Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра электронных вычислительных машин

И.В. Лукьянова, Ю.А. Луцик

АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

МЕТОДИЧЕСКОЕ ПОСОБИЕ
к курсовому проекту
для студентов специальности
”Вычислительные машины, системы и сети”
всех форм обучения

Минск 2004

УДК 004.7 (075.8)
ББК 32.97 я 73
Л 84

Лукьянова И.В.

Л 84 Арифметические и логические основы вычислительной техники: Метод. пособие к курсовому проекту для студ. спец. «Вычислительные машины, системы и сети» всех форм обуч./ И.В. Лукьянова, Ю.А. Луцик. − Мн.: БГУИР, 2004. − 35 с.: ил.
ISBN 985-444-642-5

В методическом пособии приведены исходные данные для выполнения курсового проекта и рассмотрен пример выполнения и оформления курсового проекта для одного из вариантов задания. Оно может быть использовано студентами специальности 40 02 01 ”Вычислительные машины, системы и сети”, магистрантами и аспирантами.

УДК 004.7 (075.8)
ББК 32.97 я 73

ISBN 985-444-642-5

СОДЕРЖАНИЕ

1. ЗАДАНИЕ К КУРСОВОМУ ПРОЕКТУ

Курсовой проект предполагает синтез цифровых схем арифметических устройств, выполняющих операции сложения и умножения над числами, представленными в форме с плавающей запятой в двоичной и двоично-четверичной системах счисления (с/с).

По исходным данным необходимо разработать:

1. Алгоритм выполнения операции умножения, для чего потребуется:
   - перевести заданные исходные числа в четверичную систему счисления;
   - представить числа в форме с плавающей запятой, при этом число четверичных разрядов для мантиссы равно шести, для порядка – два, плюс два разряда для знаков мантиссы и порядка;
   - произвести перемножение чисел согласно заданному алгоритму;
   - оценить погрешность вычисления после перевода результата в исходную систему счисления.

2. Алгоритм выполнения операции сложения.

3. Структурную схему вычислительного устройства, выполняющего сложение и умножение, содержащую узлы для действия над мантиссами и порядками, а также при этом необходимо определить время умножения с учетом временных задержек в комбинационных схемах.

4. Функциональные схемы основных узлов проектируемого сумматора-уножителя в заданном логическом базисе. Для этого следует провести:
   - логический синтез комбинационного одноразрядного четверичного сумматора (ОЧС) на основе составленной таблицы истинности для суммы слагаемых с учетом переноса из младшего разряда, используя при этом карты Карно-Вейча или алгоритм извлечения (Рота), оценив эффективность минимизации;
   - логический синтез одноразрядного комбинационного четверичного умножителя (ОЧУ) в случае разработки структурной схемы 1-го типа путем минимизации переключательных функций по каждому выходу схемы. Минимизация выполняется с применением алгоритма Рота или карт Карно-Вейча с последующей оценкой эффективности минимизации;
   - логический синтез одноразрядного комбинационного четверичного умножителя-сумматора (ОЧУС) в случае разработки структурной схемы 2-го типа путем минимизации переключательных функций по каждому выходу схемы. Минимизация выполняется с применением алгоритма Рота или карт Карно-Вейча с последующей оценкой эффективности минимизации;
   - логический синтез комбинационной схемы преобразователя множителя (ПМ);

- построить функциональную схему ОЧС в заданном логическом базисе и на мультиплексорах;
- построить функциональную схему ПМ и ОЧУ (ОЧУС) в заданном логическом базисе.

По результатам разработки определить время умножения на один разряд и на n разрядов множителя.

Исходные данные к курсовому проекту

Исходные данные для выполнения курсового проекта (приведены в прил.):

1. Исходные операнды - десятичные числа с целой и дробной частью, над которыми производится операция умножения (заданы в строке 1 табл. П.1);

2. Алгоритм выполнения операции умножения: А, Б, В, Г (определяется строкой 2 табл. П.1):

А – умножение начинается с младших разрядов множителя со сдвигом частичных сумм вправо,

Б – умножение начинается с младших разрядов множителя со сдвигом частичных произведений (множимого) влево,

В – умножение начинается со старших разрядов множителя со сдвигом частичных сумм влево,

Г – умножение начинается со старших разрядов множителя со сдвигом частичных произведений вправо.

3. Метод ускоренного умножения, на базе которого строится умножитель:
- для алгоритмов **А** и **Б**: умножение закодированного двоично-четверичного множимого на 2 разряда двоичного множителя одновременно в прямых кодах;
- для алгоритмов **В** и **Г**: умножение закодированного двоично-четверичного множимого на 2 разряда двоичного множителя одновременно в дополнительных кодах.

4. Двоичные коды четверичных цифр **множимого** для работы в двоично-четверичной системе счисления (вариант кодирования учитывается при выполнении арифметических операций и задается строкой 3 табл.П.1). **Множитель** представляется обычным весомозначным кодом**:** $\mathbf{0_4 - 00,\ 1_4 - 01,\ 2_4 - 10,\ 3_4 - 11}$.

5. Тип синтезируемого устройства умножения, определяемый основными структурными узлами, на базе которых строится умножитель:
- умножитель 1-го типа строится на базе ОЧУ, ОЧС и регистра-аккумулятора;
- умножитель 2-го типа строится на базе ОЧУС, ОЧС и регистра результата (см. строку 6 табл. П.1).

6. Способ минимизации и логический базис для аппаратной реализации ОЧУ, ОЧУС и ОЧС (определяется строками 4, 5, 6 табл. П.1), при этом ОЧС реализуется в заданном логическом базисе и на мультиплексорах.

2. ПРИМЕР СИНТЕЗА СУММАТОРА-УМНОЖИТЕЛЯ (РАЗРАБОТКА АЛГОРИТМА УМНОЖЕНИЯ И СТРУКТУРНОЙ СХЕМЫ СУММАТОРА-УМНОЖИТЕЛЯ)

Исходные данные:

исходные сомножители: Мн = 15,55; Мт = - 45,35;

алгоритм умножения: А;

метод умножения: умножение закодированного двоично-четверичного множимого на 2 разряда двоичного множителя одновременно в прямых кодах;

коды четверичных цифр множимого для перехода к двоично-четверичной системе кодирования: $0_4 \rightarrow 00$, $1_4 \rightarrow 11$, $2_4 \rightarrow 10$, $3_4 \rightarrow 01$;

тип синтезируемого умножителя: структурные схемы приведены для обоих типов умножителей - на рис.1 приведена структура 1-го типа (ОЧУ, ОЧС, аккумулятор), на рис.2 приведена структура 2-го типа (ОЧУС, ОЧС, регистр результата).

Арифметические операции сложения двоично-четверичных чисел с разными знаками в дополнительных кодах и умножения на 2 разряда множителя в прямых кодах должны выполняться одним цифровым устройством, именуемым **сумматор-умножитель**. Учитывая то, что суммирующие узлы обязательно входят в состав умножителя, начнем синтез с разработки алгоритма умножения.

2.1. Разработка алгоритма умножения

1. Перевод сомножителей из десятичной системы счисления в четверичную:

множимое

15 | 4	0,55	$Мн_4$ =33,2030
12 3	4	в соответствии с заданной кодировкой
3	2,20	множимого
	4	$Мн_{2/4}$ = 0101,10000100
	0,80	
	4	
	3,20	
	4	
	0,80	

множитель

45|4	0,35	$Мт_4 = -231,112$
44 11|4	4	
1 8 2	1,40	$Мт_{2/4} = -101101,010110$
3	4	*множитель представляется обычным*
	1,60	*весомозначным кодом*: 0_4 - 00, 1_4 - 01,
	4	2_4 - 10, 3_4 - 11
	2,40	для всех вариантов
	4	
	1,60	

2. Запишем сомножитель в форме с плавающей запятой в прямом коде:
Мн = 0,010110000100 $Р_{Мн} = 0.0010 + 02_{10}$ - закодировано по заданию,
Мт = 1,101101010110 $Р_{Мт} = 0.0011 + 03_{10}$ - закодировано традиционно.

3. Умножение двух чисел с плавающей запятой на два разряда множителя одновременно в прямых кодах. Это сводится к сложению порядков, формированию знака произведения, преобразованию разрядов множителя согласно алгоритму, и перемножению мантисс сомножителей.

Порядок произведения будет равен:

$Р_{Мн} =$ 0.0010	02
$Р_{Мт} =$ 0.0011	03
$Р_{Мн \cdot Мт} =$ 0.1111	11.

Результат закодирован в соответствии с заданием на кодировку множимого.
Знак произведения определяется суммой по модулю ”два” знаков сомножителей, т.е.

$$зн\ Мн \oplus зн\ Мт = 0 \oplus 1 = 1.$$

Для умножения мантисс необходимо предварительно преобразовать множитель. При умножении чисел в прямых кодах диада $11(3_4)$ заменяется на триаду $1\overline{01}$. Преобразованный множитель имеет вид: $Мт^{п}_4 = 1\overline{1}1112$ или $Мт^{п}_2 = 01\,\overline{01}\,\overline{01}\,01010110$. Перемножение мантисс по алгоритму “А” приведено в табл. 1.

4. После окончания умножения необходимо оценить погрешность вычислений. Для этого полученное произведение ($Мн \cdot Мт_4 = -0,23000331002$, $Р_{Мн \cdot Мт} = 5$) приводится к нулевому порядку, а затем переводится в десятичную систему счисления:

$Мн \cdot Мт_4 = -23000,331002$ $Р_{Мн \cdot Мт} = 0$;
$Мн \cdot Мт_{10} = -704,9884$.

