ВВЕДЕНИЕ

Целью настоящей работы является исследование и разработка методов логического проектирования преобразователей дискретной информации. В этой фразе почти каждое слово требует пояснений. Начнем с определения класса преобразователей дискретной информации. Вначале это сделаем на интуитивном уровне, а затем попытаемся уточнить.

Начнем с примеров. Мы знакомы с классом объектов, которые называем автоматами, или автоматическими устройствами. Это такие устройства, которые выполняют какие-то действия автоматически, без участия человека. Есть автоматы сравнительно примитивные. Множество таких автоматов окружают нас в обыденной жизни. К их числу можно отнести автоматы, управляющие светофорами, продающие газеты, газированную воду, и другие. Человек строит все более сложные автоматы. К сложным автоматам можно отнести автопилоты, управляющие полетом самолетов, обеспечивающие ориентацию космических кораблей, их стыковку и расстыковку. К числу сложных автоматов можно отнести и различные системы автоматического проектирования.

В настоящее время во многих случаях при реализации каких-то сложных процессов автоматы работают бок о бок с человеком. Некоторые функции этого процесса берет на себя человек, а другие — поручает автоматам. Системы автомат — человек называют автоматизированными. В автоматизированных системах проектирования проектирование, то есть принятие решений, ведет человек. Но связанные с проектированием расчеты, различные преобразования проектируемого объекта выполняются автоматически. Автомат в значительной степени облегчает труд проектировщика. Но можно представить себе и такие системы, которые состоят из одних только автоматов. Их называют автоматическими. Автоматическая система проектирования весь процесс проектирования берет на себя. Человек, если и участвует в контуре проектирования, то только в качестве источника информации для автомата. Кроме человека, автоматическая система проектирования может использовать и другие источники информации.

Попытаемся представить себе, как устроен автомат. Нас, конечно, интересует не конкретное конструктивное исполнение того или иного автомата. Можно ли с конструктивной точки зрения сравнивать автомат, продающий газированную воду, с автоматической системой проектирования? Конечно, нет. Но все же для всех автоматов мы можем выделить общие части. Первая из таких частей называется оболочкой. Задачей оболочки является изоляция автомата от окружающей его среды. Оболочка защищает автомат не только от агрессивных воздействий внешней среды, но и в информационном смысле. Агрессивные воздействия могли бы разрушить автомат, а информационные помехи не дают автомату правильно функционировать. Оболочки могут быть реализованы по-разному. Это может быть кожух аппарата. В программных автоматах это может быть защита памяти с помощью специальных ключей. И так далее. Конкретный вид оболочки зависит от того, какой это автомат и в какой среде он работает.

Автомат должен получать информацию из внешней среды. Для этой цели используются специальные чувствительные к тем или иным параметрам внешней среды устройства, которые можно назвать рецепторами. Иногда их также называют датчиками.

Автомат должен иметь средства воздействия на внешнюю среду. Их называют эффекторами. Эффекторы тоже могут быть реализованы по-разному в зависимости от назначения автомата. Это может быть ковш автоматического экскаватора, графопостроитель электронной вычислительной машины и другие.

Рецепторы и эффекторы автомата находятся за оболочкой. А что же мы имеем внутри? Воздействия внешней среды на рецепторы, как правило, носят непрерывный характер. Например, температура изменяется постепенно во времени. Для логической же обработки более удобной является дискретная или цифровая форма представления информации. Любой реальный непрерывный сигнал, имеющий ограниченный спектр, можно представить в виде дискретных отсчетов. Теорема Котельникова устанавливает связь между частотой таких отсчетов и шириной спектра непрерывного сигнала. Дискретные отсчеты могут быть квантованы по уровням. Квантованные дискретные сигналы могут быть представлены в виде кодовых комбинаций в некотором цифровом алфавите. Преобразователь, осуществляющий дискретные отсчеты непрерывного сигнала, квантование по уровням и кодирование, называется аналого-цифровым преобразователем. Такой преобразователь является составной частью большинства достаточно сложных по логике работы автоматов. Действия эффекторов тоже часто являются непрерывными. Поэтому после обработки информации требуется преобразование ее в аналоговую форму. Такой преобразователь называется цифро-аналоговым. Между аналого-цифровым и цифро-аналоговым преобразователями стоит преобразователь дискретной информации или цифровой автомат. Понятия «преобразователь дискретной информации» и «цифровой автомат» будем считать синонимами. Однако, понятие «цифровой автомат» ассоциируется с общепринятым понятием «конечный автомат», а, как будет показано позднее, класс «преобразователей дискретной информации» шире класса «конечных автоматов». Поэтому мы будем пользоваться понятием «преобразователь дискретной информации».

Таким образом, структуру автомата можно представить в следующей форме:

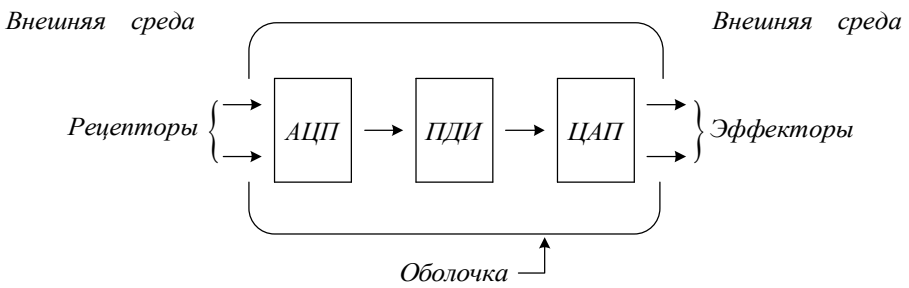


Рис. В.1

Здесь АЦП — это аналого-цифровой преобразователь, ПДИ — преобразователь дискретной информации или цифровой автомат, ЦАП — цифро-аналоговый преобразователь.

Рассмотрим электронно-вычислительную машину, как пример автомата. Рецепторами и эффекторами являются различные внешние устройства. Клавиатура терминала, например, воспринимает и реагирует на давление пальцев оператора на клавиши. На выходе терминала мы получаем кодовые комбинации букв. Значит, буквенно-цифровой терминал, кроме того, что он является рецептором, он же является и аналого-цифровым преобразователем. Этот терминал может высту-

пать и в качестве эффектора. Мы в данном случае имеем совмещение функций в одном устройстве. Примером эффектора может служить графопостроитель или АЦПУ (алфавитно-цифровое печатающее устройство). Преобразователем дискретной информации в электронно-вычислительной машине является процессор и память. Различные устройства вычислительной машины имеют свои оболочки. Это корпуса устройств, различные экраны, защищающие от внешних электростатических и электромагнитных полей.

Как уже было показано на примере электронно-вычислительной машины, в некоторых случаях мы имеем совмещение функций блоков схемы (рис. В.1). Рецепторы одновременно могут быть и эффекторами, АЦП может входить в состав рецептора. В некоторых случаях оболочка может быть рецептором. Живые существа, например, кожей ощущают внешнюю среду.

Некоторые части схемы рисунка В.1 иногда в автомате могут вообще отсутствовать. Программу, исполняемую электронно-вычислительной машиной, мы можем рассматривать как цифровой автомат. Преобразователем дискретной информации является сама программа и процессор, исполняющий ее. Все исходные и выходные данные находятся во внешней среде. Эта внешняя среда является цифровой и поэтому АЦП и ЦАП не нужны.

Теперь рассмотрим взаимодействие автомата с внешней средой. Эта среда не является аморфной, а состоит из каких-то объектов. С некоторыми из них автомат связан своими рецепторами, а с другими нет. Объекты внешней среды, с которыми автомат связан своими рецепторами, называются источниками информации. Связь автомата с источниками информации является жесткой. Электрическая лампочка, например, освещающая машинный зал, не может быть источником информации для трансмиттера, с помощью которого осуществляется ввод информации в ЭВМ с перфоленты. Даже если трансмиттер является оптическим, при его проектировании мы делаем все, чтобы посторонний свет не влиял на его работу. Эффекторы автомата тоже жестко связаны лишь с некоторыми объектами внешней среды, которые будем называть объектами воздействия. Некоторые объекты воздействия иногда одновременно могут быть и источниками информации для автомата. Тогда замыкается цепь обратной связи через внешнюю среду. Такая обратная связь не приносит каких-то качественных изменений, и автомат по-прежнему остается автоматом.

