Министерство образования Российской Федерации

Дальневосточный государственный технический университет

им. В.В. Куйбышева

Л.И. Токмакова

СИСТЕМЫ УПРАВЛЕНИЯ ЭЛЕКТРОПРИВОДОВ

Учебное пособие
для студентов специальности 1804
Часть 1

Владивосток
2002

Издано по решению Редакционно-издательского совета ДВГТУ

ББК 32

УДК 621.30

Токмакова Л.И. Системы управления электроприводов (Часть 1): Учеб. пособие для вузов: В 2 ч. Ч1. – Владивосток: Изд-во ДВГТУ, 2002. – с: ил.

В учебном пособии даны методы логического проектирования управляющих логических устройств как комбинационного, так и последовательностного типа, являющихся составной частью большого числа систем управления электроприводов, показаны приемы схемной реализации, сформированы задания для практических занятий с примерами решения.

Уделяется внимание вопросам технической диагностики управляющих логических устройств: основным понятиям и методам построения проверяющих и диагностических тестов с использованием точных - на базе таблиц функций неисправностей (ТФН) и приближенных - на базе эквивалентной нормальной формы (ЭНФ) методов, позволяющих облегчить наладку и поиск дефектов логических устройств при их эксплуатации.

Учебное пособие предназначено для студентов вузов, обучающихся по направлению «Электромеханика, электротехника и электротехнологии». Может быть полезно для специалистов, занимающихся проектированием и наладкой устройств, базирующихся на интегральных микросхемах и других логических элементах.

Рецензенты: О.В. Абрамов, зав. лабораторией управления надежностью сложных систем Института автоматики и процессов управления ДВО РАН;

В.П. Кривошеев, профессор кафедры компьютерных технологий и систем ВГУЭС, д-р техн. наук, профессор.

ISBN Изд-во ДВГТУ, 2002

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В системах управления электроприводами значительное место занимают управляющие логические устройства. Проектирование таких устройств в сложных случаях связано с большой трудоемкостью, определяемой поиском оптимальных схемных решений и просмотром ряда нестандартных ситуаций. Поэтому решение таких задач на интуитивном уровне нецелесообразно, так как не гарантирует получения оптимальных решений за приемлемое время. Методы, базирующиеся на математическом аппарате булевой алгебры, позволяют получать формальные описания функционирования управляющих устройств в виде логических зависимостей выходных сигналов устройства от входных, отражающих заданные условия функционирования.

При этом могут быть учтены и возможные нестандартные поведения командных элементов или последовательности их срабатывания, особенно если проектируются устройства с памятью (или обратными связями).

Полученные методами логического синтеза алгебраические выражения могут быть реализованы в соответствии с имеющейся элементной базой: из релейно-контактных элементов, бесконтактных логических элементов, например интегральных микросхем, или путем программного решения на компьютере, если в дальнейшем предполагается управление объектом с помощью компьютера.

На базе логических методов аналогично решаются и задачи анализа работы управляющих устройств, в том числе и задачи технической диагностики, когда с использованием логических зависимостей строятся проверяющие и диагностические тесты, облегчающие процесс наладки таких устройств и поиска неисправностей. Методы построения тестов позволяют получить оптимальные по длине тесты, что в свою очередь сокращает время диагностирования объекта и упрощает структуру средства диагностирования или объем памяти, занятой соответствующей программой в компьютере.

1. ПРОЕКТИРОВАНИЕ УПРАВЛЯЮЩИХ ЛОГИЧЕСКИХ УСТРОЙСТВ (УЛУ)

1.1. Математические основы проектирования УЛУ

1.1.1. Функции, выполняемые логическими элементами

Функции, выполняемые логическими элементами, условимся обозначать конечными большими буквами латинского алфавита:

$$X, Y, Z, \ldots, X_1, X_2, \ldots, X_i, Y_1, Y_2, \ldots, Y_i, Z_1, Z_2, \ldots, Z_i;$$

переменные - малыми начальными буквами:

$$a, b, c, \ldots, a_1, a_2, \ldots, a_i, b_1, b_2, \ldots, b_i, c_1, c_2, \ldots, c_i.$$

С учетом того, что логические функции и переменные могут иметь одно из двух значений: 0 или 1, число возможных комбинаций значений n переменных может быть 2^n, а число функций от n переменных - 2^{2^n}.

Таким образом, от одной переменной может быть четыре функции, от двух - 16, от трех - 256 и т.д.

Функции одной переменной a: x_1 - повторение, x_2 - инверсия, x_3 - единичная, x_4 - нулевая

a	x_1	x_2	x_3	x_4
0	0	1	1	0
1	1	0	1	0

$x_1 = a$, $x_2 = \bar{a}$

$x_3 = 1$, $x_4 = 0$

Если контакт релейно-контактного элемента рассматривать как источник логического сигнала, то замыкающий контакт следует описывать символом переменной без отрицания, а размыкающий - с отрицанием. Последовательное соединение контактов эквивалентно логической функции И, параллельное - ИЛИ. В зависимости от сочетания и соединения контактов релейно-контактная схема может реализовывать любую логическую функцию. Наиболее распространены названия функций двух переменных. В таблице 1.1 приведены основные сведения об этих функциях и их реализации в бесконтактном и релейно-контактном вариантах.

В структурной реализации логических схем употребляются такие функции, как “память”, “задержка” и т.п.

Условимся логический сигнал с задержкой обозначать с индексом t и стрелкой, указывающей вид задержки: на появление сигнала - $\uparrow$, на исчезновение - $\downarrow$, на то и другое - $\uparrow\downarrow$. Например, $a_{t\uparrow}$ – задержка сигнала a при его изменении с 0 на 1.

1.1.2. Основные законы алгебры логики

Аксиомы: $\overline{0} = 1,\ \overline{1} = 0;\ 0 + 0 = 0,\ 0 + 1 = 1 + 0 = 1,\ 1 + 1 = 1;$
$0 * 0 = 0,\ 0 * 1 = 1 * 0 = 0,\ 1 * 1 = 1.$

Закон нулевого множества:
$a*0 = 0,\ a + 0 = a,\ 0 * a * b * c * d * \ldots * w = 0.$

Закон универсального множества:
$a * 1 = a,\ a + 1 = 1,\ 1 + a + b + c + d + \ldots + w = 1.$

Закон повторения: $a * a * a * \ldots * a = a,\ a + a + a + \ldots + a = a.$

Закон двойной инверсии: $\overline{\overline{a}} = a.$

Закон дополнительности: $a * \overline{a} = 0,\ a + \overline{a} = 1.$

Законы переместительный, сочетательный, распределительный (справедливы для функций И, ИЛИ, НЕ):
$a * b = b * a,\ a * (b * c) = (a * b) * c,\ a + b = b + a,$
$a + (b + c) = (a + b) + c,\ a * (b + c) = a * b + a * c.$

Закон поглощения: $a * (a + b) = a,\ a * (a + b) * (a + c) \ldots = a,$
$a + a * b + a * c + \ldots = a,\ a + \overline{a} * b = a + b.$

Закон склеивания:
$a * b + a * \overline{b} = a,\ (a + b) * (a + \overline{b}) = a,\ a * b + \overline{a} * c + b * c = a * b + \overline{a} * c.$

Закон Де Моргана: $\overline{a + b} = \overline{a} * \overline{b},\ \overline{a + b} = \overline{a} + \overline{b},$
$\overline{a + b + c + \ldots + w} = \overline{a} * \overline{b} * \overline{c} * \ldots * \overline{w},\quad \overline{a * b * c * \ldots * w} = \overline{a} + \overline{b} + \overline{c} + \ldots + \overline{w}.$

Теорема разложения:
$f(a, \overline{a}, b, c, \ldots, w) = a * f(1, 0, b, c, \ldots, w) + \overline{a} * f(0, 1, b, c, \ldots, w) =$
$= (a + f(0, 1, b, c, \ldots, w)) * (\overline{a} + f(1, 0, b, c, \ldots, w)).$

Следствие: $a * f(a, \overline{a}, b, c, \ldots, w) = a * f(1, 0, b, c, \ldots, w),$
$a + f(a, \overline{a}, b, c, \ldots, w) = a + f(0, 1, b, c, \ldots, w).$

1.1.3. Функциональная полнота

Система элементарных логических функций называется функционально полной, если произвольную логическую функцию можно представить в виде суперпозиции этих функций.

В функционально полном наборе функций присутствуют функции, в совокупности не обладающие нижеприведенными свойствами.

1) Свойство сохранения константы 0

$f(a, b, c, \ldots, w) = 0$, если $a = 0, b = 0, c = 0, \ldots, w = 0$, т.е. $f(0, 0, 0, \ldots, 0) = 0$.

2) Свойство сохранения константы 1

$f(a, b, c, \ldots, w) = 1$, если $a = 1, b = 1, c = 1, \ldots, w = 1$, т.е. $f(1, 1, 1, \ldots, 1) = 1$.

3) Свойство самодвойственности $\bar{f}(a, b, c, \ldots, w) = f(\bar{a}, \bar{b}, \bar{c}, \ldots, \bar{w})$.

4) Свойство линейности

$$f(a, b, c, \ldots, w) = k_0 \oplus k_1 * a \oplus k_2 * b \oplus k_3 * c \oplus \ldots \oplus k_n * w,$$

где $k_0, k_1, k_2, k_3, \ldots, k_n$ - элементы множества $\{0,1\}$.

5) Свойство монотонности

$f(a, b, c, \ldots, w)$ - монотонна, если $f_1(a_1, b_1, c_1, \ldots, w_1) \geq f_2(a_2, b_2, c_2, \ldots, w_2)$ при $a_1 \geq a_2, b_1 \geq b_2, c_1 \geq c_2, \ldots, w_1 \geq w_2$.

Самыми распространенными функционально полными системами являются однофункциональные системы И - НЕ и ИЛИ - НЕ.

Для построения УЛУ в базе И - НЕ или ИЛИ - НЕ следует исходную функцию, представленную через функции И, ИЛИ, НЕ, преобразовать к нужной базе, воспользовавшись двумя законами алгебры логики: двойной инверсии и Де Моргана.