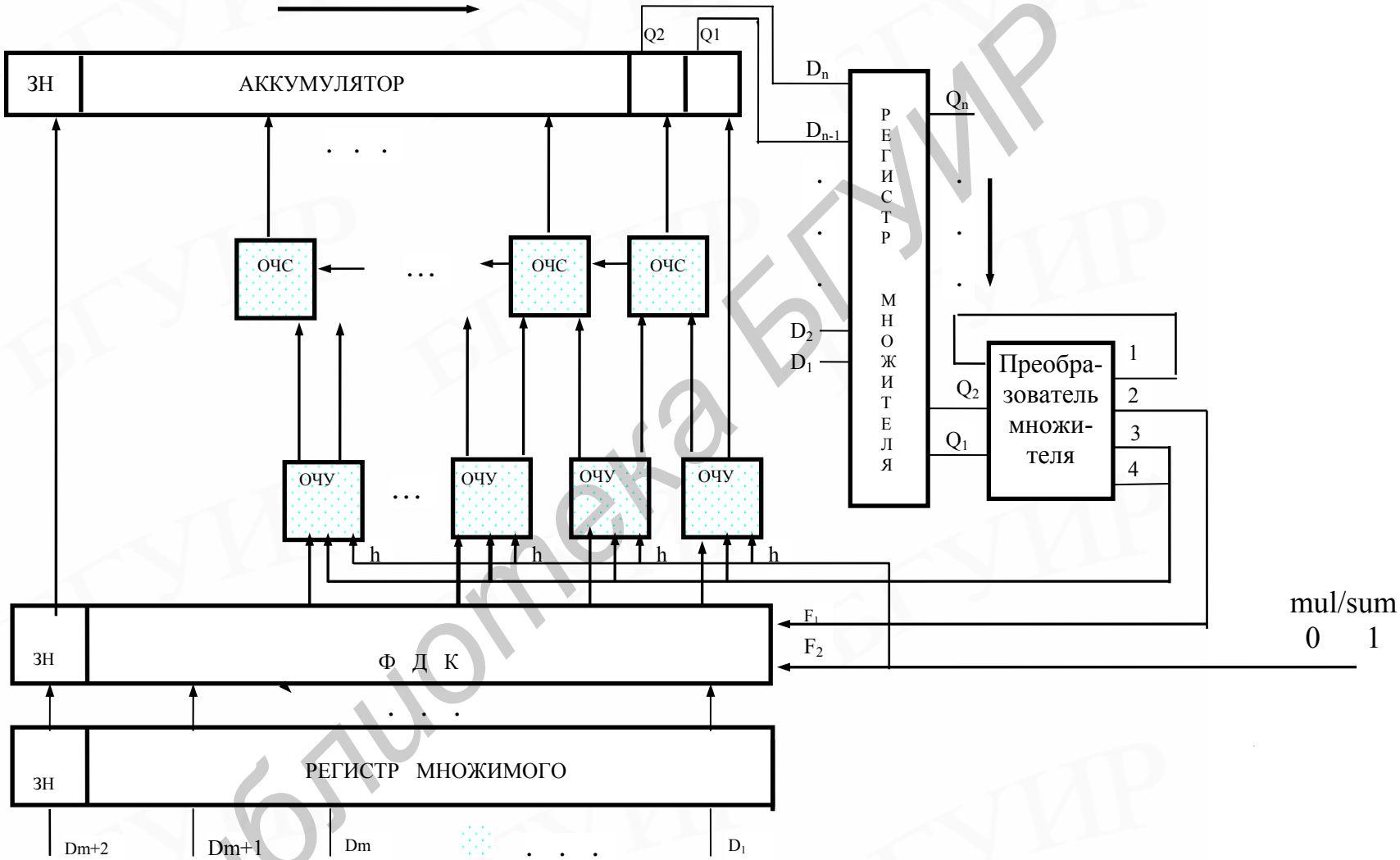


Рис. 1. Структурная схема сумматора-умножителя 1-го типа.

Алгоритм умножения «А», на 2 разряда множителя одновремен

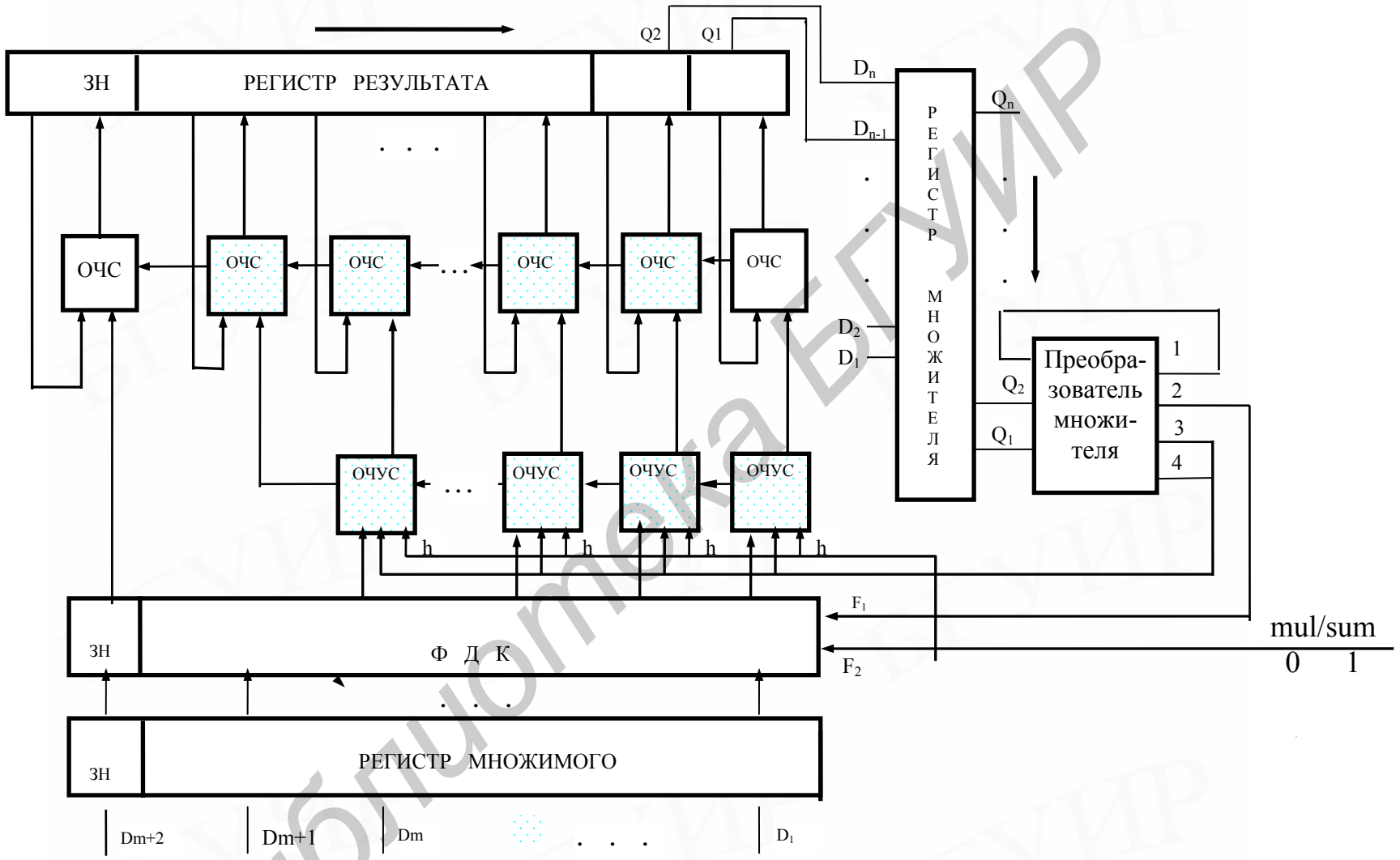


Рис. 2. Структурная схема сумматора-умножителя 2-го типа.
Алгоритм умножения «А», на 2 разряда множителя одновременно.

Таблица 1

Перемножение мантисс

Четверичная с/с			Двоично-четверичная с/с			Комментарии
0.	0000000		0.	00 00 00 00 00 00 00		$\Sigma_0^{ч} = 0$
0.	1330120		0.	11 01 01 00 11 10 00		$\Pi_1^{ч} = Мн \cdot 2$
0.	1330120		0.	11 01 01 00 11 10 00		$\Sigma_1^{ч}$
0.	0133012	0	0.	00 11 01 01 00 11 10	00	$\Sigma_1^{ч} \cdot 4^{-1}$
0.	0332030		0.	00 01 01 10 00 01 00		$\Pi_2^{ч} = Мн \cdot 1$
0.	1131102	0	0.	11 11 01 11 11 00 10	00	$\Sigma_2^{ч}$
0.	0113110	20	0.	00 11 11 01 11 11 00	10 00	$\Sigma_2^{ч} \cdot 4^{-1}$
0.	0332030		0.	00 01 01 10 00 01 00		$\Pi_3^{ч} = Мн \cdot 1$
0.	1111200	20	0.	11 11 11 11 10 00 00	10 00	$\Sigma_3^{ч}$
0.	0111120	020	0.	00 11 11 11 11 10 00	00 10 00	$\Sigma_3^{ч} \cdot 4^{-1}$
0.	0332030		0.	00 01 01 10 00 01 00		$\Pi_4^{ч} = Мн \cdot 1$
0.	1103210	020	0.	11 11 00 01 10 11 00	00 10 00	$\Sigma_4^{ч}$
0.	0110321	0020	0.	00 11 11 00 01 10 11	00 00 10 00	$\Sigma_4^{ч} \cdot 4^{-1}$
3.	3001310		1.	01 00 00 11 01 11 00		$\Pi_5^{ч} = Мн \cdot (-1)$
3.	3112231	0020	1.	01 11 11 10 10 01 11	00 00 10 00	$\Sigma_5^{ч}$
3.	3311223	10020	1.	01 01 11 11 10 10 01	11 00 00 10 00	$\Sigma_5^{ч} \cdot 4^{-1}$
3.	3001310		1.	01 00 00 11 01 11 00		$\Pi_6^{ч} = Мн \cdot (-1)$
3.	2313133	10020	1.	10 01 11 01 11 01 01	11 00 00 10 00	$\Sigma_6^{ч}$
3.	3231313	310020	1.	01 10 01 11 01 11 01	01 11 00 00 10 00	$\Sigma_6^{ч} \cdot 4^{-1}$
0.	0332030		0.	00 01 01 10 00 01 00		$\Pi_7^{ч} = Мн \cdot 1$
0.	0230003	310020	0.	00 10 01 00 00 00 01	01 11 00 00 10 00	$\Sigma_7^{ч}$

