Министерство образования Республики Беларусь

Учреждение образования
«Гомельский государственный технический университет имени П. О. Сухого»

Кафедра «Промышленная электроника»

Ю. Е. Котова, Л. А. Захаренко

СХЕМОТЕХНИКА В СИСТЕМАХ УПРАВЛЕНИЯ

ПРАКТИКУМ
по одноименной дисциплине для студентов специальности 1-53 01 07 «Информационные технологии и управление в технических системах» дневной формы обучения

Гомель 2020

УДК 621.58/621.3(075.8)
ББК 32я73
К73

Рекомендовано научно-методическим советом
факультета автоматизированных и информационных систем
ГГТУ им. П. О. Сухого
(*протокол № 10 от 03.06.2019 г.*)

Рецензент: доц. каф. «Автоматизированный электропривод» ГГТУ им. П. О. Сухого
канд. техн. наук, доц. *М. Н. Погуляев*

Котова, Ю. Е.

К73 Схемотехника в системах управления : практикум по одноим. дисциплине для студентов специальности 1-53 01 07 «Информационные технологии и управление в технических системах» днев. формы обучения / Ю. Е. Котова, Л. А. Захаренко. – Гомель : ГГТУ им. П. О. Сухого, 2020. – 72 с. – Систем. требования: PC не ниже Intel Celeron 300 МГц ; 32 Mb RAM ; свободное место на HDD 16 Mb ; Windows 98 и выше ; Adobe Acrobat Reader. – Режим доступа: https://elib.gstu.by. – Загл. с титул. экрана.

Предназначен для получения и закрепления знаний, требуемых в рамках учебной программы по предмету «Схемотехника в системах управления», на практических занятиях и при самостоятельной работе.

Для студентов специальности 1-53 01 07 «Информационные технологии и управление в технических системах» дневной формы обучения.

УДК 621.58/621.3(075.8)
ББК 32я73

ВВЕДЕНИЕ

Данное пособие предназначено для подготовки студентов дневной формы обучения по специальности «Информационные технологии и управление в технических системах» в рамках учебной программы по предмету «Схемотехника в системах управления» в разделе цифровой схемотехники. Структурно пособие разбито на семь разделов, каждый из которых представляет собой отдельное практическое занятие с необходимыми теоретическими сведениями, примерами и заданиями для самостоятельного выполнения. Прежде, чем приступить к выполнению заданий, настоятельно рекомендуется изучить теоретическую часть для понимания основных принципов, и разобрать приведенные примеры.

Стоит отметить, что пособие содержит достаточный, но не исчерпывающий уровень материала для подготовки по данной дисциплине. Поэтому рекомендуется для более углубленной проработки материала обратиться к использованным при составлении пособия источникам.

1. СИСТЕМЫ СЧИСЛЕНИЯ. ПЕРЕВОД ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГИЕ

Системы счисления

Все современные системы счисления (кроме некоторых римских цифр) являются позиционными, т.е. в них одна и та же цифра в разных позициях (слева, справа) имеет разное значение. Например, в десятичном числе 55 левая цифра означает 50, а правая, – только 5. В общем виде в позиционной системе счисления с основанием системы X число A можно представить в виде:

$$\mathrm{A} = \sum_{i=1}^{n} a_i \cdot X ,$$

где n – количество разрядов числа A, a_i – коэффициенты каждого разряда, которые могут принимать значения от 0 до $(X-1)$.

При необходимости основание системы счисления указывается внизу после числа в виде нижнего индекса.

Позиционные системы счисления, используемые в цифровых устройствах .

- Четырехразрядное десятичное число:

$5680_{10} = 5 \cdot 10^3 + 6 \cdot 10^2 + 8 \cdot 10^1 + 5 \cdot 10^0$,

где $X = 10$, – основание системы счисления, $a_0 = 5, a_1 = 8$, $a_2 = 6$, $a_3 = 5$, – коэффициенты в каждом разряде, $n = 4$, – количество разрядов числа A.

- Трехразрядное восьмеричное число:

$372_8 = 3 \cdot 8^2 + 7 \cdot 8^1 + 2 \cdot 8^0 = 250_{10}$,

где $X = 8$ (восьмеричная система счисления), коэффициенты в разрядах числа A: $a_0 = 2, a_1 = 7$, а $n = 3$, – количество разрядов числа A.

- Двухразрядное шестнадцатеричное число:

$4E_8 = 4 \cdot 16^1 + 14 \cdot 16^0 = 78_{10}$,

где $X = 16$ (шестнадцатеричная система счисления), коэффициенты в разрядах числа A: $a_0 = E = 14_{10}$, $a_1 = 4$ (шестнадцатеричные цифры от 0 до 9 записываются так же, как и соответствующие десятичные цифры, а шестнадцатеричные цифры

10, 11, 12, 13, 14, 15 записываются заглавными латинскими буквами *A*, *B*, *C*, *D*, *E*, *F* соответственно); $n = 2$, – количество разрядов числа *A*.

- Четырехразрядное двоичное число:

$1101_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 13_{10}$,

где $X = 2$, – двоичная система счисления, коэффициенты в разрядах числа *A*: $a_0 = 1, a_1 = 0$, $a_2 = 1$, $a_3 = 1$, $n = 4$, – количество разрядов числа *A*.

Двоичный разряд, который может принимать только два значения: 0 или 1, называют «бит», происходящее от сокращения *BIT* английских слов

BI *NARY DIGI* ***T*** – «двоичная цифра». В английском языке слово *bit* означает также кусочек.

Преобразование чисел из одной системы счисления в другую.

Преобразовать десятичное число в двоичное можно путем деления на 2: сначала самого числа, а затем каждого промежуточного частного. При этом каждый неделимый остаток дает очередную цифру соответствующего разряда искомого двоичного числа. Первый полученный таким образом остаток даст цифру младшего разряда, а последний, – старшего разряда двоичного числа. Например, десятичное число 53_{10} преобразуем в двоичное (рис1.1):

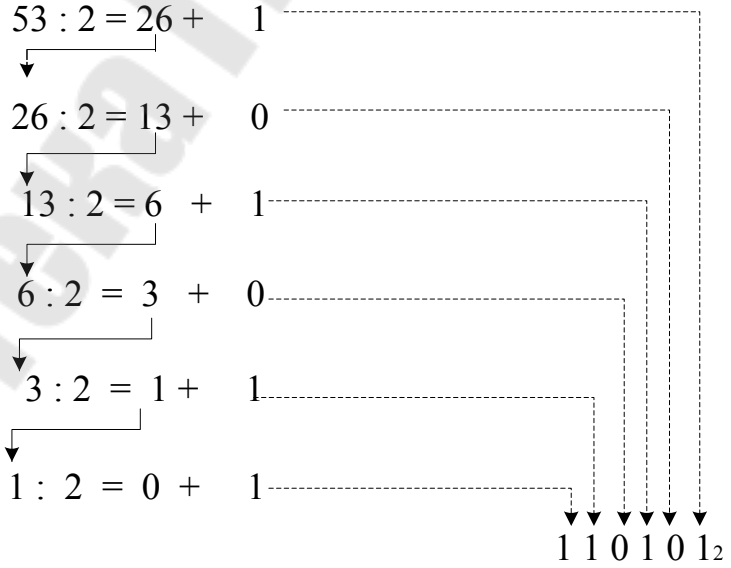


Рис.1.1. Пример преобразования десятичного число 53_{10} в двоичное

Аналогично можно преобразовывать числа с другими основаниями. Пример перевода десятичного числа 3480_{10} в шестнадцатеричное приведен ниже (рис1.2).

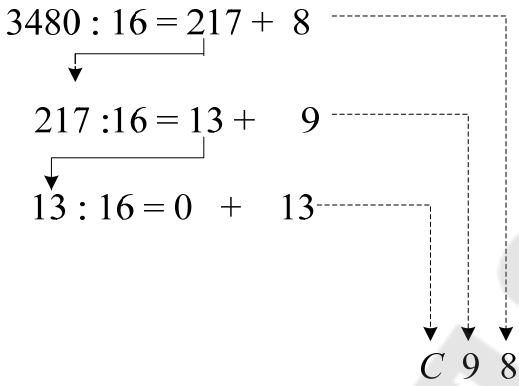


Рис.1.2. Пример преобразования десятичного число 3480_{10} в шестнадцатиричное

Так же, как большие десятичные числа для удобства чтения разбивают при записи на тройки, так и большие двоичные числа обычно разбивают на четверки, – тетрады.

В цифровой аппаратуре, в основном при индикации показаний десятичными цифрами или при задании параметров десятичными задатчиками, широко применяются различные двоично-десятичные коды.

Самый распространенный из них *BCD*-код (сокращенное *BINARY CODED DECIMAL*, – двоично-кодированная десятичная цифра), который порой называют позиционным 8421-кодом, или натуральным двоично-десятичным кодом. В этом коде каждая десятичная цифра представляется своей отдельной тетрадой, – четверкой двоичных цифр, например:

$$496_{10} = \begin{matrix} 4 & 9 & 6 \\ 0100 & 1001 & 0110_{BCD} \end{matrix}$$

Еще один распространенный двоично-десятичный код, – код с избытком 3 (*EXCESS* – 3 *CODE*). В нем каждая десятичная цифра кодируется двоичной тетрадой, в которой взвешенная сумма разрядов больше этой десятичной цифры на три. Так, десятичная цифра 9 записывается тетрадой 1100, для которой взвешенная сумма разрядов $8 \cdot 1 + 4 \cdot 1 + 2 \cdot 0 + 1 \cdot 0 = 12$, что на 3 больше числа девять.

В двоично-кодированных датчиках перемещения или угла поворота часто применяется код Грея (*GRAY CODE*). В этом коде комбинации двоичных цифр, представляющие числа, соседние по

величине, отличающиеся лишь в одной кодовой позиции, т.е. при последовательном переходе от одного числа к другому всегда изменяется только один из двоичных разрядов.

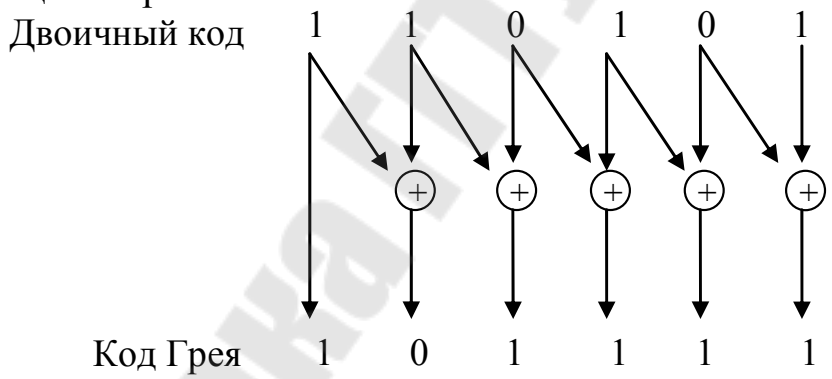
Число B, записанное в двоичном коде, можно преобразовать в число G в коде Грея с помощью следующего выражения: $G_i = B_i \oplus B_{i+1}$

Число G, записанное в коде Грея, можно преобразовать в число B в двоичном коде с помощью следующего выражения: $B_i = G_i \oplus B_{i+1}$.

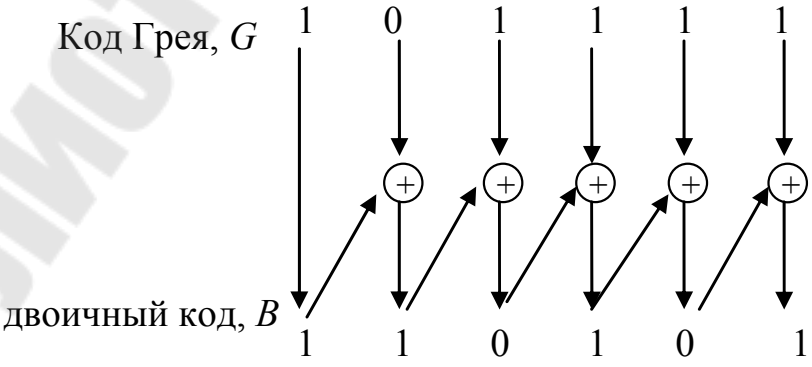
Здесь знак $\oplus$ означает сумму по модулю два, т.е. функцию, которая равна единице, если входные переменные различны, и, – нулю, в случае их равенства. Алгебраически это можно записать следующим образом:

$$0 \oplus 0 = 0$$
$$0 \oplus 1 = 1$$
$$1 \oplus 0 = 1$$
$$1 \oplus 1 = 1.$$

Число в коде Грея можно также получить из двоичного кода следующим образом:



Обратное преобразование кода Грея в двоичный код производят по похожей схеме:



ЗАДАНИЕ НА ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №1

Номер своего варианта задания определяются из предложенной таблицы (табл.1.1) согласно номеру № студента в списке журнала группы.

Таблица 1.1

№ варианта	A	B	C	D
1	123	7145	00110111	A14
2	231	6567	11100110	B67
3	286	6012	10111010	60F
4	331	5567	11111000	55C
5	386	5012	00011111	501
6	441	4665	11010011	6E5
7	496	4600	01111010	46C
8	542	4515	11001110	F15
9	597	4450	11100101	4A5
10	652	4385	10111100	3AA
11	707	4321	10010111	AF1
12	174	6936	11100110	6EA
13	242	6456	11011100	64B
14	297	5901	10001111	590
15	342	5456	11110001	F54
16	397	4901	01101110	49E
17	452	4652	11001101	A52
18	507	4587	01110101	5F8
19	553	4502	11110001	4AB
20	608	4437	00111110	CF7
21	663	4372	10100111	BA2
22	192	6893	11101100	C24
23	253	6345	10011101	6F5
24	308	5890	11001011	F40
25	353	5345	01111001	A04
26	408	4890	00101111	4890
27	463	4639	11001101	4639
28	518	4574	10111001	4574
29	564	4489	00011111	4489
30	619	4424	11101010	4424

1. Перевести заданные числа *A* и *B* из десятичной системы в двоичную, восьмеричную и шестнадцатиричную системы счисления.

2.Записать числа *A* и *B* в *BCD*-коде и в коде с избытком 3 (*EXCESS* – 3 *CODE*).

3. Перевести полученные двоичные числа *A* и *B* в код Грея.

4. Перевести число *C*, заданное в коде Грея, в двоичное, а затем в десятичное.

5. Перевести число *D*, заданное в шестнадцатиричном коде в десятичное, а затем в двоичное и в код Грея.

2. АРИФМЕТИЧЕСКИЕ ДЕЙСТВИЯ (СЛОЖЕНИЕ, ВЫЧИТАНИЕ, УМНОЖЕНИЕ, ДЕЛЕНИЕ) С МНОГОРАЗРЯДНЫМИ ДВОИЧНЫМИ ЧИСЛАМИ

Сложение многоразрядных чисел

Сложение двух двоичных чисел осуществляется точно так же, как и в случае десятичных чисел. Более того, двоичное сложение даже проще, так как есть лишь несколько правил, которые необходимо будет запомнить. Рассмотрим десятичное сложение:

$$\begin{array}{rrrr} & 3 & 7 & 6 \\ + & 4 & 6 & 1 \\ \hline & 8 & 3 & 7 \end{array}$$

Сначала складываются младшие значащие разряды, которые в данном примере дают в сумме 7. После этого прибавляются разряды во второй позиции справа. Их сумма равна 13, поэтому делается перенос 1 из разряда десятков в старший разряд. Складывая ее с цифрами, которые там уже имеются, получаем 8.

Аналогичные операции нужно выполнить и в случае двоичного сложения. Однако теперь дело приходится иметь только с четырьмя возможными вариантами сложения, так как в любой позиции можно складывать лишь два двоичных числа (бита):

$0 + 0 = \mathbf{0}$
$1 + 0 = \mathbf{1}$
$1 + 1 = \mathbf{0} = 1\mathbf{0} = \mathbf{0}$ + перенос 1 в следующий разряд
$1 + 1 + \mathbf{1} = 1\mathbf{1} = \mathbf{1}$ + перенос 1 в следующий разряд

Последний случай иллюстрирует ситуацию, когда складываются единицы, стоящие в одном разряде, а к ним после переноса из младшего разряда добавляется еще одна единица. Теперь рассмотрим несколько примеров сложения пары двоичных чисел (в скобках рядом приведены их десятичные эквиваленты):

$$\begin{array}{rrrrrr} & 0 & 1 & 1 & (3) \\ + & 1 & 1 & 1 & (6) \\ \hline 1 & 0 & 0 & 1 & (9) \end{array}$$

$$\begin{array}{rrrrrr} & 1 & 0 & 1 & 1 & (0) \\ + & 1 & 1 & 1 & 1 & (15) \\ \hline 1 & 1 & 0 & 0 & 0 & (24) \end{array}$$

Обратный код

Обратный код двоичного числа (*l's complement*) получают заменой каждого 0 на 1 и каждой 1 на 0. Иначе говоря, каждый бит числа изменяется на противоположный (обратный). Этот процесс показан далее.

```
 1  0  1  1  0  1   двоичное число
 ↓  ↓  ↓  ↓  ↓  ↓
 0  1  0  0  1  0   обратный код двоичного числа
```

Таким образом, обратный код числа 101101_2 будет равен 010010_2.

