ОСНОВЫ ЛОГИКИ

Проектирование цифровой техники с применением ПЛИС и аппаратного языка разработки System Verilog

Н. Г. Зайцев

Кандидат технических наук, преподаватель кафедры КИПР ТУСУР, начальник сектора цифровой электроники ООО «ЛЭМЗ-Т»

Системы счисления

Десятичная система счисления

1000-и 100-и 10-ки 1-ы

$$9742_{10} = 9 \times 10^3 + 7 \times 10^2 + 4 \times 10^1 + 2 \times 10^0$$

Девять тысяч

Семь сотен

Четыре десятка

Две единицы

Системы счисления

Двоичная система счисления

16 8 4 2 1

$$10110_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 +$$

один по Шестнадцать | ноль восьмерок | одна четверка

$$+1 \times 2^1 + 0 \times 2^0 = 22_{10}$$

одна двойка | ноль единиц

1-битные двоичные числа	2-битные двоичные числа	3-битные двоичные числа	4-битные двоичные числа	Десятичные эквиваленты
0	00	000	0000	0
1	01	001	0001	1
	10	010	0010	2
	11	011	0011	3
		100	0100	4
		101	0101	5
		110	0110	6
		111	0111	7
			1000	8
			1001	9
			1010	10
			1011	11
			1100	12
			1101	13
			1110	14
			1111	15

Системы счисления

Шестнадцатеричная система счисления

разряд 256-ти, разряд 16-ти, разряд 1-ц

$2ED_{16} = 2 \times 16^2 + E \times 16^1 +$

два по двести пятьдесят шесть

четырнадцать по шестнадцать

$+ D \times 16^0 = 749_{10}$

Тринадцать по одному

Шестнадцатеричная цифра	Десятичный эквивалент	Двоичный эквивалент
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

Системы счисления

Байт, полубайт

Группа из восьми битов называется *байт (byte)*. Байт представляет $2^8 = 256$ цифровых комбинаций. Размер модулей, сохраненных в памяти компьютера, обычно измеряется именно в байтах, а не битах.

Группа из четырех битов (половина байта) называется *полубайт (nibble)*. Полубайт представляет $2^4 = 16$ цифровых комбинаций. Одна шестнадцатеричная цифра занимает один полубайт, а две шестнадцатеричные цифры – один байт. В настоящее время полубайты уже не находят широкого применения, однако этот термин все же стоит знать, да и звучит он забавно (в английском языке *nibble* означает откусывать что-либо маленькими кусочками).

101100

наибольший значащий бит

наименьший значащий бит

a)

DEAFDAD8

наибольший значащий байт

наименьший значащий байт

b)

Системы счисления

Сложение двоичных чисел

перенос

$$\begin{array}{r} {\scriptstyle 1\ 1\ \ } \\ 4277 \\ +\,5499 \\ \hline 9776 \end{array}$$

a)

$$\begin{array}{r} {\scriptstyle 1\ 1\ \ } \\ 1011 \\ +\,0011 \\ \hline 1110 \end{array}$$

b)

Примеры сложения с переносом:
десятичное (*a*), двоичное (*b*)

$$\begin{array}{r} {\scriptstyle 1\ 1\ 1\ \ } \\ 0111 \\ +\,0101 \\ \hline 1100 \end{array}$$

Пример двоичного сложения

$$\begin{array}{r} {\scriptstyle 1\ 1\ \ \ 1\ \ } \\ 1101 \\ +\,0101 \\ \hline 10010 \end{array}$$

Пример двоичного сложения с переполнением

Системы счисления

Знак двоичных чисел

- Двоичные числа без знака (*Unsigned*)
- Прямой код (*Sign/Magnitude*)
- Дополнительный код (*Two's Complement*)

Историческая заметка

Ракета Ариан-5 ценой 7 млрд долл., запущенная 4 июня 1996 г., отклонилась от курса и разрушилась через 40 секунд после запуска. Отказ был вызван тем, что в бортовом компьютере произошло переполнение 16-разрядных регистров, после которого компьютер вышел из строя.

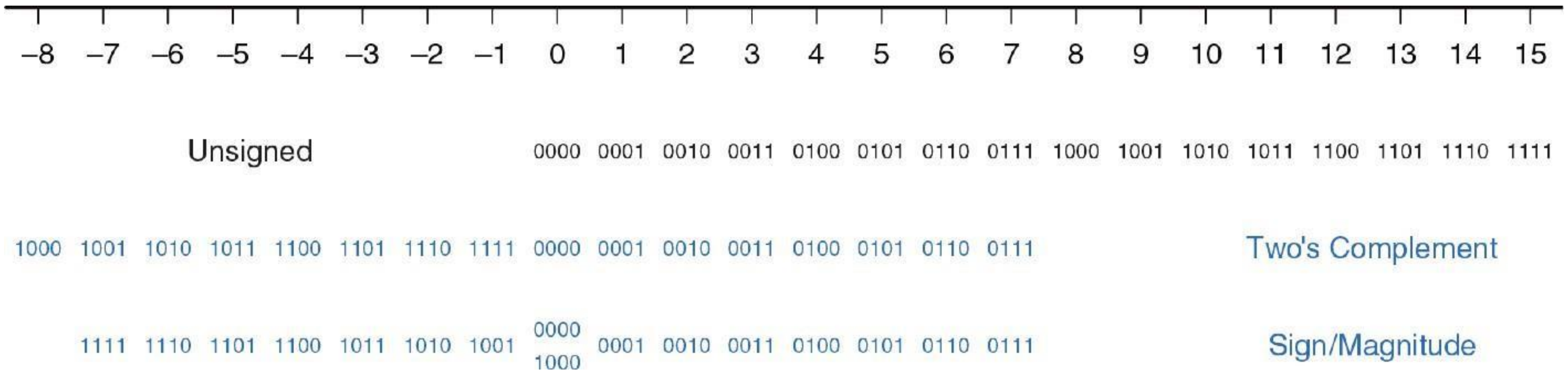
Программное обеспечение Ариан-5 было тщательно протестировано, но на ракете Ариан-4. Однако новая ракета имела двигатели с более высокими скоростными параметрами, которые, будучи переданными бортовому компьютеру, и вызвали переполнение регистров.

