В. Ф. Романов

Лекции по теории автоматов

Часть 2

Логические основы цифровых автоматов

Владимир 2009

Учебное пособие для студентов очной и заочной форм обучения специальностям в области вычислительной техники, информатики и управления.

ОГЛАВЛЕНИЕ

ЧАСТЬ 2. ЛОГИЧЕСКИЕ ОСНОВЫ ЦИФРОВЫХ АВТОМАТОВ

2.1. Способы задания логических функций

Логические функции (ЛФ) задаются на множестве аргументов, каждый из которых определяется на множестве {0, 1}({*ложь, истина*}), при этом сами функции принимают значения из этого же множества. Общая форма записи функции от *n* переменных: $f(x_1, x_2, \ldots, x_n)$. Ниже перечислены основные способы задания ЛФ.

1. Таблица истинности (ТИ); пример приведен ниже в п. 4.

2. Аналитическая форма (формула, содержащая обозначения логических переменных, логических операций, круглые скобки). Примеры: *дизъюнктивная нормальная форма* (ДНФ) или *конъюнктивная нормальная форма* (КНФ) логической функции.

3. Карты Карно – используются как способ задания логических функций и как средство их минимизации (см. раздел 2.2).

4. Числовой способ; для пояснения используем таблицу истинности:

$f(x_1, x_2, x_3) = \vee (0, 2, 3)$ – в скобках указаны номера наборов, на которых функция равна единице;

$f(x_1, x_2, x_3) = \wedge (1, 4, 5, 6, 7)$ – в скобках указаны номера наборов, на которых функция равна нулю.

Таблица истинности ЛФ

x_1	x_2	x_3	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

5. *Кубическая* форма. Функция определяется списком (комплексом) 0-кубов (*нуль-кубов*) – наборов, на которых ее значение равно "1". Для функции, заданной таблицей 1, записывается следующий комплекс 0-кубов:

$$f = \begin{cases} 000 \\ 010 \\ 011 \end{cases}$$

Введем ряд важных определений.

Импликанта логической функции f – некоторая логическая функция φ, определяемая импликацией $\varphi \to f$, что означает равенство нулю функции φ на тех наборах, на которых $f = 0$.

Запишем функцию, заданную в таблице 1, в виде совершенной дизъюнктивной нормальной формы (СДНФ):

$$f(x_1, x_2, x_3) = \overline{x_1}\,\overline{x_2}\,\overline{x_3} \vee \overline{x_1}\,x_2\,\overline{x_3} \vee \overline{x_1}\,x_2\,x_3. \qquad (1)$$

Импликантой является любой конъюнктивный терм, а также любое подмножество термов, соединенных знаком дизъюнкции. Если $f = 0$, то каждый терм СДНФ должен быть равен нулю.

Первичная импликанта (ПИ) – импликанта типа элементарной конъюнкции, никакая часть которой не является импликантой. Так, в приведенном примере $\overline{x_1}\,x_2\,x_3$ – импликанта функции f; $\overline{x_1}\,x_2$, $\overline{x_1}\,\overline{x_3}$ – первичные импликанты, полученные склеиванием записанных термов (первого и второго, второго и третьего, соответственно).

Сокращенная ДНФ представляет собой дизъюнкцию первичных импликант:

$$f = (\overline{x_1}\,x_3 \vee \overline{x_1}\,x_2). \qquad (2)$$

В общем случае сокращенная ДНФ задает функцию f в виде $f = p_1 \vee p_2 \vee \ldots \vee p_к$, где $p_1, p_2, \ldots, p_к$ – первичные импликанты.

Сокращенная ДНФ единственна, так как единственным является список ПИ. Но в сокращенной ДНФ могут присутствовать лишние импликанты, которые можно удалить. В результате исключения лишних импликант получим *тупиковую ДНФ (ТДНФ)*. ТДНФ – форма представления ЛФ в виде дизъюнкций ПИ, в которых ни одна имликанта не является лишней. ТДНФ не единственна.

Минимальная ДНФ (МДНФ) – это ТДНФ, имеющая минимальную цену покрытия.

Цена покрытия отражает количество термов и переменных функции. Чем меньше цена покрытия, тем ближе форма представления функции к МДНФ.

МДНФ в общем случае не единственна, может быть несколько равноценных форм. Используются две разновидности цены покрытия: $C^a = \sum_r q_r(n-r)$, $C^b = C^a + l$.

Здесь r – обозначение размерности куба (эта характеристика определена в разделе 2.2);

q_r – число r-кубов в ДНФ;

n – число переменных функции;

l – общее число термов в ДНФ.

C^a для формулы (1) равна 9: $C^a = 3 \cdot 3$, где первый множитель – число r-кубов (в данном случае 0-кубов); второй множитель – число переменных в 0-кубах.

C^a для формулы (2) равна 4: $C^a = 2 \cdot (3\text{-}1)$, так как формула содержит два 1-куба.

Для формул (1) и (2): $C^b = 12$ и $C^b = 6$, соответственно.

Все приведенные определения могут быть даны (в модифицированном виде) и для двойственных понятий, а именно – для конъюнктивных форм представления функции.

2.2. Минимизация логических функций

Задача получения минимальной формы представления логической функции многих переменных относится к числу труднорешаемых, поскольку объем вычислений, необходимых для точного решения задачи, может экспоненциально возрастать с ростом числа переменных. Описываемый ниже метод Квайна – Мак-Класски является фундаментальным алгоритмическим методом, приложимым не только к задаче минимизации ЛФ, но и к ряду других задач дискретной оптимизации, в которых требуется найти минимальное покрытие бинарной матрицы.

2.2.1. Метод Квайна – Мак-Класски. Изложение метода ведется на примере, что не ограничивает общность представления метода. Зададим функцию:

$f(x_1, x_2, x_3, x_4) = \vee (3, 4, 5, 7, 9, 11, 12, 13)$.

Метод включает три этапа:

1. Нахождение первичных импликант.

2. Построение таблицы покрытий.

3. Отыскание минимального покрытия.

Выполнение этапов.

1. Выпишем 0-кубы (комплекс K^0), затем перепишем его, сгруппировав кубы по числу единиц. Два 0-куба склеиваются, образуя 1-куб, если они отличаются одной координатой (на ее месте записывается символ "X"), и после склеивания обязательно отмечаются, например, знаком "$\vee$". Кубы внутри групп не склеиваются. Склеиваться могут только 0-кубы, из соседних групп, что сокращает объем перебора.

Комплекс $K^0(f)$:

0	0	1	1		0	1	0	0	∨
0	1	0	0		0	0	1	1	∨
0	1	0	1		0	1	0	1	∨
0	1	1	1	=	1	0	0	1	∨
1	0	0	1		1	1	0	0	∨
1	0	1	1		0	1	1	1	∨
1	1	0	0		1	0	1	1	∨
1	1	0	1		1	1	0	1	∨

Совокупность 1-кубов образует кубический комплекс $K^{(1)}$. Для склеивания перепишем комплекс $K^{(1)}$, расположив 1-кубы по группам, в которых расположение крестов в столбцах одинаково, а внутри каждой группы сгруппируем подгруппы по числу единиц. Склеивание 1-кубов производится внутри групп, причем пары, подлежащие склеиванию, выбираются из соседних подгрупп. Склеенные кубы отмечаются.

