Н.П. БРУСЕНЦОВ

ОБ ИСПОЛЬЗОВАНИИ ТРОИЧНОГО КОДА И ТРЕХЗНАЧНОЙ ЛОГИКИ В ЦИФРОВЫХ МАШИНАХ

Вопрос о целесообразности применения троичного кода в цифровой технике возник вместе с появлением быстродействующих автоматических цифровых машин и на протяжении всего последующего периода неоднократно рассматривался в ряде монографий по цифровым устройствам и специально написанных на эту тему работ [1–8]. Заключения, высказанные по этому вопросу, весьма различны и во многих случаях являются выражением скорее субъективного мнения того или иного автора, чем результатов объективного исследования.

Независимо от теоретического решения данного вопроса количество практических работ по созданию элементов троичной техники, а также по разработке математического аппарата трехзначной логики из года в год увеличивается, что можно рассматривать как свидетельство перспективности троичных цифровых машин.

В настоящей работе предпринята попытка по возможности всестороннего рассмотрения на основе накопленного опыта основных вопросов, связанных с использованием трехзначной логики и троичного кода в цифровых машинах. В работе использованы результаты разработок, проводимых под руководством автора в Вычислительном центре Московского государственного университета и данные из литературных источников, на которые в соответствующих местах имеются ссылки.

§ 1. Массивы *p*-значных элементов

В цифровых машинах информация представляется массивами дискретных элементов. Элемент называется *p-значным*, если ему можно присвоить каждое из *p* каких-либо попарно различных значений.

Массив – это упорядоченная (пронумерованная) совокупность элементов. Массив, в котором значность всех элементов одинакова, называется *однородным*. Однородный массив *p*-значных элементов называется *p-ичным*.

В зависимости от системы нумерации элементов массивы могут быть одномерными и многомерными. Одномерные массивы элементов называются также *словами*.

Слово a, состоящее из элементов $a1$, $a2$, ..., an, мы будем обозначать $a[1{:}n]$. Число n элементов, содержащихся в слове, будем называть *длиной* слова.

Значение, принимаемое массивом, определяется совокупностью значений, присвоенных элементам этого массива. Массив, содержащий n p-значных элементов, способен принимать РП различных значений. Представление значений p-ичным массивом называется *p-ичным кодом*.

Совокупность p различных значений, присваиваемых p-значному элементу, называется *p-значным алфавитом*. Алфавит, в котором все значения рав-

ноправны и характеризуются единственным свойством – попарной различимостью, мы будем называть *абстрактным*.

Современная математическая логика и цифровая техника базируются на двузначных (p=2) алфавитах и двоичных кодах. Алфавиты с p>2 в отличие от двузначных называются *многозначными*, а соответствующие коды – *недвоичными*. Простейшим многозначным алфавитом является *трехзначный* (p=3) алфавит. Простейшим недвоичным кодом – *троичный код*.

§ 2. Экономность троичного кода

В качестве одного из преимуществ троичного кода нередко указывают на то, что этот код является самым экономным в том смысле, что количество оборудования, необходимое для реализации p-ичного массива, обладающего заданным числом различных значений, оказывается минимальным при p=3. Оценка затрат оборудования производится при этом в предположении, что количество оборудования в p-значном элементе пропорционально p. При таком предположении отношение количества оборудования, необходимого для реализации p-ичного массива, к количеству аналогичного оборудования, необходимому для реализации двоичного массива той же значности, выражается функцией [3]:

$$f(p)=\frac{p}{2\log_2 p},$$

которая имеет минимум при p=2,718...=e, а для целых p при p=3: $f(3) \cong 0{,}946$.

Таким образом, если элементы удовлетворяют указанному предположению, то троичный код оказывается на 5,4% экономнее двоичного.

Реальная экономность кода, вообще, может быть оценена лишь применительно к конкретной физической реализации элементов. Можно указать реализации (например, биакс, индуктивный параметром, взаимно скомпенсированная пара магнитных сердечников, поляризованное реле, электромагнитная линия задержки), в случае которых стоимость двузначного и трехзначного элементов одинакова. Это обусловлено тем, что в двузначном варианте возможности данных реализаций просто недоиспользуются. При таких реализациях применение троичного кода вместо двоичного дает экономию оборудования на 37%, или в 1,59 раза. К этой категории относятся постоянные магнитные запоминающие устройства, оперативные запоминающие устройства, в которых используется по два сердечника на элемент, и такой перспективный тип запоминающих устройств, как устройства на многослойных ферритовых пластинках [9, 10].

При запоминании на магнитной поверхности применение троичного кода позволяет увеличить информационную плотность в 1,59 раза в том случае, когда запись осуществляется достаточно длинными массивами. В случае же поэлементной записи троичный код может быть экономнее двоичного на 5,4%.

Вопросы построения троичных запоминающих устройств подробно рассмотрены в статье С.П. Маслова [11]. Пути экономного построения трехзначной логики рассматриваются в последующих параграфах.

§ 3. Различимость трехзначных сигналов

Присваивание элементам значений алфавита осуществляется с помощью сигналов. Распознавание значения, представленного сигналом, сводится к измерению величины физического параметра (например, силы тока, разности потенциалов, напряженности поля), являющегося носителем сигнала.

Для того чтобы элемент безошибочно воспринимал представленные сигналами значения алфавита, он должен измерять соответствующие величины с достаточной точностью. Чем больше значность р, тем жестче допуск на измеряемые величины – уровни сигнала, и тем выше должна быть точность измерений.

На рис. 1 показан график, характеризующий распознавательную способность некоторого трехзначного элемента. Элемент принимает значения $\alpha 1$, $\alpha 2$, $\alpha 3$ в зависимости от величины параметра φ. Интервалу $\Delta 1\varphi$ соответствует значение $\alpha 1$, интервалам $\Delta 2\varphi$ и $\Delta 3\varphi$ – значения $\alpha 2$ и $\alpha 3$. Интервалы $\Delta 12\varphi$ и $\Delta 23\varphi$ составляют область неопределенности – в пределах этой области значение, представленное величиной φ, данным элементом не может быть однозначно опознано.

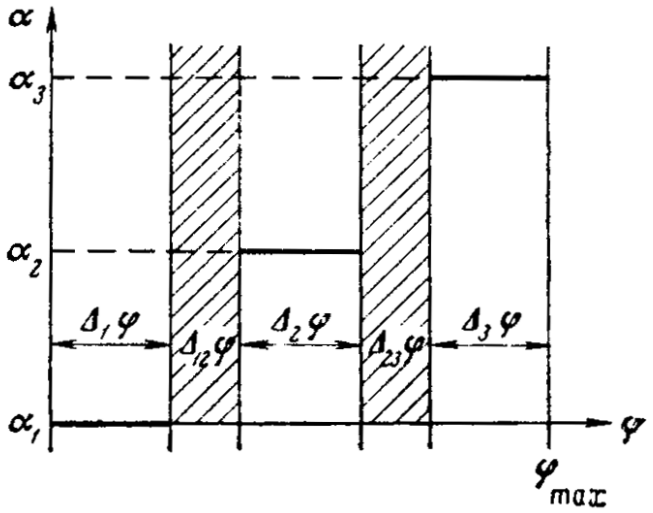

Рис. 1

Ширина области неопределенности для элементов конкретного типа определяется крутизной амплитудной характеристики элемента на участках перехода от одного состояния к другому [12] и нестабильностью положения этих участков при изменении рабочих условий и вследствие технологического разброса. Чем шире область неопределенности, тем жестче требования к стабильности уровней сигнала, которыми представлены значения алфавита. В идеальном случае, когда ширина области неопределенности равна нулю, усредненный допуск на уровни сигнала р-значного элемента составляет

$$\frac{\Delta_{\text{СР}}^{\text{ИД}}}{\varphi_{\max}} = \frac{\Delta_1\varphi+\Delta_1\varphi+\ldots+\Delta_p\varphi}{p\varphi_{\max}} = \frac{1}{p}$$

где $\varphi_{\max}$ – максимальный уровень сигнала.

В действительности допуск оказывается значительно ужесточенным из-за существования интервалов неопределенности:

$$\frac{\Delta_{СР}\varphi}{\varphi_{max}} = \frac{1-k_{СР}(p-1)}{p},$$

где

$$k_{СР} = \frac{\Delta_{12}\varphi+\Delta_{23}\varphi+\ldots+\Delta_{(p\ldots1)p}\varphi}{(p-1)\varphi_{max}}$$

отношение усредненного интервала неопределенности к максимальной величине сигнала.

Величины усредненного допуска

$$\frac{\Delta_{СР}\varphi}{\varphi_{max}},$$

вычисленные для идеального ($k_{ср}$=0) и ряда реальных значений кср в зависимости от значности элемента p, приведены в табл. 1.

Таблица 1

p \ $k_{ср}$	0	0,3	0,4	0,5	0,6
2	0,5	0,35	0,30	0,25	0,20
3	0,33	0,11	0,07	–	–
4	0,25	0,02	–	–	–

Из таблицы видно, что с увеличением значности p ужесточение допуска происходит настолько быстро, что возможность устойчивой передачи в реальных условиях даже трехзначных сигналов оказывается под сомнением, а передача сигналов большей значности представляется просто нереальной. Поэтому при необходимости передавать более чем два значения обычно используется надлежащее число двузначных сигналов, которые передаются либо по нескольким параллельным проводам, либо по одному проводу последовательно. Естественно, что в последнем случае время передачи увеличивается с возрастанием значности передаваемого алфавита.