Системы счисления

Диапазон *N*-битных чисел

Система	Диапазон
Двоичные числа без знака (unsigned)	$[0, 2^{N} - 1]$
Прямой Код (Sign/Magnitude)	$[-2^{N-1} - 1, 2^{N-1} - 1]$
Дополнительный код	$[-2^{N-1}, 2^{N-1} - 1]$

Числовая шкала и 4-битовое двоичное кодирование



Логические элементы

Логические вентили (*logic gates*) - это простейшие цифровые схемы, получающие один или более двоичных сигналов на входе и производящие новый двоичный сигнал на выходе.

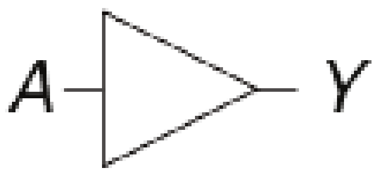
При графическом изображении логических вентилей для обозначения одного или нескольких входных сигналов и выходного сигнала используются специальные символы.

Взаимосвязь между входными сигналами и выходным сигналом логического вентиля может быть описана с помощью таблицы истинности (*truth table*) или уравнением булевой логики.

Логические элементы

Базовые элементы

BUF

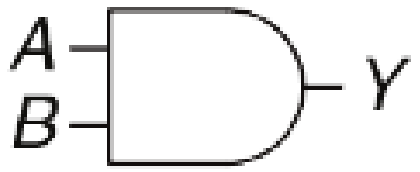


$Y = A$

A	Y
0	0
1	1

Буфер

AND

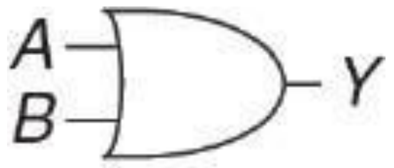


$Y = AB$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Вентиль И

ИЛИ



$Y = A + B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Вентиль ИЛИ

XOR

$Y = A \oplus B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Вентиль ИСКЛЮЧАЮЩЕЕ ИЛИ

Логические элементы

Базовые инверсные элементы

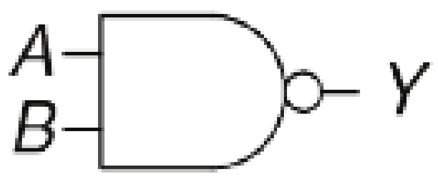
NOT

$Y = \overline{A}$

A	Y
0	1
1	0

Вентиль НЕ

NAND

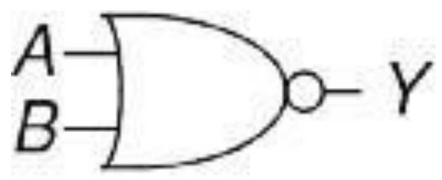


$Y = \overline{AB}$

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Вентиль И-НЕ

NOR

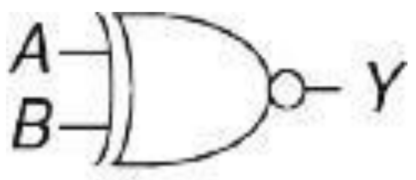


$Y = \overline{A + B}$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Вентиль ИЛИ-НЕ

XNOR



$Y = \overline{A \oplus B}$

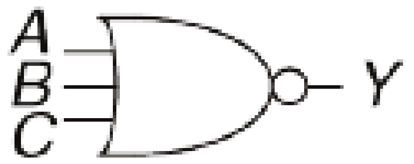
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Вентиль ИСКЛЮЧАЮЩЕЕ ИЛИ-НЕ

Логические элементы

Логические элементы с количеством входов больше двух

NOR3

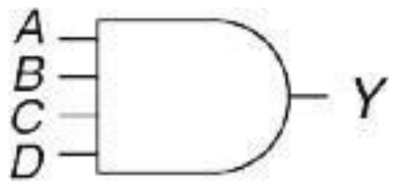


$$Y = \overline{A + B + C}$$

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Вентиль ИЛИ-НЕ с тремя входами

AND4



$$Y = ABCD$$

A	C	B	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Вентиль И с четырьмя входами

Булевы уравнения

Булевы уравнения используют переменные, имеющие значение ИСТИНА или ЛОЖЬ, поэтому они идеально подходят для описания цифровой логики.

Три базовые операции булевой алгебры:

1. **Дополнение** (*complement*) - это величина, обратная логической переменной. Дополнение 0 равно 1, а дополнение 1 равно 0. Переменная или ее дополнение называются литералом. Например, A, $\overline{A}$, B и $\overline{B}$ - литералы.
2. **Конъюнкция, произведение** (*product*) **или импликанта** - операция «И» над одним или несколькими литералами.

   Минтерм (*minterm*, элементарная конъюнктивная форма) - это произведение, включающее все входы функции. $A\overline{B}\overline{C}$ - это минтерм для функции переменных A, B и C.
3. **Дизъюнкция или сумма** - операция «ИЛИ» над одним или более литералами.

   Макстерм (*maxterm*, элементарная дизъюнктивная форма) - это сумма всех входов функции. $A+\overline{B}+C$ является макстермом функции переменных A, B и C.

Булевы уравнения

Булевы уравнения используют переменные, имеющие значение ИСТИНА или ЛОЖЬ, поэтому они идеально подходят для описания цифровой логики.

