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

Кафедра электронных вычислительных машин

# АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Методические указания и контрольные задания
для студентов специальности 1-40 02 01
«Вычислительные машины, системы и сети»
заочной формы обучения



Минск БГУИР 2010

УДК 004.315(076)
ББК 32.973.26-04я73
А81

С о с т а в и т е л и:
Ю. А. Луцик, И. В. Лукьянова, А. В. Бушкевич



**Арифметические** и логические основы вычислительной техники : А81 метод. указания и контрол. задания для студ. спец. 1-40 02 01 «Вычислительные машины, системы и сети» заоч. формы обуч. / сост. Ю. А. Луцик, И. В. Лукьянова, А. В. Бушкевич. – Минск : БГУИР, 2010. – 27 с. : ил.

Включены необходимые сведения для выполнения контрольной работы и методические указания к изучению каждой из восьми тем дисциплины «Арифметические и логические основы вычислительной техники», вопросы для самоконтроля, литература, а также варианты заданий к контрольной работе для студентов заочной формы обучения.

**УДК 004.315(076)**
**ББК 32.973.26-04я73**

**Введение**

Дисциплина «Арифметические и логические основы вычислительной техники» занимает важное место в подготовке и формировании специалиста-инженера, которому предстоит изучить целый ряд дисциплин, базирующихся на сведениях и знаниях из указанной дисциплины, таких, например, как «Схемотехника ЭВМ», «Структурная и функциональная организация ЭВМ» и др.

Основная цель и задача дисциплины – овладение:

– информационными основами цифровых автоматов (ЦА);

– методами представления чисел в ЭВМ, алгоритмами выполнения арифметических и логических операций;

– логическими основами ЦА на основе изучения булевой алгебры;

– основными понятиями теории конечных автоматов применительно к синтезу управляющих автоматов.

Дисциплина «Арифметические и логические основы вычислительной техники» содержит следующие темы:

1. Системы счисления. Перевод чисел из одной системы счисления в другую.
2. Кодирование чисел прямым, дополнительным и обратным кодом.
3. Представление чисел в ЭВМ.
4. Арифметические операции в двоичной системе счисления.
5. Двоично-десятичные (BCD) коды.
6. Знакоразрядные системы счисления и СОК.
7. Логические основы цифровых автоматов.
8. Основы теории конечных автоматов.

**Литература**

*Основная*

1. Савельев, А. Я. Арифметические и логические основы цифровых автоматов / А. Я. Савельев. – М. : Высш. шк., 1980.

2. Савельев, А. Я. Прикладная теория цифровых автоматов / А. Я. Савельев. – М. : Высш. шк., 1987.

3. Лысиков, Б. Г. Арифметические и логические основы цифровых автоматов / Б. Г. Лысиков. – Минск : Выш. шк., 1980.

4. Поснов, Н. Н. Арифметика вычислительных машин в упражнениях и задачах: системы счисления, коды / Н. Н. Поснов. – Минск : Университетское, 1984.

5. МикроЭВМ, микропроцессоры и основы программирования / А. Н. Морозевич [и др.]. – Минск : Выш. шк., 1990.

6. Акушинский, И.Я. Машинная арифметика в остаточных классах / И. Я. Акушинский, Д. И. Юдицкий. – М. : Сов. радио, 1968.

7. Питерсон, У. Коды, исправляющие ошибки / У. Питерсон, Э. Уэлдон; пер. с англ. – М. : Мир, 1976.

8. Баранов, С. И. Синтез микропрограммных автоматов / С. И. Баранов. – Л. : Энергия, 1976 (1979).

9. Скляров, В. А. Синтез микропрограммных автоматов на матричных БИС / В. А. Скляров ; под ред. С. И. Баранова. – Минск : Выш. шк., 1980.

10. Луцик, Ю. А. Учебное пособие по курсу «Арифметические и логические основы вычислительной техники» / Ю. А. Луцик, И. В. Лукьянова. – Минск : БГУИР, 2004.

11. Луцик, Ю. А. Электронный учебно-методический комплекс по курсу «Арифметические и логические основы вычислительной техники» [Электронный ресурс] / Ю. А. Луцик, И. В. Лукьянова.

*Дополнительная*

12. Поспелов, Д. А. Арифметические основы вычислительных машин дискретного действия / Д. А. Поспелов. – М. : Высш. шк., 1970.

13. Карцев, М. А. Арифметика цифровых машин / М. А. Карцев. – М. : Наука, 1969.

14. Каган, Б. М. Электронные вычислительные машины и системы / Б. М. Каган. – М. : Энергия, 1979.

15. Арифметика вычислительных машин в упражнениях и задачах. – Минск : Университетское, 1984.

## Тема 1. Системы счисления. Перевод чисел из одной системы счисления в другую

Понятие системы счисления. Разновидности систем счисления. Основные характеристики позиционных систем счисления: веса разрядов, основание системы счисления. Критерии выбора системы счисления. Двоичная система счисления и ее достоинства. Понятие разрядного числа, вес единицы разряда для целых и дробных чисел. Два основных метода перевода чисел из одной системы счисления в любую другую:

1. Метод подбора степени (величины обратной степени) основания системы счисления.

2. Метод деления (умножения) на основание системы счисления.

[1, гл. 2; 2, гл. 1; 3, гл. 2]

*Методические указания*

В данном разделе необходимо изучить понятие системы счисления, типы систем счисления, способы перевода чисел из одной системы счисления в другую.

Следует различать два типа систем счисления – непозиционные и позиционные.

Непозиционная система счисления – система, для которой значение символа не зависит от его положения в числе. Примером может служить система счисления

с одной цифрой 1. Для записи любого числа в ней необходимо написать количество единиц, равное числу. Другой пример – это римская система счисления.

Позиционной системой счисления называется система записи любых по величине чисел, в которой значение цифры зависит от ее положения в числе, т. е. *веса*. Число цифр в позиционной системе счисления ограничено.

Основание (базис) r позиционной системы счисления – максимальное количество различных знаков или символов, используемых для изображения числа в данной системе счисления. Таким образом, основание может быть любым числом, кроме 1 и бесконечности. При r = 10 для записи чисел используются десять символов (0, 1, 2, ..., 9), при r = 2 – два символа (0, 1), при r = 8 – восемь символов (0, 1, 2, ..., 7) и т. д., системы счисления в данном случае называются соответственно десятичной, двоичной, восьмеричной и т. д.

Любое число в системе счисления с основанием r может быть записано в общем виде:

$$A = a_n \cdot r^n + a_{n-1} \cdot r^{n-1} + ... + a_1 \cdot r^1 + a_0 \cdot r^0 + a_{-1} \cdot r^{-1} + ... + a_{-rn-1} \cdot r^{-(rn-1)} + a_{-rn} \cdot r^{-rn}, \quad (1)$$

или

$$A_i = \sum_{i=-m}^{n} a_i r^i, \quad (2)$$

где любая разрядная цифра $a_i \in \{0, \ldots, r-1\}$, а $r^i$ – вес соответствующего разряда.

Запись числа в форме (1) назовем записью числа в развернутой форме. Свернутой формой записи чисел называется запись чисел в виде

$A = a_1 a_2 \ldots a_k$.

В вычислительной технике для представления данных и выполнения арифметических операций над ними удобно использовать двоичную, восьмеричную и шестнадцатеричную системы счисления.

Перевести число из системы счисления с основанием $r_A$ в систему счисления с основанием $r_B$ значит найти не только символы числа в новой системе счисления, но и веса этих символов.