Результат прямого перемножения операндов дает следующее значение:

$$Мн_{10} \cdot Мт_{10} = 15{,}55 \cdot 45{,}35 = 705{,}1925.$$

Абсолютная погрешность:

$$\Delta = 705{,}1925 - 704{,}9884 = 0{,}2041.$$

Относительная погрешность:

$$\delta = \frac{\Delta}{Мн \cdot Мт} = \frac{0{,}2041}{704{,}9884} = 0{,}0002895 \qquad (\delta = 0{,}2895\%).$$

Эта погрешность получена за счет приближенного перевода из десятичной системы счисления в четверичную обоих сомножителей, а также за счет округления полученного результата произведения.

2.2. Разработка структурной схемы сумматора-умножителя

В курсовой работе предполагается разработка двух типов структур сумматора-умножителя. Структура 1-го типа строится на базе заданных узлов –

ОЧУ, ОЧС и аккумулятора (накапливающего сумматора), а структура 2-го типа строится на базе заданных узлов – ОЧУС и ОЧС.

Приведем пример синтеза структурных схем сумматора-умножителя 1-го (см. рис. 1) и 2-го типа (см. рис. 2) для алгоритма умножения “А”.

Управление обеими схемами осуществляется внешним сигналом mul/sum, который определяет вид текущей арифметической операции.

Синтез структуры сумматора-умножителя 1-го типа

Структурная схема сумматора-умножителя 1-го типа для алгоритма умножения “А” приведена на рис.1.

Если устройство работает как сумматор (на входе mul/sum – “1”), то оба слагаемых последовательно (за 2 такта) заносятся в регистр множимого, а на управляющий вход формирователя дополнительного кода (ФДК) F_2 поступает «1». Следует учесть, что числа представлены в форме с плавающей запятой. Поэтому, прежде чем складывать мантиссы, необходимо выровнять порядки. В блоке порядков необходимо обеспечить сравнение порядков, используя сумматор порядков, и в зависимости от знака результата сдвигать первое или второе слагаемое. Реализация сдвига мантиссы числа с меньшим порядком будет зависеть от используемого алгоритма умножения. Этим будет определяться порядок подачи слагаемых на операцию и то, где будет сдвигаться мантисса (в регистре множимого или в регистре результата). На выходах ФДК формируется дополнительный код одного из слагаемых с учетом знака. Это слагаемое может быть записано в регистр результата, при этом управляющие сигналы, поступающие на входы «h» всех ОЧУ, дают возможность переписать на выходы ОЧУ разряды слагаемого без изменений (см. рис.3). При необходимости выравнивания порядков в регистре-аккумуляторе может выполняться сдвиг мантиссы первого слагаемого. Если на вход «h» поступает «0», то ОЧУ перемножает разряды Мн и Мт.

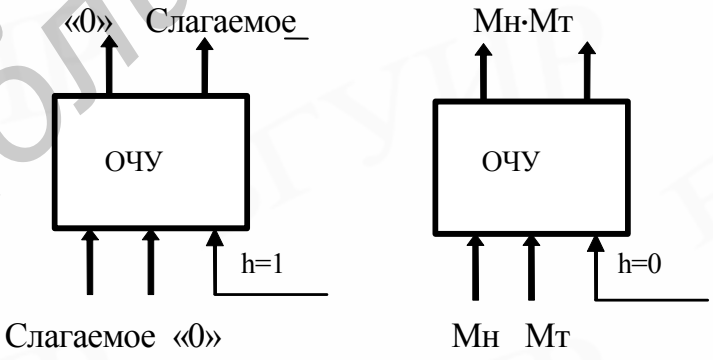


Рис. 3. Режимы работы ОЧУ

Одноразрядный четверичный сумматор предназначен для сложения двух двоично-четверичных цифр, подаваемых на его входы (рис. 4).

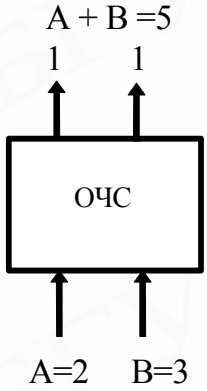


Рис. 4. Одноразрядный четверичный сумматор

В ОЧС первое слагаемое складывается с нулем, так как на старших выходах ОЧУ будут формироваться только коды нуля. Затем первое слагаемое попадает в регистр-аккумулятор, который изначально обнулен. На втором такте второе слагаемое из регистра множимого через цепочку ОЧУ и ОЧС попадает в аккумулятор, где складывается с первым слагаемым. Таким образом аккумулятор (накапливающий сумматор) складывает операнды и хранит результат. Разрядность аккумулятора должна быть на единицу больше, чем разрядность исходных слагаемых, чтобы предусмотреть возможность возникновения при суммировании переноса.

Если устройство работает как умножитель (на входе mul/sum – “0”), то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК F_2 поступает «0». Диада множителя поступает на входы преобразователя множителя (ПМ). Задачей ПМ является преобразование диады множителя в соответствии с алгоритмом преобразования. При этом в случае образования единицы переноса в старшую диаду множителя она должна быть учтена при преобразовании этой старшей диады (выход 1 ПМ). В регистре множителя в конце каждого такта умножения содержимое сдвигается на 2 двоичных разряда, и в последнем такте умножения регистр обнуляется. Это позволяет использовать регистр множителя для хранения младших разрядов произведения при умножении по алгоритму “А“ (регистр множителя служит как бы “продолжением” регистра результата). Выход 2 ПМ переходит в единичное состояние, если текущая диада содержит отрицание ($\overline{01}$). В этом случае инициализируется управляющий вход F_1 формирователя дополнительного кода (ФДК), и на выходах ФДК формируется дополнительный код множимого с обратным знаком (умножение на -1). Принцип работы ФДК в зависимости от управляющих сигналов приведен в табл.2.

На выходах 3,4 ПМ формируются диады преобразованного множителя, которые поступают на входы ОЧУ вместе с диадами множимого (см. рис.1). ОЧУ предназначен лишь для умножения двух четверичных цифр. Если в процессе умножения возникает перенос в следующий разряд, необходимо преду-

Таблица 2

Режимы работы формирователя дополнительного кода

Сигналы на входах ФДК		Результат на выходах ФДК
F_1	F_2	
0	0	Дополнительный код множимого
0	1	Дополнительный код слагаемого
1	0	Меняется знак Мн
1	1	Меняется знак слагаемого

смотреть возможность его прибавления. Для суммирования результата умножения текущей диады Мн · Мт с переносом из предыдущей диады предназначены ОЧС. Следовательно, чтобы полностью сформировать частичное произведение четверичных сомножителей, необходима комбинация цепочек ОЧУ и ОЧС. Частичные суммы формируются в аккумуляторе. На первом этапе он обнулен, и первая частичная сумма получается за счет сложения первого частичного произведения (сформированного на выходах ОЧС) и нулевой частичной суммы (хранящейся в аккумуляторе). Далее в аккумуляторе происходит сложение i-й частичной суммы с (i+1)-м частичным произведением, результат сложения сохраняется. Содержимое аккумулятора сдвигается на один четверичный разряд вправо в конце каждого такта умножения по алгоритму «А».

На четырех выходах ОЧУ формируется результат умножения диад Мн·Мт. Максимальной цифрой в диаде преобразованного множителя является двойка, поэтому в старшем разряде произведения максимальной цифрой может оказаться только «1» :

$$3_{\text{max}}^{\text{Мн}} \cdot 2_{\text{max}}^{\text{Мт}} = 12.$$

Это означает, что на младшие входы ОЧС никогда не поступят диады цифр, соответствующие кодам «2» и «3», следовательно, в таблице истинности работы ОЧС будут содержаться 16 безразличных входных наборов.

Частичные суммы хранятся в аккумуляторе и регистре множителя, так как алгоритм умножения «А» предполагает возможность синхронного сдвига этих устройств. Количество тактов умножения определяется разрядностью Мт.

Синтез структуры сумматора-умножителя 2-го типа

Структурная схема сумматора-умножителя 2-го типа для алгоритма умножения “А” приведена на рис.2.

Если устройство работает как сумматор, то оба слагаемых последовательно (за 2 такта) заносятся в регистр множимого, а на управляющий вход формирователя дополнительного кода F_2 поступает «1». Необходимо обеспечить выполнение алгоритма сложения чисел, представленных в форме с плавающей запятой, базируясь на схеме умножителя, реализующего заданный ал-

горитм умножения (см. описание структуры сумматора-умножителя 1-го типа). Первое слагаемое переписывается в регистр результата под действием управляющих сигналов, поступающих на входы «h» всех ОЧУС (рис. 5). Если на вход «h» поступает «0», то ОЧУС перемножает разряды Мн и Мт и добавляет к полученному результату перенос из предыдущего ОЧУС.