Конъюнктивная форма

A	B	Y	minterm	minterm name
0	0	0	$\overline{A}\ \overline{B}$	m_0
0	1	1	$\overline{A}\ B$	m_1
1	0	0	$A\ \overline{B}$	m_2
1	1	1	$A\ B$	m_3

$$F(A,, B) = \Pi(m_1, m_3)$$

или

$$F(A, B) = \Pi(1, 3)$$

Таблица истинности с несколькими минтермами, равными ИСТИНЕ,
и ее представление в виде функций

$$Y = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$

или

$$Y = \Pi(0, 4, 5)$$

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Произвольная таблица истинности
с тремя входами

Булевы уравнения

Булевы уравнения используют переменные, имеющие значение ИСТИНА или ЛОЖЬ, поэтому они идеально подходят для описания цифровой логики.

Дизъюнктивная форма

A	B	Y	maxterm	maxterm name
0	0	0	$A + B$	M_0
0	1	1	$A + \overline{B}$	M_1
1	0	0	$\overline{A} + B$	M_2
1	1	1	$\overline{A} + \overline{B}$	M_3

$$F(A, B) = \Sigma(M_1, M_3)$$

или

$$F(A, B) = \Sigma(1, 3)$$

Таблица истинности с несколькими макстермами, равными ИСТИНЕ,
и ее представление в виде функций

Булева алгебра

Булева алгебра основана на наборе аксиом, которые мы считаем верными. Аксиомы являются недоказуемыми в том смысле, что определение не может быть доказано. С помощью этих аксиом мы доказываем все теоремы булевой алгебры.

Аксиомы булевой алгебры

	Аксиома		**Двойственная аксиома**	**Название**
A1	$B = 0$ если $B \neq 1$	A1'	$B = 1$ если $B \neq 0$	Бинарное поле
A2	$\overline{0} = 1$	A2'	$\overline{1} = 0$	НЕ
A3	$0 \cdot 0 = 0$	A3'	$1 + 1 = 1$	И/ИЛИ
A4	$1 \cdot 1 = 1$	A4'	$0 + 0 = 0$	И/ИЛИ
A5	$0 \cdot 1 = 1 \cdot 0 = 0$	A5'	$1 + 0 = 0 + 1 = 1$	И/ИЛИ

Булева алгебра

Булевы теоремы для одной переменной

	Теорема		Двойственная теорема	Название
T1	$B \cdot 1 = B$	T1'	$B + 0 = B$	Идентичность
T2	$B \cdot 0 = 0$	T2'	$B + 1 = 1$	Нулевой элемент
T3	$B \cdot B = B$	T3'	$B + B = B$	Идемпотентность
T4		$\bar{\bar{B}} = B$		Инволюция
T5	$B \cdot \bar{B} = 0$	T'	$B + \bar{B} = 1$	Дополнительность

Булева алгебра

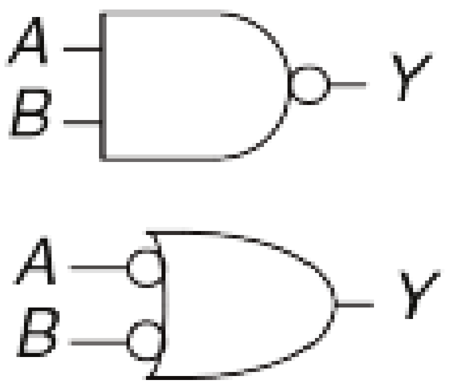
Теоремы с несколькими переменными

	Теорема		Двойственная теорема	Название
T6	$B \cdot C = C \cdot B$	T6'	$B + C = C + B$	Коммутативность
T7	$(B \cdot C) \cdot D = B \cdot (C \cdot D)$	T7'	$(B + C) + D = B + (C + D)$	Ассоциативность
T8	$(B \cdot C) + (B \cdot D) = B \cdot (C + D)$	T8'	$(B + C) \cdot (B + D) = B + (C \cdot D)$	Дистрибутивность
T9	$B \cdot (B + C) = B$	T9'	$B + (B \cdot C) = B$	Поглощение
T10	$(B \cdot C) + (B \cdot \overline{C}) = B$	T10'	$(B + C) \cdot (B + \overline{C}) = B$	Склеивание
T11	$(B \cdot C) + (\overline{B} \cdot D) + (C \cdot D) =$ $= B \cdot C + \overline{B} \cdot D$	T11'	$(B + C) \cdot (\overline{B} + D) \cdot (C + D) =$ $= (B + C) \cdot (\overline{B} + D)$	Согласованность
T12	$\overline{B_0 \cdot B_1 \cdot B_2 \ldots} = (\overline{B_0} + \overline{B_1} + \overline{B_2} \ldots)$	T12'	$\overline{B_0 + B_1 + B_2 \ldots} = (\overline{B_0} \cdot \overline{B_1} \cdot \overline{B_2} \ldots)$	Теорема де Моргана

Булева алгебра

Эквивалентные по де Моргану элементы

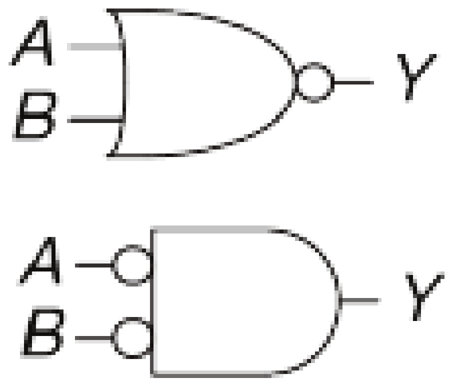
NAND



$$Y = \overline{AB} = \overline{A} + \overline{B}$$

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR



$$Y = \overline{A + B} = \overline{A}\,\overline{B}$$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Булева алгебра