Дополнительный код

Дополнительный код двоичного числа (*2's complement*) получают из обратного кода путем добавления 1 к его младшему значащему биту. Процесс преобразования числа $101101_2 = 45_{10}$ в обратный код показан далее:

```
   1  0  1  1  0  1   двоичный эквивалент 45
   0  1  0  0  1  0   каждый бит преобразуется в обратный код
+                 1   к младшему разряду  прибавляется 1
-------------------
   0  1  0  0  1  1   дополнительный код двоичного числа 45.
```

Таким образом, число 010011_2 будет дополнительным кодом числа 101101_2.

Представление чисел со знаком в системе дополнительных кодов.

Дополнительный код применяется для представления чисел со знаком.Это можно схематически представить следующим образом.

Если число положительное, то его модуль представляется в естественной форме двоичного числа, а знаковый бит содержит 0, который ставится перед старшим знаковым битом (рис.2.1).

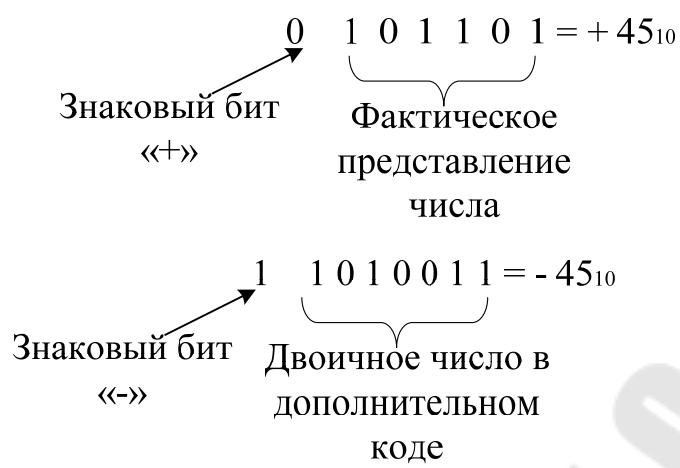


Рис.2.1. Представление чисел со знаком в системе дополнительных кодов

Таким образом, если знаковый бит у двоичного числа равен единице, это говорит о том, что число отрицательное и представлено в дополнительном коде. Для того чтобы узнать модуль этого числа требуется проделать обратную операцию, т.е. проинвертировать каждый бит числа (перевести в обратный код), а затем к младшему разряду добавить единицу.

Пример получения модуля отрицательного двоичного числа, представленного в дополнительном коде:

Знаковый бит							
1	1	0	1	0	0	Отрицательное двоичное число -12_{10} в дополнительном коде
0	0	1	0	1	1	каждый бит преобразуется в обратный код
+					1	к младшему разряду прибавляется 1
0	0	1	1	0	0	модуль отрицательного числа: $\lvert -12 \rvert = 12_{10}$

Дополнительный код используется для представления чисел со знаками, потому что этот код позволяет осуществлять операцию вычитания через сложение.

Сложение в системе дополнительных кодов

Теперь проанализируем, как осуществляются операции сложения и вычитания в цифровых машинах, которые для представления отрицательных чисел используют дополнительный код. Важно помнить, что в большинстве случаев со знаковым битов обращаются так же, как и с битами модуля

- Вариант I. Два положительных числа.

Сложение двух положительных чисел выполняется непосредственно. Рассмотрим сложение чисел +9 и +4:

+9 →	0	1	0	0	1	первое слагаемое
+4 →	0	0	1	0	0	второе слагаемое
.	0	1	1	0	1	сумма (+13)

знаковые биты

Важно отметить, что оба знаковых бита как первого, так и второго слагаемого равны 0 поэтому и знаковый бит суммы равен 0, что говорит о ее принадлежности к положительным числам. Также следует заметить, что оба слагаемые должны иметь одинаковое количество бит, так как это необходимо для использования дополнительного кода. Поэтому **всегда** нужно добавлять нули слева от значащей части числа, чтобы уравнять количество битов обоих чисел.

- Вариант II. Положительное число и отрицательное число, меньшее по модулю.

Рассмотрим сложение двух чисел: +9 и –4. Число –4 выражается в виде дополнительного кода. Таким образом, сначала надо перевести +4(00100) в –4(11100).

В этом случае знаковый бит второго слагаемого содержит 1. Обратите внимание, что теперь оба знаковых бита участвуют в процессе сложения. В последнем разряде модуля возникает перенос, который, однако, не учитывается.

знаковые биты

+9 →	0	1	0	0	1	первое слагаемое
–4 →	1	1	1	0	0	второе слагаемое
1	0	0	1	0	1	сумма (+5)

этот перенос игнорируется

Вообще, **при использовании дополнительного кода перенос из старшего разряда не учитывается**, поэтому конечный результат равен 00101, что эквивалентно +5.

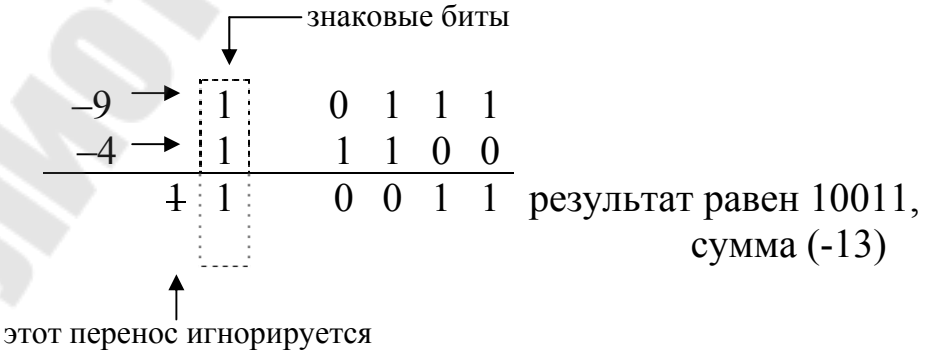
- Вариант III. Положительное число и отрицательное число, большее по модулю.

Рассмотрим сложение чисел –9 и +4.

Полученная сумма имеет знаковый бит, содержащий 1, это говорит о том, что она отрицательна. Так как сумма меньше нуля, то она имеет вид дополнительного кода, т.е. только четыре младших бита (1011) представляют собой реальную величину числа (модуль). Чтобы получить абсолютное значение числа, необходимо применить операцию отрицания (получить дополнительный код числа) к величине 11011 и прибавить 1 к младшему разряду; результат будет равен $00101 = +5$. Таким образом, число 11011 с учетом знакового бита вставляет собой число –5.

- *Вариант IV*. Два отрицательных числа.

Сложим два отрицательных числа –4 и –9. Конечный результат будет отрицательным и представляется в виде дополнительного кода, т.е. его знаковый бит равен 1. Отрицание этого числа и добавление единицы к младшему разряду (получение его дополнительного кода) даст $01101 = +13$.



Вычитание в системе дополнительных кодов

Операция *вычитания* в системе дополнительных кодов по сути состоит из операции сложения и ничем не отличается от примеров, показанных ранее. При вычитании одного двоичного числа (*вычитаемого*) из другого двоичного числа (*уменьшаемого*) следуют такой процедуре:

- Отрицание вычитаемого. Эта операция заменяет вычитаемое эквивалентной по модулю величиной, но с противоположным знаком.
- Сложение полученной величины с уменьшаемым. Сумма такого сложения и будет представлять собой *разность* между уменьшаемым и вычитаемым.

Так же, как и во всех математических операциях с применением дополнительного кода, необходимо, чтобы оба числа содержали одинаковое количество бит.

Переполнение

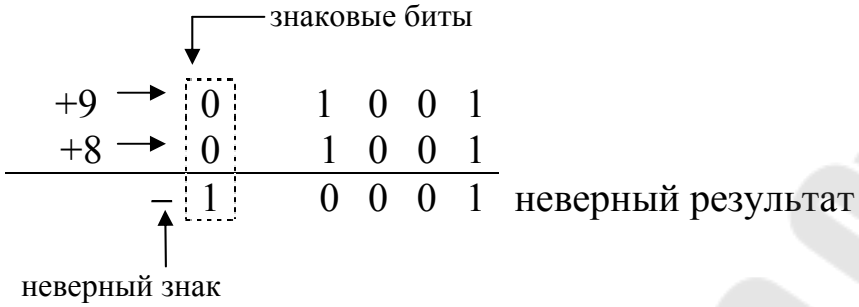
В каждом из примеров сложения, приведенных выше, слагаемые числа состояли из знакового бита и четырех битов модуля. Результаты также состояли из знакового бита и тех же четырех битов модуля. Любой перенос в шестой разряд не учитывался. Во всех рассмотренных случаях модуль конечного результата можно было уместить в четыре бита. Теперь рассмотрим сложение чисел +9 и +8.

Ответ содержит знаковый бит, соответствующий отрицательному числу, который, как совершенно очевидно, неправильный, поскольку суммировались два положительных числа. В результате должно быть +17, но для записи такого числа потребуются больше четырех битов. Таким образом, возникает *переполнение*, которое переносит единицу в знаковый бит.

Условие *переполнения* может реализоваться только в случае сложения двух положительных или двух отрицательных чисел, в результате чего может получиться неправильный ответ. Переполнение легко обнаружить при проверке знакового бита — он должен совпадать со знаковыми битами обоих слагаемых.

Поскольку операция вычитания происходит в системе дополнительных кодов и оществляется с помощью отрицания вычитаемого и последующего *сложения* полученной величины с

уменьшаемым, переполнение может возникнуть только в том учае, если уменьшаемое и вычитаемое имеют разные знаки



Например, если вычитается –8 из +9, то к числу –8 нужно применить операцию отрицания, в результате получим +8, которое затем сложится с +9, как было показано ранее. В этом случае возникнет переполнение разряда, которое приведет к неверному результату, так с модуль суммы окажется слишком большим.

Умножение и деление двоичных чисел

Пример умножения двух чисел представлен ниже (рис.2.2):

					1	1	0	1			1	3
			×			1	0	1	×			5
					1	1	0	1				
		+										
			0	0	0	0						
	+											
		1	1	0	1							
1	0	0	0	0	0	1				6	5

Рис.2.2. Пример умножения двух двоичных чисел

Так как частичное произведение многоразрядного числа на 1 равно этому числу, а умножение на 0 даёт нули во всех разрядах, то операция умножения сводится к операциям сдвига и сложения частичных произведений.

Двоичное деление сводится к следующим операциям: вычитанию делителя из делимого, сдвигу делителя на один разряд, сравнению полученного остатка с делителем (рис.2.3).

$$
\begin{array}{l|l}
110111_2\ [55_{10}] & 101\quad [5_{10}] \\
\underline{-\ 101} & \underline{\quad 1011\quad [11_{10}]} \\
111 & \\
\underline{\ 101} & \\
101 & \\
\underline{\ 101} & \\
0 &
\end{array}
$$

Рис.2.3. Пример деления двух двоичных чисел $55_{10} : 5_{10}$

Если остаток меньше делителя, делитель сдвигается ещё на один разряд, затем снова делитель сравнивается с остатком, и если остаток больше делителя, то из него вычитается делитель, затем опять следует сдвиг делителя, вновь сравнение, так до тех пор, пока после сдвига делителя его младший разряд сравняется с младшим разрядом делимого. В тех разрядах, где сравнение делителя с остатком показало, что остаток меньше делителя, в данный разряд результата деления – частного – записывается нуль, в остальные разряды – 1. Итак, деление сводится к вычитанию, сдвигу и сравнению.

ЗАДАНИЕ НА ПРАКТИЧЕСКОЕ ЗАНАТИЕ № 2

Номер своего варианта задания определяются из предложенной таблицы (табл.2.1) согласно номеру № студента в списке журнала группы.

Таблица 2.1

№ варианта	*A*	*B*	*C*	*D*	*E*
1	123	145	100110111	1101	101
2	231	567	111100110	1110	111
3	286	012	110111010	1001	010
4	331	567	111011000	1010	011
5	386	512	100011111	1100	100
6	441	465	101010011	1101	101
7	496	460	101111010	1001	110
8	542	451	101001110	1101	101
9	597	445	111001010	1011	111
10	652	438	101111000	1011	010
11	707	432	100101110	1101	011
12	174	693	111001100	1011	100
13	242	645	110111000	1100	101
14	297	590	100011110	1010	110
15	342	545	111100010	1001	101
16	397	490	101101110	1010	111
17	452	462	110011101	1001	010
18	507	457	101110101	1001	011
19	553	452	101110001	1100	100
20	608	437	100111110	1001	101
21	663	472	110100111	1010	110
22	192	683	101101100	1100	101
23	253	634	100110101	1001	111
24	308	589	110010011	1010	010
25	353	534	01111001	1100	011
26	408	489	00101111	1110	100
27	463	463	11001101	1110	101
28	518	457	101011001	1111	110
29	564	448	100011111	1110	111
30	619	442	101101010	1100	101

1. Перевести заданные числа A и B из десятичной системы в двоичную.

2. Вычислить в двоичной системе счисления $(A+B)$; $(A-B)$; $(-A-B)$.

3. Вычислить в двоичной системе счисления $(A+C)$; $(C-B)$; $(C-A)$; $(C+B)$.

4. Вычислить в двоичной системе счисления произведения DE; AE, и частного $C \div E$.

5. Для проверки правильности арифметических действий перевести результаты вычислений из двоичной системы счисления в десятичную. Сравнить полученные результаты.

3. СОСТАВЛЕНИЕ БУЛЕВЫХ ВЫРАЖЕНИЙ ПО ТАБЛИЦЕ ИСТИННОСТИ И МИНИМИЗАЦИЯ ИХ АЛГЕБРАИЧЕСКИМ СПОСОБОМ.

Дизъюнктивные и конъюнктивные формы записи функций алгебры логики

Совершенной дизъюнктивной нормальной формой (СДНФ) называют наиболее полную форму записи логического выражения. Эта форма записи представляет собой сумму, каждое слагаемое которой является произведением всех входных аргументов или их инверсий, например:

$$F = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + AB\overline{C} + ABC$$

СДНФ является избыточной, но логические функции, записанные в СДНФ, легко сравнивать между собой, их удобно преобразовывать в таблицы истинности и составлять по ним карты Карно. Булево выражение, полученное из таблицы истинности логической функции, имеет совершенную дизъюнктивную нормальную форму.

В некоторых случаях более удобной формой записи логического выражения является *совершенная конъюнктивная нормальная форма* (СКНФ). Это произведение сомножителей, каждый из которых является суммой всех входных аргументов или их инверсий, например:

$$F = (\overline{A} + B + \overline{C})(\overline{A} + B + C)(\overline{A} + \overline{B} + C)(A + B + C).$$

Так же, как и СДНФ, СКНФ является явно избыточной.

Преобразование таблицы истинности в булево выражение

Допустим, имеется логическая функция F для трех переменных A, B и C, заданная в виде следующей таблицы истинности (см.табл.3.1.):

Смысл составления булева выражения по таблице истинности в том, чтобы показать при каких сочетаниях входных переменных или их инверсий заданная функция F равна единице (при составленни выражения в СДНФ) (или нулю при составлении выражения в СКНФ).

Запишем булево выражение в совершенной дизъюнктивной нормальной форме. Для этого выберем из таблицы истинности из всех возможных восьми комбинаций входных переменных *A, B* и *C* те сочетания переменных , при которых функция *F* равна единице. В таблице 3.1 они записаны в виде логических произведений переменных P_0, P_2, P_3 и P_7, причем переменная, имеющая значение лог 0, записывается с инверсией.

Таблица 3.1.

№	*A*	*B*	*C*	*F*	Примечания
0	0	0	0	1	$P_0 = \overline{A}\,\overline{B}\,\overline{C}$
1	0	0	1	0	
2	0	1	0	1	$P_2 = \overline{A}B\overline{C}$
3	0	1	1	1	$P_3 = \overline{A}BC$
4	1	0	0	0	
5	1	0	1	0	
6	1	1	0	0	
7	1	1	1	1	$P_7 = ABC$

Поскольку функция будет иметь такое значение при любом из наборов P_0, P_2, P_3, P_7 независимо друг от друга, то их можно соединить между собой знаком ИЛИ (логическим сложением):

$$F = P_0 + P_2 + P_3 + P_7$$

Каждый из наборов P_0, P_2, P_3, P_7 является таким сочетанием входных переменных или их инверсий, которые только при совместном их воздействии обеспечивает единичное состояние выходной функции.

Следовательно, каждый такой набор состоит из всех входных переменных или их инверсий, связанных между собой функцией И, логическим умножением:

$$P_0 = \overline{A}\,\overline{B}\,\overline{C};\ P_2 = \overline{A}B\overline{C};\ P_3 = \overline{A}BC;\ P_7 = ABC$$

Исходя из этого, получаем результирующее выражение:

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

Как можно заметить, полученное выражение соответствует СДНФ.

Совершенная конъюнктивная нормальная форма (СКНФ) представляется логическим произведением дизъюнкций, каждая из которых содержит все переменные в прямом или инверсном виде не более одного раза. Рассмотрим таблицу 3.2. Она аналогична таблице

3.1, только в ней выделены из всех комбинаций переменных A, B и C те сочетания, при которых функция F равна нулю, эти комбинации обозначены P_1, P_4, P_5, P_6 и записаны в виде логической суммы переменных или их инверсий, причем в этом случае инверсной считается переменная, принимающая значение «лог. 1».

Таблица 3.2.

