Глава 12

ЛОГИЧЕСКИЕ ОСНОВЫ РАБОТЫ КОМПЬЮТЕРОВ

12.1. Общие сведения о двоичных алгебрах

В широком смысле под **алгеброй** понимают раздел математики, изучающий общие свойства операций над элементами множества произвольной природы.

Чтобы задать алгебру, нужно задать некоторое множество элементов (в частности, переменных и констант) и определяемое на них некоторое множество операций. Кроме того, требуется задать алфавит алгебры, правила записи и преобразования формул.

В школьной алгебре в качестве множества констант и переменных выступает множество вещественных чисел, а множество операций включает операции: сложение, вычитание, умножение, деление, возведение в степень и извлечение корня. Множество входящих в алгебру операций называют **базисом**. Результаты операций определяют функции: сложение — сумму, вычитание — разность, умножение — произведение и т.д.

Заметим, что набор операций школьной алгебры избыточен. Например, возведение в степень можно заменить умножением, умножение можно заменить сложением. Вычитание также можно заменить сложением, как это делается в компьютерах, имеющих только схему сумматора. Уменьшение набора операций привело бы к громоздким, трудно обозримым формулам. Увеличение набора добавлением других операций также нежелательно.

Особое место среди алгебр занимают двоичные алгебры. Из множества двоичных алгебр познакомимся только с двумя: **алгеброй**

логики и булевой алгеброй. Это алгебры двоичных переменных, констант и функций, принимающих только два значения: **истина** и **ложь** в алгебре логики и соответственно **единицу** и **ноль** в булевой алгебре. Начнем с булевой алгебры, имеющей более простой базис, состоящий всего из трех операций.

Важность знакомства с двоичными алгебрами заключается в следующем. Во-первых, они являются математической основой построения всех логических схем компьютеров, обрабатывающих информацию в двоичной системе счисления. Во-вторых, они служат математической основой решения сложных логических задач.

12.2. Двоичные переменные и функции

Двоичными называют переменные, способные принимать только два значения: 0 и 1. Двоичные функции — это функции двоичных переменных. Они также принимают только два значения — 0 и 1. Двоичные переменные и функции называют булевыми переменными и булевыми функциями (БФ).

Область определения БФ n аргументов составляют:

при n = 1 ..... значения 0 и 1 этого аргумента ($2^n = 2$);

при n = 2 ..... наборы 00, 01, 10 и 11 — 4 набора значений аргументов ($2^n = 4$);

при n = 3 ..... наборы 000, 001, 010, 011, 100, 101, 110 и 111 — 8 наборов ($2^n = 8$) и т.д.

Можно сделать вывод, что область определения БФ n аргументов состоит из 2^n наборов. Область значений БФ — это множество всего из двух значений 0 и 1.

Конечность области определения и области значений БФ позволяют задавать эти функции не только формулами, но и таблично. В виде таблицы БФ задают значениями 0 или 1 на каждом из наборов значений ее аргументов. Такие таблицы называют **таблицами истинности** (в алгебре логики в таких таблицах вместо 0 пишут Л — ложь, вместо 1 пишут И — истина).

Таблица 12.1 представляет собой пример таблицы истинности (ТИ) функций $f_m(x_1, x_2)$ и $f_{19}(x_1, x_2)$ двух аргументов x_1 и x_2.

Таблица истинности *Таблица 12.1*

a	x_1x_2	$f_m(x_1, x_2)$	$f_{13}(x_1, x_2)$
0	0 0	f(0,0)	1
1	0 1	f(0,1)	1
2	1 0	f(1,0)	0
3	1 1	f(1,1)	1

Величина a обозначает номер набора значений аргументов, m — номер функции.

Функция $f_m(x_1, x_2)$ — это задание в общем виде любой из 16-ти функций двух аргументов. f(0, 0) — значение функции при подстановке в ее формулу значений $x_1 = 0$ и $x_2 = 0$, то есть на нулевом наборе аргументов. f(0,1) — значение функции на первом наборе аргументов и т. д.

Значение булевой функции на всех наборах своих аргументов образуют двоичное число, называемое **кодом функции**. Перевод кода функции в десятичную систему дает **номер функции**. В табл. 12.1 занесена функция $f_{19}(x_1, x_2)$, с кодом 1101 и номером 13.

С ростом числа аргументов n число различных функций резко возрастает. Если число наборов определяется формулой Кн = 2n, то число функций вычисляется по формуле Кф = 2Кн. При n = 1 Кн = 2, Кф = 4; при n = 2 Кн = 4, Кф = 16; при n = 4 Кн = 16, Кф = 65536 и т.д. При н = 10 Кф сравнимо с числом элементарных частиц во Вселенной.

Все функции изучить невозможно. Поэтому, как и в обычной школьной алгебре, выбирают некоторое количество функций малого числа (один, два) аргументов и применяют их в качестве операций для построения формул функций любой сложности.

Рассмотрим булевы функции одного и двух аргументов и подробно остановимся на функциях, входящих в базис булевой алгебры, базис алгебры логики, базисы других двоичных алгебр.

12.3. Булевы функции одного аргумента

Все четыре функции на двух наборах одного аргумента показаны в табл. 12.2.

Таблица 12.2

x	$f_0(x)$	$f_1(x)$	$f_2(x)$	$f_3(x)$
0	0	0	1	1
1	0	1	0	1

Функция $f_0(x)$ называется **константа нуль**. Она независимо от значений аргумента равна нулю.

