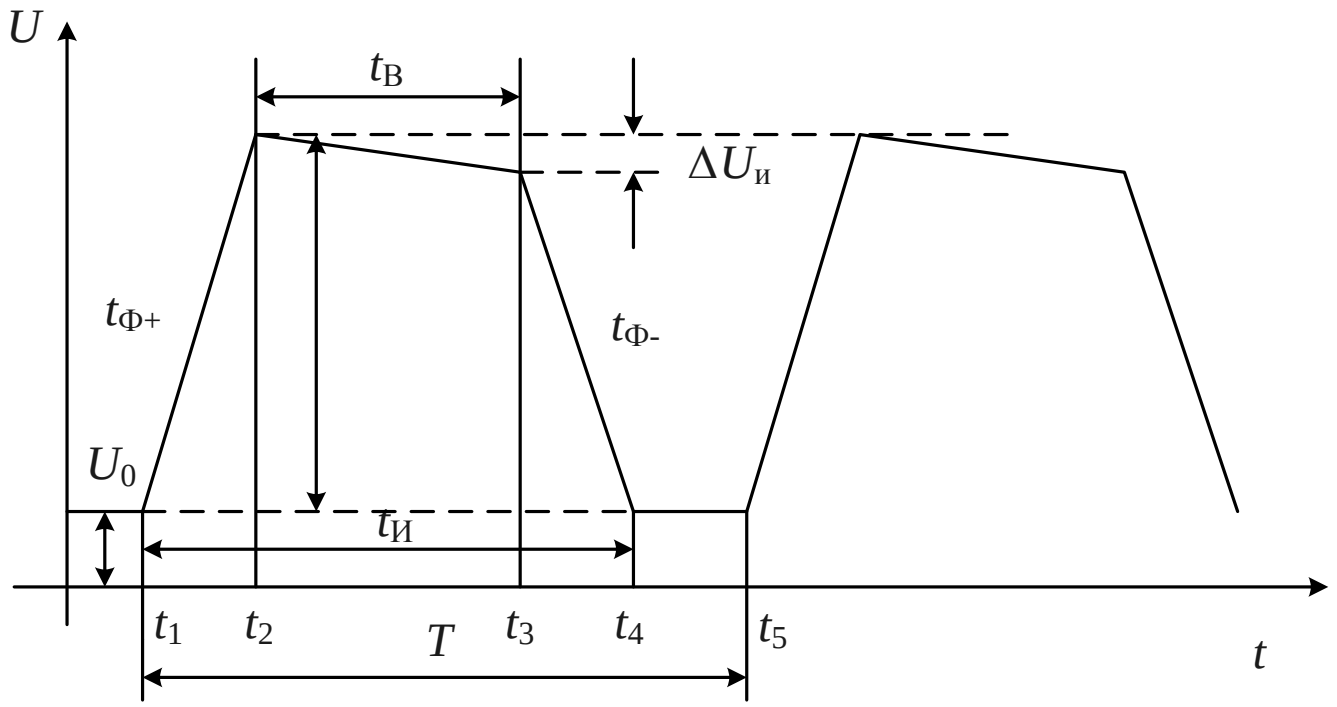
Занятие 10
Цифровые микросхемы

Цифровые интегральные микросхемы (ИМС) предназначены для преобразования и обработки дискретных импульсных сигналов и выполнения логических функций.

В цифровых ИМС используют сигналы, близкие к прямоугольным.

Основные параметры импульсных сигналов



1. Амплитудные параметры

U_0 - начальное значение сигнала (низкий уровень);

$U_и$ - амплитуда импульса;

$\Delta U_и$ - спад вершины.

2. Временные параметры

$t_и = t_4 - t_1$ - длительность импульса;

$t_в = t_3 - t_2$ - длительность вершины импульса;

$t_Ф^+ = t_2 - t_1$ - длительность переднего фронта;

$t_Ф^- = t_4 - t_3$ = длительность заднего фронта;

$T = t_5 - t_1$ - период следования импульсов;

$\frac{1}{T} = f$ - частота следования импульсов;

$Q = \frac{T}{t_и}$ - скважность импульсов;

$\eta = \frac{1}{Q} = \frac{t_u}{T}$ -коэффициент заполнения импульсов.

Если напряжение импульса превышает пороговый уровень 2,5В, считают, что импульс имеет *высокий уровень* H и равен логической «1».

Если напряжение импульса меньше порогового уровня 2,5В, считают, что импульс имеет низкий уровень L и равен логическому «0».

Для передачи, обработки и храненения данных применяют различные **системы кодирования**. Численные данные обычно передают в позиционной двоичной системе счисления. Основанием этой системы служит число 2, а коэффициентами (множителями, стоящими перед числом два в степени (справа налево): нуль, один, два, три) могут быть только две цифры-нуль (0) и единица (1). Например, для передачи числа 12 надо записать четырехразрядный код:

$$1100 \rightarrow 1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0;$$
$$1\cdot 8 + 1\cdot 4 + 0\cdot 2 + 0\cdot 1 = 12.$$

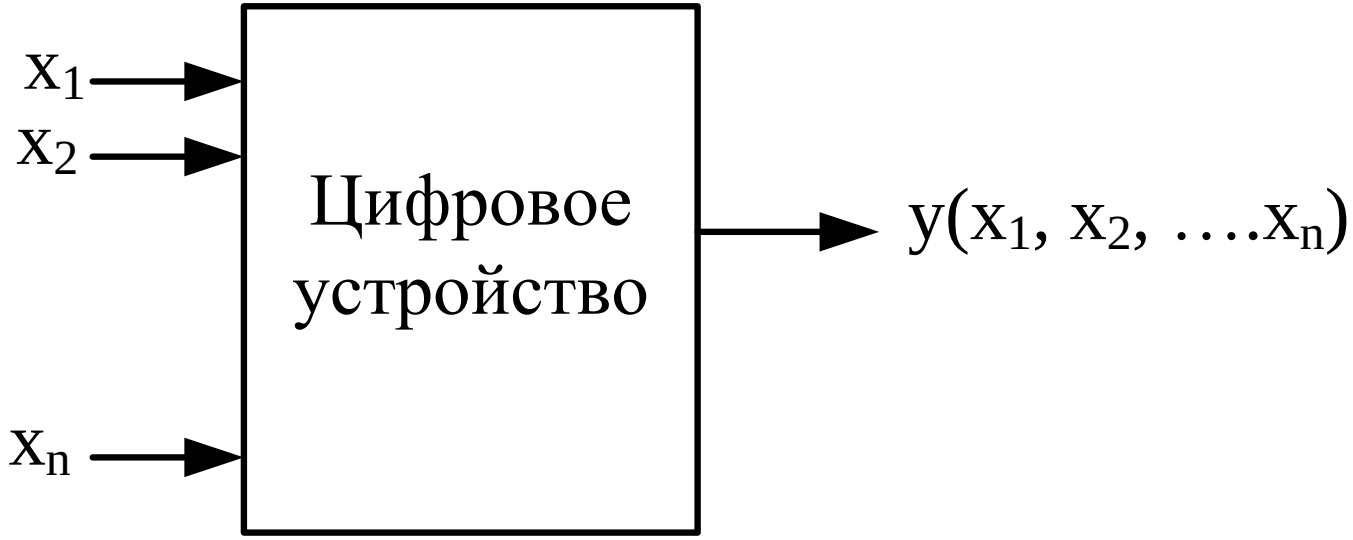
В вычислительной технике каждый разряд кода называют битом. Младший «нулевой» бит расположен справа. Старший (у нас третий бит) расположен слева. Скорость передачи данных оценивеют в килобитах в секунду (Кбит/с), в мегабитах в секунду (Мбит/с).

В любой системе кодирования должно выполняться неравенство: $N < S^n$, где N-количество кодируемых объектов; S-основание кода; n- количество разрядов в коде.

Так для хранения информации в виде текстов, формул и чисел необходимо закодировать в помощью бит около 150 различных символов. Для этого необходим восьмиразрядный код ($150 < 2^8 = 256$). Восьмиразрядный код называют байтом. Объемы памяти в компьютерах оценивают в килобайтах (Кбайт), мегабайтах (Мбайт), гигабайтах (Гбайт).

9.2. Функции алгебры логики

Цифровое устройство преобразует входную информацию, представленную в виде двоичного кода (нулей и единиц) в значение выходной функции $y(x_1, x_1, ..., x_n)$ (рис.9.2). В цифровых комбинационных устройствах выходные сигналы однозначно определяются только действующей в настоящий момент комбинацией входных переменных и не зависят от значений переменных, действовавших на входе ранее.



Однозначно выразить выходную функцию можно только задав ее значения (0 или 1) для всех комбинаций аргументов. Число различных наборов из n-величин, каждая из которых принимает два значения (0 или 1) равно 2^n. Следовательно, для функции двух переменных должно быть задано 4 значения выходной величины, для функции 3 переменных – 8 значений и т.д.

В цифровых электронных устройствах применяют ***положительную*** потенциальную логику, в которой символ «1» кодируется высоким потенциалом, а «0» низким, и ***отрицательную*** , в которой символ «1» кодируется отрицательным потенциалом, а «0» - близким к нулю. Далее мы будем изучать положительную логику.

В общем случае логическое устройство может иметь n-входов и m-выходов. Рассматривая входные сигналы $X_1, X_1, ..., X_n$ в качестве аргументов, можно соответствующие выходные сигналы представлять в виде функций $y_i(X_1, X_1, ..., X_n)$ с помощью операций алгебры логики.

Решение логических задач осуществляется с математического аппарата ***алгебры логики*** (булевой алгебры, разработанной английским математиком Джорджем Булем (1815-1864)), в которой все переменные величины могут принимать только два логических значения: «1»-логическая единица («правда») и «0» - логический нуль («ложь»).

Функции алгебры логики (ФАЛ), которые называют также переключающими функциями, представляют в нескольких формах:

- в виде таблиц ***истинности*** или ***комбинационных*** таблиц;
- в алгебраической форме (в виде математического выражения);
- комбинационной схемой, составленной из логических элементов;
- координатным способом (картой Карно).

Булева функция называется полностью определенной, если заданы 2n ее значений для каждой входной комбинации аргументов. Если чать значений функции не задана, функцию называют недоопределенной.

9.2.1. Таблица истинности

Таблицы истинности содержат всевозможные комбинации (наборы) бинарных значений входных переменных с соответствующими им бинарными значениями выходных переменных. Каждому набору входных сигналов соответсвует определенное значение выходной логической функции y_i. Число строк в таблице зависит от числа входных переменных n и равно 2^n.

В таблице 9.1 показан пример таблицы истинности комбинационного логического устройства с тремя входными сигналами.

X_1	X_2	X_3	Y=f(X)
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

9.2.2. Аналитическая запись логических операций

При аналитической записи логические операции обозначают специальными символами:

Логическое отрицание (инверсию) обозначают чертой над переменной $\overline{A}, \overline{x}$.

Логическое сложение (дизъюнкцию) обозначают знаком «$\vee$» или «+».

Логическое умножение (конъюнкцию) обозначают знаком «$\wedge$» или «$\cdot$».

Мы будем использовать более удобные знаки «+» и «$\cdot$».

Три перечисленные простейшие логические операции (логическое отрицание «НЕ», логическое сложение «ИЛИ», логическое умножение «И») представляют собой полный набор логических действий, через которые могут быть выражены логические связи, заданные таблицей истинности.

Таблица истинности простейших логических функций

И: $f = x_1 \cdot x_2$			ИЛИ: $f = x_1 + x_2$			НЕ: $f = \overline{x}_1$	
x_1	x_2	f	x_1	x_2	f	x_1	f
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Алгебраическая форма представления булевых функций используется для минимизации (упрощения формул) и для построения логических схем.

Существуют две формы алгебраических функций- дизъюнктивная и конъюктивная.

Дизъюнктивной нормальной формой (ДНФ) называют логическую сумму элементарных логических произведений, в каждое из которых аргумент или его инверсия входит один раз.

ДНФ получить из таблицы истинности с использованием следующего алгоритма:

а) для каждой входной кодовой комбинации, при которой булева функция равна единице, записывают элементарные логические произведения входных переменных. При этом, если входная переменная в кодовой комбинации равна нулю, то её записывают с инверсией. Полученные произведения называют **конституентами** единицы **(минтермами).**

б) логически суммируют все конституенты единицы.

Совершенная дизъюнктивная нормальная форма (СДНФ) получается суммированием конституент единицы.

Для таблицы истинности 9.1 имеем следующие элементарные логические произведения:

$$\overline{x}_1\overline{x}_2\overline{x}_3;\ \ \overline{x}_1 x_2\overline{x}_3;\ \ x_1\overline{x}_2\overline{x}_3.$$

Суммируя логические произведения, получаем алгебраическое выражение СДНФ:

$$f(x) = \overline{x}_1\overline{x}_2\overline{x}_3 + \overline{x}_1 x_2\overline{x}_3 + x_1\overline{x}_2\overline{x}_3.$$

Конъюнктивной нормальной формой (КНФ) называют логическое произведение логических сумм, в каждую из которых аргумент или его инверсия входят один раз.

КНФ может быть получена из таблицы истинности с использованием следлующего алгоритма:

а) Для каждой входной комбинации, при которой булева функция равна нулю, записывают элементарные логические суммы входных переменных. При этом, если входная переменная в кодовой комбинации равна единице, то ее записывают с инверсией. Полученные суммы называют конституентами нуля (макстермами).

б) Логически перемножают все полученные конституенты нуля.

Совершенная конъюктивная нормальная форма (СКНФ) получается перемножением конституент нуля.