№	A	B	C	F	Примечания
0	0	0	0	1	
1	0	0	1	0	$P_1 = A + B + \overline{C}$
2	0	1	0	1	
3	0	1	1	1	
4	1	0	0	0	$P_4 = \overline{A} + B + C$
5	1	0	1	0	$P_5 = \overline{A} + B + \overline{C}$
6	1	1	0	0	$P_6 = \overline{A} + \overline{B} + C$
7	1	1	1	1	

Каждый из наборов P_1, P_4, P_5, P_6 является таким сочетанием входных переменных и их инверсий, которые только при совместном их воздействии обеспечивает нулевое состояние выходной функции. Если записать выражение в виде произведения данных наборов получим булево выражение, записанное в совершенной конъюнктивной нормальной форме:

$$F = \left(A + B + \overline{C}\right)\left(\overline{A} + B + C\right)\left(\overline{A} + B + \overline{C}\right)\left(\overline{A} + \overline{B} + C\right)$$

Минимизация логических функций алгебраическим способом

Основой минимизации алгебраическим способом является последовательное использование законов булевой алгебры и правил преобразований. Кроме законов, известных нам из обычной алгебры, в булевой алгебре типовыми приемами можно считать следующие:

1. 1. *Многократное прибавление или умножение* какого–либо переменного, или нескольких переменных, что не изменяет функцию, поскольку:

$$A + A + \ldots + A; \qquad ABC + ABC + \ldots = ABC;$$
$$A \cdot A \cdot A \cdot \ldots = A; \qquad ABC \cdot ABC \cdot \ldots = ABC.$$

2. *Умножение членов уравнения на сумму* $A + \overline{A} = 1$, или сложение их с $A \cdot \overline{A} = 0$.

3. Использование выражений и законов булевой алгебры.

$$A + A = A \qquad A \cdot A = A$$
$$\overline{\overline{A}} = A \qquad A + \overline{A}B = A + B$$
$$A + AB = A \qquad A \cdot (A + B) = A$$
$$A + \overline{A} = 1 \qquad A \cdot \overline{A} = 0$$
$$A + (BC) = (A + B)(A + C) \qquad A + 1 = 1$$

Пример минимизации логической функции алгебраическим способом:

$$F = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC = \overline{A}BC + A\overline{B}C + AB\overline{C} +$$
$$+ ABC + \underline{(ABC + ABC)}$$

- в форму записи функции добавили – *(ABC + ABC);* после этого перегруппируем слагаемые таким образом, чтобы вынести за скобки общий множитель:

$$F = \overline{A}BC = ABC + A\overline{B}C + ABC + AB\overline{C} + ABC =$$
$$= (A + \overline{A})BC + (B + \overline{B})AC + (C + \overline{C})AB =$$
$$= AB + DC + AC$$

ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ №3

Номер варианта задания

а) выбирается равным номеру *N* студента по журналу группы;

б) выбирается равным (30 - *N*), где *N* - номер студента по журналу группы;

Таблица 3.3

N	*F*1	*F*2
1	00110111	00010110111110000
2	11100110	00111000000011100
3	10111010	11000011110100000
4	11111000	11100011000000111
5	00011111	00000111001100 11
6	11010011	11110100000011000
7	01111010	00001111011000 00
8	11001110	11000011000111 00
9	11100101	11100000110000 10
10	10111100	00110001111001 10
11	10010111	11100000110000 01
12	11100110	00011100000111 01
13	11011100	10011000111000 01
14	10001111	00010001110001 11
15	11110001	00111100000000 111
16	01101110	11110100000110 00
17	11001101	00110001111001 10
18	01110101	00010110111100 00
19	11110001	00110001111001 10
20	00111110	00010110111100 00
21	10100111	11000011101000 00
22	11101100	10011000111000 01
23	10011101	10011000111000 01
24	11001011	00010001110001 11
25	01111001	00110001111001 10
26	00101111	00111100000000 111
27	11001101	11000011101000 00
28	10111001	00011100000111 01
29	00011111	00111100000000 111
30	11101010	00010110111100 00

1. В последнюю строку таблицы истинности (табл. 3.4) вписать заданное восьмиразрядное двоичное число $F1$, взятое согласно номеру варианта из таблицы 3.3.

2. Представить логическую функцию, заданную таблицей истинности, в конъюнктивной совершенной нормальной форме.

3. Представить логическую функцию, заданную таблицей истинности, в дизъюнктивной совершенной нормальной форме.

Таблица 3.4

A	0	0	0	0	1	1	1	1
B	0	0	1	1	0	0	1	1
C	0	1	0	1	0	1	0	1
*F*1								

4. Выполнить минимизацию функций, записанных в СДНФ, используя метод непосредственных преобразований.

5. В последнюю строку таблицы истинности (табл.3.5) вписать заданное шестнадцатиразрядное двоичное число $F2$.

Таблица 3.5

A	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
B	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
C	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
D	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
*F*2																

6. Представить логическую функцию, заданную таблицей истинности, в конъюнктивной совершенной нормальной форме.

7. Представить логическую функцию, заданную таблицей истинности, в дизъюнктивной совершенной нормальной форме.

8. Выполнить минимизацию функций, записанных в СДНФ, используя метод непосредственных преобразований.

4. СИНТЕЗ КОМБИНАЦИОННЫХ СХЕМ ПО БУЛЕВЫМ ВЫРАЖЕНИЯМ

Минимизация логических функций по картам Карно

Для минимизации логических функций очень удобно пользоваться *картами Карно* или очень схожими с ними диаграммами Вейча.

Карта Карно изображает в виде графических квадратов (клеток) все возможные комбинации переменных, причем переменные, определяющие координаты клеток карты, размещают так, чтобы при переходе из одной клетки в соседнюю, как по горизонтали, так и по вертикали, изменялась только одна переменная.

Таблица истинности для четырех переменных включает 16 строк, следовательно, карта Карно должна состоять из 16 клеток, как показано на рис.4.1

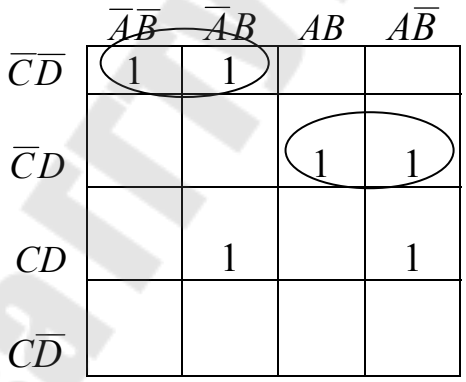


Рис.4.1 Пример карты Карно для 4–х переменных.

Если требуется получить карту Карно для какой–либо функции, сначала надо записать эту функцию в СДНФ, – в совершенной дизъюнктивно нормальной форме, или в виде таблицы истинности. Каждое слагаемое булева выражения в СДНФ, или каждая единица в столбце функции таблицы истинности, задается на карте Карно единицей в соответствующей клетке. Координаты этой клетки содержат те же входные переменные и их инверсии, что и данное слагаемое СДНФ булева выражения (или данная строка таблицы истинности).

У карты Карно для четырех переменных клетки крайнего левого столбца должны рассматриваться как соседние для клеток крайнего правого столбца, а клетки верхней строки, – как соседние для клеток нижней строки. Другими словами можно сказать, что эта карта расположена на поверхности цилиндра (склеили правый край карты с левым), изогнутого и растянутого так, что его верхний срез соединяется с нижним срезом; при этом цилиндр превращается в тор.

Правила упрощения заполненной карты Карно заключаются в следующем:

– соседние две, четыре, восемь, или другое число единиц, равное степени двойки, обводят общим контуром;

– контур должен быть прямоугольным без изгибов или наклонов;

– каждый контур превращает все входящие в него единицы в одну, т.е. объединенные таким образом слагаемые СДНФ булева выражения дают одно слагаемое в упрощенном выражении;

– те входные переменные, которые входят в координаты данного контура совместно со своими инверсиями, исключаются из слагаемого, которое дает этот контур в упрощенное выражение.

Примеры упрощения булевых выражений с помощью карты Карно для 4-х переменных:

1. $F1 = \overline{A}BC\overline{D}_1 + AB\overline{C}\overline{D}_2 + \overline{A}B\overline{C}D_3 AB\overline{C}D_4 + \overline{A}\overline{B}C\overline{D}_5 + \overline{A}BC\overline{D}_6$

В первом примере минимизации булевой функции $F1$ (см. рис. 4.2) нижний контур из двух единиц 1_5 и 1_6, соответствующие пятому и шестому слагаемым в исходном булевом выражении, дает возможность опустить B и $\overline{B}$.

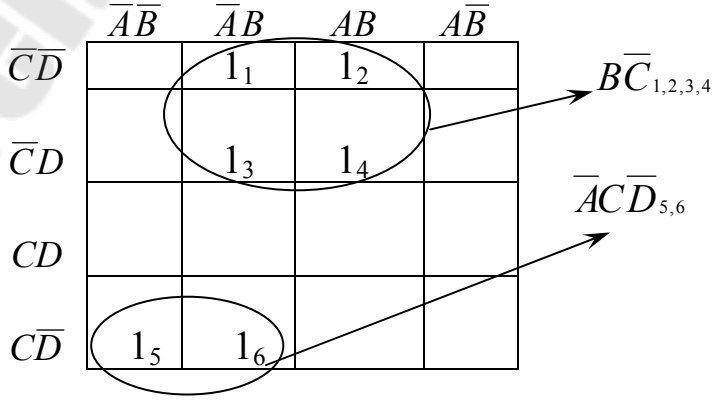


Рис.4.2. Пример минимизации булевой функции $F1$ с помощью карты Карно для 4–х переменных

После этого в нем остается произведение $\overline{A}CD$. В верхнем контуре из четырех единиц 1_1, 1_2, 1_3 и 1_4, соответствующие первым четырем слагаемым в исходном булевом выражении попарно опускаются A и $\overline{A}$, D и $\overline{D}$, так что в результате этого верхний контур дает произведение $B\overline{C}$.

Полученное упрощенное выражение: $F1 = BC + \overline{A}C\overline{D}$.

Рассмотрим еще один пример.

$F2 = \overline{A}\overline{B}\overline{C}\overline{D}_1 + \overline{A}B\overline{C}\overline{D}_2 + AB\overline{C}\overline{D}_3 + \overline{A}\overline{B}C\overline{D}_4 + \overline{A}BC\overline{D}_5$.

Во втором примере минимизации булевой функции $F2$ (см. рис.4.3) контур из двух единиц 1_2 и 1_3 , соответствующие второму и третьему слагаемым в исходном булевом выражении, дает возможность опустить A и $\overline{A}$. После этого в нем остается произведение $B\overline{C}\overline{D}$. В контуре из четырех единиц 1_1, 1_2, 1_4 и 1_5 , соответствующие другим четырем слагаемым из исходного булева выражения, попарно опускаются B и $\overline{B}$, C и $\overline{C}$, так что в результате этого верхний контур дает произведение $\overline{A}\overline{D}$.

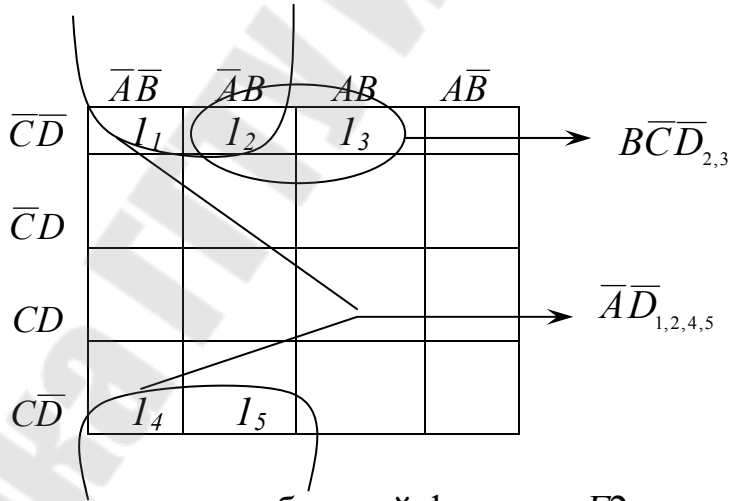


Рис.4.3. Пример минимизации булевой функции $F2$ с помощью карты Карно для 4-х переменных.

Полученное упрощенное выражение: $F2 = \overline{A}\overline{D} + B\overline{C}\overline{D}$. Карта Карно представляется в данном случае свернутой в цилиндр, в котором верхний край совмещается с нижним.

Этот пример показывает также, что контуры могут накладываться друг на друга (сколько угодно раз).

Построение комбинационных логических схем по заданным булевым выражениям

Как правило, построение и расчет любой схемы осуществляется, начиная с ее выхода.

Допустим, задано булево выражение: $F = \overline{B}A + B\overline{A} + C\overline{B}$.

1. *Первый этап*: выполняется логическое сложение, логическую операция ИЛИ, считая входными переменными функции $\overline{B}A$, $B\overline{A}$ и $C\overline{B}$ (рис.4.4).

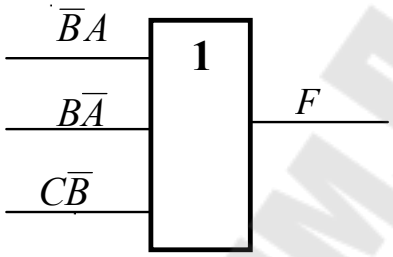


Рис.4.4. Первый этап построения комбинационных логических схем по заданному булеву выражению

2. *Второй этап*: к входам элемента ИЛИ подключаются логические элементы И, входными переменными которых являются уже A, B, C и их инверсии (рис.4.5):

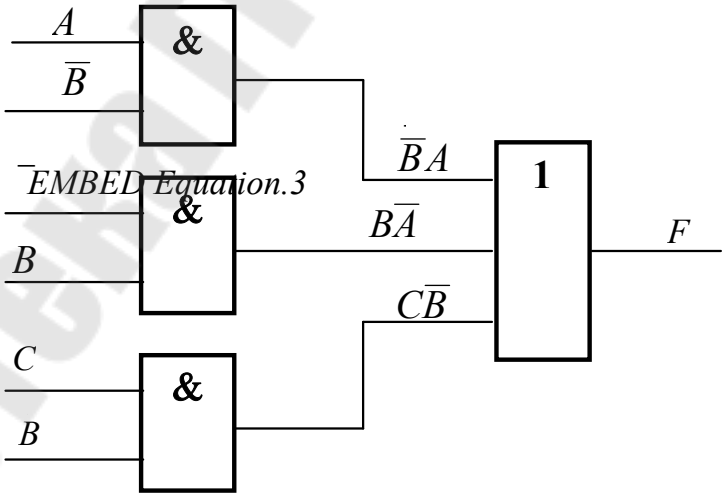


Рис.4.5. Второй этап построения комбинационных логических схем по заданному булеву выражению

3. *Третий этап*: для получения инверсий $\overline{A}$ и $\overline{B}$ на соответствующих входах ставят инверторы (рис.4.6):

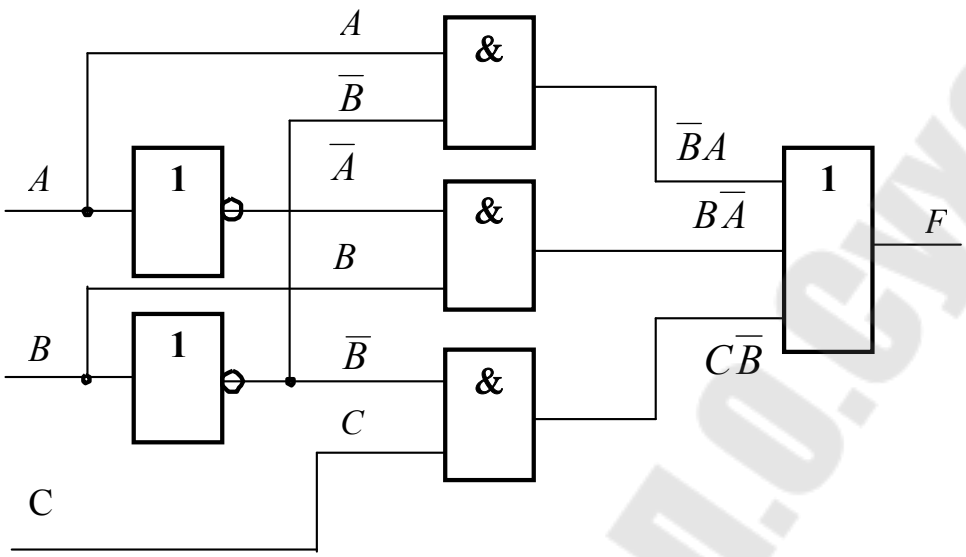


Рис.4.6. Третий этап построения комбинационных логических схем по заданному булеву выражению

Данное построение основано на следующей особенности, – поскольку значениями логических функций могут быть только нули и единицы, то любые логические функции могут быть представлены как аргументы других более сложных функций.

Таким образом, построение комбинационной логической схемы осуществляется с выхода к входу.

ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ №4

Номер варианта задания

а) выбирается равным номеру *N* студента по журналу группы;

б) выбирается равным (30 - *N*), где *N* - номер студента по журналу группы

Таблица 4.1

N	*F*1	*F*2
1	00110111	1111010011011000
2	11100110	0011000111110110
3	10111010	1001011011110000
4	11111000	1111000111100110
5	00011111	1001011011110011
6	11010011	1100001110111011
7	01111010	1001101111110001
8	11001110	1001100011100001
9	11100101	1101000111000111
10	10111100	1111011100011000
11	10010111	1110001111110001
12	11100110	1001110011011111
13	11011100	1111100011100001
14	10001111	0001111111000111
15	11110001	0011111000011111
16	01101110	1111010000011000
17	11001101	1001100011100001
18	01110101	1001100011100001
19	11110001	1111000111000111
20	00111110	0011110111100110
21	10100111	0011111111000111
22	11101100	1100001110100000
23	10011101	1001110000011111
24	11001011	1111011100011001
25	01111001	1110011011000001
26	00101111	0001111110011101
27	11001101	1001101111100001
28	10111001	0001000111111111
29	00011111	0011110011000111
30	11101010	1111010011111000

1. В последнюю строку таблицы истинности (табл. 4.2) вписать заданное восьмиразрядное двоичное число $F1$, взятое согласно номеру варианта из таблицы 4.1. а в последнюю строку таблицы истинности (табл.4.3) – шестнадцатиразрядное двоичное число $F2$.