Функция $f_1(x)$ — это **переменная x**. Она повторяет значения переменной.

Функция $f_2(x)$ называется **инверсия (отрицание x)**. Она входит в базис булевой алгебры и алгебры логики. Рассмотрим ее подробнее.

Функция $f_3(x)$ называется **константа единица**. Она независимо от значений аргумента равна единице.

Инверсия — функция одного аргумента. Логическая операция над аргументом отрицание. Часто отождествляют функцию с операцией и говорят: «функция отрицание» или «операция инверсия». Однако при строгом формальном подходе отождествлять результат с действием нежелательно.

Знак операции — черта над аргументом, например, $\bar{x}$ или $\neg x$. Такая запись читается: «не x» или «отрицание x». В языках программирования также широко используются логические операции, реализующие булевы функции. Функция инверсия аргумента x записывается так: **NOT x**.

Функцию инверсия в схемах компьютера реализует логический элемент **инвертор** (элемент НЕ). Схема инвертора показана на рис. 12.1. Работает инвертор так: если на входе 0, то на выходе 1, если на входе 1, то на выходе 0.

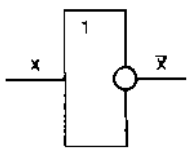

Рис. 12.1. *Условное обозначение инвертора*

12.4. Булевы функции двух аргументов

Двум аргументам соответствуют четыре набора их значений, что приводит к шестнадцати различным кодам функций (н = 2, Кн = 2^n = 4, Кф = $2^{Кн}$ = 16). Все 16 функций двух аргументов занисаны в табл. 12.3.

Булевы функции двух аргументов — Таблица 12.3

a	x_1x_2	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	00	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
1	01	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
2	10	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
3	11	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Среди функций двух аргументов имеются уже знакомые по функциям одного аргумента:

f_0 и f_{15} ..... соответственно функция константа 0 и ее инверсия — константа 1;

f_3 и f_{12} ..... переменная x_1 и ее инверсия $\bar{x}_1$;

f_5 и f_{10} ..... переменная x_2 и ее инверсия $\bar{x}_2$.

Функции f_0 и f_{15} фиктивно зависят от обоих аргументов. Изменения значений аргументов не влияют на значение этих функции. Функции f_3, f_5, f_{10} и f_{12} фиктивно зависят от одного аргумента и существенно зависят от другого аргумента.

Из оставшихся десяти функций также можно выделить пять пар — пять функций и их инверсий: f_1 и f_{14}, f_2 и f_{13}, f_4 и f_{11}, f_6 и f_9, f_7 и f_8. Рассмотрим их.

Конъюнкция

Булева функция $f_1(x_1, x_2)$ с кодом 0001 называется конъюнкцией. **Коиъюнкция** — это такая булева функция, которая равна единице тогда и только тогда, когда все аргументы функции равны единице. Другое определение — это такая функция, которая равна нулю, если хотя бы один аргумент функции равен нулю.

Таблица истинности функции конъюнкция — табл. 12.4.

Таблица 12.4
Таблица работы конъюнктора

a	$x_1\ x_2$	$f_1(x_1x_2)$
0	0 0	0
1	0 1	0
2	1 0	0
3	1 1	1

Функцию копъюпкция получаем как результат операции логическое умножение. Знак операции: & или ∧ (в теоретических работах по алгебре логики). В формулах, как и в обычной алгебре, знак чаще всего опускается: $f(x_1, x_2) = x_1 \& x_2 = x_1 \cdot x_2 = x_1x_2$. Читается формула так: «x_1 и x_2».

Запись в языках программирования: x_1 AND x_2.

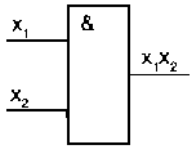

Рис. 12.2. *Обозначение конъюнктора*

Функцию конъюнкция реализует логический элемент конъюнктор (элемент И). Условное обозначение конъюнктора в логических схемах показано на рис. 12.2.

Используя принцип сунерпозиции (подстановку функции в качестве аргументов в другую функцию), фупкцию конъюнкция можно обобщить на n аргументов: $f(x_1, x_2, \ldots, x_n) = x_1x_2 \ldots x_n$.

В качестве содержательного нримера реализации функции конъюнкция рассмотрим схсму голосования «только все!». На рис. 12.3 показана цепь с N кпонками, позволяющими включать индикаторную лампочку. На электрических выключателях принято отмечать: 0 — выключено и 1 — включено. Ламночка засветится только н случае, если будут замкнуты все ключи, то есть на все N входов будут «ноданы» единицы. Такая схема реализует фупкцию конъюнкция.

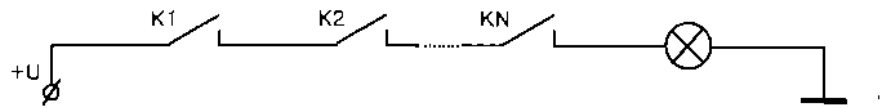

Рис. 12.3. *Реализация конъюнкции в схеме голосования «только все»*

Инверсия конъюнкции. Функция Шеффера. В таблице истинности функций двух аргументов (табл. 12.3) эта функция $f_{14}(x_1, x_2)$. Она имеет код функции 1110.