Для таблицы 9.1 имеем следующие конституенты нуля:

$$x_1 + \overline{x}_2 + \overline{x}_3;\ \overline{x}_1 + x_2 + \overline{x}_3;\ \overline{x}_1 + \overline{x}_2 + x_3\ ;\overline{x}_1 + \overline{x}_2 + \overline{x}_3$$

Перемножаем суммы и получаем алгебраическое выражение СКНФ:

$$f(x)=(x_1+x_2+\overline{x}_3)(\ x_1+\overline{x}_2+\overline{x}_3)(\ \overline{x}_1+x_2+\overline{x}_3)(\ \overline{x}_1+\overline{x}_2+x_3\ )(\ \overline{x}_1+\overline{x}_2+\overline{x}_3)$$

Законы и теоремы булевой алгебры

В алгебре логики существуют теоремы, знание которых облегчает действия с логическими переменными.

1. Коммутативный (переместительный) закон:

$x_1 \cdot x_2 = x_2 \cdot x_1;\ \ x_1 + x_2 = x_2 + x_1.$

2. Ассоциативный (сочетательный) закон:

$x_1 \cdot (x_2 \cdot x_3) = (x_1 \cdot x_2) \cdot x_3;\ \ \ x_1 + (x_2 + x_3) = (x_1 + x_2) + x_3\ .$

3. Дистрибутивный (распределительный) закон:

$x_1 \cdot (x_2 + x_3) = x_1 \cdot x_2 + x_1 \cdot x_3\ ;$

$(x_1 + x_2)(x_1 + x_3) =$

$= x_1x_1 + x_1x_3 + x_2x_1 + x_2x_3 = x_1(1 + x_3 + x_2) + x_2x_3 = x_1 + x_2x_3$

Здесь учтено: $xx = x; 1 + x = 1.$

4. Закон поглощения:

$x_1 + x_1 x_2 = x_1(1 + x_2) = x_1;$

$x_1(x_1 + x_2) = x_1 x_1 + x_1 x_2 = x_1 + x_1 x_2 = x_1(1 + x_2) = x_1$

5. Теорема де Моргана:

$\overline{x_1 \cdot x_2} = \overline{x}_1 + \overline{x}_2;$

$\overline{x_1 + x_2} = \overline{x}_1 \cdot \overline{x}_2.$

Используется при переходе от логического произведения к логической сумме и обратно

6. Правило повторения:

$x \cdot x = x; \;\; x + x = x.$

7. Правило отрицания:

$x \cdot \overline{x} = 0; \;\; x + \overline{x} = 1.$

8. Правило двойного отрицания:

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

9. Логические соотношения в случае определенности одного из аргументов:

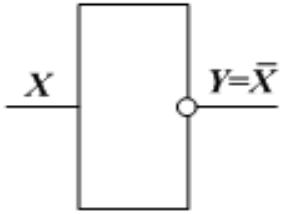
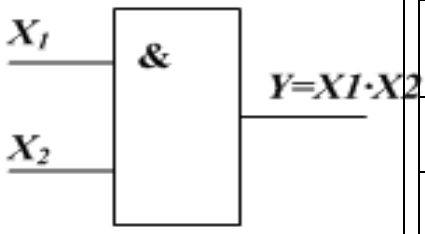
$$x_1 \cdot 1 = x_1; \quad x_1 + 1 = 1;$$
$$x_1 \cdot 0 = 0; \quad x_1 + 0 = x_1.$$

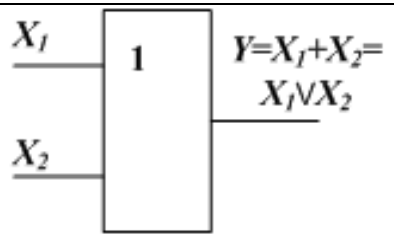
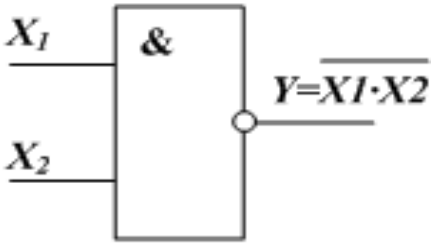
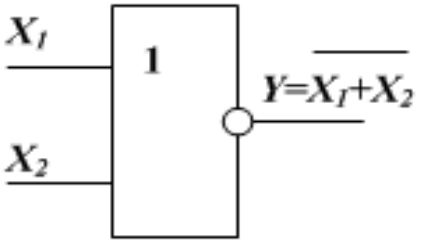
10. Инверсия нуля и единицы:

$$\overline{0} = 1; \quad \overline{1} = 0.$$

9.3. Цифровые логические элементы

Цифровые логические элементы, выполненные на интегральных микросхемах (ИМС), предназначены для преобразования и обработки дискретных сигналов и выполняют основные логические функции, представленные в таблице 9.3.

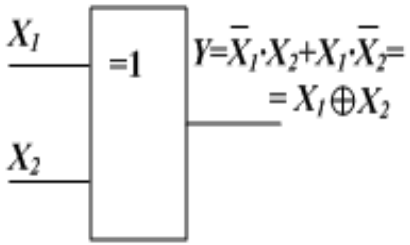
<table>
<tr><th>Элемент</th><th>Обозначение отечественных ИМС</th><th>Выполняемая функция и схема</th><th colspan="4">Таблица истинности</th></tr>
<tr><td>НЕ (отрицание)</td><td>ЛН</td><td> 
</td><td>x</td><td colspan="3">$y = \bar{x}$</td></tr>
<tr><td></td><td></td><td></td><td>1</td><td colspan="3">0</td></tr>
<tr><td></td><td></td><td></td><td>0</td><td colspan="3">1</td></tr>
<tr><td>И (логическое умножение)</td><td>ЛИ</td><td> 
</td><td>x_1</td><td>x_2</td><td>y</td></tr>
<tr><td></td><td></td><td></td><td>0</td><td>0</td><td>0</td></tr>
<tr><td></td><td></td><td></td><td>1</td><td>0</td><td>0</td></tr>
<tr><td></td><td></td><td></td><td>0</td><td>1</td><td>0</td></tr>
<tr><td></td><td></td><td></td><td>1</td><td>1</td><td>1</td></tr>
</table>

ИЛИ (логическое сложение)	ЛЛ		
И-НЕ (логическое умножение с отрицанием) (Штрих Шефера)	ЛА		
ИЛИ-НЕ (логическое сложение с отрицанием) (Стрелка Пирса)	ЛЕ		

x_1	x_2	y
0	0	0
1	0	1
0	1	1
1	1	1

x_1	x_2	y
0	0	1
1	0	1
0	1	1
1	1	0

x_1	x_2	y		
0	0	1		
1	0	0		
0	1	0		
1	1	0		

Исключающее ИЛИ	ЛП		

x_1	x_2	y		
0	0	0		
1	0	1		
0	1	1		
1	1	0		

В таблице 9.3 использованы обозначения: $\overline{x}$ - отрицание значения x; $x_1 \cdot x_2$-логическое умножение (конъюнкция); $X_1 + X_2 = X_1 \vee X_2$- логическое сложение (дизъюнкция).

Минимизация логических функций в программе TINA

Целью минимизации является получение минимального необходимого количества элементов в схеме. Причём, устройство, выполняющее любую логическую функцию, можно построить, имея элементы только одного вида («ИЛИ-НЕ» либо «И-НЕ»). Эти элементы называют ***универсальными базисами***.

Для минимизации логических элементов применяют диаграммы Карно-Вейча.

Ранее для таблицы истинности 9.1 мы получили СДНФ в виде:

$$f(x) = \overline{x}_1\overline{x}_2\overline{x}_3 + \overline{x}_1 x_2\overline{x}_3 + x_1\overline{x}_2\overline{x}_3. \qquad (9.3)$$

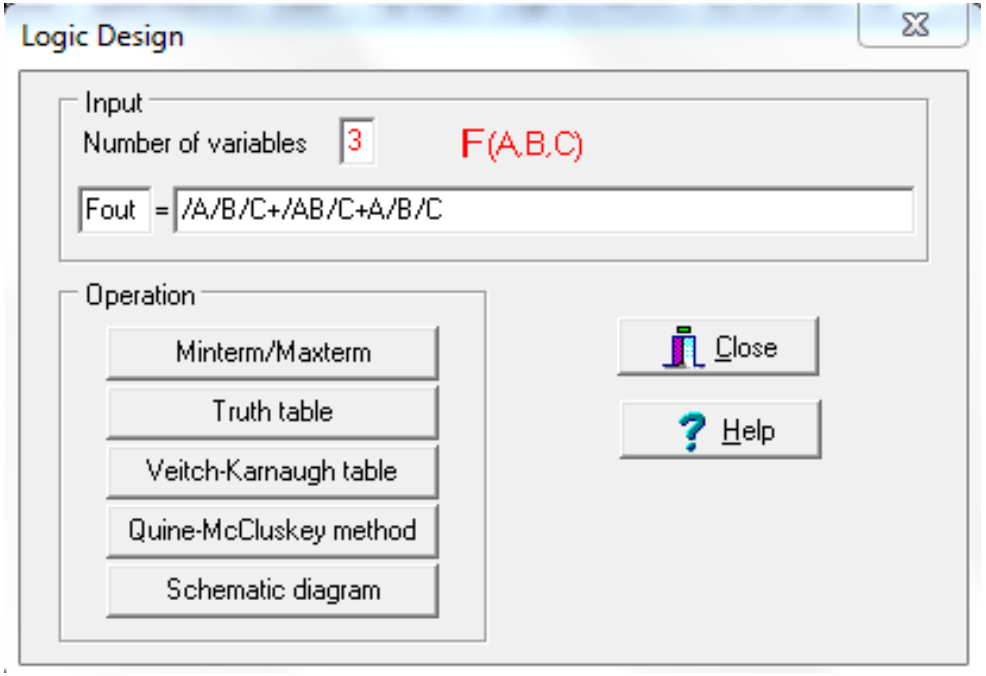
Карта Карно-графическое представление всех минитермов (2^n) для данного числа переменных (n). Каждый минитерм изображается в виде клетки, расположенной так, что минитермы, находящиеся в соседних клетках, отличаются друг от друга только одной переменной.

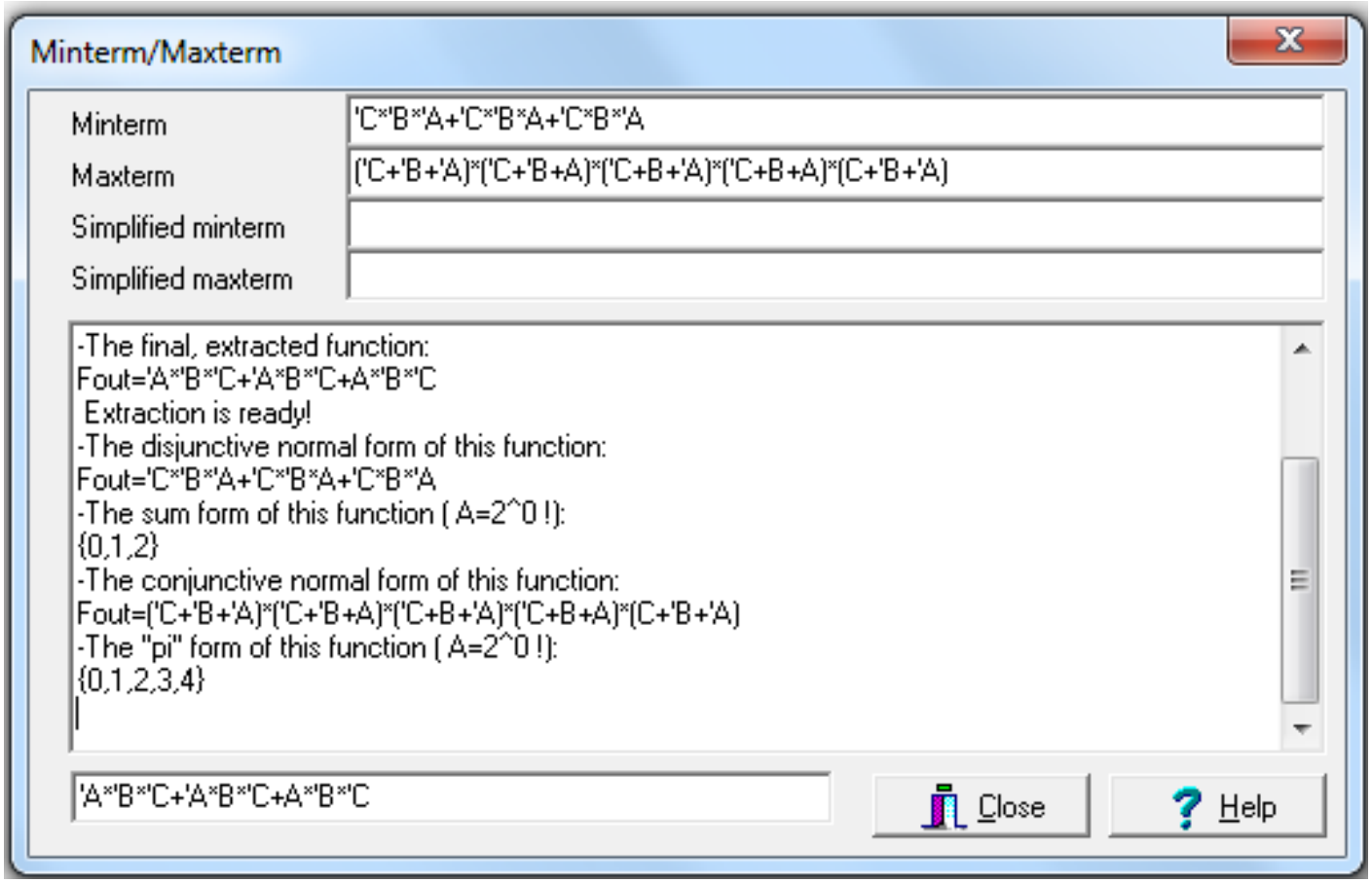
Составлять диаграммы Карно-Вейча в ручную достаточно долго и трудоемко. Поэтому мы воспользуемся программой моделирования TINA.