Комплекс $K^{(1)}(f)$:

0	1	0	X		X	1	0	0	∨
X	1	0	0		X	0	1	1	
0	X	1	1		X	1	0	1	∨
X	0	1	1	=	0	X	1	1	
0	1	X	1		1	X	0	1	
X	1	0	1		0	1	X	1	
1	0	X	1		1	0	X	1	
1	X	0	1		0	1	0	X	∨
1	1	0	X		1	1	0	X	∨

Легко видеть, что результат склеивания 1-кубов – комплекс $K^{(2)}$ – включает единственный 2-куб: $K^{(2)}(f)$ = X10X. Объединение всех кубических комплексов K^0, $K^{(1)}$, $K^{(2)}$, … образует кубический комплекс $K(f)$, а все неотмеченные кубы комплекса $K(f)$ образуют множество первичных импликант (ПИ):

ПИ =

X	0	1	1
0	X	1	1
1	X	0	1
0	1	X	1
1	0	X	1
X	1	0	X

По первичным импликантам можно выписать сокращенную ДНФ:

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

На промежуточном этапе минимизации оценим характеристики минимизации.

Для исходной СДНФ: $C^a = 32$, $C^b = 40$; для сокращенной ДНФ: $C^a = 17$, $C^b = 23$.

Наша цель – найти *минимальное покрытие* кубического комплекса K^0 подходящим подмножеством первичных импликант, т. е. отыскать МДНФ, удалив *лишние* импликанты. Этой цели служат два следующих этапа.

2. Построение таблицы покрытий.

0-кубы / ПИ	0011	0100	0101	0111	1001	1011	1100	1101
X011	*					*		
0X11	*			*				
1X01					*			*
01X1			*	*				
10X1					*	*		
X10X		*	*				*	*

3. Отыскание минимального покрытия.

Оставляем минимальное число первичных импликант, которые в совокупности покрывают все столбцы. Такие импликанты назовем *существенными*, а остальные *несущественными* или *лишними*.

Поскольку второй столбец покрывается единственным образом, то выделенная импликанта (последняя в списке), заведомо является существенной; столбцы, которые она покрывает, можно вычеркнуть, а импликанту запомнить. Подобным образом

на различных шагах поиска определяются и другие существенные импликанты, выбор которых обязателен.

В общем случае алгоритм для отыскания минимального покрытия бинарной матрицы, может быть построен с использованием следующих пунктов:

1. Удаление поглощенных строк.
2. Удаление поглощающих столбцов.
3. Выбор *заведомо существенной*, как было разъяснено выше, строки (в нашем методе – соответствующей первичной импликанты).

Эти три пункта могут повторяться сколько угодно раз в любой последовательности. Ниже приведен пример поглощенной строки.

* * *

* * ← поглощенная строка

Поглощенная строка содержит подмножество отметок "*", имеющихся в некоторой другой (*поглощающей*) строке. Поглощенная строка может быть удалена по очевидным причинам. При обнаружении пары столбцов, один из которых является поглощающим, а другой – поглощенным, удалить следует первый из них.

* *

*

* *

↑

поглощающий столбец

В нашем случае останутся три существенные импликанты:

0	X	1	1
1	0	X	1
X	1	0	X

МДНФ: $f(x_1, x_2, x_3, x_4) = \overline{x_1}\, x_3\, x_4 \vee x_1\, \overline{x_2}\, x_4 \vee x_2\, \overline{x_3}$. Характеристики найденной формы:

$$C^a = 8,\ C^b = 11.$$

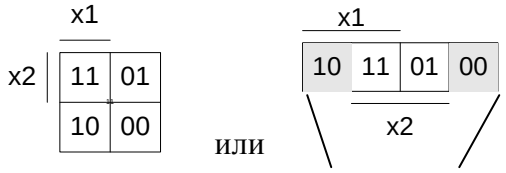
Отметим, что при возрастании "размера задачи" (числа переменных и числа 0-кубов) указанные выше пункты алгоритма на отдельных шагах его работы могут оказаться неприменимыми. В этом случае, если отыскивается точное решение, возникает необходимость перебора по дереву решений, поэтому данная задача в общей постанов-

ке является трудноразрешимой. Альтернативой точного решения является приближенное, в частности *субоптимальное* решение, которое достигается, например, применением эффективного "жадного" алгоритма. Жадный алгоритм, применительно к рассматриваемой задаче, заключается в выборе на каждом шаге строки таблицы, покрывающей наибольшее число столбцов, с последующим удалением покрытых столбцов и, соответственно, сокращением размеров таблицы.

2.2.2. Карты Карно

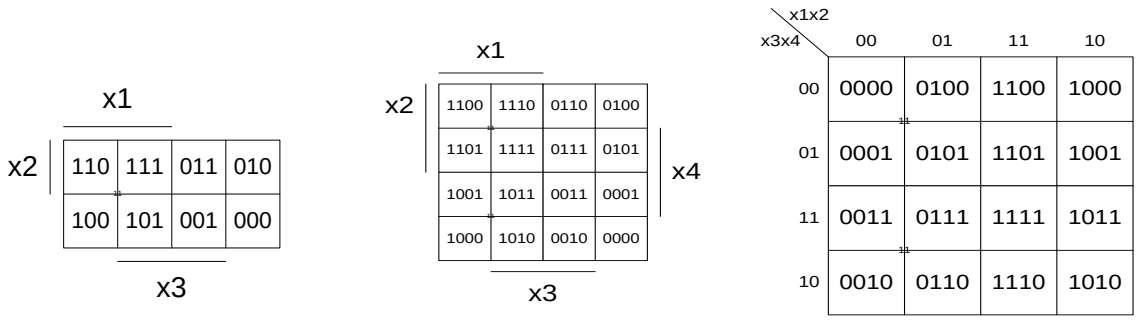
Карты Карно (КК) можно рассматривать как развертки геометрических кубов на плоскости. Карта Карно, предназначенная для задания и минимизации функций n переменных, имеет 2^n клеток. Каждая клетка соответствует набору переменных. В тех клетках, которые соответствуют значениям функции, равным 1, записываются единицы, а в остальных – нули (или вместо нулей оставляются пустые клетки).

Разметка карт. Разметка предназначена для установления взаимно однозначного соответствия между наборами ЛФ и клетками. Начнем с $n = 2$:

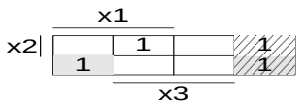


Соседние наборы различаются только одной координатой

$n = 3$: $n = 4$ (даны два варианта возможной разметки):



Пример заполнения карты и минимизации для $n = 3$.



. $f(x_1, x_2, x_3)$

Исходная функция: $f = \overline{x_1}\,\overline{x_2}\,\overline{x_3} \vee x_1\,\overline{x_2}\,\overline{x_3} \vee \overline{x_1}\,x_2\,\overline{x_3} \vee x_1\,x_2\,x_3$ ($C^a = 12$, $C^b = 16$). Соседние клетки, заполненные единицами, можно объединить в контур, используя $2^к$

соседних клеток. Каждому контуру будет соответствовать конъюнктивный терм ДНФ. Результат минимизации: $f = \overline{x_1}\ \overline{x_3} \vee \overline{x_2}\ \overline{x_3} \vee x_1\, x_2\, x_3$ ($C^a = 7$, $C^b = 10$).

Правила минимизации

1. Две соседние *единичные* клетки исходной карты (два 0-куба) образуют 1-куб. Клетки на границах также являются соседними. При этом независимая координата, обозначавшаяся ранее символом "X", при аналитической записи исчезает из терма.

2. Четыре соседние единичные клетки могут объединяться в контур, образуя 2-куб, при этом исчезают две переменные. Каждая клетка в объединении имеет две соседние клетки.