Мы получим качественное изменение, если связь автомата с внешней средой сделаем не жесткой. Рецепторы управляются преобразователем дискретной информации, и они получают возможность переключаться с одного источника информации на другой. Эффекторы тоже являются управляемыми и могут воздействовать на различные объекты внешней среды. Измененный таким образом автомат будем называть роботом. Робот качественно отличается от автомата тем, что он сам выбирает нужные ему источники информации и сам выбирает объекты воздействия. Какая же закономерность заставляет робота переключаться с одного объекта внешней среды на другой? Робот, в отличие от автомата, имеет цель. Возьмем, например, промышленный робот. В него заложена цель — изготовить какую-то деталь. Он имеет какое-то множество заготовок. Он делает их обзор, то есть переключает свои рецепторы с одной заготовки на другую, сравнивая каждую из них с чертежом, выбирает подходящую заготовку и подключает к ней свои эффекторы. Дальше

все производится так же, как и у автомата. Робот, в отличие от автомата, должен различать объекты внешней среды, то есть должен обладать системой распознавания образов. Робот воспринимает не только количественную сторону информации, но и качественную. Для него информация делится на полезную и бесполезную. Имеются и ряд других отличительных особенностей робота, которые следуют из уже перечисленных. Например, объекты воздействия, как правило, как-то размещены в пространстве и, следовательно, робот для подключения к ним своих эффекторов должен получить возможность переключения в пространстве. Это, в свою очередь, может потребовать энергетическую автономность робота.

И автомат, и робот получают информацию из внешней среды. Если мы каким-то образом перекроем поток информации, то, как автомат, так и робот перестают работать. Представим себе несколько более совершенный объект, чем робот. Будем называть его интеллектуальным роботом. Интеллектуальный робот имеет две оболочки, одна из которых охватывает другую. Между оболочками расположена дополнительная память. Дополнительная она в том смысле, что преобразователь дискретной информации тоже может обладать памятью. Дополнительную память будем называть внутренней средой интеллектуального робота. В этой среде хранятся модели различных источников информации и модели различных объектов воздействия, с которыми имеет дело интеллектуальный робот. Другими словами, внутренняя среда является моделью внешней среды. Отличительной особенностью интеллектуального робота является то, что он не просто получает информацию от источников информации, расположенных во внешней среде, а изучает их и строит модели. Создание модели является преобразованием информации, и поэтому эта функция лежит на преобразователе дискретной информации. Внешняя среда безгранична, а внутренняя среда ограничена. Значит, вторая из них лишь частично отражает первую. Если какие-то внешние источники информации в данный момент недоступны интеллектуальному роботу, то он может обратиться к его модели, расположенной во внутренней среде. Поэтому интеллектуальный робот некоторое время может работать без внешних источников информации.

Внутренняя среда содержит и модели объектов воздействия. Следовательно, прежде чем выполнить какое-то действие над внешним объектом, интеллектуальный робот все это может проделать с его моделью. Появляется элемент предвидения. Но внешняя среда изменчива, и поэтому интеллектуальному роботу постоянно приходится изменять свою внутреннюю среду. Внутреннюю среду можно рассматривать как аккумулятор информации о внешней среде.

Обычный робот обладает аккумулятором энергии. По мере того, как этот аккумулятор разряжается, робот должен испытывать энергетический голод. Интеллектуальный робот, кроме энергетического, должен иметь и информационный голод, который наступает при рассогласовании внутренней и внешней среды.

Итак, мы познакомились с некоторыми классами автоматических объектов, в состав которых входит преобразователь дискретной информации, причем, это основная их часть. На вход этого преобразователя поступает дискретная информация. А что же такое дискретная информация?

Нас интересует качественная сторона дискретной информации, а не количественная. Если

выходная информация взаимно однозначно соответствует входной информации, то количество информации при преобразовании не изменяется. Соответствие между входной и выходной информацией может и не быть взаимно однозначным, но оно должно быть, по крайней мере, функциональным. В этом случае мы можем сказать, что количество выходной информации должно быть не больше количества входной информации. Поэтому для описания преобразователя дискретной информации нам сразу же приходится отказаться от количественной меры. Мы не можем определить преобразователь дискретной информации как объект, не увеличивающий количество выходной информации по сравнению с входной.

Элементарным носителем дискретной информации является символ. Символ — это объект, который может замещаться элементами некоторого множества. Символом, например, может быть «некоторая десятичная цифра». «Некоторая десятичная цифра» может быть цифрой 0, цифрой 1, и так далее до цифры 9. Символом может быть «некоторая русская буква», которая может замещаться буквой «а», буквой «б» и так далее. Для изображения символов мы можем использовать те же цифры или буквы. Мы объявляем их символами тем, что указываем множества их возможных значений. Запись $1 \varepsilon \{0, 2, 3\}$ означает, что символ 1 может замещаться цифрами 0, 2 и 3. Если мы конкретизировали значение символа, то тем самым мы принесли информацию о значении символа.

Если информацию мы будем определять так, как это описано в предыдущем абзаце, то это никак не противоречит определению Шеннона. Ведь все равно, назовем мы символ символом, сигналом или событием. Можно было бы ввести вероятности принятия тех или иных значений символом. И тогда все свелось бы к количественной мере информации по Шеннону.

Но мы введем в рассмотрение еще одно понятие: «символ связки». Символ связки — это совокупность двух отношений, которые могут быть бинарными, тренарными и так далее. Первое из этих отношений является отношением порядка на некотором множестве символов, а второе — это некоторое (произвольное) отношение на множествах возможных значений символов из первого множества. Математическое определение символа связки будет дано позднее, а сейчас пока ограничимся тем интуитивным ощущением, которое создается данным определением. Поясним лишь это определение с помощью примеров. Пусть, например, мы имеем запись «$y = \sin x$». Тогда «$\sin x$» — это символ связки. Первое отношение на множестве $\{x, y\}$ (это множество символов) является отношением порядка и показывает, что значение символа y определяется значением символа x, то есть x является аргументом, а y — функцией. Можно сказать, что x предшествует y. Второе отношение связывает значения символа x со значениями символа y. Если x принимает значение 0, то и y принимает значение 0. Если x принимает значение $\pi/2$, то y принимает значение 1. И так далее.

Символ связки не обязательно связывает между собой только два символа. Запись «$z = x + y$» устанавливает отношение порядка на множестве символов $\{x, y, z\}$. Символы x и y предшествуют символу z. Через значения символов x и y мы можем определить значение символа

z. Если *x* принимает значение 3, а y принимает значение 8, то *z* принимает значение 11.

В общем случае символ связки мы можем представить следующим графом:

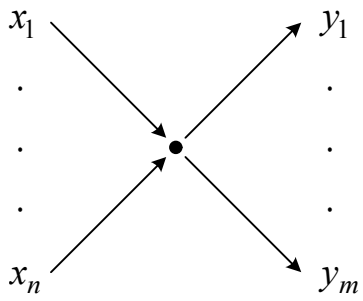


Рис. В.2

Здесь символы $x_1,...,x_m$ предшествуют символам $y_1,...,y_m$, а отношение *R* определяет соответствие между значениями символов. Это отношение не обязательно должно быть функциональным. Любую пару символов, один из которых предшествует другому, будем считать связанной с помощью символа связки. Например, связанной является пара (x_1, y_1). Отношение *R* рисунка В.2 не позволяет в общем случае, даже если оно является функциональным, по значению x_1 однозначно определить значение y_1, так как значение y_1 зависит от значений всех символов, предшествующих ему.

Символ связки устанавливает конкретное отношение предшествования на множестве символов и конкретное отношение на множестве значений символов. Но можно представить себе, что символ связки сам принимает значения из некоторого множества. Тогда будем называть его символом символа связки. Примером символа символа связки может быть следующая запись: «$y = f(x)$», где f — это некоторая функция. В этой записи отношение между значениями символов является не конкретным, но отношение предшествования — конкретным. Но символ символа связки может иметь и не конкретное отношение предшествования.

Символьное образование вида, показанного на рисунке В.2, будем называть символьной конструкцией. Здесь символы связаны между собой символами связки, либо символами символов связки. Любая композиция символьных конструкций — это тоже символьная конструкция. Понятие «композиция» будет определено позднее, а пока будем понимать это на интуитивном уровне. Отметим лишь, что запись вида: «$y = f(\varphi(x, g), t)$» является композицией двух функций, если f и φ — это функциональные отношения.

Если значения символа или символа символа связки могут выбираться произвольно, то множества этих значений будем заключать в фигурные скобки. Если же эти значения принимаются под управлением некоторой переменной, то множества значений будем заключать в угловые скобки.

Значения символов и символов символов связки могут сами быть символами или символами символов связки. Это дает возможность многоуровневого описания символьных конструкций. Значениями символов и символов символов связки могут быть и символьные конструкции.

Символьная конструкция может быть носителем информации. Мы не будем вводить количественную меру информации, хотя это можно было бы сделать. Любое сообщение, которое сужает множества значений символов или символов символов связки символьной конструкции, будем называть дискретной информацией (имеется в виду, что множества значений являются дискретными). Если же сообщение расширяет множество значений символов или символов символов связки, то мы будем называть его дезинформацией. Разрушение символьной конструкции — это тоже расширение множества возможных значений символов символов связки.

Приведем пример дезинформации. Возьмем описание некоторого образа. «В платьицах серебряных звездный хоровод. Ходит-бродит по небу месяц-звездочет. Солнце, словно печка, реки подо льдом...» Две первые фразы создают нам какую-то символьную конструкцию. Вторая фраза уточняет первую, то есть сужает множества значений символов и символов символов связки. Третья же фраза нам полностью разрушает символьную конструкцию.

