10-0	∨			
12	1100	∨	8,12	1-00	∨			
~7	0111	∨	3,7	0-11	∨			
14	1110	∨	6,7	011-	∨			
			6,14	-110	∨			
			10,14	1-10	∨			
			12,14	11-0	∨			

Промаркируем полученные простые импликанты заглавными буквами A,B,C,D.

Составление тупиковых форм из простых импликантов. Для этого используется таблица импликантов, которую называют также таблицей покрытий. Таблица содержит столько строк, сколько получено простых

импликантов, и столько столбцов, сколько обязательных конституентов содержит функция. Каждый столбец и строка отмечаются соответственно номером конституента и символом простого импликанта. Если данный импликант образован при склеивании данного конституента, то на пересечении соответствующего столбца и строки ставится метка «×». Это означает, что данный импликант покрывает данный конституент. Если какая-либо метка оказывается в столбце единственной, то она обводится кружком, что означает, что данный импликант является существенным. Совокупность существенных импликант образует ядро функции.

Задача данного этапа – выявление ядра функции и составление тупиковых форм. Поэтому после выделения существенных импликантов вычеркиваются те столбцы таблицы, которые содержат метки в строках соответствующих импликантов. Для оставшейся части таблицы записывается функция покрытий. Эта функция строится так: соединяются знаком дизъюнкции и заключаются в скобки символы простых импликантов, которые соответствуют строкам, содержащим метки в данном столбце. Это выполняется для каждого вычеркнутого столбца, затем все скобки соединяются знаком конъюнкции, поскольку рассматриваемые простые импликанты покрывают в своей совокупности оставшиеся конституенты функции. Полученная функция преобразуется путем использования закона поглощения и записывается в виде ДНФ. Каждый член ее соответствует неизбыточному набору импликантов, покрывающему оставшиеся конституенты функции. Совокупность импликантов ядра и каждого из полученных наборов соответствует одной тупиковой ДНФ функции. Если в таблице выделить ядро нельзя, то строится формула покрытий и проводятся ее преобразования.

Построим таблицу импликантов, полученных на первом этапе; она содержит четыре строки – A,D,C,D и семь столбцов – 2,3,4,6,10,12,14 (рисунок 7).

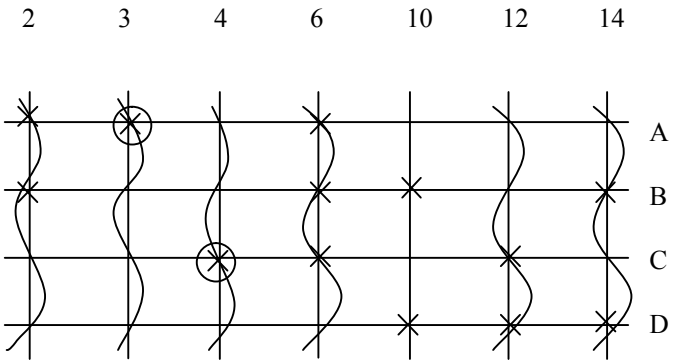


Рисунок 7

Расставим метки, выделим существенные импликанты A, C. Вычеркнем столбцы 2, 3, 4, 6, 12, 14, так как они содержат метки в строке A или C. Теперь запишем функции покрытий: это будет $\varphi = B \vee D$, так как в строках B и D содержатся метки в невычеркнутом столбце 10. Следовательно, можно построить две тупиковые ДНФ: ядро + простой импликант B, ядро + простой импликант D:

$$f_1 = x_4x_2 \vee x_3x_1 \vee x_2x_1,\ f_2 = x_4x_2 \vee x_3x_1 \vee x_4x_1.$$

Оба выражения определяют минимальную ДНФ и равноценны.

3.5 Скобочные формы ФАЛ

Рассмотренные методы минимизации позволяют получить минимальную ДНФ функции. Но полученные при этом выражения не всегда являются самыми простыми выражениями функции. Например, $f_{min} = x_1x_2 \vee x_2x_3$. Вынося за скобки x_2, получаем более простое, содержащее меньшее число операций, выражение: $f_{min} = x_2(x_1 \vee x_3)$. Такая форма представления называется скобочной. Проблема построения выражения для ФАЛ, содержащего наименьшее число операций, называется проблемой факторизации. Общего решения этой проблемы на сегодняшний день не существует, есть отдельные рекомендации и приемы по получению скобочной формы функции.

Рассмотрим следующий пример: Пусть имеем минимальное выражение ФАЛ в классе ДНФ:

$$f=x_1x_2x_5 \vee x_1x_2x_6 \vee x_1x_3x_4x_5 \vee x_2x_3x_4x_6.$$

Группируя слагаемые, содержащие общие части, и вынося за скобки общие множители, получим:

$$f = x_1x_2(x_5 \vee x_6) \vee x_3x_4(x_1x_5 \vee x_2x_6).$$

В то же время можно получить более экономичное решение, если воспользоваться законом поглощения: $x_1x_2(x_1x_5 \vee x_2x_6) = x_1x_2(x_5 \vee x_6)$. Тогда, преобразуя первое слагаемое в скобках и вынося общие части за скобки, получаем:

$$f = (x_1x_2 \vee x_2x_6)(x_1x_2 \vee x_3x_4).$$

Сравнивая исходное и конечное выражения, видим, что скобочная форма содержит почти вдвое меньше символов и включает две операции дизъюнкции и пять конъюнкции.

4 СИНТЕЗ КОМБИНАЦИОННЫХ АВТОМАТОВ

Комбинационный автомат – устройство с n входами и m выходами, у которого при идеальных (безынерционных) элементах состояние выходов в некоторый момент времени однозначно определяется состоянием входов в тот же момент. Инерционность элементов – наличие различных задержек приводит к тому, что после изменения сигналов на входах сигналы на выходе, соответствующие новому состоянию, появляются не сразу, а с некоторой задержкой, причем в переходной период возможно появление на выходе нескольких промежуточных значений сигналов. Такой процесс получил название состязаний. Рассмотрим сначала поведение комбинационных автоматов только в установившихся состояниях.

В задачу синтеза комбинационного автомата входит построение схемы структуры автомата по заданным условиям и при заданной элементной базе (например, логических элементов определенного типа), причем, как правило, ставится задача получить наиболее простую (содержащую минимальное число элементов) схему и проверить ее на состязания. Задание комбинационного автомата сводится к заданию тех функций, которые он должен реализовывать. Число функций равно числу выходов автомата.

Поэтому для задания автомата используются те же способы, что и для задания функций алгебры логики.

Рассмотрим пример синтеза комбинационного автомата, осуществляющего функцию сравнения двух двухразрядных двоичных чисел A и B, на трех выходах которого формируется уровни логической единицы при A>B, A=B и A<B соответственно. Такой автомат (цифровой компаратор – устройство сравнения) в общем виде представлен на рисунке 8.

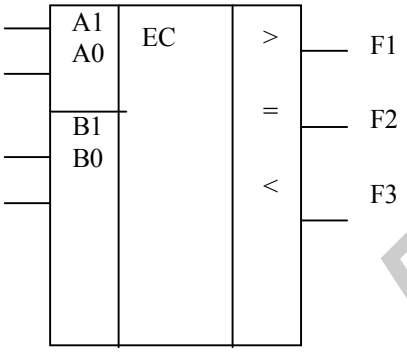


Рисунок 8

Зададим функции F1, F2 и F3, реализуемые автоматом, табличным способом:

A1	A0	B1	B0	F1	F2	F3
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0

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

На основании таблицы построим карты Карно для всех трех функций:

F1

A1A0 \ B1B0	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

F2

A1A0 \ B1B0	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

F3

A1A0 \ B1B0	00	01	11	10
00	0	1	1	1
01	0	0	1	1
11	0	0	0	0
10	0	0	1	0

F1=A1B”1∨A0B”1B”0∨A1A0B”0

F2=A”1A”0B”1B”0∨A”1A0B”1B0∨

∨A1A”0B1B”0∨A1A0B1B0=

=A1B1(A0B0∨A”0B”0)∨

∨A”1B”1(A0B0∨A”0B”0)=

=(A1≡B1)(A0≡B0)

F3=A”1B1∨A”1A”0B0∨A”0B1B0

Примером серийно выпускаемой интегральной схемы цифрового компаратора является КМОП-микросхема К564ИП2, представляющая собой компаратор для сравнения двух четырехразрядных чисел А и В (рисунок 9).

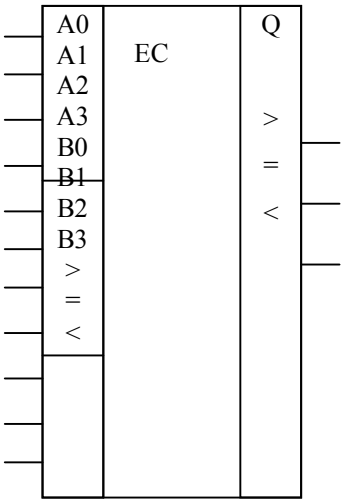


Рисунок 9

Устройство имеет три выхода A>B, A<B и A=B, на которых устанавливается высокий уровень в случае выполнения одного из условий. Входы A0..A3 и B0..B3 используются для приема сравниваемых чисел. Три входа >, < и = используются для наращивания числа разрядов сравниваемых чисел (т.е. для создания многоразрядного компаратора на нескольких интегральных схемах). Если используется только один корпус ИП2, то на вход = следует подать высокий уровень, а на входы > и < - низкие уровни. В многоразрядных компараторах входы >, < и = соединяются с сответствующими выходами схем младших разрядов.

4.1 Синтез функций алгебры логики на базе мультиплексоров

Мультиплексоры могут быть использованы при синтезе логических функций. Часто это позволяет сократить число используемых для синтеза микросхем (корпусов) по сравнению с синтезом ФАЛ на обычных логических элементах. ФАЛ n переменных может быть реализована одной микросхемой мультиплексора с n управляющими и 2n-1 информационными входами. В этом случае на информационные входы подаются константы 1 или 0 в зависимости от значения ФАЛ при соответствующем наборе переменных, подаваемых на управляющие входы. На рисунке 10 приведена схема реализации ФАЛ, заданной таблицей истинности ТИ (таблица 3).

Таблица 3

Входные переменные			Выходная функция
X_1	X_2	X_3	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

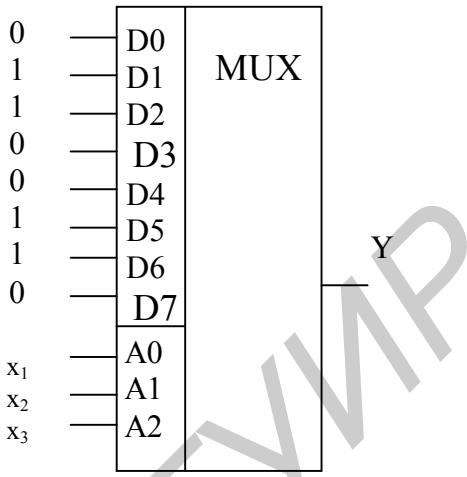


Рисунок 10

Функция трех переменных может быть реализована мультиплексором с двумя управляющими входами, если две из этих переменных подать на управляющие входы, а третью – на информационные. В этом случае заданную ФАЛ необходимо разложить по переменным, подаваемым на управляющие входы. Разложение осуществляют с использованием теоремы разложения.

Пусть, например, задана функция $f=x_1x_2 \vee x''_1x_3 \vee x_1x''_2x_3$. При разложении ее по x_1 и x_2 получим:

$f= x''_1x''_2F_0 \vee x''_1x_2F_1 \vee x_1x''_2F_2 \vee x_1x_2F_3$; $F_0=1$; $F_1=x_3$; $F_2=x_3$; $F_3=x_3$.

Схема реализации заданной функции приведена на рисунке 11. Аналогичным способом на одной микросхеме мультиплексора с n

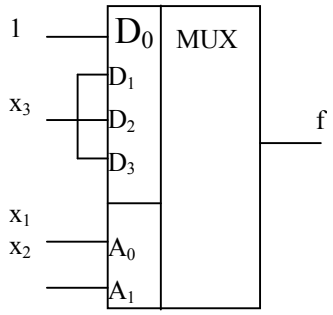


Рисунок 11

управляющими входами можно реализовать любую ФАЛ n+1 переменных.

Если число управляющих входов n_y значительно меньше числа переменных n, на информационные входы подаются функции F_i переменных $(n - n_y)$. Для реализации каждой из этих функций требуется своя схема. Функции F_i находятся в результате разложения исходной ФАЛ по n_y переменным. Следует иметь в виду, что на управляющие входы мультиплексора могут быть поданы разные группы переменных. Разлагая ФАЛ по разным группам переменных, можно получить различные по сложности схемы. Число вариантов разложения велико. В дальнейшем будем полагать, что разложение осуществляется по старшим переменным.

Приведем алгоритм реализации ФАЛ n переменных на мультиплексорах с n_y управляющими входами:

1) написание заданной ФАЛ в дизъюнктивной нормальной форме (ДНФ);

2) разложение ФАЛ по n_y переменным;

3) нахождение выражения функций F_i переменных $n\text{-}n_y$: $Fi(x_{n-ny}, x_{n-ny+1}, \ldots x_n)$. Номер функции F_i совпадает с номером набора n_y переменных, по которым раскладывается заданная ФАЛ;

4) подача на управляющие входы мультиплексора сигналов, соответствующих n_y переменным, по которым раскладывается ФАЛ;

5) подача на информационные входы мультиплексора сигналов, соответствующих значениям функций F_i.