В СДНФ (9.3) изменим обозначения входных переменных: x_1=A, x_2=B, x_3=C. Перед инверсными переменными будем ставить наклонную черту «/». Вместо наклонной черты можно ставить верхний штрих «’».

В меню Tools выбираем инструмент Logic Design, устанавливаем Number of variables – 3 и набираем функцию алгебры логики (ФАЛ) (рис.9.3).

Нажимаем Minterm/Maxterm и получаем алгебраические выражения СДНФ и СКНФ.

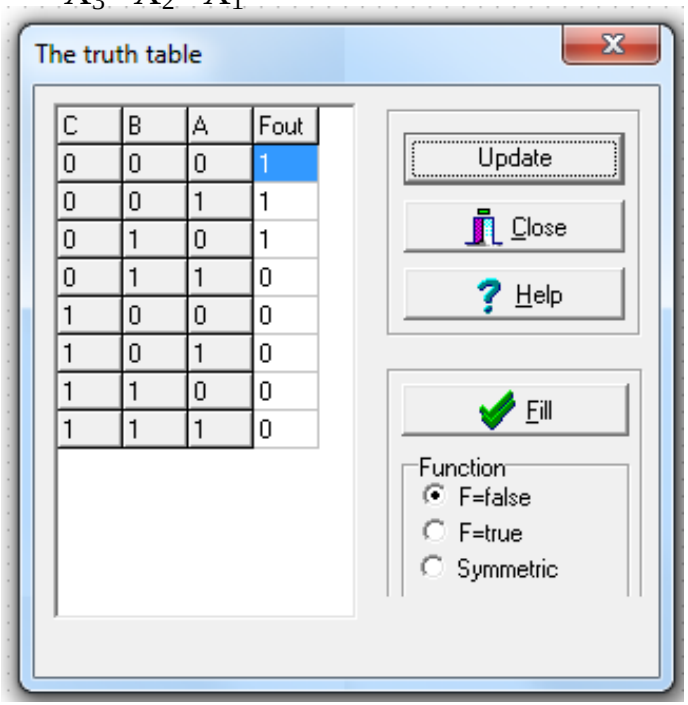




Далее нажимаем Truth table и получаем для проверки исходную таблицу истинности (рис.9.4). Сравнивая с исходной таблицей 9.1 с учетом нового расположения входных переменных, отмечаем совпадение таблиц.

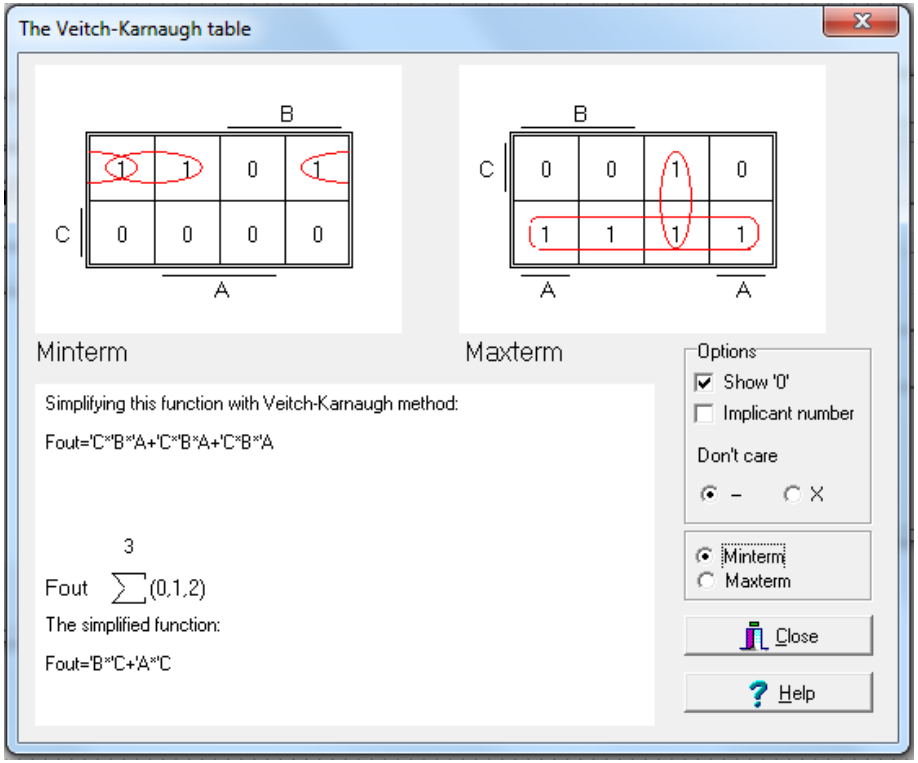
В последнем столбце таблицы могут быть записаны другие значения выходной функции, для которых требуется создать логическую схему.

X_3 X_2 X_1

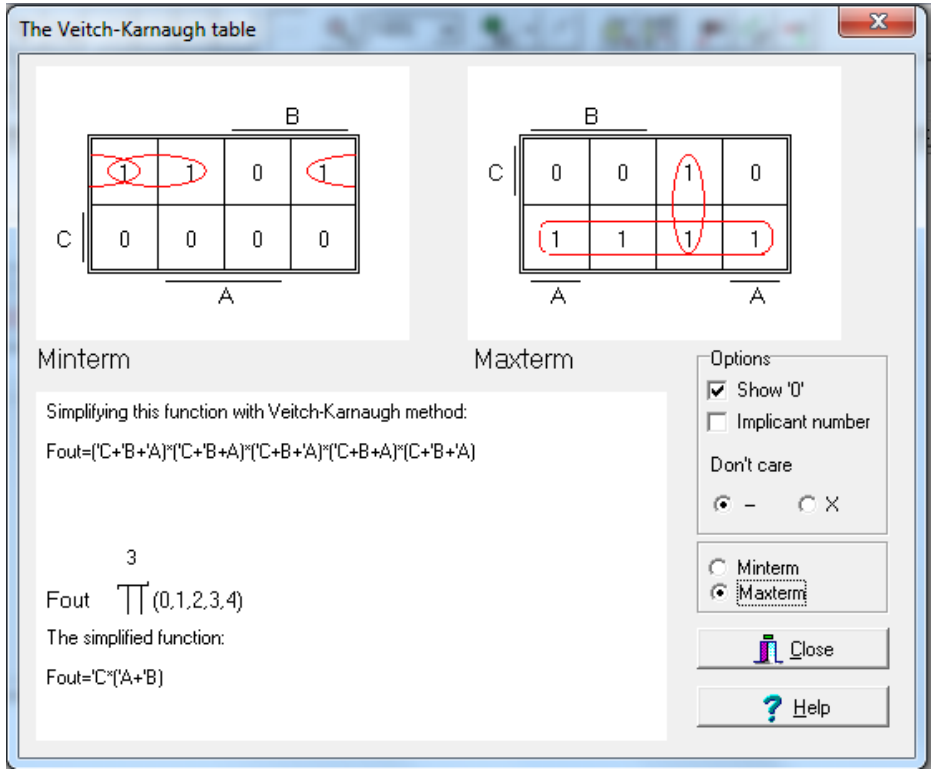


Нажав кнопку Veitch-Karnaugh table, вы получите возможность выполнить операцию графической минимизации ФАЛ (рис.9.5). Если нажать кнопку Minterm, получим выраженную через минитермы упрощенную выходная СДНФ (рис.9.5а):

