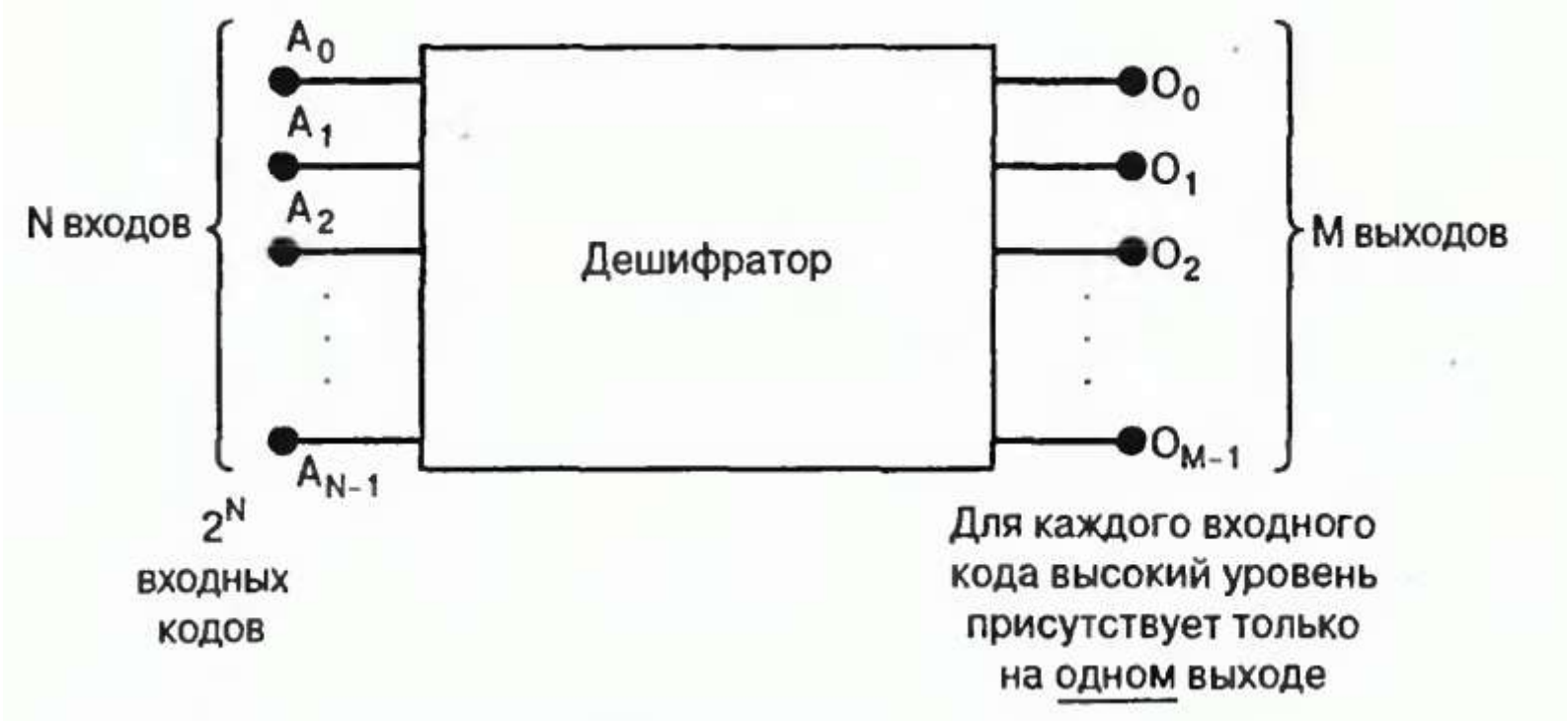
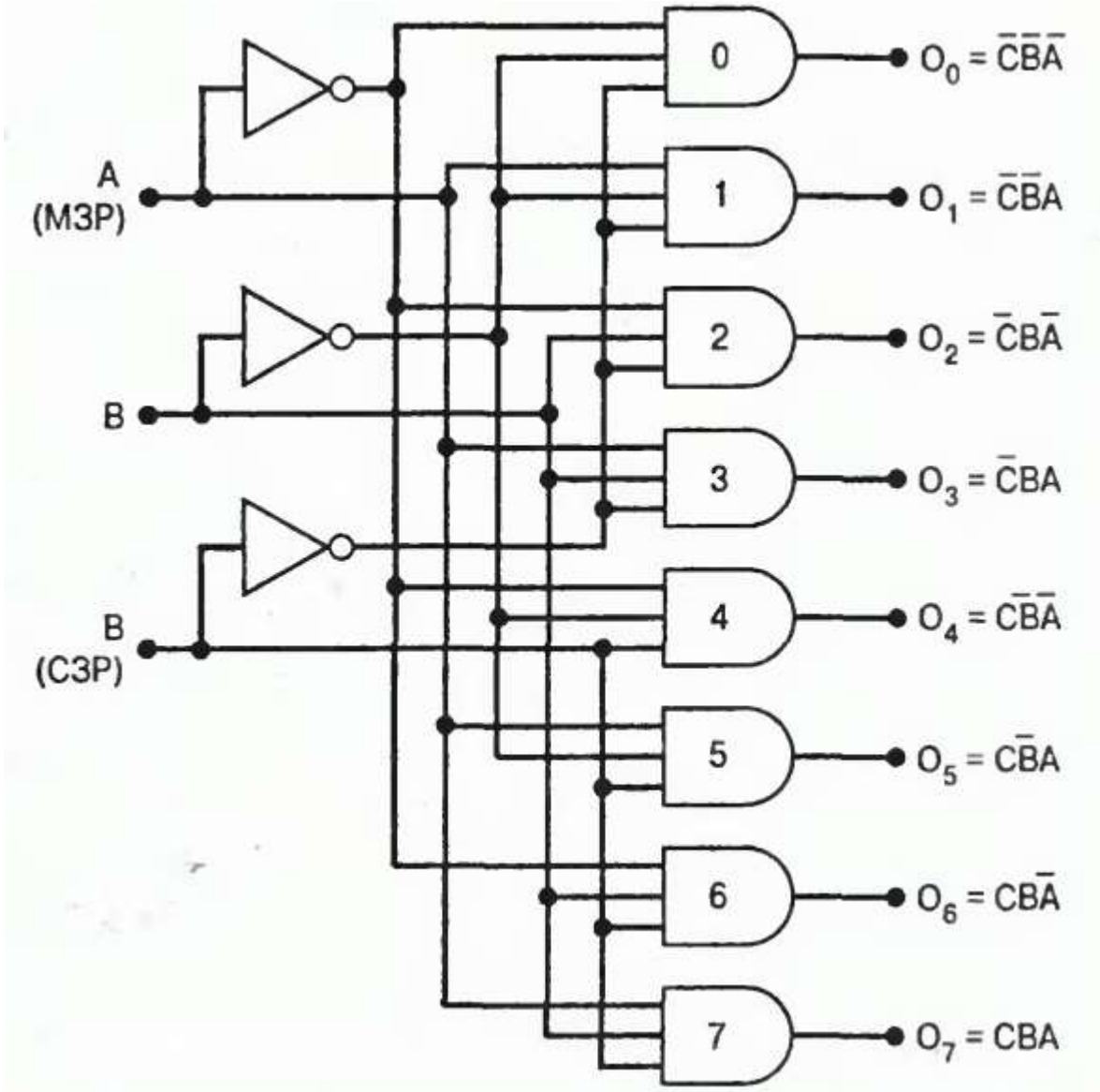
Дешифраторы



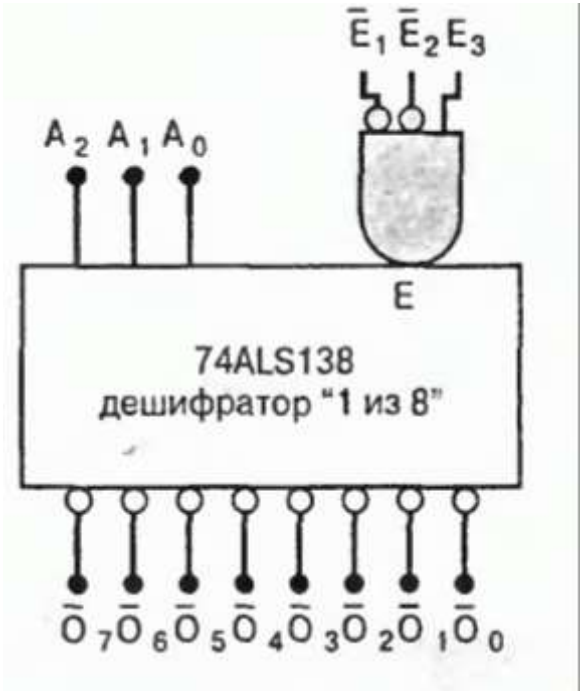
Каждый из N входов может находиться в нулевом или единичном состоянии, поэтому всего существует 2^N доступных входных комбинаций или кодов. Для каждой из этих комбинаций будет активизирован только один из M выходов, т.е. с него будет сниматься сигнал с *высоким* уровнем напряжения, на остальных же выходах будет наблюдаться *низкий* уровень сигнала. Многие дешифраторы спроектированы таким образом, что имеют выходы, для которых активным будет сигнал с *низким* уровнем напряжения, в то время как на остальных выходах будут сигналы с *высоким*. Такой дешифратор можно отличить по маленьким кружкам на выходах, обозначающим инверсию.

Некоторые дешифраторы используют не все 2^N доступных входных кода, а лишь некоторые из них. Например, дешифратор, который преобразует двоично-десятичный код в десятичный, имеет четырехбитовый вход и *десять* выходов, соответствующих *десяти* двоично-десятичным кодовым группам от 0000 до 1001. Дешифраторы такого плана часто разрабатываются с учетом того, что при поступлении неиспользуемой кодовой комбинации не будет активизирован *ни один* выход устройства*.

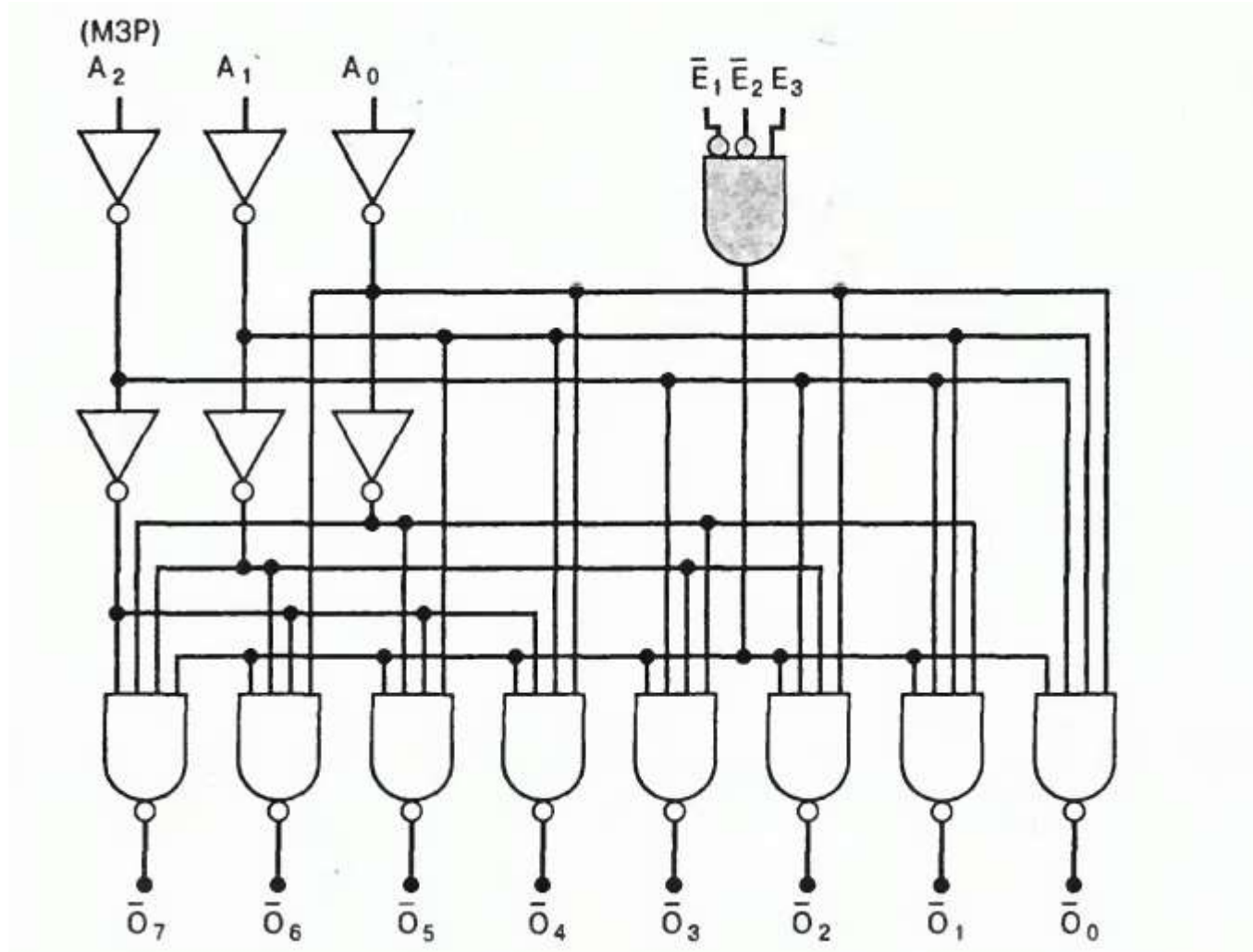
Некоторые дешифраторы имеют один или несколько разрешающих входов (в данной книге они обозначаются английским словом ENABLE), которые используются для управления работой дешифратора. Например, взглянув на рис. 9.2, на котором изображена схема дешифратора, можно представить себе общую сигнальную линию, по которой передается разрешающий сигнал. Эту линию можно было бы присоединить к четвертому входу каждого элемента И. Если на разрешающий вход ENABLE подать *высокий* уровень напряжения, то дешифратор будет функционировать так, как было описано в предыдущем абзаце, и комбинация битов со входов *A*, *B* и *C* определит, на каком выходе сигнал примет *высокий* уровень. Однако при подаче на разрешающий вход *низкого* уровня сигнала *все* выходы будут принудительно сброшены в состояния с *низкими* уровнями напряжения независимо от того, какие кодо-