Упрощение уравнений

Шаг	Выражение	Объяснение
	$\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$	
1	$\bar{B}\bar{C}\,(\bar{A} + A) + A\bar{B}C$	T8: дистрибутивность
2	$\bar{B}\bar{C}\,(1) + A\bar{B}C$	T5: дополнительность
3	$\bar{B}\bar{C} + A\bar{B}C$	T1: идентичность

Улучшенная минимизация

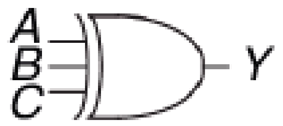
Шаг	Выражение	Объяснение
	$\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$	
1	$\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$	T3: идемпотентность
2	$\bar{B}\bar{C}\,(\bar{A} + A) + A\bar{B}\,(\bar{C} + C)$	T8: дистрибутивность
3	$\bar{B}\bar{C}\,(1) + A\bar{B}\,(1)$	T5: дополнительность
4	$\bar{B}\bar{C} + A\bar{B}$	T1: идентичность

Логические схемы

Принципиальная схема - это изображение цифровой схемы, показывающее элементы и соединяющие их проводники.

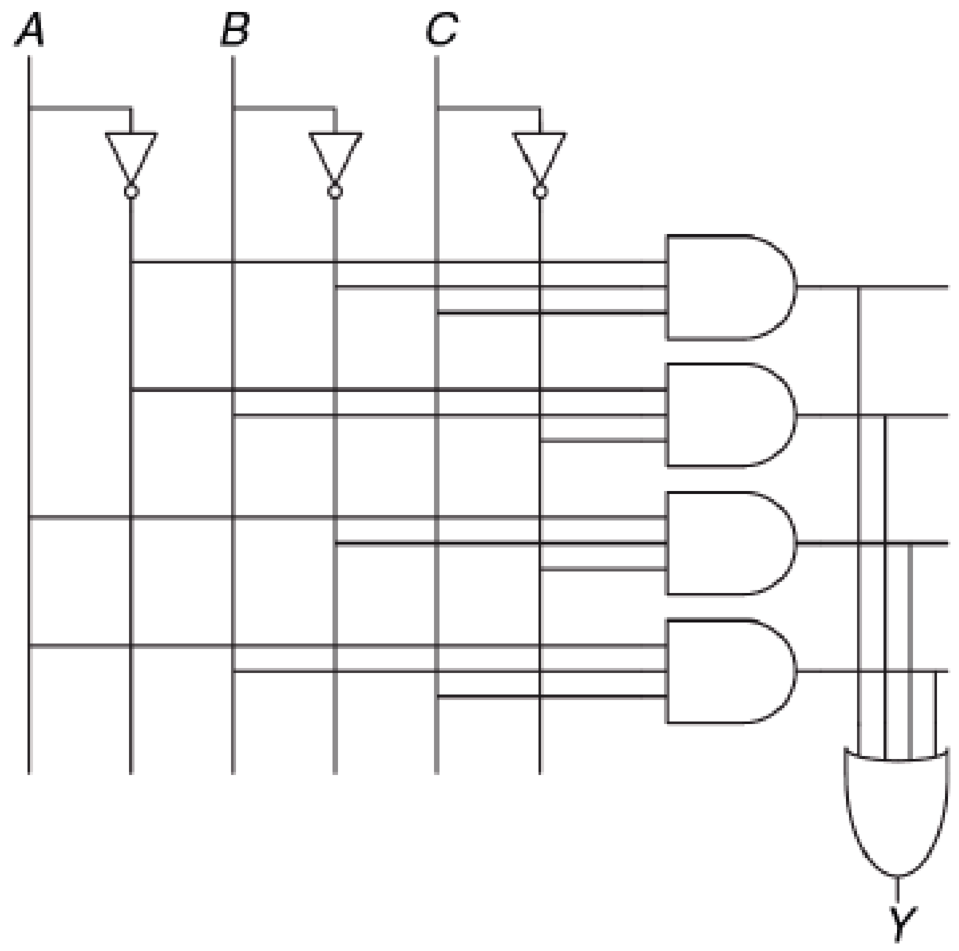
$$Y = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC.$$

XOR3



$Y = A \oplus B \oplus C$

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



Трехвходовый элемент XOR: функциональная спецификация
и реализация с двумя уровнями логики

Карты Карно

Карты Карно представляют собой наглядный метод для упрощения булевых уравнений.

Они были изобретены в 1953 г. Морисом Карно, телекоммуникационным инженером из фирмы Bell Labs. Карты Карно очень удобны в случаях, когда уравнение содержит до четырех переменных.

Но, что более важно, они дают понимание сути при манипулировании логическими выражениями.

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

(a) *a)*

Y: C \ AB	00	01	11	10
0	1	0	0	0
1	1	0	0	0

(b) *b)*

Y: C \ AB	00	01	11	10
0	$\bar{A}\bar{B}\bar{C}$	$\bar{A}B\bar{C}$	$AB\bar{C}$	$A\bar{B}\bar{C}$
1	$\bar{A}\bar{B}C$	$\bar{A}BC$	ABC	$A\bar{B}C$

(c) *c)*

Функция трех переменных: таблица истинности (*a*), карта Карно (*b*), карта Карно с минтермами (*c*)