Запись формулы функции следующая:

$$f(x_1, x_2) = \overline{x_1 \,\&\, x_2} = \overline{x_1 \cdot x_2} = \overline{x_1x_2}.$$

Инверсия конъюнкции, как видно из ее формулы, образована из двух более нростых функций: копъюнкции и инверсии. Эта функция замечательна тем, что она в единственном числе образует базис алгебры — алгебры Шеффера, позволяющей занисать в виде формулы любую двоичную логическую функцию.

В алгебре Шеффера единственная функция — функция Шеффера. Знак операции: | — штрих Шеффера. В алгебре Шеффера функция ивперсия конъюнкции записывается так: $f(x_1, x_2) = x_1|x_2$.

Логический элемент, реализующий функцию Шеффера, называется элементом Шеффера или элемснтом И-НЕ. Условное обозначение элемента И-НЕ показано на рис. 12.6.

Фупкция Шеффера также обобщается на n аргументов.

Инверсия дизъюнкции. Функция Пирса. В табл. 12.3 она находится под номером 8 и имеет код функции 1000. Это функция $f_8(x_1, x_2) = \overline{x_1 \vee x_2}$.

Функцию инверсия дизъюнкции еще называют функцией Пирса. Она, как и функция Шеффера универсальна в том смысле, что в единственном числе образует алгебру Пирса и позволяет записать формулой любую двоичную фупкцию. Знак операции: ↓ — стрелка Пирса. Запись функции Пирса: $f(x_1,x_2) = x_1 \downarrow x_2$.

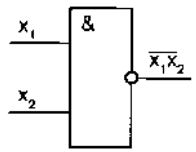

Рис. 12.6. *Обозначение элемента И-НЕ*

Логический элемент, реализующий функцию Пирса, называют элемент Пирса или элемент ИЛИ-НЕ. Условное обозначение элемента на рис. 12.7.

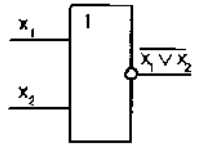

Рис. 12.7. *Условное обозначение логического элемента ИЛИ-НЕ*

Импликация. Это логическая функция двух двоичпых логических переменных x_1 и x_2. Различают импликацию от x_1 к x_2 (функция f_{13} с кодом 1101) и импликацию от x_2 к x_1 (функция f_{11} с кодом 1011). Обе эти функции приведены

Инверсия конъюнкции. Функция Шеффера. В таблице истинности функций двух аргументов (табл. 12.3) эта функция $f_{14}(x_1, x_2)$. Она имеет код функции 1110.

Запись формулы функции следующая:

$$f(x_1, x_2) = \overline{x_1 \,\&\, x_2} = \overline{x_1 \cdot x_2} = \overline{x_1 x_2}.$$

Инверсия конъюнкции, как видно из ее формулы, образована из двух более простых функций: конъюнкции и инверсии. Эта функция замечательна тем, что она в единственном числе образует базис алгебры — алгебры Шеффера, позволяющей записать в виде формулы любую двоичную логическую функцию.

В алгебре Шеффера единственная функция — функция Шеффера. Знак операции: | — штрих Шеффера. В алгебре Шеффера функция инверсия конъюнкции записывается так: $f(x_1, x_2) = x_1|x_2$.

Логический элемент, реализующий функцию Шеффера, называется элементом Шеффера или элементом И-НЕ. Условное обозначение элемента И-НЕ показано на рис. 12.6.

Функция Шеффера также обобщается на n аргументов.

Инверсия дизъюнкции. Фуикция Пирса. В табл. 12.3 она находится под номером 8 и имеет код функции 1000. Это функция $f_8(x_1, x_2) = \overline{x_1 \vee x_2}$.

Функцию инверсия дизъюнкции еще называют функцией Пирса. Она, как и функция Шеффера универсальна в том смысле, что в единственном числе образует алгебру Пирса и позволяет записать формулой любую двоичную функцию. Знак операции: ↓ — стрелка Пирса. Запись функции Пирса: $f(x_1,x_2) = x_1 \downarrow x_2$.

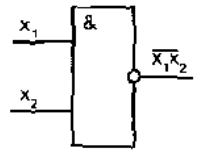

Рис. 12.6. *Обозначение элемента И-НЕ*

Логический элемент, реализующий функцию Пирса, называют элемент Пирса или элемент ИЛИ-НЕ. Условное обозначение элемента на рис. 12.7.

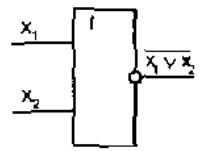

Рис. 12.7. *Условное обозначение логического элемента ИЛИ-НЕ*

Импликация. Это логическая функция двух двоичных логических переменных x_1 и x_2. Различают импликацию от x_1 к x_2 (функция f_{13} с кодом 1101) и импликацию от x_2 к x_1 (функция f_{11} с кодом 1011). Обе эти функции приведены

Таблица 12.6

Функции импликация

x_1, x_2	$x_1 \to x_2$	$x_2 \to x_1$
0 0	1	1
0 1	1	0
1 0	0	1
1 1	1	1

в табл. 12.6. Операцию, соответствующую функции импликация, также называют импликацией и обозначают стрелкой от одной логической переменной к другой.

Операция импликация входит в базис алгебры логики, где оперирует с истинными и ложными высказываниями. *Об этом подробнее будет рассказано* в главе 13.

Запись формул функций импликация в булевой алгебре и в алгебре логики следующая:

$$f_{13}(x_1, x_2) = \overline{x}_1 \vee x_2 = x_1 \to x_2;$$

$$f_{11}(x_1, x_2) = x_1 \vee \overline{x}_2 = x_2 \to x_1.$$

Логическая операция IMP (импликация) используется в языках программирования. Запись следующая: x_1 **IMP** x_2.