2. Представить логические функции $F1$ и $F2$, заданные таблицами истинности, в совершенной дизъюнктивной нормальной форме.

Таблица 4.2

A	0	0	0	0	1	1	1	1
B	0	0	1	1	0	0	1	1
C	0	1	0	1	0	1	0	1
*F*1								

Таблица 4.3

A	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
B	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
C	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
D	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
*F*2																

3. Выполнить минимизацию функций, записанных в СДНФ при помощи карт Карно.

4. Построить схемы, реализующие заданные функции, согласно упрощенным по карте Карно булевым выражения в отрицательной логике. Функцию $F1$ построить в базисе ИЛИ-НЕ, функцию $F2$ построить в базисе И-НЕ.

5. ПОСТРОЕНИЕ ПРЕОБРАЗОВАТЕЛЕЙ КОДОВ ПО ЗАДАННОЙ ТАБЛИЦЕ ИСТИННОСТИ

Преобразователи кодов

Под построением преобразователей кодов понимается синтез схем, реализующих преобразование *n*-разрядных двоичных чисел, представляющих информацию в каком-либо входном заданном коде, в *m*-разрядные двоичные числа, представляющие эту информацию в другом коде (выходном). Наиболее распространены следующие два подхода к построению преобразователей кодов.

Первый подход – синтез *n* независимых одновыходных функций по заданной таблице истинности – таблице соответствия кодов.

Допустим, имеется таблица истинности преобразователя 2-разрядного двоичного кода a_1a_0 в 3-разрядный $b_2b_1b_0$ (см.табл.5.1):

Таблица 5.1

a_1	a_0	b_2	b_1	b_0
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	1	0

Считая b_0, b_1 и b_2 независимыми одновыходными функциями, запишем для каждой из них булевы выражения:

$$b_2 = \overline{a}_1\overline{a}_0 + a_1a_0,$$
$$b_1 = a_1\overline{a}_0 + a_1a_0,$$
$$b_0 = \overline{a}_1a_0$$

Используя приведенные булевы выражения и учитывая, что произведение $a_1\overline{a}_0$ встречается в двух выходных функциях, составляем схему преобразователя на логических элементах (рис.5.1):

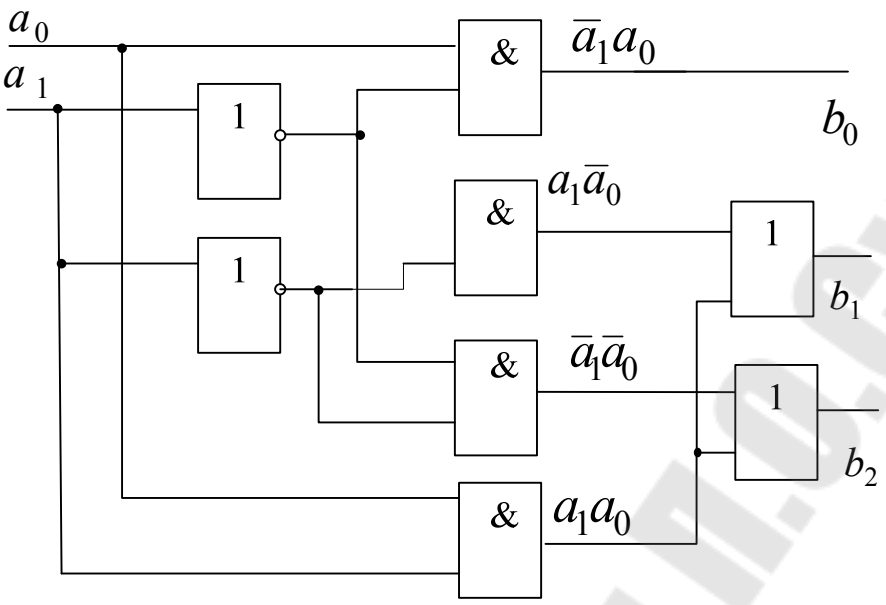


Рис.5.1 Пример построения преобразователя кодов по заданному булеву выражению.

Особенности построения логических схем в инвертирующих базисах

Первой особенностью построения логических схем в инвертирующих базисах считается непрямая зависимость между простотой булева выражения и минимальностью соответствующей ему логической схемы. Другими словами, самое минимизированное булево выражение не всегда дает схему, минимальную по количеству инвертирующих логических элементов.

Для доказательства этого построим в инвертирующим базисе И–НЕ схему для реализации двухвходовой функции «Исключающее ИЛИ», булево выражение которого в двухвходовом случае совпадает с булевым выражением для сумматора по модулю два, и имеет следующий вид: $F = \overline{A}B + A\overline{B}$.

Поскольку исходное булево выражение для двухвходовой функции «Исключающее ИЛИ», кроме функций И и НЕ содержит и функцию ИЛИ, то, чтобы исключить ИЛИ, преобразуем его следующим образом (рис.5.2):

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

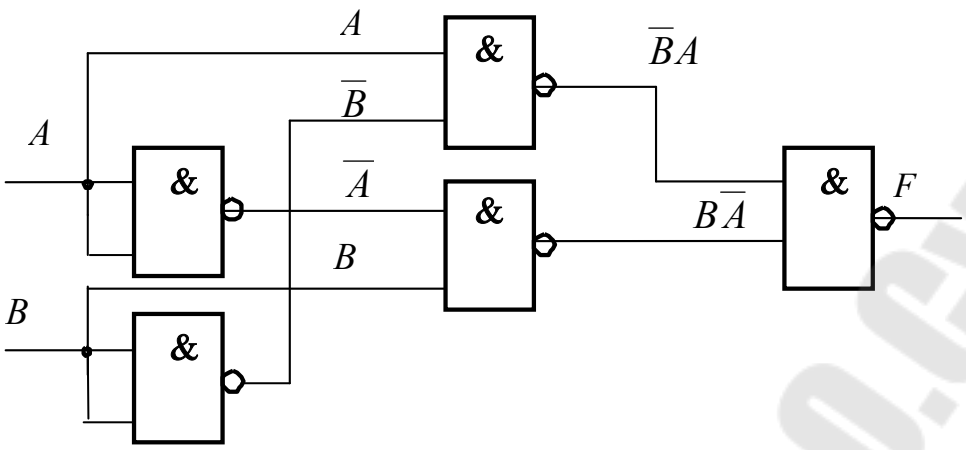


Рис.5.2 Синтез схемы в инвертирующем базисе (И-НЕ)
по булеву выражению $F = \overline{\overline{A}B} \cdot \overline{A\overline{B}}$

Полученное выражение не является минимальным, а чтобы получить действительно минимальное выражение произведем над исходным выражением следующий ряд преобразований:

$$F = \overline{A}B + A\overline{B} = \overline{A}B + A\overline{B} + A\overline{A} + B\overline{B} = \overline{A}(B + A) + \overline{B}(A + B) =$$

$$= (\overline{A} + \overline{B})(A + B) = \overline{AB} \cdot \overline{\overline{A}\overline{B}}$$

В соответствии с последним минимальным выражением построим в инвертирующем базисе схему из логических элементов И–НЕ (рис.5.3):

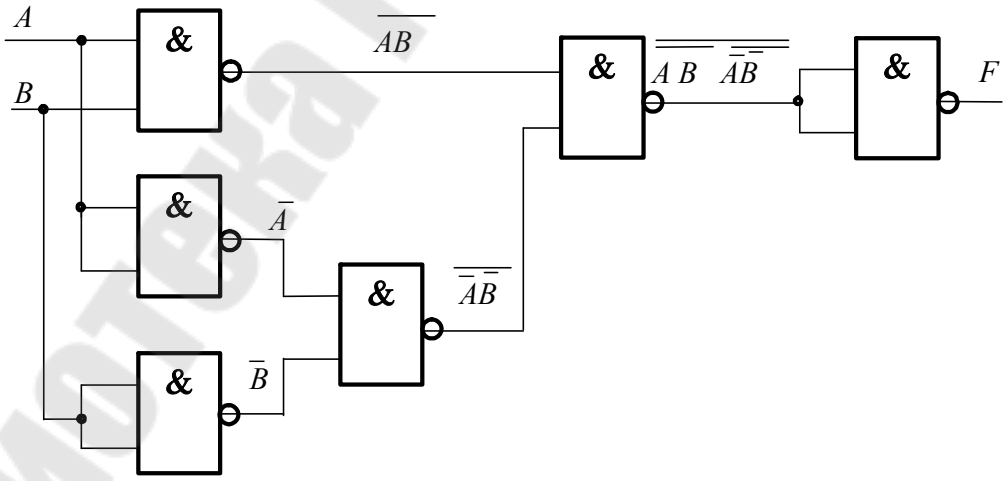


Рис.5.3 Синтез схемы в инвертирующем базисе (И-НЕ)
по булеву выражению $F = \overline{AB} \cdot \overline{\overline{A}\overline{B}}$

Теперь попробуем получить такое булево выражение, которое могло бы привести к более простой логической схеме. Для этого над минимальным выражением произведем следующий ряд преобразований:

$$F = AB + \overline{A}\,\overline{B} = \overline{AB + \overline{A}\,\overline{B}} = \overline{AB + \overline{A}\,\overline{B} + A\overline{B}B + A\overline{A}B} =$$

$$= \overline{AB \cdot AB + AB \cdot \overline{B} + \overline{A}\,\overline{B} + \overline{A} \cdot AB} = \overline{AB(AB + A\overline{B}) + \overline{A}(AB + \overline{B})} =$$

$$= (AB + \overline{B}) \cdot (AB + \overline{A}) = \overline{\overline{AB}B} \cdot \overline{\overline{AB}A}.$$

В соответствии с последним булевым выражением построим в инвертирующем базисе схему из логических элементов И–НЕ (рис.5.4.):

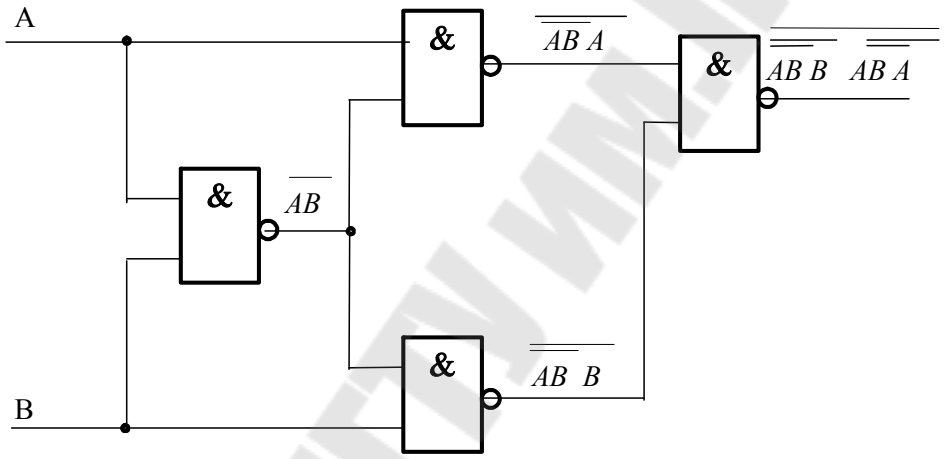


Рис.5.4. Синтез схемы в инвертирующем базисе (И-НЕ) по булеву выражению $F = \overline{\overline{AB}B} \cdot \overline{\overline{AB}A}$

Как видно данная схема оказывается в полтора раза проще, чем предыдущая, несмотря на то, что булево выражение, в соответствии с которым она построена, явно сложнее, чем полученное ранее минимальное выражение. Таким образом, очевидно, что самое минимизированное булево выражение не всегда дает минимальную по количеству инвертирующих логических элементов схему.

Вторая особенность построения логических схем в инвертирующих базисах приводится без доказательства:

Если в произвольной комбинационной схеме :

– проинвертировать все входные и выходные сигналы;

– все элементы И-НЕ заменить на ИЛИ-НЕ, то реализуемая схемой функция не изменится.

Рассмотрим пример синтеза схемы в базисе ИЛИ-НЕ двухвходовой функции «Исключающее ИЛИ» по следующему булеву выражению: $F = B\overline{A} + A\overline{B} = \overline{\overline{B\overline{A}} + \overline{A\overline{B}}} = \overline{\overline{\overline{B} + A} + \overline{\overline{A} + B}}$ (рис.5.5).

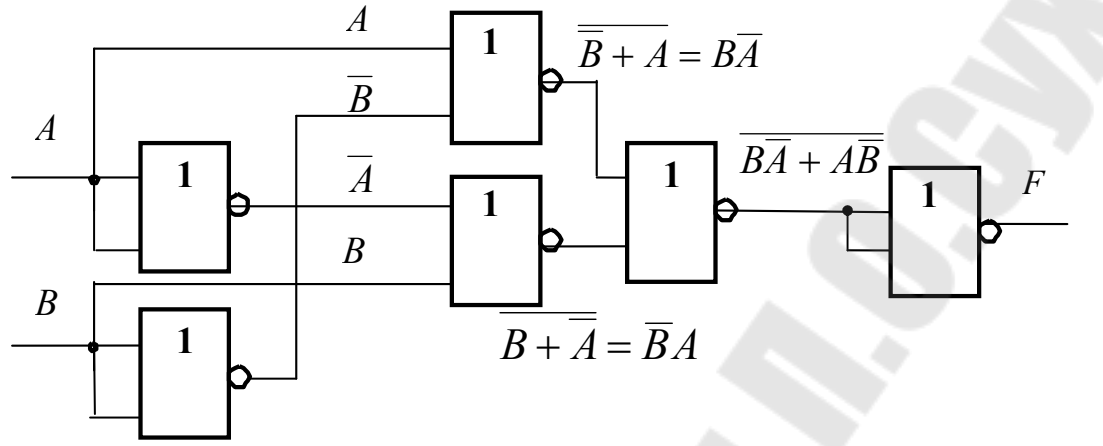


Рис.5.5. Синтез схемы в инвертирующем базисе ИЛИ-НЕ по булеву выражению $F = \overline{\overline{\overline{B} + A} + \overline{\overline{A} + B}}$

Второй способ построения преобразователей кодов – синтез схемы по принципу «дешифратор-шифратор». Для того, чтобы синтезировать схему преобразователя кодов по таблице 5.1 требуется иметь представление о структуре комбинационных устройств: дешифратора и шифратора.

Рассмотрим, что представляет собой дешифратор.

Дешифратор – это цифровое устройство, в котором входной двоичный *n*-разрядный код преобразуется в выходной унарный (один их всех), т.е. активный уровень присутствует только на одном из выходов, а именно на том, номер которого соответствует этой цифре в двоичном коде. Активным может быть как высокий уровень («*H*» – *High*, т.е. единица для положительной логики, так и низкий («*L*» – *Low*, , т.е. ноль, для отрицательной логики).

Число выходов дешифратора $N = 2^n$ где n - число разрядов *входного* кода дешифратора.Составим таблицу состояний двухразрядного дешифратора (табл.5.2).

Учитывая, что активный уровень, появляется на каждом из выходов дешифратора только для одной из входных комбинаций, булевы выражения выходных функций дешифратора содержат только одно произведение входных переменных или их дополнений, а в правой части таблицы истинности единица может быть только в одном столбце каждой строки.

Таблица 5.2. Таблица состояний двухразрядного дешифратора:

Входы		Выходы			
a_1	a_0	d_3	d_2	d_1	d_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Поэтому схема дешифратора будет содержать только логические элементы И (конъюнкторы), перемножающие входные переменные и их дополнения, реализованных по булевым выражениям, полученным из таблицы состояний: $d_0 = \overline{a}_1\overline{a}_0$, $d_1 = \overline{a}_1 a_0$, $d_2 = a_1\overline{a}_0$, $d_3 = a_0 a_1$.

На рис.5.6. показана схема 2-разрядного дешифратора, построенная на логических элементах И по приведенной выше таблице истинности.

В случае, если в схеме многоразрядного дешифратора не используется какая-либо входная комбинация переменных, такой дешифратор называется *неполным*.