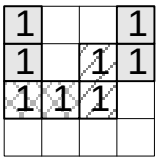
3. Восемь соседних единичных клеток могут объединяться в контур, образуя 3-куб. Каждая клетка в объединении имеет три соседние клетки.

4. Шестнадцать соседних единичных клеток могут объединяться в контур, образуя 4-куб, и т. д.

5. Каждому контуру, включающему $2^к$ клеток, в аналитической записи соответствует конъюнктивный терм минимизированной ДНФ, содержащий $n - k$ переменных; отрицания над переменными расставляются согласно разметке карты; полученные термы в формуле соединяются знаком "$\vee$".

Объединение клеток отражает факт склеивания соответствующих термов при аналитической записи. Цель использования карт Карно для получения минимальной ДНФ состоит в следующем: *покрыть все единицы карты как можно меньшим числом как можно более крупных контуров*. В общем случае возможны разные варианты покрытия; не самое удачное покрытие имеет следствием получение правильной, но не оптимальной (по цене покрытия) формулы.

Приведем пример оптимального покрытия карты при $n = 4$:



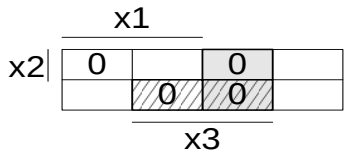
МДНФ включает три терма, один из которых содержит две переменные, а два остальных – по три переменных.

Получение МКНФ

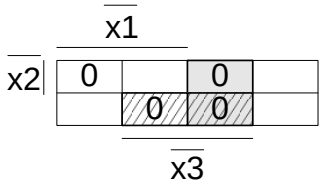
Получение МКНФ по карте Карно может основываться на предварительном получении МДНФ для инверсной функции; взятие еще одного отрицания с применением правил де Моргана приведет к записи КНФ: $\overline{f} = \ldots \Rightarrow \overline{\overline{f}} = f$.

Получение МКНФ непосредственно по карте

Используем карту с проставленными нулями, находим контуры и, подразумевая инверсную разметку карты, сразу записываем МКНФ.



Инверсную разметку можно непосредственно нанести на карту:

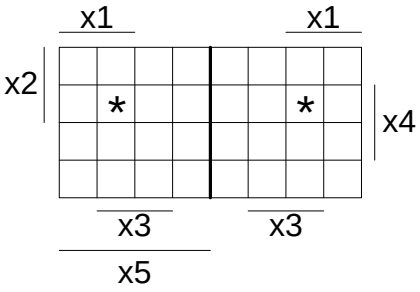


$f = (\overline{x_1} \vee \overline{x_2} \vee x_3)(x_1 \vee \overline{x_3})\,(x_2 \vee \overline{x_3})$ – МКНФ.

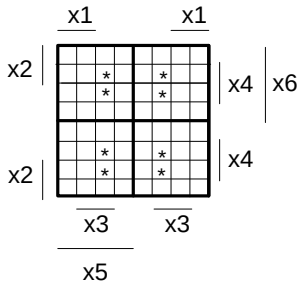
Карты Карно для n > 4

Карты Карно для $n > 4$ менее наглядны и требуют большего внимания при поиске контуров. Клетки, зеркально отраженные относительно центральных осей, также являются соседними. Ниже показаны карты и их разметка для $n = 5$ и $n = 6$. Клетки, отмеченные символом "*", – соседние и могут включаться в общий контур.

$n = 5$:



$n = 6$:



Разметка карт может осуществляться по-разному: допустимы переименования переменных и инверсная разметка. Однако в любом случае для любой переменной и ее отрицания отводятся по половине карты.

2.3. Теорема Шеннона о разложении логической функции

Любую ЛФ от n переменных можно первоначально представить в виде:

$$f(x_1, x_2, \dots, x_n) = f(1, x_2, \dots, x_n)\, x_1 \vee f(0, x_2, \dots, x_n)\, \overline{x_1}. \qquad (3)$$

Для доказательства достаточно положить поочередно в равенстве (3) значение x_1 равным 1 и 0.

Если применить (3) последовательно ко всем переменным, то мы получим следующее равенство:

$$f(x_1, x_2, \dots, x_n) = f(1,1, \dots, 1)\, x_1 x_2 \dots x_n \vee f(0,1, \dots, 1)\, \overline{x_1}\, x_2 \dots x_n \vee f(0,0, \dots, 0)\, \overline{x_1}\, \overline{x_2} \dots \overline{x_n}. \quad (4)$$

Всего в правой части (4) может быть 2^n слагаемых. В более краткой записи выражение (4) имеет вид:

$$f = \bigvee_{(l_1, l_2, \dots, l_n)} f(l_1, l_2, \dots, l_n)\, x_1^{l_1} x_2^{l_2} \dots x_n^{l_n}, \qquad (5)$$

где l_i – элемент множества $\{0, 1\}$, при этом $x_i^{l_i} = \begin{cases} x, \text{ если } l_i = 1 \\ \overline{x}, \text{ если } l_i = 0 \end{cases}$.

Удалив из (5) слагаемые, на которых f принимает значение "0", получим СДНФ:

$$f = \bigvee_{(l_1, l_2, \dots l_n)} x_1^{l_1} \cdot x_2^{l_2} \cdot \dots \cdot x_n^{l_n}. \qquad (6)$$

В (6) суммирование ведется только по наборам, на которых $f = 1$.

Аналогично, из принципа двойственности получим СКНФ:

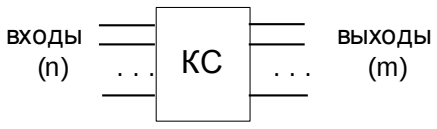
$$f = \bigwedge_{(l_1, l_2, \dots l_n)} x_1^{\overline{l_1}} \cdot x_2^{\overline{l_2}} \cdot \dots \cdot x_n^{\overline{l_n}} \qquad (7)$$

В (7) в произведение включаются только наборы, на которых $f = 0$.

Доказательство существования совершенных форм (6) и (7) и составляет содержание теоремы Шеннона.

2.4. Анализ и синтез комбинационных схем

2.4.1. Общие сведения



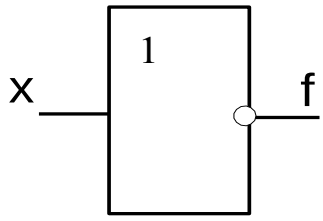
Комбинационная схема (КС) – автомат без памяти, строится на логических элементах и, как правило, не имеет обратных связей. Таблица истинности, описывающая работу КС, состоит из 2^n строк и $m + n$ столбцов.

КС можно рассматривать на логическом или физическом уровне. На логическом уровне рассматриваются "идеальные" КС, в которых выходные сигналы появляются одновременно с входными и зависят только от них (значения сигналов – 0 или 1).

Физический уровень рассмотрения связан с учетом реальных характеристик элементов и схем: переходных процессов и задержек при переключении и распространении сигналов, нагрузочной способности и потребляемой мощности.

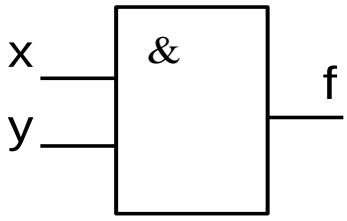
В нашем курсе основным является логический уровень рассмотрения; при этом большинство реальных характеристик логических элементов не рассматривается, за исключением задержек – в тех случаях, когда изучается их влияние на правильность работы схем. Рассмотрим основные логические элементы (ЛЭ) и реализуемые ими логические функции, которые называют также *логическими операторами* (ЛО).