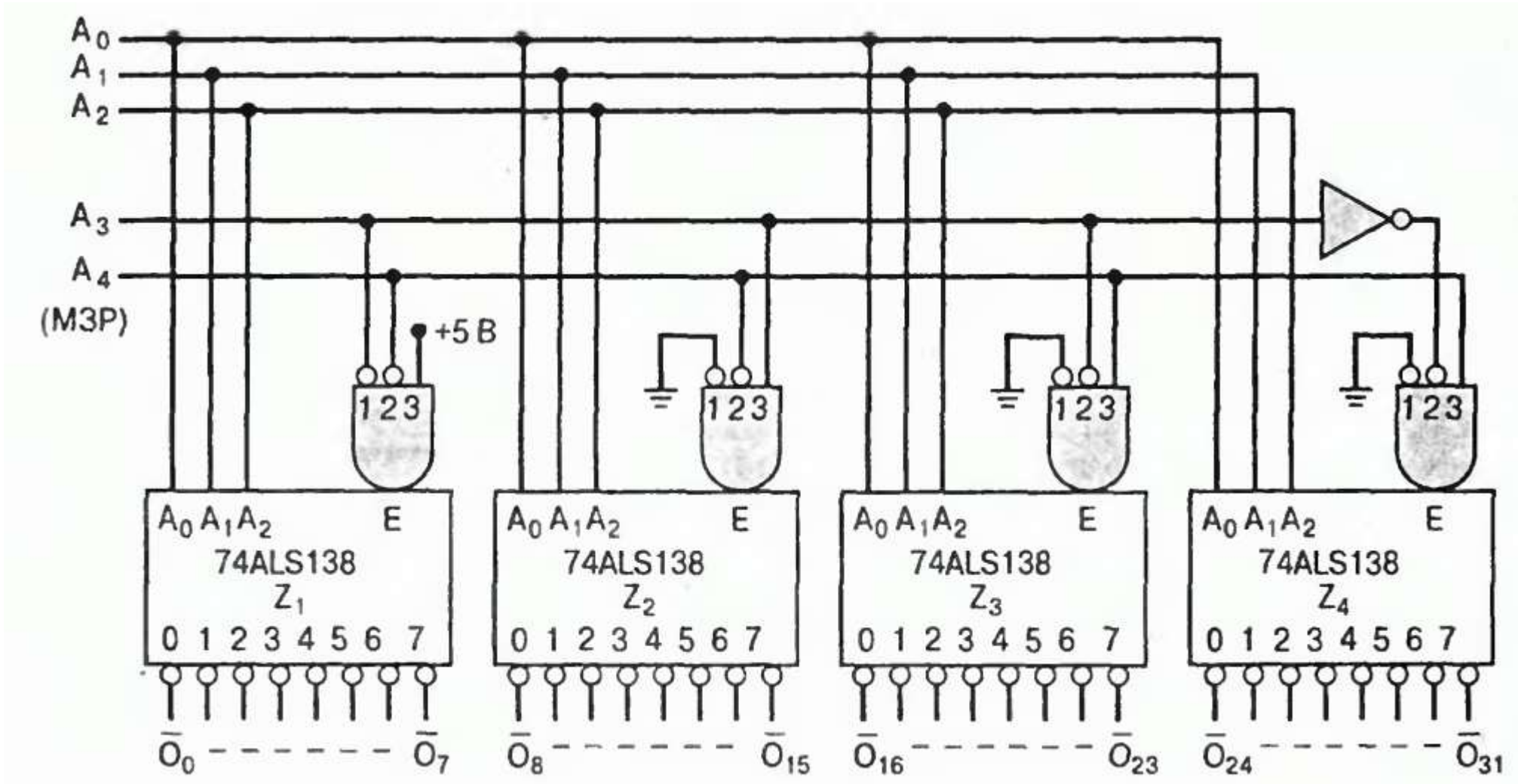


C	B	A	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

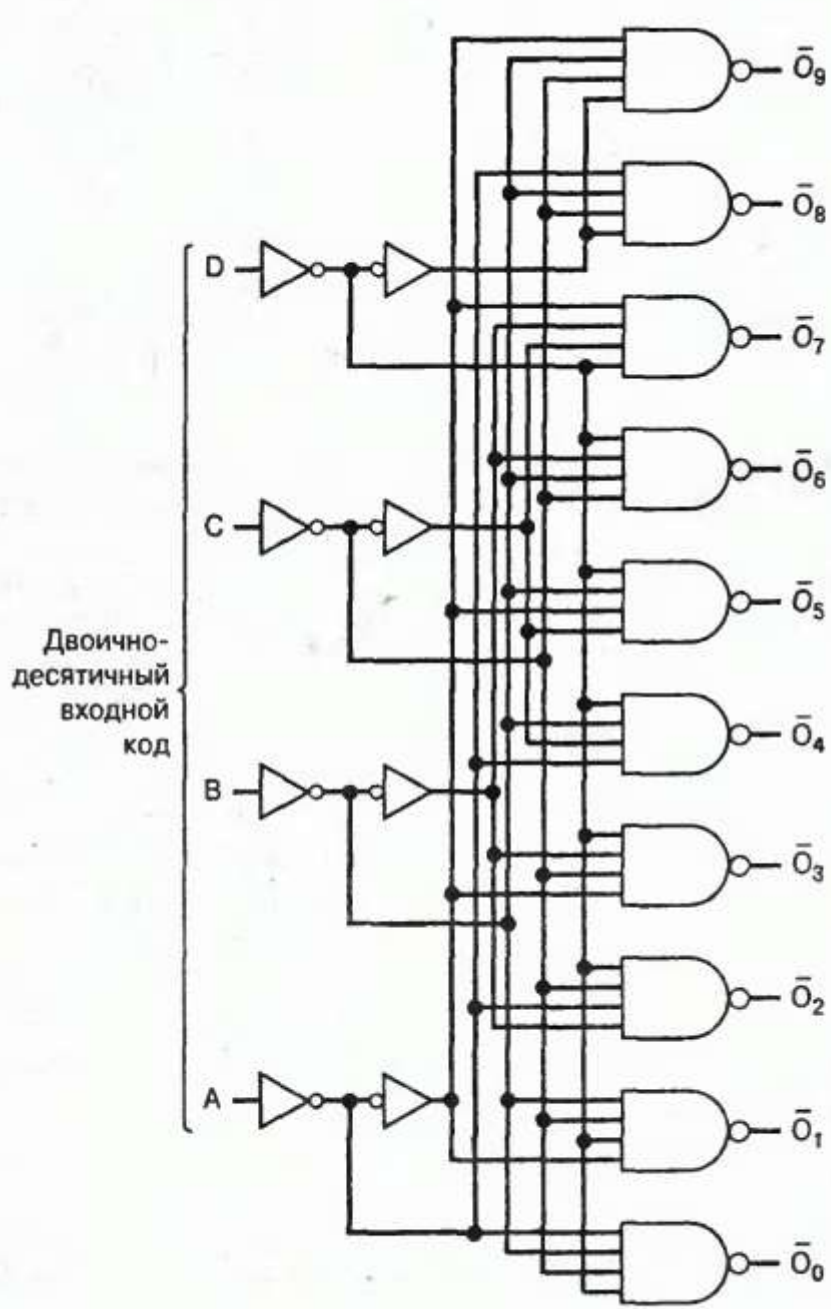


$\bar{E}_1$	$\bar{E}_2$	E_3	Выходы
0	0	1	Реакция на входной код $A_2A_1A_0$
1	X	X	Дешифрации нет — на всех выходах высокий уровень
X	1	X	Дешифрации нет — на всех выходах высокий уровень
X	X	0	Дешифрации нет — на всех выходах высокий уровень

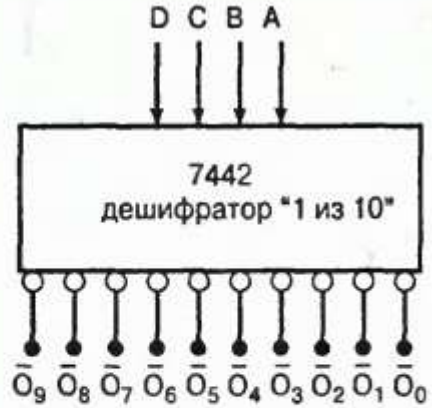



A0
A1
A2
A3
A4
(M3P)
+5 B
1 2 3
A0 A1 A2
E
74ALS138
Z1
0 1 2 3 4 5 6 7
Z2
Z3
Z4
O0 - - - - - - - O7
O8 - - - - - - - O15
O16 - - - - - - - O23
O24 - - - - - - - O31

Преобразование двоично-десятичного кода в двоичный



а)

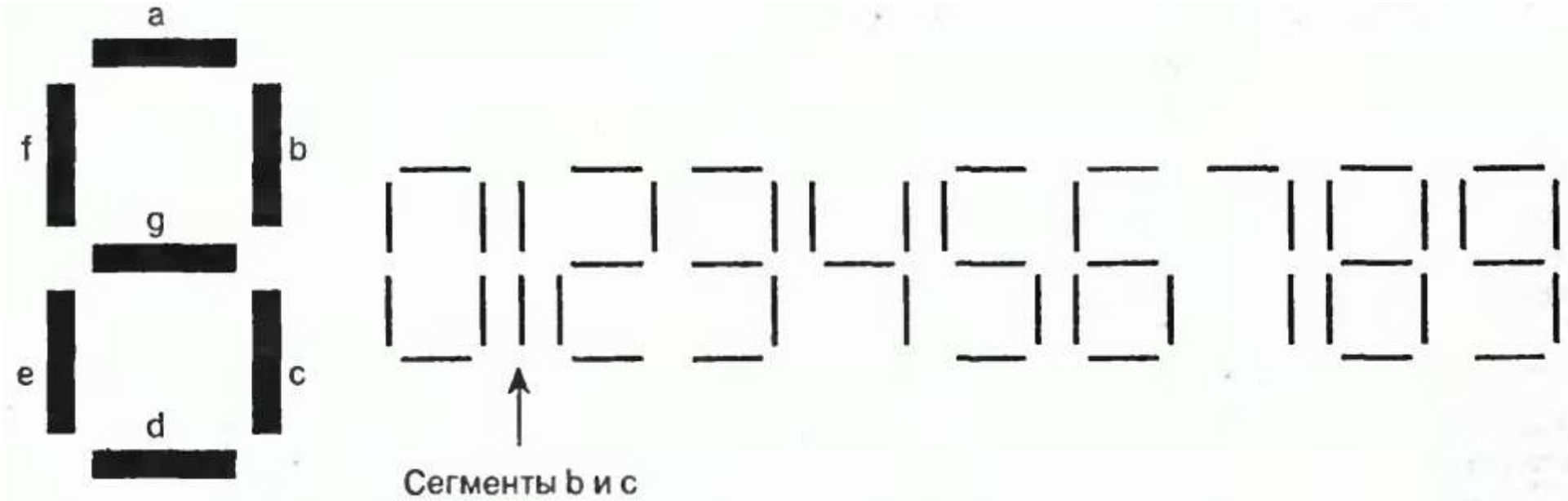


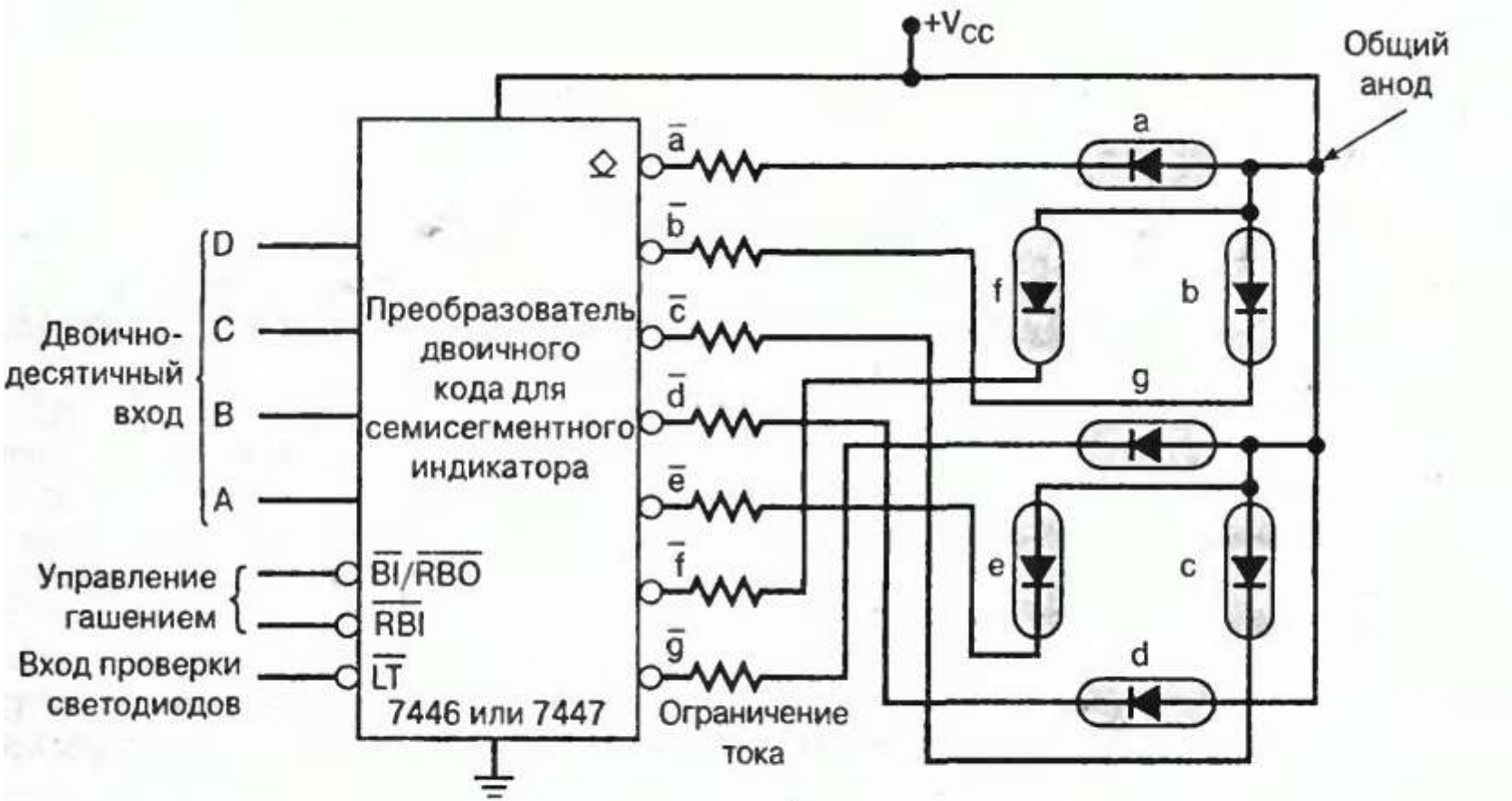
б)