Карты Карно

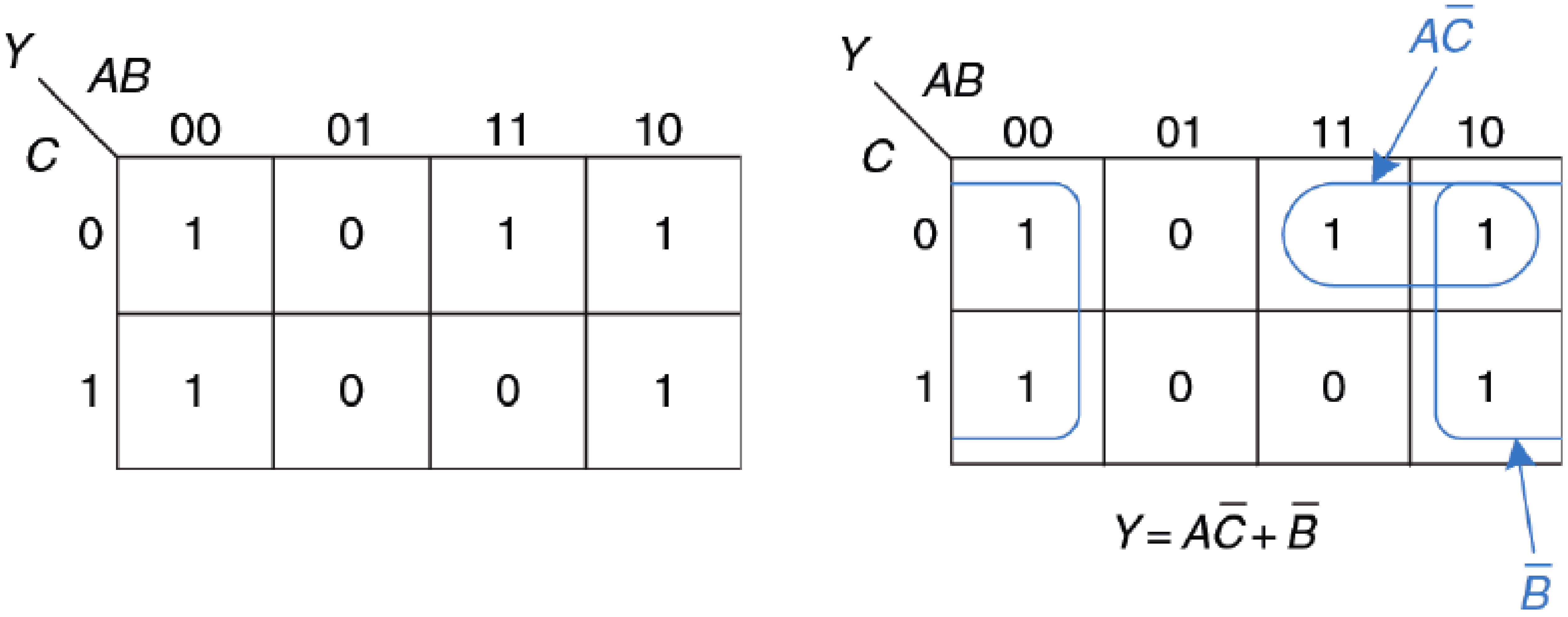
Чтение минтермов из карт Карно в точности соответствует чтению дизъюнктивной нормальной формы (ДНФ) из таблицы истинности.

Карты Карно помогают нам делать это упрощение графически, обводя единицы в соседних клетках овалами (*n*-мерными кубами).

Правила для нахождения минимального уравнения из карт Карно:

- использовать меньше всего овалов, необходимых для покрытия всех единиц;
- все клетки в каждом овале обязаны содержать единицы;
- каждый овал должен охватывать блок, число клеток которого в каждом направлении равно степени двойки (т. е. 1, 2 или 4);
- каждый овал должен быть настолько большим, насколько это возможно;
- овал может связывать края карты Карно;
- единица на карте Карно может быть обведена сколько угодно раз, если это позволяет уменьшить число овалов, которые будут использоваться.

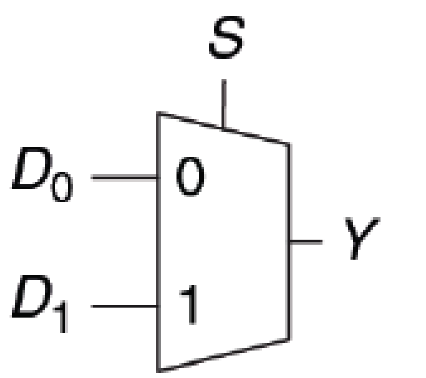
Карты Карно


Y
AB
C
00 01 11 10
0 1 0 1 1
1 1 0 0 1
Y
AB
C
00 01 11 10
0 1 0 1 1
1 1 0 0 1
$A\overline{C}$
$\overline{B}$
$Y = A\overline{C} + \overline{B}$

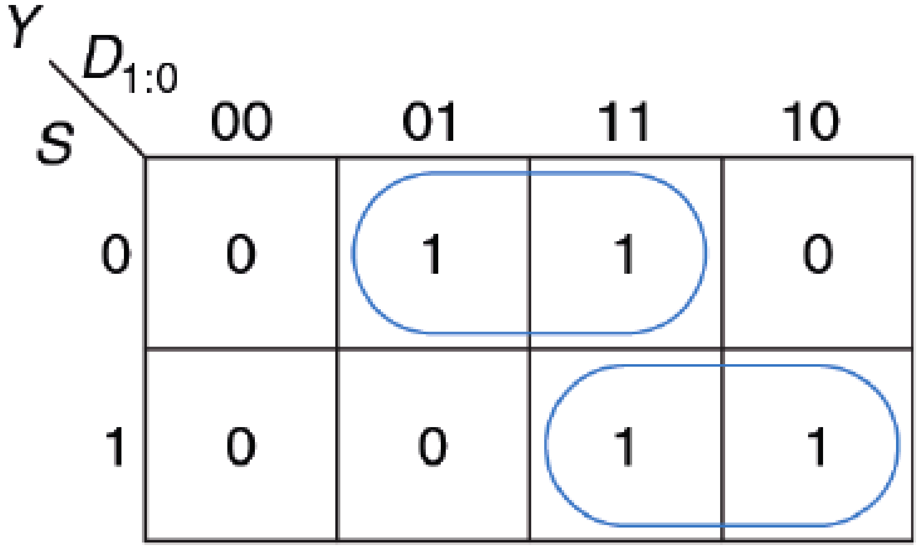
Базовые комбинационные блоки

Комбинационные логические элементы часто группируются в «строительные блоки», используемые для создания сложных систем. Это позволяет абстрагироваться от излишней детализации уровня логических элементов и вести разработку на уровне строительных блоков.