Инвертор



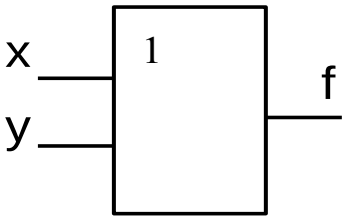
$f(x) = \overline{x}$

Конъюнктор



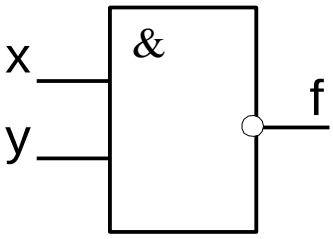
$f(x, y) = xy$

Дизъюнктор



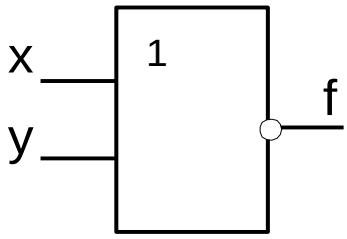
$f(x, y) = x \vee y$

И-НЕ

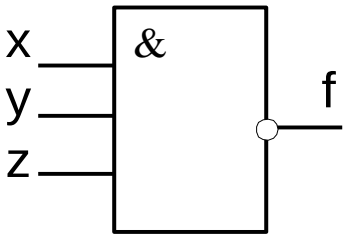


$f(x, y) = \overline{xy}$

ИЛИ-НЕ

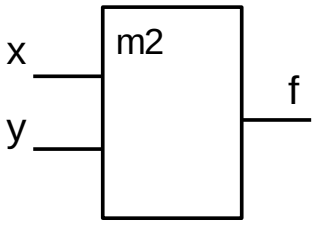


$f(x, y) = \overline{x \vee y}$



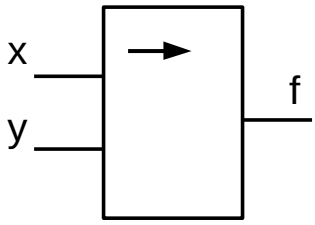
$f(x, y, z) = \overline{xy}z$

Сумматор по mod 2



$f(x, y) = x \oplus y$

Импликатор



$f(x, y) = x \rightarrow y$

На рис.1 приведены временные диаграммы сигналов на выходах логических элементов.

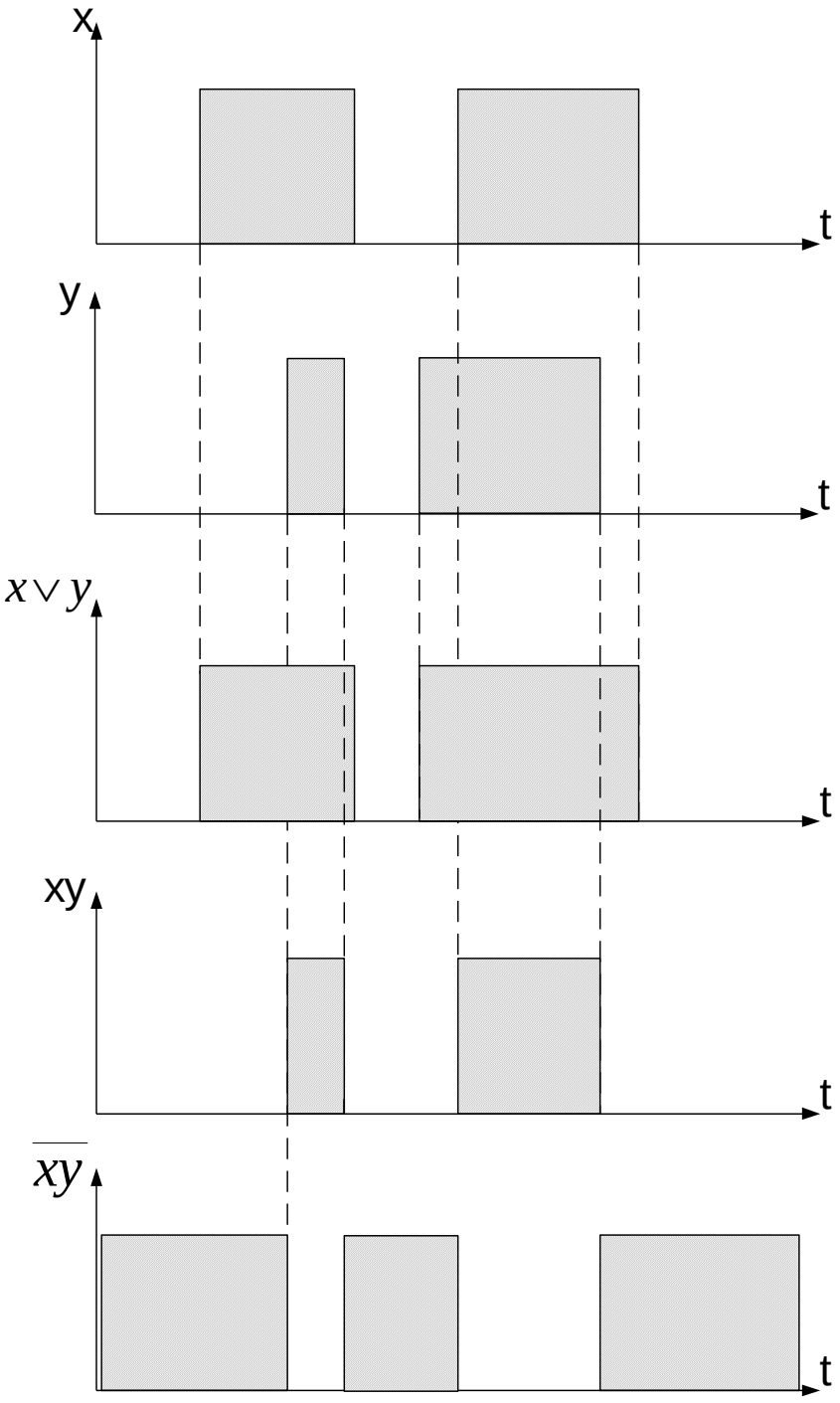


Рис.1. Временные диаграммы

На рис. 2 представлен пример изображения входного и выходного сигналов на временной диаграмме с учетом задержки на ЛЭ.

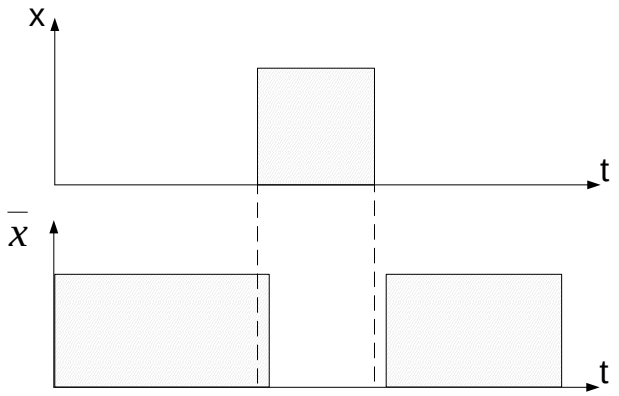
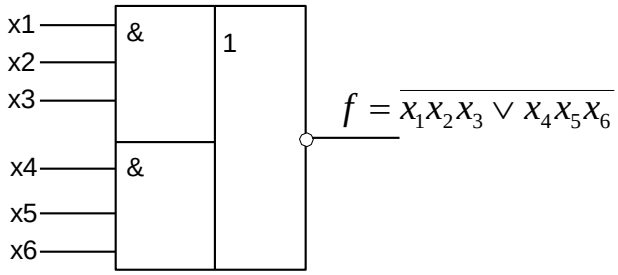


Рис. 2. Инвертор с задержкой сигнала

В схемотехнике используются и более сложные ЛЭ, реализующие суперпозиции логических операций, например:



2 И - ИЛИ - НЕ

2.4.2. Параметры реальных логических элементов и цифровых схем

1. Быстродействие (время переключения) – измеряется между фронтами входного и выходного сигналов на уровне 0,5(U_1 - U_0), где U_1 и U_0 – максимальный и минимальный уровни сигнала соответственно. В общем случае переключение выходного сигнала элемента или схемы из 0 в 1 и из 1 в 0 может осуществляться за разное время. В качестве характеристики задержки используется средняя величина: $t_{зад} = (t^{01} + t^{10})/2$.