Пример. Построить на мультиплексорах с двумя управляющими входами схему, реализующую ФАЛ:

$f=x''_1x''_2x_5x_6 \vee x''_1x_3x_5 \vee x_1x_2x''_3x''_4 \vee x_1x_2x_5x_6.$

Разложим функцию по x_1 и x_2 и получим:

$F=x''_1x''_2(x_5x_6 \vee x_3x_5) \vee x''_1x_2(x_3x_5) \vee x_1x_2(x''_3x''_4 \vee x_5x_6).$

Для реализации ФАЛ $F0= x_5x_6 \vee x_3x_5$ и $F_3 = x''_3x_4 \vee x_5x_6$ будем осуществлять разложение F_0 по x_3 и x_5, а F_3 - по x_3 и x_4:

F_0=x”$_3$x”$_5$(0)\/ x”$_3$x$_5$(x$_6$)\/ x$_3$x”$_5$(0)\/ x$_3$x$_5$(1);

F_3=x”$_3$x”$_4$(1)\/ x”$_3$x$_4$(x$_5$x$_6$)\/ x$_3$x”$_4$(x$_5$x$_6$)\/ x$_3$x$_4$ (x$_5$x$_6$).

На основе полученных выражений и алгоритма строим схему, приведенную на рисунке 12.

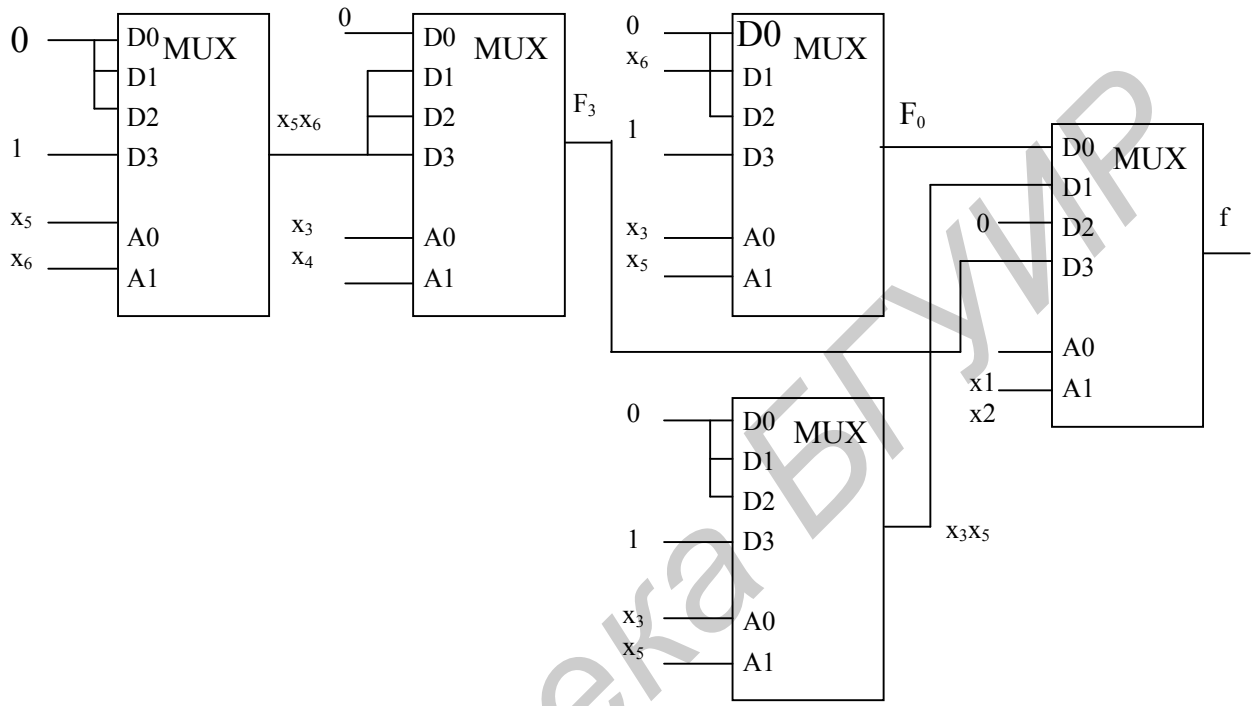


Рисунок 12

5 СОСТЯЗАНИЯ В КОМБИНАЦИОННЫХ АВТОМАТАХ

При рассмотрении работы дискретных комбинационных автоматов в статическом состоянии как постулат принималось, что a&a=0 и a\/a=1, где a и a - основное и инверсированное значение одного и того же сигнала в разных точках схемы. Но в переходные периоды, когда в схеме сигналы, идущие по разным путям, доходят до выхода не одновременно, указанные выше равенства могут нарушиться, в результате чего могут появиться кратковременные «всплески» — импульсы или паузы. Так как в общем случае эти процессы могут протекать различно в разные моменты времени, то появление указанных всплесков может носить случайный характер. Отсюда пошло название и самого процесса: *состязание цепей* или сигналов.

Аналогичное явление может быть вызвано случайным разбросом моментов изменения сигналов на нескольких входах, что получило название *состязания входов.* В общем случае могут иметь место состязания, вызываемые сочетаниями указанных причин.

Сначала рассмотрим состязания цепей, которые могут происходить в комбинационной схеме (рисунок 13,а) при изменении сигнала на одном из ее входов. Состязания определяются структурой схемы и функцией, которую она реализует. В различных структурах, реализующих одну и ту же функцию, характер состязаний может быть различным. Для возникновения состязаний необходимо (но недостаточно), чтобы между входом x, на котором происходит изменение сигнала, и выходом z было бы не менее двух путей, причем хотя бы в одном из них, но не во всех, происходило бы инвертирование сигнала.

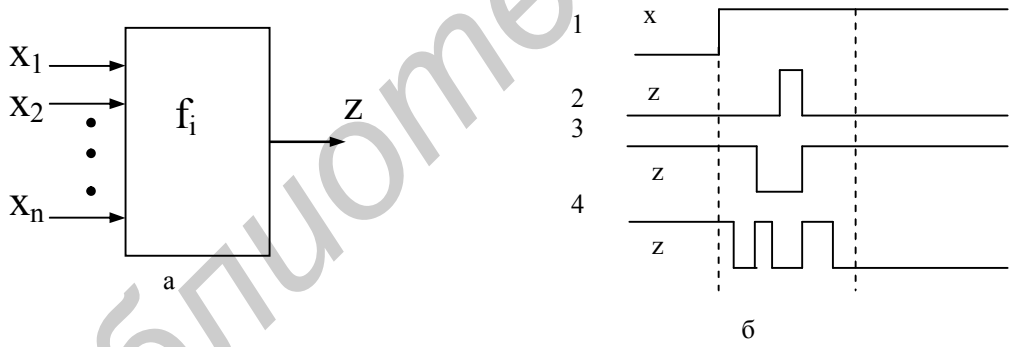


Рисунок 13

На выходе комбинационной схемы, реализующей функцию $z=f_i(x_1,x_2\ldots x_n)$, в результате состязаний после изменения значения сигнала на входе x, в переходный период возможны следующие ситуации: появление одного или нескольких единичных всплесков (эпюра 2) — импульсов при нулевом значении сигнала до и после переходного периода; появление нулевых всплесков (эпюра 3) — пауз при единичном значении сигнала до и после переходного периода; появление дребезга (эпюра 4*)* — одного или

нескольких всплесков перед окончательным изменением выходного сигнала (при $f_i(x_1,x_2\ldots x_n) = f_i(x_1,x_2\ldots x_n)$. Появление дребезга иногда называют динамическим состоянием..

Всплески могут появляться на выходе, для которого переменная x является не существенной, с чем приходится сталкиваться, например, в многовыходных схемах. Возможное число всплесков при каждом переходе зависит от числа путей между входом x и выходом z.

В одновыходной схеме с n+1 входами a, b ..., n, x, реализующей логическую функцию от n+ 1 переменной z=f(a,b,…n,x) при изменении сигнала на входе x состязания цепей могут возникнуть только при определенных сочетаниях значений сигналов на остальных входах a, b, ..., n, когда открыты пути для прохождения сигнала от входа x к выходу z.

Для j-го состояния переменных a, b, ..., n можно найти зависимость $Z(j)=\varphi_j(x_1,x_2,\ldots x_r)$, где $x_i(i=1,2,\ldots r)$ - входная переменная x, соответствующая каждому из r возможных путей в структуре между входом x и выходом z; $\varphi_j(x_1,x_2,\ldots x_r)$ - различные комбинации этих переменных, определяющих возможные варианты прохождения входного сигнала x к выходу z. Так, при двух путях x_1и x_2 в структуре возможны следующие значения:

$\varphi_j(x_1,x_2) = 0, x_i, x_ix_k, x_i\vee x_k, 1,$

а для трех путей соответственно

$\varphi_j(x_1,x_2,x_3) = 0, x_i, x_ix_k, x_i\vee x_k, x_ix_kx_j, x_i\vee x_k\vee x_j), x_i(x_k\vee x_j), x_i\vee x_kx_j, 1,$

где $i,k,j \in\{1,2,3\}$, причем x_i может означать как x, так и x.

Значения 0 и 1 показывают, что в соответствующих состояниях значение выходного сигнала не зависит от значения переменной и равно соответственно 0 или 1.

При $\varphi_j = x_i=x$ значение выхода z равно значению входа, а при

$\varphi_j = x_i = x''$ - инверсии этого значения.

Выражения вида $x_i x_k$ или $x_i \vee x_k$ при $x_i = x''_k$ указывают на наличие в схеме состязаний, т. е. на возможность появлений случайных единичных или нулевых всплесков. Выражения вида $x_i \vee x_k \vee x_j$ или $x_i \vee x_k x_j$ при $x_i = x''_k$ говорят о возможности появления дребезга, причем установившееся значение выхода z определяется значением x_i.

Рассмотрим функцию f(a,b,x), заданную картой Карно:

a \ bx	00	01	11	10
0	0	0	0	1
1	0	1	1	1

После минимизации выражение для этой функции будет: $f = ax \vee bx''$. Разлагая это выражение по всем значениям переменных a и b, при x1=x и x2=x получим значение функции, равное $x \vee x$ при a=b=1.

a	b	f	f'
0	0	0	0
0	1	x1	x1
1	0	x2	x2
1	1	x1∨x2	1

Из результатов анализа видно, что для данной функции возможно появление нулевого всплеска при a=b=1. Если дополнить выражение этой функции объединением смежных единиц, принадлежащих разным минтермам, то получим выражение $f' = ax \vee bx'' \vee ab$. Проанализировав это выражение, увидим, что данная реализация свободна от состязаний.

Выявив состояния, в которых возможны состязания, можно решить вопрос о принятии необходимых мер по их устранению. Например, если состязания появляются в неиспользуемых состояниях, то устранять их нет необходимости. Можно не бояться состязаний, если устройства, на которые

воздействует данный автомат, инерционные, т. е. не реагируют на кратковременные импульсы или паузы, которые могут возникнуть в результате состязаний.

Если необходимо устранить состязания цепей, то это может быть сделано следующим изменением структуры: а) выбором частного решения, в котором отсутствовали бы состязания, что не всегда возможно, так как это может привести к изменению реализуемой функции; б) исключением из схемы отдельных путей, вызывающих состязания, используя, если это возможно, преобразование вида x\/xa = x\/a; в) введением перекрывающих цепей, шунтирующих часть схемы, в которых возникает нулевой всплеск, или цепей, запирающих ту часть схемы, в которой возникает единичный всплеск. Последний способ может быть применен всегда, причем реализуемая функция не меняется и, если это требуется, может не изменяться основная часть схемы. Последнее бывает важно, когда данная схема является частью цепей нескольких выходов.

Рассмотрим устранение всплесков. При нулевых всплесках перекрывающая цепь Ф (рисунок 14,а) должна в переходный период создавать на выходе сигнал со значением 1 во всех состояниях, в которых возможно появление этого всплеска, и со значением 0 – во всех состояниях, в которых сигнал на выходе равен 0 при любых значениях x. Эта цепь не должна зависеть от переменной, изменяющей свое значение.

Выражение Ф может быть прибавлено не только к структурной формуле F всей системы, но и к любому выражению, входящему в F и включающему хотя бы одну переменную x, изменение которой приводит к появлению всплеска.

При единичных всплесках перекрывающая цепь G должна закрывать путь прохождения всплеска и приводить значение выходного сигнала к 0 на время переходного процесса. Иначе говоря, результирующая схема получается как последовательное соединение исследуемой F и перекрывающей G цепей (рисунок 14,б).