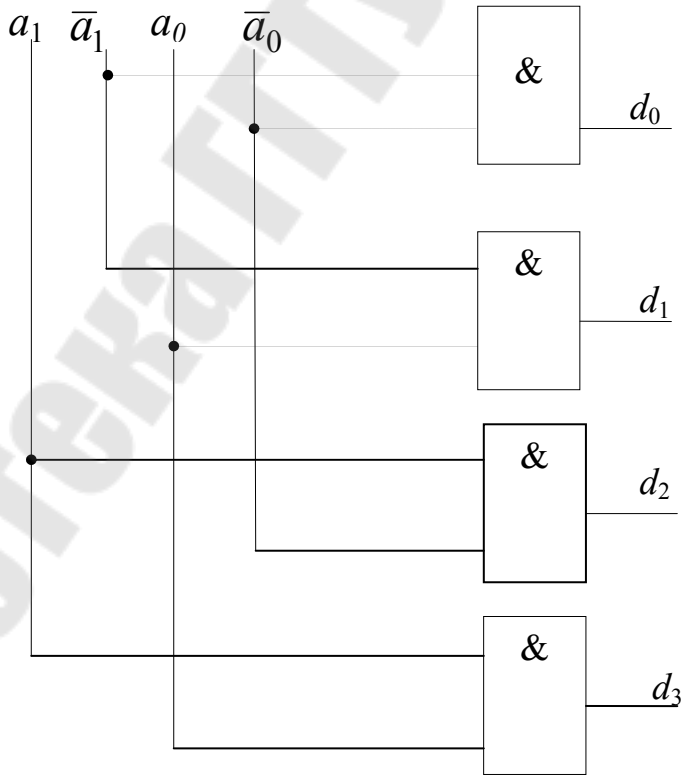


Рис.5.6. Пример построения 2-разрядного дешифратора

На рисунке 5.7. изображено условное графическое изображение дешифратора с разрядностью 3:8

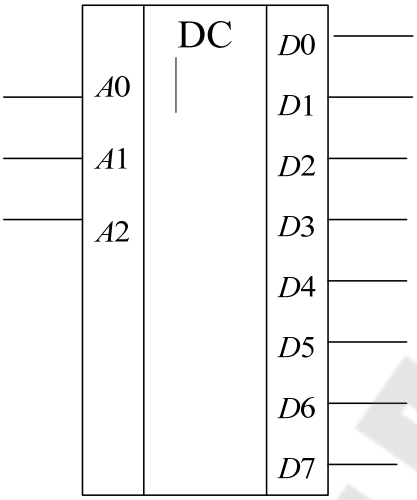


Рис.5.7. Условное графическое изображение дешифратора (от «*DC – DECODER*») с разрядностью 3:8

Шифратор выполняет функцию, обратную дешифратору. Только один из входов шифратора может быть активным, остальные должны оставаться пассивными, т.е. логическую единицу можно подавать только на один из входов, а на все остальные входы при этом надо подавать нули (для положительной логики, для отрицательной – наоборот).

Число входов такого преобразователя, равно $M = 2^n$, где n - число разрядов *выходного* кода шифратора. Учитывая это обстоятельство, запишем таблицу истинности для 3-разрядного шифратора (табл.5.3).

Таблица 5.3.Таблица состояний трехразрядного шифратора

Входы	Выходы		
d	b_2	b_1	b_0
d_0	0	0	0
d_1	0	0	1
d_2	0	1	0
d_3	0	1	1
d_4	1	0	0
d_5	1	0	1
d_6	1	1	0
d_7	1	1	1

Составим булевы выражения на основании табл.5.3: $b_0 = d_1 + d_3 + d_5 + d_7$; $b_1 = d_2 + d_3 + d_6 + d_7$; $b_0 = d_4 + d_5 + d_6 + d_7$.

Схема шифратора, построенная по этим выражениям должна содержать только элементы ИЛИ (дизъюнкторы) для положительной логики, суммирующие те входные переменные, для которых есть единица в соответствующем разряде выходного кода, (или только элементы И (конъюнкторы) для отрицательной логики).

На рис.5.8. представлена схема, реализующая 3-х разрядный шифратор.

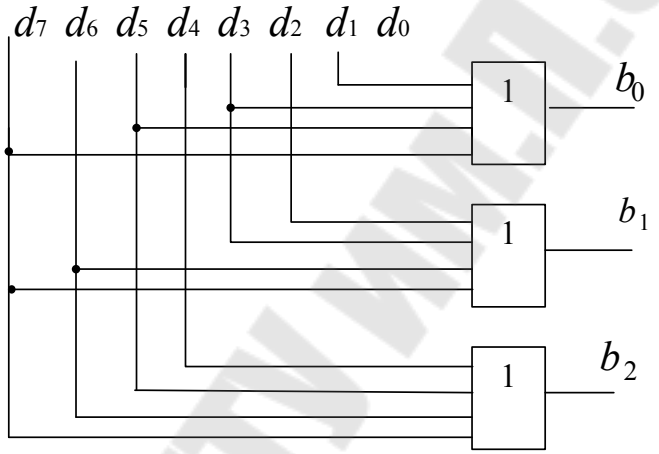


Рис.5.8. Схема, реализующая 3-х разрядный шифратор

Если в схеме преобразователя кодов один из выходных разрядов всегда неактивный, то соответствующий элемент ИЛИ не включается в схему шифратора и такой шифратор также называется *неполным.*

На рисунке 5.9 изображено условное графическое изображение дешифратора с разрядностью 8:3

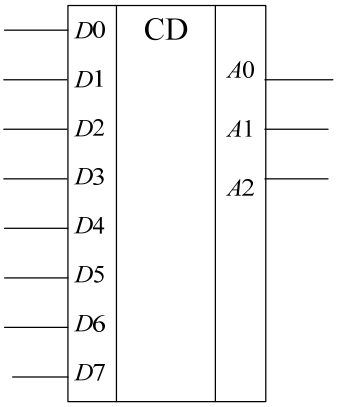


Рис.5.9. Условное графическое изображение шифратора дешифратора (от «*CD – CODER*») с разрядностью 8:3

Метод построения преобразователя кодов способом «дешифратор-шифратор» заключается в следующем:

Схема преобразователя кода образуется соединением выходов дешифратора и входов шифратора в соответствии с таблицей входных и выходных переменных преобразователя, при переводе двоичного кода в десятичный. Таблица 5.1. в этом случае преобразуется в таблицу 5.4.

Таблица 5.4. Таблица состояний преобразователя кодов

a_1	a_0	b_2	b_1	b_0	A_{10}	B_{10}
0	0	1	0	0	0	4
0	1	0	0	1	1	1
1	0	0	1	0	2	2
1	1	1	1	0	3	6

Преобразователь кодов, синтезированный вторым способом по таблице 5.4 представлен на рисунке 5.10.

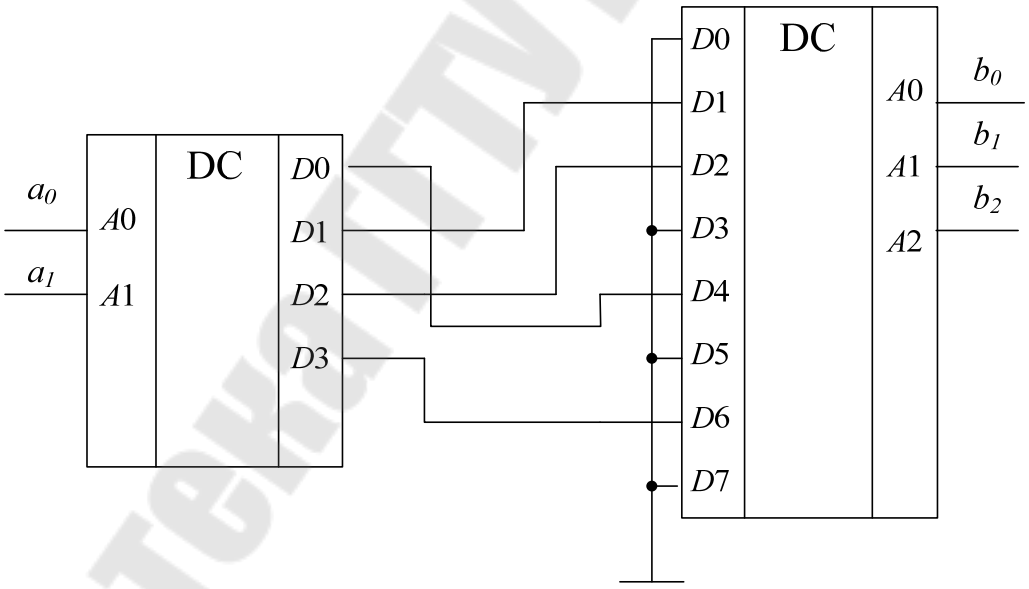


Рис.5.10 Преобразователь кодов, синтезированный вторым способом по таблице 5.4

ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ № 5

Номер варианта в таблицах 5.5 и 5.6 выбирается равным N – номеру студента по журналу группы.

Таблица 5.5

№ вар	Y
1	00110111
2	11100110
3	10111010
4	11111000
5	00011111
6	11010011
7	01111010
8	11001110
9	11100101
10	10111100
11	10010111
12	11100110
13	11011100
14	10001111
15	11110001
16	01101110
17	11001101
18	01110101
19	11110001
20	00111110
21	10100111
22	11101100
23	10011101
24	11001011
25	01111001
26	00101111
27	11001101
28	10111001
29	00011111
30	11101010

Таблица 5.6

№ вар	b_2	b_1	b_0
1, 6, 12, 15,28	1	1	0
	0	0	1
	0	1	1
	0	1	1
	1	0	0
2, 7, 11, 17, 20	1	1	0
	1	0	1
	0	1	0
	0	1	1
	0	0	0
3, 5, 10, 14 23,26	1	0	1
	1	0	1
	1	1	1
	0	1	0
	0	1	0
8, 10, 16, 24,29	1	0	1
	0	0	0
	1	1	1
	1	0	1
	0	1	1
13, 18 21, 25 27	1	1	1
	0	1	0
	1	1	0
	1	0	0
	0	0	1
4, 9,22, 19, 30	1	0	0
	0	1	1
	1	1	1
	1	1	0

1. В последнюю строку таблицы истинности (табл. 5.7) вписать восьмиразрядное двоичное число *Y*, взятое из таблицы 5.5, получить таблицу истинности логической функции для четырех входных переменных преобразователя кодов (дешифратора).

Таблица 5.7 Таблица истинности логической функции

a_0	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**
a_1	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**
a_2	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**
a_3	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**1**
Y								

2. Получить таблицу истинности преобразователя 4-разрядного ($a_3\ a_2\ a_1\ a_0$) двоичного кода в 3-разрядный ($b_2\ b_1\ b_0$); выходные переменные ($b_2\ b_1\ b_0$) взять из таблицы 5.6.

a_3	a_2	a_1	a_0	b_2	b_1	b_0
Входы			Выходы			

3. Выполнить синтез преобразователя 4-разрядного ($a_3\ a_2\ a_1\ a_0$) двоичного кода в 3-разрядный ($b_2\ b_1\ b_0$), используя логические элементы И-НЕ.

4. Выполнить синтез преобразователя 3-разрядного ($b_2\ b_1\ b_0$) двоичного кода в 4-разрядный, ($a_3\ a_2\ a_1\ a_0$) используя логические элементы ИЛИ-НЕ.

b_2	b_1	b_0	a_3	a_2	a_1	a_0
Входы			Выходы			

5. Выполнить синтез преобразователя 4-разрядного ($a_3\ a_2\ a_1\ a_0$) двоичного кода в 3-разрядный ($b_2\ b_1\ b_0$) и преобразователя 3-разрядного ($b_2\ b_1\ b_0$) двоичного кода в 4-разрядный, ($a_3\ a_2\ a_1\ a_0$) методом шифратор-дешифратор.

6. МУЛЬТИПЛЕКСОР КАК УНИВЕРСАЛЬНЫЙ ЛОГИЧЕСКИЙ ЭЛЕМЕНТ

Мультиплексор

Мультиплексированием (*MULTIPLEX*) называют передачу данных от нескольких источников по одному каналу поочередно.

Мультиплексор (или селектор данных) – это функциональное устройство, осуществляющее подключение (коммутацию) одного из нескольких информационных входов данных к выходу. Номер выбранного входа соответствует двоичному коду, подаваемому на адресные входы мультиплексора. В цифровой схемотехнике мультиплексор имеет один выход; m информационных входов данных $D_0, D_1, D_2, D_3, \ldots, D_m$ и n адресных входов A_1, A_2, $A_3, \ldots, A_n$, причем $m = 2^n$.

На рисунке 6.1 приведен пример условного обозначения мультиплексора на схемах. Число информационных входов D_0–D_3, коммутируемых на выход Y, составляет m=4 . Такой мультиплексор имеет размерность 4:1.

Рис.6.1. Условное графическое обозначение мультиплексора 4:1 с разрешающим входом

Булево выражение для выходной функции мультиплексора вглядит следующим образом:

$$Y = \overline{A}_1 \overline{A}_0 D_0 + \overline{A}_1 A_0 D_1 + A_1 \overline{A}_0 D_2 + A_1 A_0 D_3$$

При учете разрешающего входа E, булево выражение запишется следующим образом:

$$Y = E\left(\overline{A}_1 \overline{A}_0 D_0 + \overline{A}_1 A_0 D_1 + A_1 \overline{A}_0 D_2 + A_1 A_0 D_3\right)$$

Таблица истинности мультиплексора с двумя адресными входами приведена ниже (табл.6.1).

Табл.6.1. Таблица состояний мультиплексора.

E	A_1	A_0	Y
1	0	0	D_0
1	0	1	D_1
1	1	0	D_2
1	1	1	D_3

Наличие разрешающего входа E позволяет увеличивать число информационных входов вдвое путем последовательного соединения разрешающих входов двух мультиплексоров (наращивание разрядности).

Кроме прямого назначения, мультиплексор может быть использован как универсальный логический элемент, т.е. на основе мультиплексора возможно реализация любой заданной логической функции.

Если на адресные входы мультиплексора подать входные переменные, зная, какой выходной уровень должен отвечать каждой комбинации этих переменных, то, предварительно установив на входах данных уровни напряжения «лог.1» и «лог.0», согласно таблице истинности, можно получить устройство, реализующее заданную логическую функцию.

Использование мультиплексоров дает следующие преимущества при синтезе комбинационных схем:

- не требуется упрощение логического выражения, составленного по таблице истинности;
- минимизируется число требуемых интегральных схем;
- соответственно, синтез комбинационных схем упрощается.

Для реализации заданной таблицей истинности логической функции с использованием мультиплексора необходимо определить десятичные номера каждого из логических выражений таблицы истинности, для которых выходной сигнал принимает значение: $Y = 1$. Входы мультиплексора, соответствующие этим номерам, соединить с «лог.1». Все остальные входы соединить с «лог. 0.» Входные переменные заданной функции подать на адресные входы, в соответствии с весом разрядов.

Рассмотрим пример: пусть требуется реализовать на мультиплексоре логическую функцию, заданную таблицей истинности (см. табл.6.2).

Таблица 6.2. Пример задания логической функции

b_2	b_1	b_0	B_{10}	Y
0	0	0	0	0
0	0	1	1	1
0	1	0	2	0
0	1	1	3	1
1	0	0	4	0
1	0	1	5	0
1	1	0	6	1
1	1	1	7	1

Определим десятичные номера каждого из логических выражений таблицы истинности, для которых выходной сигнал принимает значение: Y=1. Это десятичные цифры 1,3,6,7. На входы данных мультиплексора, соответствующие этим номерам, подадим «лог.1», остальные соединим с землёй.

Входные переменные заданной функции ($b_2b_1b_0$) подадём на адресные входы мультиплексора. Реализация заданной логической функции на мультиплексоре представлена на рис.6.2.

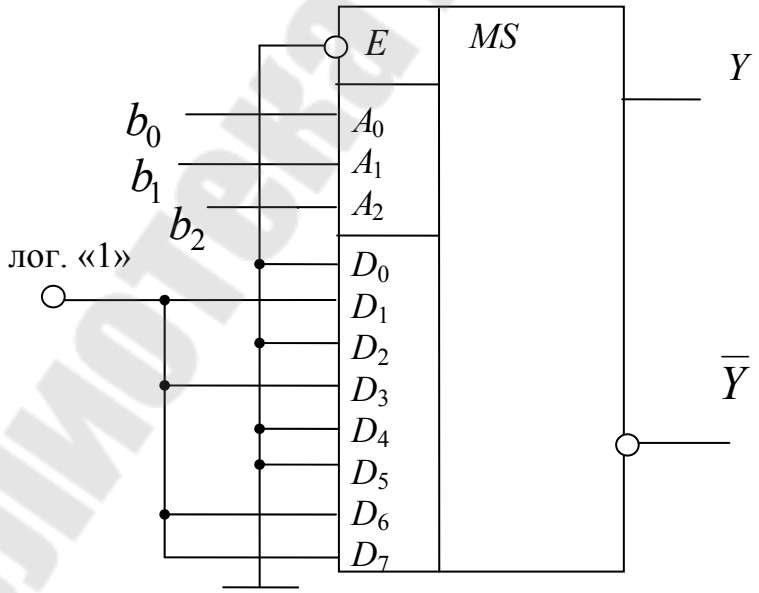


Рис.6.2. Реализация заданной табл.6.2. логической функции

Рассмотрим пример реализация функции (m+1) переменных на мультиплексорах с m адресными входами. Реализуем функцию четырех переменных ($b_3b_2b_1b_0$), заданную таблицей истинности (табл.6.3) при помощи мультиплексора 8:1.

Таблица 6.3.
Логическая функция
4-х переменных

b_3	b_2	b_1	b_0	Y
0	0	0	0	0
0	0	0	1	1
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	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Таблица 6.4. Совпадение функции
с «выделенной переменной»

Переменные				Значение функции	
b_3	b_2	b_1	b_0	Y	
0	0	0	**0**	**0**	Совпадает с b_0
0	0	0	**1**	**1**	
0	0	1	0	**0**	«лог.0»
0	0	1	1	**0**	
0	1	0	0	**0**	«лог.0»
0	1	0	1	**0**	
0	1	1	0	**0**	«лог.0»
0	1	1	1	**0**	
1	0	0	**0**	**1**	Совпадает с $\overline{b_0}$
1	0	0	**1**	**0**	
1	0	1	**0**	**1**	Совпадает с $\overline{b_0}$
1	0	1	**1**	**0**	
1	1	0	0	**1**	«лог.1»
1	1	0	1	**1**	
1	1	1	0	**1**	«лог.1»
1	1	1	1	**1**	

Функция четырех переменных может быть реализована на мультиплексоре размерностью 8:1 следующим образом: переменные заданной функции ($b_3b_2b_1$) подключим к адресным входам A_2, A_1, A_0 соответственно. Переменную b_0, неподключенную к адресным входам, называют «выделенной». Без выделенной переменной наборы переменных $b_3b_2b_1$ образуют пары. В таблице 6.4 эти пары выделены пунктирными линиями.