Имеется, однако, возможность передачи трехзначного алфавита по одному проводу без увеличения времени и без ужесточения допусков. Речь идет о передаче сигналами положительной и отрицательной полярности (рис. 2). Реализация этой возможности при использовании троичного кода вместо двоичного повышает пропускную способность каналов в 1,59 раза: при последова-

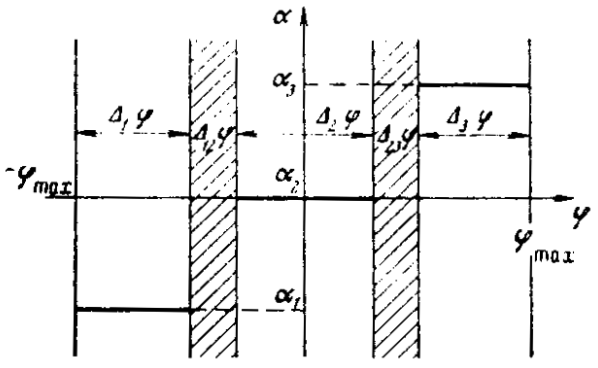

Рис. 2

тельной передаче по одному проводу в 1,59 раза сокращается затрачиваемое время, при параллельной передаче в 1,59 раза уменьшается число необходимых проводов.

Еще больший эффект достигается при однопроводной передаче трехзначных управляющих сигналов. В двузначном варианте каждый такой сигнал передается по двум проводам, поэтому с переходом к однопроводному трехзначному варианту количество соединений уменьшается в 2 раза. Опыт показывает, что в цифровой машине попарное объединение двузначных сигналов в трехзначные сравнительно легко осуществимо для подавляющего большинства управляющих сигналов. Получаемая за счет этого экономия количества соединений особенно важна в связи с миниатюризацией машин. Она может иметь существенное значение также в автоматических системах регистрации параметров и управления процессами [13].

§ 4. Предметная и числовая интерпретация p-значного абстрактного алфавита

Интерпретация значений, принимаемых элементами массивов и массивами в целом, осуществляется путем преобразования этих значений. В отрыве от преобразований обладающие различными значениями элементы представляются просто различающимися по некоторым признакам объектами, а массивы – хаотическими наборами этих объектов.

Одно и то же абстрактное значение получает различную интерпретацию в различных операционных устройствах машины. Например, в арифметическом устройстве оно рассматривается как число определенного вида, а в устройстве управления – как команда. В основе всякой интерпретации лежит интерпретация абстрактного алфавита, осуществляемая заданием операций, которыми устанавливаются те или иные отношения между значениями алфавита.

Одна из простейших интерпретаций p-значного алфавита связана с заданием операции циклической подстановки [14], которая определяет некоторую последовательность для p значений в замкнутом кольце. Существует $(p-1)!$ вариантов этой операции.

Дальнейшей интерпретацией алфавита может быть упорядочение его путем последовательной нумерации значений. При этом p-значный алфавит a можно представить словом $\alpha[1{:}p]$ с фиксированными, попарно различными значениями элементов $\alpha 1, \alpha 2, \ldots \alpha p$. Такую интерпретацию мы будем называть *предметной* или *логической*. Она может быть осуществлена заданием операции *дизъюнкции*:

$$\vee (\alpha_j, \alpha_k) = \alpha_{\max}(j, k)$$

или операции *конъюнкции*:

$$\wedge (\alpha_j, \alpha_k)\ \alpha_{\min}(j, k),$$

$j, k \in \{1, 2, \ldots p\}$. При этом значение a_1, удовлетворяющее условиям

$$\vee (\alpha_1, \alpha_k) = \alpha_k,$$

$$\wedge (\alpha_1, \alpha_k) = \alpha_i,$$

называется *первым* значением, а значение a_p удовлетворяющее условиям

$$\vee (\alpha_p, \alpha_k) = \alpha_p,$$

$$\wedge (\alpha_p, \alpha_k) = \alpha_k,$$

называется *последним значением алфавита*. В двузначной логике первым значением является «ложь», последним – «истина».

С предметной интерпретацией алфавита связана также операция *обращения (инверсии)*, определяемая формулой

$$\bar{\alpha}_k = \alpha_{p-k+i}.$$

Существует $p!$ различных вариантов упорядочения p-значного алфавита.

Другая интерпретация абстрактного алфавита, которую мы назовем *числовой*, или *арифметической*, заключается в том, что значениям приписываются веса. При этом значения p-значного алфавита рассматриваются как p последовательных целых чисел, включая ноль.

Эта интерпретация осуществляется заданием операций сложения и умножения, которыми, в частности, определяются значения 0 и 1:

$$\alpha_k + 0 = \alpha_k,$$

$$\alpha_k \times 0 = 0,$$

$$\alpha_k \times 1 = \alpha_k.$$

Предметная интерпретация алфавита не связана с понятием веса значений и поэтому представляется более свободной, чем числовая. В логике это подчеркивают тем, что употребляют нечисловые обозначения значений, например: *Л* и *И* или *F* и *T*, хотя, по-видимому, удобнее было бы употреблять числа, рассматривая их как порядковые номера предметных значений.

При $p \geq 3$ в одном и том же алфавите могут содержаться положительные и отрицательные значения. Особый интерес представляют алфавиты, в которых число положительных значений равно числу отрицательных значений. Такие алфавиты мы будем называть *симметричными*. Простейшим из них является трехзначный симметричный алфавит.

§ 5. Числовая интерпретация p-ичных слов

Интерпретация слов является развитием интерпретации алфавита. При интерпретации алфавита рассматриваются входящие в него фиксированные значения, при интерпретации слова – элементы, которым эти значения можно присваивать.

Основу логической интерпретации слов составляет рассмотренная в § 1 нумерация элементов слова. В основу числовой интерпретации слов положено различие элементов по *масштабу весов*, проявляющееся в том, что каждому элементу слова приписывают определенный разряд. Разряд характеризуется целым числом β: численное значение элемента (β-того разряда равно $\alpha_k p^\beta$, где α_k – численное алфавитное значение, присвоенное данному элементу.

Для осуществления числовой интерпретации слов должны быть заданы арифметические операции, однозначно определяющие соответствие множе-

ства принимаемых словами абстрактных значений числам. При этом главным условием, ограничивающим выбор операций, должно быть, очевидно, требование их экономной реализуемости в смысле затрат оборудования и времени. Практически, однако, операции предопределены выбором кода чисел, который обычно основан на том, что слово $a[1{:}n]$ интерпретируется как целое число $a^{\text{нат}}$, определяемое формулой:

$$a^{\text{нат}} = a_1 p^{n-1} + a_2 p^{n-2} + \ldots + a_{n-1} p + a_n = \sum_{i=1}^{n} a_i p^{n-1}.$$

Представление чисел, удовлетворяющее этой формуле, мы будем называть *натуральным кодом*.

Наиболее важны варианты натурального кода, соответствующие p-значным алфавитам, численные значения которых образуют полные системы наименьших неотрицательных или абсолютно наименьших вычетов по модулю p. При этом численные значения слов длины n являются соответственно наименьшими неотрицательными и абсолютно наименьшими вычетами по модулю p^n.

В натуральном коде простейшими арифметическими операциями являются операции умножения на целую степень p и сложения.

Умножение на p^1 сводится к тому, что каждому элементу слова, начиная с первого, присваивается значение следующего за ним по порядку номеров элемента, а последнему элементу присваивается значение 0. Умножение на p^{-1} выполняется в обратном порядке: каждому элементу, начиная с n-го, присваивается значение предшествующего ему элемента, а элементу с номером 1 присваивается значение 0. Другими словами, для умножения на p^1 выполняется один сдвиг значений влево, а для умножения на p^{-1} – один сдвиг вправо. Для умножения на p^q, q – целое, требуется, очевидно, q сдвигов влево или вправо в зависимости от знака q.

Операция сложения двух чисел, представленных в натуральном коде словами $a[1{:}n]$ и $b[1{:}n]$, заключается в после довательном определении значений элементов s_i слова-суммы $s[1{:}n]$, начиная с $i{=}n$, по формуле:

$$s_i = \operatorname{mod}_{\alpha_1}^{\alpha_p}(a_i + b_i + c_i), \quad \alpha_p - a_1 = p,$$

где c_i – значение переноса с i-того места:

$$c_i = \begin{cases} 0, \text{ если } i > n, \\ (a_i{+}b_i{+}c_i{+}1{-}s_i)p^{-1}, \text{ если } i \le n. \end{cases}$$

Функция $\operatorname{mod}_{\alpha_1}^{\alpha_p}(x)$ осуществляет замену значения x сравнимым с ним по модулю p алфавитным значением. В алфавитах с неотрицательными значениями это будет наименьший неотрицательный вычет, в симметричных алфавитах – абсолютно наименьший вычет по модулю p.

Если для суммы установлена та же длина слова, что и для слагаемых, то правильное представление значений суммы существует только в тех случаях, когда $c_1{=}0$. Ситуация $c_1{\neq}0$ называется *переполнением*.