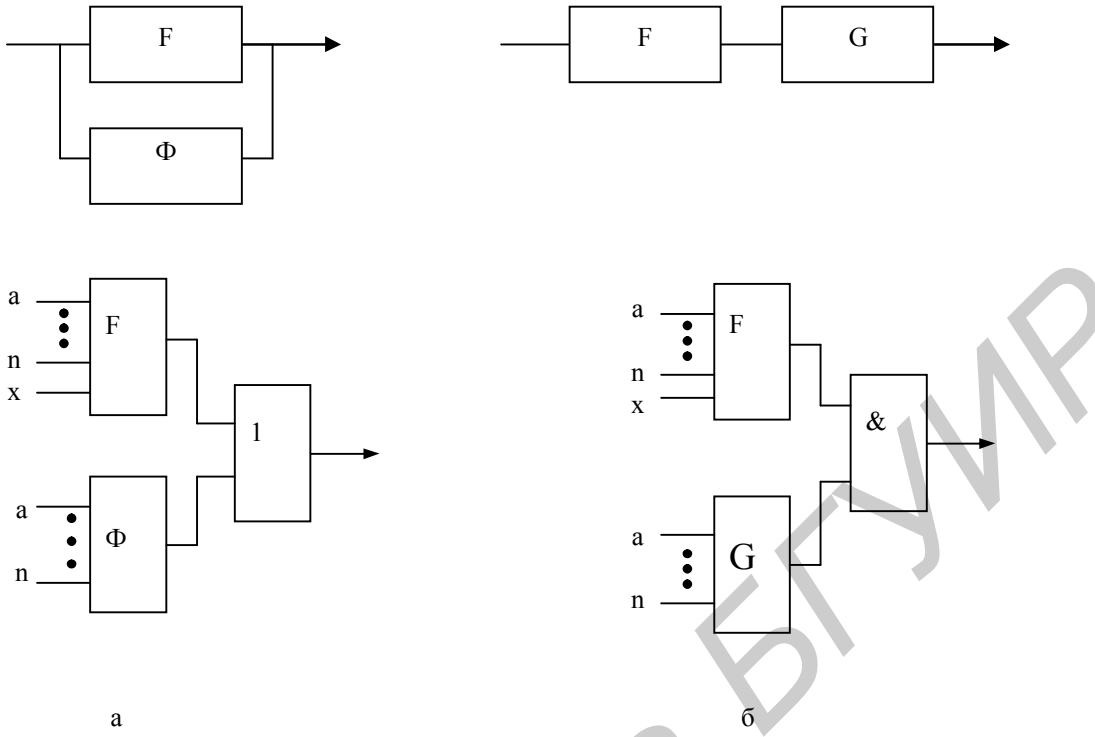


Рисунок 14

Появление дребезга, как было указано, вызывается наличием минимум трех путей, из которых в переходной период образуются цепи вида x∨xx или x(x∨x). Следовательно, в структуре можно выделить подсхемы, создающие всплески вида xx или x∨x и по отношению к ним применить указанные выше меры устранения всплесков.

Ликвидировать состязания входов подобными мерами не удается. Этого можно добиться введением на входах задержек с тем, чтобы изменение входных сигналов всегда происходило в такой последовательности, при которой всплески невозможны. Кроме этого, эффективной мерой устранения последствий переходных процессов является введение синхронизации, разрешающей изменение сигнала на выходе устройства после завершения переходных процессов.

6 ПОСЛЕДОВАТЕЛЬНОСТНЫЕ ЦИФРОВЫЕ УСТРОЙСТВА

Асинхронные последовательностные цифровые устройства (ПЦУ) состоят из комбинационной схемы (КС) и асинхронных элементов памяти в цепях обратных связей (рисунок15,а)

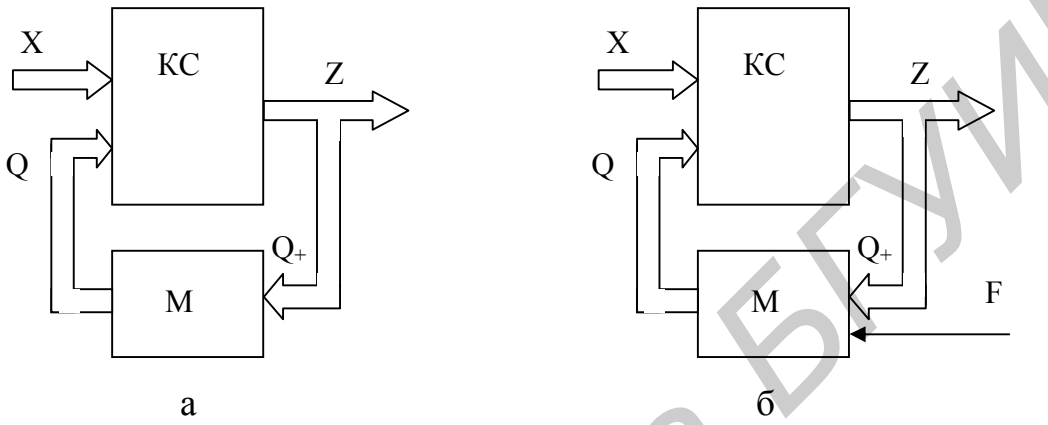


Рисунок 15

Совокупность входных сигналов устройства $X=(x_1,x_2\ldots x_n)$ называется состоянием входа автомата, совокупность выходных сигналов $Z=(z_1,z_2\ldots z_n)$– состоянием выхода, совокупность выходных сигналов элемента памяти $Q = (q_1,q_2\ldots q_n)$ – внутренним состоянием автомата. Основным назначением асинхронных элементов памяти является задержка изменений выходных сигналов КС Q_+, что обеспечивает упорядоченность воздействий на КС входных и внутренних сигналов. Одним из свойств асинхронных ПЦУ является наличие в них состязаний элементов памяти, обусловленных неидентичностью времени задержки сигналов Q_+. Кроме того, на выходах КС недопустимо появление кратковременных ложных значений сигналов Q_+, которые, поступив через ЭП на входы КС, могут вызвать ложные срабатывания автомата.

В схемах с обратными связями при выполнении определенных условий могут возникать автоколебательные процессы. При синтезе асинхронных ПЦУ необходимо соблюдать следующие условия:

1 При переходах не должны возникать автоколебательные процессы.
2 КС должна синтезироваться свободной от состязаний.
3 Состояния входа должны меняться только на соседние.
4 Значение задержки сигналов в ЭП должно быть больше максимально возможного времени протекания переходных процессов в КС.
5 Частота изменений состояний входа должна быть ограничена некоторым значением , при котором в ПЦУ еще успевают заканчиваться все переходные процессы между двумя последовательными изменениями входа.
6 Должны отсутствовать критические состязания ЭП – состязания, приводящие к неправильному функционированию автомата.

Первое и шестое условия являются необходимыми, так как невыполнение их всегда приводит к недетерминированности переходов. Остальные условия являются достаточными, так как при несоблюдении некоторых из них автомат может работать правильно.

Синхронные ПЦУ используют синхронные элементы памяти (рисунок15,б), в которых изменения информационных сигналов Q_+. воздействуют на ЭП только при наличии тактового сигнала F. Таким образом, введение тактового сигнала позволяет исключить из рассмотрения переходные процессы в ПЦУ. Тактовый сигнал выполняет функцию временного селектирования информационных сигналов Q_+, поэтому требуется, чтобы информационные сигналы были истинными только в тактовые моменты времени. Минимальное значение периода тактового сигнала должно быть не меньше максимальной продолжительности переходного процесса в ПЦУ. При соблюдении этого условия КС можно синтезировать без учета состязаний и использовать произвольное кодирование внутренних состояний автомата.

6.1 Элементы памяти

Простейшим элементом памяти является асинхронный R-S - триггер (рисунок 16).

Триггер имеет два информационных входа: R – вход установки состояния Q=0 и S – вход установки состояния Q=1, причем значения R=1 и S=1 одновременно подавать запрещено. Составив по словесному описанию таблицу истинности (рисунок 17), а затем карту Карно (рисунок 18), получим для функции переходов:

$$Q_+ = S \vee \overline{Q}R$$

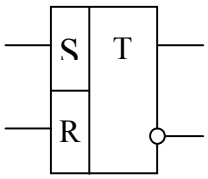


Рисунок 16

S	R	Q	Q_+
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	1	0	~
			~

Рисунок 17

SR \ Q	0	1
00	0	1
01	0	0
11	~	~
10	1	1

Рисунок 18

Синхронный S-R - триггер (рисунок 19) имеет дополнительный вход синхронизации C, высокий уровень на котором разрешает переключение триггера под воздействием информационных сигналов на входах S и R.

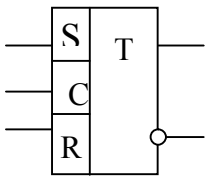


Рисунок 19

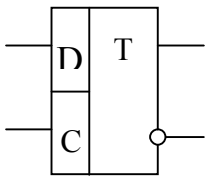
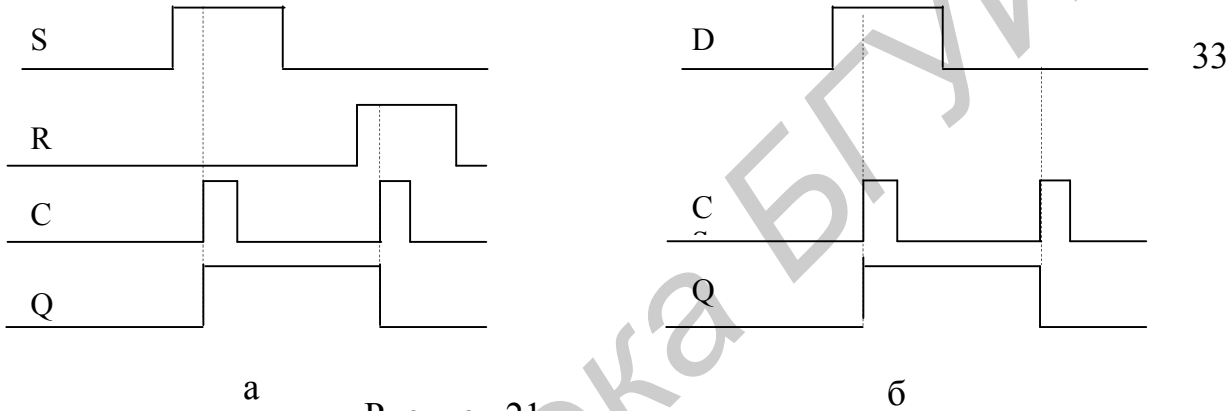


Рисунок 20

Другим типом триггера является синхронный D-триггер (рисунок 20), имеющий один информационный вход D, информация с которого при наличии высокого уровня на входе синхронизации C запоминается триггером. Временные диаграммы работы синхронного S-R- и D-триггеров представлены на рисунке 21.



а б

Рисунок 21

Триггеры, описанные выше, тактируются уровнем и называются статическими. Недостатком статических триггеров является возможность сбоя при изменении информационного сигнала во время действия тактирующего уровня. Этого недостатка лишены динамические триггеры, у которых тактирование осуществляется перепадом напряжения на тактовом входе. Примером динамического триггера является двухтактный S-R-триггер (рисунок 22). Этот триггер состоит из двух синхронных статических триггеров, первый из которых является ведущим, а второй – ведомым. В литературе такие триггеры иногда называются M-S (Master – Slave) триггерами. Работа M-S-триггера поясняется временными диаграммами рисунка 23.

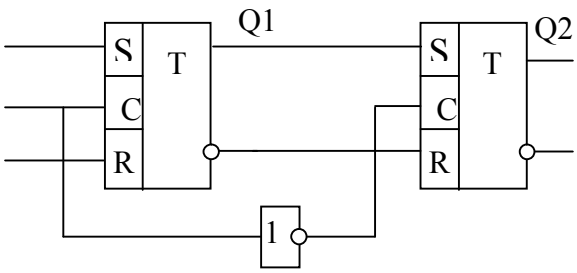


Рисунок 22

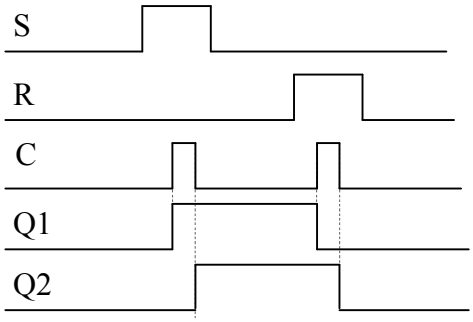


Рисунок 23

Как видно из временных диаграмм, выходной сигнал первого триггера Q1 принимает значение сигнала на информационных входах при наличии высокого уровня на входе С. Второй триггер в это время находится в режиме хранения, так как на его тактовом входе присутствует низкий уровень. Запись информации из первого триггера во второй (ведомый) происходит в момент переключения тактирующего сигнала из 1 в 0. Длительность тактового сигнала при этом не имеет значения.

Если у такого триггера соединить прямой выход со входом R, а инверсный – со входом S, получим счетный триггер, который с каждым тактом меняет свое состояние на противоположное. При введении дополнительной логики (как показано на рисунке 24) получаем универсальный J-K-триггер.

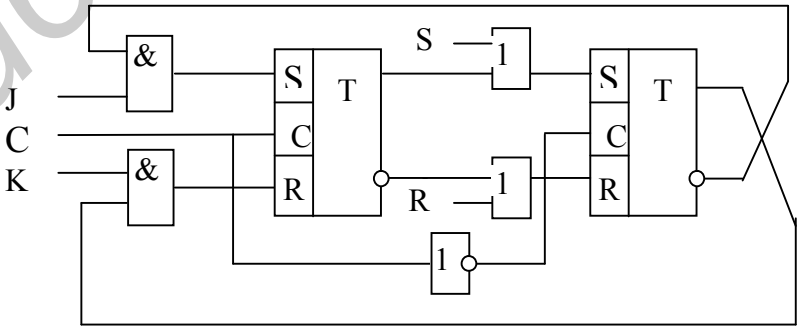


Рисунок 24

Входы J и K по своему назначению аналогичны входам S и R с той разницей, что при подаче высокого уровня на входы J и K триггер превращается в счетный. Дополнительные входы S и R являются входами асинхронной установки и сброса триггера.