Н а п р и м е р:

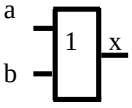
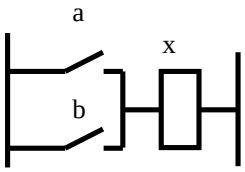
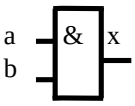
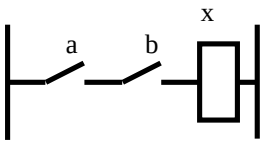
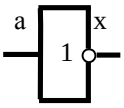
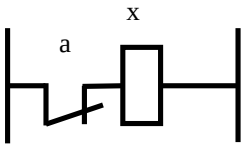
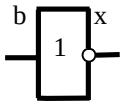
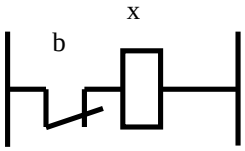
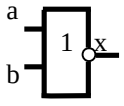
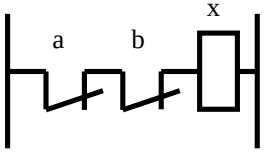
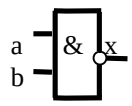
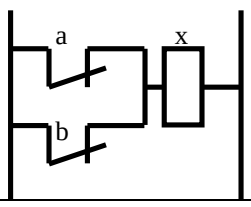
$$x = (a + b) * \bar{c} + d = \overline{\overline{(a + b) * \bar{c} + d}} = \overline{\overline{a} * \overline{\bar{b}} * \bar{c} + d} = \overline{\overline{a} * \overline{\bar{b}} * \overline{\bar{c}} * \overline{d}}$$

или

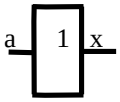
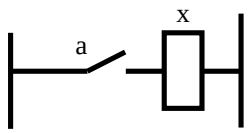
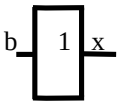
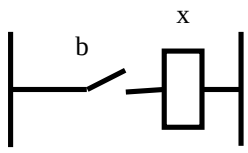
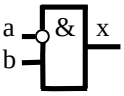
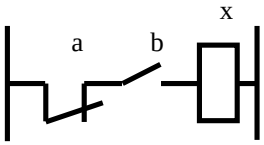
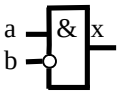
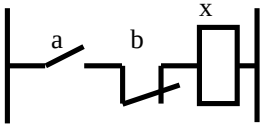
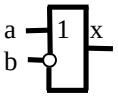
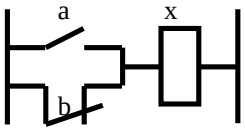
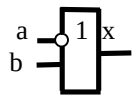
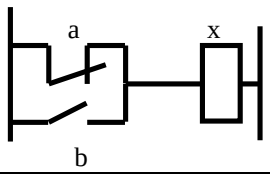
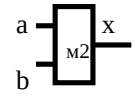
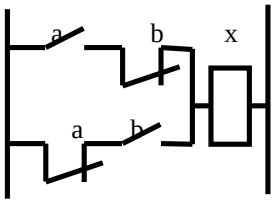
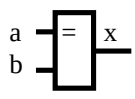
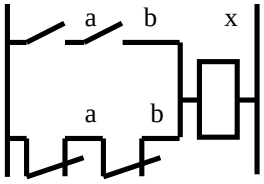
$$x = (a + b) * \bar{c} + d == \overline{\overline{\overline{(a + b) * \bar{c}} + d} = \overline{\overline{a + b} + \overline{\bar{c}} + d} = \overline{\overline{a + b} + c + d}}.$$

Таблица 1.1

Логические функции двух переменных

№№ п/п	Наименование логической функции	Логическое выражение	Обоз. логического элемента	Релейно-контактный эквивалент	Значение функции x=f(a,b)				Символич. обозн.
					a: 0	1	0	1	
					b: 0	0	1	1	
1	2	3	4	5	6	7	8	9	10
1	Нулевая	x=0	-	Обрыв	0	0	0	0	0
2	ИЛИ (дизъюнкция, логическая сумма)	x=a+b (x=avb)			0	1	1	1	+
3	И (конъюнкция, логическое произведение)	x=a*b (x=a^b, x=a&b)			0	0	0	1	*
4	НЕ a (отрицание a, инверсия a)	$x=\bar{a}$			1	0	1	0	-
5	НЕ b (отрицание b, инверсия b)	$x=\bar{b}$			1	1	0	0	-
6	ИЛИ-НЕ (стрелка Пирса, отрицание суммы)	$x=\overline{a+b}=\bar{a}*\bar{b}$ ($x=a\downarrow b$ $x=\overline{avb}$)			1	0	0	0	↓
7	И-НЕ (штрих Шеффера, отрицание произведения)	$x=\overline{a*b}=\bar{a}+\bar{b}$ ($x=a/b$, $x=\overline{a\wedge b}$, $x=\overline{a\ \&\ b}$			1	1	1	0	/

Продолжение таблицы 1.1

1	2	3	4	5	6	7	8	9	10
8	Повторение a	x=a			0	1	0	1	
9	Повторение b	x=b			0	0	1	1	
10	Запрет a	$x=\bar{a}*b$			0	0	1	0	b←a
11	Запрет b	$x=a*\bar{b}$			0	1	0	0	a←b
12	Импликация a	$x=a+\bar{b}$			1	1	0	1	b→a
13	Импликация b	$x=\bar{a}+b$			1	0	1	1	a→b
14	Неравнознач-ность (сумма по mod 2)	$x=a*\bar{b}+$ $+\bar{a}*b$ (x=a⊕b)			0	1	1	0	⊕
15	Равнознач-ность	$x=a*b+$ $+\bar{a}*\bar{b}$ (x=a≡b)			1	0	0	1	≡
16	Единичная	x=1	-		1	1	1	1	1

В первом случае двойная инверсия взята от логических сумм, а затем применен закон де Моргана: отрицание суммы есть произведение отрицаний переменных.

Во втором случае двойная инверсия взята от логических произведений, а затем применен закон де Моргана: отрицание произведения есть сумма отрицаний переменных.

Таким образом, первое выражение приведено к базе И-НЕ, второе - к ИЛИ - НЕ.

1.2. Применение карт Карно для изображения и преобразования логических функций

1.2.1. Понятие о карте Карно

Рассмотрим логическую функцию двух входных переменных f (a, b).

Две входные логические переменные a и b могут образовывать всего четыре возможных набора своих значений (таблица 1.2).

Таблица 1.2

Наборы

a	b
0	0
0	1
1	0
1	1

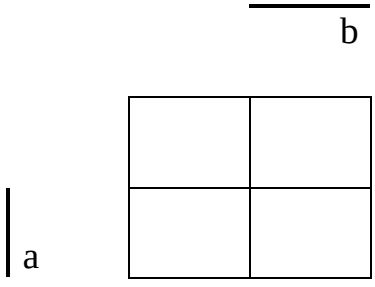


Рис. 1.1. Карта Карно

Каждому набору можно поставить в соответствие клетку карты Карно (рисунок 1.1). В эту клетку записывается значение функции (0 или 1) для данного набора. Входные переменные располагаются по внешним сторонам карты напротив ее строк и столбцов. При этом значение каждой из входных переменных относится ко всей строке (или столбцу) и равно 1, если напротив строки (или столбца) стоит под скобкой обозначение этой переменной; для остальных строк (столбцов) значение этой переменной равно 0. Эти значения входных переменных не пишутся на карте, а подразумеваются.

Следует отметить, что каждая из входных переменных делит карту Карно на две равные части, в одной из которых значение этой переменной равно 1, а в другой 0.

Каждой клетке карты соответствует один определенный набор, а каждая сторона клетки представляет собой границу между значениями переменных.

1.2.2. Составление карт Карно по таблицам истинности

В таблицах истинности, с помощью которых задаются логические функции, число строк равно числу всех возможных наборов (напомним, что число строк определяется как 2^n, где n - число входных переменных). Поэтому таблицы истинности для функций больше чем двух переменных становятся громоздкими. Изображение логических функций посредством карт Карно является более компактным, так как каждому набору в ней соответствует клетка, а не строка.

Карта Карно может составляться непосредственно по таблице истинности. Для этого строится карта с числом клеток, равным числу строк таблицы. По внешним сторонам карты определенным образом располагаются входные переменные. Для каждого набора (строки) таблицы отыскивается соответствующий набор (клетка) карты. В эту клетку проставляется значение функции для данного набора.

Например, для функции трех переменных f = f (a, b, c), заданной таблицей истинности (таблица 1.3), карта Карно имеет вид, представленный на рис. 1.2.

Число клеток карты Карно определяется величиной 2^n, где n равно числу входных переменных. Отсюда следует, что прибавление каждой новой переменной удваивает число клеток, т.е. увеличивает карту вдвое. При этом, как это будет видно в дальнейшем, новые переменные должны располагаться так, чтобы иметь общую площадь со всеми прежними переменными.

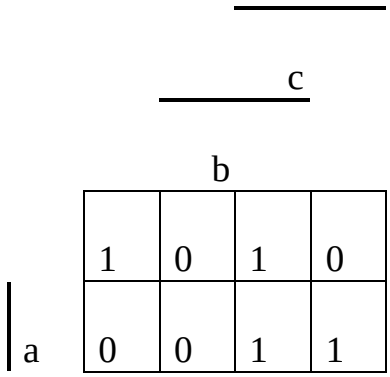


Рис. 1.2. Карта Карно для функции f = f (a, b, c).

Таблица 1.3

Таблица истинности

a	b	c	f
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

1.2.3. Составление карт Карно по алгебраическим выражениям

Карта Карно для логической функции, заданной алгебраическим выражением, может быть составлена в следующем порядке:

по числу переменных, входящих в выражение заданной функции, строится карта Карно и располагаются переменные;

заданное алгебраическое выражение приводится к совершенной дизъюнктивной нормальной форме (СДНФ);

в карте Карно для каждого конституента единицы СДНФ находится соответствующая клетка (с таким же набором переменных), в которую записывается 1, в остальные клетки карты записываются 0.

В качестве примера рассмотрим логическую функцию, заданную алгебраическим выражением

$$f = \bar{a}c + bc + a\bar{b}.$$

Эта функция является функцией трех переменных a, b, и c, и ее карта должна иметь $2^n = 2^3 = 8$ клеток;

приведение к СДНФ:

$$\bar{a}c(b+\bar{b})+bc(a+\bar{a})+\bar{a}b(c+\bar{c})=\bar{a}cb+\bar{a}c\bar{b}+bca+bc\bar{a}+\bar{a}bc+\bar{a}b\bar{c}=$$

$$=\bar{a}bc+\bar{a}\bar{b}c+abc+\bar{a}b\bar{c},$$

функция имеет значения, равные 1, в клетках карты, соответствующих конституентам СДНФ, т.е. карта Карно для заданной функции имеет вид, представленный на рисунке 1.3.