Неравнозначность. Эту логическую функцию двух аргументов еще называют **исключительным ИЛИ** (или один, или другой, но не оба), а также суммой по модулю 2. Код функции — 0110, номер — 6. Функция принимает единичное значение только на тех наборах значений двух аргументов, на которых эти значения различны (01 и 10). Функции неравнозначность многоместная, ей соответствует логическая операция сумма по модулю 2. Операция входит в базис алгебры Жегалкина и имеет знак операции: ⊕. Иногда ее включают в базис алгебры логики. Запись формулы функции неравнозначность в булевой алгебре, алгебре Жегалкина и в языках программирования следующая:

$$f_6(x_1, x_2) = \overline{x}_1 x_2 \vee x_1 \overline{x}_2, \quad f_6(x_1, x_2) = x_1 \oplus x_2, \; x_1 \text{ XOR } x_2.$$

Равнозначность. Является обратной функцией к неравнозначности. Функция равнозначность (код функции 1001, номер функции 9) принимает значение 1 только на наборах с одинаковыми значениями аргументов (00 и 11). В качестве операции двойной импликации включается в базис алгебры логики (знак операции: ↔). Функцию равнозначность также называют **эквивалентностью**. Запись фор-

мулы функции в булевой алгебре, алгебре логики и в языках программирования следующая:

$$f_9(x_1, x_2) = \overline{x}_1\overline{x}_2 \vee x_1x_2, \quad f_9(x_1, x_2) = x_1 \leftrightarrow x_2, \; x_1 \, \mathrm{EQR} \, x_2.$$

Функции запрета. Это функции f_2 и f_4. Они являются обратными функциями к функциям импликации, широко использовались при проектировании и описании работы логических элементов ЭВМ 1-го и 2-го поколений.

12.5. Алгебры. Сравнение по набору операций

Таблица 12.7

Школьная и двоичные алгебры

Алгебра	Множество переменных и констант	Базис алгебры (набор операций)
Школьная	*вещественные числа*	+, −, *, /, $\sqrt{\ }$, степень
Булева	*0 и 1 (наборы)*	&, ∨, ¬
Шеффера	– " –	| — *штрих Шеффера*
Пирса	– " –	↓ — *стрелка Пирса*
Жегалкина	– " –	&, ⊕, 1
Алгебра логики	– " –	&, ∨, ¬, →, ↔

Из перечисленных в табл. 12.7 двоичных алгебр широко используются на практике только две — алгебра Буля и алгебра логики. Алгебры Шеффера и Пирса активно изучались на предмет возможности построения компьютера из однотипных элементов. К тому же, ввиду технологических особенностей, наиболее просто реализуются именно логические элементы Шеффера (И-НЕ) и Пирса (ИЛИ-НЕ).

12.6. Булева алгебра. Основные законы и тождества

Многие законы и правила преобразования формул булевой алгебры совпадают с законами в школьной алгебре. Но среди них есть тождества присущие только булевой алгебре. Перечислим все законы булевой алгебры. Это тождества, используемые для преобразования формул логических функций.

1.	$XY = YX$	переместительные законы
	$X \vee Y = Y \vee X$	(коммутативность)
2.	$X(YZ) = (XY)Z$	сочетательные законы
	$X \vee (Y \vee Z) = (X \vee Y) \vee Z$	(ассоциативность)
3.	$X(Y \vee Z) = XY \vee XZ$	распределительные законы
	$X \vee YZ = (X \vee Y)(X \vee Z)$	(дистрибутивность)
4.	$XX = X$	законы тавтологии
	$X \vee X = X$	(идемпотентность)
5.	$X \cdot 1 = X$	законы выполнения операций
	$X \vee 0 = X$	над константами
	$X \cdot 0 = 0$	
	$X \vee 1 = 1$	
6.	$X\overline{X} = 0$	законы инверсии для
	$X \vee \overline{X} = 1$	конъюнкции и дизъюнкции
7.	$\overline{X \cdot Y} = \overline{X} \vee Y$	законы де Моргана
	$\overline{X \vee Y} = \overline{X} \cdot \overline{Y}$	
8.	$XY \vee X\overline{Y} = X$	законы склеивания
	$(X \vee Y)(X \vee \overline{Y}) = X$	
9.	$X \vee XY = X$	
	$X(X \vee Y) = X$	законы поглощения
10.	$\overline{\overline{X}} = X$	закон снятия двойной инверсии

Докажем справедливость некоторых тождеств. Из способов доказательств выберем два. Первый способ заключается в приведении путем преобразования одной части равенства к другой части. Если это удается сделать, то тождество можно считать доказанным. Второй способ состоит в определении кодов функций левой и правой части равенства. Если коды равны, то и функции равны, что также доказывает справедливость тождества.

Первый способ. Первая строка распределительного закона не вызывает сомнений. Докажем справедливость второй строки:

$$X \vee YZ = (X \vee Y)(X \vee Z).$$

Правую часть равенства преобразуем к виду левой части. Справа будем записывать используемые при преобразовании законы и правила:

$(X \vee Y)(X \vee Z) = XX \vee XY \vee XZ \vee YZ =$ 'раскрываем скобки, $X \cdot X = X$

$X \vee XY \vee XZ \vee YZ =$ 'выносим X за скобки

$X(1 \vee Y \vee Z) \vee YZ =$ 'используем $X \vee 1 = 1$ и $X \cdot 1 = X$

$X \vee YZ$ 'результат преобразований

Справедливость тождества доказана.