Необходимо отметить, что целая и дробная части числа переводятся отдельно. Следовательно, все методы перевода чисел можно подразделить на две группы: перевода целых и дробных чисел.

При переводе небольших чисел можно использовать метод подбора степеней оснований систем счисления. При этом сначала определяется наибольшая степень основания ($r_B^{n-1}$) новой системы, т. е. вес старшей цифры, а затем и сама цифра. Далее находятся веса и цифры последующих разрядов вплоть до младшего. При переводе правильных дробей используется метод подбора величин, обратных степеням основания системы счисления.

Более универсальным для перевода целых чисел является метод деления на основание новой системы счисления, при этом первый остаток от деления является младшей цифрой, последний остаток – старшей цифрой. Перевод правильных дробей производится последовательным умножением на основание

новой системы счисления, что соответствует делению исходной дроби на величину, обратную основанию.

*Вопросы для самоконтроля*

1. Почему система счисления называется позиционной?
2. Какие символы содержит система с основанием 8, 16?
3. Чем объяснить широкое применение двоичной системы?
4. Как записывается основание любой системы счисления?
5. Чему всегда равен вес младшего разряда целого числа?
6. Как связан вес старшего разряда целого числа с числом разрядов?
7. Почему первый остаток от деления исходного числа на основание новой системы является младшим разрядом числа в новой системе?
8. На что можно было бы делить при переводе дроби из некоторой системы счисления в новую систему?
9. Чему равен вес старшего разряда дроби?
10. Какому закону подчиняется изменение весов разрядов?

## Тема 2. Кодирование чисел прямым, дополнительным и обратным кодами

Задача кодирования чисел. Кодирование знака. Прямой код числа. Дополнение числа и образование дополнительного кода. Замена операции вычитания операцией сложения. Образование обратного кода. Сравнительная оценка дополнительного и обратного кодов. Образование модифицированных кодов. Представление нуля в различных кодах.

[1, гл. 3; 2, гл. 1; 3, гл. 2]

*Методические указания*

Необходимо изучить задачи и способы кодирования чисел, ознакомиться с использованием различных кодов для выполнения арифметических операций.

Кодирование чисел позволяет заменить операцию арифметического вычитания операцией алгебраического сложения с помощью двоичного сумматора. Для кодирования знака числа используется специальный двоичный разряд, называемый знаковым. При этом знак плюс кодируется двоичной цифрой 0, а минус – цифрой 1 (для системы счисления с основанием r – цифрой r – 1). Для машинного представления отрицательных чисел используют три основных вида кодов: прямой, обратный и дополнительный. Общая схема кода числа: код знака . код числа.

Для арифметических операций над числами в прямом коде используется сумматор прямого кода. В этом сумматоре отсутствует цепь поразрядного переноса между старшим значащим и знаковым разрядами, т. е. на этом сумматоре невозможно выполнение операции алгебраического сложения.

Недостатком прямого кода является сложность выполнения операции сложения чисел с разными знаками. Для этого используются дополнительный и обратный коды. Положительные числа во всех кодах кодируются одинаково: знак «+» кодируется нулем, затем следует значащая часть числа. Отрицательные числа перед значащей частью содержат единицу, которая кодирует знак числа.

Основой дополнительного кода является дополнение A', которое и составляет значащую часть числа, причем $A' = r^n - A$, если A – целое число, и $A' = 1 - A$, если A – правильная дробь. Следует отметить, что для двоичного числа дополнительный код образуется достаточно просто, поскольку для получения дополнения нужно все цифры исходного числа инвертировать и в младший разряд добавить единицу. При формировании обратного кода в отличие от дополнительного, единица в младший разряд не добавляется.

При выполнении некоторых арифметических операций может возникать явление переполнения разрядной сетки. Причиной переполнения может служить суммирование двух чисел с одинаковыми знаками (для чисел с разными знаками переполнение не возникает), которые в сумме дают величину, большую или равную единице (при сложении правильных дробей), или величину $r^n$ (при сложении целых чисел).

Пример: $A = +0{,}101$ $\quad [A]_{доп} = 0{,}101$

$B = +0{,}110$ $\quad [B]_{доп} = \underline{0{,}110}$

$[A + B]_{доп} = 1{,}011$

В результате сложения двух положительных чисел получено отрицательное число, что является ошибкой. Результат неверен также и по величине.

Для обнаружения переполнения можно использовать следующие признаки:

– знаки слагаемых не совпадают со знаком суммы;

– есть перенос только в знаковый или только из знакового разряда.

Функция переполнения имеет вид: $f = \Pi_1\Pi_2 + \overline{\Pi}_1\overline{\Pi}_2 = \Pi_1 \oplus \Pi_2$.

Если при сложении чисел с фиксированной запятой возникло переполнение, то вырабатывается сигнал переполнения разрядной сетки и вычисления прекращаются.

Для обнаружения переполнения разрядной сетки можно использовать модифицированные коды. Модифицированные коды отличаются от обычных кодов тем, что знак числа кодируется двумя разрядами. При выполнении алгебраического сложения или вычитания два знаковых разряда участвуют в операции как равноправные цифровые разряды. После выполнения операции содержимое знаковых разрядов определяет знак результата (левый знаковый разряд) и наличие переполнения (несовпадение знаковых разрядов): комбинация 01 фиксирует переполнение при сложении положительных чисел (положительное переполнение), а 10 – отрицательных (отрицательное переполнение).

Пример: A = +0,101 $[A]^{мод}_{доп} = 00,101$

B = +0,110 $[B]^{мод}_{доп} = \underline{00,110}$

$[A]^{мод}_{доп} + [B]^{мод}_{доп} = 01,011$

A = –0,101 $[A]^{мод}_{доп} = 11,011$

B = –0,110 $[B]^{мод}_{доп} = \underline{11,010}$

$[A]^{мод}_{доп} + [B]^{мод}_{доп} = 10,101$

Функция переполнения имеет вид: $f = Зн1 \cdot \overline{Зн2} + \overline{Зн1} \cdot Зн2 = Зн1 \oplus Зн2$.

*Вопросы для самоконтроля*

1. Для чего необходимо кодирование чисел?
2. Можно ли обойтись только одним прямым кодом?
3. Почему единица переноса из знакового разряда отбрасывается при сложении в дополнительных кодах, если она возникает? Что нужно сделать в аналогичном случае при использовании обратных кодов?
4. Почему образование дополнительного (обратного) кода сложнее для систем счисления с основанием, большим двух?
5. Для чего предназначены модифицированные коды?

## Тема 3. Представление чисел в ЭВМ

Форматы представления: с фиксированной и плавающей запятой. Представление целых и дробных чисел. Диапазоны изменения чисел, представленных в этих форматах, и их сравнительная оценка. Нормализация чисел.

*Методические указания*

Необходимо изучить, каким образом представляются в памяти ЭВМ целые и дробные числа, понятия нормализации и округления.

При изучении этой темы необходимо обратить внимание на то, что мантисса числа с плавающей запятой является правильной дробью, а порядок – целым числом со знаком. Числа с фиксированной запятой представляются одной мантиссой (порядок равен нулю), которая может быть числом целым или правильной дробью. Диапазон изменения чисел с плавающей запятой больше, чем с фиксированной, при одном и том же количестве разрядов, однако сложение двух чисел в первом случае требует больше времени, чем во втором, за счет выравнивания порядков. Для выполнения операций над порядками необходимо также дополнительное оборудование. Нормализации подлежат только числа с плавающей запятой путем сдвига мантиссы вправо или влево так, чтобы после

запятой находилась цифра, отличная от нуля. Сдвиг мантиссы на один разряд сопровождается добавлением (вычитанием) единицы из порядка числа.