2. Потребляемая мощность:

- статическая – измеряется при неизменном выходном сигнале; в качестве характеристики может использоваться средняя величина: $(P^0 + P^1)/2$, где P^0 – мощность, потребляемая при хранении значения 0, P^1 – при хранении значения 1;

- динамическая – измеряется при переключении сигнала с определённой частотой; в среднем потребляемая мощность растёт с увеличением частоты.

3. Нагрузочная способность (коэффициент разветвления по выходу) – определяет число ЛЭ, которые могут подключаться к выходу данного ЛЭ. Увеличение нагрузочной способности приводит к повышению потребляемой мощности и понижению быстродействия элемента.

4. Коэффициент объединения по входу – выражается количеством одинаковых по назначению входов ЛЭ (2, 3, 4, 8 – ординарные значения коэффициентов). Использование многовходовых элементов связано с понижением быстродействия и ухудшением помехоустойчивости схемы; уменьшается и степень интеграции ИС.

2.4.3. Правила соединения логических элементов в схемах

Сформулируем правила соединения ЛЭ на логическом уровне, т. е. без учета их реальных характеристик, в частности нагрузочной способности:

1. ЛЭ имеют конечное число входов и один выход и реализуют некоторый логический оператор.

2. Выход ЛЭ можно подключить к любому числу входов других ЛЭ.

3. В качестве значений входов и выходов ЛЭ могут быть лишь константы «0» или «1».

4. Никакие два выхода ЛЭ нельзя соединять вместе.

Комбинационные схемы целесообразно строить и изображать по ярусам (каскадам). Рассмотрим пример КС для функции двух переменных (рис. 3):

$$f(x,y) = x \oplus y = x\bar{y} \vee \bar{x}y$$

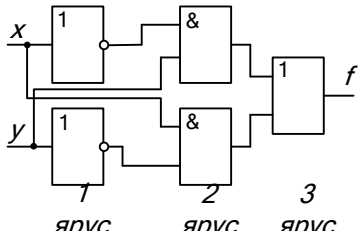


Рис. 3. Трехъярусная комбинационная схема.

Ярусное строение произвольной КС сводится к следующему:

- 1-й ярус содержит ЛЭ, входы которых являются входами всей схемы;

- 2-й ярус образуют ЛЭ, к входам которых подключаются в общем случае входы схемы и выходы элементов 1-го яруса;

- *i*-й ярус образуют ЛЭ, к входам которых подключаются выходы элементов предыдущих ярусов (*i* - 1, … , 1), а также входы схемы.

Существование СДНФ и СКНФ говорит о том, что теоретически любую КС можно сделать трехъярусной.

2.4.4. Задачи анализа и синтеза КС

Задача анализа ставится для заданной КС и может включать ряд подзадач, в частности, следующих:

- выявление реализуемой ЛФ;

- оптимизация схемы, например, исключение дублирования ее частей;

- отыскание тестов для схемы (множеств входных наборов, которые позволяют провести контроль или диагностику схемы).

Пример: найти математическое описание схемы (рис. 4).

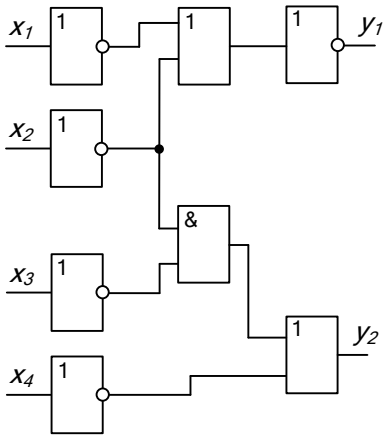


Рис. 4. Комбинационная схема

ЛФ представленной схемы легко находятся по ее структуре:

$y_1 = \overline{\overline{x_1} \vee \overline{x_2}} = x_1 x_2$; $y_2 = \overline{x_2}\ \overline{x_3} \vee \overline{x_4}$.

Задача синтеза КС – определить содержимое "чёрного ящика" (рис. 5):

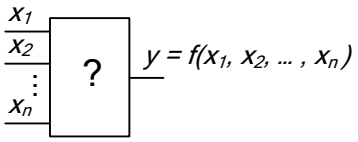


Рис. 5. "Черный ящик" с заданной функцией выхода

Этапы синтеза с учетом того, что КС может быть многовыходной:

а) составление математического описания, адекватно отображающего назначение схемы;

б) анализ выходных логических функций и их совместная минимизация в заданном базисе логических элементов;

в) переход к структурной схеме, реализующей полученные функции.

2.4.5. Синтез КС в заданном базисе

В основе синтеза лежит структурирование формул согласно правилам алгебры логики, среди которых особое место занимают правила де Моргана.

Пусть необходимо создать КС, которая реализует функцию $f = x_1x_2x_3x_4\overline{x_5} \vee \overline{x_2}\ \overline{x_3}$ на двухвходовых элементах И-НЕ. Отметим, что этот элемент соответствует логической операции "штрих Шеффера", образующей в логике функционально полную систему. Заметим также, что инвертор может быть получен соединением входов элемента И-НЕ.

Структурирование формулы:

$$f = x_1x_2x_3x_4\overline{x_5} \vee \overline{x_2}\ \overline{x_3} = \overline{\overline{x_1x_2x_3x_4\overline{x_5}} \vee \overline{\overline{x_2}\ \overline{x_3}}} = \overline{\overline{x_1x_2x_3x_4\overline{x_5}} \cdot \overline{\overline{x_2}\ \overline{x_3}}}.$$

Преобразуем отдельно подформулу: $x_1x_2x_3x_4\overline{x_5} = \overline{\overline{x_1x_2} \cdot \overline{x_3x_4} \cdot \overline{x_5}} = \overline{\overline{x_1x_2} \cdot \overline{\overline{x_3x_4} \cdot \overline{x_5}}}$.

Теперь можно записать: $f = \overline{\overline{x_1x_2} \cdot \overline{x_3x_4} \cdot \overline{x_5} \cdot \overline{x_2}\ \overline{x_3}}$. Схема, реализующая полученную формулу, изображена на рис. 6.