6.2 Двоичные счетчики

Среди последовательностных цифровых автоматов важное место занимают счетчики. Счетчики строятся на основе синхронных динамических триггеров и делятся на синхронные и асинхронные. По способу кодирования внутренних состояний счетчики делятся на двоичные, двоично-десятичные (декадные), счетчики Джонсона, счетчики с произвольным модулем счета и др. Кроме этого, все типы счетчиков можно разделить на суммирующие (Up-counter), вычитающие (Down-counter) и реверсивные (Up-Down-counter).

6.2.1 Асинхронные счетчики.

Любой счетчик, на тактовые входы триггеров которого подается не один и тот же, а два или большее число сигналов, является асинхронным счетчиком. Обычно двоичный асинхронный счетчик состоит из счетных триггеров, соединенных последовательно таким образом, что выход предыдущего триггера соединен с тактовым входом последующего. На рисунке 25 приведены схема и условное обозначение четырехразрядного асинхронного двоичного счетчика. При поступлении счетных импульсов

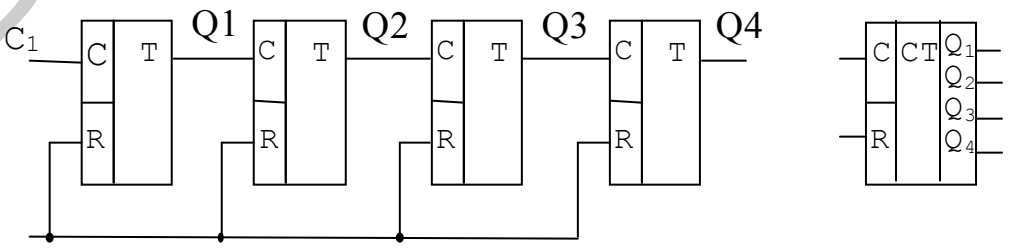


Рисунок 25

триггеры счетчика будут последовательно проходить состояния, описываемые последовательно возрастающими двоичными числами.

Состояния триггеров счетчика рисунка 25 можно описать следующими кодовыми комбинациями:

n	Q4	Q3	Q2	Q1	n	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

Временные диаграммы работы счетчика представлены на рис.26.

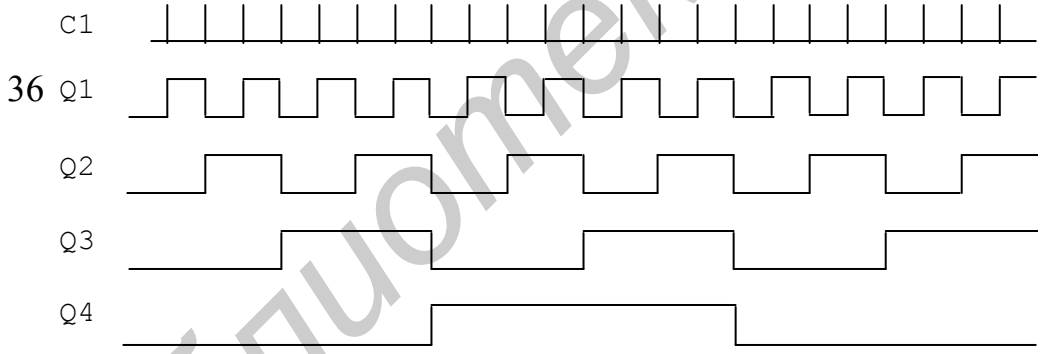


Рисунок 26

Асинхронным рассматриваемый счетчик называется потому, что в тех случаях, когда с приходом очередного счетного импульса переключаются сразу несколько триггеров, то переключаются они не одновременно, а с некоторой задержкой относительно друг друга. Если, например, все четыре триггера в счетчике рисунка 26 находятся в единице, то очередной входной импульс опрокинет первый триггер, изменение потенциала на его выходе приведет к опрокидыванию второго триггера, выходной сигнал второго триггера в свою очередь опрокинет третий, а уже после этого опрокинется

четвертый триггер. Отсюда следует: недостаток асинхронного счетчика - задержка в установлении соответствующего кода после прихода счетного импульса. Кроме того, последовательное срабатывание триггеров может привести к появлению коротких ложных импульсов на выходе дешифратора,

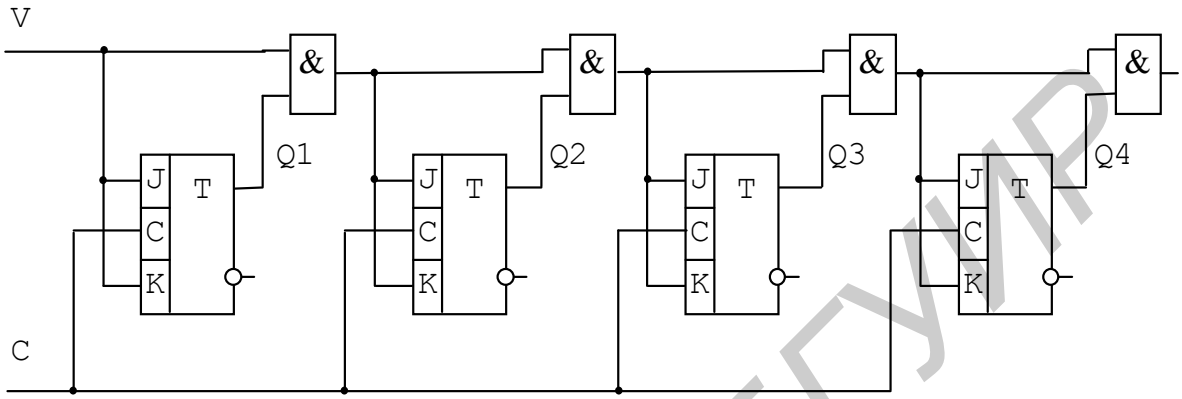


Рисунок 27.

подключенного к выходам разрядов счетчика. Действительно, описанный выше переход 1111 - 0000 осуществляется через кратковременные промежуточные состояния 1110 - 1100 - 1000, которые и будут выявлены дешифратором.

6.2.2 Синхронные двоичные счетчики.

Синхронные двоичные счетчики отличаются от асинхронных тем, что при срабатывании от входного импульса нескольких триггеров они переключаются одновременно. Это достигается благодаря тому, что тактовые импульсы подаются одновременно на входы синхронизации всех триггеров счетчика. При этом счетчик должен быть построен так, чтобы каждому импульсу соответствовали срабатывания только определенных триггеров.

Пример построения синхронного счетчика на JK-триггерах приведен на рисунке 27. В этом счетчике на входы J и K последующих триггеров подаются конъюнкции сигнала разрешения счета V и сигналов с прямых выходов предыдущих триггеров.

6.2.3 Реверсивные счетчики

Реверсивные счетчики - это счетчики, которые могут работать как в режиме суммирования, так и в режиме вычитания. Рассмотрим пример синтеза двухразрядного двоичного реверсивного счетчика, функциональное обозначение которого приведено на рисунке 28.

На вход С счетчика поступает последовательность тактовых импульсов Т, а на вход U/D - управляющее напряжение Х. Пусть при Х=0 счетчик работает на суммирование, а при Х = 1 - на вычитание. Построим счетчик на Т -триггерах. Рассмотрим временную диаграмму и таблицу истинности для Т-триггера (рисунок 28).

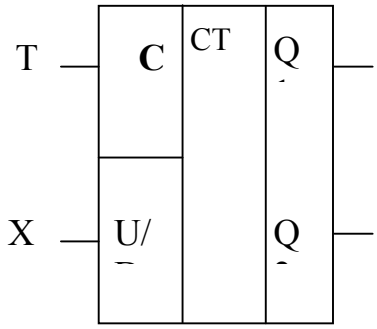
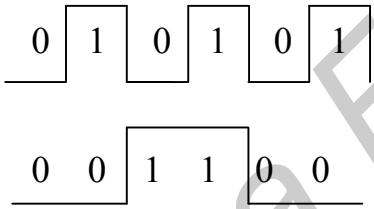


Рисунок 28



T	0	1	0	1	0	1
Q	0	0	1	1	0	0

Рисунок 29

Из рисунка 29 видно, что триггер переключается из 0 в 1 и из 1 в 0 при изменении логического уровня сигнала на тактовом входе из 1 в 0.

Составим таблицу истинности для синтезируемого счетчика .

№ такта	T	X	Q2	Q1	T2	T1
0	1	0	0	0	0	1
1	1	0	0	1	1	1
2	1	0	1	0	0	1
3	1	0	1	1	1	1
0	1	1	0	0	1	1
1	1	1	1	1	0	1
2	1	1	1	0	1	1
3	1	1	0	1	0	1

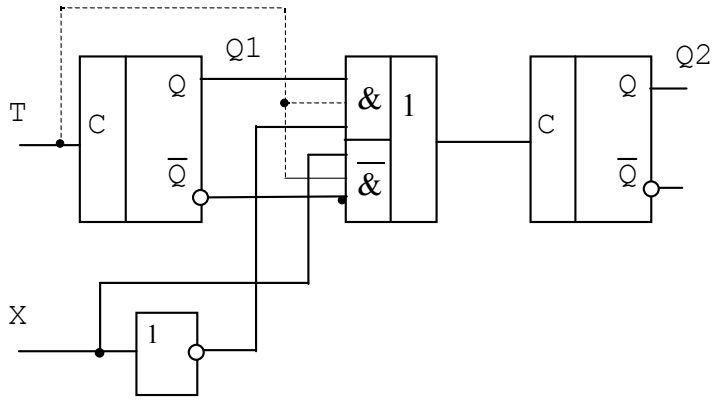


Рисунок 30

Первые четыре такта в таблице соответствуют работе счетчика на суммирование (X=0), следующие четыре такта - на вычитание (X=1). Выходы старшего и младшего разрядов счетчика Q2 и Q1 соответствуют числам в двоичной системе счисления. Используя исходные данные T, X, Q2 и Q1, определяющие алгоритм работы счетчика, и таблицу истинности для

X \ Q2Q1	00	01	11	10
0	0	1	1	0
1	1	0	0	1

выбранного типа триггера, заполняем столбцы, соответствующие сигналам на входах триггеров счетчика T1 и T2. Из таблицы видно, что тактовый сигнал на входе первого триггера счетчика повторяет входной тактовый сигнал. Для определения же тактового сигнала на входе второго триггера составим карту Карно.

Проведя минимизацию, получим выражение для тактового сигнала на входе второго триггера счетчика: T2 = Q1X”VQ”1X.

Таким образом, получим схему асинхронного реверсивного счетчика.

Из синтезированной схемы (рисунок 30) видно, что управляющий сигнал X управляет подключением входа последующего триггера к прямому или инверсному выходам предыдущего. Недостатком такого простейшего

счетчика является то, что записанная в нем информация может изменяться при изменении направления счета. Для того чтобы данный счетчик преобразовать в синхронный, достаточно добавить связи, обозначенные на рисунке 30 пунктирными линиями. При этом второй триггер будет переключаться по тактовому сигналу.

7 СЧЕТЧИКИ С ПРОИЗВОЛЬНЫМ КОЭФФИЦИЕНТОМ СЧЕТА

7.1 Счетчики на основе регистров

Счетчики на основе регистров (кольцевые счетчики) - это замкнутые в кольцо регистры сдвига, по которым под воздействием входных импульсов циркулирует одна или несколько кодовых единиц. Счетчик при этом имеет максимальный коэффициент пересчета, равный количеству входящих в него триггеров.

Вариант кольцевого счетчика на D-триггерах показан на рисунке 31. В этом счетчике в исходном состоянии все триггеры, кроме первого, находятся в нуле, а первый - в единице. Приходящий входной импульс k опрокидывает триггер, который был в единице, в состояние “нуль”. Поскольку выход Q этого триггера соединен со входом D следующего триггера, то последний при этом устанавливается в состояние “единица”.

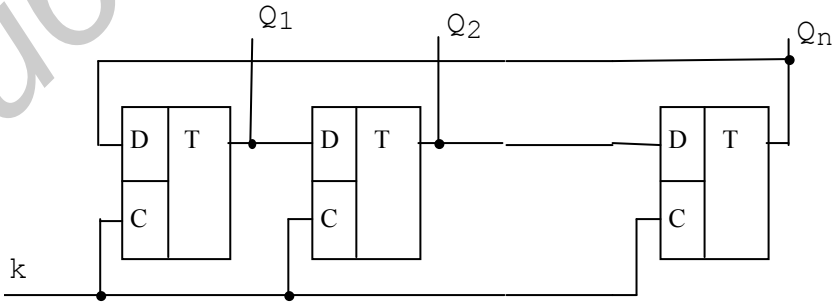


Рисунок 31

Работа простейшего кольцевого счетчика, состоящего из пяти триггеров, может быть описана следующей таблицей кодовых комбинаций (по кольцу продвигается одна единица):

k	Q1	Q2	Q3	Q4	Q5	k	Q1	Q2	Q3	Q4	Q5
0	1	0	0	0	0	3	0	0	0	1	0
1	0	1	0	0	0	4	0	0	0	0	1
2	0	0	1	0	0	5	1	0	0	0	0

Недостатком данного кольцевого счетчика является возможность сбоев, вызванных появлением лишних или исчезновением нужных кодовых единиц в кольце. Причем эти сбои, раз возникнув, могут существовать во время счета неопределенно долго, если не принять специальных мер к их устранению. В качестве такой меры может быть использовано, например, введение в счетчик логической цепи, разрешающей перепись единицы из последнего триггера в первый только при условии, что все остальные триггеры находятся в состоянии “нуль”. На рисунке 32 приведена схема подобного кольцевого счетчика, в котором устраняются сбои, проявляющиеся как в появлении лишних единиц в кольце, так и в потере единственной необходимой единицы. Здесь выходы всех триггеров соединены со входами элемента “ИЛИ-НЕ”, выход которого, в свою очередь, присоединен к управляющему входу первого триггера. Все время, пока хотя бы один триггер находится в единице, на выходе элемента “ИЛИ-НЕ” будет нулевой уровень. Когда последний триггер установится в нуль и все предыдущие также будут находиться в нуле, на выходе элемента “ИЛИ-НЕ” появится уровень логической единицы. Следующий тактовый импульс установит в единицу первый триггер.