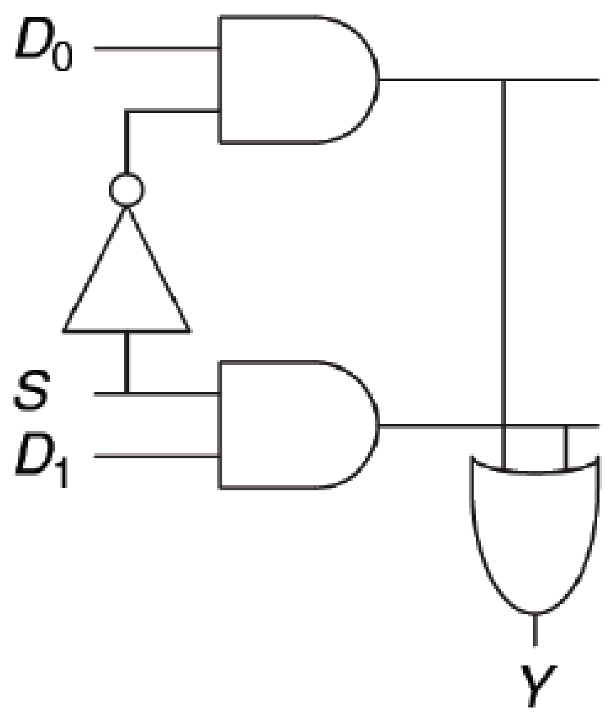
Мультиплексор



S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

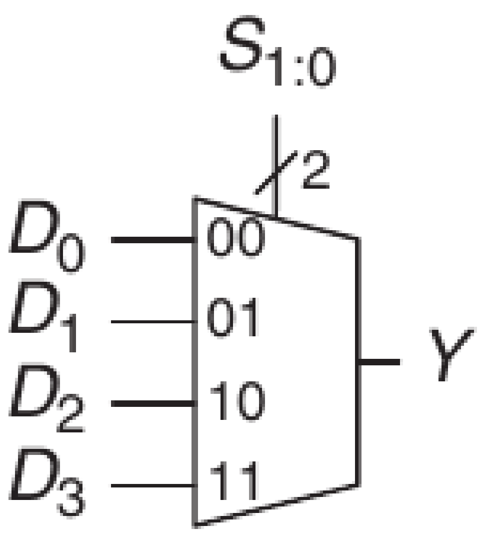


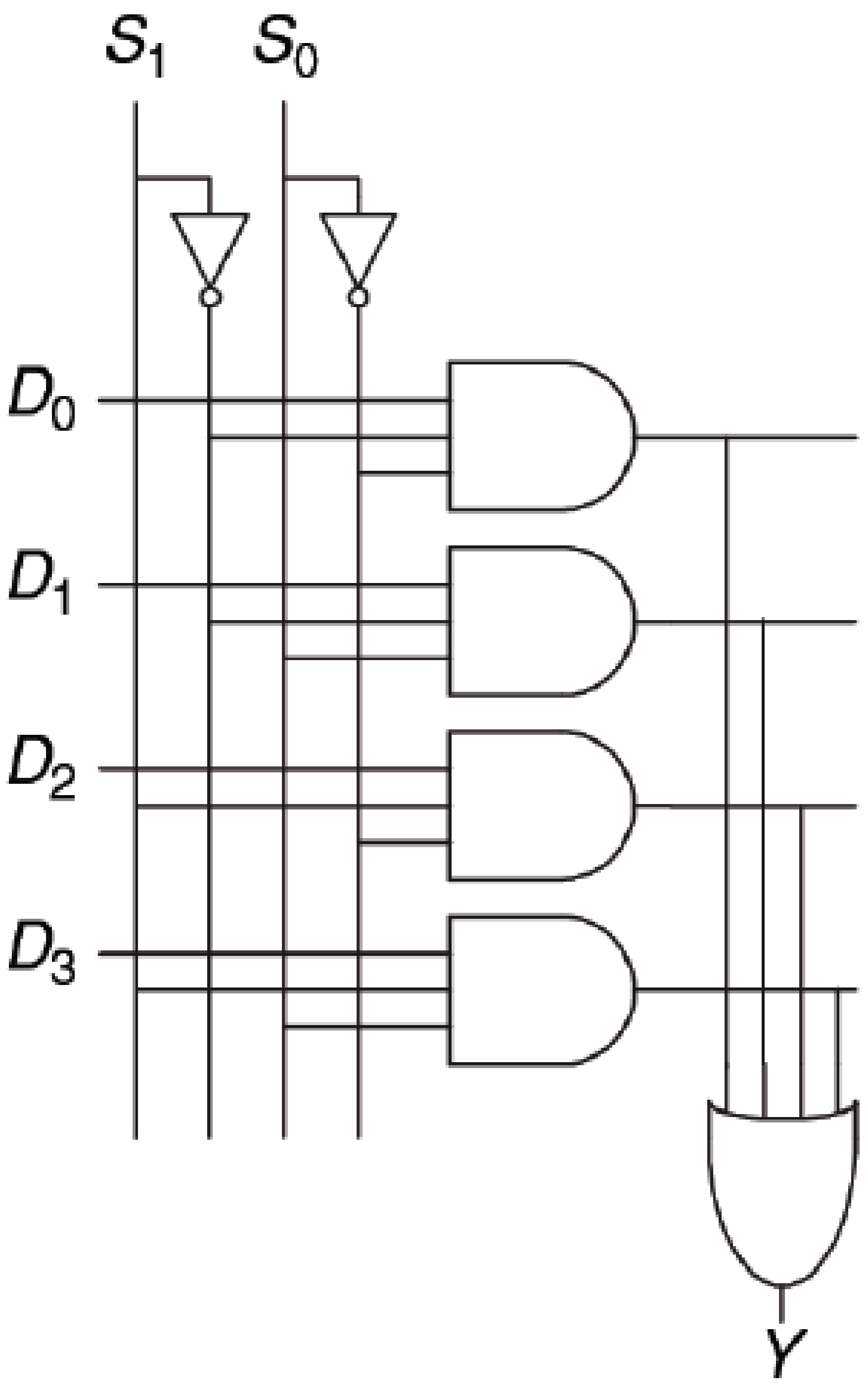
$$Y = D_0\overline{S} + D_1 S$$

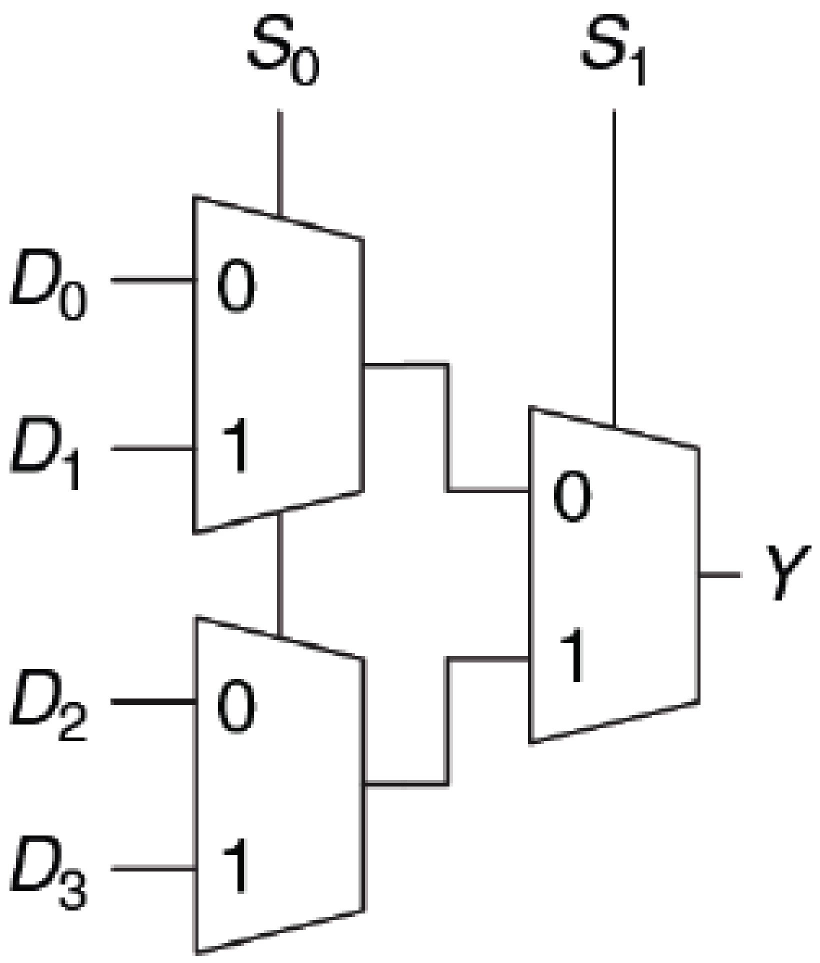


Базовые комбинационные блоки