СДНФ: $f(x) = \bar{B} \cdot \bar{C} + \bar{A} \cdot \bar{C}$



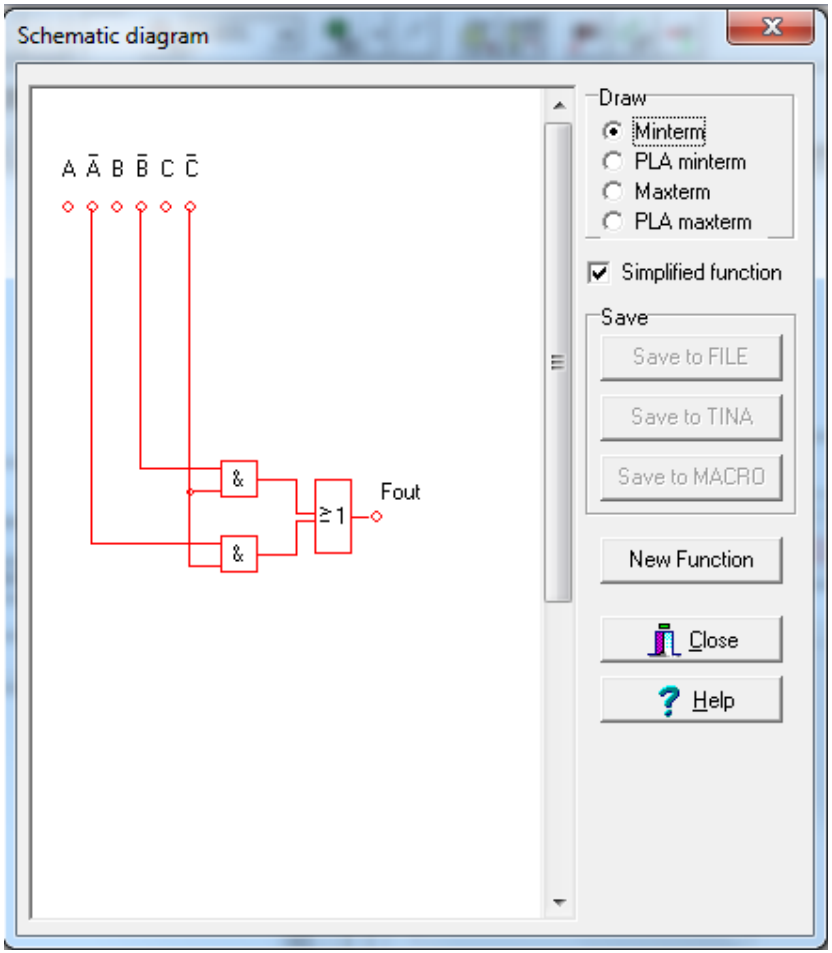
СДНФ

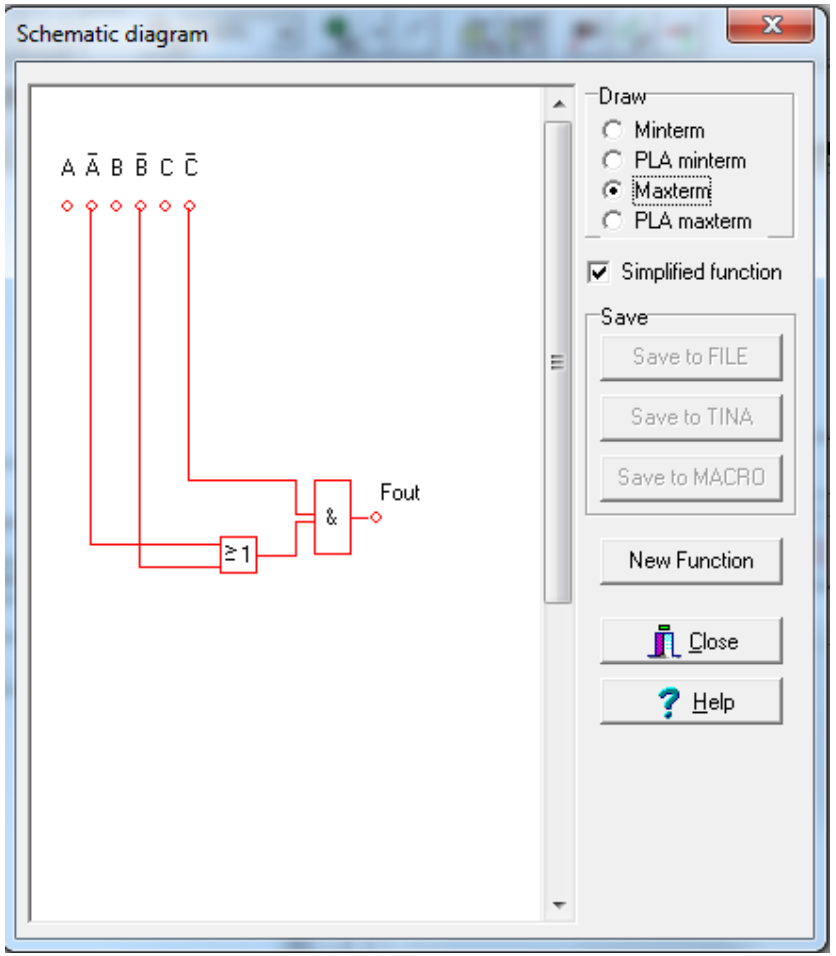


СКНФ

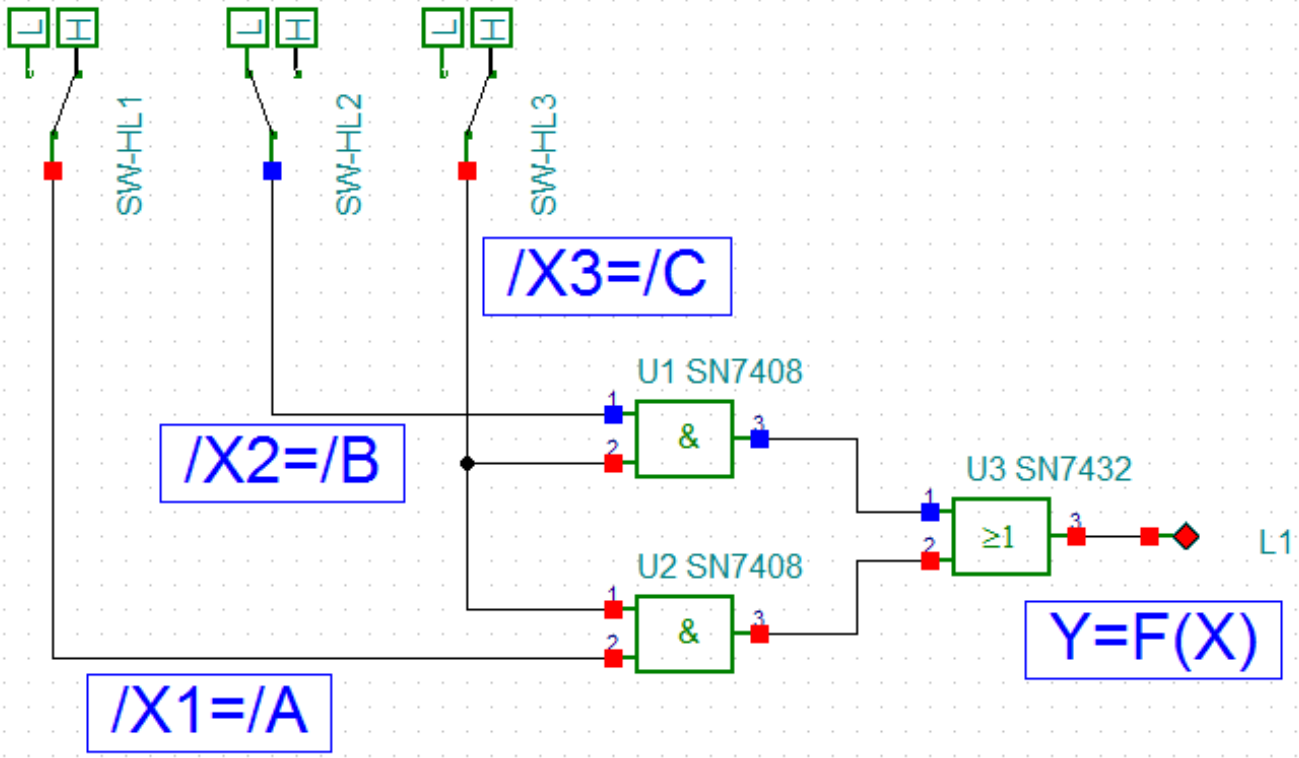
СКНФ: $f(x) = \bar{C} \cdot (\bar{A} + \bar{B})$

И, наконец, нажав Schematic Diagram и выбрав Minterm, получим схемную реализацию ФАЛ на элементах 2И, 2ИЛИ (рис.9.6а). Если выбрать Maxterm, получим схемную реализацию на элементах 2ИЛИ, 2И (рис.9.6б).





Модель по СДНФ



Схемная реализация ФАЛ

Обозначения: $\bar{X}_1 = \bar{A}, \bar{X}_2 = \bar{B}, \bar{X}_3 = \bar{C}$

Для третьей строки ТИ:

$$\bar{X}_1 = \bar{A} = 1, \bar{X}_2 = \bar{B} = 0, \bar{X}_3 = \bar{C} = 1. \quad Y = F(X) = 1$$

Ключевые схемы

Ключевые схемы предназначены для формирования на выходе цифровых сигналов высокого или низкого уровня под действием переключающих входных сигналов.

Диодные ключи

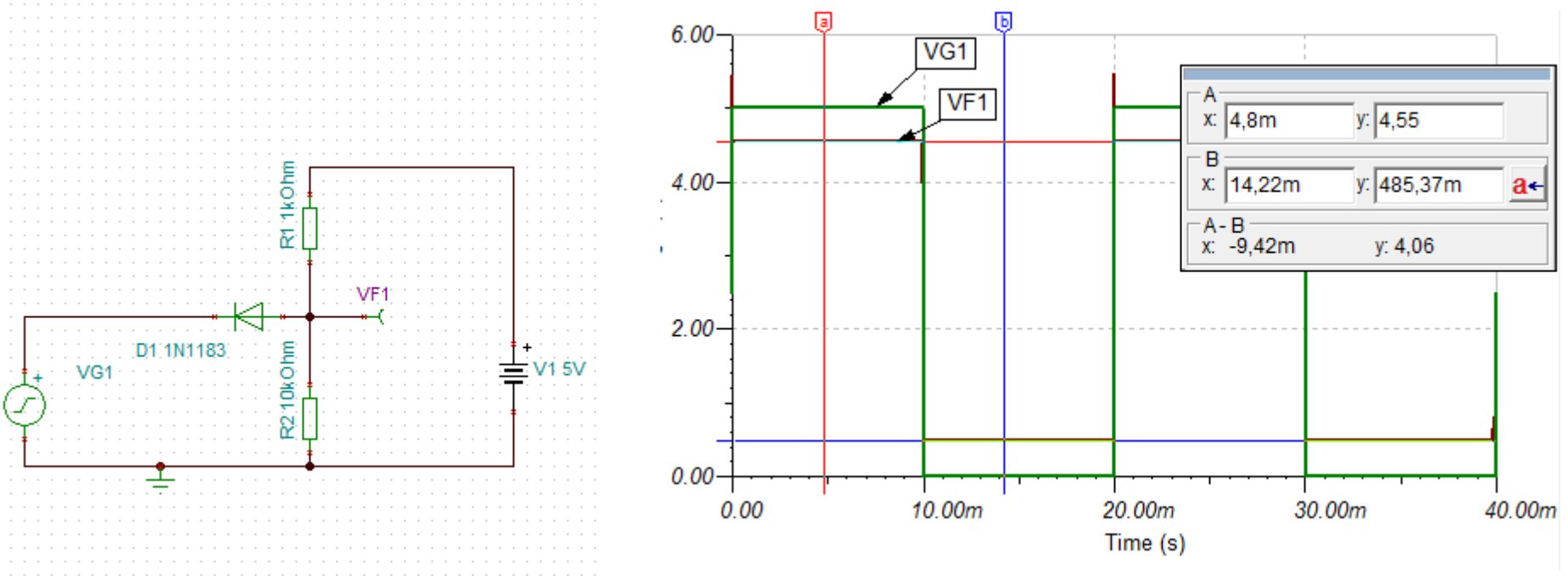


Рис.10.1

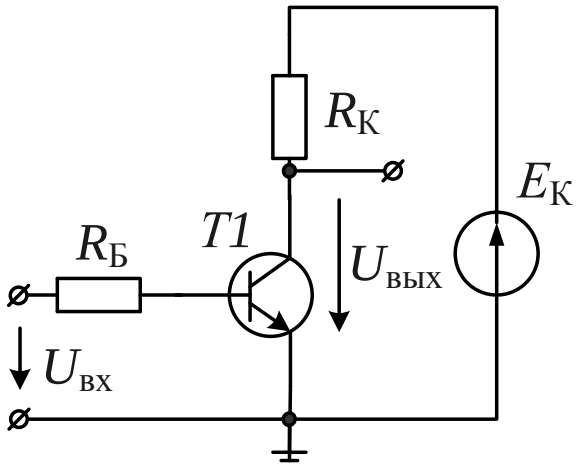
Если $U_{вх} = 0, U_{вых} = U_{пр} = 0,5 - 0,7\,B.$

Если $U_{вх} = +E,$ диод закрыт, $U_{вых} = \dfrac{ER_2}{R_1 + R_2} \approx 0{,}9E,$

$(R_2 \gg R_1)$.

Время переключения 10 пс – 1нс. Используют в быстродействующих устройствах. Недостаток: большая мощность переключения.

Ключи на биполярных транзисторах

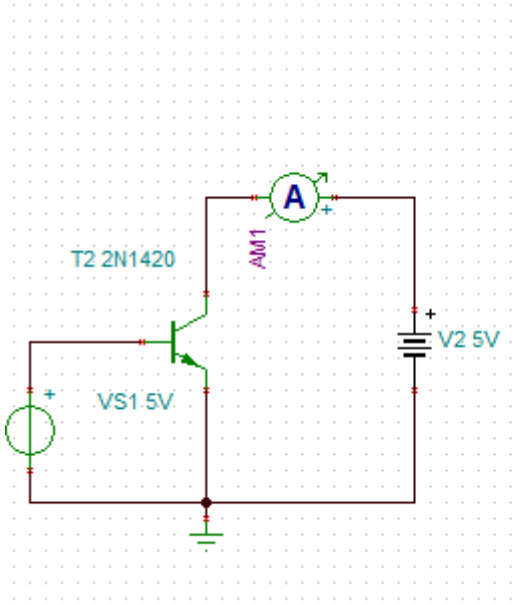


Дано: $E_К = 5B,\ R_Б = 10кОм, R_К = 100Ом$

Низкий уровень $U_{вх0} = 0{,}8B.$

Высокий уровень $U_{вх1} = 4B,$

Расчет входной цепи



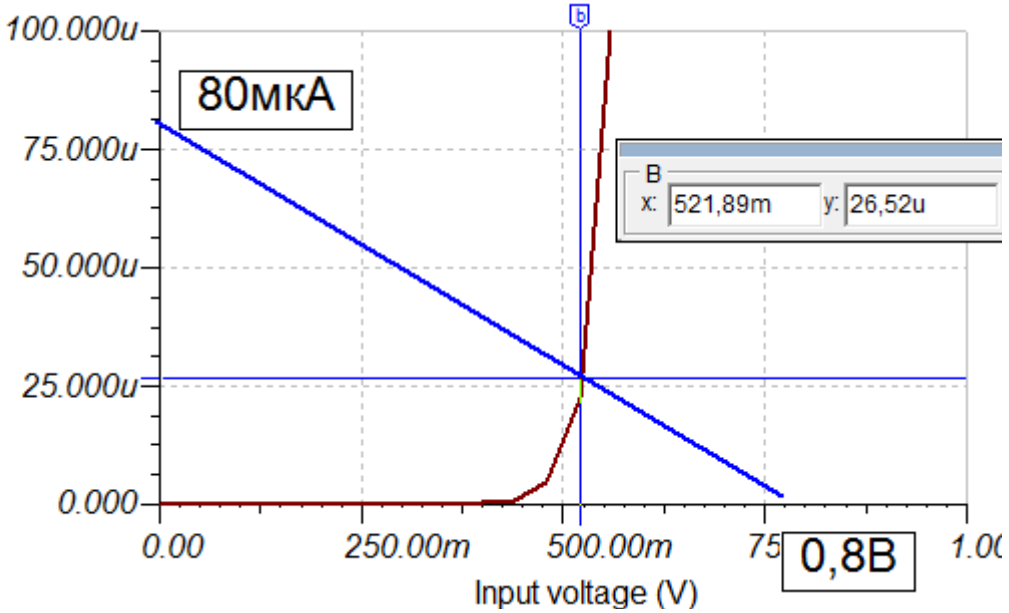
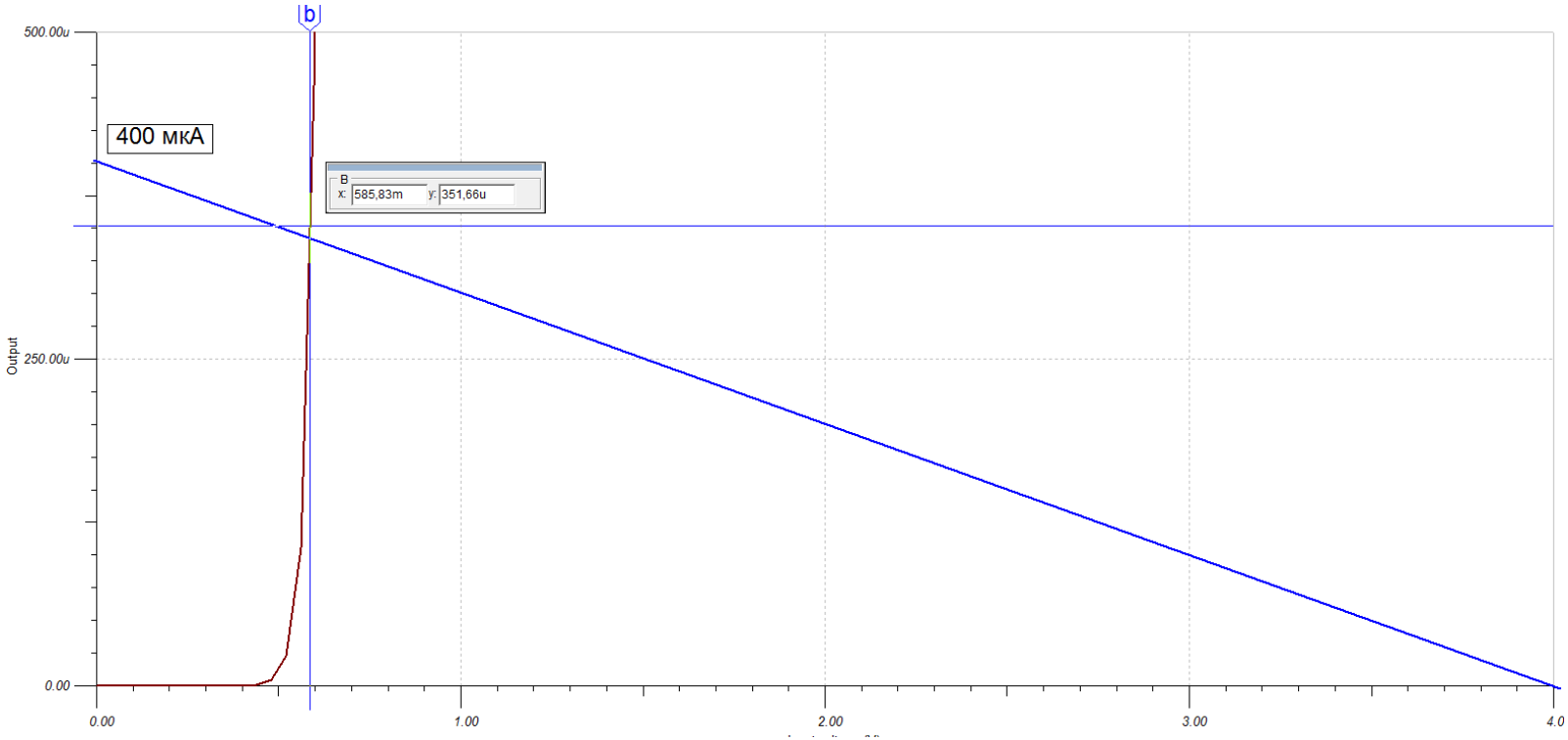