Теперь рассмотрим соотношения между выделенной переменной b_0 и выходом Y для каждой пары.

При этом возможны четыре варианта:

– выход не зависит от переменной и равен нулю;

– выход не зависит от переменной b_0 и равен единице;

– выход Y зависит от переменной b_0 и равен ей;

– выход Y зависит от переменной b_0 и совпадает с ее инверсией $\overline{b_0}$.

Исходя из вышеизложенного, на информационные входы мультиплексора и подается лог. 0, лог. 1, D или $\overline{D}$. Пример реализации функции 4-х переменных, заданной таблицей 6.3 на базе мультиплексоре с тремя адресными входами изображен на рис.6.3.

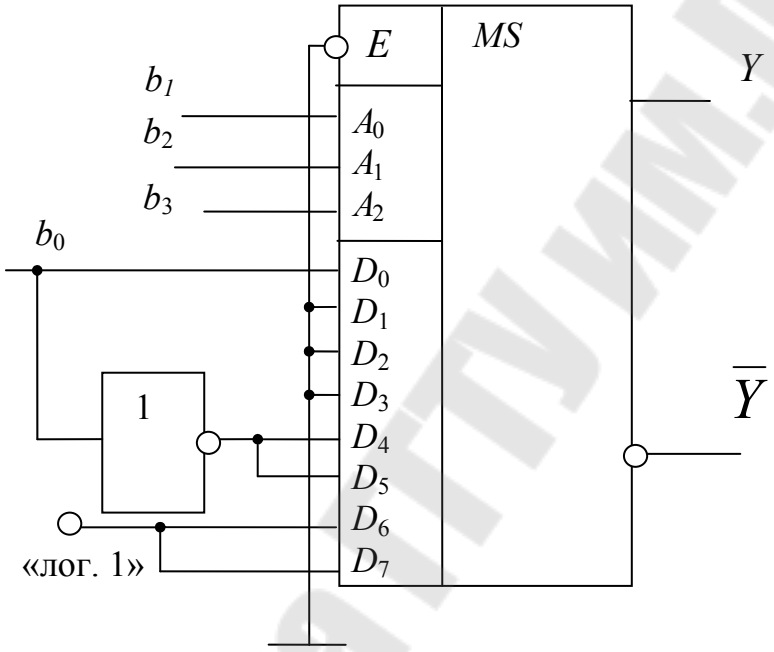


Рис.6.3. Реализация заданной табл.6.3. функции на мультиплексоре

ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ № 6

Номер варианта № в таблице 6.5 выбирается равным номеру студента по журналу группы.

Таблица 6.5

№ вар	Y
1	00110111
2	11100110
3	10111010
4	11111000
5	00011111
6	11010011
7	01111010
8	11001110
9	11100101
10	10111100
11	10010111
12	11100110
13	11011100
14	10001111
15	11110001
16	01101110
17	11001101
18	01110101
19	11110001
20	00111110
21	10100111
22	11101100
23	10011101
24	11001011
25	01111001
26	00101111
27	11001101
28	10111001
29	00011111
30	11101010

1. В последнюю строку таблицы истинности (табл. 6.6) вписать заданное восьмиразрядное двоичное число *Y*, взятое из таблицы 6.5. Получить таблицу истинности логической функции для четырех входных переменных.

Таблица 6.6.Таблица истинности логической функции

b_0	**0**	**0**	**1**	**0**	**1**	**1**	**1**	**1**
b_1	**0**	**0**	**0**	**1**	**0**	**1**	**1**	**1**
b_2	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**
a_3	**1**	**1**	**0**	**1**	**1**	**0**	**0**	**1**
Y								

3. Реализовать логическую функцию, заданную полученной таблицей истинности на мультиплексоре с 4 адресными входами (мультиплексор 16:1).

4. В последнюю строку таблицы истинности (табл. 6.6) вписать восьмиразрядное двоичное число, равное инверсии заданного числа $\overline{Y}$. Получить таблицу для четырех переменных.

5. Реализовать логическую функцию, заданную полученной таблицей истинности на мультиплексоре с 3 адресными входами (мультиплексор 8:1).

7. СИНТЕЗ ПОСЛЕДОВАТЕЛЬНОСТНЫХ ЦИФРОВЫХ УСТРОЙСТВ

Триггеры

Простейшим представителем последовательностных устройств является триггер, обобщенная структура которого имеет вид представленный на рис. 7.1, а функция, описывающая состояние выхода в *i*-тый момент времени выглядит следующим образом:

$$y_i = f\left(X_i, y_{i-1} \right)$$

Здесь X_i – входное воздействие, представляющее собой набор сигналов $x_0, x_1 \ldots x_{n-1}$. Таким образом, выходное состояние триггера зависит как от входного воздействия, так и от его предшествующего состояния. Как следует из структуры, в состав триггера входит комбинационное устройство (КУ) и узел памяти (УП), кроме того присутствует цепь обратной связи.

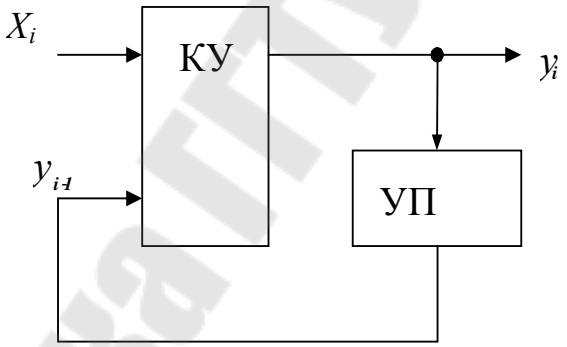


Рис.7.1. Структура триггера, как последовательностного устройства

Приведенная выше функция называется *функцией возбуждения* и обычно записывается в форме $y_{i+1} = f\left(X_i, y_i\right)$, которая представляет состояние триггера после воздействия. Часто состояние y_{i+1} обозначается y^+.

Пусть X_i представляет собой двухразрядное слово представленное сигналами R и S. Пусть при $R = 0, S = 0$ триггер сохраняет свое предыдущее состояние выхода Q, то есть $Q^+ = Q$, если

$R=0, S=1$ переходит в единичное, а когда $R=1, S=0$ – в нулевое. Последнее означает, что, $Q^+=0$, независимо от того в каком состоянии находился триггер до воздействия. Данный алгоритм характерен для большинства разновидностей триггеров, которые отличаются друг от друга лишь по реакции на комбинацию сигналов $R=S=1$ (рис. 7.2). Возможны следующие варианты: $Q^+=0$, $Q^+=1$, $Q^+=Q$, $Q^+=\overline{Q}$ и наконец эту комбинацию можно считать запрещенной, то есть не подавать ее на входы управления. Формально при ее наличии состояние триггера будет неопределенным $Q^+=*$, так как не подавая данную комбинацию нельзя ничего сказать и о выходном состоянии.

S	R	Q^+_{RS}	Q^+_R	Q^+_S	Q^+_E	Q^+_{JK}
0	0	Q	Q	Q	Q	Q
0	1	0	0	0	0	0
1	0	1	1	1	1	1
1	1	*	0	1	Q	$\overline{Q}$

Рис.7.2. Таблица функционирования различных триггеров;

Первый вариант триггера (рис.7.2) называется *RS*-триггером, который является простейшим в этом классе второй – *R*-триггером, третий – *S*-триггером, , четвертый – *E*, а последний *JK*-триггером. Функции возбуждения этих триггеров должны описываться соотношениями вида $Q^+=f(R,S,Q)$. То, что состояние таких устройств зависит от комбинации входных сигналов *R* и *S*, следует из заданного алгоритма работы. Однако, так как сохранение комбинации управляющих сигналов к следующему моменту времени, формально может считаться новым воздействием, то при переходе $R=0, S=0 \rightarrow R=0, S=0$ Q^+ останется равным Q, следовательно в общем случае можно записать, что $Q^+=f(Q)$. Таким образом предложенный алгоритм описывает работу некоторого последовательностного устройства.

Для синтеза *RS*-триггера, в состав переменных необходимо ввести значения Q. Тогда таблица его функционирования примет

такой вид, представленный на рис. 7.3,а). Часто триггер кроме прямого имеет и инверсный выход $\overline{Q}$. Условно графическое обозначение такого триггера на принципиальных схемах приведено на рис. 7.3,б).

S	R	Q	Q^+_{RS}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	*
1	1	1	*

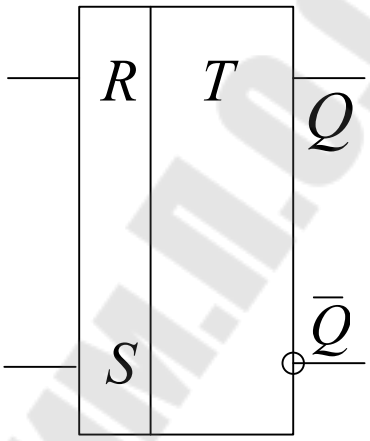


Рис.7.3. *RS*-триггер: а) таблица функционирования;
б) УГО.

Воспользовавшись картой Карно (рис. 7.4), получим $Q^+_{RS} = S + Q\overline{R}$.

Q^+_{RS}	$S\,R$	$S\,\overline{R}$	$\overline{S}\,\overline{R}$	$\overline{S}\,R$
Q	*	1	1	0
$\overline{Q}$	*	1	0	0

Рис.7.4 Карта Карно *RS*-триггера

Особое место среди рассмотренных ранее вариантов занимает *JK* -триггер, который при единичных значениях управляющих сигналов меняет свое состояние на противоположное. Вход J выполняет функции S, а $K - R$ входа на комбинациях, разрешенных для *RS*-триггера. Упрощенная и полная таблицы его функционирования приведены на рис. 7.5.

J	K	Q_{JK}^{+}
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

J	K	Q	Q_{JK}^{+}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Рис.7.5. Тблицы функционирования *JK*-триггер

Полученная после минимизации по карте Карно (рис.7.6) функция возбуждения имеет вид : $Q_{JK}^{+} = Q\overline{K} + \overline{Q}J$.

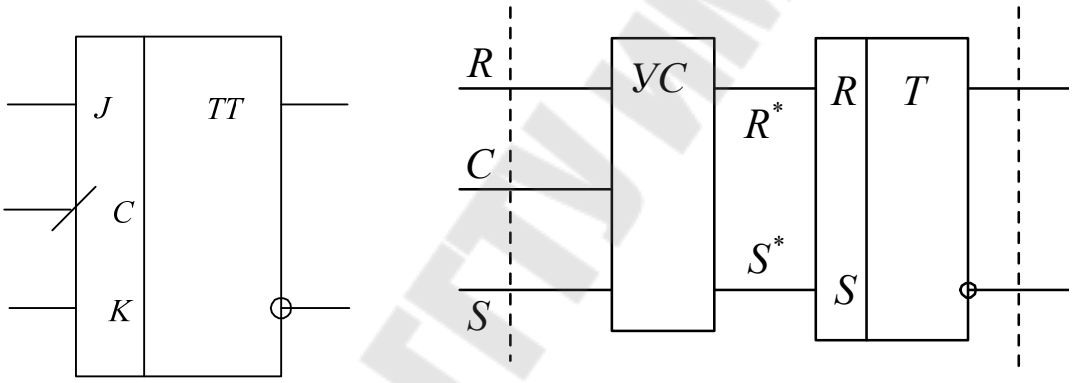
Q_{JK}^{+}	JK	$J\overline{K}$	$\overline{J}\,\overline{K}$	$\overline{J}K$
Q	0	1	1	0
$\overline{Q}$	1	1	0	0

Рис.7.6 Карта Карно *JK*-триггера

Если входы *J* и *K* рассмотренного выше триггера объединить, то получится счетный, или *T*-триггер, имеющий один вход. Он меняющий свое состояние каждый раз с приходом спада управляющего сигнала. Его функция возбуждения может быть получена из соответствующей функции *JK*-триггера и имеет вид $Q_T^{+} = Q\overline{T} + \overline{Q}T$. Так как его состояние меняется на противоположное после каждого воздействия, то формально функция возбуждения *T*-триггера может быть записана в виде $Q_T^{+} = \overline{Q}$.

Все вышерассмотренные триггера относятся к классу асинхронных. Их особенность состоит в том, что они реагируют на изменения управляющих сигналов непосредственно в момент их поступления. В ряде ситуаций, особенно при построении сложных цифровых устройств, различие в моментах прихода таких сигналов на разные триггера приведет к неодновременности их переключения, что может вызвать нарушение работы связанных с ними устройств, в частности вследствие возникновения состязаний.

Эта проблема может быть решена при использовании синхронных или синхронизируемых триггеров, которые реагируют на входные воздействия лишь после прихода специального управляющего сигнала, синхронизации. Соответствующий вход триггера обозначается буквой *C*. Синхронный триггер можно представить как совокупность асинхронного и некоторого устройства синхронизации, подключаемого к его входам (рис. 7.7 б).



а) б)

Рис.7.7. Синхронные триггеры: а) условное графическое обозначение синхронного *JK*-триггера, срабатывающего по фронту синхроимпульса; б) функциональная схема синхронного триггера

Синхронные триггеры, реагирующие на входные воздействия при поступлении определенного уровня сигнала синхронизации называются триггерами, синхронизируемыми уровнем, или *триггерами-защелками*. Кроме них в классе синхронных триггеров имеется еще одна разновидность – триггеры синхронизируемые фронтом (срабатывают по фронту синхроимпульса).

Их отличие от синхронизируемых уровнем состоит в том, что реакция на внешние сигналы управления возникает лишь при поступлении фронта (или среза) сигнала синхронизации. На принципиальных схемах такие триггера обозначаются обычным

образом, но у символа входа синхронизации вводится значок в виде наклонной черты, как показано на рис. 7.7, а).

Счетчики

К группе последовательностных относятся узлы, называемые *счетчиками*. Это устройства, по состоянию выходов которых можно определить количество входных воздействий (импульсов), поступивших на их входы к моменту наблюдения.

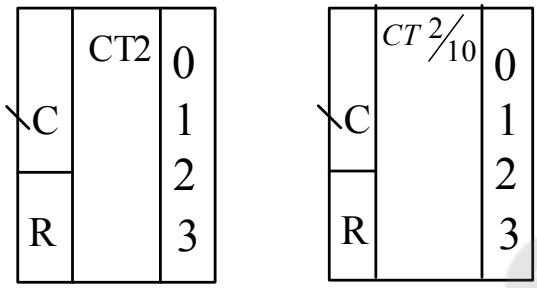
Одним из основных параметров счетчика является *модуль счета* (емкость) *M*, представляющий собой число устойчивых различимых состояний счетчика. Требование различимости необходимо, так как если состояние счетчика переходит само в себя, то никакой информации о количестве воздействий получить невозможно. Аналогичная картина будет наблюдаться, если состояния неустойчивы, то есть меняются самопроизвольно без внешний воздействий. Когда число поступивших входных воздействий превысит модуль счета, то произойдет переполнение счетчика, он вернется в исходное состояние и начнется новый процесс смены его состояний. Входным воздействием обычно является импульс поступающий на специальный, так называемый счетный вход.

По значению модуля счетчики делятся на двоичные, двоично-десятичные (десятичные), счетчики с произвольным и управляемым модулем счета. В двоичных счетчиках модуль кратен степени двойки, то есть принимает значения 2,4,8,16 и т.п., в десятичных он кратен степени десяти (10,100,1000….). В счетчиках с произвольным модулем он может иметь любое фиксированное значение, а в счетчиках с управляемым модулем, имеется возможность менять модуль счета под воздействием внешних сигналов.

Счетчики могут быть суммирующими, вычитающими и реверсивными. В *суммирующих*, число соответствующее формируемому счетчиком коду, с приходом очередного счетного импульса увеличивается на единицу, в *вычитающих* – уменьшается. *Реверсивные* счетчики в зависимости от установленного режима работы могут функционировать и как суммирующие и как вычитающие.

Часто в структуру счетчика вводятся дополнительные узлы, позволяющие установить все его разряды в нулевое состояние при поступлении соответствующего сигнала на специальный вход R. На

принципиальных схемах четырехразрядные двоичные и двоично-десятичные счетчики, как функциональные элементы отображаются как показано на рис. 7.8.



а) б)

Рис. 7.8. Условное графическое обозначение а) двоичного и б) двоично-десятичного счетчиков

Важным параметром счетчика является быстродействие, обычно оцениваемое по максимальной частоте следования входных сигналов, при которой работа системы как счетчика не нарушается.

На выходах суммирующего двоичного счетчика с модулем $M = 2^m$, формируются m разрядные кодовые комбинации, порядок смены которых соответствует изменению двоичного кода, описывающего состояния счетчика от 0 до *M-1*.

Работа счетчика может быть описана с помощью временных диаграмм, отражающих состояния его разрядов после поступления очередного импульса на счетный вход, посредством *графа переходов* и таблицы состояний.

Граф переходов представляет собой символьную запись процедуры смены состояний счетчика и может описывать порядок их смены как в виде комбинаций значений разрядов, так и в виде десятичных чисел. В этом случае состояние всех разрядов счетчика в текущий момент времени интерпретируется как двоичная кодовая комбинация, которая переводится в десятичную систему счисления. При этом *i*-тому разряду приписывается соответствующий вес, равный 2^i, где *i*– номер разряда. Нумерация разрядов начинается с нуля.