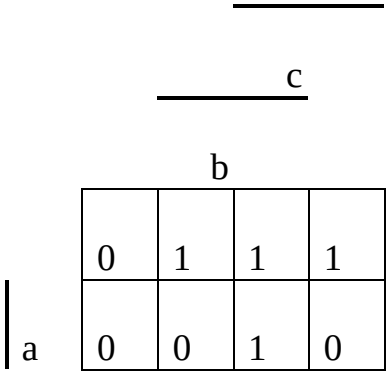


Рис. 1.3. Карта Карно для функции $\mathbf{f}=\bar{a}c+bc+\bar{a}b$.

Можно воспользоваться приведением исходного выражения к совершенной конъюнктивной нормальной форме (СКНФ). В этом случае в клетки карты, соответствующие конституентам нуля, записывается 0, в остальные 1.

Основное применение карты Карно находят при решении обратной задачи, т.е. при определении алгебраических выражений функций по картам, полученным в результате логического синтеза релейных устройств.

1.2.4. Свойства карты Карно

1. Наборы значений переменных для соседних клеток карты Карно отличаются значением лишь одной переменной. При переходе из одной клетки в соседнюю всегда изменяется значение лишь одной переменной от своего прямого значения к его инверсии и обратно.

Рассмотрим карту для четырех переменных на рисунке 1.4. Значения функции 0 или 1 в клетках карты пока во внимание не принимаются.

Для клеток второго столбца и клеток третьей строки выписаны соответствующие им наборы:

2-й столбец	3-я строка
$\bar{a}\bar{b}\bar{c}d$	$ab\bar{c}\bar{d}$
$\bar{a}b\bar{c}d$	$ab\bar{c}d$
$ab\bar{c}d$	$abcd$
$a\bar{b}\bar{c}d$	$abc\bar{d}$

Видно, что выражения для соседних клеток отличаются друг от друга значением только одной переменной.

2. Соседними между собой являются также крайние левые клетки карты с крайними правыми и крайние верхние клетки карты с крайними нижними (как если бы карты были бы свернуты в цилиндры по вертикали и по горизонтали).

В этом легко убедиться, сравнивая первые и последние выражения, записанные выше для второго столбца и третьей строки.

		$\overline{c}$			
		d			
a	b	0	0	0	0
		0	0	1	0
		0	1	1	0
		0	0	1	0

Рис. 1.4. Карта Карно для функции четырех переменных.

3. Рассмотрим карту для пяти переменных (рисунок 1.5), не принимая пока во внимание значения функции в клетках карты.

Выпишем наборы значений переменных для клеток второй строки, следуя слева направо:

1. $\bar{a}b\bar{c}\bar{d}\bar{e}$; 5. $\bar{a}bcd\bar{e}$;
2. $\bar{a}b\bar{c}\bar{d}e$; 6 $\bar{a}bcde$;
3. $\bar{a}b\bar{c}de$; 7. $\bar{a}bc\bar{d}e$;
4. $\bar{a}b\bar{c}d\bar{e}$; 8. $\bar{a}bc\bar{d}\bar{e}$.

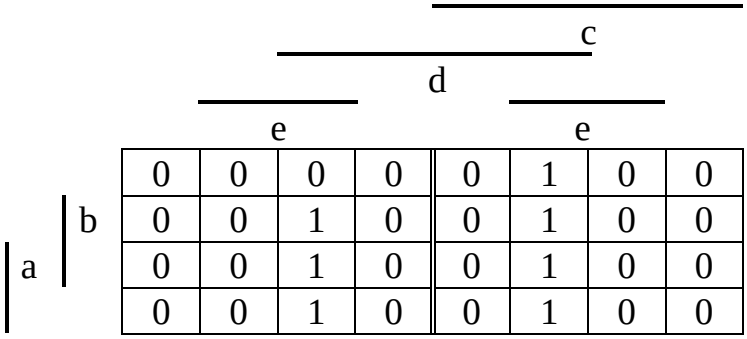


Рис. 1.5. Карта Карно для функции пяти переменных

Укажем на некоторые клетки в такой карте, которые называются соседними. Клетка 1 является соседней с клеткой 2 (отличается значением переменной e), с клеткой 8 (отличается значением переменной c) и с клеткой 4 (отличается значением переменной d). Клетка 5 является соседней с клеткой 4 (переменная c), 6 (переменная e) и 8 (переменная d).

Аналогично устанавливается соседство других клеток.

Таким образом, все клетки, отличающиеся значением только одной переменной, являются соседними, несмотря на то, что иногда они расположены не рядом.

Это свойство карты является очень важным для определения минимальных алгебраических выражений функции.

1.2.5. Определение по карте Карно алгебраических выражений логических функций

Для некоторой логической функции, представленной посредством карты Карно, можно записать несколько алгебраических выражений различной сложности в дизъюнктивной и конъюнктивной форме.

Приведем правила, которыми следует при этом руководствоваться:

1. Все единицы (при записи функции в дизъюнктивной форме) и все нули (при записи функции в конъюнктивной форме) должны быть заключены в прямоугольные контуры. Единичные контуры могут объединять несколько единиц, но не должны содержать внутри себя нулей. Нулевые контуры могут объединять несколько нулей, но не должны содержать внутри себя единиц. Одноименные контуры могут накладываться друг на друга, т.е. одна и та же единица (или нуль) может входить в несколько единичных (нулевых) контуров.

2. Площадь любого контура должна быть симметричной относительно границ переменных, пересекаемых данным контуром. Другими словами, число клеток в контуре должно быть равно 2^n, где n=0, 1, 2…, т.е. число клеток выражается числами 1, 2, 4, 16, 32 …

3. Во избежание получения лишних контуров, все клетки которых вошли уже в другие контуры, построение следует начинать с тех единиц или нулей, которые могут войти в единственный контур.

4. В контур можно объединять только соседние клетки, содержащие единицы или нули. Соблюдение этого правила особенно необходимо проверять при числе переменных, большем четырех, когда соседние клетки могут быть расположены не рядом и поэтому контуры могут претерпевать видимый разрыв.

5. Каждой единичной клетке соответствует конъюнкция входных переменных, определяющих данную клетку. Каждой нулевой клетке соответствует дизъюнкция инверсий входных переменных, определяющих данную клетку.

6. В контуре, объединяющем две клетки, одна из переменных меняет свое значение, поэтому выражение для контура из двух клеток не зависит от этой переменной, а представляется всеми остальными переменными. Это правило относится и к контурам, охватывающим число клеток более двух, и имеет такую формулировку: выражения, соответствующие контурам, не содержат тех переменных, чьи границы пересекаются площадью, ограниченной данным контуром.

7. Выражение логической функции может быть записано по соответствующей ей карте Карно в дизъюнктивнойили конъюнктивнойформах.

Дизъюнктивная форма составляется в виде дизъюнкции конъюнкций, соответствующих единичным контурам, выделенным на карте для определения функции; конъюнктивная - в виде конъюнкции дизъюнкций, соответствующих нулевым контурам.

8. Для контуров, охватывающих различное число клеток, получаются выражения различной сложности. Поэтому для данной логической функции можно записать по ее карте Карно несколько отличающихся по сложности алгебраических выражений. Наиболее сложное выражение соответствует случаю, когда каждой клетке соответствует свой контур. Это выражение представляет собой СДНФ или СКНФ данной функции. С увеличением размеров контуров алгебраическое выражение упрощается. На этом свойстве основывается метод минимизации логических функций с помощью карт Карно.

П р и м е р. Определить алгебраическое выражение логической функции по соответствующей ей карте Карно, приведенной на рисунке 1.6.

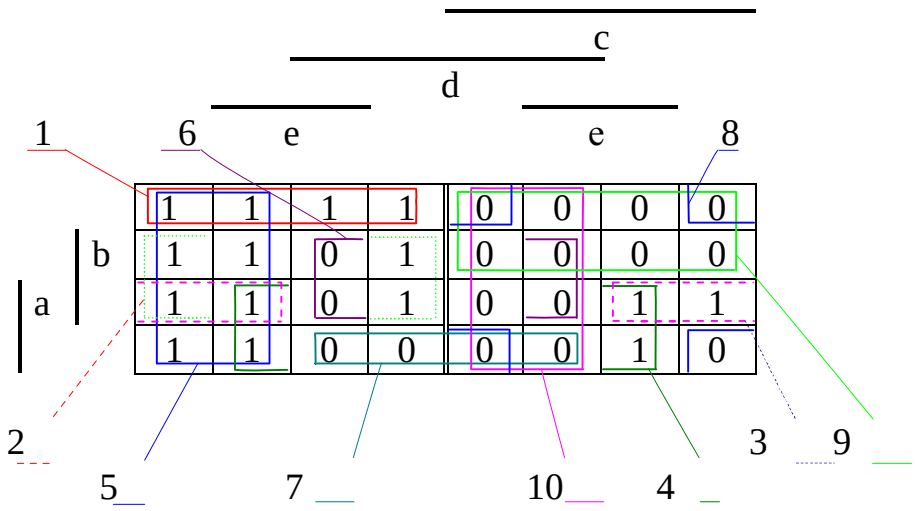


Рис. 1.6. Карта Карно к примеру

На данной карте каждая из клеток имеет непосредственные границы четырех переменных a, b, d, e, а граница пятой переменной c показана двойной линией.