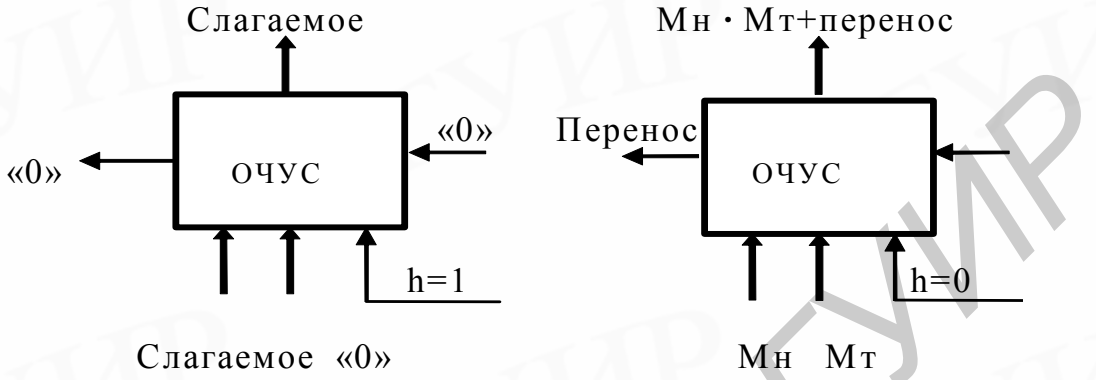


Рис. 5. Режимы работы ОЧУС

В ОЧС первое слагаемое складывается с нулем, записанным в регистре результата, и переписывается без изменений в регистр результата. На втором такте второе слагаемое из регистра множимого через цепочку ОЧУС попадает на входы ОЧС и складывается с первым слагаемым, хранящимся в регистре результата. Сумма хранится в регистре результата. Разрядность регистра результата должна быть на единицу больше, чем разрядность исходных слагаемых, чтобы предусмотреть возможность возникновения при суммировании переноса.

Если устройство работает как умножитель, то множимое и множитель помещаются в соответствующие регистры, а на управляющий вход ФДК F_2 поступает «0». Диада множителя поступает на входы преобразователя множителя. Единица переноса в следующую диаду, если она возникает, должна быть добавлена к следующей диаде множителя (выход 1 ПМ). В регистре множителя после каждого такта умножения содержимое сдвигается на 2 двоичных разряда, и в конце умножения регистр обнуляется. Это позволяет использовать регистр множителя для хранения младших разрядов произведения при умножении по алгоритму «А». Выход 2 ПМ переходит в единичное состояние, если текущая диада содержит отрицание ($\overline{01}$). В этом случае инициализируется управляющий вход F_1 формирователя дополнительного кода, и на выходах ФДК формируется дополнительный код множимого с обратным знаком (умножение на -1). Принцип работы ФДК в зависимости от управляющих сигналов отражен в табл.2.

На выходах 3,4 ПМ формируются диады преобразованного множителя, которые поступают на входы ОЧУС вместе с диадами множимого (см. рис.2). На трех выходах ОЧУС формируется результат умножения диад Мн · Мт + перенос из предыдущего ОЧУС. Максимальной цифрой в диаде преобразованного

множителя является двойка, поэтому перенос, формируемый ОЧУС, может быть только двоичным:

$$\begin{array}{ccccc} 3 & \cdot & 2 & = & 1\ 2. \\ \max & & \max & & \max \\ \text{Мн} & & \text{Мт} & & \text{перенос} \end{array}$$

Так как на входы ОЧУС из регистра Мт не могут поступить коды «3», в таблице истинности работы ОЧУС будут содержаться 16 безразличных входных наборов.

Частичные произведения, получаемые на выходах ОЧУС, складываются с накапливаемой частичной суммой из регистра результата с помощью цепочки ОЧС (на первом такте выполняется сложение с нулем).

Частичные суммы хранятся в регистре результата и регистре множителя, так как алгоритм умножения «А» предполагает возможность синхронного сдвига этих регистров. Количество тактов умножения определяется разрядностью Мт.

2.3. Разработка функциональных схем основных узлов сумматора-умножителя

Логический синтез одноразрядного четверичного умножителя

Одноразрядный четверичный умножитель - это комбинационное устройство, имеющее 5 входов (2 разряда из регистра Мн, 2 разряда из регистра Мт и управляющий вход h) и 4 выхода. Принцип работы ОЧУ представлен с помощью таблицы истинности (табл.3).

Разряды множителя закодированы : 0 - 00; 1 - 01; 2 - 10; 3 - 11.

Разряды множимого закодированы : 0 - 00; 1 - 11; 2 - 10; 3 - 01.

Управляющий вход h определяет тип операции: 0 - умножение закодированных цифр, поступивших на информационные входы; 1 - вывод на выходы без изменения значения разрядов, поступивших из регистра множимого.

В табл.3 выделено 8 безразличных наборов, так как на входы ОЧУ из разрядов множителя не может поступить код “11”.

Таблица 3

Таблица истинности ОЧУ

Мн		Мт		Упр.	Старшие разряды		Младшие разряды		Пример операции
x_1	x_2	y_1	y_2	h	P_1	P_2	P_3	P_4	в четверичной с/с
0	0	0	0	0	0	0	0	0	0·0=00

0	0	0	0	1	0	0	0	0	Выход - код «00»
0	0	0	1	0	0	0	0	0	0·1=00
0	0	0	1	1	0	0	0	0	Выход - код «00»
0	0	1	0	0	0	0	0	0	0·2=00
0	0	1	0	1	0	0	0	0	Выход - код «00»
0	0	1	1	0	x	x	x	x	0·3=00
0	0	1	1	1	x	x	x	x	Выход - код «00»
0	1	0	0	0	0	0	0	0	3·0=00
0	1	0	0	1	0	0	0	1	Выход - код «03»
0	1	0	1	0	0	0	0	1	3·1=03
0	1	0	1	1	0	0	0	1	Выход - код «03»
0	1	1	0	0	1	1	1	0	3·2=12
0	1	1	0	1	0	0	0	1	Выход - код «03»
0	1	1	1	0	x	x	x	x	3·3=21
0	1	1	1	1	x	x	x	x	Выход - код «03»
1	0	0	0	0	0	0	0	0	2·0=00
1	0	0	0	1	0	0	1	0	Выход - код «02»
1	0	0	1	0	0	0	1	0	2·1=02
1	0	0	1	1	0	0	1	0	Выход - код «02»
1	0	1	0	0	1	1	0	0	2·2=10
1	0	1	0	1	0	0	1	0	Выход - код «02»
1	0	1	1	0	x	x	x	x	2·3=12
1	0	1	1	1	x	x	x	x	Выход - код «02»
1	1	0	0	0	0	0	0	0	1·0=00
1	1	0	0	1	0	0	1	1	Выход - код «01»
1	1	0	1	0	0	0	1	1	1·1=01
1	1	0	1	1	0	0	1	1	Выход - код «01»
1	1	1	0	0	0	0	1	0	1·2=02
1	1	1	0	1	0	0	1	1	Выход - код «01»
1	1	1	1	0	x	x	x	x	1·3=03
1	1	1	1	1	x	x	x	x	Выход - код «01»

Минимизацию переключательных функций проведем с помощью карт Вейча. Для функции P_3 заполненная карта приведена на рис.6, где символом “x” отмечены наборы, на которых функция может принимать произвольное значение.

x_1

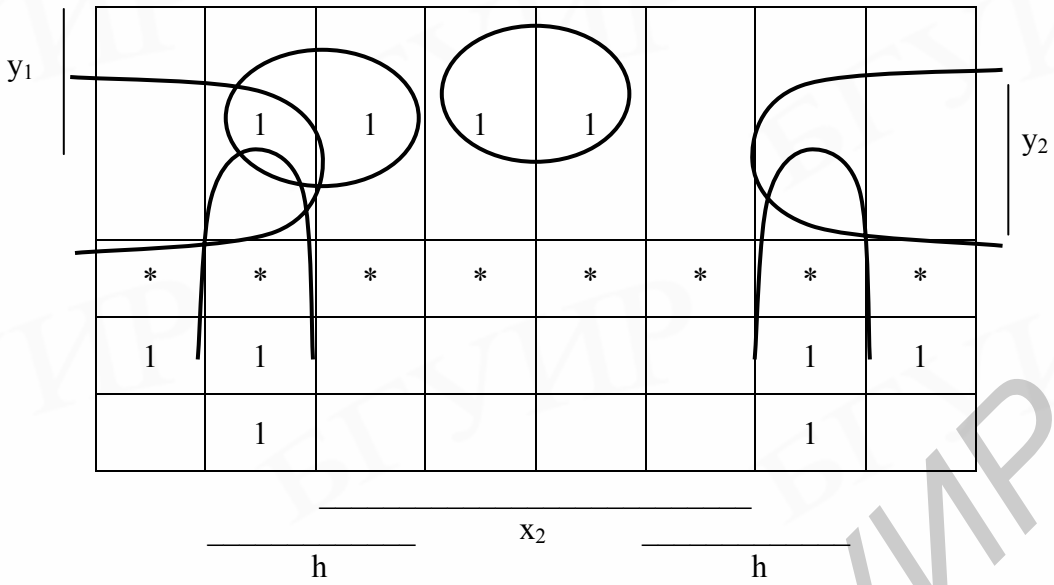


Рис.6. Минимизация функции при помощи карты Вейча

Следовательно, $P_3 = \overline{x_2}y_2 + x_1y_1h + x_2y_1\overline{h} + \overline{x_2}\,\overline{y_1}h$.

Эффективность минимизации можно оценить отношением числа входов схем, реализующих переключательную функцию до и после минимизации:

$$K = (10*5 + 10 + 5)/18 = 3{,}6.$$

Логический синтез одноразрядного четверичного сумматора

Одноразрядный четверичный сумматор - это комбинационное устройство, имеющее 5 входов (2 разряда одного слагаемого, 2 разряда второго слагаемого и вход переноса) и 3 выхода. Принцип работы ОЧС представлен с помощью таблицы истинности (табл.4).