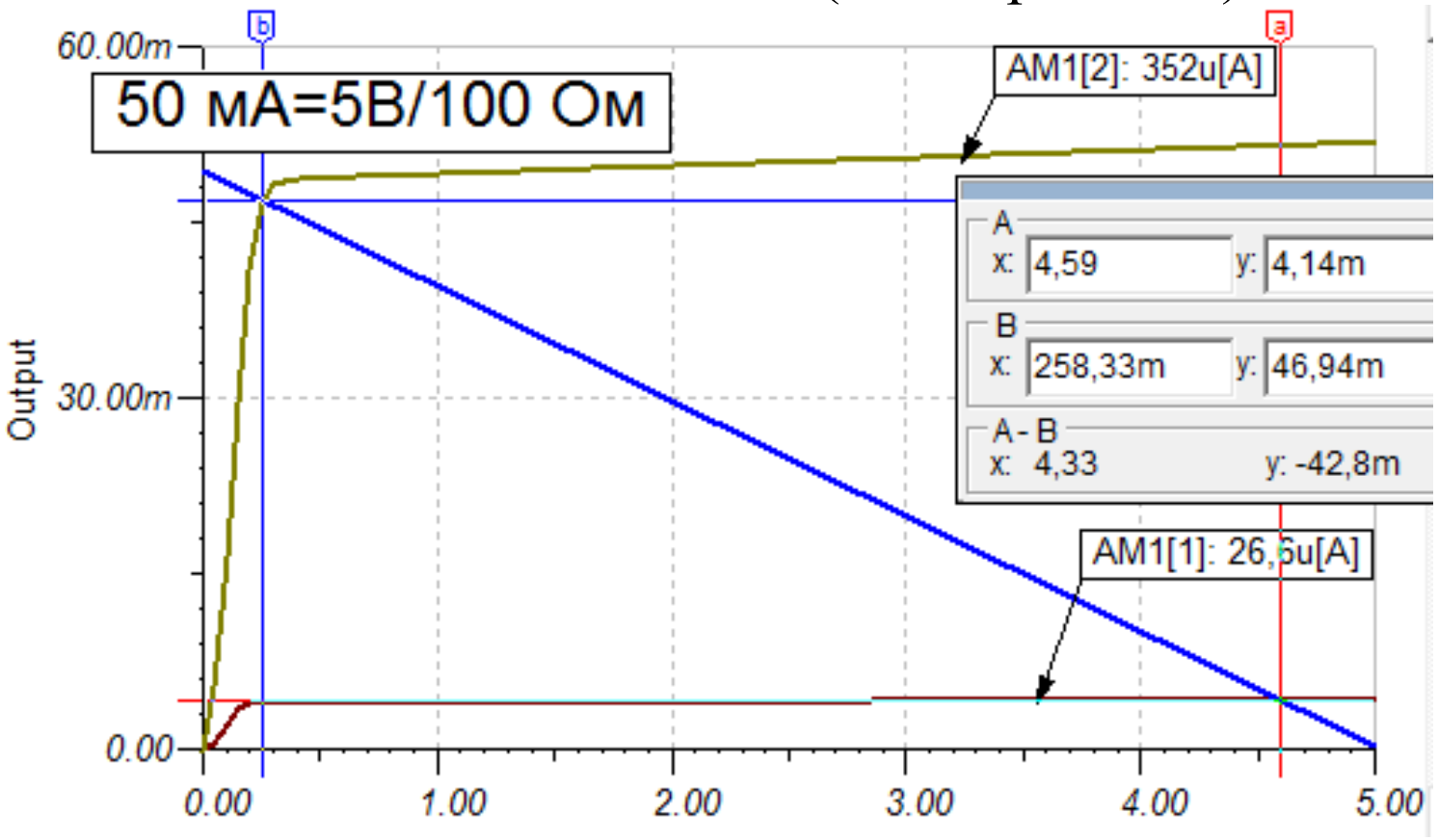
Рис.10.2

Низкий уровень $i_{Б0} = 26{,}6\,мкА.$



Высокий уровень: $i_{Б1} = 352\,мкА.$

Расчет выходной цепи (схема рис.10.4)



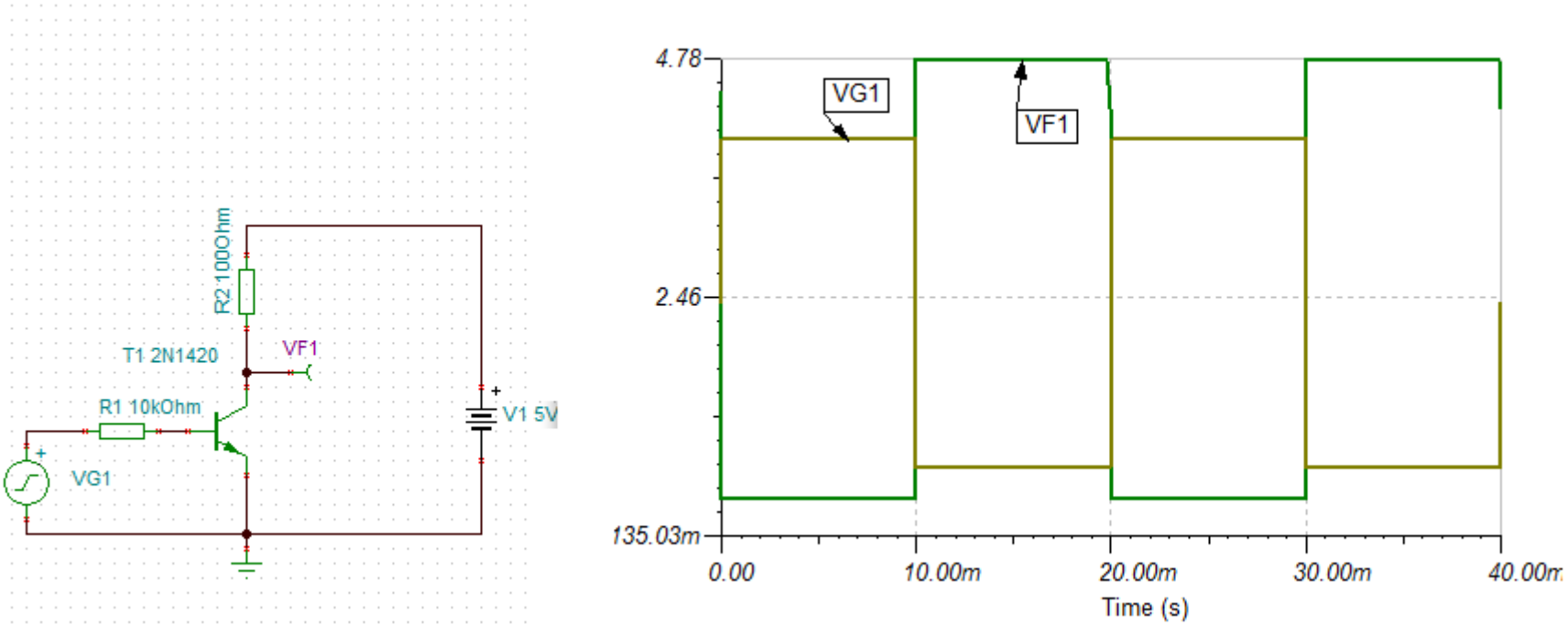


Рис.10.4

Транзисторный ключ инвертирует входной сигнал.

Недостатки:

Мощность, потребляемая от источников.

Низкий уровень:

$$P^0_{вх} = U_{вх0} \cdot i_{Б0} = 0{,}8 \cdot 26{,}6 \cdot 10^{-6} = 20{,}8\,мкВт$$

$$P^0_{Ек} = E_к i_{к\,min} = 5 \cdot 4{,}14 \cdot 10^{-3} = 20{,}7\,мВт$$

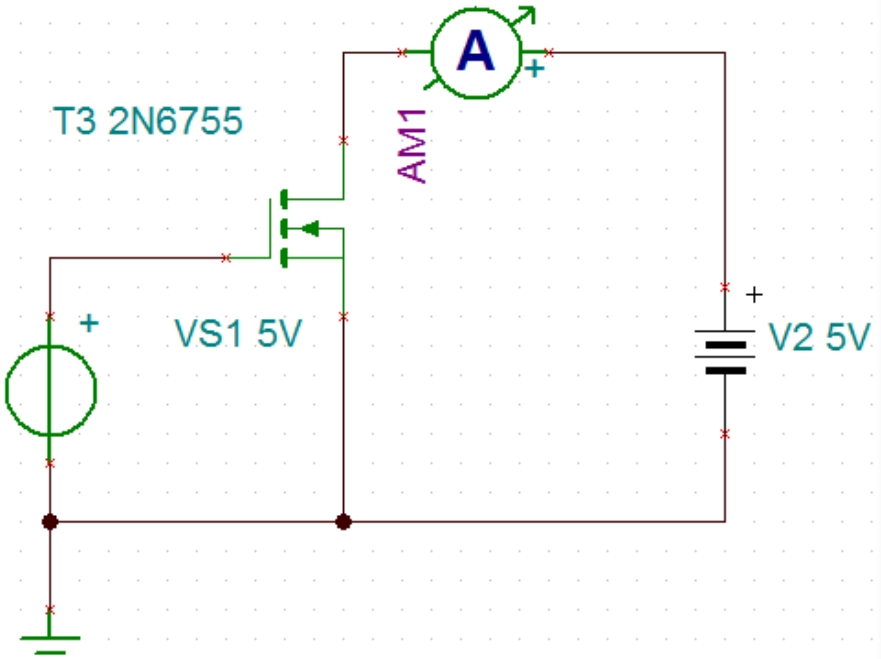
Высокий уровень:

$$P^1_{вх} = U_{вх1} \cdot i_{Б1} = 4 \cdot 352 \cdot 10^{-6} = 1{,}4\,мВт;$$

$$P^1_{Ек} = E_к i_{к\,max} = 5 \cdot 46{,}9 \cdot 10^{-3} = 243\,мВт.$$

Ключи на полевых транзисторах

Применяют комплементарные полевые транзисторы с изолированным затвором и индуцированным каналом.



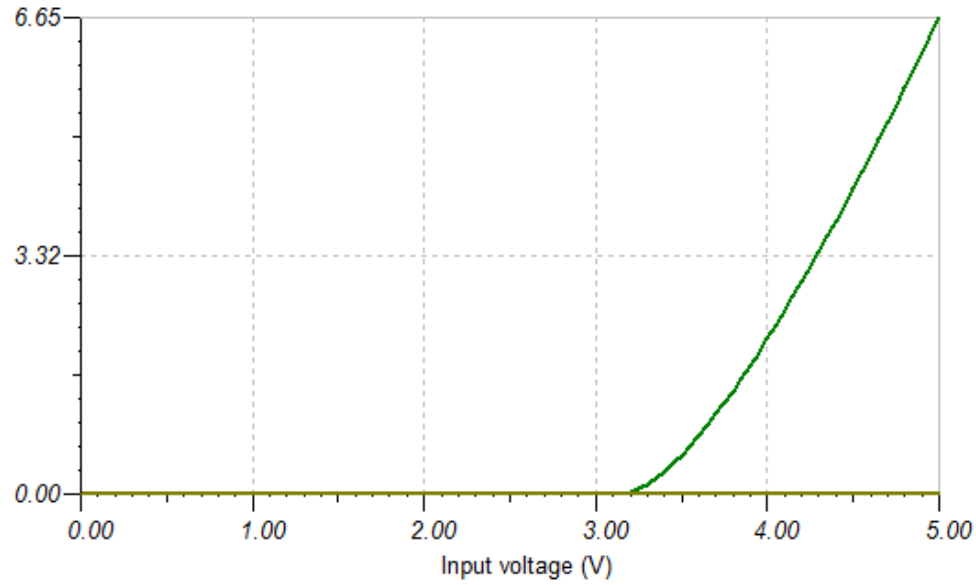
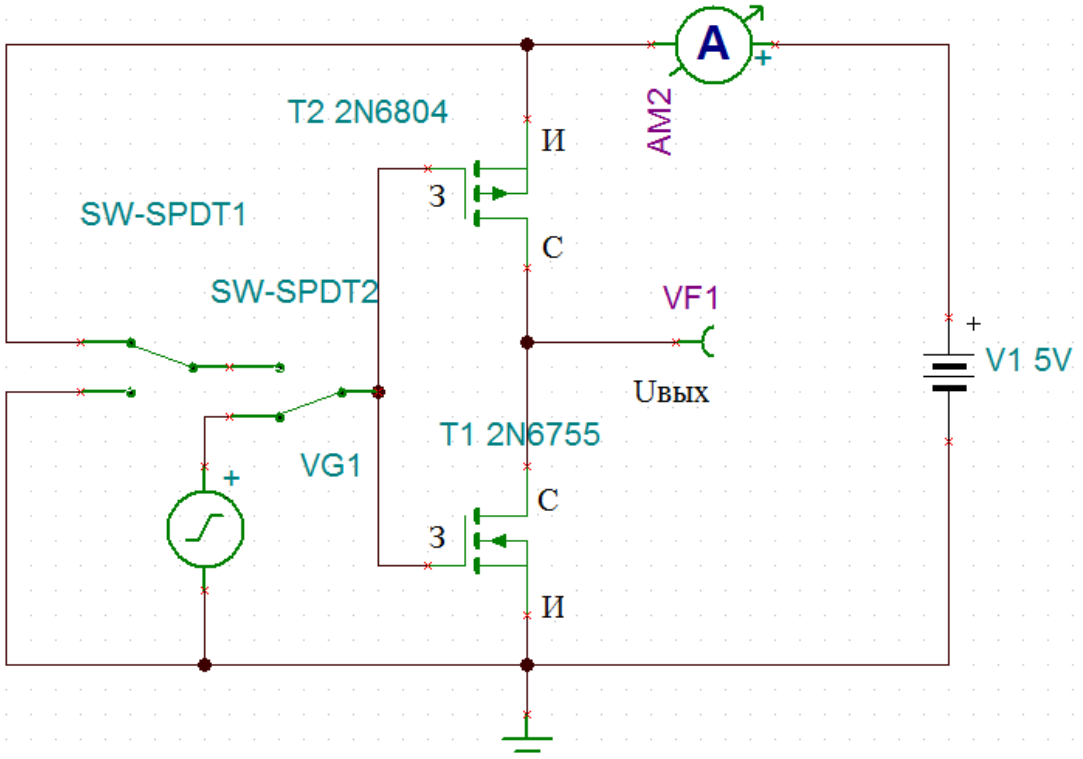
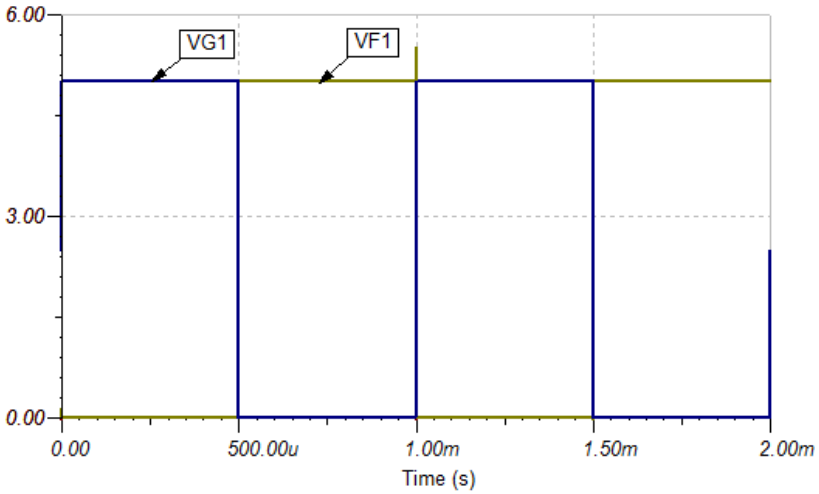