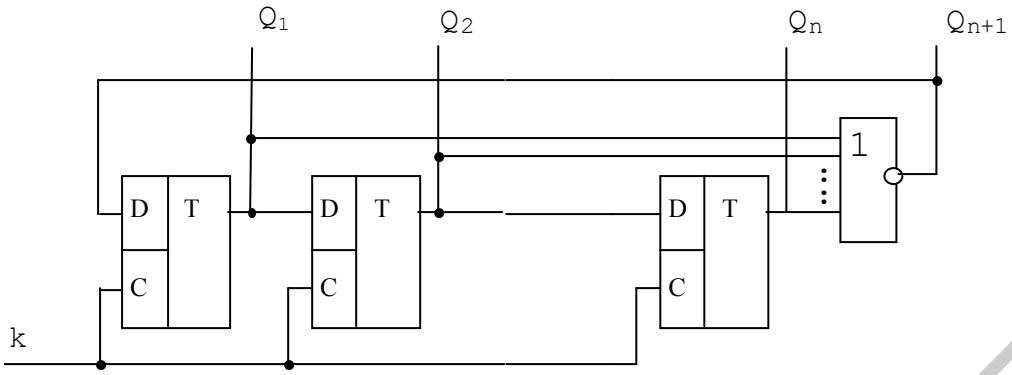


Рисунок 32

В этом счетчике при числе триггеров n получим коэффициент пересчета n+1. Следовательно, элемент “ИЛИ-НЕ” как бы заменяет (n+1)-й триггер в кольце, и с его выхода можно снимать сигнал, соответствующий выходу отсутствующего (n+1)-го триггера Q_{n+1}.

7.2 Счетчик Джонсона.

Счетчиком Джонсона называют кольцевой счетчик, который также строится на основе замкнутого регистра сдвига, но с одной перекрестной связью.

На рисунке. 33 показан пример подобного счетчика, построенного на D-триггерах. На входы всех D-триггеров, кроме первого, поданы сигналы с выходов Q предыдущего. На вход же D первого триггера подан сигнал с инверсного выхода последнего (перекрестная связь).

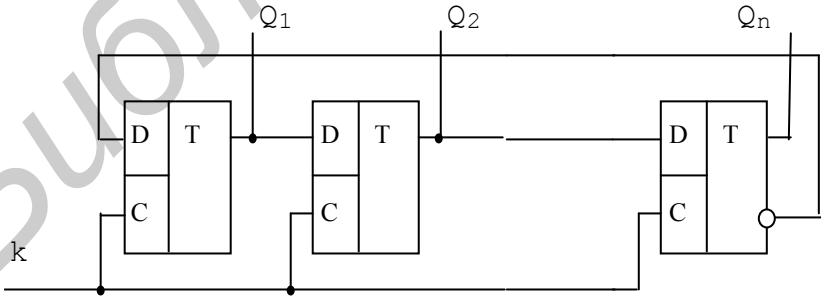


Рисунок 33

В отличие от рассмотренных выше простейших кольцевых счетчиков счетчик Джонсона имеет коэффициент пересчета, вдвое больший числа составляющих его триггеров. В частности, если счетчик составлен из пяти

триггеров, то он будет иметь десять устойчивых состояний, описываемых следующими кодовыми комбинациями:

k	Q1	Q2	Q3	Q4	Q5	k	Q1	Q2	Q3	Q4	Q5
0	0	0	0	0	0	6	0	1	1	1	1
1	1	0	0	0	0	7	0	0	1	1	1
2	1	1	0	0	0	8	0	0	0	1	1
3	1	1	1	0	0	9	0	0	0	0	1
4	1	1	1	1	0	10	0	0	0	0	0
5	1	1	1	1	1						

Как видно, при счете вначале от первого триггера до последнего распространяется “волна единиц”, а затем “волна нулей”. Код, в котором работает счетчик Джонсона, называют также кодом Либау-Крейга.

В счетчике Джонсона, как и в других кольцевых счетчиках, возможны сбои в виде лишних волн нулей или единиц. Для их устранения в десятичный счетчик может быть введена, например, логическая цепь, показанная на рисунке 34. Здесь вход D первого триггера соединен через два элемента “И-НЕ” с прямыми выходами 1-го и 5-го и с инверсным выходом 4-го триггера. Такое соединение обеспечивает переход счетчика под воздействием входных импульсов из любой запрещенной комбинации в одну из разрешенных.

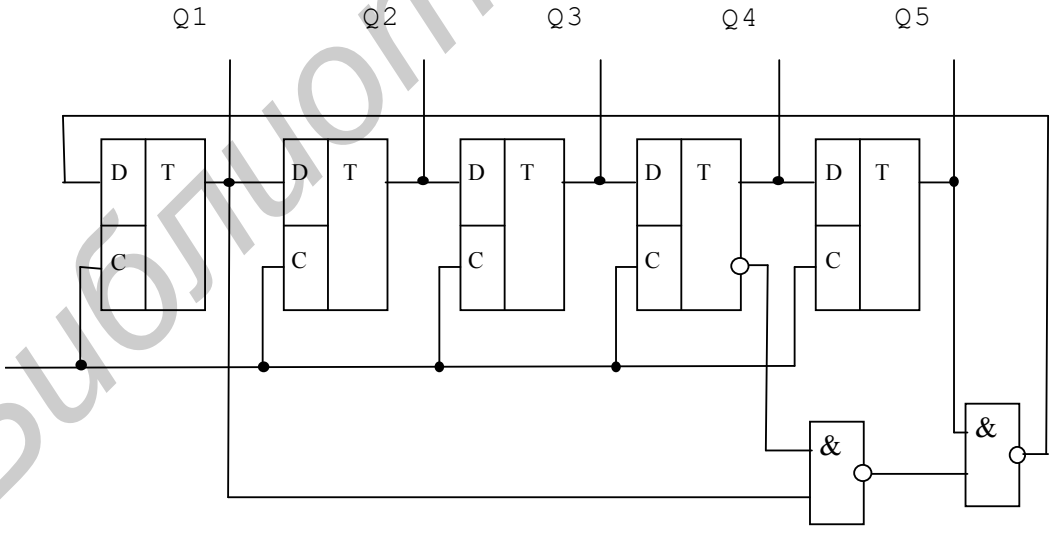


Рисунок 34

7.3 Синтез счетчиков с произвольным коэффициентом счета.

Кроме двоичных счетчиков, имеющих коэффициент счета, равный степени числа 2, в технике находят широкое применение счетчики с другими целочисленными коэффициентами счета. Такие счетчики могут быть построены путем введения соответствующих дополнительных связей в двоичные счетчики. В зависимости от типов триггеров, входящих в состав счетчика, в каждом отдельном случае приходится применять различные методы получения желаемого коэффициента пересчета.

Рассмотрим примеры синтеза счетчиков с коэффициентом пересчета, равным пяти.

Синтез синхронного счетчика на JK-триггерах.

Для построения счетчика воспользуемся таблицей переходов JK триггера.

J	K	Q	Q_{+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Каждая строка в этой таблице соответствует переключению сигнала на тактовом входе из единицы в нуль. Новое состояние триггера Q_{+1} зависит от сигналов на входах J и K в момент переключения тактового сигнала и от состояния триггера Q в этот момент. Из таблицы видно, что уровень "единица" на входе K устанавливает триггер в нуль, на входе J - в единицу, а при наличии единицы на входах J и K триггер меняет свое состояние на противоположное.

Рассмотрим таблицу состояний счетчика с коэффициентом счета, равным пяти.

№ такта	Q3	Q2	Q1	J3K3	J2K2	J1K1
1	0	0	0	0~	0~	1~
2	0	0	1	0~	1~	~1
3	0	1	0	0~	~0	1~
4	0	1	1	1~	~1	~1
5	1	0	0	~1	0~	0~
6	0	0	0	0~	0~	1~

В исходном состоянии все триггеры счетчика находятся в состоянии "нуль". Для того чтобы с приходом первого тактового импульса младший разряд счетчика Q1 переключился в единицу, необходимо, чтобы на входе J1 триггера младшего разряда присутствовал уровень единицы. Состояние входа K при этом не имеет значения, так как и при нуле и при единице на входе K произойдет переключение триггера. Во втором такте младший разряд счетчика переключится из единицы в нуль, а разряд Q2 - из нуля в единицу. Соответственно во втором такте единица должна присутствовать на входе J2 и на входе K1. Продолжая рассмотрение необходимых состояний на входах J и K, заполняем соответствующие позиции в таблице.

Закончив заполнение таблицы, проанализируем полученные результаты. Из таблицы видно, что на входах K первого и третьего триггеров не требуется уровень нуля. Отсюда можно сделать вывод о том, что на входы K этих триггеров счетчика можно подключить постоянно уровень единицы (для ТТЛ логики, например, это будет иметь место, если входы оставить свободными). Остается определить сигналы на входах J и K остальных триггеров счетчика. Воспользуемся для этого картами Карно:

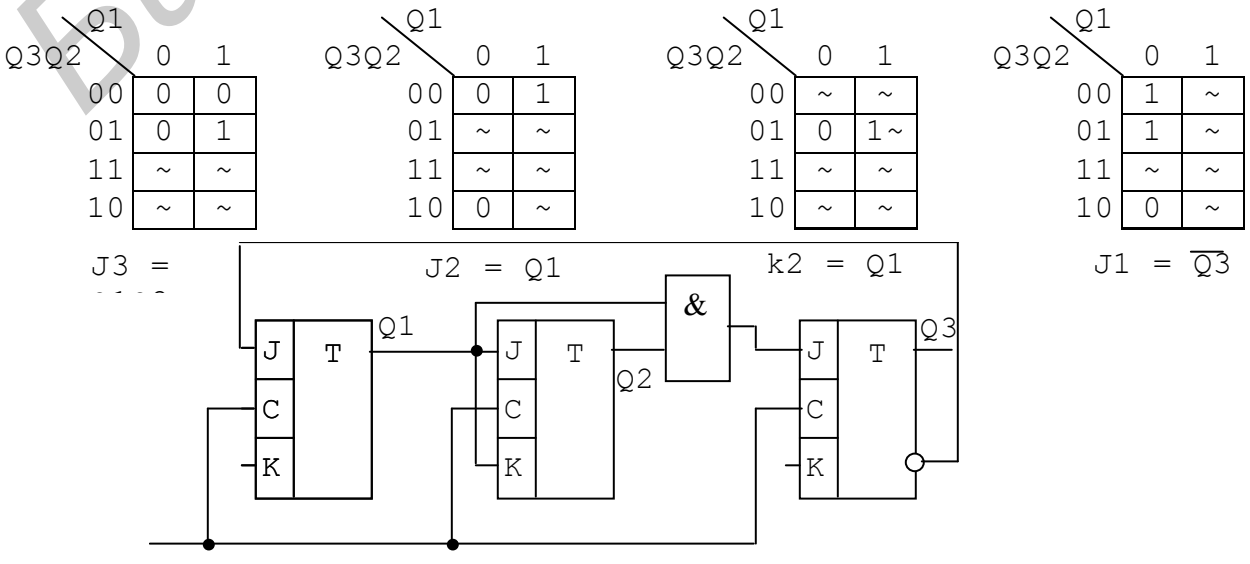


Рисунок 35

На основе полученных выражений для сигналов на J входах триггеров построим схему счетчика (рисунок 35).

При синтезе асинхронных счетчиков следует учитывать то, что в отличие от синхронного, у которого необходимо учитывать сигналы на управляющих входах всех триггеров в моменты существования соответствующих фронтов тактовых импульсов, у асинхронного имеют значения лишь сигналы на управляющих входах в моменты переключения предыдущего триггера.

Синтез асинхронного счетчика следует начинать с определения сигналов, которые должны быть поданы на тактовые входы триггеров. Поэтому вход С данного триггера должен быть подключен к тому выходу одного из предыдущих триггеров, на котором переходы из единицы в нуль, во-первых, совпадают по времени со срабатываниями данного триггера и, во-вторых, период повторения этих переходов равен минимальному времени между двумя срабатываниями данного триггера.

Проведем синтез асинхронного двоично-десятичного счетчика, работающего в нормальном двоичном коде (код 8-4-2-1), используя тактируемые фронтом D-триггеры.