Сначала будем определять выражения функции по ее единичным значениям. Руководствуясь правилом 3, находим на карте те единицы, которые можно включить только в один контур. Таких единиц четыре, они записываются согласно правилу 5 в виде конъюнкций

$$\bar{a}\bar{b}\bar{c}de,\ ab\bar{c}d\bar{e},\ abcd\bar{e},\ a\bar{b}cd\bar{e}.$$

Учитывая правила 1, 2, 4, 8, заключаем найденные единицы в обязательные контуры возможно большего размера (жирные сплошные линии), которым согласно правилам 5, 6 соответствуют следующие выражения:

$$\bar{a}\bar{b}\bar{c},\ b\bar{c}\bar{e},\ ab\bar{d},\ a\bar{d}e.$$

После этого остаются лишь две единицы $\bar{a}\bar{b}\bar{c}de, a\bar{b}\bar{c}d\bar{e}$, которые можно заключить в один общий контур, обозначенный на карте тонкой сплошной линией. Этому контуру соответствует выражение $\bar{c}\bar{d}$. Применяя правило 7, можно записать для заданной логической функции алгебраическое выражение в дизъюнктивной форме:

$$f = \underset{1}{\bar{a}\bar{b}\bar{c}} + \underset{2}{b\bar{c}\bar{e}} + \underset{3}{ab\bar{d}} + \underset{4}{a\bar{d}e} + \underset{5}{\bar{c}\bar{d}}.$$

Переходя к определению выражения по нулевым значениям, получаем три обязательных контура (жирные пунктирные линии):

$$(\bar{b}+\bar{d}+\bar{e}),\ (\bar{a}+b+\bar{d}),\ (b+\bar{c}+e).$$

Остальные нули могут быть заключены в два наибольших контура $(a+\bar{c})(\bar{c}+d)$, изображенных тонкими пунктирными линиями. Алгебраическое выражение функции в конъюнктивной форме, записанное

по нулевым контурам, имеет вид:

$$f = (\bar{b} + \bar{d} + \bar{e})\,(\bar{a} + b + \bar{d})\,(b + \bar{c} + e)\,(a + \bar{c})\,(\bar{c} + \bar{d}).$$
$$\qquad 6 \qquad\quad 7 \qquad\quad 8 \qquad 9 \quad 10$$

Следует заметить, что на карте можно образовать еще другие единичные или нулевые контуры (один из них показан штрих пунктирной линией), однако они являются лишними, так как заключенные в них единицы или нули уже вошли в обязательные контуры, без которых обойтись нельзя.

1.3. Синтез УЛУ по таблицам переходов

1.3.1. Построение таблиц переходов

Таблица переходов представляет собой матрицу, устанавливающую связь между состояниями объекта и значениями входных и выходных переменных. Число строк матрицы соответствует числу состояний, число столбцов равно $2^n + k$, где n - число входных, а k - число выходных переменных. В клетке матрицы проставляется символ последующего состояния, в которое переходит объект из данного исходного (соответствующая строка) при данной комбинации значений входных сигналов (соответствующий столбец). Число исходных состояний, определяющее число строк таблицы, следует выбирать на основе анализа операций управления от нерабочего состояния до операции, завершающей процесс. Комбинации значений выходных переменных для любого исходного состояния должна отличаться от комбинации значений выходных переменных соседних состояний по меньшей мере значением одной выходной переменной. В случае, если при переходе от одного исходного состояния к другому ни одно из значений выходных переменных не изменится, необходимо вводить промежуточную переменную, отличающую эти состояния. Если при составлении таблицы получится одинаковое заполнение двух или нескольких строк, это означает, что в таблицу введены лишние состояния. В этом случае производится сжатие таблицы переходов по определенным правилам [1].

Заполнение клеток таблицы переходов производится путем анализа входных наборов и их влияния на переход устройства в последующее состояние. Сначала целесообразно выявить практически неосуществимые наборы и в клетках соответствующих столбцов проставить знак условного состояния (~). К таким наборам можно, например, отнести появление сигнала о наличии массы при отсутствии изделия, пребывания механизма одновременно в двух крайних противоположных положениях и т.д. В

остальных клетках путем логических рассуждений определяется, сохранит ли устройство свое исходное состояние или перейдет в другое и какое именно, включая условное (~). В качестве примера см. таблицу переходов на рис. 1.7.

Исход. сост.	Последующие Состояния																Выходы	
																	x	y
1	1	1	1	3	~	~	~	~	1	2	1	1	3	1	2	1	0	0
2	1	1	1	1	~	~	~	~	2	2	2	2	2	2	2	2	1	0
3	3	3	3	3	~	~	~	~	1	1	1	1	3	3	3	3	0	1

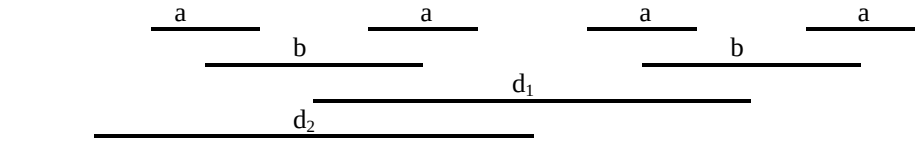


Рис. 1.7. Пример таблицы переходов.

1.3.2. Выбор дополнительных переменных

Схемы, реализующие функции, описываемые таблицами переходов - это, как правило, схемы с обратными связями (или с памятью). Поэтому для их реализации требуются дополнительные (к входным) переменные, осуществляющие запоминание состояния устройства. Дополнительные переменные комбинациями своих значений должны по-разному кодировать состояния устройства. В качестве дополнительных переменных в первую очередь используются выходные переменные. Если число выходных переменных оказывается недостаточным для различения состояния устройства, то необходимо вводить новые дополнительные переменные. Общее число дополнительных переменных P определяется по формуле

$$2^P \geq S \geq 2^{P-1},$$

где S - число исходных состояний таблицы переходов. Расчет P по приведенной формуле производится лишь при отсутствии каких-либо ограничений на кодирование символов состояний теми или иными комбинациями значений дополнительных переменных. В ряде случаев, когда переходы из одного состояния в другое будут происходить при изменении значений более чем одной дополнительной переменной, могут возникнуть так называемые " состязания" логических элементов (из-за различных времен срабатывания элементов). Эти "состязания" могут быть "критическими" - опасными для работы устройства.

Для устранения " критических состязаний" комбинации значений дополнительных переменных, соответствующие номерам устойчивых состояний, следует выбирать так, чтобы все необходимые переходы

осуществлялись при изменении состояния только одного дополнительного элемента. Выполнение этого условия может потребовать увеличить число Р.

Другим способом устранения " критических состязаний" является включение дополнительных элементов задержки в цепи обратных связей. Время задержки должно быть таким, чтобы обеспечивалось достижение устойчивого состояния комбинационной части устройства, прежде чем изменится комбинация дополнительных элементов.

1.3.3. Построение карт Карно

После решения вопроса с кодированием состояний по таблице переходов строятся карты Карно для выходных и дополнительных элементов. Карта Карно представляет собой матрицу, состоящую из 2^{n+p} клеток, причем каждой клетке соответствует одна, отличная от всех других, комбинация значений входных и дополнительных переменных (координаты клетки).В первоначальной карте Карно вместо символа состояния таблицы переходов в клетке с определенными координатами проставляется комбинация значений выходных и дополнительных переменных.

Затем первоначальная карта разбивается на столько карт, сколько значений проставлено в ее клетке. В итоге для каждой выходной и дополнительной переменной строится своя карта Карно, по которой и получается алгебраическое выражение для соответствующей выходной или промежуточной функции [1].

1.3.4. Применение элементов памяти [2]

В качестве кодирующих элементов могут быть выбраны и стандартные элементы памяти (SR- триггеры, D-триггеры, счетчики, регистры и т.п.). В этом случае в карте Карно вместо значения дополнительного сигнала проставляются значения функций возбуждения элементов памяти. При этом, если значение сигнала соответствует устойчивому состоянию устройства, то функции возбуждения тоже проставляются для устойчивого состояния, если состояние неустойчивое, то учитывается процесс перехода дополнительного элемента $0 \rightarrow 1$ или $1 \rightarrow 0$ и функции возбуждения элемента памяти проставляются с учетом этого перехода [2].

Таким образом, таблица состояний элементов памяти (ЭП) или карта Карно составляется по таблице переходов, где каждой строке таблицы поставлено в соответствие определенное состояние элементов памяти.

Единица в клетке матрицы свидетельствует о наличии сигнала на выходе ЭП, нуль - о его отсутствии и тильда (~) - о безразличном состоянии.

Способ построения таблицы зависит от свойств и возможностей применяемого элемента.

Если в качестве ЭП используется электромагнитное реле, то на его вход должен подаваться сигнал в следующих случаях:

1. Во всех устойчивых состояниях строк таблицы переходов, у которых значение элемента памяти равно 1;

2. Во всех неустойчивых состояниях таблицы переходов, соответствующих номерам устойчивых состояний, указанных в предыдущем пункте.

При использовании в качестве ЭП триггера RS, который переключается в состояние 1 при поступлении сигнала на вход S и отсутствии сигнала на входе R, а в состоянии 0 при наличии сигнала на входе R и отсутствии сигнала на входе S, в таблице ЭП необходимо указывать, на какой из входов подается сигнал при включении или выключении триггера. Таблица переходов RS -триггера приведена на рисунке 1.8.

	S \ R			
0	0	1	~	0
1	1	1	~	0

Рис. 1.8. Таблица переходов RS - триггера.

	S	R
0→0	0	~
0→1	1	0
1→0	0	1
1→1	~	0

Из таблицы переходов (рисунок 1.8) видно, что переключение триггера x из состояния 0 в состояние 1 производится при переходе 1 →2, переключение x 1 →0 при переходе 2 →1, y (0 →1) при 1 →3, y (1 →0) при 3→1.

Поэтому в неустойчивых состояниях 2 должен подаваться сигнал на вход S и отсутствовать на входе R. В клетках таблицы, соответствующих состоянию 2, проставляется комбинация S_1R_1 - 10. В неустойчивом состоянии 1 (в строке 2) проставляется S_1R_1 - 01. Там, где x=1 устойчиво (в строке 2 (2)) проставляется S_1R_1 - ~0, x = 0 устойчиво (в строках 1 и 3 (1 и 3)) - 0~ (рисунок 1.9).

Тогда вместо одной карты для каждой дополнительной переменной будет по две (для каждой функции возбуждения: S и R).

По картам Карно в этом случае будут получены алгебраические выражения для выходных функций и для функций возбуждения элементов

памяти. По полученным логическим зависимостям строится структура схемы управления.

d_2															
		d_1													
		b										b			
a				a				a					a		
0~ (0~)	0~ (0~)	0~ (0~)	0~ (10)	~	~	~	~	0~ (0~)	10 (0~)	0~ (0~)	0~ (0~)	0~ (10)	0~ (0~)	10 (0~)	0~ (0~)
01 (0~)	01 (0~)	01 (0~)	01 (0~)	~	~	~	~	~0 (0~)	~0 (0~)	~0 (0~)	~0 (0~)	~0 (0~)	~0 (0~)	~0 (0~)	~0 (0~)
0~ (0~)	0~ (0~)	0~ (0~)	0~ (0~)	~	~	~	~	0~ (01)	0~ (01)	0~ (01)	0~ (01)	0~ (0~)	0~ (0~)	0~ (0~)	0~ (0~)

Рис.1.9. Таблица состояний функций возбуждения RS - триггеров x(y).