Второй способ. Докажем справедливость правила де Моргана. Словами его можно сформулировать так: 1) инверсия конъюнкции равна дизъюнкции инверсий и 2) инверсия дизъюнкции равна конъюнкции инверсий. Докажем первую часть правила: $\overline{X \cdot Y} = \overline{X} \vee \overline{Y}$. Для доказательства используем табл. 12.8, строки которой соответствуют наборам значений аргументов X и Y.

Таблица 12.8

XY	$\overline{XY}$	$\overline{X} \vee \overline{Y}$
0 0	$\overline{0 \cdot 0} = \overline{0} = 1$	$\overline{0} \vee \overline{0} = 1 \vee 1 = 1$
0 1	$\overline{0 \cdot 1} = \overline{0} = 1$	$\overline{0} \vee \overline{1} = 1 \vee 0 = 1$
1 0	$\overline{1 \cdot 0} = 0 = 1$	$\overline{1} \vee \overline{0} = 0 \vee 1 = 1$
1 1	$\overline{1 \cdot 1} = \overline{1} = 0$	$\overline{1} \vee \overline{1} = 0 \vee 0 = 0$

Подставляя в левую и правую части равенства значения аргументов, вычисляем значения функций. В итоге получили, что коды обеих функций совпали. Тождество доказано.

Выполненный пример показывает возможность перехода от записи функции формулой к записи ее в таблице истинности. Для этого достаточно, подставляя в формулу функции все наборы значений аргументов, определить значение функции на этих наборах.

Правила преобразования формул

Кроме перечисленных выше законов для преобразования и упрощения формул булевых функций используются тождества, получившие название **правил** или **операций**. Перечислим их.

Правило отрицания:

$$\overline{f}(x, y, \ldots, z)_{\&,\vee} = f(\overline{x}, \overline{y}, \ldots, \overline{z})_{\vee,\&}.$$

Правило утверждает, что для получения отрицания некоторого выражения достаточно заменить в нем знаки дизъюнкции знаками конъюнкции, знаки конъюнкции знаками дизъюнкции, а все аргументы — их отрицаниями. Если в выражении имеются константы 0 и 1, то их также нужно заменить противоположными значениями.

Примеры.

$$\overline{x_1 \vee \overline{x}_2\left(\overline{x_3 \vee \overline{x}_4 \cdot x_5}\right)} = \overline{x}_1\left(x_2 \vee x_3\overline{x_4 \vee \overline{x}_5}\right);$$

$$\overline{\overline{x_1 \vee x_2 x_3} \vee 0} = x_1 \cdot (x_2 \vee \overline{x}_3) \cdot 1$$

Правило свертки:

$$x \vee \overline{x}y = x \vee y;$$
$$x(\overline{x} \vee y) = xy.$$

Правило обобщенного склеивания (теорема русского математика П.С. Порецкого):

$$xy \vee \overline{y}z \vee xz = xy \vee \overline{y}z;$$

$$(x \vee y)(\overline{y} \vee z)(x \vee z) = (x \vee y)(\overline{y} \vee z).$$

Справедливость тождеств можно доказать показанными выше способами.

12.7. Канонические формы булевых функций

Одна и та же функция может быть задана множеством различных формул. Поэтому функции трудно сравнивать. **Канонические формы функций** — это запись функций по единым правилам. Такие формы еще называют **совершенными**. Различают дизъюнктивную и конъюнктивную совершенные нормальные формы. Рассмотрим только дизъюнктивную форму.

Совершенная дизъюнктивная нормальная форма

Дизъюнктивной нормальной формой (ДНФ) булевой функции называют дизъюнкцию конъюнкций (логическую сумму логических произведений). Формула функции в ДНФ не имеет скобок и общих для нескольких аргументов отрицаний.

Пример ДНФ функции: $F(X,Y,Z) = X \vee \overline{Y}Z \vee \overline{X}Z \vee X\overline{Y}Z$.

Конституентой единицы называют полную (все аргументы) конъюнкцию отрицаемых или неотрицаемых аргументов. Обозначают — К(а), где **а** — номер того единственного набора, на котором К(а) = 1.

Пример: К(5) = $X\overline{Y}Z$. К(5) равна 1 только на 5-м наборе значений аргументов (101).

Совершенной дизъюнктивной нормальной формой (СДНФ) функции называют дизъюнкцию конституент единицы, равных единице на тех же наборах, что и функция.

Пусть заданы две булевы функции 3-х аргументов таблицей истинности (табл. 12.9). Функцию F(X, Y, Z) запишем в СДНФ. Функцию W(X, Y, Z) используем для следующих построений.

По определению СДНФ выпишем из таблицы конституенты единицы для наборов, на которых функция F равна 1.