При $p{=}2$ не существует единого натурального кода для положительных и отрицательных чисел. Поэтому проблему представления чисел со знаком в двоичном коде приходится решать на уровне слов при помощи специальных кодов. Мы рассмотрим наиболее часто встречающийся в современных цифро-

вых машинах и, по-видимому, наиболее совершенный из этих кодов – *дополнительный код*.

В дополнительном коде значения p-ичного слова a [1:n] интерпретируются как абсолютно наименьшие вычеты по модулю p^n натуральных числовых значений этого слова:

$$a^{\text{доп}} = \left| a^{\text{нат}} \right|_{p^n} = \left| \sum_{i=1}^{n} a_i p^{n-1} \right|_{p^n}.$$

Прямые скобки с индексом p^n под правой скобкой обозначают операцию получения абсолютно наименьшего вычета по модулю p^n. Для четного p условимся, что отрицательных вычетов больше, чем положительных.

Сложение в дополнительном коде сводится к сложению в натуральном коде:

$$a^{\text{доп}} + b^{\text{доп}} = \left| a^{\text{нат}} \right|_{p^n} + \left| b^{\text{нат}} \right|_{p^n} = \left| a^{\text{нат}} + b^{\text{нат}} \right|_{p^n}.$$

При этом, однако, усложняется фиксация переполнения. Например, в случае p=2, чтобы обеспечить фиксацию переполнения, применяется модифицированный дополнительный код, в котором отсутствие переполнения гарантируется при соблюдении условия $-2^{n-2} \le a^{\text{доп}} < 2^{n-2}$, контролируемого по значениям первых двух элементов слова.

Умножение на целую степень p в дополнительном коде выполняется в виде модифицированного сдвига, который в случае p=2 отличается от обычного сдвига тем, что при сдвиге вправо элемент с номером 1 до конца операции сохраняет свое первоначальное значение.

Изменение знака числа в дополнительном коде осуществляется путем обращения значений всех элементов слова и затем сложения обращенного слова с константой 1.

Благодаря тому, что представление числа со знаком реализовано в дополнительном коде на уровне слов, в нем, так же как в прямом и обратном кодах, сумматоры работают со словами фиксированной длины. Дело в том, что первый и последний элементы слова в сумматоре чисел со знаками должны рассматриваться как особые. Особенность первого элемента, соответствующего старшему разряду числа, обусловлена необходимостью фиксировать случаи переполнения сумматора. Особенность последнего элемента, соответствующего младшему разряду, связана с операцией изменения знака числа, при выполнении которой производится прибавление единицы младшего разряда. Невозможность непосредственного сложения чисел, представленных словами разной длины, приводит к тому, что для представления, например, малых приращений приходится применять слова той же длины, что и для полноразрядных чисел, или строить специализированные сумматоры, как это делается в цифровых дифференциальных анализаторах.

Неприятным следствием введения знака числа на уровне слов является также необходимость двух вариантов операции сдвига: логический сдвиг и арифметический (в дополнительном коде – модифицированный)сдвиг.

Наконец, немаловажным недостатком как дополнительного, так и обратного кодов является трудность их восприятия и понимания человеком, свидетельством которой служат пространные и нередко основательно запутанные описания этих кодов в учебниках.

Представление чисел со знаком можно осуществить на уровне элементов двумя путями:

1) путем использования алфавита с положительными и отрицательными весами значений, что возможно при $p \geq 3$;

2) путем специальной интерпретации слов, которая известна под названием «система с отрицательным основанием» [15] и заключается в том, что слово $a[1:n]$ рассматривается как число аотр, определяемое формулой

$$a^{\text{отр}} = \sum_{i=1}^{n} a_i (-p)^{n-i}.$$

Главное и, по-видимому, единственное преимущество второго пути состоит в том, что он не исключен при $p=2$. Однако, оптимальное представление чисел со знаком находится на первом пути. Этим представлением является натуральный код в симметричном алфавите. Мы будем называть его *симметричным* кодом.

В симметричном коде сложение чисел со знаками технически равноценно сложению положительных чисел в натуральном коде с неотрицательным алфавитом, а изменение знака числа выполняется просто поэлементной инверсией соответствующего слова. При этом не только возможно непосредственное сложение чисел разной длины, но также можно любую часть слова интерпретировать как отдельное число со знаком и выполнять арифметические операции над словами как над совокупностями таких чисел. Это открывает новые перспективы развития и вместе с тем упрощения структуры операционных устройств цифровых машин и создает полезные возможности для программирования [16].

Заметим, что преимущества симметричного алфавита могут быть реализованы также при непозиционном представлении чисел, например, в системе остаточных классов [17].

Основная трудность реализации арифметических преимуществ симметричного кода заключается в том, что они находятся за пределами привычного и технически обжитого двузначного алфавита. Однако трудность эта не так уж велика: для того чтобы реализовать все указанные преимущества, достаточно перейти от двузначного алфавита к трехзначному.

§ 6. Округление чисел

Округление необходимо всякий раз, когда длина слова недостаточна для точного представления числа, и поэтому приходится прибегать к приближенному представлению. Округление называется *правильным*, если всякое непредставимое точно данным словом число представляется ближайшим к нему значением этого слова, причем вероятность представления чисел с избытком и с недостатком одинакова.

Наиболее совершенный из применяемых в двоичных машинах способов округления заключается в том, что для округления числа, представленного в прямом или в дополнительном коде, в старший из отсекаемых разрядов прибавляется 1. Если в этом разряде числа содержалась цифра 1, то имеет место перенос в младший из сохраняемых разрядов, причем в худ-

шем случае перенос может распространиться по всей длине слова и даже вызвать переполнение.

Таким образом, операция округления двоичных чисел в прямом и в дополнительном коде связана с необходимостью производить сложение, т. е. для ее реализации требуется сумматор и соответствующее время. Эти затраты, которые иногда расцениваются как чрезмерные [2], в двоичной системе счисления, как и во всякой системе с четным основанием, не обеспечивают правильного округления. Алгоритм работает неудовлетворительно в тех случаях, когда значение, содержащееся в отсекаемых разрядах числа, составляет точно половину единицы младшего сохраняемого разряда, т. е. когда округляемое число оказывается на одинаковом удалении от двух ближайших значений слова. Это иллюстрируется табл. 2, из которой видно, что при округлении данным способом при указанном условии в дополнительном коде все числа, а в прямом коде абсолютные величины всех чисел, оказываются представленными с избытком.

Таблица 2

x	Прямой код		Дополнительный код	
	x	$x_{окр}$	x	$x_{окр}$
+2,5	0010,1	0011=+3	0010,1	0011=+3
+1,5	0001,1	0010=+2	0001,1	0010=+2
+0,5	0000,1	0001=+1	0000,1	0001=+1
−0,5	1000,1	1001=−1	1111,1	0000= 0
−1,5	1001,1	1010=−2	1110,1	1111=−1
−2,5	1010,1	1011=−3	1101,1	1110=−2

Дело можно исправить путем дальнейшего усложнения алгоритма округления. А именно в тех случаях, когда округляемое число находится на одинаковом удалении от двух ближайших к нему значений слова, можно условиться округлять его всегда до четного значения. Это соответствует известному правилу округления в десятичной системе, согласно которому в тех случаях, когда отсекается цифра 5, в младший сохраняемый разряд прибавляется 1 лишь при условии, что в этом разряде содержится нечетная цифра.

Реализация этого способа округления в машине при использовании двоичного прямого или дополнительного кода связана, вообще говоря, с необходимостью просматривать все отсекаемые разряды. Прибавление единицы в старший отсекаемый разряд производится непременно, если в младшем сохраняемом разряде находится единица. Если же в этом разряде находится нуль, то прибавление должно производиться лишь в том случае, когда хотя бы в одном отсекаемом разряде, помимо старшего, содержится единица.

Табл. 3 показывает, что усовершенствованный алгоритм дает одинаковое и более правильное округление как в прямом, так и в дополнительном коде.

Важность правильного округления чисел в цифровой машине не очевидна, и этот вопрос до последнего времени, по-видимому, даже не обсуждался. На существенность его обратил внимание В.В. Воеводин [18].

Наиболее просто проблема округления решается в симметричном коде. Значение отсекаемой справа части числа в симметричном коде никогда не превышает по абсолютной величине половину единицы младшего сохраняе-

Таблица 3

x	Прямой код		Дополнительный код	
	x	$x_{окр}$	x	$x_{окр}$
+2,5	0010,1	0010=+2	0010,1	0010=+2
+1,5	0001,1	0010=+2	0001,1	0010=+2
+0,5	0000,1	0000=+0	0000,1	0000= 0
−0,5	1000,1	1000=−0	1111,1	0000= 0
−1,5	1001,1	1010=−2	1110,1	1110=−2
−2,5	1010,1	1010=−2	1101,1	1110=−2

мого разряда, поэтому всякое отсечение автоматически является правильным округлением числа. При этом не только получается экономия оборудования и времени, затрачиваемых для выполнения округления, но и сокращается количество необходимых арифметических операций, так как операции умножения, сдвига, сложения и вычитания с плавающей запятой, выполняемые обычно в двух вариантах с округлением и без округления результата, становятся однозначными.

Симметричный троичный код – простейший код, обладающий этим ценным свойством.