Рис.10.5





Если $U_{вх0} = 0 < U_{порТ1}$, Т1 выключен.

$U_{зиТ2} = U_{вх0} - E = -E$, Т2 включен, $U_{вых} \approx E$.

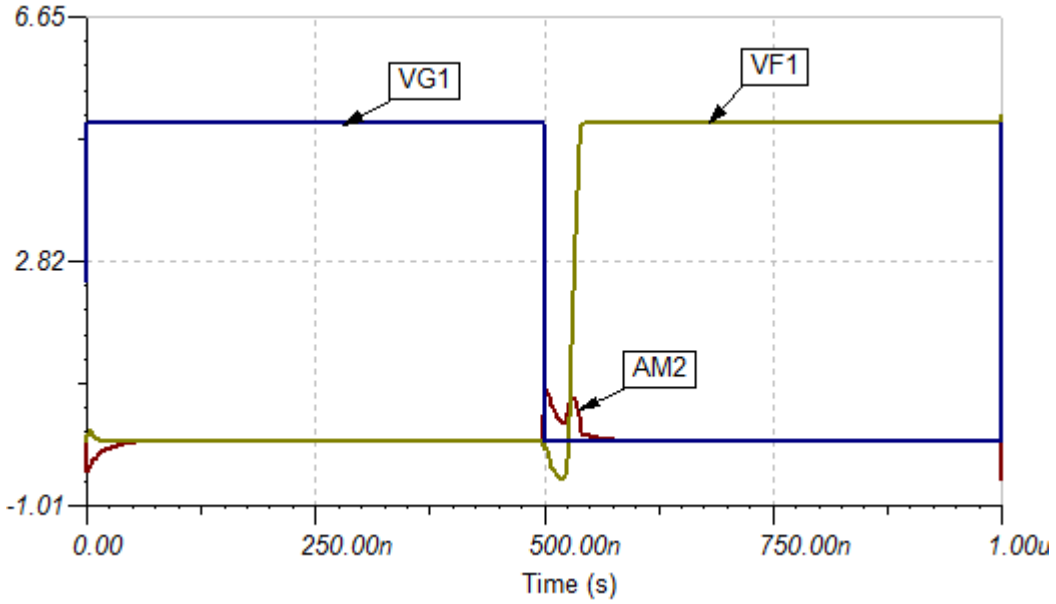
Если $U_{вх1} = E > U_{порТ1}$, Т1 включен.

$U_{зиТ2} = U_{вх1} - E = E - E = 0$. Т2 выключен. $U_{вых} \approx 0$.

Получили инвертер на КМОП.

Достоинства ключей на МОП транзисторах

1. Технологичность – изготавливаются в одном цикле с другими МОП элементами.
2. Высокая плотность упаковки.
3. Нет резисторов.
4. Низкое потребление мощности (только при переключении), очень малый ток.



Структура базового элемента И-НЕ

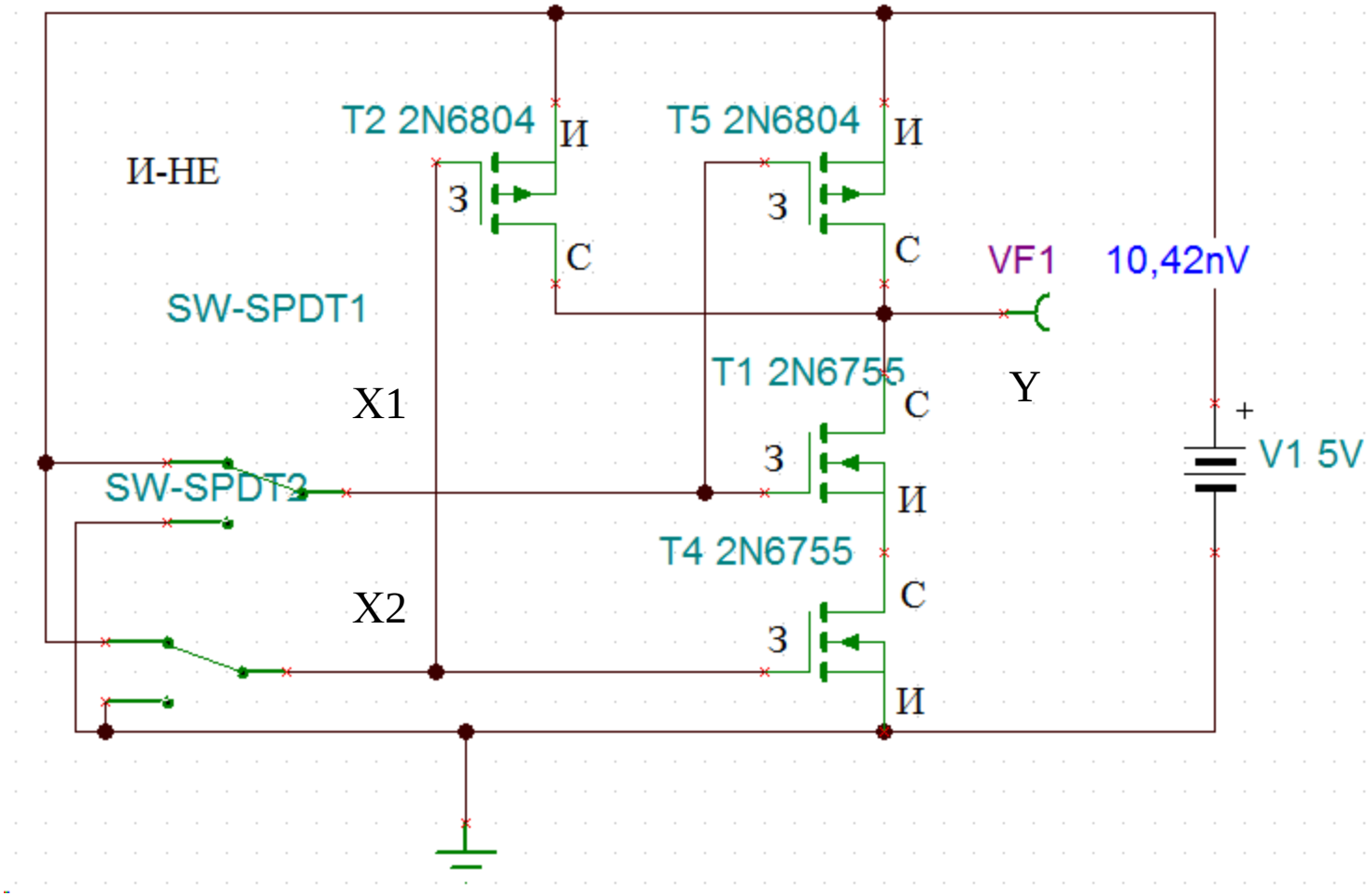


Рис.10.6

Структура базового элемента ИЛИ-НЕ

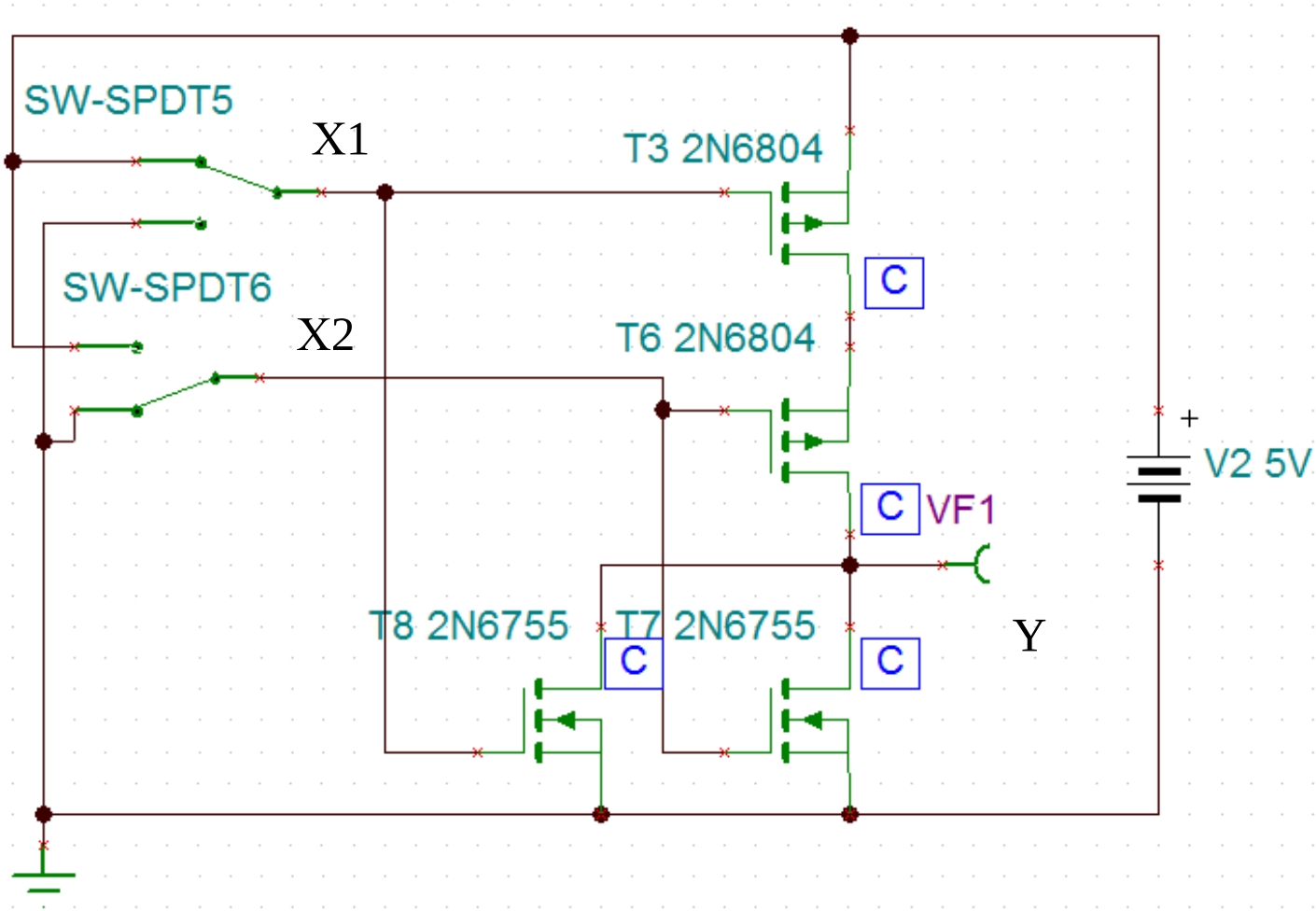


Рис.10.7

Функционально полные системы

Это набор логических элементов, с помощью которых можно построить любые комбинационные схемы с разными логическими функциями.

1. И, ИЛИ, НЕ
2. И-НЕ
3. ИЛИ-НЕ

Типы серийных логических элементов

Цифровые интегральные схемы бывают:

1. ИС малой степени интеграции с числом транзисторов <100.
2. ИС средней степени интеграции $n \sim 10^3$.
3. БИС – большие интегральные схемы $n \sim 10^4 \div 10^5$.
4. СБИС – сверхбольшие интегральные схемы $n > 10^6$.

Наиболее популярные ИС малой степени интеграции

ТТЛ – транзисторно-транзисторная логика. Входная логика на многоэмиттерных транзисторах, ключи с динамической нагрузкой. Е=5В.

ТТЛШ – транзисторно-транзисторная логика с диодами Шоттки. Большое быстродействие.

n-МОП логика. Все элементы по МОП транзисторах. Не высокое быстродействие, малое потребление.