Итак, если дискретная информация — это некоторая символьная конструкция, то преобразователь дискретной информации преобразует одну символьную конструкцию в другую. Но от того что мы выяснили, что делает преобразователь дискретной информации, задача его исследования не становится более легкой.

Из множества всевозможных символьных конструкций выделим класс, элементы которого имеют следующий вид:

$$x_1 \xrightarrow{R_1} x_2 \xrightarrow{R_2} x_3 \xrightarrow{R_3} \cdots \xrightarrow{R_{n-2}} x_{n-1} \xrightarrow{R_{n-1}} x_n \qquad \text{(B.1)}$$

Символьные конструкции такого вида будем называть линейными. В линейной символьной конструкции символом связки или символом символа связки связаны между собой лишь два соседних символа. Если $n=1$, то линейную символьную конструкцию вида (B.1) будем называть буквой. Множество значений символа M_1 будем называть алфавитом. Примером может служить символ, представляющий букву русского алфавита. Если $n>1$ и элементами символьной конструкции (B.1) являются буквы, то мы эту конструкцию будем называть словом. Если же элементами линейной символьной конструкции являются слова, то мы будем называть ее одномерным массивом слов. Линейная символьная конструкция, составленная из одномерных массивов слов, будет двухмерным массивом слов. И так далее.

Отношение предшествования в линейных символьных конструкциях является строго определенным, и поэтому символы связки и символы символов связки мы можем опускать, а лишь подразумевать их наличие. Но при этом мы должны иметь средства, позволяющие отличить одну букву от другой, средства, разделяющие слова, средства, разделяющие массивы слов. Отношения между значениями символов могут быть заданы отдельно, и совокупность всех этих отношений можно назвать грамматикой.

После того, как символьная конструкция будет определена математически, будет доказано, что любая символьная конструкция может быть линеаризирована. Это говорит о том, что любую

дискретную информацию мы можем представить буквами, словами и массивами слов различного числа измерений.

Теперь преобразователь дискретной информации мы можем представить состоящим из трех блоков, соединенных последовательно. Блок-схема ПДИ представлена на рис. В.3. Блок Л — это линеаризатор символьных конструкций, блок ПЛСК — это преобразователь линейных символьных конструкций, а блок ДЛ — это делинеаризатор символьных конструкций.

Рис. В.3

Преобразователь линейных символьных конструкций представляет собой композицию преобразователей букв, преобразователей слов и преобразователей массивов слов различного числа измерений. В преобразователе букв входная и выходная информация представлена множествами букв, в преобразователе слов — множествами слов, а в преобразователе массивов слов — множествами массивов слов. Возможны и преобразователи , изменяющие число измерений символьных конструкций. Например, на входы могут поступать слова, а на выходе получается буква. Или, наоборот, на вход поступает буква, а на выходе получается слово. Может на вход поступать слово, а на выходе получаться массив слов. Могут быть смешанные преобразователи, на входы которых поступают линейные символьные конструкции различного числа измерений. Смешанные преобразователи можно трактовать как управляемые. Например, если на входы поступают слова и буква, то мы можем считать, что осуществляется преобразование слов под управлением буквы.

Отношение вход-выход преобразователя линейных символьных конструкций является функциональным. Выходная информация преобразователя однозначно определяется входной.

Композиция преобразователей линейных символьных конструкций (букв, слов и массивов слов различного числа измерений) может быть представлена символьной конструкцией. Символы связки этой конструкции определяют отношения предшествования на множествах символов и функциональные отношения на множествах значений символов. Символы принимают значения букв, слов и массивов слов.

Остановимся на том, каким образом можно задавать функциональные отношения, реализуемые преобразователями линейных символьных конструкций. Начнем с преобразователей букв. На входы поступают *n* входных букв, а на выходах получаются *m* выходных букв (буквы понимаются как символы). Алфавиты букв в общем случае могут быть различными. Множество входных букв можно линейно упорядочить и представить как символьный вектор $\vec{X}$. Множество выходных букв также линейно упорядочивается и представляется как символьный вектор $\vec{Y}$.Если каждому из алфавитов входных и выходных букв мы сопоставим числовые значения, то числовые зна-

значения мы можем сопоставить векторам $\vec{X}$ и $\vec{Y}$. Пусть, например, на входы преобразователя букв поступают буквы x_1, x_2 и x_3, а на выходах получаются буквы y_1 и y_2. Буква x_1 может принимать значения «а» и «б», буква x_2 — значения «б», «в» и «г», а буква x_3 — значения «д» и «е». Выходные буквы y_1 и y_2 принимают значения «а» и «к». Алфавиту $\{а, б\}$ сопоставим цифровой алфавит $\{0,1\}$, алфавиту $\{б, в, г\}$ — цифровой алфавит $\{0,1,2\}$, алфавиту $\{д, е\}$ — алфавит $\{0,1\}$, а алфавиту $\{а, к\}$ — алфавит $\{0,1\}$. Тогда, если преобразователь букв набор $\langle ббе \rangle$ входных букв преобразует в набор $\langle ка \rangle$ выходных букв, то мы можем это представить так, как будто он число 101 преобразует в число 10. Входное число записано в неоднородной системе счисления, а выходное — в однородной двоичной системе счисления.

Иногда алфавиты входных и выходных букв сразу нам заданы в цифровой форме. Числа, записанные в неоднородной системе счисления, мы можем представить в привычной нам десятичной позиционной системе счисления. Будем их представлять как целые положительные числа. Если в некоторой позиционной системе счисления (однородной или неоднородной) число записано комбинацией цифр $\alpha_1\alpha_2...\alpha_n$, то соответствующее ему десятичное число N мы можем вычислить с помощью следующей арифметической формулы:

$$N = p_1 \cdot \alpha_1 + p_2 \cdot \alpha_2 + ... + p_n \cdot \alpha_n \qquad (В.2)$$

где $p_1, p_2, ..., p_n$ — это весовые коэффициенты.

Весовые коэффициенты определяются следующим образом: коэффициент p_n самого младшего разряда числа приравнивается единице. Коэффициент p_{n-1} равняется числу возможных значений цифры n-ного разряда. Весовой коэффициент $i-1$-го разряда определяется по следующей формуле:

$$p_{i-1} = p_i \cdot k_i \qquad (В.3)$$

где k_i — это число возможных значений цифры i-го разряда, а i принимает значения от единицы до n.

Переведем в десятичную систему счисления, например, число 101, которое записано в неоднородной системе счисления. Первая цифра может принимать два значения, вторая — три, а третья — два. Тогда весовые коэффициенты разрядов будут следующими: 6, 2 и 1. По формуле (В.2) мы получаем десятичное число 7. Выходное число 10 имеет веса разрядов 2 и 1. По формуле (В.2) мы получаем десятичное число 2. Таким образом, преобразованию набора букв $\langle ббе \rangle$ в набор букв $\langle ка \rangle$ соответствует преобразование десятичного числа 7 в десятичное число 2.

Преобразователь букв будет полностью заданным, если для каждой из возможных комбинаций входных букв мы укажем соответствующую ей комбинацию выходных букв. В числовой форме это означает, что для каждого входного числа мы указываем соответствующее выходное число. Но для возможности обратного перехода к буквам мы должны указать веса разрядов вход-

ного числа, записанного в неоднородной системе счисления, и веса разрядов выходного числа. Кроме того, должны быть заданы буквенные алфавиты каждого разряда входного и каждого разряда выходного чисел.

Соответствие между входными и выходными числами мы можем задать с помощью множества пар чисел. Первое число каждой пары является входным, а второе — выходным. Как любое конечное множество, так и множество пар мы можем линейно упорядочить. Упорядочим его по признаку монотонного возрастания первых чисел пар. Тогда в описании преобразователя букв мы можем опустить первые элементы пар. Получится последовательность выходных чисел, которую мы будем называть числовой последовательностью преобразователя букв. Это будет его моделью.

Приведем пример числовой последовательности преобразователя букв. Пусть мы имеем двоичный сумматор, обладающий тремя двоичными входами и двумя двоичными выходами. Два входа, которым дадим веса 4 и 2, являются входами *i*-х разрядов слагаемых. Третий двоичный вход, которому дадим вес 1, является входом сигнала переноса из младшего разряда многоразрядного двоичного сумматора. Двоичный выход, которому дадим вес 2, является выходом сигнала переноса в старший разряд многоразрядного двоичного сумматора. Выход, которому дадим вес 1, является выходом М-го разряда суммы. Алфавиты всех букв $\{0,1\}$ одинаковы и сразу заданы в цифровой форме. Рассматриваемый преобразователь букв мы можем задать следующей числовой последовательностью:

$$01121223 \qquad \text{(В.4)}$$

Комбинация состояний входов определяется номером элемента числовой последовательности. Номера отсчитываются, начиная с нуля, слева направо. Комбинация состояний выходов определяется числовым значением выбираемого элемента числовой последовательности.