Входы

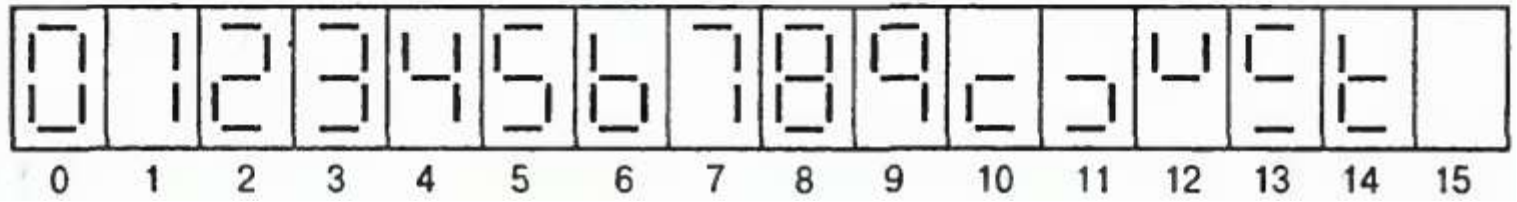
D	C	B	A	Активный выход
Н	Н	Н	Н	$\bar{O}_0$
Н	Н	Н	В	$\bar{O}_1$
Н	Н	В	Н	$\bar{O}_2$
Н	Н	В	В	$\bar{O}_3$
Н	В	Н	Н	$\bar{O}_4$
Н	В	Н	В	$\bar{O}_5$
Н	В	В	Н	$\bar{O}_6$
Н	В	В	В	$\bar{O}_7$
В	Н	Н	Н	$\bar{O}_8$
В	Н	Н	В	$\bar{O}_9$
В	Н	В	Н	Нет
В	Н	В	В	Нет
В	В	Н	Н	Нет
В	В	Н	В	Нет
В	В	В	Н	Нет
В	В	В	В	Нет

В — высокий уровень напряжения
Н — низкий уровень напряжения

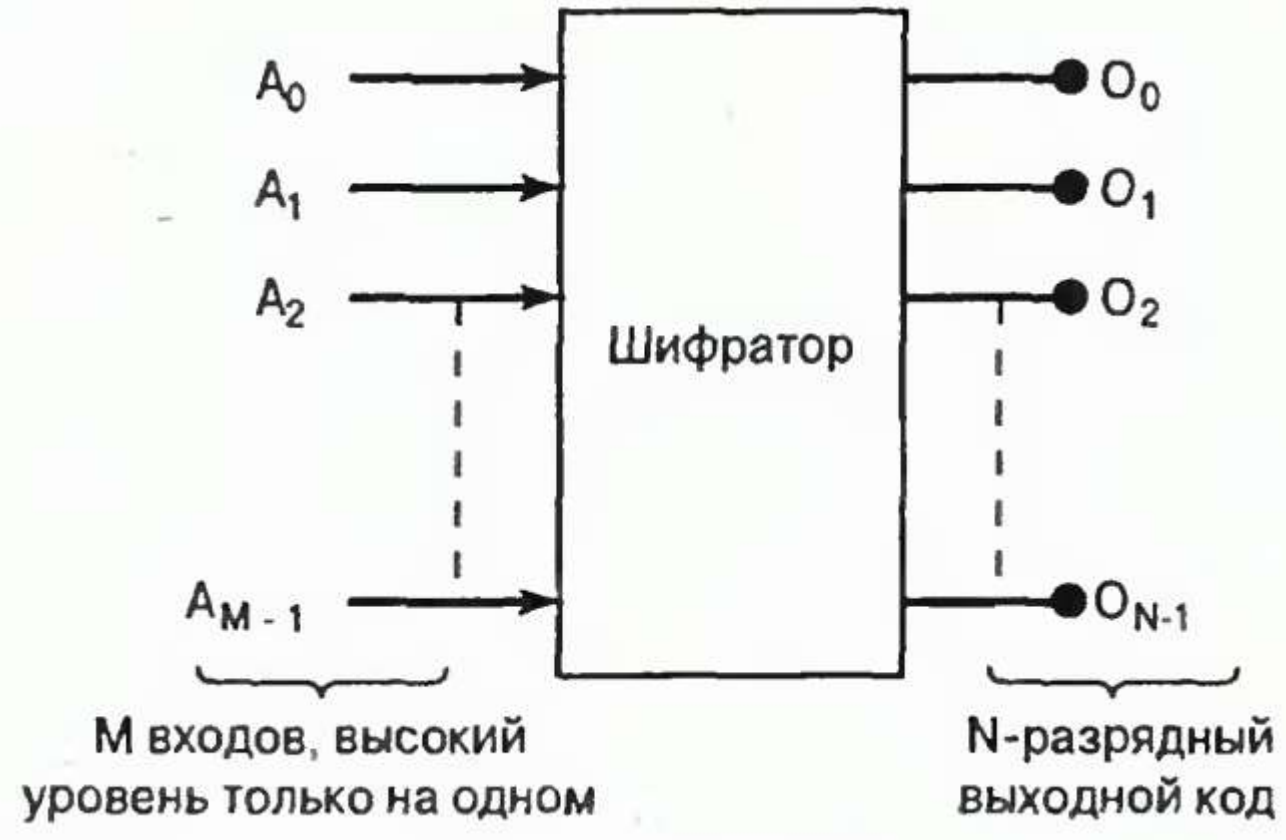

a
f
b
g
e
c
d
Сегменты b и c

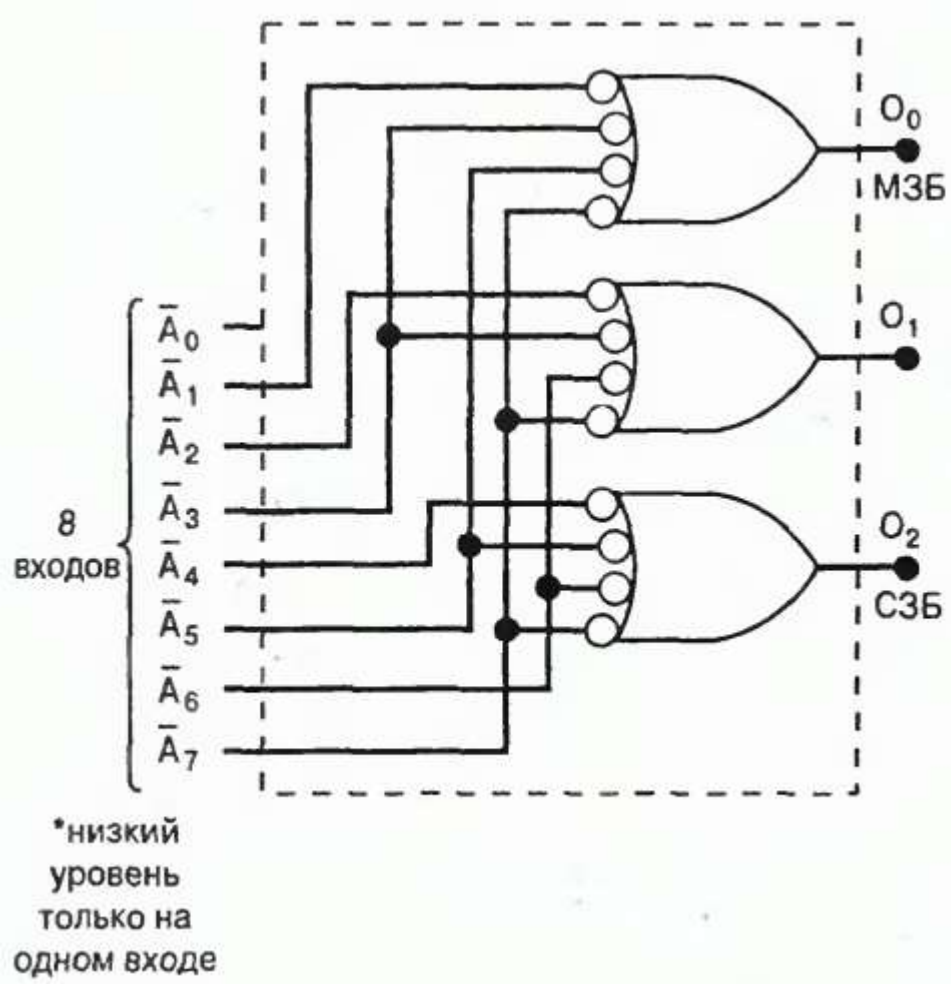

+Vcc
Общий анод
a
b
c
d
e
f
g
Двоично-десятичный вход
D
C
B
A
Преобразователь двоичного кода для семисегментного индикатора
Управление гашением
BI/RBO
RBI
Вход проверки светодиодов
LT
7446 или 7447
Ограничение тока

а)


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Шифраторы


A0
A1
A2
AM - 1
Шифратор
O0
O1
O2
ON-1
M входов, высокий уровень только на одном
N-разрядный выходной код

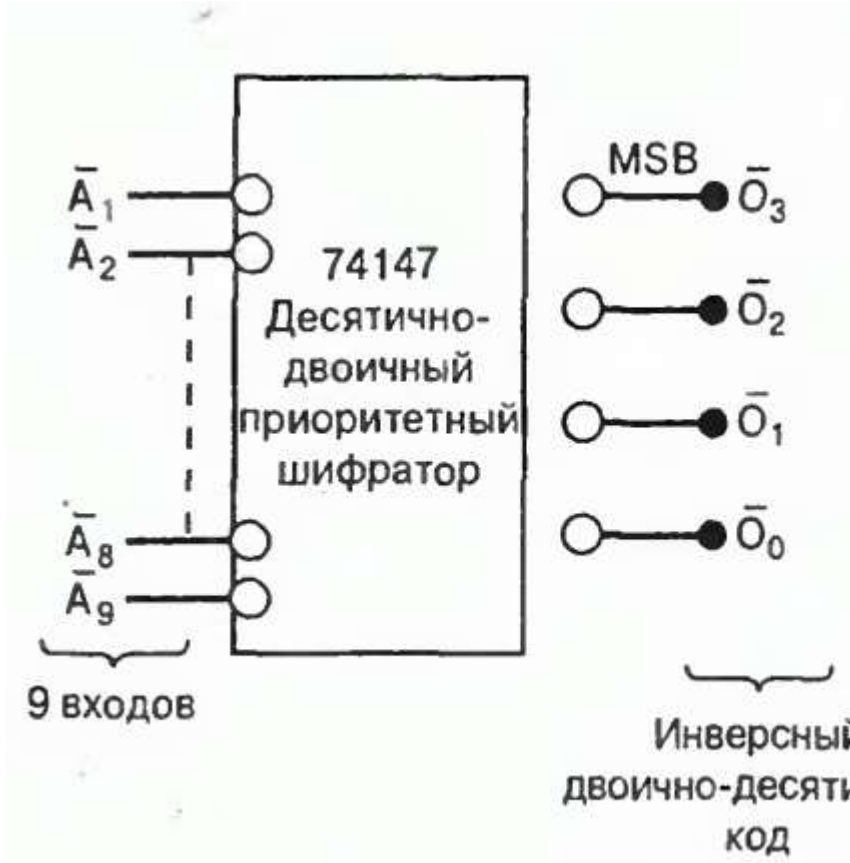


Входы								Выходы		
$\bar{A}_0$	$\bar{A}_1$	$\bar{A}_2$	$\bar{A}_3$	$\bar{A}_4$	$\bar{A}_5$	$\bar{A}_6$	$\bar{A}_7$	O_2	O_1	O_0
X	1	1	1	1	1	1	1	0	0	0
X	0	1	1	1	1	1	1	0	0	1
X	1	0	1	1	1	1	1	0	1	0
X	1	1	0	1	1	1	1	0	1	1
X	1	1	1	0	1	1	1	1	0	0
X	1	1	1	1	0	1	1	1	0	1
X	1	1	1	1	1	0	1	1	1	0
X	1	1	1	1	1	1	0	1	1	1

Рис. 9.13. Логическая схема шифратора, выполняющего преобразование из восьмеричного кода в двоичный (с восемью входами и тремя выходами). Для правильной работы в каждый момент времени должен быть активизирован только один вход

Что произойдет, если низкий уровень будет одновременно установлен на двух входах?

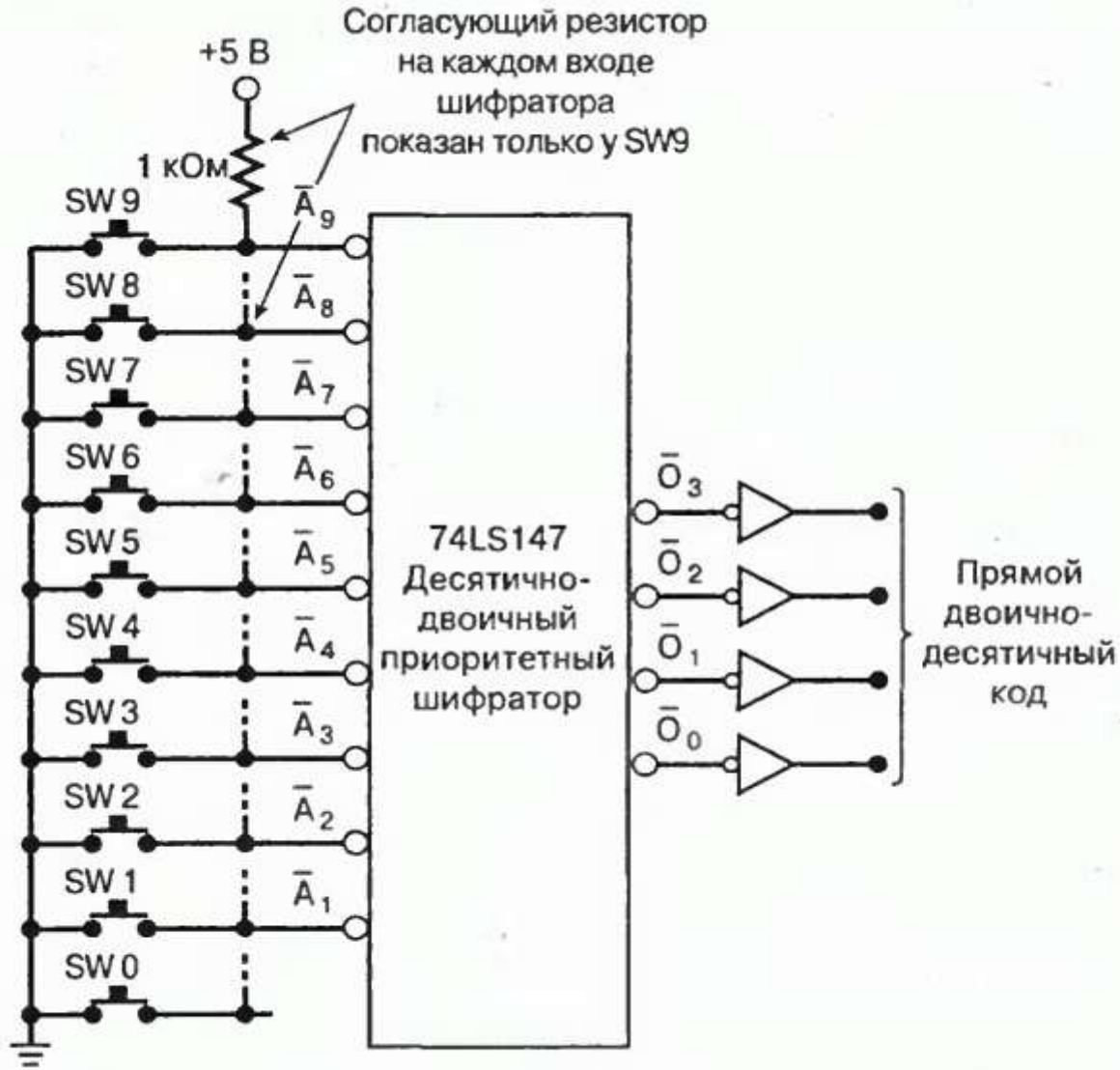
Приоритетные шифраторы



$\bar{A}_1$	$\bar{A}_2$	$\bar{A}_3$	$\bar{A}_4$	$\bar{A}_5$	$\bar{A}_6$	$\bar{A}_7$	$\bar{A}_8$	$\bar{A}_9$	$\bar{O}_3$	$\bar{O}_2$	$\bar{O}_1$	$\bar{O}_0$
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