Проанализируем таблицу состояний данного счетчика, приведенную ниже:

k	Q4 Q3 Q2 Q1 (8) (4) (2) (1)	D4	D2
0	0 0 0 0	~	~
1	0 0 0 1	0	1
2	0 0 1 0	~	~
3	0 0 1 1	0	0
4	0 1 0 0	~	~
5	0 1 0 1	0	1
6	0 1 1 0	~	~
7	0 1 1 1	1	0
8	1 0 0 0	~	~
9	1 0 0 1	0	0
10	0 0 0 0	~	~

Из анализа таблицы можно сделать вывод, что на тактовые входы первого (младшего) триггера следует подать входные импульсы, тактовые входы второго и четвертого следует подключить к выходу первого, а тактовый вход третьего подключить к выходу второго триггера. Из этой таблицы также следует, что первый и третий триггеры счетчика должны работать просто в счетном режиме, так что у каждого из этих триггеров следует соединить вход D с его инверсным выходом, тем самым образуя Т-триггеры.

Что касается входов D второго и четвертого триггеров, то определим сигналы, которые должны присутствовать на этих входах, анализируя каждый раз то состояние, которое триггер должен принять с приходом соответствующего перепада на его тактовый вход. Если, например, триггер должен установиться в единицу с приходом следующего тактового импульса, то во время существования кодовой комбинации, предшествующей этому импульсу, нужно обеспечить сигнал “единица” на входе D этого триггера. Если при приходе очередного импульса на вход счетчика на тактовом входе данного триггера нет тактирующего перепада, то не имеет значения, какой сигнал будет на управляющем входе этого триггера. Следуя этому правилу, заполняем графы D2 и D4.

Проведем минимизацию выражений для D4 и D2, используя карты Карно:

D4

Q4Q3 \ Q2Q1	00	01	11	10
00	~	0	0	~
01	~	0	1	~
11	~	~	~	~
10	~	0	~	~

D2

Q4Q3 \ Q2Q1	00	01	11	10
00	~	1	0	~
01	~	1	0	~
11	~	~	~	~
10	~	0	~	~

Проведя минимизацию, получим D4 = Q2Q3; D2 = Q”2Q”4.

Синтезированный таким образом двоично-десятичный счетчик представлен на рисунке 36.

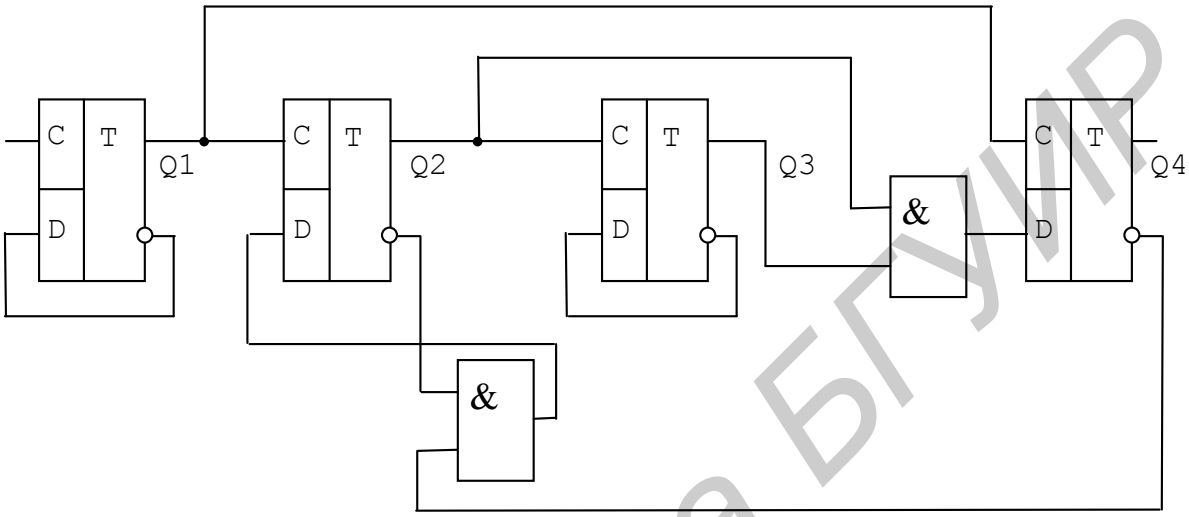


Рисунок 36

7.4 Управление коэффициентом счета.

Существуют различные способы управления коэффициентом счета. Простейший из них заключается в организации сброса счетчика в нуль при достижении им соответствующего состояния. Для этого выходы разрядов счетчика подключают ко входам дешифратора и выходным сигналом последнего осуществляют сброс счетчика. Переключая выходы дешифратора, как показано на рисунке 37, можно изменять коэффициент

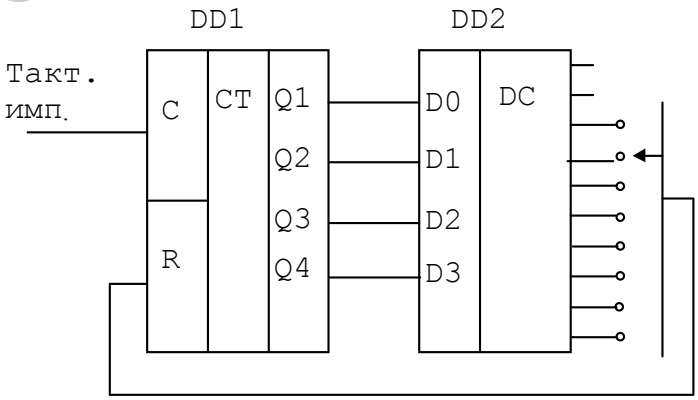


Рисунок 37

счета.

Если к выходам разрядов счетчика DD1 подключить дешифратор двоично-десятичного кода DD2, то, изменяя положение переключателя, можно изменять коэффициент счета в пределах от 2 до 8. Пусть, например, переключатель подключен к шестому выходу дешифратора, при этом при установлении на выходах разрядов счетчика кода числа 6 перепад напряжения на выходе дешифратора сбросит счетчик в нуль и начнется новый цикл счета. Следует учесть, что при дешифрации состояния "нуль" счетчик остановится, а при дешифрации состояния "единица" счетчик будет сбрасываться при поступлении каждого тактового импульса.

Счетчики находят широкое применение в качестве делителей частоты тактовых импульсов. При коэффициенте счета, не равном степени числа 2, период последовательности выходных импульсов, снимаемой с выходов последнего разряда счетчика, будет неравномерным, что эквивалентно паразитной фазовой модуляции, или будет иметь скважность, не равную двум. Для устранения этого недостатка при четном коэффициенте деления 2N можно поступать так: сначала последовательность входных импульсов подают на вход счетчика с коэффициентом счета N, а затем к его выходу подключают счетный триггер, который делит частоту еще в два раза и обеспечивает равномерность периода выходных колебаний. Для обеспечения равномерности периода при нечетном коэффициенте деления используют включение в цепь обратной связи счетчика элемент “ИСКЛЮЧАЮЩЕЕ ИЛИ”, как показано на рисунке 38.

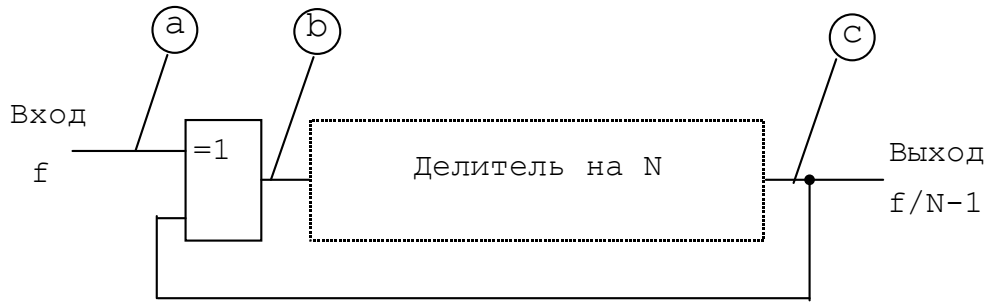


Рисунок 38

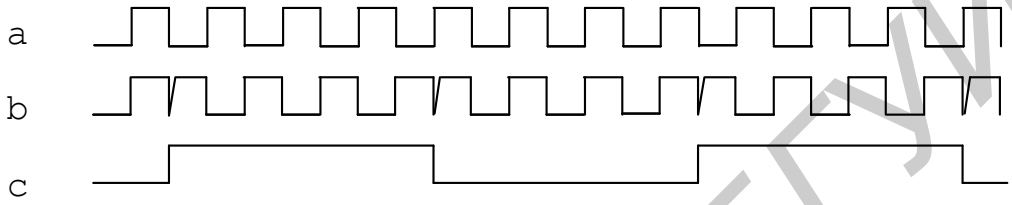


Рисунок 39

Работа счетчика поясняется временными диаграммами рисунка 39. При изменении логического уровня на выходе счетчика происходит изменение фазы на выходе элемента “ИСКЛЮЧАЮЩЕЕ ИЛИ”, при котором появляется дополнительный узкий импульс на входе делителя. Диаграммы построены для коэффициента деления счетчика N = 8. Результирующий коэффициент деления при этом равен 7, причем скважность выходных импульсов равна 2. Следует отметить, что если снимать выходной сигнал с промежуточных выходов делителя, то можно получить дробные коэффициенты деления. Так, если в рассмотренном примере выходной сигнал снять с выхода предпоследнего разряда, то коэффициент деления тактовой частоты составит (8 - 1)/2 = 3,5. Естественно, последовательность импульсов при этом будет неравномерной.

В системах связи, в измерительной технике и т.п. часто требуется изменять коэффициент деления программным способом. Для этой цели удобно использовать универсальные счетчики с предустановкой. Такие счетчики имеют дополнительные входы, по которым осуществляется запись в счетчик определенного числа - коэффициента деления (установка разрядов

счетчика в соответствующее состояние). После записи счетчик, работающий обычно на вычитание, с каждым тактом уменьшает свое состояние на единицу. После обнуления всех разрядов необходимо повторить запись. Таким образом можно изменять коэффициент деления “на лету”, т.е. в процессе счета, получая при этом любые коэффициенты деления. Выходной сигнал при этом обычно снимают с выхода переноса, где в момент перехода счетчика в нуль формируется импульс, длительностью равный тактовому.

В качестве делителей частоты с переменным коэффициентом деления используются также выпускаемые промышленностью программируемые счетчики (называемые также преобразователями “код-частота”). Упрощенная схема такого счетчика приведена на рисунке 40.

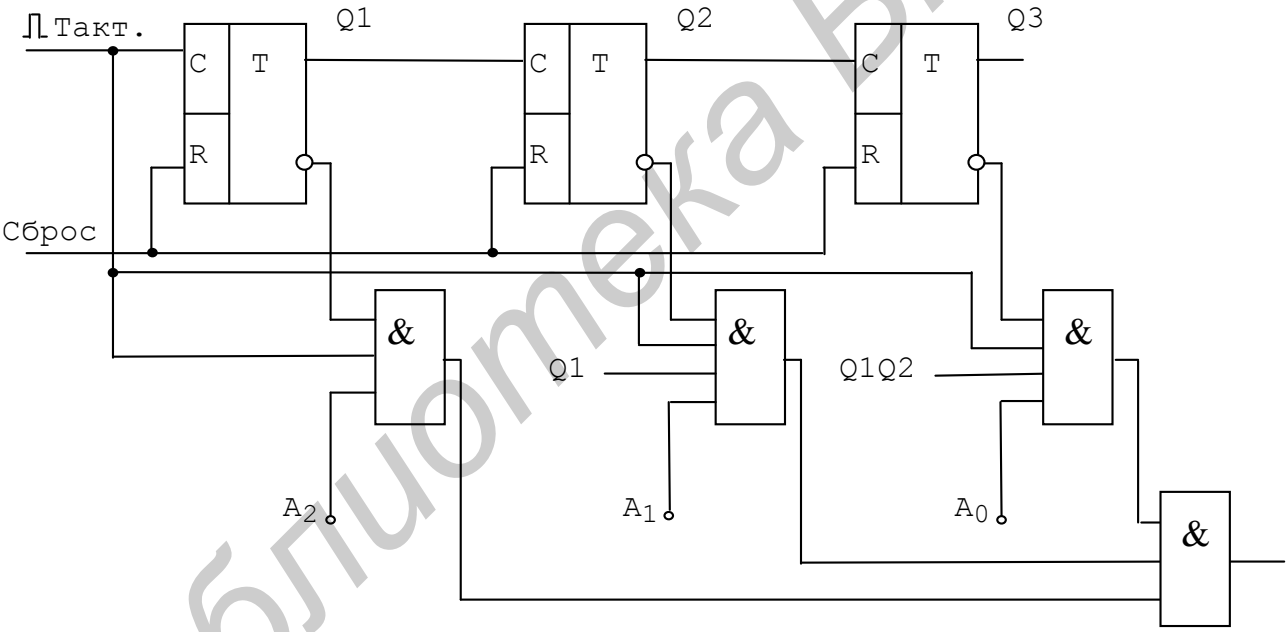


Рисунок 40

Как видно из рисунка 40, на входы элементов “И” подаются внешние управляющие сигналы A_2 - A_0. На временных диаграммах рисунка 41 показаны сигналы на выходах элементов “И” для случаев подачи единичного уровня на входы A_2, A_1, A_0. На выходах этих элементов появляются импульсы, частота следования которых соответственно равна f/2 при A2= = 1, f/4 при A1 = 1 и f/8 при A0 = 1. Эти импульсы не совпадают между собой во времени. Действительно, например, импульс на выходе третьего элемента