Пусть, например, мы хотим определить комбинацию состояний выходов разряда двоичного сумматора при комбинации состояний входов $\langle 100 \rangle$. В десятичной системе счисления этой комбинации соответствует число 4. Отсчитываем слева направо, начиная с нуля, элементы числовой последовательности (В.4). По указанному адресу находим число 1. В двоичной системе счисления оно записывается как 01. Это говорит о том, что сигнал переноса в старший разряд многоразрядного двоичного сумматора равен нулю, а *i*-й разряд суммы равен единице.

Для удобства чтения числовой последовательности мы в нее можем вводить служебные знаки, например «пробелы». Отметим также, что числа числовой последовательности мы не обязательно должны представлять в десятичной, а можем в любой удобной для нас системе счисления. Но эти вопросы обсудим потом, когда вплотную займемся преобразователями букв.

Достоинством числовых последовательностей по сравнению с другими возможными способами описания преобразователей букв, например, в случае, когда алфавиты всех входных и выходных букв являются двоичными, то преобразователи букв мы можем описывать с помощью систем булевых формул, является компактность. Это немаловажное обстоятельство при выполнении

операций, связанных с проектированием преобразователей букв. Вторым достоинством является абстрактность числовой последовательности. Она нам указывает лишь соответствие «вход-выход» преобразователя букв. Мы не интересуемся тем, как реализован преобразователь букв. Он может быть реализован на логических элементах «И», «ИЛИ» и «НЕ», может быть реализован на микросхемах малой, средней или большой степени интеграции, может быть реализован в виде программы для электронной вычислительной машины и так далее. Можно провести аналогию с теорией электрических цепей. Электрические цепи мы можем анализировать с помощью законов Ома и Кирхгофа, можем анализировать методом контурных токов или узловых потенциалов. При этом учитывается конкретная структура цепей. Но мы можем воспользоваться и теорией четырехполюсников. Любой линейный четырехполюсник мы можем задать, например, *y*-матрицей, которая задает нам соответствие «вход-выход» и не интересуется внутренней структурой четырехполюсника. Третьим достоинством числовых последовательностей является то, что информация здесь представлена в числовой форме, а это удобно для использования в электронных вычислительных машинах при автоматизированном или автоматическом проектировании.

Но, как это всегда бывает, не могут быть только одни достоинства. То, что в числовой последовательности устранена всякая избыточность описания преобразователя букв, приводит к тому, что при «ручном» проектировании мы можем совершать ошибки, которые не могут быть обнаружены. Если в числовой последовательности вместо одного числа поставили другое, не выходящее за рамки максимально возможного для данной последовательности числа, то мы получим просто числовую последовательность другого преобразователя букв. Кроме того, следует отметить, что при выполнении операций над числовыми последовательностями мы встречаемся с трудностями, связанными с их непрерывностью.

При абстрактных рассуждениях о числовых последовательностях мы будем обозначать их заглавными буквами латинского алфавита или буквами с индексами, например, T_0, T_1 и так далее. Так как числовая последовательность реализует функциональное отношение, то при абстрактных рассуждениях мы можем записывать ее как $f(x_1, x_2, ...x_n)$.

Остановимся теперь на числовых моделях управляемых преобразователей букв. Пусть управляющая буква принимает значения от 0 до $k-1$. При каждом конкретном значении управляющей переменной, мы имеем обычный преобразователь букв, который характеризуется некоторой числовой последовательностью. При значении 0 этой переменной он описывается числовой последовательностью T_0, при значении 1 — последовательностью T_1, ..., при значении $k-1$ — последовательностью T_{k-1}. Такой управляемый преобразователь букв может быть задан набором числовых последовательностей следующего вида: $\langle T_0, T_1, ..., T_{k-1} \rangle$. Можно представить и в виде матрицы следующего вида:

$$\begin{pmatrix} T_0 \\ T_1 \\ \dots \\ T_{k-1} \end{pmatrix} \tag{B.5}$$

Номер строки этой матрицы указывает нам значение управляющей переменной. Номера строк отсчитывают сверху вниз, начиная с нуля.

Перейдем теперь к числовым моделям преобразователей слов. Такие преобразователи можно разделить на два класса: одномерные и многомерные преобразователи слов. В первом из этих классов нам не требуется для преобразования переходить в пространство с большим, чем одно, измерением, а во втором — требуется. Каждый из этих классов можно, в свою очередь, разделить на два класса. Преобразователи первого класса будем называть итеративными, а преобразователи второго — квазиитеративными. Класс одномерных итеративных преобразователей слов делится на классы правых итеративных преобразователей слов, левых итеративных преобразователей слов и ненаправленных итеративных преобразователей слов. Многомерные преобразователи слов тоже могут быть итеративными, либо квазиитеративными.

Правый одномерный итеративный преобразователь слов может быть определен следующей системой функций:

$$\begin{cases} a_{i+1} = f(a_i, \vec{X}_i), \\ \vec{Y}_i = \varphi(a_i, \vec{X}_i) \end{cases} \tag{B.6}$$

Здесь индекс i изменяется слева направо от единицы до I (I может быть и не ограниченным). Если на вход преобразователя слов поступают n входных слов, а на выходе получаются m выходных слов, то $\vec{X}$ и $\vec{Y}$ — это объединенные векторы i-х букв всех входных слов и i-х букв всех выходных слов соответственно. Переменная a_i — это внутренняя переменная i-го итеративного звена. Все звенья правого итеративного преобразователя слов одинаковы. Первая из функций системы (B.6) называется функцией переходов, а вторая — функцией выходов. Функция переходов определяет значение внутренней переменной на входе $i+1$-го итеративного звена через значение внутренней переменной i-го звена и состояние входа $\vec{X}_i$. Значения переменных a_{i+1} и a_i будем называть новым и старым значениями внутренней переменной соответственно.

Внутреннюю переменную a_i, так как она не связана непосредственно с входами преобразователя слов, можно рассматривать как управляющую. Тогда систему (B.6) мы можем рассматривать как уравнения, описывающие два управляемых преобразователя букв. Для того, чтобы как-то различать функцию переходов и функцию выходов, числовую модель первого управляемого преобразователя будем представлять в виде матрицы (B.5). Будем называть ее матрицей переходов. Числовую модель второго управляемого преобразователя букв будем представлять в виде набора числовых последовательностей, но в качестве разделителей будем использовать не запятые, а пробелы. Числовые модели двух управляемых преобразователей букв будем разделять запятой. Для того, чтобы подчеркнуть, что одномерный итеративный преобразователь слов является правым

итеративным, модели управляемых преобразователей букв будем заключать в квадратные скобки. Для того, чтобы по имеющейся числовой модели преобразователя слов и по заданному набору входных слов мы однозначно могли определить набор выходных слов, нужно указать начальное значение внутренней переменной. Начальное значение внутренней переменной будем указывать в нижнем левом углу матрицы переходов.

Приведем пример числовой модели правого итеративного преобразователя слов. На входы преобразователя поступают два многоразрядных двоичных числа, а на выходе получается выходное двоичное число, равное максимальному из двух двоичных чисел. Сравнение входных чисел производится в обычном арифметическом смысле.

$$0\begin{bmatrix}0120\\1111\\2222\end{bmatrix},\begin{bmatrix}0111 & 0101 & 0011\end{bmatrix} \tag{B.7}$$

Проиллюстрируем работу этой числовой модели. Входные числа будем записывать одно под другим. Над ними будем записывать внутренне слово, буквами которого являются значения внутренней переменной. Внутреннее слово отделяется от входных с помощью черты. Под всеми словами будем записывать выходное слово, которое отделено от входных чертой. Внутреннее слово строится буква за буквой слева направо. Самая левая буква внутреннего слова равняется нулю.

Внутреннее слово строится следующим образом. Слева записывается начальное значение внутренней переменной 0. Затем мы рассматриваем самую левую комбинацию входных букв. По адресу, определяемому этой комбинацией, в числовой последовательности нулевой строки матрицы переходов мы находим новое значение внутренней переменной и приписываем его справа к начальному значению этой переменной. Затем мы передвигаемся вправо на один шаг и проделываем то же самое, но уже с числовой последовательностью той строки матрицы переходов, номер которой мы нашли на предыдущем шаге. И так далее. Если входные и выходное слова *i*-буквенные, то для определения *i*-й буквы внутреннего слова мы используем комбинацию $i-1$-х букв входных слов.

Теперь покажем как строится выходное слово. Заметим, что порядок его построения произволен. Мы его можем строить слева направо, справа налево, начиная с середины и так далее. Выбирается какая-то комбинация букв входных слов. Над ней указана буква внутреннего слова, которое мы построили раньше. Числовая последовательность функции выходов состоит из отрезков, номера которых отсчитываются слева направо, начиная с нуля. Номер отрезка, которым мы должны воспользоваться при определении буквы выходного слова, равен выбранной букве внутреннего слова. В этом отрезке мы находим искомую букву выходного слова по адресу, определяемому комбинацией букв входных слов.