§ 7. Преимущество трехзначной логики

Подобно тому как всякое число различных значений, представленное в данном p-ичном коде, можно представить также в любом другом $p \geq 2$ коде, всякое преобразование значений, реализуемое в логике данной значности, может быть осуществлено в любой p-значной $p \geq 2$ логике. При этом экономность реализаций преобразования, так же как экономность кода, зависит от значности

Понятие оптимального в смысле экономной реализации преобразований значения p, по-видимому, имеет смысл лишь для конкретного класса преобразований и используемых технических средств. В частности, господство двузначной логики в современных цифровых машинах объясняют обычно тем, что практически всем используемым в настоящее время физическим элементам свойственны лишь два устойчивых состояния.

Применительно к машинам это объяснение в общем правдоподобно, хотя физические элементы с числом состояний большим, чем два, вполне осуществимы и не получили распространения, может быть, потому, что до сих пор все усилия были сосредоточены на развитии двузначных элементов. Однако логика не исчерпывается цифровыми машинами, и ее двузначная основа определилась независимо от физических элементов машин.

Никто не доказал абсолютных преимуществ рассуждения по принципу «да»–«нет», наоборот, имеются примеры, показывающие преимущества многозначных логик [19, 30]. По-видимому, основная причина преобладания двузначной логики заключается в том, что эта логика простейшая.

Преимущество трехзначной логики перед двузначной можно наглядно показать на примере рассуждения путем ответов на общие (т. е. не содержа-

щие вопросительных слов) вопросы. В двузначной логике на каждый такой вопрос допустим один из двух ответов: «да», «нет». На практике не реже, и даже чаще, случается ответ, не являющийся ни утверждением, ни отрицанием, например: «неизвестно», «неопределенно», «безразлично», «бессмысленно» и т. п. – в общем, «нет ответа». Эта ситуация в двузначной логике равносильна тупику. Чтобы обойти тупик, надо перед тем, как поставить данный, вопрос x, убедиться в том, что на него есть ответ, т. е. задать предварительный вопрос: «Есть ли ответ на вопрос x?»

В трехзначной логике предварительный вопрос не нужен, так как имеется возможность трех ответов на основной вопрос. Ясно, что рассуждение в трехзначной логике происходит быстрее и проще. По-видимому, на практике мы чаще всего пользуемся именно трехзначной логикой.

Пути реализации указанного преимущества трехзначной логики в цифровых машинах, которые на современном этапе представляют собой полностью детерминированные системы, не очевидны. Однако и в детерминированных системах с использованием трехзначной логики может быть связана существенная экономия средств, благодаря тому, что, во-первых, трехзначная ситуация в этих системах случается весьма часто, и во-вторых, подавляющее большинство двузначных управляющих сигналов можно попарно объединить в трехзначные сигналы.

§ 8. Трехзначная логика и теория силлогистики

Убедительным свидетельством эффективности трехзначной логики может служить применение ее для исследования силлогистики – логической системы, наиболее непосредственно отражающей способ рассуждений человека.

Теория силлогистики, созданная Аристотелем в IV в. до н. э., сохранилась в сущности неизменной до наших дней. Алгебраизация силлогистики была начата еще Лейбницем. Гильберт и Аккерман [20] показали, что в комбинированном исчислении высказываний и предикатов можно обосновать только 15 из 19 истинных модусов классической силлогистики. В связи с этим существует мнение, что силлогистика представляет собой оригинальную дедуктивную систему, не выводимую в исчислении предикатов, и ее современное изложение базируется на специальной системе аксиом, определяющей внешние свойства основных силлогистических отношений A, E, I, O без раскрытия их внутренней логики [21, 22].

Далее будут построены выражения отношений A, E, I, O, позволяющие осуществить вывод всех тождеств силлогистики в трехзначном исчислении предикатов.

Рассмотрим алфавит α[1:3], над которым определены переменные x, y, z, … и операции:

дизъюнкция $\vee(\alpha_j, \alpha_k) = \alpha j \vee \alpha_k = a_{\max}(j, k)$,

конъюнкция $\wedge(\alpha_j, \alpha_k) = \alpha_j \wedge \alpha_k = a_{\min}(j, k)$,

инверсия $\overline{\alpha}k = \alpha_{3-k+1}$ $\quad j, k \in \{1, 2, 3\}$.

Значение α_3 обозначим цифрой 1 и будем интерпретировать как утверждение – «да», значение α_1 обозначим цифрой 0 (отрицание – «нет»), значение α_2 – буквой *i* (неопределенность – «то ли да, то ли нет»).

В рассматриваемой системе выполняются следующие основные тождества булевой алгебры: Закон двойного обращения:

$$\overline{\overline{x}} = x$$

Коммутативность дизъюнкции и конъюнкции:

$$x \wedge y = y \wedge x,$$

$$x \vee y = y \vee x.$$

Ассоциативность конъюнкции и дизъюнкции:

$$x \wedge (y \wedge z) = (x \wedge y) \wedge z,$$

$$x \vee (y \vee z) = (x \vee y) \vee z.$$

Дистрибутивность конъюнкции относительно дизъюнкции и дистрибутивность дизъюнкции относительно конъюнкции:

$$x \wedge (y \vee z) = x \wedge y \vee x \wedge z,$$

$$x \vee y \wedge z = (x \vee y) \wedge (x \vee z)$$

(здесь и далее предполагается, что знак конъюнкции Л связывает сильнее, чем знак дизъюнкции $\vee$).

Правила де Моргана:

$$x \overline{\wedge} y = x \vee \overline{y},$$

$$\overline{x \vee y} = \overline{x} \wedge \overline{y}.$$

Законы поглощения:

$$x \wedge (x \vee y) = x,$$

$$x \vee (x \wedge y) = x,$$

в частности, законы идемпотентности:

$$x \wedge x = x,$$

$$x \vee x = x.$$

Тождества с константами 0 и 1 :

$$x \wedge 0 = 0,$$

$$x \wedge 1 = x,$$

$$x \vee 0 = x,$$

$$x \vee 1 = 1,$$

$$\overline{0} = 1,$$

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

Недействительными оказываются законы

противоречия: $x \wedge \overline{x} = 0$,

исключенного третьего: $x \wedge \overline{x} = 1$.

Термины силлогистики мы интерпретируем как признаки x, y, z, которыми могут обладать объекты некоторого множества M.

Рассмотрим дизъюнкцию $\underset{M}{\vee} x$ значений, принимаемых переменной x на объектах множества M, взятую по всем объектам этого множества. Если признак x присущ хотя бы одному объекту из M, то $\underset{M}{\vee} x = 1$. Если о каждом объекте из M известно, что признак x ему не присущ, то $\underset{M}{\vee} x = 0$. Если же хотя бы для одного объекта $m_j \in M$ присущность признака x не определена, т. е. $x(mj) = i$, и нет ни одного объекта, которому признак x присущ, то $\underset{M}{\vee} x = \mathrm{i}$.

Сначала предположим, что неопределенность исключена, т. е. $\underset{M}{\vee} x \neq i$, $\underset{M}{\vee} y \neq \mathrm{i}$, и построим формулы, соответствующие основным силлогистическим отношениям:

Axy – «всякое x есть y»,

Exy – «никакое x не есть y»,

Ixy – «некоторое x есть y»,

Oxy – «некоторое x не есть y».

Будем исходить из того, что эти отношения удовлетворяют следующим тождествам силлогистики:

$$Axy = \overline{O}xy, \quad Exy = \overline{I}\,xy \text{ (законы противоположности)},$$

$$Exy = Eyx \text{ (закон обращения для } E),$$

$$Exy \vee Oxy = Oxy \text{ (закон подчинения для } E).$$

Из двух последних получим

$$Exy \vee Oyx = Oyx.$$

Наиболее очевидны условия выполнения отношений I и O:

$Ixy = 1$ (и, следовательно, Exy=0), если $\underset{M}{\vee} x \wedge y = 1$,

Oxy = 1 (и, следовательно, $Axy = 0$), если $\underset{M}{\vee} x \wedge \overline{\mathrm{y}} = 1$.

Чтобы было выполнено отношение Exy, требуется соблюдение трех условий:

1) $\underset{M}{\overline{\vee}} x \wedge y = 1$ В силу закона противоположности $Exy = \overline{I}\,xy$,

2) $\underset{M}{\vee} x \wedge y = 1$ в силу закона подчинения для E,

3) $\underset{M}{\vee} \overline{x} \wedge y = 1$ в силу тождества $Exy \vee Oxy = Oyx$.

Таким образом:

$Exy = 1$ (и, следовательно, $Ixy = 0$), если

$$(\underset{M}{\overline{\vee}} x \wedge y) \wedge (\underset{M}{\vee} x \wedge y) \wedge (\underset{M}{\vee} \overline{x} \wedge y) = 1$$

В случае, когда выполнено условие 1), но не выполнены условия 2) или 3), отношения Ixy и Exy неопределены: $Ixy = Exy = i$. Действительно, поскольку $\underset{M}{\vee} x \wedge y \neq 1$, то $Ixy \neq 1$ и $Exy \neq 0$, а поскольку $(\underset{M}{\vee} x \wedge \overline{y}) \wedge (\underset{M}{\vee} \overline{x} \wedge y) \neq 1$, то $Exy \neq 1$ и $Ixy \neq 0$, т. е. о каждом из отношений Ixy, Exy нельзя сказать, что оно выполнено, и вместе с тем нельзя сказать, что оно не выполнено.