Таблица 12.9

Переход от таблицы истинности функции к СДНФ

a	XYZ	F(X,Y,Z)	W(X,Y,Z)
0	000	0	0
1	001	1	1
2	010	1	0
3	011	0	1
4	100	1	0
5	101	0	1
6	110	0	1
7	111	1	0

$$F = K(1,2,4,7) = K(1) \vee K(2) \vee K(4) \vee K(7) =$$

$$\underset{001}{XYZ} \vee \underset{010}{X\overline{Y}Z} \vee \underset{100}{\overline{X}YZ} \vee \underset{111}{XYZ}$$

$$\overline{X}\overline{Y}Z \vee \overline{X}Y\overline{Z} \vee X\overline{Y}\overline{Z} \vee XYZ$$

Правило перехода от таблицы к формуле функции в СДНФ следующее: для перехода нужно записать дизъюнкцию полных конъюнкций по числу единиц в коде функции (в примере — 4 единицы), подписать под ними наборы, на которых функция равна единице, и поставить отрицания аргументов, соответствующих нулям в этих наборах.

Переход от схемы к формуле функции

Переход от логической схемы (ЛС), построенной из логических элементов (ЛЭ) к формуле булевой функции требует знания булевых функций, реализуемых ЛЭ. Начиная от входов ЛС, используя принцип суперпозиции (подстановки функций в качестве аргументов в другие функции), получаем на выходе функцию, реализуемую всей ЛС. Рассмотрим этот переход на примере ЛС, показанной на рис. 12.8.

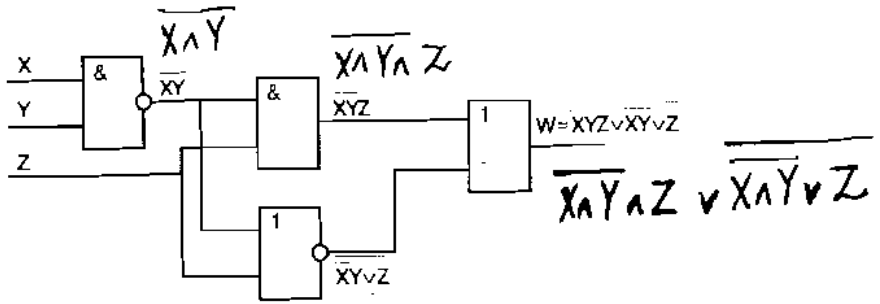

Рис. 12.8. *Переход от логической схемы к формуле функции*

Переход от формулы к СДНФ и таблице истинности

Такой переход можно сделать, выполнив следующие действия:

1. При наличии общих для нескольких переменных отрицаний, используя правило де Моргана, опускаем их на переменные. При наличии двойных отрицаний убираем их:

$$W = \overline{\overline{X}YZ \vee \overline{XY} \vee Z} = (\overline{\overline{X}} \vee \overline{Y})Z \vee \overline{\overline{XY}\overline{Z}} = (\overline{\overline{X}} \vee \overline{Y})Z \vee XY\overline{Z}$$

2. Раскрываем все скобки и добавляем недостающие переменные умножением членов полученной ДНФ на скобки, равные единице:

$$W = \overline{X}Z \vee \overline{Y}Z \vee XYZ = X(Y \vee \overline{Y})Z \vee (X \vee \overline{X})\overline{Y}Z \vee XY\overline{Z}$$

3. Раскрываем скобки и оставляем в формуле только по одной из повторяющихся конституент (используем правило $X \vee X = X$). В результате получим СДНФ функции:

$$W = \overline{X}YZ \vee \underline{\overline{X} \cdot \overline{Y}Z} \vee X\overline{Y}Z \vee \underline{\overline{X} \cdot \overline{Y}Z} \vee XY\overline{Z} =$$

$$= XYZ \vee \overline{X} \cdot \overline{Y}Z \vee X\overline{Y}Z \vee XYZ$$

4. Осталось определить номера наборов, на которых полученные конституенты единицы равны единице. Это соответственно наборы: 011, 001, 101 и 110 (наборы 3, 1, 5 и 6). Записываем в таблице истинности на этих наборах значение функции 1, на остальных наборах 0. Функция W(X,Y,Z) записана в табл. 12.9.

Переход от алгоритма работы к логической схеме

Такой переход выполняется в следующей последовательности:

Задача → алгоритм → таблица истинности → → формулы функций → логическая схема

Пример 1.

Построить логическую схему, суммирующую два одноразрядных двоичных числа A и B и вырабатывающую их сумму S и перенос P.

Алгоритм сложения можно задать описательно, в виде блок-схемы, другими способами. Ввиду простоты алгоритма, запишем его непосредственно таблицей истинности — табл. 12.10 работы логической схемы.

По рассмотренным выше правилам выписываем формулы суммы и переноса:

$$S = \overline{A}B \vee A\overline{B}\ ;\ P = AB.$$

Простейший анализ формул показывает, что для построения схемы требуется два инвертора (элемента НЕ), три конъюнктора (элемента И), и один дизъюнктор (элемент ИЛИ). Логическая схема показана на рис. 12.9.

Таблица 12.10

AB	S	P
0 0	0	0
0 1	1	0
1 0	1	0
1 1	0	1

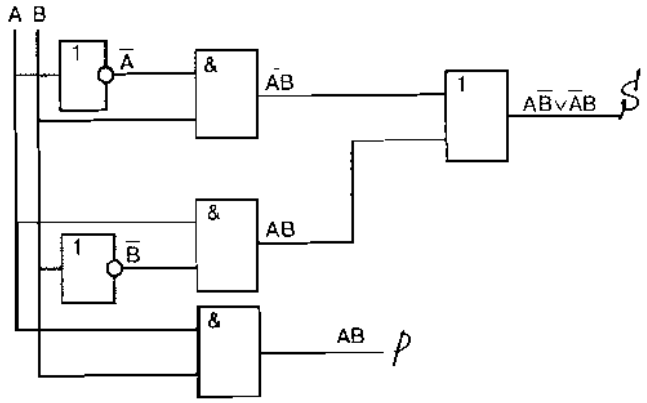

Рис. 12.9. *Логическая схема к примеру 1*

Обратим внимание на важный момент построения логической схемы. В качестве булевых переменных выступают цифровые разряды двоичного числа. Пример показывает применение булевой алгебры для построения всех логических схем компьютера, преобразующего информацию, представленную в двоичной системе счисления.