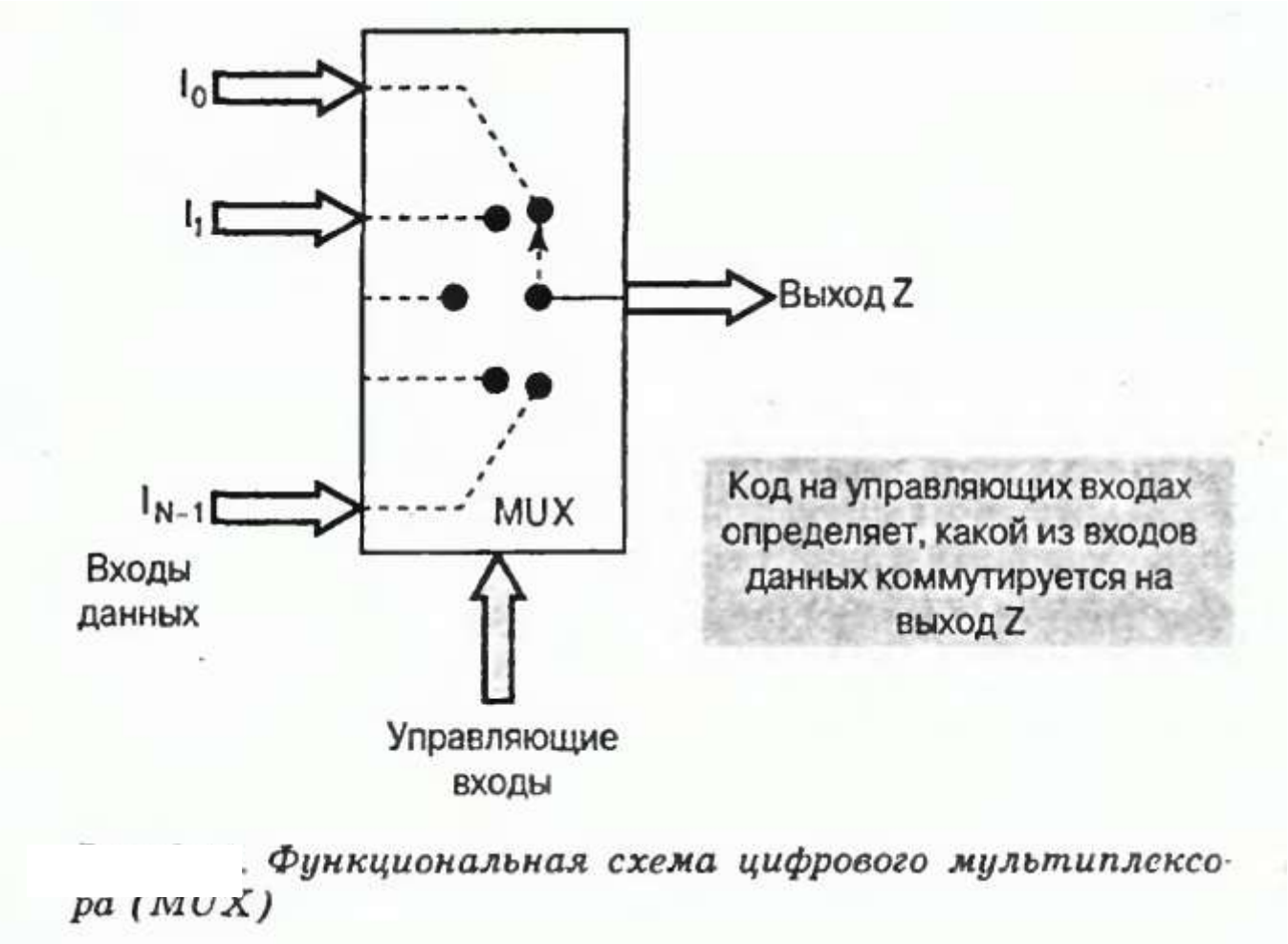
X — 0 или 1

Шифратор выдает инверсный двоично-десятичный код



Ключевой шифратор, преобразующий десятичный код в двоично-десятичный

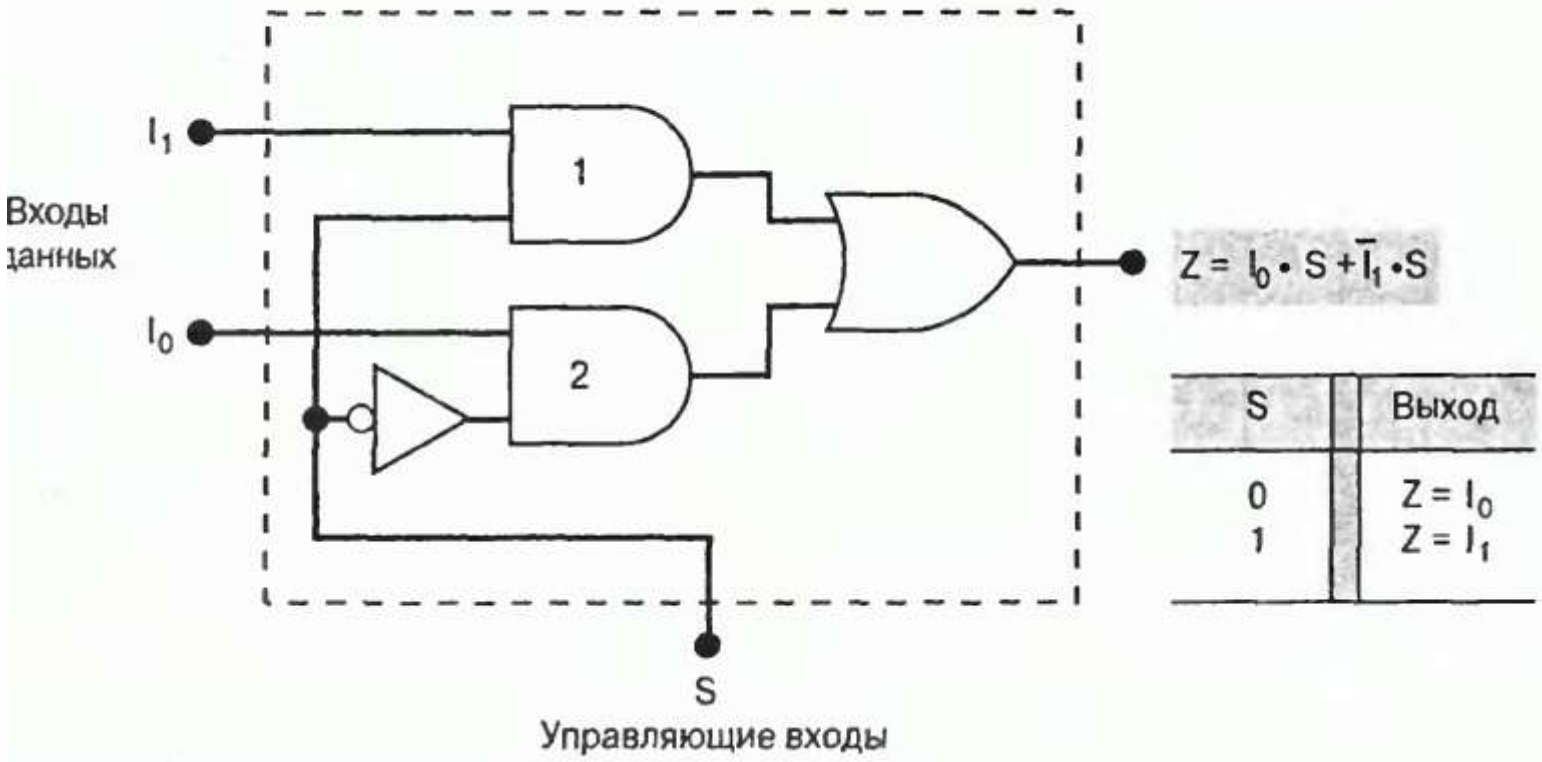
Мультиплексоры



Функциональная схема цифрового мультиплексора (MUX)

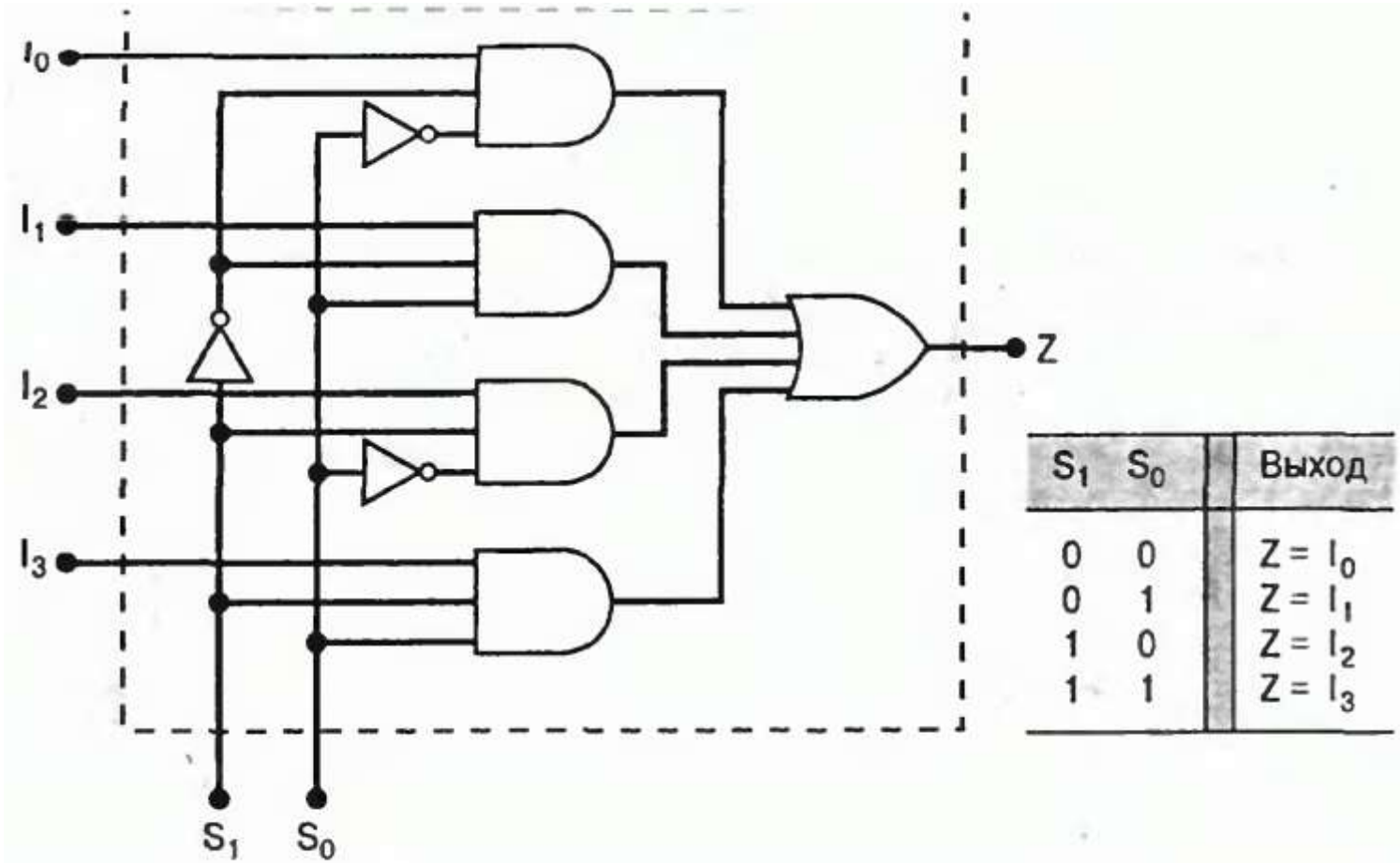
Цифровой мультиплексор, или *селектор данных*, представляет собой логическую схему, которая принимает несколько оцифрованных сигналов и выбирает один из них и передает на выход. Передача требуемого сигнала на выход контролируется входами выбора данных (SELECT), которые иногда еще называют входами выбора адреса (ADDRESS).