“И” появляется тогда, когда все предыдущие триггеры находятся в единице, а третий триггер - в нуле. А если так, то в этот момент не может появиться импульс на выходе второго элемента “И”, так как одно из условий его появления - нулевое состояние второго триггера. Импульсы с выходов элементов “И” суммируются элементом “ИЛИ”. Изменяя сигналы на входах A2 - A0, можно изменять выходную частоту преобразователя. Если значения сигналов A2 - A0 соответствуют разрядам двоичного числа A, то зависимость между частотой и числом A будет следующей:

$$f = A_2 f/2 + A_1 f/4 + A_0 f/8 = f(A_2 2^2 + A_1 2^1 + A_0 2^0)/8.$$

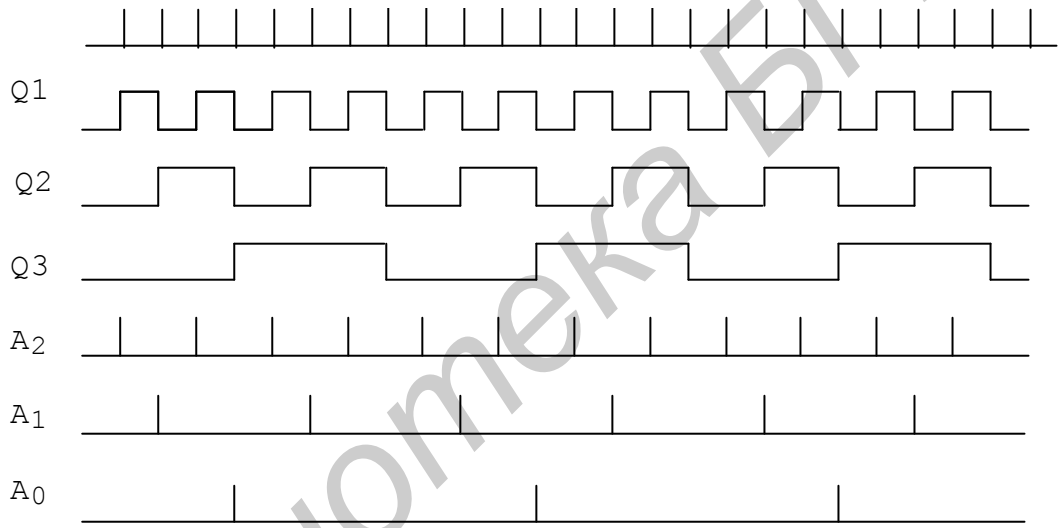


Рисунок 41

Подобный шестиразрядный делитель К155ИЕ8 позволяет получать колебания с частотами, кратными 1/64 тактовой частоты. Недостатком таких преобразователей код-частота является неравномерность периода выходного сигнала.

8 СИНТЕЗ НАДЕЖНЫХ ЦИФРОВЫХ УСТРОЙСТВ

Резервирование дискретного устройства предусматривает установку одного или нескольких идентичных экземпляров устройства. Резервироваться

может как все устройство в целом, так и отдельные его элементы или блоки. При поблочном резервировании каждый функциональный блок дискретного устройства заменяют несколькими идентичными блоками На рисунке 42 приведена структурная схема резервирования дискретного устройства в целом. Число k устанавливаемых идентичных устройств называют кратностью резервирования или избыточностью. Выходы $y_1, y_2, \ldots y_k$ отдельных дискретных устройств *(ДУ)* подаются на входы восстанавливающего органа *(ВО),* который исправляет ошибки, возникающие на выходах ДУ. Если все ДУ исправны, двоичные сигналы y_i на их выходах равны между собой. На выходе *ВО* в этом случае образуется сигнал z=y1=y2=…=yk.

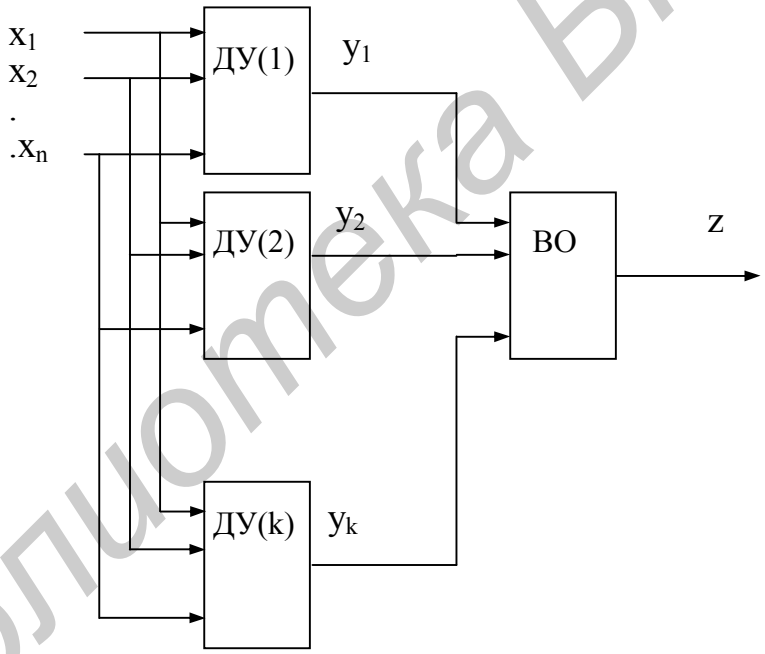


Рисунок 42

При неисправности какого-либо из *ДУ* на его выходе может появиться ложный сигнал. Ошибки могут быть двух типов: вместо правильного сигнала логического нуля появляется ложный сигнал логической единицы (ошибка типа 0→1). Принято также появление сигнала логической единицы вместо логического нуля называть ложным срабатыванием, а появление нуля

вместо единицы -ложным несрабатыванием. Правильный сигнал z на выходе избыточной структуры восстанавливается только в случае непревышения количеством ложных сигналов на выходах ДУ некоторого числа, т. е. *ВО* исправляет ограниченное число ошибочных сигналов y_i.

Восстанавливающая способность зависит от избыточности и логической функции ВО. На рисунке 43 показан пример поблочного резервирования дискретного устройства.

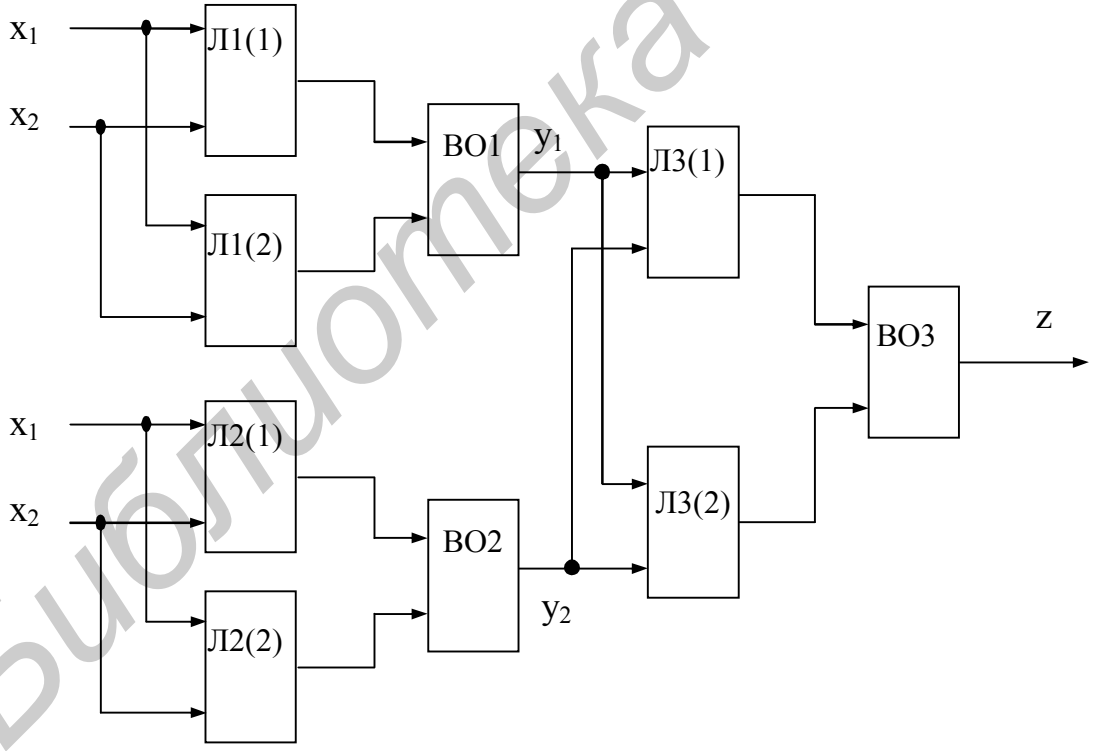


Рисунок 43

Рассматриваемое устройство состоит из трех логических

функциональных блоков Л1-Л3. При резервировании каждый блок заменяют k (в данном случае k=2) идентичными блоками с установкой отдельного ВО.

Как правило, поблочное резервирование дает большее повышение надежности по сравнению с резервированием всего устройства, но и требует большего числа восстанавливающих органов.

При решении конкретной задачи резервирования дискретного устройства следует выбрать значение избыточности k и логическую функцию ВО. Очевидно, что чем выше избыточность k, тем больше ошибок на выходах функциональных блоков будет исправляться. Значение k выбирают, исходя из конкретного заданного значения параметра надежности разрабатываемого устройства. Логическую функцию ВО выбирают в зависимости от того, какой отказ дискретного устройства является наиболее нежелательным с точки зрения воздействия на объект управления (ложное срабатывание, несрабатывание или то и другое).

Наиболее часто в качестве ВО применяют элемент, реализующий пороговую логическую функцию

$$z = M_k^r = f(y_1, y_2 \ldots y_k) = \left\{ \begin{array}{l} 0, если \sum_{i=1}^{k} y - r < 0; \\ 1, если \sum_{i=1}^{k} y_i - r \geq 0 \end{array} \right\}$$

где r – порог ВО;

k – избыточность;

y_i – логический сигнал на выходе i-го функционального блока.

Восстанавливающий орган, описываемый приведенным выражением, работает так. Если на r и более из k входов подается сигнал логической единицы, на выходе также формируется сигнал логической единицы, в противном случае – сигнал логического нуля.

Вычислим функцию $z=M_k^r$. Так. при r=1 $z = y_1 \vee y_2 \vee y_3$;

при r=2 $z=y_1y_2 \vee y_1y_3 \vee y_2y_3$; при r=3 $z=y_1y_2y_3$.

Рассмотрим более подробно случай дублирования дискретных устройств (k = 2). В этом случае могут быть использованы два типа ВО, реализующие функции вида $z=y_1 \vee y_2$ и $z=y_1y_2$.

Если в качестве ВО применен элемент ИЛИ, то в схеме исправляется одна ошибка типа $1\rightarrow0$, но не исправляются ошибки типа $0\rightarrow1$. Действительно, пусть на выходу ДУ(1) появляется ошибка типа $1\rightarrow0$, а ДУ(2) в это время исправен (на его выходе есть сигнал логической единицы). Тогда в соответствии со свойствами элемента ИЛИ на выходе восстанавливающего органа z = 1, т. е. происходит исправление указанной ошибки. С другой стороны, если на выходе одного из ДУ возникает ошибка типа $0\rightarrow1$, она в соответствии со свойствами элемента ИЛИ непосредственно передается на выход ВО вне зависимости от состояния другого ДУ. Если же в качестве ВО используется элемент И, то, наоборот, в схеме исправляется одна ошибка типа $0\rightarrow1$, но не исправляются ошибки типа $1\rightarrow0$. В связи с этим, если с точки зрения воздействия на объект управления необходимо уменьшить вероятность ложного срабатывания схемы, то в качестве ВО следует применять элемент И, если же необходимо уменьшить вероятность ложного несрабатывания, - элемент ИЛИ.

Оценим надежность дублирования схемы с элементом ИЛИ в качестве ВО. Введем обозначения: $p_0(q_0)$- вероятность отсутствия (появления) на выходе функционального блока ошибки типа $1\rightarrow0$; $p_1(q_1)$- вероятность отсутствия (появления) на выходе функционального блока ошибки типа $0\rightarrow1$. Имеют место равенства: $p_0=1-q_0$; $p_1=1-q_1$. Расчет надежности проводится с использованием таблицы повреждений, которой в рассматриваемом случае является таблица 4. В ней показаны значения выходной переменной избыточной схемы z при различных состояниях исправности всех k ее логических блоков.

Таблица 4

V_1	V_2	Z	
		При yi=0-	При y_i=1
0	0	0	1
1	0	1	1
0	1	1	1
	1	1	1

Переменные V_i ($i \in \{1,2,\ldots k\}$) используют для обозначения состояний логических блоков. Если $V_i = 0$, то соответствующий блок исправен и выдает сигнал y_i, если же $V_i = 1$, то неисправен и выдает сигнал y_i. В двух последних столбцах таблицы записывают значения выходной переменной z при заданном наборе состояний всех блоков для двух значений правильного сигнала на выходе логического блока $y_i=0$ и $y_i = 1$. Для получения значений z в логическую функцию ВО следует подставить значения величин y_i, если $V_i =0$, и y_i, если $V_i=1$.