$x(y) => R_1, S_1 (R_2, S_2)$

1.4. Синтез УЛУ на основе циклограмм

1.4.1. Понятия и определения

Метод структурного синтеза на основе циклограмм является развитием известного в инженерной практике метода синтеза на основе таблиц включения. При проектировании устройств управления механизмами, работающими циклически, метод записи условий их работы посредством циклограмм имеет преимущество наглядности, а поэтому рекомендуется к использованию.

Циклограмма - это графическое изображение последовательности работы отдельных элементов управляющего логического устройства во времени. Работа элементов дискретного действия в логическом устройстве характеризуется появлением и исчезновением сигналов в определенной последовательности.

Наличие сигнала изображается на циклограмме отрезком горизонтальной прямой. Слева от отрезка, отражающего работу элемента, на границе циклограммы проставляется обозначение соответствующего сигнала (рисунок 1.10, а). Последовательность работы элементов определяется положением концов отрезков, изображающих их работу, относительно левой границы циклограммы (рисунок 1.10, б). На циклограмме отображается любое изменение состояний элементов и указывается собственное время их срабатывания (рисунок 1.10,в).

Воздействие одного элемента на другой изображается на циклограмме стрелкой, указывающей направление воздействия (рисунок 1.10,в).

В циклограмме время не оценивается количественно, поэтому она выполняется без масштаба. Отмечается лишь факт срабатывания элемента, факт наличия или отсутствия сигнала. При наличии специального элемента задержки его сигнал на циклограмме обозначается буквой Т, а время, по истечении которого он появляется или исчезает, - буквой t (рис 1.11).

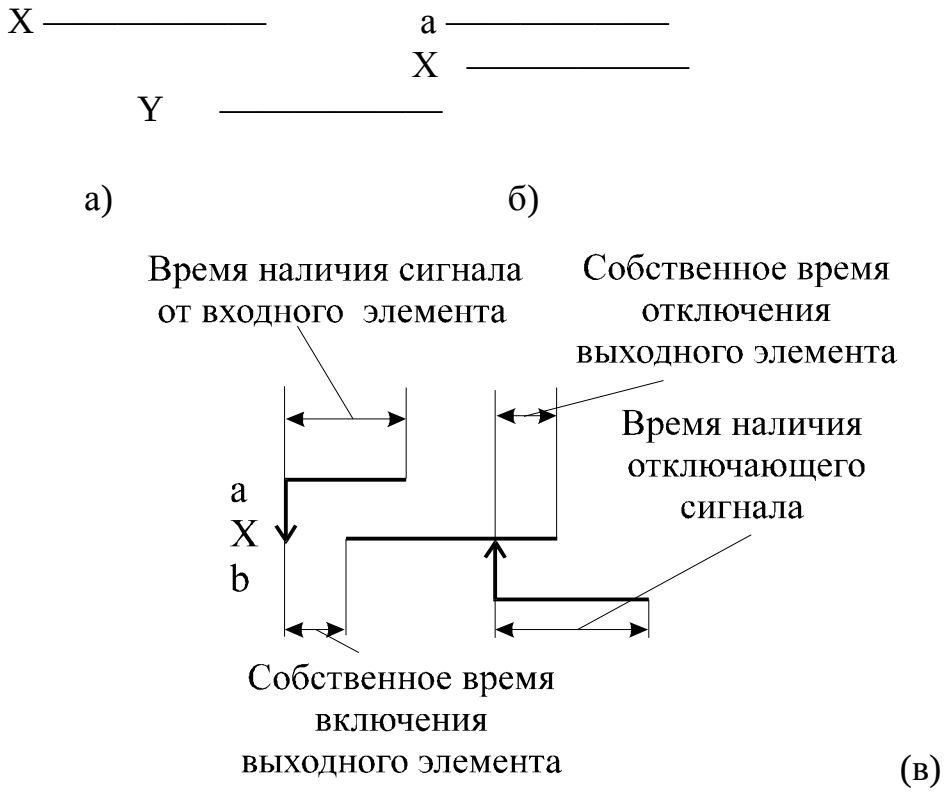


Рис. 1.10. Изображение сигналов и периодов работы элементов на циклограмме (а - обозначение сигналов, б - обозначение последовательности работы, в-собственное время работы элемента)

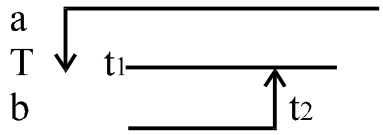


Рис. 1.11. Изображение на циклограмме сигналов элементов задержки.

На рисунке 1.12, а показаны варианты воздействия элемента а на элемент Х при его включении. В первом варианте элемент Х включается при появлении сигнала а, а во втором варианте - при исчезновении сигнала а. На рисунке 1.12,б показаны варианты воздействия элемента а на элемент Х при его отключении.

Тактами называются периоды, в течение которых в схеме не изменяется состояние ни одного из входных, промежуточных или

выходных сигналов. Каждое изменение состояния одного или одновременно нескольких элементов является началом такта.

Периодом включения элемента называется непрерывный ряд тактов, в течение которого этот элемент находится во включенном состоянии. *Периодом отключения* элемента называется непрерывный ряд тактов, в течение которого этот элемент находится в отключенном состоянии. Период включения элемента обозначается чертой. В периоде отключения элемента черта на циклограмме отсутствует.

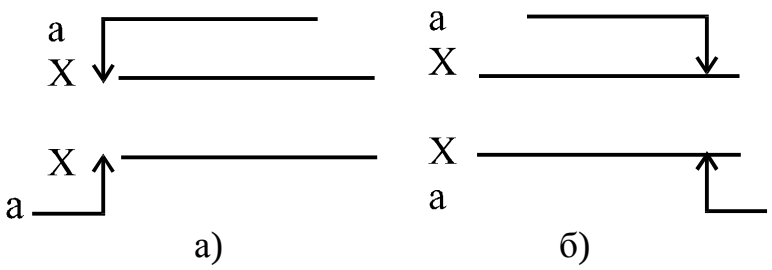


Рис.1.12. Варианты воздействия элемента а на элемент Х
(а - при включении; б - при отключении).

Включающим тактом называется такт, предшествующий периоду включения данного элемента. *Отключающий тактом* называется такт, предшествующий периоду отключения данного элемента.

Включающий период состоит из включающего такта и периода включения без отключающего такта. *Отключающий период* состоит из отключающего такта и периода отключения без включающего такта (понятие отключающего периода вводится при наличии нескольких периодов включения). Приведенные определения иллюстрируются циклограммой на рисунке 1.13.

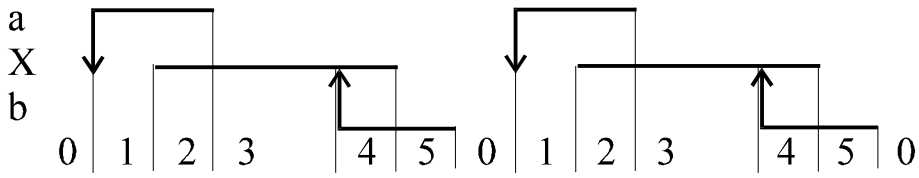


Рисунок 1.13. Изображение на циклограмме характерных тактов и периодов
(1 - включающий такт; 4 - отключающий такт; такты 2, 3, 4 - период включения; такты 1, 2, 3 - включающий период; такты 4, 5, 0 – отключающий период; такты 5, 0, 1 - период отключения).

Элементы, изменяющие свое состояние во включающем и отключающем тактах рассматриваемого периода включения, называются основными элементами.

1.4.2. Составление алгебраических выражений по циклограммам. Условия "срабатывания" и "несрабатывания" элемента

Алгебраические выражения или структурные формулы, описывающие УЛУ, составляются для каждого выходного и промежуточного элемента, затем составляется выражение общее для всего логического устройства.

Общая структурная формула анализируется с точки зрения возможности минимизации с целью упрощения схемы устройства. Для минимизации алгебраических выражений используются алгебраические равносильные преобразования.

Изменение состояния какого-либо выходного или промежуточного элемента, т.е. его включение и отключение, обусловлено изменением состояний других, воздействующих на него элементов.

Условия, вызывающие изменения состояния элемента, возникают во включающем и отключающем тактах и называются условиями *срабатывания* во включающем такте и условиями *несрабатывания* в отключающем такте.

Условие срабатывания обозначается f '. В условие срабатывания входит сигнал элемента, изменяющего свое состояние во включающем такте. Если в этом такте изменяют свое состояние несколько элементов, то в условие срабатывания достаточно ввести сигнал одного (любого из одновременно срабатывающих элементов) основного элемента, от которого на циклограмме отходит стрелка в направлении рассматриваемого выходного или промежуточного элемента. В условие срабатывания записывается сигнал этого элемента в том состоянии, которое он имеет во включающем такте.

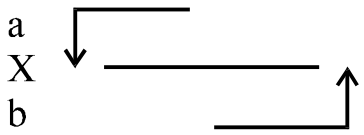


Рис.1.14. Условия срабатывания и несрабатывания элемента X под воздействием элементов a и b

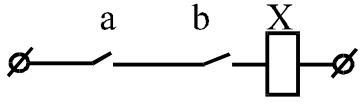


Рис.1.15. Схема, реализующая условия циклограммы на рис. 1.14.

Условия несрабатывания элементов возникают в отключающем такте. В эти условия входит сигнал элемента, изменяющего свое состояние в этом такте. При наличии нескольких элементов, изменяющих свое состояние в

отключающем такте, следует ввести в условие несрабатывания сигнал любого из основных элементов, от которого на циклограмме отходит стрелка в направлении рассматриваемого выходного или промежуточного элемента. В условие несрабатывания сигнал основного элемента записывается в том состоянии, в котором он находится в отключающем такте, со знаком инверсии. Условие несрабатывания обозначается f'' и приписывается к предыдущему условию срабатывания со знаком логического умножения. Таким образом, структурная формула для одного периода включения какого-либо исполнительного или промежуточного элемента имеет вид:

$$X = f'(x)\,\bar{f}''(x).$$

На рисунке 1.14 показаны в качестве примера условия срабатывания элемента X под воздействием элемента a и условие его несрабатывания под воздействием элемента b. Во включающем такте появляется сигнал a. Он вызывает срабатывание элемента X и появление одноименного сигнала X. Условие срабатывания для этого случая запишется так:

$$f'(x) = a.$$

Условие несрабатывания возникает в отключающем такте и записывается следующим образом:

$$\bar{f}''(x) = \bar{\bar{b}}.$$

Один знак инверсии поставлен потому, что предусмотрен формулой условий несрабатывания, а второй - потому, что в отключающем такте сигнал b исчезает (b = 0).