Разряды обоих слагаемых закодированы : 0 - 00; 1 - 11; 2 - 10; 3 - 01.

Если ОЧС синтезируется для схемы 1-го типа, то в таблице истинности необходимо выделить 16 безразличных наборов, так как со старших выходов ОЧУ не могут прийти коды “2” и “3”. Если ОЧС синтезируется для схемы 2-го типа, то безразличные наборы в таблице истинности отсутствуют.

Таблица 4

Таблица истинности ОЧС

a_1	a_2	b_1	b_2	p	П	S1	S2	Пример операции в четверичной с/с
0	0	0	0	0	0	0	0	0+0+0=00
0	0	0	0	1	0	1	1	0+0+1=01
0	0	0	1	0	x	x	x	0+3+0=03
0	0	0	1	1	x	x	x	0+3+1=10
0	0	1	0	0	x	x	x	0+2+0=02
0	0	1	0	1	x	x	x	0+2+1=03

0	0	1	1	0	0	1	1	0+1+0=01

Окончание табл. 4

a_1	a_2	b_1	b_2	p	П	S1	S2	Пример операции в четверичной с/с
0	0	1	1	1	0	1	0	0+1+1=02
0	1	0	0	0	0	0	1	3+0+0=03
0	1	0	0	1	1	0	0	3+0+1=10
0	1	0	1	0	x	x	x	3+3+0=12
0	1	0	1	1	x	x	x	3+3+1=13
0	1	1	0	0	x	x	x	3+2+0=11
0	1	1	0	1	x	x	x	3+2+1=12
0	1	1	1	0	1	0	0	3+1+0=10
0	1	1	1	1	1	1	1	3+1+1=11
1	0	0	0	0	0	1	0	2+0+0=02
1	0	0	0	1	0	0	1	2+0+1=03
1	0	0	1	0	x	x	x	2+3+0=11
1	0	0	1	1	x	x	x	2+3+1=12
1	0	1	0	0	x	x	x	2+2+0=10
1	0	1	0	1	x	x	x	2+2+1=11
1	0	1	1	0	0	0	1	2+1+0=03
1	0	1	1	1	1	0	0	2+1+1=10
1	1	0	0	0	0	1	1	1+0+0=01
1	1	0	0	1	0	1	0	1+0+1=02
1	1	0	1	0	x	x	x	1+3+0=10
1	1	0	1	1	x	x	x	1+3+1=11
1	1	1	0	0	x	x	x	1+2+0=03
1	1	1	0	1	x	x	x	1+2+1=10
1	1	1	1	0	0	1	0	1+1+0=02
1	1	1	1	1	0	0	1	1+1+1=03

Определим множество единичных кубов:

L = {01001, 01110, 01111, 10111}

и множество безразличных кубов:

N ={00010, 00011, 00100, 00101,
01010, 01011, 01100, 01101,
10010, 10011, 10100, 10101,
11010, 11011, 11100, 11101}.

Сформируем множество C_0 = L U N.

Первым этапом алгоритма Рота является нахождение множества простых импликант. Для реализации этого этапа будем использовать операцию умножения (*) над множествами C_0, C_1 и т.д., пока в результате операции будут образо-

ваться новые кубы большей размерности. Первый шаг умножения (C_0*C_0) приведен в табл. 5. В результате этой операции сформируем новое множество кубов:

C_1 = {0001x, 0010x, 0101x, 010x1, 0110x, 0111x, 011x0, 011x1, 01x01, 01x10, 01x11, 0x010, 0x011, 0x100, 0x101, 1001x, 1010x, 101x1, 10x11, 1101x, 1110x, 1x010, 1x011, 1x100, 1x101, x0010, x0011, x0100, x0101, x1010, x1011, x1100, x1101}.

Множество Z_0 кубов, не участвовавших в образовании новых кубов, пустое.

В табл. 6 приведен следующий шаг поиска простых импликант с помощью операции C_1*C_1. В результате образовалось множество C_2 кубов второй размерности:

C_2 = {011xx, 01x1x, 01xx1, 0x01x, 0x10x, 1x01x, 1x10x, x001x, x010x, x101x, x110x, xx010, xx011, xx100, xx101}.

Множество Z_1 кубов, не участвовавших в образовании новых кубов:

Z_1 = {101x1, 10x11}.

В табл. 7 приведен следующий шаг поиска простых импликант – операция C_2*C_2. В результате образовалось множество C_3 кубов третьей размерности:

C_3 = {xx01x, xx10x}.

Множество Z_2 кубов, не участвовавших в образовании новых кубов:

Z_2 = {011xx, 01x1x, 01xx1}.

Таблица 5

Поиск простых импликант (C_0*C_0).

C_0*C_0	01001	01110	01111	10111	00010	00011	00100	00101	01010	01011	01100	01101	10010	10011	10100	10101	11010	11011	11100	11101
01001	-----																			
01110		-----																		
01111		0111x	-----																	
10111				-----																
00010					-----															
00011					0001x	-----														
00100							-----													
00101							0010x	-----												
01010		01x10			0x010				-----											
01011	010x1		01x11			0x011			0101x	-----										
01100		011x0					0x100				-----									
01101	01x01		011x1					0x101			0110x	-----								
10010					x0010								-----							
10011				10x11		x0011							1001x	-----						
10100							x0100								-----					
10101				101x1				x0101							1010x	-----				
11010									x1010				1x010				-----			
11011										x1011				1x011			1101x	-----		
11100											x1100				1x100				-----	
11101												x1101				1x101			1110x	-----

Таблица 6

Поиск простых импликант (C_1*C_1).

C_1*C_1	0001x	0010x	0101x	010x1	0110x	0111x	011x0	011x1	01x01	01x10	01x11	0x010	0x011	0x100	0x101	1001x	1010x	101x1	10x11	1101x	1110x	1x010	1x011	1x100	1x101	x0010	x0011	x0100	x0101	x1010	x1011	x11
0001x	-----																														
0010x		-----																													
0101x	0x01x		-----																												
010x1				-----																											
0110x		0x10x			-----																										
0111x			01x1x		011xx	-----																									
011x0							-----																								
011x1				01xx1			011xx	-----																							
01x01									-----																						
01x10										-----																					
01x11									01xx1	01x1x	-----																				
0x010												-----																			
0x011												0x01x	-----																		
0x100														-----																	
0x101														0x10x	-----																
1001x	x001x															-----															
1010x		x010x															-----														
101x1																		-----													
10x11																			-----												
1101x			x101x													1x01x				-----											
1110x					x110x												1x10x				-----										
1x010												xx010										-----									
1x011													xx011									1x01x	-----								
1x100														xx100										-----							
1x101															xx101									1x10x	-----						
x0010																										-----					
x0011																										x001x	-----				
x0100																												-----			
x0101																												x010x	-----		
x1010																										xx010				-----	
x1011																											xx011			x101x	-----	
x1100																												xx100				---
x1101																													xx101			x11

Таблица 7

Поиск простых импликант (C_2*C_2)

C_2*C_2	011xx	01x1x	01xx1	0x01x	0x10x	1x01x	1x10x	x001x	x010x	x101x	x110x	xx010	xx011	xx100	xx101
011xx	-----														
01x10		-----													
01xx1			-----												
0x01x				-----											
0x10x					-----										
1x01x				xx01x		-----									
1x10x					xx10x		-----								
x001x								-----							
x010x									-----						
x101x								xx01x		-----					
x110x									xx10x		-----				
xx010												-----			
xx011												xx01x	-----		
xx100														-----	
xx101														xx10x	-----

Таблица 8

Поиск простых импликант (C_3*C_3)

C_3*C_3	xx01x	xx10x
xx01x	-----	
xx10x		-----

Результат C_3*C_3 приведен в табл. 8. Новых кубов (четвертой размерности) не образовалось. Получено множество Z_3 = {xx01x, xx10x}.

На этом заканчивается этап поиска простых импликант, так как $|C_4| \le 1$. Множество простых импликант

$Z = Z_0 \cup Z_1 \cup Z_2 \cup Z_3$ = {101x1, 10x11, 011xx, 01x1x, 01xx1, xx01x, xx10x}.

Следующий этап – поиск L-экстремалей на множестве простых импликант. Для этого используется операция # (решетчатое вычитание). В табл. 9 из каждой простой импликанты поочередно вычитаются все остальные простые импликанты Z#(Z\z), результат операции (последняя строка таблицы) указывает на то, что L-экстремалями стали следующие простые импликанты:

E = 01xx1, xx01x, xx10x.

Необходимо проверить, нет ли среди них таких, которые стали L-экстремалями за счет безразличных кубов. Для этого в табл. 10 из кубов множества L вычитаются остатки простых импликант, полученные в табл. 9 (результат выполнения операции Z#(Z\z)). По результатам табл. 10 L-экстремалью, не связанной с безразличными наборами, стал куб 01xx1 (остаток от вычитания из него всех остальных простых импликант – 01001 – относится ко множеству

единичных наборов L исходного задания функции). Этот куб обязательно должен войти в минимальное покрытие.