Многовходовый мультиплексор





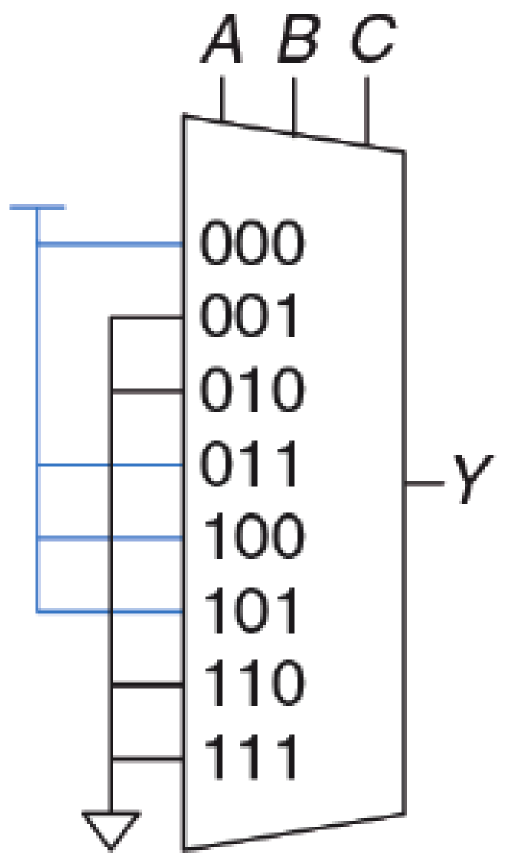


Базовые комбинационные блоки

Логика на мультиплексорах

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

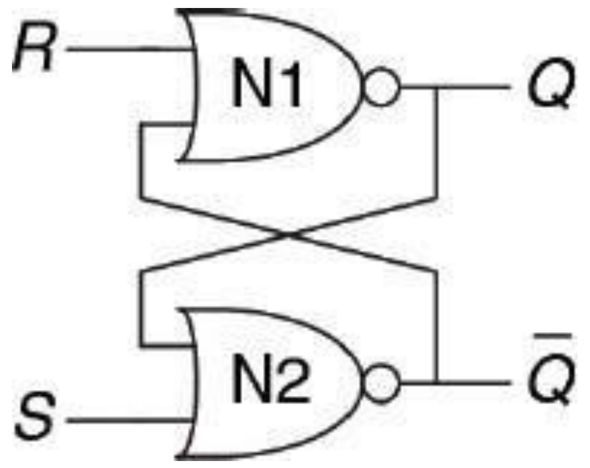
$$Y = A\overline{B} + \overline{B}\,\overline{C} + \overline{A}BC$$