Рассмотренная трехзначная ситуация выражается формулами:

$$Ixy = (\underset{M}{\vee} x \wedge y) \vee i \wedge \overline{(\underset{M}{\vee} x \wedge \bar{y}) \wedge (\underset{M}{\vee} \bar{x} \wedge y)},$$

$$Exy = (\underset{M}{\bar{\vee}} x \wedge y) \wedge ((\underset{M}{\vee} x \wedge \bar{y}) \wedge (\underset{M}{\vee} \bar{x} \wedge y) \wedge i).$$

Можно показать, что эти формулы сохраняют правильный смысл также в случаях, когда неопределенность x и y не исключена.

Соответствующие формулы для отношений Oxy и Axy получаются посредством тождеств:

$$Oxy = \bar{I}\, xy$$

$$Axy = \bar{O}xy$$

Заметим, что все известные в силлогистике варианты отношений A, E, I, O можно выразить при помощи операции обращения через любое из этих отношений, например через Ixy:

$$Ixy = \mathrm{I}yx,$$

$$Eyx = Exy = Ixy,$$

$$Oxy = Ixy,$$

$$Oyx = Ixy,$$

$$Axy = Ixy,$$

$$Ayx = Ixy.$$

При этом оказывается, что существующая теория силлогизма упускает из вида два из 8 возможных отношений этого типа, а именно:

$$\mathrm{I}\overline{xy},\ \overline{\mathrm{I}xy}.$$

В рассмотренной трехзначной логике выводимы все без исключения тождества теории силлогизма, как традиционной, так и дополненной отношениями $\mathrm{I}\overline{xy}$ и $\overline{\mathrm{I}xy}$, в том числе 384 истинных модуса из 4096 модусов, возможных в дополненной теории.

Вследствие того, что средства двузначной логики недостаточны для эффективного построения силлогистики, эта важнейшая на практике логическая система, в сущности, не воспринята современной математической логикой. Характерное для математиков, например, отождествление отношения Axy с теоретико-множественным включением $x \subset y$ [23, стр. 301] – свидетельство совершенного непонимания силлогистики. Включение же силлогистики в математическую логику может придать языку последней гибкость и естественность, отсутствие которых является его существенным недостатком. Заметим в связи, с этим, что являющиеся необходимыми элементами языка математики выражения «достаточность x для y» и «необходимость x для y» являются синонимами соответственно Axy и Ayx.

Рассмотренная трехзначная алгебра с операциями $-$, $\wedge$, $\vee$ может быть применена в цифровой технике при использовании двоичного кода «с активным нулем», т. е. при такой реализации двоичных устройств, при которой двоичный код передается трехзначными сигналами, например: положительный импульс – 1 , отрицательный импульс – 0, отсутствие импульса – «нет передачи».

Система операций $-$, $\wedge$, $\vee$ в трехзначной логике не обладает функциональной полнотой даже при наличии констант, но для данных применений это не существенно.

§ 9. Алгебры трехзначной логики

Первые работы, посвященные алгебрам трехзначной (и вообще, многозначной) логики, относятся к 20-м годам нашего века [14, 24, 25]. В 50-х годах положено начало применению этих алгебр для проектирования логики цифровых устройств [26–28]. Последнее десятилетие характеризуется высокой и быстро возрастающей активностью в данной области – ежегодно появляется несколько десятков публикаций.

Многозначным логикам свойственно значительно большее по сравнению с двузначной логикой разнообразие операций. В частности, в трехзначной логике имеется 27 одноместных и 19 683 двухместных операции. Вопрос о том, какие из операций следует включить в алгебру, предназначенную для проектирования p-ичных цифровых устройств, должен решаться с учетом следующих требований:

1. Система операций должна быть функционально полной [29, 30] в p-значной логике.

2. Система операций должна быть гибкой, т. е. должна обеспечивать возможность представления достаточно простыми суперпозициями [30] большинство важных в цифровых устройствах функций.

3. Операции, включенные в систему, должны допускать экономную физическую реализацию их наличными техническими средствами.

Для алгебр, базирующихся на предметной интерпретации алфавита, типичны операции конъюнкции, дизъюнкции, инверсии, циклической подстановки и ряд других одноместных операций, из которых наиболее популярно следующее семейство [31]:

$$j_i(\alpha_k) = \begin{cases} \alpha_p, & \text{если } k = \mathrm{l} \\ \alpha\mathrm{1}, & \text{если } k \neq \mathrm{l}, \ k, \mathrm{l} \in \{1, 2, \ldots, p\}. \end{cases}$$

Семейство операций образует функционально полную в p-значной логике систему при наличии констант $\alpha_1, \alpha_2, \ldots, \alpha_p$ вместе с операциями конъюнкции и дизъюнкции, а также вместе с каждой одной из них и инверсией. Однако существенно большей гибкостью обладает алгебра, включающая все перечисленные операции.

Трехзначный вариант такой алгебры мы получим, дополнив алгебру, рассмотренную в § 8, операциями $j_1()$, $j_2()$, $j_3()$. При этом можно сформулировать следующие тождества, являющиеся трехзначными аналогами законов противоречия и исключенного третьего двузначной логики:

$$j_l(x) \wedge j_m(x) = 0, \ \mathrm{l} \neq m, \ \mathrm{l}, \mathrm{m} \in \{1, 2, 3\},$$

$$\bigvee_{l=1}^{3} J_l(x) = 1.$$

Алгебры, связанные с числовой интерпретацией алфавита, включают операции типа сложения и умножения, а также ряд одноместных операций, из которых наиболее часто употребляется семейство

$$(\alpha_k)^l = \begin{cases} 1, & \text{если } \alpha_k = l \\ 0, & \text{если } \alpha_k \neq l, \ \alpha_k, \mathrm{l} \in \{\alpha_1, \alpha_2, \ldots, \alpha_p\}. \end{cases}$$

В качестве операции типа сложения и умножения обычно используется модульное сложение и умножение, т. е. операции, результаты которых опре-

деляются как наименьшие вычеты по модулю p соответственно суммы и произведения, являющиеся значениями используемого алфавита $\alpha[1{:}p]$. Эти операции, для обозначения которых мы применяем знаки $|+|$ и $|\times|$, можно определить при помощи описанной в § 5 функции $\mathrm{mod}_{\alpha_1}^{\alpha_p}(\,)$:

$$\mathrm{aj}\ |+|\ \mathrm{ak} = \mathrm{mod}_{\alpha_1}^{\alpha_p}(a_j + a_k),$$
$$\mathrm{aj}\ |\times|\ \mathrm{ak} = \mathrm{mod}_{\alpha_1}^{\alpha_p}(a_j \times a_k).$$

Модульные операции удобны тем, что с ними связаны те же правила преобразования выражений, которым удовлетворяют обычные арифметические операции. Иногда считают, что использование модульных операций обеспечивает экономную реализацию операционных устройств цифровой машины [5, 32], однако экономия получается, по-видимому, только на бумаге, потому что экономная реализация самих модульных операций еще не изобретена.

Другой вид операций сложения и умножения в p-значном алфавите связан с функцией *ограничения* $\mathrm{restr}_{\alpha_1}^{\alpha_p}(\,)$, которая определяется так:

$$\mathrm{restr}_{\alpha_1}^{\alpha_p}(a) = \begin{cases} \alpha_p, & \text{если } a \geq \alpha_p, \\ a, & \text{если } \alpha_1 \leq a \leq \alpha_p, \\ \alpha_1, & \text{если } a \leq \alpha_1, \end{cases}$$

где α_1, α_p – соответственно первое и последнее числовые значения алфавита, a – целое число.

Операции сложения и умножения с ограничением, которые мы будем называть также *пороговыми* операциями сложения и умножения и обозначать знаками $\overline{+}$ и $\overline{\times}$, выражаются с помощью функции $\mathrm{restr}_{\alpha_1}^{\alpha_p}(\,)$ в виде:

$$a_j \mathbin{\overline{+}} a_k = \mathrm{restr}_{\alpha_1}^{\alpha_p}(a_j + a_k),$$
$$a_j \mathbin{\overline{\times}} a_k = \mathrm{restr}_{\alpha_1}^{\alpha_p}(a_j \times a_k).$$

Преимущество пороговых операций перед модульными заключается в том, что, во-первых, для них имеется достаточно простая физическая реализация и, во-вторых, они эффективны при описании логики как арифметических, так и управляющих устройств, в то время как модульные операции для описания логики управления неудобны. Показательно, что при p=2 пороговое сложение совпадает с булевой дизъюнкцией, а модульное сложение – с операцией неэквивалентности. Заметим, что пороговое сложение не удовлетворяет закону ассоциативности.

Далее мы рассмотрим трехзначную алгебру, которая сложилась в процессе проектирования вычислительной машины «Сетунь» [7] и ряда троичных устройств, разработанных в Вычислительном центре Московского университета.

Рассматриваемая алгебра включает определенные над алфавитом $\alpha[1{:}3] = \overline{1}\ 0\ 1$ переменные x, y, z, . . . операции порогового сложения и умножения (пороговое умножение в случае алфавита $\overline{1}\ 0\ 1$ совпадает с обычным арифметическим умножением, поэтому далее эта операция обозначается знаком $\times$), а также одноместные операции инверсии (изменения знака) и отделения двузначных компонент трехзначной переменной:

минус-компонента:

$$\alpha_k{}^- = \begin{cases} 1, \text{ если } \alpha_k = \overline{1}, \\ 0, \text{ если } \alpha_k \neq \overline{1}, \end{cases}$$

нуль-компонента:

$$\alpha_k{}' = \begin{cases} 1, \text{ если } \alpha_k = \overline{1}, \\ 0, \text{ если } \alpha_k \neq \overline{1}, \end{cases}$$

плюс-компонента:

$$\alpha_k{}^+ = \begin{cases} 1, \text{ если } \alpha_k = \overline{1}, \\ 0, \text{ если } \alpha_k \neq \overline{1}. \end{cases}$$

Возможности преобразования выражений в данной алгебре определяют следующие тождества.