Структурная формула для периода включения элемента X:

$$X = f'(x)\,\bar{f}''(x) = ab.$$

Релейно-контактная схема, соответствующая полученному выражению, изображена на рисунке 1.15. Рассмотрение этой схемы показывает, что она неработоспособна, так как входной элемент a не обеспечивает необходимых условий включения элемента X и поэтому следует произвести проверку схемы и структурной формулы и ввести дополнительные элементы.

При наличии нескольких периодов включения и отключения какого-либо выходного или промежуточного элемента условия срабатывания и несрабатывания составляются для каждого периода. Для первого периода составляется выражение в виде $f'_{1(x)}\,\bar{f}''_{1(x)}$, для второго периода $f'_{2(x)}\,\bar{f}''_{2(x)}$ и для n-го периода $f'_{n(x)}\,\bar{f}''_{n(x)}$.

Общая структурная формула для входного или промежуточного элемента X записывается так:

$$X = f'_{1(x)}\,\bar{f}''_{1(x)} + f'_{2(x)}\,\bar{f}''_{2(x)} + \ldots + f'_{n(x)}\,\bar{f}''_{n(x)},$$

т.е. в виде логической суммы логических произведений условий срабатывания и условий несрабатывания для всех периодов включения.

Введение только основных элементов в условие срабатывания и несрабатывания иногда бывает недостаточным для обеспечения работы устройства по данной циклограмме. В этих случаях приходится вводить промежуточные элементы. Для определения необходимости введения промежуточных элементов и их числа необходимо проводить три проверки реализуемости циклограммы. Все три проверки проводятся в отдельности для каждого включающего периода.

Первая проверка заключается в анализе того, существуют ли записанные ранее условия срабатывания (f') в течение всего включающего периода. Если функция f' неизменна в этом периоде, то эти условия срабатывания для данного периода включения являются достаточными. Если же функция f' вторично изменяет свое значение в течение включающего периода, то необходимо в схему ввести дополнительный элемент (P') таким образом, чтобы он изменял свое состояние до изменения состояния f' и оставался в этом состоянии далее в течение всего анализируемого включающего периода. Таким образом, если на циклограмме мысленно совместить линии действия сигналов f' и P', то это будет непрерывная прямая в течение всего включающего периода рассматриваемого элемента. Желательно в качестве промежуточного элемента при возможности использовать выходной элемент, применяя самоблокировку, или другие элементы циклограммы, линии действия сигналов которых удовлетворяют условиям выбора P'. Сигнал дополнительного элемента (P') или самоблокировки приписывается к условию несрабатывания $\bar{f}''$ в виде логического произведения $P'\bar{f}''$, а произведение в свою очередь приписывается к первоначальному условию срабатывания (f') в виде суммы, причем его состояние принимается обратным тому, которое он имеет во включающем такте. Эта проверка производится для каждого периода включения данного элемента.

Введение самоблокировки в результате первой проверки в первом периоде включения не требует дополнительных рассуждений. Применение самоблокировки в последующих периодах включения следует увязывать с предыдущими. Если в первом периоде включения была введена самоблокировка, то для использования ее в последующих периодах необходимо учитывать имеющиеся комбинации сигналов в сочетании с сигналом, используемым для самоблокировки. При наличии хотя бы одной из этих комбинаций в течение всех последующих периодов рассматриваемого элемента использование самоблокировки возможно в случае, если сигнал или сигналы, соответствующие условиям несрабатывания для данного включающего периода, не встречались в

предыдущих включающих периодах или встречались, но также в качестве условий несрабатывания. Тогда в виде произведения они приписываются к используемой комбинации сигналов с самоблокировкой во всех структурных формулах включающих периодов рассматриваемого элемента. Ниже будет рассмотрен пример, иллюстрирующий наряду с другими и это положение.

Для n-го периода включения при введении дополнительного элемента P ' структурная формула элемента X запишется так:

$$X_n = f'_{n(x)} + p'\bar{f}''_{n(x)}$$

или при самоблокировке

$$X_n = f'_{n(x)} + x\bar{f}''_{n(x)};$$

возможна и такая запись:

$$X_n = (f'_{n(x)} + p')\,\bar{f}''_{n(x)}$$

или

$$X_n = (f'_{n(x)} + x)\,\bar{f}''_{n(x)}.$$

Функциональная равносильность этих записей при определенной

1 0 1

0

последовательности появления значений входных сигналов (a-a-b-b) очевидна из приведенных на рисунке 1.16 релейно-контактных схем.

При срабатывании элемента a получает питание элемент X, самоблокируется и отключается лишь при срабатывании элемента b.

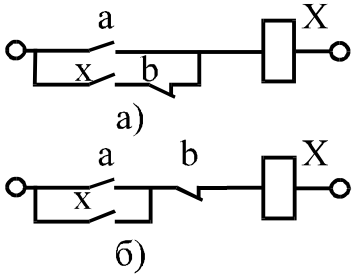


Рис. 1.16. Функционально равносильные варианты релейно-контактных схем с самоблокировкой.

а) $X = a + x\bar{b}$; б) $X = (a + x)\bar{b}$.

В схемах автоматики часто требуется, чтобы после срабатывания отключающего элемента схему нельзя было вновь запустить в работу до тех пор, пока он не придет в исходное состояние. Например, если нажата

кнопка *стоп*, отключающая какой-либо аппарат, то одновременное нажатие кнопки *пуск* не должно вызывать даже кратковременного его включения.

С этой точки зрения целесообразней использовать структурную формулу типа $X = (a + x)\bar{b}$.

Вторая проверка предназначена для выяснения, существуют ли записанные ранее условия несрабатывания ($\bar{f}$ ") в течении всего включающего периода. Если функция $\bar{f}$ " неизменна в этом периоде, то эти условия несрабатывания для данного периода включения являются достаточными.

Если же функция $\bar{f}$ " вторично изменяет свое состояние в течении включающего периода, то необходимо ввести в схему промежуточный элемент (P"), изменяющий свое состояние в периоде включения рассматриваемого элемента, но после того, как изменит свое значение функция $\bar{f}$ ". Если функция $\bar{f}$ " изменяет свое состояние несколько раз, то промежуточный элемент (P") должен изменять свое состояние после последнего состояния функции $\bar{f}$ ".

В качестве промежуточного элемента желательно использовать при возможности другие выходные или уже введенные в циклограмму промежуточные элементы, изменяющие свое состояние так, как необходимо для обеспечения условий несрабатывания данного элемента. Условия несрабатывания в этом случае записываются в виде инверсии произведения двух сигналов f " и p ", т.е. $\overline{f\,''p''}$. Значение сигналов f " и p" берутся такими, которые они имеют в отключающем такте.

Третья проверка предназначена для контроля того, чтобы после отключения выходного элемента не создались вновь условия для его повторного (неправильного) включения. С этой целью следует проверить каждую комбинацию сигналов в выражениях $f' \bar{f}''$, $f' + p' \bar{f}''$, $f' \overline{f''p''}$ или $f' + p' \overline{f''p''}$ после того, как будут раскрыты все скобки и исключены общие знаки инверсии, т.е. получено выражение функции в дизъюнктивной нормальной форме, причем выражение вида $X_n = f'_{n(x)} \bar{f}''_{n(x)}$ получается, если первоначально принятые условия срабатывания и несрабатывания удовлетворяют всему включающему периоду данного элемента; выражение

$$X_n = f'_{n(x)} + p' \bar{f}''_{n(x)}$$

получается, если в условия срабатывания вводится дополнительный элемент; выражение

$$X_n = f'_{n(x)} \overline{p''f''}_{n(x)}$$

получается, если в условие несрабатывания вводится дополнительный элемент;

выражение

$$X_n = f'_{n(x)} + p'\overline{p''f''}_{n(x)}$$

получается, если дополнительные элементы вводятся в условие срабатывания и несрабатывания.

Каждое слагаемое в ДНФ функции X в данном периоде включения после организации двух проверок дает определенную комбинацию сигналов и их инверсий. Ни одна из этих комбинаций не должна встречаться в отключающих периодах элемента. Если в отключающем периоде появляется одна из этих комбинаций, то в эту комбинацию должен быть введен еще хотя бы один дополнительный промежуточный элемент P'''. Состояния этого элемента должны быть различны при данной комбинации во включающем и отключающем периодах.

В качестве промежуточного элемента также желательно использовать уже введенные в циклограмму промежуточные или выходные элементы, если они удовлетворяют условиям применения этого промежуточного элемента. Сигнал вводимого промежуточного элемента приписывается в виде произведения к тому слагаемому, которое дает комбинацию, встречающуюся во включающем и в отключающем периодах, состояние этого элемента принимается таким, какое он имеет при данной комбинации во включающем периоде.

Таким образом, в результате третьей проверки общий вид структурной формулы для исполнительного элемента X в n-м периоде его включения будет:

$$X_n = f'_{n(x)}\,\overline{f''}_{n(x)}p''',$$

если введен дополнительный элемент после третьей проверки;

$$Xn = f'n(x) + p'\overline{f''}_{n(x)}p''',$$

если введен дополнительный элемент из условия срабатывания после первой проверки и во второе слагаемое по третьей проверки;

$$X_n = f'_{n(x)}\,p''' + p'\overline{f''}_{n(x)},$$

если введен дополнительный элемент в условия срабатывания после первой проверки и в первое слагаемое после третьей проверки.

Для выражения вида

$$X_n = f'_{n(x)}\,\overline{p''f''}_{n(x)} = f'_{n(x)}\,(\overline{p''} + \overline{f''}_{n(x)}) = f'_{n(x)}\,\overline{p''} + f'_{n(x)}\,\overline{f''}_{n(x)}$$

при необходимости p''' приписывается или в виде $f'_{n(x)}\,\overline{p'}p'''$ или $f'_{n(x)}\,\overline{f''}_{n(x)}p'''$.