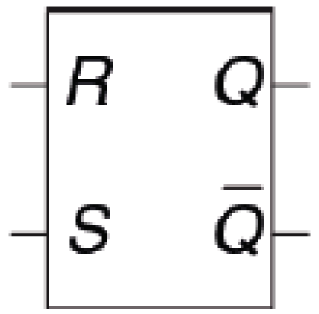
Элементы последовательной логики

Последовательные логические схемы - это логические схемы, значение на выходе которых зависит как от текущих, так и от предыдущих входных значений. Следовательно, последовательные логические схемы обладают памятью**.**

RS-триггер

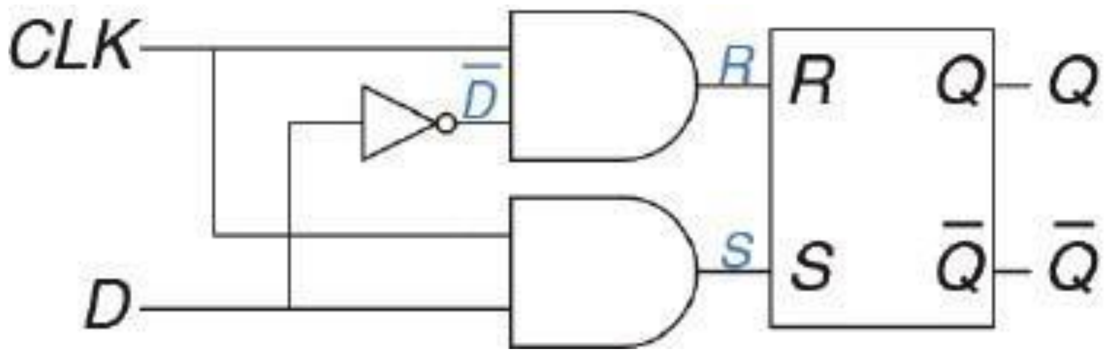


S	R	Q	$\overline{Q}$
0	0	$Q_{пред}$	$\overline{Q}_{пред}$
0	1	0	1
1	0	1	0
1	1	0	0

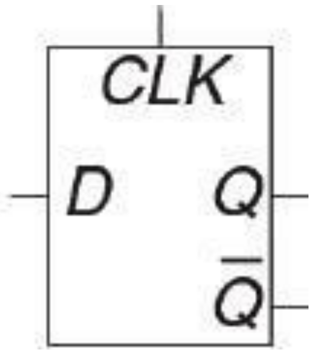


Последовательная логика

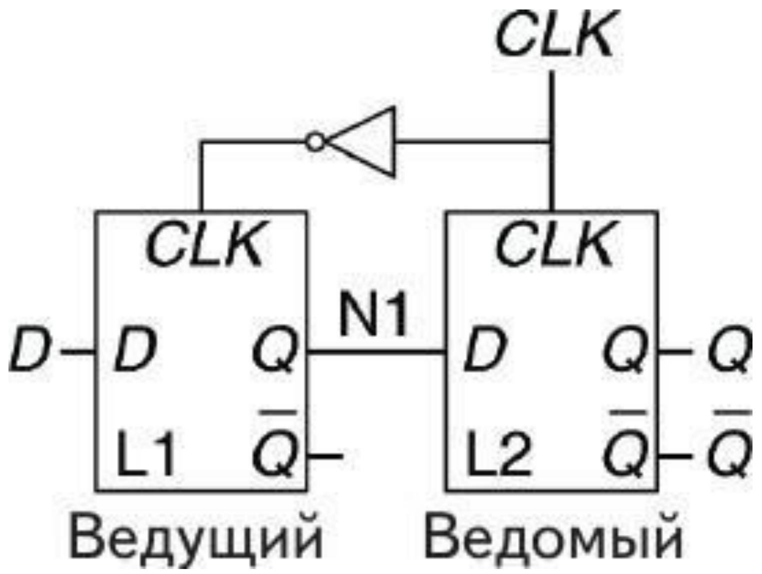
D-защелка

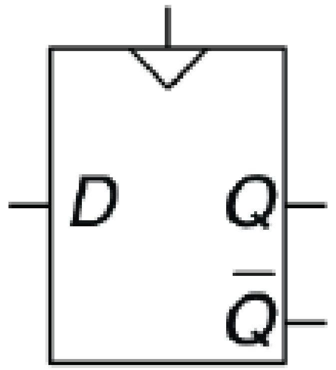


CLK	D	$\overline{D}$	S	R	Q	$\overline{Q}$
0	X	$\overline{X}$	0	0	Q_{prev}	$\overline{Q}_{prev}$
1	0	1	0	1	0	1
1	1	0	1	0	1	0



D-триггер





Регистр *N*-разрядный регистр - набор из *N* триггеров с общим тактовым сигналом. Таким образом, все биты регистра обновляются одновременно. Регистр является ключевым блоком при построении большинства последовательностных схем.

Спасибо за внимание!