КМОП – логика на комплементарных МОП транзисторах. Рабочее напряжение от 3 до 15 В. Сверхнизкое потребление мощности. Высокое быстротействие.

Примеры комбинационных схем
И-И-НЕ

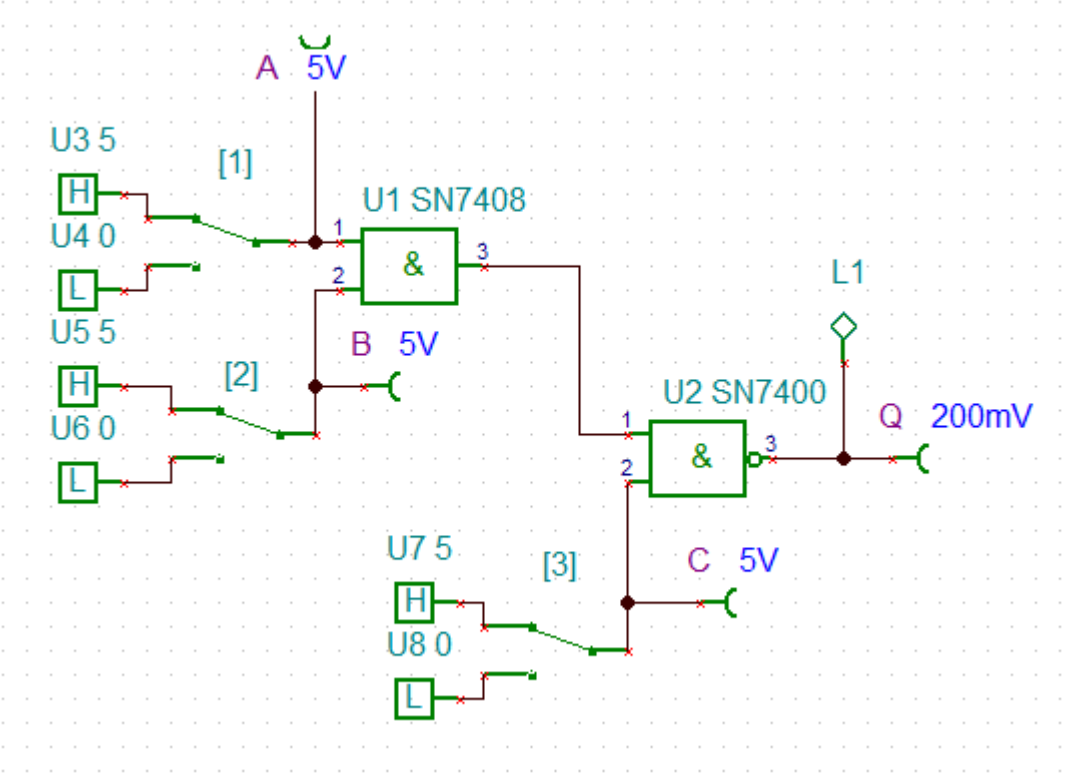
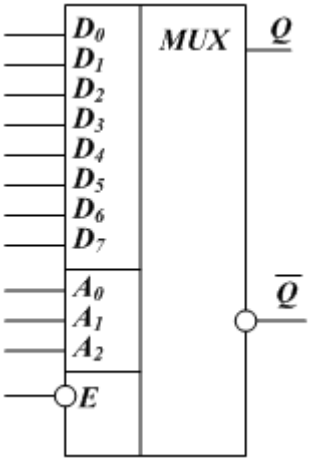
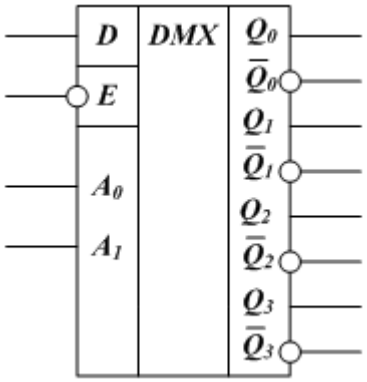
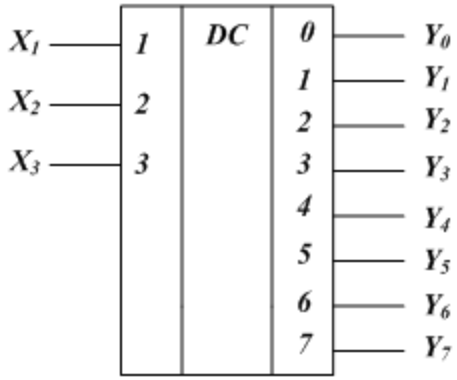


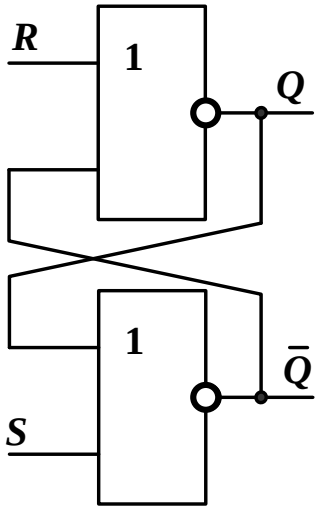
Рис.10.8

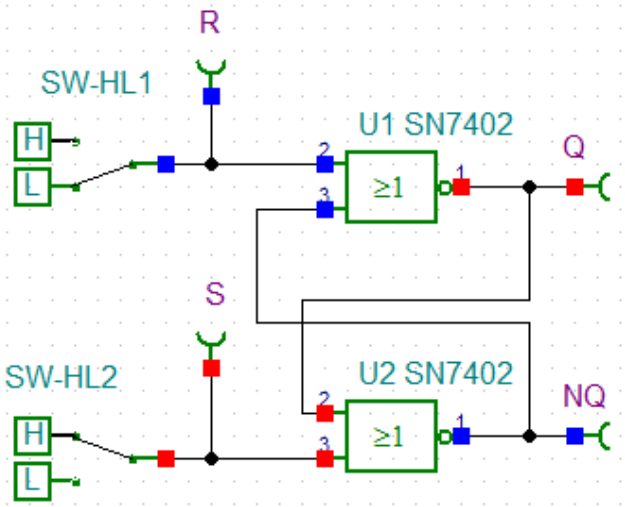
	
Условное графическое изображение мультиплексора (8→1)	Условное графическое изображение демультиплексора (1→4)

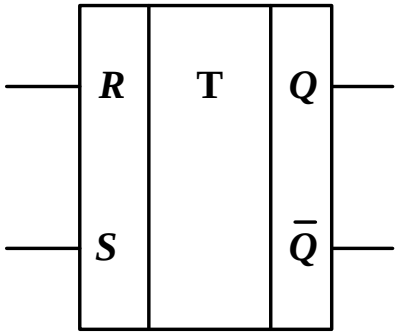


Условное графическое изображение дешифратора 3 х 8

Триггеры







Асинхронный ***RS*** **– триггер** содержит одну ячейку памяти, может быть выполнен на двух элементах ИЛИ-НЕ (рис.9.24) и имеет два прямых информационных входа:

R – раздельный вход сброса триггера (Q=0);

S – раздельный вход установки триггера (Q=1).

Триггер называется асинхронным, если переключение его происходит сразу при изменении информационных сигналов. Работа асинхронного RS – триггера на элементах ИЛИ-НЕ отображается таблицей переходов 9.4.

Таблица 9.4

R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	-

Функциональное состояние ***RS***- триггера определяется уравнением:

$$Q_{n+1} = \bar{R}_n S_n + \bar{R}_n Q_n \qquad (9.7)$$

где Q_n и Q_{n+1} - соответственно предыдущее и новое состояние триггера.

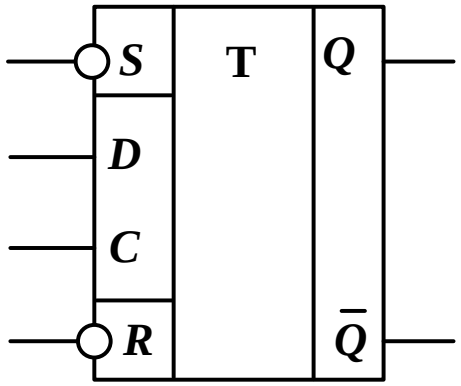
В синхронных триггерах имеется синхронизирующий вход ***C*** и переключение происходит при поступлении на этот вход синхронизирующего импульса. Причем момент переключения может соответствовать переднему или заднему фронту синхроимпульса.

9.9.2. *D*-триггер

***D*- триггер** (рис.9.25) имеет информационный вход *D* (data – данные). Информация со входа *D* заносится в триггер по положительному перепаду на счетном входе *C* триггера. Помимо счетного *C* и информационного *D* – входов, триггер имеет асинхронные установочные $\overline{R}$ и $\overline{S}$ входы. Установочные входы приоритетны. Они устанавливают триггер независимо от сигналов на входах *C* и *D*. После окончания установки входы $\overline{R}$ и $\overline{S}$ следует перевести в неактивное состояние $\overline{R}=\overline{S}=1$. Далее по переднему фронту импульса на счетном входе C на выход триггера пересылаются данные с информационного входа D.

Уравнение *D*- триггера имеет вид:

$$Q_{n+1}=\left[\overline{C}Q_n+CD+S\right]\overline{R} \qquad (9.8)$$



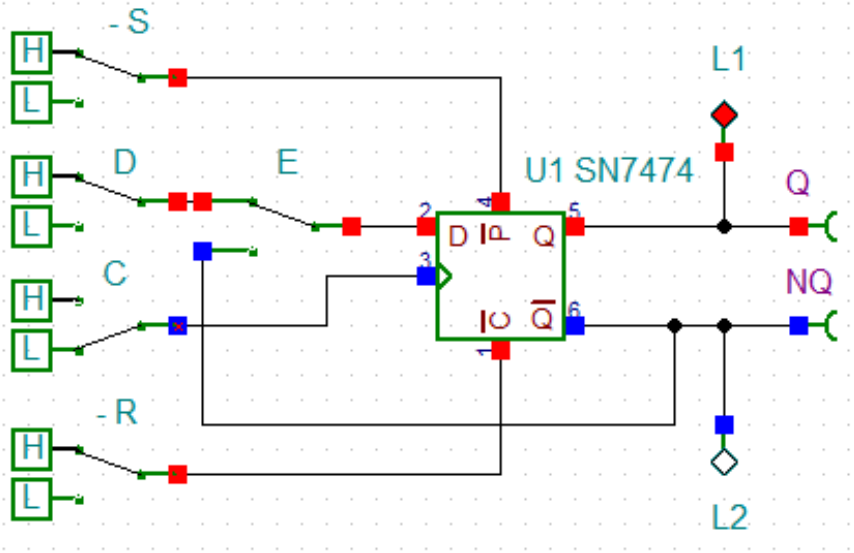


Таблица 9.5 переходов *D*-триггера

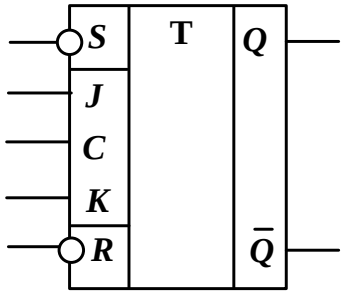
$\overline{R}$	$\overline{S}$	D	C	Q_n	Q_{n+1}
1	0	-	-	0	1
0	1	-	-	1	0
1	1	0	⌐	0	0
1	1	1	⌐	0	1

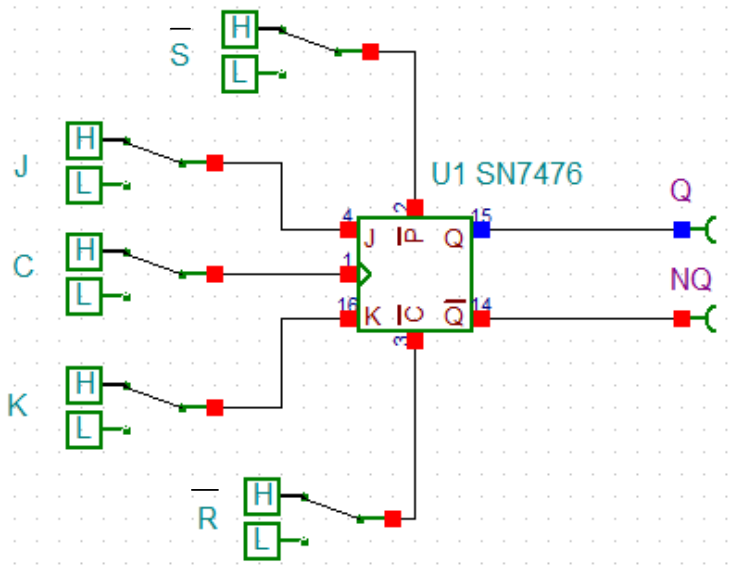
9.9.3. *JK*-триггер

***JK* –триггер** (рис.9.26) является наиболее универсальным, так как на его основе могут быть построены любые из рассмотренных выше триггеров. *JK*- триггер имеет: входы *J* (*jump*-прыжок) и *K* (*kill*- отключение) установки триггера в состояния *Q*=1 и *Q*=0 соответственно; синхронизирующий вход *C*; раздельный вход *S* асинхронной установки триггера (*Q*=1); раздельный вход *R* асинхронного сброса триггера (*Q*=0). В схеме (рис.9.26) входы *S* и *R* имеют низкий активный уровень. Причем, входы *S* и *R* имеют приоритетное значение. После асинхронной установки в модели надо установить $\bar{R} = \bar{S} = 1.$

Функциональное состояние *JK*- триггера определяется уравнением:

$$Q_{n+1} = \left[C(J\bar{Q}_n + \bar{K}Q_n) + \bar{C}Q_n + S \right] \bar{R} \qquad (9.9)$$