Таблица 9

Поиск L-экстремалей

Z#(Z\z)	101x1	10x11	011xx	01x1x	01xx1	xx01x	xx10x
101x1	-----	zz0zz 10011	yyzz0 011xx	yy0z0 01x1x	yy0zz 01xx1	01yz0 xx01x	01zz0 0x01x x110x xx100
10x11	zzz0z 10101	-----	yyz00 011xx	yyzz0 01x1x	yyz0z 01xx1	01zz0 0x01x x101x xx010	y1zy0 0yzy0 01zyy 0x10x x110x xx100
011xx	yyzzz 10101	yyyzz 10011	-----	zz0zz 0101x	zz0zz 010x1	z0yzz 1zyzz 10yzz 0x01x x101x xx010	z0zzz 1zzzz 10zzz 0010x 1110x 1x100 x0100
01x1x	yyzyz 10101	yyyzz 10011	zzz0z 0110x	-----	zzz0z 01001	z0zzz 1zzzz 10zzz 0001x 1101x 1x010 x0010	zyzyz yzzyz y0zyz 1yzyz 0010x 1110x 1x100 x0100
01xx1	yyzzz 10101	yyzzz 10011	zzzz0 01100	zzzz0 01010	-----	zyzz0 yzzz0 y0zzy 1yzzy 0001x 1101x 1x010 x0010	zyzz0 yzzz0 y0zzy 1yzzy 0010x 1110x 1x100 x0100
xx01x	zzyyz 10101	zzzzz ∅	zzyyz 01100	zzzzz ∅	zzzyz 01001	----- ----- ----- -----	zzyyz zzyyz zzyyz zzyyz 0010x 1110x 1x100 x0100
xx10x	zzzzz ∅		zzzzz ∅		zzyzz 01001	zzyyz zzyyz zzyyz zzyyz 0001x 1101x 1x010 x0010	----- ----- ----- -----

Таблица 10

Проверка L-экстремалей

L ∩ Ê	01001	01110	01111	10111
01001	01001	∅	∅	∅
0001x	∅	∅	∅	∅
1101x	∅	∅	∅	∅
1x010	∅	∅	∅	∅
x0010	∅	∅	∅	∅
0010x	∅	∅	∅	∅
1110x	∅	∅	∅	∅
1x100	∅	∅	∅	∅
x0100	∅	∅	∅	∅

Далее необходимо проанализировать, какие из исходных единичных кубов (множество L) не покрыты найденной L-экстремалью. Этот анализ осуществляется с помощью табл. 11.

Таблица 11

Поиск непокрытых исходных наборов

L # E	01001	01110	01111	10111
01xx1	zzzzz ∅	zzzzy 01110	zzzzz ∅	yyzzz 10111

Из табл. 11 видно, что L-экстремалью не покрыты два единичных куба (01110 и 10111). Чтобы их покрыть, воспользуемся множеством простых импликант, не являющихся L-экстремалями (табл.12).

Таблица 12

Покрытие оставшихся кубов

L ∩ Ž	01110	10111
101x1	∅	10111
10x11	∅	10111

011xx	01110	∅
01x1x	01110	∅
xx01x	∅	∅
xx10x	∅	∅

Из табл. 12 видно, что каждый из непокрытых единичных кубов может быть покрыт двумя равнозначными способами. Следовательно, существуют 4 тупиковые (минимальные) формы:

$F_{min1} = \overline{a}_1 a_2 p + \overline{a}_1 a_2 b_1 + a_1 \overline{a}_2 b_1 p = \overline{a}_1 a_2 (p + b_1) + a_1 \overline{a}_2 b_1 p$;

$F_{min2} = \overline{a}_1 a_2 p + \overline{a}_1 a_2 b_1 + a_1 \overline{a}_2 b_2 p$;

$F_{min3} = \overline{a}_1 a_2 p + \overline{a}_1 a_2 b_2 + a_1 \overline{a}_2 b_1 p$;

$F_{min4} = \overline{a}_1 a_2 p + \overline{a}_1 a_2 b_2 + a_1 \overline{a}_2 b_2 p$.

Функциональную схему ОЧС (рис. 7) построим по F_{min1} :

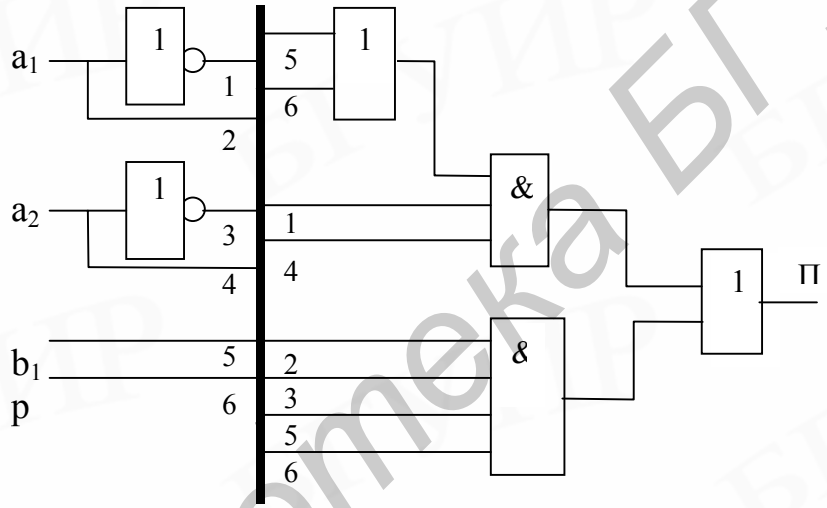


Рис.7. Функциональная схема ОЧС

Логический синтез одноразрядного четверичного умножителя-сумматора

ОЧУС - это комбинационное устройство, имеющее шесть входов (два разряда из регистра множимого, два разряда из регистра множителя, вход переноса и управляющий вход h) и три выхода. Принцип работы ОЧУС представлен с помощью таблицы истинности (табл. 13).

Разряды множителя закодированы : 0 - 00; 1 - 01; 2 - 10; 3 - 11.

Разряды множимого закодированы : 0 - 00; 1 - 11; 2 - 10; 3 - 01.

Управляющий вход h определяет тип операции: 0 - умножение закодированных цифр, поступивших на информационные входы, и добавление переноса; 1 - вывод на выходы без изменения значения разрядов, поступивших из регистра множимого. В табл. 13 выделено 36 безразличных наборов, так как на входы ОЧУС из разрядов множителя не может поступить код 11, при работе ОЧУС

как сумматора на вход переноса не может поступить единица, а при умножении на ноль или единицу на вход переноса также не может поступить единица.

Таблица 13

Таблица истинности ОЧУС

Пер	Мн		Мт		Упр.	Перенос	Результат		Результат операции в четверичной с/с
P_1	x_1	x_2	y_1	y_2	h	P	Q_1	Q_2	
0	0	0	0	0	0	0	0	0	0·0+0=00
0	0	0	0	0	1	0	0	0	Выход – код «00»
0	0	0	0	1	0	0	0	0	0·1+0=00
0	0	0	0	1	1	0	0	0	Выход – код «00»
0	0	0	1	0	0	0	0	0	0·2+0=00
0	0	0	1	0	1	0	0	0	Выход - код «00»
0	0	0	1	1	0	x	x	x	0·3+0=00
0	0	0	1	1	1	x	x	x	выход - код «00»
0	0	1	0	0	0	0	0	0	3·0+0=00
0	0	1	0	0	1	0	0	1	выход - код «03»
0	0	1	0	1	0	0	0	1	3·1+0=03
0	0	1	0	1	1	0	0	1	выход - код «03»
0	0	1	1	0	0	1	1	0	3·2+0=12
0	0	1	1	0	1	0	0	1	выход - код «03»
0	0	1	1	1	0	x	x	x	3·3+0=21
0	0	1	1	1	1	x	x	x	выход - код «03»
0	1	0	0	0	0	0	0	0	2·0+0=00
0	1	0	0	0	1	0	1	0	выход - код «02»
0	1	0	0	1	0	0	1	0	2·1+0=02
0	1	0	0	1	1	0	1	0	выход - код «02»
0	1	0	1	0	0	1	0	0	2·2+0=10
0	1	0	1	0	1	0	1	0	выход - код «02»
0	1	0	1	1	0	x	x	x	2·3+0=12
0	1	0	1	1	1	x	x	x	выход - код «02»
0	1	1	0	0	0	0	0	0	1·0+0=00
0	1	1	0	0	1	0	1	1	Выход - код «01»
0	1	1	0	1	0	0	1	1	1·1+0=01
0	1	1	0	1	1	0	1	1	выход - код «01»
0	1	1	1	0	0	0	1	0	1·2+0=02
0	1	1	1	0	1	0	1	1	выход - код «01»
0	1	1	1	1	0	x	x	x	1·3+0=03
0	1	1	1	1	1	x	x	x	выход - код «01»
1	0	0	0	0	0	x	x	x	0·0+1=01

1	0	0	0	0	1	x	x	x	выход - код «00»
1	0	0	0	1	0	x	x	x	0·1+1=01
1	0	0	0	1	1	x	x	x	выход – код «00»
1	0	0	1	0	0	0	1	1	0·2+1=01
1	0	0	1	0	1	x	x	x	Выход - код «00»

Окончание табл. 13

Пер	Мн		Мт		Упр.	Перенос	Результат		Результат операции
P_1	x_1	x_2	y_1	y_2	h	P	Q_1	Q_2	в четверичной с/с
1	0	0	1	1	0	x	x	x	0·3+1=01
1	0	0	1	1	1	x	x	x	Выход - код «00»
1	0	1	0	0	0	x	x	x	3·0+1=01
1	0	1	0	0	1	x	x	x	выход - код «03»
1	0	1	0	1	0	x	x	x	3·1+1=10
1	0	1	0	1	1	x	x	x	выход - код «03»
1	0	1	1	0	0	1	0	1	3·2+1=13
1	0	1	1	0	1	x	x	x	выход – код «03»
1	0	1	1	1	0	x	x	x	3·3+1=22
1	0	1	1	1	1	x	x	x	выход – код «03»
1	1	0	0	0	0	x	x	x	2·0+1=01
1	1	0	0	0	1	x	x	x	выход - код «02»
1	1	0	0	1	0	x	x	x	2·1+1=03
1	1	0	0	1	1	x	x	x	выход - код «02»
1	1	0	1	0	0	1	1	1	2·2+1=11
1	1	0	1	0	1	x	x	x	выход – код «02»
1	1	0	1	1	0	x	x	x	2·3+1=13
1	1	0	1	1	1	x	x	x	выход - код «02»
1	1	1	0	0	0	x	x	x	1·0+1=01
1	1	1	0	0	1	x	x	x	выход - код «01»
1	1	1	0	1	0	x	x	x	1·1+1=02
1	1	1	0	1	1	x	x	x	выход - код «01»
1	1	1	1	0	0	0	0	1	1·2+1=03
1	1	1	1	0	1	x	x	x	выход - код «01»
1	1	1	1	1	0	x	x	x	1·3+1=10
1	1	1	1	1	1	x	x	x	выход - код «01»

Синтез выходов ОЧУС в методическом пособии не рассматривается.