Пусть, например, нам заданы следующие два двоичных слова: 10110110 и 10111000. С помощью числовой модели (B.7) определим максимальное из этих двоичных слов.

$$\begin{array}{r} 00000111 \\ \hline 10110110 \\ 10111000 \\ \hline 10111000 \end{array} \qquad \text{(B.8)}$$

Строим внутреннее слово. Старшей буквой его является 0. В нулевой строке матрицы переходов по адресу 3, которому соответствует комбинация 11 букв входных слов, находим число 0. Это будет следующей буквой внутреннего слова, и приписываем ее справа. В нулевой строке матрицы переходов по адресу 0, которому соответствует комбинация 00 букв входных слов, находим число 0. Это будет третья буква внутреннего слова. И так далее. После пятой буквы мы переходим в первую строку матрицы переходов.

Построим теперь выходное слово. Порядок его построения произволен. Поэтому найдем лишь первую, пятую и шестую буквы. В нулевом отрезке числовой последовательности функции выходов по адресу 3 находим число 1. Это будет первая буква выходного слова. В нулевом отрезке числовой последовательности функции выходов по адресу 1, которому соответствует комбинация 01 значений входных переменных, находим число 1. Это пятая буква выходного слова. В первом отрезке числовой последовательности функции выходов по адресу 2, которому соответствует двоичная комбинация 10, находим число 0. Это шестая буква выходного слова. И так далее.

Числовая модель левого итеративного преобразователя слов отличается от числовой модели правого лишь тем, что вместо квадратных скобок мы используем круглые. С помощью системы функции переходов и функции выходов левый итеративный преобразователь описывается так:

$$\begin{cases} b_{i-1} = f(b_i, \vec{X}_i), \\ \vec{Y}_i = \varphi(b_i, \vec{X}_i) \end{cases} \qquad \text{(B.9)}$$

Левое итеративное преобразование слов производится так же, как и правое, но в противоположном направлении.

Приведем пример числовой модели левого итеративного преобразователя слов. На входы преобразователя поступают два двоичных числа, представленных в позиционном дополнительном коде. На выходе получается разность этих чисел (из первого вычитается второе), представленная в том же дополнительном коде.

Напомним, как представляются числа в позиционном дополнительном коде. Самый старший разряд числа является знаковым. Он всегда двоичный, в какой бы позиционной системе не было представлено число. Веса разрядов вычисляются по формуле (В.3), но вес самого старшего разряда устанавливается отрицательным. Числовое значение определяется по формуле (В.2). Пусть, например, мы имеем двоичное число 0101. Подставив цифры этого числа в формулу (В.2), получим: 0+4+0+1= +5. Пусть теперь мы имеем число 1101. Подставив цифры этого двоичного числа в формулу (В.2) с учетом того, что вес старшего разряда берется со знаком минус, получим -8+4+0+1= -3 (десятичное число минус три).

Числовая модель вычитателя записывается так:

$$0\begin{pmatrix}0100\\1101\end{pmatrix},\begin{pmatrix}0110 & 1001\end{pmatrix} \tag{B.10}$$

Проиллюстрируем ее работу на примере определения разности следующих двух двоичных чисел: 0001101011 и 1111001101. Первое из них соответствует десятичному числу +107, а второе — числу -51.

$$\begin{array}{r} 1100111000 \\ \hline 0001101011 \\ 1111001101 \\ \hline 0010011110 \end{array} \tag{B.11}$$

Самая младшая буква внутреннего слова устанавливается равной нулю. По адресу 3 числовой последовательности нулевой строки матрицы переходов стоит число 0. Букву 0 приписываем слева к ранее установленной букве 0. По адресу 2 нулевой строки матрицы переходов находим число 0. Букву 0 приписываем слева к ранее найденным буквам внутреннего слова. По адресу 1 нулевой строки находим число 1. Слева приписываем единицу. И так далее. Внутреннее слово записываем над входными словами в (B.11).

Выходное слово строится с помощью числовой последовательности функций выходов в произвольном порядке. Будем строить его слева направо. В первом отрезке по адресу 1 стоит число 0. Следующая буква выходного слова тоже будет нулем. В нулевом отрезке числовой последовательности функции выходов стоит единица, и мы заносим ее в выходное слово. На выходе получается двоичное число 0010011110, которому соответствует десятичное число + 158.

Ненаправленный итеративный преобразователь слов можно трактовать как вырожденный случай направленного итеративного преобразователя (правого или левого), когда внутренняя переменная является константой. Он характеризуется одной лишь функцией выходов:

$$\vec{Y}_i = f(\vec{X}_i) \tag{B.12}$$

Числовой моделью такого преобразователя является числовая последовательность, которую мы будем заключать в скобки для того, чтобы отличить ее от числовой последовательности преобразователя букв. Скобки могут быть как квадратными, так и круглыми.

Приведем пример ненаправленного итеративного преобразователя слов. На вход преобразователя поступает слово в алфавите $\{а, б, в\}$, а на выходе получается слово в том же алфавите. Всюду, где во входном слове встречается буква «*а*», в выходном слове мы заменяем ее буквой «*в*», букву «*б*» заменяем буквой «*а*», а букву «*в*» — буквой «*б*». Заменим заданный алфавит на цифровой: $\{0,1,2\}$. Тогда рассматриваемый преобразователь слов можно задать следующей числовой моделью:

$$[201] \tag{B.13}$$

Поясним работу этой числовой модели на примере преобразования слова *аббавббабв*. Этому слову соответствует число 0110211012. Запишем входное число и под ним с помощью (B. 13) построим выходное число, разделив эти числа чертой.

$$\frac{0110211012}{2002100201} \qquad \text{(В.14)}$$

Порядок преобразования входного слова произволен. Проведем преобразование слева направо. По адресу 0 в числовой модели (В. 13) стоит цифра 2, по адресу 1 — цифра 0 и так далее. Полученное выходное слово мы вновь можем записать в алфавите $\{a, б, в\}$. Получится слово *ваавбаавваб*.

Если ненаправленный итеративный преобразователь слов не имеет ни одной внутренней переменной, то квазиитеративный преобразователь слов их имеет сразу две. Одну из них будем называть правой, а вторую — левой. Функция выходов зависит от обеих внутренних переменных. Между собой же эти переменные являются взаимонезависимыми. Квазиитеративный преобразователь слов можно определить с помощью следующей системы функций:

$$\begin{cases} a_{i+1} = f(a_i, \vec{X}_i), \\ b_{i-1} = \varphi(b_i, \vec{X}_i), \\ \vec{Y}_i = \varphi(a_i, b_i, \vec{X}_i) \end{cases} \qquad \text{(В.15)}$$

Первая функция этой системы, которую будем называть правой функцией переходов, характеризует управляемый преобразователь букв. Управление производится правой внутренней переменной. Модель этого управляемого преобразователя букв будем задавать в виде матрицы переходов, заключенной в квадратные скобки. В левом нижнем углу указывается начальное значение правой внутренней переменной. Вторую функцию системы будем называть левой функцией переходов и задавать мы ее будем матрицей, заключенной в круглые скобки. Управление производится левой внутренней переменной. В левом нижнем углу матрицы указывается начальное значение левой внутренней переменной. Третью функцию системы будем задавать в виде числовой последовательности функции выходов. Эта числовая последовательность разделена на отрезки и группы отрезков. Это сделано для того, чтобы показать, что управление производится сразу двумя внутренними переменными. При переходе от одной группы отрезков к другой изменяется значение правой внутренней переменной. Группы отсчитываются слева направо, начиная с нулевой. При переходе от одного отрезка к другому внутри группы изменяется значение левой внутренней переменной. Отрезки внутри каждой группы отсчитываются слева направо, начиная с нулевого. Числовую последовательность функции выходов будем заключать в круглые или квадратные скобки.

Примером числовой модели одномерного квазиитеративного преобразователя слов может быть следующая:

$$0\begin{bmatrix} 0120 \\ 1111 \\ 2222 \end{bmatrix}, \quad 0\begin{pmatrix} 0120 \\ 1121 \\ 2122 \end{pmatrix}, \quad \begin{bmatrix} 0110 & 0100 & 0010 & 0110 & 0110 & 1001 \\ 0110 & 1001 & 0110 \end{bmatrix} \qquad \text{(В.16)}$$

На входы этого квазиитеративного преобразователя слов поступают два целых положительных двоичных числа, а на выходе получается абсолютная величина их разности, записанная в двоичной системе счисления.

Пусть мы имеем двоичные числа 10010101 и 10100101. Первому из них соответствует десятичное число 149, а второму — 165. Запишем заданные двоичные числа одно под другим. Над ними, в следующем этаже, построим левое внутреннее слово. Еще в более высоком этаже построим правое внутреннее слово. На самом нижнем этаже строится выходное слово. Этажи разделены между собой подчеркиванием.

$$\begin{array}{r} 00011111 \\ \hline 11200000 \\ \hline 10010101 \\ 10100101 \\ \hline 00010000 \end{array} \qquad \text{(B.17)}$$

Правое внутреннее слово строится слева направо с помощью правой матрицы переходов. Левое внутреннее слово строится справа налево с помощью левой матрицы переходов. Выходное слово строится с помощью числовой последовательности функции выходов в произвольном порядке. Получается число, десятичным эквивалентом которого является 16.