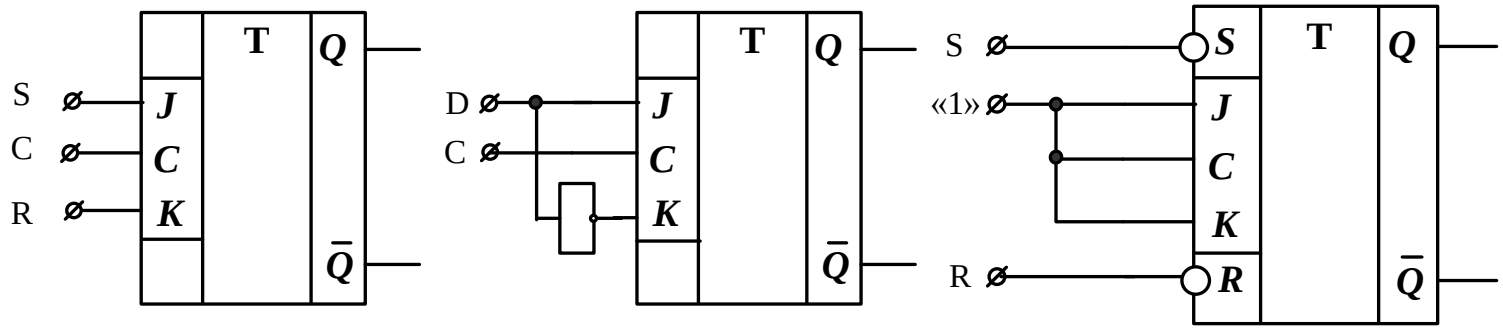




JK- триггер

Таблица переходов

$\overline{R}$	$\overline{S}$	*J*	*K*	*C*	Q_n	Q_{n+1}
1	0	-	-	-	1	0
0	1	-	-	-	0	1
1	1	1	0	⎍	0	1
1	1	0	1	⎍	1	0
1	1	1	1	⎍	Меняется на противополож-ное	

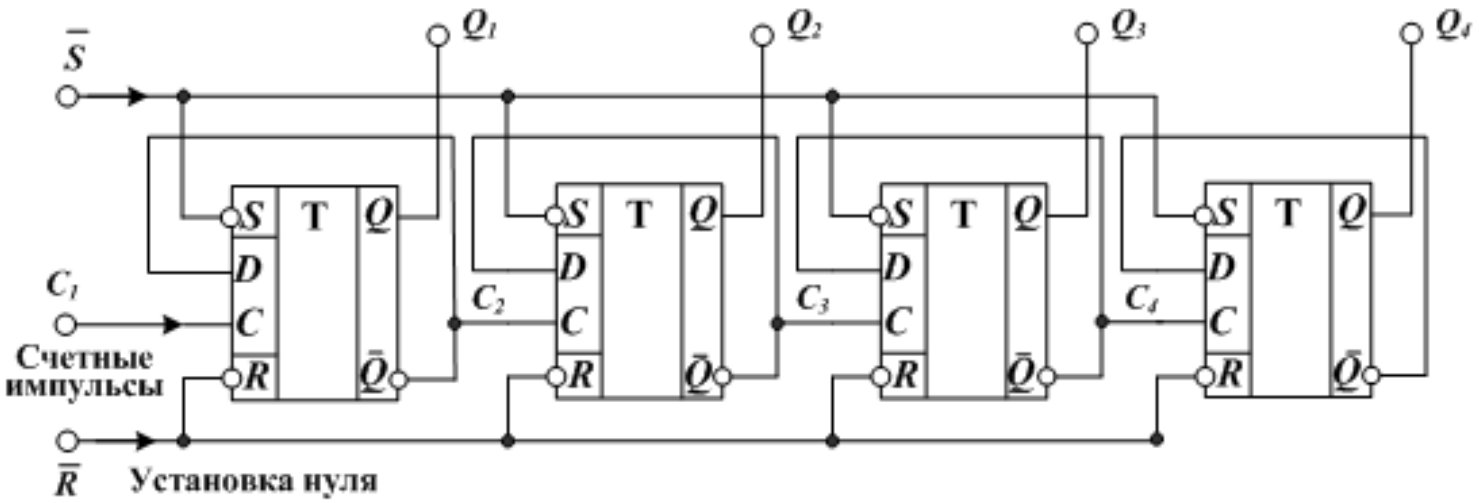


Синхронный RS-триггер D-триггер Асинхронный RS-триггер

На триггерах строят счетчики и регистры.

Счетчики импульсов

Счетчики предназначены для подсчета числа импульсов. С поступлением каждого импульса на вход *C* счетчик меняет свое состояние на единицу. Счетчики бывают суммирующие, вычитающие, реверсивные.



Асинхронный счетчик на *D*- триггерах

В начале счета все триггеры устанавливаются в нуль и с входов $\overline{R}$ и $\overline{S}$ снимаются активные уровни.

На счетный вход С1 первого триггера поступают счетные импульсы. Каждый импульс изменяет состояние триггеров так, что на выходах Q1 – Q4 формируется двоичный код, соответствующий числу счетных импульсов.

Модуль счета $K_{сч} = 2^4 = 16.$

Модель асинхронного счетчика

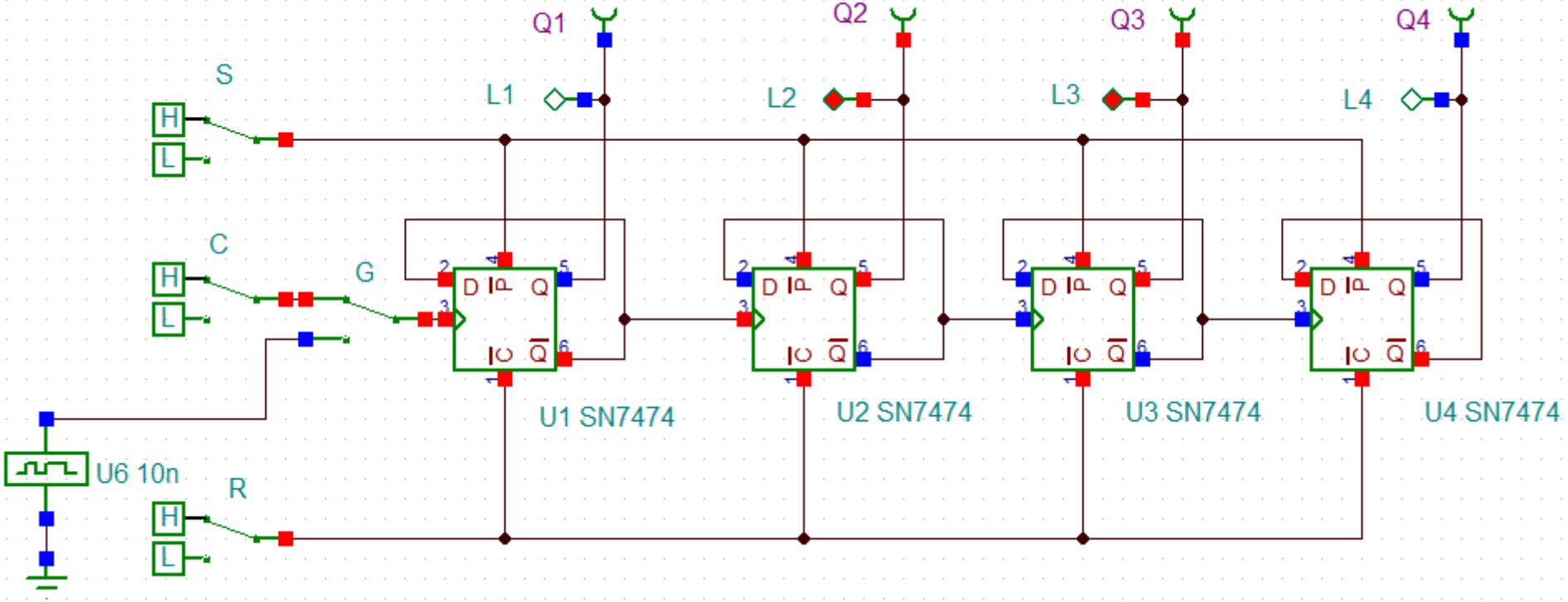
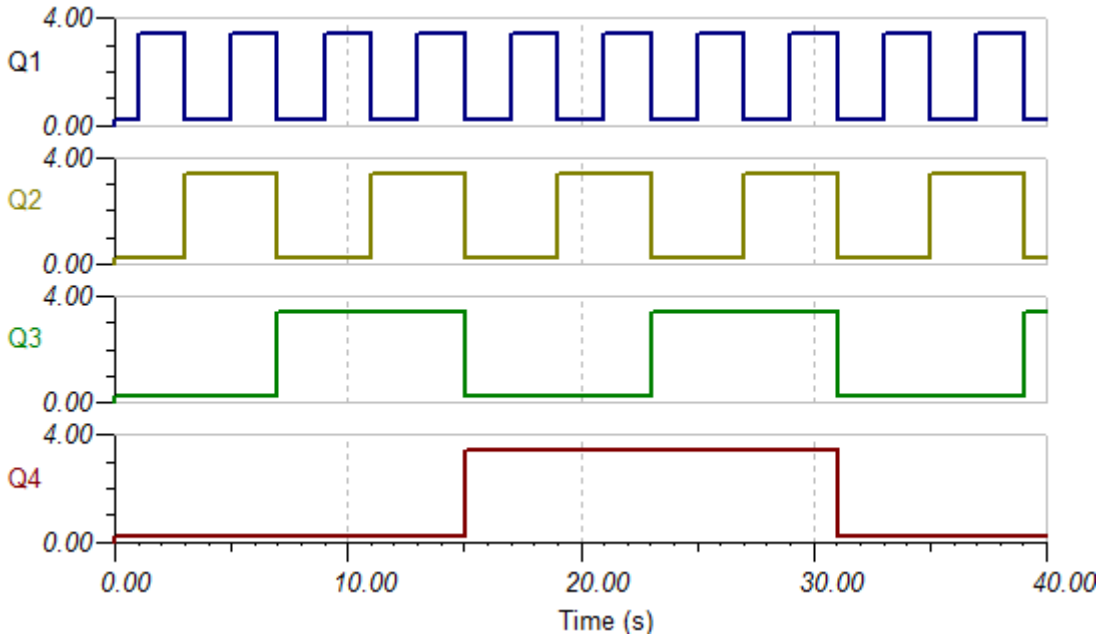


Рис.10.9



Временные диаграммы двоичного счетчика

9.11. Регистры

Регистром называется устройство цифровой техники, предназначенное для записи, хранения и (или) сдвига информации, представленной в виде многоразрядного двоичного кода.

По способу приема информации регистры подразделяют на:

последовательные (сдвигающие), в которых информация записывается и считывается только в последовательной форме;

параллельные (статические), в которых информация записывается и считывается только в параллельной форме;

последовательно-параллельные, в которых информация записывается или считывается как в последовательной, так и в параллельной формах.

Простейшие регистры выполняют на триггерах. Схема последовательного сдвигающего регистра на *JK*- триггерах показана на рис.9.32.

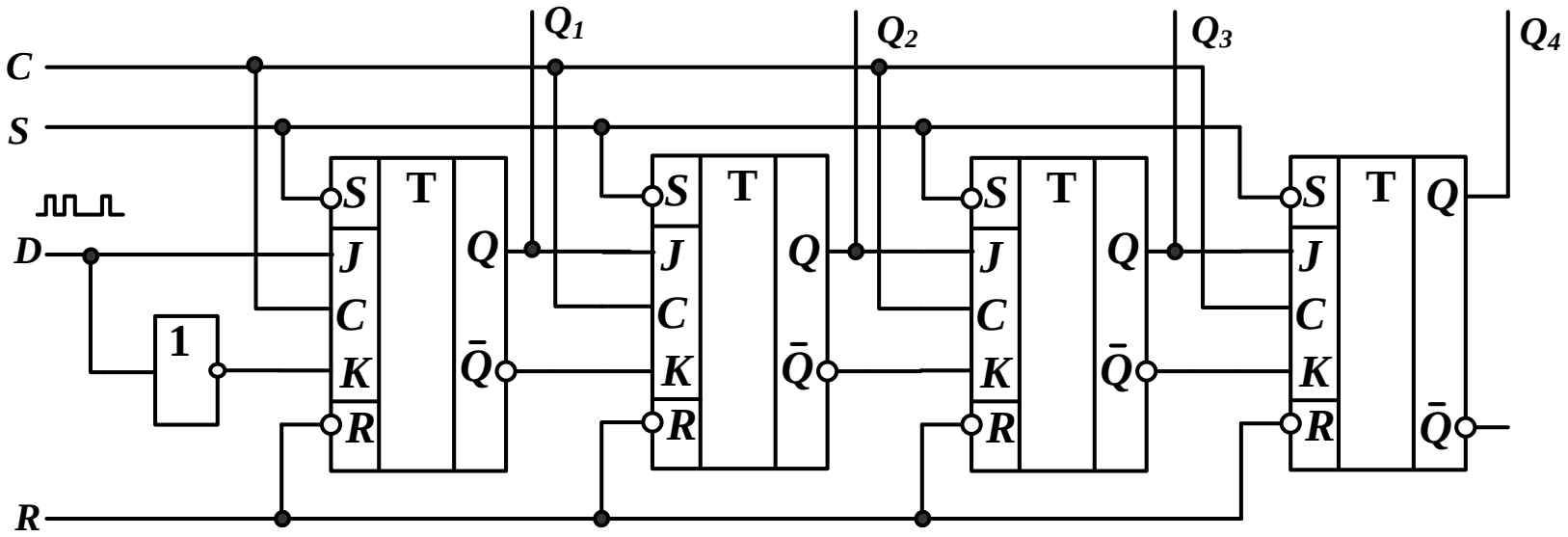


Рис.9.32. Четырехразрядный сдвигающий регистр с последовательным вводом