Существуют два основных способа представления данных в ЭВМ: с фиксированной и плавающей запятой.

В первом случае для сокращения длины разрядной сетки и упрощения обработки данных положение запятой может быть зафиксировано схемотехнически. При этом в слове данных сохраняются только две структурных компоненты: поле знака и поле цифр.

| ± | целая часть | дробная часть |
|---|---|---|

Диапазон представления чисел для этого формата:

$A_{max} = (2^k - 1) + (1 - 2^{-l})$,

где k – число разрядов целой, а l – дробной части числа (k + l = n).

В зависимости от размеров целой и дробной частей возможно следующее:

1) k = 0, l = n, $A_{max} = 1 - 2^{-n}$

| ± | , | 1 | 1 | . | . | . | | 1 |
|---|---|---|---|---|---|---|---|---|

2) k = n, l = 0, $A_{max} = 2^n - 1$

| ± | 1 | 1 | . | . | . | | 1 |
|---|---|---|---|---|---|---|---|

3) k = 0, l = n, $A_{min} = 2^{-n}$

| ± | , | 0 | 0 | . | . | . | | 0 |
|---|---|---|---|---|---|---|---|---|

4) k = n, l = 0, $A_{min} = 1$

| ± | 0 | 0 | . | . | . | | 0 |
|---|---|---|---|---|---|---|---|

Очевидно, что ограничение длины разрядной сетки приводит к ограничению диапазона хранимых чисел и потере точности их представления. Поэтому на практике широко используется и другая форма представления чисел – с плавающей запятой. В общем виде числа с плавающей запятой имеют следующий вид:

$A = \pm m_A r^{\pm pA}$,

где $m_A$ – мантисса, а pA – порядок числа A. Порядок (с учетом знака) показывает, на сколько разрядов и в какую сторону сдвинута запятая при замене формы записи числа с естественной на нормальную.

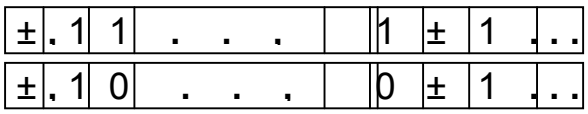
Например, $A_{10} = 239{,}745 = 0{,}239745 \cdot 10^3 = 239745 \cdot 10^{-3}$.

Наиболее распространено и удобно для представления в ЭВМ ограничение вида $r^{-1} \le |m_A| < 1$.

Форма представления чисел, для которых справедливо данное ограничение, называется нормализованной. Так как абсолютное значение мантиссы в этом случае лежит в диапазоне от $r^{-1}$ до $1 - r^{-n}$, где n – число разрядов мантиссы без знака, то положение разрядов числа в его машинном изображении непостоянно. Отсюда и название этой формы представления чисел – с плавающей запятой. Формат машинного изображения чисел с плавающей запятой должен включать знаковые поля (мантиссы и порядка), поле мантиссы и поле порядка числа. Он имеет следующий вид:

| ± | мантисса $m_A$ | ± | порядок p |
|---|---|---|---|

Для данного формата разрядной сетки можно записать следующий диапазон представления чисел:




$$A_{max} = (1 - 2^{-n})\, 2^{(2^q - 2)}$$

$$A_{min} = 2^{-2^q}$$

Для упрощения операций над порядками применяют представление чисел с плавающей запятой со смещенным порядком: $p' = p + N$, где N – целое положительное число (смещение), $N = \max(-p)$. Обычно $N = 2^k$, где k – число двоичных разрядов в поле цифр несмещенного порядка. В этом случае поле знака порядка избыточно, так как $p'$ всегда положительно. Такие смещенные порядки называют характеристиками. В зависимости от типа данных числа с плавающей запятой в памяти ЭВМ хранятся в одном из следующих трех форматов:


При выполнении арифметических операций над числами с плавающей запятой может получаться результат, выходящий за пределы диапазона представления чисел, при этом выход за правую границу диапазона принято называть переполнением порядка (получение очень большого числа), а выход за левую границу – исчезновением порядка (потерей порядка) – получение очень малого числа, близкого к нулю.

В стандарте IEEE крайние значения порядка (характеристики) зарезервированы и не используются для представления обычных чисел. Максимальное значение характеристики, представленное всеми единицами при положительном знаке числа, зарезервировано для представления значения $+\infty$ при нулевой мантиссе. При знаке минус число с максимальной характеристикой используется для представления $-\infty$ и неопределенности. Значение с минимальной характеристикой, равной нулю, зарезервировано для представления денормализованных чисел (положительных и отрицательных), а также для представления нуля (представляется всеми нулями), причем различают +0 и –0.

*Вопросы для самоконтроля*

1. Чему равен диапазон представления целых и дробных чисел с фиксированной запятой?

2. Чему равен диапазон представления чисел с плавающей запятой, если n – число разрядов мантиссы, q – число разрядов порядка?

3. В чем достоинства и недостатки формы с плавающей запятой?

4. Что такое нормализация и в чем состоит условие нормализации? Когда может возникнуть нарушение нормализации?

## Тема 4. Арифметические операции в двоичной системе счисления

1. Арифметические операции над числами с фиксированной запятой (точкой)

Сложение чисел с разными знаками в дополнительном и обратном кодах. Переполнение. Причины и признаки переполнения. Модифицированные коды и их возможности для обнаружения переполнения.

Умножение чисел в прямых кодах. Четыре машинных алгоритма умножения, их сравнительная оценка. Структурная схема операционных устройств умножения. Определение знака произведения.

Ускоренные методы умножения: с сохранением переносов, умножение на два (в прямых и дополнительных кодах) и четыре разряда одновременно, матричные методы умножения.

Машинные алгоритмы деления. Два основных алгоритма деления: с восстановлением и без восстановления остатка. Деление в дополнительных кодах.

2. Арифметические операции над числами с плавающей запятой

Сложение, умножение и деление чисел с плавающей запятой. Действия над порядками и мантиссами при выполнении этих операций. Необходимость выравнивания порядков при сложении чисел. Умножение и деление чисел с плавающей запятой.

[1, гл. 3, гл. 4, гл. 5; 2, гл. 3; 3, гл. 3]

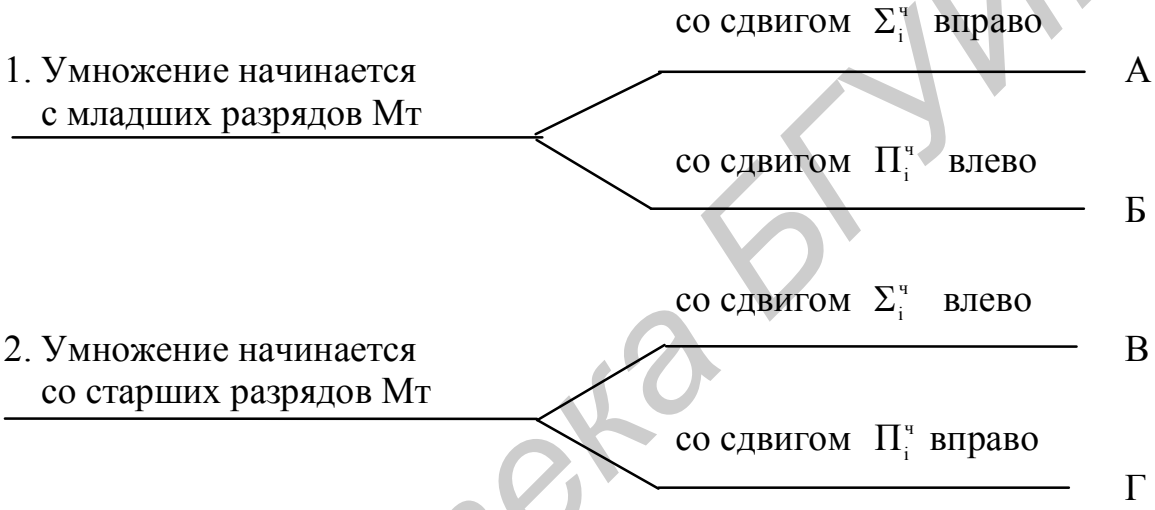
*Методические указания*

В данной теме необходимо изучить машинноориентированные алгоритмы выполнения арифметических операций.