Пример 2.

Построить логическую схему одноразрядного двоичного сумматора на три входа, вырабатывающего значение суммы S и переноса в следующий разряд Q.

Обозначим входные сигналы: A и B — значения одноименных разрядов двух двоичных чисел, P перенос из предыдущего разряда.

Зная двоичную арифметику и используя модель работы такого сумматора (рис. 12.10.а), заполняем таблицу истинности 12.11. Условное обозначение сумматора показано на рис. 12.10.б.

Таблица 12.11

a	ABP	S	Q
0	000	0	0
1	001	1	0
2	010	1	0
3	011	0	1
4	100	1	0
5	101	0	1
6	110	0	1
7	111	1	1

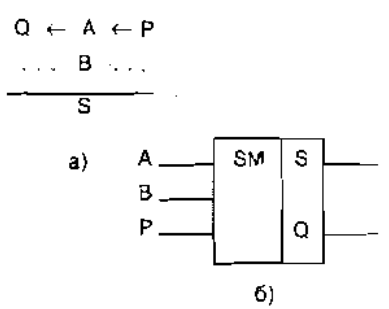

Рис. 12.10. *Модель (а) и обозначение сумматора (б)*

Из таблицы по рассмотренным в предыдущем примере правилам выписываем формулы функций S и Q в виде дизъюнкции конституент единицы. Как видно из таблицы, каждая формула будет содержать по четыре дизъюнктивных члена — по числу единиц в кодах функций:

$$S = \overline{A}\overline{B}P \vee \overline{A}B\overline{P} \vee A\overline{B}\overline{P} \vee ABP; \quad Q = \overline{A}BP \vee A\overline{B}P \vee AB\overline{P} \vee ABP.$$

Вторую формулу можно упростить. Упрощение формул уменьшает потребное для их реализации количество логических элементов. В упрощенном виде формула переносов запишется так: $Q = AB \vee AP \vee BP$. Убедиться в справедливости полученной формулы можно обратным переходом к СДНФ функции Q.

Решение задач минимизации сложности логических формул было актуально при построении схем компьютеров первого и второго поколения, когда стоимость каждого логического элемента была высока.

Постройте самостоятельно логические схемы, реализующие полученные формулы работы одноразрядного сумматора на три входа.

12.8. Применение логических операций при программировании

Результаты операций отношения (знаки операций := , <>, >,<, > = ,< =) принимают значение истина (TRUE), если условие выполняется и ложь (FALSE), если условие не выполняется. Отношения можно интерпретировать как простые высказывания, которые могут быть истинными или ложными.

С помощью логичееких операций AND, OR, NOT, XOR, EQV и IMP из простых отношений можно строить сложные, составные конструкции и использовать их в качестве сложных условий, например, в операторах IF.

Программа 12.1 печатает результаты операций отношения величин чисел a, b и c.

Программа 12.1

```
REM Значения результатов отношений (истина — –1, ложь — 0)
a = 5: b = 3: c = 1
f1 = a > b                                  'f1 = –1
f2 = a < b                                  'f2 = 0
f3 = a < b OR b > c                         'f3 = –1
f4 = a > b AND NOT (b > c)                  'f4 = 0
PRINT "f1 = "; f1, "f2 = "; f2, "f3 = "; f3, "f4 = "; f4
END
```

В качестве компонент отношений могут выступать арифметические и символьные выражения. В выражениях соблюдается следующая очередность выполнения операций: арифметические и символьные, отношения, логические. При этом очередность выполнения логических операций такая: NOT, AND, OR, XOR и EQV, IMP.

Рассмотрим пример использования сложного логического условия. Пусть требуется табулировать функцию, заданную графиком на рис. 12.11. Это можно сделать с помощью следующей программы 12.2.

Программа 12.2

```
REM Табуляция функции, заданной графиком
INPUT "x = ",x
IF x<-2 OR x>-1 AND x<1 OR x>2 THEN y = 0 ELSE y = 1
PRINT "При x = "; x , "y = "; y
END
```

Эту же функцию можно задать так:

```
IF x> = -2 AND x< = -1 OR x> = 1 AND x< = 2 THEN y = 1 ELSE y = 0.
```

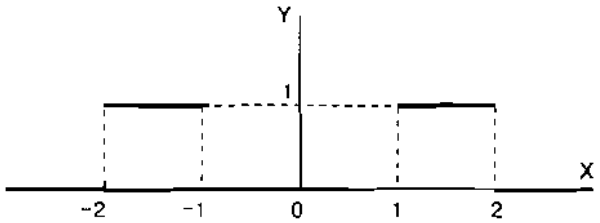

Рис. 12.11. *График функции*

12.9. Моделирование логических функций

Некоторые из рассмотренных выше задач связаны с преобразованиями формул булевых выражений, вычислением их значений на различных наборах значений аргументов. К таким задачам можно отнести доказательства справедливости основных законов и тождеств булевой алгебры, переход от произвольной формулы функции к таблице ее истинности, другие задачи. Заметим, что подобные тождественные преобразования даже несложных функций вызывают затруднения, требуют внимательности, связаны с возможными ошибками. В таких случаях следует привлекать компьютер.