Для выражения вида

$$X_n = f'_{n(x)} + p'\overline{p''f''}_{n(x)} = f'_{n(x)} + p'\overline{p''} + p'\overline{f''}_{n(x)}$$

при необходимости в результате третьей проверки p''' приписывается в виде $f'_{n(x)}p'''$, или $p'\,\overline{p'}\,p'''$, или $p'\overline{f''_{n(x)}}p'''$ в зависимости от того, какая из комбинаций ($f'_{n(x)}$, или p', $\overline{p''}$, или p', $\overline{f''_{n(x)}}$) встречаются во включающем и отключающем периодах.

После того как проведены три проверки для каждого периода включения данного элемента, полученную структурную формулу

$$X = X_1 + X_2 + \ldots + X_n = f'_{1(x)}\,\bar{f}''_{1(x)} + f'_{2(x)}\,\bar{f}''_{2(x)} + \ldots + f'_{n(x)}\,\bar{f}''_{n(x)}$$

можно упрощать, используя алгебраические равносильности и другие преобразования с целью получения рациональной структурной схемы.

Аналогично получаются структурные формулы для остальных выходных элементов. Для промежуточных элементов структурные формулы составляются так же, как и для выходных.

Общая структурная формула всего логического устройства составляется в виде суммы произведений общей формулы каждого выходного или промежуточного элемента на сигнал этого элемента

$$F = (X_1 + X_2 + \ldots + X_n)X + (Y_1 + Y_2 + \ldots + Y_n)Y + (P_1 + P_2 + \ldots + P_n)P + \ldots$$

Общая структурная формула при возможности преобразовывается с целью упрощения общей схемы логического устройства.

Основным преобразованием этой формулы, как правило, является вынесение общих множителей за скобки, что при реализации алгебраических выражений дает возможность выделять в схеме узлы, общие для нескольких выходов, и вычерчивать их один раз, не повторяя для каждого выхода в отдельности.

При синтезе логических устройств, работающих по нескольким циклам, следует придерживаться приведенного ниже порядка.

Составляются циклограммы работы всех элементов для каждого цикла. Все циклограммы объединяются в одну, общую для устройства, подразделенную на циклы.

По циклограмме каждого цикла составляются структурные формулы, производятся три проверки, предусмотренные методом, и корректировки формул.

Производится четвертая проверка, в результате которой устанавливается, не встречается ли одинаковые комбинации переменных в циклограммах разных циклов. Если одинаковые комбинации переменных встречаются, то вводятся дополнительные элементы, так же, как и в при третьей проверке. При этом, если в условия срабатывания входит несколько основных сигналов (например элемент X включится при появлении сигналов a_1, или a_2, или a_3), то в структурную формулу условия срабатывания записываются в виде логической суммы

$$f'_{(x)} = a_1 + a_2 + a_3.$$

Если в условия несрабатывания входит несколько основных сигналов (например, исполнительный элемент X отключится, если будут сигналы или b_1, или b_2, или b_3), то в структурной формуле условия несрабатывания записываются в виде инверсии логической суммы

$$\overline{f''} = \overline{b_1 + b_2 + b_3} = \overline{b}_1\overline{b}_2\overline{b}_3.$$

Функции f' и $\overline{f''}$ могут иметь и более сложные выражения, которые составляются на основании словесных формулировок условий срабатывания и несрабатывания при наличии нескольких основных сигналов в соответствующих тактах.

1.4.3. Составление циклограмм по алгебраическим выражениям

Чтение и анализ работы сложных схем циклически работающих логических устройств представляют известную трудность, в особенности, если эти схемы построены из логических элементов (например, интегральных микросхем). В этих случаях целесообразно построить циклограмму, дающую наглядное представление о работе анализируемой схемы.

Построение циклограммы можно выполнить на основании предварительно составленных алгебраических выражений для выходных и промежуточных элементов.

Этот способ целесообразно также использовать для проверки правильности синтеза схемы по циклограмме. По полученным в результате синтеза алгебраическим выражениям строится циклограмма и сравнивается с исходной. Совпадение начертаний циклограмм свидетельствует о правильности выполнения синтеза. Процедура может быть представлена следующими операциями:

- записывается система алгебраических выражений для выходных и промежуточных элементов;
- задается последовательность изменения (появление и исчезновение) командных сигналов;
- уравнения по п.1 решаются для исходного состояния командных и промежуточных сигналов. Результаты решения отмечаются изображением сигналов в такте 0 (черта для сигнала 1, отсутствие черты - для 0);
- полученные в 0-м такте значения промежуточных сигналов при прежних значениях командных подставляются в уравнения п.1. При наличии изменений в результатах последние отражаются в следующем такте (такте 1);
- при отсутствии изменений в результатах рассматривается очередное изменение командных сигналов. Это отражается в следующем такте;

• снова решаются уравнения по п.1 с новыми значениями командных сигналов. Результаты отмечаются в следующем такте.

Далее процедура повторяется до тех пор, пока не будут учтены все заданные изменения командных сигналов и реакций на эти изменения.

В результате будет построена циклограмма работы устройства (пример смотри в 1.6).

1.5. Применение ЭВМ при синтезе и реализации логических функций

Здесь рассматриваются методы, не требующие сложных разработок программ, такие как метод синтеза по таблицам переходов, минимизации булевых функций, устранения критических состязаний при кодировании состояний и т.п.

1.5.1. Метод замены входных переменных [1]

Метод замены входных переменных представляет собой многошаговый процесс построения структуры релейного устройства в направлении от входов к выходам.

Исходными данными для построения комбинационной структуры этим методом являются непротиворечивая таблица состояний логической функции f от n входных переменных, заданной полностью определенной или недоопределенной, и функционально полный набор логических элементов.

Исходная таблица состояний представляет собой таблицу, в которой каждое определенное значение функции f (N - число определенных значений функции f) закодировано соответствующим, отличным от всех других набором значений n входных переменных (или, другими словами, значения f закодированы по-разному наборами значений кодирующих переменных). Целью синтеза функции f является уменьшение числа кодирующих переменных от n до 1 с одновременным уменьшением числа по-разному закодированных определенных значений заданной функции f от N до 2.

На первом шаге синтеза по исходной таблице состояний и набору логических элементов составляется *расширенная таблица состояний*, которая содержит столбец значений заданной функции f, столбцы значений переменных исходной таблицы состояний (x_1, x_2, …, x_n) и столбцы значений выходных функций логических элементов в рабочих вариантах их включения.

Вариантом включения логического элемента называется упорядоченная относительно его входов совокупность подаваемых на эти входы переменных. Варианты включения одного элемента отличаются друг от друга составом, числом или порядком его входных переменных. Варианты включения некоторого логического элемента, при которых входная функция этого элемента отличается от исходных переменных на данном шаге и констант, называются *рабочими вариантами*.

Полученная таким образом расширенная таблица состояний избыточна и поэтому из нее можно выделить неизбыточные таблицы состояний. Неизбыточной таблицей называется такая непротиворечивая таблица, удаление из которой столбца любой одной переменной сделает ее противоречивой. Такое выделение соответствует выделению различных вариантов правильного (непротиворечивого) кодирования значений синтезируемой функции *f*.

Эти варианты кодирования могут различаться между собой, например, числом n_s, кодирующих переменных при *s*-м варианте кодирования, т.е. числом столбцов в неизбыточной таблице состояний, или числом N_s поразному закодированных определенных значений функции *f*, т. е. числом строк в таблице состояний, и, кроме того, числом, составом, суммарной стоимостью и схемами включения требуемых для реализации такой таблицы логических элементов. Процедура получения оптимального решения в этом случае громоздка.

При получении приближенного решения процедура построения неизбыточных таблиц состояний может быть проведена в виде последовательности отдельных шагов. При этом должна быть предварительно назначена функция предпочтения, экстремальное значение которой используется при выборе частных решений на каждом шаге процедуры, и должна быть доказана сходимость процедуры, т. е. возможность получения полного решения за конечное число шагов. В этом случае сокращается общее число операций, хотя может быть получен вариант решения, отличный от оптимального.

На каждом шаге из числа кодирующих переменных расширенной таблицы состояний выбирается одна кодирующая переменная искомой сокращенной таблицы состояний.

Шаги процедуры построения сокращенной таблицы состояний выполняются до тех пор, пока не будет получена непротиворечивая таблица. Эта таблица является исходной для осуществления следующего шага процедуры синтеза. По ней снова строится расширенная таблица состояний. Логические элементы, соответствующие кодирующим переменным новой таблицы состояний, включаются в синтезируемую структуру по схемам, соответствующим вариантам включения этих элементов. Синтез завершается получением таблицы состояний с одной кодирующей

переменной, значения которой совпадают с соответствующими значениями заданной для реализации функции.

Для обеспечения сходимости процесса при использовании функционально полного набора логических элементов достаточно получать монотонное изменение функции предпочтения от шага к шагу, например, достаточно, чтобы число кодирующих переменных синтезируемой функции на *s*-м шаге синтеза было меньше, чем на *s*-1-м, или равным ему при условии, что в новой таблице состояний будет меньшее число различных строк.

В качестве функции предпочтения *R*, оценивающей свойства кодирующих переменных, примем число пар различных значений функции *f*, кодируемых одинаково значениями этих переменных.

Для одной кодирующей переменной

$$R=N_{0(0)}N_{1(0)}+N_{0(1)}N_{1(1)},$$

здесь, например, $N_{0(0)}$ - число нулевых значений функции, кодируемых нулевым значением переменной.

Например:

f	x_1
0	1
1	1
1	1
0	0
0	1

$$N_{0(0)} = 1,\ N_{1(0)} = 0,\ N_{0(1)} = 2,\ N_{1(1)} = 2,$$
$$R = 1 \times 0 + 2 \times 2 = 4$$

Для двух кодирующих переменных

$$R = N_{0(00)}N_{1(00)} + N_{0(10)}N_{1(10)} + N_{0(01)}N_{1(01)} + N_{0(11)}N_{1(11)};$$

здесь, например, $N_{1(00)}$ - число единиц функции, кодируемых комбинацией 00 кодирующих переменных. Для трех кодирующих переменных будет соответственно 8 возможных вариантов, для четырех - 16 и т. д.

При выборе кодирующих переменных из расширенной таблицы состояний следует выбирать сначала одну переменную с $R_{мин1}$, затем к этой

переменной подбирать следующую, рассматривая попарное кодирование функции f, обеспечивающее $R_{мин2}$, к двум выбранным переменным таким же образом подбирать третью и т. д., пока на k-м шаге не получится $R_{мин\,k} = 0$. Значение функции предпочтения $R = 0$ говорит о том, что получена новая непротиворечивая таблица состояний, по которой (если она содержит более одной кодирующей переменной) следует строить очередную расширенную таблицу состояний.