Эта тема по объему достаточно обширна и потребует много времени на изучение. Необходимо обратить внимание на возможность использования только суммирующего устройства, если числа, вступающие в операцию сложения, кодировать соответствующим образом. Важным моментом процесса сложения является возможное переполнение, которое искажает результат как по величине, так и по знаку. Переполнение должно быть обнаружено. Признаков обнаружения переполнения существует несколько, включая использование модифицированных кодов.

Машинные методы умножения в прямых кодах отличаются от ручного умножения на бумаге только тем, что полное произведение получается путем

накапливания в сумматоре частичных сумм ($\sum_i^{ч}$), формируемых путем последовательного добавления в сумматор к предыдущей $\sum_{i-1}^{ч}$ очередного частичного произведения $П_i^{ч}$. Последняя частичная сумма $\sum_n^{ч}$ и является произведением. Сложением в каждом такте управляют разряды множителя: если в очередном разряде множителя содержится единица, то к частичной сумме добавляется частичное произведение (множимое). При этом в зависимости от метода умножения выполняется сдвиг либо множимого, либо частичной суммы. Наряду с этим умножение можно начинать как с младших, так и со старших разрядов множителя. Для умножения используют модули сомножителей. В связи с этим можно определить четыре алгоритма умножения. Назовем их алгоритмами А, Б, В и Г:




Сумматор может воспринимать только два слагаемых $\sum_{i-1}^{ч}$ и $П_i^{ч}$, тогда как при умножении на бумаге все n частичных произведений складываются одновременно. Знак произведения определяется отдельно путем сложения по mod2 знаков сомножителей.

Машинные методы умножения чисел в дополнительных кодах принципиально не отличаются от умножения чисел в прямых кодах. Различие заключается в том, что в зависимости от сочетания знаков множимого и множителя в конце умножения вводятся поправки. При умножении целых чисел участвует и знаковый разряд множителя, а в случае дробных чисел умножение производится только на дополнение, но поправки оказываются одинаковыми. При этом знак произведения формируется автоматически [3].

С целью уменьшения затрат времени на умножение используют логические методы ускорения операции умножения. Время умножения определяется числом разрядов множителя (числом тактов умножения) и длительностью такта. Соответственно сократить время можно за счет уменьшения времени сложения в каждом такте, что обеспечивает метод умножения с хранением переносов, или за счет уменьшения числа тактов умножения, что происходит при использова-

нии метода умножения на несколько разрядов множителя одновременно. Наиболее широкое распространение получили методы умножения одновременно на два и четыре разряда множителя как в прямых, так и в дополнительных кодах [3].

Деление − это процесс многократного вычитания делителя. Первое вычитание делителя (Дт) из делимого (Дм) называется пробным. Если полученный остаток меньше нуля, деление можно продолжать, записав в частное (Чт) ноль целых. Если остаток больше нуля, процесс деления чисел с фиксированной запятой (точкой) далее не производится (переполнение частного). Деление чисел с плавающей запятой продолжается до конца, после чего полученное Чт сдвигается вправо, а к порядку Чт добавляется «1». Если при выполнении деления получен нулевой i-й остаток, то деление прекращается, и в оставшиеся разряды частного записываются нули. Процесс деления заканчивается, если достигнута заданная точность частного или получен нулевой остаток.

Машинный способ деления с восстановлением остатка является естественным процессом получения цифр частного: если делитель не содержится в делимом, то вычитание первого из второго дает отрицательный остаток, что и послужит признаком записи нуля в частное. Если в результате вычитания остаток ($A_i$) оказался положительным, в частное записывается единица.

Деление без восстановления остатка может быть обосновано на основании анализа предыдущего метода и позволяет не только упростить схему управления делением, но и сократить время операции. Знак Чт при делении чисел в прямых кодах определяется так же, как и знак произведения при умножении.

Основой деления чисел в дополнительных кодах является метод деления без восстановления остатка. Однако здесь как при формировании цифр Чт, так и очередного добавления к $A_{i-1}$ Дт или $[-Дт]_{доп}$ производится сравнение знака остатка и знака делителя. Деление можно производить с любым сочетанием знаков Дм и Дт, при этом знак формируется автоматически в результате пробного вычитания. Следует заметить, что при делении чисел в дополнительном коде частное получаем в обратном коде. Чтобы получить результат в дополнительном коде, необходимо после получения нужного числа разрядов в случае отрицательного результата добавить единицу в младший разряд.

При выполнении арифметических операций над числами с плавающей запятой, кроме операций над мантиссами, производятся операции и над порядками.

При сложении чисел сначала определяется разность порядков, при этом обычно из первого вычитается второй с использованием дополнительного кода. После этого производится сдвиг мантиссы числа с меньшим порядком вправо на количество разрядов, равное разности порядков. При умножении чисел порядки сомножителей складываются, а при делении из порядка Дм вычитается порядок Дт.

*Вопросы для самоконтроля*

1. Сформулируйте причину и признаки, по которым можно обнаружить переполнение.

2. Укажите возможные действия машины в случае возникшего переполнения.

3. Сформулируйте машинные алгоритмы умножения и их техническую реализацию в виде простейших структурных схем.

4. Какие поправки и в каких случаях необходимо вводить при умножении чисел в дополнительных кодах?

5. Как определяется знак произведения при умножении чисел в прямых кодах?

6. Как производится умножение на пары разрядов 10 и 11 при умножении одновременно на два разряда множителя в прямых и дополнительных кодах?

7. В чем состоит принцип умножения на четыре разряда множителя одновременно?

8. В чем состоит принципиальное отличие деления двоичных чисел от деления чисел в других системах?

9. Назовите два основных способа деления.

10. Как определяется знак частного при делении чисел в прямых кодах?

11. Сформулируйте алгоритм деления в дополнительных кодах и изобразите его в виде граф-схемы.

12. Что требуется предварительно сделать, чтобы сложить два числа с плавающей запятой?

13. В чем отличие умножения и деления чисел с плавающей запятой от аналогичных операций над числами с фиксированной запятой.

## Тема 5. Двоично-десятичные (BCD) коды

Понятие двоично-десятичного, или BCD-кода. Требования к BCD-коду. Код прямого замещения 8421. Суммирование положительных чисел в BCD-кодах. Суммирование чисел с разными знаками в BCD-кодах. Одноразрядный двоично-десятичный сумматор. BCD-код с избытком 3. BCD-код с избытком 6 для одного из слагаемых.