Таблица повреждений имеет 2^k строк, соответствующих всем возможным комбинациям значений V_i. Например, таблица 4 содержит четыре строки, так как при двух блоках существуют четыре комбинации значений V_i. Первая строка соответствует случаю исправного состояния обоих блоков, вторая и третья строки - соответственно случаям неисправного состояния первого и второго блоков, четвертая строка — случаю неисправного состояния обоих блоков. Существенными повреждениями избыточной структуры называют повреждения логических блоков, вызывающие появление сигнала z = yi. В таблице 4 для случая $y_i = 0$ существенными являются неисправности, соответствующие второй - четвертой строкам таблицы, а для случая $y_i = 1$ - четвертой строке.

Пользуясь таблицей повреждений, можно определить характеристики надежности различных избыточных структур при произвольной функции ВО, считая, что последний абсолютно надежен. Повреждения различных блоков являются независимыми событиями, а появление различных состояний повреждений блоков - несовместимыми событиями, составляющими полную группу событий. Поэтому, рассматривая, например, в таблице 4 столбец z при $y_i = 1$, можно найти вероятность Q_0 появления существенных повреждений избыточной структуры, вызывающих ошибку типа $1 \rightarrow 0$, и вероятность P_0 правильного срабатывания структуры суммированием вероятностей появления тех сочетаний повреждений блоков, при которых имеет место указанная ошибка (или, если это удобнее, при

которых не имеет места указанная ошибка). Рассматривая столбец z при y_i = 1, определяем, что ошибка типа $1\rightarrow0$ имеет место только в одном случае, когда $y_1=V_1=1$, т. е. при неисправности обоих блоков дублированной структуры. Поэтому

$$Q_0=q_0^2=(1-p_0)^2;\ P_0=1-Q_0=1-(1-p_0)^2=2p_0-p_0^2.$$

Рассматривая столбец z при y_i = 0, можно найти вероятность P_i отсутствия существенных повреждений избыточной структуры, вызывающих на выходе ошибку типа $0\rightarrow1$. Из таблицы 4 видно, что ошибка типа $0\rightarrow1$ возможна при повреждении одного или двух логических блоков и невозможна только в случае исправности обоих блоков. Поэтому $P_1=p_1^2$.

Если в схеме дублирования ВО реализует функцию И, надежность избыточной структуры: $P_0=p_0^2$; $P_1=1-(1-p_1)^2=2p1-p_1^2$. Таким образом, при избыточности k=2 использование ВО, реализующего логическую функцию ИЛИ, увеличивает P_0 и уменьшает P_1, а использование ВО, реализующего функцию И, наоборот, увеличивает P_1, и уменьшает P_0 по сравнению с неизбыточной структурой.

Часто резервирование дискретного устройства с использованием логического элемента ИЛИ называют параллельным, а элемента И - последовательным резервированием.

Если необходимо исправлять более одной ошибки одного типа или одновременно ошибки обоих типов, приходится выбирать избыточность k> 2. При k= 3 возможны три вида ВО, реализующие функции M_3^1,M_3^2,M_3^3. Использование функций M_3^1 (ИЛИ) и M_3^2 (И) позволяет исправлять по две ошибки соответственно типов $1\rightarrow0$ и $0\rightarrow1$, а использование функции $M_3^2=y_1y_2\vee y_1y_3\vee y_2y_3$ - одиночные ошибки обоих типов.

Рассмотрим работу схемы резервирования с использованием ВО, реализующего функцию M_3^2 (рисунок 45) при одиночных неисправностях.

Пусть на выходе ДУ (1) имеется ошибка типа $1\rightarrow0$ (y_1, = 0).

Остальные два блока при этом исправны ($y_2=y_3=1$). Тогда на выходах элементов И1 и И3 также возникает ошибка типа $1\rightarrow 0$, но на выходе элемента И2 сигнал логической единицы сохраняется, так как этот элемент не связан с выходом неисправного блока. Данный сигнал через элемент ИЛИ передается на выход устройства, и, следовательно, ошибка исправляется. Если на выходе есть ошибка типа $0\rightarrow 1$ ($y_1 = 1$), а на выходах двух остальных исправных блоков - сигнал логического нуля, на выходах всех элементов И будет присутствовать сигнал логического нуля, так как каждый из них связан с выходом хотя бы одного исправного блока. В этом случае на выходе устройства также будет сигнал логического нуля, и, следовательно, указанная ошибка исправляется. В то же время две ошибки в схеме не исправляются. Если, например, на выходах двух блоков возникает ошибка типа $1\rightarrow 0$, на выходах всех элементов И устанавливается сигнал логического нуля, и ошибка передается на выход устройства.

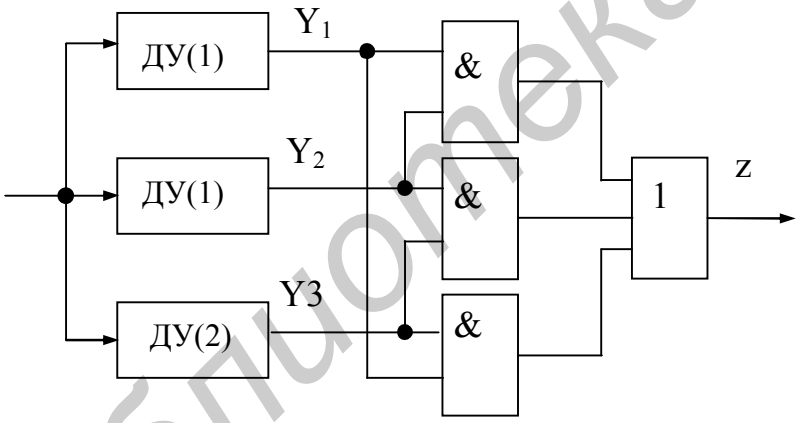


Рисунок 44

Функцию M_3^2 назвали мажоритарной, а реализующий ее элемент - мажоритарным. Работает мажоритарный элемент в соответствии со следующим правилом: на выходе этого элемента присутствует сигнал логической единицы в том и только в том случае, если на большинство его входов поданы сигналы логической 1. Число входов мажоритарного элемента всегда нечетное. Например, на выходе элемента, реализующего

функцию M_3^2, сигнал логической единицы появляется при подаче такого же сигнала на любые из двух входов элемента. Такой элемент получил наиболее широкое распространение на практике в качестве ВО, так как он при сравнительно небольшой избыточности (в 3 раза) обеспечивает повышение надежности относительно ошибок обоих типов. Характеристики надежности в этом случае следующие (с условием абсолютной надежности ВО):

$$P_0=3p_0^2-2p_0; \qquad P1=3p_1^2-2p_1.$$

В общем случае при построении резервированного устройства с избыточностью k и использовании ВО, реализующего функцию M_k^r , в нем исправляются k-r ошибок типа $1\rightarrow0$ и r- 1 ошибок типа $0\rightarrow1$.

При достаточно высокой надежности логических блоков p_0 и p_1 выбор большого k позволяет получить высокую надежность логической части избыточной структуры. В этом случае надежность структуры в целом будет определяться и ограничиваться надежностью восстанавливающего органа. Если дискретное устройство резервируется целиком, структура содержит только один ВО. Поэтому следует обратить особое внимание на повышение надежности ВО (например, резервированием его внутренних элементов). Очевидно, что надежность ВО должна быть, по крайней мере, не ниже надежности остальной части избыточной структуры.

При поблочном резервировании избыточная структура содержит большое число ВО. Поэтому влияние ВО на общую надежность устройства увеличивается. Для повышения надежности возможна установка вместо одного k одинаковых восстанавливающих органов ВО (рисунок 45).

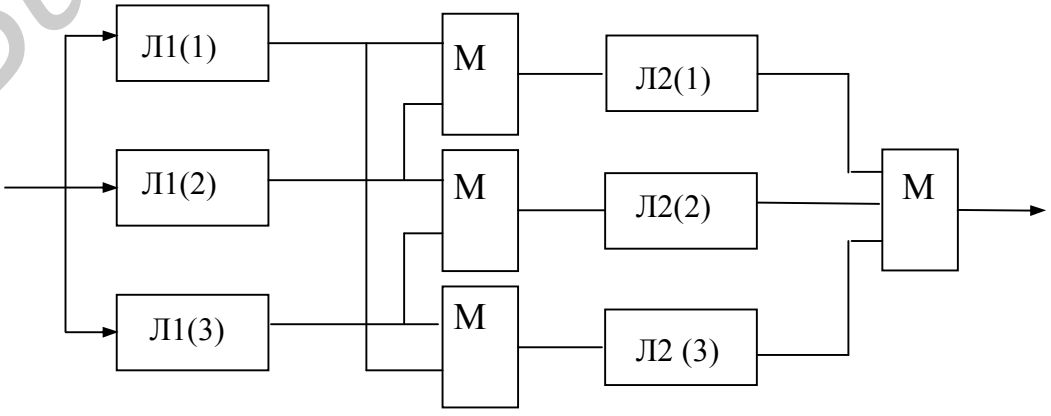


Рисунок 45

На этом рисунке приведен пример схемы, в которой вместо одного установлены три мажоритарных элемента. Такая структура получила название троированной. Она находит широкое применение при построении надежных вычислительных и логических устройств. Структура не отказывает при любых повреждениях одного из k=3 одинаковых логических блоков и одного из k=3 восстанавливающих органов.

9 ЭТАПЫ Проектирования цифровых устройств

Рассмотренные выше вопросы и методы синтеза служат основой для проектирования цифровых устройств. Процесс проектирования обычно разделяют на четыре этапа: системное или архитектурное, логическое, техническое и технологическое.

На этапе системного проектирования, которое относят к предпроектной стадии разработки, решается задача выбора архитектуры проектируемого устройства, т.е. состава основных блоков и структуры их взаимодействия, составления общего алгоритмического описания и формулировки условий работы отдельных блоков. Особое значение этап системного проектирования имеет в тех случаях, когда проектируемое устройство достаточно сложно и составление его общего алгоритмического описания возможно лишь в самых общих формулировках. На этапе системного проектирования такая система разделяется на ряд подсистем, например, по функциональному признаку. Некоторые из них, в свою очередь, оказываются структурно-сложными и также разделяются на ряд функциональных блоков и т.д. При каждом разделении на блоки определяются структура их взаимосвязей и характер взаимодействия, а также дается общее алгоритмическое описание работы каждого блока. Поскольку научно обоснованных методов системного проектирования в настоящее время не существует, в большинстве случаев архитектуру устройства выбирают исходя из эвристических соображений, основываясь на имеющихся знаниях и опыте.

На этапе логического проектирования решается задача синтеза структуры устройства в целом и отдельных его блоков. Под структурой устройства понимается совокупность физических элементов и соединений между ними, составляющих данное устройство. При синтезе чаще всего оперируют функциональной схемой устройства, т.е. графическим изображением структуры устройства в виде условных блоков и элементов, а также путей передачи сигналов между ними и сигналов со входов на выходы

устройства. Поэтому конечным результатом логического проектирования является функциональная схема устройства. Логическое проектирование ведется с учетом различных ограничений: обеспечения требуемого быстродействия, надежности и устойчивости его работы, простоты и однородности его структуры и т.д.; основным критерием при выборе окончательного решения чаще всего является экономичность схемы.

На этапе технического проектирования решается задача отображения функциональной схемы устройства в реальных элементах с учетом их конструктивного оформления. При этом производится разбиение элементов схемы по модулям (или корпусам), затем размещение модулей по конструктивным элементам, построение монтажных схем и т.д. Конечным результатом этапа технического проектирования является комплект технической документации для производства печатных плат и изготовления устройства.

Технологическое проектирование решает задачи разработки технологических процессов изготовления устройства и соответствующей технологической документации.

Завершаются все этапы проектирования изготовлением образца по разработанной технической документации, его наладка и проведение опытных испытаний.

Все этапы проектирования тесно связаны между собой и взаимозависимы (рисунок 46).

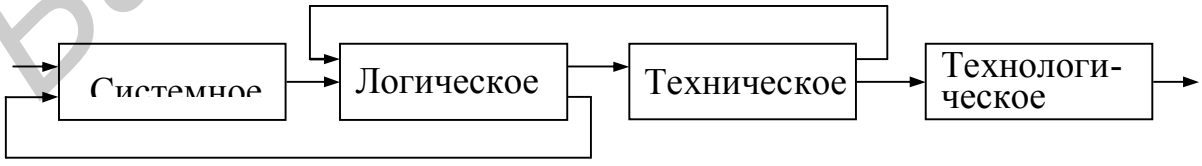


Рисунок 46

ЛИТЕРАТУРА

1. Ершова Э.Б., Рогинский В.Н., Маркин Н.П. Основы дискретной автоматики в электросвязи. М.: Связь, 1979.
2. Гутников В.С. Интегральная электроника в измерительных устройствах. Л.: Энергоатомиздат, Ленинград. отд-ние, 1988.
3. Пухальский Г.И., Новосельцева Т.Я. Проектирование дискретных устройчтв на интегральных микросхемах. М.: Радио и связь, 1990.
4. Соловьев В.В. Проектирование функциональных узлов цифровых систем на программируемых логических устройствах. Мн.: Бестпринт, 1996.