$$Z = I_0\bar{S} + I_1 S.$$



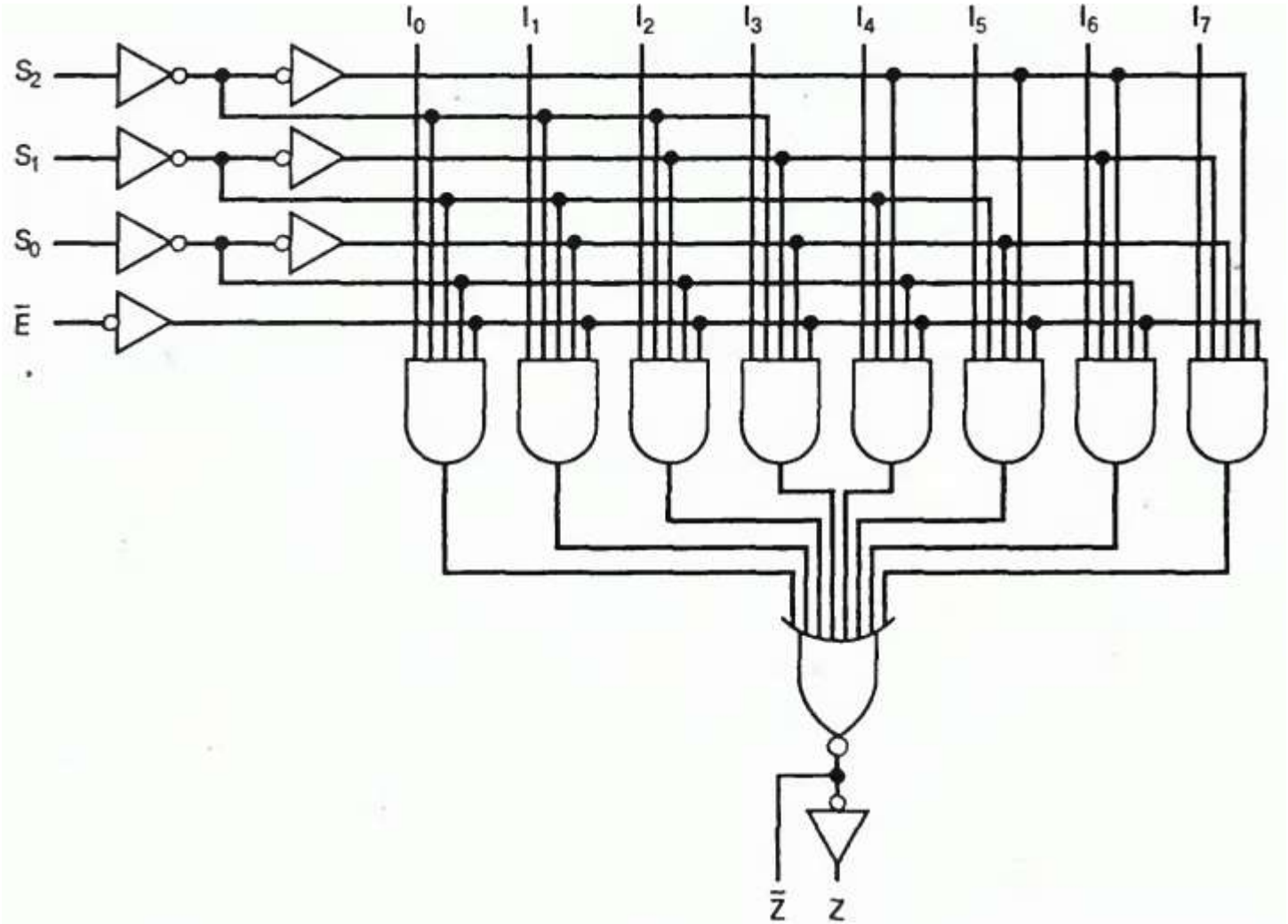
S	Выход
0	$Z = I_0$
1	$Z = I_1$

Мультиплексор с двумя входами данных

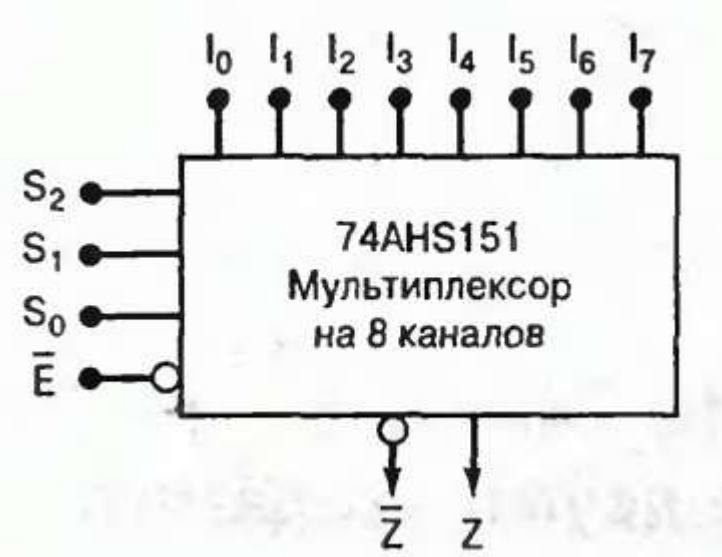


S_1	S_0	Выход
0	0	$Z = I_0$
0	1	$Z = I_1$
1	0	$Z = I_2$
1	1	$Z = I_3$

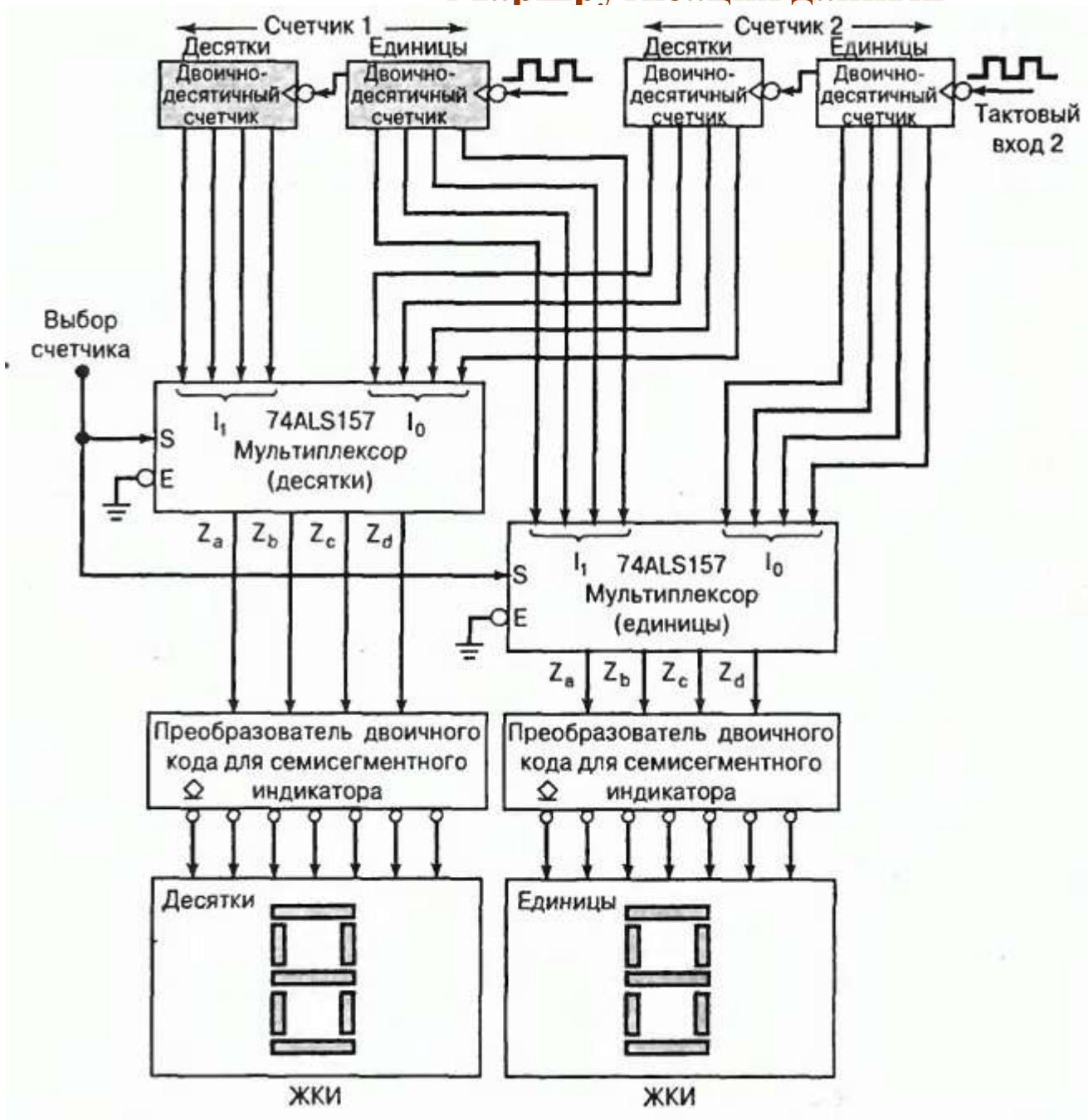
Мультиплексор с четырьмя входами данных



Входы				Выходы	
$\bar{E}$	S_2	S_1	S_0	$\bar{Z}$	Z
B	X	X	X	B	H
H	H	H	H	$\bar{I}_0$	I_0
H	H	H	B	$\bar{I}_1$	I_1
H	H	B	H	$\bar{I}_2$	I_2
H	H	B	B	$\bar{I}_3$	I_3
H	B	H	H	$\bar{I}_4$	I_4
H	B	H	B	$\bar{I}_5$	I_5
H	B	B	H	$\bar{I}_6$	I_6
H	B	B	B	$\bar{I}_7$	I_7

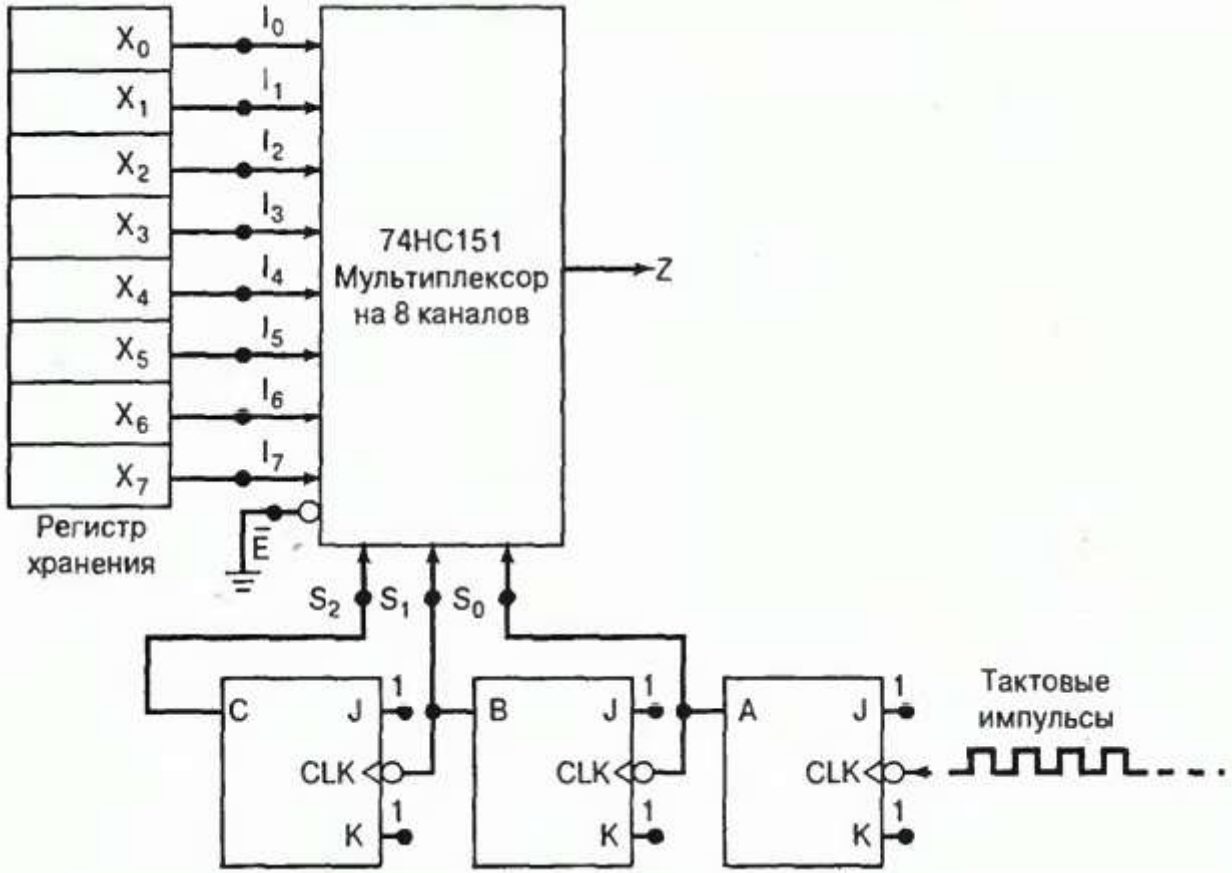


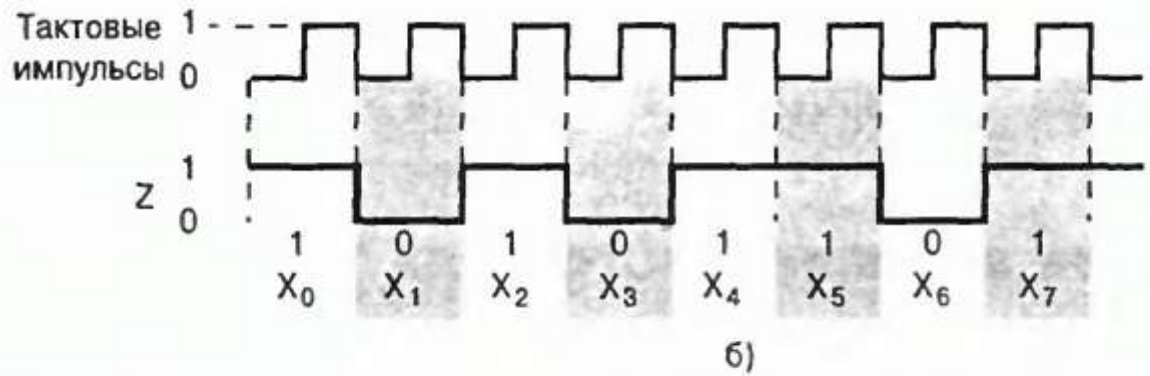
Маршрутизация данных



Преобразование параллельного кода в последовательный

Многие цифровые системы обрабатывают двоичные данные в параллельной форме (все биты одновременно), поскольку такой метод быстрее последовательной обработки. Однако если данные необходимо передавать на относительно большие расстояния, параллельная передача, требующая слишком большого количества линий передач, уже не может удовлетворить запросы. Именно поэтому перед тем, как передать двоичные данные на большие расстояния, их часто приходится преобразовывать в последовательный вид. Один из методов, с помощью которого можно выполнить **преобразование данных из параллельного вида в последовательный**, заключается в применении мультиплексора,



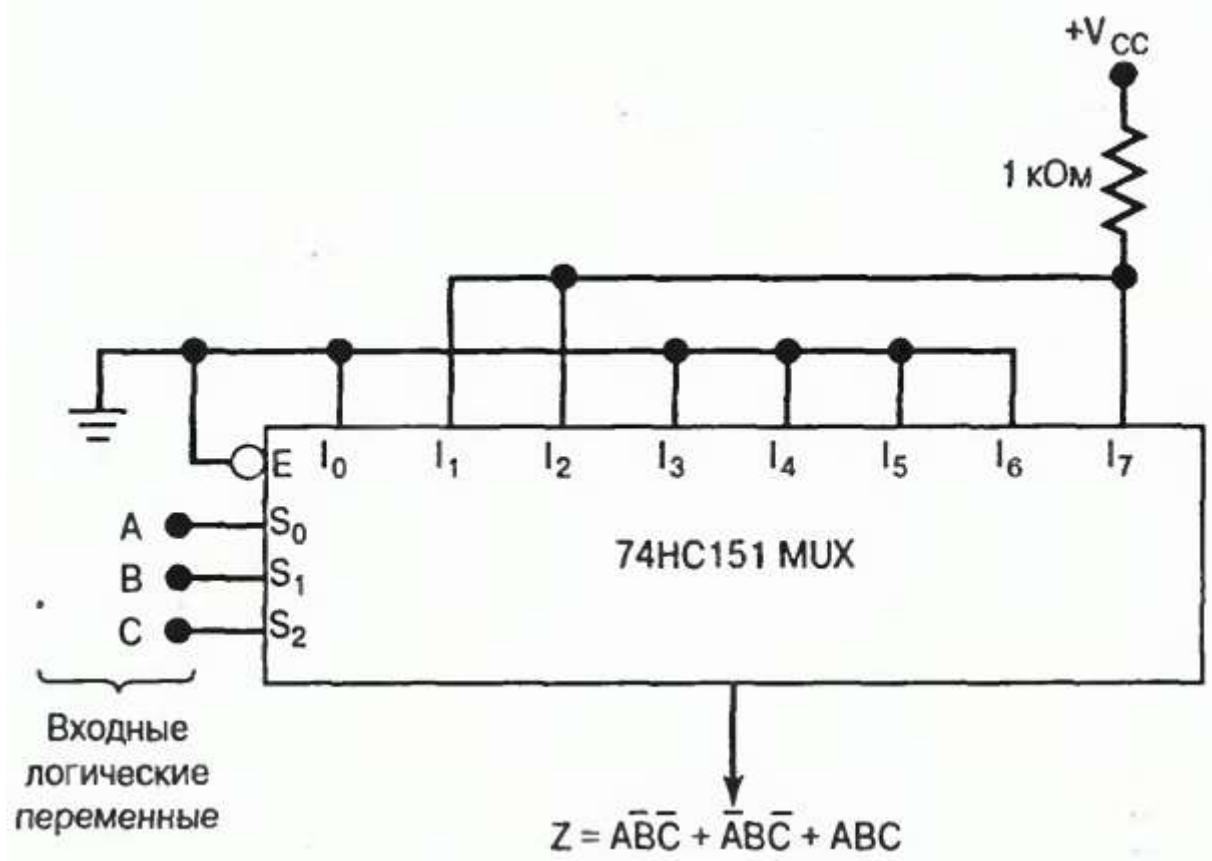


б)