*Методические указания*

Необходимо изучить способы представления чисел в BCD-кодах и выполнение арифметических операций над этими числами.

Общее число различных способов кодирования очень велико, однако для практических целей используются лишь немногие. Наиболее широкое применение нашел весомозначный код прямого замещения 8421. Этот код, взаимодополняемый до 15, усложняет введение поправок. Сложение чисел в BCD-

кодах производится в два этапа: первый этап – это собственно сложение чисел с выполнением межтетрадных переносов (переносов из одного десятичного разряда в другой), на втором этапе производится коррекция результата в тех тетрадах (разрядах), где это необходимо, при этом переносы между тетрадами не производятся, таким образом коррекция результата по времени короче, чем сложение. При сложении чисел с одинаковыми знаками добавление 6 (0110) к тетраде производится в тех случаях, когда сумма двух цифр равна или больше десяти. Если сумма, кроме того, больше 16, то добавление 6 компенсирует покинувшие тетраду вместе с шестнадцатеричным переносом шесть единиц. Если шестнадцатеричного переноса не было, а сумма равна или больше десяти, то добавлением 6 удаляются из тетрады десять единиц, а передача десятка в следующую тетраду осуществляется по наличию единиц в разрядах тетрады суммы: с весом 8 и 4 или с весом 8 и 2. В связи с этим необходимо рассмотреть схему одноразрядного десятичного сумматора и изучить ее функционирование при сложении двух тетрад.

Числа с разными знаками складываются в обратном коде. При сложении чисел с разными знаками следует обратить внимание на наличие или отсутствие переноса из тетрады суммы. Если при положительном знаке результата перенос из тетрады отсутствует, значит в эту тетраду произошел заем, и для коррекции результата необходимо вычесть 6 (т. е. добавить в тетраду дополнение 6, равное 1010). Если же при отрицательном знаке результата есть перенос из тетрады, значит, в эту тетраду произошел заем, и для коррекции результата необходимо добавить 6 (т. е. добавить в тетраду 0110). Это объясняется тем, что автоматически производится заем, равный 16 единицам, а не 10, как это необходимо для десятичных чисел, представленных в BCD-кодах.

Аналогичные случаи следует рассмотреть и для коррекции суммы в кодах с избытком 3 и 6.

*Вопросы для самоконтроля*

1. В чем состоит основной недостаток кода 8421?
2. Сформулируйте правила введения поправок в тетрады суммы при сложении чисел с одинаковыми и разными знаками.
3. Изобразите схему одноразрядного десятичного сумматора и изучите его работу на примерах сложения тетрад, сумма которых не больше 9; больше 9, но не больше 15; больше 15.
4. Достоинства кода с избытком три  по сравнению с кодом 8421?

**Тема 6. Знакоразрядные системы счисления и СОК**

Системы счисления с четным и нечетным основанием, симметричные и кососимметричные. Избыточные системы. Суммирование в избыточных системах. Положительный и отрицательный переносы. Распространение переноса.

Система СОК. Система остаточных классов. Выбор оснований; вычисление остатков (вычетов) по основаниям. Сложение, вычитание, умножение, перевод чисел из десятичной системы в СОК и обратно.

[1, гл. 7; 6]

*Методические указания*

Необходимо рассмотреть принципы представления чисел в знакоразрядных системах счисления и СОК, выполнение арифметических операций над этими числами.

Следует обратить внимание на то, что обычные системы счисления только с положительными весами цифр называются смещенными. Однако можно построить системы счисления как с положительными, так и с отрицательными весами разрядов. Интерес представляют избыточные системы с одной, двумя и более цифрами, что позволяет ограничить распространение переноса. СОК – система непозиционная. Любой остаток, вычисленный по одному из принятых оснований, характеризует все число. Поэтому при сложении чисел распространение переноса ограничивается в пределах суммирования остатков, представленных двоичными эквивалентами. При переводе десятичных чисел в СОК вычисленные заранее остатки по основаниям разрядных цифр и их весов хранятся в памяти ЭВМ. В памяти ЭВМ находятся и базисы $B_1 ... B_n$ для перевода чисел из СОК в десятичную систему.

*Вопросы для самоконтроля*

1. Почему не требуется кодирование в дополнительном и обратном кодах чисел, представленных в знакоразрядных системах счисления?
2. Как производится сложение чисел с разными знаками?
3. В чем преимущество избыточных систем?
4. Какие числа могут быть основаниями в системе СОК?
5. Какие достоинства у системы СОК?

**Тема 7. Логические основы цифровых автоматов**

Логические переменные (аргументы) и переключательные функции (ПФ) как простые и сложные высказывания. Задание ПФ: таблица истинности, логическое выражение, логическая схема. Связь между числом переменных и количеством наборов, на котором ПФ принимает значение нуля или единицы. ПФ двух аргументов, из них наиболее широко используемые: И, ИЛИ, НЕ, И–НЕ, ИЛИ–НЕ, сложение по mod2 и др.

Основные соотношения (правила) булевой алгебры. Две основные формы представления ПФ: дизъюнктивная и конъюнктивная, и запись ПФ в этих формах по таблице истинности. Элементарная конъюнкция, ее ранг; совершенные формы ПФ. Минимизация ПФ методом Квайна, Квайна − Мак-Класки, картами

Карно − Вейча: задача минимизации; понятие покрытия ПФ; соседние наборы, кодовое расстояние, простая импликанта, обязательная простая импликанта. Минимальная форма ПФ, тупиковые формы, скобочная форма ПФ. Минимизация неполностью определенных ПФ. Функционально полные наборы ПФ; выражение логических связей И, ИЛИ, НЕ и их реализация функционально полным набором И–НЕ (ИЛИ–НЕ).

Упрощение логических выражений путем использования законов булевой алгебры при синтезе одноразрядного двоичного сумматора (вычитателя); полусумматора; сумматора на двух полусумматорах и других схем.

[1, гл. 9, гл. 10; 2, гл. 2]

*Методические указания*

Необходимо изучить основные понятия алгебры логики, их использование в описании и синтезе дискретных устройств, методы минимизации логических функций.

Для формального описания устройств вычислительной техники при их анализе и синтезе используется аппарат алгебры логики. Алгебру логики называют также булевой алгеброй. Основными понятиями алгебры логики являются двоичные переменные и переключательные (булевы) функции.

Двоичные переменные могут принимать только два значения: 0 (ложь) и 1 (истина), и обозначаются символами $x_1$, $x_2$, …, $x_n$. Двоичные (логические, булевы) переменные являются аргументами булевых (переключательных) функций (ПФ).

Основными понятиями, лежащими в основе представления булевых функций в различных формах, являются понятия элементарной конъюнкции и элементарной дизъюнкции.

Элементарной конъюнкцией называется логическое произведение любого конечного числа различных между собой булевых переменных, взятых со знаком инверсии или без него.

Например, логические выражения вида $x_1\bar{x}_2\bar{x}_3$, $\bar{x}_1x_4$, $x_1x_2x_4$ являются элементарными конъюнкциями, а выражения вида $\overline{x_1x_2x_3}$, $\overline{\bar{x}_1x_4}$, $\overline{x_1x_2x_4}$ не являются элементарными конъюнкциями.

Элементарной дизъюнкцией называется логическая сумма любого конечного числа различных между собой булевых переменных, взятых со знаком инверсии или без него.