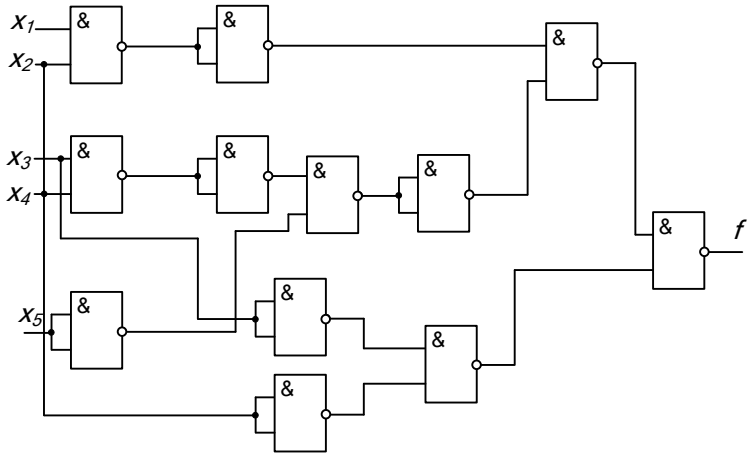


Рис. 6. Комбинационная схема, построенная в заданном базисе

2.4.6. Синтез КС с несколькими выходами

Задача синтеза КС с n входами и k выходами отличается от задачи синтеза КС с одним выходом тем, что необходимо исключить дублирование в схемах, представляющих k логических функций. Исключение дублирования может быть основано на выделении первичных импликант для заданной системы ЛФ. Последовательность действий:

- отыскание ПИ для системы ЛФ;

- представление каждой ЛФ через первичные импликанты;

- синтез КС, отображающий только эти ПИ и связи между ними.

Пример. Даны две функции: $y_1 = x_1x_2 \vee \overline{x_2}x_3$, $y_2 = x_1\overline{x_2}\,\overline{x_3} \vee x_1x_2\overline{x_3} \vee x_1x_2x_3$.

Первичные импликанты в данном случае легко получить простым склеиванием термов. Выражение функций y_1 и y_2 через ПИ приводит к формулам:

$$y_1 = \underline{x_1x_2} \vee \overline{x_2}\,x_3,\ y_2 = x_1\overline{x_3} \vee \underline{x_1x_2}.$$

В схеме, построенной по полученным формулам, общей (подчеркнутой) ПИ соответствует один логический элемент (рис. 7).

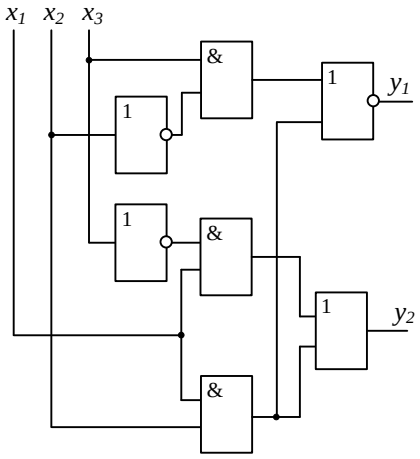


Рис. 7. КС с общей частью для выходных функций

Дешифратор. Дешифратор (сокращения: ДШ или DC) – комбинационная схема с несколькими входами и выходами, преобразующая код на входе в единичный сигнал на одном из выходов. ДШ с n входами имеет 2^n выходов, которые нумеруются числами 0, 1, ... , m, где $m = 2^n - 1$ (рис. 8, справа – дешифратор с тремя входами).

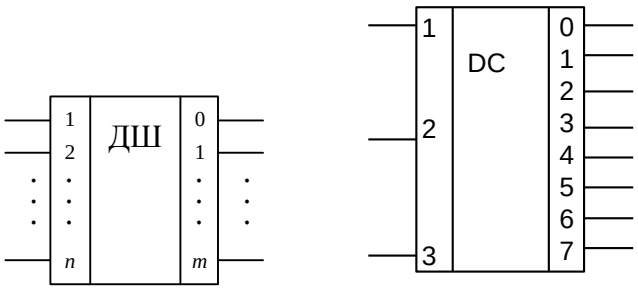


Рис. 8. Дешифратор

В каждой строке таблицы истинности, описывающей ДШ, значение "1" будет записано только для одной выходной функции. Ниже приведена ТИ для дешифратора с тремя входами.

Таблица истинности

Наборы			Функции							
x_1	x_2	x_3	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7
0	0	0	**1**	0	0	0	0	0	0	0
0	0	1	0	**1**	0	0	0	0	0	0
0	1	0	0	0	**1**	0	0	0	0	0
0	1	1	0	0	0	**1**	0	0	0	0
1	0	0	0	0	0	0	**1**	0	0	0
1	0	1	0	0	0	0	0	**1**	0	0
1	1	0	0	0	0	0	0	0	**1**	0
1	1	1	0	0	0	0	0	0	0	**1**

Комбинационная схема дешифрации трехразрядного входного кода, представленного прямыми и инверсными сигналами, может быть построена на трехвходовых конъюнкторах – в соответствии с приведенной выше таблицей истинности (рис. 9).

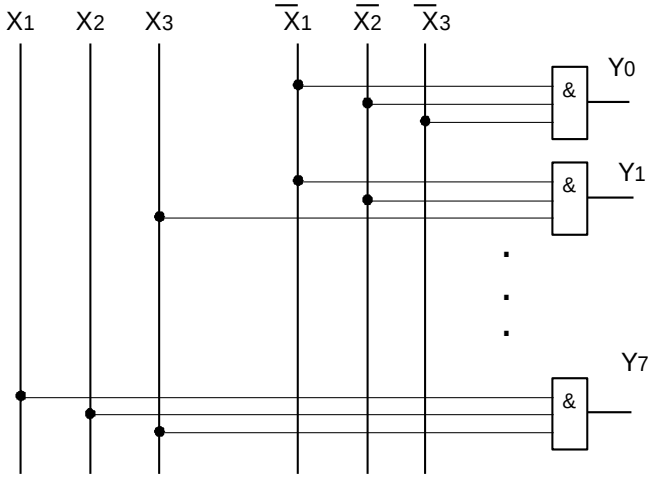


Рис. 9. Комбинационная схема дешифрации

Дешифратор как преобразователь кодов

Дешифратор, помимо основного назначения, может использоваться в качестве преобразователя кодов, что упрощает в ряде случаев проектирование комбинационных схем. Пусть требуется построить КС, которая описывается приведенной ниже таблицей.

Таблица преобразования кодов

	x_1	x_2	x_3	y_1	y_2	y_3	y_4
p_0	0	0	0	0	0	0	0
p_1	0	0	1	0	0	1	1
p_2	0	1	0	0	1	0	1
p_3	0	1	1	0	1	1	0
p_4	1	0	0	1	1	0	0
p_5	1	0	1	1	1	1	1

Слева от таблицы записан столбец с новыми переменными p_i ($i = 0, \dots, 5$), обозначающими входные наборы. Эти же переменные обозначают выходы дешифратора с тремя входами – x_1, x_2, x_3; разряды преобразованного кода при этом описываются соотношениями: p_2

$$y_1 = p_4 \vee p_5;\ y_2 = p_2 \vee p_3 \vee p_4 \vee p_5;\ y_3 = p_1 \vee p_3 \vee p_5;\ y_4 = p_1 \vee p_2 \vee p_5. \qquad (6)$$

Читатель легко построит по соотношениям (6) комбинационную схему, которая подключается к выходам дешифратора.

2.5. Не полностью определенные логические функции

Не полностью (частично) определенная логическая функция от n переменных – это функция, заданная на наборах, число которых менее 2^n.

Если число наборов, на которых функция не определена равно m, то посредством различных доопределений можно получить из этой функции 2^m полностью определенных функций. Доопределения могут выполняться исходя из некоторых условий, например, из условий наилучшей минимизации.

В дальнейшем неопределенные значения ЛФ, а также и сами наборы, на которых ЛФ не определена, будем отмечать символом "*". При подходящих условиях доопределения этот символ заменяется нулем или единицей. Поскольку от выбора доопределения зависит конечный результат минимизации, выполняя его, можно руководствоваться правилами, которые отчасти аналогичны правилам метода Квайна – Мак-Класски:

1. Получаем из частично определенной функции f функцию φ_0 путем замены всех неопределенных значений (отмеченных *) нулями.