Идемпотентность сложения:

$$x \mathbin{\overline{+}} x = x.$$

Коммутативность сложения и умножения:

$$x \mathbin{\overline{+}} y = y \mathbin{\overline{+}} x,$$

$$x \times y = y \times x.$$

Ассоциативность умножения:

$$x \times (y \times z) = (x \times y) \times z.$$

Дистрибутивность умножения относительно сложения:

$$x \times (y \mathbin{\overline{+}} z) = x \times y \mathbin{\overline{+}} x \times z.$$

Тождества с операцией инверсии:

$$\overline{\overline{x}} = x,$$

$$\overline{x \mathbin{\overline{+}} y} = \overline{x} \mathbin{\overline{+}} \overline{y},$$

$$\overline{x} \times \overline{y} = x \times \mathrm{y},$$

$$\overline{x \times y} = x \times \overline{y} = \overline{x} \times y.$$

Тождества с константами:

$$x \mathbin{\overline{+}} \overline{x} = 0, \qquad x \times 1 = \overline{x},$$

$$x \mathbin{\overline{+}} 0 = x, \qquad x \times 0 = 0,$$

$$\mathrm{x} \times 1 = \mathrm{x}.$$

Тождества с компонентами:

(Сокращенная запись x^{ik} означает $(x^i)^k$, $i, k \in \{-, ', +\}$.)

$$x^{k-} = 0, \qquad x^+ \times x^- = 0,$$

$$x^{k''} = x^k, \qquad x \times x' = 0,$$

$$x^{k+} = x^k, \qquad x \times x^- = \overline{x^-},$$

$$\overline{x^-} \mathbin{\overline{+}} x^+ = x, \qquad \overline{x} \times x^- = x^-,$$

$$\overline{x^+} \mathbin{\overline{+}} x^- = \bar{x}, \qquad x^{+'} \times x^- = x^-,$$
$$x^{-'} \mathbin{\overline{+}} x^+ = x^{-'}, \qquad x \times x^+ = x^+,$$
$$x^{+'} \mathbin{\overline{+}} x^- = x^{+'}, \qquad \bar{x} \times x^+ = \overline{x^+},$$
$$x^{+'} \mathbin{\overline{+}} x^{-'} = 1, \qquad x^{-'} \times x^+ = x^+,$$
$$x^- \mathbin{\overline{+}} x' \mathbin{\overline{+}} x^+ = 1,$$
$$x \times x \mathbin{\overline{+}} \bar{x} = x'' \times x^{+'} = \bar{x},$$
$$x \times \bar{x} \mathbin{\overline{+}} 1 = x^{-'} \times x^{+'} = x',$$
$$x \times \bar{x} \mathbin{\overline{+}} x = x^{-'} \times x'' = x^+,$$

Поскольку компоненты x^i, y^i, z^i ..., $i \in \{-, ', +\}$, принимают значения из двузначного алфавита 0 1, над которым операция порогового сложения является операцией дизъюнкции, операция умножения – операцией конъюнкции и операция нуль-компонента – операцией отрицания, то выражения, в которых нет знака инверсии и все буквы находятся под знаками компонент, можно рассматривать как выражения булевой алгебры, имея в виду, что знак $\overline{+}$ означает дизъюнкцию, знак $\overline{\times}$ – конъюнкцию, знак $\bar{}$ – отрицание.

В рассмотренной пороговой алгебре достаточно просто и наглядно выражаются наиболее типичные функции, реализуемые операционными элементами цифровых устройств.

Функция *смесителя* трехзначных в алфавите $\overline{1}$ 0 1 сигналов x, y, z выражается при помощи порогового сложения:

$$x \mathbin{\overline{+}} y \mathbin{\overline{+}} z.$$

Функции *вентилей*, осуществляющих передачу трехзначного сигнала x в зависимости от значения управляющего вентилем сигнала u: $u = \overline{1}$, $u = 0$, $u = 1$, выражаются соответственно в виде:

$$\mathrm{x} \times \mathrm{u}^-, \qquad \mathrm{x} \times \mathrm{u}', \qquad \mathrm{x} \times \mathrm{u}^+,$$

Функция *сложного вентиля* [8, 26], передающего один из трехзначных сигналов x, y, z в зависимости от значения управляющего сигнала u:

$$T(x, y, z, u) = x \times u^- \mathbin{\overline{+}} y \times u' \mathbin{\overline{+}} z \times u^+.$$

Функция сумматора по модулю 3.

$$x \mathbin{\overline{+}} y = x \times y' \mathbin{\overline{+}} \mathrm{x}' \times \mathrm{y} \mathbin{\overline{+}} \mathrm{x}^- \times \mathrm{y}^- \mathbin{\overline{+}} \overline{\mathrm{x}^+ \times \mathrm{y}^+}.$$

§ 10. Алгебраическое представление функций *p*-значной логики

Первоочередной задачей алгебраического синтеза цифровых устройств является задача представления произвольной функции p-значных переменных суперпозицией операций данной конкретной алгебры. Достаточно универсальным является следующее решение этой задачи.

Рассмотрим класс алгебр p-значной логики, определяемый схемой одноместной операции $(\,)^{\mathrm{srl}}$:

$$\alpha_k{}^{srl} = (\alpha_k)^{srl} = \begin{cases} \alpha_s, & \text{если } k = l, \\ \alpha_r, & \text{если } k \neq l, \end{cases}$$

и двумя свойствами двухместной коммутативной операции sr:

$$\alpha_k sr\ \alpha_s = \alpha_s sr\ \alpha_k = \alpha_s,$$
$$\alpha_k sr\ \alpha_r = \alpha_r sr\ \alpha_k = a_k,$$

где $s, r, s \neq r$ – параметры, значениями которых определяется конкретная алгебра, l – параметр, значением которого определяется конкретный вид одноместной операции $k, l, s, r \in \{1, 2, \ldots, p\}$.

Условимся, что перестановка букв s, r в обозначениях операций соответственно изменяет их смысл:

$$\alpha_k{}^{rsl} = \begin{cases} \alpha_r, & \text{если } k = l, \\ \alpha_s, & \text{если } k \neq l, \end{cases}$$

$$\alpha_k rs\ \alpha_r = \alpha_r,$$
$$\alpha_k rs\ \alpha_s = \alpha_k.$$

Для n-местной операции sr введем обозначение $\overset{n}{\underset{i=1}{SR}}$:

$$\overset{n}{\underset{i=1}{SR}}\, x_i = x_1 sr\ x_2 sr\ \ldots\ sr\ x_n.$$

Произвольная функция n переменных p-значной логики $f(x_1, x_2, \ldots, x_n)$ в алгебре рассматриваемого класса, определенной заданием значений параметров $s, r, s \neq r$, может быть представлена в виде:

$$f(x_1, x_2, \ldots, x_n) = \overset{p}{\underset{i=1}{SR}}\, x_n{}^{sri}\ rs\ f(x_1, x_2, \ldots, x_{n-1}, \alpha_i).$$

В результате n-кратного применения этой формулы получим выражение функции f через значения, принимаемые ею на всевозможных наборах значений переменных $x_1, x_2, \ldots, x_n$. Например, для функции одной переменной:

$$f(x) = \overset{p}{\underset{i=1}{SR}}\, x^{sri}\ rs\ f(\alpha_i),$$

для функции двух переменных:

$$f(x_1, x_2) = \overset{p}{\underset{i=1}{SR}}\ \overset{p}{\underset{j=1}{SR}}\, x_1{}^{sri}\ rs\ x_2{}^{srj}\ rs\ f(\alpha_i, \alpha_j) \text{ и т. д.}$$

Данное представление справедливо в широком классе алгебр, включающем алгебры, связанные как с предметной, так и с числовой интерпретацией p-значног о алфавита. В частности, при $r = l$, $s = p$ операции sr и rs являются операциями типа дизъюнкции и конъюнкции, а операция $(\ )^{srl}$ совпадает с рассмотренной в § 9 операцией $j_l(\ )$ [31, 33]. При $\alpha_r = 0$, $\alpha_s = 1$ операции sr и rs являются операциями типа сложения и умножения, а операция $(\ )^{srl}$ – операцией отделения l-той компоненты p-значной переменной [7, 27].