На рис. 8 приведена карта Вейча для минимизации функции переноса P.

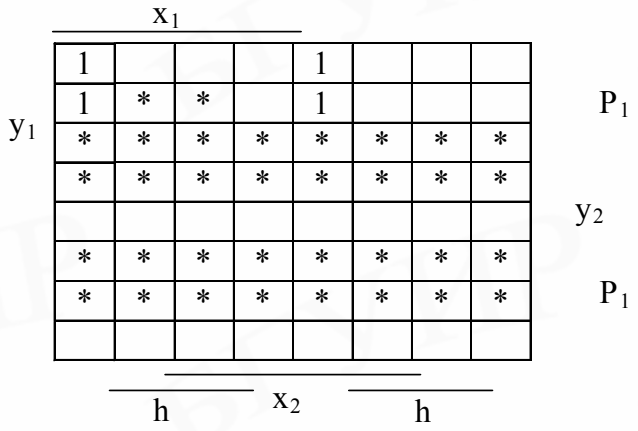


Рис.8. Карта Вейча для минимизации функции P

Синтез комбинационных схем устройств умножения на основе мультиплексоров

Мультиплексор – это логическая схема, имеющая n информационных входов, m управляющих входов и один выход. При этом должно выполняться условие $n = 2^m$.

Принцип работы мультиплексора состоит в следующем. На выход мультиплексора может быть пропущен без изменений любой (один) логический сигнал, поступающий на информационные входы. Порядковый номер информационного входа, значение с которого в данный момент должно быть передано на выход, определяется двоичным кодом на управляющих входах.

На рис. 9 показан мультиплексор, имеющий 4 информационных (I0–I3) и два управляющих (S0, S1) входа, так называемый ”один из четырех”. В табл. 14 определена зависимость выходного сигала Y от сигналов на входах мультилексора. Сигналы x_1, x_2, x_3, x_4 - это логические сигналы, поступающие на вход мультиплексора, которые могут принимать значения ноль или единица.

Из табл. 14 видно, что под действием управляющей комбинации $\overline{S0}\,\overline{S1}$ на выход будет пропущен сигнал, поданный на вход I0 (в нашем случае это x_1), а управляющая комбинация $\overline{S0}S1$ пропускает на выход сигнал, поданный на вход I2 (в нашем случае это x_3).

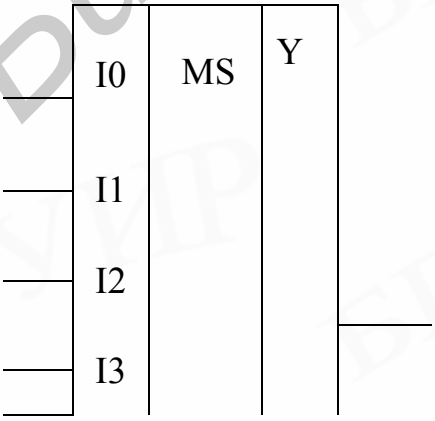


Таблица 14

Работа мультиплексора

S0	S1	I0	I1	I2	I3	Y
0	0	x_1	x_2	x_3	x_4	x_1
0	1	x_1	x_2	x_3	x_4	x_2
1	0	x_1	x_2	x_3	x_4	x_3
1	1	x_1	x_2	x_3	x_4	x_4

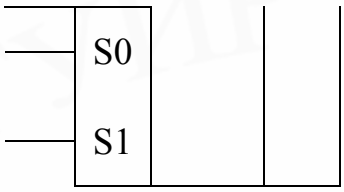


Рис.9. Мультиплексор

Мультиплексор может быть использован для синтеза комбинационных схем. С помощью мультиплексора ”один из четырех” легко реализовать любую переключательную функцию (ПФ) от двух переменных. Пример реализации функций ИЛИ и И на мультиплексоре по таблицам истинности 15 и 16 приведен соответственно на рис. 10 и 11.

Таблица 15

Функция ”ИЛИ”

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	1

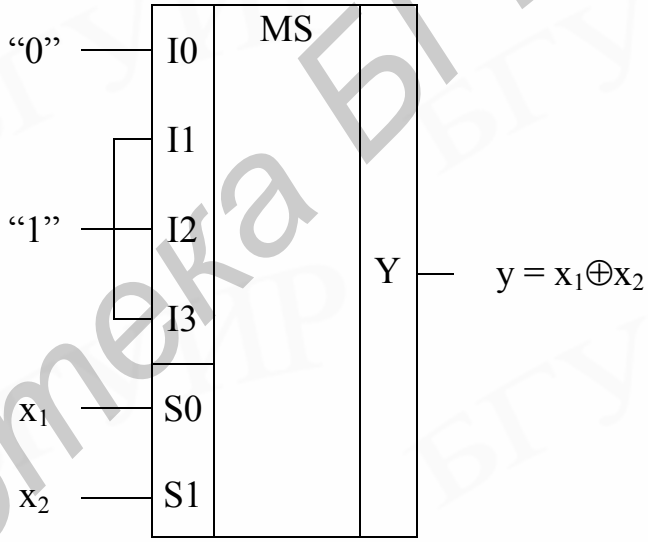
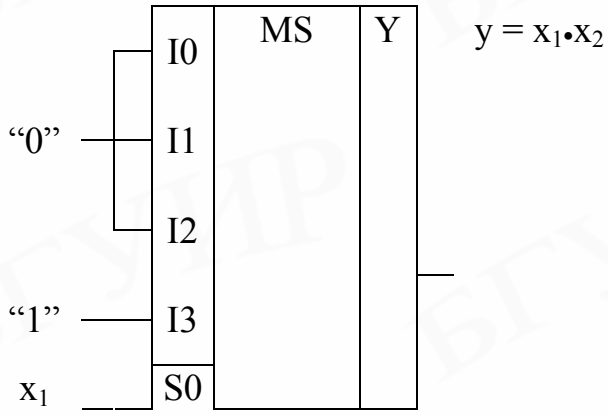


Рис.10. Реализация функции ”ИЛИ” на мультиплексоре

Таблица 16

Функция ”И”

x_1	x_2	$x_1 \cdot x_2$
0	0	0
0	1	0
1	0	0
1	1	1



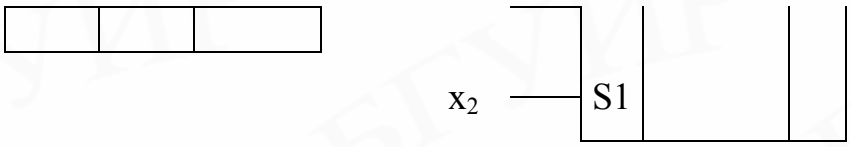


Рис.11. Реализация функции ”И” на мультиплексоре

На основе мультиплексора можно синтезировать ПФ более чем от двух переменных. Для этого на входе мультиплексора, возможно, придется разместить некоторую дополнительную логическую схему. Для синтеза этой дополнительной схемы все наборы в таблице истинности (табл. 17) целесообразно поделить на группы так, чтобы в каждой группе наборы переменных x_1, x_2 были одинаковы. Таких групп с одинаковыми наборами 00, 01, 10, 11 будет четыре.

Для синтеза входной логической схемы независимыми переменными будут только x_3 и x_4, которые в свою очередь образуют четыре различных набора в каждой группе. Записывая для единичных значений ПФ логические выражения для входных переменных x_3 и x_4, строим затем по этим выражениям для каждого входа I0 – I3 логическую схему. Например, для первой и четвертой групп $y=x_3\overline{x_4}+\overline{x_3}x_4 = x_3 \oplus x_4$, а для второй и третьей групп y = 1, поскольку ПФ на всех восьми наборах равна 1.

Мультиплексор с входной логической схемой для реализации ПФ четырех переменных показан на рис. 12.

Таблица 17

Таблица истинности

№	x_1	x_2	x_3	x_4	функция
0	0	0	0	0	$x_3 \oplus x_4$
1	0	0	0	1	
2	0	0	1	0	
3	0	0	1	1	
4	0	1	0	0	“1”
5	0	1	0	1	
6	0	1	1	0	
7	0	1	1	1	
8	1	0	0	0	“1”
9	1	0	0	1	
10	1	0	1	0	
11	1	0	1	1	
12	1	1	0	0	$x_3 \oplus x_4$
13	1	1	0	1	
14	1	1	1	0	
15	1	1	1	1	

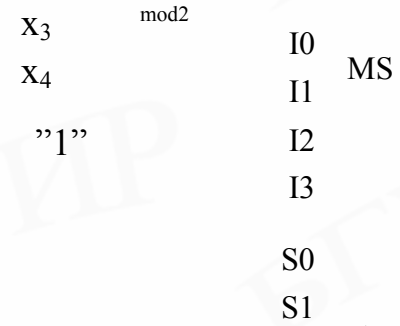


Рис.12. Реализация ПФ четырех переменных

Переключательные функции пяти переменных можно реализовать на мультиплексоре “один из восьми”, применяя аналогичный подход. Здесь управляющее поле определяется тремя переменными x_1, x_2, x_3, поэтому число групп с одинаковыми значениями этих переменных будет равно восьми (табл. 18). Заметим еще раз, что каждая такая группа управляет одним из восьми входов $I0-17$.