2. Получаем из f функцию φ_1 путем замены всех неопределенных значений единицами. Функции φ_0 и φ_1 являются полностью определенными.

3. Находим минимальное покрытие комплекса $К^0(\varphi_0)$ первичными импликантами функции φ_1.

Сформулированные правила обоснованы следующими соображениями. Функция φ_1 содержит все ПИ, которые могут встретиться в покрытиях функции f при различных доопределениях, а функция φ_0 содержит минимальное число термов в СДНФ (и, следовательно, минимальное число ноль-кубов); кроме того, обе функции адекватно представляют функцию f на тех наборах, где f определена.

Минимизация частично определенных функций при помощи карт Карно.

При использовании карт Карно для частично определенных функций в клетках карты, кроме 0 и 1, записываются символы "*", которые соответствуют произвольным значениям функций. При нахождении оптимальных контуров, можно любые выбранные клетки, помеченные *, присоединять к клеткам, помеченных единицами (при нахождении МДНФ), или нулями (при нахождении МКНФ). Такое присоединение соответствует выбору некоторого варианта доопределения и способствует наилучшей минимизации.

Пример 1. Построение дешифратора для выходных кодов синхронного кодового кольца (СКК). СКК – счетчик (автомат Мура), на выходе которого при подаче входных импульсов образуется циклическая последовательность кодов:

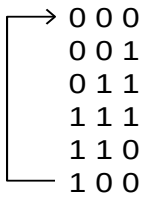
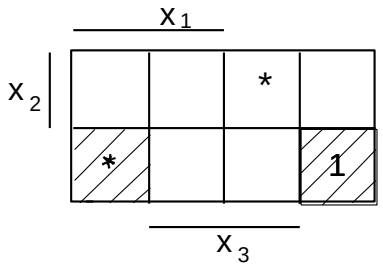


Таблица выходных функций дешифратора

Наборы			Функции					
x_1	x_2	x_3	f_0	f_1	f_2	f_3	f_4	f_5
0	0	0	**1**	0	0	0	0	0
0	0	1	0	**1**	0	0	0	0
0	1	1	0	0	**1**	0	0	0
1	1	1	0	0	0	**1**	0	0
1	1	0	0	0	0	0	**1**	0
1	0	0	0	0	0	0	0	**1**
0	1	0	*	*	*	*	*	*
1	0	1	*	*	*	*	*	*

Карта Карно для функции f_0 имеет следующий вид:



В заштрихованный контур включен символ *, следовательно, $f_0 = \overline{x_2}\ \overline{x_3}$. Аналогично, на картах Карно для каждой из функций $f_1, \ldots, f_5$ будут присутствовать два таких символа, один из которых может быть присоединен к единственной единице. В целом это приведет к упрощению схемы и повышению ее надежности, так как вместо трехвходовых элементов конъюнкции будут использованы двухвходовые.

Пример 2. Построение КС с двумя выходами.

Пусть необходимо синтезировать схему, выходы которой описываются функциями $y_1 = f_1(x_1, x_2, \ldots, x_n)$, $y_2 = f_2(x_1, x_2, \ldots, x_n)$. Допустим, что схема для y_1 уже построена, тогда можно использовать следующий прием: ввести частично определенную функцию $y_2' = f_2'(x_1, x_2, \ldots, x_n, x_{n+1})$, где $x_{n+1} = y_1$. Значения функции y_2' определены только на половине наборов. Далее выполняется минимизация функции y_2' одним из рассмотренных выше способов, разработанных для частично определенных функций, что обычно приводит к упрощению выражения, описывающего второй выход схемы и, соответственно, к целесообразному построению схемы в целом.

2.6. Особенности проектирования комбинационных схем с учетом задержек

Функциональная надежность логической схемы характеризует точное выполнение схемой алгоритма ее функционирования. Однако в реальной схеме, вследствие задержек сигналов на логических элементах, при переключении значений входов на выходах могут возникать сигналы, не соответствующие алгоритму функционирования. Задержки порождают "состязания" сигналов, распространяющихся по параллельным цепям, и вызывают неустойчивую работу *цифровых схем* (ЦС); последний термин служит для обозначения как комбинационных схем, так и схем с памятью.

Реальный логический элемент (РЛЭ) можно представить как последовательное соединение идеального ЛЭ с элементом задержки (ЭЗ) (рис. 10).

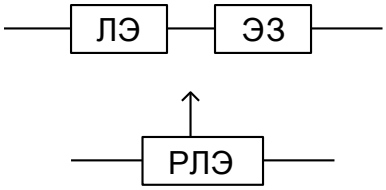


Рис. 10. Модель представления реального логического элемента

Задержки бывают двух видов:

1. Безинерциальная – сдвигает сигнал на время т, не искажая его.

2. Инерциальная – сдвигает сигнал на время т, если длительность сигнала больше, чем т, и не пропускает сигналы меньшей длительности (рис. 11).

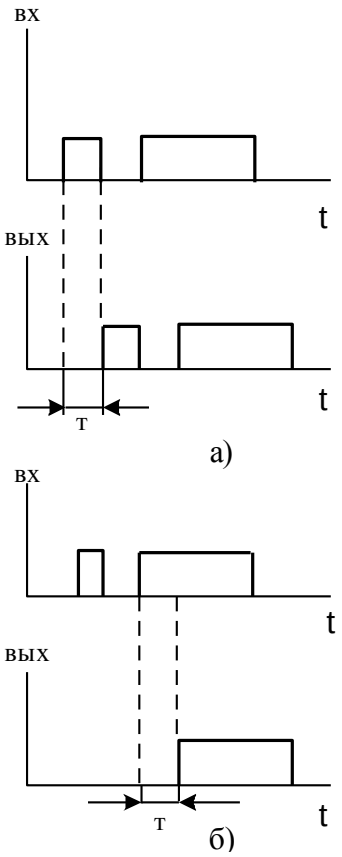


Рис. 11. Виды задержки сигнала: а – безинерциальная; б - инерциальная

Различают *статические* и *динамические* состязания в КС.

Статические состязания могут возникать тогда, когда для двух последовательных состояний входов схемы значение выхода должно оставаться неизменным.

Пример. Входной и выходной сигналы для схемы, изображенной на рис. 12, приведены на рис. 13 (задержки сигналов на всех ЛЭ одинаковы и равны τ).