Рассмотренное представление функций p-значной логики осуществлено в основном средствами двузначной логики: p-значные переменные $x_1, x_2, \ldots, x_n$ входят в него не непосредственно, а в виде совокупности двузначных переменных x_i^{srk}, $i = 1, 2, \ldots, n$; k = 1, 2, …, p, полученных в результате разложения:

$$x_i = \overset{p}{\underset{p=1}{SR}}\, \alpha_k\, rs\; x_i^{srk}.$$

К равноценному представлению можно прийти путем прямого построения p-значной функции в виде:

$$f(x_1, x_2, \ldots, x_n) = \overset{p}{\underset{l=1}{SR}}\, \alpha_l\, rs\; f^{srl},$$

где f^{srl} – двузначная функция np двузначных переменных x_i^{srk}. При этом переменные x_i^{srl} и функции f^{srl} в случае предметной интерпретации алфавита и соответствующие нулькомпоненты в случае числовой интерпретации можно опустить. Так в трехзначной логике с алфавитом $\bar{1}$ 0 1 достаточно минус- и плюс-компонент [7]. В частности, в рассмотренной выше пороговой алгебре функция n переменных будет представлена в виде:

$$f(x_1, x_2, \ldots, x_n) = f^- \mp f^+,$$

где f^- и f^+ – двузначные функции $2n$ двузначных переменных x_1^-, x_1^+, x_2^-, x_2^+, …, x_n^-, x_n^+.

§ 11. Пороговая реализация трехзначных функций

Подавляющее большинство описанных в литературе примеров троичного синтеза, использующих разные алгебры и методы минимизации, показывает, что являющийся непременным и обычно единственным синтезируемым объектом троичный сумматор получается в 3–4 раза более сложным (по количеству деталей на разряд), чем двоичный сумматор.

Известен, однако, способ осуществления операций, применение которого позволяет снизить указанное отношение до 2 и даже ниже. Этот способ, называемый обычно *пороговой логикой* [34], основан на представлении функций логики в виде арифметических выражений с ограничением. Частными случаями представленных этим способом функций являются рассмотренные в § 9 пороговые сложение и умножение.

Практически пороговое представление функции n переменных $f(x_1, x_2, \ldots, x_n)$ p-значной логики осуществляется в виде:

$$f(x_1, x_2, \ldots, x_n) = \mathrm{restr}_{\alpha_1}^{\alpha_p}(a_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n),$$

где $a_0, a_1, \ldots, a_n$ – целые числа. Функция, для которой это представление существует, называется *пороговой функцией*.

В случае симметричного трехзначного алфавита функция $\mathrm{restr}_{\alpha_1}^{\alpha_p}()$ совпадает с функцией sign ( ):

$$\mathrm{sign}(a) = \mathrm{restr}_{-\bar{1}}^{1}(a),$$

которая будет использоваться в дальнейшем изложении.

Рассматривая функцию sign ( ) вместе с описанной в § 9 трехзначной пороговой алгеброй, мы будем пользоваться также ее плюс- и минус-компонентами: sign^+, sign^-. Заметим, что плюс-компонента sign^+ составляет основу двузначной пороговой логики:

$$\mathrm{sign}^{+}(a) = \mathrm{restr}_{0}^{1}(a).$$

Компоненты sign^{+} и sign^{-} удовлетворяют следующим тождествам:

$$\mathrm{sign}^{+}(a) = \mathrm{sign}^{-}(\overline{a}),$$
$$\mathrm{sign}^{-}(a) = \mathrm{sign}^{+}(\overline{a}),$$
$$\mathrm{sign}^{+}(a) \mp \overline{\mathrm{sign}^{-}}(a) = \mathrm{sign}(a).$$

Далее мы рассмотрим систему трехзначной пороговой логики, осуществленную в вычислительной машине «Сетунь» [35, 36].

Общий вид функции, реализуемой логическими элементами (ячейками) устройств машины «Сетунь», выражается при помощи установленных выше обозначений формулой:

$$\mathrm{sign}^{+}\left(a_0 + \sum_{i=1}^{n} a_i x_i\right) \mp \overline{\mathrm{sign}^{-}}\left(\mathrm{b}_0 + \sum_{i=1}^{n} b_i x_i\right),$$

где $x_i \in \{\overline{1}, 0, 1\}$ a_0, b_0, a_i, b_i – целые числа.

Известные системы трехзначных пороговых функций являются частными случаями рассматриваемой системы и получаются из нее путем соответствующих ограничений.

При $a_0 = b_0 = 0$, $a_i = b_i = 1$, n – нечетно, получаем трехзначные мажоритарные функции вида $\mathrm{sign} \sum_{i=1}^{2k+1} x_i$, $\mathrm{k} = 1, 2, \ldots, m$ [37].

При $a_0 = b_0 = -\eta$, $a_i = b_i = \xi_i$ – трехзначные пороговые функции вида $\mathrm{sign}\left(\sum_{i=1}^{n} \xi_i x_i - \eta\right)$ [38].

При $a_0 = -_{\mathrm{b}}0$, $a_i = b_i$ – трехзначные двухпороговые функции [39, 40].

Функциональная способность реальных ячеек, примененных в устройствах машины «Сетунь», ограничена следующим выбором

параметров: $a_0, b_0, a_1, b_1 \in \{\overline{1}, 0, 1\}$, $\mathrm{n} \leq 4$. Ограничения обусловлены недостаточной нагрузочной способностью используемых физических элементов – магнитных усилителей с питанием импульсами тока [12] и стремлением максимально облегчить монтаж ячеек, который выполнялся вручную, выразившимся, в частности, в том, что количество возможных выводов было уменьшено с целью увеличить расстояние между ними.

Расширение в разумных пределах диапазона значений, приписываемых параметрам a_0, b_0, a_1, b_1, n, не связано с ужесточением требований к точности применяемых деталей, и оно было осуществлено в последующих модификациях ячеек. При этом оказалось, что расширение функциональных способностей ячеек в указанном смысле приводит к существенной экономии оборудования главным образом в операционных устройствах, причем практически достаточным является следующий диапазон значений:

$$a_0, b_0, a_i, b_i \in \{\overline{3}, \overline{2}, \overline{1}, 0, 1, 2, 3\}, \mathrm{n} \leq 6.$$

Иллюстрацией экономии, обусловленной расширением функциональных способностей ячеек, может служить реализация сумматора с тремя входами (не считая вход переноса), используемого в множительном устройстве маши-

ны «Сетунь». Функцию, выполняемую этим сумматором, можно осуществить на двух модифицированных ячейках:

$$s_i = \text{sign}\,(a_i + b_i + \text{d}_i + \overline{3}c_i + c_{i+1}),$$

$$c_i = \text{sign}^{+}\,(1 + a_i + b_i + d_i + c_{i+1}) \mp \overline{\text{sign}^{-}}\,(1 + a_i + b_i + d_i + \text{c}_{i+1}).$$

Последовательный вариант сумматора с учетом необходимых элементов задержки содержит 6 ячеек, в то время как ф ункционально равноценный сумматор, используемый в машине «Сетунь», выполнен на 17 ячейках.

Вместе с тем при реализации функций, характерных для управляющих устройств, применение модифицированных ячеек не дает ощутимой экономии, потому что наиболее популярные из этих ф ункций могут быть осуществлены немодифицированной ячейкой:

$$x \times u^{+} = \text{sign}^{+}\,(\overline{1} \mp x \mp u) \mp \overline{\text{sign}^{-}}\,(1 \mp x \mp \overline{u}),$$

$$x \times u = \text{sign}^{+}\,(\overline{1} \mp x \mp \overline{u}) \mp \text{sign}\,(1 \mp x \mp u).$$

Поскольку реализуемые данными ячейками трехзначные функции представлены парами двузначных компонент, то на троичную ячейку должно приходиться в 2 раза больше оборудования, чем на двоичную (за исключением корпуса ячейки, который в обоих случаях требуется один). Поэтому экономность троичных устройств по отношению к двоичным, осуществленным этим же способом, можно характеризовать отношением $\frac{N_2}{2N_3}$, где N_2 и N_3 – количества соответственно двоичных и троичных ячеек в двоичном и троичном вариантах устройства: для параллельного арифметического устройства $N_2 > 1{,}59\,N_3$, для устройства управления $N_2 > 2N_3$.

Однако раздельное использование двоичных компонент, составляющих троичную ячейку, практически не всегда возможно, В частности, двоичные ячейки, аналогичные ячейкам «Сетуни», выполняются также в виде взаимно скомпенсированных пар магнитных усилителей, т. е. по затратам оборудования равноценны троичным ячейкам. Подобное положение имеет место при осуществлении трехзначных и двузначных функций с помощью индуктивных параметронов [41]. В таких случаях относительная величина выигрыша, обусловленного применением троичного кода, определяется отношением $\frac{N_2}{N_3}$.

§ 12. Практические результаты

До настоящего времени известна единственная цифровая машина, работающая в троичном коде – разработанная в Вычислительном центре МГУ малая вычислительная машина «Сетунь» [36]. В этой машине реализованы далеко не все полезные свойства троичного кода и трехзначной логики: ограниченно используется возможность выполнения операций над словами разной длины, не применяется однопроводная передача трехзначных сигналов, сравнительно слабо использованы возможности трехзначной пороговой логики. Кроме того, в «Сетуни» нет операций с плавающей запятой, для которых

преимущества троичного кода особенно существенны. Несмотря на это «Сетунь» наглядно продемонстрировала выгодность троичного кода. Достаточно сказать, что даже в условиях мелкосерийного производства и фактически в макетном исполнении «Сетунь» была значительно дешевле машин ее класса (в том числе выпускавшихся в последующее время), превосходя их в отношении быстродействия и диапазона эффективных применений [42].

Оценка экономии, полученной за счет использования троичного кода и трехзначной логики в операционных и управ ляющих устройствах машины «Сетунь», показывает, что по сравнению с двоичными устройствами, построенными на базе той же техники, необходимое количество деталей и потребляемая мощность уменьшились в 2 раза при одновременном увеличении быстродействия более чем на 50%.