Примером логического выражения, являющегося элементарной дизъюнкцией, могут служить $\overline{x_1 + \bar{x}_2 + \bar{x}_3}$, $x_1 + \overline{\bar{x}_4}$, $x_1 + x_2 + x_4$, а выражения вида $\overline{x_1 + x_2 + x_3}$, $\overline{x_1 + x_4}$, $\overline{x_1 + x_2} + x_4$ не являются элементарными дизъюнкциями.

Дизъюнктивной нормальной формой (ДНФ) булевой функции называется дизъюнкция конечного числа элементарных конъюнкций:

$f_{ДНФ} = x_1\bar{x}_2\bar{x}_3 + x_1x_4 + x_2x_3\bar{x}_4 + x_1\bar{x}_2x_3$.

Число переменных, входящих в элементарную конъюнкцию, определяет *ранг* этой конъюнкции.

Совершенной ДНФ (СДНФ) логической функции от n аргументов называется такая ДНФ, в которой все конъюнкции имеют ранг n. СДНФ записывается по таблице истинности согласно правилу: для каждого набора переменных, на котором булева функция принимает единичное значение, записывается конъюнкция ранга n, и все эти конъюнкции объединяются дизъюнктивно; переменная имеет знак инверсии, если на соответствующем наборе имеет нулевое значение.

$f_{СДНФ} = x_1x_2\overline{x_3} + \overline{x_1}\overline{x_2}x_3 + \overline{x_1}x_2\overline{x_3} + x_1x_2x_3.$

В общем виде это можно записать следующим образом:

$$f(x_1,x_2,...,x_n) = Vf(y_1,y_2,...,y_n)\,\&\,x_1^{\sigma_1}\,\&\,x_2^{\sigma_2}...\&\,x_n^{\sigma_{1n}},$$

где

$$x^{\sigma} = \begin{cases} \overline{x}, \text{ если } \sigma = 0, \\ x, \text{ если } \sigma = 1. \end{cases}$$

Элементарные конъюнкции, образующие СДНФ, называют также конституентами (составляющими) единицы (минтерм), так как они соответствуют наборам, при которых функция принимает значение, равное единице. Построение СДНФ по таблице истинности называют составлением булевой функции по условиям истинности.

Конъюнктивной нормальной формой (КНФ) булевой функции называется конъюнкция конечного числа элементарных дизъюнкций.

$f_{КНФ} = (x_1 + \overline{x_2} + \overline{x_3})(\overline{x_1} + \overline{x_4})(x_2 + \overline{x_3} + x_4)(x_1 + x_2 + x_3).$

Совершенной КНФ (СКНФ) логической функции от n аргументов называется такая КНФ, в которой все дизъюнкции имеют ранг n. СКНФ записывается по таблице истинности согласно правилу: для каждого набора переменных, на котором булева функция принимает нулевое значение, записывается дизъюнкция ранга n, и все эти дизъюнкции объединяются конъюнктивно; переменная имеет знак инверсии, если на соответствующем наборе имеет единичное значение:

$f_{СКНФ} = (x_1 + x_2 + \overline{x_3})(x_1 + \overline{x_2} + \overline{x_3})(\overline{x_1} + x_2 + x_3)(x_1 + x_2 + x_3).$

Элементарные дизъюнкции, образующие СКНФ, называют конституентами (составляющими) нуля (макстерм), так как они соответствуют наборам, при которых функция принимает нулевое значение. Построение СКНФ по таблице истинности называют составлением булевой функции по условиям ложности.

Чтобы получить совершенную дизъюнктивную нормальную форму, следует взять все наборы, на которых значение функции равно единице, и записать для каждого из них конъюнкцию переменных и их отрицаний. Если в наборе значение переменной равно нулю, то переменную следует взять с отрицанием, если единице – без отрицания. Из получившихся конъюнкций нужно построить дизъюнкцию.

Чтобы получить совершенную конъюнктивную нормальную форму, следует взять все наборы, на которых значение функции равно нулю, и записать для каждо-

го из них дизъюнкцию переменных и их отрицаний. Если в наборе значение переменной равно нулю, то переменную следует взять без отрицания, если единице – с отрицанием. Из получившихся дизъюнкций нужно построить конъюнкцию.

Основные отношения и законы алгебры логики, такие, как переместительный, сочетательный, распределительный законы для конъюнкции и дизъюнкции, правило склеивания и закон де Моргана, следует запомнить. Необходимо уметь записать ПФ в ДНФ, СДНФ, КНФ и СКНФ по таблице истинности и оценить сложность реализации логической схемы, например по числу входов всех элементов схемы.

При проектировании цифровых автоматов широко используются методы минимизации булевых функций, позволяющие получать экономичные схемы. Общая задача минимизации булевых функций может быть сформулирована следующим образом: найти аналитическое выражение заданной булевой функции в форме, содержащей минимально возможное число букв. Следует отметить, что в общей постановке данная задача пока не решена, однако достаточно хорошо исследована в классе дизъюнктивно-конъюнктивных форм.

Минимальной дизъюнктивной нормальной формой (МДНФ) булевой функции называется ДНФ, содержащая наименьшее число букв (по отношению ко всем другим ДНФ, представляющим заданную булеву функцию).

Для минимизации ПФ, заданной в СДНФ или СКНФ, необходимо воспользоваться расчетно-табличным алгоритмом Квайна или Квайна − Мак-Класки. Основной операцией на первом этапе является операция склеивания, применение которой приводит к нахождению простых импликант. На втором этапе этого алгоритма из общего множества импликант выявляются сначала обязательные простые импликанты, а затем и полное минимальное покрытие. Более удобным и наглядным для минимизации булевых функций от небольшого числа переменных является графический способ минимизации с помощью карт (диаграмм) Карно (Вейча). Карта Вейча представляет собой развертку n-мерного куба на плоскости. При этом вершины куба представляются клетками карты, каждой из которых поставлена в соответствие конститутиента единицы или нуля. Переменные, обозначающие клетки диаграммы, расставляются таким образом, чтобы наборы, записанные в двух смежных клетках, имели кодовое расстояние, равное единице. Поскольку такие наборы располагаются в смежных клетках, они получили название *соседних наборов*. В клетку карты, соответствующую конституенте единицы, заносится 1, иначе − 0. Таким образом, для минимизации функции она должна быть представлена в форме СДНФ. Минимизация булевой функции с использованием карт в дизъюнктивной (конъюнктивной) форме заключается в объединении единичных (нулевых) клеток в контуры, каждому такому контуру соответствует простая импликанта.

Можно сформулировать следующие правила минимизации:

- количество клеток карты в одном контуре должно быть равно $2^n$;
- для контура, содержащего $2^n$ клеток, должно быть **n** осей симметрии;

• количество контуров должно быть минимально;

• число единиц в контуре должно быть максимально;

• контуры могут пересекаться, т. е. некоторая клетка может входить в несколько контуров.

Если выбраны самые большие контуры и использовано по возможности меньшее их число, то будет получена самая простая дизъюнктивная нормальная форма. Дальнейшее упрощение можно получить за счет выполнения скобочных преобразований. Выражению с меньшим числом вхождений букв соответствует схема, имеющая меньшее число входов элементов, так что упрощение функций ведет к упрощению реализующих их схем.

Существуют системы булевых функций, с помощью которых можно аналитически представить любую сколь угодно сложную булеву функцию. Проектирование цифровых автоматов основано на знании таких систем булевых функций. Последнее особенно важно для определения набора элементарных логических схем, из которых можно построить произвольный цифровой автомат. Проблема функциональной полноты является центральной проблемой функциональных построений в алгебре логики.