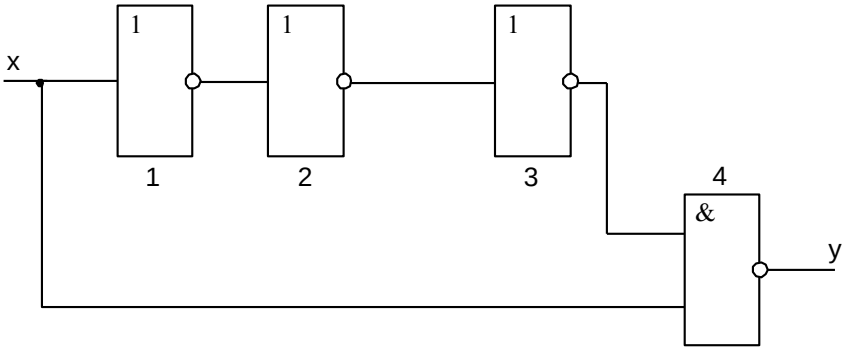


Рис.12. Комбинационная схема, построенная на реальных ЛЭ

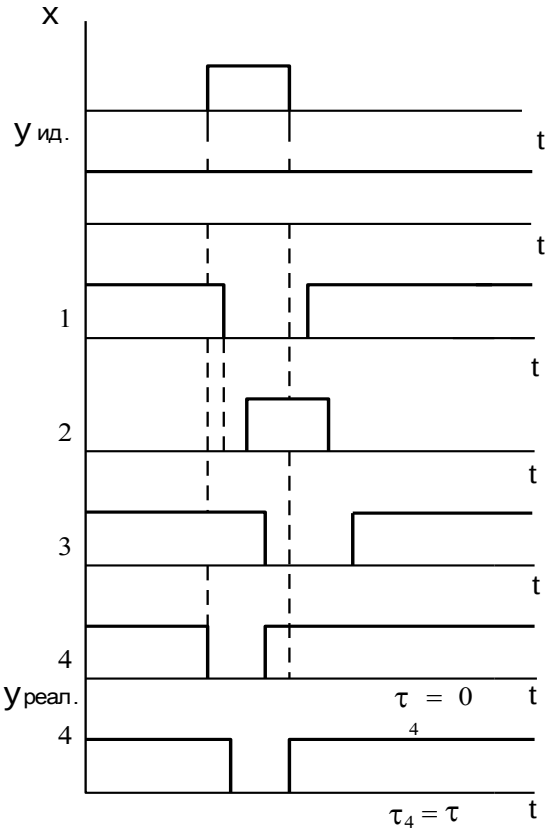


Рис.13. Временные диаграммы, иллюстрирующие статические состязания

Динамические состязания могут возникать в ситуации, когда на выходе схемы алгоритмический переход должен иметь вид 0 – 1 или 1 – 0. Правильный алгоритмический переход может предваряться ложным импульсом. Однако при этом передний фронт ложного импульса имеет то же направление перехода, что и передний фронт последующего алгоритмического сигнала (рис. 14).

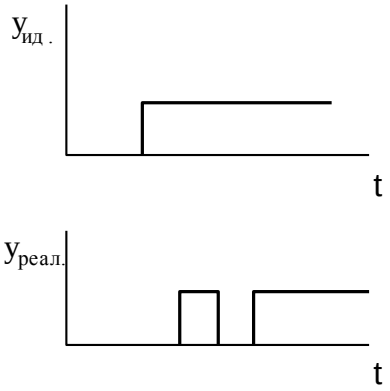


Рис. 14. Временные диаграммы, иллюстрирующие динамические состязания

Из двух видов состязаний более опасны статические, так как возникающий при статических состязаниях неалгоритмический импульс может быть зафиксирован элементом памяти, что может вызвать неправильную работу схемы в целом. При динамических состязаниях алгоритмический переход подтверждает предварительно возникший переход из-за ложного импульса, что является положительным фактором.

2.7. Способы проектирования комбинационных схем, свободных от состязаний

1. Введение структурной избыточности. Метод Хаффмена.

Основная идея метода: для получения схемы, свободной от состязаний, необходимо и достаточно для каждой пары смежных (т. е. отличающихся на один бит) состояний входов, для которых выходная функция схемы имеет одноименные значения (нулевые либо единичные), найти по крайней мере один терм функции, который покрывает оба состояния. Применительно к картам Карно это означает, что каждая пары соседних одинаково отмеченных клеток карты должна быть включена в общий контур (рис.15).

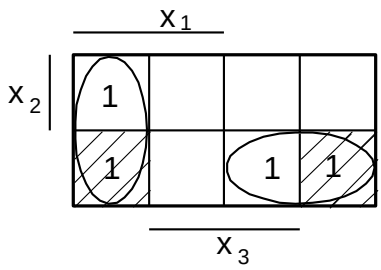


Рис. 15. Включение в покрытие избыточного контура

Минимальное покрытие карты Карно определяет функцию $y_1 = x_1 \overline{x_3} \vee \overline{x_1}\,\overline{x_2}$, соответствующую наиболее простой схеме. Введение избыточного контура (на рисунке

этот контур заштрихован) приводит к функции $y_2 = x_1\overline{x_3} \vee \overline{x_1}\,\overline{x_2} \vee \overline{x_2}\,\overline{x_3}$, описывающей более сложную, но и более надежную схему.

2. Введение дополнительных "стробирующих" (как правило, импульсных) сигналов, которые разрешают прохождение определенных сигналов схемы лишь после окончания переходных процессов (рис. 16).

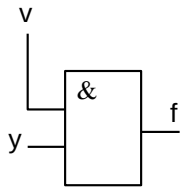


Рис. 16. Использование стробирующего сигнала (v), подключенного к элементу "И"

Как видно из рисунка, выходной сигнал y некоторой схемы или логического элемента появляется в виде результирующего сигнала f только при значении $v = 1$.

3. Коррекция возможных состязаний в схеме путем подбора задержек логических элементов. Этот способ может применяться лишь для отдельных, обычно экспериментальных, схем и не рассчитан на массовое производство.

ЗАДАЧИ И УПРАЖНЕНИЯ

1. Минимизировать при помощи карт Карно функцию, заданную в разделе 2.2.1:

а) получить МДНФ;

б) получить МКНФ.

2. Одноразрядный двоичный сумматор имеет на входе двоичные операнды a_i, b_i, $p_{i\text{-}1}$, где a_i и b_i – слагаемые, $p_{i\text{-}1}$ – сигнал переноса из предыдущего (i-1)-го разряда. Выходными функциями сумматора от трех указанных переменных являются сигнал суммы c_i и сигнал переноса в следующий разряд p_i. Выполнить следующие задания:

а) составить таблицу истинности для сумматора;

б) описать выходные сигналы в виде логических функций в базисе $\{\vee, \&, \neg\}$;

в) минимизировать логические функции при помощи карт Карно;

г) составить схему с двумя выходами для сумматора.

3. Синтезировать КС для функции, заданной в разделе 2.4.5, на двухвходовых элементах ИЛИ-НЕ.

4. Построить комбинационный преобразователь кодов, описанный таблицей в разделе 2.4.6, с использованием дешифратора.

5. Найти МДНФ для частично определенной логической функции

$$f(x_1, x_2, x_3, x_4) = \vee (0, 1^*, 2^*, 4^*, 6, 7^*, 8^*, 9, 11^*, 13^*, 14, 15^*)$$

методом, использующим построение полностью определенных функций φ_0 и φ_1 (см. раздел 2.5). *Примечание*. На наборах, номера которых отмечены *, значение функции не определено.

6. Минимизировать при помощи карт Карно все выходные функции дешифратора для синхронного кодового кольца (пример 1 из раздела 2.5).

7. Использовать метод, изложенный в разделе 2.5 (пример 2), для усовершенствования схемы одноразрядного двоичного сумматора. Исходить из того, что схема для функции p_i построена, поэтому можно считать p_i четвертой переменной функции c_i.

Литература

1. Карпов Ю. Г. Теория автоматов. *Учебник для ВУЗов*. – СПб.: ПИТЕР, 2002. - 206 с.
2. Савельев А. Я. Прикладная теория цифровых автоматов. *Учебник для ВУЗов* – М., ВШ, 1987. - 270 с.
3. Каган Б. М. Электронные вычислительные машины и системы. – М., "Энергия", 1979. - 527 с.
4. Захаров В. Н., Поспелов Д. А., Хазацкий В. Е. Системы управления. – М., "Энергия", 1982. - 346 с.