Программа 12.2 строит и заполняет таблицу истинности. Она последовательно вырабатывает все наборы значений аргументов и вычисляет значения функции на этих наборах. Программа легко дополняется для генерации наборов 4, 5 и более аргументов. В рассмотренном примере программа работает с формулой функции из пункта «Переход от формулы к СДНФ и таблице истинности»: $W = \overline{XYZ} \vee \overline{\overline{X}\,\overline{Y}} \vee Z$.

Программа 12.3

```
REM Переход от формулы к таблице истинности
CLS : PRINT " x"; "  y"; "  z", " w": PRINT
FOR x = 0 TO 1
   FOR y = 0 TO 1
      FOR z = 0 TO 1
         w = NOT (x AND y) AND z OR NOT (NOT (x AND y) OR z)
         PRINT x; y; z, w
NEXT z, y, x
END
```

В результате работы программы будет выведена на экран монитора таблица истинности функции W(x, y, z) аналогичная табл. 12.9.

При моделировании логических функций возможно получение в коде функций значений −1 вместо 1 и −2 вместо 0. Это легко изменить перед печатью результатов.

Задания для самостоятельной работы

1. В таблице истинности заданы четыре булевы функции трех переменных: f_1, f_2, f_3 и f_4 (табл. 12.12). Для каждой из этих функций выполнить следующее:

Таблица 12.12

α	$x_1\ x_2\ x_3$	f_1	f_2	f_3	f_4
0	0 0 0	1	0	1	1
1	0 0 1	0	1	0	1
2	0 1 0	0	0	0	0
3	0 1 1	1	1	0	0
4	1 0 0	1	1	1	1
5	1 0 1	0	0	1	1
6	1 1 0	0	1	1	0
7	1 1 1	1	0	1	1

   а) перейти от таблицы истинности к записи функции в СДНФ (совершенной дизъюнктивной нормальной форме);

   б) используя законы и правила преобразования формул булевых функций, в частности операции склеивания и поглощения, упростить запись функции (уменьшить в формуле число вхождений переменных и их отрицаний);

   в) реализовать полученную запись функции логической схемой на элементах И, ИЛИ и НЕ.

2. Для заданных на рис. 12.12 логических схем, реализующих булевы функции трех аргументов, выполнить следующие действия:

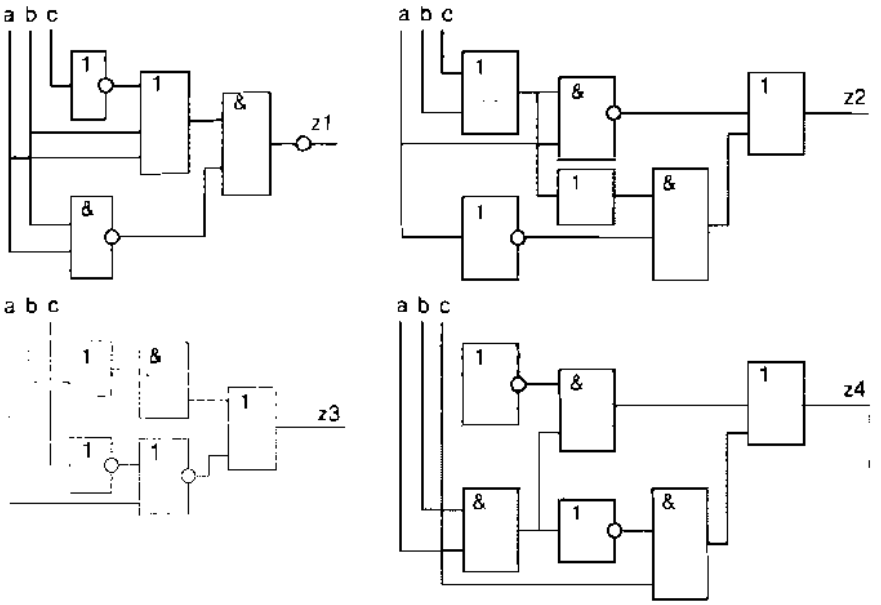

Рис. 12.12. *Логические схемы к заданию 2 самостоятельной работы*

а) записать формулы реализуемых логическими схемами функций;

б) перейти от произвольной формы записи формул функций к записи их в СДНФ;

в) перейти от СДНФ функций к записи их в таблице истинности;

г) подать на входы схемы набор значений двоичных переменных, например, 101 и проследить значения сигналов на выходах логических элементов и на выходах логических схем; сравните полученные значения со значениями функций на этом наборе в таблице истинности.

3. Как было показано, любую булеву функцию можно записать в СДНФ. Это говорит о функциональной полноте булевого базиса (конъюнкция, дизъюнкция и инверсия). Чтобы убедиться в функциональной полноте базисов алгебр Шеффера, Пирса и Жегалкина, достаточно с помощью функций, входящих в их базисы, реализовать функции булева базиса.

Например, инверсия в базисах алгебр Шеффера, Пирса и Жегалкина реализуется так: $\overline{x} = x \mid x = x \downarrow x = 1 \oplus x$. Реализуйте в этих базисах функции конъюнкцию и дизъюнкцию.