а) Преобразователь из параллельного кода в последовательный; б) формы сигналов при $X_7X_6X_5X_4X_3X_2X_1X_0 = 10110101$

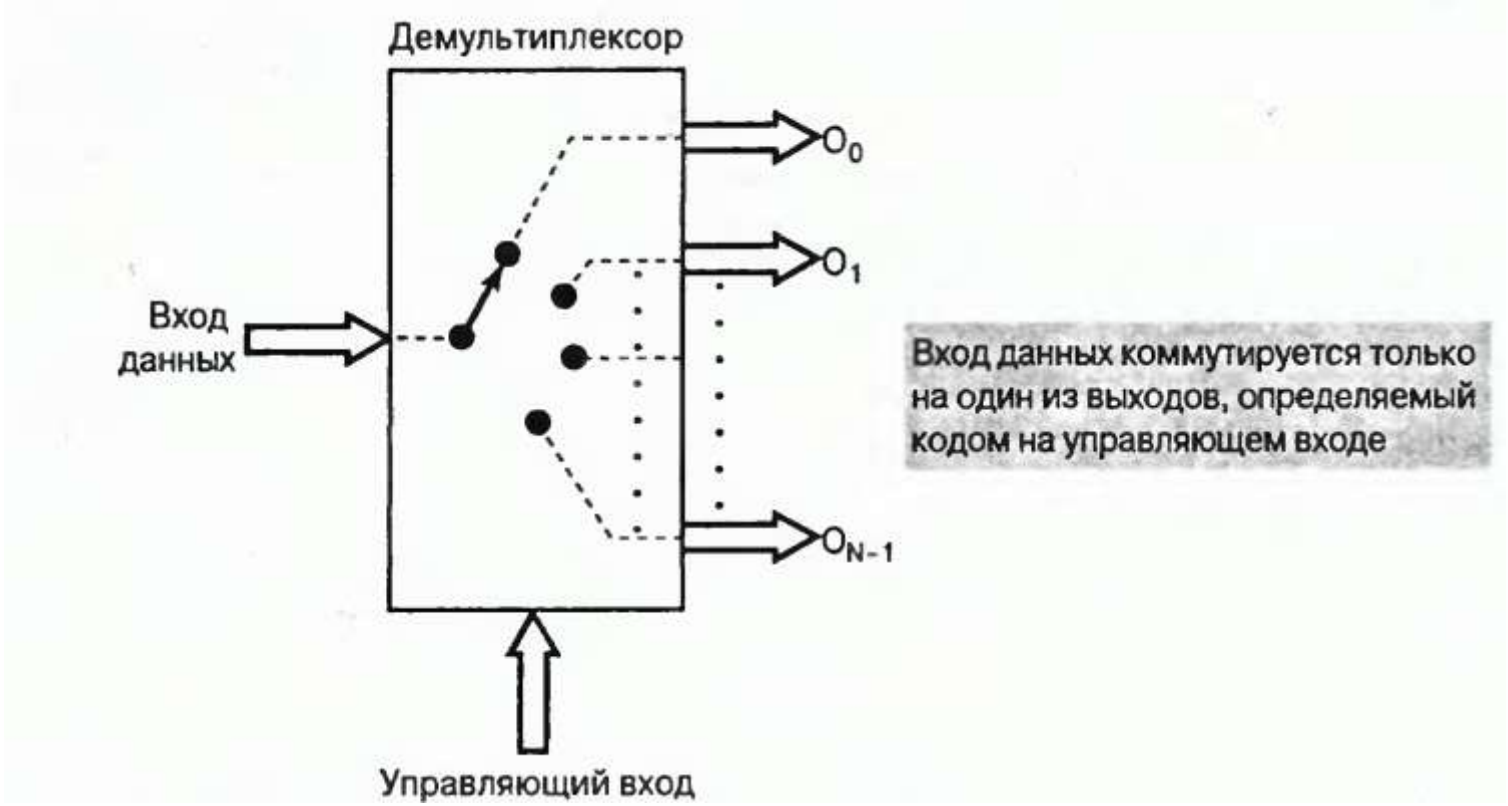
Формирование логической функции

Мультиплексоры могут использоваться для реализации логических функций непосредственно по таблице истинности, при этом необходимость упрощать выражения отпадает. Когда мультиплексор применяется для этих целей, входы выбора используются в качестве логических переменных, а на каждый вход данных подается постоянный сигнал с *высоким* или *низким* уровнем, в зависимости от данных таблицы истинности.



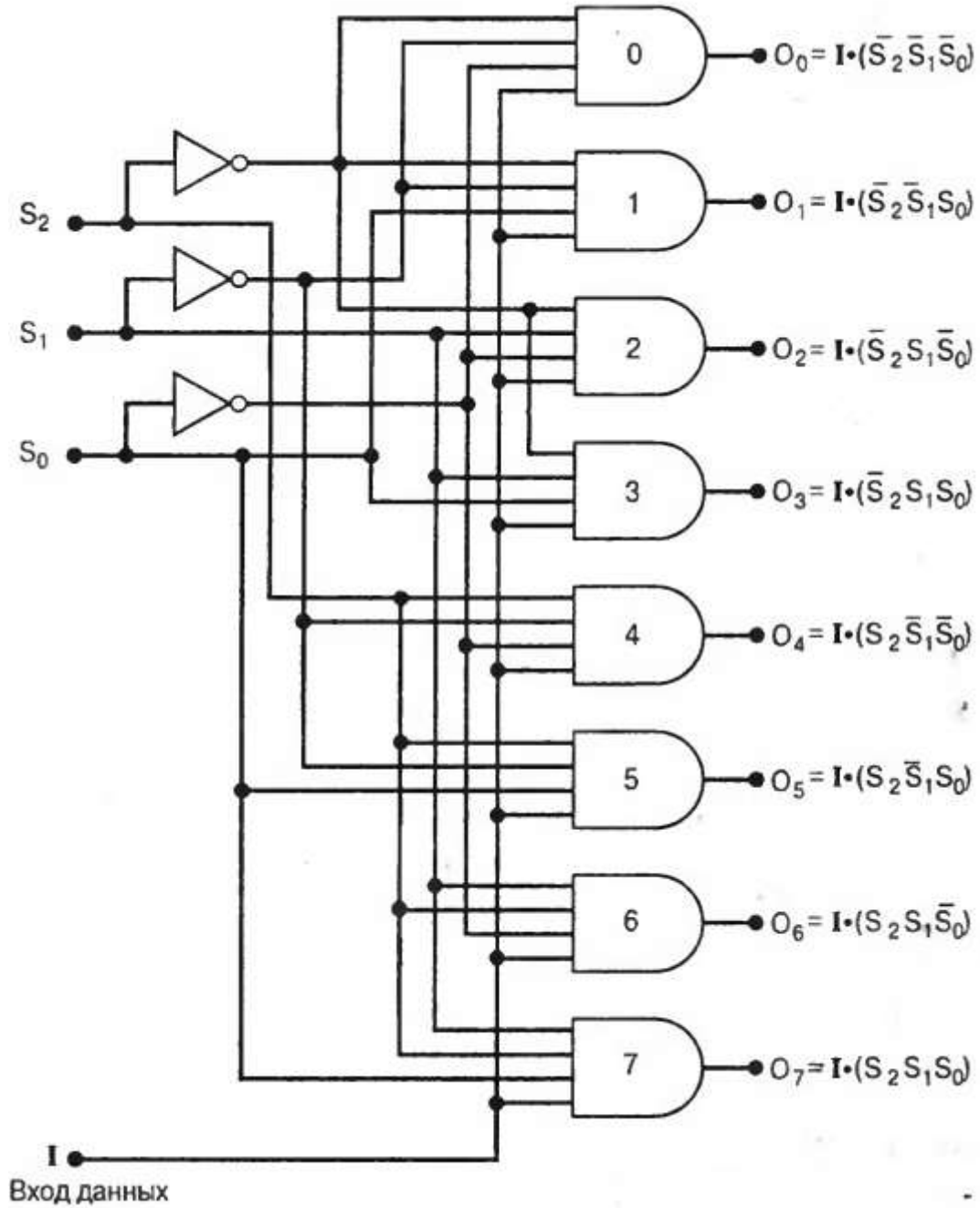
C	B	A	Z
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Демультиплексоры



Вход данных коммутируется только на один из выходов, определяемый кодом на управляющем входе

Мультиплексор принимает несколько входных сигналов и передает *один* из них на выход. **Демультиплексор (DEMUX)** осуществляет обратную операцию: он берет один входной сигнал и распределяет его на несколько выходов. На рис. изображена функциональная схема цифрового демультиплексора. Входы и выходы показаны толстыми стрелками, так как они могут представлять собой по несколько сигнальных линий сразу. Код, который поступает на вход выбора, определяет, на какой выход поступит сигнал со входа данных. Другими словами, на демультиплексор поступает один сигнал от источника данных, который проходит на 1 из N выходов, т.е. мы имеем дело с многопозиционным переключателем.

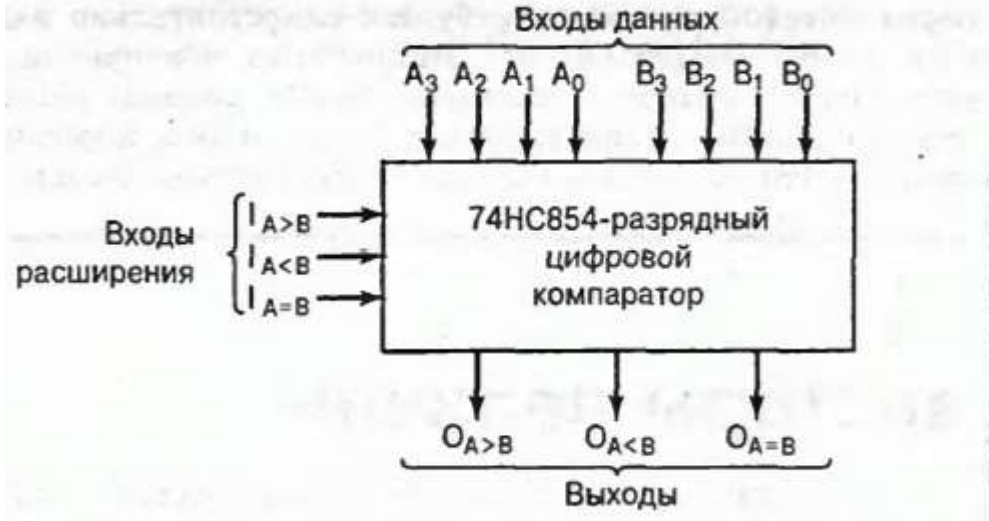


Управляющий код			Выходы							
S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

Схема демультиплексора, показанная на рис. очень напоминает дешифратор с 3 входами и 8 выходами (Слайд 7), с той лишь разницей, что к каждому элементу был добавлен дополнительный четвертый вход (*I*). Многие дешифраторы на основе ИС имеют разрешающий вход, который подведен к элементам дешифратора. Такой тип микросхем может использоваться в качестве демультиплексора. Входы, на которые поступают кодовые группы (т.е. биты *A*, *B* и *C* на Слайд 7), служат входами выбора, а разрешающий вход ENABLE представляет вход данных *I*. По этой причине производители интегральных схем часто называют подобные микросхемы *дешифраторами/демультиплексорами* и акцентируют внимание на том, что одна ИС может использоваться в качестве любого из этих устройств.

Компараторы

Еще одним полезным членом семейства интегральных схем средней степени интеграции является *компаратор величин*. Компаратором величин называется комбинационная логическая схема, сравнивающая две входных двоичных величины и формирующая сигналы, которые показывают, какая из величин имеет большее значение.