Таблица переходов отображает процедуру смены состояний разрядов счетчика. В левой части строки записывается состояние счетчика до поступления входного воздействия, а в правой после него.

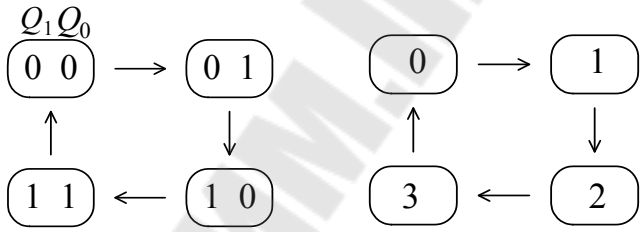
Для построения двоичного счетчика с модулем $M = 2^n$ потребуется n триггеров. Количество возможных комбинаций их

состояний будет равно $M = 2^n$, что совпадает с модулем счета. То есть в этом случае число возможных комбинаций будет равно количеству рабочих.

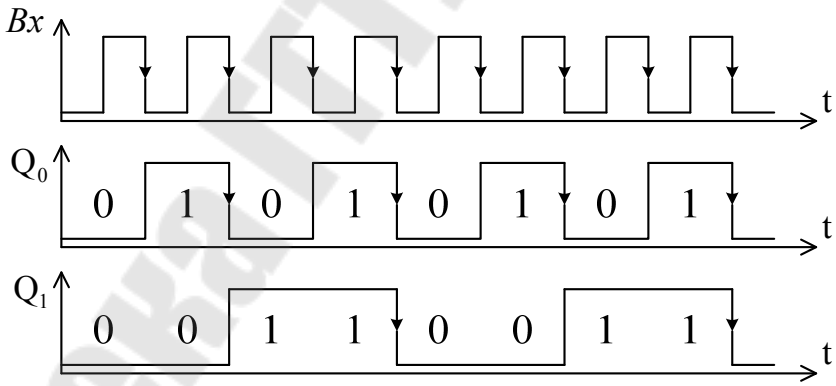
Для построения счетчика с модулем 4 потребуется система из двух триггеров у которых может быть четыре состояния. Для суммирующего счетчика десятичное число, описывающее его состояние и равное количеству поступивших импульсов должно увеличиваться с приходом каждого из них на единицу. Младший разряд счетчика Q_0 должен иметь вес 2^0, а старщий Q_1 - 2^1. Варианты описания его работы приведены на рис. 7.9.

Q_1	Q_0	Q_1^+	Q_0^+
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0

а)



б)



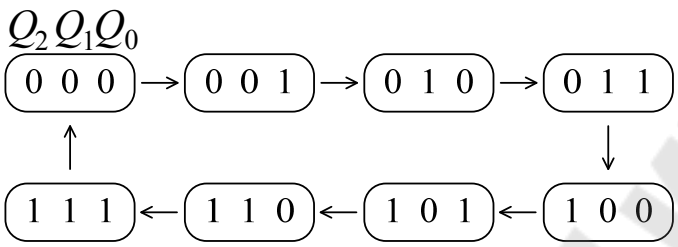
в)

Рис.7.9. Варианты описания работы счетчика с модулем 4: а) с помощью таблицы состояний; б) с помощью графов переходов; в) с помощью временных диаграмм

В счетчиках можно использовать как асинхронные, так и синхронизируемые фронтом *JK*-триггеры. Синтез синхронных счетчиков на синхронных триггерах удобно проводить, определив граф переходов, описывающий его функционирование. Для

суммирующего счетчика с модулем 8 он имеет вид, представленный на рис 7.10 а).

На основании этого графа составляется таблица переходов, в левой части которой указываются состояния разрядов счетчика до, а в правой – после переключения (рис. 7.10 б).



а)

Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

б)

Рис.7.10 - а)граф для суммирующего счетчика с модулем 8;
б) таблица переходов для суммирующего счетчика с модулем 8

Состояния разрядов Q_i^+ можно считать некоторыми функциями предшествующих, то есть $Q_i^+ = F_i(Q_0, Q_1 ... Q_{n-1})$. Конкретный вид соответствующей функции зависит от задаваемого алгоритма работы счетчика

Используя для минимизации карты Карно, функции можно привести к виду, представленному на рис. 7.11.

Далее, задавшись типом конкретного триггера, требуется в соответствии с его функцией возбуждения, подобрать такие наборы управляющих сигналов, чтобы переключение триггера происходило по законам, определяемым полученными функциями. В этом случае, триггер, функционируя в соответствии со своим алгоритмом работы, будет выполнять функции разряда счетчика.

Q_0^+	Q_0Q_1	$\overline{Q_0}Q_1$	$\overline{Q_0}\,\overline{Q_1}$	$Q_0\overline{Q_1}$
Q_2	0	1	1	0
$\overline{Q_2}$	0	1	1	0

Q_0^+	Q_0Q_1	$\overline{Q_0}Q_1$	$\overline{Q_0}\,\overline{Q_1}$	$Q_0\overline{Q_1}$
Q_2	0	1	0	1
$\overline{Q_2}$	0	1	0	1

Q_0^+	Q_0Q_1	$\overline{Q_0}Q_1$	$\overline{Q_0}\,\overline{Q_1}$	$Q_0\overline{Q_1}$
Q_2	0	1	1	1
$\overline{Q_2}$	1	0	0	1

$$Q_0^+ = \overline{Q_0}$$
$$Q_1^+ = Q_1\overline{Q_0} + \overline{Q_1}Q_0$$
$$Q_2^+ = Q_2\overline{Q_0} + Q_2\overline{Q_1} + \overline{Q_2}$$

Рис.7.11. Карты Карно и упрощенные выражения для суммирующего счетчика с модулем счета 8

Функция возбуждения JK-триггера выглядит следующим образом $Q^+_{JK} = Q\overline{K} + \overline{Q}J$. Для нулевого разряда счетчика она будет совпадать с найденной, если $K_0 = J_0 = 1$. Аналогичный триггер будет функционировать как первый разряд при $K_1 = J_1 = Q_0$. Для подбора управляющих сигналов триггера, выполняющего функцию второго разряда, полученную функцию необходимо преобразовать к виду:

$$Q_2^+ = Q_2\overline{Q_0} + Q_2\overline{Q_1} + \overline{Q_2}Q_1Q_0 = Q_2(\overline{Q_0} + \overline{Q_1}) + \overline{Q_2}Q_1Q_0 = Q_2\overline{Q_1Q_0} + \overline{Q_2}Q_1Q_0.$$

Отсюда следует, что $K_2 = J_2 = Q_1Q_0$.

Тогда:

$$Q_0^+ = Q_0\overline{K}_0 + \overline{Q}_0J_0 \qquad Q_0^+ = \overline{Q}_0 = Q_0 \cdot 0 + \overline{Q}_0 \cdot 1$$
$$Q_1^+ = Q_1\overline{K}_1 + \overline{Q}_1J_1 \qquad Q_1^+ = Q_1\overline{Q}_0 + \overline{Q}_1Q_0$$
$$Q_2^+ = Q_2\overline{K}_2 + \overline{Q}_2J_2 \qquad Q_2^+ = Q_2\overline{Q_0Q_1} + \overline{Q}_2Q_0Q_1$$

Если данные наборы сигналов подать на соответствующие входы триггеров, то они будут выполнять функции разрядов суммирующего синхронного счетчика с модулем 8, принципиальная схема которого приведена на рис. 7.12. Задержка формирования выходного кода в такой структуре будет равна задержке срабатывания триггера τ.

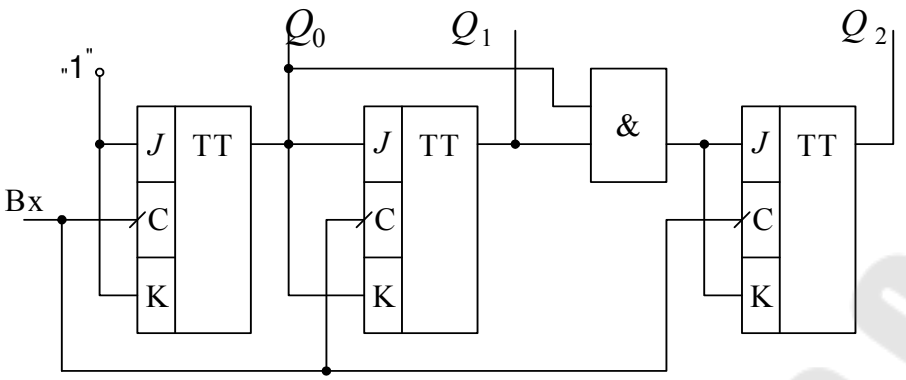


Рис.7.12 Принципиальная схема суммирующего синхронного счетчика с модулем 8,

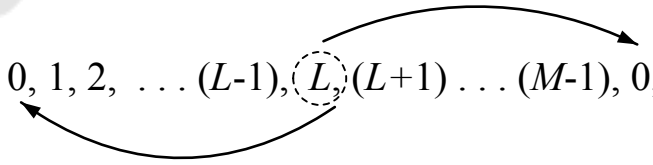
Узлы для формирования управляющих сигналов при использовании *JK*-триггеров получаются проще, чем для триггеров других типов, что обуславливает их широкое применение для построения счетчиковых структур.

Пусть имеется двоичный суммирующий счетчик с модулем M. Последовательность смены его состояний представлена на рис. 7.13. Из состояния, соответствующего коду числа $(M-1)$, счетчик будет переходить в исходное, нулевое.

0, 1, 2, . . . (*L*-1), *L*, (*L*+1) . . . (*M*-1), 0, 1, . . .

Рис.7.13 Последовательность смены состояний двоичного суммирующего счетчика с модулем счета M

В счетчиках с недвоичным модулем количество рабочих состояний L отличается от 2^n. Для построения таких устройств можно использовать двоичные счетчики с $M > L$, у которых часть состояний, а именно $(M-L)$ исключается из числа рабочих



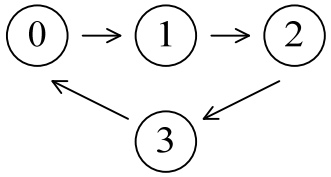
0, 1, 2, . . . (*L*-1), *L*, (*L*+1) . . . (*M*-1), 0, 1, . . .

Рис.7.14 Последовательность смены состояний недвоичного суммирующего счетчика с модулем счета L

Для реализации счетчика с модулем L необходимо из рабочего цикла исходного двоичного исключить $(M-L)$состояний (рис. 7.14). Эти состояния могут исключаться произвольным образом, то есть $(M-L)$ первых, последних, либо выбранных в соответствии с заданными требованиями. Такие состояния будут нерабочими.

Количество счетчиков с модулем L, которые можно реализовать на базе двоичного с модулем M определяется соотношением $N=\frac{M!}{(M-L)!}$, где восклицательный знак обозначает факториал, то есть результат произведения чисел от 1 до аргумента данной функции.

При использовании счетчика с модулем M=4 в качестве исходного, число вариантов счетчиков с модулем 3 будет равно $N=\frac{4!}{(4-3)!}=\frac{1\cdot 2\cdot 3\cdot 4}{1}=24$.



а)

0, 1, 2	1, 2, 3	2, 3 0	3 0, 1
2, 1, 0	3 2, 1	0, 3 2	1, 0, 3

б)

0, 1, 2	1, 2, 3	2, 3, 0	3, 0, 1
1, 2, 0	2, 3, 1	3, 0, 2	0, 1, 3
2, 0, 1	3, 1, 2	0, 2, 3	1, 3 0
0, 1, 2	3, 2, 1	0, 3, 2	1, 0, 3
1, 2, 0	2, 1, 3	3, 2, 0	0, 3, 1
2, 0, 1	1, 3, 2	2, 0, 3	3, 1, 0

в)

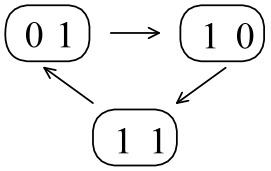
Рис.7.15 а)граф переходов двухразрядного двоичного счетчика; б) несовпадающие графы переходов счетчиков с модулем 3; в)условные графы переходов, описывающих порядок смены состояний 24 вариантов счетчиков с модулем 3

Если граф переходов двухразрядного двоичного счетчика представить, как показано на рис. 7.15, а), то условные графы переходов, описывающих порядок смены состояний 24 вариантов счетчиков с модулем 3, будут иметь вид, представленный на том же рисунке 7.15, в).

Записи, объединенные в группы описывают работу одного и того же счетчика с различными начальными состояниями. Количество счетчиков с несовпадающими графами переходов будет равно 8, общая формула в этом случае выглядит следующим образом $N = \frac{M!}{(M-L)!\cdot L}$, а графы их переходов приведены на рис. (7.15, б).

Для синтеза таких счетчиков можно использовать методику, описанную при построении двоичных счетчиков на синхронных триггерах.

Рассмотрим вариант построения счетчика с модулем 3, граф переходов которого и таблица переключений изображены на рисунке 7.16.



Q_1	Q_0	Q_1^+	Q_0^+
0	1	0	1
1	0	1	1
1	1	0	1

Рис. 7.16 Граф переходов и таблица переключений счетчика с модулем 3

Функции, описывающие состояния его разрядов могут быть представлены как: $Q_0^+ = Q_0\overline{Q_1} + \overline{Q_0}Q_1$, $Q_1^+ = Q_1\overline{Q_0} + \overline{Q_1}Q_0$, откуда $K_0 = \overline{Q_1}, J_0 = Q_1$ и $K_1 = Q_0, J_1 = Q_0$. Схема такого счетчика приведена на рис. 7.17.

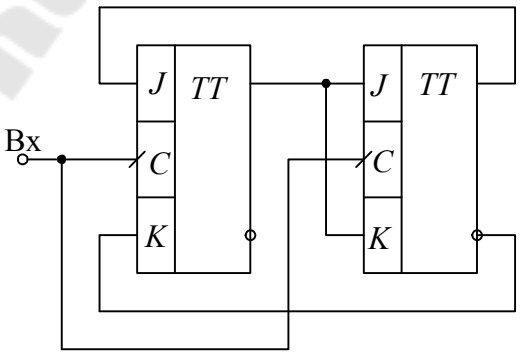


Рис.7.17 Схема счетчика с модулем счета 3, построенного по таблице переключений и графу на рис.7.16

Для рассматриваемого счетчика $Q_0^+ = Q_1^+ = 0$, то есть счетчик в этом состоянии зациклится, и его полный граф переходов будет иметь вид, представленный на рис. 7.18.

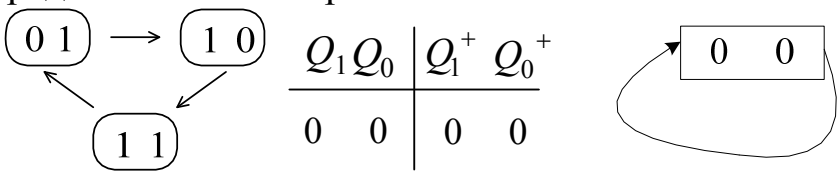


Рис.7.18 Полный граф переходов счетчика с модулем счета 3

Нерабочим состоянием здесь является комбинация 0-0. Несмотря на то, что этой комбинации на выходах счетчика не должно быть, однако, после включения питания или под воздействием помех, разряды счетчика могут перейти в это состояние. Такая ситуация называется *сбоем*. Некоторые варианты счетчиков через несколько тактов могут самостоятельно выйти из сбоя, то есть восстановить рабочий цикл. Однако возможна ситуация, когда сбой будет сменяться сбоем и работоспособность счетчика не восстановится. Для проверки необходимо провести анализ развития событий после возникновения сбоя. С этой целью значения разрядов счетчика соответствующие сбою подставляются в формулы, описывающие их состояния после переключения. Q_0^+

В некоторых случаях, использование стандартных процедур минимизации функций, описывающих состояния разрядов счетчика после переключения, приводит к выражениям, которые не позволяют подобрать требуемые комбинации управляющих сигналов на входах *JK* триггеров. Это может потребовать отказа от полной минимизации с целью получения необходимой структуры функций.