Этот метод можно распространить на синтез структур устройств, условия работы которых заданы в виде *циклограмм* или *таблиц включений*. Исходная таблица состояний может быть записана по таблице включений следующим образом. Число строк таблицы состояний будет равно числу тактов таблицы включения. Комбинациям значений входных переменных в i-м такте ставится в соответствие значение выходных переменных в i+1-м такте. Например, на рисунке 1.17 приведена таблица включений для функции $f(x_1, x_2, x_3)$, а в таблице 1.4 - таблица состояний. Значение функции предпочтения для такой исходной таблицы состояний, как правило, не равно нулю, следовательно, структура f будет не комбинационной, а с обратными связями или с элементами памяти.

Поэтому в набор логических элементов добавляются элементы памяти (задержки, триггеры) и процедура синтеза может быть теперь представлена следующим образом.

1. Записать заданную таблицу включений в форме таблицы состояний для выходных функций, расположив строки сверху вниз в порядке следования тактов таблицы включений. При этом каждой комбинации значений входных переменных в (i-м такте ставится в соответствие значение выходных функций в i+1-м такте.

2. Составить расширенную таблицу состояний, столбцами которой являются не только выходные функции логических элементов в рабочих вариантах их включения (как это делается в методе замены входных переменных при синтезе комбинационных структур), но также выходные функции элементов памяти, реализуемые ими при всех возможных начальных состояниях и различных функциях возбуждения, являющихся входными переменными элементов памяти.

3. Пользуясь расширенной таблицей состояний данного шага, найти с помощью функции предпочтения лучшую таблицу следующего шага.

4. После получения непротиворечивой таблицы состояний с $R = 0$ задача становится комбинационной и может решаться без использования элементов памяти.

Таблица 1.4

Таблица состояний

Номер такта	Переменные			
	Входные			Выходная
	x_1	x_2	x_3	Z
0	0	0	0	0
1	1	0	0	1
2	1	0	0	1
3	0	0	0	1
4	0	1	0	1
5	0	1	1	0
6	0	1	1	0
7	0	0	1	1
8	0	0	1	1
9	0	0	0	0
10	0	0	0	0

Входы и выходы	Такты										
	0	1	2	3	4	5	6	7	8	9	10
x_1											
x_2											
x_3											
Z											

Рис.1.17. Таблица включений

1.5.2. Непосредственное вычисление логических функций.

Предварительно логическая функция записывается через элементарные функции И, ИЛИ, НЕ, наиболее доступные для программной реализации на базе микропроцессоров и ЭВМ. Запись может быть как в скобочной, так и в дизъюнктивной нормальной форме, хотя последняя по трудоемкости реализации на ЭВМ предпочтительнее.

Процедура реализации логической функции может быть продемонстрирована на примере.

$$X = a(b\bar{c} + de) + \bar{k}$$

Если принять во внимание, что логические переменные соответствуют реальным физическим сигналам объекта управления, то первая операция будет представлять собой считывание с объекта сигналов a, b, c, d, e, k в порты ввода ЭВМ и запоминание их. Затем эти сигналы извлекаются из памяти и над ними производятся логические операции:

инверсия c, логическое перемножение $b\bar{c}$ и запись в соответствующую ячейку памяти, логическое перемножение de и запись в память, логическое сложение извлеченных из ячеек памяти $b\bar{c} + de$, перемножение результата с извлеченным из памяти значением сигнала a (получаем $a(b\bar{c} + de)$), инверсию $\bar{k}$ складываем с полученным результатом и получаем значение функции X, которую в качестве выходного сигнала выводим на соответствующий порт вывода ЭВМ, затем дешифруем, в случае необходимости, усиливаем и в результате доводим до исполнительного органа объекта. Этот метод достаточно прост в реализации, но с ростом числа переменных увеличивается длина программы и время ее реализации. Кроме того, каждая функция требует своей отдельной программы. Очевидно, этот метод целесообразно использовать для реализации относительно несложных функций при небольшом числе входных переменных, требующих ограниченного (приемлемого) числа портов ввода.

1.5.3. Метод отображения входного набора [2].

Рассмотрим функцию

$$y = \bar{a}b\bar{c} + d$$

Положим, что требуется вычислить значение этой функции на входном наборе переменных a, b, c, d равном 1000. Будем считать, что такой входной набор определяет входное слово 1000. Функция y включает в себя две конъюнкции: $\bar{a}b\bar{c}$ и d. Если прямому вхождению переменной в конъюнкцию поставить несоответствие 1, а инверсному вхождению - 0, то конъюнкции $\bar{a}b\bar{c}$ будет соответствовать набор 010 x, а конъюнкции d - набор xxx1. Символом « x» обозначены переменные, не входящие в рассматриваемую конъюнкцию. Построенные таким образом наборы далее будем называть словами-конъюнкциями. Очевидно, что если входное слово совпадает по всем разрядам с одним из слов-конъюнкций рассматриваемой функции, то значение функции на данном входном наборе будет равно 1. В противном случае функция равна 0. Если конъюнкция образована не всеми переменными, то сравниваются только те разряды слова-конъюнкции, которые не содержат символа «x».

Такой подход положен в основу программного вычисления булевых функций по методу отображения входного набора. Поскольку в этом случае время вычисления булевых функций, а также объем занимаемой

памяти будут пропорциональны числу конъюнкций в ДНФ реализуемой функции, то предварительно необходимо минимизировать искомую функцию, т. е. построить ее кратчайшую ДНФ (с минимальным числом конъюнкций).

Обратимся к реализации такого подхода. Как и ранее, в качестве искомой функции возьмем функцию ***y***. Будем считать, что входные переменные уже введены в память микропроцессора и упорядочены в одно входное слово, каждый разряд которого соответствует одной входной переменной. С каждым из слов-конъюнкций искомой функции сопоставляется маскирующее слово. В разрядах маскирующего слова записаны 1, если соответствующие разряды слова-конъюнкции являются определенными, т. е. содержат 1 или 0. Если некоторый разряд слова-конъюнкции является неопределенным (т.е. содержит символ « x»), то в соответствующий разряд маскирующего слова записывается 0. Хранение маскирующих слов и слов-конъюнкций осуществляется в стеке. Пусть стек организован в оперативном ЗУ, а обращение к нему осуществляется через регистр указателя стека, в котором постоянно хранится адрес первого слова стека. По мере продвижения данных из стека в рабочие регистры микропроцессора содержимое регистра указателя стека автоматически увеличивается на единицу. Последним словом стека всегда является нулевое слово, т. е. слово, все разряды которого равны нулю.

Программа первоначально осуществляет поразрядное сложение по модулю 2 разрядов входного слова и текущего слова-конъюнкции. Если соответствующие разряды этих двух слов совпадают, то разряд результирующего слова будет нулевым. В случае несовпадения этих разрядов соответствующий разряд результирующего слова будет единичным. Для исключения влияния неопределенных разрядов в слове-конъюнкции производится поразрядное логическое умножение результирующего слова и соответствующего маскирующего слова. Если все разряды полученного при этом слова будут нулевыми, то функции присваивается единичное значение. При считывании из стека нулевого слова вычисления прекращаются. При использовании этого метода искомая функция будет равна единице только в том случае, когда наблюдается полное совпадение входного слова и одного из слов-конъюнкций в тех разрядах, для которых в маскирующем слове записаны единицы.

По сравнению с методом непосредственного вычисления булевых функций рассмотренный подход обладает рядом преимуществ. Во-первых, программа вычислений в значительной степени является универсальной и единственное изменение, которое требуется при переходе к реализации новой функции, сводится либо к изменению содержимого стека, либо к организации дополнительного стека и занесению адреса первого слова нового стека в регистр указателя стека. Во-вторых, данный метод является

более быстродействующим (особенно при большом числе входных переменных), так как для вычисления функции требуется сравнение входного слова не со всеми словами-конъюнкциями, а до обнаружения первого совпадения. В-третьих, более рационально используется в память микропроцессора за счет размещения в одной ячейке нескольких входных переменных.

Недостатки данного подхода заключаются в возможной значительной длине стека при большом числе конъюнкций в ДНФ реализуемой функции, а также в необходимости сохранения содержимого регистра указателя стека при обращении к этому регистру системных программ (например, при обработке прерываний) или других программ пользователей.

1.5.4. Использование таблицы состояний.

В память ЭВМ помещается таблица состояний, которая предварительно вычисляется по логическому выражению функции. В этой таблице каждой комбинации значений входных сигналов ставится в соответствии значение функции. В процессе реализации функции программа считывает с объекта значения входных сигналов, формирует из них набор, сравнивает полученный набор с записанными в таблице состояний, на совпадающем наборе фиксирует значении функции и выводит его в порт вывода для дальнейшей транспортировки до исполнительного органа. Для упрощения программы в качестве кодов-наборов значений входных переменных могут быть использованы коды адресов памяти, которым поразрядно ставятся в соответствие значения входных сигналов, а в самой ячейке памяти записывается соответствующее значение выходного сигнала. Тогда набор значений входных сигналов, считываемых с объекта, сравнивается с кодами ячеек памяти и в случае совпадения на выход выдается значение функции.

Этот метод хорош своей универсальностью, так как не требует изменений в программе при работе с различными функциями, обеспечивает большое быстродействие, не требует предварительных затрат времени на формирование таблицы состояний.

1.6. Практические занятия

Задание 1.

1.1. Упростить исходную схему (или алгебраическое выражение).

1.2. Построить бесконтактную схему по п.1.1 в заданной элементной базе.

Задание 2.

2.1. По заданной таблице состояний построить карту Карно.

2.2. Получить алгебраические выражения:

2.2.1. по единицам;

2.2.2. по нулям;

2.2.3. с учетом неопределенностей.

Задание 3.

3.1. По заданному графу переходов построить таблицу переходов.

3.2. Построить бесконтактную схему на любой элементной базе:

3.2.1. при простом кодировании;

3.2.2. при использовании элементов памяти (RS триггеров, …).

Задание 4.

4.1 По циклограмме получить алгебраическое выражение.

4.2. Построить схему в релейно-контактном варианте.

4.3. Построить схему в бесконтактном варианте.

4.4. Проанализировать полученный результат, построить циклограмму.