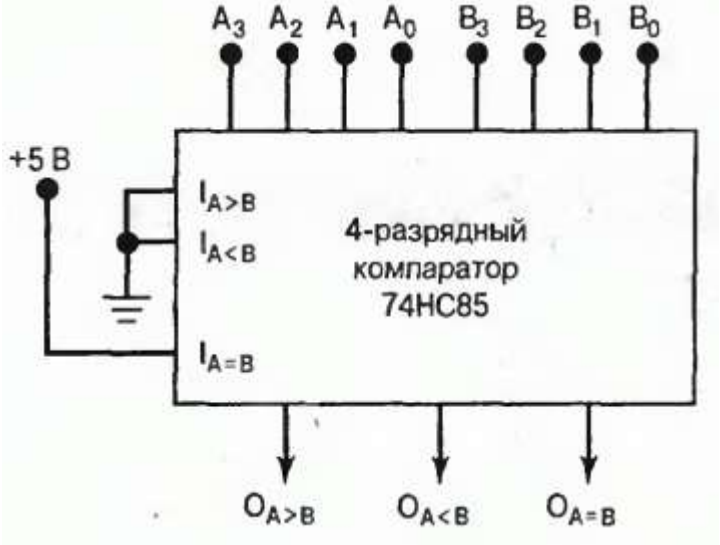
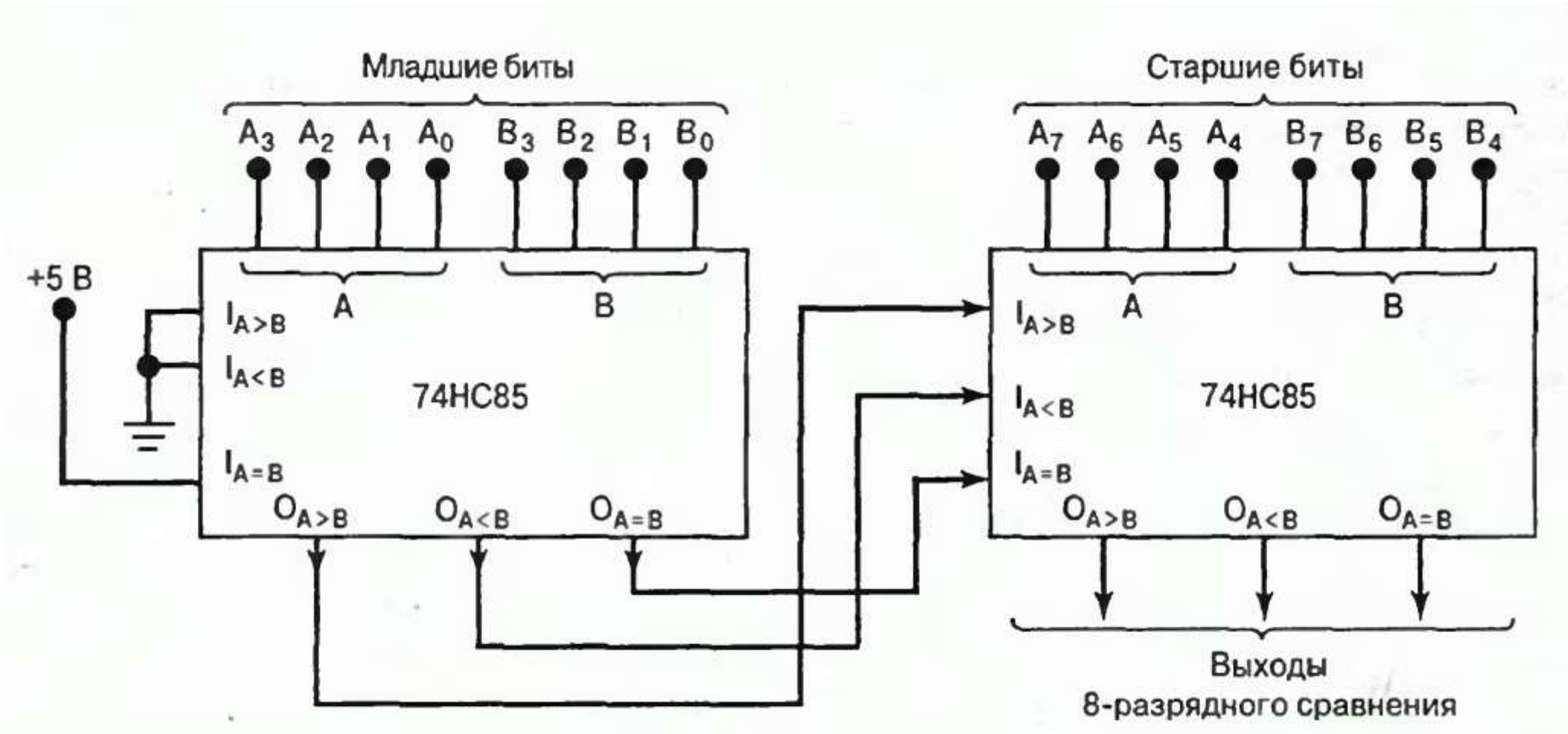


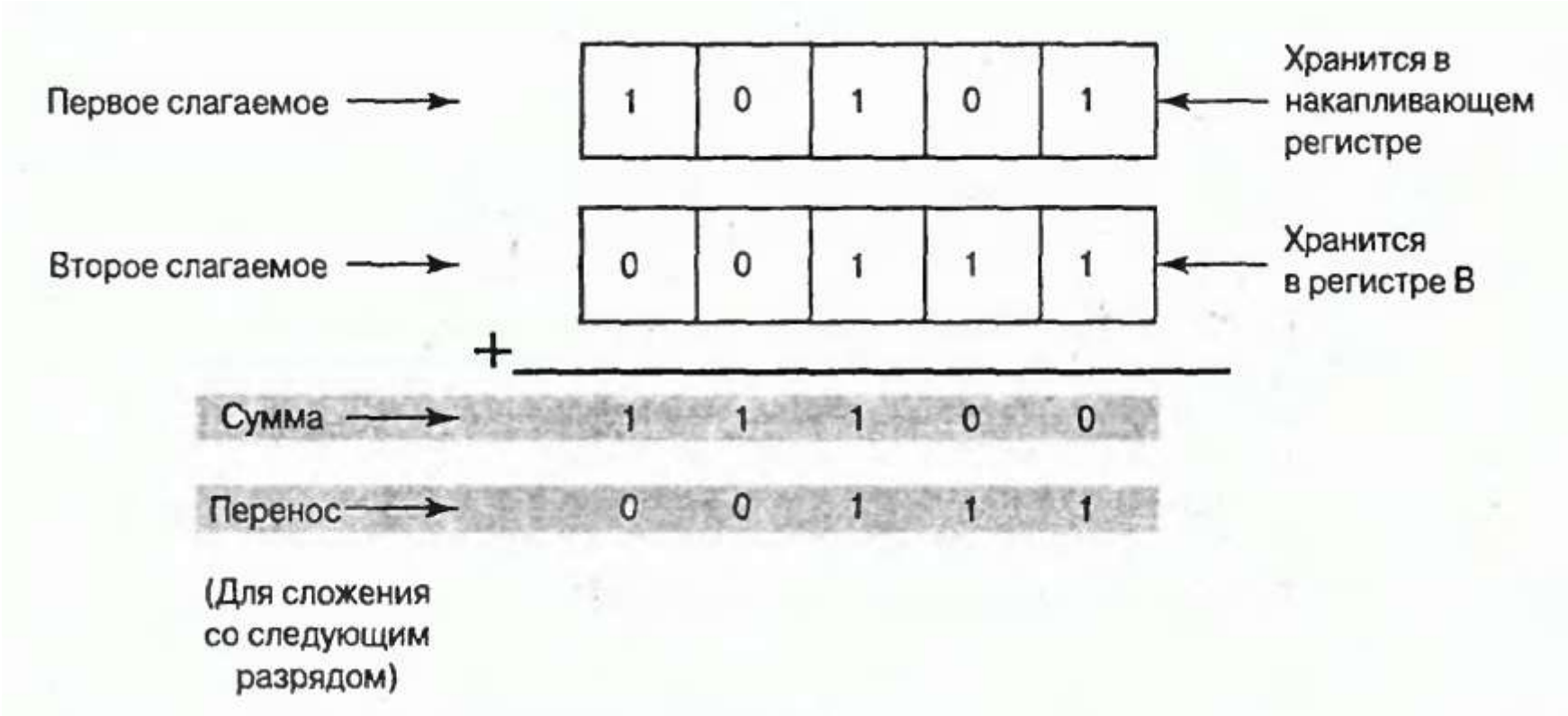
Таблица истинности

Входы сравнения				Входы расширения			Выходы		
A_3, B_3	A_2, B_2	A_1, B_1	A_0, B_0	$I_{A>B}$	$I_{A<B}$	$I_{A=B}$	$O_{A>B}$	$O_{A<B}$	$O_{A=B}$
$A_3 > B_3$	X	X	X	X	X	X	В	Н	Н
$A_3 < B_3$	X	X	X	X	X	X	Н	В	Н
$A_3 = B_3$	$A_2 > B_2$	X	X	X	X	X	В	Н	Н
$A_3 = B_3$	$A_2 < B_2$	X	X	X	X	X	Н	В	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 > B_1$	X	X	X	X	В	Н	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 < B_1$	X	X	X	X	Н	В	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 > B_0$	X	X	X	В	Н	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 < B_0$	X	X	X	Н	В	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	В	Н	Н	В	Н	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	Н	Н	Н	Н	В	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	X	X	В	Н	Н	В
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	Н	Н	Н	В	В	Н
$A_3 = B_3$	$A_2 = B_2$	$A_1 = B_1$	$A_0 = B_0$	В	В	Н	Н	Н	Н

В — высокий уровень напряжения
Н — низкий уровень напряжения
X — несущественно

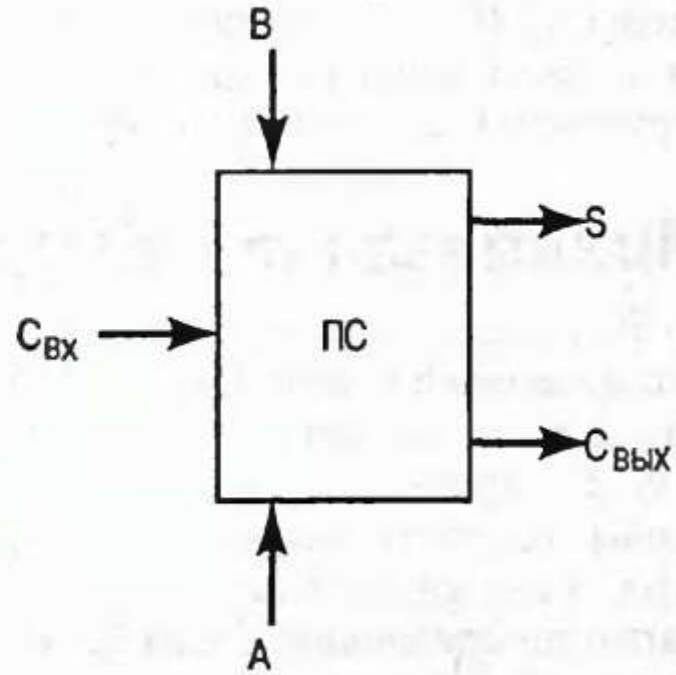

Младшие биты
Старшие биты
A3 A2 A1 A0 B3 B2 B1 B0
A7 A6 A5 A4 B7 B6 B5 B4
+5 В
A
B
IA>B
IA<B
IA=B
74HC85
OA>B OA<B OA=B
Выходы
8-разрядного сравнения

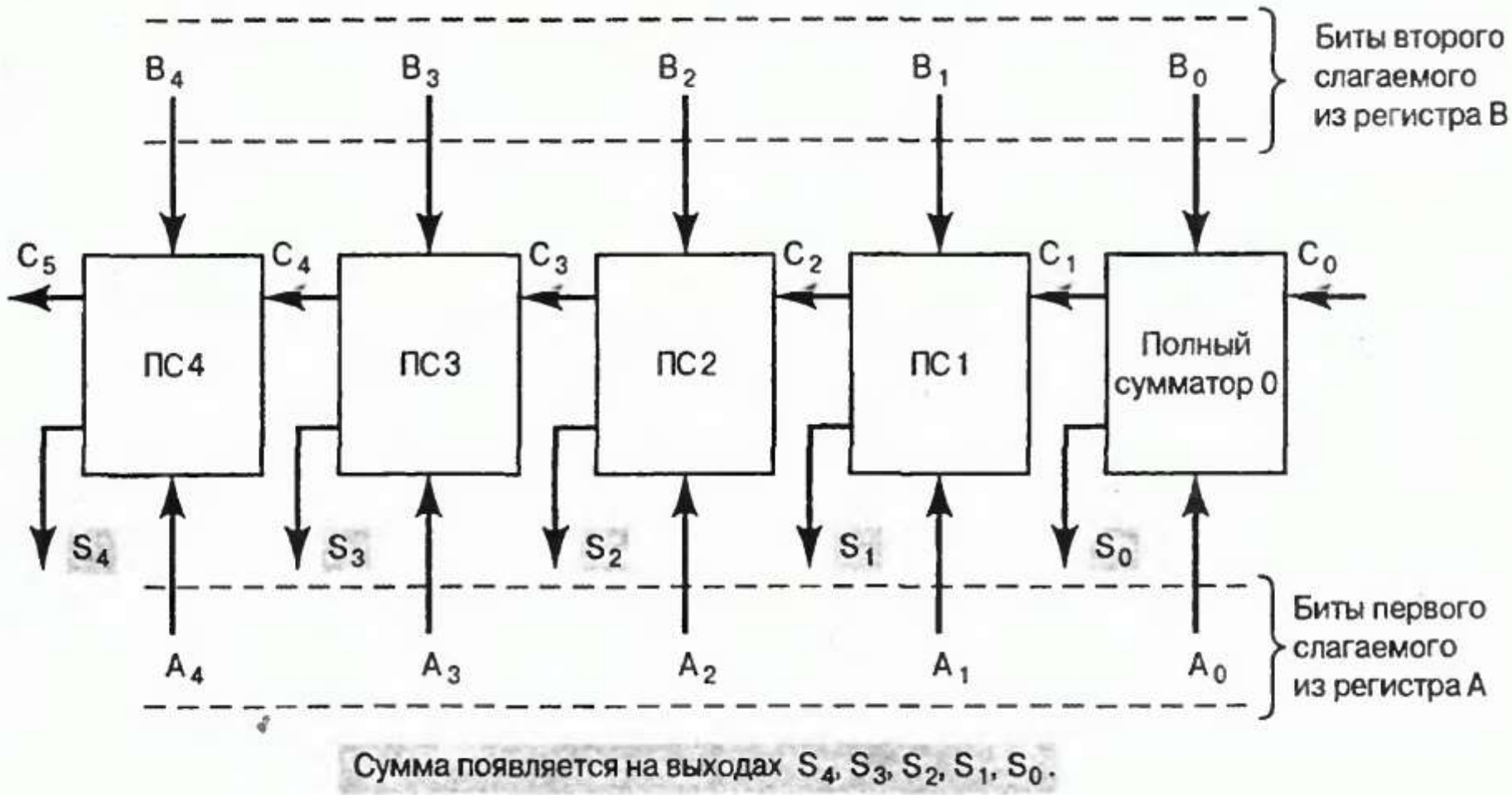
Принцип работы сумматора



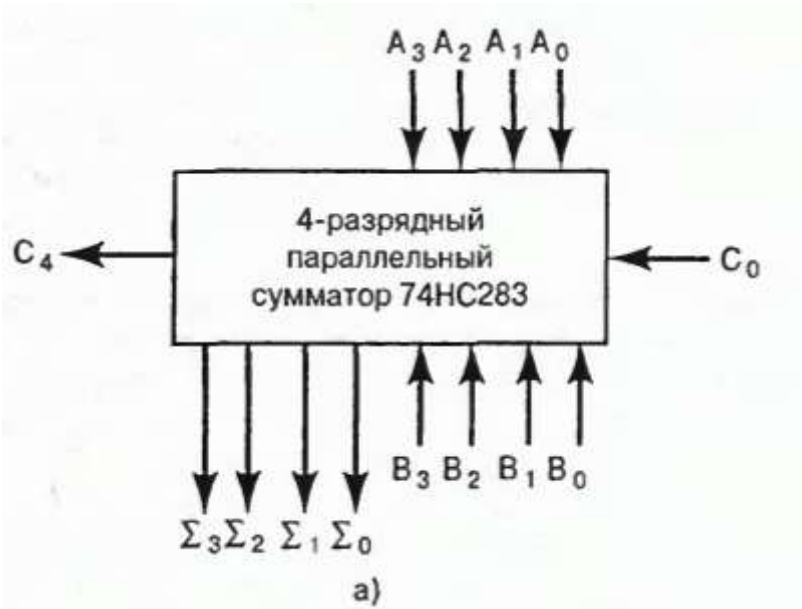
Вход битов первого слагаемого	Вход битов второго слагаемого	Вход битов переноса		Выход битов суммы	Выход битов переноса
A	B	$C_{вх}$		S	$C_{вых}$
0	0	0		0	0
0	0	1		1	0
0	1	0		1	0
0	1	1		0	1
1	0	0		1	0
1	0	1		0	1
1	1	0		0	1
1	1	1		1	1