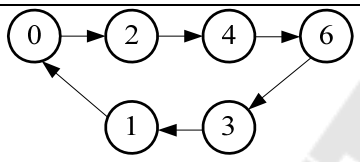
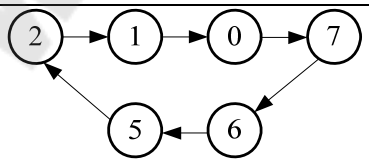
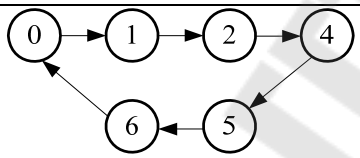
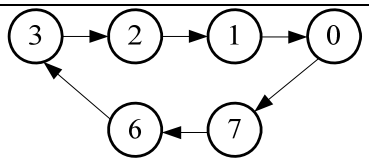
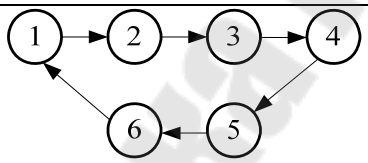
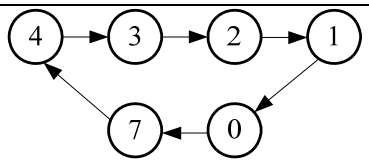
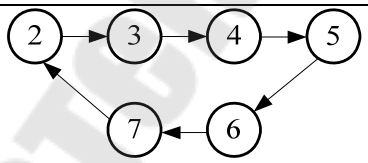
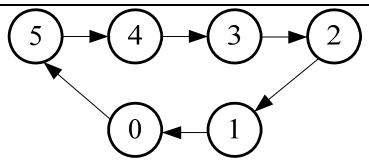
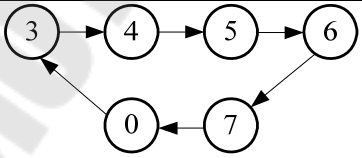
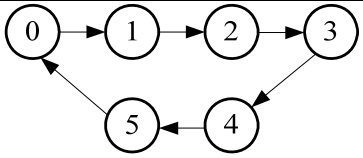
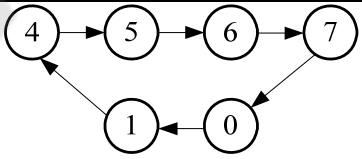
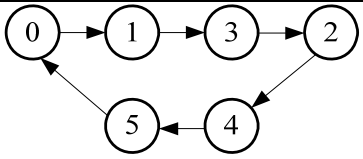
ЗАДАНИЕ НА ПРАКТИЧЕСКУЮ РАБОТУ № 7

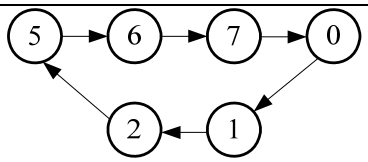
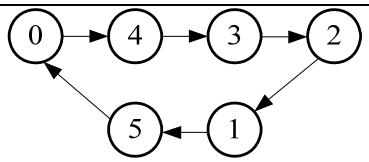
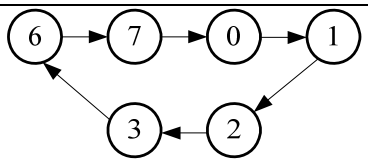
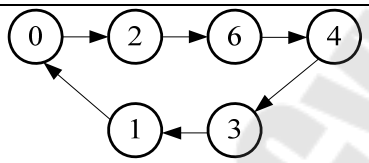
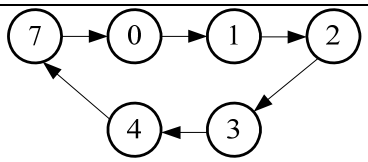
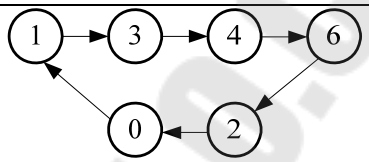
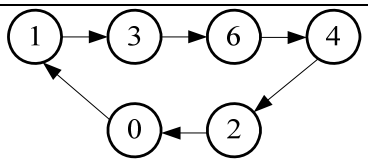
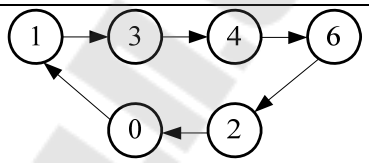
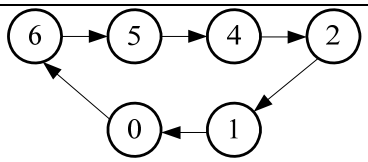
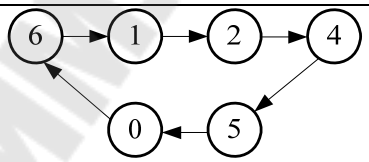
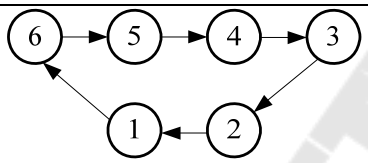
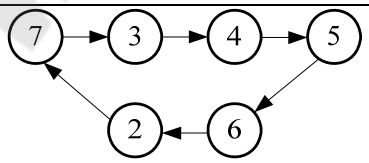
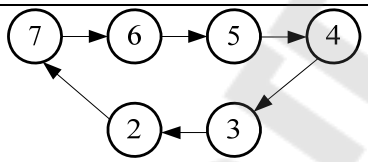
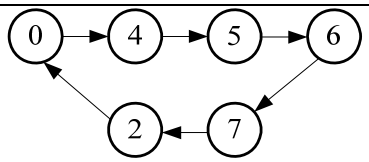
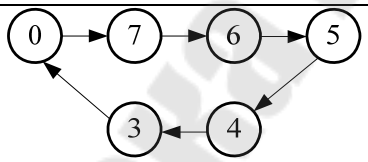
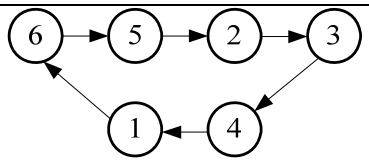
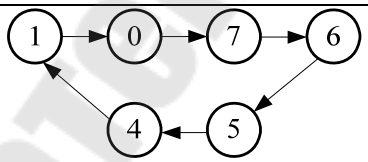
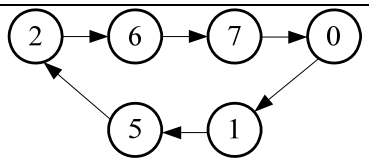
Номер варианта № в таблице 7.1 выбирается равным номеру студента по журналу группы.

В соответствии с вариантом задания требуется разработать принципиальную схему счетчика с заданным графом переходов и проанализировать ситуации при возникновении сбоев в его работе.

Для этого по заданному графу составляется таблица переходов, определяются функции описывающие состояния разрядов счетчика после переключения, и подбираются совокупности управляющих сигналов на входах триггеров для его реализации. Далее осуществляется анализ работы счетчика при возникновении сбоев. В конце приводится принципиальная схема разработанного устройства и временные диаграммы работы счетчика.

Таблица 7.1

N		*N*	
1		16	
2		17	
3		18	
4		19	
5		20	
6		21	

7		22	
8		23	
9		24	
10		25	
11		26	
12		27	
13		28	
14		29	
15		30	

Рассмотрим пример синтеза трехразрядного счетчика с модулем 6 на синхронных *JK*-триггерах с графом переходов, приведенным на рисунке 7.19.

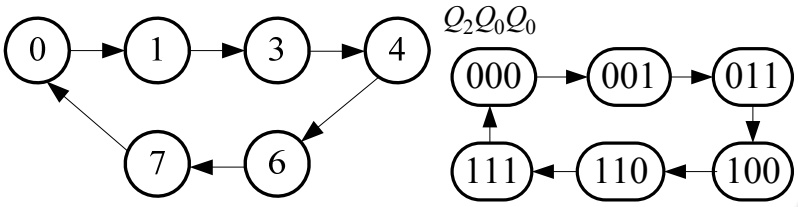


Рис. 7.19.Граф переходов трехразрядного счетчика с модулем 6

Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+
0	0	0	0	0	1
0	0	1	0	1	1
0	1	1	1	0	0
1	0	0	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

В соответствии с графом получаем следующую таблицу переходов

Карты Карно для функций, описывающих состояния разрядов счетчика после переключения, имеют вид, изображенный на рисунке 7.20.

Q_0^+	Q_1Q_0	$Q_1\overline{Q_0}$	$\overline{Q_1}\overline{Q_0}$	$\overline{Q_1}Q_0$
Q_2	0	1	0	*
$\overline{Q_2}$	0	*	1	1

Q_2^+	Q_1Q_0	$Q_1\overline{Q_0}$	$\overline{Q_1}\overline{Q_0}$	$\overline{Q_1}Q_0$
Q_2	0	1	1	*
$\overline{Q_2}$	1	*	0	0

Q_1^+	Q_1Q_0	$Q_1\overline{Q_0}$	$\overline{Q_1}\overline{Q_0}$	$\overline{Q_1}Q_0$
Q_2	0	1	1	*
$\overline{Q_2}$	0	*	0	1

Рис.7.20. Карты Карно для функций, описывающих состояния разрядов счетчика после переключения

Контура склейки выбираются таким образом, чтобы в выражения для соответствующих функций Q_i^+ входили прямые и инверсные значения функций Q_i. Это необходимо для подбора управляющих сигналов на J, K входах соответствующих триггеров, функции возбуждения которых имеют вид $Q_i^+ = Q_i\overline{K_i} + \overline{Q_i}J_i$.

Функция Q_0^+, описывающая состояние младшего разряда счетчика имеет вид $Q_0^+ = Q_0\overline{Q_1} + \overline{Q_0}Q_1 + \overline{Q_0}\overline{Q_2}$. Для подбора управляющих

сигналов она должна быть преобразована к двухкомпонентной форме $Q_0^+ = Q_0\overline{K_0} + \overline{Q_0}J_0$.

После преобразования получаем:

$$Q_0^+ = Q_0\overline{Q_1} + \overline{Q_0}Q_1 + \overline{Q_0}\,\overline{Q_2} = Q_0\overline{Q_1} + \overline{Q_0}\left(Q_1 + \overline{Q_2}\right) = Q_0\overline{Q_1} + \overline{Q_0}\,\overline{\overline{Q_1}Q_2},$$

Отсюда следует, что на входы управления триггера младшего разряда счетчика должны подаваться сигналы: $K_0 = Q_1 \qquad J_0 = \overline{\overline{Q_1}Q_2}$.

Функция Q_1^+, описывающая состояние следующего разряда счетчика имеет вид $Q_1^+ = Q_1\overline{Q_0} + \overline{Q_1}Q_2 + \overline{Q_1}Q_0$. Для подбора управляющих сигналов она должна быть преобразована к двухкомпонентной форме $Q_1^+ = Q_1\overline{K_1} + \overline{Q_1}I_1$.После преобразования получаем:

$$Q_1^+ = Q_1\overline{Q_0} + \overline{Q_1}Q_2 + \overline{Q_1}Q_0 = Q_1\overline{Q_0} + \overline{Q_1}\left(Q_2 + Q_0\right) = Q_1\overline{Q_0} + \overline{Q_1}\,\overline{\overline{Q_2}\,\overline{Q_0}}.$$

Отсюда следует, что на входы управления триггера разряда Q_1 счетчика должны подаваться сигналы: $K_1 = Q_0 \qquad J_1 = \overline{\overline{Q_2}\,\overline{Q_0}}$.

Функция Q_2^+, описывающая состояние старшего разряда счетчика имеет вид $Q_2^+ = Q_2\overline{Q_0} + \overline{Q_2}Q_1$. Она уже представлена в форме $Q_2^+ = Q_2\overline{K_2} + \overline{Q_2}J_2$. На входы управления триггера старшего разряда счетчика должны подаваться сигналы: $K_2 = Q_0 \qquad J_2 = Q_1$.

Состояниями сбоя для проектируемого счетчика являются комбинации сигналов 0 1 0 и 1 0 1 на его выходах $Q_2\,Q_1\,Q_0$. Для анализа ситуации после сбоя подставим значения данных сигналов в формулы, описывающие состояния разрядов счетчика после переключения.

Первый сбой: $Q_2 = 0,\ Q_1 = 1,\ Q_0 = 0$.

$$Q_0^+ = Q_0\overline{Q_1} + \overline{Q_0}Q_1 + \overline{Q_0}\,\overline{Q_2} = 0\cdot 0 + 1\cdot 1 + 1\cdot 1 = 1$$

$$Q_1^+ = Q_1\overline{Q_0} + \overline{Q_1}Q_2 + \overline{Q_1}Q_0 = 1\cdot 1 + 0\cdot 0 + 0\cdot 0 = 1$$

$$Q_2^+ = Q_2\overline{Q_0} + \overline{Q_2}Q_1 = 0\cdot 1 + 1\cdot 1 = 1$$

Второй сбой $Q_2 = 1,\ Q_1 = 0, Q_0 = 1$.

$$Q_0^+ = Q_0\overline{Q_1} + \overline{Q_0}Q_1 + \overline{Q_0}\,\overline{Q_2} = 1\cdot 1 + 0\cdot 0 + 0\cdot 0 = 1$$

$$Q_1^+ = Q_1\overline{Q_0} + \overline{Q_1}Q_2 + \overline{Q_1}Q_0 = 0\cdot 1 + 1\cdot 1 + 1\cdot 1 = 1$$

$$Q_2^+ = Q_2\overline{Q_0} + \overline{Q_2}Q_1 = 1\cdot 0 + 0\cdot 0 = 0$$

Таблица переходов в случае сбоев имеет вид:

Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+
0	1	0	1	1	1
1	0	1	0	1	1

Из таблицы следует, что в случае возникновения сбоя, счетчик в следующем такте перейдет в рабочее состояние, то есть он обладает свойствами самовосстановления.

Принципиальная схема разработанного варианта счетчика с модулем счета 6 изображена на рисунке 7.21.

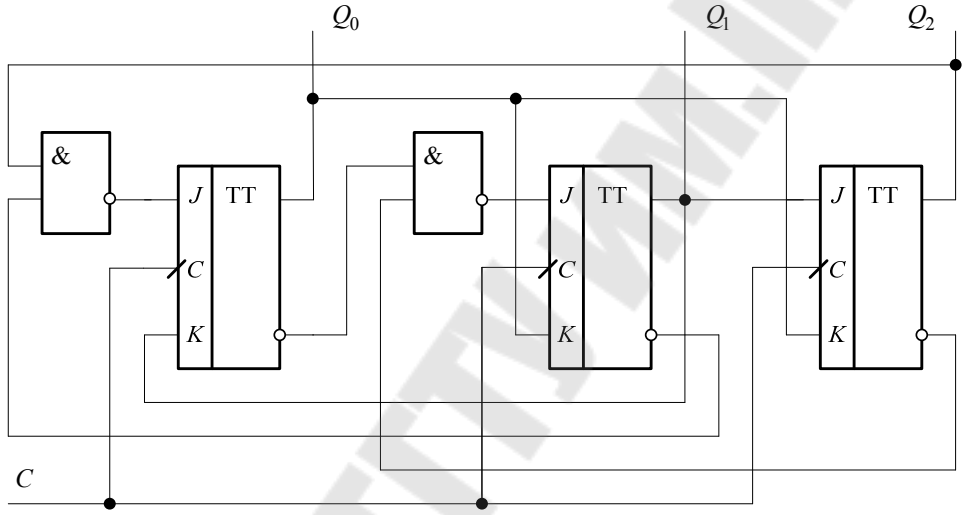
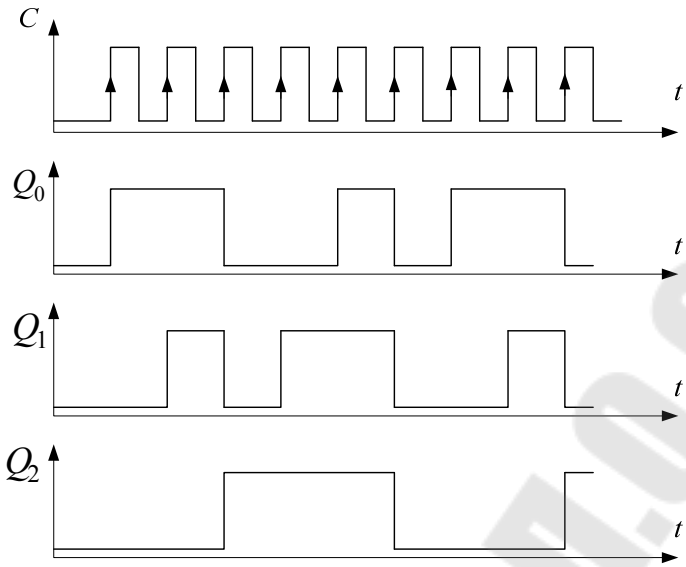


Рис.7.21 Принципиальная схема разработанного счетчика с модулем счета 6

Временные диаграммы функционирования счетчика 6 изображены на рисунке 7.22.



Q_0	0	1	1	0	0	1	0	1	1
Q_1	0	0	1	0	1	1	0	0	1
Q_2	0	0	0	1	1	1	0	0	0

Рис.7.22 Временные диаграммы функционирования разработанного счетчика с модулем счета 6

Литература.

1. Безуглов Д. А. Цифровые устройства и микропроцессоры: учебное пособие для вузов / Д. А. Безуглов, И. В. Калиенко. – Ростов н/Д: Феникс, 2006. – 400 с

2. Браммер Ю. А. Импульсные и цифровые устройства: учебное пособие для вузов / Ю. А. Браммер, И. Н. Пащук. – М.: Высш. шк., 1999. – 351 с.

3. Белоус А.И., Емельянов В.А., Турцевич А.С. Основы схемотехники микроэлектронных устройств – Москва: Техносфера, 2012. – 472 с.

4. Быстров, Ю. А. Электронные цепи и микросхемотехника : учебник для вузов / Ю. А. Быстров, И. Г. Мироненко. – М. : Высш. шк, 2002.

5. Новиков Ю. В. Введение в цифровую схемотехнику : учеб. пособие. - Москва : ИНТУИТ : БИНОМ, 2007. – 343 с

6. Пухальский Г.И., Новосельцева Т.Я. Проектирование дискретных устройств на интегральных микросхемах: Справочник. – М.: Радио и связь, 1990. – 304 с.

7. Храбров, Е. А. Цифровая электроника : учебное пособие для вузов / Е. А. Храбров, Ю. Е. Котова. – Гомель : ГГТУ им. П. О. Сухого, 2013. – 271 с.

СОДЕРЖАНИЕ

Котова Юлия Евгеньевна
Захаренко Леонид Александрович

СХЕМОТЕХНИКА В СИСТЕМАХ УПРАВЛЕНИЯ

Практикум
по одноименной дисциплине для студентов специальности 1-53 01 07 «Информационные технологии и управление в технических системах» дневной формы обучения

Подписано к размещению в электронную библиотеку
ГГТУ им. П. О. Сухого в качестве электронного
учебно-методического документа 09.03.20.

Рег. № 47Е.
http://www.gstu.by