Одномерные преобразователи слов, как и преобразователи букв, могут быть управляемыми. Но способов управления преобразованием слов имеется гораздо больше, чем способов управления преобразованием букв. Эти способы будут рассмотрены позднее. Сейчас лишь отметим, что управление может производиться управляющей буквой, либо управляющим словом. Как в том, так и в другом случае, модель управляемого одномерного преобразователя слов мы можем представить в виде упорядоченного набора числовых моделей преобразователей слов. Каждая модель набора соответствует конкретному значению управляющей буквы. Эта буква может иметь постоянное значение для всех разрядов входных слов, и тогда мы говорим, что управление производится буквой. Если для различных разрядов слов управляющая буква может принимать различные значения, то управление производится словом.

Минимальное число измерений преобразователя одномерных массивов слов равняется двум. Будем называть их измерениями *i* и *j*. В измерении *i* при постоянном значении *j* изменяются номера букв входных и выходных слов преобразователя. В измерении *j* изменяются номера слов в одномерном массиве слов. Преобразователи в рассматриваемом классе могут быть правыми итеративными в обоих измерениях, правыми в измерении *i*, но левыми в измерении *j,* левыми в измерении *i*, но правыми в измерении *j*, левыми в обоих измерениях. Это невырожденные случаи двухмерных преобразователей одномерных массивов слов. В вырожденных случаях в одном или сразу в двух измерениях преобразователи могут быть ненаправленными. Все перечисленные преобразователи одномерных массивов слов относятся к классу итеративных. Мы можем строить также квазиитеративные и смешанные преобразователи. Смешанные преобразователи являются квазиитеративными в одном каком-то измерении и итеративными в другом. Квазиитеративные преобразователи одномерных массивов слов являются квазиитеративными в обоих измерениях.

Пусть мы имеем преобразователь одномерных массивов слов, который является правым итеративным в обоих измерениях. Такой преобразователь мы можем определить следующей сис-

темой функций:

$$\begin{cases} A_{j+1} = f(A_j, \vec{X}_j), \\ \vec{Y}_j = \varphi(A_j, \vec{X}_j) \end{cases} \qquad \text{(B.18)}$$

Здесь $\vec{X}_j$ и $\vec{Y}_j$ — это не векторные буквы, как раньше, а векторные входные и выходные слова. Внутренняя переменная A_j, теперь является не буквой, а словом. Первую из этих функций по-прежнему будем называть функцией переходов, а вторую — функцией выходов.

Внутреннюю переменную A_j можно рассматривать как управляющую. Тогда каждая из функций системы (B. 18) будет определять управляемый словом преобразователь слов. Управляемый словом преобразователь слов задается в виде набора числовых моделей преобразователей слов, каждая из которых соответствует какому-то значению буквы управляющего слова. При изменении индекса *i* номера буквы в словах, если при этом изменяется буква управляющего слова, мы переходим от одной модели к другой. Набор числовых моделей преобразователей слов мы можем задать как в виде последовательности этих моделей, так и в виде матрицы, в которой номера строк соответствуют различным значениям буквы управляющего слова. Условимся функцию переходов записывать в виде матрицы, строками которой являются модели преобразователей слов, а функцию выходов — в виде последовательности таких моделей. Тогда числовая модель правого итеративного в обоих измерениях преобразователя одномерных массивов слов может быть представлена в следующем виде:

$$\begin{bmatrix} M_0^A \\ M_1^A \\ \dots \\ M_K^A \end{bmatrix}, \left[M_0^Y \; M_1^Y \dots M_K^Y \right] \qquad \text{(B.19)}$$

Кроме этого, должны быть заданы два начальных слова, одно из которых развернуто в измерении *i*, а второе — в измерении *j*. В (B. 19) строками матрицы переходов являются числовые модели преобразователей слов функции переходов для различных значений буквы управляющего слова. Элементами числовой последовательности функции выходов являются числовые модели преобразователя слов функции выходов для различных значений буквы управляющего слова. Условимся записывать начальные внутренние слова, одно из которых развернуто в измерении *i*, а другое — в измерении *j,* в левом нижнем углу матрицы переходов одно под другим. Эти слова можно записывать как десятичные числа.

Отметим одну особенность числовой модели (B.19). Здесь строками матрицы переходов являются числовые модели, состоящие только из матриц переходов. Числовые последовательности функций выходов не отличаются от разверток матриц переходов, и поэтому их указывать не будем.

Преобразование одномерных массивов слов с помощью числовой модели (B. 19) производится следующим образом. Начальное слово, развернутое в измерении *i,* нам задано. Это началь-

ное значение управляющего слова, которое обозначим как β_1. С помощью этого слова и моделей числовой последовательности функции выходов мы строим слово α_1, первая буква которого нам известна. Мы выбираем ее из начального слова, развернутого в измерении *j*. После построения слова α_1, мы с его помощью и с помощью числовых моделей матрицы переходов строим внутреннее слово β_2. Затем мы строим слово α_2. И так далее. В результате мы получим два внутренних массива, один из которых обозначим как β, а второй — как α. При построении внутренних массивов слов используются только матрицы переходов числовых моделей. После построения внутренних массивов мы с помощью числовых последовательностей функций выходов числовых моделей строим выходной массив слов. Порядок построения внутренних массивов фиксирован. Для правых итеративных преобразователей в обоих измерениях массивы строятся слева направо и сверху вниз. Порядок построения выходного массива слов произволен.

Пусть, например, мы на входах преобразователя имеем два одномерных массива двоичных чисел. Мы эти массивы преобразуем в выходной массив двоичных чисел. Первое число выходного массива равняется максимальному из первой пары чисел входных массивов. Второе число выходного массива равняется максимальному числу из первых двух пар чисел входных массивов, то есть максимальному из четырех чисел. Третье число выходного массива равняется максимальному числу из трех первых пар входных массивов чисел. И так далее. Этот преобразователь является правым в обоих измерениях и его можно задать с помощью следующей числовой модели:

$$\begin{bmatrix} \beta \\ \begin{bmatrix} 0111 \\ 0101 \\ 0011 \\ 0111 \\ 0000 \\ 0101 \\ 0011 \end{bmatrix} \\ \begin{bmatrix} 1111 \\ 0101 \\ 0011 \\ 0111 \\ 1111 \\ 1111 \\ 1111 \end{bmatrix} \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ 0 \\ 0 \end{matrix} \begin{bmatrix} \alpha \\ \begin{bmatrix} 1111 \\ 0101 \\ 0011 \\ 0111 \\ 1111 \\ 1111 \\ 1111 \end{bmatrix} \end{bmatrix}, \begin{bmatrix} 0111 & 0101 & 0011 & 0111 & 0000 & 0101 & 0011 \end{bmatrix}$$

$$\begin{bmatrix} \alpha \\ \begin{bmatrix} 4560 \\ 1111 \\ 2222 \\ 3123 \\ 4444 \\ 4545 \\ 4466 \end{bmatrix} \end{bmatrix}, \begin{bmatrix} 0111 & 0101 & 0011 & 0111 & 1111 & 1111 & 1111 \end{bmatrix} \qquad \text{(B.20)}$$

Над матрицами указаны символы α и β, которые показывают, что ими мы пользуемся при определении соответствующих внутренних массивов слов. Проиллюстрируем работу этой числовой модели на примере преобразования следующих двух одномерных массивов слов:

$$\begin{array}{cc} 00001 & 00000 \\ 00000 & 00001 \\ 00101 & 00101 \\ 11010 & 10101 \end{array} \qquad \text{(B.21)}$$

Построение внутренних массивов слов начнем с того, что запишем заданные нам начальные слова, одно из которых развернуто в измерении *i*, а второе — в измерении *j*.

$$\begin{array}{cc} \beta & \alpha \\ 00000 & 0..... \\ ..... & 0..... \\ ..... & 0..... \\ ..... , & 0..... \end{array} \qquad \text{(B.22)}$$

Строим первое слово массива α. Нам известна первая буква этого слова. В нулевой строке нулевой матрицы по адресу 0, соответствующему комбинации $\langle 00 \rangle$ первых букв первых слов перерабатываемых массивов (B.21), стоит цифра 0, которую приписываем справа к заданной первой букве слова α_1. Мы при определении второй буквы слова α_1, пользуемся нулевой матрицей α, так как первая буква управляющего слова β_1, нулевая. Вторая буква управляющего слова β_1 тоже нулевая, комбинация вторых букв первых слов входных массивов (B.21) тоже $\langle 00 \rangle$. Поэтому третьей буквой внутреннего слова отбудет снова 0. И так далее. После построения слова α_1 мы получим следующие внутренние массивы:

$$\begin{array}{cc} \beta & \alpha \\ 00000 & 00000 \\ ..... & 0..... \\ ..... & 0..... \\ ..... , & 0..... \end{array} \qquad \text{(B.23)}$$