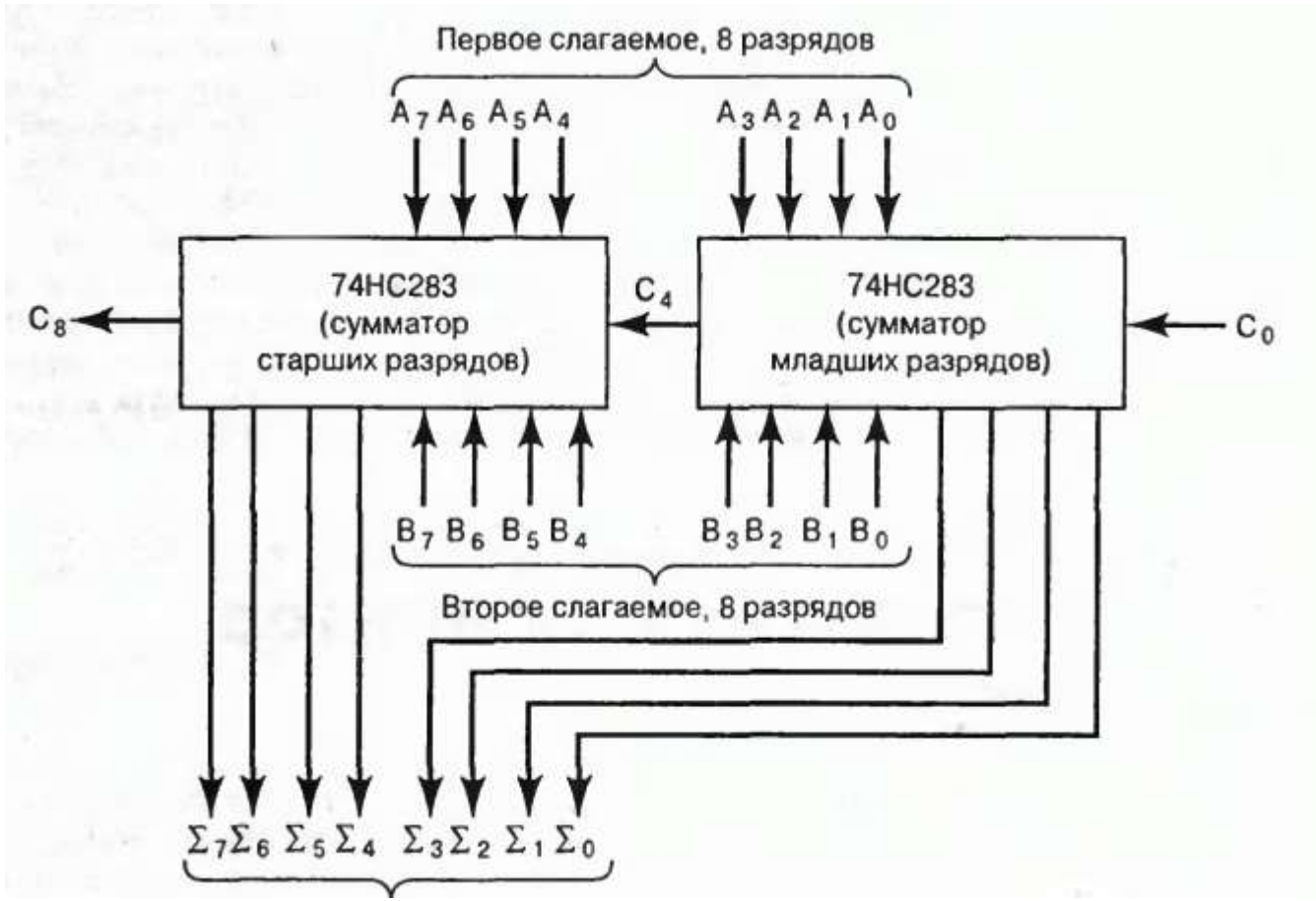
Таблица истинности полного сумматора





Блок-схема параллельного пятибитового сумматора, использующего отдельные полные сумматоры


A3 A2 A1 A0
4-разрядный
параллельный
сумматор 74HC283
C4
C0
B3 B2 B1 B0
Σ3 Σ2 Σ1 Σ0
а)


Первое слагаемое, 8 разрядов
A7 A6 A5 A4
A3 A2 A1 A0
74HC283
(сумматор
старших разрядов)
74HC283
(сумматор
младших разрядов)
C8
C4
C0
B7 B6 B5 B4
B3 B2 B1 B0
Второе слагаемое, 8 разрядов
Σ7 Σ6 Σ5 Σ4
Σ3 Σ2 Σ1 Σ0

Преобразователи кодов

Двоично-десятичный код в семисегментный

Двоично-десятичный код в двоичный

Двоичный код в двоично-десятичный

Двоичный код в код Грея

Код Грея в двоичный

ASCII-код в EBCDIC-код*

EBCDIC-код в ASCII-код

* EBCDIC (Extended Binary Coded Decimal Interchange Code) — расширенный двоично-десятичный код обмена информацией — алфавитно-цифровой код, разработанный компанией IBM, подобный по своей структуре ASCII-коду

В качестве примера схемы преобразователя кода рассмотрим устройство, преобразующее двоично-десятичный код в прямой двоичный. Однако прежде чем изучать данную схему, еще раз вспомним, что собой представляет двоично-десятичный код.

Двухразрядные десятичные значения от 00 до 99 могут быть представлены в виде двоично-десятичного кода с помощью двух четырехбитовых кодовых групп. Например, число 57_{10} можно представить как

$$\overbrace{0101}^{5}\quad\overbrace{0111}^{7}$$ (Двоично-десятичный код)

Прямой двоичный код числа 57 будет равен

$$57_{10} = 111001_2.$$

Наибольшее двухразрядное десятичное значение 99 можно представить в следующем виде:

$$99_{10} = 10011001 \text{ (двоично-десятичный код)} = 1100011_2.$$

Обратите внимание, что двоичное представление того же числа потребовало всего семь бит.

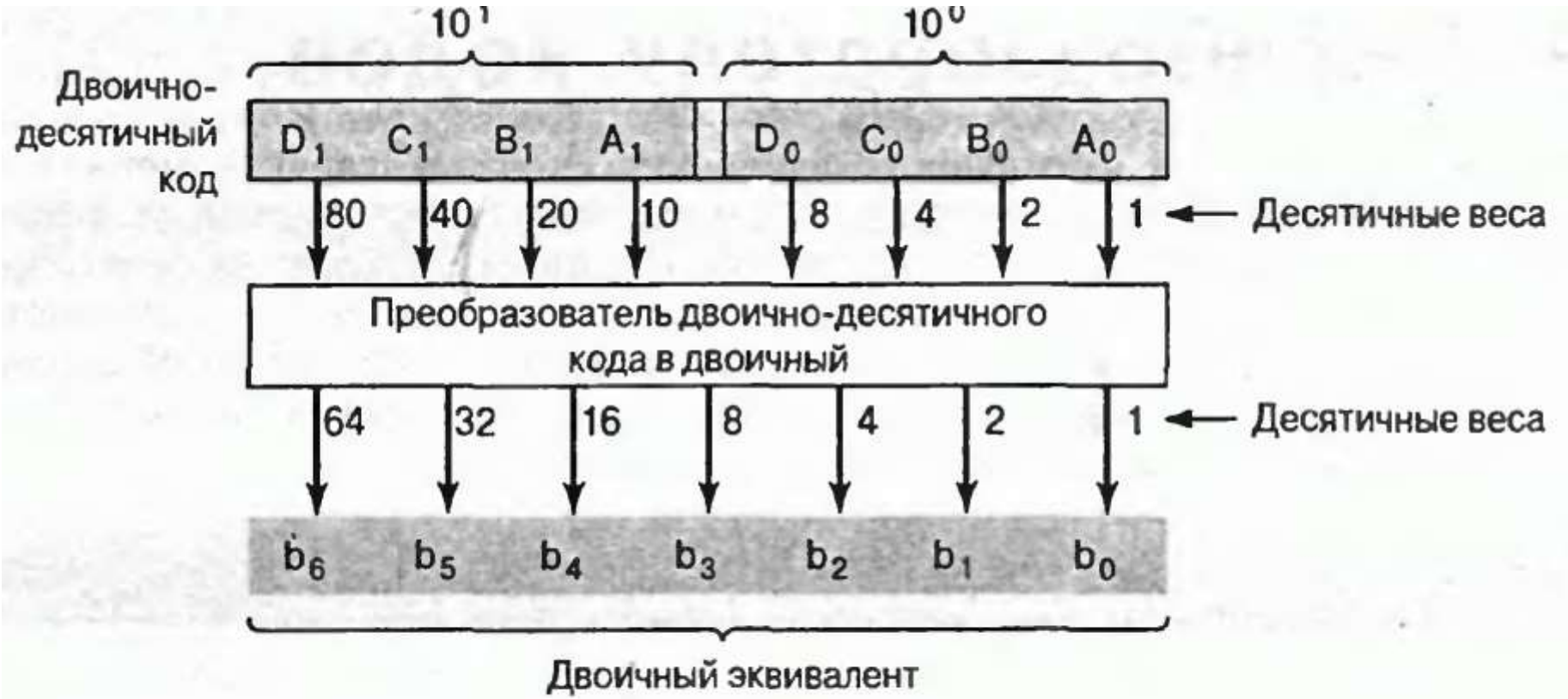


Рис. 9.39. Основной принцип работы преобразователя двухразрядного двоично-десятичного кода в двоичный код

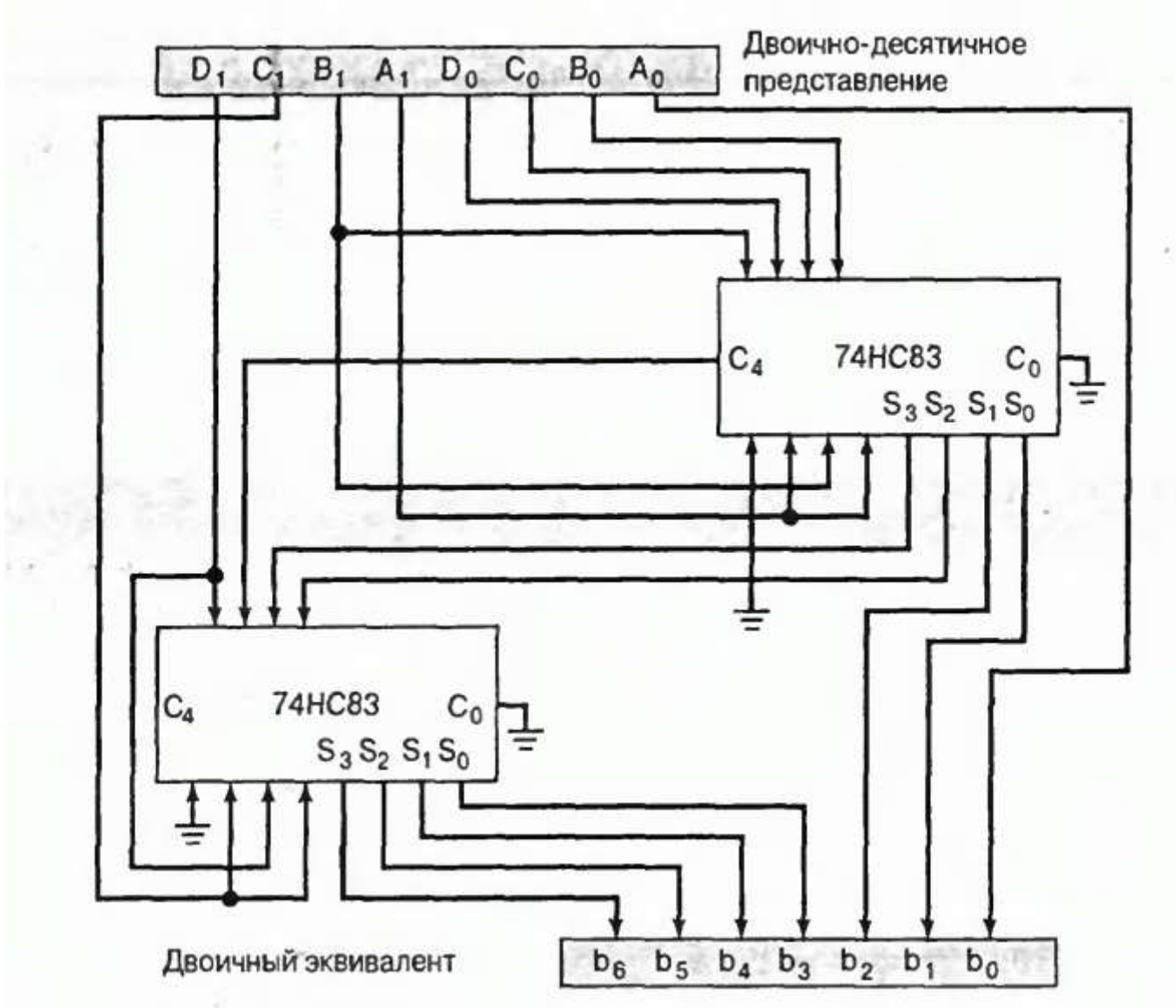
В двоично-десятичном виде биты каждой кодовой группы имеют десятичные веса, равные, соответственно, 8, 4, 2, 1; однако соседние группы отличаются друг от друга множителем 10 (вес десятичного разряда). На рис. показаны веса битов при двухразрядном двоично-десятичном представлении чисел.

Десятичный вес каждого бита в двоично-десятичном представлении может быть преобразован в двоичный эквивалент. Результаты приведены в табл. Используя указанные веса, можно преобразовать двоично-десятичный код в двоичный, следуя простому правилу:

необходимо рассчитать двоично-десятичную сумму двоичных эквивалентов всех двоично-десятичных битов, которые равны 1.

Двоично-десятичный бит	Десятичный вес	Двоичный эквивалент						
		b_6	b_5	b_4	b_3	b_2	b_1	b_0
A_0	1	0	0	0	0	0	0	1
B_0	2	0	0	0	0	0	1	0
C_0	4	0	0	0	0	1	0	0
D_0	8	0	0	0	1	0	0	0
A_1	10	0	0	0	1	0	1	0
B_1	20	0	0	1	0	1	0	0
C_1	40	0	1	0	1	0	0	0
D_1	80	1	0	1	0	0	0	0


0 1 0 1 0 0 1 0
двоично-десятичное представление
0000010 (двоичная 2)
0001010 (двоичное 10)
+ 0101000 (двоичное 40)
0110100 (двоичное 52)
1 0 0 1 0 1 0 1
двоично-десятичное представление
0000001 (двоичная 1)
0000100 (двоичное 4)
0001010 (двоичное 10)
+ 1010000 (двоичное 80)
1011111 (двоичное 95)


D1 C1 B1 A1 D0 C0 B0 A0
Двоично-десятичное представление
C4 74HC83 C0
S3 S2 S1 S0
C4 74HC83 C0
S3 S2 S1 S0
Двоичный эквивалент
b6 b5 b4 b3 b2 b1 b0