Функционально полной системой булевых функций (ФПСБФ) называется совокупность таких булевых функций ($f_1$, $f_2$, ..., $f_k$), посредством которых можно записать произвольную булеву функцию f.

Необходимо обратить внимание на то, что набор логических элементов (функций) И, ИЛИ, НЕ является функционально полным, но избыточным. Обязательным логическим элементом в таком наборе является элемент НЕ. Чтобы сделать набор функционально полным, к элементу НЕ достаточно добавить И или ИЛИ. Для получения навыков использования законов (правил) булевой алгебры целесообразно синтезировать одноразрядный двоичный сумматор и другие широко применяемые схемы. Синтез начинается со словесного описания работы схемы; затем составляется таблица истинности, по которой записываются логические выражения ПФ. Производится их упрощение путем применения правил булевой алгебры, и, наконец, строится схема.

*Вопросы для самоконтроля*

1.Что такое переключательная функция?

2. Перечислите способы задания ПФ.

3. Приведите примеры простых импликант и назовите их ранг.

4. Перечислите основные законы булевой алгебры.

5. В чем достоинства карт Вейча и Карно?

6. Почему набор И, ИЛИ, НЕ является функционально полным и избыточным?

7. Назовите некоторые другие функционально полные наборы.

8. С чего начинается синтез логической схемы?

9. В чем особенности анализа и синтеза последовательных схем по сравнению с комбинационными схемами?

## Тема 8. Основы теории конечных автоматов

Понятия: абстрактный и структурный автоматы. Два класса автоматов: Мили и Мура. Способы задания абстрактных автоматов. Метод синтеза автоматов по таблицам переходов/выходов. Понятия: микропрограмма, микропрограммный автомат (МПА). Принцип микропрограммного управления. Явления: гонки и риск сбоя, способы их устранения. Матричные программируемые логические устройства (ПЛУ). Граф-схема алгоритма (ГСА) и синтез МПА по ГСА.

[8; 9]

*Методические указания*

Необходимо изучить понятия абстрактного и структурного автоматов, микропрограммного автомата, способы построения структурного автомата.

Наряду с комбинационными схемами в цифровой технике широкое распространение получили последовательностные схемы, или автоматы, или, иначе, комбинационные схемы, объединенные с элементами памяти.

Под термином *автомат* можно понимать некоторое реально существующее устройство, функционирующее на основании как сигналов о состоянии внешней среды, так и внутренних сигналов о состоянии самого автомата. В этом плане ЭВМ может быть рассмотрена как цифровой автомат. Под цифровым автоматом понимается устройство, предназначенное для преобразования цифровой информации. Однако под термином *автомат* можно понимать и математическую модель некоторого устройства.

Абстрактный автомат – математическая модель цифрового устройства, определяемая вектором $S = (A, Z, W, \delta, \lambda, a_1)$, где A – множество внутренних состояний автомата, Z – входной, W – выходной абстрактный алфавит, $\delta$ и $\lambda$ − функции выходов и переходов, $a_1$ – начальное состояние. По виду функций переходов и выходов автоматы подразделяются на два класса: Мили ($a(t + 1) = \delta(a(t), z(t))$; $w(t) = \lambda(a(t), z(t))$) и Мура ($a(t + 1) = \delta(a(t), z(t))$; $w(t) = \lambda(a(t))$).

Существуют два основных способа задания абстрактных автоматов: таблицами переходов/выходов и ориентированным графом. В отличие от абстрактных автоматов, имеющих один вход (на который подаются абстрактные входные слова из множества Z) и один выход (на котором формируются абстрактные выходные слова из множества W), структурный автомат имеет множество двоичных входов и выходов, на которые подаются входные и формируются выходные двоичные сигналы. В литературе рассмотрен канонический метод синтеза структурного автомата, реализующего заданный таблицами переходов/выходов абстрактный автомат. Абстрактный автомат в данном случае является математической моделью устройства, которое необходимо реализовать в

виде реальной логической схемы. В качестве элементов памяти автомата используются простейшие схемы, предназначенные для приема, хранения и передачи одного бита информации − триггеры. Триггер имеет один или более входов и два выхода (прямой и инверсный). Выходные сигналы триггера зависят только от его состояния и изменяются только при смене состояния триггера. Таким образом, триггеры являются элементарными автоматами Мура (элементарными, так как они имеют только два устойчивых состояния).

Триггеры можно классифицировать по следующим признакам:

1) по способу записи информации: несинхронизируемые (асинхронные) и синхронизируемые (синхронные) триггеры. В асинхронных триггерах запись информации происходит под действием информационных сигналов, в синхронных, кроме информационных, на вход должны быть поданы разрешающие сигналы;

2) по способу синхронизации: синхронные триггеры со статическим управлением записью, синхронные двухступенчатые триггеры, синхронные триггеры с динамическим управлением записью;

3) по способу организации логических связей: триггеры с раздельной установкой состояния (RS-триггеры), триггеры со счетным входом (Т-триггеры), универсальные триггеры с раздельной установкой состояний (JK-триггеры), триггеры с приемом информации по одному входу (D-триггеры), комбинированные триггеры (RST-, JKRS-, DRS-триггеры и т. д.), триггеры со сложной входной логикой.

При построении дискретных схем вычислительной техники принято подразделять их на операционное устройство и устройство управления (иначе операционный и управляющий автоматы). В основу синтеза управляющих автоматов положен принцип микропрограммного управления. Рассмотрим некоторые понятия. Элементарное действие, выполняемое за один такт автоматного времени (за один такт работы автомата), называется *микрооперацией*. Условия, влияющие на порядок выполнения автоматом микроопераций, называются *логическими условиями*. Проверка значений логических условий в каждом такте работы автомата позволяет определить группу выполняемых микроопераций. Совокупность операций, выполняемых за один такт, называется *микрокомандой*.

Принцип, согласно которому алгоритм работы некоторого устройства описывается в перечисленных выше терминах, называется принципом микропрограммного управления. Конечный автомат, алгоритм работы которого может быть описан на основе принципа микропрограммного управления, называется микропрограммным автоматом (МПА). Для записи микропрограмм в компактной форме используются специализированные языки. Одним из способов графического представления микропрограммы является граф-схема алгоритма (ГСА). ГСА представляет собой ориентированный связный граф. ГСА может содержать вершины четырех типов: начальную, операторную, условную и конечную.

ГСА должна удовлетворять следующим основным требованиям:

– в ГСА должна быть одна начальная и одна конечная вершины;

– входы и выходы вершин соединяются с помощью дуг;

– каждая вершина должна лежать на одном из путей следования из начальной вершины в конечную;

– один из выходов условной вершины может соединяться с ее входом;

– в каждой условной вершине записывается одно из логических условий $x_i$ (допускается запись одинаковых условий в различных вершинах);

– в каждой операторной вершине записывается микрокоманда (допускается пустая микрокроманда и повтор микрокоманды в различных вершинах).

Нарушение функционирования автомата может быть вызвано явлениями, получившими название «гонки» и «риск сбоя».

Гонки возникают из-за неодновременного срабатывания элементов памяти автомата вследствие разброса во времени переключения триггеров, а также из-за различия по времени поступления сигналов на входы триггеров. Таким образом, если на некотором переходе в автомате одновременно изменяют свое состояние несколько элементов памяти, то между ними возникает «состязание». Если из промежуточного состояния автомат в конечном счете переходит в требуемое состояние, то «состязания» называются некритическими, если же переход происходит в ложное состояние, то «состязания» называются критическими, или гонками. Кроме того, в промежуточном состоянии может быть сформирован кратковременный неверный выходной сигнал. Это явление называется риском сбоя. Существуют программные и аппаратные способы борьбы с этими явлениями.