По известным внутренним словам α_1 и β_1, а также по заданным первым словам входных массивов (B.21) мы можем построить внутреннее слово β_2. Так как все буквы внутреннего слова β_1 нулевые, то мы на всех шагах будем пользоваться нулевой матрицей β. Так как все буквы слова α_1 нулевые, то мы будем пользоваться только нулевой строкой нулевой матрицы β. Первые четыре пары букв входных массивов (B.21) дают комбинации $\langle 00 \rangle$. По адресу 0 нулевой строки нулевой матрицы β стоит 0. Поэтому первые буквы внутреннего слова β_2 будут нулевыми. Пара пятых букв первых входных слов дает комбинацию $\langle 10 \rangle$. По адресу 2 в нулевой строке нулевой матрицы β стоит цифра 1. Она будет пятой буквой внутреннего слова β_2. В результате на этом шаге мы получим следующие внутренние массивы слов:

$$\begin{array}{cc} \beta & \alpha \\ 00000 & 00000 \\ 00001 & 0..... \\ ..... & 0..... \\ ..... , & 0..... \end{array} \qquad \text{(B.24)}$$

По известному внутреннему слову β_2 и по заданным вторым словам входных массивов (B.21) с помощью матриц числовой модели строится внутреннее слово α_2. По найденным внутренним словам β_2 и α_2, а также по заданным вторым словам входных массивов (B.21) строится внутреннее слово β_3. И так далее. В результате получим следующие внутренние массивы слов:

$$\begin{array}{cc} \beta & \alpha \\ 00000 & 00000 \\ 00001 & 00000 \\ 00001 & 00033 \\ 00101 , & 03222 \end{array} \qquad \text{(B.25)}$$

По найденным внутренним массивам слов и по заданным входным массивам строится выходной массив слов. Эта операция является ненаправленной. Буквы внутренних слов массива β определяют номера моделей функции выходов. Буквы слов внутреннего массива α определяют номера числовых отрезков. Комбинации соответствующих пар букв слов входных массивов определяют адреса чисел, которые мы должны выбрать. В каждом числовом отрезке нумерация ведется слева направо, начиная с нуля. С помощью (B.21) и (B.25) построим следующий выходной массив слов:

$$\begin{array}{c} 00001 \\ 00001 \\ 00101 \\ 11010 \end{array} \qquad \text{(B.26)}$$

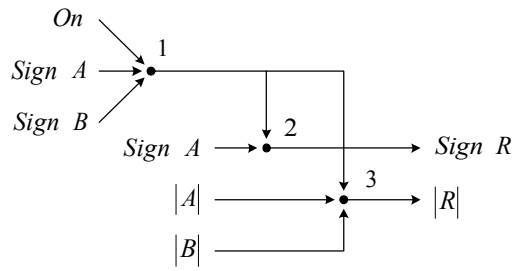
Числовые модели левых в обоих измерениях преобразователей одномерных массивов слов отличаются от рассмотренных лишь тем, что все скобки берутся круглыми. Если в одном измерении преобразователь является правым, а в другом левым, то внешние или внутренние скобки являются квадратными, а другие — круглыми. Если в одном или в обоих измерениях итеративный преобразователь одномерных массивов слов является ненаправленным, то из модели устраняются соответствующие матрицы переходов. Если в измерении *i* преобразователь является квазиитеративным, а в измерении *j* — правым итеративным, то модели M с индексами в (B.19) являются моделями квазиитеративных преобразователей слов. Если же преобразователь одномерных массивов слов является квазиитеративным в измерении *j*, то вместо одной матрицы переходов в (B.19) мы будем иметь две — правую и левую. И так далее. Числовые модели преобразователей одномерных массивов слов сложнее моделей преобразователей слов, но и объем перерабатываемой ими информации гораздо больше.

Преобразователи двухмерных массивов слов можно рассматривать как итерации или квазиитерации преобразователей одномерных массивов слов. Если вместо моделей M с индексами в

(В.19) мы подставим модели преобразователей одномерных массивов слов, то получим модель преобразователя двухмерных массивов слов. И так далее. Чем больше число измерений преобразователя дискретной информации, тем более сложными являются их числовые модели. Но с помощью символов мы можем сделать описание таких моделей многоуровневым и оно станет не таким сложным.

Теперь, когда мы познакомились с числовыми моделями преобразователей букв, преобразователей слов и преобразователей массивов слов, возвратимся вновь к блоку ПЛСК рисунка В.3. Было сказано, что он представляет собой композицию преобразователей символьных конструкций различного числа измерений. Такую композицию мы можем задать с помощью символьной конструкции, символы связки которой заданы числовыми моделями преобразователей, а символы символов связки — множествами или упорядоченными наборами таких моделей. На графе символьной конструкции символы связки и символы символов связки нумеруются. Затем под этими номерами указываются числовые модели или множества числовых моделей. Описание может быть многоуровневым, когда под соответствующим номером дается новая символьная конструкция.

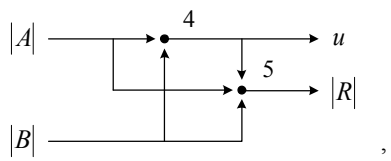
Приведем пример символьной конструкции преобразователя линейных символьных конструкций.



1. 01101001;

2. $<01;\ Sign\ A \xrightarrow{u} 6 \rightarrow Sign\ R,$ 6. $<10;01>>;$

3. $<\begin{pmatrix}0001\\0111\end{pmatrix}0,(0110\ \ 1001);$



,

4. $\begin{pmatrix}0010\\1011\end{pmatrix}0,(0010\ \ 1011);$

5. $<\begin{pmatrix}0010\\1011\end{pmatrix}0,(0110\ \ 1001);$

$\begin{pmatrix}0100\\1101\end{pmatrix}0,(0110\ \ 1001)>>.$

(В.27)

Это символьная конструкция алгоритма сложения и вычитания двоичных чисел разного знака, представленных в прямом коде. В этом коде числа представляются в виде знака и абсолютной величины. Результат операции представляется в этом же коде. Здесь Оп — это вход кода вы-

полняемой операции. Если это двоичная переменная принимает значение 0, то выполняется операция сложения. В противном случае из числа A вычитается число B. Двоичные переменные Sign A, Sign B и Sign R — это знаки входного и выходного чисел. Если такая переменная принимает значение 0, то соответствующее число положительное. В противном случае оно отрицательно. $|A|$, $|B|$ и $|R|$ — это абсолютные величины входных и выходного чисел, представленных в двоичной системе счисления.

В символьной конструкции (В.27) преобразователь 1 является преобразователем букв, преобразователь 2 — управляемым преобразователем букв, а преобразователь 3 — управляемым преобразователем слов. При значении 1 управляющей переменной преобразователя 1 преобразователь 3 состоит из преобразователей 4 и 5. Первый из них является преобразователем типа «слово-буква», а второй — управляемым преобразователем слов. В целом символьная конструкция (В.27) представляет квазиитеративный преобразователь слов, несмотря на то, что в ней используются итеративные преобразователи одного направления.

Проиллюстрируем работу этой символьной конструкции на примере сложения чисел +1 и -5. В двоичной системе эти числа записываются как 0001 и 1101. Самый старший разряд — знаковый. На входы преобразователя 1 поступает комбинация $\langle 001 \rangle$. По адресу 1 числовой последовательности этого преобразователя расположена цифра 1. При значении 1 управляющей переменной преобразователь 3 представляется символьной конструкцией, состоящей из преобразователей 4 и 5. С помощью числовой модели преобразователя 4 проведем операцию над словами 001 и 101. В результате получим букву 0. С помощью нулевой модели преобразователя 5 произведем операцию над словами 001 и 101. Получим слово 100. Это абсолютная величина результата. При значении 1 управляющей переменной преобразователя 1 преобразователь 2 характеризуется символьной конструкцией, состоящей из управляемого преобразователя букв 6. Управляющая переменная u имеет значение 0. Выбрав нулевой отрезок числовой модели преобразователя 6, мы по адресу 0, так как знак числа A равен нулю, найдем единицу. В результате на выходе получим число 1100, которое в десятичной системе записывается как -4.

Отметим, что символьная конструкция описывает связи между числовыми моделями, которые в свою очередь, описывают соответствия «вход-выход». Здесь никак не учитываются средства реализации. В настоящей работе предлагается следующий подход к проектированию преобразователей дискретной информации. Вначале проектируется абстрактная символьная конструкция, а затем она транслируется на язык конкретной реализации. Такими языками могут быть различные языки программирования и языки аппаратной реализации на заданной элементной базе. Трансляция на конкретные языки здесь называется этапом покрытия заданными средствами. При покрытии мы имеем уже спроектированную символьную конструкцию и числовые модели элементарных преобразователей, на которых мы хотим реализовать преобразователь дискретной информации. Это могут быть модели интегральных микросхем, модели команд электронной вычислительной машины и другие. Один покрывающий преобразователь может, в зависимости от сложности

его числовой модели, покрыть сразу несколько узлов спроектированной ранее символьной конструкции. Иногда перед покрытием символьной конструкции требуется предварительная ее трансформация. Например, при формальном программировании для однокристальной ЭВМ мы предварительно переработку информации должны сделать последовательной.