Примененный в машине «Сетунь» способ пороговой реализации логики не связан с ужесточением допусков и обеспечивает высокую надежность устройств. Главная трудность применения этого способа заключается в том, что техника осуществления операционных устройств на основе электромагнитных явлений в настоящее время недостаточно разработана, причем основное внимание уделяется использованию этих явлений в запоминающих устройствах. Перспективы развития указанной техники, по-видимому, связаны с успехами интегральной технологии магнитных запоминающих устройств, а также с возможностью использования эффекта Холла.

§ 13. Заключение

Троичный код является самым простым (в отношении структуры и физической реализации) кодом, который допускает использование симметричного алфавита, т. е. может быть симметричным кодом. Этим обусловлены его главные преимущества перед другими кодами применительно к цифровым машинам. Симметричному коду присущи такие ценные свойства, как естественное представление чисел со знаком на уровне элементов и правильное самоокругление чисел при отсечении младших разрядов.

Вследствие указанных свойств в симметричном троичном коде коренным образом упрощается логика и сокращается число необходимых вариантов арифметических операций, открывается возможность оперирования со словами разной длины, существенно упрощается структура операционных устройств, получается значительная экономия времени, затрачиваемого на выполнение операций, а в ряде случаев также оборудования.

Троичный код является единственным недвоичным кодом, использование которого может быть не связано с ужесточением требований к стабильности сигналов и к различительной способности воспринимающих сигналы устройств. При этом пропускная способность каналов в троичном коде в 1,59 раза больше, чем в двоичном. В частности, в операционных устройствах последовательного действия сложение в троичном коде выполняется в 1,59 раза, а умножение в 2,5 раза быстрее, чем в двоичном.

Трехзначная логика в цифровых устройствах весьма эффективна, так как трехзначные ситуации достаточно часты и, кроме того, имеется возможность попарного объединения двузначных управляющих сигналов в трехзначные.

Наиболее экономная реализация трехзначной логики связана с применением электромагнитных пороговых элементов. Средствами полупроводниковой техники троичные устройства реализуются, как правило, с большими затратами оборудования, чем функционально равноценные двоичные устройства. Однако в условиях интегральной технологии связанная с троичным кодом возможность упрощения структуры устройств за счет усложнения элементов и экономия количества соединений между элементами могут оказаться важнее обычной экономии деталей.

ЛИТЕРАТУРА

1. Синтез электронных вычислительных и управляющих схем. Перевод с англ. под ред. В. И. Шестакова. М., ИЛ, 1954.

2. Ричардс Р. К. Арифметические операции на цифровых вычислительных машинах. М., ИЛ, 1957.

3. Карцев А. А. Арифметические устройства электронных цифровых машин. М., Физматгиз, 1958.

4. Брусенцов Н. П. Вычислительная машина «Сетунь» Московского государственного университета. В сб.: «Материалы научно-технической конференции «Новые разработки в области вычислительной математики и вычислительной техники». Киев, 1960, стр. 226–234.

5. Hallworth R. P., Heath F. G. Semiconductor circuits for ternary logic. IEE Monograph No. 482E, Nov. 1961.

6. Alexander W. The ternary computer. «Electronics and Power», Febr. 1964, pp. 36–29.

7. Брусенцов Н. П. Опыт разработки троичной вычислительной машины. «Вести. Моск. унта», сер. матем., 1965, № 2, стр. 39–48.

8. Turecki A. The ternary number system for digital computers. «Comput. Design», 1968, v. 7, No. 2, pp. 66–71.

9. Shahbender R. Laminated ferrite memories – review and avaluation. RCA Rev. 1968, v. 29. No. 2, pp. 180–198.

10. Бардиж В. В., Перекатов В. И., Шойфер Ю. И. Особенности ЗУ на слоистых ферритовых матрицах. В сб.: «Третье Всесоюзное совещание по запоминающим устройствам вычислительных машин». Л., 1968, стр. 137–138.

11. Маслов С. П. Вопросы построения запоминающих устройств для троично-кодированной информации (в настоящем сборнике).

12. Брусенцов Н. П. Цифровые элементы типа быстродействующих магнитных усилителей с питанием импульсами тока. В сб: «Цифровые магнитные элементы и устройства». Изд-во МГУ, 1966, стр. 4–66.

13. Тэрас М., Охока Т., Мураяма К. Система управления производственными процессами с корректирующим вычислительным устройством, включающая телеметрический селектор с троичным кодом. «Тр. I Конгресса IFAC», IV. М., Изд-во АН СССР, 1961 , стр. 326–334.

14. Post E. L. Introduction to a general theary of elementary propositions. Amer. J. Math, 1921, 43, pp. 163–185.

15. Pawlak Z., Wakuhcz A. Use of expansion wall a negative bas1s in the arithmometer of a digital computer. Bull. Acad. Polon. Sci., ser. Math., 1957, v. 5, No. 3, pp. 233–236.

16. Жоголев Е. А. Особенности программирования и математическое обслуживание для машины «Сетунь». В сб.: «Вычислительные методы и программирование», вып. 5. Изд-во МГУ, 1966, стр. 294–318.

17. Баня Е. Н. Применение в вычислительных машинах системы счисления остаточных классов (СОК) с наименьшими по абсолютной величине основаниями. «Кибернетика», 1967, № 8, стр. 36–38.

18. Воеводин В. В. Ошибки округления и устойчивость в прямых методах линейной алгебры. Ротапринт ВЦ МГУ, 1969.

19. Бочвар Д. А. Об одном трехзначном исчислении и его применении к анализу парадоксов классического расширенного функционального исчисления. «Математич. сборник», 1938, т. 4, вып. 2, стр. 287–308.

20. Гильберт Д., Аккерман В. Основы теоретической логики. М., ИЛ, 1947.

21. Лукасевич Я. Аристотелевская силлогистика с точки прения современной формальной логики. М., ИЛ, 1959.

22. Субботин А. Л. Теория силлогистики в современной формальной логике. М., «Наука», 1965.

23. Бурбаки Н. Теория множеств. М., «Мир», 1965.

24. Lukasiewicz J. O logike trojwartosciowej. Ruch Filozoticzny, 1920, v. 5, pp. 169–171.

25. Bernstein B. A. Representation of three element algebras. Amer. J. Math., 1924, v. 46, No. 2, pp. 110–116.

26. Lee C. Y., Chen W. H. Several-valued combinational switching circuits. Commun. and Electron., July 1956, No. 25, pp. 278–282.

27. Моисил Г. К. Алгебраическая теория дискретных автоматических устройств (перевод с румынского изд. 1959 г.). М., ИЛ, 1963.

28. Шестаков В. И. Перфокарточный метод синтеза многотактных систем многопозиционных реле. «Автоматика и телемеханика», т. 20, № 11, стр. 1496–1506, 1959.

29. Slupecki J. Kryteriom petnosci wielowartosciowych systemow logiki zban. «Comptes rendus des seances de la Societe des Sciences et des Letteres de Varsovie», Cl. III, 1939, v. 32, pp. 102–109.

30. Яблонский С. В. Функциональные построения в *к*-значной логике. «Тр. матем. ин-та им. В. А. Стеклова», т. 51, стр. 5–142, 1958.

31. Rosser J. B., Turquette A. R. Many-valued logics. Amsterdam, North Holland Pubb. Согпр., 1952.

32. Поспелов Д. А. Об одной постановке задачи минимизации в многозначных логиках. В сб.: «Многозначные элементы и структуры». М., «Советское радио», 1967, стр. 112–114.

33. Айзенберг Н. Н., Рабинович З. Л. Некоторые классы функционально полных систем операций и канонические формы представления функций в многозначной логике. «Кибернетика», 1965, № 2, стр. 37–45.

34. Дертоузос М. Пороговая логика. М., «Мир», 1967.

35. Брусенцов Н. П. Построение логических схем на магнитных усилителях с питанием импульсами тока. В сб.: «Магнитные элементы. Труды Всесоюзного совещания во Львове 10–16 сентября 1962 г.». Киев, «Наукова думка», 1964, стр. 361–367.

36. Брусенцов Н. П., Маслов С. П., Розин В. П., Тишулина А. М. Малая цифровая вычислительная машина «Сетунь». М., Изд-во МГУ. 1965.

37. Варшавский В. И. Трехзначная мажоритарная логика. «Автоматика и телемеханика», 1964, т. 15, № 5, стр. 673–684.

38. Боголюбов И. Н. Минимизация перебора при синтезе трехстабильного порогового элемента. «Кибернетика», 1967, № 6, стр. 85.

39. Hanson W. H. Ternary threshold logic. «IEEE Trans on electron. Comp.», 1963, v. EC-12, No. 3, pp. 191–197.

40. Merril R. Some properties of ternary threshold logic. «IEEE Trans, on Electron. Сошр.», 1964, v. EC-13, No. 5, pp. 632–635.

41. Иваськин Ю. Л. Вопросы синтеза схем цифровых автоматов из логических троичных элементов, использующих трехстабилыный параметрон. В сб.: «Многозначные элементы и структуры». М , «Советское радио», 1967, стр. 132–142.

42. Аннотированный указатель программ для вычислительной машины «Сетунь». Составители Н. П. Брусенцов и В. А. Морозов. Ротапринт ВЦ МГУ, 1968.