Таблица 18

Таблица истинности

	x_1	x_2	x_3	x_4	x_5	y
0	0	0	0	0	0	0
1	0	0	0	0	1	1
2	0	0	0	1	0	1
3	0	0	0	1	1	0
4	0	0	1	0	0	1
5	0	0	1	0	1	1

6	0	0	1	1	0	1
7	0	0	1	1	1	1
8	0	1	0	0	0	0
9	0	1	0	0	1	1
.	.	.	.	.	.	.
24	1	1	0	0	0	0
25	1	1	0	0	1	0
26	1	1	0	1	0	0
27	1	1	0	1	1	0
28	1	1	1	0	0	0
29	1	1	1	0	1	0
30	1	1	1	1	0	0
31	1	1	1	1	1	1

Входная логическая схема синтезируется только для входных переменных x_4 и x_5 исходя из заданных единичных значений ПФ, оказавшихся в той или иной группе. Например, для нулевой группы $y = \bar{x}_4\bar{x}_5 + x_4x_5 = \overline{x_4 \oplus x_5}$; для первой группы $y = \bar{x}_4\bar{x}_5 + \bar{x}_4x_5 = \bar{x}_4$ и т.д., а для седьмой и восьмой групп $y = \bar{x}_4\bar{x}_5 + x_4\bar{x}_5 = \bar{x}_5$ и $y = \bar{x}_4x_5 + x_4x_5 = x_5$. Реализация нескольких ПФ, как например ОЧС, потребует для каждой ПФ отдельного мультиплексора (рис. 13).

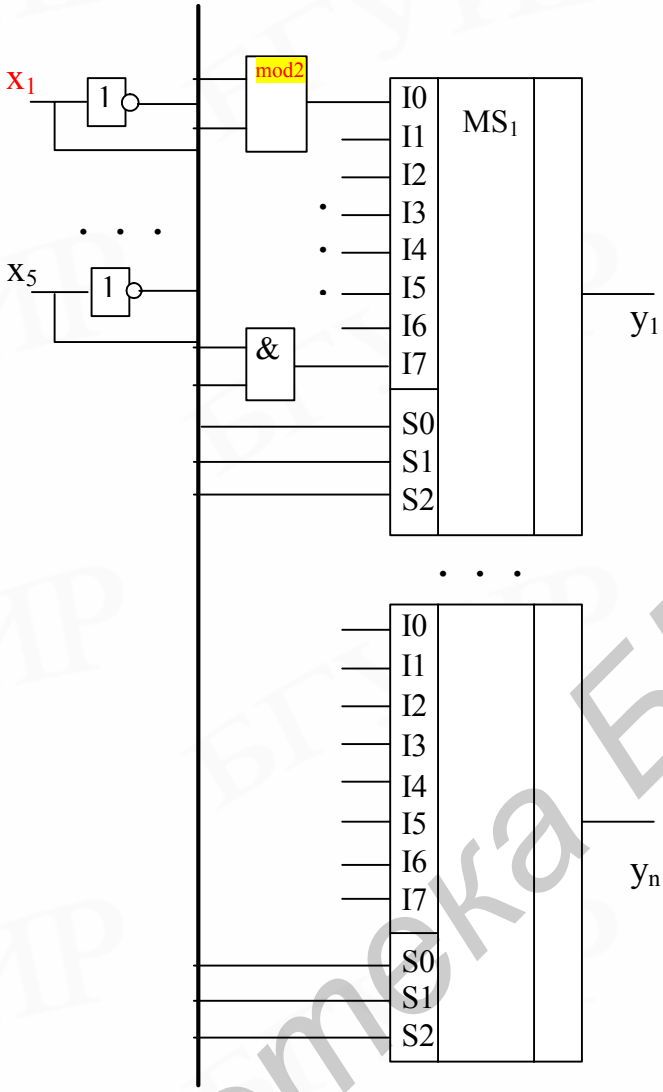


Рис. 13. Реализация функций $y_1 \ldots y_n$ на n мультиплексорах

ЛИТЕРАТУРА

1. Савельев А.Я. Прикладная теория цифровых автоматов. М.: Высш. шк., 1985.

2. Лысиков Б.Г. Арифметические и логические основы цифровых автоматов. Мн.: Выш. шк., 1980.

3. Лысиков Б.Г. Цифровая вычислительная техника. Мн., 2003.

4. Луцик Ю.А., Лукьянова И.В., Ожигина М.П. Учебное пособие по курсу "Арифметические и логические основы вычислительной техники". Мн.: МРТИ, 2001.

5. Луцик Ю.А., Лукьянова И.В. Арифметические и логические основы вычислительной техники: Учеб. пособие. Мн.: МРТИ , 2004.

ПРИЛОЖЕНИЕ

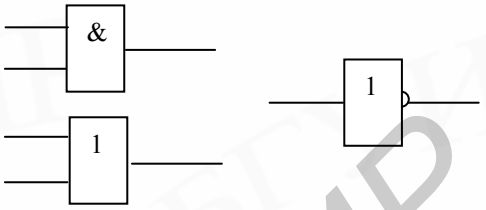
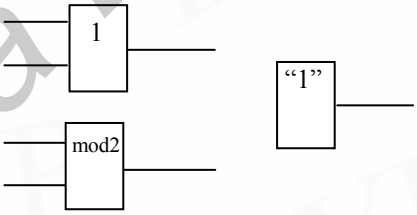
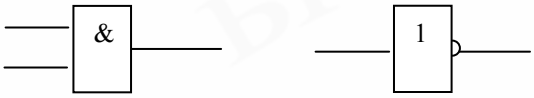
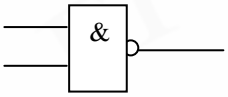
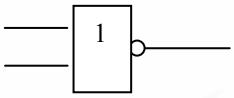
Таблица П.1

Исходные данные к курсовой работе

№вар.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	Сомножители в десятичной системе счисления																							
Мн	43, 34	81, 92	65, 91	15, 44	36, 39	48, 51	68, 39	28, 69	73, 48	49, 27	72, 34	67, 83	56, 59	29, 63	52, 26	48, 27	84, 19	16, 35	31, 50	72, 95	37, 32	28, 12	78, 11	42, 97
Мт	32, 65	44, 35	17, 39	47, 31	53, 25	69, 11	16, 71	21, 59	49, 13	38, 70	35, 44	25, 37	18, 27	63, 29	83, 31	72, 23	55, 13	67, 21	45, 17	67, 65	28, 15	69, 97	25, 17	55, 39
2	Алгоритм выполнения операции умножения																							
	А	Б	В	Г	А	Б	В	Г	А	Б	В	Г	А	Б	В	Г	А	Б	В	Г	А	Б	В	Г
3	Варианты кодирования четверичных цифр двоичными кодами																							
“0”	00	00	00	00	00	01	01	01	01	01	01	10	10	10	10	11	11	11	11	00	10	10	11	11
“1”	01	10	10	11	11	00	00	10	10	11	11	00	00	01	11	01	00	00	01	01	01	11	10	10
“2”	11	01	11	01	10	10	11	00	11	00	10	01	11	00	00	10	01	10	00	10	11	01	01	00
“3”	10	11	01	10	01	11	10	11	00	10	00	11	01	11	01	00	10	01	10	11	00	00	00	01
4	Логический базис для реализации ОЧС (см. табл. П.2) и метод минимизации.																							
	А1	А2	А3	А4	А5	А6	А7	А1	А2	А3	А4	А5	А6	А7	А1	А2	А3	А4	А5	А6	А7	А1	А2	А3
	Карты Карно-Вейча												Алгоритм Рота											
5	Логический базис для реализации ОЧУ(ОЧУС, см. табл П.2) и метод минимизации.																							
	А4	А5	А6	А7	А1	А2	А3	А4	А5	А6	А7	А1	А2	А3	А4	А5	А6	А7	А1	А2	А3	А4	А5	А6
	Алгоритм Рота												Карты Карно-Вейча											
6	Тип реализуемой структурной схемы																							
	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	1	2	1	2	1

Таблица П.2

Логический базис для реализации схем

Функционально полный логический базис		Базовые логические элементы
A1	$\overline{x}$ x_1x_2 $x_1 + x_2$	
A2	x_1x_2 $x_1 \oplus x_2$	
A3	x_1+x_2 $x_1 \oplus x_2$	
A4	x_1x_2 $\bar{x}$	
A5	x_1+x_2 $\bar{x}$	
A6	$\overline{x_1x_2}$	
A7	$\overline{x_1+x_2}$	

Св. план 2004, поз.58

Учебное издание

Лукьянова Ирина Викторовна,
Луцик Юрий Александрович

АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

МЕТОДИЧЕСКОЕ ПОСОБИЕ
к курсовому проекту
для студентов специальности
”Вычислительные машины, системы и сети”
всех форм обучения

Редактор Н.А. Бебель
Корректор Е.Н. Батурчик

Подписано в печать 15.11.2004. Формат 60x84 1/16. Бумага офсетная.
Гарнитура «Таймс». Печать ризографическая. Усл. печ. л.2,21.
Уч.-изд. л. 1,5. Тираж 200 экз. Заказ 81.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
Лицензия на осуществление издательской деятельности №02330/0056964 от 01.04.2004.
Лицензия на осуществление полиграфической деятельности №02330/0133108 от 30.04.2004.
220013, Минск, П. Бровки, 6