*Вопросы для самоконтроля*

1. Какой автомат называется абстрактным, структурным?
2. В чем различия автоматов Мили и Мура, способов их задания?
3. В чем состоит метод синтеза автоматов по таблицам переходов/выходов?
4. Что такое принцип микропрограммного управления?
5. Метод синтеза МПА по ГСА (его основные этапы).
6. Явление риска сбоя при работе МПА и методы борьбы с ним.
7. Критические состязания (гонки) и методы борьбы с ними.

**Контрольная работа (Часть 1)**

Эта часть работы предусматривает изучение раздела курса, касающегося арифметических основ вычислительной техники. Перед тем как приступить к выполнению работы, необходимо изучить разделы, посвященные системам счисления, кодированию чисел, изучить машинно-ориентированные алгоритмы выполнения арифметических операций, в частности, сложения чисел с пла-

вающей запятой, умножения чисел в прямых и дополнительных кодах (при выполнении работы следует использовать все четыре алгоритма умножения), методы ускорения умножения, методы деления чисел, методы сложения чисел в BCD-кодах.

1. Для выполнения арифметических операций выбрать из таблицы два десятичных числа (назовем их A и B), расположенных на пересечении первой цифры номера варианта (указывается преподавателем) по горизонтали и второй – последняя цифра номера зачетки по вертикали. Например, варианту 37 соответствуют A = 25,97 и B = 95,18.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 47,64<br>16,46 | 74,12<br>28,95 | 19,25<br>83,17 | 15,27<br>93,17 | 99,15<br>23,76 | 24,66<br>86,79 | 28,31<br>86,17 | 11,47<br>89,25 | 94,58<br>14,43 | 92,43<br>17,83 |
| 1 | 19,76<br>98,41 | 95,79<br>18,15 | 16,72<br>93,78 | 99,77<br>15,18 | 19,38<br>87,15 | 18,69<br>97,17 | 21,63<br>79,09 | 17,59<br>71,55 | 17,28<br>91,19 | 11,96<br>84,25 |
| 2 | 13,39<br>77,13 | 97,48<br>12,68 | 87,13<br>10,49 | 79,83<br>13,77 | 29,17<br>88,80 | 77,11<br>15,28 | 96,17<br>22,89 | 18,95<br>94,16 | 17,73<br>71,19 | 95,69<br>11,56 |
| 3 | 11,54<br>84,77 | 83,19<br>24,65 | 19,17<br>81,39 | 19,54<br>97,15 | 18,99<br>92,59 | 12,83<br>89,17 | 13,27<br>89,22 | 25,97<br>95,18 | 14,88<br>94,67 | 11,25<br>89,16 |
| 4 | 13,27<br>94,14 | 14,87<br>83,41 | 87,17<br>22,86 | 72,15<br>19,49 | 13,65<br>92,18 | 83,23<br>19,71 | 25,73<br>83,12 | 83,56<br>21,15 | 27,92<br>95,28 | 10,88<br>86,14 |
| 5 | 12,85<br>99,33 | 21,35<br>89,13 | 11,77<br>98,18 | 25,74<br>92,15 | 12,92<br>87,39 | 84,17<br>21,81 | 91,14<br>15,86 | 19,24<br>89,12 | 71,97<br>16,79 | 89,22<br>11,79 |

2. Числа A и B перевести делением на основание системы счисления в 12-разрядные двоичные, которые будут состоять из целой и дробной частей. Аналогичный перевод произвести в системы счисления с основаниями 4, 8 и 16 и получить соответственно 6, 4 и 3-разрядные числа. После этого, заменив цифры чисел в этих системах счисления соответственно двоичными диадами, триадами и тетрадами, удостовериться, что в каждом случае получены двоичные изображения десятичных чисел A и B, ограниченных числом разрядов дробной части.

3. Представить двоичные числа A и B в форме с плавающей запятой.

4. Просуммировать эти числа в дополнительном и обратном кодах для всех случаев сочетания знаков слагаемых ($A > 0$; $B > 0$), ($A < 0$; $B > 0$), ($A > 0$; $B < 0$), ($A < 0$; $B < 0$). Обратить внимание на случаи переполнения и денормализации результата, для которых порядок суммы должен быть изменен после нормализации результата.

5. Перемножить двоичные числа A и B, ограниченные старшими шестью разрядами. Перемножение производить в дополнительных кодах для всех случаев сочетания знаков, как в п. 4.

6. Над двоичными числами A и B из п.5 произвести операцию деления, приняв за делимое меньшее из двух чисел. Деление произвести в дополнительных кодах для всех случаев сочетания знаков.

7. Перемножить двоичные числа A и B из п. 5, используя метод ускоренного умножения с сохранением переносов. Числа умножать в прямом коде.

8. Перемножить двоичные числа A и B из п. 5, используя метод ускоренного умножения на два разряда множителя одновременно. Перемножение производить в дополнительных кодах для случаев сочетания знаков ($A > 0$; $B > 0$), ($A < 0$; $B < 0$).

9. Выполнить сложение исходных десятичных чисел в BCD-кодах для случаев сочетания знаков ($A > 0$; $B > 0$), ($A < 0$; $B > 0$), ($A > 0$; $B < 0$).

10. При выполнении перечисленных выше арифметических операций производить контроль правильности получаемого результата, переводя его в десятичную систему счисления и сравнивая с результатом действия в десятичной системе счисления.

**Контрольная работа (Часть 2)**

Перед тем как приступить к выполнению этой части работы, необходимо изучить разделы курса, касающиеся логических основ вычислительной техники и основ теории автоматов, в частности, принцип микропрограммного управления, понятия абстрактного и структурного автомата, модели автомата Мили и Мура, понятие граф-схемы алгоритма (ГСА), понятие микропрограммного автомата (МПА), способы построения МПА, заданного ГСА, методы минимизации булевых функций.

1. Составить ГСА (примерно на 30 операторных и условных вершин).

2. Отметить ГСА метками МПА Мура.

3. Построить по отмеченной ГСА граф МПА, структурную таблицу МПА и реализовать схему МПА на логических элементах.

4. Выполнить пп. 2 и 3 для МПА Мили.

## СОДЕРЖАНИЕ

Св. план 2009, поз. 61

Учебное издание

# АРИФМЕТИЧЕСКИЕ И ЛОГИЧЕСКИЕ ОСНОВЫ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

Методические указания и контрольные задания
для студентов специальности 1-40 02 01
«Вычислительные машины системы и сети»
заочной формы обучения

Составители:
**Луцик** Юрий Александрович
**Лукьянова** Ирина Викторовна
**Бушкевич** Алексей Владимирович

Редактор Т. Н. Крюкова
Корректор Л. А. Шичко
Компьютерная верстка Е. С. Чайковская

---

Подписано в печать 10.06.2010. Формат 60x84 1/16. Бумага офсетная.
Гарнитура «Таймс». Отпечатано на ризографе. Усл. печ. л. 1,74.
Уч.-изд. л. 1,5. Тираж 100 экз. Заказ 27.

---

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0494371 от 16.03.2009. ЛП №02330/0494175 от 03.04.2009.
220013, Минск, П. Бровки, 6