Существует множество методов логического проектирования преобразователей дискретной информации, ориентированных на конкретные средства реализации. Но средства реализации очень изменчивы. Это приводит к тому, что либо от этих методов приходится отказываться, либо их нужно в значительной степени переделывать. Можно представить себе ситуацию, когда методы логического проектирования, ориентированные на аппаратную реализацию, могут вообще не понадобиться. Микро-ЭВМ станут настолько малогабаритными, быстродействующими и дешевыми, что станет выгоднее реализовывать различные аппараты на микро-ЭВМ, чем изготавливать эти аппараты.

Ориентированные на конкретные средства реализации методы логического проектирования обладают еще одним недостатком. В настоящее время многие автоматические устройства частично реализованы аппаратно, а частично — программно. Для проектирования частей устройства мы должны применять различные методы. При этом мы не можем, кроме как на интуитивном уровне, рассмотреть многие вопросы проектирования, касающиеся всего устройства в целом.

Существуют и абстрактные теории, связанные с проектированием преобразователей дискретной информации, например, абстрактная теория конечных автоматов. Конечный автомат определяется как совокупность трех множеств и двух функций, заданных на этих множествах. Все множества конечны. Первое из них называется множеством состояний входа, второе — множеством состояний выхода, а третье — множеством состояний памяти. Функции, которые определяются на этих множествах, называются функцией переходов и функцией выходов. Абстрактный конечный автомат эквивалентен одномерному итеративному преобразователю слов. То есть, теория конечных автоматов рассматривает лишь один из классов преобразователей дискретной информации. Поэтому некоторые вопросы, связанные с проектированием реальных автоматов, не могут быть решены формально в теории конечных автоматов. Примером может служить теория микропрограммных автоматов, в которой неформально составляется алгоритм работы проектируемого автомата.

Полная формализация процесса логического проектирования нужна для того, чтобы его можно было сделать полностью автоматическим. В автоматической системе проектирования человек может присутствовать, но как пассивное звено. От него не требуется принятия каких-либо решений. Он может выступать лишь как источник информации. Он должен сообщить системе, что во что должно преобразовываться, то есть сообщить ей числовую последовательность соответствия «вход-выход» проектируемого преобразователя. Вместо человека система может использовать и какой-либо автоматический источник информации. Это, например, может быть программа, позволяющая вычислить соответствие «вход-выход». Можно представить себе следующую ситуацию. Мы имеем какой-либо аппарат, логика поведения которого нас удовлетворяет. Но он выпол-

нен на устаревшей элементной базе. Нужно перепроектировать этот аппарат. Устаревший аппарат через специальные переходные устройства мы можем подключить к ЭВМ, и тогда он будет для нее источником числовой последовательности соответствия «вход-выход». Если робот на другой планете должен спроектировать алгоритм своего поведения в незнакомой ему обстановке, то источником информации для него может быть только пассивная внешняя среда, а она не может дать формулы, схемы или алгоритмы. Воздействуя же на внешнюю среду и наблюдая за ее откликом, робот может получить числовую последовательность соответствия «вход-выход».

В настоящей работе в качестве исходной информации принята числовая последовательность соответствия «вход-выход». По этой последовательности строится числовая модель преобразователя. Этот этап называется обычно абстрактным синтезом. Затем производится декомпозиция найденной модели, то есть преобразователь дискретной информации разделяется на более простые части. В результате получается символьная конструкция проектируемого преобразователя. При декомпозиции принимается во внимание только критерий уменьшения сложности. Когда мы получим минимально сложную символьную конструкцию, можно перейти к этапу покрытия заданными средствами реализации. В сложных случаях декомпозицию можно производить в процессе абстрактного синтеза. Именно так, по соответствию «вход-выход» в главе 1.8 формально строится символьная конструкция (В.27). Затем эта символьная конструкция транслируется на язык граф-схем алгоритмов. Здесь используется только один неформальный шаг — это сопоставление числовым моделям преобразователей операций, которые нам знакомы и мы можем их представить с помощью математической символики. Но этот шаг в этой главе является неформальным только потому, что еще пока не рассмотрены вопросы формального покрытия.

Работа состоит из трех частей: нулевой, первой и второй. Нулевая часть является расширенным введением. Здесь даются краткие сведения из теории множеств, определяются символы, символьные функции и символьные конструкции. Даются процедуры линеаризации и делинеаризации символьных конструкций. Здесь более подробно, чем во введении, рассматриваются различные классы преобразователей линейных символьных конструкций и определяются их числовые модели. Здесь же делается постановка задачи проектирования преобразователей дискретной информации, в том числе и автоматического. Показано, что если для целей проектирования мы будем использовать язык символьных конструкций, то задача автоматического проектирования может быть замкнута на себя, то есть проектирующий автомат может проектировать и себя. В нулевой части три главы, первая из которых посвящена математике, вторая — информации и ее преобразованию, а третья — проектированию преобразователей дискретной информации и числовым моделям преобразователей.

Первая часть посвящена непосредственному проектированию символьных конструкций преобразователей дискретной информации. Она состоит из пяти глав. В первой главе рассматривается абстрактный синтез, композиция и декомпозиция преобразователей букв. Во второй главе рассматриваются одномерные итеративные преобразователи слов. Тут тоже рассматриваются вопросы композиции, декомпозиции и абстрактного синтеза и анализа. Но здесь еще освещаются и

методики проектирования различных тестов. Диагностические тесты могут использоваться при автоматическом проектировании числовых моделей одномерных итеративных преобразователей слов. Для этих же целей могут быть использованы и проверочные тесты. Ввиду того, что класс одномерных итеративных преобразователей слов изоморфен классу конечных автоматов, то во второй главе некоторые результаты, полученные в теории конечных автоматов, используются как готовые. В третьей главе рассматриваются вопросы абстрактного синтеза, композиции, декомпозиции и тестирования одномерных квазиитеративных преобразователей слов. Здесь показано, что если преобразователь слов содержит внутреннее управление, то он является квазиитеративным не смотря на то, что может быть составлен из одних только итеративных преобразователей слов одного направления итераций. В этой же главе дается методика трансляции символьных конструкций преобразователей на язык граф-схем алгоритмов. В четвертой главе рассматриваются многомерные итеративные и квазиитеративные преобразователи дискретной информации. В виду того, что модели при увеличении числа измерений преобразуемой информации усложняются, здесь рассматриваются вопросы многоуровневого проектирования. В этой же главе рассматриваются преобразования, требующие перехода в пространство с большим числом измерений. В пятой главе первой части рассматривается покрытие одних числовых моделей другими. Реализующие средства здесь тоже представляются на абстрактном уровне.

Вторая часть работы посвящена реализации преобразователей дискретной информации. Она состоит из трех глав. В первой главе дается аппаратная реализация. При аппаратной реализации мы должны учитывать многие физические особенности аппаратов, которые мы не учитываем при абстрактном проектировании символьных конструкций. К числу таких особенностей можно отнести гонки состояний памяти и комбинационные гонки. В главе дается краткий обзор современной базы. Во второй главе второй части рассматривается программная реализация преобразователей дискретной информации. Здесь тоже вводятся некоторые вопросы, которые не рассматривались при абстрактном проектировании символьных конструкций. К их числу можно отнести распределение памяти и трассировку пересылки данных. В третьей главе второй части предлагается модель вычислительной машины, не имеющей отдельного процессора. Машина разделена на ячейки, в каждой из которых имеются средства для хранения массивов информации, средства для хранения числовых моделей преобразователей и средства для осуществления преобразования по заданным числовым моделям. Программой для такой машины является символьная конструкция, которая может проектироваться автоматически. Быстродействие такой машины будет существенно повышено за счет того, что преобразование может осуществляться сразу во многих точках, причем, в этих точках мы имеем как бы специализированные процессоры, предназначенные для решения только заданной обработки конкретного массива. Некоторые преобразования могут осуществляться лишь после того, как выполнены другие. Могут потребоваться пересылки данных. Но во многих случаях пересылку данных можно заменить пересылкой числовых моделей. Числовые модели содержат в себе меньше информации (меньше чисел), чем перерабатываемые ими массивы данных.

Предлагаемая работа носит исследовательский характер. Целью исследования является проверка возможности полной формализации процесса логического проектирования. Многие вопросы проектирования, которые человек решает сравнительно легко на интуитивном уровне, здесь должны быть заменены цепочками формальных преобразований. А это создает трудности при восприятии методов. Методы логического проектирования являются цифровыми. Они ориентированы на вычислительную машину, а не на «ручное» проектирование. Ввиду того, что предлагаемое исследование пока не доведено до конца, грамотный обзор литературы, который следует дать во введении, пока дать нельзя. Он будет дан позднее. Пока лишь можно отметить, что на это исследование автора вдохновила книга Криницкого, посвященная роботам.